Blockchain in Dairy Supply Chains: A Literature Review

MAKSYM KOVAL, University of Twente, The Netherlands

A blockchain system is a P2P distributed network. Initially applied in the financial domain, the qualities of the distributed ledger increase trust and transparency among all of the nodes, decrease the possibility of a single point of failure and, importantly, alleviate the responsibility of accounting from third parties. Even though the most famous application of blockchain is of financial nature (Bitcoin), the aforementioned qualities can be beneficial to other domains. With the development of globalization, food safety is a cause of concern for the producers and the customers. Standardized sharing of quality assurance information (QAI) with transparency and data immutability of blockchain could increase the trust of customers by enhancing the traceability of products. In addition to food safety, the impact of blockchain adoption can also carry economic benefits and logistics optimization for SC actors. This research presents a state-of-the-art literature review of blockchain technology (BT) in dairy supply chains (DSC) and generic supply chains (SC), elicits a list of technical system development recommendations and showcases the considerations software engineers will have to face while developing such systems with the Hyperledger Sawtooth framework.

Additional Key Words and Phrases: blockchain, supply-chain, dairy, sawtooth, ethereum

1 INTRODUCTION

Blockchain Technology (BT) is assimilated with qualities such as trust and transparency. Blockchains transparency comes from the fact that it is a distributed system (relative to its centralised alternatives [20]) where each node in the network has the same overview of complete stored continuously synchronised data and its trust through its data immutability [21][16]. Whenever a new block (i.e. new data) enters the blockchain, it is chained to the previous blocks in the chain by referencing their header hash. In combination with distributed consensus algorithms, this makes it almost practically difficult for someone to compromise block data integrity, thus elevating trust among the network nodes [27].

The most famous application of BT is Bitcoin, introduced in Satoshi Nakamoto's whitepaper [16]. Even though Bitcoin exists in the financial domain, the qualities of trust and transparency that it offers can be applied to the dairy supply chain (DSC) domain [22].

With the development of globalization, it is now more common to find people buying food products in their offseason. Thus food safety is becoming more and more of a pressing concern for supply chain actors to protect the welfare of consumers and maintain their trust [6]. A typical scandalous example of where such trust has been compromised is the 2008 Chinese milk melamine contamination [26].

In addition to food safety, the literature points out that it could optimize logistics performance and provide economic benefit [14].

That said, blockchain possesses significant drawbacks that prevent it from being readily adopted into DSCs [25][6]. Supply chain actors must share quality assurance information to facilitate food safety, yet they lack knowledge on how the shared data can be formatted to enable sufficient standardization and traceability without compromising confidentiality, such that there is no loss in competitive advantage [6].

When compared to its centralized alternatives, BT does not provide an apparent economic benefit [6] or, in some cases, not to all actors that are part of the chain [14]. In addition, since the technology is relatively new, there is a lack of experts in the industry and a limit on interoperability with the existing systems to enable smooth integration [10].

In light of such complexities, software engineers have to make design decisions that are in line with the business need and the aforementioned context. Past research presents a variety of business requirements [25] that should be satisfied while developing such a system and analyses existing implementations [4]. However, there appears to be a lack of literature that indicates the reasoning behind the choices made in software design.

This research conducted a literature review to investigate the business case for supply chains, including the societal benefits, and analysed existing implementations of BT systems (Section 3). The elicited result is a list of software engineering recommendations (Section 3.4). Additionally an example system is considered to demonstrate the design issues Hyperlerdger Sawtooth engineer will face during development (Section 4).

It answers the following research questions:

(1) How can BT enhance DSCs?
   (a) How can BT produce economic bgenefit in DSCs?
   (b) How can BT enhance food safety in DSCs?
   (c) How can BT enhance communication and logistics processes in DSCs?
(2) What challenges prevent the adoption of BT in DSCs?
(3) How does the supply chain context impact the software design considerations of a BT system?

2 THEORETICAL BACKGROUND

This section provides a background knowledge on the two development frameworks used to implement systems presented in later sections (Ethereum [24] and Hyperledger Sawtooth [19]). Additionally, a description and definitions of terms of the assumed supply chain model is provided.

2.1 Ethereum

Ethereum is a distributed public network, with Ether being the cryptocurrency of the network. Following Bitcoin, it uses a proof-of-work-based consensus algorithm. In addition to being a cryptocurrency platform, its contrived nature is attributed to its smart contract functionality. Ethereum has two types of accounts: External and internal (smart contracts). A user of an Ethereum network with an external

TScIT 37, July 8, 2022, Enschede, The Netherlands

account is capable of making regular transactions (transfers of Ether to other accounts), but a smart contract is not. Just like an internal account, a smart contract has an address and is capable of receiving transactions; additionally, when a smart contract is created (or deployed), it is supplied with code by the developer and is distributed among the nodes of the network. Upon a smart contract receiving a transaction, it may trigger its related code depending on the behaviour specified by the developer. As smart contracts may have features that rely on data storage (e.g. data upload), each smart contract maintains its own Merkle Patricia trie, which is an optimised Merkle Radix trie that allows efficient data addressing and root hash computation for later verification [17][24]. A smart contract maintains its balance and is taxed for each machine-level instruction that is executed due to its related code execution and its deployment. A part of this tax becomes a bounty for the miner who had run the related machine code.

2.2 Hyperledger Sawtooth

The Hyperledger Sawtooth project is part of the family of projects from the Hyperledger Foundation maintained by the Linux Foundation. Hyperledger focuses primarily on developing enterprise blockchain-related tools and frameworks [1][19]. Thus, Hyperledger Sawtooth is a framework for building a distributed blockchain system. The key speciality of Sawtooth (among other Hyperledger projects) is that its design emphasises the discretion between the application layer and core network layer such that if one were to write an application for Sawtooth, they need not be concerned with blockchain and networking-related aspects. Additionally, Sawtooth ships with a permission feature that allow placing restrictions on transactors and validator nodes and a dynamic consensus feature that allows switching consensus algorithms during network operation. Currently, Sawtooth supports three non-development consensus algorithms: Sawtooth PBFT, Sawtooth PoET and Sawtooth Raft. A node participating in a Sawtooth network typically runs the following processes: A REST API, a Validator, a Consensus Engine and a set of transaction processors . The REST API receives requests from the network and forwards them to the Validator. The Validator performs validation on the request formatting and delivers its payloads to the appropriate transaction processors. Finally, the transaction processor executes the request and, if required, updates the associated Merkle Radix tree, which in turn will cause an update to the underlying blockchain [23]. Transaction families (TFs) in Sawtooth are sometimes compared to Ethereum smart contracts since they provide the definition of application behaviour. Yet, this metaphor is imperfect as they do share a number of differences and are not "deployed" in the same as smart contracts in Ethereum. As defined by Sawtooth, TFs are made of three components the data, transaction processor and client. Arguably the transaction processor is the most relevant as it processes the received transactions from the Validator and updates the state (Merkle Radix trie).

2.3 Supply chain model

For this paper, a simplified model presented by Aung and Cheng [3] is assumed. The model considers two end-points, the producer and the consumer, and defines an actor as an entity that represents a step in the diagram (e.g. Retailer). Note that an actor may not need to be an organisation but can also be considered a single individual. Internal traceability is thus defined as the collection of processes that enable traceability of goods (e.g. collection of quality assurance information) within the immediate vicinity of the actors' area. On the other hand, external traceability is only concerned with processes that occur intermediately (e.g. while the good is being transported from one to actor to another along the chain)

3 LITERATURE REVIEW

This literature review overviews the context of blockchains in SCs by considering qualitative SC business research, existing implementations and system architectures to form a set of holistic technical design recommendations for a BT system in an SC context.

3.1 Identified benefits, challenges and drawbacks

Related to Blockchain-based system implemented in SC are the root qualities brought to the business will be the immutability and traceability, which in combination enhance the accountability and responsibility of actors in an SC [11][25]. The benefits that arise through these qualities are complex in nature; the most notable of which are:

- Increase in profit margins and logistic operations optimisation for some SC actors [14].
- Social benefits. In the case of DSCs, they are local embedding, rural development, decrease in food fraud, animal health and welfare, proximity to food markets, food safety, educating and promoting healthy eating, assisting food access and social acceptability and transparency [15].

Note that even those qualities can be elevated through the usage of a non-BT decentralised system or a centralised system. These systems are typically not capable of preventing fraud and counterfeiting of information to the same extent as a BT system [11].

Even though the benefits are ample, there appear to be a significant number of challenges that oppose them, which primarily reside in the business side adoption and management of the deployed technology [10].

Firstly, even though it is commonly mentioned in the literature that trust is one of the benefits of BT system adoption, it is essential to clarify that even though such a system may enhance or preserve trust in an SC on the basis of an existing trust, the literature suggests that it is unlikely that BT adoption is capable of creating trust in an environment in which it is scarce [25][14].

Secondly, the management of a BT system requires consensus and ongoing cooperation among all of the key stakeholder representatives that are part of the chain which will include requirements engineering, the establishment of necessary business rules and policy regulations [11][25]. Otherwise, it will collapse [25].

3.2 Related Work

The analysed implementations are documented in Table 2 and Table 3. Table 2 provides surface-level information about the projects and how the customer or person of interest can access the traceability information stored in the blockchain. Where as Table 3 gives insight into how and what data is stored in the blockchain.

Wang et al. [25] and Behnke et al. [6] suggest that if the system introduces a significant shift in competitive advantage, then it is most likely the case that the project will collapse and be of no use. Thus it becomes essential to consider how can one vary the competitive advantage and traceability that is brought about by a blockchain system. A trend in literature is to suggest that richness of data (which can be contributed to by IoT devices) and access or how the system warrants confidentiality of information to actors in an SC - apart from explicitly excluding data from the system - provide dials for traceability, accountability and competitive benefit for actors (note that the benefit brought about by such a system is typically not equally distributed among all actors) [4][6][25]. Table 3 gives insight into how these aspects were regulated or, in other words, "how the dials have been set".

Azzi et al.[4] analysed two swiss startup case studies: **Modum** and **Ambrosus**.

Modum [7] developed a blockchain system to provide a solution for pharmaceutical transportation temperature verifiability by using Ethereum smart contracts that would store the valid temperature ranges and verify whether the temperature of transported pharmaceuticals aligned with the provided ranges. Usage of blockchain, in this case, creates transparency and trust through data immutability (i.e. temperature ranges) for authorities that would want to verify whether the temperature of transported pharmaceuticals had indeed been appropriate.

At the start of the shipment, the Modum mobile client would pair via Bluetooth to temperature sensors and associate the sensors with a particular medical package that is to be transported. At this point, the client would send a request to a centralised HTTP server which would deploy the appropriately configured smart contracts with the temperature ranges set by the client. As the shipment carries on, the sensors continuously record the temperature. At the end of the shipment, the track and trace code would be scanned; the recorded temperatures would be uploaded to the server (through the mobile client) and saved in the database. At the same time, the server would execute the earlier established smart contract that would contact the database in which the temperature is stored, verify temperature compliance and communicate the report back to the client.

Note: Smart contract configuration, deployment, execution, and temperature recording are communicated through the same server. Additionally, a challenge was mentioned that after an Ethereum fork, the smart contracts had to be re-written (i.e. the Ethereum fork was backwards incompatible). **Ambrosus** [12][4] uses IoT, BT and real-time sensors/tracking components to monitor manufacturing processes. Unlike Modum, Ambrosus is a generic solution for SC traceability by making the sensors "pluggable". The distinguishing aspect of Ambrosus is that it combines Ethereum and InterPlanetary File System (IPFS) to compensate for the data explosion caused by the sensors, where the particular set of sensors would directly communicate with the underlying blockchain instead of relying on a centralised server and client application. Yet, similarly to Modum, the Ambrosus network makes use of two smart contracts, one for tracking the requirements of goods and another one for recording the actual measurements.

Baralla G et al. [5] developed a Hyperledger Sawtooth system for a generic European food supply chain. The study takes more of a software engineering perspective and outlines the potential system architecture, user requirements and justification behind the design choices. As Hyperledger sawtooth is a fully custom distributed system, there are no performance issues as in Ethereum systems; thus, all data that is entered into the system can be maintained in the blockchain. That said, no tests concerning the scalability of data had been carried out to verify this (in the case sensors are added to the system that continuously produces real-time information). Additionally, as the system is based on Hyperledger Sawtooth, the network is permissioned by default [19] which in this case was configured such that anyone could reach the non-confidential or public information through a set of authentication REST APIs.

Similarly, Bumblauskas et al. [9] also used Hyperledger Sawtooth to develop the first known BT system in an egg SC deployed to a major market. The system tracks eggs from the farm to the consumer and records major intermediary information in between. Unlike the previous system, it uses IoT sensors which track and submit temperature recordings to the blockchain.

Longo et al. [14] analysed the economic impact of BT in a generic SC. The model SC is assumed to consist of wholesalers and big-box retailers. A simulation was developed that models the behaviour between the two actors by interacting with the underlying blockchain system (Ethereum). Here a common strategy that is observed in [13][12], where instead of uploading the complete data to a blockchain, only a hash sum of it is uploaded. Thus whoever gets access to the full data - which would commonly be stored in an off-chain database - should be able to recalculate the hash sum and verify its data integrity.

Niya et al. [18] developed NUTRIA, a Swiss-based dairy traceability system, following the" Foodchain" case study, which aimed at increasing the economic value of products by alleviating transparency of moral values associated with them (e.g. trust, sustainability, quality). NUTRIA is a fully decentralised Ethereum DAPP (Distributed Application) which primarily acts as a storage of actor information with its own JavaScript client. Whenever an actor in the chain produces some form of operation on dairy produce (it could be the case that the actor uses a dairy product to contrive another dairy product, as in the case of processing the milk into cheese), they generate a QR code with information to be stored in the blockchain. A unique design choice is emphasised in the paper "QR code chaining": Whenever an intermediary SC actor receives a particular product to be processed, their interaction has to be preceded by scanning the product QR code, creating a link in the chain. Once a particular product has gone through the SC and is available to the customer, the customer can access the SC and view its arrival path and addition product information. Additionally, this research points out that exemplary real-time system behaviour is not achieved, as it takes from the 30s to 120s for a data update in the blockchain to be visible to all actors. That said, since NUTRIYA stores a relatively small amount of data about the product, there appears to be no need to introduce a dual-storage system as in other architectures.

Lin Q et al. [13] prototyped a BT-enabled supply chain system while specifically focusing on the trust transfer problem, data explosion, and information classification problems in the domain of food and agriculture. The prototype is Ethereum based. One particular element that stands out from this architecture, among others, is that

each actor would maintain their server with relevant product data and only submit the insensitive data to a public ledger by using developed filter software. How the system ensures that each actor can verify product information and its integrity is described by the following systems data flow:

(1) A manufacturer generates an Radio-frequency Identification (RFID) tag, attaches it to the product, and enters the insensitive data into the blockchain (including a hash sum of the sensitive information which is maintained on the server of that particular actor).
(2) Once another actor receives the product, he/she can use the RFID label and authenticate themselves using a smart contract causing the smart contract to reply with an IP address of the server on which the sensitive data of the manufacturer is maintained.
(3) The receiver contacts the manufacturer's server (which requires their confirmation) and re-computes the hash sum of the product data to verify information integrity.

3.3 Design considerations

The non-technical considerations and SC context in which a potential BT system will be used are discussed. These considerations bring about insight and narrow down the choice of system requirements that are outlined in the technical considerations section that follows.

3.3.1 Context and non-technical considerations.

Wang et al. [25] analysed the design considerations and requirements that a set of businesses in an SC collectively must undertake in an attempt to increase the chance of creating a BT system that thrives. The design considerations they have posed are as follows:

(1) Requirements of a BT system and its benefits have to be collectively analysed where the subject collective will consist of stakeholders representing each actor in the blockchain.
(2) Deployment and usage of a BT system should benefit every actor in the chain.
(3) The collective should contain a minimum number of key supply chain actors.
(4) A BT system orchestrator should be appointed whose responsibility will be to maintain the network and communicate its values throughout the collective. In the case study conducted authors in [25], it was also mentioned that the party that acted as the orchestrator was not a direct actor in the SC and had a neutral stance concerning competition and benefit, which could have been key as some actors in the SC may be unwilling to place this responsibility onto another SC actor.
(5) Off-chain and on-chain governance protocols should be established to regulate the blockchain solution.
(6) A degree of permissioning is required to protect the sensitive data of actors depending on the use case.
(7) What data should be stored on-chain requires careful consideration.
(8) The establishment of legal and regulatory documentation is critical to ensure shared understanding among stakeholders.

A design approach has also been mentioned by Behnke et al [6]., which suggests that an immature blockchain system should contain minimum data, and as collective decisions are made, the kind of data collected by the system should increase or change. An implication can be drawn from these observations that a vital function of a "to be blockchain system" is to have sufficient flexibility such that the kind of data stored can be changed relatively easily. Additionally, the study by Niya et al. [18] suggests that one of the reasons for the increase in customer trust towards the recorded traceability data is that the subject data was collected in an automated manner (e.g. the phone would automatically gather the location when the good arrived at a specific actor location). Therefore, a hypothesis can be drawn from this that introducing relevant IoT devices (i.e. sensors) into the system could increase customer trust.

3.3.2 Technical considerations.

The technical design considerations elicited from the aforementioned existing implementations (Section 3.2) are as follows:

Consideration of the Blockchain System:
There are generally two ways in which one can implement a blockchain system:

(1) Implementation of own custom blockchain client.
(2) Choosing and developing on an existing blockchain (e.g. Ethereum, Hyperledger).

With their own implementation, the developer has full flexibility. However, the knowledge and experience that are required to develop such a system are considerably higher than the second option. This may be unattractive for a business as this option is also likely to consume most resources [2]. Due to such reason, ledger design will not be discussed, but rather the focus will be on the second development direction (choosing and developing on an existing blockchain).

From the identified literature (Table 2 and Table 3), the most common blockchain to use is Ethereum, followed by Hyperledger Sawtooth. Ethereum is a more mature public ledger technology [5]. On the other hand, Hyperledger Sawtooth is relatively immature and is not a public ledger but rather a framework or tool that allows for the creation of custom permissioned blockchain systems[1]. The relevant characteristics of Ethereum and Hyperledger Sawtooth are presented in Table 1.

Consideration of Architecture:
Different systems adopted a different architecture depending on the underlying choice of the blockchain (Ethereum or Hyperledger Sawtooth). Firstly characterisations of Ethereum architecture are provided, followed by Hyperlerledger Sawtooth.

Most of the Ethereum aforementioned solutions adopted a dual storage architecture. In the context of this paper, a dual storage architecture is a blockchain system in which typically fully detailed data is stored off-chain in some form of a database or storage, and only a partial copy of that data is stored on the Ethereum ledger through a smart contract [4]. The identified reasons for such a design decision are as follows:

(1) Data explosion - If the system continuously sends data through smart contract execution (this is the case, especially in systems that make use of IoT devices [4][12][7]), the system may

[1] Hyperledger documentation - https://sawtooth.hyperledger.org/docs/1.2/

Ethereum[5]	Hyperledger Sawtooth[5]
- Mature (released in July 2015)	- Relatively immature (released in January 2018)
- Reliable non-custom consensus	- Customizable and "pluggable" consensus
- Uses the Ethereum network	- Allows for the creation of a custom distributed ledger
- Data immutability	- Data immutability
- Performance and cost are dependent on network load which cannot be centrally regulated.	- Performance and cost are highly dependent on individual development decisions and business decisions in the SC.

Table 1. Ethereum and Hyperledger Sawtooth characteristics

begin to stall and underperform due to the nature of Ethereums public ledger [13].

(2) Confidentiality - Actors in the SC would not want to upload complete information in clear text to the blockchain, as they deem that information confidential [6][25]. The solution to this then becomes to store full data in an off-chain storage system and only keep the hash sum of the stored off-chain data on-chain. This design ensures that any actor or customer in the chain can still verify data integrity if they get access to complete data [14]. As it is most likely the case that actors already have some form of database with information that they would want to store on-chain, creating an additional storage system may be considered unnecessary and, instead, a kind of filter software that synchronises the off-chain system with the blockchain is also a possible alternative [13].

The identified choices of architecture for Ethereum like blockchain systems are as follows:

(1) Dual distributed system - In a dual distributed system, the off-chain data is stored in another distributed system, such as IPFS (seen in [4][12]).
(2) Distributed system - The case in which complete data is stored on the blockchain (Seen in [18]).
(3) Distributed system with centralised storage - A dual distributed storage system in which the complete data is stored on a centralised server, yet the access to that data is regulated per actor in the supply chain (Seen in [7]).
(4) Distributed system with isolated storage - A dual distributed storage system in which every actor in the SC maintains their own server, which contains the full or complete data that that particular actor has produced. It is then implied that each actor's server would communicate with the blockchain (e.g. by filtering and uploading the relevant information to the blockchain) (Seen in [14][13]).

Hyperledger Sawtooth does not generally face the issues of data explosion and confidentiality since the network access is completely custom. That said, an architectural decision was identified: whether or not the key pairs used to submit requests to the hyper ledger network are stored on a *middle-man* server. A node of a Sawtooth network runs a set of Sawtooth processes. The node is not limited to running a validator[2] but typically also runs a Sawtooth REST API that would allow clients to submit data over it. That said, the data submitted to the REST API must already be signed by the sender's private key. Introducing a new server (e.g. REST API) in between the client and the Sawtooth REST API can prevent the clients from managing their own key set and instead let the *middle-man* handle it (seen in [9]). If the *middle-man* is introduced, then it can either have a shared keyset or a keyset per user. If a shared keyset is used, then each request submitted by any client will be signed by the same private key associated with the *middle-man*. An alternative approach involves maintaining a database of user keysets and associating them with user credentials such that whenever a user authorises a request to the blockchain, the *middle-man* will sign it on behalf of the user (seen in [9]). At times a hybrid of the two approaches is prudent as that would allow some users not to authenticate and still be able to send a certain kind of a request, while an authenticated user may be able to send requests of a different kind. Additionally, a *middle-man* with a database can make use of a Hyperledger Sawtooth event system that allows the blockchain data to be continuously uploaded to the database. Sawtooth developers noted that this leads to a decrease in query time as the *middle-man* need not to interact with the blockchain upon client request and can instead query the database. However, the database is only useful in the case of data being read from the blockchain and the *middle-man* still has to interact with it upon any data insertion operations.

[2]validator - a Hyperledger Sawtooth process that manages the data stored on the blockchain and performs transaction validation

3.4 Recommendations

After analysing system requirements from the presented literature, a list of recommendations for engineering a blockchain with a supply chain use case was formed and is as follows:

Overall, Hyperledger Sawtooth is a more suitable development environment for supply chains than Ethereum and thus should be chosen as the implementation framework, and that due to the below advantages:

(1) Greener customisable consensus - Hyperledger Sawtooth provides a dynamic consensus feature (i.e. a Sawtooth network can switch its consensus algorithm during operation) and supports less energy-consuming algorithms such as PoET [8].
(2) Permissioned at its core - Sawtooth ships with permissibility features that supply chain actors desire [5].
(3) Higher performance - Ethereum underperforms when compared to Sawtooth. This could have significance in the case it is desired for a system to have a continuous real-time data upload [5].

Software design should facilitate high adaptability of the data stored in the system.

Assuming that the software development cycles will take place in a consortium with the most relevant supply chain actor representatives [25]. It is of utmost importance for software engineers to design their programs in such a way that it is easy and fast to adapt the data stored in the system. The reason for that is not only that each actor would want to keep a different kind of data on the blockchain but also that it is likely that actors will not have a specific standardisation format for the data to be published [6]. Thus, the process of data standardisation will become part of the development cycle, and software design should be able to accommodate such entropy to reduce time and development costs.

Software design should support IoT and sensor integrations.

Depending on the goods being transported, the stakeholders may want to include sensors that continuously submit data to the blockchain. This is especially true in the case of food and pharmaceuticals, where the product may be temperature sensitive. Software design that can accommodate a variety of sensors and the data that they are recording may significantly reduce the development time and prevent codebase re-writes.

4 EXAMPLE IMPLEMENTATION CONSIDERATIONS

This section provides an example of key considerations that have to be made while developing a Sawtooth application. The example assumes a simple supply chain which enables actors to store localisation and sensor measurement data onto the blockchain as a good traverses through the chain[3].

4.1 Data

The state (Merkle Radix trie) of Sawtooth will store three kinds of objects: Actor(s), Product (s) and Shipment(s). Each object is coded in Google Protobufs, which is a technology encouraged by the Sawtooth community and enables efficient object serialisation and de-serialisation that is platform-independent. The use of this technology is particularly prudent when the client does not run the same kind of programming language (e.g. in the case the client is running JavaScript and the transaction processor is implemented in Python). Actor objects are restricted to id and organisation name fields. The latter of which is self-explanatory, yet the former is assumed to be the public key of the actor, which is used to sign the transactions submitted to the network. Products have an ID, a product name and a list of associated Shipment (s). Where each Shipment maintains a list of measurements (sent from the sensors), a location, time of shipment departure and the associated actor that initiated the shipment.

4.2 Addressing

Whenever a Transaction Processor attempts to save any encoded data onto the blockchain, it does so by firstly addressing the Merkle Radix, which acts as the database of current data that has been accepted by the network consensus (note that each Validator maintains its own version of the Merkle Radix tree that is synchronised throughout the network). The height of the Merkle Radix tree is 35, which corresponds to the addressing scheme of Sawtooth, where each address has a length of 35 bytes. Conventionally the first 3 bytes determine the application namespace, and the other 32 bytes are left for the developer to use appropriately. For our example system, each object was associated with a unique address. Each address of each type of object (Actor, Shipment, Product) has a prefix of the first 6 hex characters (3 bytes) of the hashed application name. Followed by its object type encoding of 2 hex characters (1 byte), followed by the first 62 hex characters (31 bytes) of the object identifier hash (e.g. the hash of the actor identifier would be the hash of the actor public key).

Note that this approach is simplistic and does not prevent object collision to a satisfactory level if this system were to be deployed to production, and a more sophisticated method of ID creation and management is required.

[3]implementation repository - https://github.com/1MaxKoval/supplyledger

5 CONCLUSION

This paper provides a comprehensive overview of blockchain in the context of supply chains and dairy supply chains. It looks at the benefits of blockchain systems, discusses and analyses a number of existing adoption challenges, implementations and architectures, and considers the business needs and requirements software engineers would have to form by while developing such systems. A result of this research is a list of recommendations that argues in favour of Hyperledger Sawtooth over Ethereum as the development framework and that higher emphasis should be placed on software design with regard to data adaptability and IoT support. Additionally, the paper highlighted the challenges Sawtooth developers face by providing an example case system.

REFERENCES

[1] Shubhani Aggarwal and Neeraj Kumar. 2021. Hyperledger. In *Advances in computers*. Vol. 121. Elsevier, 323–343.

[2] Esteban Angulo and Xavier Ferre. 2014. A case study on cross-platform development frameworks for mobile applications and UX. In *Proceedings of the XV International Conference on Human Computer Interaction*. 1–8.

[3] Myo Min Aung and Yoon Seok Chang. 2014. Traceability in a food supply chain: Safety and quality perspectives. , 172-184 pages. Issue 1. https://doi.org/10.1016/j.foodcont.2013.11.007

[4] Rita Azzi, Rima Kilany Chamoun, and Maria Sokhn. 2019. The power of a blockchain-based supply chain. *Computers and Industrial Engineering* 135 (9 2019), 582–592. https://doi.org/10.1016/j.cie.2019.06.042

[5] Gavina Baralla, Andrea Pinna, and Giacomo Corrias. 2019. Ensure traceability in european food supply chain by using a blockchain system. *Proceedings - 2019 IEEE/ACM 2nd International Workshop on Emerging Trends in Software Engineering for Blockchain, WETSEB 2019*, 40–47. https://doi.org/10.1109/WETSEB.2019.00012

[6] Kay Behnke and M. F.W.H.A. Janssen. 2020. Boundary conditions for traceability in food supply chains using blockchain technology. *International Journal of Information Management* 52 (6 2020). https://doi.org/10.1016/j.ijinfomgt.2019.05.025

[7] Thomas Bocek, Bruno B. Rodrigues, Tim Strasser, and Burkhard Stiller. 2017. Blockchains Everywhere - A Use-case of Blockchains in the Pharma Supply-Chain. (2017).

[8] M. Bowman, D. Das, A. Mandal, and H. Montgomery. 2021. *On Elapsed Time Consensus Protocols*. Vol. 13143 LNCS. 559–583 pages. https://doi.org/10.1007/978-3-030-92518-5_25

[9] Daniel Bumblauskas, Arti Mann, Brett Dugan, and Jacy Rittmer. 2020. A blockchain use case in food distribution: Do you know where your food has been? *International Journal of Information Management* 52 (6 2020). https://doi.org/10.1016/j.ijinfomgt.2019.09.004

[10] Pankaj Dutta, Tsan Ming Choi, Surabhi Somani, and Richa Butala. 2020. Blockchain technology in supply chain operations: Applications, challenges and research opportunities. *Transportation Research Part E: Logistics and Transportation Review* 142 (10 2020). https://doi.org/10.1016/j.tre.2020.102067

[11] Sohail Jabbar, Huw Lloyd, Mohammad Hammoudeh, Bamidele Adebisi, and Umar Raza. 2021. Blockchain-enabled supply chain: analysis, challenges, and future directions. *Multimedia Systems* 27, 787–806. Issue 4. https://doi.org/10.1007/s00530-020-00687-0

[12] M Kirejczyk, A Kędracki, I Rukhavets, and V Trifa. 2017. Ambrosus White Paper. (2017). https://www.allcryptowhitepapers.com/ambrosus-whitepaper/

[13] Qijun Lin, Huaizhen Wang, Xiaofu Pei, and Junyu Wang. 2019. Food Safety Traceability System Based on Blockchain and EPCIS. *IEEE Access* 7 (2019), 20698–20707. https://doi.org/10.1109/ACCESS.2019.2897792

[14] Francesco Longo, Letizia Nicoletti, Antonio Padovano, Gianfranco d'Atri, and Marco Forte. 2019. Blockchain-enabled supply chain: An experimental study. *Computers and Industrial Engineering* 136 (10 2019), 57–69. https://doi.org/10.1016/j.cie.2019.07.026

[15] Sachin Kumar Mangla, Yigit Kazancoglu, Esra Ekinci, Mengqi Liu, Melisa Özbiltekin, and Muruvvet Deniz Sezer. 2021. Using system dynamics to analyze the societal impacts of blockchain technology in milk supply chainsrefer. *Transportation Research Part E: Logistics and Transportation Review* 149 (5 2021). https://doi.org/10.1016/j.tre.2021.102289

[16] Satoshi Nakamoto. 2008. Bitcoin: A Peer-to-Peer Electronic Cash System. www.bitcoin.org

[17] Muhammad Saqib Niaz and Gunter Saake. 2015. Merkle hash tree based techniques for data integrity of outsourced data. *GvD* 1366 (2015), 66–71.

[18] Sina Rafati Niya, Danijel Dordevic, Markus Hurschler, Sarah Grossenbacher, and Burkhard Stiller. 2020. A Blockchain-based Supply Chain Tracing for the Swiss Dairy Use Case. *Proceedings - 2020 2nd International Conference on Societal Automation, SA 2020*. https://doi.org/10.1109/SA51175.2021.9507182

[19] Kelly Olson, Mic Bowman, James Mitchell, Shawn Amundson, Dan Middleton, and Cian Montgomery. 2018. Sawtooth: an introduction. *The Linux Foundation* (2018).

[20] Haroon Shakirat Oluwatosin. 2014. Client-server model. *IOSRJ Comput. Eng* 16, 1 (2014), 2278–8727.

[21] Max Raskin and David Yermack. 2018. Digital currencies, decentralized ledgers and the future of central banking. In *Research handbook on central banking*. Edward Elgar Publishing.

[22] Shuvam Shingh, V Kamalvanshi, Sarthak Ghimire, and Sudarshan Basyal. 2020. Dairy supply chain system based on blockchain technology. *Asian J. Econ. Bus. Account* 14 (2020), 13–19.

[23] SK Vivek, RS Yashank, Yashas Prashanth, N Yashas, and M Namratha. 2020. E-voting system using hyperledger sawtooth. In *2020 International Conference on Advances in Computing, Communication & Materials (ICACCM)*. IEEE, 29–35.

[24] Dejan Vujičić, Dijana Jagodić, and Siniša Ranđić. 2018. Blockchain technology, bitcoin, and Ethereum: A brief overview. In *2018 17th international symposium infoteh-jahorina (infoteh)*. IEEE, 1–6.

[25] Yingli Wang, Catherine Huirong Chen, and Ahmed Zghari-Sales. 2021. Designing a blockchain enabled supply chain. *International Journal of Production Research* 59 (2021), 1450–1475. Issue 5. https://doi.org/10.1080/00207543.2020.1824086

[26] Changbai Xiu and K. K. Klein. 2010. Melamine in milk products in China: Examining the factors that led to deliberate use of the contaminant. *Food Policy* 35 (10 2010), 463–470. Issue 5. https://doi.org/10.1016/j.foodpol.2010.05.001

[27] Zibin Zheng, Shaoan Xie, Hong-Ning Dai, Xiangping Chen, and Huaimin Wang. 2018. Blockchain challenges and opportunities: A survey. *International journal of web and grid services* 14, 4 (2018), 352–375.

A IMPLEMENTATION TABULATION

(Name) and Reference	Underlying Blockchain	Domain	System Objective	Traceability information accessibility
(Modum) Bocek et al. [7], Azzi et al. [4]	Ethereum	Pharmaceuticals	Demonstrate accordance with a Good Distribution Practice (GDP) regulation by providing evidence of pharmaceutical temperature during transportation	Authorities or persons of interest can access the data through a QR code referencing the smart contract and full temperature data.
(Ambrosus) Kirejczyk et al [12], Azzi et al. [4]	Ambrosus network (Ethereum + IPFS)	Pharmaceuticals and Food	Demonstrate traceability and immutability of quality assurance information.	QR code access through any client that uses the Ambrosus Javascript library.
Baralla et al. [5]	HyperLedger Sawtooth	Food	Traceability	QR code access through mobile client.
Bumblauskas et al. [9]	HyperLedger Sawtooth	Food (Eggs)	Minimisation of recall impact, food fraud, consumer ethical concerns through traceability.	Label scan of the egg packaging through a mobile client.
Longo et al. [14]	Ethereum (UnicalCoin)	Generic SC with whole-suppliers and big-box retailers	Evaluate the impact trust has on the economy of the SC.	No accessibility implemented per se, as the purpose of the study was to simulate a BT system to evaluate its economic impact.
(NUTRIA) Niya et al. [18]	Ethereum	Food (Dairy)	Enhance consumer trust through transparency and traceability.	QR code scanning through a mobile client.
Lin et al. [13]	Ethereum	Food	- Minimise maintenance costs and provide information confidentiality through the proposed architecture in a BT system. - Create varying levels of accessibility with the highest levels reserved for government regulators. - Demonstrate product traceability to the consumer.	Submission of an identification code of a good and address the smart contract. The consumer would then receive a URL(s) pointing to the full product information (from the server(s) of actors in the chain).

Table 2. *Prima facie* system information

(Name) and Reference	Data Storage Architecture	Confidentiality Warranty	Richness of Data	Usage of IOT
(Modum) Bocek et al. [7], Azzi et al. [4]	Dual storage.	Regulated full data access through URL. Confidential information on the blockchain is represented as a hash sum, while full-data is stored on a centralised server.	Temperature	Temperature sensors that continuously capture data and send it to the server. The sensors upload data to the server by making a Bluetooth connection to a system that has an Internet connection and has access to the Modum front-end (e.g. mobile phone).
(Ambrosus) Kirejczyk et al [12], Azzi et al. [4]	Dual storage. Combines IPFS with Ethereum to track immutability in IPFS nodes.	Authentication is implemented by maintaining a list of entities that can access the system (on the blockchain).	Any data that can be captured by sensors.	A bundle of sensors continuously captures relevant data which are connected to a centralised microcontroller that directly communicates with the underlying blockchain.
Baralla et al. [5]	Dual storage. The architecture uses a REST API for managing blockchain authentication, which maintains a set of credentials. All information relevant to traceability is stored on-chain.	Hyperledger Sawtooth is a permissioned network with public access by design.	Generic product information and relevant documentation.	Supported but not used.
Bumblauskas et al. [9]	Dual storage. The architecture uses a REST API for managing blockchain authentication, which maintains a set of credentials. All information relevant to traceability is stored on-chain.	Hyperledger Sawtooth is a permissioned network with public access by design.	Internal traceability information, Information with regard to individual eggs (pickup time, egg type, certification data, collection, temperature, brand, ...)	Temperature sensors.
Longo et al. [14]	Dual storage.	No authentication implemented per se, as the purpose of the study was to simulate a BT system to evaluate its economic impact	Inventory forecast and inventory information.	None
(NUTRIA) Niya et al. [18]	Single storage.	None.	Geolocation and time when the dairy product is at a certain chain actor, Producer identities, website URLs, certificates, product licenses, ...	None
Lin et al. [13]	Dual storage.	Each actor in the chain maintains its own server with full confidential information and regulates access to it.	Documentation and generic product data (non-specific).	None

Table 3. Prototype data architecture