Supporting ubiquitous and fully decentralized outbound logistics through blockchain technology

Guilherme Cerqueira Gomes

Master's Dissertation

Supervisor: Prof. Pedro Amorim

U.PORTO

FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

Mestrado Integrado em Engenharia e Gestão Industrial

2018-07-02

Abstract

Supply chains are complex, integral parts of almost any industry and generate massive amounts of information. Supply chains with several thousand different organizations are fairly common, and, for efficient operation, there's a significant portion of data that needs to be be visible along the product's journey. This transparency is not the only concern, as governmental authorities require more verifiable ways of proving accountability. That's where already existent technologies like EDI and cloud computing display a few shortcomings.

This thesis sprouts from very early work in analyzing possible blockchain implementations to increase the value proposition from a company that manages supply chains for fashion brands, and a research project at the Institute for Systems and Computer Engineering, Technology and Science, located in Portugal. The main achievable improvements range from reducing manual labour and material costs directly and cutting down global expenditure on supply chains by eliminating unnecessary idle times, reducing the volume of held items at customs due to incorrect information or lack thereof and shortening lead times.

This dissertation attempts to discern the characteristics of existing blockchain frameworks, the architectural challenges that need to be overcome when implementing the technology in a scenario of dynamic business relationships, such as integration with preexisting database systems, the standardization of information and the tools available to accomplish such tasks.

The work was accomplished by revising existing literature on blockchain technology applied to logistics, which is not as vast as financial blockchain applications, to substantiate the use of a specific framework. Additionally, a mapping of the company, its stakeholders and processes was made to better comprehend the data needs of the supply chain. Hyperledger Fabric was chosen for its greater flexibility, higher performance and more comprehensive documentation, when compared to other projects.

The Fabric framework was deconstructed to grasp its components' functionalities and successfully mirror the business reality in the pseudo-decentralized software application. The logic in every step of this dissertation was to maintain the project as agnostic as possible, or in other words, an effort was made to ensure that the developed work didn't solely apply to the company where the case study was developed. This translates into the analysis of data structures and processes that are common to every supply chain, and staying away from internal processes that don't produce shareable data, as well as the creation of an API that facilitates communication between off-chain storage and the blockchain ledger.

The results obtained with a stakeholder were promising and are in tune with what was to be expected, especially when comparing to previous use cases from considerably bigger industry players. On a specific process, potential changes from weeks of manual labour to a few hours of computation were achieved. The findings do support the hypothesis that blockchain can in fact disrupt how information is shared along the supply chain and change the landscape for operations in logistics.

Resumo

As cadeias de abastecimento são partes complexas e fundamentais de quase qualquer indústria e geram quantidades enormes de informação. Não é incomum cadeias de abastecimento contarem com milhares de organizações de vários ramos, e, para um funcionamento operacional eficiente, há uma porção significativa de dados que requer visibilidade ao longo do trajeto do produto. Esta transparência não é a única preocupação, visto que as entidades governamentais requerem métodos mais verificáveis de provar responsabilidade por ações. É neste aspecto que as tecnologias existentes como EDI e *cloud computing* demonstram algumas deficiências.

Esta tese tem origem de trabalho desenvolvido de forma a analisar a viabilidade de implementar *blockchain* para aumentar a proposta de valor de uma empresa que gere cadeias de abastecimento de marcas de moda, e um projecto de investigação do Instituto de Engenharia de Sistemas e Computadores, Tecnologia e Ciência, em Portugal. Os melhoramentos atingíveis vão de reduzir trabalho manual e custos em material de form directa a cortes nas despesas globais que são incorridas nas cadeias de abastecimento ao eliminar tempos de inatividade, reduzir o volume de items detidos nas alfândegas devido a informação incorreta ou falta da mesma e encurtar *lead times*.

Esta dissertação tenta discernir as características das estruturas de *blockchain* existentes, os desafios arquitetónicos que precisam de ser superados ao implementar a tecnologia num cenário de relações comerciais dinâmicas, como por exemplo, a integração com sistemas de bases de dados preexistentes, standardização de dados e ferramentas disponíveis para realizar essas mesmas tarefas.

O trabalho foi realizado através da revisão da literatura existente sobre a tecnologia *blockchain* aplicada à logística, que não é tão vasta como a refernete a aplicações financeiras de blockchain, para substanciar o uso de uma *framework* específica. Adicionalmente, foi feito um mapeamento da empresa, dos *stakeholders* e processos, para melhor compreender as necessidades de dados da cadeia de abastecimento. O Hyperledger Fabric foi escolhido pela sua maior flexibilidade, melhor desempenho e documentação mais abrangente, quando comparado a outros projetos.

O *framework* Fabric foi desconstruído para compreender as funcionalidades dos seus componentes e espelhar com sucesso a realidade empresarial na aplicação de *software* pseudo-descentralizado. A lógica em cada passo desta dissertação foi manter o projeto o mais agnóstico possível, ou por outras palavras, foi feito um esforço para garantir que o trabalho desenvolvido não se aplica apenas à empresa onde foi desenvolvido o caso de estudo. Isso traduz-se numa análise de estruturas de dados e processos que são comuns a todas as cadeias de abastecimento, evitando os processos internos que não produzem dados partilháveis, e ainda a criação de uma API que facilita a comunicação entre o armazenamento *off-chain* e os registos na *blockchain*.

Os resultados obtidos com um *stakeholder* foram promissores e estão em sintonia com o que seria de esperar, especialmente quando comparados com casos de uso anteriores de companhias consideravelmente maiores. Num processo específico, mudanças potenciais de semanas de mão-de-obra para horas de computação foram alcançadas. Os resultados confirmam a hipótese de que a tecnologia *blockchain* pode de facto pertubar positivamente a forma como a informação é partilhada ao longo da cadeia de abastecimento e mudar o cenário das operações logísticas.

Acknowledgements

This dissertation constitutes the final step in my 5 year journey to obtain a Master's degree in Industrial Engineering and Management at FEUP.

I'd like to extend my gratitude to my family for all they have taught me and for their patience in dealing with my personality.

My friends, for shaping me and sharing some of my best and some of my least great moments, thank you.

To all the teachers, from kindergartner to college, you have my respect.

A special thanks to my supervisors at HUUB, INESC TEC and FEUP for the support throughout this dissertation.

"Man is pre-eminently a creative animal, predestined to strive consciously for an object and to engage in engineering—that is, incessantly and eternally to make new roads, wherever they may lead."

Fyodor Dostoevsky

Contents

Acronyms and Symbols

API	Application Programming Interface
AWS	Amazon Web Services
BFT	Byzantine Fault Tolerance
BPMN	Business Process Model Notation
CA	Certificate Authority
CRM	Customer Relationship Management
DB	Database
DFS	Distributed File System
DMS	Distribution Management System
DSC	Decentralized Supply Chain
ELD	Electronic Logging Device
EU	European Union
IoT	Internet-of-Things
OMS	Order Management System
PBFT	Practical Byzantine Fault Tolerance
PoA	Proof-of-Authority
PoET	Proof-of-Elapsed-time
PoS	Proof-of-Stake
PoW	Proof-of-Work
RBFT	Redundant Byzantine Fault Tolerance
SBFT	Simplified Byzantine Fault Tolerance
SDK	Software Development Kit
SKU	Stock Keeping Unit
SLA	Service level agreement
SO	Sales Order
PO	Purchase Order
WMS	Warehouse Management System

List of Figures

List of Tables

Chapter 1

Introduction

Supply chains have become increasingly global and exponentially more complex, with sometimes thousands of stakeholders, spanning multiple geographies and accounting for enumerable data interchanges (Wu et al., 2017) carried out through a myriad of silloed steps from production, to storage, to shipment, to sale (Schrauf and Berttram, 2016). The digitization of supply chains (Digital Supply Chain – DSC) represents a collaborative effort of companies with different incentives to reach the common goal of increasing integration between stakeholders (Korpela et al., 2017) which is key for the success of any supply chain. Visibility, traceability, accountability all benefit from a higher degree of transparency provided by a DSC (Schrauf and Berttram, 2016).

The landscape in supply chain management has, unequivocally, shifted into tech companies that provide the much sought-after digital upper-hand. Since 2012, the world of supply chain and logistics has seen $8.4 billion in tech funding since 2012, with approximately two-fold growth from year to year (Wu, 2016). These investments cover niche startups that focus on drones and croud-sourcing drivers to truck driver networks and self-driving vehicles. However, there's an overarching theme amongst all: integration. Not the integration inherently associated with SCM, "from end user through original suppliers" (Lambert et al., 1998), but the integration of stakeholders as a service, or in simpler terms, offering a single service of managing all the clients' supply chain needs.

Integration, be it vertical or horizontal, is not a modern concept in SCM by any means, and its advantages have been more than researched and empirically proven. The aim of this thesis is to evaluate how blockchain may or may not be a step forward towards fully integrated supply chains, i.e. supply chains with end-to-end visibility and effective information flows, and how it may be a valuable tool for SCM as a service.

Recent advances and research in this area show promising results for blockchain. The original financial scope associated with blockchain, thanks to Bitcoin, has since been experimentally broadened to many other fields like healthcare, politics, and, most pertinent to this dissertation, supply chain. The decentralized and the self-sustaining nature of this technology may eliminate the need for third-parties to centralize data and increase integration costs (Korpela et al., 2017). The blockchain allows for a verifiable ways of transacting information across several stakeholders, information which is crucial for key activities like carrier validation, customs clearance and financial auditing (Apte and Petrovsky, 2016; Korpela et al., 2017; Kshetri, 2018; Blechschmidt and Stöcker, 2017).

1.1 Case Study description

This thesis serves as a case study for a supply chain and logistics partner tech startup. The now 3-year old company focuses on reducing the complexity of managing a supply chain for small to medium textile fashion brands. The main goal is to leverage economies of scale by managing larger volumes of packages that would be impossible for said small brands to produce. HUUB achieves this by gathering and connecting a myriad of stakeholders' information to extract meaningful data. Coincidently, the network resembles the well-disseminated distribution topology paradigm known as spoke-hub, illustrated bellow.

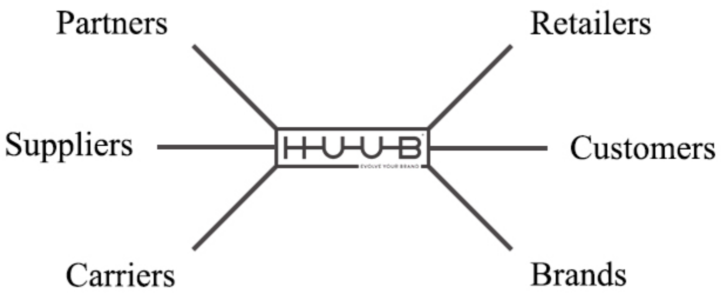


Figure 1.1: HUUB's positioning relative to its stakeholders

The whole process of on-boarding partners and managing supply chains is greatly expedited with HUUB's proprietary system, Spoke, which is accessible through a web platform. This information system supports warehouse (WMS), order fulfillment (OMS) and distribution (DMS) features, that already account for great improvements in coordination of information.

Currently, the company boasts a portfolio of 40+ brands and has served more than 80 countries. Since the deployment of Spoke, it has handled more than 180.000 products, pertaining to 20.000 SKUs and an approximate total of 6.600 shipments, with two logistics centers, one in Maia, Portugal and another in the Netherlands, via a partnership with DAMCO.

1.2 Motivation

No matter which platform is being used, whether it's private blockchain with a focus on business data interchange or a public decentralized database for external audits and customer visibility, blockchains still beg the question: Are blockchains better than a centralized database?

There is no simple answer, because they simply don't offer the same advantages. Blockchains disintermediate the process of sharing data with other parties, in the sense that there's no need to hire and pay employees to maintain the integrity of the stored data, instead, computers and code do that with smart cryptography. They can also be designed to be more robust than regular databases more easily, because once the network has several nodes connected, they automatically sync each other to redundantly store copies of the information. On the other hand, traditional databases offer more confidentiality and faster access, at the expense of centralization of control and human error.

However, as many industry players note, visibility down the supply chain is often hard, even with centralized databases, with delays, sometimes up to days, in the sharing of information. Permissioned blockchains offer the advantages of blockchain with some of the benefits of centralized databases, with pseudo-real time transaction information and the trustless security that is expected from the technology.

1.3 Scope

The scope of this project, as an exploratory dive into the application of the novel technology in business-to-business transactions in logistics, will try to answer one important question: how can the "hype" be filtered into a functioning, well documented, proof-of-concept implementation of blockchain?

The current state of research in the field of blockchain applied to supply chains is filled with opaque use cases from big companies, like IBM with Maersk, or truly well documented studies on possibilities and conundrums associated with the technology. There seems to be a deficit of a combination of both. This thesis aims at understanding the caveats of the technology through literature review and going through all the essential steps of designing and deploying a blockchain application that can improve supply chain operations.

Kshetri (2018) documented a series of use cases, and by adding reports that complement his research, Maersk and Walmart's shipment and food tracking (Popper, Nathaniel and Lohr, Steve, 2017; Higgins, Stan, 2017b), Provenance's efforts in Indonesian tuna safety (Provenance, 2016) and Intel's tuna tracking (Del Castillo, Michael, 2017), AliBaba's food fraud (Tas, Bindi, 2017), Lockheed Martin interest in blockchain for security (Higgins, Stan, 2017a), Chronicled, Modum and Gemalto's tackle of the pharmaceutical industry (Chronicled, 2017; Campbell, Rebecca, 2016; Start-up Ticker, 2017; Mathieson, SA, 2017), Everledger's crusade against counterfeiting (Lechmere, Adam, 2016) and Bext360 coffee farmers awareness (Kolodny, Lora, 2017; Scott, Michael, 2017) can be presented as an elucidative landscape of blockchain implementations in Table 1.1.

1.4 Thesis outline

This dissertation is composed of 6 chapters, being this the end section of the first chapter. Chapter 2 overviews the state of the art in blockchain technology, points out the challenges that can exist

Table 1.1: Comparison of blockchain use cases in several industries by big players and this thesis

	Scope	Dimension	Technology	Public	Private	Research Paper
IBM & Maersk (2017)	Shipping Containers	Customs-to-customs	Hyperledger	-	✓	-
Provenance (2016)	Food	Producer-to-Point-of-sale	Provenance	✓	-	-
Alibaba, AusPost, Blackmores & PwC (2017)	Food	N/D	N/D	✓	-	-
Lockheed Martin & GuardTime Federal (2017)	Security	N/D	N/D	N/D	N/D	-
Chronicled & Linklab (2017)	Pharmaceutical	Producer-to-Point-of-sale	N/D	-	✓	-
Modum & University of Zurich (2017)	Pharmaceutical	Producer-to-Point-of-sale	Modum	-	✓	-
Everledger (2016)	Counterfeiting	Producer-to-Seller	Everledger	-	✓	-
IBM &Walmart (2016)	Food	Producer-to-seller	Hyperledger	N/D	N/D	-
Gemalto (2017)	Pharmaceutical	N/D	N/D	N/D	N/D	-
Intel (2016)	Food	Producer-to-Point-of-sale	Hyperledger	✓	-	-
Bext360 (2017)	Food	Producer-to-Seller	Stellar	✓	-	-
This work (2018)	Fashion	Warehouse-to-customer	Hyperledger	✓	✓	✓

when implementing such a solution and provides an in-depth view of the reasons behind choosing Hyperledger Fabric as the preferred frameworks, and not Ethereum or IOTA, as examples.

Chapter 3 describes the company structure, a relevant topic to understand the different business and commercial interactions they have with their stakeholders. An analysis of the outbound process as-is and its limitations is also presented, as a way of discerning the bottlenecks that derive from the assymetry of information along the supply chain. The central drawback of the current methods is the time spent on validation of documents and communication between stakeholders. The proposed process to-be shortens and simplifies the flow of operations by leveraging blockchain technology as a hub of shareable data.

In Chapter 4 the caveats of Hyperledger Fabric are demystified, such as the configuration of the network, i.e. how different entities are represented digitally and interact with the ledger, how smart contracts deal with data and business logic, and how different entities grant/revoke/gain/lose permissions and are authenticated in the network. As a last point, the use of the Node SDK abstraction for interaction with the ledger is explicated.

Chapter 5 exposes the architecture behind the Web API interface developed to serve as a test for automatic input and output of data directly from stakeholders private off-chain databases and the blockchain. It is also in this chapter that the on-boarding of an outside entity is introduced and how data was gathered to measure possible gains of implementing the technology.

The final chapter, Chapter 6, wraps up this dissertation with an examination of the accomplished work and its shortcomings, with a glance into what might be developed in future research.

Chapter 2

Literature Review

This chapter begins by contextualizing blockchain, with the most proliferated technologies, Bitcoin and Ethereum. A further analysis of available technologies is made to provide a clear view of the possibilities of implementation frameworks to use, but also to highlight novel features and their relevance in logistics and validate the choice of framework. The majority of concerns stressed by other authors, like data uniformization or confidentiality of information in a system that was originally designed to be public and anonymous, are also addressed. Hyperledger Fabric, for its finer fit with the objectives for this thesis is delved into with more detail.

2.1 Blockchain

Picture a database that is distributed through several computers, so that if one computer is shut down or hacked, there's no single point of failure, so the information is still accessible. These databases are called Distributed File Systems, or DFS, and they have been around since 1984, when Sun Microsystems introduced its Network File System (NFS), using UDP/IP Protocols and Ethernet, according to Levy and Silberschatz (1990). Nakamoto (2008) suggests that if we make it so that the data can only be written to this database when the whole network of computers agree, through consensus algorithms, and if every new piece of information is contained in a block that somehow relates to the information that is already stored in the database, we have a chain of timestamped correlated blocks that contain data that was consensually put there: a blockchain.

Blockchain, not to be confused with Bitcoin, even though they are often mentioned alongside each other, is not a product, it's a technology. Some authors, like Iansiti and Lakhani (2017) and Hancock and Vaizey (2016), compare its disruptive possibilities to that of TCP/IP because it essentially allows computers, or in other words nodes, to communicate with each other through a set of rules, designed in part by Nakamoto (2008), like digital signatures (with cryptographic hashes), consensus algorithms (PoW, PoS, PoET, PoA, etc.) and encrypted data, and later optimized by others, as we will see further ahead.

These characteristics allow for a distributed and immutable ledger, that stores data, be it transactions or business-related information, in a tamper-free and, in a way, ownerless fashion, making it trustless, i.e. no need for trust in a third party that maintains and secures the database (Greenspan, 2016).

Bitcoin, first introduced in 2008 with blockchain, is the cryptocurrency that comprises the origin of blockchain's introduction. It is a digital coin, that takes advantage of blockchain technology to execute transactions between peers and record data/transactions in a public distributed ledger. Though Bitcoin might have a future in the world of Fintech (Greenspan, 2015), the problem remained that the business world, and more specifically Supply Chain Management related firms, cannot leverage Bitcoin in a productive way. It was only with the introduction of Ethereum, where Buterin (2014a) re-formulated the concept of "smart contracts", that the corporate world started taking blockchain more seriously (Pilkington, 2015), and areas such as Healthcare, Pharma, SCM, Construction started seeing exponencial growth in research.

2.2 Ethereum

Ethereum, similarly to Bitcoin, is a blockchain network where users can transfer data with a cryptocurrency (Ether), but it adds a major functionality: smart contracts. Simply put, smart contracts are programmable actions that are triggered by an event or series of events within the network (Buterin, 2014a), e.g. payment of goods is triggered when goods are received. Ethereum has multiple client languages (ways of interacting with the network), including Go, widely used, Rust, C++, Python and more, but it uses its own programming language, Solidity, for smart contracts, which might entail a learning curve (Ethereum Community, 2017).

Ethereum has one main advantage over Bitcoin, which is the time and energy it takes to mine a block. Despite both using Proof-of-Work algorithms to reach consensus, Ethereum has a variable built-in difficulty parameter that is constantly updated to make it more or less difficult to mine a new block. Mining is a term that is related to the PoW algorithm and it means that all the computers in the network are looking for a rare hash combination that will let the computer write a queued block to the blockchain. So, in Ethereum, if the network is taking to long to find this solution, then the difficulty is lowered, and vice-versa, in order to achieve an average block-writing time of 12s Buterin (2014b). Notwithstanding, this doesn't make Ethereum perfect for Supply Chain Management decentrilized applications, far from it.

Like Bitcoin, Ethereum is a public ledger, i.e. anybody can access it, which obviously means it should immediately be discarded for business uses where sensitive and private data is involved Weber et al. (2016). Even though some efforts have been and are being made to create permissioned Ethereum-based ledgers (which are blockchains that use the technology behind the main blockchain, e.g. Ethereum, and alter or add features on top of that), like Quorum, HydraChain and ErisDB, the technology behind Ethereum is still oriented at transactions and crypto-economic incentives (Buterin, 2014a), and not so much at business data logging. This is evidenced by the use of a crypto-currency, which is virtually useless for purposes of recording data that is not directly related to a transaction.

Consequently, more blockchains have been developed that have a bigger focus on business intelligence and streamlining process workflows.

2.3 Relevant Blockchain technologies

It is important to note that the following list is not exhaustive, for there is a growing number of blockchain platforms available, that, if explored in their full extent, would heavily expand the scope of this thesis and unnecessarily extend its length.

Accordingly, public/permissionless blockchain platforms have been discarded from the get-go, and many other that are irrelevant to the topic here presented due to their "niche" applications, like identity management, health records, document registries, just to name a few, and, evidently, cryptocurrencies. The ones presented which do not specifically serve the purpose of this thesis have, however, a benchmark role, being among the most successful in their field of application. These have great value, regardless, but are simply not purposefully-made for cross-industry business networks, a requirement in Supply Chain Management.

Corda Corda's novel offer is pluggable consensus by pluggable notaries (there is no blockchain) that provide their guarantees through whatever algorithm they use (Hearn, 2016). It has been contributed to by R3, a consortium of more than 100 financial institutions.

ShipChain Although ShipChain does show potential for benefits in Supply Chain Management, it is designed as a permissioned Ethereum side-chain and more specifically aimed at shipping networks (ShipChain, 2017).

BigChainDB BigChainDB is aimed at digital transaction of assets. It boasts impressively transaction speeds, when compared to the other contemporary blockchains, thanks to its own consensus algorithm (Mcconaghy et al., 2016).

Modum Modum is very regulation-oriented, specifically the pharmaceutical or diamond trade industry regulations and is based on Ethereum. It greatly relies on IoT, e.g. temperature and humidity sensors, for data gathering and the necessary gonvernmental certifications as well as the much needed visibility and data-integrity (Modum.io AG, 2016).

IOTA IOTA is a fee-less and "blockchainless" blockchain platform for IoT and other applications that solves the scaling challenges of previous blockchains with its Tangle underlayer (IOTA Support, 2017). The term "blockchainless" stems from the fact that instead of a chain of blocks related to the previous and next blocks, it acts as a grid of interrelated blocks that has gains in its efficiency the bigger it gets (Popov, 2017).

Quorum Quorum is yet another open-source permissioned Ethereum fork. The objective of Quorum, as JP Morgan Chase (2016) puts it, is to make as little modifications as possible to the Ethereum Go client, to allow for easier synchronization with future versions.

Ripple Ripple (2018) alledges it's system "connects banks, payment providers, digital asset exchanges and corporates via RippleNet to provide one frictionless experience to send money globally". The financial asset oriented blockchain has demonstrable higher transaction speeds than its competitors (Travis, 2017) and a myriad of financial partners that aid it in being one of the most used cryptocurrencies as of the writing of this paper.

Startups To clarify any remaining doubts related to unmentioned blockchain projects, Skuchain (Hyperledger Fabric), Everledger (non-specified Hyperledger), Provenance (proprietary) and others are startups and not excactly blockchain frameworks, i.e. they offer a service, and not a explorable open-source framework. OriginTrail, for instance, funded through token pre-sales, in other words, private sale of cryptocurrency, is not of interest for this thesis as they do not present the necessary access to the underlying technology, as that is not their purpose, even though they do present a solution for Supply Chain Management.

2.4 Hyperledger

Hyperledger is an open-source project started by The Linux Foundation that houses several blockchain platforms, jointly developed with other companies (Hyperledger Architecture Working Group, 2017). The main goal of Hyperledger, contrary to other blockchain projects aimed at specific industries like Fintech (Hearn, 2016) or Healthcare (Bocek et al., 2017), is cross-industry transparency, accountability and trust among partners, and may have great advantages in the supply chain, where there is a great deal of interest misalignment in the involved parties (Chen et al., 2017).

Hyperledger also offers a coinless ecosystem, with the possibility of creating tokens (equivalent to cryptocurrencies) using smart contracts, if needed.

Another key aspect which is transversal to the Hyperledger frameworks is their modular approach to build, deploy and run blockchains and the underlying modules: Composer, Explorer and Cello, that accelerate and simplify these steps. For example, the use of REST APIs to interact with the network or web-friendly applications to query or write to the blockchain (Paul, 2018).

2.4.1 Sawtooth (Intel)

As we can apprehend from Intel Corporation (2015), along with the same advantages provided by Ethereum, by not using the PoW algorithm, and instead using PoET the energy consumption and computational power required for consensus in the network are greatly reduced while still allowing scalability. Thanks to its modularity, different consensus algorithms can be implemented to respond to network size changes.

There are, unsurprisingly, more aspects that differentiate Sawtooth from public decentralized ledgers like Bitcoin and Ethereum, among which we can indicate the following:

- Smart contracts are not programmed from the ground-up, but instead are based on pre-defined "templates", called transaction families, this makes the network safer as you cannot program ill-intended smart-contracts as easily.

- Transactions can be handled in a commonly used serial fashion, but also in parallel, while being broadcasted and validated in batches, so, if one transaction is not valid within a batch of valid transactions, the batch itself is not validated and is not recorded in the blockchain, which doesn't necessarily mean that the batch is lost, it is "saved" just not written to the blockchain (batches are not to be confused with blocks, i.e. blocks still exist but instead of containing single transactions they contain batches of transactions, that may contain one or more transactions).

- There's a developer-friendly separation between the application level and core blockchain system, which allows for an SDK that is available in many of the popular programming languages – Java, Python, C, C++, JavaScript, Go.

2.4.2 Fabric (Linux Foundation)

Androulaki et al. (2018) presents Fabric as a modular, business focused, open-source blockchain framework, which makes it a strong candidate for the objectives at hand. The two main features that separate Fabric and Sawtooth are: Membership Service Providers (MSPs) and channels.

With Sawtooth, regardless of it being a permissioned blockchain, i.e. only validated users can interact and explore the data stored in it, permissions/roles are handled with high flexibility, while Fabric provides a more robust and stricter approach to security through MSPs. These deal with the authentication of clients, and, again, thanks to its modularity, the Certificate Authority that handles the users' validation can either be Fabric's default Fabric-CA or an External Certificate Authority (ECA), so that every member can use their preferred authenticator. Additionally, transaction privacy is also improved in Fabric through channels, which are essentially separate smaller chains, can be connected to a main blockchain, that provide a secure way of sharing sensitive information only with selected members.

These two features are fundamental in designing a supply chain business network because partnerships change overtime and new clients are brought into the mix constantly. Along with these distinguishing features, Fabric counts on:

- Smart contracts being written in Go, also referred to as chaincode.

- There is a State Database that contains the world state for easier/faster validation procedures. By default, Fabric uses LevelDB (key/value tuples) but it can be configured to use CouchDB (JSON objects).

- Interaction with the network can be done with several APIs – Node.js, Java SDK or simply CLI.

- Apache Kafka, a real-time data streaming pipeline platform, is used as a very fast (finality is reached in seconds) permissioned voting-based consensus algorithm.

2.4.3 Other Hyperledger frameworks

The other currently active Hyperledger frameworks are promising in their own right, however, as previously stated, this thesis target lies in advancing SCM, and as such these don't serve the purpose as efficiently.

- Iroha focuses on mobile applications;
- Burrow is made to be a permissioned Ethereum-based blockchain, with the attached vulnerabilities;
- Indy has its core capabilities in identity management.

2.5 Technology summary

To better understand the subtle differences between the different candidate frameworks, Table 2.1 delineates the primary characteristics taken into consideration when choosing the framework that best fitted the needs of this project. In addition to the technologies' whitepapers, Tuan et al. (2017) provides a comprehensive comparison of relevant parameters like the state database, consensus algorithms used, scalability and smart contract language and execution. James-Lubin, Kieren (2015), Scherer and Eriksson (2017) and Goswami (2017) also share some more knowledge on network scalability and other performance factors. The table content is self-explanatory with the exception of the state-database feature that should be clarified: both transaction and account based models are typically associated with cryptocurrencies, but while in transaction based models users' data storage is based on unspent transaction outputs, account based uses a global list of accounts, their balance and code (Ethereum Community, 2014).

Table 2.1: Comparison of characteristics in relevant blockchain technologies

	Bitcoin	Ethereum	Hyperledger Frameworks	Corda	ShipChain	BigChainDB	Modum	IOTA	Quorum	Ripple
Purpose	Payments	General	General	Financial	Shipping	Asset transaction	Regulatory/IoT	IoT	General	Payments
Currency	Crypto	Crypto	None (but tokens can be issued with chaincode)	None	Crypto	Asset	Crypto	None	None	Crypto
Permissioned	No	No (new advances and altchains allow for permissioned versions)	Yes (Sawtooth allows permissionless chains)	Yes	Yes	Yes (but possibility for public chains)	Yes	No	Yes	No
Pseudo-anonymous	Yes	No	No	Yes (but can also be identity driven)	No	No	No	Yes	No	Yes
Auditable	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Immutable ledger	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
State database	Transaction Data	Account data	Key-Value database (LevelDB) or JSON objects (CouchDB)	Arbitrary Data	Account data	Transaction data (JSON objects) (MongoDB)	No info on data structures but uses PostgreSQL	Account Data	Account data	Account data
Modularity	No	No	Yes	No	No	No	No	No	No	No
Smart contracts	No (not easily achieved)	Yes (Solidity) (run on EVM)	Yes (Go, Java) (run on Dockers)	Yes (Kotlin, Java) (Run on JVM)	Yes (run on EVM)	Yes (but predefined and not arbitrary)	Yes (run on EVM)	No (but they're planned)	Yes (Go) (run on EVM)	No
Consensus protocol	PoW	PoW	PBFT, PoET, RBFT, YAC	Transaction validity and uniqueness (notary nodes)	PoW	BCA (BigchainDB Consensus Algorithm) Paxos-based	PoW	Directed Acyclic Graph ("Tangle")	QuorumChain (voting-based)	Ripple Protocol
Performance Scalability	Low	Low	High	High	Low	Very High	Low	High	Medium	High
Novel features	Introduction of blockchain	Smart contracts	Modular architecture	Pluggable Notaries	Track & Trace (ELD)	Big data, High throughput (100,000tx/s)	Passive monitoring with IoT in pharmceutical industry	No blockchain, but instead Tangle. Fee-less	Ethereum advantages but permissioned	Transactions in any currency

2.6 Architectural challenges

Many of the architectural problems described in the literature have or quickly become obsolete due to the rapid developments in the field. Rimba et al. (2017) compares the cost of blockchain with existing cloud-based platforms (AWS), however not only is the price of the cryptocurrency used extremely unstable (Ether = 11$ as of the writing of the paper, Ether = 900$ at the moment of writing this thesis) but the use of cryptocurrencies for private networks is unnecessary. Min et al. (2016) proposes a framework for faster and more dynamic permissioned blockchain transactions, that goes into detail on possible peer protocols and consensus algorithms to use (rendered useless with Hyperledger Fabric's channels and PBFT).

Kim and Laskowski (2016) address a very important point related to inter-organization data sharing and the need for ontology-based modelling. The paper suggests using TRUs (traceable resource units), first mentioned by Kim et al. (1995), as a way of having simple activities, such as pick-ups and drop-offs, virtually interact with physical objects via a set of updates (consumption and production of TRUs). Adding to Kim and Laskowski (2016), we can refer to Wu et al. (2017), the paper has a clear focus on the modelling and representation of data, defending the use of EDI (electronic data interchange) standards, such as EDI-214 translated to JSON format, to ensure blockchain interpretability. The authors also propose a framework with private chains for trading partners and a public ledger for extra validation of custody events (with no specific information, to preserve data privacy), diving into detailed descriptions of the types of events that can occur, distinguishing between private and public custody events (today improved with Hyperledger's channels) and how they are written to the blockchain.

The layer of blockchain technology that handles event triggers,as the ones mentioned above, is the Smart Contract layer, which can contain previously specified arbitrary rules (Buterin, 2014a) but also raise a question that Weber et al. (2016) tries to answer: How can business processes can be efficiently translated and monitored to/with smart contract code? The paper delves into this issue and proposes a translator that derives smart contract scripts from process specifications, e.g. Business Process Model and Notation (BPMN). Then these smart contracts can be either used to monitor collaborative processes and their state of execution or mediate business processes with continuous message reception/sending. The scope of this paper is, however, quite broad and not applicable to physical flow of products, as the ones in SCM, but does, regardless, encompass an important aspect of blockchain architectural design.

Another integral building block of this technology is consensus: a term that refers to the moment all the nodes in the network agree on what the current state of the ledger and add new information, that consequently changes the ledger to a new state on all nodes simultaneously (reaching consensus is also referred to as finality). A common and very real problem among distributed computer systems, as is blockchain, is the Byzantine generals problem (an adaptation of the proven-to-be-unsolvable (Akkoyunlu et al., 1975) Two Generals problem). It can be summed up to the Byzantine army, i.e. the network, having loyal generals, i.e participants that are interested in ensuring the networks integrity and their nodes, and possible traitorous generals, i.e. participants that, for a myriad of reasons, want to alter the data on the blockchain, and how they reach consensus on attacking, i.e. committing new information on the blockchain.

Proof-of-work is the most widespread probabilistic solution to this problem where the nodes work to solve a complicated encryption problem, consuming computational resources in process (Nakamoto, 2008). Solving the puzzle is finding a hash that has a leading number of zeros, by continuously generating hash strings, by altering nonces, which are integers that serve as inputs for the hashing algorithm, that, when combined with the previous block hash and the current block-to-be merkle tree root hash, output the desired hash. This makes it so that every block is correlated to all the previous blocks, and if any changes are attempted the attacker has to perform a currently immensely computation-intensive process in order to update all the subsequent blocks that inherently depend on the block that's being changed.

The PoW algorithm chooses who will decide the block creation based on chance (because solving the cryptographic puzzle is done by brute-forcing), and whoever has more computational power has a higher chance of winning, hence, theoretically, an entity with 51% of the computational power can take over the blockchain (Vukolić, 2016). To solve this the Proof-of-stake algorithm automatically assigns a chance of winning this lottery according to the stake the participants have in the network, e.g. someone with 3% of all bitcoin has 3% chance of being elected as the leader that commits the block to the chain.

Both consensus protocols incentivize the participants to act faithfully by rewarding them with currency, which is unsurprisingly infeasible for currency-less blockchain networks, like the proposed Hyperledger Fabric. Additionally, the Problems with PoW involve, but are not limited to, the 51% attack mentioned above and high power consumption, and, concurrently, PoS exhibits a lack of tolerance against double spending (explained further down) and increases centralization by prioritizing participants with higher stake.

The Practical Byzantine Fault Tolerance (PBFT) algorithm is a possible solution where each peer/node receives a message pertaining to the new piece of data and makes a decision, using computations based on the locally stored blockchain data, on whether the potential new transaction is valid or not. When all the nodes cast their decision, the network has a consensus and either commits the change or discards it, storing, however, the attempt and the perpetrator. This algorithm comes at a cost: anonymity, which is notwithstanding an aspect that should be forfeited in business networks where the parties involved have interest in transparency. Originally, in rc0.6.0, Hyperledger Fabric made use of PBFT, but with rc1.0.0 it now uses a Kafka-based ordering service, which may offer higher throughputs but no Byzantine fault tolerance (Tuan et al., 2017), which is not a requirement for membership-based networks (Nielsen, 2017). Despite that, a future Kafka-based ordering service with BFT is planned, though no release date is available at the moment of writing this thesis.

A considerable number of blockchain frameworks use proprietary consensus algorithms with different capabilities. Baliga (2017) describes in detail the most popular algorithms (PoW, PoS, PoET, etc.) but has a shortfall on Tangle (IOTA Support, 2017), Kakfa (Hyperledger Architecture Working Group, 2017), Raft (Nielsen, 2017) and Ripple Protocol (Schwartz et al., 2014). Table

2.2 illustrates the key differences between the algorithms of interest.

Table 2.2: Comparison of commonly used and proprietary consensus algorithms

	PoW	PoS	BFT (PBFT, RBFT, SBFT, BFT SMaRt)	Kafka	PoET	Tangle	Raft	Ripple Protocol
Examples	Bitcoin, Ethereum	Tendermint	Hyperledger Fabric and Indy, Corda	Hyperledger Fabric	Hyperledger Sawtooth	IOTA	Corda, Quorum	Ripple
Finality	Probabilistic	Probabilistic	Immediate	Immediate	Probabilistic	Probabilistic	Immediate	?
Throughput	Low	High	Higher	High	Moderate	High (inversely proportional to # of tx)	High	Very High
Cost	Yes	Yes	None	None	None	None	None	Yes
Peer Scalabillity	High	High	Low	Low	High	High	?	High
Trust	Decentralized	Decentralized	Semi-centralized	Semi-centralized	Decentralized	Decentralized	Centralized	Decentralized
Byzantine Fault tolerance (any f % are subverted)	Yes (f <25%)	?	Yes (f <33%)	No	No	?	No	Yes (f <20%)
Crash Fault tolerance (any t % crash)	Yes (t <50%)	?	Yes (t <50%)	Yes (t <50%)	Yes (t <50%)	?	Yes (t <50%)	Yes (t <50%)

2.7 Hyperledger Fabric's operations

Hyperledger Fabric's "unique" transaction flow (simulating, ordering, validating, as opposed to ordering, executing) prevents double-spending because each transaction is first validated, or rather verified against the current state of the ledger, and then if validated it is committed locally on the validating peers and broadcast (via the orderer, using Apache Kafka to handle high throughputs) to the non-validating peers if consensus has been achieved by the endorsing peers and while not violating the endorsing policy. A more detailed flow of a transaction follows:

- A peer initiates a transaction, by sending a transaction proposal created using the SDK which contains the invoke request, i.e. the chaincode function that is going to be called as well as necessary data, and the set of required cryptographic materials, i.e. the clients credentials, that will digitally sign the proposal.
- The endorsing peers execute the following validations:
  - the structure of the transaction is correct (syntax-wise);
  - the transaction is unique and has not happened before (replay-attack protection);
  - the signature is valid, by leveraging the MSP;
  - the client proposing the transaction has indeed permission to do so (writers policy for a given channel);
- And then execute the transaction locally, without actually altering the ledger, against the current world state to produce a read set (response to the transaction execution) and write set that is signed and sent back to the SDK.
- The SDK analyses the responses and cross-checks them to ensure coherence, i.e. they're all the same, and checks if the endorsing policy criteria have been met.

- A transaction message is sent to the Ordering Service, which orders them chronologically by channel and creates block of transactions per channel.
- These blocks are broadcast to all network peers, that do a final endorsing policy check and read set validation to ensure that no changes have happened between the execution of the transactions in the 3rd step and this step. Transactions are labeled as valid or invalid;
- The block is appended to the ledger and every valid transactions' write set is committed to the world state and the client is notified.

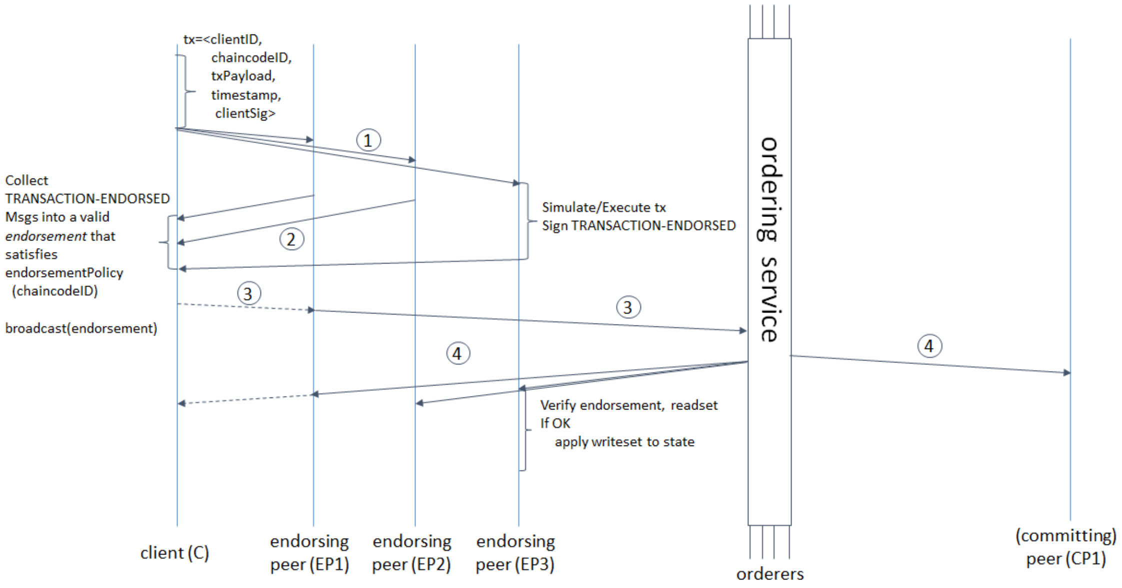

tx=<clientID, chaincodeID, txPayload, timestamp, clientSig>
1
Collect TRANSACTION-ENDORSED Msgs into a valid endorsement that satisfies endorsementPolicy (chaincodeID)
2
Simulate/Execute tx
Sign TRANSACTION-ENDORSED
ordering service
3
3
broadcast(endorsement)
4
4
Verify endorsement, readset
If OK
apply writeset to state
client (C)
endorsing peer (EP1)
endorsing peer (EP2)
endorsing peer (EP3)
orderers
(committing) peer (CP1)

Figure 2.1: Sequence diagram of transaction flow (common case). Source: Hyperledger (2017)

One of the integral underlying pillars of a fabric network is chaincode, what is commonly referred to in blockchain lingo as smart contracts. These pieces of code have a particular interest in business networks as they will not only translate query and write functions into machine understandable commands, but also the business logic, e.g. the inter-organization relationships. We have, for example, the channel policies: if our network is comprised of 3 organizations, A, B and C, then it is safe to assume that, at the moment of deploying the network, the firms will agree on an endorsing policy which states that, for instance, all 3 organizations have to agree in case one of them, A, wishes to add a 4th organization, D. This is not to say that all organizations are, in this case, administrators of the network, that responsibility will usually fall in a reduced number of parties, it does however provide a way of ensuring that administrators don't take over the inherently decentralized network. The question is then how will the policy be changed. The incumbent organizations and the entrant will agree, similarly to a real contract, on whether the the policy stays the same, i.e. A, B and C need to agree (but D doesn't have a say), or if D is included with an AND operator. To achieve this modification, the proposer, A, creates the appropriate cryptographic material (credentials) for the new organization peers, and issues a change in the configuration of the channel to which D is to be added. This process also includes the new policy for future alterations (e.g. A AND B AND C AND D). When finished, the deployed chaincode has a higher version number than the previous, and all the actors, versions, dates, etc. are stored in the ledger, so that there's a fool-proof way of tracking alterations to the network.

It is clear to see that the blockchain landscape is full of emerging startups and open-source projects, each with their unique features and corresponding advantages, therefore the choice of framework for the present work relies on a holistic view and judgement from the author. The Hyperledger Fabric project developed primarily by IBM gathers a set of necessary conditions, like business partnership-oriented channels, easily upgradeable smart contracts, needed to keep up with ever changing relations in supply chains, accountabillity due to its semi-centralized nature, a consensus algorithm (Kafka Ordering Service) that, albeit doesn't account for a throughput of the likes of BigChainDB's algorithm, still allows for considerable, and sufficient, flow of transactions, and very valuable modularity for expeditious scalability.

Although this dissertation doesn't offer new technical insights in terms of developing new blockchain technologies, it does gather a multitude of authors perspectives that hold tremendous value and how the exposed concerns have been resolved, partially or completely, with more recent technologies. It also proposes the inclusion of most of the theoretical formulations into a materialized proof-of-concept application.

Chapter 3

Problem Description

The interest of applying blockchain in the context of this case study been developed stems from an intrinsic complex network of stakeholders and a complex flow of information that's not unidirectional. The company interacts with dozens of other parties to ensure that a product leaves the supplier and reaches the customer in the least amount of time for the least cost. This inevitably leads to scalability concerns when those dozen partners become hundreds or thousands. That is to say that this complex flow of information is clearly a threat to Supply Chain Management, as previously stated in Chapter 1. Although the problem is universal, and every internet blog, magazine, news outlets has stated the benefits of blockchain, the present chapter intends to shine light upon a somewhat specific stage of the supply chain, that takes place under a single firm, trying to remain, however, as agnostic as possible, to allow for other scholars or field professionals to extrapolate the results in their own interest.

As such, this chapter will present an overview of the structure of the organization as well as an analysis on the process of communicating information and transferring documents, how, when and by whom the documents are created, handed over, verified and analyzed. By rethinking how the information is generated and transferred between different parties, the goal is to find find processes that can either be automated through the use of blockchain or ideally removed altogether (Hammer, 1990).

A key aspect that is delved into with more detail further on in Chapter 4 is the modular mindset that is used in the software development department, which is, in a way, related to how the different internal departments and external partners communicate between one another, and naturally, for a successful implementation of this project, how a blockchain application fits into this ecosystem.

3.1 Internal structure

The company follows a very standard functional organizational structure with 6 departments: Information & Technology, Account Management, Operations, Business Intelligence & Artificial

Intelligence, Marketing & Sales, Financial & HR. The first three mentioned are of special importance for their integral functions as integrators with the stakeholders, digitally for IT and AM, and physically for Operations.

The **Information & Technology** department develops and maintains the company's digital cornerstone, Spoke, and its modules, WMS, OMS and DMS. This information system is a vital part of the company's value proposition, and as such a primary concern is that of making the platform as versatile as possible while providing a simple to use, "plug-and-play" solution.

Account managers are responsible for two phases of a client-brand's life cycle with a service company, as the one where this project is being developed. An initial on-boarding stage, that is characterized by a tour of the system and of its inner workings and the clarification of any doubts, is followed by personalized insights that are facilitated by the high amount of data the system consumes and outputs. Ideally, the system will become so easy to use that the initial walk-through can be greatly reduced, and the AM department can focus on CRM. This department has also the responsibility of arranging the necessary documents for transport and exporting.

Operations is in charge of warehousing activities, i.e. reception of items to stock, picking items from stock and reception of items to fulfill a sale orders. The fulfillment of orders has special relevance, and is already aided by the modules made available through the company's software, Spoke. However, there is still a heavy document-related component to outbound logistics, namely a packet of several documents, needed for customs and financial regulation, which is handled both by AMs and Operations' staff.

Data analytics are mainly handled by the **Business Intelligence & Artificial Intelligence** department. The big amount of data that is generated from the thousands of orders that are managed by the company is analysed with machine learning and optimization algorithms, which need large data sets for ideal performance. That is why IoT, alongside blockchain, is an interest for this department, as it enables far more data gathering with fewer mistakes, with the added benefits that blockchain provides.

Marketing & Sales essentially find and convert leads, or prospective client brands, or, in other words, actively search for brands that fit the profile, that is, brands that can benefit from a comprehensive end-to-end SCM solution, and turn them into clients.

Lastly, but equally essential, the **Financial & HR** department ensures financial and regulatory standards are being met. It is also its responsibility to design KPIs and study the cost structure to improve economic performance. This project impacts the cost structure indirectly by reducing work hours needed to perform tasks up and down the supply chain, so it stands to reason that it has a measurable impact for this department.

3.2 External partners

The relationships with other organizations are fundamental for the goals of cost reduction that the company proposes as a holistic manager of the entire supply chain from production to delivery, which can be either to wholesalers or to actual end-customers. It is then only natural that the problem analysis includes an explanation of how the different stakeholders play a part in the managed supply chains. The actors can be categorized as 6 main types of stakeholders, and those are: client-brands, suppliers, carriers, customs, end-customers and tax authorities.

Client-brands are revenue drivers for the company and the stakeholder that directly benefits from the offered services. The majority of money-related transactions occur between the either the client-brands and the managing company, or between the client-brands and other stakeholders but routed through and handled by the managing company. Concurrently, there is also a big volume of data that is mainly communicated to and from this partner, namely Sales Orders, Purchase Orders, SKU descriptions and stock availability, transport updates, and invoices.

Suppliers have a more restricted role of producing the right amount of the right product at the right time. The orders that are ready are fetched from the supplier and, depending on whether they'll be shipped or stocked, they'll be processed at the company's warehouse for the next stage. The transactions with this partner are mainly physical, e.g. a box of t-shirts with the respective invoice, and thus are not too complex. Regardless, the performance of suppliers can gravely affect the subsequent phases of the supply chain so it is important to take these interactions into consideration when designing a blockchain network for SCM.

Carriers , however, despite also having a seemingly restricted scope of action, have a vital role in ensuring the timeliness of the outbound process. The reason being that they are the bridge between the warehouse and the end-customer, and are, in case of exports, directly obstructed by customs. The flow related to this partner assumes then both the physical aspect, similar to the suppliers', and the informational one, similar to the client-brands, and it is, along with the SCM service company, a focal point for the bulk of the data that is relayed between stakeholders, e.g. carrier-specific labels, invoices, bill of landing, packing lists, certificate of origin, etc. In the common case of exporting, the SCM company, carrier and client-brands have to exchange information that is ultimately "compiled" by the carrier and handed to customs for verification.

Customs represent a severe bottleneck for SCM, nonetheless necessary by law. The bottleneck manifests itself through long document validation times, which can result in non-clearance of exports due to some data misalignment or missing documents. The aforementioned stakeholders goal is to make sure that the information that arrives at customs is valid. However, even valid information takes time to verify manually.

End-customers are indirectly affected by all these "behind-the-scenes" processes but directly affected but their failure. In other words, if all the partners do their job on time correctly the end-customer does not notice anything relevant, but if one document is misplaced or a bar code is misread (anecdotal examples), the end-customer might receive the product one day too late and, innately, be unsatisfied. Even though this stakeholder is more intimately related to the client-brands, it is still of high priority for SCM companies to uphold their SLAs, which are linked to the end-customer satisfaction.

Tax authorities are a background actor but still pose a relevant role in SCM, since there are large amounts of money that are being transferred across parties. Information sharing is, in this case, the main driver to take into account TAs for the simple fact that often times invoices or other necessary documents are bounced around parties before being delivered to TAs, which is inefficient.

Figure 3.1 encapsulates how these different stakeholders interact within the supply chain.

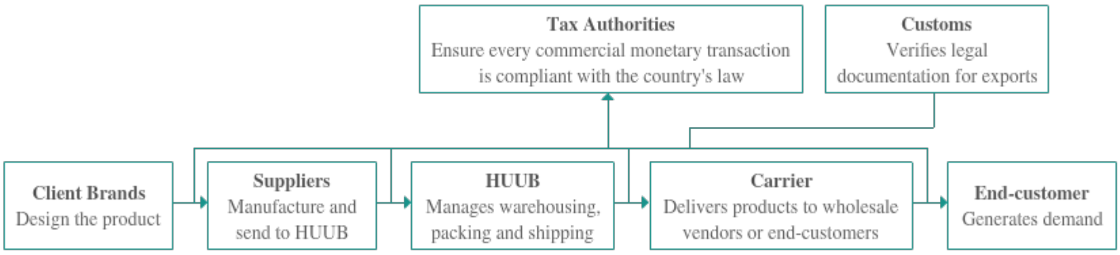

Tax Authorities
Ensure every commercial monetary transaction is compliant with the country's law
Customs
Verifies legal documentation for exports
Client Brands
Design the product
Suppliers
Manufacture and send to HUUB
HUUB
Manages warehousing, packing and shipping
Carrier
Delivers products to wholesale vendors or end-customers
End-customer
Generates demand

Figure 3.1: Simplified view of stakeholders' sequence of activities

3.3 Process as-is

The sharing of information in the supply chain is generally not arbitrary, as in there's a process that triggers another process downstream, each carrying data with them. As mentioned previously when describing the company, the scope of action of a SCM service provider usually entails overseeing operations from supplier to end-costumer, however, simply because enlarging the scope of this project would only marginally increase the findings but linearly increase the time of research, a closer look of just the outbound process was taken, specifically the one set off by an e-commerce order, which, in the eye of the company's staff, warrant the most attention due to their complexity.

This outbound process starts when an order is placed by an end-customer on an e-commerce platform. The process can immediately trigger a response on the SCM company's side when a software integration is in place, which is often the case, but it may rely on email communication of information related to the order, e.g. an invoice.

This trigger sets off a series of internal order processing operations, e.g. pick from stock, reception to SO, packaging, etc. Since the core information for the order processing, like SKU, quantity, shipment location, just to name a few, is present, the order starts being processed, all the while generating documents needed for shipping and export, yet, sometimes, it is only when these operations are being wrapped up that missing documents are detected, due to the separation between warehouse operations and document handling.

A new "backwards" communication line is then opened to request the missing documents, or pieces of information. After being processed internally, the SCM company requests a document, e.g. a label, from the carrier, which will have to be physically signed by both entities. The packaged is then handed of to the carrier, that has its own silo of information, often accessible through an API (usually under the form of a tracking number request). The carrier processes the order internally as well, those operations are mostly opaque to outside parties (as they should be to preserve competitive advantage), and hand-in the packages for customs clearance. Customs' officers manually validate the documentation and either clear for export or withhold the items and open yet another line of communication, that back tracks from party to party until the original creator of the document, that holds the necessary data to correct the mistake. When everything is error-free the package can be internationally shipped and a similar mirrored process occurs at the destination, i.e. customs receive the package, validate it, communicate any non-conformity by relaying it intermittently, hand it off to the carrier, and finally deliver it to the end-customer. Figures 3.2 through 3.4 depicts portions of a flow chart to visually illustrate the sequential nature of this outbound process and the silos of information, represented in blue bellow some processes indicate the disconnection of information in what should be a transparent environment. The sharing of information is happening, as it should, but in a very inefficient manner with enormous amounts of time wasted on communication and manual validations that are warranted by the physical nature of the documents that are exchanged.

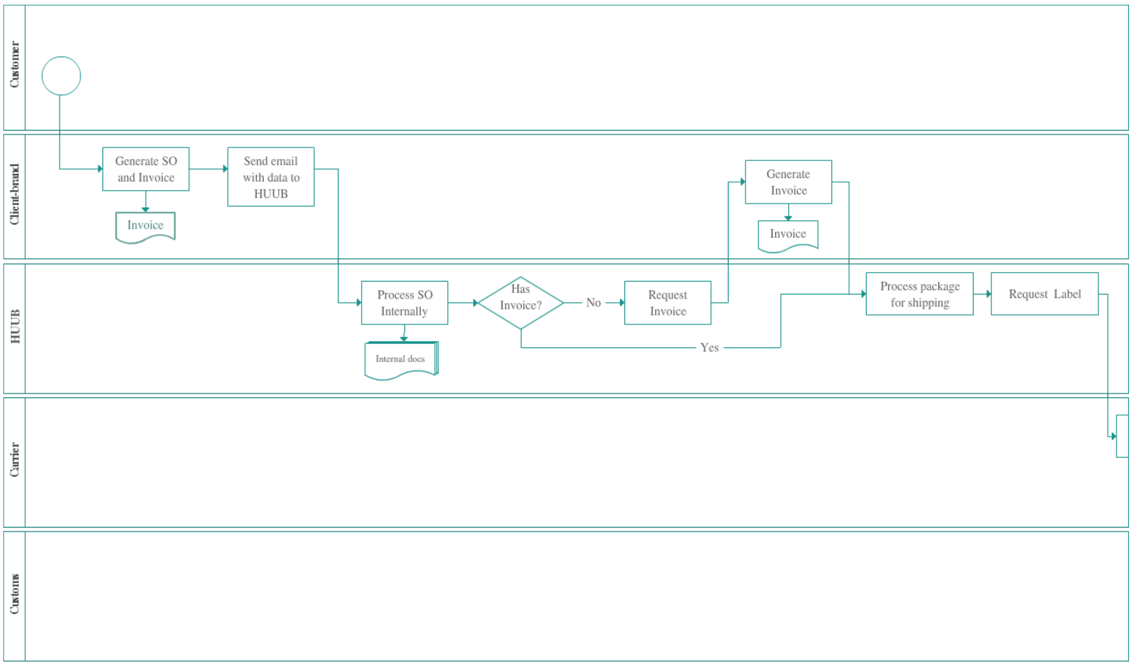


Figure 3.2: 1st stage of the flow chart of outbound operation for e-commerce export order

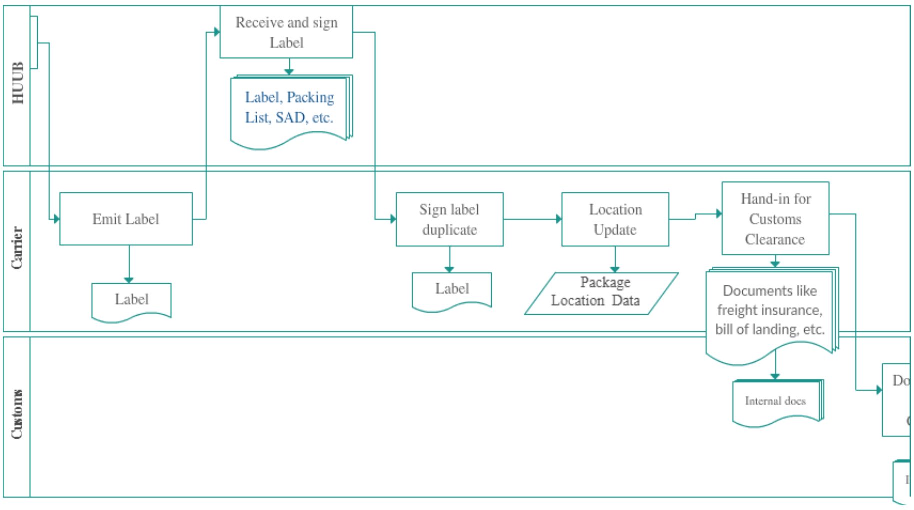


Figure 3.3: 2nd stage of the flow chart of outbound operation for e-commerce export order

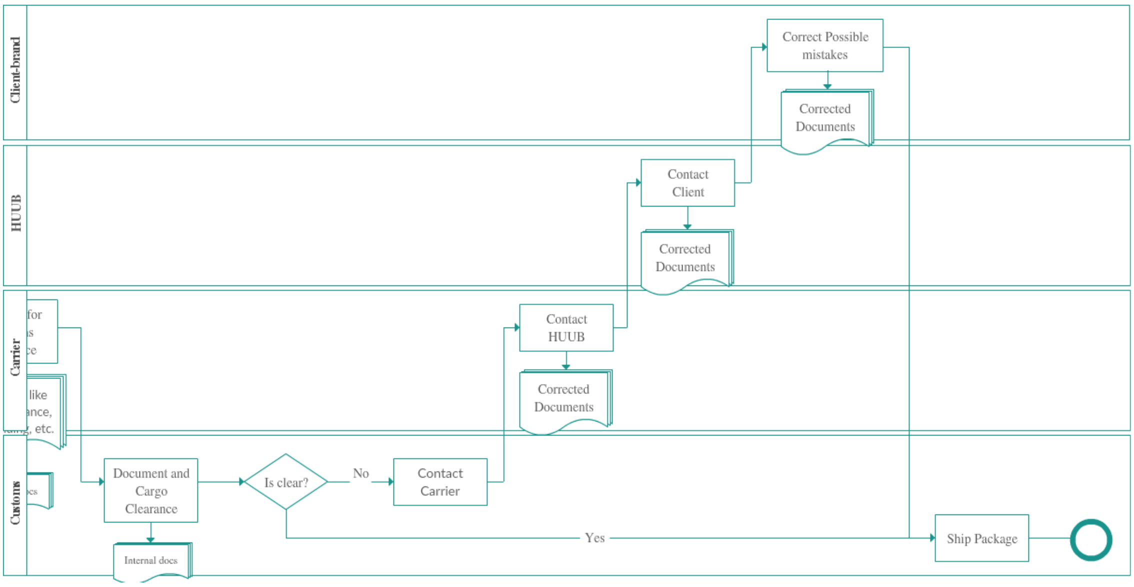


Figure 3.4: 3rd stage of the flow chart of outbound operation for e-commerce export order

3.4 Limitations

Nowadays, there are digital equivalents of these documents do exist, such as EDI mentioned in Section 2.6, but these digital counterparts lack the necessary accountability. In other words, an EDI document, i.e. a computer file that contains all the data associated with a relevant physical object, lacks the necessary infrastructural support to verifiably determine who actually created that file. That is to say that there are ways of turning physical document flows into digital transactions, though there isn't a secure way for customs or other parties to verify origin and ownership. So, the current way still relies on signatures and hand-to-hand interactions that carry their own proof.

The situation mentioned above translates into an inability of turning manual tasks into computer automated programs that can run instantaneously when a trigger is activated. A simple operation that can be further described is the manual verification of export codes on invoices, for regulatory and taxation purposes. Currently, this process falls on the carrier's hands and, from the moment the documents are in the carrier's possession until they are sent to customs for authorization of export, it goes through 2 main stages depicted in Figure 3.5. The Gantt diagram illustrates how long the document handling portion of the process is, for an arbitrary 50 minute package picking route performed by the carrier to collect N packages. To clarify, the 50 minute route is arbitrary due to the fact that the physical document verification done on-site is negligible, so the 50 minutes depend solely on the time of picking up packages and the time it takes to drive from point to point. That being said, the 10 minute gap between activity 1 and activity 2, as well as the 20 minutes of document validation for each package, are not arbitrary values and are actually obtained from measurements. It takes approximately 10 minutes from the moment of document drop-off by the courier to document pick-up by the office worker. It also takes approximately 20 minutes to verify the documentation for a complex package with several items. A Gantt diagram was chosen because even though several workers may be working in parallel, a single worker can only start a new activity when the previous is finished.

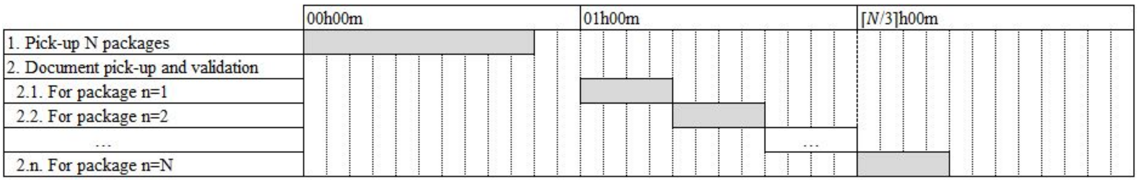


Figure 3.5: Gantt diagram of activities involved in the carrier's preparation of a package for export (5 minute interval)

These measurements demonstrate, for this specific operation, the copious amount of manual labour that is required for simple cross table verifications, due to the sheer fact that a human cannot simply instantaneously access bits of information on tables with thousands of entries.

3.5 Process to-be

The manual processes, as described, present a high level of automatability that is hurdled by bureaucratic and legal necessities that cannot be altered, at least for the foreseeable future. The proposed blockchain architecture, when in place, can assure both the correctness and speed of process execution as well as acountability, or proof-of-ownership, required by the government and other stakeholders. The hub-and-spoke physiognomy of the blockchain network, i.e. a central datawarehouse with several points of access by different entities, can additionally grant instantaneous access for more timeliness in processes that can start simultaneously.

The time reduction and asynchronicity described above results in a flow of operations that has less activity precedence. Referring back to the flow chart described in Figures 3.2 through 3.4, a comparison is presented in Figure 3.6 (a "bird's eye" view of the mentioned figures) and Figure 3.7.

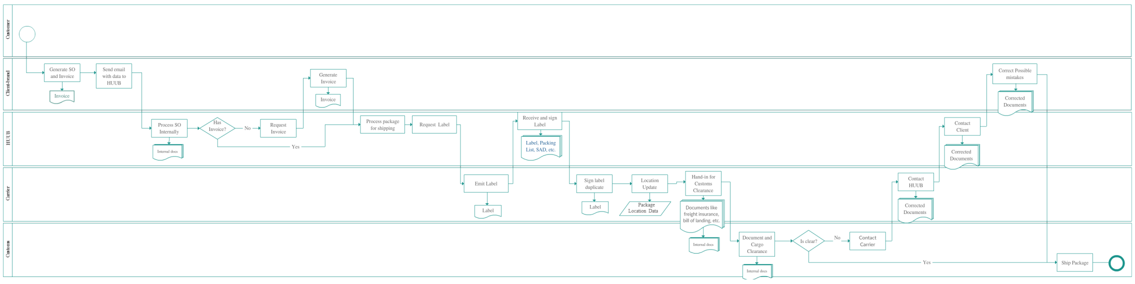


Figure 3.6: Flow chart of outbound operation for e-commerce export order

It is worthy to mention the change in color, from blank to blue, of just some of the document creation moments. The reason is related to the fact that there is a need for some data to be shared in the network, though only partially. This prevents clogging the network data pipelines with useless or confidential information. The sharing of data is restricted to documents that are currently already accessed by several parties, albeit in an inefficient way.

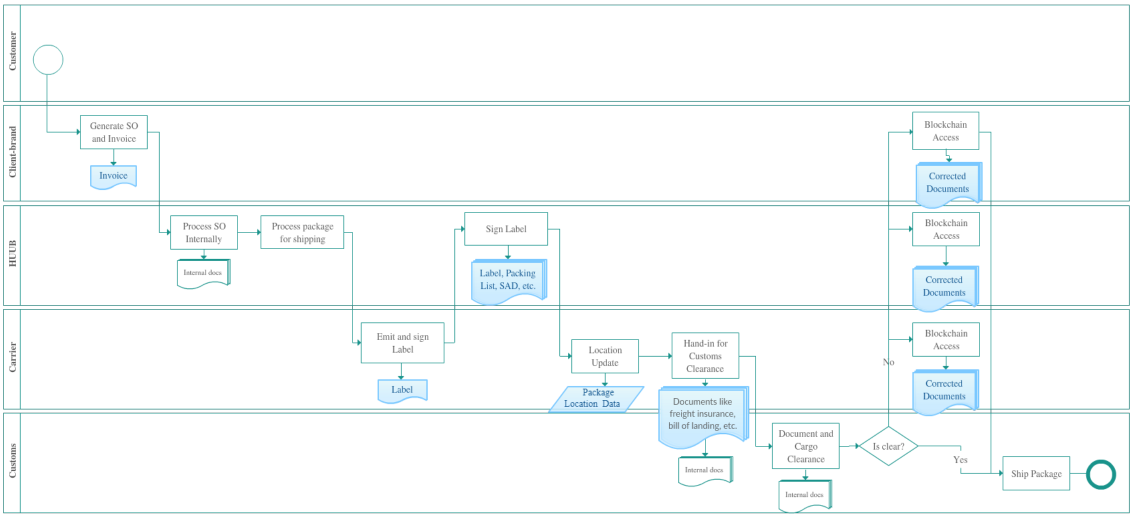


Figure 3.7: New flow chart of outbound operation for e-commerce export order with proposed underlaying blockchain architecture (data stored in the ledger is colored blue)

It is also fairly simple to assume the sequence of activities represented in Figure 3.5 becomes obsolete, not only because of the gains in speed of the tasks, performed by a computer and not office workers, but more importantly because they can happen even before the courier reaches the carrier's facilities, i.e. immediately after the creation of the documents.

Chapter 4

Methodology

Given the novelty of this project, there's a lack of formal methodology, but there are similarities to action research methodology that can serve as a guideline for this chapter. The aim is to translate what has been described in Chapters 2 and 3 into a clear scheme for the proposed software architecture. Firstly, a thorough walk-through of the different components that make up a Hyperledger network is undertaken, explaining how different actors coexist in the network, how data and business processes are expressed in smart contract logic, followed by how authentication and interaction with the network is performed.

4.1 Approach

Firstly, stakeholders were identified and an analysis was carried of the processes and their value, to understand and discern flows where blockchain could add value and flows where blockchain would only add redundancy or even retract value. Secondly, where the business logic and theory, previously researched, were converted to technical software specifications specific to Hyperledger (see Section 4.2). The final stage was designed to cover the practical, yet restricted, implementation of an experimental pilot.

4.2 Hyperledger Fabric

The documentation available online for Hyperledger Fabric is very comprehensive and includes a set of examples that allow fast and easy deployments of a simple networks that work at a basic level. This means that for developers it's fairly straightforward to simply download a small amount of documents, install the necessary dependencies and, effectively, run a blockchain network.

There are available frameworks to construct and deploy a blockchain networks and application in Hyperledger Fabric, but these add a layer of abstraction on top of Fabric that, for this project, is not desired, since one of the focuses here is to understand the caveats of the technology and understand how real business-related processes are interpreted digitally in the blockchain.

To build a network that correctly mimics the real-world business model, there are several aspects of Fabric that warrant special configuration. There are, however, modules that serve a more technological purpose and don't interfere with the business implications or the networks performance, on a significant level, and for that reason are not explained with more detail. For example, Membership Service Providers (MSP) and Certificate Authorities (CA) serve to translate real-world identities into verifiable cryptographic digital ones. The documentation provides a simple analogy, that can be further simplified, as follows:

- A store can take payments from Visa and Mastercard.
- Regardless of a client having enough funds to pay for a purchase with a valid American Express card, the store only accepts the above mentioned cards.
- Visa, Mastercard and American Express are analogous to CAs, that issue verifiable certificates, and the store's list of accepted methods is similar to an MSP, that determines which CAs are "trusted".

More specifically, Fabric uses standards from the X.509 Public Key Infrastructure (PKI) and, even though the MSPs and CAs can be interchanged, it's only a matter of preference, and neither performance nor business processes. Thus it is not relevant to delve into this matter further, as the available components fully serve their purpose. It is, nevertheless, important to understand the role of MSPs and CAs in the network, as these modules guarantee that only trusted entities with the necessary permissions can interact with the network, hence this introductory note.

As an additional point, it is necessary to start with a distinction between the network and the application, to immediately discard potential future confusion. The network development and deployment relies on a series of programming libraries and software that need to be installed in the systems where the blockchain runs, like cURL, Go Programming Language, Docker and Docker Compose, NodeJs and NPM, and Python. Not unexpectedly, Hyperledger's Fabric source code that orchestrates the different components (Peers, Orderers, Fabric CA) as well as tools for intermediary steps when deploying or updating the network configuration (cryptogen, configtx, configtxlator). The application was developed with the official Hyperledger Fabric Node SDK, but the access is done through the browser via an API endpoint, hence the only requisite is access keys.

4.2.1 Network configuration

One of the questions posed by the literature concerns what information exists on the blockchain, referred henceforth as on-chain data, and what information remains in private offline servers or private cloud services, similarly, off-chain data. As a general rule, information that is solely shared across two parties has no need to be on-chain. In most of these situations, the higher performance and reliability of current services, like AWS, are sufficient, and adding on-chain data unnecessarily increases redundancy. Likewise, information that is not shared with other stakeholders, e.g. linked to internal operations, should also remain off-chain. The blockchain purpose is then to serve as a repository of information that needs to be shared and accessed by two or more stakeholders along the supply chain, and not as a communal database for every entity to, independently of its usefulness, "dump" their data and clutter the network.

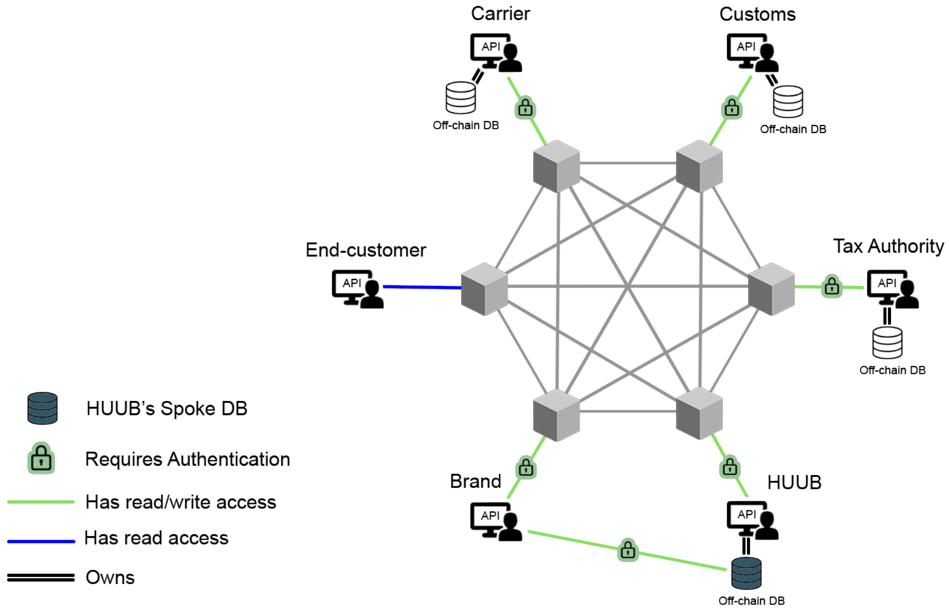


Figure 4.1: Overview of the blockchain network and stakeholders' access to data

Considering Fabric's capabilities, the network can be blueprinted with the schemes ahead. Figure 4.1 presents a generalist view of the different stakeholders in the SCM context access the several data infrastructures.

For a more explicit view of how stakeholders with similar characteristics but different permissions interact, Figure 4.2 illustrates Fabric's feature known as channels. Channels enable the co-existence of competitive stakeholders while still maintaining confidentiality, by existing completely independently from on another, in separate blockchains. Channels are still governed by the overarching network. For example, Carrier A works with Brands 1 and 3 and therefore has access to those two channels. On the other hand, Carrier B, who only works with Brand 2, can only access the channel designated to them, and not other Brands' channels. HUUB and the government have access to all channels because the former manages the entire supply chain and the latter for legal reasons. That's not to say that accredited authorities can access companies' confidential information, but merely the legal documents that require visibility throughout th supply chain.

Before further explaining how different entities are created and permissions are set, it is key that the basic level of network functionalities are understood. The network is structured with several services, of which peers and orderers perform the main blockchain operations, as described in Section 2.7. The source code for these services is made available through Docker container images. A container image is a "lightweight, stand-alone, executable package of a piece of software that includes everything needed to run it: code, runtime, system tools, system libraries, settings"

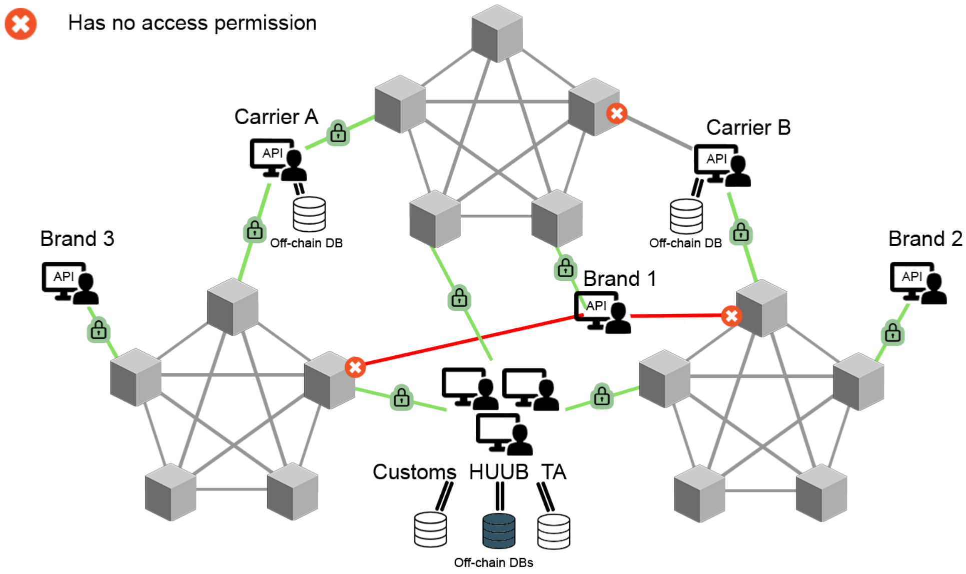


Figure 4.2: Channel's view of the blockchain network and stakeholders' access to data

(Docker Inc., 2018). In other words, they are machine virtualizations that offer high performance due to their simplicity and self-containment.

Being that different organizations have different stakeholders, the configuration of the network is dependent on the organizational needs of the parties involved. This configuration is defined in *.yaml* files that are interpreted by Docker Compose to create the necessary Docker containers that will perform tasks like writing a transaction or querying a value to or from the blockchain. A transaction is nothing more than an object that is stored in the blockchain, it doesn't objectivly mean that an asset is being transacted between entities.

Entities and their permissions are initially set by one ore more organizations that conceptualize and develop the first iteration of the blockchain network. These rules are set empirically and after discussion and agreement from the parties involved. Another set of *.yaml* configuration files with the agreed upon permissions, number of organizations, peer nodes, channels is consumed by the tools mentioned at the end of Section 4.2 and a number of cryptographic files is generated. These files are interpreted by Fabric, at the Docker container level, and are untamperable to an extent, because channels and their chaincode will dictate who can write, query the ledger or update the network configuration, e.g. add or remove an organization. These operations are explained next in Section 4.2.2.

4.2.2 Smart contracts

Unlike other blockchain technologies, Fabric relies heavily on smart contracts to function, and designates them as chaincode, i.e. compiled code that runs on the blockchain, essentially on the peer nodes of the network. The process of chaincode installation and instatiation determines the set of possible actions, written in the form of functions like *invoke* or *query*, that can be performed on a channel. A channel, on the other hand, has a list of which entities can and cannot call those actions. Without channels and chaincode, the blockchain does not exist. This is important for the flexibity that Fabric claims, because if there's a need to create a channel, in other words a ledger, where one entity can write/query data but all other participants can only query, the framework allows the fulfillment of that requirement.

The complex flow described in Section 2.7 is not tackled in the scope of this project, as it regards lower level network operations that are handled by Fabric autonomously, and thus it is not tackled. Despite that, there are two proposed levels of significance in chaincode that should be clarified: the data layer relates to the core functionalities of the network, e.g. writing a new transaction, and the layer that translates the business reality into computer interpreted code. Both layers rely on each other to operate, as one provides the blockchain functionality and the other the data structure.

4.2.2.1 Data layer

The data layer is a set of chaincode *.go* files that contain the functions that will read the data provided and, for example in *json* format, and convert it into bytes that are the written to the ledger as transactions. This layer doesn't handle user validation, but it can contain certain rules like the number of arguments an asset can have or deter unwanted values. As an example, it is the data layer that determines if an asset with 4 arguments can be written to the blockchain with only 3 of the arguments containing information, making the 4th argument optional. Similarly, if the argument has to be of type string or integer, it is also defined in these files.

As mentioned in Chapter 2, blockchain is "immutable", i.e. the only actions that can take place are *setting*, writing new data, and *getting*, querying written data. This does not pose a threat to updatibility of written data, or in other words, if wrong data is written by mistake or ill-intent, it can be amended to the correct data, but the blockchain has recorded the two incidents, and when querying that data, the function can either retrieve the most current state or its history.

The literature proposes that a distinction be made between *invoking* chaincode functions and the functions themselves. For this project, one more separation took place between *get* and *set* functions, for folder and file structure user readability.

4.2.2.2 Business Logic layer

The translation of business logic proves to be the most cumbersome, for it entails a deep understanding of the business and its nuances and how those characteristics can be correctly expressed in a language that Fabric, or other frameworks for that matter, can use. This layer encompasses aspects not only from the chaincode level but also from the network level.

In regards to chaincode itself, there is a requirement of creating models, or in other words structures, of assets that are stored. This is unequivocally related to data, but it pertains to a reality that is business dependent. It can be viewed as the equivalent of creating tables with different columns in a relational database like MySQL. As an example, if a client-brand has to be identified through a name, location and a legal identifier, then a model for a client-brand is defined as having these three arguments.

Inter-company relationships, as they occur in the supply chain, demand a more careful approach, however. Updating chaincode is the secure way of guaranteeing that new participants or removed ones have the right permissions to access the blockchain. The way this is handled is non trivial, but comprehensive. It needs to be clear that every operation that is performed on data in the blockchain is backed by chaincode, and chaincode runs on peers (Docker containers) that belong to specific entities. In other words, for an organization to read or write data, regardless of whether they have permission to do so, it needs at least one peer and specific chaincode installed and instantiated on that peer. Secondly, when chaincode is installed it only applies to the channel it has been installed to. Thirdly, the first instance of chaincode defines which entities can update said chaincode, and this can be made to be as simple as, for example, in a network of 5 participants, "Participant 1 has the right to update the chaincode regardless of others permissions" or "If Participants 1, 2 and 3, or 2, 3 and 4, or 1, 3 and 5, agree on updating the chaincode, they can regardless of the other participants approval". This policy, as Hyperledger denotes it, enforces democracy in the network and versioning of chaincode ensures the possibility of adding or removing participants from the network, which is a fundamental premise of supply chain management. To illustrate, Figure 4.3 depicts how chaincode enacts changes in the nature of inter-company relationships, where the key aspects are the policy, that could be "Organization 2 and 3 can update the chaincode", the removal of Org1, by not installing the new version of code on its peers, and the addition of Org4, by installing the new version of code on its peers.

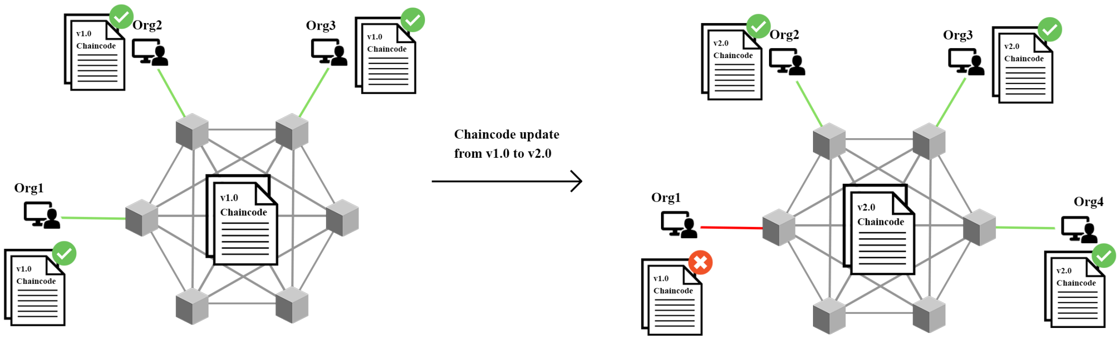


Figure 4.3: How policy and versioning enable dynamic relationships in the network.

4.2.3 Authentication

Security is a big concern for this project, as it deals with very sensitive information, and in a supply chain network there are stakeholders with competitive differences who have an imperative need for confidentiality. It is also crucial for a company that manages data from different stakeholders to not have that sensitive information easily accessible for others, for example prices that can be viewed in invoices.

Fabric handles authentication with a public key infrastructure, that means that every entity needs a certificate, public key and private key to interact with the network. For simplicity, in this project these were generated with the tools provided by Hyperledger, mentioned above in Section 4.2. To shed some light on how crucial this PKI is for a proof-of-concept of a blockchain implementation, it is how the problems mentioned in Section 3.4 are solved. By having every entity in the network mandatorily identified, every interaction that an entity has with the network is almost instantaneously verified. Additionally since each participating stakeholder holds their own certificates, public and private keys, and no others, the blockchain is "ownerless", in a way. The term is between quotation marks because there has to be a group of organizations that initially develop and deploy the network, but no single entity has ruling power over it.

4.3 Interaction

For a first network deployment, that is, turning all the *.yaml* configuration files, chaincode *.go* files and RSA 2048 keys and *.pem* cryptographic files into a blockchain that can receive requests for creating and reading data, the command line is the only tool, assuming no layer of abstraction is being used on top of Hyperledger Fabric. One of the simplest ways to automate command line commands is through bash scripts, which are made available by IBM and Hyperledger. With this in mind, and referring back Section 2.4.2, there are two official SDKs: Java and NodeJS, a well-known Javascript environment. The SDK provides a layer of abstraction that, in the author's opinion, don't oversimplify the architecture of a Fabric blockchain network, unlike the unused layers of abstraction given by, for instance, Hyperledger Composer. More objectively, the Java or NodeJS SDKs extend the reach of the network interoperability to Javascript rather then simplifying the development of the network itself.

Given the project's time span, the NodeJS SDK was chosen due to the already existent experience with Javascript, not only from the author's side but from the IT professionals consulted throughout this case study. The choice was not based on performance or simplicity of code, and the other SDKs, Python and Go, were not considered for not being official releases from Hyperledger and potentially not offering the same support and documentation needed.

To produce a simple-to-use Web API, a NodeJS web application framework was used, Express, which is, similarly to other tools mentioned previously, recommended by Hyperledger, and a logical option given the choice of SDK. The Python SDK was contemplated with conjunction with Flask, but was opted out for the already mentioned criteria.

Chapter 5

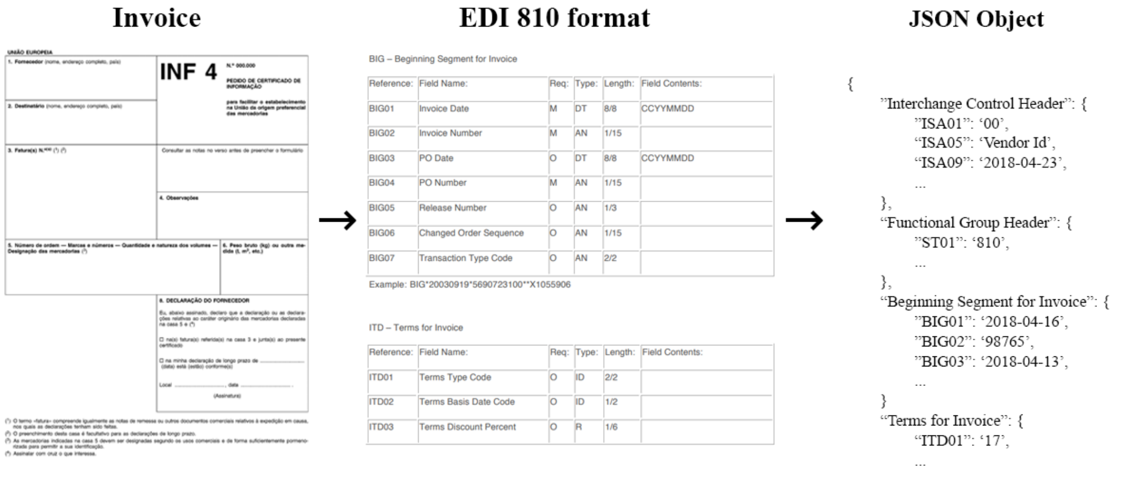
Implementation and Results

In this chapter, the technical implementation of the concepts discussed in the previous chapters is described. The main pratical considerations that were taken have to do with the off-chain/on-chain data storage duality, the physical to digital translation of documents and the microservice oriented mindset of the case study. The focus of this project on the outbound stage of logistics, coupled with the on-boarding of one stakeholder, a major carrier, allowed for a narrow but insightful view of the possible results of deploying this project to production. The measurements and results come from real data, however they are just illustrative, as the project and technology is yet too immature to be used by government entities, where it would have a more significant positive impact.

5.1 Web API and Blockchain Interface

As proposed by the literature, using a standard EDI format to convert non-standard documents into simple *.json* format that can be easily interpreted and relatable to chaincode models. A problematic document that raised the majority of concerns for the carrier was the invoice, a document that is generated by the client-brand, passed on to HUUB, then the carrier, then customs and finally the end-customer. Figure 5.1 exemplifies change that would take place, that is, instead of generating a printable *.pdf* document, the invoice would be generated in *.json* format and written to the blockchain.

A second technical aspect of the implementation pertains to the idealization that this project would serve as a humanless backend support infrastructure. In other words, that there would be no human interaction needed, and that the relation between off-chain data, that is in fact inserted manually by office workers, warehouse operators, couriers, etc., and on-chain data would be supported by IT systems. In practic terms, this means that there is a connection established between off-chain databases and the blockchain. There were two components that directly affected how this support structure was: the company's database is managed with PostgreSQL and the SDK for blockchain interaction is written in Javascript in a NodeJS environment. The solution was to use Sequelize, a Object-relational mapping (ORM) tool for NodeJS. Fabric's NodeJS SDK and Sequelize rely on



Invoice

UNIÃO EUROPEIA

1. Fornecedor (nome, endereço completo, país)	INF 4 N.º 000.000 PEDIDO DE CERTIFICADO DE INFORMAÇÃO para facilitar o estabelecimento na União da origem preferencial das mercadorias
2. Destinatário (nome, endereço completo, país)	
3. Fatura(s) N.º(s) (1) (2)	Consultar as notas no verso antes de preencher o formulário
	4. Observações
5. Número de ordem — Marcas e números — Quantidade e natureza dos volumes — Designação das mercadorias (3)	6. Peso bruto (kg) ou outra medida (l, m³, etc.)
	8. DECLARAÇÃO DO FORNECEDOR Eu, abaixo assinado, declaro que a declaração ou as declarações relativas ao caráter originário das mercadorias declaradas na casa 5 e (4) □ na(s) fatura(s) referida(s) na casa 3 e junta(s) ao presente certificado □ na minha declaração de longo prazo de .... (data) está (estão) conforme(s) Local ...., data .... (Assinatura)

(1) O termo «fatura» compreende igualmente as notas de remessa ou outros documentos comerciais relativos à expedição em causa, nos quais as declarações tenham sido feitas.
(2) O preenchimento desta casa é facultativo para as declarações de longo prazo.
(3) As mercadorias indicadas na casa 5 devem ser designadas segundo os usos comerciais e de forma suficientemente pormenorizada para permitir a sua identificação.
(4) Assinalar com cruz o que interessa.

→

EDI 810 format

BIG – Beginning Segment for Invoice

Reference:	Field Name:	Req:	Type:	Length:	Field Contents:
BIG01	Invoice Date	M	DT	8/8	CCYYMMDD
BIG02	Invoice Number	M	AN	1/15	
BIG03	PO Date	O	DT	8/8	CCYYMMDD
BIG04	PO Number	M	AN	1/15	
BIG05	Release Number	O	AN	1/3	
BIG06	Changed Order Sequence	O	AN	1/15	
BIG07	Transaction Type Code	O	AN	2/2	

Example: BIG*20030919*5690723100**X1055906

ITD – Terms for Invoice

Reference:	Field Name:	Req:	Type:	Length:	Field Contents:
ITD01	Terms Type Code	O	ID	2/2	
ITD02	Terms Basis Date Code	O	ID	1/2	
ITD03	Terms Discount Percent	O	R	1/6	

→

JSON Object

```
{
    ”Interchange Control Header”: {
        ”ISA01”: ‘00’,
        “ISA05”: ‘Vendor Id’,
        “ISA09”: ‘2018-04-23’,
        ...
    },
    “Functional Group Header”: {
        ”ST01”: ‘810’,
        ...
    },
    “Beginning Segment for Invoice”: {
        ”BIG01”: ‘2018-04-16’,
        ”BIG02”: ‘98765’,
        ”BIG03”: ‘2018-04-13’,
        ...
    }
    “Terms for Invoice”: {
        “ITD01”: ‘17’,
        ...
```

Figure 5.1: How an invoice would morph from a *.pdf* document to a *.json* object through the EDI-810 standard. Sources: EUR-LEx (2017) and Direct Commerce (2014)

promises, that essentially associate a success or failure response to an asynchronous requests, allowing for asynchronous methods to be called synchronously. A sequence diagram that succinctly describes this flow of assynchronous/synchronous requests is illustrated in Figure 5.2. The label document is usually requested by the SCM company when the package is being prepared for shipping. It's a document that is exchanged between stakeholders and that benefits from being written to the blockchain for the numerous reasons stated in Chapter 4. However, this example exposes a situation where blockchain doesn't fully infringe on the current process. While the traditional way of requesting a label is still being used (steps 4 through 6), additional blockchain interactions, (steps 2, 3, 7 and 8) assure that future stakeholder access to the documents is available and the document is verifiable, without the need for physical signatures.

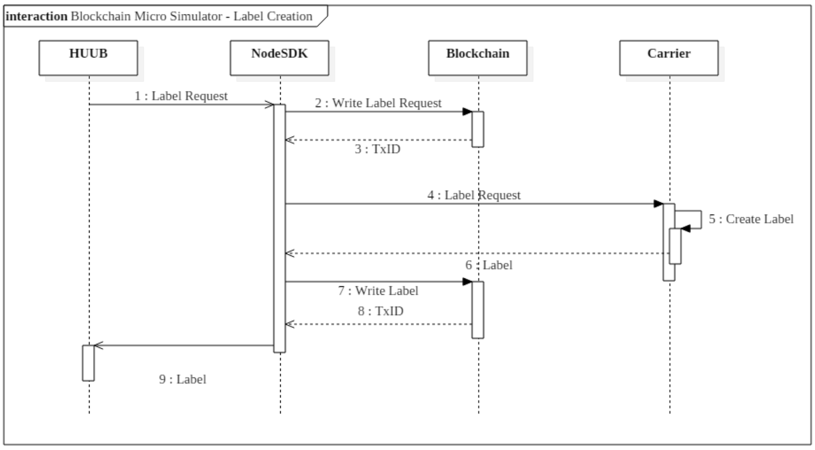


Figure 5.2: Sequence diagram of flow of requests to write to the blockchain, request a document from a stakeholder and incorporate the data in the blockchain and the off-chain database, for later use

The NodeSDK has also the functionality of issuing cryptographic material and validating users, the process is similar to what is described before in Section 4.2.3, but here instead of having cryptographic keys and certificates associated with the organization directly, the files are associated with a user that can act as a member of that organization. An overall view of all the implementation caveats is presented in Figure 5.3.

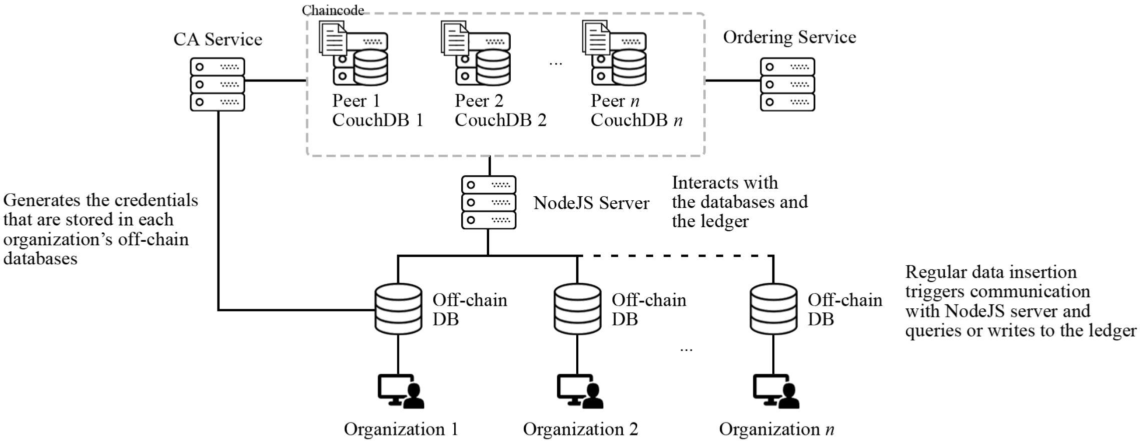

Chaincode
CA Service
Peer 1
CouchDB 1
Peer 2
CouchDB 2
...
Peer n
CouchDB n
Ordering Service
Generates the credentials that are stored in each organization's off-chain databases
NodeJS Server
Interacts with the databases and the ledger
Off-chain DB
Off-chain DB
Off-chain DB
...
Regular data insertion triggers communication with NodeJS server and queries or writes to the ledger
Organization 1
Organization 2
Organization n

Figure 5.3: Proposed system architecture for SCM inter-organization blockchain application

For development purposes, a simple user interface was developed that allows the process to be clearly identified, even though the production scenario wouldn't require such UI, as the process would happen in the background. It is presented in Appendix A.

5.2 Data collection

In order to measure the potential gains that could arise from implementing a solution similar to what is proposed in this thesis, a major carrier, that shall remain unnamed for confidentiality requirements, was approached to gather some relevant values. Since export was deemed to be the flow that experiences the higher number of delays due to mishandling of data, the research had an emphasis on exported goods. The data includes:

Document idle time Time between the drop-off of packages and respective documentation at the carrier's facilities and pick-up of said documents by the department that handles document preparation for Customs.

Data: Varies between 5 and 10 minutes.

Document handling time The time it takes for a worker to manually verify that the information on a document is correct, namely for invoice with several products that require cross checking between the document data and government data tables.

Data: Varies between 1 and 20 minutes.

Goods "trapped" at Customs Number of packages stuck at Customs' facilities due to lack of documents, wrong documents, correct documents with misinformation per day. The main reason was noted to be lack of information in the correct documents, and not the other 2 possibilities.

Data: Varies between 0 (off-peak) and 5 (peak season) packages.

Resolution time Time between the communication of the problem mentioned above and the resolution by providing the lacking or correct information.

Data: Varies between 1 and 7 days.

Number of communications How many emails are exchanged between the carrier and the SCM company to achieve a valid set of package and documents that can be cleared by Customs.

Data: Varies between 1 and 7 emails.

Returns Number of return per month and the time allocated for documental and physical verification.

Data: Varies between 0 and 5 returns per month with approximately 2 hours per returned package.

5.3 Results

Because this analysis was restricted to a small segment of the enormous chain of processes that SCM entails, it is difficult to quantify the full impact of such a solution on the entire supply chain. Even with such a minute portion of data, some of the gains are more easily assessed than others. Regardless, with the available data and simulations run on the developed prototype, the following assertions can be made:

- The document handling time is reduced to less than 5 seconds, but it is also important to note that computers are able to process several documents in parallel with several threads.
- The rate of incidence of lack of information is reduced to 0, because data insertion at the moment of creation would require validation. An EDI-810 invoice with insufficient information would flag a warning and writing to the blockchain would not occur.
- For the above stated reason, the number of withheld goods would potentially decrease.
- The interval of time that packages are stuck at Customs waiting for clearance would diminish, as the delay between communications of errors would be reduced. Communications would be broadcasted from the organization that detects the error directly to the concerning organizations, and not sequentially.
- Document validation related to returns, namely proof of previous export, would also be expedited because the information is verifiable and available on the network. However the

physical inspection is a mandatory process that happens simultaneously and is not subject to any modification. The process would take the same time, despite the decrease of documental weight.

5.3.1 Direct costs

The direct cost reduction can be estimated with the results obtained regarding the quantities that concern HUUB and for overall cost reduction related to the total volume the carrier has an extrapolation can be made, assuming their processes are homogeneous and independent of the customer. To estimate the expenditure decrease in manual labour, four assumptions were made: the average labour cost in Portuguese industry is 11.6€/hour (Eurostat, 2017), the average time lies between the values (1 minute and 20 minutes) obtained from the carrier, 10 minutes and 30 seconds, the average electricity cost per kWh in Portugal is 0.14€ (PORDATA, 2017) and the computer power consumption is 220W.

Table 5.1: Cost comparison between current manual labour and blockchain-enabled computer processing for package validation by a carrier

	Manual Labour		Blockchain-enabled Network	
	Avg. Time	Avg. Cost	Avg. Time	Avg. Cost
Per package	10m30s	€ 2.03	5s	€ 0.00004
Per 100 packages	17h30m	€ 203.00	8m20s	€ 0.0043
Per 1000 packages	175h00m	€ 2,030.00	1h23m	€ 0.0428

5.3.2 Indirect costs

The indirect costs, that are also reduced by implementing the proposed technologies and flows of operation, are virtually impossible to quantify at a such a small scale. Although there's no suitable way to measure them, there are implicit cost reductions that are pervasive to the adopting firms. Namely, paper and printer ink, by eliminating the need for printing documents, something that is inherent to process digitization, and labour hours related to communication and validations, since messages are not relayed between parties, but rather transmitted directly to the interested party and data insertion is standardized and monitored from the start.

5.3.3 Discussion

These results are merely demonstrative and should not serve as a basis to extrapolate economic gains from implementing blockchain technology in SCM. The calculations presented in Section 5.3.1 are limited to a crucial yet small process in a long chain of tasks performed by very disparate entities. The effects of migrating to a blockhain-enabled supply chain network will heavily depend on its complexity and length. A leading international carrier that delivers a volume of upwards of 5 billion items is only one node in several supply chains that contain more companies with similarly high annual throughput. The longer the supply chain is, the higher the necessity for information sharing. The studied processes exemplify this effect perfectly: information that is created at one moment is transferred from stakeholder to stakeholder until it reaches a stakeholder that might invalidate said information, a point where it then is transferred backwards. Again, the cost cutbacks associated with the elimination of these unnecessary backtracks are more noticeable when volume is higher and more stakeholders are involved.

The outcome of these tests, however, corroborate the disruptive nature of the technology. Even though the illustrative, almost anecdotal, calculations account for nearly 100% improvements, a resemblance to another technology and its unarguable disruption can be identified: e-mailing. Before e-mail, letters were written, put in an envelope with an address, handed over to the post-office where a myriad of workers would ensure its eventual arrival at the destination. E-mail turned a process that involved several people across multiple days, depending on distance, to a few seconds of digital exchange.

No comparison should be needed between the blockchain enabled network and a central database provided by a third-party. As presented in Chapter 2, centralized databases' performance is indisputably better than a decentralized one, but all is for naught if there is not a trustless mechanism in place that guarantees the veracity of information stored.

Chapter 6

Conclusions

The scope of this thesis, even though closely related to SCM, does not have as much of a particular focus on optimizing operation planning, for example, as on leveraging an emerging technology to expedite existing overarching processes. The company where this project was developed, HUUB, was a stellar candidate for a case study on how to leverage blockchain technology for two reasons: HUUB acts as a connector between brands and their supply chain and manages everything from the moment a product leaves the manufacturing plant until it reaches the final customer, dealing with hundreds of stakeholders; and secondly, the internal microservice architecture and focus on software development provide a good fit for the implementation of new technologies.

As an action-driven research thesis, the first clear objective was to map what has been done with blockchain technology, with the intent of filtering the so called "hype" that has flooded the internet. Among the existing blockchain projects, Bitcoin and Ethereum could not have been left out simply for their pioneering traits. Yet, there have been great developments in recent years, and open-source projects with improved features have sprouted, that inherently tackle a lot of the problems of Bitcoin and Ethereum posed by other authors, like data standardization, entities' permissions, business logic translation and cost. Hyperledger Fabric proved to be the most relevant framework for its business emphasis, semi-centralized nature, that allows for accountability and higher performance, when compared to fully decentralized, and robust documentation alongside popular programming languages. With the findings, the proposition is such that Blockchain enters the supply chain landscape not as a "full-on" central database substitute, but rather as a tool that allows processes to transform from manual tasks, that delay necessary operations, into digital, unmanned, background processes that require significant less labour to operate.

The umbrella project, Hyperledger, for the chosen framework, Fabric, has a few layers of abstraction available to ease the deployment of business blockchain networks, but to fully understand the caveats of the technology, none were used (e.g. Hyperledger Composer and Cello). Howbeit, the NodeJS SDK was used for the development of a Web API. The framework can handle multiple business configurations easily, with several stakeholders that have different access permissions; changes in company relationships that demand revoking or granting membership to the network, and changes in business processes, like new document standards or new operations along the supply chain; and interaction through the SDK is a definitely appreciated method of writing and querying the ledger.

With the issues posed in the literature and the business where HUUB finds itself, the solution that seemed to make more sense was, again, not to replace current storage systems in every stakeholder, but provide a central hub of shared data that can be continually accessed, alongside the ongoing, efficient, internal data silos. Standard digital document formats, like EDI-810 for invoices, and API endpoints equip the network a set of rules that guarantee access to data, that has to be necessarily transferred between stakeholders, and easier integration with the internal data structures of each firm. The resulting product is a ledger that supports the accountability required by law and the advantages of turning a manual process into a automated digital set of validations that take seconds to perform, rather than hours.

The results obtained are limited in size but demonstrate, in a meaningful way, how the technology can cut down costs from thousands of dollars to a few cents, in a very specific validation task, and it can be argued that similar processes along the supply chain can experience the same gains. A key takeway is that the disruptive nature of blockchain is only possible with a big enough network. As mentioned in previous chapters, 2 entities that share information bilaterally have no need for blockchain technology. But this case study evidences that there is a great deal of information asymmetry, with data being scattered along the supply chain. Even for a small company like HUUB, the degree to which blockchain can aid in information sharing, in a secure, verifiable way, is clear. Moreover, business logic and legal requirements don't pose such a disproportional barrier to implementing a viable solution. They serve as guidelines and functional prerequisites that can easily be translated into code with the support of Hyperledger Fabric's framework.

This project remains as an experiment where an actual proof-of-concept product was developed, and as such there are a few caveats that would need further development if a production-ready service was to be launched. The authentication method in a Fabric network requires that each entity be the sole holder of the certificates and keys, which can only be achieved through the implementation of Docker Swarm. In the development and research environment of this thesis, this could not be accomplished due to time constraints. However, the results do not suffer in validity, as Docker Swarm acts as a mere orchestrator for different Docker containers on separate physical machines, and doesn't directly affect the ledger's operations and configurations.

As a final note, a few targets remain open for research in the field of blockchain applied to SCM that can deeply impact the logistics landscape. This thesis tackled a small portion of a rather small supply chain (when compared with automotive supply chains, for example), with restricted stakeholder interactions. The potential for disruption of blockchain can only unlocked if large-scale tests invoving several entities, private and governmental, migrate from the ethereal plane they are at currently, with incredible yet ambiguous results documented in news articles and blog posts, to well documented research papers. Furthermore, from the company interactions with external parties and its internal operations, there is good reason to believe that IoT in conjuction with blockchain would output better results than if both technologies were to be implemented separately. Mainly for the obvious reason of human error elimination and the higher recognition speeds of sensors compared to humans. This would go hand in hand with the proposed structure of background humanless tasks. "Disruption" is perhaps an overused term when discussing blockchain, but with a complex technology like blockchain, that is not trivial to understand, its disruptive nature can only become a reality in SCM through a collaborative and well documented effort from a large number of companies.

Bibliography

Akkoyunlu, E. A., Ekanadham, K., and Huber, R. V. (1975). Some constraints and tradeoffs in the design of network communications. *SIGOPS Oper. Syst. Rev.*, 9(5):67–74.

Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., De Caro, A., Enyeart, D., Ferris, C., Laventman, G., Manevich, Y., Muralidharan, S., Murthy, C., Nguyen, B., Sethi, M., Singh, G., Smith, K., Sorniotti, A., Stathakopoulou, C., Vukolić, M., Cocco, S. W., and Yellick, J. (2018). Hyperledger fabric: A distributed operating system for permissioned blockchains. In *Proceedings of the Thirteenth EuroSys Conference*, EuroSys '18, pages 30:1–30:15, New York, NY, USA. ACM.

Apte, S. and Petrovsky, N. (2016). Will blockchain technology revolutionize excipient supply chain management? *Journal of Excipients and Food Chemicals*, 7(3).

Baliga, A. (2017). Understanding Blockchain Consensus Models. *Persistent Systems Ltd.*

Blechschmidt, B. and Stöcker, D. C. (2017). How Blockchain Can Slash the Manufacturing "Trust Tax". *Cognizant.*

Bocek, T., Rodrigues, B. B., Strasser, T., and Stiller, B. (2017). Blockchains everywhere - A use-case of blockchains in the pharma supply-chain. In *Proceedings of the IM 2017 - 2017 IFIP/IEEE International Symposium on Integrated Network and Service Management*, pages 772–777.

Buterin, V. (2014a). A next-generation smart contract and decentralized application platform. *Ethereum*, pages 1–36.

Buterin, V. (2014b). Toward a 12-second Block Time. *Ethereum Blog*. Accessed on October 25, 2017. https://blog.ethereum.org/2014/07/11/toward-a-12-second-block-time/.

Campbell, Rebecca (2016). Modum.io's Tempurature-Tracking Blockchain Solution Wins Accolades at Kickstarter Accelerator 2016. Accessed on March 16, 2018. https://bitcoinmagazine.com/articles/modum-io-s-tempurature-tracking-blockchain-solution-wins-accolades-at-kickstarter-accelerator-1479162773.

Chen, S., Shi, R., Ren, Z., Yan, J., Shi, Y., and Zhang, J. (2017). A Blockchain-Based Supply Chain Quality Management Framework. In *2017 IEEE 14th International Conference on e-Business Engineering (ICEBE).*

Chronicled (2017). Pharma Companies Tap Startups to Develop Protocol for Tracking and Verifying Prescription Drugs using Blockchain. Accessed on March 16, 2018. i.

Del Castillo, Michael (2017). Intel Demos Seafood Tracking on Sawtooth Lake Blockchain. Accessed on March 16, 2018. https://www.coindesk.com/intel-demos-seafood-tracking-sawtooth-lake-blockchain/.

Direct Commerce (2014). Electronic Invoicing Guide. Accessed on May 8, 2018. http://www.directcommerce.com/wp-content/uploads/2014/10/EDI-810-File-Format2.pdf.

Docker Inc. (2018). What is a container. Accessed on March 6, 2018. https://www.docker.com/what-container.

Ethereum Community (2014). Ethereum Wiki - Design Rationale. Accessed on March 12, 2018. https://github.com/ethereum/wiki/wiki/Design-Rationale.

Ethereum Community (2017). Ethereum Homestead Documentation. *Github*. http://www.ethdocs.org/en/latest/.

EUR-LEx (2017). REGULAMENTO DE EXECUÇÃO (UE) 2015/2447 DA COMISSÃO de 24 de novembro de 2015. Accessed on May 8, 2018. https://eur-lex.europa.eu/legal-content/PT/TXT/?uri=CELEX{%}3A02015R2447-20170614.

Eurostat (2017). Hourly labour costs in euro, breakdown by economic activity in 2017. Accessed on June 15, 2018. http://ec.europa.eu/eurostat/statistics-explained/index.php?title=File:Hourly_labour_costs_in_euro,_breakdown_by_economic_activity_in_2017_CORR.png.

Goswami, S. (2017). Scalability Analysis of Blockchains Through Blockchain Simulation. *UNLV Theses, Dissertations, Professional Papers, and Capstones*, 2096.

Greenspan, G. (2015). Avoiding the pointless blockchain project. *Multichain*. Accessed on October 24, 2017. http://www.multichain.com/blog/2015/11/avoiding-pointless-blockchain-project/.

Greenspan, G. (2016). Blockchains vs centralized databases. *Multichain*, pages 1–6. Accessed on October 24, 2017. https://www.multichain.com/blog/2016/03/blockchains-vs-centralized-databases/.

Hammer, M. (1990). Reengineering Work: Don't Automate, Obliterate. *Harvard Business Review*, 68(4):104–112.

Hancock, M. and Vaizey, E. (2016). Distributed ledger technology: Beyond block chain. Technical report, Government Office for Science.

Hearn, M. (2016). Corda: A distributed ledger. *Whitepaper*, pages 1–56.

Higgins, Stan (2017a). Defense Giant Lockheed Martin Integrates Blockchain. Accessed on March 16, 2018. https://www.coindesk.com/defense-giant-lockheed-martin-integrates-blockchain/.

Higgins, Stan (2017b). Walmart: Blockchain Food Tracking Test Results Are 'Very Encouraging'. Accessed on March 16, 2018. https://www.coindesk.com/walmart-blockchain-food-tracking-test-results-encouraging/.

Hyperledger (2017). Hyperledger Fabric v1.1 - Architecture Reference. Accessed on March 9, 2018. http://hyperledger-fabric.readthedocs.io/en/release-1.1/architecture.html.

Hyperledger Architecture Working Group (2017). Hyperledger Architecture , Volume 1. *hyperledger.org*, 1.

Iansiti, M. and Lakhani, K. R. (2017). Harvard Business Review - The Truth About Blockchain. *Harvard Business Review*, 95(1):118–128.

Intel Corporation (2015). Architecture - Sawtooth v1.0.1 Documentation. *sawtooth.hyperledger.org*.

IOTA Support (2017). An introduction to IOTA. Accessed on March 14, 2018. https://iotasupport.com/whatisiota.shtml.

James-Lubin, Kieren (2015). Blockchain scalability. Accessed on March 12, 2018. https://blockgeeks.com/guides/blockchain-scalability/.

JP Morgan Chase (2016). Quorum Whitepaper. *https://github.com/jpmorganchase/quorum-docs/blob/master/QuorumWhitepaperv0.1.pdf*.

Kim, H. M., Gruninger, M., and Fox, M. S. (1995). An ontology of quality for enterprise modelling. In *4th Workshop on Enabling Technologies, Infrastructure for Collaborative Enterprises (WET-ICE'95), April 20-22, 1995, Berkeley Springs, West Virginia, USA, Proceedings*, pages 105–116.

Kim, H. M. and Laskowski, M. (2016). Towards an Ontology-Driven Blockchain Design for Supply Chain Provenance. *SSRN Electronic Journal*.

Kolodny, Lora (2017). Bext360 is using robots and the blockchain to pay coffee farmers fairly. Accessed on March 16, 2018. https://techcrunch.com/2017/04/11/bext360-is-using-robots-and-the-blockchain-to-pay-coffee-farmers-fairly/?guccounter=1.

Korpela, K., Hallikas, J., and Dahlberg, T. (2017). Digital Supply Chain Transformation toward Blockchain Integration. In *Proceedings of the 50th Hawaii International Conference on System Sciences*.

Kshetri, N. (2018). 1 Blockchain's roles in meeting key supply chain management objectives. In *International Journal of Information Management*.

Lambert, D. M., Cooper, M. C., and Pagh, J. D. (1998). Supply chain management: Implementation issues and research opportunities. *The International Journal of Logistics Management*, 9(2):1–20.

Lechmere, Adam (2016). Wine Vault Offers Security in a Digital Age. Accessed on March 16, 2018. https://www.wine-searcher.com/m/2016/12/wine-vault-offers-security-in-a-digital-age.

Levy, E. and Silberschatz, A. (1990). Distributed file systems: concepts and examples. *ACM Computing Surveys*, 22(4):321–374.

Mathieson, SA (2017). Blockchain starts to prove its value outside of finance. Accessed on March 16, 2018. https://www.computerweekly.com/feature/Blockchain-starts-to-prove-its-value-outside-of-finance.

Mcconaghy, T., Marques, R., Müller, A., De Jonghe, D., Mcconaghy, T., Mcmullen, G., Henderson, R., Bellemare, S., and Granzotto, A. (2016). BigchainDB: A Scalable Blockchain Database (DRAFT). *BigchainDB*, pages 1–65.

Min, X., Li, Q., Liu, L., and Cui, L. (2016). A permissioned blockchain framework for supporting instant transaction and dynamic block size. In *2016 IEEE Trustcom/BigDataSE/ISPA*, pages 90–96.

Modum.io AG (2016). Modum Whitepaper. *www.modum.io*.

Nakamoto, S. (2008). Bitcoin: A Peer-to-Peer Electronic Cash System. *www.bitcoin.org*, pages 1–9.

Nielsen, P. M. (2017). Raft-based consensus for Ethereum/Quorum.

Paul, M. S. (2018). Hyperledger — Chapter 2 | Hyperledger Frameworks & Modules.

Pilkington, M. (2015). Blockchain Technology: Principles and Applications. *Research Handbook on Digital Transformations*, pages 1–39.

Popov, S. (2017). The Tangle. *https://www.iotatoken.com/IOTA_Whitepaper.pdf*.

Popper, Nathaniel and Lohr, Steve (2017). Blockchain: A Better Way to Track Pork Chops, Bonds, Bad Peanut Butter? Accessed on March 16, 2018. https://www.nytimes.com/2017/03/04/business/dealbook/blockchain-ibm-bitcoin.html.

PORDATA (2017). Preços da electricidade para utilizadores domésticos e industriais (Euro/ECU). Accessed on June 15, 2018. https://www.pordata.pt/Europa/Preços+da+electricidade+para+utilizadores+domésticos+e+industriais+(Euro+ECU)-1477.

Provenance (2016). From shore to plate: Tracking tuna on the blockchain. Accessed on March 16, 2018. https://www.provenance.org/tracking-tuna-on-the-blockchain.

Rimba, P., Tran, A. B., Weber, I., Staples, M., Ponomarev, A., and Xu, X. (2017). Comparing Blockchain and Cloud Services for Business Process Execution. In *Proceedings - 2017 IEEE International Conference on Software Architecture, ICSA 2017*.

Ripple (2018). Ripple's Website. *https://www.ripple.com/*.

Scherer, M. and Eriksson, J. (2017). Performance and Scalability of Blockchain Networks and Smart Contracts. *UMEA University*.

Schrauf, S. and Berttram, P. (2016). How digitization makes the supply chain more efficient, agile, and customer-focused. *Strategy&*.

Schwartz, D., Youngs, N., and Britto, A. (2014). The Ripple Protocol Consensus Algorithm. *Ripple Labs Inc White Paper*.

Scott, Michael (2017). Innovation Percolates When Coffee Meets the Blockchain. Accessed on March 16, 2018. https://www.nasdaq.com/article/innovation-percolates-when-coffee-meets-the-blockchain-cm774790.

ShipChain (2017). ShipChain - Whitepaper.

Start-up Ticker (2017). Modum and University of Zurich team up on blockchain research project. Accessed on March 16, 2018. https://www.startupticker.ch/en/news/october-2017/modum-and-university-of-zurich-team-up-on-blockchain-research-project.

Tas, Bindi (2017). Alibaba and AusPost team up to tackle food fraud with blockchain. Accessed on March 16, 2018. https://www.provenance.org/tracking-tuna-on-the-blockchain.

Travis, M. (2017). Ripple: The Most (Demonstrably) Scalable Blockchain. *High Scalability Blog*. Accessed on March 14, 2018. http://highscalability.com/blog/2017/10/2/ripple-the-most-demonstrably-scalable-blockchain.html.

Tuan, T., Dinh, A., Liu, R., Zhang, M., Chen, G., Ooi, B. C., and Wang, J. (2017). Untangling Blockchain: A Data Processing View of Blockchain Systems. *CoRR*.

Vukolić, M. (2016). The quest for scalable blockchain fabric: Proof-of-work vs. bft replication. In Camenisch, J. and Kesdoğan, D., editors, *Open Problems in Network Security*, pages 112–125, Cham. Springer International Publishing.

Weber, I., Xu, X., Riveret, R., Governatori, G., Ponomarev, A., and Mendling, J. (2016). Untrusted Business Process Monitoring and Execution. In *International Conference on Business Process Management*, pages 329–347.

Wu, H., Li, Z., King, B., Miled, Z. B., Wassick, J., and Tazelaar, J. (2017). A distributed ledger for supply chain physical distribution visibility. *Information (Switzerland)*.

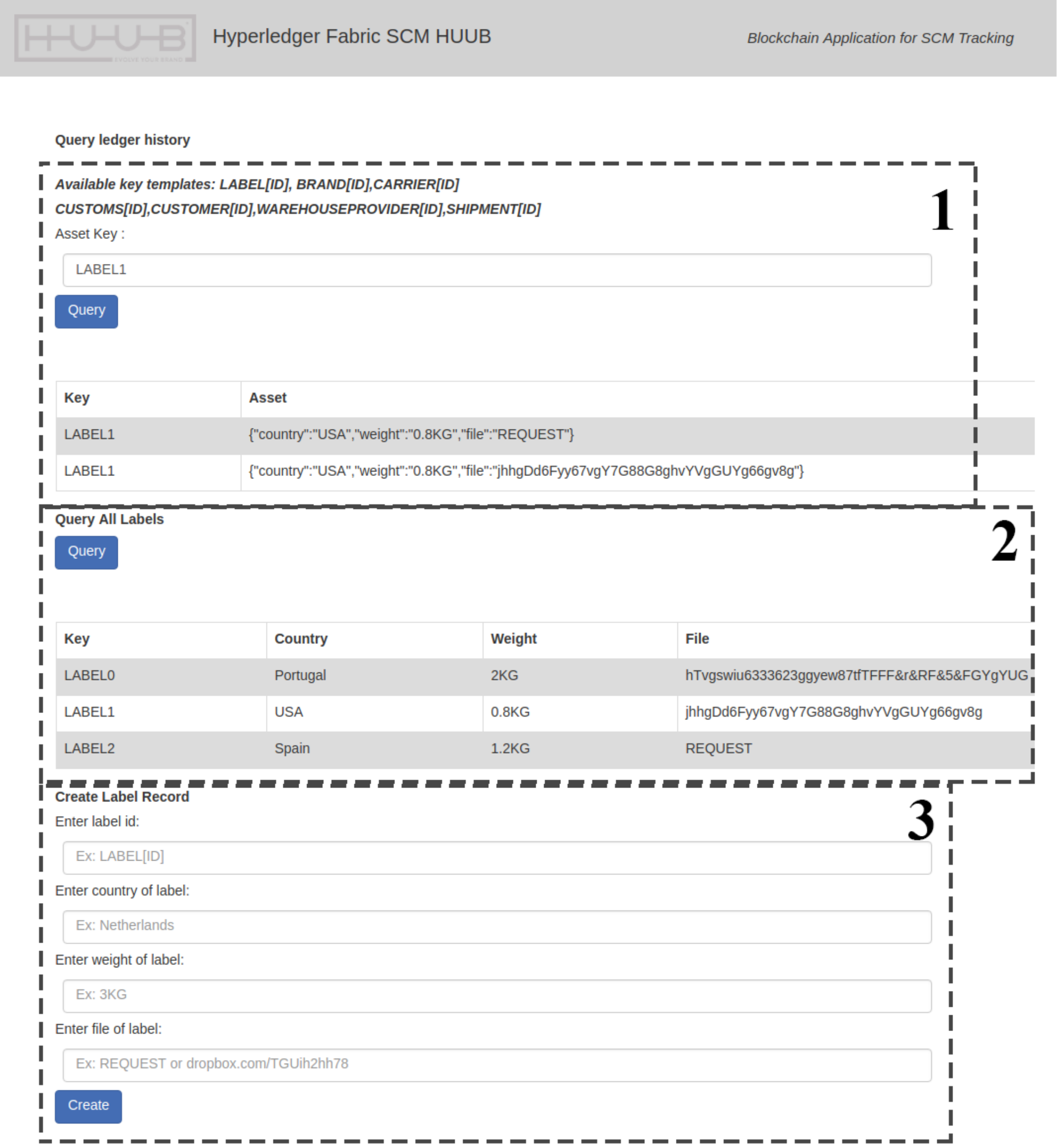
Wu, K. (2016). Startups in Supply Chain and Logistics - Investments, acquisitions, and trends to watch.

Appendix A

Auxiliary interface for ledger interaction

This interface was built on top of the examples provided by IBM and Hyperledger, with the distinct feature of interacting with off-chain databases. The manual input via text boxes is only for development purposes, as insertions would ideally be made automatically.

Section 1 of the interface illustrates one of the main features of blockchain: immutability of records. An asset can go through several states, for example, a label goes from requested, to created, to signed. The asset history is stored in the peers and is not editable. This allows an asset to be rectified in case of incorrect information, but allows users to verify any past alteration. Section 2 is a simple query that returns the current state of all labels. Section 3 conceals a slightly more complex process than the other sections, which execute commands to the NodeJS SDK server directly. This input performs the steps described in Section 5.1, by first writing to a local PostgreSQL database, that then triggers the asynchronous events that will write the request in the ledger, request a label from another entity and write the requested label to the ledger. As a side note, the "file" field is a test field where a series of mathematical functions write the result of some arbitrary operations to simulate computational time on the other entity's side.



Hyperledger Fabric SCM HUUB

Blockchain Application for SCM Tracking

Query ledger history

Available key templates: LABEL[ID], BRAND[ID],CARRIER[ID] CUSTOMS[ID],CUSTOMER[ID],WAREHOUSEPROVIDER[ID],SHIPMENT[ID]

Asset Key : LABEL1

Key	Asset
LABEL1	{"country":"USA","weight":"0.8KG","file":"REQUEST"}
LABEL1	{"country":"USA","weight":"0.8KG","file":"jhhgDd6Fyy67vgY7G88G8ghvYVgGUYg66gv8g"}

Query All Labels

Key	Country	Weight	File
LABEL0	Portugal	2KG	hTvgswiu6333623ggyew87tfTFFF&r&RF&5&FGYgYUG
LABEL1	USA	0.8KG	jhhgDd6Fyy67vgY7G88G8ghvYVgGUYg66gv8g
LABEL2	Spain	1.2KG	REQUEST

Create Label Record

Enter label id: Ex: LABEL[ID]

Enter country of label: Ex: Netherlands

Enter weight of label: Ex: 3KG

Enter file of label: Ex: REQUEST or dropbox.com/TGUih2hh78

Figure A.1: Graphical interface for writing and querying the blockchain ledger