Smart Contracts and their Application in Supply Chain Management

by

Angwei Law

B.S. Biomedical Engineering
University of Wisconsin-Madison, 2010

Submitted to the System Design and Management Program
in Partial Fulfillment of the Requirements for the Degree of

Master of Science in Engineering and Management

at the

Massachusetts Institute of Technology

September 2017

Signature of Author Signature redacted
System Design and Management Program
August 11, 2017

Certified by Signature redacted
Abel Sanchez
Executive Director, MIT Geospatial Data Center
Thesis Supervisor

Accepted by Signature redacted
Joan Rubin
Executive Director, System Design and Management Program

This page is intentionally left blank.

Smart Contracts and their Application in Supply Chain Management

by

Angwei Law

Submitted to the System Design and Management Program on August 11, 2017
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Engineering and Management

Abstract

Blockchain is a distributed ledger technology that records exchanges of value between parties securely, permanently, and in an easily verifiable manner. It is the technology underlying cryptocurrencies such as Bitcoin and Ethereum. Although initially used for financial transactions, blockchain's applications extend beyond finance and can impact a wide variety of industries. One such application is in supply chain management. With multiple stakeholders and business transactions, supply chains are inherently complex. Many challenges exist, including the lack of transparency and traceability, difficulty in managing risks and disruptions, and the need to build trust and reputation. Blockchain technology ushers in the potential to solve these challenges through the use of smart contracts. Smart contracts are digital agreements between transacting parties that are written in computer code and deployed to the blockchain, where they will self-execute when predetermined conditions are met. They reduce complexity in a supply chain through automated verification and execution of the multiple business transactions involved. A decentralized, immutable record also ensures all stakeholders have equal access to information and helps build trust. Smart contracts improve the transparency, traceability and efficiency of a supply chain, allowing it to be more agile while strengthening relationships among stakeholders.

In this thesis, I create a proof-of-concept to explore the application of smart contracts in supply chain management. The proof-of-concept consists of three smart contracts, coded in Solidity, that can be integrated to determine the provenance of goods, track the chain of custody as goods flow through a supply chain, automatically execute payment upon fulfillment of criteria, and maintain an open database of stakeholders with a score indicating their reputation. I validate the proof-of-concept using the Ethereum platform, which was specially conceived for smart contract and decentralized application development. Preliminary testing shows that the contracts are able to interact with one another and execute their functions as intended. Further testing is necessary to evaluate the performance of the contracts on the live Ethereum network, and integration with smart sensors should be explored to create a more viable real-world solution.

Thesis Supervisor: Abel Sanchez
Title: Executive Director, MIT Geospatial Data Center

This page is intentionally left blank.

Acknowledgments

The past year has been an incredibly enriching life experience for me, both in the academic and personal sense. It is never easy leaving family and friends to live on the other side of the world for an extended period of time. There were challenges, but thankfully more ups than downs. The diversity, innovativeness and drive that I experienced at MIT were unparalleled, and I thoroughly enjoyed the opportunity to immerse myself in its collective brilliance. Of course, none of this would have been possible without the support that I received.

I would like to start by thanking Dr. Abel Sanchez, without whom this thesis would not have been possible. In spite of his busy schedule, he took the time to provide clear guidance and shape my ideas along the way. It was also his course on JavaScript and HTML, co-taught with Prof. John R. Williams, that piqued my interest in computer programming in general. Their enthusiasm and dynamism made learning a joy, and I was encouraged to explore a coding-related topic for my thesis despite not having much experience with it.

I am also grateful to the Singapore government and the SDM program for giving me this opportunity to develop myself both personally and professionally. The SDM faculty and staff have shown great dedication towards making our experience in the program a fulfilling and pleasant one. The core syllabus, though rigorous, provided a good platform to develop my systems thinking skills and work in diverse teams. I thank my fellow classmates in SDM, Sloan, and the School of Engineering for broadening my perspectives through our interactions, both in and out of class.

Last but most definitely not least, I would like to thank my parents, fiancée, and friends outside of MIT for their never-ending support and encouragement throughout the past year. It heartens me to know that they are always just a message or call away, and that they are always there to share in my joy and frustration. Their belief and motivation have given me focus, built my confidence and driven me to achieve my best in everything that I do.

This page is intentionally left blank.

Table of Contents

List of Figures

List of Abbreviations

API	Application Programming Interface
B2B	Business-to-Business
BaaS	Blockchain-as-a-Service
D.A.O.	Decentralized Autonomous Organization
EDI	Electronic Data Interchange
EOA	Externally Owned Account
EOQ	Economic Order Quantity
EVM	Ethereum Virtual Machine
Fintech	Financial Technology
GSCF	Global Supply Chain Forum
IDE	Integrated Development Environment
IoT	Internet of Things
M2M	Machine-to-Machine
NGO	Non-Governmental Organization
OEM	Original Equipment Manufacturer
OTC	Over-The-Counter
P2P	Peer-to-Peer
PoC	Proof-of-Concept
PoS	Proof-of-Stake
PoW	Proof-of-Work
RFID	Radio Frequency Identification
SCM	Supply Chain Management
SKU	Stock Keeping Unit
SVN	Stakeholder Value Network
UI	User Interface
WIP	Work-In-Process

1 Introduction

A supply chain refers to "the network of all the individuals, organizations, resources, activities and technology involved in the creation and sale of a product" [1]. This involves multiple processes, from the procurement of raw materials by suppliers, to design and fabrication by manufacturers, delivery of the finished product to consumers, and can even include post-sales logistics support. Supply chain management (SCM) integrates all of these steps and orchestrates the people, processes and technologies required to ensure a smooth flow. Given the complexity of most supply chain networks, there are many challenges associated with managing supply chains.

These challenges can be broadly categorized into two stages: the planning stage and the coordination stage. The main challenge associated with supply chain planning is demand forecasting, due to the inherent uncertainty of consumer demand for particular products [2]. As a result, it is difficult to get an accurate projection of the supply required to match the demand. This causes a related challenge, which is inventory management. As inventory is expensive to hold, too much supply relative to demand (overstocking) can result in major write-downs. Conversely, too little supply relative to demand (understocking) can result in lost sales. Hence, there are costs associated with a mismatch of supply and demand [3]. Various forecasting models have been developed to deal with this, and advancements in data analytics can help provide organizations with more and better information to manage these challenges [4].

Several other challenges exist in the coordination stage of the supply chain. With various entities involved at different stages of the supply chain, the presence of multiple decision makers complicates management and leads to information asymmetry. This lack of information sharing and traceability can make it difficult to chart progress, sense and respond to disruptive events, and also affects collaboration among parties [5]. This is tied in to the challenge of risk management, where organizations must remain flexible in an increasingly volatile world, in order to cope with sudden changes such as natural disasters or production disruptions [3], [6]. Another

related challenge is the requirement to ensure product safety and quality. In order to do so, there is a need for greater transparency throughout the supply chain [3], [7], [8]. As consumers become more discerning, they are also increasingly demanding to know the provenance of a product before committing to a purchase [8], [9]. Underpinning all of these challenges is the need for trust and to build strong relationships among supply chain partners. This will help to maintain the reliability and efficiency of the supply chain [10], [11]. Organizations continue to grapple with such issues and although progress has been made with better technology and management processes, there is still room for improvement. One potential way to address these coordination challenges is through the use of blockchain technology.

Blockchain technology can be summarized as "an open, distributed ledger that can record transactions between two parties efficiently and in a verifiable and permanent way" [12]. It is a relatively early-stage technology, with the most prominent implementation to-date being the cryptocurrency, Bitcoin. There are three main advantages of a blockchain: First, it is distributed and run on computers globally, so there is no central database to hack. Second, it is open and public so that it can be viewed at any time. Third, it is encrypted in order to maintain security [13]. This allows blockchain technology to reduce two key costs – the cost of verification and the cost of networking. As all transactions carried out are cryptographically verified by consensus and permanently stored on the blockchain, there is no need for additional costs to be incurred to verify trustworthiness, and it is also much easier to review the audit trail if problems arise down the line. By combining blockchain technology with a cryptotoken such as Bitcoin, it is also possible to eventually create a distributed marketplace without the need for costly intermediaries (e.g. Uber, Airbnb, eBay) [14].

Blockchain technology can potentially be applied to a wide variety of industries, including finance, insurance, medical, as well as supply chain [15]. The implications for SCM are promising, as it could provide a panacea to the current challenges faced and help the overall supply chain become more efficient. By using a decentralized database to provide a shared reality across non-trusting entities such as suppliers, manufacturers and even consumers [16], a blockchain can

improve transparency and traceability in the supply chain process. As pertinent information regarding the physical goods and other relevant events are securely and reliably recorded on the blockchain, smart contracts can also be used to automatically execute when specific conditions are met [14], [17]. This provides flexibility and can help to simplify the complex multi-party systems that a supply chain typically consists of. Although the use case is clear, actual implementations of blockchain technology in SCM are yet to proliferate widely. Hence, more research is required to investigate the possibility and practicality of such an application.

In this thesis, I first describe the concepts of blockchain technology and smart contracts. Then, I delve into the challenges of supply chain management and explore how blockchain and smart contracts, in particular, can be used to address those challenges. Next, I introduce a concept that uses smart contracts to determine provenance, track goods, execute payment, and manage reputation in the supply chain process. I validate it by developing a proof-of-concept and testing it on Ethereum, which is a decentralized application development platform using blockchain technology. Finally, I discuss the benefits and challenges of such an implementation, explore similar real-world concepts, and look towards the future of blockchain.

2 Blockchain Technology

2.1 Blockchain definition and principles

At its core, blockchain is a distributed ledger technology that "leverages the resources of a large peer-to-peer network to verify and approve transactions" [13]. These transactions are recorded chronologically in blocks, and the blocks are linked together cryptographically and stored permanently on the blockchain, creating an immutable chain. There is no single copy of the blockchain; rather, copies are distributed globally across the participants in the network and updated simultaneously [14]. This ensures that the ledger cannot be manipulated by any single party without the consensus of the majority in the network. The type of information stored in the transactions can be varied (e.g. currency, intellectual property, identity data, location data, etc.), making blockchain technology highly customizable for different applications.

In their article, Iansiti and Lakhani espoused five basic principles underlying blockchain technology [12]. They are:

1) Distributed database

The information contained within the blockchain is not controlled by any single party. Every participant in the network is able to access the entire database and its complete history. Since all the information is accessible, participants can also verify the records of transactions directly, without the need for an intermediary.

2) Peer-to-peer transmission

Communication within the network occurs directly between participants instead of through a central node or intermediary. Each participant constitutes a node, and each node stores and sends information to all other nodes.

3) Transparency with pseudonymity

All transactions and their associated values are visible to participants with access to the blockchain network. Every participant is assigned a private key and a public key. The private key is used to generate digital signatures and should never be disclosed, while the public key is an alphanumeric address that is used for transactions. This allows transactions to be traceable to the public address of participants, but they can choose how much of their identity information to share.

4) Irreversibility of records

Once a transaction has been verified and approved by the network, it will be recorded in the database and updated across all nodes. At this point, the records cannot be altered as they are linked to every preceding transaction record (hence forming the 'chain'). The process of adding transactions to the blockchain is accomplished by a subset of participants (known as 'miners') that compete to solve difficult mathematical problems [18]. Computational algorithms are used to ensure that the records are permanent, in chronological order, and accessible to all participants on the network.

5) Computational logic

The digital nature of the ledger allows transactions on the blockchain to be programmed using computational logic. Hence, users can build algorithms and rules to automatically trigger transactions between different nodes when certain criteria are met. This provides the basis for using smart contracts in blockchain.

2.2 How blockchain works

As its name implies, a blockchain is made up of a chain of blocks. Each block consists of data of transaction records and the corresponding information contained within each transaction. Every transaction (and hence block) has a timestamp associated with when it was recorded to the blockchain. Subsequent blocks require the identifier (or hash) of the preceding block, and this is the link that chains all the blocks together (see Figure 1).

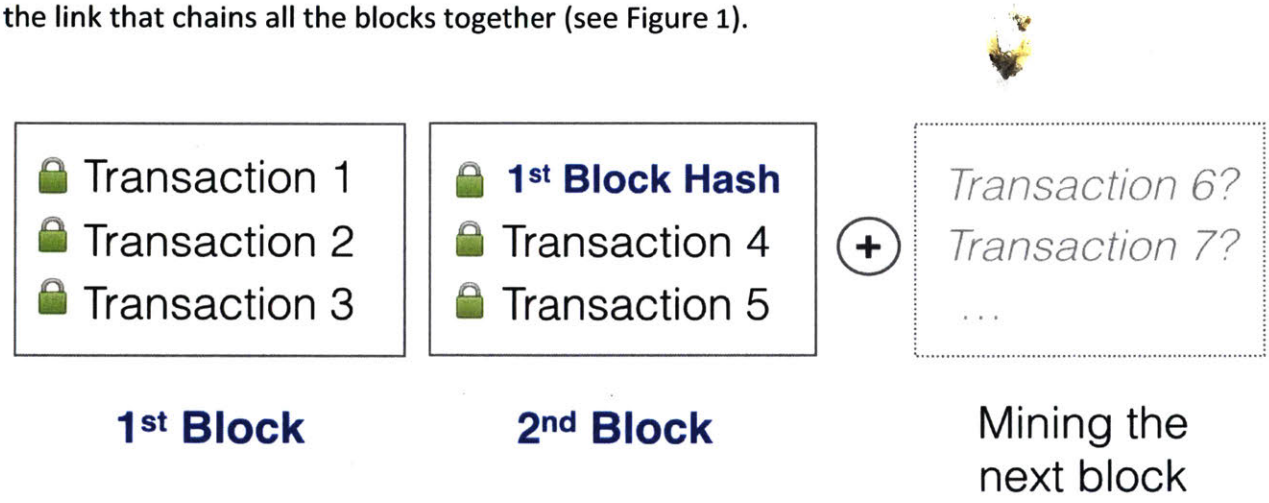


Figure 1: Blocks in a blockchain (Source: [19])

In order to add new blocks to the blockchain, they must be 'mined'. A common way to do this is through a proof-of-work (PoW) system, where miners "perform computationally costly tasks to participate in what essentially constitutes a lottery for the right to add the next block to the chain" [19]. Miners compete to solve the 'puzzle', and the successful miner broadcasts its proof-of-work to the other miners for verification. Consensus is achieved once there is a slight majority (51%), after which the new block is added (see Figure 2).

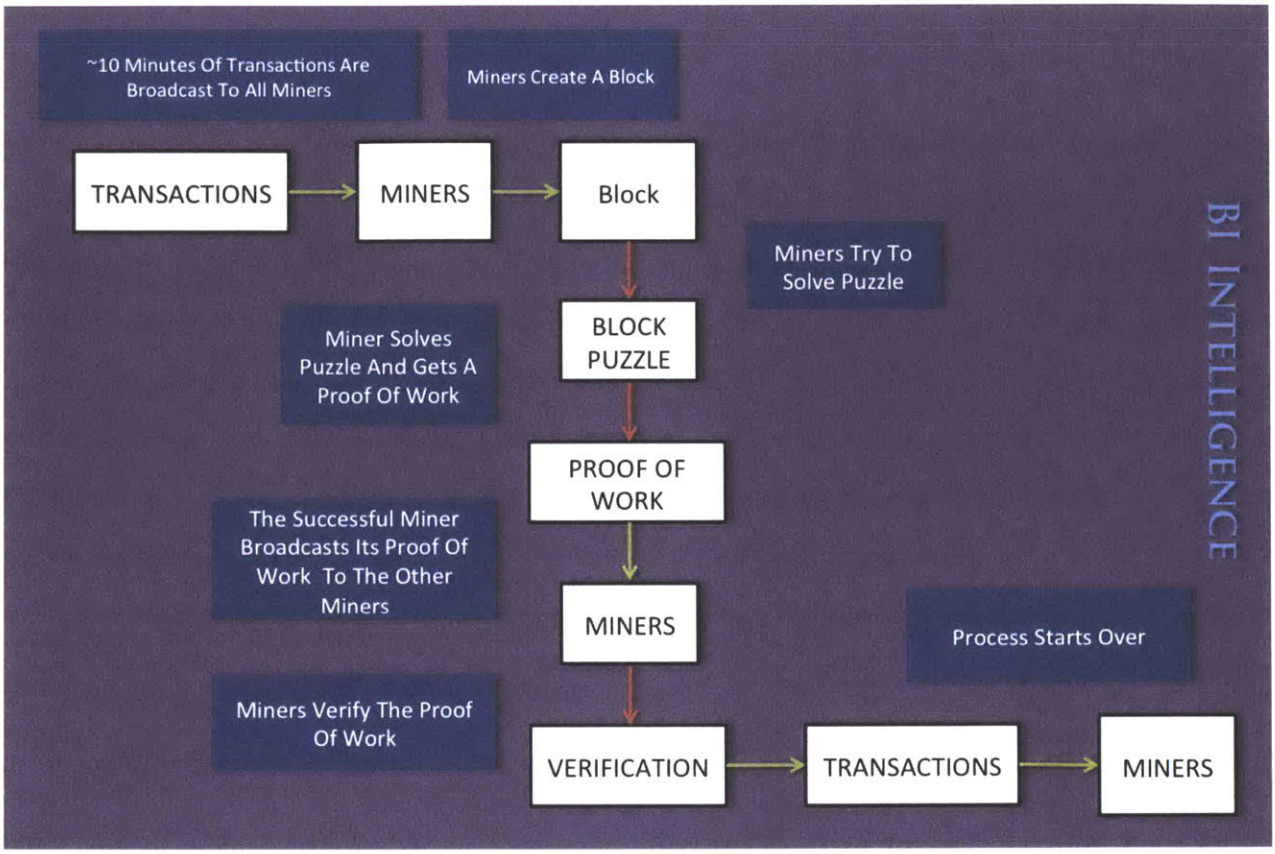


Figure 2: Blockchain mining procedure (Source: [20])

Once a block is successfully added, it is propagated to all nodes in the network to ensure that they are all updated with the latest blockchain. In addition, a reward is allocated to the miner for the work performed. This is usually in the form of a cryptotoken tied to the blockchain (e.g. Bitcoin or Ether). Hence, transactions that are carried out on the blockchain will incur a cost in order to incentivize miners to commit computing resources to solve the puzzles. As the chain becomes longer and more computing power has been devoted to support it, it becomes more difficult to tamper with a past transaction. This is because all the subsequent blocks will have to be amended and consensus must be achieved, which will require a disproportionate amount of resources [19].

2.3 Advantages of blockchain

As mentioned, it becomes more difficult to reverse a prior transaction as the blockchain grows longer. This is an inherent advantage of blockchain technology – its **immutability**. Once a transaction is recorded and propagated to the network, it becomes permanently stored on the blockchain and can be accessed subsequently for verification. This provides an audit trail for reliable verification. Since each block is linked to the preceding block, this also discourages rogue parties from manipulating the transaction records as they would require massive computing power to change all the subsequent records and obtain a majority consensus. This is succinctly summarized by Tapscott, who states that "to steal anything of value, a thief would have to rewrite the entire history on the blockchain. Collective interest ensures the blockchain's safety and reliability" [13].

Catalini and Gans add that blockchain technology can reduce two key costs – the **cost of verification** and the **cost of networking** [19]. Whenever an exchange of value occurs, key attributes of the transaction need to be verified by the parties involved. This usually involves a trusted intermediary, such as a central bank or financial institution in financial transactions, or a third party like eBay or Airbnb in an exchange of goods or services. These intermediaries typically charge a fee for providing verification services. In addition, many also require some form of information disclosure from the parties involved (e.g. social security number, credit card information). This increases the risk of data leakage should any of the intermediaries' security be compromised. Blockchain can circumvent this by securely storing key transaction attributes on a distributed ledger and allowing them to be verified at a future time without incurring additional cost. This helps to improve market efficiency as "credentials, reputation systems, provenance and other attributes of individuals, goods and services can be more cheaply tracked with higher integrity throughout the economy" [21].

The cost of networking can be reduced by combining blockchain's distributed ledger technology with a native cryptographic token such as Bitcoin. The token can be used as an incentive to grow

and secure a new platform, and an entire marketplace can be bootstrapped without the need for a costly intermediary. This can allow participants to achieve large-scale consensus on the allocation of scarce resources [22]. For example, a ride-sharing marketplace could be created to directly match drivers to riders, where payments are settled via a cryptotoken and reputations are maintained through transaction attributes stored on the blockchain. This would increase competition with traditional ride-sharing intermediaries such as Uber and Lyft, and they would be pushed to innovate and find other ways to add value. This architectural change is a good thing according to Catalini, as "in the long run, consumers are likely to benefit not only from the resulting lower costs and increased competition, but also from the new types of applications and services that could not be provided in a cost-effective manner without the technology" [21].

2.4 Limitations of blockchain

Although blockchain technology has the potential to disrupt a wide range of industries and fundamentally change the way we approach exchanges of value, it has not achieved widespread adoption yet. There are some limitations that need to be overcome before blockchain can become more scalable. The proof-of-work protocol used to generate blocks is **computationally wasteful** by design. Although this may help to preserve the integrity of the blockchain by deterring attacks, a large amount of resources and energy need to be committed. As the chain grows longer, even more computational power is required, exacerbating the problem [23]. This could preclude certain applications of blockchain, for example in Internet of Things (IoT) devices. Standard IoT devices would not have the requisite processing power to participate directly in a blockchain, so a workaround needs to be developed in order to scale up adoption [24].

As larger blockchains require more resources, **processing speed** is also adversely affected as more time is required to verify and approve transactions. For comparison in a financial application, an established intermediary like Visa can process more than 50,000 transactions per second at peak, whereas Bitcoin can only process 7 transactions per second [19]. Hence,

blockchain's computational efficiency must be increased significantly before it can seek to challenge the incumbents.

Even though blockchain technology relies on consensus of the majority to ensure security of the system, it is not completely immune to **security breaches**. Exploiting the need for consensus, it is plausible that a single rogue entity could garner enough share of the network to achieve 51% majority, when it would then be able to introduce its version of the blockchain and validate it [25]. Although this is unlikely due to the distributed nature of blockchain, it is not impossible especially as computational requirements grow, making it commercially viable for only a small number of influential participants to continue mining for reward. Another problem that could arise is if users are able to exploit security flaws in the code underlying a blockchain. This occurred when a hacker exploited a weakness in the code of the Decentralized Autonomous Organization (D.A.O.) – a blockchain-based venture capital fund – and was able to siphon one-third of the cryptocurrency that it had raised from investors [26], [27]. Although the flaw was fixed subsequently, it has affected confidence in blockchain's viability on a larger scale.

To overcome some of these limitations, alternative protocols such as proof-of-stake (PoS) have been proposed over proof-of-work. In proof-of-stake, miners do not compete to solve the computational puzzle; instead, they are chosen deterministically based on their relative wealth (or stake) in the cryptotoken. Proponents argue that this wastes less resources, can result in faster transactions, and can decrease the likelihood of a 51% attack [18].

2.5 Implementations of blockchain – Bitcoin and Ethereum

To date, the most prominent and successful implementation of blockchain technology is Bitcoin. Introduced through a white paper published by Satoshi Nakamoto on October 31, 2008, Bitcoin is a peer-to-peer (P2P) electronic payment system "based on cryptographic proof instead of trust, allowing any two willing parties to transact directly with each other without the need for a trusted third party ... For the first time in history value could be reliably transferred between two distant,

untrusting parties without the need for a costly intermediary" [18] such as financial institutions or payment service providers. As Bitcoin began to gain popularity beyond the coding world and into mainstream society, exchanges were opened for it to be traded like other currencies and service providers began accepting Bitcoin as a legitimate form of payment. Although Bitcoin has suffered its fair share of criticism (particularly for its use in criminal transactions in the black market) and volatility as a result, there is still much optimism about its future applications and that of blockchain technology as a whole. This is reflected in the steady increase in the value of Bitcoin, where it has risen by more than 100% in 2017 alone (see Figure 3).

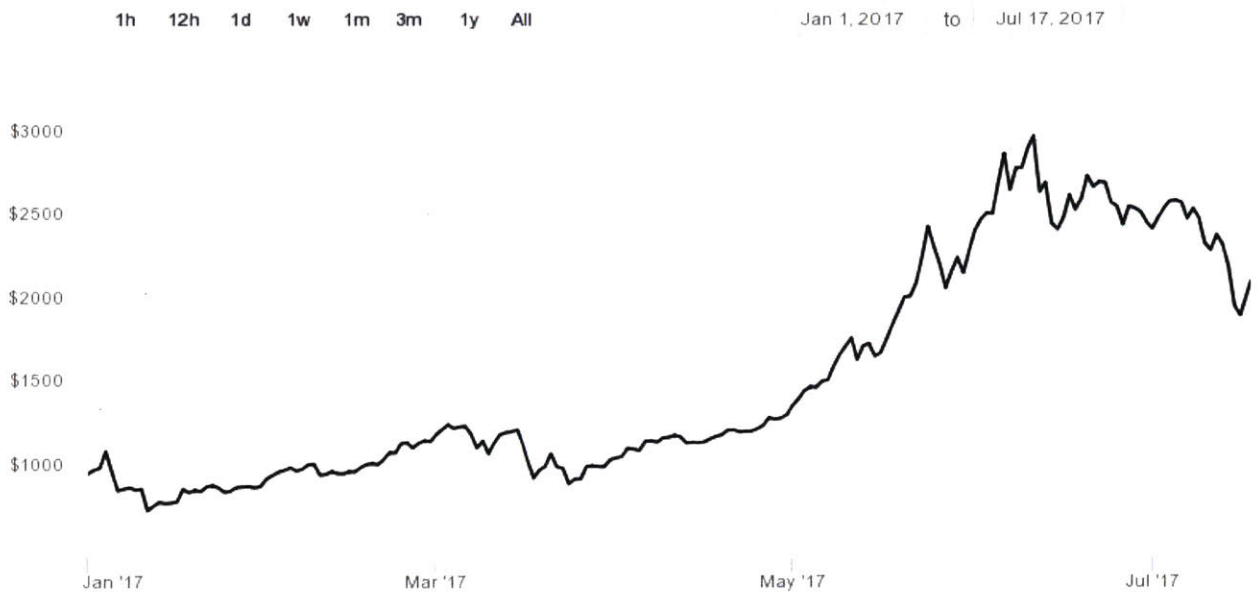


Figure 3: Price of Bitcoin from January to July 2017 in USD (Source: [28])

The success of Bitcoin has spawned several alternate cryptocurrencies that are based on different underlying protocols. In particular, Ethereum aims to take blockchain technology beyond finance and serve as a platform for decentralized application development that can be used for a myriad of industries. It was founded in 2014 by Vitalik Buterin, Gavin Wood and Jeffrey Wilcke. Ethereum's differentiating factor is that it is programmable to suit users' requirements, rather than giving users a set of pre-defined operations such as Bitcoin transactions. As a result, it could in theory be used to automatically and reliably carry out any transaction involving trust, security,

or permanence [29]. Although a relatively new implementation of blockchain, Ethereum has garnered a lot of hype over its potential to change the way many businesses operate. This has attracted interest from technology giants like IBM and Microsoft, who have separately made big pushes to explore the applications of Ethereum. IBM is leading a collaborative project known as the Hyperledger Foundation, while Microsoft has formed the Enterprise Ethereum Alliance with other corporate giants [30]. Although there are still fewer real-world uses for Ethereum compared to Bitcoin, its programmability is gaining it favor and it is rapidly catching up in terms of market value (see Figure 4).

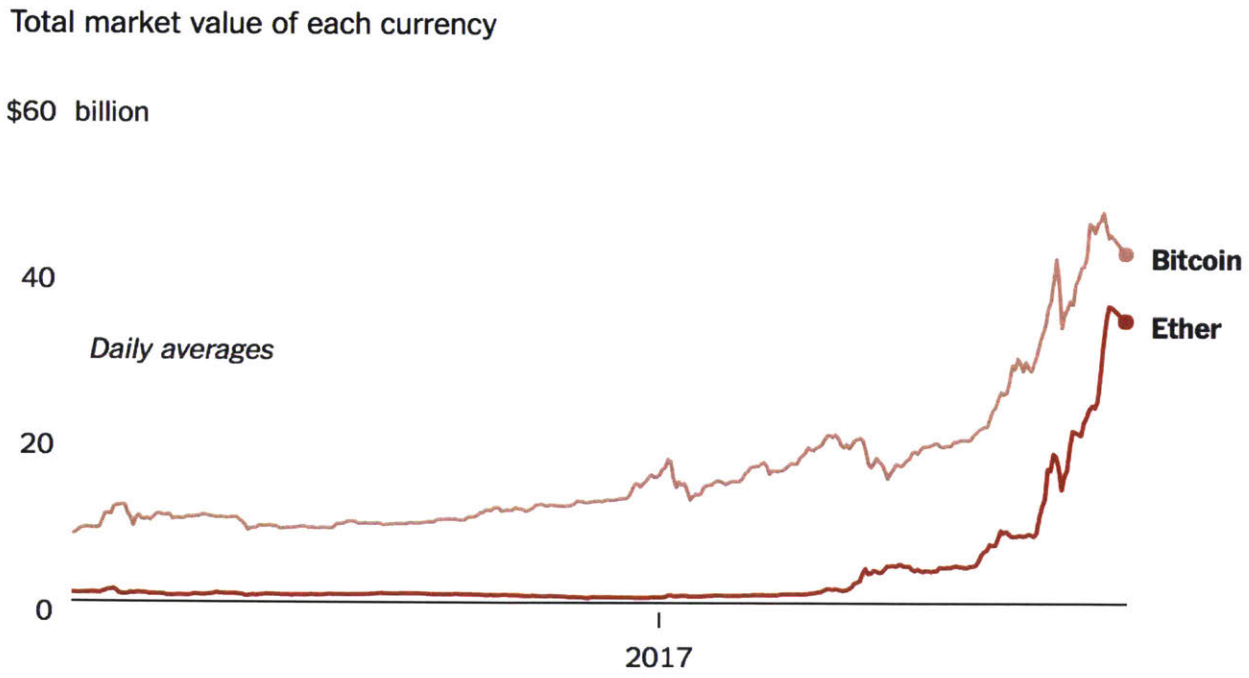


Figure 4: Market value of Bitcoin and Ethereum as of June 2017 (Source: [31])

3 Smart Contracts

3.1 Smart contract definition

The Ethereum platform's programmability is predicated on its ability to create and execute smart contracts. The term 'smart contract' was introduced by Nick Szabo in 1996, when he described it as "a set of promises, specified in digital form, including protocols within which the parties perform on these promises" [32]. The Smart Contracts Alliance, in its white paper on smart contracts, further elaborates on each element of this definition [33]:

1) **"a set of promises"** – These promises refer to the contractual terms and/or the rules-based operations that are designed by the parties and agreed on in order to carry out their exchange.

2) **"specified in digital form"** – This implies that a smart contract operates electronically and its terms and operations are all specified by writing lines of code in software.

3) **"protocols"** – These protocols in the form of algorithms define the rules that each party must abide by in order to perform actions in relation to the smart contract (e.g. release of payment).

4) **"within which the parties perform"** – This highlights the self-executing nature of a smart contract, where once initiated, the outcomes are typically irrevocable. Performance is thus automated.

In a nutshell, smart contracts are agreements between transacting parties that are written using computer code and programmed to self-execute when certain conditions are met. These can be integrated onto a blockchain platform like Ethereum to provide the verification and integrity necessary for such an automated system to work.

3.2 How smart contracts work

To understand how smart contracts work, it is also important to know how the Ethereum platform functions and how smart contracts come into play. The basic unit in Ethereum is the account, and the Ethereum blockchain tracks the state of every account. Each time there is an exchange of value or information between accounts, the accounts' state transitions are captured on the blockchain. There are two types of accounts: Externally Owned Accounts (EOAs) and Contract Accounts. EOAs are controlled by private keys that are assigned to human users, whereas Contract Accounts are governed by their internal code and can only be activated by an EOA [29]. Contract Accounts are where smart contracts reside within the blockchain.

A summary of how smart contracts work in a blockchain is shown in Figure 5. To create a smart contract, parties must first identify an opportunity for them to collaborate and agree on the desired outcomes for each party. This could include any possible exchange of value such as goods or services. Next, they must set the terms and conditions that have to be met in order for an exchange to occur. These could be triggered by the parties themselves, by external events, or by certain milestones. All of the requirements and contractual agreements are then programmatically written using computer logic and code. The smart contract can then be deployed to the blockchain, where it will self-execute when initiated by the specific conditions.

In order to initiate a smart contract, users must use an EOA to carry out a transaction with a Contract Account. This is encrypted with the initiator's private key and transmitted to other nodes in the blockchain. The other users can verify the authenticity of the transaction using the public key generated, to ensure that the initiator was indeed the one who triggered the transaction [33]. Once consensus has been obtained from the majority, the transaction is added to the blockchain, the smart contract is successfully executed and its outcomes are recorded. As the state of the blockchain changes, it is updated across all nodes in the network and the outcomes cannot be altered.

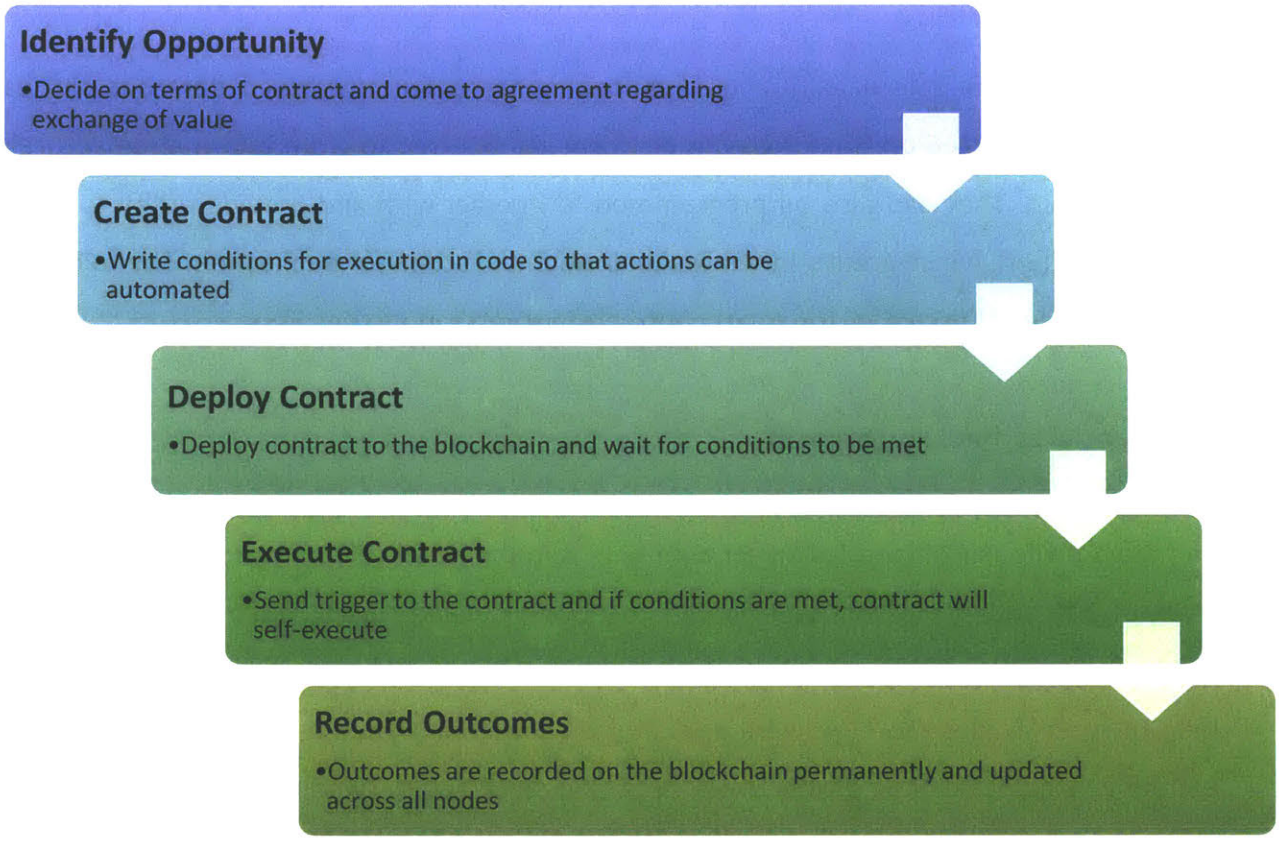


Figure 5: Smart contract process in a blockchain

Due to the computational resources required to maintain the proof-of-work system, every transaction that creates a change in state in the Ethereum platform (including deploying and executing smart contracts) requires a transaction fee. This is paid by the initiator of the transaction in the form of Ether, Ethereum's native value-token [29]. The amount depends on the extent of computing power and memory storage the transaction requires. This discourages frivolous or malicious computational tasks and provides impetus to write smart contracts with as efficient code as possible.

3.3 Smart contract use cases

Beyond the traditional role of contracts in managing agreements between different parties, smart contracts deployed on the Ethereum platform can be used to store information securely on the blockchain. They can also be programmed to interact with and provide utility to other smart contracts [34], which greatly increases their potential scope for application. The following are some examples of use cases for smart contracts in diverse industries [33]:

1) Financial technology (fintech)

The most obvious use case for smart contracts is in the financial services sector, given that many of the contracts existing today relate to monetary value. Apart from traditional banking services, smart contracts can also be used in securities trading and management. This can help to make trades more efficient and shorten settlement periods. It can also automate certain processes such as dividend payment and stock splits. Symbiont, a smart securities company, is already exploring this as part of the Delaware Blockchain Initiative [15]. Continuing in the vein of fintech, smart contracts are being considered for over-the-counter (OTC) derivatives trading as well, to help automate processes and enable real-time valuation of positions.

Apart from facilitating trades, smart contracts can also be used to record important financial data, serving as an accurate and transparent accounting ledger. This will help to reduce accounting fraud, smoothen the audit process, and can reduce the costs associated with managing accounting data.

2) Digital identity management

Since smart contracts can securely store data on the blockchain, they can be used to store an individual's personal data, which the individual has full control over. With personal control,

individuals can then choose to disclose only pertinent information to the necessary parties during transactions. For example, instead of giving our entire passport to a bank to set up an account, the bank can simply query our digital identity to provide the details required. This reduces the risk of data leakage or a single point-of-failure should the third parties hold on to such sensitive information and become compromised (as is the case currently).

3) Property

Smart contracts can be used in property in two ways. The first is by managing mortgages. Many different parties are involved in mortgage transactions, which adds complexity and friction to the various processes. Smart contracts can help to simplify these processes and reduce the likelihood of errors by automating them.

The second way smart contracts can be used is for land title recording. Coupled with digital identity management, smart contracts can help to improve transaction integrity, efficiency and transparency of property transfers and reduce the risk of fraud. Some countries have begun exploring this, including Georgia, Ghana and Honduras [15].

4) Insurance

In the insurance industry, smart contracts can be used to automate the claims process to increase processing speed and payment efficiency. For example, the current automotive insurance claims process is usually very disjointed and drawn out. By storing all the policy details in smart contracts and equipping cars with smart sensors, damage can be assessed and claims processed on-the-spot in the event of an accident. This saves both time and cost by avoiding any duplication of work to verify reports and policies.

5) Medical

Smart contracts can aid the medical industry in terms of managing and sharing sensitive patient data. This can help when conducting clinical trials, by automating and tracking patient consent, ensuring privacy of patient data, streamlining setup processes, and facilitating cross-institutional sharing of information. Similarly, the same benefits can be used for the conduct and sharing of life-saving medical research, such as cancer research. By addressing privacy concerns and increasing transparency, institutions might be more willing to collaborate and share data with one another, helping to boost important research in the field.

6) Supply chain

Current supply chain processes are long and complex. Smart contracts can help to simplify this and improve visibility across the supply chain. By coupling them with IoT devices that can track the location of goods, smart contracts can allow for tracking of inventory and the chain of custody throughout a supply chain. With this information, companies will be better equipped to deal with disruptions or manage incidents such as recalls. This also reduces the risk of theft or fraud along the supply chain.

On top of tracking the journey of goods, smart contracts also allow companies and consumers to determine their provenance. This is especially important in the food industry (for health reasons), but also as environmental sustainability becomes a bigger concern. Companies like Walmart, Maersk, BHP Billiton, and Everledger have started exploring the use of smart contracts to track a diverse range of goods, including meat, shipping containers, mining samples, and even diamonds [35]–[38].

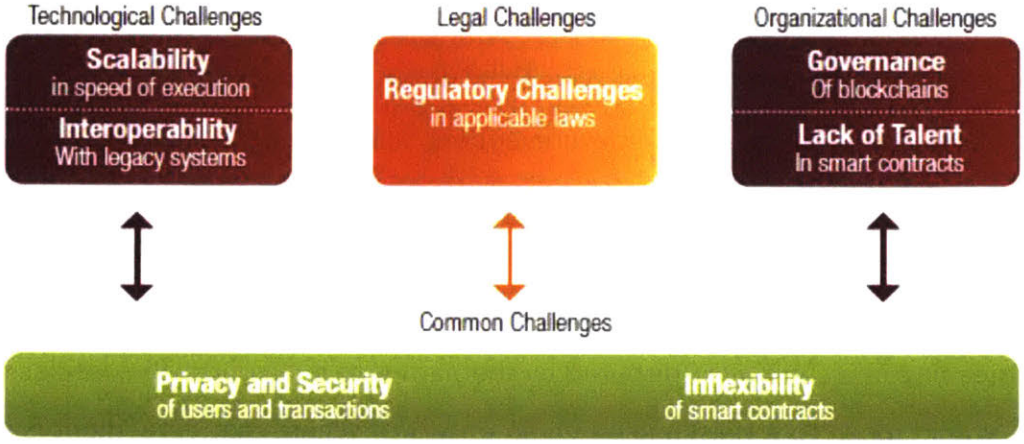
3.4 Challenges facing smart contracts

For all the potential applications of smart contracts, it is not without challenges as well. The main challenge involves the **regulatory environment** and how smart contracts will affect established **legal practices**. Being a nascent technology, there are scarcely any existing regulations governing the use of smart contracts. Measures will have to be drawn up to mitigate concerns over privacy, security, consumer protection and crime prevention. With an open blockchain that is highly visible and accessible yet can store sensitive information, security is of utmost importance. It is also unclear how current contract law will come into play in the event of smart contract disputes. As mentioned by Green, an Associate Professor of Law at the University of Oxford, "Currently, when the meaning of a contract is disputed by the parties to it, a court will consider what that agreement would mean to a reasonable human observer. Where that agreement is written in computer code, however, and intended for communication to an artificial intelligence, the significance of a reasonable human observer's interpretation is a matter of contention" [39]. Current legal practices will have to be updated in order to manage smart contracts appropriately.

Another challenge is that the success of smart contracts is highly dependent on the code programmed into them and the individuals writing the programs. Smart contracts **execute exactly as instructed**, so a lot of care and effort must be put into understanding what the code entails, both for contract developers and the parties entering into agreement. There have already been instances where the code in smart contracts were exploited, resulting in large financial losses (e.g. D.A.O., CoinDash) [26], [40].

Some other **technological challenges** include scalability of smart contract technology (as processing speed on the blockchain is still slow) and their interoperability with existing legacy systems. Should smart contracts be implemented on a larger scale in the future, the transition from legacy systems to the new technology must be managed carefully to ensure no loss of data or security. There are also **organizational issues** such as determining the overall governance of

blockchains (public vs. private blockchains) and attracting enough talent into the smart contract space to contribute positively to its growth. A summary of the challenges is shown in Figure 6.



Source: Capgemini Consulting Analysis

Figure 6: Challenges facing smart contracts (Source: [41])

4 Supply Chain Management

4.1 Overview of supply chain

A supply chain encompasses all of the activities that go into the delivery of goods or services, beginning at the earliest stage of creation and ending at the final stage of destruction or extinction. With the impact of globalization, supply chains typically cross the boundaries of both organizations and countries. They can also vary significantly in terms of length (i.e. the number of tiers across the supply chain) and depth (i.e. the number of suppliers or customers within each tier), depending on the good or service in question. The complexity of a generic supply chain network is illustrated in Figure 7.

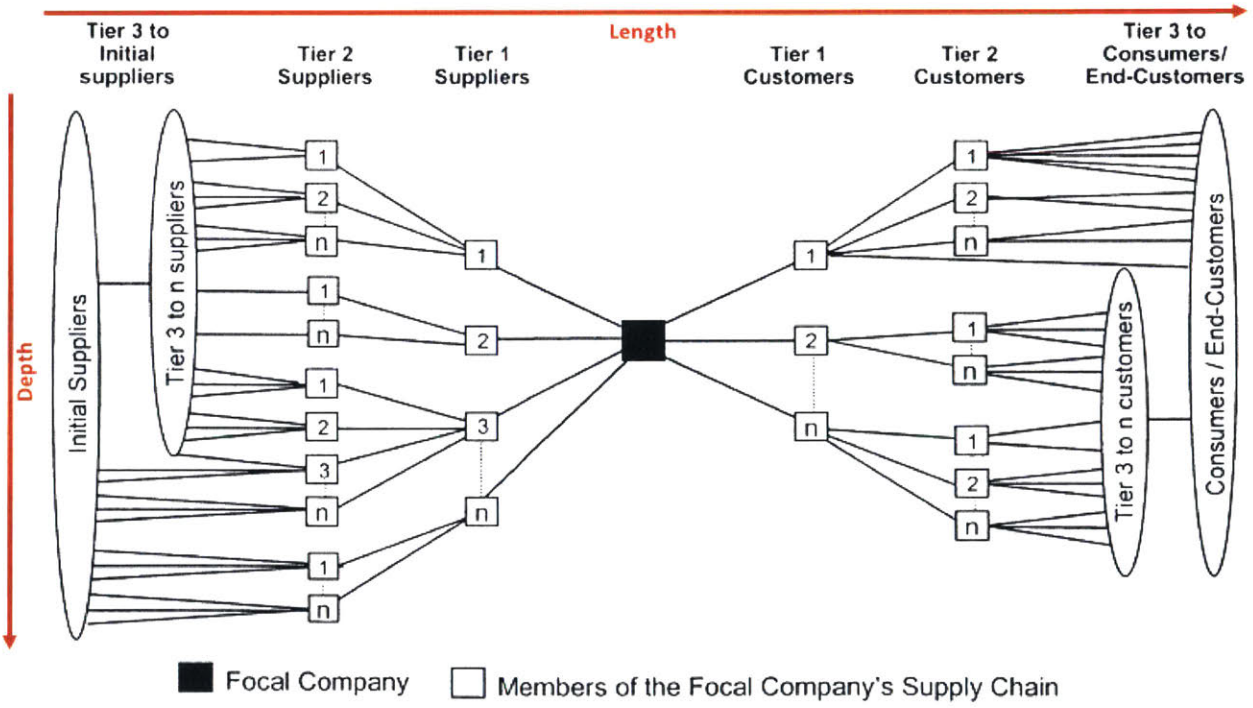


Figure 7: Generic supply chain network (Source: adapted from [42])

Given the complexity and diversity of supply chains, it is imperative that the multiple processes and relationships across the entire chain – inter-company, intra-company, and business-to-

consumer – are managed properly in order to ensure smooth operation of the supply chain. Therefore, the field of supply chain management (SCM) is crucial to the success of global businesses. The Global Supply Chain Forum (GSCF), a group of non-competing firms and a team of academic researchers, defines SCM as "the integration of key business processes from end user through original suppliers that provides products, services, and information that add value for customers and other stakeholders" [42].

To better identify the various processes involved in a supply chain, a stakeholder analysis of a generic supply chain was conducted (see Figure 8). The stakeholder value network (SVN) shows that multiple exchanges take place between the stakeholders, and these exchanges can be financial, good/service, information, or political. It is important to understand the needs and goals of the individual stakeholders in order to manage all of these processes:

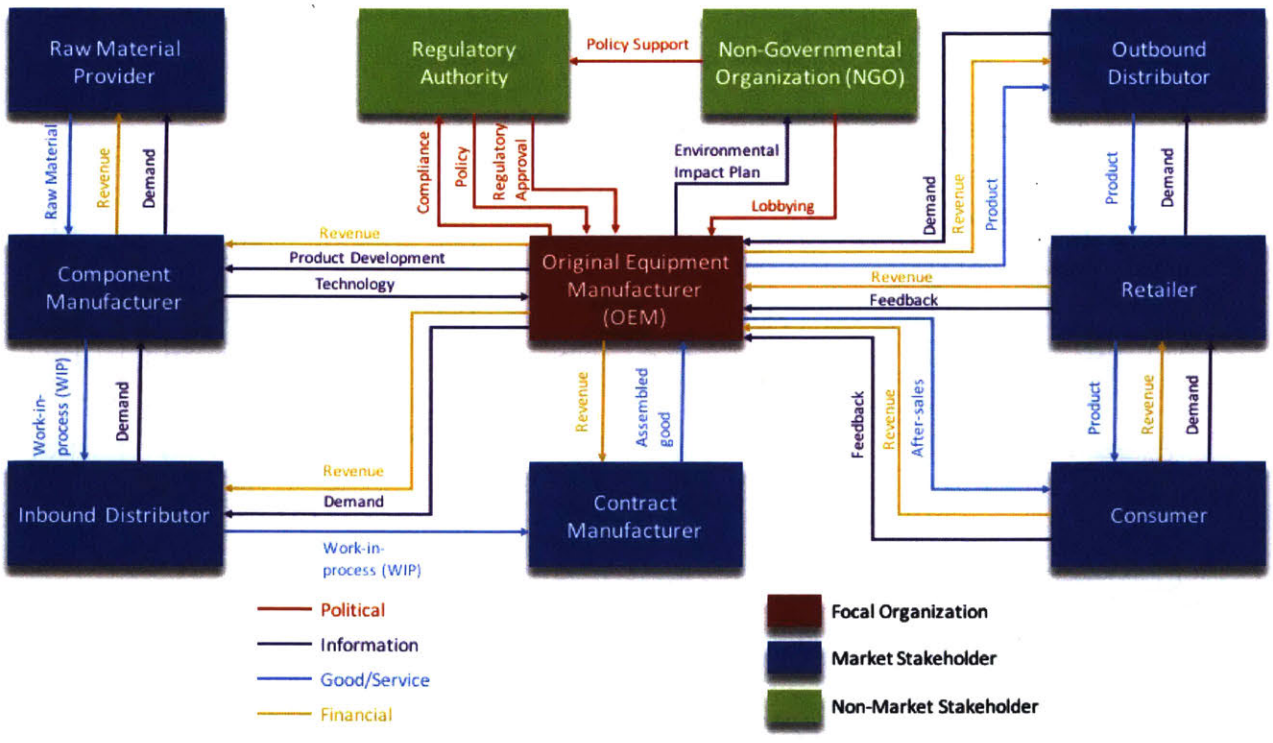


Figure 8: Stakeholder value network for a generic supply chain

1) **Original Equipment Manufacturer (OEM)** – the OEM serves as the focal organization for this analysis and has links to all other stakeholders. Its goal is to develop a product and successfully bring it to market, so that it can earn revenue and grow its brand. It is responsible for ensuring that its suppliers meet the deadline and product requirements, that the product adheres to regulatory, safety and environmental guidelines, and that consumers are satisfied with the product. Hence, there are multiple business processes that the OEM must manage, including demand forecasting, inventory management, manufacturing flow, product development, marketing and customer service.

2) **Raw Material Provider** – the raw material supplier sits at the earliest stage of the supply chain, where it is responsible for providing the raw material required to manufacture the product. Its goal is to provide sufficient quantity and quality of raw material at a market-competitive price so that it can earn revenue and get repeat customers in the future. Its main point-of-contact is with the component manufacturer.

3) **Component Manufacturer** – the component manufacturer obtains raw material and uses it to build the individual components of the product. Its goal is to build high quality components to earn revenue as well as the trust of the OEM, so that it can be relied on for subsequent orders. The manufactured components, or work-in-process (WIP), are passed to a distributor for management. The component manufacturer also liaises directly with the OEM for product development, where the OEM's engineering team provides the specification and the component manufacturer provides the technology to create the components.

4) **Inbound Distributor** – the inbound distributor obtains WIP from the component manufacturer and distributes it to the contract manufacturer for assembly. Its goal is to manage the WIP inventory level to ensure that there is sufficient supply to meet demand, but not too much supply to incur high inventory holding costs. As a result, it has to manage

its relationships with the component and contract manufacturers closely, as well as getting accurate demand forecasts from the OEM.

5) **Contract Manufacturer** – the OEM can choose to outsource final assembly to a contract manufacturer, who obtains WIP from the distributor and builds the complete product. Its goal is to carry out the duties specified in its contract in order to earn revenue and maximize profits. Its main relationship is with the OEM to ensure that the terms of the contract are honored.

6) **Outbound Distributor** – the outbound distributor receives the final product from the OEM and distributes it to the retailer for sale. Its goal is to manage the product inventory level to match supply and demand, at the same time keeping inventory costs low. Hence, it has to maintain strong relationships with the retailer, and work with the OEM on demand forecasting.

7) **Retailer** – the retailer receives the product from the distributor and is responsible for selling it to the consumer. Its goal is to maximize sales of the product, and hence its own profit. Therefore, it is beneficial for the retailer to work well with the distributor to ensure that it continues to receive sufficient product to sell. It can also communicate with the OEM to provide feedback on product improvement so that a better product can be developed to generate more sales.

8) **Consumer** – the consumer is at the end of the supply chain and is the main source of demand and revenue. Hence, it is important for stakeholders like the OEM and retailer to manage their relationships with the consumer very carefully. The consumer's goal is to gain satisfaction and/or improve quality of life through purchasing a high quality and useful product. The consumer provides product feedback to the OEM, who in turn provides after-sales support and works to develop an improved product. The OEM also observes consumer trends to forecast demand and gain ideas for new products.

9) **Regulatory Authority** – the regulatory authority is a non-market stakeholder, as it is not directly involved in the market for the product. However, it plays an important role in providing oversight for the market. Its goal is to ensure that the product meets regulatory and safety guidelines, so that it is safe for use and the risk of hazard is minimized. Since regulatory approval can make or break a product's ability to go to market, it is crucial for the OEM to manage this relationship.

10) **Non-Governmental Organization (NGO)** – the NGO is also a non-market stakeholder. There are various NGOs championing different causes that can have an impact on the product. Their goal is to increase public awareness of issues and possibly to increase their sphere of influence. For example, if a product might have an environmental impact, it is important for the OEM to gain buy-in from environmental NGOs prior to product launch, otherwise they could face a backlash from the market. This is increasingly stark in a more open and highly-connected world where public opinion can be swayed in the blink of an eye. Therefore, the OEM must also manage this relationship with care.

The stakeholder analysis has shown that even for a generic supply chain, multiple relationships and business processes exist. Often, these are not one-to-one relationships but are interconnected. Furthermore, as a supply chain grows in both length and depth, these relationships become even more complex and critical to manage.

4.2 Challenges in managing supply chains

Given the complexity of supply chain networks and the business processes involved, there are many challenges associated with SCM. These challenges can be broadly categorized into two stages: the planning stage and the coordination stage. The planning stage involves determining the level of supply and demand, whereas the coordination stage concerns the actual physical movement of goods.

Planning stage

1) Demand forecasting

Depending on the type of product, forecasting consumer demand can be one of the biggest challenges in SCM. For consumer staples or essential products (e.g. food, home appliances), demand tends to be non-cyclical and relatively predictable. However, for consumer discretionary or lifestyle products (e.g. fashion, technology), demand uncertainty can be high as there are many factors affecting a consumer's decision. Consumer trends can also change very quickly; the latest gadget or fashion style can become obsolete or passé within a year. In a 2010 McKinsey survey on supply chain challenges, volatility of customer demand was the most frequently cited challenge by company executives [4]. Supply chain managers have developed various models to forecast demand, using historical data, linear programming, and other time series analysis methods. With the rise of big data, there is the potential to use more and better data analytics to help improve the accuracy of demand forecasting by tracking other observable characteristics.

2) Inventory management

Closely associated with demand forecasting, inventory management involves the fine balancing act between ensuring sufficient supply to meet demand and preventing a supply glut, which cuts into profit margins. If supply is too low, profits are affected due to lost sales. Consumer and client relationships are also negatively affected, which may have a cascading effect on subsequent products. If supply is too high, there is too much inventory and inventory holdings costs will lower profit. In his book, *The Goal*, Goldratt talks about high inventory levels being one of the main culprits affecting the profitability of organizations [43]. Furthermore, as interest rates gradually increase, inventory holding costs will be driven up as well [2]. Apart from managing the inventory level, it is important to take into account lead

times and manage the timing as well, so that delays are minimized [3]. There are several existing inventory management models, such as the Newsvendor model, economic order quantity (EOQ) model, periodic review and continuous review models. These models work by determining the optimum quantity and timing to replenish inventory so as to maximize profits and reduce costs. Similar to demand forecasting, the growth of data analytics can help to improve the accuracy of such models.

Coordination stage

1) Information sharing and product traceability

Given the multiple stakeholders across a supply chain, each with their own goals (on top of the ostensible overall aim), it is inevitable that information asymmetry exists. Whether this asymmetry is intentional or not, it poses a challenge in managing the supply chain without complete information. An example of the lack of information sharing affecting the supply chain is in product traceability. Without the availability of real-time information to track the progress through the production cycle (from raw material to component manufacturing, assembly, distribution, and all the transportation in between), it is difficult to update forecasts and the ability to respond to negative events is decreased. If the product is delayed or there is insufficient supply to meet demand, this could jeopardize future customer relationships and have a longer-lasting detrimental effect on the organization's brand [44].

The lack of information sharing is also a barrier to building trust and collaboration between supply chain stakeholders. If parties suspect that information is being withheld by other parties, they are less likely to trust one another and may incur additional costs to do double work or for verification. This may also cause delays along the chain. Research has shown that information sharing helps to improve supply chain coordination and product quality, and that information serves as "the 'glue' that holds together the business structures that allow supply chains to be agile in responding to the competitive challenges" [5]. The McKinsey survey adds

that more and better information will be required to solve evolving SCM challenges moving forward [4]. Therefore, technology that can help to facilitate the sharing of information and tracking of progress throughout a supply chain may prove very beneficial.

2) Managing risks and disruptions

Risk management is an essential aspect of any organization. The ability to plan for and react to contingencies will help to minimize the impact of disruptions and ensure that operations can continue in the event of an emergency. This is no different for supply chains, and arguably even more important, given the wide geographical and organizational spread. In an increasingly volatile world where natural disasters or production disruptions can occur at any time, supply chain managers must have adequate backup plans to ensure that the supply chain is flexible enough to handle sudden changes without too much delay or impact to profits [3].

Aside from having backup plans, another way to better prepare to manage risks and disruptions is through increased visibility along the supply chain [6]. This is related to information sharing and product traceability, whereby having better visibility of the progress will allow organizations to remain nimble and stand ready to respond quicker in the event of a disruption. With better information, they can also use a more targeted approach to address issues.

3) Push for transparency

As the regulatory environment and consumer behavior continue to evolve, there is a push towards greater transparency in the provenance of goods and the supply chain process as a whole, in order to better ensure product safety and manage recalls [7], [8]. This is especially relevant in the food industry, where food safety and quality are of utmost importance. For example, McDonald's, the world's largest fast food provider, had drawn a lot of flak

previously for using products that were not fresh and untrustworthy. To combat the criticism, McDonald's reviewed its supply chain practices and made them more transparent. Although that has helped, it takes a much longer time to mend the negative reputation ingrained in people's minds [9]. More recently, the Chipotle food poisoning scandal in 2015 underscored the importance of transparency and tracking the provenance of goods. Due to viral outbreaks which sickened hundreds of people across the U.S., Chipotle was forced to close several stores and conducted an extensive review of its food safety practices. Yet, it was unable to ascertain the actual cause of the outbreak since it could not track its entire supply chain properly [45]. Had Chipotle been able to track the provenance of its food items, it could have responded to the outbreak much quicker and with more certainty, potentially averting the crisis that ensued.

Transparency and provenance are not limited to the food industry or safety issues. Consumers have become more discerning about the products that they buy, especially with the plethora of choices today. Often, they choose products that are aligned with their values and beliefs. For example, people might want assurance that the product originates from a reliable source that does not use child labor, does not harm the environment, or does not engage in any other unethical practices (e.g. 'blood diamonds'). All of these have implications for the image of both the product and organization. Since it is essential to manage the customer relationship, having a means to ascertain the provenance of a product is increasingly important. Greater transparency also helps to keep parties in the supply chain accountable, as they are obliged to adhere to the standards set rather than cut corners for their own gain (e.g. using low quality materials to maximize their own profit) [3].

4) Building trust and reputation

All of the coordination challenges discussed thus far have an overarching principle linking them together – building trust across the supply chain. Sharing of information and increased transparency have shown to increase the level of trust among supply chain partners. In turn,

trust helps to foster commitment and decreases behavioral uncertainty in the supply chain [10], as parties are more aligned with a common goal and values. In addition, trust can provide a competitive advantage; when all parties are on the same page, the supply chain will be more flexible and can adapt to situations and make decisions faster. On the contrary, a lack of trust can cause inefficiencies in a supply chain as additional time and money is wasted on verification, inspections and duplicate work [11].

One way to engender trust is through building up and maintaining a good reputation [10]. If suppliers have proven over time that they are reliable and produce high quality products, manufacturers are more likely to trust them. Similarly, suppliers will feel more comfortable working with manufacturers who have shown that they keep to their word and do not alter requirements at the last minute. Hence, it would be beneficial to leverage technology to provide a globally accessible database where reputable supply chain partners can be easily identified [46]. Having a common and transparent platform will also incentivize organizations to improve their standards constantly or risk getting left behind.

4.3 Addressing supply chain challenges using blockchain and smart contracts

The introduction of blockchain technology ushers in the potential to solve some of the coordination challenges in SCM. By using smart contracts to handle the multiple exchanges between parties, blockchain technology can reduce complexity and allow for greater transparency and trustless verification across the supply chain. This will help to speed up the supply chain, make it more agile, and foster stronger relationships among partners.

Smart contracts can improve SCM in three key ways: transparency, traceability and efficiency. The benefits are summarized in Figure 9 and elaborated further below:

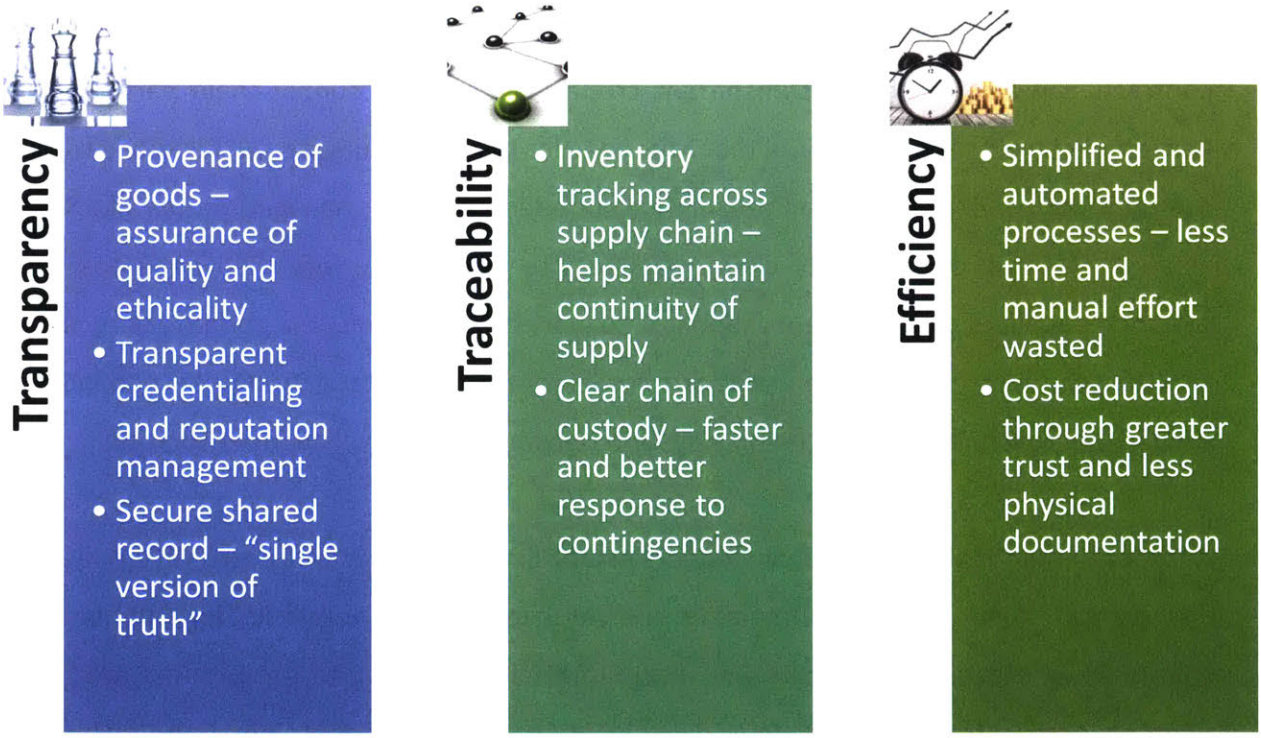


Figure 9: Summary of smart contract benefits in supply chain management

1) Transparency

Smart contracts can enhance transparency in the supply chain by recording the **provenance** of goods. By storing information such as the date, location and quality on the blockchain, the origin of a product can be easily verified. This will provide assurance to manufacturers that their raw materials are coming from reliable sources, and consumers can have more confidence that they are purchasing a legitimate product. There are already companies entering this field, such as Everledger, which is trying to improve the transparency of the diamond supply chain to combat issues such as forced labor and 'blood diamonds' (where the sale of diamonds is used to fund violence). Another example is Provenance, which is promoting transparency and socially acceptable practices by helping companies to disclose how they build their products and their environmental impact [47].

By allowing a digital form of identification to be created and stored on the blockchain, smart contracts can also help with **transparent credentialing** across supply chain partners [48]. Parties will be able to easily verify that other parties have the requisite certifications to carry out their duties. Reputation and reliability can be recorded and managed on the blockchain as well. This will allow supply chain managers to make more informed decisions when selecting suppliers, and also spurs suppliers to work hard to maintain a good track record. In addition, this may also provide a platform for lesser-known but quality suppliers to increase their global reach. In a sense, the playing field becomes more level and meritocratic.

Another benefit is that "blockchain ledgers establish a shared, secure record of supply chain information flows – a **single version of the truth** across networks for supply chain transactions, processes, and partners" [49]. Information relating to the product (e.g. serial number, origin), transportation (e.g. date, location, number of containers) and accounting (e.g. purchase order, receipt, shipment notification) can all be stored on the blockchain and accessible to relevant parties. This openness helps to ensure that all parties are on the same page and reduces the potential for conflict. Should there be disputes, they can also be resolved faster since there is a common database to refer to.

2) Traceability

Smart contracts can improve traceability within the supply chain by allowing inventory to be tracked at every step along the way, from its raw material source to end-user delivery. This can be accomplished through the use of serial numbers, radio frequency identification (RFID) tags, or smart sensors. The latter is particularly promising, especially with the rapid growth of IoT and connected devices. Smart sensors can potentially provide a wealth of data such as location, environmental conditions, and even product quality (especially for perishable goods). With more information and real-time updates on product status, supply chain managers will be able to make better and faster decisions. For example, if they know that a

batch of goods has deteriorated in quality midway through the supply chain process, they can activate a reserve batch instead of waiting until the bad batch arrives and then making a decision. This can help to reduce delays and keeps the supply chain agile. Organizations will also be more well-equipped to deal with disruptions like natural disasters, factory strikes, or delivery accidents. In a world with increasing product competition, being able to **maintain continuity of supply** can go a long way towards enhancing consumer confidence and their impressions of a brand [50].

The ability to trace the **chain of custody** of inventory throughout the supply chain becomes especially useful in the event of product recalls or safety incidents. As discussed previously, the Chipotle viral outbreak could have been contained and resolved faster had Chipotle been able to trace its food sources to the right suppliers. Once the contaminated food source was identified, they could have then determined which other stores had received goods from the same supplier and removed those items immediately. This could have limited the number of people who fell sick and reduced the damage done to Chipotle's reputation. The same can be applied to other goods that are recalled. For example, in the Takata exploding airbag issue that led to a massive recall of vehicles worldwide, blockchain could have allowed automakers and Takata to identify the affected customers and vehicles more quickly and accurately. The increased efficiency could help to lower their costs, and customers would also be safer and more satisfied with the quick resolution [51].

3) Efficiency

Using smart contracts in the supply chain can enhance efficiency in two aspects: process and cost. **Process efficiency** is improved as smart contracts executed on a distributed ledger help to simplify the complex multi-party systems present in typical supply chains. Given their self-executing nature, smart contracts can be used to automatically execute "contractual rights and obligations, including the terms for payment and delivery of goods and services" [48]. This eliminates much of the cumbersome paperwork traditionally required and reduces time

wastage. By programming the contracts to execute only when certain milestones are met, parties can also be more trusting and not have to worry about unfulfilled obligations. For instance, by only executing the contract to pay a supplier once the product has successfully reached a predetermined port within a specified timeframe, the manufacturer is protected against excessive loss from rogue suppliers; the supplier is also incentivized to ensure that it meets the requirements set in the contract or risk not getting paid.

The other way efficiency is enhanced is through **cost reduction**. Relying on trusted computer code that is easily customizable to execute contractual agreements precludes the requirement for numerous physical documents that need to be maintained by each party's purchasing, accounting or legal department [51]. This helps to lower costs and the amount of manual effort undertaken. Having a database where identity and reputation can be easily verified also saves on costs associated with conducting certification checks and building new trusted business relationships.

Referring to the functional complexity-automation capability framework developed by blockchain experts Don and Alex Tapscott [51], the benefits presented seem to correspond closely with solving the 'low-hanging fruit' opportunities illustrated (see Figure 10). Smart contracts can build on the progress achieved by Electronic Data Interchange (EDI) software through more efficient and real-time status tracking. Business-to-business (B2B) transactions are also made more transparent and trusted. Payments can be executed automatically and without the need for a costly intermediary. Product specifications can also be securely recorded and made accessible to all parties. Once real-world implementations prove that these 'low-hanging fruit' can be successfully resolved using smart contracts, further developments can be explored to address the more complex functions.

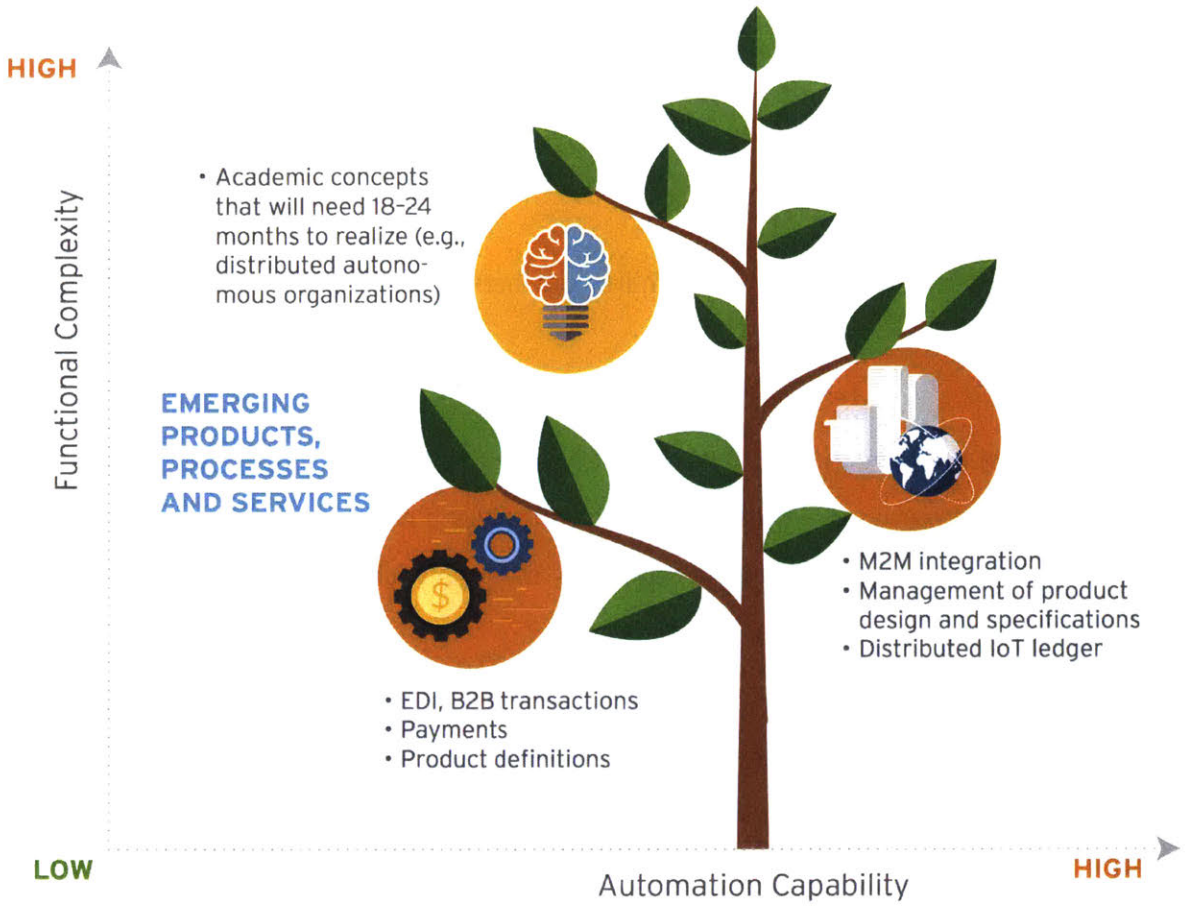


Figure 10: Functional complexity-automation capability framework (Source: [51])

5 Proposed Concept

5.1 Concept overview

A concept is proposed that seeks to use smart contracts to address three main challenges in supply chain management:

1) Determining the **provenance** of goods
2) **Tracking** the progress of goods through the supply chain
3) Building trust through an **open database** of supply chain partners, including their reputation

To help illustrate the concept, a simplified, generic supply chain for a basic consumer electronic product is used. The supply chain flow is shown in Figure 11. The OEM (red) is the focal organization and is in charge of coordinating the supply chain. The other parties are designated as either supply side (green) or demand side (blue). Supplier A sources the raw material and ships it to supplier B, who manufactures the individual components. The parts are then shipped to the OEM, who assembles the final product. The product is shipped to the distributor, who distributes it to the retailer to sell to the consumer.

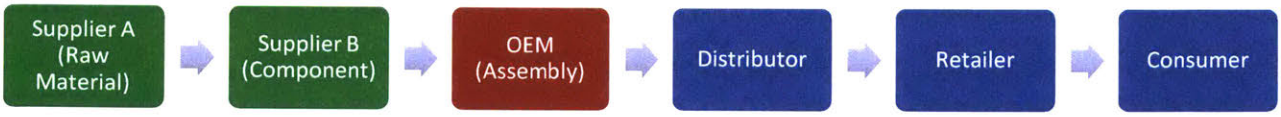


Figure 11: Supply chain stakeholders and flow for proposed concept

The proposed concept will run on a blockchain that all supply chain partners will be able to access (consumers can access certain functions such as checking provenance). It can be deployed and administered by either the focal organization or a neutral third party (e.g. blockchain startup like

Provenance, Modum.io, etc.). A caveat is that some level of existing trust in the party administering the blockchain is still required, so it is not completely trustless. However, this will allow more control and mediating actions can be easily carried out in the event something goes awry.

To determine **provenance**, supplier A – the origin of the supply chain – will be required to record its details and the details of its raw material to the blockchain. Through integration with a smart sensor, the location and time from which the raw material is shipped can be immutably stored. This will allow any of the other parties, including the consumer, to access the information through the blockchain (see Figure 12). By providing a quick way to determine provenance, raw material providers are kept accountable and parties can easily verify the source of their product.

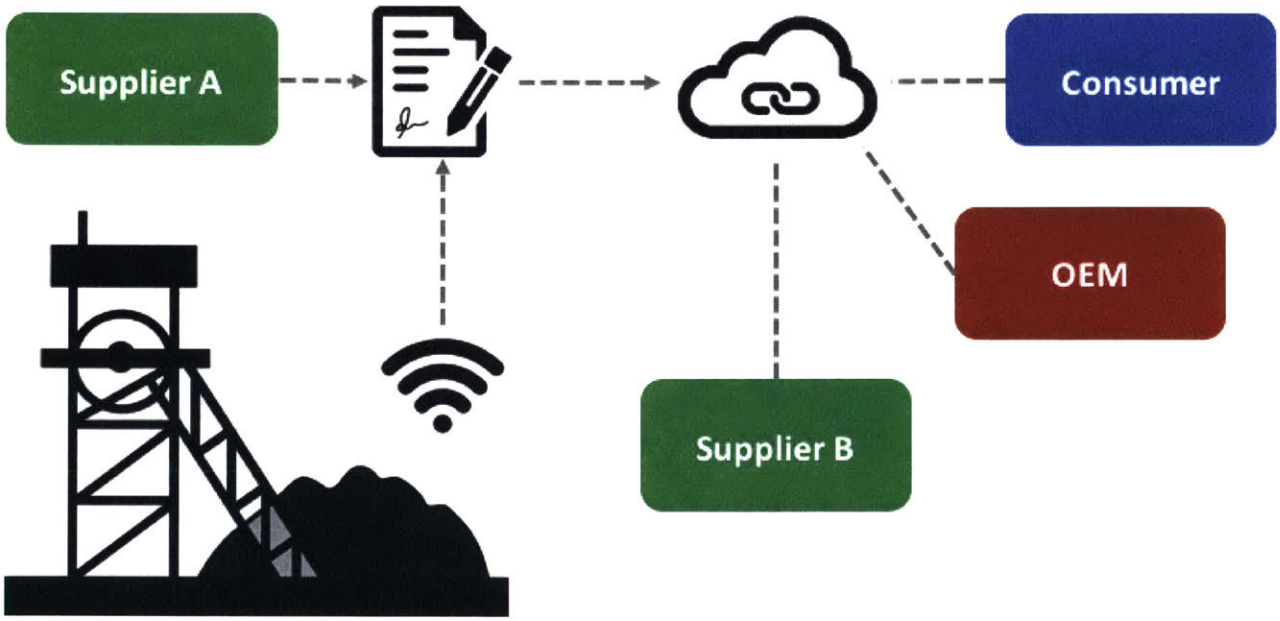


Figure 12: Using smart contracts to determine provenance

For **tracking** of the product along the supply chain, each party will be required to record the details to the blockchain whenever they send out a shipment or receive a shipment. When a shipment is received, the receiving party will confirm that everything is in order. If the shipment arrives on time and at the correct location (verified by smart sensors), a payment in the form of

cryptotokens is triggered to the shipping party. Figure 13 illustrates an example: the OEM predetermines the acceptable lead time and correct location for each leg of shipment (step 1). It also holds all the cryptotokens initially. Supplier A records the details on the blockchain when it ships raw material to supplier B (step 2). When supplier B receives the shipment, it records the details as well (step 3). The smart contract checks that the shipment tallies and whether the predetermined criteria have been met. If everything is in order (step 4a), a payment is executed from the OEM to supplier A (step 5); otherwise, an alert is triggered so that parties can rectify any problems (step 4b). Automated tracking and payment will help to simplify the processes within the supply chain and provide real-time updates on product status, so that parties can remain agile in the event of unforeseen circumstances.

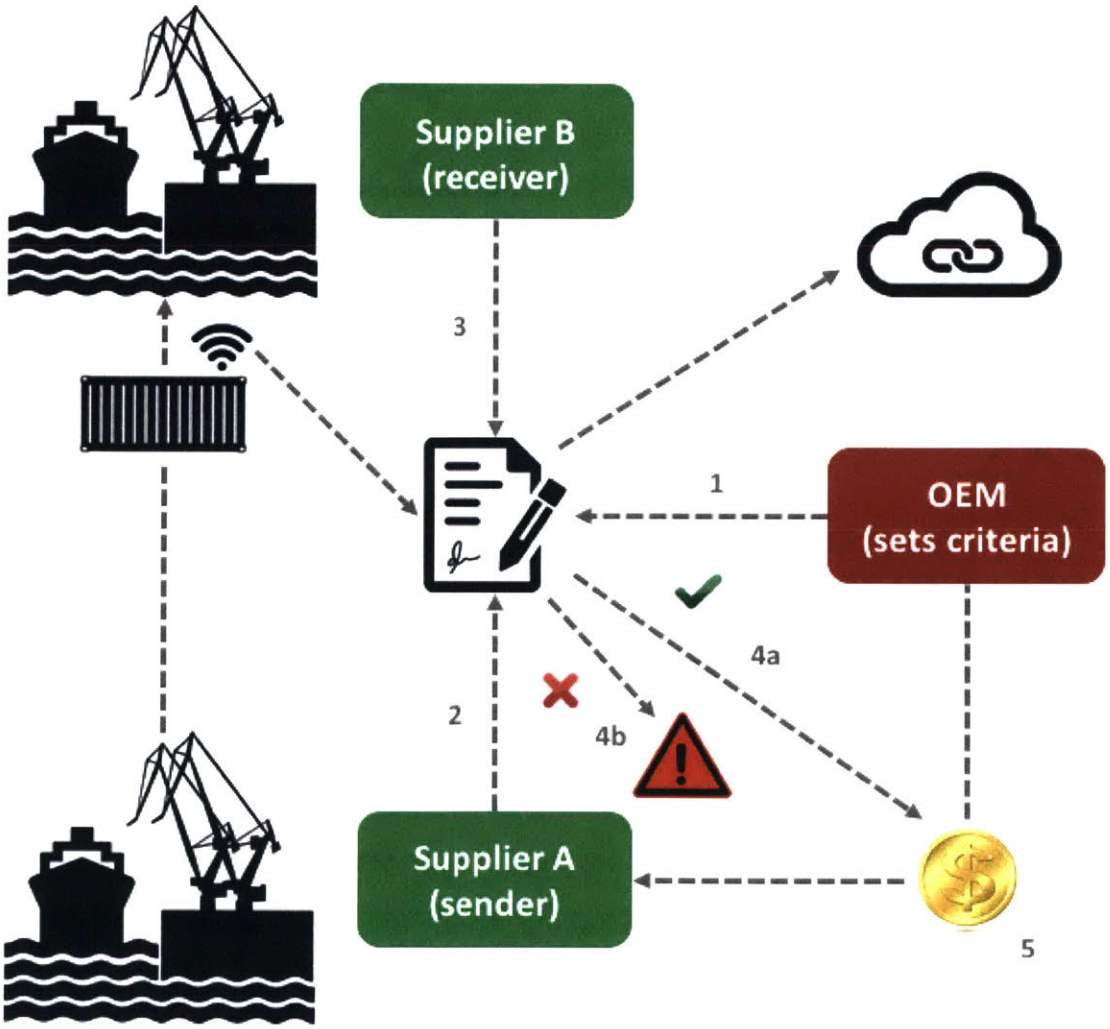


Figure 13: Using smart contracts to track goods and execute payment

Finally, to maintain an **open database** of information, all parties in the supply chain are required to record their details on the blockchain. In addition, a reputation score can be reflected. A simple way to determine reputation of a party is to calculate their number of successful shipments (i.e. correct item, quantity, location and on time) as a percentage of their total number of shipments made. This can be included as part of the smart contract for tracking goods, and the database smart contract can access the reputation score by calling that contract. This can be updated as more shipments are made (see Figure 14). The open database will incentivize parties to work towards maintaining a good reputation, and help build trust among parties with greater transparency.

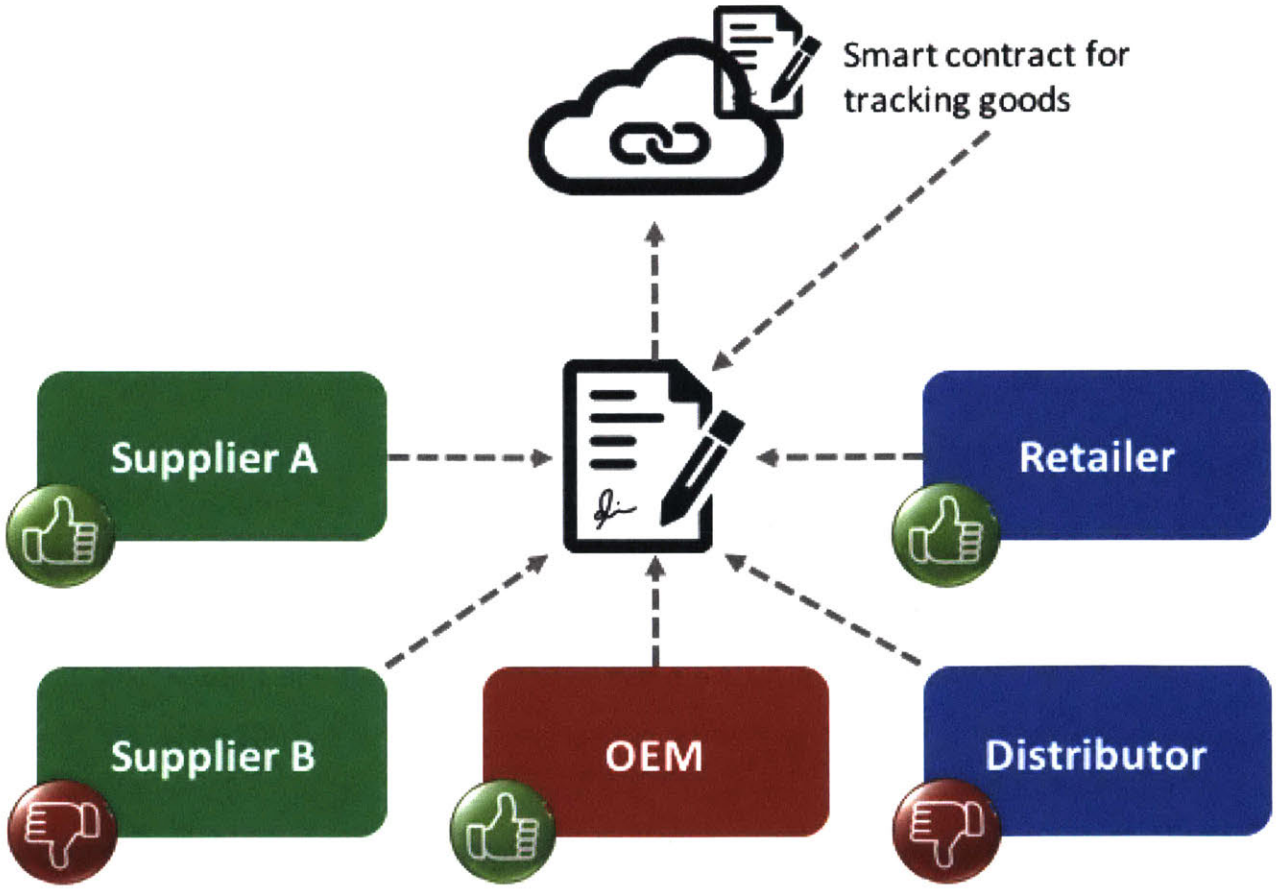


Figure 14: Using smart contracts for open database and reputation management

5.2 Proof-of-concept development

A proof-of-concept has been developed that can be run on Ethereum, which is an open blockchain platform for building and using decentralized applications. The proof-of-concept consists of three different smart contracts: Provenance, Tracking and Reputation. The smart contracts are written in Solidity, which is an object-oriented programming language that is widely used for writing smart contracts. Its syntax is similar to that of JavaScript and it is designed to target the Ethereum Virtual Machine (EVM) [52].

The Provenance smart contract allows supply chain parties and consumers to check the provenance of goods using their serial number or tag. It consists of the following functions that can be called once the contract has been deployed to the blockchain (see Appendix A for the complete smart contract code):

1) **addProducer** – this allows producers/suppliers to input their details and record them to the blockchain. Solidity uses a mapping to map the producer's Ethereum public address to a struct containing its details such as name, phone number, city, state and country of origin, as well as certification. Hence, the details can be easily retrieved subsequently with the producer's public address.

2) **removeProducer** – this allows the administrator of the contract (either focal organization or neutral third party) to remove a producer/supplier from the database in the event of any changes. Administrator access is required to prevent unnecessary tampering.

3) **findProducer** – this allows any party to display the details of a producer/supplier by entering their Ethereum public address. Consumers can use this to verify that the producer supplying their product is legitimate and certified. This function is costless as it does not require making any changes to the blockchain.

4) **certifyProducer** – this allows the administrator of the contract to certify a particular producer/supplier (perhaps upon receipt of necessary documentation). The certification status will be recorded on the blockchain and displayed together with the producer's other details. Administrator access is required to provide an additional layer of verification.

5) **addProduct** – this allows a producer/supplier to record to the blockchain each time their product is tagged at its origin. If integrated with a smart sensor, it can automatically record the location of the product. Once recorded to the blockchain, the producer's public address and block timestamp will also be automatically recorded. A mapping is used to link the product's serial number/tag with its details.

6) **removeProduct** – this allows the administrator of the contract to remove a product from the database in the event of any changes. Administrator access is required to prevent unnecessary tampering.

7) **findProduct** – this allows any party to display the details of a product by entering its serial number. It returns the producer's public address as well as location, date and time of origin. This function is costless as it does not require making any changes to the blockchain. Consumers can use this function together with the findProducer function to determine the provenance of their products.

The Tracking smart contract allows parties in a supply chain to track the shipment of goods and automatically execute payment in the form of tokens once every leg of shipment is completed, provided that certain predetermined criteria are met. There are two components to the contract – one for managing the tokens and the other for managing the shipments. Events are also used to display messages and details when transactions are executed on the blockchain. This provides more information than the default transaction hash that is displayed. The smart contract consists of the following functions (see Appendix B for the complete smart contract code):

1) **sendToken** – this allows tokens to be sent from one Ethereum account to another, by specifying the respective public addresses and the token amount. It has logic built in to check if there are enough tokens in the sender's account and to automatically update the balances of both accounts once the transaction has succeeded. Event messages are published to the blockchain to alert parties of the transactions.

2) **getBalance** – this allows any party to check the token balance of an account by entering the Ethereum public address. This function is costless as it does not require making any changes to the blockchain. Upon deployment of the contract to the blockchain, the number of initial tokens can be set and all tokens are initially held by the administrator. Separate arrangements will have to be made to agree on the value of the tokens and the allocation of initial tokens.

3) **recoverToken** – this allows the administrator of the contract to recover tokens from any account. It is a form of check and balance to prevent parties from abusing the sendToken function (since it can be used to send tokens from any account to another account). Similar to the sendToken function, logic is built in to check for sufficient balance and automatically update balances.

4) **setContractParameters** – this allows the administrator of the contract to predetermine the conditions that have to be met (and have been agreed upon by parties) before a shipment is successful and payment is released. It includes details for the shipping lead time, the shipping destination, and the payment amount (in the form of tokens).

5) **sendShipment** – this allows the sender to record details of a shipment on the blockchain once it has been dispatched. A mapping is used to map the tracking number/tag of the shipment to details such as the item and quantity. A smart sensor can be integrated to provide real-time location data. Once recorded to the blockchain, the time of dispatch

and sender's address are also captured. An event is triggered to inform parties that an item has been shipped.

6) **receiveShipment** – this allows the receiver to record details of a shipment on the blockchain once it has arrived. Two layers of logic are incorporated into this function (see Figure 15). The first checks that the item and quantity received match the item and quantity shipped. If they match, an event is triggered to log that the item has been received successfully. Then, the next layer of logic checks to see if the predetermined shipping lead time and destination have been adhered to. If they have, the sendToken function is automatically triggered to send payment to the shipping party. Otherwise, events are logged to capture the errors (i.e. payment not sent or item/quantity do not match).

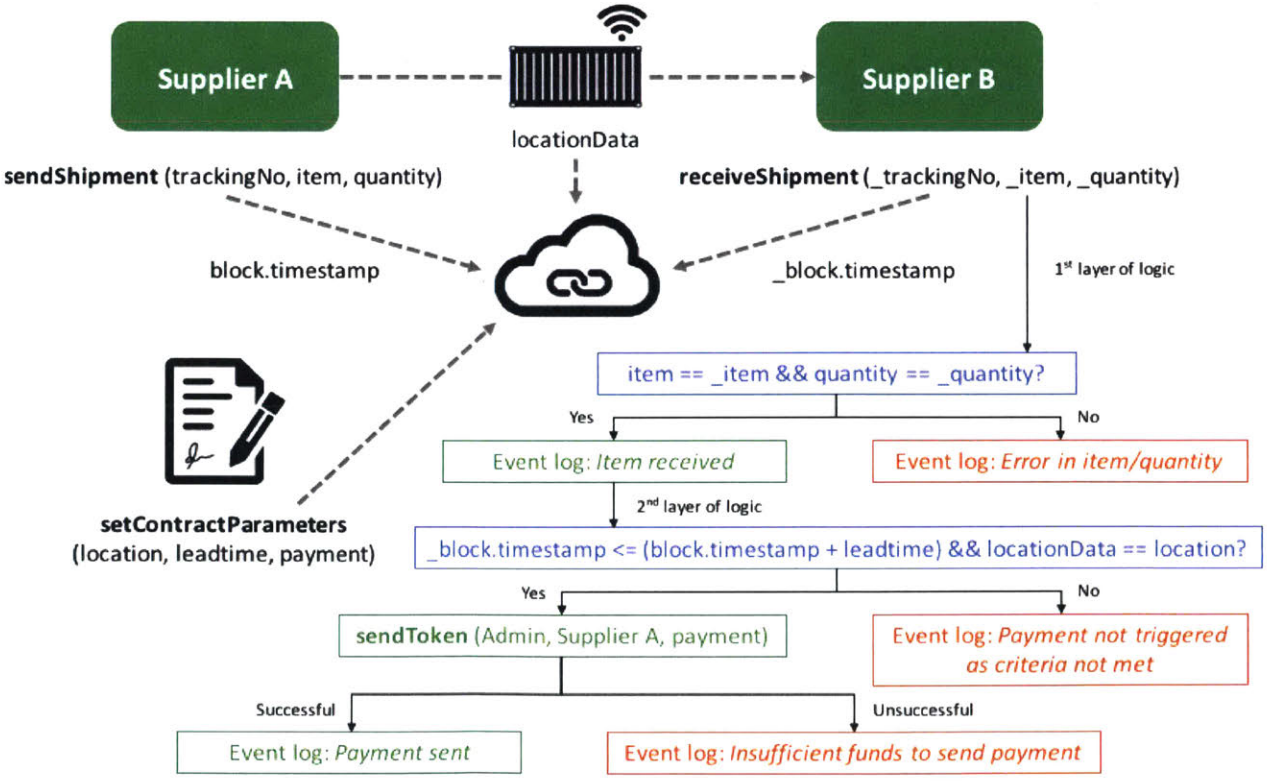


Figure 15: Logic flowchart for receiveShipment function

7) **deleteShipment** – this allows the administrator of the contract to remove a shipment from the database in the event of any changes. Administrator access is required to prevent unnecessary tampering.

8) **checkShipment** – this allows any party to display the details of a shipment by entering their tracking number. It returns the item, quantity, location (real-time if integrated with smart sensor), shipping timestamp, and sender. This function is costless as it does not require making any changes to the blockchain (caveat: it will not be costless if real-time location data is used).

9) **checkSuccess** – this allows any party to check the number of successful shipments made by a party in the supply chain, as well as their total number of shipments made. Successful shipments are defined as those that have met the predetermined criteria set in the contract (i.e. not late and delivered to the correct location). This function is costless as it does not require making any changes to the blockchain.

10) **calculateReputation** – this allows any party to calculate the reputation score of a supply chain partner. The reputation score is determined by taking the number of successful shipments as a percentage of the total number of shipments made by a particular party, and is expressed as an integer between 0 and 100. If a party has not made any shipments, its default reputation score is 0. This function is costless as it does not require making any changes to the blockchain. It is also essential to the proper functioning of the Reputation smart contract.

The Reputation smart contract maintains an open database of suppliers/parties in a supply chain that can be accessed by all. In addition to supplier details, it also tracks the reputation of each party. This is done by calling the Tracking smart contract that has been deployed on the blockchain and accessing the reputation score calculated in there. To make it more convenient to browse the list of suppliers, functions to filter by type of goods and reputation have been

created. The smart contract consists of the following functions (see Appendix C for the complete smart contract code):

1) **addSupplier** – this allows suppliers/parties to input their details and record them to the blockchain. A mapping is used to map the supplier's Ethereum public address to a struct containing its details such as name, phone number, city, state and country of origin, type of goods it specializes in, as well as reputation. Hence, the details can be easily retrieved subsequently with the supplier's public address. The address is also pushed into an array containing all of the parties' addresses.

2) **removeSupplier** – this allows the administrator of the contract to remove a supplier/party from the database in the event of any changes. Its public address is also removed from the array of consolidated addresses. Administrator access is required to prevent unnecessary tampering.

3) **findSupplier** – this allows any party to display the details of a supplier/party by entering their Ethereum public address. Parties can use this to browse suitable and preferred suppliers since information such as the type of goods and reputation are available. This function is costless as it does not require making any changes to the blockchain.

4) **allSuppliers** – this allows any party to display the complete list of suppliers/parties recorded on the blockchain. The array containing the list of Ethereum public addresses is returned, which parties can use to get further details. This function is costless as it does not require making any changes to the blockchain.

5) **filterByGoodsType** – this allows any party to search for suppliers by the type of goods that they specialize in. An array is created in memory with the same length as the complete supplier array. Logic is used to iterate across the complete supplier array and search for those with goods type matching the one specified in the query. The matches

are then returned in the new array. This function is costless as it does not require making any changes to the blockchain.

6) **filterByReputation** – similar to the previous function, this allows any party to search for suppliers by their reputation score. An array is created in memory with the same length as the complete supplier array. Logic is used to iterate across the complete supplier array and search for those with a reputation score equal to or higher than the score specified in the query. The matches are then returned in the new array. This function is costless as it does not require making any changes to the blockchain.

7) **checkReputation** – this allows any party to display the reputation score of a specified supplier/party by entering their Ethereum public address. It works by calling the deployed Tracking contract and running the calculateReputation function. Since both functions do not require making any changes to the blockchain, they are costless.

8) **updateReputations** – this allows the administrator of the contract to derive the latest updated reputation scores of all parties and update them on the blockchain. Since the reputation scores are stored when parties add their details through the addSupplier function, they may become outdated over time as more shipments are made. Hence, the administrator can use this function to update the reputations periodically. It iterates across the complete supplier array and for each public address, updates the reputation score using the calculateReputation function from the Tracking contract. Administrator access is assigned to prevent unnecessary usage.

A summary of the key components of each contract and their interactivity is shown in Figure 16:

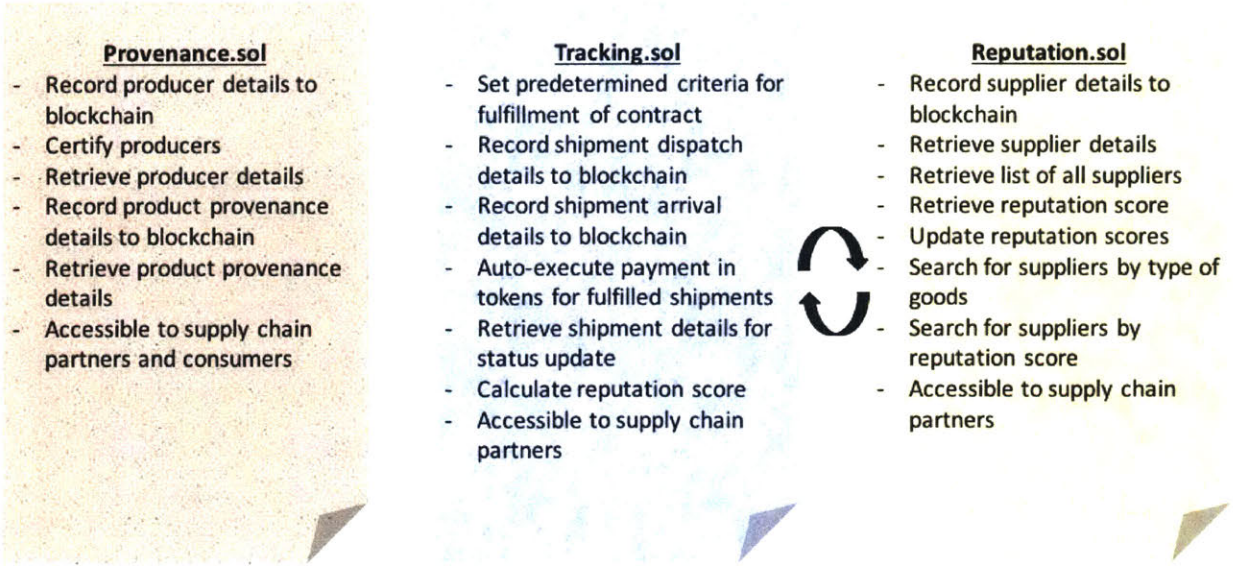


Figure 16: Summary of smart contract components and interactivity

5.3 Proof-of-concept validation

Remix, a browser-based Solidity compiler and integrated development environment (IDE) [53], is used to provide a preliminary validation of the proof-of-concept. It simulates the EVM, containing all of the tools necessary to create, compile, deploy and test smart contracts as they would be done on the actual Ethereum network, but does not actually connect to the live network. Since deploying contracts and executing transactions on the live Ethereum blockchain incurs costs in the form of Ether, Remix provides a developer-friendly environment to iterate, test and debug contracts without actually incurring real transaction fees. Its intuitive web interface also allows for quick and easy testing to be carried out. Contracts can be deployed and transactions executed with simple button clicks, and the output and costs are displayed correspondingly. Five test accounts are also available, so that different parties can be simulated when testing the contracts. Figure 17 provides a screenshot of the Remix IDE, where the costs of deploying a contract (Provenance.sol) and executing a transaction (in this case the addProducer function) and its output are displayed:

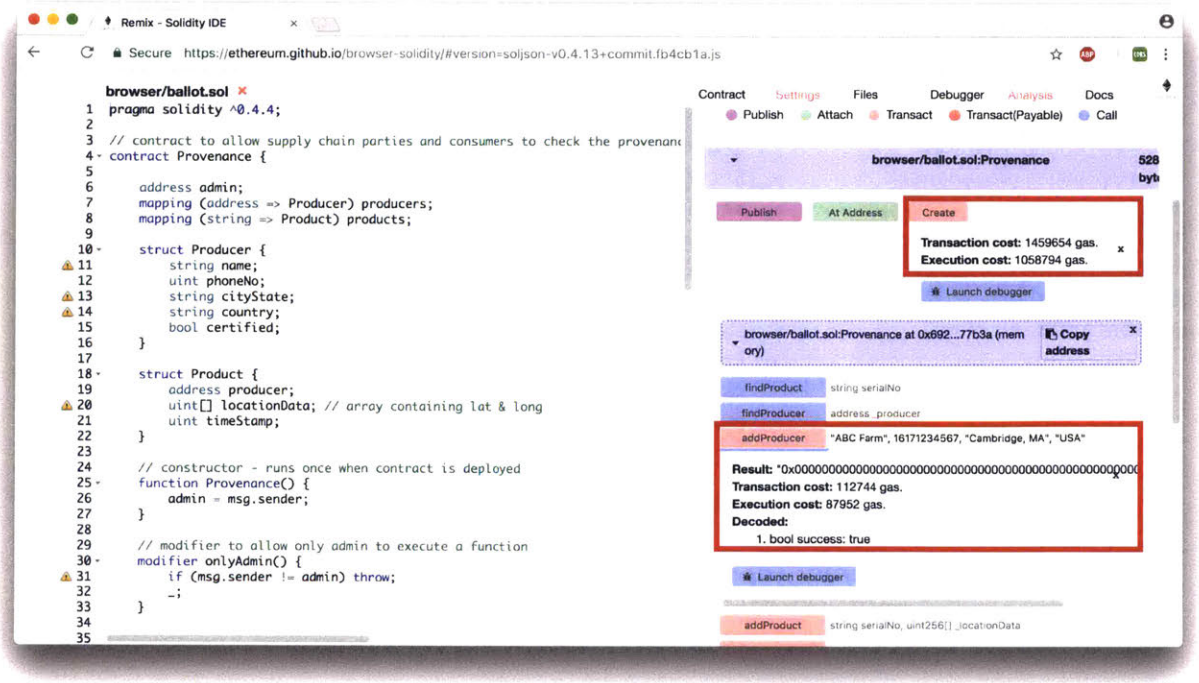


Figure 17: Remix - browser-based Solidity compiler and IDE

To validate the Provenance smart contract, one of the test accounts is used to simulate a producer/supplier, *ABC Farm*. It is able to add its details using the addProducer function, as well as add one of its products, *A00001*, using the addProduct function. Since there is no integration with smart sensors yet, arbitrary latitude and longitude data are used to provide the location. When the same account is used to try to get the producer to certify itself, an error is displayed (see Figure 18). This provides an additional layer of verification as only the administrator account (the one used to deploy the contract) can be used to certify the producer.

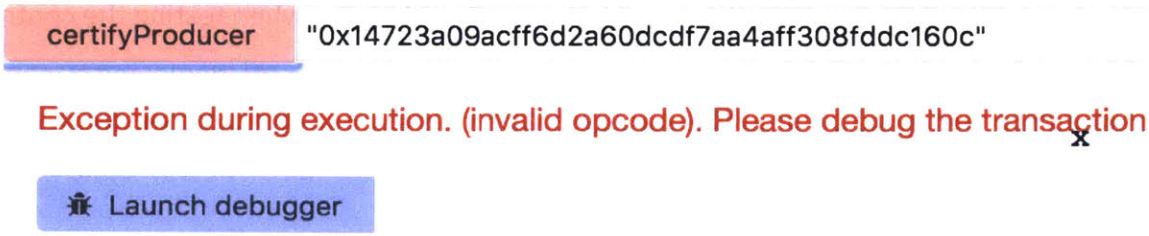


Figure 18: Error message when attempting to execute an administrator-only function

Another account is then used to simulate the consumer, who is able to retrieve details of his product, *A00001*, using the findProduct function. He can see the producer's public address, location of the product's origin, as well as the timestamp of when the product was recorded to the blockchain (the timestamp is displayed in Unix epoch time, which can be easily converted to human readable date and time). With the producer's public address, he can then use the findProducer function to retrieve details of the producer. Given the information provided, he can determine if the product originated from the same location as its producer, whether the producer is properly certified, and he can also contact the producer for any further clarifications (see Figure 19 for the output displayed by both functions).

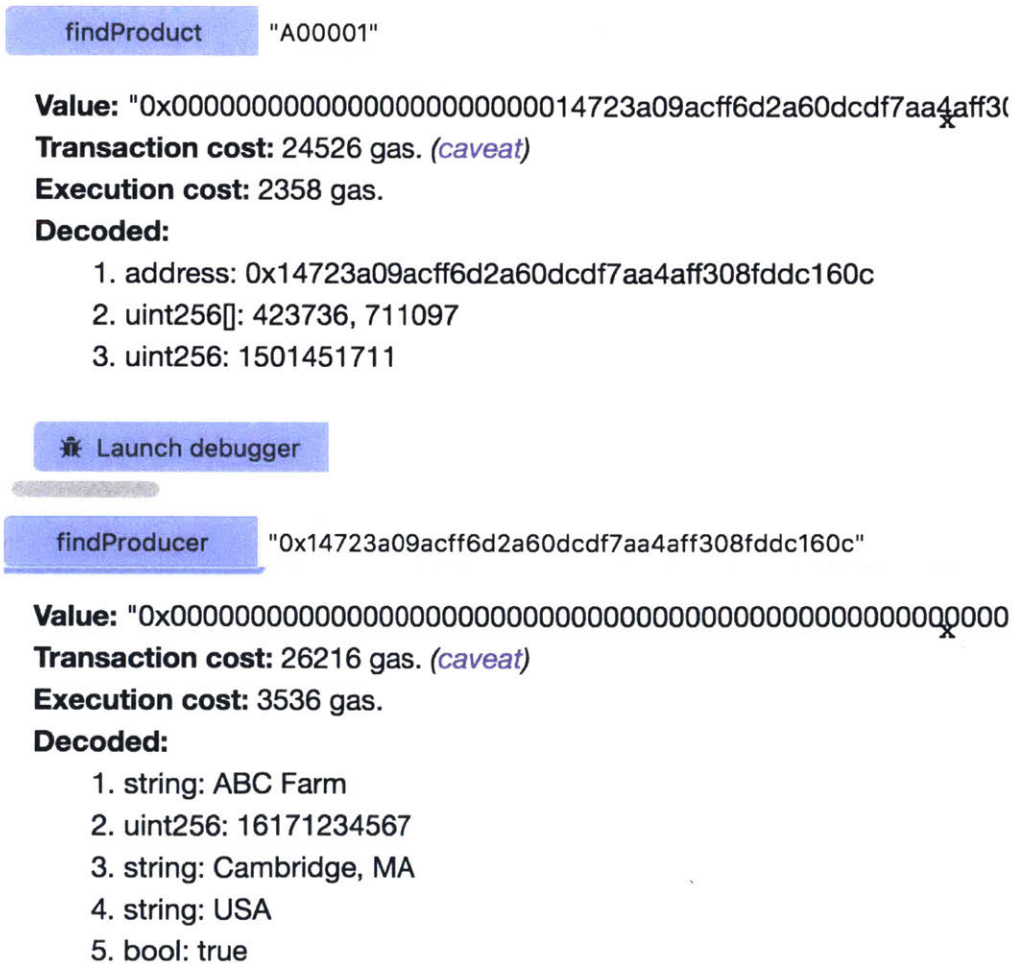


Figure 19: Outputs of findProduct and findProducer functions

To validate the Tracking smart contract, a scenario similar to the one described in Figure 15 is simulated. The contract is first deployed, with 1000 tokens initialized to the administrator. The administrator then sets the contract parameters for the shipment from supplier A to supplier B. When supplier A sends the shipment out, it records the details to the blockchain. When supplier B receives the shipment, it too records the details to the blockchain. Assuming some of the items went missing during the shipping process such that the quantity recorded by supplier B does not tally with the quantity recorded by supplier A, an alert is triggered so that necessary checks can be done (see Figure 20).

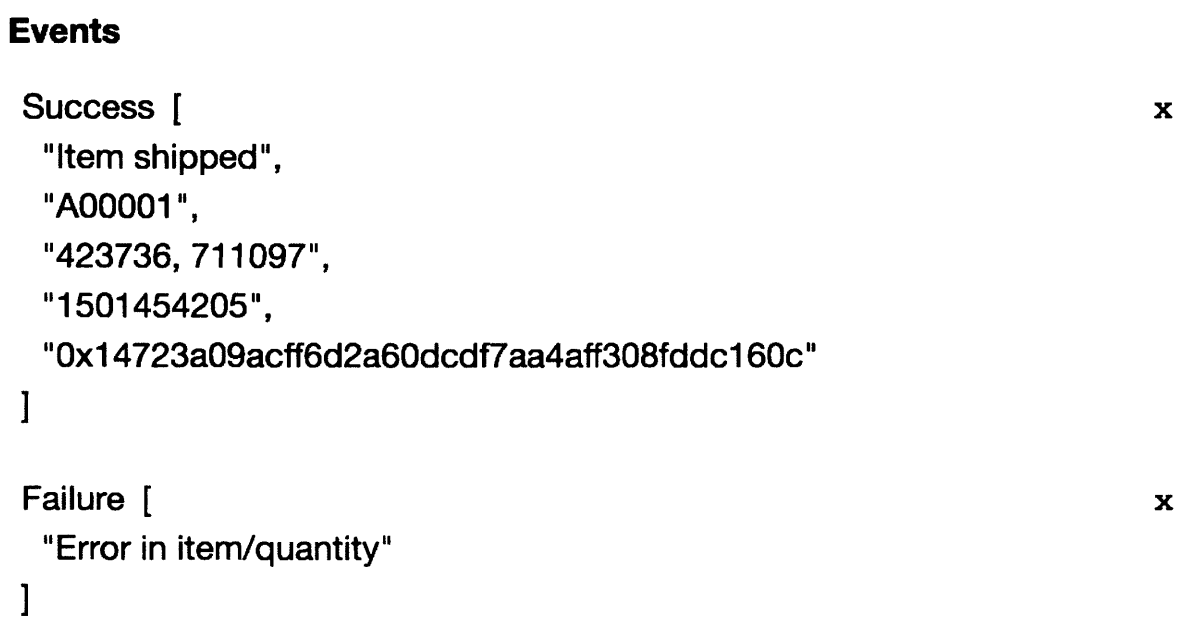


Figure 20: Alert triggered when item and/or quantity do not match

Now assume that the shipment is intact and that the items and quantities match. However, the item took longer to arrive than initially set by the administrator (based on the blockchain timestamp). In this case, even though the item has been successfully received, the payment is not triggered as the predetermined criteria have not been met (see Figure 21).

Events

```
Success [                                                    x
  "Item shipped",
  "A00001",
  "423736, 711097",
  "1501454205",
  "0x14723a09acff6d2a60dcdf7aa4aff308fddc160c"
]

Success [                                                    x
  "Item received",
  "A00001",
  "368508, 762858",
  "1501454703",
  "0x4b0897b0513fdc7c541b6d9d7e929c4e5364d2db"
]

Failure [                                                    x
  "Payment not triggered as criteria not met"
]
```

Figure 21: Alert triggered when predetermined contract criteria are not met

Only when both levels of logic are satisfied is the payment executed (i.e. item and quantity match, predetermined criteria met). The predetermined token amount (50 tokens, in this case) is automatically sent from the administrator's account to supplier A's account. A check using the getBalance function confirms this (see Figure 22).

Figure 22: Confirmation of token payment when all conditions are satisfied

The same scenario is applied to validate the Reputation smart contract. It calls the same Tracking contract that has been deployed in the previous test. Given that supplier A has made one shipment and it has been successfully received, its reputation score is now 100 (1 out of 1). After adding supplier A's other details using the addSupplier function and displaying them using the findSupplier function, we can see that the reputation score has been incorporated successfully (see Figure 23). This also shows that the contracts are able to interact with each other while deployed on the blockchain.

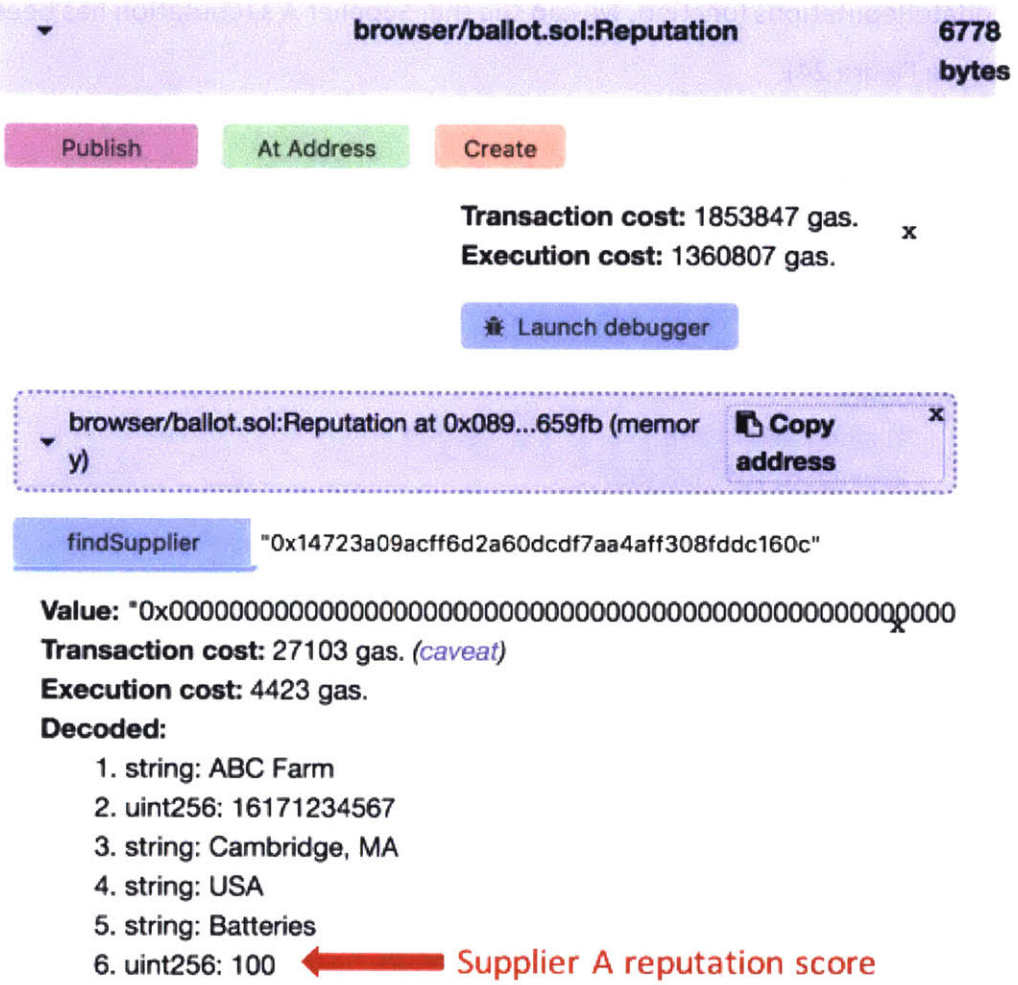


Figure 23: Reputation contract deriving reputation score from Tracking contract

A few more suppliers are added according to the table below. Supplier A (*ABC Farm*)'s reputation score is also decreased from 100 to 75:

Name	Public Address	Goods Type	Reputation
ABC Farm	0x14723a09acff6d2a60dcdf7aa4aff308fddc160c	Batteries	75
DEF Company	0x4b0897b0513fdc7c541b6d9d7e929c4e5364d2db	Wafers	50
GHI Company	0x583031d1113ad414f02576bd6afabfb302140225	LCD	100
JKL Company	0xdd870fa1b7c4700f2bd7f44238821c26f7392148	Batteries	0

Using the updateReputations function, we can see that Supplier A's reputation has been updated accordingly (see Figure 24).

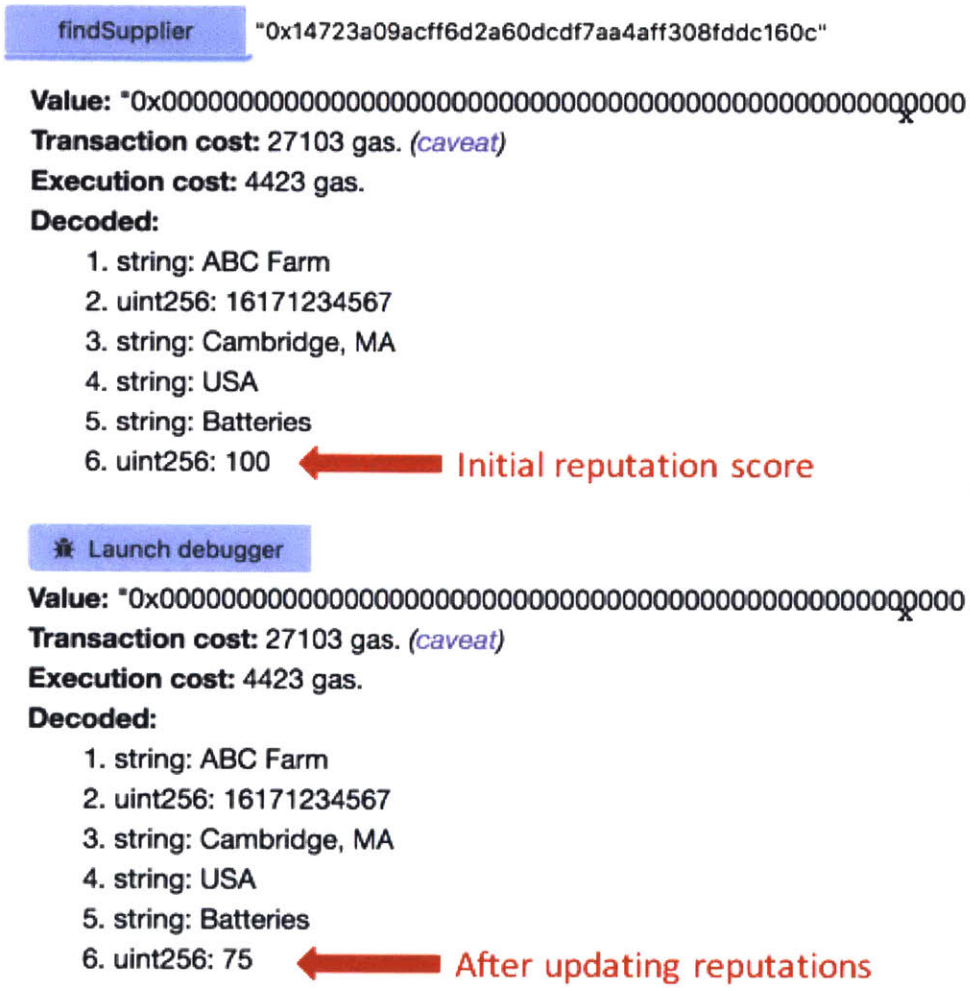


Figure 24: Updated reputations after calling updateReputations function

The two filtering functions can also be tested. Filtering by goods type for "Batteries", we see that the output displays the public addresses for *ABC Farm* and *JKL Company*. Filtering by reputation for any greater than or equal to 75, we see that the output displays the public addresses for *ABC Farm* and *GHI Company*. Hence, both filtering functions work as expected (see Figure 25).

Figure 25: Outputs of filterByReputation and filterByGoodsType functions

All three smart contracts have been comprehensively tested using Remix and all of their functions work as intended. Further validation can be done using the testrpc node [54] together with the Truffle development framework [55], which would allow for testing using more accounts (ten instead of five). Alternatively, the contracts can be tested on a private test network or deployed to Ethereum's public testnet, Ropsten. These networks function exactly the same as the live network, but use 'test' Ether that can be mined for free rather than real Ether.

6 Discussion

6.1 Proof-of-concept

As a preliminary implementation, the proof-of-concept provides a simple way to address some of the supply chain management challenges stated in Chapter 4. It increases transparency by recording the provenance of goods and certification of supply chain partners on the blockchain, so that the information is easily accessible to all parties, including consumers. This can provide assurance that the goods are coming from legitimate sources and will be helpful in the event of disruptions such as food safety recalls. The proof-of-concept also improves traceability as the chain of custody of goods can be tracked as they are shipped along the supply chain. Real-time updates help to manage risk by keeping the supply chain agile and ready to respond to any disruptions. By integrating automatic payments with successful delivery of goods, supply chain efficiency is enhanced as well. Finally, having an open database with details and reputations of all supply chain partners helps to build up trust within the supply chain. Parties will not have to waste time and money for extra verification checks, and are also incentivized to work towards maintaining a good reputation.

Managing these smart contracts on the blockchain is beneficial as it provides an immutable record that can be easily audited and is openly accessible to the relevant parties. On top of the transaction hashes that are generated each time a transaction is successfully recorded to the blockchain, the use of events helps to clearly indicate when pertinent transactions have occurred (e.g. when goods have shipped, have been received, or when payment has been executed). They also trigger alerts whenever certain transactions are unsuccessful, so that parties can identify the problem quickly and easily and work towards rectifying it as soon as possible. This is more efficient than manual systems currently in use, where errors can go unnoticed for long periods of time. Having a standardized contract template also ensures that all parties share information in the same way. This reduces the risk of misinterpretation or loss of information due to different types of forms used, and eliminates the need to maintain different methods of data entry. In

general, efficiency of the entire supply chain is improved as more information is shared and easily obtained, helping parties to remain on the same page.

Although the proof-of-concept has been extensively tested using Remix to simulate the EVM, it has not been tested on an actual Ethereum network. This would be the logical next step of validation, where transaction times and costs will become a bigger factor. Should certain transactions require a significant amount of 'gas' (the cost to execute transactions, that is paid in Ether), they might prove too costly to execute on a frequent basis. In this case, a more computationally efficient way to code the functions might be necessary. Another improvement would be to develop a user interface (UI) that allows parties to easily interact with the deployed smart contracts. This can be in the form of a mobile application or web browser using the Web3 JavaScript application programming interface (API).

A significant benefit of using smart contracts in supply chain management is premised on their ability to interact with IoT devices such as smart sensors to obtain real-time information. Although the proof-of-concept was developed with this interactivity in mind, testing was not carried out with smart sensors. Since Ethereum requires all nodes to agree on the outcome of a computation, the smart contracts can only execute strictly deterministic functions and cannot perform native operations such as random number generation or API calls [29]. Hence, in order to obtain external data such as those from smart sensors, they need to rely on 'oracles'. An oracle – or data feed – is a trusted third-party service that sources real-world data that is requested and pushes it to the blockchain to be used by smart contracts [56]. Specifically, hardware oracles obtain information from the physical world, such as smart sensors. The challenge they face is how to ensure the data is transmitted securely without tampering. Companies such as Ledger are working to develop cryptographically attestable anti-tampering sensors that can be used as hardware oracles linking real-world data with the blockchain (see Figure 26) [57]. This should be explored when later-stage testing of the concept is done on the live Ethereum network.

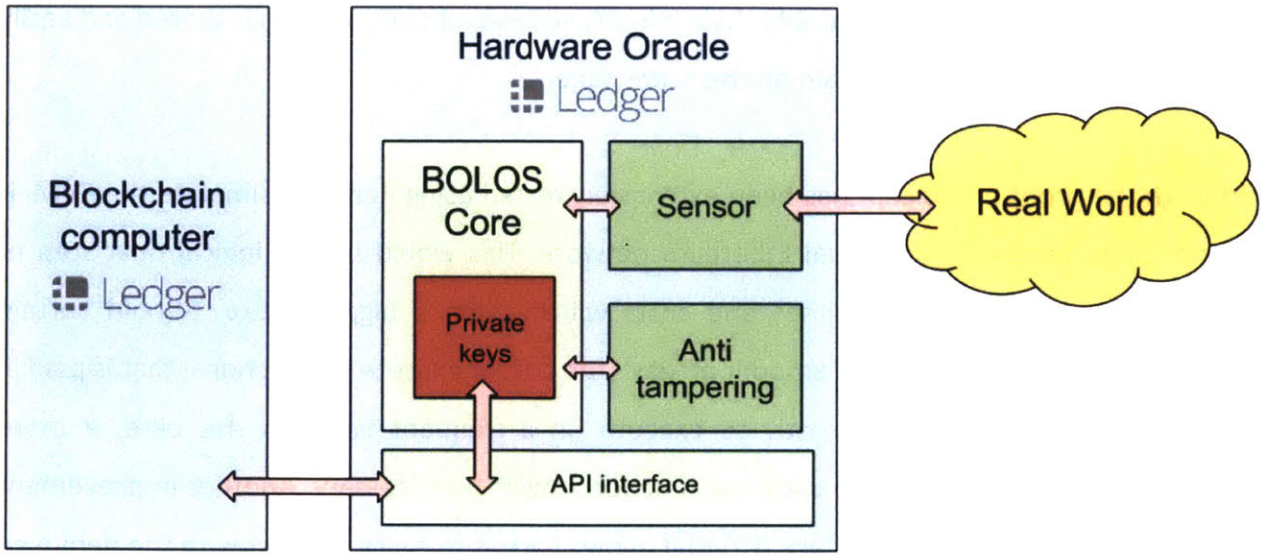


Figure 26: Secure hardware oracle proposed by Ledger (Source: [57])

Although blockchain promises to allow trustless exchanges with its decentralized technology, the proof-of-concept in its current state still requires some level of trust, as there are some functions that are controlled by the administrator of the contracts. Hence, there has to be implicit trust that the administrator – whether it is the focal organization in the supply chain or a neutral third-party blockchain solutions provider – will not abuse its additional power. There remains some risk of fraud through inaccurate reporting of information by supply chain partners or tampering with goods. Even as the introduction of tamper-proof hardware oracles can go some way towards preventing this, more research can be done to explore other ways to reduce the risk of security breaches.

Finally, the reputation score calculated in the proof-of-concept is a very simplistic way of representing the overall reputation of a party, since it is only based on the proportion of successful shipments made (i.e. shipment history). A more sophisticated mechanism can be explored, perhaps using additional factors such as reliability, online reviews, or product range. This can help to provide a more robust and accurate representation of a party's reputation.

6.2 Similar real-world concepts

The use of smart contracts in supply chain management has the ability to solve many of its challenges and could even potentially revolutionize the industry. Therefore, it is no surprise that major industry players are devoting significant resources to blockchain technology. Microsoft has launched its Blockchain-as-a-Service (BaaS), built on the Microsoft Azure platform, to collaborate with organizations and experiment with new business processes [58]. It is also a member of the Enterprise Ethereum Alliance, exploring the use of Ethereum and smart contracts to streamline future business operations [30]. IBM has made its blockchain software open-source, starting the Hyperledger Foundation to collaborate on blockchain technologies across industries and countries, in an effort to accelerate their development towards mainstream commercial adoption. It has partnered with giants in the supply chain industry, working with Maersk to track shipping containers across the globe and prevent tampering of shipping documents. It has also conducted pilot tests with Walmart to track the movement of pork from Chinese farms to stores, and produce from Latin America to the United States [36].

Apart from the bigger players, several blockchain startups have also emerged in the supply chain field. Everledger, part of the Hyperledger community, uses blockchain technology to track the provenance of diamonds and their movement from mines to jewelry stores. This allows consumers to screen for diamonds that have been mined in regions where there is forced labor or where diamond sales have been used to fund violence [59].

Provenance, a London-based company, uses a mobile application to link parties across a supply chain (including the consumer). It promotes transparency and traceability by encouraging companies to provide information on the source of their products and the environmental impact of their business practices. In essence, the aim is for every product to tell a story of its journey from beginning to end. It has successfully implemented a pilot to track tuna in Indonesia from catch to consumer [60].

Another company, Mojix, is proposing to use Microsoft's BaaS platform to introduce real-time, hands-free inventory tracking using RFID. By tracking the delivery of goods each step along the way, greater trust is fostered and overhead costs are reduced. The technology can also be used to manage inventory for retailers, giving them "up-to-the-minute status on inventory and replenishment information at the SKU (stock keeping unit), department or store level" [61].

Swiss-based startup, Modum.io, combines smart sensors with blockchain technology to help enhance efficiency of the pharmaceutical supply chain. The sensors record environmental conditions such as temperature during shipment. Upon delivery, the collected data is checked against a smart contract deployed on the Ethereum blockchain to determine if the standards set out in the contract have been met, after which follow-on actions are triggered [62]. This is especially important in the pharmaceutical industry, which sets stringent requirements on product standards.

6.3 Future of blockchain

The future bodes well for blockchain technology. As more organizations devote resources to explore the use of blockchain in enhancing business operations across a diverse range of industries, there is hope that blockchain could have as significant an impact as the Internet did, given its potential for widespread application. In the midst of all the hype, however, it is important to understand and work towards overcoming the current challenges that are preventing blockchain technology from achieving more mainstream adoption.

Although blockchain technology promises increased transparency and trust among parties, the technology itself will not be effective if there is no corresponding mindset shift. In today's highly competitive corporate landscape, organizations tend to be uncomfortable with the notion of complete sharing of information with other parties [63]. This is exacerbated by the fact that on the blockchain, many of these parties could be unseen network partners, different from the traditional way of conducting business exchanges. In order for the blockchain to succeed,

participating organizations must be open to this new change. Thus, it is important to continue to educate the public about blockchain technology, particularly since it is a relatively new and unfamiliar technology. This will ultimately help organizations to see the value proposition that blockchain brings to them.

As is the case for nascent technologies, there is still a lack of standards and regulations to govern the use of blockchain. The legal implications of smart contracts have to be further explored [39], and industries have to come up with best practices that are most applicable to their particular industry. With the presence of both public and private blockchains, standards and agreements will also be needed to ensure interoperability [48]. Being a public ledger that is not owned by any single entity, divisions can arise when addressing the technology's development. For example, both Bitcoin and Ethereum have had to deal with disagreements in their open-source communities, making it difficult to agree on protocol upgrades and causing 'forks', where smaller groups split off from the main group. Thus, much volatility still exists in the progression of blockchain technology.

The transition period when blockchain technology begins to see more widespread implementation will also have to be carefully managed. Its benefits and risks must be weighed against existing systems, and there should also be solutions to integrate blockchain with legacy systems as seamlessly as possible, to ensure minimal loss in performance or data security [51].

In its 2016 Hype Cycle for Emerging Technologies, Gartner placed blockchain at the 'Peak of Inflated Expectations', with a five- to ten-year timeframe before mainstream adoption (see Figure 27). This is the period characterized by the greatest hype surrounding the technology. There is much publicity due to some early success stories, and more and more players try to enter the field. As blockchain proceeds towards the 'Trough of Disillusionment', interest may start to fade as experiments and implementations fail to deliver. Many will fail and exit the market, but those that succeed will continue to grow and help push the technology towards maturity [64]. If organizations are open to changing their business processes and proper standards and

regulations are implemented to govern blockchain, there is strong potential for the technology to reach the 'Slope of Enlightenment' and beyond, eventually achieving mainstream adoption.

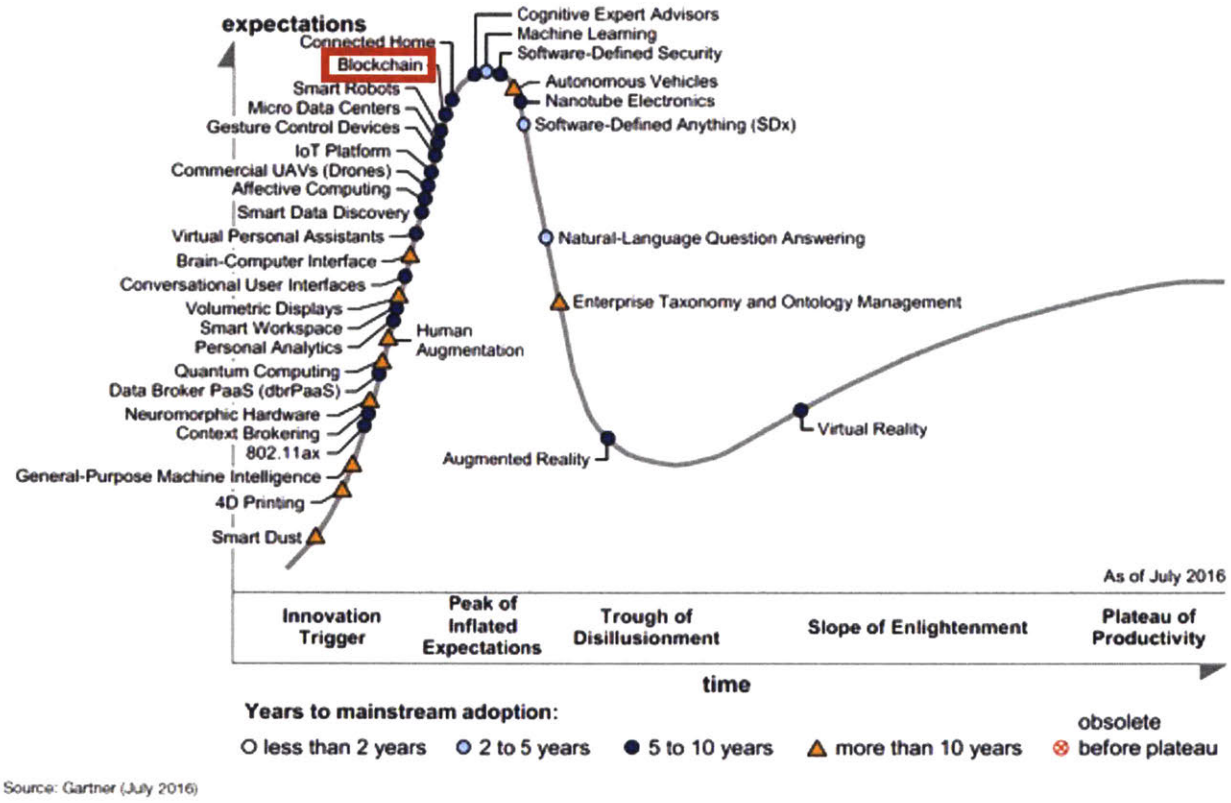


Figure 27: Hype Cycle for Emerging Technologies, 2016 (Source: adapted from [65])

7 Conclusion

The seeds of blockchain technology were sown in 2008, when Satoshi Nakamoto published a white paper introducing Bitcoin, a peer-to-peer (P2P) electronic payment system based on cryptography. As Bitcoin gained more prominence beyond the coding community, its underlying technology – blockchain – began to garner more interest as well. Since then, many different applications of blockchain technology have been conceived, moving beyond the financial realm.

One area in which blockchain promises to be particularly beneficial is supply chain management. Currently, many challenges exist in coordinating supply chains. With so many stakeholders involved, process flow is complex and there tends to be inadequate sharing of information. Without an efficient method to keep track of goods in real-time, the supply chain is also ill-equipped to manage risk and respond to disruptions quickly. Safety issues continue to be a concern, as evidenced by the Takata airbag and Chipotle food poisoning incidents. Hence, there is greater pressure on the supply chain to become more transparent and openly declare the provenance of its goods. Finally, trust is also a big issue when managing suppliers and partners across a diverse, global supply chain.

Blockchain has the potential to address these supply chain management challenges with its distributed ledger technology that generates an open, immutable record of transactions that is easily accessible by all relevant parties. Specifically, the use of smart contracts – self-executing agreements – deployed on the blockchain can help to increase transparency, traceability and efficiency across a supply chain. They can be used to determine provenance, provide transparent credentialing and reputation management, track the flow of goods through a supply chain, and automatically execute payments. This helps to build up trust among supply chain partners and reduces time and cost inefficiencies associated with overlapping effort and additional verification checks. It also speeds up the supply chain and makes it more agile in managing risks and disruptions, since real-time data can be used to guide decision-making.

To validate this concept, a proof-of-concept was created and tested on Ethereum, a blockchain platform for decentralized application development. Three different smart contracts were coded to determine provenance, track goods, and maintain an open database, respectively. The contracts provide a standardized template for supply chain partners to record, share and access information, helping to eliminate ambiguity. Parties (including consumers) can check the provenance of their goods and certification status of the corresponding producer. They can also track the chain of custody as goods flow from one party to the next through the supply chain. By integrating smart sensors, real-time location data can be provided. Upon fulfillment of the shipments and contractual agreements, payment in the form of cryptotokens can be automatically executed. Finally, a database of supply chain partners can be recorded on the blockchain. This will include their reputation scores, which are derived from the number of successful shipments made. This can incentivize parties to maintain a good reputation, and helps to build trust among supply chain partners with open access to information. Although the proof-of-concept works as intended after preliminary testing, more testing on the live Ethereum blockchain and integration with smart sensors will be needed to further develop it into a viable solution.

The list of companies exploring the use of blockchain and smart contracts continues to grow, as hype over the technology continues unabated. There are already several startups looking to disrupt the supply chain industry, tracking a wide variety of goods ranging from produce to pharmaceutical products and even diamonds. The surge in blockchain-related companies brings much promise, as the technology can have far-reaching implications across multiple industries, but also the potential of unfulfilled expectations. There are still several challenges that must be overcome before blockchain can reach mainstream adoption. Organizations must be receptive to a shift towards a more open and transparent culture, and standards and regulations have to keep pace with the growth of the technology. If these and other challenges are adequately addressed, it would not be a stretch to believe that one day, blockchain could fundamentally change the way we exchange value and have the same revolutionary impact that the Internet had when it was first introduced.

Bibliography

[1] "supply chain (SC)," *WhatIs.com*. [Online]. Available: http://whatis.techtarget.com/definition/supply-chain. [Accessed: 13-Jul-2017].

[2] H. Canitz, "The Biggest Challenges Supply Chain Leaders Will Crush in 2016," *Supply Chain 24/7*. [Online]. Available: http://www.supplychain247.com/article/the_biggest_challenges_supply_chain_leaders_will_crush_in_2016. [Accessed: 13-Jul-2017].

[3] N. Boariu, "Major Issues Facing Supply Chain Managers," *Procurify*, 10-Jun-2015. [Online]. Available: https://blog.procurify.com/2015/06/10/4-major-issues-facing-your-supply-chain-manager/. [Accessed: 13-Jul-2017].

[4] T. Gyorey, M. Jochim, and S. Norton, "The challenges ahead for supply chains: McKinsey Global Survey results," *McKinsey & Company*. [Online]. Available: http://www.mckinsey.com/business-functions/operations/our-insights/the-challenges-ahead-for-supply-chains-mckinsey-global-survey-results. [Accessed: 13-Jul-2017].

[5] M. Hudnurkar, S. Jakhar, and U. Rathod, "Factors Affecting Collaboration in Supply Chain: A Literature Review," *Procedia - Social and Behavioral Sciences*, vol. 133, pp. 189–202, May 2014.

[6] A. Renner, "5 Data-Driven Supply Chain Challenges to Overcome in 2016," *Spend Matters*, 23-Dec-2015. [Online]. Available: http://spendmatters.com/2015/12/23/5-data-driven-supply-chain-challenges-to-overcome-in-2016/. [Accessed: 13-Jul-2017].

[7] A. Marucheck, N. Greis, C. Mena, and L. Cai, "Product safety and security in the global supply chain: Issues, challenges and research opportunities," *Journal of Operations Management*, vol. 29, no. 7, pp. 707–720, Nov. 2011.

[8] J. H. Trienekens, P. M. Wognum, A. J. M. Beulens, and J. G. A. J. van der Vorst, "Transparency in complex dynamic food supply chains," *Advanced Engineering Informatics*, vol. 26, no. 1, pp. 55–65, Jan. 2012.

[9] S. New, "McDonald's and the Challenges of a Modern Supply Chain," *Harvard Business Review*, 04-Feb-2015. [Online]. Available: https://hbr.org/2015/02/mcdonalds-and-the-challenges-of-a-modern-supply-chain. [Accessed: 13-Jul-2017].

[10] I.-W. G. Kwon and T. Suh, "Factors Affecting the Level of Trust and Commitment in Supply Chain Relationships," *Journal of Supply Chain Management*, vol. 40, no. 1, pp. 4–14, Mar. 2004.

[11] S. Beth *et al.*, "Building Relationships," *Harvard Business Review*, 01-Jul-2003. [Online]. Available: https://hbr.org/2003/07/building-relationships. [Accessed: 13-Jul-2017].

[12] M. Iansiti and K. R. Lakhani, "The Truth About Blockchain," *Harvard Business Review*, 01-Jan-2017. [Online]. Available: https://hbr.org/2017/01/the-truth-about-blockchain. [Accessed: 15-Jul-2017].

[13] D. Tapscott and A. Tapscott, "How Blockchain Will Change Organizations," *MIT Sloan Management Review*, 07-Dec-2016. [Online]. Available: http://sloanreview.mit.edu/article/how-blockchain-will-change-organizations/. [Accessed: 15-Jul-2017].

[14] Z. Church, "Blockchain, explained," *MIT Sloan School of Management*, 25-May-2017. [Online]. Available: http://mitsloan.mit.edu/newsroom/articles/blockchain-explained/?utm_source=mitsloanlinkedinu&utm_medium=social&utm_campaign=blockchainexplainer. [Accessed: 12-Jul-2017].

[15] R. Marvin, "Blockchain in 2017: The Year of Smart Contracts," *PCMAG*, 12-Dec-2016. [Online]. Available: https://www.pcmag.com/article2/0,2817,2510524,00.asp. [Accessed: 15-Jul-2017].

[16] B. Warburg, *How the blockchain will radically transform the economy*. 2016.

[17] C. Catalini, "How Blockchain Applications Will Move Beyond Finance," *Harvard Business Review*, 02-Mar-2017. [Online]. Available: https://hbr.org/2017/03/how-blockchain-applications-will-move-beyond-finance. [Accessed: 15-Jul-2017].

[18] M. Pilkington, "Blockchain Technology: Principles and Applications," Social Science Research Network, Rochester, NY, SSRN Scholarly Paper ID 2662660, Sep. 2015.

[19] C. Catalini and J. S. Gans, "Some Simple Economics of the Blockchain," Social Science Research Network, Rochester, NY, SSRN Scholarly Paper ID 2874598, Nov. 2016.

[20] J. Heggestuen, "These Are The Five Main Reasons Bitcoin Is Beginning To Flourish As A Payment Technology," *Business Insider*, 02-Jun-2014. [Online]. Available: http://www.businessinsider.com/five-main-reasons-bitcoin-is-beginning-to-flourish-as-a-payment-technology-2014-5. [Accessed: 15-Jul-2017].

[21] C. Catalini, "The Firm as a Nexus of Smart Contracts? How Blockchain and Cryptocurrencies Can Transform the Digital Economy," *Yale Journal on Regulation: Notice & Comment*, 07-Jun-2017. [Online]. Available: http://yalejreg.com/nc/the-firm-as-a-nexus-of-smart-contracts-how-blockchain-and-cryptocurrencies-can-transform-the-digital-economy-by-christian-catalini/. [Accessed: 12-Jul-2017].

[22] C. Catalini, "How Blockchain Technology Will Impact the Digital Economy," *Oxford Law Faculty*, 24-Apr-2017. [Online]. Available: https://www.law.ox.ac.uk/business-law-blog/blog/2017/04/how-blockchain-technology-will-impact-digital-economy. [Accessed: 12-Jul-2017].

[23] B. McCabe, "The Pitfalls and Limitations of Blockchain," *Medium*, 02-Apr-2017. [Online]. Available: https://medium.com/@billsoftnet/the-pitfalls-and-limitations-of-blockchain-eba7424c0cf9. [Accessed: 17-Jul-2017].

[24] J. Compton, "How Blockchain Could Revolutionize The Internet Of Things," *Forbes*, 27-Jun-2017. [Online]. Available: http://www.forbes.com/sites/delltechnologies/2017/06/27/how-blockchain-could-revolutionize-the-internet-of-things/. [Accessed: 17-Jul-2017].

[25] F. Reese, "As Bitcoin Halving Approaches, 51% Attack Question Resurfaces," *CoinDesk*, 06-Jul-2016. [Online]. Available: http://www.coindesk.com/ahead-bitcoin-halving-51-attack-risks-reappear/. [Accessed: 17-Jul-2017].

[26] N. Popper, "A Hacking of More Than $50 Million Dashes Hopes in the World of Virtual Currency," *The New York Times*, 17-Jun-2016.

[27] D. Z. Morris, "Blockchain-based Venture Capital Fund Robbed of $60 Million in Apparent Hack," *Fortune*, 18-Jun-2016. [Online]. Available: http://fortune.com/2016/06/18/blockchain-vc-fund-hacked/. [Accessed: 17-Jul-2017].

[28] "Bitcoin Price Index - Real-time Bitcoin Price Charts," *CoinDesk*. [Online]. Available: http://www.coindesk.com/price/. [Accessed: 17-Jul-2017].
[29] "What is Ethereum?," *Ethereum Homestead Documentation*. [Online]. Available: http://ethdocs.org/en/latest/introduction/what-is-ethereum.html. [Accessed: 17-Jul-2017].
[30] N. Popper, "Business Giants to Announce Creation of a Computing System Based on Ethereum," *The New York Times*, 27-Feb-2017.
[31] N. Popper, "Move Over, Bitcoin. Ether Is the Digital Currency of the Moment.," *The New York Times*, 19-Jun-2017.
[32] N. Szabo, "Smart Contracts: Building Blocks for Digital Markets," 1996. [Online]. Available: http://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart_contracts_2.html. [Accessed: 17-Jul-2017].
[33] Smart Contracts Alliance, "Smart Contracts: 12 Use Cases for Business & Beyond," Chamber of Digital Commerce, White Paper, Dec. 2016.
[34] A. Hertig, "How Do Ethereum Smart Contracts Work?," *CoinDesk*, 30-Mar-2017. [Online]. Available: http://www.coindesk.com/information/ethereum-smart-contracts-work/.
[35] R. Hackett, "Walmart and IBM Are Partnering to Put Chinese Pork on a Blockchain," *Fortune*, 19-Oct-2016. [Online]. Available: http://fortune.com/2016/10/19/walmart-ibm-blockchain-china-pork/. [Accessed: 18-Jul-2017].
[36] N. Popper and S. Lohr, "Blockchain: A Better Way to Track Pork Chops, Bonds, Bad Peanut Butter?," *The New York Times*, 04-Mar-2017.
[37] P. Rizzo, "World's Largest Mining Company to Use Blockchain for Supply Chain," *CoinDesk*, 23-Sep-2016. [Online]. Available: http://www.coindesk.com/bhp-billiton-blockchain-mining-company-supply-chain/. [Accessed: 18-Jul-2017].
[38] "A Digital Global Ledger," *Everledger*. [Online]. Available: https://www.everledger.io/. [Accessed: 18-Jul-2017].
[39] S. Green, "Smart Contracts," *Oxford Law Faculty*, 07-Mar-2017. [Online]. Available: https://www.law.ox.ac.uk/business-law-blog/blog/2017/03/smart-contracts. [Accessed: 18-Jul-2017].
[40] W. Zhao, "$7 Million Lost in CoinDash ICO Hack," *CoinDesk*, 17-Jul-2017. [Online]. Available: http://www.coindesk.com/7-million-ico-hack-results-coindash-refund-offer/. [Accessed: 18-Jul-2017].
[41] "Smart Contracts in Financial Services: Getting from Hype to Reality," *Capgemini Consulting*, 11-Oct-2016. [Online]. Available: https://www.capgemini-consulting.com/blockchain-smart-contracts. [Accessed: 18-Jul-2017].
[42] D. M. Lambert and M. C. Cooper, "Issues in Supply Chain Management," *Industrial Marketing Management*, vol. 29, no. 1, pp. 65–83, Jan. 2000.
[43] E. M. Goldratt and J. Cox, *The Goal: A Process of Ongoing Improvement*, 30th Anniversary Edition. Great Barrington, Mass: North River Press, 2014.
[44] C. Hidjaja, "Top Supply Chain Management Challenges in 2017 & How to Overcome Them," 20-Apr-2017. [Online]. Available: https://blog.vision33.com/top-supply-chain-management-challenges-in-2017-and-how-to-overcome-them. [Accessed: 19-Jul-2017].
[45] S. Berfield, "Inside Chipotle's Contamination Crisis," *Bloomberg*, 22-Dec-2015.

[46] S. Swartz, "Challenges for Today's Global Supply Chain: Cost, Profitability and Personalization," *Inbound Logistics*, 06-Oct-2014. [Online]. Available: http://www.inboundlogistics.com/cms/article/challenges-for-todays-global-supply-chain-cost-profitability-and-personalization/. [Accessed: 20-Jul-2017].
[47] B. Dickson, "Blockchain has the potential to revolutionize the supply chain," *TechCrunch*, 24-Nov-2016. [Online]. Available: http://social.techcrunch.com/2016/11/24/blockchain-has-the-potential-to-revolutionize-the-supply-chain/. [Accessed: 20-Jul-2017].
[48] M. J. Casey and P. Wong, "Global Supply Chains Are About to Get Better, Thanks to Blockchain," *Harvard Business Review*, 13-Mar-2017. [Online]. Available: https://hbr.org/2017/03/global-supply-chains-are-about-to-get-better-thanks-to-blockchain. [Accessed: 20-Jul-2017].
[49] R. Khaitan, "Blockchain: One Truth Across Networks for Supply Chain," *Watson Customer Engagement*, 11-Apr-2017. [Online]. Available: https://www.ibm.com/blogs/watson-customer-engagement/2017/04/11/blockchain-supply-chain/. [Accessed: 20-Jul-2017].
[50] S. Rollings, "Addressing Today's Continuity of Supply Challenge," *Supply & Demand Chain Executive*, 04-Jan-2016. [Online]. Available: http://www.sdcexec.com/article/12154230/addressing-todays-continuity-of-supply-challenge. [Accessed: 20-Jul-2017].
[51] P. Satyavolu and A. Sangamnerkar, "Blockchain's Smart Contracts: Driving the Next Wave of Innovation Across Manufacturing Value Chains," Cognizant, White Paper, Jun. 2016.
[52] "Solidity," *Solidity 0.4.14 Documentation*. [Online]. Available: https://solidity.readthedocs.io/en/develop/#. [Accessed: 28-Jul-2017].
[53] *browser-solidity: Browser-Only Solidity IDE and Runtime Environment*. Ethereum, 2017.
[54] *testrpc: Fast Ethereum RPC client for testing and development*. EthereumJS, 2017.
[55] *truffle: The most popular Ethereum development framework*. Truffle Suite, 2017.
[56] "Blockchain Oracles," *BlockchainHub*. [Online]. Available: https://blockchainhub.net/blockchain-oracles/. [Accessed: 31-Jul-2017].
[57] E. Larchevêque, "Hardware Oracles: bridging the Real World to the Blockchain," *Ledger*, 31-Aug-2016. [Online]. Available: https://blog.ledger.co/hardware-oracles-bridging-the-real-world-to-the-blockchain-ca97c2fc3e6c. [Accessed: 31-Jul-2017].
[58] N. Lund, "How blockchain can transform the consumer goods supply chain," *Microsoft Enterprise*, 08-May-2017. [Online]. Available: https://enterprise.microsoft.com/en-us/articles/industries/retail-and-consumer-goods/blockchain-can-transform-consumer-goods-supply-chain/. [Accessed: 31-Jul-2017].
[59] K. S. Nash, "IBM Pushes Blockchain into the Supply Chain," *Wall Street Journal*, 14-Jul-2016.
[60] "From shore to plate: Tracking tuna on the blockchain," *Provenance*, 15-Jul-2016. [Online]. Available: https://www.provenance.org/tracking-tuna-on-the-blockchain. [Accessed: 31-Jul-2017].
[61] J. Donaldson, "Mojix Brings Transformational RFID, Big Data Analytics and Blockchain Technology to NRF Retail's Big Show 2017," *Mojix, Inc.*, 12-Jan-2017. [Online]. Available:

https://www.mojix.com/mojix-rfid-big-data-analytics-blockchain-technology/. [Accessed: 31-Jul-2017].
[62] "Data Integrity for IoT Devices Powered by Blockchain," *Modum.io*. [Online]. Available: https://modum.io/. [Accessed: 31-Jul-2017].
[63] J. McKendrick, "Why Blockchain May Be Your Next Supply Chain," *Forbes*, 21-Apr-2017. [Online]. Available: https://www.forbes.com/sites/joemckendrick/2017/04/21/why-blockchain-may-be-your-next-supply-chain/. [Accessed: 01-Aug-2017].
[64] P. Smith, "Blockchain technologies entered the trough of disillusionment in 2016, but 2017 will be brighter," *TechCrunch*, 30-Dec-2016. [Online]. Available: http://social.techcrunch.com/2016/12/30/blockchain-technologies-entered-the-trough-of-disillusionment-in-2016-but-2017-will-be-brighter/. [Accessed: 01-Aug-2017].
[65] "Gartner's 2016 Hype Cycle for Emerging Technologies Identifies Three Key Trends That Organizations Must Track to Gain Competitive Advantage," *Gartner, Inc.*, 16-Aug-2016. [Online]. Available: http://www.gartner.com/newsroom/id/3412017. [Accessed: 01-Aug-2017].

Appendix A: Provenance Smart Contract Code

```
// contract to allow supply chain parties and consumers to check the
provenance of goods
contract Provenance {

    address admin;
    mapping (address => Producer) producers;
    mapping (string => Product) products;

    struct Producer {
        string name;
        uint phoneNo;
        string cityState;
        string country;
        bool certified;
    }

    struct Product {
        address producer;
        uint[] locationData; // array containing lat & long
        uint timeStamp;
    }

    // constructor - runs once when contract is deployed
    function Provenance() {
        admin = msg.sender;
    }

    // modifier to allow only admin to execute a function
    modifier onlyAdmin() {
        if (msg.sender != admin) throw;
        _;
    }

    // function for producer to add their details to database
    function addProducer(string _name, uint _phoneNo, string
_cityState, string _country) returns (bool success) {

        // don't overwrite existing entries and ensure name isn't null
        if (bytes(producers[msg.sender].name).length == 0 &&
bytes(_name).length != 0) {
            producers[msg.sender].name = _name;
            producers[msg.sender].phoneNo = _phoneNo;
            producers[msg.sender].cityState = _cityState;
            producers[msg.sender].country = _country;
            producers[msg.sender].certified = false;
```

```
            return true;
        }
        else {
            return false; // either entry already exists or name
entered was null
        }
    }

    // function to remove producer from database (can only be done by
admin)
    function removeProducer(address _producer) onlyAdmin returns (bool
success) {
        delete producers[_producer];
        return true;
    }

    // function to display details of producer
    function findProducer(address _producer) constant returns (string,
uint, string, string, bool) {
        return (producers[_producer].name,
producers[_producer].phoneNo, producers[_producer].cityState,
producers[_producer].country, producers[_producer].certified);
    }

    // function to certify producer as legitimate (can only be done by
admin)
    function certifyProducer(address _producer) onlyAdmin returns
(bool success) {
        producers[_producer].certified = true;
        return true;
    }

    // function for producer to add their product to database
    function addProduct(string serialNo, uint[] _locationData) returns
(bool success) {

        // ensure no duplicate serial numbers and serial number isn't
null
        if (products[serialNo].producer == 0x0 &&
bytes(serialNo).length != 0) {
            products[serialNo].producer = msg.sender;
            products[serialNo].locationData = _locationData;
            products[serialNo].timeStamp = block.timestamp;
            return true;
        }
        else {
            return false; // either serial number already in use or
serial number entered was null
```

```
        }
    }

    // function to remove product from database (can only be done by
admin)
    function removeProduct(string serialNo) onlyAdmin returns (bool
success) {
        delete products[serialNo];
        return true;
    }

    // function to display details of product
    function findProduct(string serialNo) constant returns (address,
uint[], uint) {
        return (products[serialNo].producer,
products[serialNo].locationData, products[serialNo].timeStamp);
    }
}
```

Appendix B: Tracking Smart Contract Code

```
// contract to allow supply chain parties to track shipment of goods
and automatically execute payment in tokens
contract Tracking {

    address admin;
    uint[] contractLocation; // array containing lat & long
    uint contractLeadTime; // in seconds
    uint contractPayment; // in tokens
    mapping (string => Shipment) shipments;
    mapping (address => uint) balances;
    mapping (address => uint) totalShipped; // total number of
shipments made
    mapping (address => uint) successShipped; // number of shipments
successfully completed

    struct Shipment {
        string item;
        uint quantity;
        uint[] locationData;
        uint timeStamp;
        address sender;
    }

    // events to display messages when certain transactions are
executed
    event Success(string _message, string trackingNo, uint[]
_locationData, uint _timeStamp, address _sender);
    event Payment(string _message, address _from, address _to, uint
_amount);
    event Failure(string _message);

    // constructor - runs once when contract is deployed
    // determine initial token supply upon contract deployment
    function Tracking(uint _initialTokenSupply) {
        admin = msg.sender;
        balances[admin] = _initialTokenSupply; // all tokens held by
admin initially
    }

    // modifier to allow only admin to execute a function
    modifier onlyAdmin() {
        if (msg.sender != admin) throw;
        _;
    }
```

```
    // function to send tokens from one account to another
    function sendToken(address _from, address _to, uint _amount)
returns (bool success) {
        if (balances[_from] < _amount) {
            Failure('Insufficient funds to send payment');
            return false;
        }
        balances[_from] -= _amount;
        balances[_to] += _amount;
        Payment('Payment sent', _from, _to, _amount);
        return true;
    }

    // function to show token balance of an account
    function getBalance(address _account) constant returns (uint
_balance) {
        return balances[_account];
    }

    // function to recover tokens from an account (can only be done by
admin)
    // in the event that the sendToken function gets abused
    function recoverToken(address _from, uint _amount) onlyAdmin
returns (bool success) {
        if (balances[_from] < _amount) {
            Failure('Insufficient funds for recovery');
            return false;
        }
        balances[_from] -= _amount;
        balances[msg.sender] += _amount;
        Payment('Funds recovered', _from, msg.sender, _amount);
        return true;
    }

    // function to set contract parameters for next leg of shipment
(can only be done by admin)
    function setContractParameters(uint[] _location, uint _leadTime,
uint _payment) onlyAdmin returns (bool success) {
        contractLocation = _location; // set next location that will
receive shipment
        contractLeadTime = _leadTime; // set acceptable lead time for
next leg of shipment
        contractPayment = _payment; // set payment amount for
completing next leg of shipment
        return true;
    }

    // function for party to input details of shipment that was sent
```

```
    function sendShipment(string trackingNo, string _item, uint
_quantity, uint[] _locationData) returns (bool success) {
        shipments[trackingNo].item = _item;
        shipments[trackingNo].quantity = _quantity;
        shipments[trackingNo].locationData = _locationData;
        shipments[trackingNo].timeStamp = block.timestamp;
        shipments[trackingNo].sender = msg.sender;
        totalShipped[msg.sender] += 1;
        Success('Item shipped', trackingNo, _locationData,
block.timestamp, msg.sender);
        return true;
    }

    // function for party to input details of shipment that was
received
    function receiveShipment(string trackingNo, string _item, uint
_quantity, uint[] _locationData) returns (bool success) {

        // check that item and quantity received match item and
quantity shipped
        if (sha3(shipments[trackingNo].item) == sha3(_item) &&
shipments[trackingNo].quantity == _quantity) {
            successShipped[shipments[trackingNo].sender] += 1;
            Success('Item received', trackingNo, _locationData,
block.timestamp, msg.sender);

            // execute payment if item received on time and location
correct
            if (block.timestamp <= shipments[trackingNo].timeStamp +
contractLeadTime && _locationData[0] == contractLocation[0] &&
_locationData[1] == contractLocation[1]) {
                sendToken(admin, shipments[trackingNo].sender,
contractPayment);
            }
            else {
                Failure('Payment not triggered as criteria not met');
            }
            return true;
        }
        else {
            Failure('Error in item/quantity');
            return false;
        }
    }

    // function to remove details of shipment from database (can only
be done by admin)
```

```
    function deleteShipment(string trackingNo) onlyAdmin returns (bool
success) {
        delete shipments[trackingNo];
        return true;
    }

    // function to display details of shipment
    function checkShipment(string trackingNo) constant returns
(string, uint, uint[], uint, address) {
        return (shipments[trackingNo].item,
shipments[trackingNo].quantity, shipments[trackingNo].locationData,
shipments[trackingNo].timeStamp, shipments[trackingNo].sender);
    }

    // function to display number of successfully completed shipments
and total shipments for a party
    function checkSuccess(address _sender) constant returns (uint,
uint) {
        return (successShipped[_sender], totalShipped[_sender]);
    }

    // function to calculate reputation score of a party (percentage
of successfully completed shipments)
    function calculateReputation(address _sender) constant returns
(uint) {
        if (totalShipped[_sender] != 0) {
            return (100 * successShipped[_sender] /
totalShipped[_sender]);
        }
        else {
            return 0;
        }
    }
}
```

Appendix C: Reputation Smart Contract Code

```
// contract to store database of supply chain parties and their
reputations
contract Reputation {

    // call the Tracking contract at its deployed address
    // need to include Tracking contract code at the end
    Tracking track =
Tracking(0x0dcd2f752394c41875e259e00bb44fd505297caf);

    address admin;
    mapping (address => Supplier) suppliers;
    address[] suppliersByAddress; // array of all suppliers' accounts

    struct Supplier {
        string name;
        uint phoneNo;
        string cityState;
        string country;
        string goodsType;
        uint reputation;
    }

    // constructor - runs once when contract is deployed
    function Reputation() {
        admin = msg.sender;
    }

    // modifier to allow only admin to execute a function
    modifier onlyAdmin() {
        if (msg.sender != admin) throw;
        _;
    }

    // function for supplier to add their details to database
    function addSupplier(string _name, uint _phoneNo, string
_cityState, string _country, string _goodsType) returns (bool success)
{

        // don't overwrite existing entries and ensure name isn't null
        if (bytes(suppliers[msg.sender].name).length == 0 &&
bytes(_name).length != 0) {
            suppliers[msg.sender].name = _name;
            suppliers[msg.sender].phoneNo = _phoneNo;
            suppliers[msg.sender].cityState = _cityState;
            suppliers[msg.sender].country = _country;
```

```
            suppliers[msg.sender].goodsType = _goodsType;
            suppliers[msg.sender].reputation =
track.calculateReputation(msg.sender);
            suppliersByAddress.push(msg.sender);
            return true;
        }
        else {
            return false; // either entry already exists or name
entered was null
        }
    }

    // function to remove supplier from database (can only be done by
admin)
    function removeSupplier(address _supplier) onlyAdmin returns (bool
success) {
        delete suppliers[_supplier];

        // delete entry from array and shorten array
        for (uint i = 0; i < suppliersByAddress.length; i++) {
            if (suppliersByAddress[i] == _supplier) {
                for (uint index = i; index < suppliersByAddress.length
- 1; index++) {
                    suppliersByAddress[index] =
suppliersByAddress[index + 1];
                }
                delete suppliersByAddress[suppliersByAddress.length -
1];
                suppliersByAddress.length--;
            }
        }
        return true;
    }

    // function to display details of supplier
    function findSupplier(address _supplier) constant returns (string,
uint, string, string, string, uint) {
        return (suppliers[_supplier].name,
suppliers[_supplier].phoneNo, suppliers[_supplier].cityState,
suppliers[_supplier].country, suppliers[_supplier].goodsType,
suppliers[_supplier].reputation);
    }

    // function to display all suppliers' accounts in database
    function allSuppliers() constant returns (address[]) {
        return suppliersByAddress;
    }
```

```
    // function to search for suppliers by type of goods
    function filterByGoodsType(string _goodsType) constant returns
(address[]) {
        address[] memory filteredGoods = new
address[](suppliersByAddress.length);
        for (uint i = 0; i < suppliersByAddress.length; i++) {
            if (sha3(suppliers[suppliersByAddress[i]].goodsType) ==
sha3(_goodsType)) {
                filteredGoods[i] = suppliersByAddress[i];
            }
        }
        return filteredGoods;
    }

    // function to search for suppliers by reputation (returns those
with same or higher reputation)
    function filterByReputation(uint _reputation) constant returns
(address[]) {
        address[] memory filteredRep = new
address[](suppliersByAddress.length);
        for (uint i = 0; i < suppliersByAddress.length; i++) {
            if (suppliers[suppliersByAddress[i]].reputation >=
_reputation) {
                filteredRep[i] = suppliersByAddress[i];
            }
        }
        return filteredRep;
    }

    // function to display reputation of supplier (calls Tracking
contract)
    function checkReputation(address _supplier) constant returns
(uint) {
        return track.calculateReputation(_supplier);
    }

    // function to update reputations of all suppliers (calls Tracking
contract)
    // can only be done by admin
    function updateReputations() onlyAdmin returns (bool success) {
        for (uint i = 0; i < suppliersByAddress.length; i++) {
            suppliers[suppliersByAddress[i]].reputation =
track.calculateReputation(suppliersByAddress[i]);
        }
        return true;
    }
}
```