sustainability

MDPI

Article

Blockchain-Based Formal Model for Food Supply Chain Management System Using VDM-SL

Hira Hameed [1], Nazir Ahmad Zafar [1], Eman H. Alkhammash [2] and Myriam Hadjouni [3,*]

[1] Department of Computer Science, COMSATS University Islamabad, Sahiwal Campus, Sahiwal 57000, Pakistan
[2] Department of Computer Science, College of Computers and Information Technology, Taif University, P.O. Box 11099, Taif 21944, Saudi Arabia
[3] Department of Computer Sciences, College of Computer and Information Science, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia
* Correspondence: mfhaojouni@pnu.edu.sa

Abstract: In modern society, the food supply chain management system has become an important research area realized nationally and internationally, which has established a collaborative relationship between producers, manufacturers, processors and retailers. Now, consumers are more interested in food quality, safety and the authentication of the products. Food safety has become an important issue in public health in the food market because people of all types and races around the world are affected due its poor quality. The traditional supply chains are centralized and face different issues such as lack of transparency, accountability and audit ability. The general issues in supply chain management include lack of communication, trust among the stakeholders, and interference of entities and wastage of food. A lot of work has been completed by researchers to address the above issues, but still, there is a need for effective mechanisms for the modeling of supply chain management systems. In this paper, a trusted, self-organized, traceable formal system based on blockchain and Internet of Things (IoT) is developed by using wireless sensors networks and finite automata. In the proposed model, smart contracts are designed to assure the automated payment procedures. The proposed model reduced the need for centralized authority. Unified Modeling Language (UML) is used to realize the requirements, and automata is used to capture behavior of the system. A blockchain-based model is used to provides the privacy and security mechanism for the transitions record. Wireless sensors are used to sense the information, and actors are used for decision making in case of any violation in the contact. A lot of work has been completed by researchers on smart contracts. Different smart contracts are designed to achieve the better traceability among producers, transporter/logistics and consumers. Our system provides the smart contract algorithm to show the interaction of entities in the food supply chain management system. Vienna Development Method-Specification Language (VDM-SL) is used to describe the formal system and the VDM-SL toolbox is used for the validation and analysis of the system.

Keywords: blockchain; IoT; smart contract; formal method; validation; VDM-SL; centralized; trusted

Citation: Hameed, H.; Zafar, N.A.; Alkhammash, E.H.; Hadjouni, M. Blockchain-Based Formal Model for Food Supply Chain Management System Using VDM-SL. *Sustainability* **2022**, *14*, 14202. https://doi.org/10.3390/su142114202

Academic Editor: Marc A. Rosen

Received: 28 September 2022
Accepted: 25 October 2022
Published: 31 October 2022

1. Introduction

Supply chain management (SCM) is the entire production process, including the raw materials, data and assets, which it converts it into final products that reach their final destination. It is the foundation of industry. By managing the supply chain, the extra cost of delivering the product to the consumer is reduced. A proper and well-managed supply chain system is essential for the smooth working of any industry. Even minor interruptions in the supply chain can destroy the entire market and cause huge financial losses for the companies involved. A well-maintained and immutable supply chain is required to detect the origin of fake products. The supply chain is a network in which people, organizations as well as production processes are involved that convert raw materials into final products and

distribute the products to customers [1]. The supply chain network is divided into different stages, including the extraction of raw materials, processing, distribution, and delivery of finished products, and all these stages take months to complete. If the finished product is lack of quality, tracing the root cause of the problem becomes extremely difficult [2].

To ensure product safety, it is important to monitor all processes of agricultural products, and there is the need for an effective management system in the agricultural and food supply chain. The agricultural and food supply chain's dynamic nature makes it difficult to monitor and trace products. Increasing the concerns about food safety and contamination risks have converted the focus to improved traceability in the supply chain. With the rapid growth of the economy and the continuous improvement of people's living standards, food safety has gained global attention in modern society. Therefore, people need a food traceability system that can control and track the entire food production cycle, food processing, transportation, storage and sales [3].

To gain the trust of consumers, supply chain experts must be reliable and efficient in the product delivery. The integrity, consistency and reputation of the supply chain methods are critical issues for supply chain authorities. Many governments have implemented standards to improve supply chain traceability systems' efficiency, transparency, and safety. Governments in different countries are strictly enforcing these standards [2]. The supply chain system has a complex network due to the large number of transaction records. Traditional supply chains are centralized and suffer from different issues such as failure of single point and a lack of transparency shown in Figure 1. Since they depend on a third party for payment, these systems are inefficient and unreliable [4].

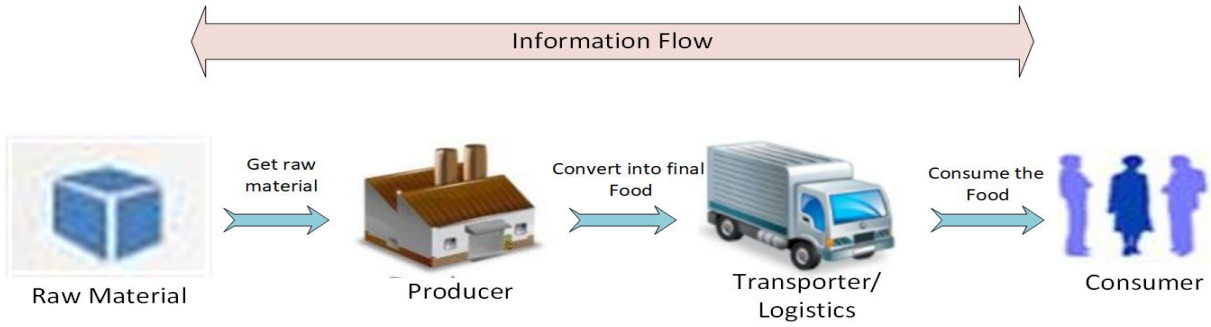

Figure 1. Food Supply Chain Management without Blockchain.

Furthermore, a centralized storage system is incapable of handling large amounts of data, which affects the overall efficiency of the network. Furthermore, researchers are focusing their efforts on improving the agriculture food supply chain's immutability, trust, accountability and full traceability [5].

One of the possible solutions to address the above-mentioned issues is the use of blockchain technology and IoT technology. Blockchain technology is immutable, transparent, and decentralized, allowing for secure communication between entities without the use of central authorities [6]. Blockchain is a decentralized system, and these systems are executed by the use of smart contracts. The traditional smart contract summarizes all the rules of the system and assures their execution by automation. In the blockchain technology, each of the entities has a transaction history copy of all transitions completed in the blockchain, which is known as a ledger. It contains different blocks, and these blocks constitute the set of transactions. These blocks are associated by hashing, and each hash is found by using the hash function. The hash function receives a block as an input and return the static length value [7].

With its various properties, blockchain is a safe scheme that addresses the threats of the central system and plays an important role in the food supply chain system. The Internet of Things has the ability to affect the global food supply chain by increasing supply chain performance. Modern IoT-based technologies are needed to track food quality and

increase the level of visibility of monitored data in order to improve safety and avoid the wastage of food [8].

Several methodologies are being proposed to address the existing challenges, but due to the introduction of emerging technology, many changes are still needed. The authors in [9] describe the reference model for the process of wine traceability. The reference model focuses on the important issues for which effective traceability is necessary to avoid fraud. After traceability, the reference model is used for business transformation. In [10], the authors propose a traceability system based on IoT Services and Smart RFID Tags, in which temperature sensors are built into RFID tags to monitor the condition of food during transportation and resolve the interconnection issue and cost implementation. The authors in [11] focus on how innovative smart packaging solutions improve overall food supply quality and safety by increasing device traceability and reducing food losses. In [12], the authors proposed auditable protocols for fair payment and temperature-proof transactions using smart contracts. The authors did not consider the credibility of entities in this work. In this study [13], the authors proposed a blockchain-based reputation system which logs reviews and increases trust among trade entities. The authors in [4] proposed the decentralized proof of delivery system that uses blockchain and Ethereum smart contracts to provide tamper-proof logs that are immutable, accountable, and traceable. Ethereum smart contracts are used to monitor all transactions and interactions related to the selling of digital products. In [2], the authors proposed an agriculture food supply chain based on blockchain in which transactions are defined to upload data to the IPFS, which provides smart contracts to show entity interaction.

In this paper, a blockchain-based traceable formal model is proposed by the integration of approaches for a food supply chain management system. A lot of work has been completed on simulation in a food supply chain, but this does not provide any mechanism for validation and verification. In this paper, we provide the mechanism for the validation and verification of a food supply chain management system. Unified Modeling Language (UML) is used to represent the interaction between the objects and realize the requirements. UML-based models are used to describe the flow of the system from one object to another. Non-Deterministic finite automata (NFA) are used to capture the behavior and actions to be performed over the system. The proposed model used sensors for tracking the products and actors for decision making on the basis of smart contracts. Smart contracts are proposed to assure the automated payment. Different smart contracts are designed to achieve better traceability among producers, transporter/logistics and consumers. Our system provides the smart contract algorithm to show the interaction of entities in a food supply chain management system. Furthermore, the blockchain-based model reduced the need for a centralized system. Formal methods are used to analyze the system constraint, system checking, syntax checking and system modeling, which ensure the system consistency, validity, security and privacy. Formal methods provide the mathematical strategies and methods for developing the system model. VDM-SL is the technique of formal methods which is used to formalize the system. The specifications and correctness of the system are analyzed and validated by using the VDM-SL toolbox. Therefore, in this study, we present the fully verified and validated blockchain-IoT based formal model of a food supply chain management system. The major contributions of this paper are:

- Develop a blockchain-IoT based formal model for food supply chain management by using an integration of approaches;
- Design smart contracts to show the interaction of entities;
- To automate the system, we convert the UML diagrams into an automated model by using finite automata;
- Enhance the efficiency of the system by collecting the required information and provide an effective communication mechanism;
- Describe a formal model and ensure its correctness through various validation and analysis techniques.

The remaining paper is structured as follows. Related studies that use blockchain and IoT in food supply chains are presented in Section 2. In Section 3, we present the system model, sequence diagram and algorithm that show the interaction between entities. In Section 4, we describe the formal model for food supply chain management.

2. Related Work

There are several works in this area, and some of the most important ones are discussed here. In [14], the authors in the Big Data and blockchain fusion application environment proposed an efficient supply chain framework. To revise the demand function, the author chooses a green agri-food supply chain from one producer and one retailer built the ISBD and analyzed a model that takes into consideration changes in the agri-food industry. In [15], the authors address the benefits and challenges of implementing blockchain-based traceability systems as well as blockchain-based solutions for food traceability issues and an analysis of blockchain characteristics and functions. The authors in [16] proposed a full grain supply chain based on blockchain technology and design a multimode storage system that combines chain storage. This method is a global standard for ensuring traceability, food security, and essential safety processes. In [17], the authors proposed a blockchain-based supply chain delivery system for eggs. The main goal is to monitor goods from farm to fork using enabled technologies such as IoT and blockchain. Traceability and accountability reduce the risk, fraud and loss of products. In [18], the author promotes the blockchain ecosystem. A study is being performed on the application of blockchain technology and its ecosystem of data storage as well as innovations and explained principles of blockchain technology. In [19], the authors recognized the enablers for blockchain adoption in an ASC (Agriculture Supply Chain). This is the first research to promote BT adoption enablers and ISM (Interpretive Structural Modeling) and DEMATEL's (Decision-Making Trial and Evaluation Laboratory) integrated approach. The authors in [20] proposed blockchain-based frameworks and real-time logistic status tracking. This scheme will be possible to implement in order to eliminate costs associated with cash backlogs. In [21], the authors develop a decentralized and automated traceability framework that is based on smart contracts and blockchain. The proposed model is based on a set of main performance indicators which have been pre-defined. The authors in [22] proposed a restaurant prototype traceability framework and used the FQI (Food Quality Index) algorithm to create a Food Quality Index value using product identifiers and blockchain. A comparison of the related work in this area considering traceability, efficiency and food safety is presented in Table 1.

A blockchain and IoT food traceability system is proposed in [30], which integrates blockchain technology, fuzzy logic and IoT technology. Shipment transit time and stakeholder assessment are being taken into consideration in the creation of integrated consensus mechanisms. In [31], the author proposes a new blockchain and utilizes some special tags. A special tag obtains information about the product that is reliable in terms of manufacturing. The authors in [32] associate blockchain technology with the Internet of Things (IoT) in order to give accountability and effectiveness in the supply chain. With strong authentication, blockchain provides tamper-proof data. The authors proposed a trust chain in [33], which uses a blockchain to track relations among supply chain members and dynamically allocates trust and reputation scores. In [34], the authors proposed the AgriBlock IoT, which is completely decentralized and provides a blockchain traceability solution for an agriculture and food supply chain management system. To gain traceability, the Ethereum and Hyperledger saw tooth used two different blockchain implementations. The author in [35] develops and implements an agricultural blockchain service system for farm-to-fork traceability using IoT sensors. Smart contracts are used to help tamper-proof data storage, and the Ethereum blockchain technology is also used. However, in the agriculture and food supply chain, safety and privacy are the main issues. In this regard, the authors in [36] proposed blockchain technology for IC manufacturing supply chain security and trust. We create a smart contract that specifies the products to be produced as well as the program code to be executed. The authors in [37] proposed a system for food supply chain

information security which is based on blockchain. However, this system does not provide the required traceability for markets.

Table 1. A comparison of related work considering traceability, efficiency and food safety.

Reference #	Year	Technology	Contribution	Issues/Limitation
[3]	2018	Blockchain and IOT technology	A trusted, self-organized and ecological smart agriculture system is built by using blockchain and IOT technology. Use IOT devices instead of manual verification and recording and reduce the human interference.	To find problems and process by law executors is difficult.
[23]	2019	Blockchain smart contracts	All product-transporting histories are verified by using smart contracts. Design the event response mechanism to verify the identities for the transaction. Build a decentralized application.	Possibility of manual input errors. Improvement needed in consumer consumption experience
[24]	2019	Blockchain and EPCIS	To improve the traceability system. The main issues, including data blast, trust and sensitive information disclosure, need to be solved.	Amount of data limit the system performance. Consensus algorithm is not used to improve the system throughput.
[25]	2019	Ethereum Blockchain and smart contracts	Eliminates the trusted centralized authority and provides a transaction record, safety with high integrity, reliability and security.	Lack of automated payments and proof of delivery
[26]	2016	Blockchain technology and RFlD	In this paper, the author covers the entire process of data collected and information in an agriculture and food supply chain.	Lack of decentralized system.
[27]	2009	RFID technology	Automation is used for an internal traceability system. High automation brings accuracy, completeness, and reliability.	It require a large amount for starting, and its maintenance is smaller.
[28]	2017	Blockchain and Internet of Things technology	Improve the efficiency, transparency and deliver real-time information to members.	The cost of RFID is very high.
[29]	2020	Blockchain technology	Improve the traceability, accountability and automated payment.	Delivery mechanism and credibility are not addressed.

In [38], the effect of blockchain technology on agriculture and the food supply chain is examined. The author describes the exiting project, discusses the overall implementation and challenges and concludes that blockchain is a promising technology for developing a transparent food supply chain. The authors in [39] consider the grape wine supply chain and use a rating-based conjoint approach to identify some potential drivers of blockchain technology adoption. The author in this study discovers that in order of relative importance and utility, disintermediation, traceability, price, confidence, enforcement, and coordination and control can all affect the supply chain actors' decision processes.

In [40], the authors show the comprehensive survey on the role of integrating both blockchain and IoT for developing smart applications in agriculture. In this paper, novel blockchain models are proposed that are used for major challenges in IoT, and it also discussed the security and privacy challenges. The authors in [41] focus on the FSC (Food Supply Chain) and conduct a literature review in order to better understand the role of internet of thing and other technology such as blockchain on FSC management. This review also shows the literature dealing with the decision support system, which enhances the use

of the IoT in decision making. In [42], the authors proposed a system on the smart contracts to track the flow of food supply chain systems and break down the information between enterprises to eliminate the need for central institutions to improve the transition record and security. Farmers store the record detail of crops in the IPFS file, and the IPFS file stores hashes in a smart contract.

In [43], the authors design and develop a software framework that allows the Internet of Things to interact directly with an Ethereum-based blockchain. The proposed solution represents another way to interact with the Internet of Things without relying on a centralized intermediary and third-party services. In [44], the authors introduced a new integrated approach to logistics and food quality research by implementing ALADIN, which is a pre RTL simulator design used in a new simulation environment. It uses discrete event models to integrate food quality transition models and sustainability indicators. In paper [45], the aspects of supply chain management that are affected by traceability are presented. The paper has two goals: the first is to examine how current supply chain management is influenced by traceability principles, standards, and technologies. The authors' second aim is to illustrate what they assume are the future developments and perspectives in this field of study. The author in [46] provides a comprehensive examination of blockchain and its possible applications in the industrial sector, including Industrial Internet of Things-based sensor networks and the effect of blockchain technology on manufacturing-related areas such as traceability. With a comprehensive examination of its technological viability, the author focuses on a possible implementation of blockchain in SCQM. The concept of blockchain and its existing applications in logistics and supply networks are presented by the author in [47]. The current and potential future applications of blockchain in logistics and supply chain are described and justified in this paper. In [48], the author proposed a concept for a smart logistics monitoring system. Only four modules are described here: system management, order, customers, and manufacturer. It uses wireless sensors, RFID, RTLS, and actors to communicate with the system's various components. VDM-SL is used to create formal specifications for the components of our proposed model.

The author in [49] proposed an automated blockchain-based formal model for a parking system. The author of this paper used VDM-SL for validation and the Unified Modeling Language for requirements. The authors in [50] proposed the registration system, which is based on ubiquitous computing, and this system is modeled by a unified modeling language, NFA and VDM-SL. A different operation is discussed in this paper. According to the research presented above, the use of blockchain technology in supply chain management systems is rising exponentially. In supply chain management systems, blockchain technology is being used to improve traceability, food safety, and transparency. As a result, a traceable formal model based on blockchain is proposed by acquiring motivation from IoT technology and blockchain.

3. System Model

The main issues faced in the food supply chain management system are inconsistency and incorrectness. The existing supply chain system is centralized, depends on a third party and is not validated and verified by formal modeling methods. Food quality and safety has become a growing concern for the public health system in recent years. It is estimated that millions of people fall ill after eating contaminated food. Therefore, this system needs to remove these inconsistencies. To overcome these issues, we used a blockchain to increase the security and privacy and reduce the need for a centralized system. The IoT-based food supply chain used a sensor to sense the food condition. Our proposed model develops the blockchain–IoT-based formal model integrated with a UML, sensor, controller, non-deterministic finite automata and formal methods.

Our system model is discussed in this section. We define the traceable formal model for a food supply chain from farm to fork. The proposed model provides the secure system between different entities of the food supply chain system and uses the blockchain to handles all transaction records of the different events. The supply chain system consists of

large entities that carry out the complete process of food manufacture and delivery of food products from farm to end users. That is why it is difficult to trace and track the whole procedure. To achieve the complete traceability, we use the unique identifier for food and record each transaction. To ensure privacy, only the authorized user can access the data and ensure that the authorized user carried out the transactions. In our proposed system, smart contracts are designed to show the interaction of different entities including the producer, transporter/logistics and consumer, as shown in Figure 2. Only registered users perform the specific transaction, and users who are not registered are not allowed to perform any transaction. Different entities are registered and interact by using smart contracts.

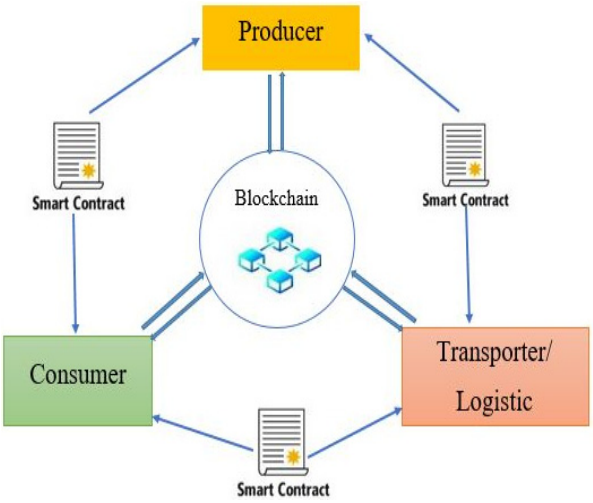

Figure 2. Blockchain-Based Supply Chain Management.

Each entity has a role and interacts with the smart contract. These entities are as follows.

- Producer: The producer is the first participant of the food supply chain and produces a large amount of food. He sells the food to the transporter/logistics. He records the food information and will receive the payment after 30 days and return the remaining amount to the consumer.
- Transporter/Logistics: A transporter buys the finalized food from the producer and is responsible for selling this food to the consumer. He is responsible for the fair delivery of the food from the producer to the consumer.
- Consumer: The consumer plays an important role in the food supply chain. He buys the food from the transporter. According to their needs, they can buy the right product and communicate with the producer.
- Manager: The manager authenticates the entities that are new in the network. He adds and verifies all entities of the network.

3.1. Sequence Diagram

Sequence diagrams are used to show the interaction of object and detail the information of the operations which are performed. They capture the behavior of the object. Sequence diagrams are used to describe functions which are performed in the system and how they are performed. In Figure 3, we show the interaction of different entities. There are different entities such as the producer, transporter, consumer, sensor, manager and controller. All entities' data are stored in the smart contract. This sequence diagram shows what operations are performed.

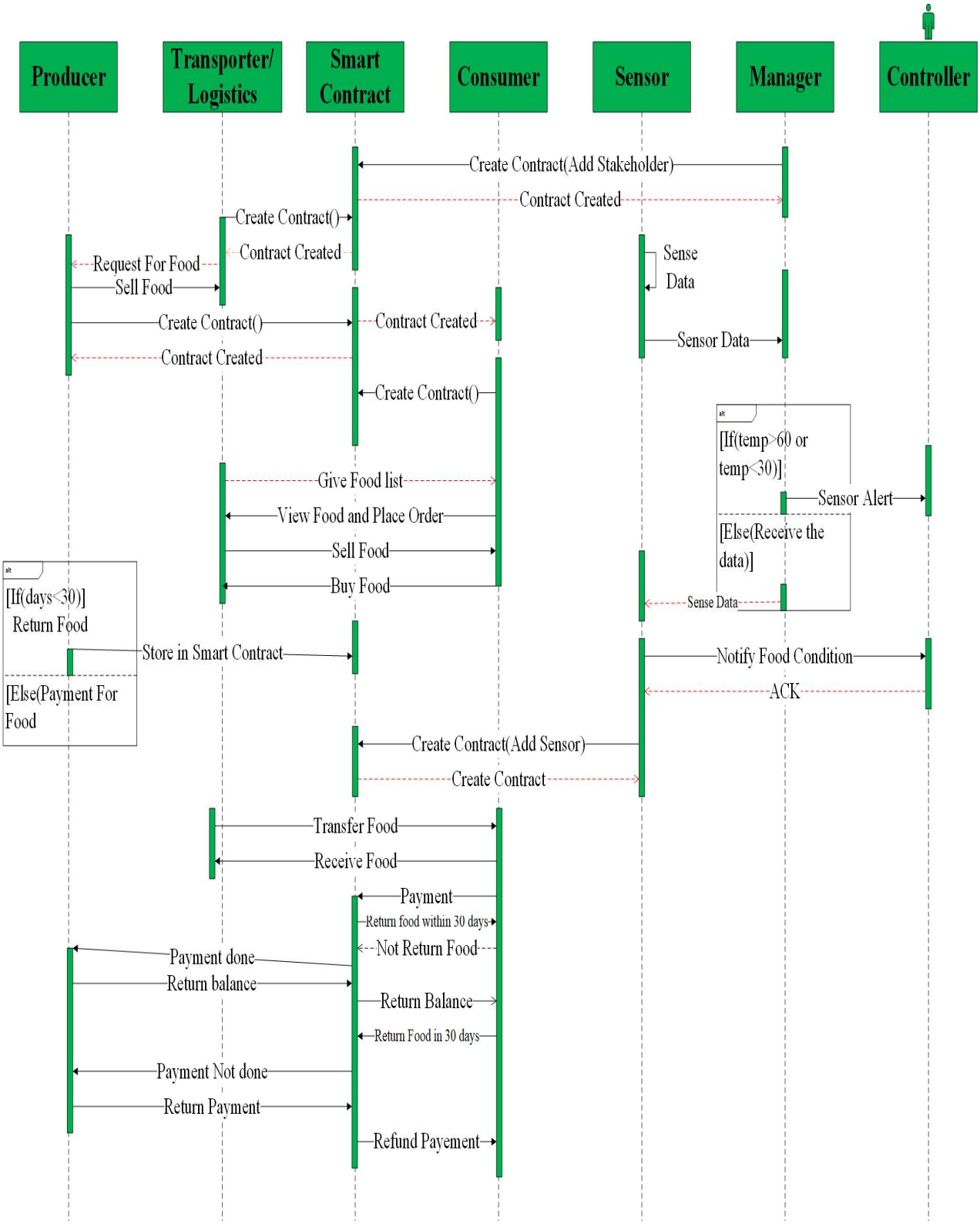

Figure 3. Sequence Diagram.

Every entity has an address and name and these data are stored in the smart contract. In Figure 3, the sequence flow of the system is shown, where a transporter requests for food and creates the smart contract. When the smart contract is created, it stores the information of the transporter. The producer creates the contract, sells the food to the transporter and

gives detail information about the food. The transporter gives the food list to the consumer, and the consumer creates the smart contract. Then, the information of the consumer is added into the smart contract. The consumer views the list of food, checks the food details, selects their desired food and orders it. After ordering the food, the transporter checks the order and sells the food to the consumer. The producer stores some conditions in the smart contract, such as food can only be returned within 30 days. The producer receives the payment if the consumer does not return the food within 30 days, according to the smart contract. The producer transfers the money in their account and returns the remaining money to the consumer. The consumer creates the smart contract and stores the money record in the smart contract.

The consumer checks the food; if he satisfied, then he transfers the payment to the producer's account. If the consumer is not satisfied with the food, then he return the food within 30 days, and the producer refunds the payment to the consumer. The manager is the major entity in supply chain management. He creates the contract and adds the producer, transporter and consumer. He is responsible for authentication of the entities. Only authorized entities can access the transaction and record this transaction. If the entities complete the contract, then manager destroys this contract. To check the food condition, sensors and a controller are used. A smart contract is created to add the sensor, and the sensor senses the data and gives the sensor data to the manager. The sensor is used to sense the condition of the food; if the food condition is not satisfactory, then the senor sends an alert to the manager. The manager check the sense data; if the temperature is greater than 60 or less than 30, then the manager sends this information to the controller. Otherwise, he receives the data and sends the information to the sensor to sense the data again. The sensor sends the notify condition to the controller.

3.2. Automated Food Supply Chain Management

Automata theory presents the mathematical properties of the system. Automata theory is used to capture the behavior of the system and describes the complete functionality of the system. These models help remove ambiguities from the method while also enhancing continuity and accuracy. UML uses natural language to define the system's static and dynamic properties, which helps to minimize ambiguity. As a result, automata models are built to increase the system's performance and accuracy. Non-deterministic finite automata (NFA) has five elements ($Q, \sum, q_0, F, \delta$)

(1) Q represents the set of state;
(2) $\sum$ represents the input letters;
(3) q_0 represents the start state;
(4) F represents the final state;
(5) δ represents the transition function Transition function is formally represented as $Q \times \delta \rightarrow P(Q)$.

NFA is used to model the food supply chain management system and captures the behavior of the system efficiently. In NFA, the states represent the producer, transporter, consumer and manager, and the edge describes which function is performed. Figure 4 shows the union of three NFAs that represent the flow of food, sensor information and payment process. The states of NFA are {P, T, S, C, M, S', C'} which represent the producer, transporter, smart contract, consumer, manager, sensor and controller.

We define the union of three NFA shown in Figure 4. Let the flow of food automata be represented by A1 = (Q1, $\sum$, q1, F1, δ1), food sensor information automata be represented by A2 = (Q2, $\sum$, q2, F2, δ2) and payment process automata be represented by A3 = (Q3, $\sum$, q3, F3, δ3). The union of automata is A = (Q, $\sum$, q0, F, δ), which is represented by A1UA2UA3.

1. Q = {q0}UQ1UQ2UQ3. A is the states of A1, A2 and A3 and q0 is the new state.
2. The state q is the new state of A.

3. F = F1UF2UF3 are the set of final states. The final state of A is all final states of A1, A2 and A3.
4. Now, define the transition function δ of union of three NFAs where the input letter is a and state is q.

$$\delta(q,a) = \begin{cases} \delta_1(q,a) & q \in A_1 \\ \delta_2(q,a) & q \in A_2 \\ \delta_3(q,a) & q \in A_3 \\ (q_1,q_2,q_3) & q = q_0 \text{ and } a = \epsilon \\ \emptyset & q = q_0 \text{ and } a \neq \epsilon \end{cases}$$

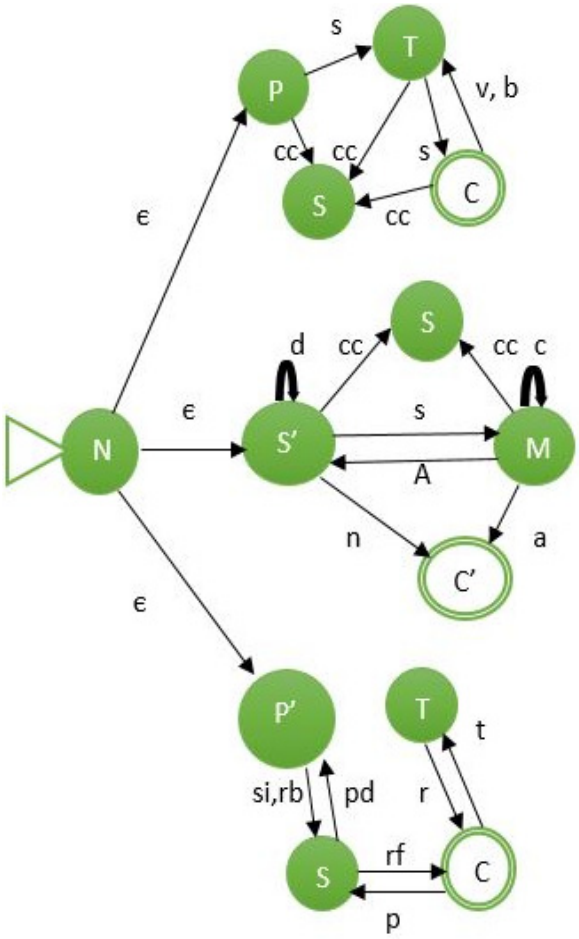

Figure 4. Union of Three NFA.

We define the concatenation of three NFAs shown in Figure 5. Let the flow of food automata be represented by $A_1 = (Q_1, \sum, q_1, F_1, \delta_1)$, food sensor information automata be represented by $A_2 = (Q_2, \sum, q_2, F_2, \delta_2)$ and payment process automata be represented by $A_3 = (Q_3, \sum, q_3, F_3, \delta_3)$. The concatenation of automata is $A = (Q, \sum, q_0, F, \delta)$, which is represented by $A_1 \circ A_2 \circ A_3$.

1. $Q = Q_1 U Q_2 U Q_3$. A is the states of A_1, A_2 and A_3.
2. The state q_1 is the new state of A_1.
3. F_3 is the final state of A_3.
4. Now, define the transition function δ of concatenation of three NFAs where the input letter is a and state is q.

$$\delta(q,a) = \begin{cases} \delta_1(q,a)\ q \in A_1 \text{ and } q\ \in F_1 \\ \delta_1(q,a)\ q \in F_1 \text{ and } a\ \neq\ \epsilon \\ \delta_1(q,a) \cup \{q_2\} \cup \{q_3\}\ \ q \in F_1 \text{ and } a = \epsilon \\ \delta_2(q,a)\ \ q \in A_1 \\ \delta_3(q,a)\ \ q \in A_3 \end{cases}$$

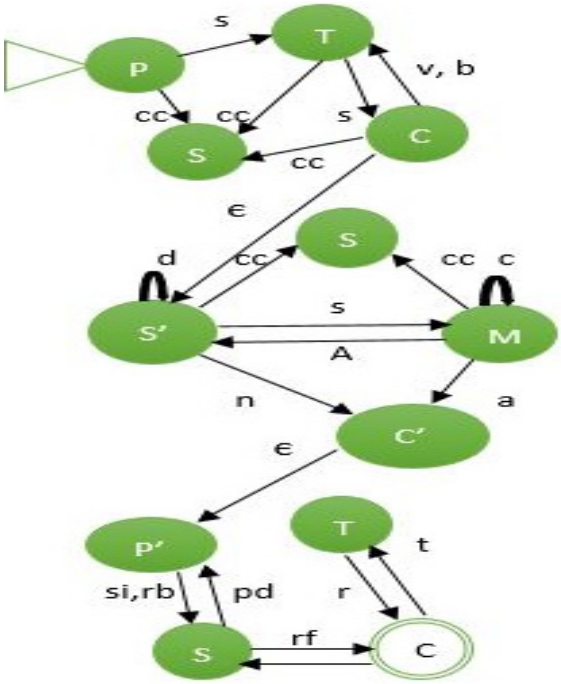

Figure 5. Concatenating of three NFA.

3.3. Smart Contract

A smart contract is a self-executing contract in which the terms of the buyer–seller agreement are written directly into lines of code. The code as well as the agreements are distributed across a decentralized blockchain network. Transactions are trackable and irreversible, and the code controls their execution. In this section, the smart contracts algorithms are discussed that show the interaction of different entities, and the language that is mostly used to write the smart contract on Ethereum is solid. We keep a record of the sensor, stakeholder and food information in the smart contracts to achieve better traceability. In Figure 6, different smart contracts are discussed.

There are four fields in the contract: user name, address, ID, and type of sensor. In these algorithms, we discuss three entities: transporter/logistics, producer, and consumer. We use p to represent the producer, t to represent the transporter, c to represent the consumers, and s to represent the sensor. The new stakeholder is verified by the manager, and the manager then enters this information into the contract and blockchain.

Algorithm 1 adds the sensor and stakeholder, the manager enters the stakeholder's address and name, and this contract is stored in the blockchain. In this algorithm, the food information status is discussed. For consumers, the food status is necessary during the ordering process. When customers place an order for food, they search the list and see which items are available and which are not. In this algorithm, we also discussed the order information status. If the order is completed successfully, the information about the customer ordering food is updated; otherwise, the consumer does not order food.

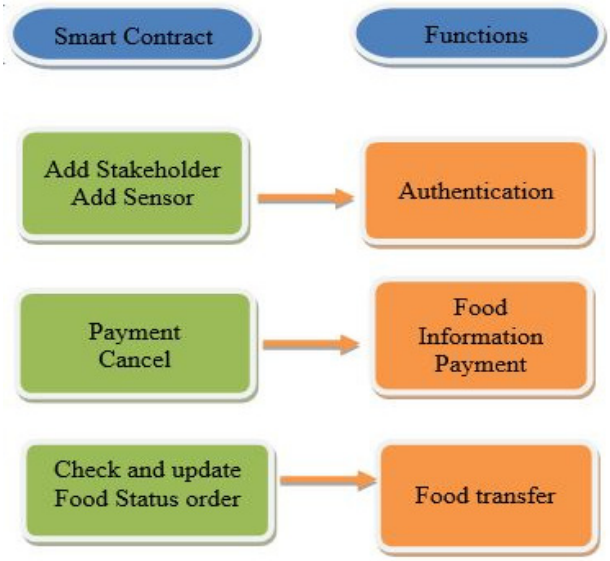

Figure 6. Smart Contract.

Algorithm 1: Add entities and food and order status.

```
Input: manager, user address, name
id, stype, consumers, transporter, s
1. If message. sender == manager
2. If keccak256(name) == keccak256(p)
3. Entity producer
4. Else If keccak256(name) == keccak256(t)
5. Entity transporter
6. Else If keccak256(name) == keccak256(c)
7. Entity consumers
8. Else If keccak256(stype) == keccak256(s)
9. successfully added sensor
10. If (message. sender == transporter
      && s == available)
11. Food is available
12. Else
13. Food not available in list
14. If (message. sender == consumers
      && s == Order)
15. Order food by consumers
16. Else
17. Not order food by consumers
18. End
```

The food is selected by the consumer for order. After receiving the approval of the order, the user can choose a series number randomly, which is denoted by l and then obtain the hash of l, which is denoted as Keccak256 (l) ==. (h_1). The obtained hash is sent to the producer with food details. l is the pickup code that is decided by the consumer.

The payment function is defined in Algorithm 2. If the customer receives the product and does not return it within 30 days, the contract will give the payment of food to the producer's account and return the amount of the contract to the consumer.

The payment refund function is defined in Algorithm 2. The contract returns payment back to each person if the consumer refuses to accept the food. However, if the consumer returns back the food within 30 days, the consumer receives an extra transportation fee. The hash primitive to be verified is represented by l.

Algorithm 2: Transfer payment and refund function.

Input: producer, consumers l
1. If (message. sender == producer && Keccak256(l) == (h_1) && time < 30days)
2. {
3. Payment transfer to producer
4. }
5. If (message. sender == consumers && keccak256(l) == (h_2) && reject food
6. {
7. Give payment to producer and consumer;
8. }
9. Else if (message. sender == consumers && keccak256(l) == (h_2) && return within days)
10. {
11. Give payment and transport fee to Consumer;
12. }
13. End

The logistics company uses Algorithm 3 to validate the product's acceptance, and if the acceptance is successful, then the information is updated to confirm the acceptance. If the product is returned, the consumer uploads the return details as well as the explanation for the return.

Algorithm 3: Food validation Information.

Input: transporter, consumers, l,purpose
1 If (message. sender == transporter && Keccak256(l) == (h_2) && s == true)
2 {
3 Food is received by consumer;
4 }
5 Else if (message. sender == consumers)
6 {
7 Food refuse purpose;
8 }
9 End

4. Formal Specification

The food supply chain management system is verified and validated by using the VDM-SL toolbox, which is the formal method. Formal methods are used to check the consistency and correctness of the system. Formal specification of the food supply chain management system is described by VDM-SL toolbox. Different notations of VDM-SL are used in the system model: numeric type, composite object, sets and types. The model has two parts: dynamic and static. In the static model, different data type are defined, and the composite object is also defined, which is important data type of VDM-SL toolbox. The composite type has different fields, which are used to represent the different operations of the model. Invariant is defined on the composite object, which defines the scope of the system. These conditions must be fulfilling in the system specification. In the dynamic model, all operations and functions are defined. In operation, different pre and post conditions are used to identify the conditions on the system. In the given specification, consumerid, managerid, stakeholderid, orderid, stype and sensor identifier are declared as the token type whose internal information is not required. The ctype is declared as enumeration types. The password and chipid are declared as natural numbers. The name is declared as a string which is a sequence of characters. A composite type time has three fields, i.e., hour, min, and sec. Invariant is used to define the time composite object in which hours are less than 24, minutes are less than 60 and seconds are less than 60. The payment status, order information, notification status, sensor alert, food status, login status, user status, stakeholder type and feedback are declared as enumeration types, which have the name value.

types
string = seq of char;
Consumerid = token;
Managerid = token;
Stakeholderid = token;
Orderid = token;
Identifier = token;
name = string;
password = nat;
sensor = token;
Stype = token;
chipid =nat;
ctype = <foodpacket>;
Time: hour: nat
min: nat
sec: nat
inv mk_Time (h, m, s) == h < 24 and m < 60 and s < 60;
foodstatus = <Available> | <Not_Available>;
paymentstatus = <Paid> | <NotPaid>;
Orderinfo = <valid> | <Invalid>;
Typestakeholder = <producer> | <transporter> | <consumer>;
Notificationstatus = <order> | <Not_order>;
SAlert = <temperature_increase> | <temperature_decrease>;
Userstatus = <login> | <logout>;
loginstatus = <Successful> | <Unsucessful>;
feedback = <good> | <Not_good>;

The stakeholder has seven fields: i.e., identifier, name, password, food type, order, stakeholder type and status, which is a composite object. The stakeholder may be a producer or transporter, and every stakeholder has a password and login detail to access the system.

Stakeholder:: sid: Stakeholderid
sname: name
spassword: password
foodtype: foodstatus
order: Orderinfo
Stakeholdertype: Typestakeholder
Sstatus: Userstatus;

A composite object consumer has six fields, i.e., consumer identifier, name, password, order, balance, status and pay bill. Pay bill has three fields: day, amount and payment status. Invariant indicates that days must be less than 30. Invariant is used to define the composite type: a consumer who orders food must have a balance greater than zero on their credit card.

Consumer:: cid: Consumerid
cname: name
cpassword: password
order: Order
balance: nat
cstatus: Userstatus
paybill: set of Paybill
inv mk_Consumer(-,-,-,-,b,-,-)== have_amount(b);
Paybill:: days:nat
Amount:nat
Status: paymentstatus

inv mk_Paybill(d,-,-) == Days(d);

The sensor has three fields: identifier, type and message, which is a composite object, and the sensor has the unique identifier. A chip has two fields: chip identifier and chip type, which is a composite object. Food has two fields: food identifier and food name, which is a composite object. The food identifier is used to uniquely identify the food. Order has two fields: order identifier and order type, which is a composite object.

Sensor:: sensorid: Identifier
type: Stype
msg: SAlert;
Chip:: chid: chipid
chip: ctype;
Food:: fid: chipid
fname: name;
Order:: oid: Orderid
Otype: Orderinfo;

A composite object actor has an action field. The actor works as a controller that checks the food condition after low and high temperature.

action = <Low_temperature> | <High_temperature>;
Actor:: Action: action;
A composite type manager has two fields manager identifier and manager name.
Manager:: mid: Managerid
Mname: name;

A composite object smart contract has ten fields, i.e., manager identifier, food identifier, stakeholder identifier, consumer identifier, sensor identifier, stakeholder registration, consumer registration, sensor registration and bill reduction. The smart contract stores all the information of all these fields.

SmartContract:: mid: Managerid
fid: chipid
sid: Stakeholderid
cid: Consumerid
sensorid: Identifier
Sreg: map Stakeholderid to Stakeholder
Creg: map Consumerid to Consumer
Sensorreg: map Identifier to Sensor
sensorinfo: map Identifier to Sensor
Cbill: map Consumerid to Consumer;
The state “food supply chain management” consists of twenty attributes.
state FoodSupplyChainManagement of
stakeholdermember: set of Stakeholder
registerstakeholder: map Stakeholderid to Stakeholder
regconsumer: map Consumerid to Consumer
consumers: set of Consumer
stakeholder_sell: map chipid to Stakeholder
sensordetail: map Identifier to Sensor
Sensorreg:map Identifier to sensor
chips: set of Chip
assignchip: map chipid to ctype
regfood: map chipid to Food
food: set of Food
checkfood: map name to Food
order: set of Order
orderdetail: map Orderid to Order

manager: map Managerid to Manager
Transaction: map Consumerid to SmartContract
Record: map Stakeholderid to SmartContract
Sensorrecord: map Identifier to SmartContract
Foodrecord: map chipid to SmartContract
record: map Consumerid to SmartContract
end

This function takes the consumer balance as input and gives the result in the Boolean form. If the balance is greater than 0, then it returns true; otherwise, it returns false.

functions

have_amount(balancein: nat) result: bool
pre true
post result <=> (balancein >0);

The day function takes the day as input and give the result in the Boolean form. If the day is less than 30 than it returns true otherwise returns false.

Days(dayin: nat) Result: bool
pre true
post Result <=> (dayin < 30);

The "Add sensor" operation adds a sensor to the system. The Add sensor operation takes the identifier, type and message as input. In the pre-condition, the sensor does not belong to the register sensor set. In post, we add the new sensor and store the information of the sensor into the smart contract.

Operations

Addsensor(sensorIn: Identifier, typeIn: Stype, msgIn: Salert)
ext wr Sensorreg:map Identifier to sensor
wr Sensorrecord: map Identifier to SmartContract
pre sensorIn not in set dom Sensorreg
post Sensorreg = Sensorreg ~munion{sensorIn |-> mk_Sensor(sensorIn,typeIn,msgIn)} and Sensorrecord(sensorIn).sensorid = sensorIn;

The "sensor alert" operation senses the data from the food, and in post condition, the sensor sends the alert for temperature over the blockchain.

sensor_alert()
ext rd sensordetail: map Identifier to Sensor
wr Sensorrecord: map Identifier to SmartContract
pre true
post exists s in set dom Sensorrecord & Sensorrecord(s).sensorinfo
= sensordetail;

The "registration stakeholder" operation takes the identifiers of stakeholder, name, password, food type, order and stakeholder type as input. In the pre-condition, the stakeholder does not belong to the registered stakeholder set, and in the post condition, it adds the new record of the stakeholder and stores these data into the smart contract.

registrationstakeholder(IdIn: Stakeholderid, nameIn:name, passwordIn: password, food-typein: foodstatus,orderIn: Orderinfo, typeIn: Typestakeholder)
ext wr registerstakeholder: map Stakeholderid to Stakeholder
wr Record: map Stakeholderid to SmartContract
pre IdIn not in set dom registerstakeholder
post registerstakeholder = registerstakeholder ~munion{IdIn |-> mk_Stakeholder(IdIn,nameIn,passw
typeIn, <login>)} and Record(IdIn).sid = IdIn;

The "stakeholder login" operation takes the name, password and stakeholder type as input and gives the login status of the stakeholder. In the pre-condition, the person is

a stakeholder, and the status of this person is logout. In the post condition, if the input name, password and stakeholder type match with the stakeholder name and password, then the user is an authorized person and is able to successfully login; otherwise, the user is unauthorized.

Login(nameIn:name, passwordIn:password, typeIn: Typestakeholder)
output: loginstatus
ext rd stakeholdermember: set of Stakeholder
wr Record: map Stakeholderid to SmartContract
pre exists m in set stakeholdermember & m.Sstatus= <logout>
post forall m in set stakeholdermember &
if m.sname = nameIn and m.spassword = passwordIn and m.Stakeholdertype =typeIn then output = <Successful> else output= <Unsucessful>;

The "remove" operation takes the stakeholder identifier. (i) In the pre-condition, the stakeholder member is not empty and input ID belongs to the registered stakeholder. (ii) In the post condition, the identifier person is deleted from the register stakeholder.

remove(IdIn: Stakeholderid)
ext wr registerstakeholder: map Stakeholderid to Stakeholder
rd stakeholdermember: set of Stakeholder
rd Record: map Stakeholderid to SmartContract
pre stakeholdermember <> {} and IdIn in set dom registerstakeholder
post registerstakeholder = {IdIn}<-:registerstakeholder~;

The "registration consumer" operation takes the identifier of the stakeholder, name, password, order and stakeholder type. In the pre-condition, the consumer is able to login successfully, the consumer does not belong to the register consumer set, and the type of stakeholder is consumer. In the post condition, we add the new record of the consumer and store this information into the smart contract.

registrationconsumer(IdIn: Consumerid, nameIn:name, passwordIn: password, orderIn: Order, typeIn: Typestakeholder)
ext wr regconsumer: map Consumerid to Consumer
wr record: map Consumerid to SmartContract
pre Login(nameIn,passwordIn,typeIn)=
<Successful> and
typeIn=<consumer> and IdIn not in set dom regconsumer
post regconsumer = regconsumer ~munion{IdIn |-> mk_Consumer(IdIn,nameIn,passwordIn, orderIn,0,<login>,{})} and record(IdIn).cid = IdIn;

The "get chip id" operation takes the identifier of chip and give the result as chip id. In the pre-condition, chips are not empty, and the chip identifier belong to the chip set. In the post condition, the result is the chip identifier.

get_chip_id(chId: Chip) result:chipid
ext rd chips: set of Chip
pre chips <> {} and chId in set chips
post result = chId.chid;

The "register food" operation takes the food identifier and food name. (i) In the pre-condition, the food identifier does not belong to the register food. (ii) In the post condition, we check that the get chip id of food is equal to the food identifier and add the new food record, and the food identifier is stored in the smart contract.

registerfood(fId: Chip, Fname: name)
ext wr regfood: map chipid to Food
rd food: set of Food
wr Foodrecord: map chipid to SmartContract
pre fId.chid not in set dom regfood

post get_chip_id(fId) = fId.chid and regfood = regfood~ munion {fId.chid |-> mk_Food(fId.chid,Fnar
and Foodrecord(fId.chid).fid= fId.chid;

The "get food status" takes the name of the food as input. In the pre-condition, we check that the food belongs to the set of food and the name of the food matches. In the post condition, if the name of the food belongs to the food list, then the food is available; otherwise, the food is not available.

get_food_status(Fname: string) output: foodstatus
ext rd food: set of Food
rd Foodrecord: map chipid to SmartContract
rd checkfood: map name to Food
pre exists f in set food & f.fname=Fname
post if Fname in set dom checkfood then output=<Available> else output=<Not_Available>;

The get food operation obtains the food from the list. In the get food operation, it takes the food identifier as input and returns the food list.

getfood(fId: chipid) foodlist: set of Food
ext rd food: set of Food
pre true
post foodlist= {F |F in set food & F.fid = fId};

In the sell food operation, the stakeholder sells the food. The "Sell food" operation takes the stakeholder identifier, stakeholder type, name, password and food identifier as input. In the pre-condition, the food identifier must belong to the registered food, and the stakeholder identifier must belong to the registered stakeholder. In the post condition, the chip id of the food and stakeholder must be able to login successfully, and the stakeholder type is producer; then, we delete the food from the stakeholder.

Sell_food(IdIn:Stakeholderid,typeIn: Typestakeholder, nameIn:name, passwordIn: password, fId: Chip)
ext wr registerstakeholder: map Stakeholderid to Stakeholder
wr regfood: map chipid to Food
wr stakeholder_sell: map chipid to Stakeholder
pre fId.chid in set dom regfood and IdIn in set dom registerstakeholder
post get_chip_id(fId) = fId.chid and Login(nameIn, passwordIn,typeIn)=<Successful> and
typeIn=<producer> and stakeholder_sell = {fId.chid }<-: stakeholder_sell~;

The "get order status" takes the order identifier as input and returns the order information. In the pre-condition, we check that the order belongs to the set of the order and the name of the order matches. In the post condition, if the name of the order belongs to the order list then the order is valid; otherwise, the order is not valid.

getorderstatus(Oid: Orderid) Output: Orderinfo
ext rd order: set of Order
rd orderdetail: map Orderid to Order
pre exists o in set order & o.oid=Oid
post if Oid in set dom orderdetail then Output=<valid>
else Output=<Invalid>;

The "place order" takes the consumer identifier and order identifier as input and returns the notification order, which is placed or not. In the pre-condition, we check that the order belongs to the set of the order and the name of the order matches. In the post condition, if the consumer identifier belongs to the registered consumer, then the order is place; otherwise, the order is not placed.

placeorder(Cid: Consumerid, Oid: Orderid)result: Notificationstatus
ext rd order: set of Order
wr regconsumer: map Consumerid to Consumer
pre exists o in set order & o.oid=Oid

post if Cid in set dom regconsumer then result=<order>
else result=<Not_order>;

The "get balance" takes the consumer identifier as input and returns the balance as output. In the pre-condition, the consumer identifier belongs to the set of the consumer, and the consumer balance is not zero. In the post condition, we obtain the total balance of the consumer and return this as output.

getbal(IdIn: Consumerid)balOut: nat
ext rd regconsumer: map Consumerid to Consumer
rd consumers: set of Consumer
pre exists IdIn in set consumers & IdIn.balance <>0
post balOut = (regconsumer(IdIn)).balance;

The "payment" operation takes the consumer identifier, name, password, stakeholder type, day and amount as input and gives the output as login status. In the pre-condition, the stakeholder must be able to login successfully and the type of stakeholder must be consumer. In the post condition, the let in clause is used to define the payment operation. First, let in gives the name Balance to the old consumer balance. Second, the let in clause gives the name payment to the old payment transition record of the consumer. The final let in gives the name new payment, consisting of the days, amount and status that is paid, and it adds the appropriate amount of the food and takes union with the new payment. When the payment is completed, it gives the output successfully and stores these data into the smart contract.

Payment(namein:string, passwordin:password, IdIn:Consumerid, typein: Typestakeholder, dayin:nat, amountin:nat)
out: loginstatus
ext wr regconsumer: map Consumerid to Consumer
wr consumers: set of Consumer
wr record: map Consumerid to SmartContract
pre Login(namein, passwordin, typein)= <Successful> and
typein= <consumer>
post let Balance= (regconsumer~(IdIn)).balance
in let payment= (regconsumer~(IdIn)).paybill
in let newpayment= mk_Paybill(dayin,amountin,<Paid>)
in regconsumer= regconsumer~ ++ {IdIn |-> mu(regconsumer~(IdIn), balance
|-> Balance+amountin, paybill |-> payment union {newpayment})} and
out= <Successful> and record(IdIn).Cbill = regconsumer;

The "Refund" operation takes the consumer identifier, name, password, stakeholder type, day and amount as input and gives the output as login status. In the pre-condition, the stakeholder must be able to login successfully and the type of stakeholder must be consumer. In the post condition, the let in clause is used to define the payment operation. First, let in gives the name Balance to the old consumer balance. Second, the let in clause gives the name payment to the old payment transition record of the consumer. The final let in gives the name new payment, consisting of the days, amount and status that is not paid, and it subtracts the appropriate amount of the food and takes union with the new payment. When the payment is completed, it gives the output successfully and stores these data into a smart contract.

Refund(namein:string, passwordin:password, IdIn:Consumerid, typein: Typestakeholder, dayin:nat, amountin:nat)
out: loginstatus
ext wr regconsumer: map Consumerid to Consumer
wr consumers: set of Consumer
wr record: map Consumerid to SmartContract
pre Login(namein, passwordin, typein)= <Successful> and

typein= <consumer>
post let Balance= (regconsumer~(IdIn)).balance
in let payment= (regconsumer~(IdIn)).paybill
in let newpayment= mk_Paybill(dayin,amountin,<NotPaid>)
in regconsumer= regconsumer~ ++ {IdIn |-> mu(regconsumer~(IdIn), balance
|-> Balance-amountin, paybill|-> payment union {newpayment})} and
out= <Successful> and record(IdIn).Cbill = regconsumer;

5. Formal Analysis and Result

A food supply chain management system is formally verified by using the VDM-SL toolbox. By using the VDM-SL toolbox syntax checker and type checker, the system syntax and type are checked. Pretty Printer is represented by "P" and ensures that the pre/post condition and invariants are not consistent in the system. In Figure 7, system analysis is shown. In this figure, syntax is represented by "S" and indicates no syntax error in the system, type is represented by "T" and indicates no type error in the system, and C++ is represented by "C" and analyzed for automatic translation.

Table 2 shows the system analysis of the food supply chain. It indicates there is no syntax, type, code generation and pretty print error in any operation. All operations of the food supply chain system are consistent and correct.

Table 2. Formal analysis of food supply chain management.

Operation	Syntax Checker	Type Checker	C++	Pretty Printer
Add sensor	√	√	√	√
Sensor alert	√	√	√	√
Registration stakeholder	√	√	√	√
Stakeholder Login	√	√	√	√
Remove	√	√	√	√
Registration consumer	√	√	√	√
Get chip id	√	√	√	√
Register food	√	√	√	√
Get food status	√	√	√	√
Get food	√	√	√	√
Sell food	√	√	√	√
Get order status	√	√	√	√
Place order	√	√	√	√
Get balance	√	√	√	√
Payment	√	√	√	√
Refund	√	√	√	√

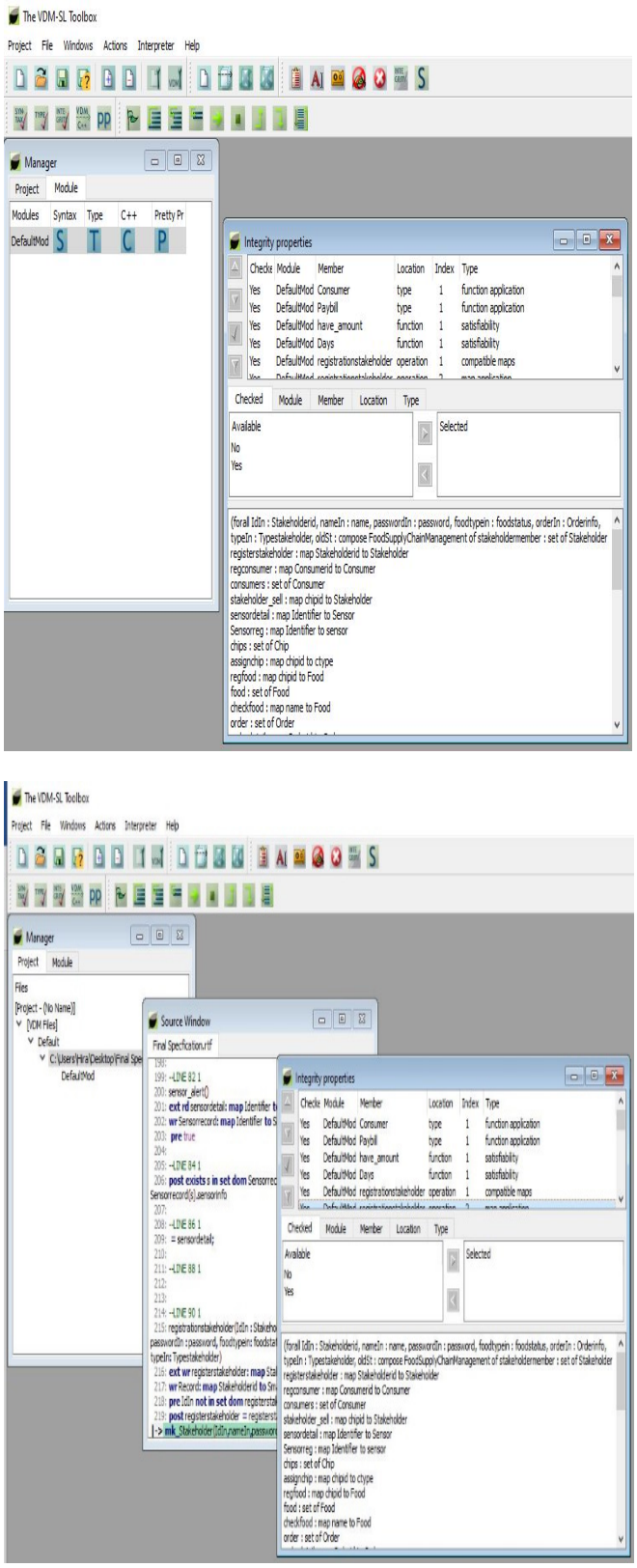

Figure 7. System Correctness Analysis.

6. Conclusions

In this paper, a food supply chain management system is represented and presents different actions performed against different operations. The traditional supply chain is

centralized and not secure. Each entity in the supply chain needs a secure system for transactions. To solve this issue, we use the blockchain that stores all transaction details and all information of the stakeholder. Only registered user can access the stored information in the blockchain. A public blockchain is used to overcome the food supply chain management security issue; i.e., the registration of the stakeholder, payment detection, refund record, food record and sensor alert are stored as transactions in the blockchain. In the supply chain management, we define the different stakeholder, consumer and manager data that are stored in the blockchain. The aim of this system to monitor food conditions and inform authorized users about the food supply chain management system. The food supply chain management is monitored by using a sensor, and these sensors sense the data and send this information to the manager. The manager sends this information to the controller, and the controller takes the actions to control the abnormal condition. To make the system more efficient, different sensors are used. For the system's security, blockchain technology is used in food supply chain management. Food issues increase every day, and a lot of people fall ill by eating these foods. Blockchain and IoT technology are used to make the system more reliable, and IoT is used to monitor the whole process of the food supply chain management system and connect every entity to each other. To understand the system functionality, sequence diagrams are used in the system model. The whole functionality of the proposed system and behavior is presented by the sequence diagram. In this paper, the union and concatenation of automata are used to describe the complete functionality of the system and automate the system. Smart contracts are used to describe the relationship between different entities such as producers, transporters and consumers. Different smart contract algorithms are written to automate the payment process and ensure the security. Only authorized users can access the smart contract. Every authorized user has an identifier and unique address. In supply chain management, a lot of work has been completed on simulation, but simulation does not provide validation and verification. Formal methods are used for validation and verification. In the formal method, the VDM-SL toolbox is used to show the system's correctness and verification. In future work, we will apply the simulation on the food supply chain. We will also apply the model checking on all the components of blockchain-based food supply chain management and verify the system properties by using the model checking.

Author Contributions: Conceptualization, H.H., N.A.Z., E.H.A., M.H.; methodology, H.H., N.A.Z., E.H.A., M.H.; software, H.H., N.A.Z., E.H.A.; validation, H.H., N.A.Z., E.H.A., M.H.; formal analysis, H.H., N.A.Z., E.H.A., M.H.; investigation, H.H., N.A.Z., E.H.A., M.H.; resources, H.H., N.A.Z., E.H.A., M.H.; data curation, H.H., N.A.Z., E.H.A., M.H.; writing—original draft preparation, H.H., N.A.Z., E.H.A.; writing—review and editing, H.H., N.A.Z., E.H.A., M.H.; visualization, H.H., N.A.Z., E.H.A., M.H.; supervision, H.H., N.A.Z., E.H.A.; funding acquisition, M.H. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the Princess Nourah bint Abdulrahman University Researchers Supporting Project (number PNURSP2022R193), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Hereby, we consciously assure that for the manuscript "Blockchain-Based Formal Model for Food Supply Chain Management System Using VDM-SL" the following is fulfilled: 1. This material is the authors' original work, which has not been previously published elsewhere. 2. The paper is not currently being considered for publication elsewhere. 3. The paper reflects the authors' research and analysis truthfully and completely. 4. The paper properly credits the meaningful contributions of co-authors and co-researchers. 5. The results are appropriately placed in the context of prior and existing research. 6. All sources used are properly disclosed (correct citation). 7. All authors have been personally and actively involved in substantial work leading to the paper and will take public responsibility for its content. We agree with the above statements and declare that this submission follows the policies of Swarm Intelligence as outlined in the Guide for Authors and in the Ethical Statement.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Dwivedi, K.S.; Amin, R.; Vollala, S. Blockchain based secured information sharing protocol in supply chain management system with key distribution mechanism. *J. Inf. Secur. Appl.* **2020**, *54*, 102554. [CrossRef]
2. Shahid, A.; Almogren, A.; Javaid, N.; Al-Zahrani, F.A.; Zuair, M.; Alam, M. Blockchain-Based Agri-Food Supply Chain: A Complete Solution. *IEEE Access* **2020**, *8*, 69230–69243. [CrossRef]
3. Lin, J.; Shen, Z.; Zhang, A.; Chai, Y. Blockchain and IoT based food traceability for smart agriculture. In Proceedings of the 3rd International Conference on Crowd Science and Engineering, New York, NY, USA, 28–31 July 2018.
4. Hasan, H.R.; Salah, K. Proof of Delivery of Digital Assets Using Blockchain and Smart Contracts. *IEEE Access* **2018**, *6*, 65439–65448. [CrossRef]
5. Zhao, G.; Liu, S.; Lopez, C.; Lu, H.; Elgueta, S.; Chen, H.; Boshkoska, B.M. Blockchain technology in agri-food value chain management: A synthesis of applications, challenges and future research directions. *Comput. Ind.* **2019**, *109*, 83–99. [CrossRef]
6. Ferdousi, T.; Gruenbacher, D.; Scoglio, C.M. A Permissioned Distributed Ledger for the US Beef Cattle Supply Chain. *IEEE Access* **2020**, *8*, 154833–154847. [CrossRef]
7. Juma, H.; Shaalan, K.; Kamel, I. A Survey on Using Blockchain in Trade Supply Chain Solutions. *IEEE Access* **2019**, *7*, 184115–184132. [CrossRef]
8. Mondal, S.; Wijewardena, K.P.; Karuppuswami, S.; Kriti, N.; Kumar, D.; Chahal, P. Blockchain Inspired RFID-Based Information Architecture for Food Supply Chain. *IEEE Internet Things J.* **2019**, *6*, 5803–5813. [CrossRef]
9. Gayialis, S.P.; Kechagias, E.P.; Papadopoulos, G.A.; Panayiotou, N.A. A Business Process Reference Model for the Development of a Wine Traceability System. *Sustainability* **2022**, *14*, 11687. [CrossRef]
10. Urbano, O.; Perles, A.; Pedraza, C.; Rubio-Arraez, S.; Castelló, M.L.; Ortola, M.D.; Mercado, R. Cost-Effective Implementation of a Temperature Traceability System Based on Smart RFID Tags and IoT Services. *Sensors* **2020**, *20*, 1163. [CrossRef]
11. Chen, S.; Brahma, S.; Mackay, J.; Cao, C.; Aliakbarian, B. The role of smart packaging system in food supply chain. *J. Food Sci.* **2020**, *85*, 517–525. [CrossRef]
12. Wang, S.; Tang, X.; Zhang, Y.; Chen, J. Auditable Protocols for Fair Payment and Physical Asset Delivery Based on Smart Contracts. *IEEE Access* **2019**, *7*, 109439–109453. [CrossRef]
13. Shahid, A.; Sarfraz, U.; Malik, M.W.; Iftikhar, M.S.; Jamal, A.; Javaid, N. Blockchain-Based Reputation System in Agri-Food Supply Chain. In *International Conference on Advanced Information Networking and Applications*; Springer: Cham, Switzerland, 2020.
14. Liu, P.; Long, Y.; Song, H.-C.; He, Y.-D. Investment decision and coordination of green agri-food supply chain considering information service based on blockchain and big data. *J. Clean. Prod.* **2020**, *277*, 123646. [CrossRef]
15. Feng, H.; Wang, X.; Duan, Y.; Zhang, J.; Zhang, X. Applying blockchain technology to improve agri-food traceability: A review of development methods, benefits and challenges. *J. Clean. Prod.* **2020**, *260*, 121031. [CrossRef]
16. Zhang, X.; Sun, P.; Xu, J.; Wang, X.; Yu, J.; Zhao, Z.; Dong, Y. Blockchain-Based Safety Management System for the Grain Supply Chain. *IEEE Access* **2020**, *8*, 36398–36410. [CrossRef]
17. Bumblauskas, D.; Mann, A.; Dugan, B.; Rittmer, J. A blockchain use case in food distribution: Do you know where your food has been? *Int. J. Inf. Manag.* **2020**, *52*, 102008. [CrossRef]
18. Lin, W.; Huang, X.; Fang, H.; Wang, V.; Hua, Y.; Wang, J.; Yin, H.; Yi, D.; Yau, L. Blockchain Technology in Current Agricultural Systems: From Techniques to Applications. *IEEE Access* **2020**, *8*, 143920–143937. [CrossRef]
19. Kamble, S.S.; Gunasekaran, A.; Sharma, R. Modeling the blockchain enabled traceability in agriculture supply chain. *Int. J. Inf. Manag.* **2019**, *52*, 101967. [CrossRef]
20. Chang, S.E.; Chen, Y.-C.; Lu, M.-F. Supply chain re-engineering using blockchain technology: A case of smart contract based tracking process. *Technol. Forecast. Soc. Chang.* **2019**, *144*, 1–11. [CrossRef]
21. Casino, F.; Kanakaris, V.; Dasaklis, T.K.; Moschuris, S.; Rachaniotis, N.P. Modeling food supply chain traceability based on blockchain technology. *IFAC-PapersOnLine* **2019**, *52*, 2728–2733. [CrossRef]
22. George, R.V.; Harsh, H.O.; Ray, P.; Babu, A.K. Food quality traceability prototype for restaurants using blockchain and food quality data index. *J. Clean. Prod.* **2019**, *240*, 118021. [CrossRef]
23. Wang, S.; Li, D.; Zhang, Y.; Chen, J. Smart Contract-Based Product Traceability System in the Supply Chain Scenario. *IEEE Access* **2019**, *7*, 115122–115133. [CrossRef]
24. Lin, Q.; Wang, H.; Pei, X.; Wang, J. Food Safety Traceability System Based on Blockchain and EPCIS. *IEEE Access* **2019**, *7*, 20698–20707. [CrossRef]
25. Salah, K.; Nizamuddin, N.; Jayaraman, R.; Omar, M. Blockchain-Based Soybean Traceability in Agricultural Supply Chain. *IEEE Access* **2019**, *7*, 73295–73305. [CrossRef]
26. Tian, F. An agri-food supply chain traceability system for China based on RFID & blockchain technology. In Proceedings of the 2016 13th International Conference on Service Systems and Service Management (ICSSSM), Kunming, China, 24–26 June 2016.
27. Gandino, F.; Montrucchio, B.; Rebaudengo, M.; Sanchez, E.R. On Improving Automation by Integrating RFID in the Traceability Management of the Agri-Food Sector. *IEEE Trans. Ind. Electron.* **2009**, *56*, 2357–2365. [CrossRef]

28. Tian, F. A supply chain traceability system for food safety based on HACCP, blockchain & Internet of things. In Proceedings of the 2017 International Conference on Service Systems and Service Management, Dalian, China, 16–18 June 2017.
29. Behnke, K.; Janssen, M. Boundary conditions for traceability in food supply chains using blockchain technology. *Int. J. Inf. Manag.* **2019**, *52*, 101969. [CrossRef]
30. Tsang, Y.P.; Choy, K.L.; Wu, C.H.; Ho, G.T.S.; Lam, H.Y. Blockchain-Driven IoT for Food Traceability with an Integrated Consensus Mechanism. *IEEE Access* **2019**, *7*, 129000–129017. [CrossRef]
31. Nasurudeen Ahamed, N.; Karthikeyan, P.; Anandaraj, S.P.; Vignesh, R. Sea Food Supply Chain Management Using Blockchain. In Proceedings of the 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 6–7 March 2020.
32. Madumidha, S.; Ranjani, P.S.; Varsinee, S.S.; Sundari, P.S. Transparency and traceability: In food supply chain system using blockchain technology with Internet of Things. In Proceedings of the 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, India, 23–25 April 2019.
33. Malik, S.; Dedeoglu, V.; Kanhere, S.S.; Jurdak, R. TrustChain: Trust management in blockchain and IoT supported supply chains. In Proceedings of the 2019 IEEE International Conference on Blockchain (Blockchain), Atlanta, GA, USA, 14–17 July 2019.
34. Caro, M.P.; Ali, M.S.; Vecchio, M.; Giaffreda, R. Blockchain-based traceability in Agri-Food supply chain management: A practical implementation. In Proceedings of the 2018 IoT Vertical and Topical Summit on Agriculture-Tuscany (IOT Tuscany), Tuscany, Italy, 8–9 May 2018.
35. Pan, C.-T.; Lee, M.-J.; Huang, N.-F.; Luo, J.-T.; Shao, J.-J. Agriculture Blockchain Service Platform for Farm-to-Fork Traceability with IoT Sensors. In Proceedings of the 2020 International Conference on Information Networking (ICOIN), Barcelona, Spain, 7–10 January 2020.
36. Kulkarni, A.; Hazari, N.A.; Niamat, M. A Blockchain Technology Approach for the Security and Trust of the IC Supply Chain. In Proceedings of the 2019 IEEE National Aerospace and Electronics Conference (NAECON), Dayton, OH, USA, 15–19 July 2019.
37. Tse, D.; Zhang, B.; Yang, Y.; Cheng, C.; Mu, H. Blockchain application in food supply information security. In Proceedings of the 2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), Singapore, 10–13 December 2017.
38. Kamilaris, A.; Fonts, A.; Prenafeta-Boldó, F.X. The rise of blockchain technology in agriculture and food supply chains. *Trends Food Sci. Technol.* **2019**, *91*, 640–652. [CrossRef]
39. Saurabh, S.; Dey, K. Blockchain technology adoption, architecture, and sustainable agri-food supply chains. *J. Clean. Prod.* **2020**, *284*, 124731. [CrossRef]
40. Torky, M.; Hassanein, A.E. Integrating blockchain and the internet of things in precision agriculture: Analysis, opportunities, and challenges. *Comput. Electron. Agric.* **2020**, *178*, 105476. [CrossRef]
41. Ben-Daya, M.; Hassini, E.; Bahroun, Z.; Banimfreg, B.H. The role of internet of things in food supply chain quality management: A review. *Qual. Manag. J.* **2020**, *28*, 17–40. [CrossRef]
42. Wang, L.; Xu, L.; Zheng, Z.; Liu, S.; Li, X.; Cao, L.; Li, J.; Sun, C. Smart Contract-Based Agricultural Food Supply Chain Traceability. *IEEE Access* **2021**, *9*, 9296–9307. [CrossRef]
43. Grecuccio, J.; Giusto, E.; Fiori, F.; Rebaudengo, M. Combining Blockchain and IoT: Food-Chain Traceability and Beyond. *Energies* **2020**, *13*, 3820. [CrossRef]
44. Van Der Vorst, J.G.; Tromp, S.-O.; van der Zee, D.-J. Simulation modelling for food supply chain redesign; integrated decision making on product quality, sustainability and logistics. *Int. J. Prod. Res.* **2009**, *47*, 6611–6631. [CrossRef]
45. Dabbene, F.; Gay, P.; Tortia, C. Traceability issues in food supply chain management: A review. *Biosyst. Eng.* **2014**, *120*, 65–80. [CrossRef]
46. Li, J.; Maiti, A.; Springer, M.; Gray, T. Blockchain for supply chain quality management: Challenges and opportunities in context of open manufacturing and industrial internet of things. *Int. J. Comput. Integr. Manuf.* **2020**, *33*, 1321–1355. [CrossRef]
47. Dujak, D.; Sajter, D. Blockchain Applications in Supply Chain. In *SMART Supply Network*; Kawa, A., Maryniak, A., Eds.; Springer: Berlin/Heidelberg, Germany, 2019; pp. 21–46.
48. Saddiq, S.; Zafar, N.A.; Ullah, F. Formal modeling of smart logistics monitoring. In Proceedings of the 2017 1st International Conference on Electronics, Materials Engineering and Nano-Technology (IEMENTech), Kolkata, India, 28–29 April 2017.
49. Afzaal, M.F.; Rehman, A.; Latif, S.; Zafar, N.A. Blockchain based Automated Formal Model for Safety and Security in Smart Parking System. In Proceedings of the 2019 4th International Conference on Emerging Trends in Engineering, Sciences and Technology (ICEEST), Karachi, Pakistan, 10–11 December 2019.
50. Rehman, A.; Latif, S.; Zafar, N.A. Non-deterministic formal modeling of registration system towards smart campus. In Proceedings of the 2018 12th International Conference on Mathematics, Actuarial Science, Computer Science and Statistics (MACS), Karachi, Pakistan, 24–25 November 2018.