Eirik Harald Lund

Scalability of Blockchain-based Supply Chain Management Systems

Master's thesis in Computer Science
Supervisor: Maria Letizia Jaccheri, Xiaoying Bai
July 2019

Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

NTNU
Norwegian University of
Science and Technology

Eirik Harald Lund

Scalability of Blockchain-based Supply Chain Management Systems

Master's thesis in Computer Science
Supervisor: Maria Letizia Jaccheri, Xiaoying Bai
July 2019

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

NTNU
Norwegian University of
Science and Technology

Abstract

Eirik Harald LUND

Scalability of Blockchain-based Supply Chain Management Systems

Sustainability and sustainable development are topics of growing public interest. The UN have proposed several goals for sustainable development for the global society, in order to face the growing climate crisis. Among the goals from the UN is a call for technological innovation, in order to improve and disrupt industries and services on all levels.

Blockchain is a fairly novel technology, which offers possibilities of radically changing industries. Among the possible use cases of the blockchain technology, is that of supply chain management systems. Supply chains are an essential part of nearly every industry, and involves parties all around the globe. However, there are several weaknesses with current supply chains. Utilizing blockchain technology to improve supply chain management systems, offers a possibility improve upon current practices. To learn more about the implications of using blockchain technology for supply chain management systems, we propose the following research questions:

- How will a blockchain-based supply chain management system perform on different scales?
- How viable is a blockchain-based supply chain management systems for use in sustainable industries?

In order to answer our research questions, we created a simulated supply chain based on the open source project Hyperledger Fabric. Using the simulation, we performed 60 tests with varying configurations, from which we gathered data on the performance of the system. The data was then contextualized and analyzed.

We found that as the system grows, the time needed to submit transactions to the blockchain increases by up to 50% at the largest scale tested. The total time per transaction is still low, but more research should be done to see if the increase persists when the system grows even larger.

The size of the ledger grows at a constant rate to the number of transactions. The number of nodes in the network does not appear to affect the performance. However, with higher numbers of products, there appears to be a slowdown in the system.

In conclusion, our blockchain-based supply chain management system appears to scale well. However, improved simulations and further testing are necessary in order to investigate more use cases in more detail.

Sammendrag

Det er en økende interesse for bærekraftighet og bærekraftig utvikling. FN har kommet med en rekke mål for bærekraftig utvikling for verdenssamfunnet for å kunne motvirke den voksende klimakrisen. Ett av FNs mål omhandler behovet for teknologiske nyvinninger, for å forbedre og endre industrier og tjenester av alle slag.

Blockchain er en forholdsvis ny teknologi, med potensiale for å drastisk endre industrier. Blant de mulige bruksområdene finner vi systemer for håndtering av forsyningskjeder. Forsyningskjeder er en essensiell del av nær sagt alle industrier, og kan involvere kommunikasjon mellom alle verdensdeler. Det er imidlertid visse problemer med konvensjonelle forsyningskjeder. Ved å bruke blockchainteknologi til å forbedre systemer for håndtering av forsyningskjeder kan man potensielt forbedre dagens standard. For å lære mer om følgene av å bruke blockchainteknologi i forbindelse med systemer for håndtering av forsyningskjeder foreslår vi de følgende forskningsspørsmålene:

- Hvordan kommer et blockchainbasert system for håndtering av forsyningskjeder til å yte på ulike skalaer?
- Hvor gjennomførbart er det å bruke blockchainbaserte system for håndtering av forsyningskjeder i bærekraftige industrier?

For å besvare forskningsspørsmålene våre lagde vi en simulering av en forsyningskjede, basert på den åpne kildekoden til prosjektet Hyperledger Fabric. Vi gjennomførte 60 ulike tester med simuleringen, som vi brukte til å samle data om systemets ytelse. Disse dataene ble så satt i kontekst og analysert.

Vi fant at tiden som trengs for å registrere en transaksjon øker med så mye som 50% etter hvert som systemet vokser. Antallet noder i systemet later ikke til å påvirke ytelsen nevneverdig. Med et høyere antall produkter virker det imidlertid som at systemet blir tregere.

Resultat viser at vårt blockchainbaserte system for håndtering av forsyningskjeder later til å skalere godt. Bedre simuleringer og videre testing er imidlertid nødvendig for å undersøke flere bruksområder i mer detalj.

Acknowledgements

I would like to thank my supervisors, Prof. Letizia Jaccheri and Prof. Xiaoying Bai, for their great help and advice in working on this thesis. I would also like to give my thanks to PhD Candidate Orges Cico and Dr. Jingyue Li for their guidance in both academic and practical matters. I would also like to thank the IPIT Network as a whole, for offering me the opportunity to carry out my thesis work at Tsinghua University in China. Finally, I would like to thank the international sections at NTNU and Tsinghua for their help in making the academic exchange possible.

This work was funded by the Research Council of Norway under the project IPIT Project Number: 274816.

Contents

List of Figures

List of Tables

Listings

List of Abbreviations

SDG	**S**ustainable **D**evelopment **G**oals
PoW	**P**roof **o**f **W**ork
PoS	**P**roof **o**f **S**take
NTNU	Norwegian University of Science and Technology
IPIT	**I**nternational **P**artnerships for Excellent Education and Research in **I**nformation **T**echnology
REST	**Re**presentational **S**tate **T**ransfer
IoT	**I**nternet **o**f **T**hings

Chapter 1

Introduction

1.1 Motivation

Sustainability is becoming an increasingly important topic on the global agenda. The UN have released reports urging the global community to take action to strive for sustainable development and against environmental crime [39][44]. There is a need for technological innovations in several fields, in order to both renew and improve current practices. By improving sustainability, it becomes possible to improve the lives of people around the world, and also lay the foundations for economic growth in the future.

Blockchain is a technology that was originally proposed as the backbone of the digital currency Bitcoin [38]. However, it has since proven to have a variety of use cases, such as energy trading and supply chains [6][5]. Blockchain is a technology that holds possibilities of innovating, possibly even disrupting, a wide variety of fields and industries [52][46].

Supply chains are an essential part of most industries, from agriculture to pharmaceuticals [53][1]. By utilizing blockchain technology in supply chain management systems, one can improve the traceability and transparency of supply chains that are traditionally highly complex and opaque [8]. Improved traceability is a key feature in order to combat illegal trade and exploitation of vulnerable resources and areas.

However, introducing new technologies in well-established fields and operations is a challenging task. There are often social and traditional aspects involved [26]. In this master thesis, we aim to explore the scalability of blockchain-based supply chain management systems. If a software solution does not scale well, it is badly suited for industry adoption. If, on the other hand, it can be demonstrated to function well with increased workloads, it can be offered as a viable alternative, and can aid in the development of sustainable industries.

1.2 Sustainability

Sustainability is a vast issue, with applications in nearly all industries. For the purposes of this thesis, we look at sustainability in the context of the UN's 17 goals for sustainable development (SDG) [39]. The goals comprise a wide variety of topics, from human rights to economic development. The aspects we would like to highlight, are the calls for technological innovation and improvements in the industries.

Global energy production is still highly reliant on fossil fuels, although renewables are on the rise [51]. However, as many as three billion people lack access to clean energy to use for cooking [42]. Food loss and food waste are major issues, with as much as 1/3 of the global food production going to waste, while hundreds of millions starve [50][41]. All the while, the UN and other organizations are issuing

warnings about the impending dangers of climate change. The issues are on a global scale, and can only be addressed through disruptive changes to current practices. As part of meeting the global challenges, the UN calls for technological innovation [40]. However, as highlighted above, there are significant differences between the rich and poor in the world. For an innovation to truly aid sustainability, it must be accessible to all, not only to those who can already afford to pay the price.

1.3 Blockchain

Blockchain is a fairly novel technology, having been proposed in 2008 and put into use in 2009 [38]. Although the blockchain technology was originally created for the purpose of supporting the digital currency Bitcoin, it has come to be used in numerous fields. Even within the field of cryptocurrencies, Ethereum offers a vastly different approach from Bitcoin; an entire ecosystem with its own programming language, centered around a cryptocurrency based on blockchain [11].

Blockchain is a distributed technology. Rather than having a centralized server, with high requirements of performance and reliability, there are several small peers, who all share the responsibility of maintaining the data. Another key feature of the Blockchain technology, is its immutability. Any data that registered in the blockchain is stored permanently, and cannot be changed.

The distributed and immutable nature of the blockchain makes it a good candidate for building supply chain management systems. A problem with conventional systems, is that each party involved will have their own proprietary database. This means that each time a product changes hands, it must be registered as a completely new product. With a distributed system, all parties are aware of the existence of each product, and they all have access to the same base data. Consequently, when a product is handed over, it is only necessary to register a transaction of the trade having taken place. Hence, a complete log of the path taken by the product from source to destination is available. The immutability means that no parties can conduct fraud by changing the properties of a product that has been registered.

1.4 Objective

Several blockchain-based supply chain management systems exist at the time of writing. The objective of our research is to look into how such systems perform with different work loads, and to judge what the performance implies with regards to sustainability. To this end, we propose the following research questions:

- How will a blockchain-based supply chain management system perform on different scales?
- How viable is a blockchain-based supply chain management system for use in sustainable industries?

In order to answer the research questions, it is necessary to perform a series of tests. Without access to a real supply chain, the alternative is to use a simulation. By defining the structure of a supply chain, where a product is transferred from source to destination, it is possible to create a blockchain-based management system, where all transactions are logged.

Using the simulation, it is then possible to carry out tests, and measure the performance of the system. The data from the measurements can be analyzed to provide

insight into how a blockchain-based supply chain management system works, and how it is affected by changing various parameters.

1.5 Contributions

By gathering and analyzing data from the tests, we aim to make certain contributions to the literature. The research questions define the direction and scope of the research, but it is also necessary to have an idea of what is expected to be the outcome.

Research question 1 relates to the scalability of blockchain-based supply chain management systems. Scalability can be measured, and hence offers a starting point for a quantitative discussion of the matter. Through this discussion, we aim to contribute to the understanding of such systems, and how the systems might perform under increasing loads and growing network architectures.

Research question 2 looks at how viable blockchain-based supply chain management systems are for sustainable industries. Viability is not measurable in a quantitative manner, as such making it more natural to take a qualitative approach to evaluate this research question. By linking the results of the tests with sustainability, we want to provide a discussion regarding the potential benefits and drawbacks to utilizing a new technology, in the context of the UN goals for sustainable development.

Figure 1.1 illustrates the structure of the topics of the thesis. The two fundamental concepts are blockchain and sustainability. Blockchain is a technology with a wide variety of use cases, and for this thesis the emphasis is on supply chain management systems. Through utilizing blockchain-based supply chain management systems, it is possible to address the UN goals calling for reducing world hunger, and the general call for technological innovations across industries and borders. In order to look at a supply chain, a simulation will be used, as a fully functioning system would be too complex.

Through the process of developing the simulation, experiences with working with blockchain technology and supply chain management systems will be gathered. These experiences can then be used for a discussion of the availability and complexity of such systems. The measurements taken from running tests on the simulation, will provide data to analyze the performance and scalability of the system, as well as provide insight into the technical requirements to run a similar system on a larger scale.

1.6 Outline

The rest of the thesis is outlined as follows:

In Chapter 2 we present the background for the thesis. This includes a part on the motivation behind choosing our research topic, prior work done by us, as well as describing the problem we will be looking into. The motivation serves as a background for both the current research, as well as the systematic mapping study carried out in preparation for it. We present the methodology and findings of the study, and relate it to this thesis. We also discuss the threats to the validity of our study.

Chapter 3 describes the methodology utilized in this thesis. We first present our working hypothesis and research questions, with an explanation of the reasoning and motivation behind them. We conclude that running a simulation will be necessary in order to achieve our goals. Next we discuss the use of existing solutions,

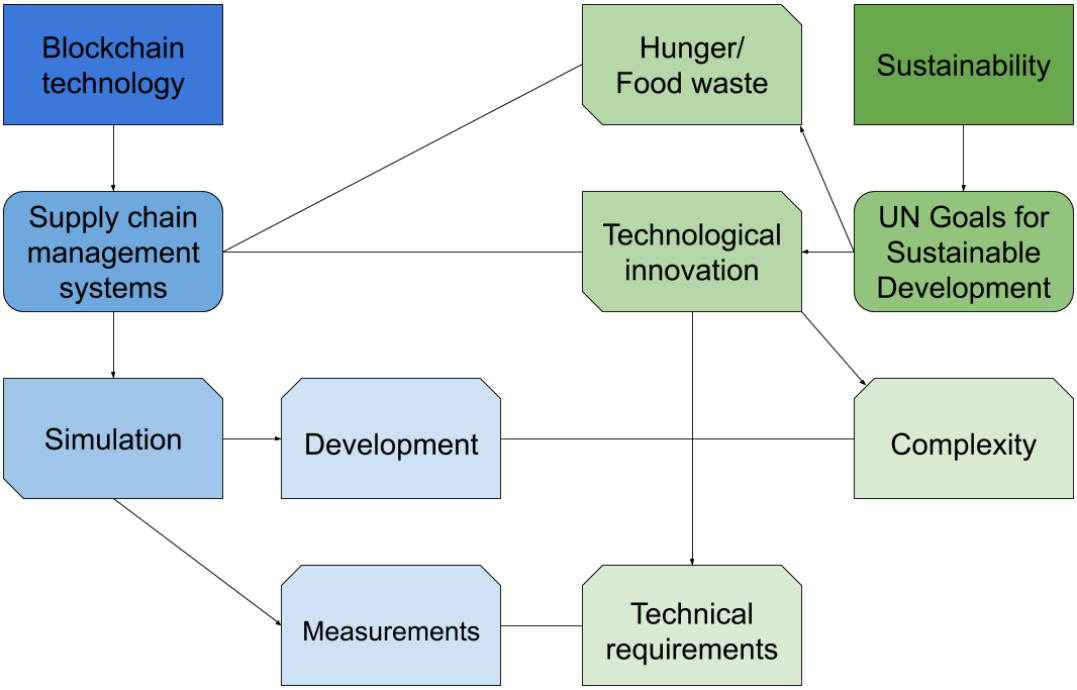

FIGURE 1.1: The relationships between the various parts of this thesis

upon which we can build our work. We mention benefits and drawbacks to using open source projects, and ultimately decide that they are the most beneficial starting point for developing our simulation.

In chapter Chapter 4, we talk about the main phase of the research. Initially we discuss the phases we went through, and the different attempts we made before arriving at our final solution. Next we present the stages of development of our simulation. We show code snippets and explain their functionality and reasoning. At the end of the chapter, we explain how we will perform the tests, and with what parameters.

In Chapter 5 we present the results from carrying out our research. We present the series of tests carried out, with the corresponding data. We then introduce a selection of graphs, in order to better illustrate our findings. We then talk about the findings, pointing out the most important parts, and explaining the causes, with a final discussion of the shortcomings of our simulation and tests.

For the discussion in Chapter 6 we first talk about the various challenges encountered while working on this thesis. Following that, we focus on discussing and analyzing our findings.

In Chapter 7, we discuss our experiences from working on this thesis. We present our final findings, discuss possibilities and recommendations for future work, before finally reaching the conclusion.

Chapter 2

Motivation and Background

In this chapter we will present the motivation and background of the thesis. First we include a discussion of the motivation behind the research topic. Next is a presentation of a literature review, in the form of a systematic mapping study, carried out in preparation for this research. We explain the methodology, the process and the findings. Finally we relate those findings to the research topic of this thesis.

2.1 Blockchain

Blockchain is described as a distributed ledger [38]. What this means, is that rather than having a centralized server which contains all the data to be stored, the data is distributed among all the users. The network consists of a collection of nodes. The nodes are competing to solve a mathematical problem, and the node that wins gets the privilege of creating the next block to be added to a chain of blocks, containing new transactions. The data that is stored is immutable, meaning that it cannot be changed; anything that is stored on the blockchain will stay there forever. In order to make changes, one has to perform transactions. To validate each transaction, the peers in the network need to reach a consensus. A consensus would mean that a majority of the peers agree upon which transactions have taken place, and these are then permanently stored in blocks on the blockchain. In the context of Bitcoin, users A and B can have 1 and 0 bitcoins respectively. It would be impossible for a user C to make the claim that user A transfers the bitcoin of user A bitcoin to user C, unless C controls the majority of nodes in the network. However, user A can tell the other peers that they want to transfer their bitcoin to user B, and this transaction will eventually be added to a block, which will then inform all peers that it is now B who has the bitcoin, while A has 0.

However, the innovation around blockchain-based technologies has certain drawbacks. The network of peers running Bitcoin consumes the same amount of energy as the country of Ireland [55]. The main reason the Bitcoin network consumes such a staggering amount of electricity is because of how a bitcoin is generated. As it is a completely digital currency, there are no coins to be minted or bills to be printed. Instead, there is a process commonly referred to as *mining*, where the peers compete to be the first to complete a challenging mathematical problem, with the winner gaining the privilege of creating the next block and receiving a reward of bitcoins [38].

Bitcoin is set up such that a new block will be created approximately every ten minutes, by automatically adjusting the difficulty of the mathematical problem [38]. It then becomes obvious, that when more peers join the network to compete, the

computational power needed to solve the problem also increases. In fact, specialized bitcoin miners will often choose locations for their operations, in places where electricity is cheap, in order to maximize their profits [60].

The Ethereum network struggles with the same problem as the Bitcoin network. The total energy expenditure of the peers in the network, has reached a level comparable to the nation of Bolivia [7]. The primary reason behind the massive energy consumption in both these networks, is the technique utilized for the mining. Both Bitcoin and Ethereum use a concept called *Proof of Work* (PoW) [38][12].

The basic principle of PoW, is to have the miners prove they have put in an effort in order to be allowed to create the next block. If the validation had been implemented as a simple matter of voting, it would have been too easy to overwhelm the network with a large number of computers, such as a botnet. By making a requirement of computational power, it becomes a lot more expensive, but with the consequence of requiring high amounts of power.

However, PoW is not the only validation algorithm. In fact, the Ethereum Foundation is looking to implement an alternative algorithm in the near future [12]. The new algorithm uses a concept called Proof of Stake (PoS) [13]. Rather than expending computational power to solve a mathematical problem, PoS is based on a weighted voting system. "Stake" comes from the fact that the weight for a validator's vote is based on the amount of coins they make as a deposit. The winner is then awarded coins, similar to how the miner who managed to solve the problem would receive a reward. As the basis for PoS is based on simply voting and waiting, the massive energy consumption generated by performing math operations is avoided.

Nevertheless, cryptocurrencies are not the only field in which Blockchain technology can be utilized. In fact, Blockchain is often considered to have a wide variety of use cases, with the potential to cause drastic changes [52][46]. Among the possibilities are energy production, infrastructure and even protection of intellectual property [33][31][16]. As Blockchain can be though of simply as a technique for storing data and transactions in a distributed manner, the possibilities are vast.

2.2 Sustainability

According to the UN and several other organizations, the world is facing a climate crisis on a global scale. To address the climate crisis, the UN published a list of 17 goals for sustainable development (SDG) [39]. These goals identify key issues for a sustainable environment for future generations, and discuss how to face them.

Furthermore, the UN is behind another report called the World Atlas of Illicit Flows [44]. Among the issues highlighted in this report is that of environmental crime. There are several aspects to environmental crime, one of which is sale of illegally procured goods as legitimate products. Not knowing the origin of a product can be a serious concern, as it might turn out that it comes from an area where it is supposed to be illegal to operate. Even if a shipment is supposed to come from a legitimate source, bad products can be mixed in, to hide their origin.

Being able to know the origin of a product is called traceability. As a product moves along a supply chain, from origin to destination, it can be increasingly difficult to be sure of its true source [14]. However, this knowledge can be key to ensure the sustainability of certain industries, such as seafood [4]. Transparent, traceable supply chains are hence highly related to the matter of sustainability. Such supply chains can also help avoid losses during shipping, as well as help guarantee the quality of the product at arrival [5][53].

2.2.1 Goal 2

SDG 2 is called "Zero hunger" [41]. Even though enough food for the entire world's population is produced, hundreds of millions live under starvation. This is not only an issue of distribution, but also of food waste and food loss. It is estimated that as much as 1/3 of all food produced goes to waste [9]. *Food waste* refers to food that is not eaten by a consumer, while *food loss* refers to food that is lost during production or transportation.

Food loss is an example of an issue with a variety of consequences; there is a direct economic loss for each food item which is lost, and to compensate for lost produce, it is necessary to produce more, which takes up areas that could be used for other purposes [56]. Furthermore, the food that is lost could have been used to feed some of the people who lack food security.

According to ReFED, food loss happens along the entirety of the supply chain [50]. However, an estimated 84% of the loss happens other places than at the producer, meaning that it occurs during transportation. Hence, the argument can be made that improving our supply chains can help reduce food loss. If the amount of food that is needlessly lost during transportation is reduced, that can in turn aid in reducing world hunger.

2.2.2 Goal 7

SDG 7 makes the case for achieving clean and affordable energy for all [42]. According to the UN's numbers, 13% of the people in the world lack proper access to electricity. Furthermore, three billion people use non-clean fuel sources for heating and cooking. Using such fuels in those ways not only contributes to greenhouse emissions, but also compromises people's health, due to indoor air pollution. This goes to show that even ignoring warnings of climate change, providing clean energy to more people has a humanitarian motivation.

As a whole, so-called modern renewables stood for an approximate 10.4% of the total energy consumption in the world in 2016 [51]. Meanwhile, fossil fuels accounted for 79.5%. Decisions at a national and institutional level are necessary in order to make any significant changes to these numbers. Even so, individual choices are possible and do make a difference. By making technologies for private energy production and sharing more widely available, individuals and communities can come to supply themselves with clean energy.

Localized and privatized energy production has multiple facets. By setting up energy grids separate from the main grid of a country or area, microgrids are created [30]. Microgrids can be further developed to create so-called smart grids, where electricity is sold and bought automatically as needed using smart contracts [28].

2.2.3 Goal 13

SDG 13 is a call for action to combat climate change [40]. Goal 13 is somewhat less specific than the two goals discussed above. While SDG two and seven are aimed towards clear industries, SDG 13 is more comprehensive, involving all nations and industries as a whole. Tackling issues at a global scale is both overwhelming and infeasible, but we make note of the following point from the website [40]:

- Major institutional and technological change will give a better than even chance that global warming will not exceed [the 1.5°C] threshold

The threshold referred to above, is the goal of keeping global warming under a 1.5°C increase compared to global temperatures in 1850 to 1900. This temperature goal is in accordance with the Paris Agreement of 2015 [43]. The temperature goal was set forth as the limit if we are to avoid significant climate change and rising sea levels.

"Major institutional and technological change" points to a need for technological innovation across nations and industries. Rather than getting lost in the big picture of waiting for governments and major corporations to make the decisions, it is possible to take steps towards utilizing more sustainable technologies at all scales.

In order to make changes at all levels, it is essential that new technologies be available to everyone, not just the rich. While countries in Europe and North America have come comparatively far with adopting renewable energy into their electrical grids, people in developing countries in Sub-Saharan Africa depend on unsustainable biomass [51]. Furthermore, the primary victims of environmental crime are often the local population in rural, under-developed parts of the world [44]. By making innovative and sustainable technology widely available, the livelihoods of people around the world, not just in developed countries, can be improved.

2.3 Literature Review

During the fall of 2018, we conducted a literature review with a systematic mapping study, with the research questions: 1) How is blockchain technology related to sustainability? And 2) how can blockchain technology be used to develop sustainable technology? These questions were inspired by the UN goals for sustainable development [39]. Of the goals, goal 13 is the most relevant [40]. The goal is aimed towards climate action, saying that the global society needs to act in order to fight climate change. One of the ways to fight, is through technological innovation. Blockchain is a technology lauded as being able to disrupt several fields [52]. As such we believe it relevant to look into what is being done with blockchain technology in the context of sustainable development and sustainable technologies.

The systematic mapping study was transformed into a conference paper of eight pages, submitted to and accepted by the 2nd International Workshop on Emerging Trends in Software Engineering for Blockchain, held in Montreal 2019 [57]. The full published paper can be seen in Appendix C.

2.3.1 Motivation and Background

We identified two main areas to examine: smart grids and supply chain management. Aside from goal 13, goal 7 relates to affordable and clean energy for everyone. One way to work towards this goal is to make local microgrids available to people. However, looking at a community level, not everyone might be able to afford a solar power setup or similar for themselves. Or the power cells might be owned by the community as a whole. Then it becomes imperative to have a way to distribute the power generated. Instead of having the users sell and purchase electricity themselves, it is more sensible to have an automated system do it. By utilizing computer technology, it is possible to make so-called smart grids, where energy distribution is done automatically according to supply and demand. Through making smart grids more viable for everything from undeveloped, off the grid areas, to people wanting to decrease their dependence on centralized power generation, it is possible to work

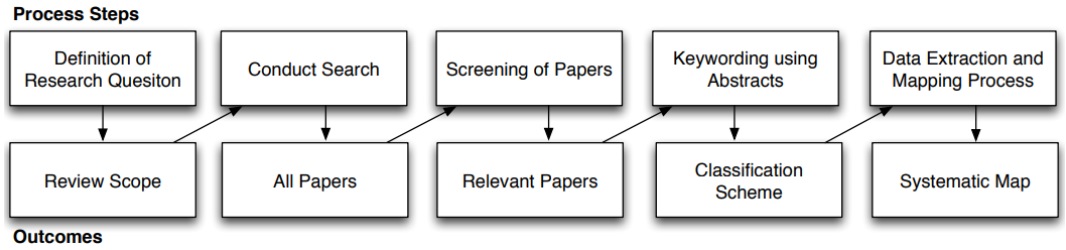

FIGURE 2.1: The process for a systematic mapping study in software engineering [47].

towards UN goal 7. We wanted to look into how blockchain technology is being utilized in this field, and what potential benefits it might offer.

The motivation to look at blockchain technology comes from several points. For one, it is considered a technology with a potential to disrupt several industries, leading to innovative solutions and possibilities [52]. In tandem with the UN's call for technological innovation to address the issue of sustainable development, it seems logical to look at the potential for utilizing blockchain technology for this purpose.

Furthermore, blockchain technology is in an interesting position regarding sustainability, as certain widespread applications of it consume staggering amounts of electricity. According to a study from 2018, the network consumes a similar amount of power to the country of Ireland [55]. The Ethereum network, also a cryptocurrency, but with a purpose-built programming language to create native applications, also consumes a high amount of power. At the time of writing, the Ethereum network has an energy consumption similar to the Democratic Republic of the Congo. These numbers are clearly at odds with the idea of using blockchain technology for environmentally friendly purposes, and hence serve as motivation to explore the topic further.

2.3.2 Methodology

We used the methodology for systematic mapping studies in software engineering, as described by Petersen et al [47]. Systematic mapping studies have primarily been utilized in the field of medicine, but it as also applicable in others. Petersen et al. present a thorough methodology for how such studies can be carried out in the field of software engineering.

[Describe in a couple of paragraphs what was done, and how the steps relate to the figure]

The first step was to formalize the research questions in order to define the topic and scope of the project. Well-designed research questions are essential to conducting research, as they might mislead the investigations if they are poorly specified. The research questions used for the project were:

Based on the research questions, we developed a set of search strings to be used in selected online libraries. These search strings were based on cursory searches in the relevant libraries, as well as the topics of the project. By doing cursory searches, one can see whether one can expect to find a reasonable number of papers, and whether these papers appear to be relevant. The search strings we used were:

Using these search strings, we identified a total of 535 papers. By removing duplicates and otherwise unusable papers, we were left with 486. The next step is to review the papers, primarily by looking at titles and abstracts, to judge whether they are relevant. Finally, we were left with 62 papers. However, two of these papers were

Research Question	**Explanation**
1: How is blockchain technology related to sustainability?	The blockchain technology is being used for a variety of purposes. Some of the use cases might have a net positive effect on climate change, whereas others could cause significant harm.
2: How can blockchain technology be used to develop sustainable technology?	In which direction should further research and development of blockchain technology be headed?

TABLE 2.1: Research questions.

Search strings
blockchain AND "climate change"
(blockchain OR cryptocurrency OR bitcoin) AND ("climate change" OR green OR "green energy")
(blockchain OR bitcoin OR cryptocurrency) AND (climate OR green OR "green certificate")
blockchain AND "supply chain"
(blockchain OR bitcoin OR cryptocurrency) AND (climate OR green OR "green certificate") AND ("supply chain")

TABLE 2.2: Search strings.

not available to us through the licenses provided by NTNU, and were hence left out. In the end, the study was then based on 60 papers. While carrying out the review, it is useful to make note of recurrent keywords, in order to categorize the papers.

Creating categories, or knowledge areas, is the next step. As we found no previous studies on the same topics, we had to create our own classification scheme. This is done by looking through the papers, and finding commonalities. After deciding on a set of categories, an attempt is made to put every paper into appropriate categories. If a significant number of papers either do not fit well, or fit into two or more categories, it is necessary to further specify the categories. For our classification scheme we used the following:

The final step is to put all papers into their corresponding categories, and determine the type of research presented in each paper, as well as the contribution type. The research types and contribution types were also taken from Petersen et al [47]. The result of our systematic mapping study can be seen in figure 2.2.

One significant thing that we noted, was that the amount of papers published per year had increased noticeably over the last couple of years. This trend can be seen in figure 2.3. As such it can be noted that there is a growing interest in the topic of sustainability, in relation to blockchain technology. We hypothesize that this is because of a growing interest in sustainable development in general.

However, despite the significant increase in papers over the years, it cannot yet be called a considerable amount of papers, by pure numbers. This points towards the technology still being quite new and research in the field still limited. Further supporting this view, we can look at figure 2.4. In this graph, we can see that there is a lot of validation research compared to the other kinds of research. Validation research looks at whether a proposed solution is viable to be put into use. It is hence a step along the way for the technology to be ready for the public. However, there

Knowledge area	Description
Supply chain for visibility	Making supply chains more transparent to the actors involved, to keep track of each step along the way
Supply chain for security	Improving the resilience of supply chains against malicious actors, including counterfeit protection
Supply chain for quality	Improving supply chains to ensure the quality of the end product, reducing spoilage and product loss
P2P energy trading	How to make Peer-to-peer (P2P) energy trading available to the public
Energy trading security	Discussing the security of using smart grids, or P2P energy trading in general
Blockchain energy efficiency	Proposals to improve the energy efficiency of future blockchain implementations

TABLE 2.3: Table of knowledge areas

are almost no experience papers, meaning that almost no research has reached a point where it can be evaluated with regards to how it has performed in use.

2.3.3 Smart grids

Smart grids are a term used for micro grids with automatic trading of energy built in. Micro grids are electrical grids that are not connected to the main power grid of the area. Micro grids have several use cases, from people wanting to produce their own power and sell the surplus to make a profit, to getting electricity to an area not covered by the main grid, such as poor, rural areas. By making these grids "smart", one can utilize algorithms that automatically buy and sell energy, so that the users do not need to manually take care of it themselves. In order words, it streamlines the process and makes such systems easier for the end users to take advantage of them.

We found that there is a considerable amount of research going on in this field. However, as the field is still fairly young, there is somewhat of a lack of proper solutions, meaning proposals ready to be put into use by the public.

Nevertheless, there are promising and interesting solutions. Mihaylov et al. propose a cryptocurrency based around supplying energy to a grid [37]. This is a stark contrast to the Bitcoin network, which at the time of writing consumes a comparable amount of power to the country Ireland [55]. The proposal revolves around so-called *prosumers*, as opposed to consumers. While a consumer stands for a demand, a prosumer is a supplier. By producing surplus energy, a prosumer can sell energy to consumers connected to the grid, in order to earn what Mihayl et al. call NRGcoins, the currency of the network.

2.3.4 Supply chain management systems

Supply chain management systems are essential to keeping track of goods as they make their way from production to consumption. However, they are not foolproof, and errors and flaws can lead to significant losses. These losses can be because of goods being lost along the way, or because of insufficient storage. For example,

if you are shipping fish, it is essential to keep the fish in units with temperature control, or else it will spoil. If the supply chain management system does not allow for monitoring the temperature, it is impossible to verify whether it has been stored in a proper manner.

Traditional Systems

There are several issues with traditional supply chain management systems, identified in the mapping study, that we would like to highlight. The first is that they are primarily centralized. That is to say, one depends on one or a handful of centralized databases. The individual parties do not necessarily have control over the data registration and storage themselves. This can be problematic, because it becomes very easy for malicious agents to tamper with the records. One can never be entirely sure that a server is adequately secured. With a centralized database, there is a single point of failure for the information security of the network.

While we have referred to traditional supply chain management systems as centralized, they are not usually completely centralized, with a single database used by all participants. Rather, the different companies involved will have their own databases, with more or less proprietary systems. This means that information cannot flow freely from one to the other, a product will be stored with a different format in each database. Each time a product changes hands, a new database entry will have to be made, instead of registering a change of owner.

With an individual company having its own database, it also heightens the risk of fraud. In the World Atlas of Illicit Flows, the authors highlight an issue where products harvested from a sustainable source are mixed with illegally harvested goods [44]. When a company has full access to both the goods and the database registering the goods, it is fully possible for them to register fake information. Thus it can be claimed that the illegally harvested goods come from the same legitimate source as the rest of the shipment.

Another issue we want to address is the lack of continuous monitoring. In traditional supply chain management systems, it would only be possible to monitor the product being shipped at certain points. Conversely, this means that for the majority of the duration of the shipping, the condition of the shipment is unknown. An obvious concern in this case is, as mentioned above, perishable goods where factors like temperature control are crucial.

Advantages to Using Blockchain

Our mapping study found that many blockchain-based supply chain management systems offer methods to have continuous monitoring of the goods being shipped. This can be achieved through means of using Internet of Things (IoT) devices, that upload information to the supply chain [29][5].

The information that can be uploaded is only limited by what can practically be included with a tag. For example, for a shipment where storage conditions are vital, a temperature sensor is a natural inclusion. Being able to know whether the environment of the products have at some point become too hot is extremely useful. For fresh food, it is vital in order to keep it from spoiling. An even more important use case would be a shipment of medicine such as vaccines. If a vaccine is subjected to too high a temperature, it can become inactive, and hence useless. Being able to detect such events is thus highly valuable.

Challenges

A problem that has been given some attention, but as of yet not been solved, is that of how to make tamper-proof product tags. Some researchers have suggested tags using RFID or NFC technology [54][53], however these are not reliable. They can be cloned or the data can be modified [1]. Tag cloning can be avoided with sufficiently complex identifiers, such as the craquelure method proposed by Hepp et al [15]. Regardless of how refined a tag might be, the biggest problem remains: the tag itself can simply be moved from one product to another, or the contents of a container can be changed. Blockchain technology is rooted in software, and consequently does not offer a way to deal with this issue. As such, we will in this thesis not try to tackle the issue of physical identifiers.

2.4 Supply chains

As the main topic of this master thesis well be supply chains, we will in this section we will go into more detail regarding supply chains and supply chain management systems. More specifically, the topic is blockchain-based supply chains. Smart grids also have a lot of potential, but from the mapping study, it seemed as though there are good solutions available, that have been developed further than what exists for supply chain management systems. With regards to supply chain management systems, one of the main issues is to have reliable solutions ready to be implemented in real life scenarios. By looking into the viability of blockchain-based solutions, we are hoping to contribute to the development and evolution of supply chain management systems.

2.4.1 Findings

What we found in the mapping study was that there are several existing proposals for blockchain-based supply chain management systems, but they are still at a stage where they are being researched. For example, we found no papers looking into the impact of actually implementing such a system in real life. The only experience paper we found, meaning papers looking at something already put into to real-life use, was about the use of ontologies for blockchain design [27].

One of the issues that first motivated our choosing of the topics blockchain and sustainability, was that of environmental crime. However, we did not find any papers looking into how to utilize blockchain technology to prevent this.

2.4.2 Future Work

In the mapping study, we also proposed possible future work. One of the avenues of research we suggested was improved user involvement, specifically including the end user. One of the benefits of providing proof of provenance through tracing a product's life cycle using the blockchain, is that an end user can be sure of the source of the product they consider buying. This can be a possible tool to combat environmental crime, as there would an incentive for legit producers to prove the origins of the products they are selling, since a consumer will most likely prefer to buy a product that can be proven not to be from an endangered area. Even so, we only found a single paper where the issue of user involvement was highlighted [3].

A clearer approach on the topic of environmental crime would also be of interest. The need and interest for this use case is highlighted not only by the UN report on

illicit trade, but also papers such as Fish2.0 and Future of Fish [14][8]. These papers point out the opacity and complexity of supply chains in the context of the seafood industry.

However, simply creating a software solution is not enough. In a paper by Jabbar and Bjørn, they describe the complex issue of getting an industry to accept a new technology [26]. They found that in addition to the information structure, there is a social aspect, with practices between the various partners built up over time. Replacing a part of a deeply ingrained practice is a significant challenge. Rather than just introducing a new solution, it is beneficial to know that it can actually improve current practices.

Finally, we have claimed that our research is quite novel. In that case, in stands to follow that further research on the same topic would be of interest. Research is a collaborative effort, and it is meaningless if no one else offers input, or further improves upon what has already been done.

2.4.3 Threats to Validity

Even when trying to follow best practices and conduct research in a sensible way, there are risks of errors or simply missing things. One cannot take everything into account, and there are as such risks of one's research having flaws that might make it invalid in certain aspects.

Missed Papers

Although we made an effort to find a broad selection of relevant papers, we have also realized that we missed some. For example, there are in fact blockchain-based supply chain management solutions that have been made for the industry, and are available right now [14][8]. However, by using a limited selection of online libraries and search terms, there are some papers that will not be found.

Classification Schema

As we found no previous classifications for a topic similar to ours, we were left with the task of creating our own. Coming up with a schema from scratch grants us the freedom of finding the solution we think is the best, but at the same time it places upon us the responsibility of making something new. A classification schema should ideally be based on objective criteria, and be subjected to review by independent researchers. However, as ours was a new schema, created for the express purpose of our research, it is bound to be influenced by our own opinions and biases. Ideally, the development of classification schema would follow an iterative process, where subsequent researchers improve upon the categories first created.

Topic Selection

In our research, we decided to limit ourselves to the topics of supply chains and smart grids. While we might have gotten relevant or interesting regarding how these relate to sustainability, we may have missed the bigger picture of using blockchain technology for sustainable development. As such, we might have overlooked points relevant to the overarching topic we are aiming to research.

We tried to mitigate this factor by including more general search strings, which would include papers using blockchain in different ways to combat climate change. However, few such papers were found. There are a wide variety of factors that might

play into this. There could simply be a lack of such papers that have been published, or we could have looked in the wrong way, in the wrong places.

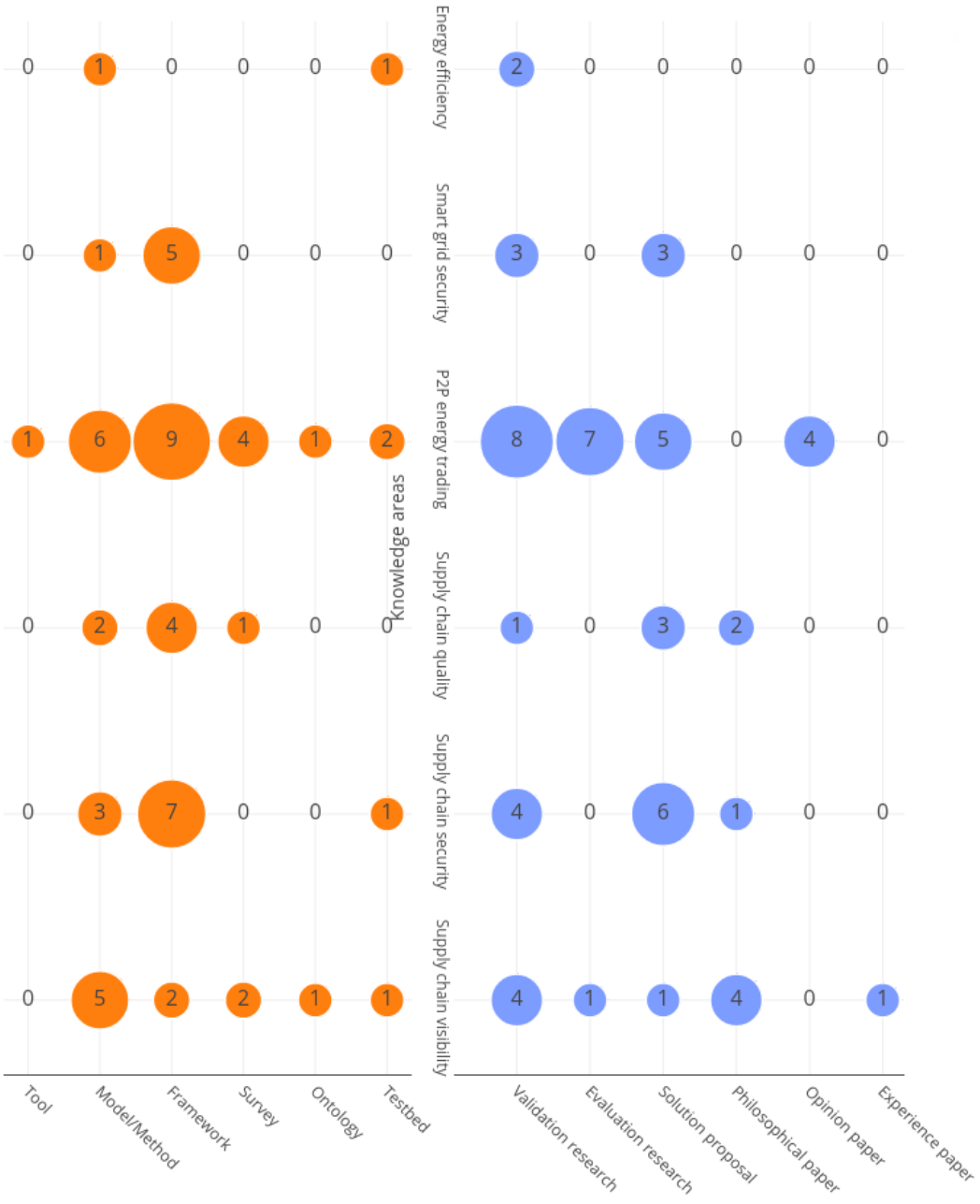

FIGURE 2.2: Bubble plot presenting the results of the systematic mapping study.

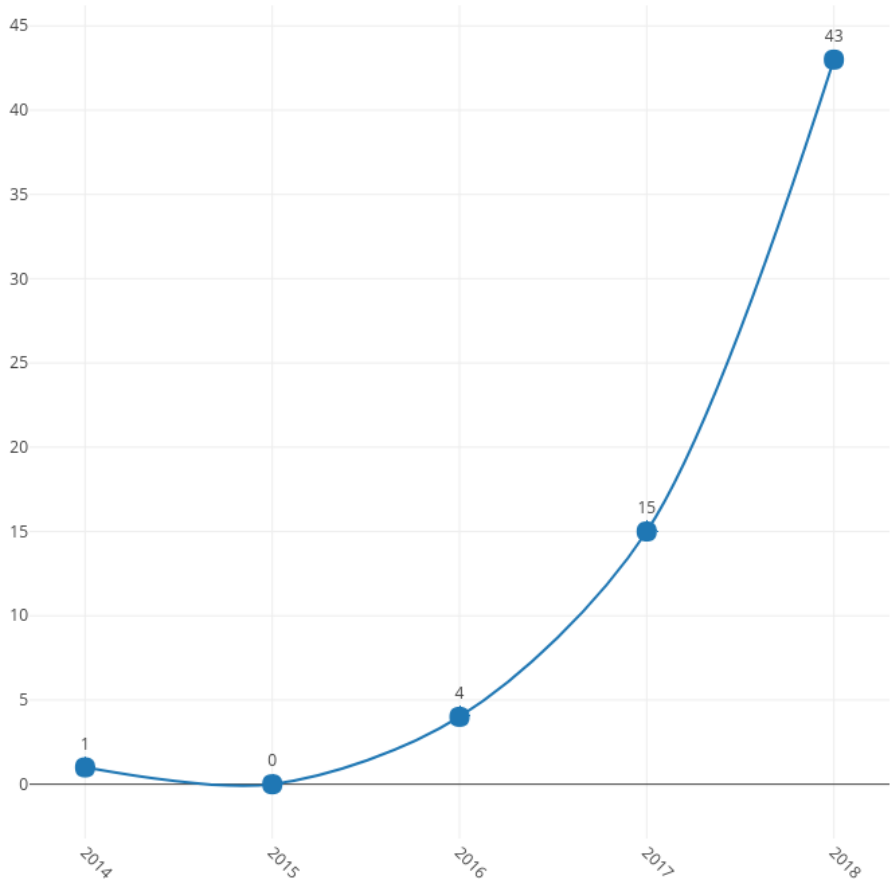

FIGURE 2.3: Number of publications per year

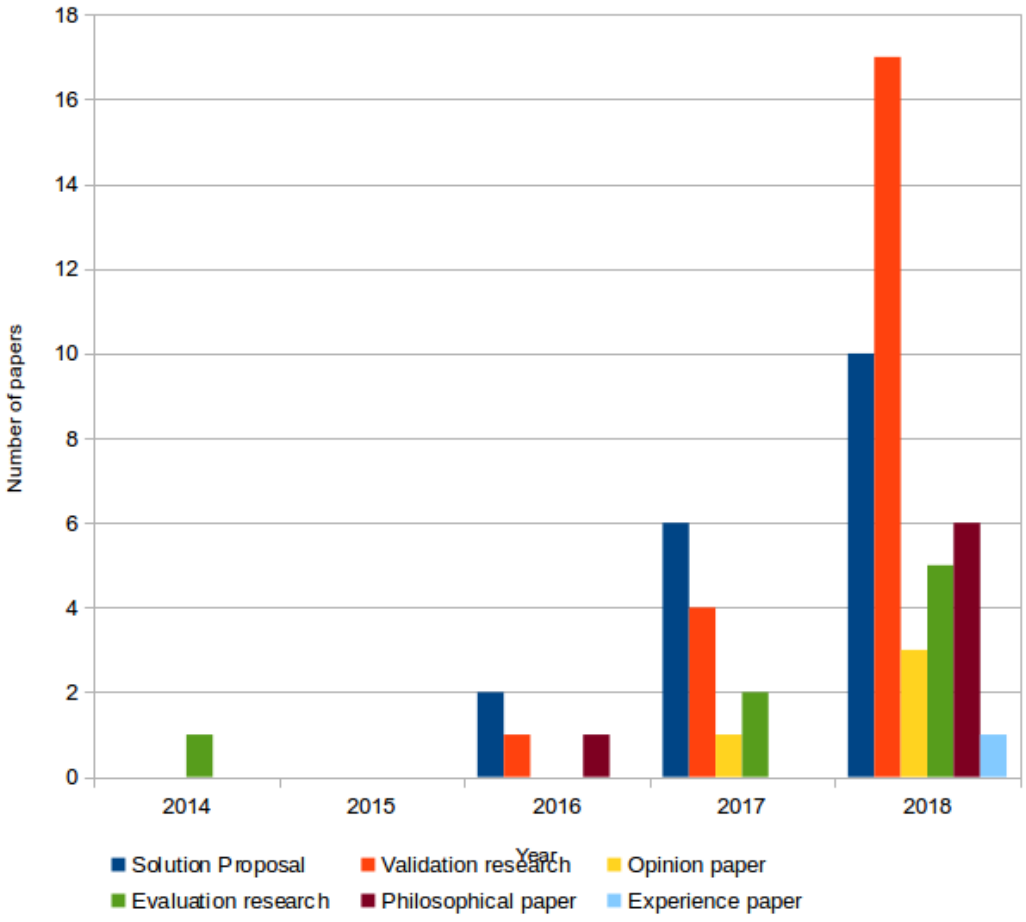

FIGURE 2.4: Number of papers by type of paper by year

Chapter 3

Methodology

Chapter 3 describes the methodology utilized in this master thesis. First the research questions and hypothesis are presented and explained. Next is an explanation of our decision to base our simulation on an open source project. We then describe the tests that will be run on the simulation, before concluding with the working assumptions for the supply chain for the purposes of this research.

3.1 Hypothesis and Research Questions

The basis for this thesis is the relationship between blockchain-based supply chain management systems and sustainable development. The systematic mapping study completed prior to this thesis suggests that blockchain-based supply chains are worth exploring further. While they are still a ways away from being implemented on a significant scale in the real world, they show promise of becoming more widespread. However, more knowledge is required regarding how such systems perform, as potential users need to be convinced that a new solution is worthwhile, and worth the effort of replacing the old [26].

Furthermore, with a growing interest in sustainability and sustainable development, improving supply chain management systems in order to reduce losses is a highly relevant topic. As put forth in chapter 2, we have found that there is a notable amount of research on how to implement supply chain management systems utilizing blockchain technology.

We hypothesize that a blockchain-based supply chain management system is scalable. By scalable, we mean that the performance loss and resources necessary when facing increasing workloads does not increase at too high of a rate. For example, if the required storage capacity increases exponentially under certain conditions, it cannot be considered scalable, as it would very quickly become unreasonable to store that much data.

In order to test our hypothesis and answer the research questions, it will be necessary to have a system to test. A full-scale implementation would be very unrealistic, as supply chains are usually highly complex and geographically dispersed. Hence it will be necessary to have a simulation where we can control certain attributes and parameters.

3.2 Existing Solutions

Building a full blockchain-based supply chain management system from scratch would be a vast undertaking. An alternative is to use a pre-existing code pattern or solution proposal as a basis, and further build upon it. There are several solutions available through various open source projects. Open source projects are to be

Research Questions	**Explanation**
How will a blockchain-based supply chain management system perform on different scales?	In order to assess whether a blockchain based supply chain management system is scalable, it is necessary to perform tests that show how it performs when subjected to changing parameters.
How viable is a blockchain-based supply chain management system for use in sustainable industries?	Sustainable technologies should be accessible and usable by everyone. If the cost of production or implementation is too high, either in pure monetary terms or in the workload required, it is highly unlikely that the technology will be put into widespread use.

TABLE 3.1: The research questions for this thesis

preferred, as they provide you with full access to the code and allow you to make changes as you see fit. If one were to purchase a proprietary solution, it would only be possible to utilize what is made available. Proprietary solutions also put stricter limits on what you are allowed to do with it, while open source solutions in general have very permissive licenses that lets you change them as you see fit. Open source projects motivate innovation and make it easier for people to collaborate and further develop ideas [10].

A downside of open source projects, is that they are in general community driven [25]. This means that there might be a lower standard of quality, as there isn't necessarily as much of an incentive to create a product that can be justifiably sold at a high price. As such, one has to consider the quality of the documentation and code architecture, and not only the quality of the code itself or how well the implementation fits one's requirements.

Despite the weaknesses, we still believe open source solutions to be the best option. There are very few proprietary solutions available, and those that might exist would most likely be challenging to test and modify. Additionally there would be the problem of licenses, where we might not be allowed to actually publish research on them [10][25]. Building a full solution from scratch would require more time than we had available, which only leaves the option of open source projects.

Furthermore, by using an open source project as a base, we are obliged to keep our changes to the code open source as well [25]. Hence anything we create will be accessible to others, for them to make use of or improve as they want. By keeping our contributions available to others, we play a small role in providing new developments and innovations to the world as a whole.

3.3 Tests

After getting an open source solution running, we will conduct various tests. The goal with the tests is to see how certain parameters affect the performance of the system. That is, to see how well the system scales, and ideally identify bottlenecks. By doing this, we hope to see how such a system might perform when put into use, and on what scale it might be viable.

3.3.1 Parameters

There are a lot of possible parameters to test for when looking into performance and scalability. In order to make the work doable, we will only be testing a limited set of parameters. These parameters are selected based on what is viable to change in our setup and that will offer relevant insight.

The main parameters to be used in this thesis are the number of nodes in the supply chain and and the number of products to be handled. The first parameter can be varied in regards to the amount of producers, shippers and consumers. Changing the number of shippers will also increase the amount of steps that a product needs to be moved, hence increasing the number of transactions in the supply chain. Changing the number of products directly influences the number of transactions. We can also modify the frequency at which the products are created and transported.

The main parameters we want to explore are:

Parameter	**Scale**
Number of products	1,000s
Number of nodes	10s
Block size	Megabytes

TABLE 3.2: The parameters we will test for

By changing the first two parameters, there will some indirect effects: The number of nodes influences how many peers that it will be necessary for the network to notify. Additionally, the number of products as well as the number of nodes will directly influence the number of transactions necessary for the simulation to run its course.

3.3.2 Measurements

To look at the effects of the parameters, we will be measuring certain numbers, which can be seen in table 3.3.

The first point is when a new product is registered to the blockchain. This would be an event such as a fisherman catching a fish and registering it with the various data, for example weight and species of fish. This event is very central to a supply chain, as it is the starting point of the whole process. By measuring the time it takes to register a new product, we can see how long time it takes for all the participating parties to become aware of something happening.

The size of the ledger is an important measurement. It is unavoidable to have to store the data for all products, no matter if it is a blockchain-based system, or a more traditional centralized database. However, when distributing the database, every participant needs to store the same data. If it turns out that changing the parameters significantly affects the storage space required, one might have to reconsider the design. Growth at a too high pace is unsustainable, especially if we want our solution to be accessible to small actors.

The total execution time is to an extent a secondary variable to measure. A simulation will by definition have certain levels of abstraction from a real-life scenario. In a real supply chain, a product will be registered and shipped at a given time. The time taken for a shipment to arrive at its destination can be several days. Running a simulation where each step of execution takes days is infeasible at this point. As

Measure	**Explanation**
Time taken for product creation	Registering products to the blockchain is a key functionality of this system. If it takes too long time, it might inconvenience the parties involved in some way. While the time taken by an individual registry is not too important, if it turns out to scale badly, it will eventually be impossible to use
Time taken for updating the location parameter of a product	One of the main arguments for making the change from conventional supply chains to blockchain-based ones, is that the blockchain combined with IoT allows for continuous data updates
Time taken for changing ownership of product	Registering change in ownership is also a key feature. The importance of this measurement is also in the scalability
Total time of execution	By looking at the total time of execution, we can get an impression of the possibilities for parallelism in the network. If the time for carrying out a single action varies or changes with the different parameters, but the total time taken does not, the importance of the single event matters less
Size of ledger	The size of the ledger is perhaps the most direct measurement that can be taken. It has a direct influence on the required computer setup a company will need, and it will be interesting to see how the necessary storage might compare to a conventional database

TABLE 3.3: The variables we will measure

such, the total execution time can offer some added insight and aid analysis, but is in itself not a highly relevant measurement.

3.4 Practical Setup

Regardless of the specifics regarding the implementation, the testing setup will be the same. Initially, we will make a proof of concept on a local machine. This will be a supply chain management system simulation, running with all nodes on a single machine. The plan is to utilize cloud computing offered by Microsoft Azure [36]. This is in order to better simulate a distributed system, and to lessen the workload on a personal computer. The specifics of the configuration we will start with are as follows:

- Number of servers: 4

- Cores: 2
- RAM: 2GB
- HDD: 50GB
- Network Bandwidth: 20Mbps
- Operating System: Ubuntu 18.04

Both the local computer and the servers run Ubuntu. Using Linux makes it a lot easier to connect with the servers and set them up than when using Windows. Ubuntu is a widely used distribution of Linux, with a lot of information available. The authors are also more familiar with Ubuntu than with other distributions, which makes development for and with it easier.

We do not expect the simulation to be very resource intensive, and therefore choose a fairly lightweight setup. It might be necessary to expand the disk storage, if it turns out that the ledger grows very large. We do not want the servers to be the bottleneck of our simulation, and we will change the specifications if they turn out to be lacking.

3.4.1 Technical Requirements

The code will have to include certain functionalities, no matter the implementation details. That is to say, the core features of a supply chain management system. For our purposes, this includes being able to:

- Create new products
- Change the owner of the product
- Querying the blockchain for the products registered
- Determine whether a product has reached its destination

Being able to create a new product is a given, as it is the basis for the entire supply chain. However, it has some wider implications beyond simply logging the existence of an item. Physically speaking it would be connected to a tag of some kind, for example an NFC chip [53]. There is also the matter of notifying all the involved parties. In a centralized approach, one would only have to deal with the database one is in charge of, but in a distributed system, the registration must be broadcast. This can be a very involved process, but is already implemented in some open source projects like the Hyperledger Fabric [12].

Since the blockchain is immutable, changing the owner of a product also becomes a somewhat more challenging process, but it is also in many ways the basis of a blockchain-oriented method of operation. That is to say, the original purpose of the blockchain was to support Bitcoin, for making monetary transactions [38]. Changing the owner is also a transaction, as such it is not very different from dealing with currencies. In a practical implementation, it would be required for the receiving party to register the product with its tag in order to finalize the transaction. Hence it is confirmed that the product has reached its new location, and the transaction is immutably stored in the blockchain.

In order to keep track of the products in the supply chain, it is necessary to be able to query them. Querying the blockchain refers to asking for a list of the data

stored in the ledger. In our case the data would be information about the products being shipped. The information must include the identification of the product, but can be anything we think necessary or relevant. It will also be natural to want to query only for the products owned by oneself, as those will be the products you can interact with.

Similarly to the creation of a new product, determining that it has reached its destination is both very natural and fairly simple. In a real-world implementation, it would be done when the receiving party, the consumer, registers the transaction for receiving the product. Realistically, it would not even be necessary to take any special action for it, one could simply query the blockchain for products belonging to a consumer. In a simulation, we will take a little more care, in order to avoid doing unnecessary work, checking whether already arrived products should still be sent farther.

3.5 Assumptions

A real life supply chain is a highly complex infrastructure, which can span across the globe, with complex relations between the involved parties. A simulation will never be fully realistic, and we will here list some assumptions we are making in order to make the problem more manageable. We will also clarify some practical points.

3.5.1 Contents of the Supply Chain

The supply chain we are basing our work on is one meant for tracing shipments of fish from their source to when they arrive at their final location. More precisely, this can be from the time a fisherman catches the fish and attaches the sensor tag to it, to when a restaurant receives the fish and prepares it for a diner. As we presented at the end of chapter 2, there is a need for improved supply chains in the seafood industry, so we see it as a fitting example for our simulation.

Realistically, it would be very expensive to have a sensor for on and every fish. Considering the requirement that a tag's identifier be unique, at the very least the identifying part of it would have to be replaced each time if we want to reuse the tags. Also, putting a tag directly on a fish could possibly compromise the quality of the fish. As such, it would make more sense to have a crate or similar in which to store a certain quantity of fish, and then put the tag on the outside to be scanned. The temperature sensor could be inside the crate, as it is the temperature around the fish that matters, rather than that of the surroundings.

All that being said, it makes little difference to our simulation. As this will be a completely digital simulation, the physical specifics have minimal influence on our design.

3.5.2 Structure of the Supply Chain

In our simulation we will be using a simplified view of a supply chain. Let us call the series of steps a product takes from source to destination a path. A realistic supply chain might have several possible paths. This complexity is not the key point to the scalability of the supply chain management system. Even if there are several possible paths, what matters more would be the number of nodes, rather than their relation to each other. As such, we will simplify the shipping part of the supply chain, to a linear path. That is to say, there will be several producers, shippers and

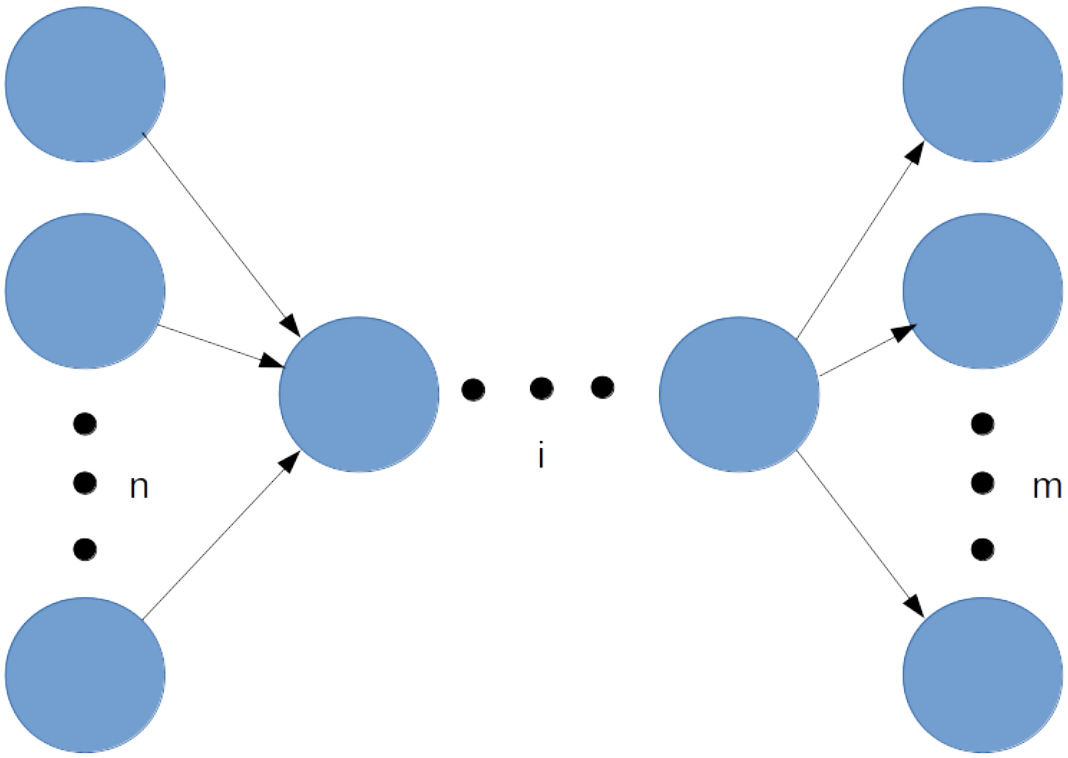

FIGURE 3.1: Configuration of nodes in the supply chain

consumers, but the shippers will be set up such that a product is sent through the same ones, in the same order, every time.

The structure of our supply chain network can hence be described as seen in 3.1. There can be any number n producing nodes, that are independent of each other. There can be i shipping nodes, connected in a series, with a $1to1$ relationship. Finally there can be any number m consumer nodes. After the product has arrived at the last shipping node in the series, it is sent to its corresponding consumer node.

Figure 3.1 shows a visual representation of the node setup. The leftmost column of nodes are the producing nodes, while the rightmost column represent the consuming nodes. Between them is a series of nodes representing the shipper nodes.

When performing tests using the simulation, we will hence vary the total number of nodes, using different configurations of the different kinds of nodes. For the initial test, to set a benchmark, we will use a very simple setup with one of each node, in a $1-1-1$ relation. For the subsequent tests, we will vary the parameters described in table 3.2. The results will be measured in terms of the variables given in table 3.3.

Chapter 4

Main Research

Chapter 4 present the main research of the thesis. We present the process of deciding upon which framework to use, with the arguments and reasoning behind the decisions. We then present the development process, which includes explanations of how to set up the development environment and of how the code works. Finally is an explanation of the testing procedure to follow, and how the data will be presented in Chapter 5.

4.1 Hyperledger

Among the open source projects with potential for use in supply chain management systems, the Hyperledger project appeared to have the most potential. The project is described as "an open source collaborative effort created to advance cross-industry blockchain technologies" [17]. It is hosted by the Linux Foundation, and is completely open source. The Linux Foundation believe open source projects are the only way to grow a sustainable and transparent blockchain system in order to gain mainstream adoption [17]. The collaboration involves industry partners such as Intel and IBM.

Under the Hyperledger umbrella, there are several blockchain technologies, meant for being used in the industry. This is actually an interesting point to keep in mind. During the literature review done prior to this thesis, we discovered that there was a lack of technology being put into use. The blockchain solutions were still at a research level, and nowhere near being used in the industry. With Hyperledger, there appears to be less of an emphasis on research, while they focus more on creating practical solutions for the industry from the get-go.

The project Hyperledger Grid would be a natural option when developing a supply chain management system, as it is made specifically for supply chain use cases [20]. However, when looking into it, it seemed to be very lacking in several aspects. For example, the official documentation is only a very bare-bones summary of what can be found in the code, with very lacking explanations regarding how to put it to use. Combined with a lack of examples available, it would be a challenging job to understand how to use it, and then actually put it to use, in the limited time available.

4.1.1 Sawtooth

After reading about the various solutions offered by the Hyperledger projects, we first decided to look at the Hyperledger Sawtooth. It is a flexible and modular platform for building blockchain solutions [21]. There is also a ready-made supply chain example solution made using Sawtooth available [22].

We then began looking into the Hyperledger Sawtooth Supply Chain. The source code and build instructions are available on Github, which makes getting started easier. However, when we started the build process we encountered problems. Following the instructions lead to an error, with no instructions on how to fix it. After trying a multitude of fixes, we finally came across a post which described the problem and the solution: the current version available on Github does not use the right dependencies, which leads to a build error. The post included the git patches necessary to fix the issue, which made it possible to build the program.

With the setup done, it was possible to run the program. It is an elaborate system, with parts running on five different localhost servers. Two of the localhosts are for the APIs of the system. There is a REST API for the Sawtooth Blockchain, but also one specifically for the Sawtooth Supply Chain. REST stands for representational state transfer and is a style for designing communication for web services [58]. The APIs are to facilitate creating applications that can communicate with the system. Two of the remaining localhosts offer mock websites with interfaces for checking up on products and assets, as well as adding and updating new products. Finally, the last localhost offers an interface showing stats for the server and communication on the blockchain.

The localhost page set up by the application presents the data in an easily accessible format. However, it is more challenging to track the data as it is created, and to monitor the blocks being added to the blockchain. An additional issue is that the repository itself does not contain the actual code used to set up and run the underlying blockchain, it is merely an application using it. As such, locating the information we want and monitoring how the blockchain changes becomes challenging. It could possibly be doable to use the REST API to listen for block updates, but we would prefer a more low-level solution, where we directly monitor the blocks and the ledger as they are updated, rather than getting notifications and deriving the changes from that.

While trying to tackle the challenges related to working with the project, we also discovered a present weakness of the Hyperledger Sawtooth: it is still a young project. The documentation offers little help with creating an application such as the one we want, and there are few examples available. There are also very few examples and tutorials made by unrelated parties, which hinders learning. While there are certain parameters that we could still test to some extent, we would not be able to test how block size and block generation frequency would affect the system. Faced with these problems, we decided to try other options.

4.2 Hyperledger Fabric

Based on the problems we faced with using the Hyperledger Sawtooth, we wanted to look at a more mature project with more popularity. Even though the Hyperledger Grid is supposed to designed for creating supply chain solutions, it is as mentioned above too new and without any resources to go with it. Our decision then fell on the Hyperledger Fabric [19]. The Hyperledger Fabric is a general-purpose blockchain architecture, meant to be used by the industry. The documentation for the project is extensive and features a heap of examples. Additionally, there are several example implementations made by both industry partners and individuals [23][35].

4.2.1 blockchainbean2

Looking at the various options available, we wanted to start with using the project *blockchainbean2* from IBM in cooperation with the Brooklyn Roasting Company [23]. It is a code pattern for creating a fair trade supply chain management solution. The idea is to improve traceability in order to guarantee the provenance of coffee beans used by the Brooklyn Roasting Company. That is to say, they cooperate with farmers to have them put trackers on their shipments of coffee beans. The shipments can then be tracked from point to point until they end up at their final destination. In this way, the Brooklyn Roasting Company can have more confidence in the provenance of their goods, and the farmers can be properly compensated, as there is proof of the origin of what they produce.

We decided upon this project because of the technologies involved and the promises it makes. The project utilizes the Hyperledger Fabric v1.4, Node.js and Loopback 4. Node.js is a run-time environment for JavaScript for server-side code. Node.js is also one of the primary languages or systems supported by the Hyperledger Fabric, which means that there are more resources available for it than for languages with less support, such as Java and Python. Loopback is a Node.js framework that simplifies the process of creating REST APIs. The latter two technologies are not integral to us, but make it easier to develop and deploy the solutions.

However, we immediately encountered an issue when trying to use the *blockchainbean2* project: cloning the repository from Github did not work. Attempts were made both with and without a VPN, but to no avail. We then tried downloading a ZIP file of the project, which did not work either. In the end, we could not figure out why this problem was occurring. The project was deemed not to be worth the effort of trying more and was abandoned.

4.2.2 Tuna App and Blockchain Supply Chain

When looking into more options, we found *Tuna App* made by the people behind the Hyperledger Project itself. It is a full example solution of a simple blockchain-based supply chain, made to showcase how a company can utilize the Hyperledger Fabric in their own supply chain. The specific use case is the tracking of fish from the time it is caught by the fisherman, to when it ends up on a diner's plate in a restaurant, to help the restaurant serve the customer the best fish possible. However, two main issues appeared when trying to use this project: 1) After setting it up, it was still unreliable to use. While the setup ran normally without any errors, the web application included would sometimes not properly connect to the ledger, making it impossible to actually look up fish. 2) it was not clear how one could modify the Fabric settings in order to run the tests we want.

4.2.3 Summary

After trying a variety of different ready-made solutions, to no avail, we finally decided to create our own proprietary solution. However, as there is a significant workload involved when setting up an entire blockchain solution, we based it around a framework that we knew worked. Specifically we utilized the *fabcar* example from Hyperledger's official *Fabric Samples* repository. It is therefore not truly a proprietary solution, but rather an extension of an open source project. Fabcar can be called a bare-bones implementation of an ownership registration network. One can register

a product, in the original implementation they use cars, with certain attributes, including the owner. It is possible to add new products, query the existing ones, and also to change the owner of products.

The code provided by the repository sets up a simple Fabric-based network on a single machine. This network provides a basis from which one can develop an application. The most important part of the network for the purposes of this thesis, is the chaincode, as this is what allows us to carry out the transactions we want to simulate.

To start with, we will develop a version that could be run on a local machine, with all the necessary parts as well as all peer nodes running on a single computer. *Fabcar* is constructed to work like this, so it will consist of modifying and expanding the functionality included in the repository.

After having a working simulation that can be run locally, we will work on getting it to run on a distributed system. A distributed version is in many ways significantly more complex. When running a simulation on a single computer, all the nodes are implicitly connected directly to each other, as they are physically at the same location. In a distributed scenario, there is no guarantee regarding where the various nodes are located in relation to each other. The nodes in the network could theoretically be at opposite sides of the globe.

4.3 Development

In this section we will present the stages of development of the application. We first present the basis of what needs to be developed, highlighting the necessary features to be implemented. We will also present some key parts of the code used in the simulation, and explain how it works, and why it is set up as it is. Additionally, we mention some challenges we encountered while developing the simulation. However, we relegate a more thorough discussion of the challenges to chapter 6. The full source code can be found in Appendix A.

4.3.1 Setting Up Hyperledger Fabric

Getting Hyperledger Fabric to work properly can be a challenging effort. We recommend following the steps outlined on the official website closely [18]. Even when following the steps, it is easy to make mistakes. There are many parts involved, and the versions are changed frequently. Fabric is very specific with regards to the setup it needs to work properly.

One important thing to note is that we are using version 1.4.0 of the Hyperledger Fabric. Specifically, after cloning the Git repository of *Fabric Samples*, we change to the v1.4.0 branch:

```
1 $ git checkout v1.4.0
```

LISTING 4.1: Changing the branch of the repository

This is a version of Fabric that is well-documented and stable. It is important to do this right after downloading the files, before starting any development or even setting up the environment. Furthermore, we have tried using the exact same setup, but with a different release version, and had it not working. It is hence essential to use the same version to get our simulation to run properly.

For the remaining parts required to run Hyperledger Fabric, we are using the following specific versions:

Program	Version
cURL	7.58.0
Docker	18.09.4
Docker Compose	1.23.2
Go	1.11.0
Node.js	8.15.1
npm	6.4.1
Python	2.7.12

4.3.2 Lacking Features

While *fabcar* is a working example, it does not quite fit our needs out of the box. The most challenging change that needs to be made, is to allow for setting up the network for communication between several computers. *Fabcar* is built on an also-included *basic-network* from the Fabric samples. This network is only made for being run on a single host machine. While this does allow for some testing, such as seeing how the ledger grows, it is impossible to test for how network communication affects the performance and scalability.

Some changes also need to be made to the chaincode. *Chaincode* is a term used by Hyperledger, and is in principle a smart contract. It essentially provides an API for peers in the network to carry out transactions, query the ledger, and in other ways interact with the blockchain. For example, if a peer wants to register a new product on the blockchain, it must utilize a chaincode function, say $createProduct(productID, x, y, z)$, which will carry out the necessary steps to create the transaction for the blockchain.

The included chaincode allows for changing the owner of a product, but one must specify to whom the ownership will be transferred. We want a more streamlined function, where one can simply provide the identification number of a product, and have it be sent to its next stop. This way, we avoid having to hard-code the transfers for each participant. If we were to use the included *changeOwner* method, the next owner of a product would have to be specified in the "main" program running on each node, i.e. the core of the simulation. Instead, the logic of the transfer will be defined in the chaincode, following the logic of $Producer- > Shipper- > Consumer$.

Furthermore, we want to be able to detect whether a product has reached its destination and register the fact. As the original example was made only to keep track of the specifications and owner of a car, there was no need to keep track of a destination. Data fields for the source, destination and whether it has been delivered can simply be added to the data structure provided. Some additional logic must also be added to the chaincode in order to identify when the destination has been reached.

Using Hyperledger Fabric and the example of shipping fish, the data fields for the product look like this:

```
const fish = {
    docType: 'fish',
    species,
    size,
    owner,
   source: owner,
   destination,
   coordinates,
   arrived: '0',
};
```

The field *docType* is to identify the kind of product that is being stored. It is slightly redundant in our case, as we only look at a single kind of product, but makes little difference. *Species* and *size* would be important to a consumer, as they could compare what they receive to what has been registered on the blockchain. In case of fraud, theft or similar tampering, one could cross-check the physical shipment with the registered data. *Owner*, *source* and *destination* are names of parties involved, such as fishermen and shipping companies. The *coordinates* field will be updated with some frequency, and represents sensor data in general. Any number of additional fields could be added, but the ones listed here are enough for the purposes of our simulation.

Arrived will always be instantiated as 0, meaning "not arrived". Since a Producer node will never also be a Consumer node, it is simply impossible for a newly created product to already have arrived, and it is hence unnecessary to require the *arrived* field to be specified when registering it.

4.3.3 Peer Nodes

The peers in our simplified supply chain management system can be put into three categories, based on their role. This further defines what actions they need to be able to carry through, which will influence the design of the code run by each node. The categories are as follows:

Producer	Nodes that generate new products and register them in the supply chain, before sending them off to a *Shipper* node
Shipper	Nodes that receive products and send them to their next point along the supply chain, ending in a *Consumer* node
Consumer	Nodes that receive products from *Shipper* nodes, and register them as delivered. They do not send products any further

Producer Nodes

Producer nodes are in our case fishermen or otherwise fish providing entities. These nodes will create new fish that are registered with their data on the blockchain. To have some variety in the data, the data fields will be somewhat randomized. Changing the number of products generated will be one the parameters of our simulation. After some time, the products will be sent to the next node, which will be a Shipper node.

The code run by producer nodes is as follows:

```
1  // n is the number of products to be generated
2  for (var i = 0; i < n; i++) {
3     // Generate data for new product
4     const destination = consumers[Math.floor(Math.random()*consumers.length)];
5     const weight = (Math.random() * 5 + 2).toFixed(2); // Generates a
          pseudorandom number [2, 7)
6
7     const x = (Math.random() * 360 - 180).toFixed(2); // Pseudorandom number
          [-180, 180)
8     const y = (Math.random() * 360 - 180).toFixed(2);
9     const coor = `(${x}, ${y})`; // Format the coordinates
10
```

```
11    const spec = species[Math.floor(Math.random()*species.length)]; // Get
          the name of the species of fish from a list
12
13     // Call chaincode function with data to generate new fish
14    await contract.submitTransaction('createFish', 'FISH' + nodeNumber + i,
          spec, `${weight}`, owner, destination, coor);
15
16    // Update list of products owned by this peer
17    products = (await contract.evaluateTransaction('queryOwned',
          owner)).toString();
18    fishNumbers = products.match(/FISH[0-9]*/g || []);
19    if (null == fishNumbers) {
20        continue;
21    }
22    for (var j = 0; j < fishNumbers.length; j++) {
23         // Get next product in list
24         const product = fishNumbers[j];
25
26         // Update location of product
27        const x = (Math.random() * 360 - 180).toFixed(2);
28        const y = (Math.random() * 360 - 180).toFixed(2);
29        const coor = `(${x}, ${y})`;
30        await contract.submitTransaction('updateCoordinates', product, coor);
31        elapsedTime = parseHrtimeToSeconds(process.hrtime(startTime));
32        console.log("update " + user + " " + elapsedTime);
33
34        // Move product to next node
35        startTime = process.hrtime();
36        await contract.submitTransaction('sendToNext', product);
37        elapsedTime = parseHrtimeToSeconds(process.hrtime(startTime));
38        console.log("send " + user + " " + elapsedTime);
39    }
40 }
```

LISTING 4.2: Code for Producer nodes

Line 19 makes a check if the list of products currently owned by the node running the code is empty. In the case that the list is empty, the rest of the code is meaningless, and can be skipped for the current iteration.

It is worth noting the numbering system we are utilizing for fish identification. The structure is defined as $nodeNumber + i$, where *nodeNumber* is the number identifying the node. This is in order to avoid two nodes creating a fish with the same number. Hence, the first fish created by node 1 would be "FISH10", while node 2 would create "FISH20". Since we always append a fixed number unique to each node at the beginning of identifying number, there will never be an overlap.

Shipper Nodes

Shipper nodes are, as the name implies, the intermediary shipping companies whose job is to take a product from point A to B. There can be several shipping companies involved in the process. For simplicity, we are considering a case where each product takes a linear path from source to destination. By linear, we mean that they are all shipped by the same shipping companies, in the same order every time.

```
1  while (true) {
2    // Update list of products owned by this peer
```

```
3     const products = (await contract.evaluateTransaction('queryOwned', owner,
          false)).toString();
4     const fishNumbers = products.match(/FISH[0-9]*/g || []);
5      if (null == fishNumbers) {
6          continue;
7     }
8     var product = null;
9     for (var j = 0; j < fishNumbers.length; j++) {
10         // Get next product in list
11         startTime = process.hrtime();
12         product = fishNumbers[j];
13
14         // Update location of product
15         const x = (Math.random() * 360 - 180).toFixed(2);
16         const y = (Math.random() * 360 - 180).toFixed(2);
17         const coor = `(${x}, ${y})`;
18         await contract.submitTransaction('updateCoordinates', product, coor);
19         elapsedTime = parseHrtimeToSeconds(process.hrtime(startTime));
20         console.log("update " + user + " " + elapsedTime);
21
22         // Move product to next node
23         startTime = process.hrtime();
24         await contract.submitTransaction('sendToNext', product);
25         elapsedTime = parseHrtimeToSeconds(process.hrtime(startTime));
26         console.log("send " + user + " " + elapsedTime);
27
28    }
29 }
```

Lines three to seven query the blockchain for the products currently owned by the calling node. If there are none, the rest of the execution is skipped, and the list is updated again until there are products in the list.

The next part of the code is applied to every product in the list of owned products. First a set of coordinates is generated. The coordinates are randomly generated, as this simulation does not use location data in a meaningful way. The coordinates of the product are then updated through a transaction. The product is then sent to its next destination. The time needed for both these operations is measured and logged.

Consumer Nodes

Consumer nodes are the end customers. End customers can be food stores, restaurants, cafes and so on. In our simulation, Consumer nodes receive products and register them as delivered. They do not generate new products or send any further along, they are end nodes.

```
1  while (true) {
2     // Update list of products owned by this peer
3     const products = (await contract.evaluateTransaction('queryOwned', owner,
          '0')).toString();
4     const fishNumbers = products.match(/FISH[0-9]*/g || []);
5     if (null == fishNumbers) {
6         continue;
7     }
8     var product = null;
9     for (var j = 0; j < fishNumbers.length; j++) {
10        // Get next product in list
```

```
        startTime = process.hrtime();
        product = fishNumbers[j];

        // Register fish as having arrived
        await contract.submitTransaction('hasArrived', product, owner);
        elapsedTime = parseHrtimeToSeconds(process.hrtime(startTime));
        console.log("arrived " + user + " " + elapsedTime);
    }
}
```

Consumer nodes will first update their list of products that they own. If the list is empty, they will simply continue to the next iteration and check again, until there are products to be found. For each product, the Consumer nodes will then register in the blockchain that it has arrived at its final destination.

4.3.4 Chaincode

The chaincode is in many ways the most important part of a network based on Hyperledger Fabric. In order for the peers in the network to interact with the blockchain, they have to go through the chaincode. In many ways, it functions as an API for blockchain communication.

Creating Fish

The basics of the *createFish()* method are the same as in the original *Fabcar* version. However, there are a couple of key differences:

```
async createFish(ctx, fishNumber, species, size, owner, destination,
    coordinates) {
    console.info('============= START : Create Fish ===========');
    const fish = {
        docType: 'fish',
        species,
        size,
        owner,
       source: owner,
       destination,
       coordinates,
       arrived: '0',
    };

    // Check if the owner is a real producer, and that the destination is a
        real consumer
   if (!hasElement(data.producers, fish.source) ||
       !hasElement(data.consumers, fish.destination)) {
       console.log('Incorrect source or destination');
       return -1;
   }

   // Register fish in blockchain
    await ctx.stub.putState(fishNumber, Buffer.from(JSON.stringify(fish)));
    console.info('============= END : Create Fish ===========');
}
```

LISTING 4.3: The method *createFish()*

The method takes a series of arguments. The arguments are the first difference, as these naturally had to be exchanged for the necessary data points for tracking fish. The second difference is the part starting from the comment on line 14 until line 18. The purpose of this part is to verify that the source and destination are valid. This verification saves up some effort later on, where these fields can be assumed to be correct. That being said, as this is a simulation where we can control every aspect of the code and nodes, it is not strictly necessary, as we can always avoid trying to use incorrect values. Even so, it serves as a fail-safe in case we make a typo when setting up the simulation.

Changing Owner

A completely essential part of a supply chain management system, is the ability to track the owner of a given product. The basic *Fabcar* code provides both a data field for listing the owner, as well as a method for changing it. The original method *changeCarOwner* is defined as follows:

```
1 async changeCarOwner(ctx, carNumber, newOwner)
```

Providing the ID number of the car, *carNumber*, and the name of the new owner, the code queries the blockchain for the entry of the corresponding car, changes the owner, and submits the new data. While it would be possible to use this method as is, it would require hard-coding each peer for who the recipient should be. This is highly unmodular and would make it cumbersome to expand the simulation for different numbers of peer nodes. With this in mind, we created a new method, *sendToNext()*. While the new method is hard-coded to only allow for the three different kinds of nodes we have defined, it is completely modular for our purposes. The code is as follows:

```
1  async sendToNext(ctx, fishNumber) {
2      console.info('============= START : sendToNext ===========');
3
4     const fishAsBytes = await ctx.stub.getState(fishNumber); // get the fish
          from chaincode state
5      if (!fishAsBytes || fishAsBytes.length === 0) {
6          throw new Error(`${fishNumber} does not exist`);
7      }
8      const fish = JSON.parse(fishAsBytes.toString());
9
10    if (hasElement(producers, fish.owner)) {
11        fish.owner = shippers[0];
12    }
13    else if (hasElement(shippers, fish.owner)) {
14        if (shippers[shippers.length-1] === fish.owner) {
15            fish.owner = fish.destination;
16        }
17        else {
18          fish.owner = shippers[shippers.indexOf(fish.owner)+1];
19        }
20    }
21    else {
22        console.info("Fish already at destination");
23        return -1;
24    }
25
```

```
26      // Submit transaction
27      await ctx.stub.putState(fishNumber, Buffer.from(JSON.stringify(fish)));
28      console.info('============= END : sendToNext ===========');
29  }
```

LISTING 4.4: The method *sendToNext()*

The first seven lines are taken directly from the original *Fabcar* code, only modified to use fish rather than cars. An attempt is made to fetch the bytes representing the given fish from the ledger. If the fish turns out not to exist, execution is stopped. If the fish does exist, the bytes are parsed to a JSON format. Lines nine to 23 perform a check of where the fish is currently located, before changing the ownership to the next owner in line. As the supply chains utilized in these tests are linear, the logic is quite simple. For the last step of moving from the last Shipper node to the Consumer node, the new owner is simply set to the destination. The destination is verified as a legitimate destination when the fish is first registered, so it is unnecessary to perform a look-up again.

Update Location

Fabcar has no other methods for changing the various data fields of the registered products. It was thus necessary to create a method for updating the location, as providing continuous knowledge about the state of shipment is considered a central benefit to utilizing blockchain in supply chain management systems. We defined the method as follows:

```
1   async updateCoordinates(ctx, fishNumber, coordinates) {
2      console.info('============= START : updateCoordinates ===========');
3
4       const fishAsBytes = await ctx.stub.getState(fishNumber); // get the fish
            from chaincode state
5       if (!fishAsBytes || fishAsBytes.length === 0) {
6           throw new Error(`${fishNumber} does not exist`);
7       }
8       const fish = JSON.parse(fishAsBytes.toString());
9      fish.coordinates = coordinates;
10
11      await ctx.stub.putState(fishNumber, Buffer.from(JSON.stringify(fish)));
12      console.info('============= END : updateCoordinates ===========');
13  }
```

LISTING 4.5: The method *updateLocation()*

In our simulation, we simply use a text string to represent coordinates. In a more realistic scenario, one would use real geographical data, through an API provided by a service such as Google Maps.

Querying Owned Products

Fabcar originally provides the methods *queryCar()* and *queryAll()*. These can be used as-is, with the only change being that all references to "car" be changed to "fish". However, we also want the additional option of querying for products owned by the calling node only. We achieve this through the implementation of the method *queryOwned()*, defined as follows:

```
async queryOwned(ctx, owner, includeArrived) {
    const startKey = 'FISH0';
    const endKey = 'FISH99999999';

    const iterator = await ctx.stub.getStateByRange(startKey, endKey);

    const allResults = [];
    while (true) {
        const res = await iterator.next();

        if (res.value && res.value.value.toString()) {
            console.log(res.value.value.toString('utf8'));

            const Key = res.value.key;
            let Record;
            try {
                Record = JSON.parse(res.value.value.toString('utf8'));
            } catch (err) {
                console.log(err);
                Record = res.value.value.toString('utf8');
            }
   if (owner === Record.owner){
       // If we include already arrived products, then we include all
           products
       if (includeArrived) {
      allResults.push(Key);
       }
       // If not, we only want non-arrived products
       else if ('0' === Record.arrived) {
      allResults.push(Key);
       }
   }
        }
        if (res.done) {
            console.log('end of data');
            await iterator.close();
            console.info(allResults);
            return JSON.stringify(allResults);
        }
    }
}
```

LISTING 4.6: The method *queryOwned()*

The constants *startKey* and *endKey* refer to the identification numbers of the products. We have set *endKey* to be 99999999 because of how we determine the numbers, but it can be any sufficiently high number.

This method also serves as a more realistic implementation. In a real-world scenario, it could very well be that each party would only be allowed detailed information on products in their possession.

Registering Arrival

The most important task of the Consumer nodes, is to register a product as having arrived. For this purpose, we have the method *hasArrived*():

```
async hasArrived(ctx, fishNumber, owner) {
   console.info('============= START : hasArrived ===========');

    const fishAsBytes = await ctx.stub.getState(fishNumber); // get the fish
        from chaincode state
    if (!fishAsBytes || fishAsBytes.length === 0) {
        throw new Error(`${fishNumber} does not exist`);
    }
    const fish = JSON.parse(fishAsBytes.toString());
    fish.owner = newOwner;

   if (fish.owner === owner && fish.destination === owner) {
       fish.arrived = 1;
   }
   else {
       console.log("You are not authorized to do this");
   }

    await ctx.stub.putState(fishNumber, Buffer.from(JSON.stringify(fish)));
    console.info('============= END : hasArrived ===========');
}
```

4.3.5 Blocks

Finally there is the matter of the actual blocks being stored in the blockchain. These have a direct correlation with the storage space needed, and are as such of interest to us. This is the only feature of the blocks that we will use as a parameter in our simulation.

In the *basic-network* folder, which sets up the network used by *Fabcar*, there is a file called *configtx.yaml*. This is apparently the file in which some of the specifications of the blocks are defined. The following is the relevant excerpt of the code:

```
# Batch Size: Controls the number of messages batched into a block
    BatchSize:

        # Max Message Count: The maximum number of messages to permit in a
            batch
        MaxMessageCount: 10

        # Absolute Max Bytes: The absolute maximum number of bytes allowed
            for
        # the serialized messages in a batch.
        AbsoluteMaxBytes: 99 MB

        # Preferred Max Bytes: The preferred maximum number of bytes allowed
            for
        # the serialized messages in a batch. A message larger than the
            preferred
        # max bytes will result in a batch larger than preferred max bytes.
        PreferredMaxBytes: 512 KB
```

LISTING 4.7: Code snippet from *configtx.yaml*

Here we can see the fields *AbsoluteMaxBytes* and *PreferredMaxBytes*. We will be editing the latter field, *PreferredMaxBytes*, for the purposes of our tests.

4.4 Testing

In this section we describe the tests we performed using the simulation we developed, described in the section above. We first present the technical details regarding the testing, including hardware and the testing environment. We then move on to describing how the tests were carried out, and with which parameters.

As we were unable to finalize the version of the simulation meant for a distributed network, we were left with the working version running on a single computer.

4.4.1 Technical Details

Instead of the servers described in chapter 3, the tests were carried out on a laptop, with the following specifications:

Model	HP Pavilion 15 Notebook PC
Processor	Intel i5-4210U
Clock speed	1.7GHz
Cache size	3MB
Memory	8GB

This is unfortunately a somewhat dated laptop model, but the computational load of the simulation is not expected to be highly significant. Even so, it is worth keeping in mind, and will serve as a point of discussion when analyzing the results.

As we will be running the tests on a single machine, the concepts of peers and nodes get somewhat watered down. Simply by the nature of the hardware, all parts of the system will be directly connected to each other.

We will run a series of test, starting with a very low-complexity proof of concept, before gradually increasing the complexity. This is in order to check for any errors that might occur. By starting with a simpler setup, we can more quickly and easily check for errors. The functionality is the same regardless of complexity, so it will be possible to make assumptions based on our experiences. Furthermore, by gradually increasing the complexity, if something goes wrong, we can know what caused the problem by looking at which parameter was changed. For example, if the initial test with a single producer node goes well, but we experience trouble when introducing another, we can deduce that the inclusion of another node caused the issue.

4.4.2 First Tests

The first few tests will be run with one of each type of node, in order to check that the different parts of the network communicate well together. In other words, it will be a very minimal supply chain, a completely linear directed graph with three nodes.

For the very first test, we will register only ten products, again to check for functionality. We will be using the default preferred maximum block size. Hence the parameters will be as follows:

Setting Up The Test Environment

The testing environment is based from the Ubuntu terminal. The first step is to replace the chaincode located in $fabric-samples/chaincode/fabcar/javascript/lib/$. This is simply a matter of replacing the file $fabcar.js$, which is the only file in the directory.

We then move to the directory $fabric-samples/fabcar/$. Here we run the following commands:

Total number of nodes	3
Producer nodes	1
Shipper nodes	1
Consumer nodes	1
Number of products	10
Block size	512KB

TABLE 4.1: Parameters for the first test

LISTING 4.8: Setting up the testing enviroment

```
$ ./startFabric javascript
$ cd javascript
$ npm install
$ node enrollAdmin.js
```

These steps are the same as you would run in the original *Fabcar* example. The first command starts Hyperledger Fabric running in a set of Docker containers, set up to use JavaScript. The second moves to the sub-directory where the code for the JavaScript version is located. Next we install the application, before finally enrolling an administrator for the network.

At this point our application differs from the source code. We modified the code from the source file *registerUser.js*, which registers a single user called "user1", to allow for registering several users, with the same naming convention, "userX", *X* being an increasing integer. After registering the number of users that we want, we can move on to carrying out the tests.

Running The Test

This initial test requires a very minimal setup, so it will be run in the form of three terminal windows. In each of the terminals, we will enter the command to start running the program, and logging the results, in the following manner:

```
$ node peer-node.js A 1 > userlog1.txt
```

LISTING 4.9: Command to start running the code for a peer node

The first element is "node", which is the command used to run a Node.js program from a Linux terminal. The second element is the source file, here the code to be used by a peer node. Next is the name of the user, or peer. This can be anything, for example "Fishy Shipping" or "The Fish Emporium", but for simplicity, we stick to single letter names. This name is used as an identifier, as seen in the code snippets above. The number 1 refers to the name of the node. In our code we have called all the nodes "userX", where X is a number. The last part, $> userlog1.txt$, is a Linux convention, where one can direct the console output of a program to a file. In our source code, we print information regarding the time taken to execute parts of the code, which we here store in the file $userlog1.txt$. This is to gather the data for analysis.

To track the size of the ledger, we open a terminal connected to the Docker container running a peer, using the following command:

```
$ docker exec -it peer0.org1.example.com bash
```

LISTING 4.10: Command to connect to Docker container running a peer node

In this terminal, we can navigate to the directory where the peer stores its local version of the ledger, and we can determine the size of the ledger at any given time:

```
$ cd /var/hyperledger/production/ledgersData/
$ du -h
28K   ./ledgerProvider
20K   ./chains/index
24K   ./chains/chains/mychannel
28K   ./chains/chains
52K   ./chains
16K   ./configHistory
20K   ./historyLeveldb
16K   ./bookkeeper
20K   ./pvtdataStore
156K  .
```

LISTING 4.11: Commands to determine the size of the ledger

We can here see a breakdown of the size of the ledger. At first, with three registered users and zero fish, the ledger takes up a total of 156KB. We can then, at the end of execution, look at the ledger size again, and see how much it has grown.

It would be possible to continuously register the ledger size, but we do not see this as necessary. By performing tests where we change different parameters one at a time, it is fully possible to deduce how much a given parameter affects the growth. For example, if we extend the length of the supply chain, we know that there will be more transactions in the form of changing ownership. We can even calculate how many more transactions there will be, and then use this number along with how much larger the ledger grew to figure out the cost of each transaction.

We are bound to come across a variety of errors and problems as we attempt to carry out this stage of the research. These will be described in chapter 6, under the section about the challenges we encountered during the process of writing this master thesis.

Next Test

After confirming that the first test works, we will start exploring the various parameters. To begin with, we will increase the number of products being produced. We will do this incrementally, by factors of ten. The reason why we start with the number of products, is that it is the easiest parameter to change. It is a single number in the peer node code that has to be changed, rather than the changes to the whole architecture required for the other parameters.

Due to limited time, we can only test for a certain number of values. Specifically, we will be using the following approximate values:

n = 10

n = 100

n = 1,000

n = 5,000

This will give us a benchmark for what to expect when expanding the supply chain, and changing the size of the blocks later on. We will distribute the task of producing the products equally between the Producer nodes. When it is not possible

to use the exact number listed above, we will round up to the closest divisor. For example, with three nodes and ten products, each will produce four fish, for a total of twelve. With 100 products and eight nodes, each will register 13 fish, for a total of 104.

4.4.3 Subsequent Tests

At this point, we will have run a series of tests exploring one parameter. We will then start exploring the rest of the parameters we outlined in chapter 3, starting with the number of nodes. This will also involved different configurations of the nodes, according to the structure described at the end of chapter 3.

To more easily launch the simulation with higher numbers of nodes, we made the following script:

```
1  #!/bin/bash
2
3  users=("A" "B" "C" "D" "E" "F" "G" "H" "I" "J" "K" "L" "M" "N" "O" "P" "Q"
       "R" "S" "T" "U" "V" "W" "X" "Y" "Z")
4
5  read -p "How many users? " input
6
7  for i in $(seq 1 ${input}); do
8      temp="user"$i
9      echo $temp ${users[i-1]}
10     node peer-node.js ${users[i-1]} $i > userlog${i}.txt &
11 done
```

LISTING 4.12: Script to set up the execution of up to 26 nodes

The list *users* defines the usernames for nodes. For testing purposes, these are limited to the letters of the English alphabet, and have a direct relation to the node number.

Line five presents a prompt for user input. There is no input validation, but it is assumed to be an integer defining how many users will be included in the simulation run.

Line seven starts a for loop, running from one to the number received from line five. The next lines print to the terminal information on which users and nodes have been included, and start the corresponding peer nodes. The peer node is also set to write its console output to a log file. The ampersand at the end of the command is a Linux terminal option, which sets the preceding command to be run in the background, allowing for more commands to be run without waiting for the former to finish.

Number of Nodes

As described before, we are aiming to use values in the tens for the nodes. We have originally started with three, and will again work our way up gradually. We will also vary the configuration, that is to say how many of each three types of nodes we will have.

These configurations were chosen in order to test the performance in a variety of use cases. In some real-world scenarios, there might be more producers than consumers, while in others it could be the other way around. It is therefore interesting to look at several options.

5	10	15	20
3 - 1 - 1	6 - 2 - 2	8 - 4 - 3	12 - 4 - 4
1 - 3 - 1	2 - 6 - 2	4 - 8 - 3	4 - 12 - 4
1 - 1 - 3	2 - 2 - 6	4 - 3 - 8	4 - 4 - 12

TABLE 4.2: Number of nodes, and configurations for which we will test

Size of Blocks

As mentioned above, we will be modifying the parameter *PreferredMaxBytes* for the blocks. This value is originally set to 512KB, while the maximum size is at 99MB. We will be using values within this range for our tests.

size = 1MB

size = 20MB

size = 50MB

size = 99MB

Limitations

Trying out every value of each parameter with each other would result in a very large amount of tests, $5 * 12 * 4 = 240$. From trial runs while developing, we have found that a transaction will take about 2.3 seconds to perform. With thousands of transactions, carrying out over 200 tests would take a very large amount of time. To make this number more manageable with the limited amount of time at disposal, we will only be using certain values from the previous testing block. That is to say, we divide the testing into three sections:

- Number of products
- Number of nodes
- Block size

For each section, we will test every value. However, for the next, we will only be using two of the values from the previous section. This means that we will only perform $(5) + (2 * 12) + (2 * 2 * 5) = 49$ tests, which is still a fairly large number. These numbers will be:

Products	Nodes
100	20
1000	

TABLE 4.3: Test values for subsequent tests

4.4.4 Data Presentation

As we will be generating a sizeable amount of data, it is important to present it in a meaningful and consistent way.

Time Spent

To measure the time spent on various events, we use the built-in timing function in JavaScript, *process.hrtime()*. A lot of tests will be run, so rather than presenting an individual graph for the measurements of each, we will present the average.

To calculate the average time spent on each task, we will use the following script:

```
#!/bin/bash

count=0;
total=0;

for i in $( grep "produce" userlog* | awk '{ print $3 }' )
do
     total=$(echo $total+$i | bc )
     ((count++))
   done
echo Produce:
echo "scale=2; $total / $count" | bc

count=0;
total=0;

for i in $( grep "send" userlog* | awk '{ print $3 }' )
do
     total=$(echo $total+$i | bc )
     ((count++))
   done
echo Send:
echo "scale=2; $total / $count" | bc

count=0;
total=0;

for i in $( grep "update" userlog* | awk '{ print $3 }' )
do
     total=$(echo $total+$i | bc )
     ((count++))
   done
echo Update:
echo "scale=2; $total / $count" | bc
```

LISTING 4.13: Script to calculate average time for adding new products, sending a product to the next node, and updating location data

Each node logs all the blockchain operations they carry out. These actions are registering new products (produce), updating a data field (update), and sending a product to the next node (send). The script above calculates the average of each of these actions, across all

Ledger Size

For the ledger size, we will be using the storage space required at the end of the simulation. We will not be using any intermediary values from the course of the execution, as these can be highly variable, depending on variable beyond our control.

We will also be including the original size of the ledger for all values. The initial state of the ledger takes up $156KB$ of disk space.

After collecting all the data, we will contextualize it by comparing the required space to the different parameters, such as number of nodes and number of products. This will also make it easier to showcase any discoveries we make.

Chapter 5

Results

In this chapter we will present the results of our research. In the first section, we present the raw data, in form of tables with the numbers we got. Next, we will contextualize the data in a series of graphs, in order to more easily understand and analyze the findings.

5.1 Raw Data

In this first section, we will present the raw data gathered. We collected the data in a series of tables. As we split the data collection into three stages, we will present each in turn, and explain how to read the table.

5.1.1 First Tests

The first tests carried out were designed more or less as a proof of concept, in order to verify that everything was working correctly. It was carried out with the parameters found in table 4.1 from chapter 4. The data were as follows:

Products	Produce (s)	Send (s)	Update (s)	Total (s)	Size (KB)
10	2.34	2.35	2.34	75	448
100	2.39	2.39	2.39	724	2976
1,000	2.42	2.44	2.31	7314	28192
5,000	2.54	2.55	1.83	42335	138716

TABLE 5.1: The results of running the simulation with three nodes, with a $1-1-1$ distribution

5.1.2 Testing Number of Nodes

We then started to increase the number of nodes in the system. By increasing the number of nodes, the number of possible configurations also increased. For each number, we limited ourselves to three different configurations. We will present the data in order of increasing number of nodes.

Five Nodes

For the case of five nodes, we used the configurations $3-1-1$, $1-3-1$ and $1-1-3$. This lead to the following data:

Products	**Produce (s)**	**Send (s)**	**Update (s)**	**Total (s)**	**Size (KB)**
10	2.32	2.35	2.30	73	480
100	2.35	2.36	2.30	494	2812
1,000	2.45	2.36	3.31	2456	26128
5,000	2.52	2.59	1.94	14523	129568

TABLE 5.2: The results of running the simulation with five nodes, with a $3-1-1$ distribution

Products	**Produce (s)**	**Send (s)**	**Update (s)**	**Total (s)**	**Size (KB)**
10	2.37	2.35	2.36	76	604
100	2.40	2.41	2.41	723	4424
1,000	2.47	2.50	2.32	7509	42660
5,000	2.54	2.63	1.94	42612	211012

TABLE 5.3: The results of running the simulation with five nodes, with a $1-3-1$ distribution

Products	**Produce (s)**	**Send (s)**	**Update (s)**	**Total (s)**	**Size (KB)**
10	2.47	2.45	2.44	74	444
100	2.43	2.45	2.41	734	2972
1,000	2.56	2.57	2.30	7715	28184
5,000	2.79	2.94	2.31	48264	141948

TABLE 5.4: The results of running the simulation with five nodes, with a $1-1-3$ distribution

Products	**Produce (s)**	**Send (s)**	**Update (s)**	**Total (s)**	**Size (KB)**
10	4.17	2.47	2.40	76	568
100	2.46	2.51	2.39	560	3556
1,000	2.51	2.55	2.15	4661	33368
5,000	2.65	2.92	2.66	29672	164412

TABLE 5.5: The results of running the simulation with ten nodes, with a $6-2-2$ distribution

Products	**Produce (s)**	**Send (s)**	**Update (s)**	**Total (s)**	**Size (KB)**
10	2.64	2.55	2.41	117	808
100	2.53	2.51	2.35	657	6396
1,000	2.49	2.54	1.94	5073	61664
5,000	2.66	2.74	2.46	30783	309196

TABLE 5.6: The results of running the simulation with ten nodes, with a $2-6-2$ distribution

Products	**Produce (s)**	**Send (s)**	**Update (s)**	**Total (s)**	**Size (KB)**
10	2.49	2.45	2.53	84	500
100	2.49	2.48	2.40	537	3488
1,000	2.68	2.69	2.29	5457	33328
5,000	3.10	3.20	3.16	33450	164428

TABLE 5.7: The results of running the simulation with ten nodes, with a $2-2-6$ distribution

Ten Nodes

Fifteen Nodes

For the very first test with 15 nodes, we produce a total of 16 products, rather than ten. This is because of how our tests are designed, with each Producer node producing the same number. As such, the total time spent and the size of the ledger are not quite comparable to the other tests. However, for the following tests, we reach the exact numbers listed.

Products	**Produce (s)**	**Send (s)**	**Update (s)**	**Total (s)**	**Size (KB)**
10	2.41	2.25	2.31	107	936
100	1.11	2.04	2.08	473	5152
1,000	1.00	2.11	2.13	4994	47972
5,000	2.00	2.93	2.85	30493	238352

TABLE 5.8: The results of running the simulation with 15 nodes, with a $8 - 4 - 3$ distribution

Products	**Produce (s)**	**Send (s)**	**Update (s)**	**Total (s)**	**Size (KB)**
10	2.65	2.55	2.36	116	1112
100	1.18	1.96	1.93	501	7892
1,000	0.8	1.81	1.79	4107	76584
5,000	1.70	2.51	2.54	29892	381124

TABLE 5.9: The results of running the simulation with 15 nodes, with a $4 - 8 - 3$ distribution

Products	**Produce (s)**	**Send (s)**	**Update (s)**	**Total (s)**	**Size (KB)**
10	2.93	2.60	2.45	92	660
100	2.53	2.55	2.40	607	4216
1,000	2.73	2.92	2.25	5273	40556
5,000	3.7	4.52	4.59	49779	201140

TABLE 5.10: The results of running the simulation with 15 nodes, with a $4 - 3 - 8$ distribution

Twenty Nodes

When conducting the tests with twelve Producer nodes, we realized a minor flaw in our peer node code; by only prepending the number of the node to the fish identification numbers, we could not avoid creating two fish with the same number. For example, the first fish generated by node twelve would be "FISH121". This has the exact same identification as the 21st fish created by node one, "FISH1" plus "21". However, this was easily fixed by adding three additional zeros after the node number. Hence, fish number one from node twelve would be "FISH120001", and the 21st from node one becomes "FISH100021".

Products	Produce (s)	Send (s)	Update (s)	Total (s)	Size (KB)
10	1.95	2.21	2.19	94	748
100	1.11	2.10	2.02	634	5368
1,000	0.99	2.36	2.07	5432	48688
5,000	5.72	4.35	4.14	38685	240932

TABLE 5.11: The results of running the simulation with 20 nodes, with a $12 - 4 - 4$ distribution

Products	Produce (s)	Send (s)	Update (s)	Total (s)	Size (KB)
10	2.69	2.18	2.03	142	1488
100	1.07	1.08	1.15	399	10960
1,000	0.91	1.28	1.20	3029	106588
5,000	4.08	6.11	6.06	66524	546712

TABLE 5.12: The results of running the simulation with 20 nodes, with a $4 - 12 - 4$ distribution

Products	Produce (s)	Send (s)	Update (s)	Total (s)	Size (KB)
10	2.55	2.54	2.45	75	752
100	2.63	2.63	2.38	627	4956
1,000	2.88	3.06	2.72	6502	48120
5,000	5.09	6.55	6.69	71806	237184

TABLE 5.13: The results of running the simulation with 20 nodes, with a $4 - 4 - 12$ distribution

5.1.3 Testing Block Size

After having tested for a variety of different node configurations, we wanted to test for different block sizes. For the first test, we used 100 products, and the $12 - 4 - 4$ configuration of 20 nodes. The results can be seen in table 5.14.

Size	Produce (s)	Send (s)	Update (s)	Total (s)	Size (KB)
1MB	0.96	2.14	2.07	547	5364
20MB	0.90	2.13	2.07	557	5372
50MB	0.89	2.14	2.1	563	5372
99MB	0.89	2.14	2.07	555	5372

TABLE 5.14: The results of running the simulation with 20 nodes, with a $4 - 4 - 12$ distribution, and with 100 products

As we can see in table 5.14, there are no real differences to the data between the various simulation runs. There are negligible differences in execution time, and only the first run with 1MB preferred block size has a different size. However, the size difference is so small, that it can simply come from random chance.

We then performed tests for the same block sizes, using the same configuration, but with 1,000 products. The results can be seen in table 5.15.

Similar to the previous results, the differences between the different simulation runs are negligible.

Size	**Produce (s)**	**Send (s)**	**Update (s)**	**Total (s)**	**Size (KB)**
1MB	0.90	2.33	2.04	5142	48712
20MB	0.98	2.34	2.10	5241	48700
50MB	1.02	2.40	2.12	5302	48692
99MB	0.94	2.32	2.09	5186	48708

TABLE 5.15: The results of running the simulation with 20 nodes, with a $4 - 4 - 12$ distribution, and with 1000 products

5.1.4 Average Values

The following tables contain the calculated average values for the data presented in the tables above.

	10	**100**	**1000**	**5000**	**Total**
Produce	2.647	2.052	2.068	3.161	2.480
Send	2.413	2.267	2.408	3.658	2.665
Update	2.355	2.201	2.214	3.448	2.523
All	2.472	2.173	2.230	3.422	2.557

TABLE 5.16: Average time taken for the various operations

10	**100**	**1000**	**5000**	**Total**
93	538	5345	34520	10124

TABLE 5.17: Average time taken for complete execution of the simulations

Number of products:	10	100	1000	5000
Size of ledger (KB):	734.5	5014.5	47848.7	238824.9

TABLE 5.18: Average size of the ledger

5.2 Figures

Here we will present various figures, created from the data presented in the previous section. By extracting parts of the data sets and presenting them visually, it becomes easier to see how the data relate to each other. A visual presentation also facilitates the following discussion in chapter 6.

5.2.1 Ledger Size

We will here present figures showing the size of the ledger, and how it is affected by different parameters. Figure 5.1 shows the size of the ledger for different numbers of nodes.

Figures 5.1 and 5.2 show the ledger size for each simulation run, by the number of nodes, for 1,000 and 5,000 products respectively. Figure 5.3 shows the average ledger size for each total number of nodes.

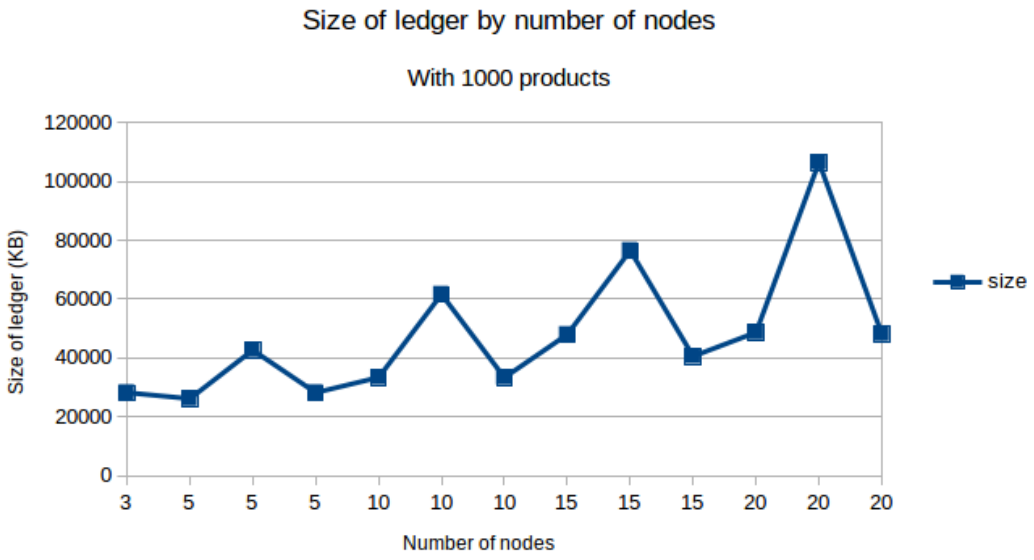

FIGURE 5.1: Ledger size by number of nodes, with 1,000 products

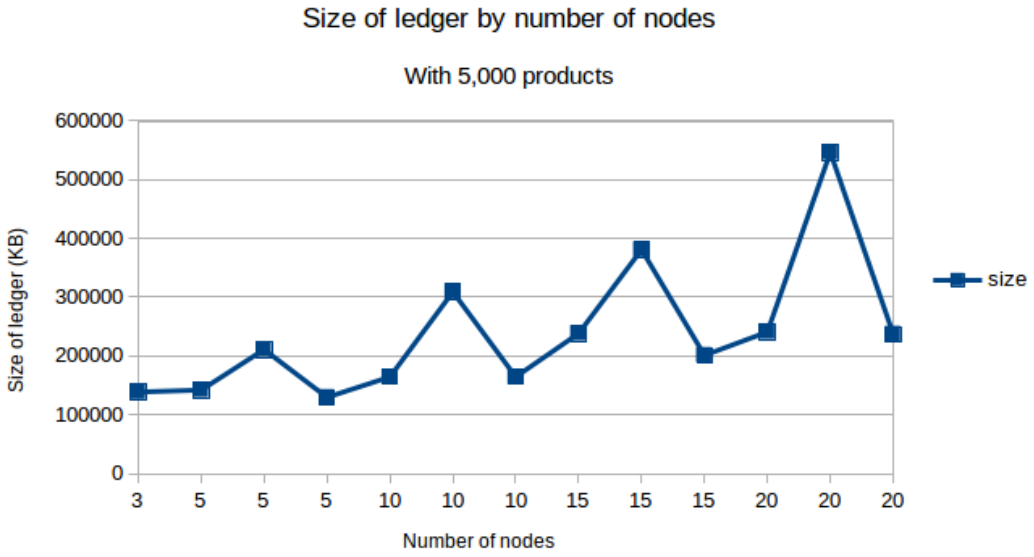

FIGURE 5.2: Ledger size by number of nodes, with 5,000 products

Figures 5.4 and 5.5 show the relationship between number of Shipper nodes and the ledger size.

The average size of the ledger compared to the number of fish stored in the blockchain can be seen in figure 5.6. In order to get a sensible presentation, we took the logarithm of each value, as we have mostly used numbers that are factors of ten apart.

5.2.2 Execution Time

We also measured the execution time in a variety of ways. Both the total execution time of the simulation, and the time taken for each individual blockchain interaction were measured.

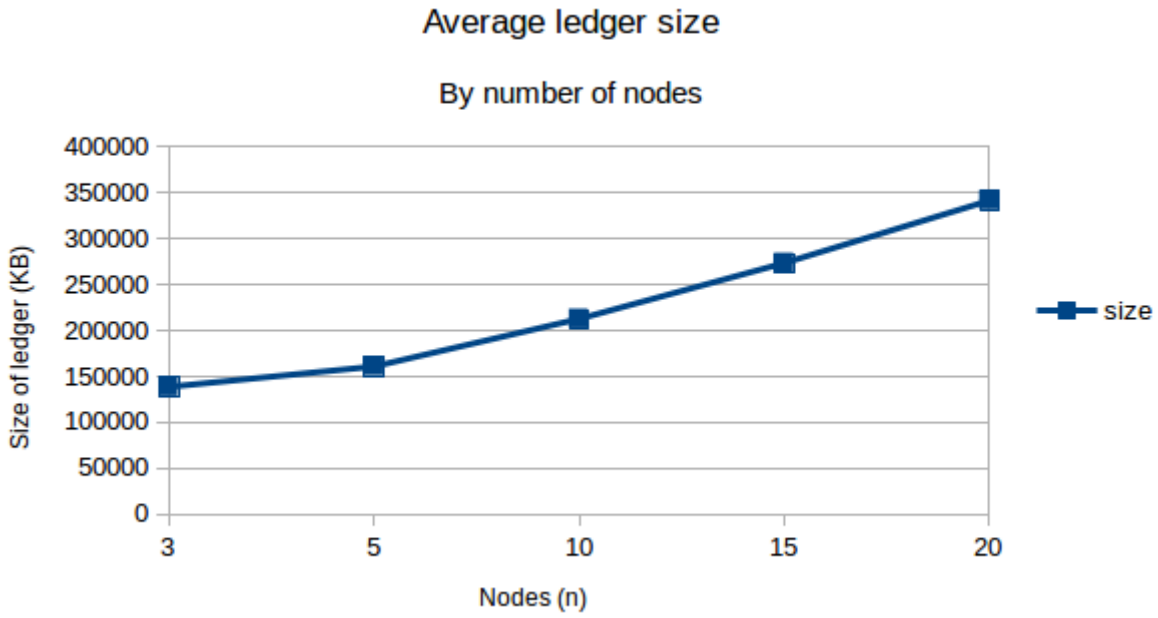

FIGURE 5.3: Average ledger size by number of nodes

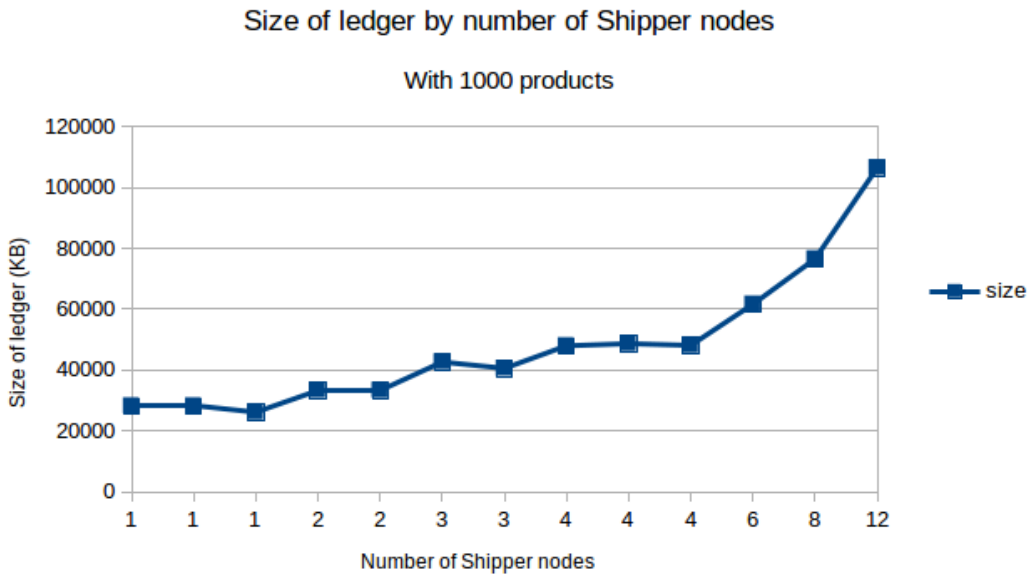

FIGURE 5.4: Ledger size by number of Shipper nodes, with 1,000 products

Single Events

For the single events, that is to say operations with blockchain interaction, we measured the average time taken for each such event, through the total execution of the simulation.

Figure 5.7 shows the average time taken to complete the individual events, for different numbers of products. We took the logarithm of the number of products to properly scale the x axis:

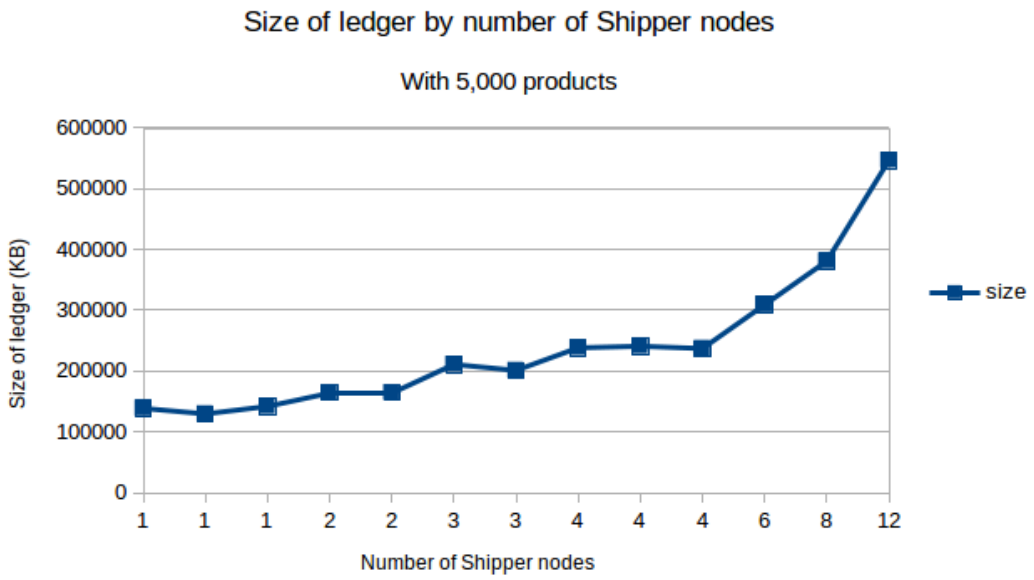

FIGURE 5.5: Ledger size by number of Shipper nodes, with 5,000 products

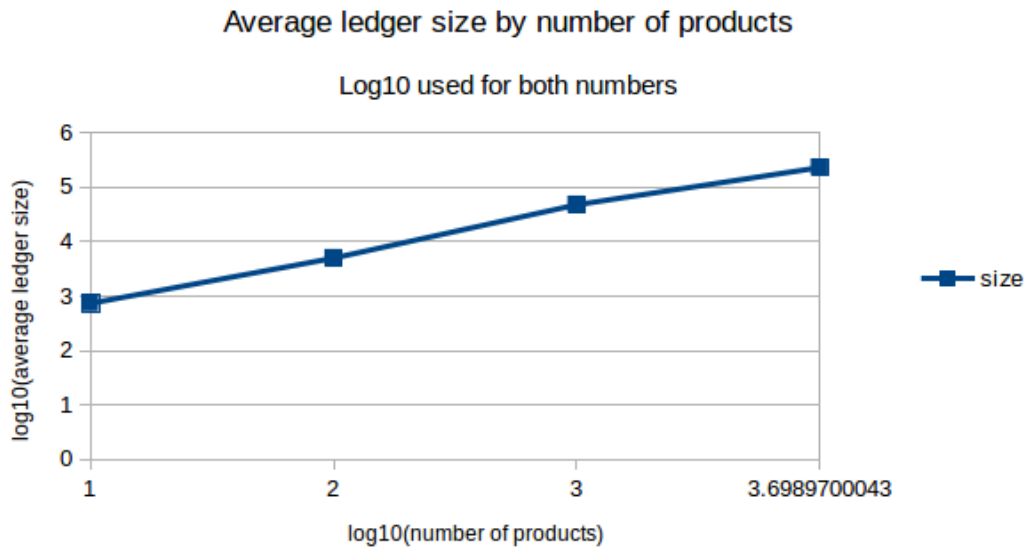

FIGURE 5.6: Logarithmic average size of the ledger, by number of products

Total Execution Time

The total execution time was measured from the time when we first started the peer nodes, to when every product had reached its destination.

Figure 5.8 shows the average total amount of time spent on executing the simulation, based on the number of products.

In figure 5.9 we show the correlation between total time spent on the simulation, and the total number of nodes in the network, when running the simulation for 1,000 products.

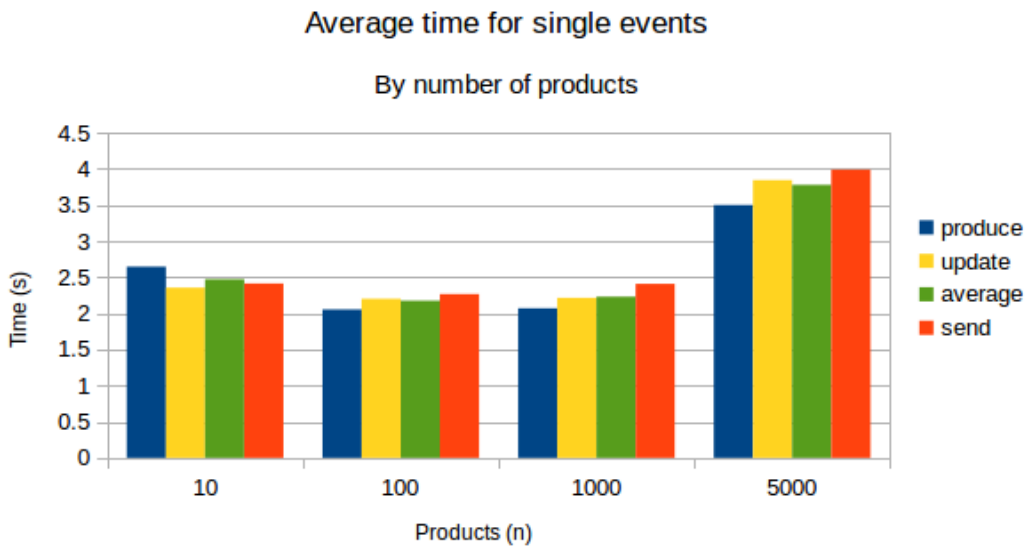

FIGURE 5.7: Average time taken for individual events, by number of products

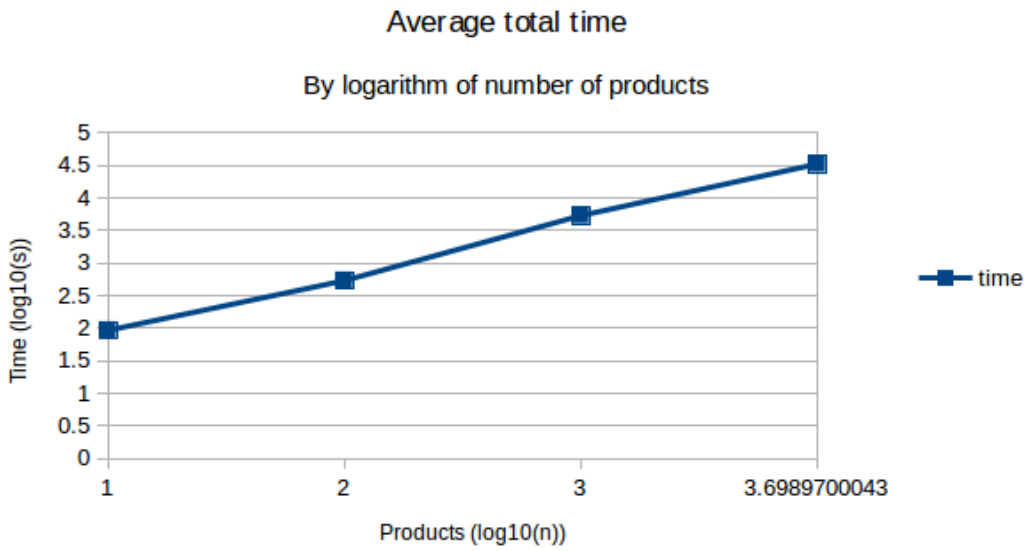

FIGURE 5.8: Total execution time, by logarithm of number of products

5.3 Comments

In this section we will talk about the findings, going by the data presented in the previous sections. The results will be discussed in more detail, and will be contextualized with our research questions.

5.3.1 Data

In some of the test scenarios, there was a very noticeable change in the average time taken for certain operations. This can for example be seen in table 5.2, in the field for

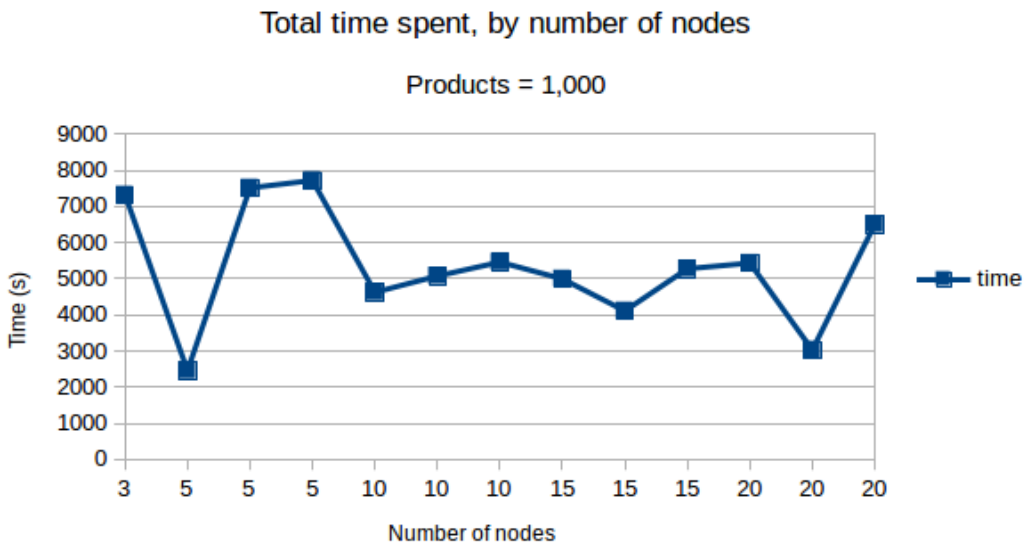

FIGURE 5.9: Total time spent, by number of nodes, with 1,000 products

updates with 1,000 products. Here, The average time increases with a full second when compared to the same operation with 100 products. It is also nearly a second longer when comparing it to the same stat with different configurations.

Ledger Size

Increasing the number of Shipper nodes significantly increases the ledger size. This increase is to be expected, as having more intermediate nodes will directly lead to more transactions being made. The change is noticeable even with as few as five nodes.

In figure 5.6, we can see the average size of the ledger compared to the number of products. We used the logarithm for each value to get the data on an easy to understand scale. In the figure, we can see that the size increases at a more or less linear rate. Since we take the average of all the data sets, the differences observed in figure 5.1 are mostly mitigated. In fact, looking at figure 5.1, we can see that there is a recurrent pattern to the ledger size, for each total number of nodes. When taking the average, this is flattened out.

Observations

One thing we observed while running the tests, was that the Producer nodes finished a lot faster than the rest of the simulation. However, this is a natural consequence of our test setup, where the Producer nodes are the only ones running truly in parallel. The Shipper nodes rely directly on each other, and are the bottleneck of our system.

The laptop used for running our simulations was relatively old, and was at the time of purchase a mid-range model. In other words, it is not a particularly powerful machine, with a mediocre processor. However, even when running the simulation with thousands of products and up to 20 simultaneous nodes, it never reached 100% CPU usage. That is to say, even though it was put under heavy strain, it did not reach its limits. As such, we can with some confidence say that it was not the bottleneck of our simulation. This claim is further substantiated by the fact that increasing

the number of nodes and products in many cases lead to increased efficiency, for example by looking at the execution times in tables 5.1 and 5.12. Rather, it would appear as though the amount of products has a bigger influence on the performance of our simulation.

The ledger size would always increase in size by multiples of four kilobytes. The smallest observed change was exactly four kilobytes, but it would occasionally be more. These changes were tracked manually rather than logged automatically. It would have been possible to automatically track the changes over time, but analyzing this log in conjunction with the rest of the data would have been a highly challenging task that we had not accounted for when planning the research.

Block Size

In our tests, changing the preferred block size turned out to be completely inconsequential. Some slight variances can be observed in tables 5.14 and 5.15. However, the differences are statistically irrelevant, as any number of factors could have affected the simulation to the degree observed in our tests.

The biggest change in the performance of the system can be seen when going from 1,000 to 5,000 products. As we only tested block size for 100 and 1,000 products, it is possible that when increasing the number, we would have seen a change.

5.3.2 Research Question 1

Our first research question was:

- How will a blockchain-based supply chain management system perform on different scales?

Execution Time

Figures 5.7 and 5.8 show that there is an increase in the average time per blockchain event, and also the average total time spent on execution, both by number of products.

In an ideal scalable system, the average time per event should be constant after running for a sufficiently long amount of time to allow for stabilization. However, in figure 5.7, it can be seen that the time per event varies between each number of products. Running the simulation with ten products results in a slightly higher average time than with 100 and 1,000. As ten is a very low number, the difference could simply be because of a lack of caching, where the computer is not given enough time to improve on the performance. More significant is the increase when going to 5,000 products. The average time taken is more than a second longer for the scenarios with 5,000 products, roughly a 60% increase.

When increasing the amount of products in the supply chain, it is natural that the total time of the simulation also increases, as there are more transactions that need to take place. Nevertheless, the average time increases in a linear fashion, as can be seen in figure 5.8.

No correlation was observed between the structure of the supply chain and the average transaction time. Produce is on average slightly faster than the other two transaction types measured. Update and Send take more or less the same amount of time, and have a higher variation in the measurements.

Location Updates

We found that updating the location of a product is as time-consuming as registering a transfer of ownership, or even the creation of an entirely new product. This is a major weakness of the system we have used. In a scenario with a high number of products, having all of them trying to update their data fields would very quickly bog down the system. It is hence a major bottleneck for a large scale supply chain.

Looking at the code, it would have been easy to predict this behaviour. In our implementation, location updates are handled programmatically in the exact same way as any other ledger update. That is to say, there is no difference between simply updating a data field, and creating a new product altogether.

Ledger Size

The size of the ledger increases both with an increased number of products, and with an increased number of Shipper nodes. We found no correlation between the pure number of nodes and the size of the ledger. Rather, with an increased amount of Shipper nodes, the amount of transactions needed for a product to reach its destination is directly increased.

Figures 5.1 and 5.2, and 5.4 and 5.5 show that the ledger size is affected directly by the configuration of the nodes. The graphs show the same pattern, with both 1,000 and 5,000 products.

5.3.3 Research Question 2

The second research question we formulated was:

- How viable are blockchain-based supply chain management systems for use in sustainable industries?

Physical Architecture

The first thing to note is that a blockchain-based supply chain management does not necessarily suffer from the same energy problem as Bitcoin or Ethereum. In these networks, a transaction fee must be paid for each transaction, as a means to motivate the miners to include the transaction in the next block. In essence, the whole network is based on expending energy to mine, as transactions can only be finalized by mining. However, in a private blockchain, no such fee is necessary. As all peers have a common goal, of moving a product from source to destination, they will all be motivated to validate a transaction without an economic reward. Such is the case for the supply chain management system used here.

The technical requirements for running the simulation are quite low. It was possible to run the simulation at all scales, on a single computer, without the computer ever freezing or otherwise getting overloaded. When distributing the execution among several physical machines, there is no reason to believe that the hardware would be a bottleneck.

5.3.4 Software

The software utilized in the technical part of this work was primarily based around Hyperledger Fabric. The decision to use Hyperledger Fabric was reached after a process of looking into what options existed.

Hyperledger Sawtooth was also considered as an option, as it is a specifically designed blockchain framework for supply chains. However, as it is a very young technology, it is lacking in both official documentation and available online resources.

Hyperledger Fabric is a more mature project, with a comprehensive documentation, and a notable amount of resources available in the form of online posts and code patterns. The code patterns are a particularly valuable resource, as they have the potential to save a significant amount of development time.

5.3.5 Shortcomings

There were significant delays when working on this thesis. As outlined above, and further expanded upon in Chapter 6, we went through several stages of development. Coupled with the delays from moving to a new country, we were hence left with a limited amount of time, which affected our research.

Not being able to run the tests on a distributed system is undeniably a shortcoming. While we did create a blockchain-based solution, it lacks the key part of a blockchain application, namely the distributed network.

The original plan was to perform tests with 10,000 and 20,000 products. However, seeing as the tests with 5,000 took between eight and 19 hours, we simply did not have enough time to carry out the tests at such a large scale.

Chapter 6

Discussion

Chapter 6 is a discussion of the process of this thesis, as well as of the findings from our research. The first section outlines the challenges encountered during the time working on the thesis. We then present some of the caveats to consider when discussing the results of the research. The following section is a discussion of the findings, contextualized with the research questions. Finally we talk about threats to the validity of our research, and how the threats can be mitigated in the future.

6.1 Challenges

It is a given when working on a project that there will be challenges, especially when it is as large of a project as a master thesis is. In this section we will outline the challenges we faced while conducting the research and writing this thesis.

6.1.1 Hyperledger Sawtooth

During the first period of the thesis work, we looked into using the Hyperledger Sawtooth project. More specifically we looked at their example implementation of a supply chain management system for tracking fish from fisherman to plate. While the Hyperledger Sawtooth Supply Chain is a well-written piece of software, it is still lacking in several aspects. The Sawtooth framework itself is very well documented, but the documentation is highly technical, with little practical information. For a more long-term project, using Sawtooth could be a good option. However, with limited time, the current state of the project is too complex.

An additional issue is that Hyperledger Sawtooth is an immature piece of software in certain aspects. This can be seen in cases such as when trying to get the system up and running: at the time of writing, simply downloading the code and following the instructions on the associated Github repository is not enough to get it working. The latest version of the code uses settings that are not compatible with the dependencies. While the solution to this problem is fairly simple, applying old patches from the repository, discovering what the problem was required more effort than it is worth.

After we had gotten the Hyperledger Sawtooth set up and running, we still had problems with data extraction. There was no obvious way to track the distributed ledger to see how it would affect the parties involved. Even though the entire system was supposed to be running locally, locating the storage was challenging. It could be that it was simply kept in the browser, but even when looking into the available data we had no luck.

An option could be to track every block update. However, this would mean that we would have to derive the results, with an assumption of the data used. Also, we could not find an explicit definition of the structure of a block, so even that would

have to be guessed from the code interacting with blocks. In the end, all these issues would have lead to too inaccurate results, and we decided to abandon the Hyperledger Sawtooth.

The Great Firewall of China

Another problem that we encountered with using Hyperledger Sawtooth is unrelated to the code itself, but rather a challenge with working from China; certain dependencies of the code are unavailable because of the Golden Shield Project, also known as the Great Firewall of China [49]. This problem was solved by utilizing the VPN from NTNU, available to all students and employees at the university.

The firewall also caused a more persistent challenge: due to the convenience of having the thesis draft continuously available to all the people involved, a shared online text editor was preferred. For stylistic reasons and as a matter of standardization, Latex was used to write up the thesis. As such, the choice fell on Overleaf, a free, online Latex editor. However, as the firewall slows down communication that crosses the borders of China, the Overleaf website was slow to connect to, and at times we experienced disconnections while in the middle of writing. While it is not a major issue or challenge, it made the process of writing the thesis more time-consuming.

6.1.2 Hyperledger Fabric

After deciding to look into other options than the Hyperledger Sawtooth, we tried the Hyperledger Fabric. Hyperledger Fabric as a project is a lot more mature than Sawtooth. This means that there it is better documented, and there is a lot of user created content available online, which makes it much easier to develop applications for, and one can draw from others' experiences when encountering error messages.

A challenge common to both Hyperledger Sawtooth and Fabric, is that they are enormous projects. When trying to understand how the system is put together, it requires attention to detail, and keeping track of a lot of components and functionalities.

Hyperledger Fabric is very specific about the versions of its various dependencies. While this can be dealt with by having a keen eye when setting everything up, it was somewhat complicated by the fact that some versions are not the latest. This lead to unnecessary time loss from needing to track down errors, and then finding the right version to install.

Supply Chain Solutions

There are several code patterns using Hyperledger Fabric, and we decided to try one called *blockchainbean2* [23]. Although it looked promising, we came across a strange problem, in that it was impossible to download the repository from Github. After several attempts, with and without using a VPN, using both git clone and the option of downloading a ZIP file from Github, we abandoned *blockchainbean2*. The cause of the downloading issue could very well be the Great Firewall. However, it was possible to clone other repositories using the exact same setup, so we are still not sure about the exact cause.

Having abandoned *blockchainbean2*, we looked for other options using Hyperledger Fabric. Two promising candidates were found: 1) *Tuna App*, which is an example implementation of a supply chain made by IBM [48], and 2) *Blockchain Supply*

Chain made by Github user *mattdean1*. However, as it turns out, there were other problems when trying to use these other two: for the Tuna app, there was no convenient way of changing the parameters of the underlying blockchain. Once again, this is a significant problem, as it hinders us from testing what we set out to do. As for the Blockchain Supply Chain, it looked to be fairly easy to modify, but we could not get it to run. As far as we could tell, the application level implementation of the program does not work with the currently available Hyperledger Fabric images. More specifically, the application was built for a specific version of the Fabric, but some of the relevant images were simply not available from the official websites. In the end, we saw no way to amend this problem.

Proprietary Solution

Having tried a variety of different implementations, to no avail, we decided to create our own. However, being left with very limited time at the point of realization, this would have to be a rudimentary simulation. Most likely, it would have been possible to get at least some of the projects we looked at to work properly, and even modify them as we wanted. The problems were mostly due to inexperience and lack of time. Had we had more time to properly delve into the details of the various implementations, it is likely that we could get them to work. Lack of experience with working with blockchain solutions, specifically ones under the Hyperledger umbrella, also limited what we were able to do.

While working on our own supply chain solution, a major hurdle was the deployment of the chaincode. If one makes changes to the chaincode, restarting the network does not cause the new one to be put into use. Even realizing that the new chaincode was not being used took a fair amount of time, leading to a long process of error tracking. After realizing the fact, coming up with a solution for it was another long process. The intended method when using Hyperledger Fabric is to utilize the *peer* program that comes with the repository. Using this program, one can upgrade the code being run by peers in the network. One of the peers is running the chaincode. However, trying to use *peer* for this purpose was fruitless. The documentation is at this point in time still very lacking when it comes to this feature, making it challenging to understand the usage. There is also little general information to be found when looking at other resources, although there are certain posts online looking at specific examples. In the end, a solution was found at Stack Overflow, with a user describing the steps they used in order to upgrade the chaincode [34]. While this solution does work, it is not ideal, as it involves bringing the entire network down and rebuilding it from scratch, rather than upgrading only a part of it.

6.1.3 Distributed Server

When we started looking into running the simulation as a distributed network, we quickly realized that it would be a complex issue. As mentioned above, Hyperledger Fabric is a massive and complex project, and setting it up properly can be quite challenging. That being said, the basic network supplied by the *Fabric Samples* repository offers a good start for developing applications. However, it is only designed for being run locally. There is a large number of different configuration files, and working out the correct ones to edit takes time to understand. Furthermore, the exact lines to edit, and features to add, is even more complicated. Luckily, we came across a tutorial, demonstrating how to go about this problem [2].

Unfortunately, a final problem was encountered: actually deploying the working simulation on a distributed server. Even before making the changes to run the simulation on a distributed network, we had to get Hyperledger Fabric to run on the servers in the first place. Setting up Hyperledger Fabric to run on the servers we wanted to use proved to be a big challenge. We had already encountered several issues when working locally, but the errors at this point were different. Unfortunately, we were in the end left with too little time to work out the problems, and had to abandon the attempts.

Unfortunately, this was partially due to bad planning of the development of the simulation. Getting Hyperledger Fabric to run on the server should have been a higher priority, and finalized earlier. The reason why this did not happen, was that we had wanted to sort out the main issues with the locally run simulation before looking into how to use the servers. As the problems turned out to be more challenging than expected, we were ultimately left with too little time to figure out the issues we faced.

6.1.4 Open Source Projects

Open source refers to when the source code of an application, program or similar is fully available to the public [25][10]. There are generally very few terms of use, which means that anyone can take the code and modify it to fit their own needs. This code can then be used for almost any purpose, for example academic research or even commercial use.

However, open source projects are most often not commercial themselves, and frequently community-driven. This means that there is no economic incentive to provide usability or providing a thorough documentation. As we have highlighted above, we had several problems when trying to work with open source projects. The best example would be when trying to utilize the Hyperledger Sawtooth; It is a highly complex system, and it is still young. This meant that the documentation at time of writing was highly limited. There had been no incentive to provide a thorough documentation or similar explanation of how to use Sawtooth at the time of publishing it.

6.2 Caveats

There are several caveats to the simulation we utilized for our research. A simulation can never perfectly represent the real-world scenario it is demonstrating, but one should strive for realism. We will in this section present certain liberties we took in developing our simulation, and discuss the consequences of not including these features. We will also discuss why we thought it acceptable or necessary to leave out these aspects.

6.2.1 Distributed Network

A supply chain will in general have parties spread across a large physical area. In the *blockchainbean2* project for example, the coffee beans are harvested in Ethiopia, then sent to Djibouti to be packaged, before they are sent and delivered to New Jersey, USA. While we did run our simulation on a distributed computing platform, that is to say a cloud computing service, we have no control over the physical distribution of said platform. For all we know, the servers could be in the same room, or even

directly connected to each other. Conversely, they could also be at different locations, perhaps a considerable distance apart.

As there is no way for us to verify the distances between the servers, we are unable to claim with certainty that changes to the distance will not affect the performance. That being said, considering the bandwidth of the servers used, the comparatively low amount of data being sent, and the speed of the modern Internet, the probability of the impact being very significant is quite low.

However, there are other risk factors that we have not looked at. For example, what if there is a node with a very slow Internet connection? Or what if a node loses its Internet connection altogether? These are factors that are hard to account for, and even harder to test for. Even so, they are events that could easily happen in the real world, and putting a system into use without having accounted for the risks could potentially be dangerous. Unfortunately, the tests required would be too laborious and time-consuming to carry out in the process of this thesis, so we will have to leave them for future efforts.

6.2.2 Physical Aspects

We did not include the physical aspects of a supply chain management system. With physical aspects, we refer to things such as the physical tag that would be attached to the product being shipped, and the sensors included in this tag. The main reason to ignore the physical part, is simply that this would fall under electrical engineering, or even infrastructural engineering. The scope of designing such a tag is well beyond the scope of this thesis.

That being said, there are certain software engineering aspects to the identification part of the tag. It is necessary to be able to identify uniquely every tag, and that it cannot be copied or otherwise faked. Uniquely identifying every product is vital to providing proof of provenance for it. However, identification verification such as this would rather fall under cryptography, and is as such also outside the scope of this thesis.

Nevertheless, in a system such as the one we are simulating, it would be an assumption that such a verification system already exists or that it is co-developed to be put into use with our supply chain management system. Something we did include, was a simple feature of sending data updates, representing the sensors. Unfortunately, because of how the simulation was set up, these were sent from the nodes themselves, and not from independent units. Even so, an update to the blockchain should be essentially the same, regardless of source.

Another key reason why we did not include the physical aspects of the supply chain, is that they should not make a significant difference. The sensors send continuous data updates, which we to some extent can simulate. Other than that, the only times the physical part comes into play is when creating new products and when changing the owner of a given product. When registering a product, it would be necessary to associate the data representation with the identification of the tag. Similarly, when trying to change the owner, one would have to verify that one is in possession of the actual product by scanning or otherwise identifying the tag.

It could also be argued that the supply chain management system is only relevant after one has proven the possession of the product. The scanning and verification are not necessarily parts of the management system itself, but rather supporting parts. As such, we are free to work on the core functionalities, while leaving the remaining parts to be developed separately.

6.2.3 Access Control

Certain parties might have a reduced set of privileges in a realistic network. For example, there is no need for the shipping companies involved to be able to add new products. In fact, it is counter intuitive, and a potential security risk. There is also the aspect of privacy and data security. Even though the various parties are cooperating, they will want to keep some data private and hidden from the others. The private data can for example be information about the employees, or how much money they spend on each transaction.

Access control is in itself a complex topic, with many different models and techniques. It would again be natural to use a ready-made solution, rather than taking the effort upon ourselves of creating one from scratch. Besides, creating one ourselves would be a security risk, as information security is a highly specialized field, with exploits and security holes being constantly exposed. As such, using an existing, cryptographically secure solution would make a lot more sense.

Nevertheless, we have chosen not to use any such solutions in our simulation. We left out access control, because we do not believe it relevant to this thesis. In a simulation, we have complete control over the system. Hence, we are able to control which part accesses what, without having to account for eventualities of illegal accesses. Besides, access control would be mostly relevant when setting up the network, which is not a topic of this thesis. One could imagine a scenario where, when connecting the various parties to the network, they would be provided an API with the functionalities they require. This way, each party will only be able to perform the tasks they need to, and there will be very limited effects on the performance of the system as a whole.

6.3 Findings

In this section we will briefly discuss some of the findings from our research.

6.3.1 Ledger Size

Looking at the figures 5.1 and 5.4, it is clear that there is a closer correlation between number of Shipper nodes and ledger size, than the pure number of nodes. In our system, the correlation is a natural consequence of the Shipper nodes being connected in a linear fashion, leading to a higher number of transactions. It is in fact the clearly dominating factor, as figure 5.1 shows a sharp spike in ledger size for each number of nodes. The difference within each category is greater than the difference when simply comparing the number of nodes.

6.3.2 Block Size

For some of our tests, we changed the preferred max size of the blocks stored in the blockchain. Changing this parameter turned out not to be significant with regards to performance or ledger size. However, no changes to the data is a result nevertheless.

The preferred max size of the blocks is at the scale of hundreds of kilobytes, to tens of megabytes. Comparatively, the size of the ledger at the end of our simulation is at most at the scale of hundreds of megabytes. For the simulations where the preferred block size was tested, the ledger only reached a size of roughly five and 47 megabytes for the different test configurations. As such, each transaction took up

a minimal part of a possible block, and the amount of blocks would simply not be affected.

That being said, we carried out two tests outside the ones described in Chapter 4. For the first, *PreferredMaxBytes* was set to 4KB. 4KB was earlier identified as the minimum change in ledger size. For the second, *AbsoluteMaxBytes* was also set to 4KB. However, neither of these values resulted in any noticeable change in behaviour or performance.

Hyperledger Fabric is a highly complex system, which means that predicting its behaviour ahead of time is impossible without being intimately familiar with its functionality. Given our inexperience with working with Fabric, it is possible that there is another parameter that could be changed, that would have made an impact on the performance.

6.4 Research Questions

Research questions are a fundamental part of all research, as they are what defines the topics and scope of any report. The research questions then also provide a basis for presenting and discussing the findings. In this section we will discuss the results presented in Chapter 5, and how they might answer our research questions.

6.4.1 Question 1

Research question 1 was as follows:

- How will a blockchain-based supply chain management system perform on different scales?

The motivation for this question came from our systematic mapping study conducted during fall last year. While we found a variety of proposals and suggestions for blockchain-based supply chain management systems, we did not come across many studies looking at their viability. Rather, the studies found were mostly quite theoretical, for example proving mathematically the correctness of their implementations. Such research is both relevant and useful, but it is lacking the practical dimension of putting the supply chain system into use. As such, we saw the possibility of contributing to the literature with a more practically oriented look at how blockchain-based supply chain management systems perform under load.

We defined scales to refer to a selection of parameters, namely:

- Number of products
- Number of nodes
- Block size

In order to measure the impact of adjusting the parameters, we looked at the following variables:

- Time to register a new product
- Time to send a product along the supply chain
- Time to update a parameter of a registered product
- Total execution time for a simulation run
- Size of the ledger stored on disk

Execution Time

The average time per event increased significantly when running the tests with 5,000 products, as can be seen in figure 5.16. However, figure 5.17 shows that the average total time increases linearly. Interestingly, despite the event times being lower on average for 100 and 1,000 products, the total time increases at the same rate as with the higher event time for 5,000 products.

Comparing the increasing event time with the linear total time suggests a level of parallelism in the system. That is to say, more than one event can be processed at the same time. Multiple transactions finishing simultaneously makes sense in a blockchain-based system. Transactions are batched together and included in a block. The block has a maximum size, meaning how many transactions can be included, meaning that unless a significant number of transactions are submitted at the same time, they will all be included in the same block.

As can be seen in the tables 5.14 and 5.15, changing the preferred block size did not notably affect the execution time. As discussed above, the data sets used in the simulations were probably not large enough to affect the blocks to a noteworthy degree.

Ledger Size

The results from Chapter 5 show that the ledger size increases roughly linearly with number of products, as seen in figure 5.6. An increase lower than linear would mean that there is a large overhead to get the system started, that becomes less relevant as the data set grows. On the other hand, an increase larger than linear would mean that there is an overhead involved with the transaction, beyond a simple data entry. A linear growth rate suggests that there is no significant overhead in either direction.

However, the size of the ledger is naturally directly correlated with the size of the blocks. In our tests, we were unable to produce a different result by changing the preferred size, as seen in figures 5.14 and 5.15. It might be that the blocks are too large compared to the data used by each transaction in our simulation.

Figures 5.1 and 5.3 show the size of the ledger by the total number of nodes. The first graph, figure 5.1, shows that there is a big variety among the simulations with an equal number of total nodes. The variation is due to differing structures of the supply chain. As can be seen in the tables of raw data, the simulations with a higher number of Shipper nodes result in a significantly larger ledger. Figure 5.3 shows the average size of the ledger for total number of nodes. There is a slight, linear increase in the ledger size with an increased number of nodes. A set of three simulation runs were carried out for each total number, with different configurations of node types. However, the average number of Shipper nodes increases for each increment. As such, it would appear as though the total number of nodes is not a major influence, but rather the number of Shipper nodes.

Evaluation

The average time per event appears to grow with a large number of products. If the transaction time becomes too large at a certain size, it could be problematic in a real-life scenario, if the supply chain is time sensitive. With the scale of the tests in this thesis, the increase is significant, but the numbers are still fairly small. The total execution time suggests a level of parallelism which might alleviate performance issues caused by the individual transaction time, although this might depend on the structure of the supply chain.

The ledger size appears to grow at a linear rate. Considering that each transaction needs to be stored as a separate data field, a linear increase is to expected. The structure of the supply chain is highly relevant to the size of the ledger, as a high number of intermediate nodes will directly increase the number of transactions and hence the size of the ledger. There is also a correlation between the total number of nodes altogether, and the size of the ledger. Thus, in a supply chain with a high number of Shipper nodes, the ledger size might increase at a higher rate than the aggregated numbers found in our research.

Overall it would appear as though the system used in this simulation is fairly scalable. Ledger size and total execution time increase at a linear rate, with quite low constant factors. Transaction times are low, but appear to increase at larger scales. However, the transactions also show potential for parallel execution, which would alleviate an increased time needed for any single transaction.

6.4.2 Question 2

Research question 2 was:

- How viable is a blockchain-based supply chain management system for use in sustainable industries?

Viability is not possible to measure in a quantifiable manner. To answer the question, we will hence have to look to analyses of the data gathered during the research.

Performance

The simulation used in this thesis had overall quite low technical requirements to be run. It was fully possible to run a full simulation of 20 nodes and 5,000 products on a single, low-end machine. The hardware should as such not be a limiting factor for any company or individuals wanting to utilize a blockchain-based supply chain management system.

Storage space depends on the structure of the supply chain, as well as the numbers of products being registered. The largest size reached in the simulation runs was 546,712KB, or around half a gigabyte. That maximum size was reached with 5,000 products and 20 nodes, 12 of which were Shipper nodes. In comparison, the Bitcoin network's blockchain has reached a size of approximately 220GB, over the course of over ten years [32]. Even on a normal, mid-range personal computer, 220GB is in no way an insurmountable amount of storage space.

In a real-life scenario, the network would naturally be run online. The relatively low amount of data generated in our simulation points to fairly low bandwidth requirements. An internet connection is required no matter what, as it is a prerequisite for a distributed network, but the quality and stability of internet access varies greatly from place to place. Low requirements to bandwidth and latency are hence highly beneficial, to facilitate widespread access and use.

Open Source

For our simulation we used an open source project as the basis. As described in Chapter 3, it means that anyone can use both the original project, and our derived work as they see fit.

However, the project being open source also means that any potential users will have limited access to support and help, should they have any problems. The Hyperledger project as a whole is quite well-supported, with industry partners offering financial support, which makes it more accessible. Even so, the developers and maintainers are under no obligation to help you, should you run into problems while using their tools.

A large company would be able to pay for a license as well as customer support to use a ready-made solution from a third party. With enough resources it is also possible to develop and maintain a proprietary solution, as the shipping giant Maersk has done in cooperation with IBM [59]. By developing a purpose-made solution, measures can be made to ensure functionality and performance. However, even Maersk and IBM did not build an entire blockchain framework from scratch, but rather based their solution around Hyperledger Fabric, which goes to show the complexity of the matter.

To truly meet the sustainable development goals, innovative and sustainable technologies should be available to and usable by all, not only large-scale corporations. If the cost of implementation or deployment is too high, a technology will become inaccessible to some, especially to poor communities. Hence, communities with limited opportunities will be prone to be victims of unsustainable practices.

Open source projects have the possibility to alleviate the cost of production. The code will by definition be available to anyone, and can be used as seen fit.

Complexity

The development of the simulation used in this thesis went through a series of stages. As mentioned above, developing an entire blockchain solution for supply chains is an immense undertaking. By looking at the source code of Hyperledger Fabric, one can appreciate the complexity of the technical solution offered by the developers behind Hyperledger.

Although the workload is reduced drastically by utilizing existing blockchain solutions, a significant amount of work is still necessary. As highlighted in the section above on challenges faced during the work on this thesis, understanding and working on Hyperledger projects can be demanding. A lot of time is necessary to sufficiently understand the structure of the framework and how it works. The simulation used for our research was heavily simplified ignoring important aspects of a system to be used in real life scenarios.

For the supply chain management system designed by IBM and Maersk, the code is available on Github [24]. The application is made to comply with laws and regulations from the U.S. Food and Drug Administration, meaning that the supply chain could in theory be deployed by companies to export food to the U.S. The complexity of the system is well above that of the one used in this thesis. Developing a supply chain management system of the same scale and quality would be a significant challenge for a small company, or a company with limited funds.

On the other hand, the availability of such code patterns and examples means that a lot of the development work can be bypassed. It is, for example, possible to for a company to take and modify the code from IBM and Maersk to make it fit their industry, hence saving the first steps of development. Bypassing the early stages of development was the main motivation for using an open source project in this thesis. Nevertheless, despite the option of using frameworks, the remaining application layer is quite complex, as can be seen in the IBM and Maersk source code.

Evaluation

There exists a variety of open source projects that offer a base for developing a blockchain-based supply chain management system, or that offer a full system altogether. By utilizing such projects, the amount of work needed get a simple solution running is quite low, and accessible to anyone regardless of budget.

However, there are significant challenges even when utilizing available open source frameworks. Projects such as Hyperledger Fabric and Hyperledger Sawtooth are huge and highly complex. The complexity is a challenge to overcome when developing applications based on such frameworks.

Regarding hardware, there is little to no reason to believe that the necessary hardware to run a blockchain-based supply chain management system will be too costly or otherwise inaccessible to anyone.

6.5 Threats to Validity

Similar to the part on our systematic mapping study in chapter 2, we will here discuss the threats to the validity for our research in this master thesis.

6.5.1 Realism

We base our research on a simulation. A good simulation must be as realistic as possible. Due to time constraints as well as limited knowledge of the subject matter, we were unable to reach the level of detail we would have liked.

One option we considered in the early phases of the work, was to have a more realistic supply chain, at the cost of not utilizing real blockchain technology. That is to say, we would be simulating the behaviour of a blockchain, but without any true blockchain components. Instead, we decided to have a less realistic supply chain, but using an actual blockchain implementation.

Testing Environment

The testing environment ended up being a single laptop, and a fairly old model at that. While it is not completely unrealistic for a small company or independent operator to be using a low-end computer for their server purposes. However, a dated laptop would most likely never be used as a server. Furthermore, we simulated a whole network on a single computer. Both these factors take away from the realism of the simulation.

While the simulation ran without problems on the laptop, without requiring all available resources, it is possible that the computer simply limited the resources available to the simulation. That is to say, the computer may have actively slowed down the execution of the simulation. As such, the average transaction event time and total execution time may be higher than necessary.

Code

The code used in our simulations placed a higher emphasis on utilizing a blockchain technology than on presenting a realistic supply chain. The code includes the most important features of a supply chain, such as keeping track of products and their owners. However, a lot of work remains to be done in order for the code to be used in a real-life setting.

Certain simplifications were made for the purpose of having a self-contained simulation, which could be run without active interaction. The products being registered are purely digital, with no physical registration taking place. Similarly, the transaction of changing ownership and registering the final arrival of a product is done without any physical interactions.

6.5.2 Tests

For our research, we performed a variety of tests, exploring certain parameters. However, for lack of time, we were only able to carry out a fairly limited number. We tested for a number of configurations, but were only able to do one test for each. As a consequence, it is possible that some of our data represent outliers. Ideally, each test scenario would have been run several times, calculating an average and deviation, in order to ascertain the validity of the data. This could have been done for the lower values of products, ten and 100, as these simulations only ran for some minutes. The larger simulations, on the other hand, reached execution times of almost 20 hours at the highest.

Furthermore, a decision had to be made between testing tall and testing wide. By "tall" we mean a smaller selection of configurations, but more detailed scenarios. We went with the "wide" option, where we instead had a larger variety of configurations. The decision to test for a larger selection of configurations was made in order to cover more potential use cases. Real-life supply chains can have any kind of structure, and it was hence judged to be of more interest to look at more options, rather than make assumptions regarding which structures would be of most interest.

By automating the tests, it would have been possible to perform a greater number of tests. However, an automation would have also required that Hyperledger Fabric be set up on several computers. Being limited to a single personal computer was a significant bottleneck for the process of conducting tests.

Chapter 7

Conclusion

In this master thesis we have looked at blockchain-based supply chain management systems, and how they perform with regards to scalability and sustainability. Our hypothesis at the start of this research was that blockchain-based supply chain management systems are scalable, and can be used in real-life use cases. Our research suggests that supply chain systems based on blockchain technology can be decently scalable, and can be potentially viable for use in sustainable industries.

7.1 Conclusion

In this master thesis we have looked at a blockchain-based supply chain management system. The objective of the thesis was to look at the scalability of such a system, and how it relates to the topic of sustainable industries. Blockchain is a fairly novel technology, with a lot of potential to change and even disrupt industries. Traditional supply chain management systems are known for being opaque and complex, making it challenging if not impossible to accurately track a given shipment. By utilizing blockchain technology, it is potentially possible to provide accurate tracking of shipments, with sensor data along the way to guarantee the quality of the shipping conditions.

With more and better open source frameworks becoming readily available, it will only become easier to utilize blockchain technology in supply chains, for companies at all scales. The technical requirements to run the software are also quite low, making the technologies accessible to anyone. Overall, blockchain-based supply chain management systems appear to have the potential to improve the general sustainability of industries. However, to the best of the authors' knowledge, the environmental and economical cost of putting sensors on every shipment has not yet been sufficiently evaluated. Furthermore, changes to the software of supply chains does not address the issues of securing the physical shipment.

The complexity of implementing a blockchain-based supply chain solution can be a potential hindrance for widespread adoption. Creating a complete system is a significant investment which most companies would likely not be able to afford. It is possible to put pre-existing solutions into use, but even then there is a non-negligible amount of work to modify a supply chain management system to fit the needs of a given company.

Our research suggests that blockchain-based supply chain management systems appear to be scalable, and can potentially be beneficial to use in sustainable industries. However, further research is necessary, especially looking into the potential benefits and consequences a real-life implementation might have.

7.2 Future work

In this section we will discuss what we believe will be useful in future research. During the course of the work on this research and thesis, we came across several points that can be improved by doing further research. We also highlight some weaknesses with our methodology and implementation, what can be done to improve on these, and also what consequences they have for the understanding of the results of our research.

7.2.1 Security

In this thesis we did not look at how different cryptographic measures affect the performance and viability. For example, a consensus algorithm with less interaction between nodes would require less computing power and data usage.

A major point is the physical part of the supply chain. In this thesis we have considered a supply chain dealing with physical products, rather than a digital supply chain such as for providing software. As such, it is vital to ensure that the product you receive is indeed the one you ordered. In order words, the shipment should be tamper-proof. Our specific example has revolved around the shipment of fish. In the scenario we are imagining, the fisherman would attach a physical tag with a unique identifier to the shipment. While this tag would have features such as uploading location and temperature data continuously, we do not provide any measures to keep a malicious actor from simply replacing the product.

As an example of tampering, imagine the following scenario: we are sending a shipment of fish. The fish are stored in a box and the tag is attached to the box. At some point during shipping, a malicious actor gets access to the box, opens it and takes the fish. This will go unnoticed until the box is manually checked, as the tag does not have a way of monitoring the contents of the box to which it is attached.

7.2.2 Implementation

Our implementation for a simulation of a blockchain-based supply chain management system has some major flaws, highlighted in Chapter 6 under the section Caveats. The outcome was to be expected, considering the limited amount of time left when we were able to start developing the simulation properly. In the future, the flaws should be corrected, which would lead to a more accurate simulation and more dependable results.

Data Updates

One of the main arguments for making the change from centralized supply chain management systems to blockchain-based ones, is the possibility of having continuous monitoring of a set of sensors. However, in our implementation, this is done in a highly inefficient way. Updating a data field of the product registry is treated as a transaction, meaning that every data update has the same complexity as an ownership registration. Treating the updates as transactions was partially an oversight during development, where the computational complexity was not taken into consideration. However, by treating everything as a transaction, the time to develop a separate solution is saved, which could then be used to develop other parts of the simulation.

In order to avoid treating data updates as full transactions, the implementation would have to be changed significantly. As the supply chain management system

used in our research is set up, there is no way to perform computationally simpler interactions or operations. An alternative could be to provide an interface connecting to the physical unit, where the data is readable. As every physical tag would be connected to the internet to provide access to the data, it would be possible to have a simple communication system, where one would query for the data when needed, while the sensor data itself would be managed locally.

However, to properly develop and test the solution described above, one would either need a hardware setup, or a more complex software setup. The physical setup would be a more realistic approach, where there is a physical storage unit to query for data, but with a much higher cost of development. Not only would the prerequisite software be more complex, but there would also be the actual monetary cost of buying and setting up hardware. The software approach would not have the same monetary costs, but would have higher requirements to the hardware of the computer or computers running the simulation. In order to store the data independently from the blockchain, each product or shipment would have to be stored as its own entity, with capabilities of communicating with the nodes.

Local vs Distributed System

We were in the end unable to test our simulation on a distributed system. In a real use case, the supply chain management system will be set up in a distributed network, possibly spanning across the world. Only testing our application on a single computer is hence very lacking.

By implementing a distributed version, it will be possible to measure the bandwidth requirements of the network. A relevant measure would be the total amount of data being communicated between the nodes. Another interesting topic of research would be how the network would deal with unstable internet connections. Researching how the network deals with different internet conditions could be highly relevant, in order for the system to be used in a wide variety of locations and situations.

7.2.3 Channels

Hyperledger Fabric utilizes a concept they refer to as *channels*. Channels serve the purpose of connecting two or more peers in the network directly, meaning that communication between them using their channel is only visible to themselves, and not to the other peers in the network. Hence they can conduct private transactions.

In a supply chain scenario, it could be imagined that there are separate channels between the various companies involved, for example between a fisherman and the shipping company they cooperate with. It would then be possible for participants in the network to register transactions with confidential information, such as how much was paid and for what, that they do not want broadcast to the entire network. By utilizing channels, it becomes possible to keep sensitive data private, which could make a blockchain-based solution more viable to a wider variety of companies.

The downside to using channels, is that they can become a bottleneck on the system. As such, using channels is a trade-off between privacy and performance. It is fully possible to develop supply chain management systems without using channels, but such systems might not appeal to all companies and industries. Offering a system where one can choose the parties between which there should be channels would offer the most flexibility. However, it would also be of interest to investigate the actual performance impact of utilizing channels. As supply chains are not as

time sensitive as other applications might be, it might be possible to offer privacy without it being detrimental to the overall productivity.

7.2.4 Testing

Due to time constraints we were unable to perform tests to the extent we would have liked. Running the simulations takes a significant amount of time; execution time went above ten hours as the number of products rose.

Increasing the number of products from 100 to 1,000 did not affect the performance of the simulation to a significant degree. However, when the number increased to 5,000, there was a notable increase in the average execution time of every event we measured. It would therefore be interesting to run the tests with even more products.

For the research in this thesis, only one test was conducted for each configuration. Consequently, we have no control for whether the data generated were outliers, or representative for an average execution. By carrying out a number of tests for each configuration, it would be possible to generate a varied data set, which could then be subjected to statistical analyses to get a more detailed view. Data generated from a statistical analyses would also be more trustworthy, as potential outliers in performance would ideally be accounted for.

7.2.5 Other Alternatives

For the research in this thesis, we ended up utilizing Hyperledger Fabric, but there are other alternatives available. Even under the Hyperledger umbrella, there is a selection of options, for example the Sawtooth project that we looked at briefly. While Sawtooth is a well-made piece of software, it is at the time of writing too lacking in practical information to be used without significant time investments.

It is also quite possible that there are alternatives completely unrelated to the Hyperledger umbrella available. Such alternatives could potentially be both easier to use and offer better performance, but have not been investigated during the work on this thesis. Due to time constraints, a decision had to be made regarding how to develop the simulation, and which technologies to use. In the future, it would be beneficial to look properly into existing options, and examining their benefits and drawbacks.

7.2.6 Internet of Things

A central argument for many blockchain-based supply chain management systems, is the possibility of offering continuous tracking of a shipment through the use of small IoT units with a selection of sensors [53][5][29]. By utilizing such sensors, it is possible to reduce losses and spoilage of goods during transportation, which can save both money and the environment.

However, there is a lack of research looking into the specifics of the sensors. If the sensors are single use, they are in themselves a pollutant. If they can be used multiple times, it is still necessary to transport them back to the producer, in order to attach them to new shipments. It would be of interest to know more about the consequences of utilizing such technologies, and whether the climate cost of using them outweighs the savings. In their paper on using blockchain technology to

improve the seafood supply chain, WWF presents the economic costs of implementation [4]. However, the study looks at a specific use case, and cannot necessarily be generalized.

7.2.7 Summary

In conclusion, we have looked at blockchain-based supply chains management systems, specifically their scalability and how they relate to sustainable industries. Our research suggests that such systems perform well as the supply chain grows, but it is necessary to perform tests at even larger scales. The availability, technical requirements and performance of the supply chain management system used in our research points towards similar systems being viable for use in sustainable industries. However, the complexity and relative immaturity of the technologies are hindrances that will be necessary to overcome. Furthermore, certain aspects of blockchain-based supply chains lack satisfactory research, such as the environmental impact of using digital sensor tags. Our research points toward blockchain-based supply chain management systems having potential, but further research is needed to improve on existing solutions, and to gather more insight into the potential impact on sustainability.

Appendix A

Source Code

Appendix A includes the full source code of any files we changed in order to use the *Fabcar* example. Including the full source code would take up a huge amount of space, and is not relevant to this thesis.

A.1 Chaincode

```
/*
 * SPDX-License-Identifier: Apache-2.0
 */

'use strict';

const { Contract } = require('fabric-contract-api');

function hasElement(array, value){
  return array.indexOf( value ) != -1;
}

const owners = [
    'A',
    'B',
    'C',
    'D',
    'E',
    'F',
    'G',
    'H',
    'I',
    'J',
    'K',
    'L',
    'M',
    'N',
    'O',
    'P',
    'Q',
    'R',
    'S',
    'T'
];

const producers = [
    'A',
```

```
38      'B',
39      'C',
40      'D',
41      'E',
42      'F',
43      'G',
44      'H',
45      'I',
46      'J',
47      'K',
48      'L'
49  ];
50
51  const shippers = [
52      'M',
53      'N',
54      'O',
55      'P'
56  ];
57
58  const consumers = [
59      'Q',
60      'R',
61      'S',
62      'T'
63  ];
64
65  const species =[
66      'Salmon',
67      'Cod',
68      'Halibut',
69  ];
70
71  var stop = 0;
72
73  class FabCar extends Contract {
74      async initLedger(ctx) {
75          console.info('============= START : Initialize Ledger ===========');
76
77          console.info('============= END : Initialize Ledger ===========');
78      }
79
80      async queryFish(ctx, fishNumber) {
81     const fishAsBytes = await ctx.stub.getState(fishNumber); // get the fish
           from chaincode state
82          if (!fishAsBytes || fishAsBytes.length === 0) {
83              throw new Error(`${fishNumber} does not exist`);
84          }
85          console.log(fishAsBytes.toString());
86          return fishAsBytes.toString();
87      }
88
89
90
91      async createFish(ctx, fishNumber, species, size, owner, destination,
            coordinates) {
92          console.info('============= START : Create Fish ===========');
```

```
93
94      if (!hasElement(producers, owner) || !hasElement(consumers, destination))
            {
95          console.log('Incorrect source or destination');
96          return -1;
97      }
98
99      const fish = {
100             docType: 'fish',
101             species,
102             size,
103             owner,
104         source: owner,
105         destination,
106         coordinates,
107         arrived: '0',
108          };
109
110
111          await ctx.stub.putState(fishNumber,
                Buffer.from(JSON.stringify(fish)));
112          console.info('============= END : Create Fish ===========');
113      }
114
115      async queryAllFish(ctx) {
116          const startKey = 'FISH0';
117          const endKey = 'FISH999999999';
118
119          const iterator = await ctx.stub.getStateByRange(startKey, endKey);
120
121          const allResults = [];
122          while (true) {
123              const res = await iterator.next();
124
125              if (res.value && res.value.value.toString()) {
126                  console.log(res.value.value.toString('utf8'));
127
128                  const Key = res.value.key;
129                  let Record;
130                  try {
131                      Record = JSON.parse(res.value.value.toString('utf8'));
132                  } catch (err) {
133                      console.log(err);
134                      Record = res.value.value.toString('utf8');
135                  }
136
137        allResults.push({ Key, Record });
138              }
139              if (res.done) {
140                  console.log('end of data');
141                  await iterator.close();
142                  console.info(allResults);
143                  return JSON.stringify(allResults);
144              }
145          }
146      }
147
```

```
148
149
150
151     async queryOwned(ctx, owner, includeArrived) {
152    const startKey = 'FISH0';
153         const endKey = 'FISH99999999';
154
155         const iterator = await ctx.stub.getStateByRange(startKey, endKey);
156
157         const allResults = [];
158         while (true) {
159             const res = await iterator.next();
160
161             if (res.value && res.value.value.toString()) {
162                 console.log(res.value.value.toString('utf8'));
163
164                 const Key = res.value.key;
165                 let Record;
166                 try {
167                     Record = JSON.parse(res.value.value.toString('utf8'));
168                 } catch (err) {
169                     console.log(err);
170                     Record = res.value.value.toString('utf8');
171                 }
172       if (owner === Record.owner){
173           // Only include products that have not arrived yet
174           if ('0' === Record.arrived) {
175              allResults.push(Key);
176           }
177       }
178             }
179             if (res.done) {
180                 console.log('end of data');
181                 await iterator.close();
182                 console.info(allResults);
183                 return JSON.stringify(allResults);
184             }
185         }
186     }
187
188
189
190
191     async updateCoordinates(ctx, fishNumber, coordinates) {
192    console.info('============= START : updateCoordinates ===========');
193
194         const fishAsBytes = await ctx.stub.getState(fishNumber); // get the
                 fish from chaincode state
195         if (!fishAsBytes || fishAsBytes.length === 0) {
196             throw new Error(`${fishNumber} does not exist`);
197         }
198         const fish = JSON.parse(fishAsBytes.toString());
199    fish.coordinates = coordinates;
200
201         await ctx.stub.putState(fishNumber,
                 Buffer.from(JSON.stringify(fish)));
202         console.info('============= END : updateCoordinates ===========');
```

```
203     }
204
205
206
207     async changeFishOwner(ctx, fishNumber, newOwner) {
208         console.info('============= START : changeFishOwner ===========');
209
210         const fishAsBytes = await ctx.stub.getState(fishNumber); // get the
                fish from chaincode state
211         if (!fishAsBytes || fishAsBytes.length === 0) {
212             throw new Error(`${fishNumber} does not exist`);
213         }
214         const fish = JSON.parse(fishAsBytes.toString());
215         fish.owner = newOwner;
216
217     if (fish.owner === fish.destination) {
218         fish.arrived = 1;
219     }
220
221         await ctx.stub.putState(fishNumber,
                Buffer.from(JSON.stringify(fish)));
222         console.info('============= END : changeFishOwner ===========');
223     }
224
225
226
227
228     async sendToNext(ctx, fishNumber) {
229       console.info('============= START : sendToNext ===========');
230
231     const fishAsBytes = await ctx.stub.getState(fishNumber); // get the fish
            from chaincode state
232         if (!fishAsBytes || fishAsBytes.length === 0) {
233             throw new Error(`${fishNumber} does not exist`);
234         }
235         const fish = JSON.parse(fishAsBytes.toString());
236
237
238       if (hasElement(producers, fish.owner)) {
239           fish.owner = shippers[0];
240       }
241       else if (hasElement(shippers, fish.owner)) {
242           if (shippers[shippers.length-1] === fish.owner) {
243       fish.owner = fish.destination;
244           }
245           else {
246          fish.owner = shippers[shippers.indexOf(fish.owner)+1];
247           }
248       }
249       else {
250           console.info("Fish already at destination");
251           return -1;
252       }
253
254         await ctx.stub.putState(fishNumber,
                Buffer.from(JSON.stringify(fish)));
255         console.info('============= END : sendToNext ===========');
```

```
256     }
257
258
259     async hasArrived(ctx, fishNumber, owner) {
260       console.info('============= START : hasArrived ===========');
261
262         const fishAsBytes = await ctx.stub.getState(fishNumber); // get the
                fish from chaincode state
263         if (!fishAsBytes || fishAsBytes.length === 0) {
264             throw new Error(`${fishNumber} does not exist`);
265         }
266         const fish = JSON.parse(fishAsBytes.toString());
267
268       if (fish.owner === owner && fish.destination === owner) {
269           fish.arrived = '1';
270       }
271       else {
272           console.log("You are not authorized to do this");
273        return -1;
274       }
275
276         await ctx.stub.putState(fishNumber,
                Buffer.from(JSON.stringify(fish)));
277         console.info('============= END : hasArrived ===========');
278     }
279 }
280
281 module.exports = FabCar;
```

A.2 Register Users

```
1  /*
2   * SPDX-License-Identifier: Apache-2.0
3   */
4
5  'use strict';
6
7  const { FileSystemWallet, Gateway, X509WalletMixin } =
       require('fabric-network');
8  const fs = require('fs');
9  const path = require('path');
10
11 const ccpPath = path.resolve(__dirname, '..', '..', 'basic-network',
       'connection.json');
12 const ccpJSON = fs.readFileSync(ccpPath, 'utf8');
13 const ccp = JSON.parse(ccpJSON);
14
15 async function main() {
16     var number = 1;
17     if (2 < process.argv.length) {
18        number = process.argv[2];
19     }
20     console.log(`Creating ${number} user(s)`);
21
22     // Create a new file system based wallet for managing identities.
```

```
23      const walletPath = path.join(process.cwd(), 'wallet');
24      const wallet = new FileSystemWallet(walletPath);
25      console.log(`Wallet path: ${walletPath}`);
26
27      // Check to see if we've already enrolled the admin user.
28      const adminExists = await wallet.exists('admin');
29      if (!adminExists) {
30     console.log('An identity for the admin user "admin" does not exist in the
            wallet');
31     console.log('Run the enrollAdmin.js application before retrying');
32     process.exit(1);
33      }
34
35      var user = '';
36      var x = 0;
37      try {
38         for (var i = 0; i < number; i++) {
39             x = String(i + 1); // 0-indexing users is weird
40             user = 'user' + x;
41
42              // Check to see if we've already enrolled the user.
43              const userExists = await wallet.exists(user);
44              if (userExists) {
45                console.log(`An identity for the user ${user} already exists in
                      the wallet`);
46                continue;
47              }
48
49              // Create a new gateway for connecting to our peer node.
50              const gateway = new Gateway();
51              await gateway.connect(ccp, { wallet, identity: 'admin',
                    discovery: { enabled: false } });
52
53              // Get the CA client object from the gateway for interacting with
                    the CA.
54              const ca = gateway.getClient().getCertificateAuthority();
55              const adminIdentity = gateway.getCurrentIdentity();
56
57              // Register the user, enroll the user, and import the new
                    identity into the wallet.
58              const secret = await ca.register({ affiliation:
                    'org1.department1', enrollmentID: user, role: 'client' },
                    adminIdentity);
59
60              const enrollment = await ca.enroll({ enrollmentID: user,
                    enrollmentSecret: secret });
61
62              const userIdentity = X509WalletMixin.createIdentity('Org1MSP',
                    enrollment.certificate, enrollment.key.toBytes());
63
64              wallet.import(user, userIdentity);
65
66              console.log(`Successfully registered and enrolled admin user
                    ${user} and imported it into the wallet`);
67
68         }
69      }
```

```
    catch (error) {
        console.error(`Failed to register user ${user}: ${error}`);
        process.exit(1);
    }

    process.exit(0);
}

main();
```

A.3 Peer Node

```
'use stict';

const { FileSystemWallet, Gateway } = require('fabric-network');
const fs = require('fs');
const path = require('path');

const ccpPath = path.resolve(__dirname, '..', '..', 'basic-network',
    'connection.json');
const ccpJSON = fs.readFileSync(ccpPath, 'utf8');
const ccp = JSON.parse(ccpJSON);

var data = require('./data.js');

function parseHrtimeToSeconds(hrtime) {
    var seconds = (hrtime[0] + (hrtime[1] / 1e9)).toFixed(3);
    return seconds;
}

function hasElement(array, value) {
  return array.indexOf( value ) != -1;
}

async function main() {
    if (process.argv.length < 3) {
   console.log("Please include the name of the node and the user number");
   return -1;
    }
    const owner = process.argv[2].toString();
    const user = 'user' + process.argv[3];
    // Extremely bad form, but does the job for now
    if (! hasElement(data.owners, owner)) {
   console.log("Please enter a valid owner");
   return -1;
    }
    const baseNumber = process.argv[3] + '000';

    try {
       // Create a new file system based wallet for managing identities.
        const walletPath = path.join(process.cwd(), 'wallet');
        const wallet = new FileSystemWallet(walletPath);
        console.log(`Wallet path: ${walletPath}`);

        // Check to see if we've already enrolled the user.
```

```
43          const userExists = await wallet.exists(user);
44          if (!userExists) {
45         console.log(‘An identity for the user ${user} does not exist in the
               wallet‘);
46              console.log(’Run the registerUsers.js application before
                   retrying’);
47              return;
48          }
49
50          // Create a new gateway for connecting to our peer node.
51          const gateway = new Gateway();
52          await gateway.connect(ccp, { wallet, identity: user, discovery: {
                enabled: false } });
53
54          // Get the network (channel) our contract is deployed to.
55          const network = await gateway.getNetwork(’mychannel’);
56
57          // Get the contract from the network.
58          const contract = network.getContract(’fabcar’);
59
60      var startTime = 0;
61      var elapsedTime = 0;
62
63      // ===== PRODUCER NODES =====
64      if (hasElement(data.producers, owner)) {
65          for (var i = 0; i < 84; i++) {
66         // Create new product
67         startTime = process.hrtime();
68         const destination =
               data.consumers[Math.floor(Math.random()*data.consumers.length)];
69         const weight = (Math.random() * 5 + 2).toFixed(2); // Generates a
               pseudorandom number [2, 7)
70
71         const x = (Math.random() * 360 - 180).toFixed(2);
72         const y = (Math.random() * 360 - 180).toFixed(2);
73         const coor = ‘(${x}, ${y})‘;
74
75         const spec =
               data.species[Math.floor(Math.random()*data.species.length)];
76         // console.log("Creating fish with ID: " + (baseNumber + i));
77
78         await contract.submitTransaction(’createFish’, ’FISH’ + baseNumber +
               i, spec, ‘${weight}‘, owner, destination, coor);
79         elapsedTime = parseHrtimeToSeconds(process.hrtime(startTime));
80         console.log("produce " + user + " " + elapsedTime);
81
82         // Update list of products owned by this peer
83         const products = (await contract.evaluateTransaction(’queryOwned’,
               owner, ’1’)).toString();
84         const fishNumbers = products.match(/FISH[0-9]*/g || []);
85         if (null == fishNumbers) {
86             continue;
87         }
88         for (var j = 0; j < fishNumbers.length; j++) {
89             // Get next product in list
90             startTime = process.hrtime();
91             const product = fishNumbers[j];
```

```
92          // Update location of product
93          const x = (Math.random() * 360 - 180).toFixed(2);
94          const y = (Math.random() * 360 - 180).toFixed(2);
95          const coor = `(${x}, ${y})`;
96          await contract.submitTransaction('updateCoordinates', product,
                coor);
97          elapsedTime = parseHrtimeToSeconds(process.hrtime(startTime));
98          console.log("update " + user + " " + elapsedTime);
99
100         // Move product to next node
101         startTime = process.hrtime();
102         await contract.submitTransaction('sendToNext', product);
103         elapsedTime = parseHrtimeToSeconds(process.hrtime(startTime));
104         console.log("send " + user + " " + elapsedTime);
105     }
106      }
107 }
108
109 // ===== SHIPPER NODES =====
110 else if (hasElement(data.shippers, owner)) {
111     while (true) {
112     // Update list of products owned by this peer
113     const products = (await contract.evaluateTransaction('queryOwned',
            owner, '0')).toString();
114     const fishNumbers = products.match(/FISH[0-9]*/g || []);
115     if (null == fishNumbers) {
116         continue;
117     }
118
119     var product = null;
120     for (var j = 0; j < fishNumbers.length; j++) {
121         // Get next product in list
122         startTime = process.hrtime();
123         product = fishNumbers[j];
124         console.log(product);
125
126         // Update location of product
127         const x = (Math.random() * 360 - 180).toFixed(2);
128         const y = (Math.random() * 360 - 180).toFixed(2);
129         const coor = `(${x}, ${y})`;
130         await contract.submitTransaction('updateCoordinates', product,
                coor);
131         elapsedTime = parseHrtimeToSeconds(process.hrtime(startTime));
132         console.log("update " + user + " " + elapsedTime);
133
134         // Move product to next node
135         startTime = process.hrtime();
136         await contract.submitTransaction('sendToNext', product);
137         elapsedTime = parseHrtimeToSeconds(process.hrtime(startTime));
138         console.log("send " + user + " " + elapsedTime);
139
140     }
141      }
142 }
143
144 // ===== CONSUMER NODES =====
145 else if (hasElement(data.consumers, owner)) {
```

```
146         while (true) {
147        // Update list of products owned by this peer
148        const products = (await contract.evaluateTransaction('queryOwned',
               owner, '0')).toString();
149        const fishNumbers = products.match(/FISH[0-9]*/g || []);
150        if (null == fishNumbers) {
151            continue;
152        }
153        var product = null;
154        for (var j = 0; j < fishNumbers.length; j++) {
155            // Get next product in list
156            startTime = process.hrtime();
157            product = fishNumbers[j];
158
159            // Register fish as having arrived
160            await contract.submitTransaction('hasArrived', product, owner);
161            elapsedTime = parseHrtimeToSeconds(process.hrtime(startTime));
162            console.log("arrived " + user + " " + elapsedTime);
163
164        }
165         }
166    }
167    else {
168        console.log(data.consumers);
169    }
170
171    result = await contract.evaluateTransaction('queryAllFish');
172    console.log(`Transaction has been evaluated, result is:
           ${result.toString()}`);
173
174    // Disconnect from the gateway.
175         await gateway.disconnect();
176
177     } catch (error) {
178         console.error(`Failed to submit transaction: ${error}`);
179         process.exit(1);
180     }
181
182 }
183
184 main();
```

A.4 Average.sh

```
1  #!/bin/bash
2
3  count=0
4  total=0
5
6  for i in $( grep "produce" userlog* | awk '{ print $3 }' )
7  do
8      total=$(echo $total+$i | bc )
9      ((count++))
10 done
11 echo Produce:
```

```
12 echo "scale=2; $total / $count" | bc
13
14 count=0;
15 total=0;
16
17 for i in $( grep "send" userlog* | awk '{ print $3 }' )
18 do
19      total=$(echo $total+$i | bc )
20      ((count++))
21    done
22 echo Send:
23 echo "scale=2; $total / $count" | bc
24
25 count=0;
26 total=0;
27
28 for i in $( grep "update" userlog* | awk '{ print $3 }' )
29 do
30      total=$(echo $total+$i | bc )
31      ((count++))
32    done
33 echo Update:
34 echo "scale=2; $total / $count" | bc
```

Appendix B

The IPIT Network

This thesis was done as part of the IPIT Network.

B.1 Introduction

The IPIT Network is the International Partnership for Excellent Education and Research in Information Technology [45]. It includes the Norwegian University of Science and Technology (NTNU), Tsinghua University (China), Nanjing University (China) and the University of Michigan (USA). The leader of the IPIT project is Professor Letizia Jaccheri at NTNU.

The project is headed by NTNU, and is motivated by a growing need for IT professionals. The goal is to grow international cooperation and expertise in the field, and thereby help educate future researchers and workers in IT. With this project, top universities in three different countries are cooperating and exchanging both students and researchers, to share knowledge and best practices, and to increase academic cooperation.

The partnership is to last for three years, during which the goal is to achieve the following:

1. Establish sustainable long term strategic research and education partnerships with the international partners to increase synergy and minimize clerical work
2. Increase quality and popularity of IDI (Department of Computer Science) international offers at Master level and increase the number of incoming students
3. Increase the number of outgoing NTNU students with the final goal to reach 40%
4. Stronger connection between research and education – students need to become more aware of research processes by acting as young researchers and to be part of experiments

As part of this, a number of students will be exchanging between the universities, both for study and research purposes. This thesis is written as part of the agreement, and is as such carried through as a co-supervision between NTNU and Tsinghua University.

B.2 Advantages

By being part of the IPIT Network, several advantages are offered. It is a network with top universities from three different countries involved, which means that the professors one can get in touch with are of a high level. The network also offers help

and advice regarding the administrative matters, such as getting the agreement for the exchange in place.

There is a lack of researchers focusing on blockchain technology in Norway. At NTNU, there is a very small number, which limits the availability of supervisors for students wishing to pursue t. However, at Tsinghua University there is a significant amount of research being done on blockchain technologies and their applications. Through the IPIT Network, cooperation with Tsinghua University is made more accessible, and it was possible to find a supervisor there to aid in the research and writing of this master thesis.

By having a co-supervision between two universities with two supervisors, there is also the added advantage of having more help available. As researchers can be tight on time, having two people to ask for advice increases the possibility of getting help quickly.

From a practical point of view, there is also an economic benefit. The IPIT Network offers a scholarship to the students and researchers involved when they go from their home university to one of the partner universities. This allows for greater economic freedom, which lets you focus more on what you are supposed to do, rather than thinking about the financial aspects of going to a foreign country. Not to mention that the scholarship helps pay for the costs associated with the visa, as well as the travel expenses to get to the host university.

B.3 Challenges

There are, however, certain challenges involved when conducting research in an international context. For one, there is a significant overhead involved in the process. This includes everything from acquiring the visa required to stay for a prolonged period of time in a different country, to actually getting settled in a completely new place and getting used to the environment.

The first step along the way is to decide on the agreement in the first place. As the IPIT Network is still new, the agreement for this thesis was one of the first to be put in place. Because of this, finding a professor for co-supervision took some time. Part of the challenge was to decide upon a research topic that fit the interests of both universities. Once the topic was decided, finding a professor went without much trouble. However, there was some uncertainty for a time, leading to doubts about whether the exchange would even happen. The details surrounding the agreement was taken care of by the IPIT administration.

The preliminary work for this thesis was done during the fall of 2018, at NTNU in Trondheim, Norway. This included a literature review and a systematic mapping study. In January 2019, the main author moved from Trondheim in preparation for going to China. After a month, during which the conference paper shown in Appendix C was written, the main author relocated to China.

During the time between the submission of the conference paper and arriving in China, there was a limited amount of work that could be done. It was limited by the fact that all the authors were at very different places, so communication became more challenging, but also because of the preparations needed in order to move. The geographical distances meant that only online communication was possible. Communication was further hindered by technical issues, such as poor Internet conditions and dysfunctional computers.

In the specific context of going from Norway to China, the application process for the visa is quite involved and time-consuming. The application is required to be

handed in at the consulate in Oslo. As the main author is not based in Oslo, this involves traveling. Also, it is required to have an appointment at the consulate in advance, which adds another step of planning.

Getting settled in a foreign country is in itself a challenge. The matter of housing was quite easily solved, as Tsinghua University has a significant number of dormitories, some of which are reserved for international students. To be guaranteed a room, it is required to apply and pay in advance within a 24 hour window on a specific day, but this posed no problems. Upon arrival at campus, it was easy to get registered and get to the assigned room.

B.4 Thoughts For The Future

All in all, the experience with working with the IPIT Project has been positive. Spending time abroad is a very valuable experience for anyone to have, and IPIT offers a way to do this while also contributing to the scientific community, as well as to international cooperation between universities.

Appendix C

Published Paper

In this appendix we present the conference paper produced from the systematic mapping study conducted during the fall of 2018, as a preliminary work for this thesis.

Blockchain and Sustainability: A Systematic Mapping Study

Eirik Harald Lund, Letizia Jaccheri, Jingyue Li, Orges Cico
Department of Computer Science
Norwegian University of Science and Technology (NTNU)
Trondheim, Norway
eirikharald@hotmail.com

Xiaoying Bai
Department of Computer Science
Tsinghua University
Beijing, China
baixy@mail.tsinghua.edu.cn

***Abstract*—Sustainability is a topic of increasing interest. The United Nations has released a list of 17 goals for sustainable development for the global community. Blockchain is a recent technological innovation that shows great promise in changing industries. In this paper, we look specifically at smart grids and supply chain management systems as areas where sustainable technological innovation can happen. To identify software engineering aspects of blockchain in smart grids and supply chain management, we start upon online libraries focusing on engineering and information technology, and we opted for the methodology of systematic mapping studies in software engineering. The search strategy identified 535 papers, of which 60 were identified as main studies for our mapping. To the best of the authors' knowledge, no previous similar studies exist. Results of the study show that the research connecting blockchain technology to smart grids and supply chain management systems is still young. None of the techniques or systems have yet been implemented in a real life setting. As such, more work has to be done before we can look at the actual implications of putting such technologies into use. Software engineering practices could prove to be very useful in the process of development. We propose that future studies can focus on bringing the technologies closer to real life implementations, as well as how to involve the end users in the development of the blockchain-based systems.**

***Index Terms*—blockchain, sustainability, green energy, supply chain management, smart grids, P2P energy trading, UN goals**

I. Introduction

With a growing focus on climate change and other environmental issue, there is also an increasing focus on sustainability. The United Nations (UN) has defined a list of goals for sustainable development, including issues ranging from food safety to green energy [17]. Blockchain technology can be a useful tool when trying to address certain issues with sustainable development.

In this paper, we conduct a systematic mapping study to discover how blockchain technology relates to sustainability, as defined by the UN goals. We defined the research questions 1) how is blockchain technology related to sustainability? 2) how can blockchain technology be used to develop sustainable technology?

Our research shows that there is an increasing amount of research into blockchain solutions that can be used to address sustainability. However, the solutions are still at relatively early stages of development. Nevertheless, the literature shows a careful optimism with regards to the possibilities of utilizing blockchain technology in the relevant fields.

The rest of the paper is organized as follows. Section II introduces the background and motivation of the study. Section III describes the methodology used. Section IV presents our findings. Discussion and conclusions are presented in Section V and Section VI respectively.

II. Background

The UN goals for sustainable development [17] [18] define 17 goals that the global community should work towards in order to achieve a sustainable future. Goals 2, 7 and 13 are *zero hunger*, *affordable and clean energy*, and *climate action* respectively, which are of particular interest to us. Goal 2 *zero hunger* claims that food waste and loss is a major issue, and part of that issue comes from the food items being spoiled during transportation or upon arrival. The food items can become spoiled because of inadequate storage facilities along the way, but it is difficult to know when or where that happened [3]. For goal 7 affordable and clean energy, it is imperative to make green energy production, e.g. solar panels, more widespread to reduce pollution from fossil fuel power generation methods. The more general goal 13 climate action calls for action to combat global climate change. One of the methods to do so that they point out is to use technological measures in a widespread manner, to reduce emissions where it is possible.

The second UN report is called the World Atlas of Illicit Trade [18] and describes how illegal goods are procured and sent around the globe, generating illegal revenue. In the Atlas, environmental crime is identified as the most profitable, as well as lowest risk, illegal operation, surpassing even drug trade in profitability. Environmental crime includes activities such as logging in protected forest areas such as in the Amazon. Part of the problem is the lack of control of where traded goods come from.

We identified blockchain technology as a possible candidate for making local, green energy production more widespread and for improving supply chain management. Blockchain is a distributed ledger technology, originally proposed as the backbone for the cryptocurrency Bitcoin, introduced in 2008

and launched in 2009. The blockchain technology could drastically change several industries and shows potential in several fields [19]. Taking a cursory look at existing literature, we identify the blockchain technology as a viable option [4] [16]. In this paper, we look at what has been done with regards to blockchain technology and implementations of smart grids and supply chain management systems, and how the literature has developed over time. We are particularly interested in software engineering related issues there.

However, blockchain technology is not without its drawbacks. Blockchain-based networks such as the Bitcoin network and the Ethereum network consume enormous amounts of energy [20] [12]. While the two networks themselves provide services that do not directly contribute to emissions, their footprint is huge.

III. Research Design

We opted for the methodology of systematic mapping studies in software engineering described by Petersen et al. [1] because it describes a systematic way to get an overview of state of the art, which is the primary goal of this paper.

A. Research Questions

TABLE I: Research questions.

Research Question	Explanation
1: How is blockchain technology related to sustainability?	The blockchain technology is being used for a variety of purposes. Some of the use cases might have a net positive effect on climate change, whereas others could cause significant harm.
2: How can blockchain technology be used to develop sustainable technology?	In which direction should further research and development of blockchain technology be headed?

The goal of this paper is to get an overview of what already exists in the literature, as well as see what is lacking and can be done in the future with regards to sustainable technology. So, two research questions are formulated, as shown in Table I. RQ1 relates to connecting the topics of blockchain technology and sustainability, and RQ2 looks at how the blockchain technology can be implemented to achieve sustainability.

B. Screening of Papers

An important step in searching for papers is to decide upon which libraries to use. Blockchain technology is primarily an IT innovation, with physical engineering, financial aspects and other fields coming into the picture after a programming solution has been created. As such, we consider software engineering to be the most relevant here. Taking into consideration that we are interested in the software engineering aspects of smart grids and supply chain management, rather than the economic or purely physical engineering, we decided upon online libraries focusing on engineering and information technology. We searched the libraries IEEE Xplore, ACM Digital Library and DBLP Computer Science bibliography and decided to use the sources available to our university. The libraries were decided upon by looking at possible candidates and doing cursory searches to scan for relevant results.

Initial test searches using the terms ”blockchain AND sustainability” and ”blockchain AND (sustainability OR sustainable)” yielded a very wide spectrum of results. To get more specific results, we decided to use more specific terms. We wanted to look more into certain use cases for blockchain technology, namely smart grids and supply chains, as we believe those to have the potential to improve the sustainability in industries. We wanted to look into green energy production and how to verify the green or sustainable origin of a product, which leads to the inclusion of ”green energy” and ”green certificate. The term ”climate change” was included to look for papers specifically looking into that specific topic, as we would like to look into how these may or may not be related. The final search strings can be seen in Table II. As blockchain technology is still fairly novel, rapid changes can be expected in the literature, so we only looked at papers from the last five years, i.e. from 2014 to 2018.

TABLE II: Search strings.

Search strings
blockchain AND ”climate change”
(blockchain OR cryptocurrency OR bitcoin) AND (”climate change” OR green OR ”green energy”)
(blockchain OR bitcoin OR cryptocurrency) AND (climate OR green OR ”green certificate”)
blockchain AND ”supply chain”
(blockchain OR bitcoin OR cryptocurrency) AND (climate OR green OR ”green certificate”) AND (”supply chain”)

The searches yielded a total of 535 papers. We used the library software Mendeley to keep track of all the papers. Some of the documents found in the libraries were tables of content for conferences and similar and were automatically excluded by Mendeley simply because they’re not meant to be cited. As such, the working set included 486 papers. To exclude irrelevant papers, we first removed all the duplicates among the papers, leaving 318. To decide upon the remainder, we have to develop inclusion and exclusion criteria. The criteria used for this study are shown in Table III. After the process of screening the papers, 60 were left. The bibliography can be seen here.

C. Groupings

As we did not find any previous mapping studies on blockchain and sustainability, we were left with the task of deciding the groupings from scratch ourselves. We decided on the grouping by following these steps:

1) Choose a sample of the papers
2) Read the titles, abstracts, conclusions and keywords if included

TABLE III: Inclusion and exclusion criteria

Inclusion	**Exclusion**
Studies related to using Blockchain to address sustainability related issues	Study related to the already proven and accepted implementations of blockchain technology, such as cryptocurrencies
Studies related to improving the energy efficiency of Blockchain services	Studies related to addressing problems with Blockchain technology not related to sustainability
Studies looking at the impact of Blockchain technology	Studies related to offering a service through Blockchain that does not address an environment or climate related topic

3) If the paper fits an existing category, put it there. If not, create a new category
4) After finishing the sample papers, do the same with the rest of the papers
5) If there is a too big of an overlap between categories, get rid of those categories and return to step 2

The papers were put into the following categories, as shown in Table IV. The three categories are further divided into their own sub-groupings, which can be seen in tables V, VI, and VII. The research types and contribution types were taken from the paper on systematic mapping studies in software engineering by Petersen et al. work [1].

TABLE IV: Groupings

Knowledge area	What is the topic of the paper?
Contribution type	What was the output of the paper?
Research type	What kind of research was conducted?

TABLE V: Knowledge areas

Supply chain for visibility	Making supply chains more transparent to the actors involved, to keep track of each step along the way
Supply chain for security	Improving the resilience of supply chains against malicious actors, including counterfeit protection
Supply chain for quality	Improving supply chains to ensure the quality of the end product, reducing spoilage and product loss
P2P energy trading	How to make Peer-to-peer (P2P) energy trading available to the public
Energy trading security	Discussing the security of using smart grids, or P2P energy trading in general
Blockchain energy efficiency	Proposals to improve the energy efficiency of future blockchain implementations

TABLE VI: Contribution types, from Peterson et al. [1]

Contribution	**Description**
Tool	Papers proposing a tool to aid further development
Model/method	Papers introducing new models or methods for addressing a problem
Framework	Papers proposing frameworks for development
Survey	Papers presenting data on what already exists, but do not propose a solution in themselves
Ontology	Papers proposing an ontology for identifying and discussing issues
Testbed	Papers proposing a testbed to aid further development

D. Knowledge areas

After the first look, most of the categories were already as they are now, but a category named ”smart cities” was also included. We excluded this term because it is a very broad term, which makes limiting it only to the topics of interest to us would be too inaccurate. There was originally a category called ”smart grids”, which was renamed ”P2P energy trading”, as the latter was deemed more accurate. Not all of the papers on energy trading between individuals specifically refer to smart grids. However, the category ”smart grid security” was kept, as the cases where security was included did deal with smart grids. Similarly, ”smart grid security” was renamed ”energy trading security”, as the former was found to be too strict. The term ”smart grids” refers to a fairly specific scenario, where there is an off-grid network of automated energy trading. ”P2P energy trading” is more general, and can cover any kind of situation where individuals are trading energy between each other.

Overall the papers fell fairly neatly into the final knowledge areas. A small number of papers could potentially fit into more than one, and one falls slightly outside a strict categorization. The rest were not problematic. No papers trying to solve the energy issues of existing blockchain-based systems were found, which is reflected in the categories.

IV. Results

A. RQ1 How is Blockchain technology related to sustainability?

In this paper, we limited the scope of the huge topic of blockchain technology and sustainability, to the fields of smart grids and supply chains, while also looking at the energy expenditure of blockchain implementations. Smart grids were

TABLE VII: Research types, from Peterson et al. [1]

Research type	**Description**
Validation research	Techniques investigated are novel and have not yet been implemented in practice. Techniques used are for example experiments, i.e., work done in the lab.
Evaluation research	Techniques are implemented in practice and an evaluation of the technique is conducted. That means, it is shown how the technique is implemented in practice (solution implementation) and what are the consequences of the implementation in terms of benefits and drawbacks (implementation evaluation). This also includes identifying problems in the industry.
Solution proposal	A solution for a problem is proposed, the solution can be either novel or a significant extension of an existing technique. The potential benefits and the applicability of the solution are shown by a small example or a good line of argumentation.
Philosophical paper	These papers sketch a new way of looking at existing things by structuring the field in the form of a taxonomy or conceptual framework.
Opinion paper	These papers express the personal opinion of somebody whether a certain technique is good or bad, or how things should be done. They do not rely on related work and research methodologies.
Experience paper	Experience papers explain on what and how something has been done in practice. It has to be the personal experience of the author.

chosen, as we believe they can be helpful in working towards the UN goals for sustainability, and in creating the ”smart cities” of the future. We also included supply chains, to see if they can be used for proofs of origin and traceability, as we believe it can be an option to reduce environmental crime and food waste, as well as other use cases. Figure 1 shows that there has been a significant increase in publications, which looks into blockchain technology used in conjunction with smart grids or supply chain management systems, over the last two years. Figure 2 shows that validation research and solution proposals appear earlier than other types of studies. Validation research looks into proposing new possibilities for addressing a given problem. The papers try to prove the validity or viability of some new technique, without actually implementing it. Solution proposals take it one step further, trying to develop such techniques to a point where they would be suitable for implementation. The increased interest here can be seen as a growing initiative for solving these problems that have caught the public eye.

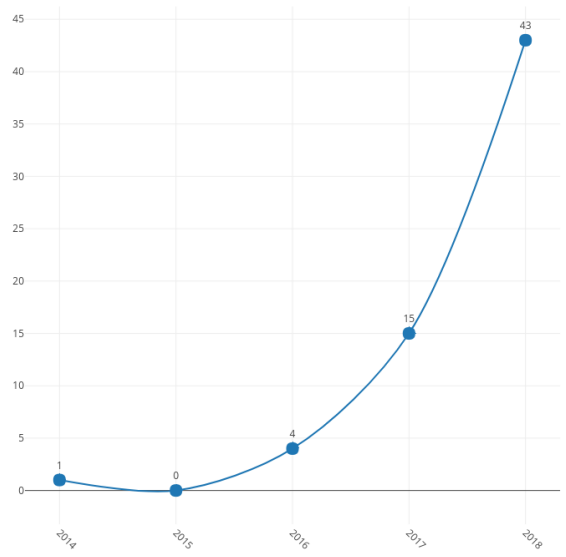

Fig. 1. Number of publications by publication year.

Figure 2 also shows the lack of evaluation research, which implies that researchers have not gotten so far as to implementing the proposed techniques in real life test scenarios. Figure 1 shows that there is very little research into the fields of blockchain and sustainability before 2018. It would be overly optimistic to expect full solutions to problems as big as these in such a short period of time. Some philosophical papers show up from 2018. It might be that there was a conventional understanding of how to utilize blockchain technology for sustainability, which went unquestioned. However, with growing interest comes growing criticism, which could explain why researchers are now trying to propose different ways of understanding and addressing the issues. The same arguments apply to opinion papers. There are almost no experience papers, hinting at just how new the field is and implying that not much has actually been tested in the real world.

B. RQ2 How can Blockchain technology be used to develop sustainable technology?

The number of solution proposals, validation papers and evaluation research makes it clear that there is a potential for utilizing blockchain technology in sustainable technology. Not all the research papers we discovered deal directly with sustainability. In fact, quite a few do not mention sustainability at all, some focusing instead on economic viability. However, economic viability is indeed an important factor for technology adoption. Technological innovation that costs more money than it saves will have a hard time becoming widespread.

As can be seen in Figure 4, we found many papers on P2P energy trading and security in smart grids. In all the papers, blockchain technology plays a central role. Hence, it

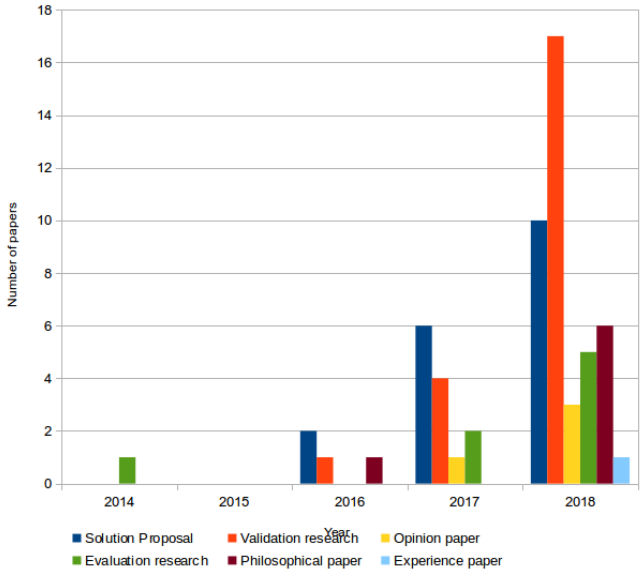

Fig. 2. Research type as frequency per year.

is obvious that researchers believe it is possible to impler these systems. The papers do not discuss whether it is doa but rather the specific details of implementation. We see s validation and evaluation research, a fair amount of solu proposals, and little else. Continuing the argument above, lack of real life tests can be seen as a sign that researchers still figuring out the best practices for how to implement smart grid systems.

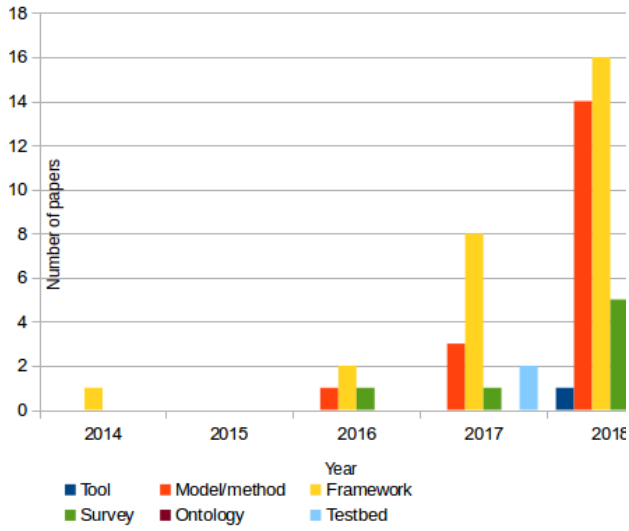

Fig. 3. Contribution type as frequency per year.

Looking at the contribution types in Figure 3, we can see that there are many framework proposals as well as models/methods. There is only one paper contributing a tool for further development. So researchers have proposed many available variations of implementations, but little on how to use these in real life. The two testbeds might prove useful in achieving that. As research progresses and the field matures, we will probably see more tools and testbeds, that will make further developments easier. In smart grids, ensuring consensus on how to implement grids might be crucial. As we are here dealing with infrastructure, compatible solutions can be important. If the solutions are not compatible, each grid will become a separate island, making further development, and perhaps connections over longer distances, much more difficult. Lombardi et al. claim that blockchain-based smart grids will have limited effects on the industry [16]. Their research shows that certain parts of the industry can be improved through the use of the new technology, such as reducing costs and simplifying transactions, but it will not be disruptive. Chitchyan and Murkin looked at more of the energy sector as a whole [11]. They ”remain cautiously optimistic”, but similarly do not expect a complete revolution.

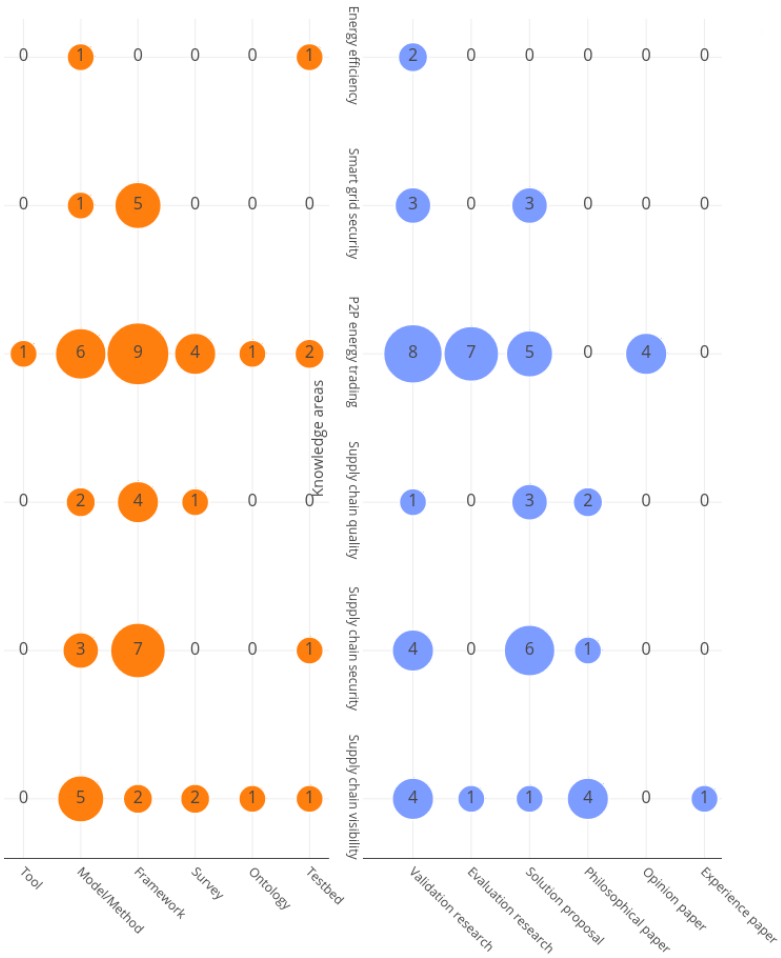

Fig. 4. Publication distribution

As with the papers on energy trading, there are many frameworks and models/methods. Also similarly, it seems that the current issue is to decide on how to go about the real life implementation. Many of the papers look at specific industries

or issues. There is as of yet not much focus on agreeing upon a generalized solution. Some companies might want their own specialized supply chain, but as with smart grids, such a system might prove to be problematic when trying to expand.

The papers on supply chains that we found mostly follow the same pattern as the ones on smart grids. However, a key difference is the lack of evaluation research, as witnessed in Figure 4. Evaluation research has to do with verifying that proposed solutions work. The lack of such research hints that there is still a lot to be done on the use of blockchain technology in supply chains. As explained in the World Atlas of Illicit Flows, environmental crime is carried through in part through "green-washing" products [18]. That is, illegal resources are harvested and then mixed with legit products. If there is no way to tell a legitimate product from a fake, then it is also impossible to know its origin. Proof of origin or Guarantee of origin has been an emphasis on certain supply chain implementations [6][4][5]. Technologies for proof of origin have a wide variety of use cases, several of them possibly benefiting the environment. It should be clear that the possibilities for using blockchain for supply chain management are there. Many researchers have looked at the issue, from various angles. Hence, the question is not whether it is possible, but rather if it is worth it. Jabbar and Bjrn highlight problems with integrating such a system in an already existing supply chain, but end on an optimistic note [14]. The questions are what, then, and what the economic advantages are? As of now, to the best of the authors' knowledge, no supply chains using blockchain have actually been put into use. Any data on possible savings or profitability in general will, therefore, be based on experiments and simulations. The literature seems to be cautiously positive with regards to the potential of blockchain-based supply chains, so it would in fact seem possible to address sustainability issues using blockchain technology. However, it might prove difficult to convince companies to actually put existing proposals into use [9].

V. Discussion

A. Smart grids

Smart and micro grids are becoming popular for a variety of use cases. Examples range from setting up off-grid solutions for rural areas to prosumers selling excess, green energy to their neighbors to replace main-grid energy coming from fossil sources. Two common problems that need to be dealt with are how to incentivize prosumers to produce an excess to sell to others, and how to guarantee the legitimacy of said transaction.

NRGcoin is a proposal by Mihaylov et al., where they set up a virtual market for energy trading in smart grids, with its own cryptocurrency [2]. The currency is generated by supplying energy to the network. Users can also buy energy from others, using the same currency. It would also be possible to exchange the currency for fiat currencies, so one would not be required to sell energy in order to obtain it. Hence, the idea is for participants to balance production and consumption out of self-interest. Mihaylov et al. identified two key issues with the methods found in the literature [2]. As trading is usually done a day in advance, it requires the user 1) predicting the supply and demand ahead of time, and as a result that the user has 2) good knowledge of finance and economy, in order to maximize profits. Their proposal to amend these involves automating a lot of the activity to have it occur with a much higher frequency. They operate with sub-stations for streets, with each one updating every 15 minutes. The prosumers can also inject energy directly into the grid at any time; they do not have to use batteries for storage or hope for net consumption to be at a high at the time. By making buying and selling easier, it becomes possible to adjust prices on the fly, reacting to the market as it changes with highs and lows in consumption. Thus, predicting the expenditure of tomorrow is removed and predicting prices long ahead of time is not needed. Buyers and sellers can place bids with various settings, such as whether to wait if there is no immediately available match, or whether they are willing to accept a different price than they have stated.

B. Supply chain

When discussing the use of blockchain technology in supply chains, researchers have covered fields such as pharmaceuticals and farming, using a variety of different implementations [8] [7]. The goal is to improve the quality of the received product and to counteract malicious actors who might want to replace a genuine article with a fake one. Another benefit, as well as important feature, is supply chain visibility for all parties involved.

An important part of blockchain driven supply chains is that they have to be easy to use. If not, the user experience will be negative, leading to an unwillingness to adopt the system. The usability issue comes in addition to the general unwillingness often found in people to adopt new technologies. How to mitigate this unwillingness was studied by Jabbar and Bjrn [14]. They looked at how to introduce blockchain technology to the shipping domain. They describe the domain as being resistant to new technologies, as the existing infrastructure has been built up over a long time. They bring up two very interesting points. First is the term "infrastructural grind", with which they refer to what occurs when two information infrastructures are converging. They claim the two will rub off on each other, effectively changing each little by little until they fit. The second is the claim that it is necessary to understand the socio-technical infrastructure. In other words, it is not enough to simply offer an industry what you believe is a better solution, you also have to understand how it would fit in with the potential users.

While there has been substantial research looking into how to apply blockchain technology in order to improve conditions for companies, not much has been done in the way of providing information to consumers. Bettn-Daz et al. propose a methodology for developing supply chain traceability systems, while also keeping the end customer in mind [10]. Even so, it is only briefly mentioned, and the paper features little in the ways of how to actually present the information to a customer.

An issue with developing such systems to collect and display data is that current proposals of supply chains are quite proprietary. That is to say, if companies were to adopt the solutions from current literature, they would likely each have their own systems. Similarly, they would also have to develop their own systems for displaying the information. The wide variety we are seeing is a recognizable pattern of innovation; new technology is introduced, followed by a multitude of attempts at utilizing it. This leads to wildly varying solutions, followed by a slow merging, as certain standards are recognized as superior in some way.

Not many researchers have looked directly at the challenge of environmental crimes, but we believe it is possible to use current proposals found in the literature for this purpose. Supply chain solutions are often aimed towards specific industries, but the underlying methods are more generic. We believe it could be possible to apply existing supply chain management methods in the field of environmental crime. Research points to providing visibility to every party involved and avoiding counterfeit products. These relate directly to the issues of malicious actors mixing illicitly procured goods with genuine products. An issue that has been highlighted as particularly challenging is bridging the gap between the virtual blockchain and the physical products [15][9]. While much research has gone into the virtual part, the part regarding how to register and track a physical product has been largely forgotten, which might be due to the difficulty of solving the problem. There are several papers trying to ensure the validity of a container or similar, for example by using RFID tags or lacquer stamps [3][13]. However, while such solutions might offer a unique identifier that is challenging to copy, they do nothing to keep a malicious actor from tampering with the contents of the container.

In the end, there are definitely promising use cases for blockchain technology in supply chain management. However, the lack of standardization makes it both risky and costly to make the change from conventional solutions. Additionally, there is as of now simply no good way to solve the issue of physical tampering with products.

C. Software engineering aspects

We followed the systematic mapping study approach for software engineering described by Petersen et al. [1]. However, we did not find studies focused on the software engineering aspects of developing a blockchain-based system in smart grid and supply chain domain. There are several possible software engineering challenges related to creating and implementing blockchain-based systems, for example:

- How to involve users in defining and eliciting software requirements?
- How to create a flexible architecture that is easy to change and evolve?
- How to verify the functional and non-functional aspects of blockchain-based systems?
- What should the empirical evaluation methods and criteria be for blockchain-based systems?

Figure 1, 2, and 3 show that most studies on the blockchain-based systems for sustainable development are in the early stages. Thus, introducing systematic and rigorous software engineering practices to developing such systems is still a gap to be filled in.

VI. Conclusion and future work

We conducted a systematic mapping study, looking at the relationship between sustainability, as defined through the UN goals, and blockchain technology. More specifically, we focused on the use cases supply chains and smart grids. By doing this, we get an overview of what has been published, and what the researchers have tried to achieve.

We found several studies in the field of smart grids. The different papers often emphasize different aspects, some, for example, looking at how to incentivize prosumers, i.e. participating parties who produce goods rather than consuming them, while others are more focused on how to make the systems resilient towards attacks and exploits. We have also discovered that there does not exist any kind of standardized methodology or framework for developing supply chain management systems. There are, however, several individual implementations to draw inspiration from and to see what they have in common. The experience of existing implementations will be used when looking into how to gather information from the supply chains. For both the main topics in this study, namely smart grids and supply chains, the literature appears to be cautiously optimistic but doubts any huge changes to industry practice. There is potential to improve the industries, but more definite advantages must be discovered and proven in order to incentivize companies to put the technologies into use.

Utilizing software engineering practices in blockchain-based systems meant for improving sustainability is highly relevant. However, our mapping study did not find studies dedicated to software engineering issues, such as development process, requirement engineering, and quality assurance, for developing blockchain-based systems for smart grid or supply chain. As most solution proposals are published in the last two years, few experience paper or evaluation studies, especially studies using empirical software engineering approaches, have been published. Our future work is to study how software engineering theories and practices can help facilitate development and quality assurance of blockchain-based systems for sustainability development.

Acknowledgment

This work was funded by the Norwegian Research Council under the project IPIT Project Number: 274816.

References

[1] Kai Petersen et al. “Systematic Mapping Studies in Software Engineering”. In: *Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering*. EASE’08. Italy: BCS Learning & Development Ltd., 2008, pp. 68–77.

[2] Mihail Mihaylov et al. “NRGcoin: Virtual currency for trading of renewable energy in smart grids”. In: *International Conference on the European Energy Market, EEM*. IEEE, May 2014, pp. 1–6. ISBN: 9781479960941. DOI: 10.1109/EEM.2014.6861213. arXiv: arXiv:1011.1669v3.

[3] Feng Tian. “An agri-food supply chain traceability system for China based on RFID & blockchain technology”. In: *2016 13th International Conference on Service Systems and Service Management, ICSSSM 2016*. IEEE, June 2016, pp. 1–6. ISBN: 9781509028429. DOI: 10.1109/ICSSSM.2016.7538424.

[4] J. Alejandro F. Castellanos, Debora Coll-Mayor, and José Antonio Notholt. “Cryptocurrency as guarantees of origin: Simulating a green certificate market with the Ethereum Blockchain”. In: *2017 5th IEEE International Conference on Smart Energy Grid Engineering, SEGE 2017* (2017), pp. 367–372. ISSN: 1932-6203. DOI: 10.1109/SEGE.2017.8052827.

[5] F. Imbault et al. “The green blockchain: Managing decentralized energy production and consumption”. In: *Conference Proceedings - 2017 17th IEEE International Conference on Environment and Electrical Engineering and 2017 1st IEEE Industrial and Commercial Power Systems Europe, EEEIC / I and CPS Europe 2017* (2017). ISSN: 0014-2956. DOI: 10.1109/EEEIC.2017.7977613.

[6] Qinghua Lu and Xiwei Xu. “Adaptable Blockchain-Based Systems: A Case Study for Product Traceability”. In: *IEEE Software* 34.6 (Nov. 2017), pp. 21–27. ISSN: 07407459. DOI: 10.1109/MS.2017.4121227.

[7] Daniel Tse et al. “Blockchain application in food supply information security”. In: *2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*. IEEE, Dec. 2017, pp. 1357–1361. ISBN: 978-1-5386-0948-4. DOI: 10.1109/IEEM.2017.8290114.

[8] Naif Alzahrani and Nirupama Bulusu. “Block-Supply Chain”. In: *Proceedings of the 1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems - CryBlock'18*. CryBlock'18. New York, NY, USA: ACM, 2018, pp. 30–35. ISBN: 9781450358385. DOI: 10.1145/3211933.3211939.

[9] Giulia Baruffaldi. “Chains in Chains - Logic and Challenges of Blockchains in Supply Chains.” In: (2018).

[10] Rafael Bettín-Díaz, Alix E. Rojas, and Camilo Mejía-Moncayo. “Methodological approach to the definition of a blockchain system for the food industry supply chain traceability”. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 10961 LNCS. 2018, pp. 19–33. ISBN: 9783319951645. DOI: 10.1007/978-3-319-95165-2_2.

[11] Ruzanna Chitchyan and Jordan Murkin. “Review of Blockchain Technology and its Expectations: Case of the Energy Sector”. In: *CoRR* abs/1803.0 (2018). DOI: arXiv:1803.03567v1. arXiv: 1803.03567.

[12] Digiconomist. *Ethereum Energy Consumption Index (beta)*. https://digiconomist.net/ethereum-energy-consumption. Accessed: 2018-10-31. Oct. 2018.

[13] Thomas Hepp et al. “Securing Physical Assets on the Blockchain”. In: *Proceedings of the 1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems - CryBlock'18* (2018), pp. 60–65. DOI: 10.1145/3211933.3211944.

[14] Karim Jabbar and Pernille Bjørn. “Infrastructural Grind”. In: *Proceedings of the 2018 ACM Conference on Supporting Groupwork - GROUP '18*. GROUP '18. New York, NY, USA: ACM, 2018, pp. 297–308. ISBN: 9781450355629. DOI: 10.1145/3148330.3148345.

[15] Nir Kshetri. “1 Blockchain's roles in meeting key supply chain management objectives”. In: *International Journal of Information Management* 39 (Apr. 2018), pp. 80–89. ISSN: 02684012. DOI: 10.1016/j.ijinfomgt.2017.12.005.

[16] F. Lombardi et al. “A Blockchain-based Infrastructure for Reliable and Cost-effective IoT-aided Smart Grids”. In: *Living in the Internet of Things: Cybersecurity of the IoT - 2018*. Institution of Engineering and Technology, 2018, 42 (6 pp.)–42 (6 pp.) ISBN: 978-1-78561-843-7. DOI: 10.1049/cp.2018.0042.

[17] United Nations. *Goal 13: Take urgent action to combat climate change and its impacts*. https://www.un.org/sustainabledevelopment/climate-change-2/. Accessed: 2018-11-26. Nov. 2018.

[18] C. Nellemann et al. *World Atlas of Illicit Flows*. Sept. 2018.

[19] Don Tapscott and Alex Tapscott. *Realizing the Potential of Blockchain*. http://www3.weforum.org/docs/WEF_Realizing_Potential_Blockchain.pdf. Accessed: 2018-11-15. June 2018.

[20] Alex de Vries. “Bitcoin's Growing Energy Problem”. In: *Joule* 2.5 (2018), pp. 801–805. ISSN: 2542-4351. DOI: https://doi.org/10.1016/j.joule.2018.04.016.

Bibliography

[1] Naif Alzahrani and Nirupama Bulusu. "Block-Supply Chain". In: *Proceedings of the 1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems - CryBlock'18*. CryBlock'18. New York, NY, USA: ACM, 2018, pp. 30–35. ISBN: 9781450358385. DOI: 10.1145/3211933.3211939.

[2] Diego Bascans. *Setup Hyperledger Fabric in multiple physical machines*. https://medium.com/1950labs/setup-hyperledger-fabric-in-multiple-physical-machines-d8f3710ed9b4. Accessed: 2019-05-06. Aug. 2018.

[3] Rafael Bettín-Díaz, Alix E. Rojas, and Camilo Mejía-Moncayo. "Methodological approach to the definition of a blockchain system for the food industry supply chain traceability". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 10961 LNCS. 2018, pp. 19–33. ISBN: 9783319951645. DOI: 10.1007/978-3-319-95165-2_2.

[4] Cook Bubba. "Blockchain: Transforming the Seafood Supply Chain". In: (2018), p. 41.

[5] Miguel Pincheira Caro et al. "Blockchain-based traceability in Agri-Food supply chain management: A practical implementation". In: *2018 IoT Vertical and Topical Summit on Agriculture - Tuscany, IOT Tuscany 2018*. IEEE, May 2018, pp. 1–4. ISBN: 9781538669303. DOI: 10.1109/IOT-TUSCANY.2018.8373021.

[6] J. Alejandro F. Castellanos, Debora Coll-Mayor, and José Antonio Notholt. "Cryptocurrency as guarantees of origin: Simulating a green certificate market with the Ethereum Blockchain". In: *2017 5th IEEE International Conference on Smart Energy Grid Engineering, SEGE 2017* (2017), pp. 367–372. ISSN: 1932-6203. DOI: 10.1109/SEGE.2017.8052827.

[7] Digiconomist. *Ethereum Energy Consumption Index (beta)*. https://digiconomist.net/ethereum-energy-consumption. Accessed: 2019-06-04. June 2019.

[8] Future of Fish. "Future of Fish. Making Sense of Wild Seafood Supply Chains." In: (2015), p. 48.

[9] The Food and Agriculture Organization. *Food Loss and Food Waste*. http://www.fao.org/food-loss-and-food-waste/en/. 2019.

[10] Electronic Frontier Foundation. *Creativity and Innovation*. https://www.eff.org/issues/innovation. 2019.

[11] Ethereum Foundation. *Ethereum*. www.ethereum.org/. Accessed: 2018-06-03. 2019.

[12] Ethereum Foundation. *Learn*. www.ethereum.org/learn/. Accessed: 2019-06-04.

[13] Ethereum Foundation. *Proof of Stake (PoS)*. https://docs.ethhub.io/ethereum-roadmap/ethereum-2.0/proof-of-stake/. Accessed: 2019-06-04.

[14] Jana Hennig and Monica Jain. "Fish2.0 Improving Transparency in Seafood Supply Chains". In: (June 2017), p. 1.

[15] Thomas Hepp et al. "Securing Physical Assets on the Blockchain". In: *Proceedings of the 1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems - CryBlock'18* (2018), pp. 60–65. DOI: 10.1145/3211933.3211944.

[16] Martin Holland, Josip Stjepandic, and Christopher Nigischer. "Intellectual Property Protection of 3D Print Supply Chain with Blockchain Technology". In: *2018 IEEE International Conference on Engineering, Technology and Innovation, ICE/ITMC 2018 - Proceedings*. IEEE, 2018, pp. 1–8. ISBN: 9781538614693. DOI: 10.1109/ICE.2018.8436315.

[17] Hyperledger. *About Hyperledger*. https://www.hyperledger.org/about. Accessed: 2019-06-27. 2019.

[18] Hyperledger. *Getting Started - Hyperledger Fabric*. https://hyperledger-fabric.readthedocs.io/en/release-1.4/getting_started.html. Accessed: 2019-04-06. Apr. 2019.

[19] Hyperledger. *Hyperledger Fabric*. https://www.hyperledger.org/projects/fabric/. Accessed: 2019-06-27. 2019.

[20] Hyperledger. *Hyperledger Grid*. https://www.hyperledger.org/projects/grid/. Accessed: 2019-06-27. 2019.

[21] Hyperledger. *Hyperledger Sawtooth*. https://www.hyperledger.org/projects/sawtooth/. Accessed: 2019-06-27. 2019.

[22] Hyperledger. *Sawtooth Supply Chain*. https://github.com/hyperledger/sawtooth-supply-chain. Accessed: 2019-06-27. 2019.

[23] IBM. *blockchainbean2*. https://github.com/IBM/blockchainbean2. Accessed: 2019-03-15. Apr. 2019.

[24] IBM. *BlockchainPublicRegulationFabric-Fooi*. https://github.com/IBM/PublicRegulationFabric-Food-IBPV20. 2017.

[25] Open Source Initiative. *Frequently Asked Questions*. https://opensource.org/faq. 2019.

[26] Karim Jabbar and Pernille Bjørn. "Infrastructural Grind". In: *Proceedings of the 2018 ACM Conference on Supporting Groupwork - GROUP '18*. GROUP '18. New York, NY, USA: ACM, 2018, pp. 297–308. ISBN: 9781450355629. DOI: 10.1145/3148330.3148345.

[27] Henry M Kim. "Toward an ontology - driven blockchain design for supply - chain provenance". In: *CoRR* abs/1610.0 (2018), pp. 18–27. DOI: 10.1002/isaf.1424.

[28] Ioannis Kounelis et al. "Fostering consumers' energy market through smart contracts". In: *Energy and Sustainability in Small Developing Economies, ES2DE 2017 - Proceedings*. IEEE, 2017, pp. 1–6. ISBN: 9781538620663. DOI: 10.1109/ES2DE.2017.8015343. URL: http://ieeexplore.ieee.org/document/8015343/.

[29] Nir Kshetri. "1 Blockchain's roles in meeting key supply chain management objectives". In: *International Journal of Information Management* 39 (Apr. 2018), pp. 80–89. ISSN: 02684012. DOI: 10.1016/j.ijinfomgt.2017.12.005.

[30] Berkeley Labs. *About Microgrids*. https://building-microgrid.lbl.gov/about-microgrids. Accessed: 2018-11-15.

[31] Cristian Lazaroiu and Mariacristina Roscia. "Smart district through IoT and blockchain". In: *2017 6th International Conference on Renewable Energy Research and Applications, ICRERA 2017*. Vol. 2017-Janua. IEEE, 2017, pp. 454–461. ISBN: 9781538620953. DOI: 10.1109/DISTRA.2017.8191102.

[32] Shanhong Liu. *Size of the Bitcoin blockchain from 2010 to 2019, by quarter (in megabytes)*. https://www.statista.com/statistics/647523/worldwide-bitcoin-blockchain-size/. Accessed: 2019-06-24. May 2019.

[33] Antonio Magnani, Luca Calderoni, and Paolo Palmieri. "Feather forking as a positive force". In: *Proceedings of the 1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems - CryBlock'18*. CryBlock'18. New York, NY, USA: ACM, 2018, pp. 99–104. ISBN: 9781450358385. DOI: 10.1145/3211933.3211951.

[34] mahima. *Make changes to chaincode*. https://stackoverflow.com/questions/49855723/hyperledger-fabric-chaincode-not-updated. Accessed: 2019-05-06. Apr. 2018.

[35] mattdean1. *Blockchain Supply Chain*. https://github.com/mattdean1/blockchain-supply-chain. Accessed: 2019-06-27. Apr. 2018.

[36] Microsoft. *Microsoft Azure*. https://portal.azure.com/. Accessed: 2019-05-11. May 2019.

[37] Mihail Mihaylov et al. "NRGcoin: Virtual currency for trading of renewable energy in smart grids". In: *International Conference on the European Energy Market, EEM*. IEEE, May 2014, pp. 1–6. ISBN: 9781479960941. DOI: 10.1109/EEM.2014.6861213. arXiv: arXiv:1011.1669v3.

[38] Satoshi Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. https://bitcoin.org/bitcoin.pdf. Accessed: 2018-11-15. 2008.

[39] United Nations. *About the Sustainable Development Goals*. https://www.un.org/sustainabledevelopment/sustainable-development-goals/. Accessed: 2018-11-26. Nov. 2018.

[40] United Nations. *Goal 13: Climate Action*. https://www.un.org/sustainabledevelopment/climate-change/. Accessed: 2018-11-26. Nov. 2018.

[41] United Nations. *Goal 2: Zero Hunger*. https://www.un.org/sustainabledevelopment/hunger/. Accessed: 2018-11-26. Nov. 2018.

[42] United Nations. *Goal 7: Energy*. https://www.un.org/sustainabledevelopment/energy/. Accessed: 2018-11-26. Nov. 2018.

[43] United Nations. *What is the Paris Agreement?* https://unfccc.int/process-and-meetings/the-paris-agreement/what-is-the-paris-agreement. Accessed: 2019-06-07.

[44] C. Nellemann et al. *World Atlas of Illicit Flows*. Sept. 2018.

[45] The IPIT Network. *The IPIT Network*. https://ipit.network. Accessed: 2019-05-19. May 2019.

[46] Quoc Khanh Nguyen. "Blockchain-A Financial Technology for Future Sustainable Development". In: *Proceedings - 3rd International Conference on Green Technology and Sustainable Development, GTSD 2016*. IEEE, Nov. 2016, pp. 51–54. ISBN: 9781509036387. DOI: 10.1109/GTSD.2016.22.

[47] Kai Petersen et al. "Systematic Mapping Studies in Software Engineering". In: *Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering*. EASE'08. Swindon, UK: BCS Learning & Development Ltd., 2008, pp. 68–77. URL: http://dl.acm.org/citation.cfm?id=2227115.2227123.

[48] Project Provenance. *Tuna App*. https://www.provenance.org/tracking-tuna-on-the-blockchain#overview. Accessed: 2019-03-11. July 2016.

[49] Ping Punyakumpol. *The Great Firewall of China: Background*. https://cs.stanford.edu/people/eroberts/cs181/projects/2010-11/FreedomOfInformationChina/background/index.html. 2011.

[50] ReFED. *Rethink Food Waste*. https://www.refed.com/. 2019.

[51] REN21. *Renewables 2018 Global Status Report*. http://www.ren21.net/status-of-renewables/global-status-report/. Accessed: 2018-11-15.

[52] Don Tapscott and Alex Tapscott. *Realizing the Potential of Blockchain*. http://www3.weforum.org/docs/WEF_Realizing_Potential_Blockchain.pdf. Accessed: 2018-11-15. June 2018.

[53] Feng Tian. "A supply chain traceability system for food safety based on HACCP, blockchain & Internet of things". In: *14th International Conference on Services Systems and Services Management, ICSSSM 2017 - Proceedings*. IEEE, June 2017, pp. 1–6. ISBN: 9781509063697. DOI: 10.1109/ICSSSM.2017.7996119. arXiv: arXiv:1011.1669v3.

[54] Feng Tian. "An agri-food supply chain traceability system for China based on RFID & blockchain technology". In: *2016 13th International Conference on Service Systems and Service Management, ICSSSM 2016*. IEEE, June 2016, pp. 1–6. ISBN: 9781509028429. DOI: 10.1109/ICSSSM.2016.7538424.

[55] Alex de Vries. "Bitcoin's Growing Energy Problem". In: *Joule* 2.5 (2018), pp. 801–805. ISSN: 2542-4351. DOI: https://doi.org/10.1016/j.joule.2018.04.016.

[56] Katie Walsh. *Leaving No Food Behind: WWF Examines Post-Harvest Crop Losses and Pathways to Change*. https://www.worldwildlife.org/press-releases/leaving-no-food-behind-wwf-examines-post-harvest-crop-losses-and-pathways-to-change. 2018.

[57] WETSEB. *2nd International Workshop on Emerging Trends in Software Engineering for Blockchain*. http://www.agilegroup.eu/wetseb2019/. 2019.

[58] *What is REST*. https://restfulapi.net. Accessed: 2019-06-27.

[59] Michael White. *Digitizing Global Trade with Maersk and IBM*. https://www.ibm.com/blogs/blockchain/2018/01/digitizing-global-trade-maersk-ibm/. 2018.

[60] Joon Ian Wong and Johnny Simon. *Photos: Inside one of the world's largest bitcoin mines*. https://qz.com/1055126/photos-china-has-one-of-worlds-largest-bitcoin-mines//. Accessed: 2018-06-04. Aug. 2017.

Eirik Harald Lund

Scalability of Blockchain-based Supply Chain Management Systems

NTNU
Norwegian University of
Science and Technology