IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received March 30, 2022, accepted April 18, 2022, date of publication April 28, 2022, date of current version May 13, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3171045

Automatic Generation of Ethereum-Based Smart Contracts for Agri-Food Traceability System

LODOVICA MARCHESI, (Member, IEEE), KATIUSCIA MANNARO, MICHELE MARCHESI, (Senior Member, IEEE), AND ROBERTO TONELLI, (Member, IEEE)
Department of Mathematics and Computer Science, University of Cagliari, 09124 Cagliari, Italy

Corresponding authors: Lodovica Marchesi (lodovica.marchesi@unica.it) and Katiuscia Mannaro (katiuscia.mannaro@unica.it)

This work was supported in part by the Application of Blockchain to Authenticity 1370 and Traceability of Aliments (ABATA) Project, carried out together with NetService Inc.; and in part by the Ministero dello Sviluppo Economico, National Operational Program ‘‘Enterprises and Competitiveness,’’ under Project F/200130/01-02/X45 and Project CUP: B31B19000650008.

ABSTRACT There is a growing demand for transparency along the agri-food chain, both from customers and governments. The adoption of blockchain technology to enable secure traceability for the management of the agri-food chain, provide information such as the provenance of a food product and prevent food fraud, is emerging rapidly, due to the inherent trust and inalterability provided by this technology. However, developing the right smart contracts for these use cases is even more of a challenge than it is for those used in other fields. Several management systems for the agri-food chain based on blockchain technology and smart contract have been proposed, all however ad-hoc for a specific product or production process and difficult to generalize. In this paper, we propose a new approach to easily customize and compose general Ethereum-based smart contracts designed for the agri-food industrial domain, to be able to reuse the code and modules and automate the process to shorten development times, while keeping it safe and reliable. Starting from the definition of the real production process, we aim to automatically generate both the smart contracts to manage the system and the user interfaces to interact with them, thus producing a system that works semi-automatically. Additionally, we describe a honey production case study to show how our approach works. Future work will first extend the scope of the approach to other supply chains, furthermore, while the current platform used is Ethereum, in the future our approach will be easily extended to other blockchain platforms.

INDEX TERMS Agri-food product traceability, blockchain, smart contract, supply chain.

I. INTRODUCTION

Blockchain technology is a new distributed, decentralized and immutable ledger database that can assure immutability and integrity of data without the need of a third trusted party. This is one of the reasons for which strong expectations exist on this technology to solve problems in sectors in which several untrusted actors have to work together, such as in the case of the agri-food industry. Blockchain technology appeared for the first time in 2008 when one or more developers under the pseudonym Satoshi Nakamoto published a paper on a P2P electronic cash system [1] based on a digital currency called Bitcoin. This currency is based on a blockchain and does not need any intermediaries or central authority to transfer money from one person to another person. A blockchain is a specific type of distributed database able to store data in a secure and immutable way, and simultaneously to create transparency of the data history. It is based on a technological protocol that enables data to be exchanged with third parties within the P2P network without the need for intermediaries, because participants interact anonymously with encrypted identities, through transactions. Each transaction must be validated by a community of users through a consensus process, and then recorded in the ledger by adding it to an immutable chain of blocks holding the transactions stored in every network node. Many companies and startups are already adopting, and working on blockchain technology, trying to exploit the many advantages it promises, so we are experiencing a strong growth of ideas and applications.

Several research papers, like [2], [3], [4], [5], [6], just to cite a few, have shown that the use of the blockchain can advantageously help to achieve traceability, by storing data that are non-forgeable, and with a certain date. Consequently, companies are trying to adopt this technology in

The associate editor coordinating the review of this manuscript and approving it for publication was Mauro Gaggero.

various sectors by harnessing in particular its ability to get transparency in scenarios where numerous untrusted actors are involved.

According to Galvez *et al.* [7] today more than ever, customers are demanding transparency, especially with food. People want to feel secure and to know how a product was farmed or manufactured, and which ingredients are involved in its production.

In Europe, food legislation is particularly strict and the implementation of traceability systems are mandatory, but they are unable to fully guarantee consumers against fraud [8]. European Council formally adopted, on 13 June 2019, a new Regulation on the transparency and sustainability of the EU risk assessment in the food chain. According to European Commission and the European Food Safety Authority (EFSA) - a public agency that provides scientific advice and technical support for EU legislation - this new Regulation mainly revises the General Food Law Regulation that provides some procedures relating to food safety. It aims at increasing the transparency of the EU risk assessment in the food chain, maintaining a high level of safety [9]. The food labeling system cannot guarantee food safety and high quality. Therefore, traceability is a tool to assure food safety and quality as well as to achieve consumer confidence [10]. In our opinion, innovative methods for traceability systems based on product identification are needed in order to ensure a high level of food safety and fully guarantee consumers against fraud.

For these reasons, blockchain is gaining increasing popularity as a technology to enable traceability in a certified and immutable way in the agri-food sector, helping to avoid fraud and counterfeiting by creating an auditable record of the journey from the farm to the fork behind all physical components of the food products.

Some companies have launched pilot or proof of concept projects to implement blockchain technology in a wide range of sectors, but at present many limitations still have to be considered and addressed. Most of the published works concerning the application of blockchain technologies, for example in supply chain management, reported no detailed information about the technical implementation, and there are still few practical uses of blockchain technology.

Research on blockchain in many different application areas is going through an exploratory phase, and supply chain management is one of the main areas of interest to be studied in terms of the benefits that would be obtained on the traceability system, such as facilitating food safety and fraud prevention.

Clearly, there is a need for structured methods to facilitate and make more efficient the development of applications based on blockchain technology for the management of the traceability of the agri-food supply chain.

In our previous research works [11], [12] we explored how the Internet of Things can be combined with blockchain technologies to address potential issues in the agri-food industry, and we combined Internet of things (IoT) technology - in particular radio frequency identification (RFID) sensors and near field communication (NFC) tags - with blockchain and interplanetary file system (IPFS) technology, to guarantee transparent and auditable traceability of the goods from farm to fork, providing data that demonstrate the quality of all intermediate products. We believe that, also in the light of our experience, researchers and developers could benefit from a general-purpose approach for agri-food supply-chain management.

The goal of this work is to facilitate and make more efficient the development of blockchain applications for agri-food supply chain management, by using configurable blocks to be assembled together, so as not to start from scratch every time.

Starting from the definition of the real production process, we aim to automatically generate both the smart contracts to manage the system and the user interfaces to interact with them, thus producing a working system in a semi-automated way.

This paper is an extended version of the earlier one [13] in which we proposed configurable and modular building blocks for agri-food supply chain management systems based on a blockchain, as the first attempt to develop a semi-automatic configurable system. With respect to the previous paper, this one increases and improves the presentation of the concepts useful to represent an agri-food supply chain. It also provides a section about how the SCs are actually implemented and presents a different case study. In the previous paper, the presented case study was olive oil production, described at a very concise level. In this paper, we present honey production as a case study, described in much greater detail and explained with a step-by-step approach.

To summarize, we propose a novel approach for customizing and composing general Ethereum-based smart contracts (SCs) designed for the agri-food industrial domain in a simple way to be able to reuse the code and modules and automate the process to shorten the time of development, keeping its secure and trusted. As far as we know, this is the first attempt to develop a semi-automatic configurable system that supports the entire class of supply chains for the agri-food industrial domain.

Though the approach is targeted to the agri-food domain, it can be easily extended to many other types of supply chains, where a product, a service, or a shipment is delivered by assembling and working on parts, and/or passes through different types of transformations and state changes.

The main contributions of our work are:

- The study and development of an increased and improved general representation of food production specifically targeted to field traceability systems using blockchain technology;
- The development of a set of modules, both general SCs and UI applications, able to be easily configured to generate a system for tracking real agri-food supply chains; the SCs were also checked for security and gas-saving;
- The development of a structured way, starting from the definition of the food production process through

predefined tables, to configure these modules and to easily generate the final system also by developers with only limited knowledge of blockchain technology;

- The development of a novel case study (honey production) to show how the approach works.

The remainder of the paper is organized as follows. Section II deals with the background and related work. In Section III, we describe the proposed methodology by identifying entities and events of food production. In Section IV we present the design of the general software modules supporting these entities and events. Section V presents the case study of a traceability system for honey production, explaining how our approach works. This goal has been evaluated through a case study in which the approach is demonstrated. Finally, Section VI discusses and concludes the paper.

II. BACKGROUND AND RELATED WORK

In this section, we first introduce in subsection II-A an overview of blockchain technology and SCs and we present the state-of-the-art of analysis and design techniques for developing blockchain-based systems. Then in subsection II-B we recall the main studies that highlight the benefits of the use of blockchain systems in the agri-food supply chain domain.

A. DESIGN METHODS FOR SMART CONTRACTS DEVELOPMENT: RELATED RESEARCH

In the last years, smart contracts have increasingly gained ground in ICT applications, but smart contract development still remains a challenging task to many developers. This is largely due to its special design challenges and to the differences between SC development and traditional software development. The term ''smart contract'' is used in different ways, also based on the field in which these are deployed. Since there is no clear consensus on this terminology, Clack *et al.* [14] adopted a higher-level definition based on the two topics of automation and enforceability. They defined a smart contract as *" an automatable and enforceable agreement. Automatable by computer, although some parts may require human input and control. Enforceable either by legal enforcement of rights and obligations or via tamper-proof execution of computer code"*.

In this paper, we assume a common and quite widespread definition of SCs, intended as code scripts typically run on distributed ledgers that are designed to automatically execute specific tasks when predefined conditions are met. Therefore, a SC is a program that contains executable code and data. SCs are deployed on blockchain nodes, and through them, a decentralized application can operate without human intervention.

A decentralized application (DApp, Dapp, dapp, or more frequently dApp) is a computer application that runs on a distributed peer-to-peer (P2P) system, that is on a network of nodes, with no node acting as supervisor. A dApp is stored and executed on a blockchain to be decentralized, transparent, deterministic, and redundant. It is developed by writing smart contracts (SCs), which are small script programs running on every node of the blockchain, and may have a user interface (UI) that allows users and devices to interact with SCs. SCs are immutable - no one can tamper with the code of the contract - and distributed because of their storage inside the blockchain. In the last decade, there has been a huge development in the field of blockchain technology applied to various economic sectors, due to the advantages that the implementation of such a system could provide.

Alharby and Van Moorsel [15] found four issues that might face developers when writing smart contracts: i) the difficulty of writing correct contracts; ii) the inability to modify or terminate contracts; iii) the lack of support to identify under-optimized contracts, and finally; iv) the complexity of SC programming languages.

Subsequently, Zou *et al.* [16] conducted an empirical study to explore the potential challenges faced by developers during SC development, with a focus on Ethereum blockchain. The survey results revealed several major challenges. In particular, existing tools for SC development are still very basic. Programming SCs is different than programming in standard programming languages because the blockchain and the code residing there cannot be changed after it has been deployed.

Currently, there are different blockchain platforms (e.g., Bitcoin, Corda, Ethereum, Hyperledger Fabric, Tendermint, etc.) but not all of them support SCs. Moreover, each platform offers distinctive features, and there is no standard way to write SCs. For instance, Corda is an open-source permissioned blockchain platform – meaning a blockchain managed by a consortium of organizations, which run the validator nodes and perform the consensus mechanism – that allows transacting directly with SCs, and it is explicitly designed to account for a highly regulated environment, e.g. the financial service industry.

Ethereum[1] is one of the leading blockchain platforms, released in July 2015, that also has its own cryptocurrency, called Ether, or ETH. Ethers can be used to pay for the computations performed by participants who mine blocks. Ethereum provides a decentralized Turing-complete machine - the Ethereum virtual machine (EVM) - to execute scripts, that is SCs. It supports advanced programming languages (e.g., Solidity, Vyper, YUL) for writing SCs and dApps. The public Ethereum blockchain uses a proof-of-work consensus but is transitioning to a proof-of-stake one.

All SCs are stored on every node of the blockchain. Ethereum uses a ''gas'' mechanism, which is an internal pricing mechanism for all transactions running on it. To execute a transaction, it is necessary to pay a gas fee in Ethers, whose amount depends on the network overload. This gas mechanism is needed to avoid long computations and loops and can improve the allocation of resources and mitigate spam.

[1]https://ethereum.org

In this paper, we focus on the difficulty of writing correct SCs, that are contracts functioning as intended by their developers. If a SC does not execute as intended, some or the whole currency managed by it would disappear, or other unintended effects might be triggered by an attacker.

In an attempt to tackle this issue,Alharby and Van Moorsel [15] identified three solutions: i) to semi-automate the creation of SCs; ii) to provide developers with guidelines; iii) adoption of formal verification techniques. In our opinion, a solution to ease the process of writing SCs is to semi-automate the creation of smart contracts.

Rocha and Ducasse [17] presented a general proposal for extending existing software modeling notations to include specific blockchain concepts or integrations. According to these authors, modeling is an important part of designing software and in their preliminary work, they start the discussion on specialized modeling notations for dApps. The authors show three complementary modeling notations based on well-known software engineering models: entity-relationship model (ERM), unified modeling language (UML), and business process model and notation (BPMN). Then they apply them to an example of blockchain-oriented software (BOS) that implements part of the business logic in the blockchain by using SCs.

In the literature, there are a few proposals for the standardization of software engineering of blockchain technologies with reference also to automatic generation of SCs from formal models.

According to According to Jurgelaitis *et al.* [18], Blockchain technology and SC development lack clarity in their implementation. They propose a method based on Model-Driven Architecture, which could be used for describing blockchain-based systems in a more general language to determine whether it is possible to model blockchain structure and SC logic, and which business logic should be conveyed in SCs, and which should stay off-chain.

In [19] a model-driven engineering (MDE) tool called Lorikeet for the implementation of business processes on blockchain to manage assets was presented. Model-driven engineering is a software engineering methodology that automatically creates software system code from formal models, and helps developers to manage software complexity by only focusing on building high-level models. Lorikeet can automatically create well-tested SC code from specifications that are encoded in the business process data schema.

In a subsequent work, in [20] the authors presented a model-driven blockchain application development approach for business processes and asset management. They provide templates for customizing data schemata for both fungible and non-fungible assets registries. Moreover, they propose SC generation methods to automatically transform models into SC programming language code, namely into Solidity. The generated SCs consist of SCs for business process execution, and SCs managing standard ERC-20/ERC-721 compliant tokens. The proposed approach is implemented using their smart contract generation tool: Lorikeet.

In the field of model-driven methods, Udokwu and Norta [21] presented a goal-modeling method to systematically describe the requirements of inter-organizational collaboration systems based on blockchain technology. In a related paper, Udokwu *et al.* [22] presented and evaluated the decentralized agent-oriented modeling (DAOM) framework for building dApps that support inter-organizational collaborations. The evaluation, made by 15 industry experts, showed the usefulness and applicability of DAOM for this type of dApps, and the effectiveness of the related tool support.

Frantz and Nowostawski [23] proposed a SC development approach that supports the semi-automated translation of human-readable contract representations in terms of ADICO statements – different components that include attributes, deontic, aim, conditions, or else (consequences associated with non-conformance) – to enable the codification of laws into verifiable and enforceable computational structures in the public blockchain.

de Sousa *et al.* [24] proposed B-MERODE, to fulfill the need to develop new methods for the analysis and engineering of Business Processes (BPs) supported by a blockchain. This is a novel approach to generate SCs supporting cross-organizational collaborations, and relying on model-driven engineering and artifact-centric business processes. Finally, they demonstrated its feasibility by developing the case of a rice supply chain through B-MERODE.

Marchesi *et al.* [25] proposed a general scheme for managing BOS development processes: a software development process based on several Agile practices, and a more formal design approach using UML and including Class diagrams, Statecharts, Use Case diagrams, and Sequence diagrams. A practical example of a paradigmatic blockchain smart contract implementing a voting system was provided. This approach was subsequently upgraded and extended in [26].

Mavridou and Laszka [27] introduced a framework for designing smart contracts in terms of finite state machines. They provide a tool with a graphical editor for defining the contract specifications as automata and for translating them into SC code.

All the mentioned proposals for the analysis and design of blockchain and SCs systems are in the early stages and have not yet been extensively validated or tested in practical applications.

B. BLOCKCHAIN TECHNOLOGY IN AGRI-FOOD SUPPLY CHAIN

Blockchain, and more generally the distributed ledger technology (DLT), is a promising technology that is tamper-proof and decentralized. Self-executing and self-verifying SCs can manage transactions between mutually untrusted parties. In this context, scholarly literature on the adoption of blockchain technology in specific traceability systems for the agri-food supply chain is beginning to emerge. In particular, since 2018, a lot of research efforts have been made on the use of blockchain technology for traceability systems [2].

Tripoli and Schmidhuber [28] discussed the potential of distributed systems to transform the agri-food industry. Tribis *et al.* [29] performed a literature review of relevant papers about the adoption of blockchain technology for generic supply chain management, covering the literature until 2018, and found out that most of the papers were focused on the use of blockchain for traceability, but only one out of 40 papers dealt with the agricultural field. Another relevant literature review about the use of IoT technology in agriculture, which is a prerequisite for automated blockchain registrations, was performed by Farooq et at. [30].

Other research papers, namely [4], [31], [32], [33], [34] [35], just to cite a few, presented traceability systems that use blockchain technology and SCs.

According to these researches, this technology guarantees:

- Data integrity and provenance of documents and records on the blockchain;
- Immutability and transparency of data recorded on the blockchain, resulting in the traceability of agri-food products from root to retail;
- Compliance with the quantities of the products involved (grapes, wine, bottles), based on the annual production of the land and the yield in the various stages of processing. This is achieved with the system of tokens, which are associated with the products and cannot be altered, as they are managed on a blockchain;
- Ability to retrace the entire supply chain, simply by accessing the blockchain, and public servers storing relevant documents, starting from the QR code shown on the final product.

Chang *et al.* [36] claim that by 2023 the global blockchain supply chain market will grow to \$ 3,314.6 million, with an increase in the annual growth rate of 87%.

Various papers presented real blockchain solutions for supply chain management, which proved to be successful. Among others, AgriDigital [2] is an Australian system for managing the grain supply chain. It was released in 2017 and at the end of 2020 had more than 7000 users and a transaction value of \$ 3,793 million. Caro *et al.* [37] developed AgriBlockIoT, a fully decentralized, blockchain-based traceability solution for agri-food supply chain management, able to seamlessly integrate IoT devices, using and comparing both Ethereum and Hyperledger Sawtooth blockchains. Tian [4] studied and developed an agri-food supply chain traceability system for China based on RFID (radio frequency identification) and blockchain technology to guarantee food safety. Baralla *et al.* [38] proposed a generic agri-food supply chain traceability system based on blockchain technology implementing the "farm-to-fork" (F2F) model currently used in the European Union, which can integrate current traceability rules and processes, using Hyperledger Sawtooth, and implemented following an agile approach. Wang *et al.* [39] proposed a product traceability system based on blockchain technology, in which all product registration and transfer histories are perpetually recorded by using SCs. An event response mechanism was designed to verify the identities of both parties of the transaction and guarantee the validity of the transaction.

Yu *et al.* [40] proposed a monitoring framework that combines SCs and evaluation models for the automatic evaluation of the quality of fruit juice samples. SCs are executed to record production data on a blockchain, and can decide whether the production process is working correctly, or should be terminated for non-compliance. The feasibility of the system has been evaluated by implementing a prototype version of the quality monitoring system for flat peach juice production based on the Ethereum platform and executed in the Remix IDE.

Many studies in the literature focused on highlighting the benefits and value derived by blockchain implementation in the agri-food supply chain domain. In particular, Kamble *et al.* [35] conducted an analysis to model a traceability system based on blockchain technology in the agriculture supply chain. They identified thirteen enablers that encourage blockchain adoption in the agriculture supply chain, such as anonymity and privacy, immutability, SCs, secured and shared database, traceability, transparency, and others. Then they established hierarchical levels and relationships between the involved actors in the supply chain through interpretive structural modeling (ISM), and decision-making trial and evaluation laboratory (DEMATEL) methodologies. The enablers were identified from the existing literature and validated by experts from the field of agri-based supply chains and technology. Moreover, the authors conducted an interesting literature review revealing that BC technology offers various benefits leading to improvements in the sustainability of agricultural supply chains.

Recently, also Pranto *et al.* [41] demonstrated how the applicability of blockchain and SCs in the field of agriculture can ensure traceability of agricultural products. In their research, they described in detail two SCs and showed a gas cost analysis of the operations. Specifically, a SC called Storage Contract is used in the pre-harvesting period for monitoring the storage condition connected with the system, the second SC is called Distribution Contract and is used in the post-harvesting period. Finally, they analyzed the approach in terms of advantages and disadvantages.

In a similar work, Haque *et al.* [42] proposed a framework for providing complete transparency and unforgeable product information in the oil supply chain. They tried to conceptualize the process of end-to-end product tracking. They identified the role and function of every actor and described the structure of two SCs in terms of attributes, events, modifier, and functions. The first SC, named CheckProgress, aims to monitor the product's information and keep track of it. The second SC, the OilDistribution contract, checks the authenticity of the actors.

In all these research papers, no approach was formalized for the development of SCs. Domain concepts are entirely delegated to the individual work of software developers,

[2]https://www.agridigital.io/

potentially leading to pitfalls well known in the field of software engineering, such as poor maintainability and low levels of reuse.

To the best of our knowledge, and by comparing our work to others that deal with the use of dApps to certify the origin of food and prevent food fraud, we assert that this is the first work that proposes and formalizes a general approach to develop dApps to track agri-food supply chains, useful for most types of food production.

III. THE CONCEPTUAL APPROACH AND THE PROBLEM DOMAIN

In this section, we describe the conceptual approach we followed to facilitate the generation of smart contracts for the traceability system of agricultural food supply chains. This approach is guided by the design science methodology proposed by Hevner [43]. We applied the seven design-science research guidelines (DSRG) derived from the assumption that ''*knowledge and understanding of a design problem and its solution are acquired in the building and application of an artifact.*''.

The relatively young field of SC development is made problematic by their current lack of formalization, so it is necessary to use a sound methodology to be assured that the research is performed following widely accepted guidelines.

In particular, our research produced a viable artifact, in terms of agri-food problem domain modeling, a systematic process to describe real supply chains, a software prototype able to implement and demonstrate the validity of the approach. The business problem tackled is certainly important and relevant. These facts cover guidelines 1 and 2 of DSRG.

The contributions of our work are relevant, as described in the Introduction. The analysis of the agri-food process domain has been performed using sound software engineering techniques and the design and development of the software prototype have been performed using ABCDE method. We evaluated alternatives, both in the problem domain analysis and in the data representation and software domain, resulting in effective artifacts. Our results, as presented in this work, can be understood by a technology-oriented audience. The organizational and business aspects, suitably presented, could be effectively communicated also to a management-oriented audience. These facts cover guidelines 4-7 of DSRG.

Guideline 3 on design evaluation is satisfied by the presentation and evaluation of our approach to some agri-food producers of honey, olive oil, wine, and pecorino cheese. This evaluation was not made in a systematic way but through meetings and presentations different for each producer. Nevertheless, the reception of the project results was good, and we were encouraged to continue in the direction outlined.

Our entire approach was carried on taking advantage of ABCDE (Agile BlockChain Dapp engineering) method to design and implement dApps [26]. The approach aims to:

- Document in a transparent and immutable way all relevant events relevant to production;
- Allow authorities, laboratories, and certified experts to asseverate the production, giving proof of their identity and their certifications;
- Integrate manual registrations and automatic registrations made by Internet of Things (IoT) devices, which are increasingly widespread;
- Keep track of the quantities produced so that these cannot be increased by introducing products of non-certified origin;
- Give evidence of all stages of production to the authorities responsible for verifying the specifications;
- Allow retailers and end consumers to learn about the history of the products purchased, from the field to the purchased product, using an app.

A. THE PROBLEM DOMAIN IN AGRI-FOOD SUPPLY-CHAIN

Before building a software system, software engineers need to capture the knowledge of the problem domain, that is all the information that defines the problem the software system aims to solve. The first step is to get a deep and consistent understanding of the area under analysis.

Most agri-food supply chain systems get their primary inputs from one or more primary sources (soil, herd, beehives, lake, sea, and so on.), then a primitive resource is produced (harvest, milk, raw honeycombs, fishes, etc.), this product is transformed, possibly several times, until the final product is packed and delivered to customers. Events relevant to the process can occur in each of these phases. Most of the processes in the agri-food industry manage ''batches'', where a batch means a specific quantity of product that is intended to have uniform character and quality.

In general, agri-food products are identified through a batch management system, both in the case in which the product is marketed without undergoing significant transformations and when it is processed to obtain output products that are significantly different from input ones. The reliability of a production batch can be guaranteed through an efficient and transparent system of product and process traceability.

Certifying the origin, ingredients, and processing methods to guarantee a high-quality standard of an agri-food product is a problem too complex to be tackled top-down, therefore we propose a bottom-up approach to correctly identify incoming, processing, and outgoing goods.

By using analysis techniques and object-oriented design, we performed the analysis of the agri-food industrial domain from the software engineering viewpoint, to find common objects and objects linked to specific processes. In particular, this phase aims to:

- Investigate and define the roles of the major actors involved in the system;
- Determine the entities that emerge and recur in this type of system;
- Decide how to be able to represent a general-purpose system;

- Determine the relationships between these entities and the events of interest for the traceability system (events that need to be made permanent for the traceability).

1) IDENTIFYING THE ACTORS

The first step in use case analysis is to identify the major actors. The agri-food supply chain domain is primarily characterized by autonomous and independent actors that in recent years are increasingly interacting with globally interconnected systems. We consider the agri-food supply chain as a sequence of processes from production to the final product, that involves directly or indirectly individuals or groups of actors with different roles, at various levels and steps of the production process. For instance, there are the producer, its suppliers, people who make up the workforce, retailers, and consumers themselves. Not all processes involve all of these actors, but they are found in most agri-food processes. Note also that if an actor is an organization, like the producer firm or the analysis lab, the actions related to this actor are performed by one or more human people who have the right to represent the organization.

In our model, each actor is identified by a unique address and can send transactions to the blockchain from this address. The actor owns the private key associated with the address, thus being the only person able to send messages from that address. For human actors, a mechanism that associates a human identity with the address is also needed, which is performed by the Address Catalog described in section III-A2.

The key actors of a typical agri-food production system are:

- **Administrator/Owner**: administers the software system by managing and controlling the reading and writing access to the system by other actors and their permissions. This is a role present in most business process management systems.
- **Farm**: produces the raw materials that are the inputs of the supply chain. It is able to generate tokens associated with the produced materials. It can manage the information certified in the system.
- **Supplier**: supplies materials, services, or devices needed for producing the target goods. It is able to generate tokens associated with the produced materials. It can manage the information certified in the system.
- **Transformer**: transforms the raw material, or already transformed material, into intermediate goods or into the final good. It is able to take ownership of tokens associated with the produced materials. A transformer can work with multiple input materials or goods, and produce multiple goods. It can manage the information certified in the system.
- **Wholesaler**: buys the target good in large quantities and sells it to other wholesalers or to retailers. It is able to take ownership of tokens associated with target goods. It could manage information certified in the system.
- **Retailer**: buys the target good to sell it to End Customers. It is able to take ownership of tokens associated with the target good. It could manage information certified in the system.
- **End Customer**: buys the target good from Retailers. S/he is usually not provided with an address.
- **Certification Authority**: Public or private authority in charge of controlling and certifying a given production. It can certify the goodness of amounts and documents, or to directly produce certificates. Usually, it manages the information certified in the system. For instance, the Regional Authority, or Protection Consortium that performs inspections to verify the conformity of the products and the work of each actor of the supply chain. The inspections can be performed by viewing the data documents stored by the nodes of the blockchain.
- **Professional**: is a person with a given degree and experience, qualified to certify the goodness of amounts and documents. The list of professionals qualified in a given field is usually maintained by a Certification Authority.
- **Analysis Lab**: is a laboratory able to perform physical, chemical, and/or biological analysis of given materials, products, or goods. The list of Labs qualified in a given field is usually maintained by a Certification Authority. It can manage the information certified in the system.
- **Warehouse**: receives, stores, and sends goods. It is able to take ownership of the tokens associated with the target good, or simply to register their storage, leaving the ownership to the original one.
- **Device**: is an IoT device connected to the Internet, able to send transactions with measurements relevant to the supply chain. It can provide, for instance, weights, temperatures, Ph values, RFID tracks of shipments, positions, etc.

There can be different sub-types of each actor. Some actors could bear different roles together; for instance, a transformer or a wholesaler could also be a retailer, if enabled to sell goods directly to the end customer.

2) ENTITIES

The next step is to decompose the design of an agri-food supply chain into its ''entities'', identified as either an object or a process, understood as a concept that has an identity and is meaningful for the model. In this phase, we focus on their identification and not on the relationships between objects and processes. We identified the main entities involved in an agri-food supply chain, that have distinct identities and share common features. They are:

- **Address Catalog**: for each address, the identity and the role(s) of the owner are specified; the catalog is managed by the Administrator/Owner of the system.
- **Producer**: a farmer or a firm producing or transforming agri-food products. Its representative(s) are identified by blockchain addresses.
- **Productive Resource**: it represents something that produces the main raw agri-food products. Typically, it is a field (orchard, vineyard, wheat field, vegetable garden,

greenhouse, olive grove, etc.), a group of animals (flock, herd, poultry, etc.), a set of beehives. It is owned by a Producer. The system can hold information and documents on it and can register events related to its cultivation or farming.

- **Product**: it represents a production batch of something that comes from a Productive Resource, or from the transformation of other products. For instance, grapes are produced from a vineyard, must from pressed grapes, and wine from must. It is linked to a Producer which is in charge of its processing. The system can hold information and documents on it and can register events related to its processing.
- **Token**: a given quantity (a number) created and assigned to a given address. The token represents the ownership of a specific amount of material, good, or asset. It can be split and transferred to other address(es). Since the token, once created, cannot be increased, it guarantees that only the original material/good/asset is managed by the system. Many types of tokens can be managed by the system.
- **Notarization** of documents assessing the process (treatments, harvest, chemical analysis, quantity produced in subsequent steps, etc.) and assuring the parties that the document is authentic and can be trusted. It enables verification of the originality of a document that must be kept on a server and available to download to authorized users. The notarization must include:
  - hash of the document;
  - registration date (always available as date and time of the transaction);
  - link to the document in the server, or information on how to access it;
  - possible metadata.

3) DATA TYPES

Our object model represents the part of the world that is of interest to the agri-food supply chain domain. Some attributes of the model, represent groups of related data that are the same in every agri-food supply chain, such as type, name, unique identifier, owner. However, an agri-food supply system may hold much more data, specific to the particular production and production process.

Since our goal is to develop a general-purpose system, able to be configured for every agri-food production process, we use a flexible data representation. A first set of types allowed by the model are basic types, such as int, float, string, date, text (multiline string), enum. Other data types are more structured and represent the information needed to access data on the Internet or to notarize and check the notarization of data. The allowed types are:

- **int**: numeric input field (digits with an initial '+' or '-' sign). It may have minimum and maximum value constraints.
- **float**: numeric input field with decimal point. It may have minimum and maximum value, and precision of decimal part constraints.
- **string**: one-line string input field. It may have a maximum length. Be careful to filter out any control characters.
- **enum**: input field of a string chosen from a given list. The admissible values are given in a list.
- **text**: multi-line string input field. It may have a maximum length. Be careful to filter out any control characters.
- **link**: a one-line string input field containing a URL. It checks that the URL corresponds to an existing page or file. If the operator clicks on the URL, it shows its content in a pop-up.
- **hashlink**: a one-line string input field containing a URL pointing to a file, with another read-only input field holding the hash digest of the file. It checks that the URL corresponds to an existing file. It calculates the hash of the file with the given algorithm and shows it. If the operator clicks on the URL, it shows its content in a pop-up.
- **upload**: a local file input field that allows the operator to navigate the file system, choose a file, and then activate an "upload" button. If the operator clicks on the file name, it shows its contents in a pop-up.
- **hashupload**: a local file input field as above. It calculates the hash of the file with the given algorithm and shows it, allowing the upload of the file. If the operator clicks on the file name, it shows its contents in a pop-up.
- **upload or hashupload with photo**: it allows the operator to activate the camera of a device, to take a photo, to view it, to discard it and take another, to confirm its sending as a.jpg file, as in the case of "upload". Show a second read-only input field with the file hash digest if hashupload. If the operator clicks on the file name, it shows its contents in a pop-up.

4) EVENTS IN AGRI-FOOD SUPPLY CHAIN

The presented concepts are general for an agri-food supply chain and would be valid also for information systems not based on a blockchain. On the contrary, the events that we are going to describe are directly linked to a supply chain management system based on SCs executed on a blockchain. In other words, they are events registered and enabled by blockchain technology.

In the specific case of Solidity language, an event is an inheritable member of the contract, which stores the arguments in the transaction's log for notifying services outside of the blockchain. In addition, for a better understanding of an agri-food supply chain process, we have grouped all these events into two macro categories:

1) **Transformation events**, which create a product starting from one or more resources, transform one or more products into one or more others or divide the product

into more sub-products that are of the same type, but of lower quantities;

2) **Documentation events**, which associate data related to the production process to a product (or resource), but do not transform it and do not create other products.

We identified the most common events managed by a supply chain management system that uses a blockchain:

- **Asseveration**: a given entity stored on the blockchain (hash of a document, data, etc.) is certified by a transaction sent from an address of a person/body able to certify it. Also, the compliance of a product and token creation or transformation can be asseverated.
- **Creation of tokens** associated with some products of the process, at a given date. The creation can be provided with further data about the physical product associated with the tokens, and even of the notarization of documents attesting the truthfulness of the creation.
- **Product Merging**: the act of merging products of the same type, producing other products of the same type. Different batches of the same material can be merged, producing one or more new batches. The tokens associated with the products must be burned, and new tokens associated with the new products are created, preserving the overall number of tokens.
- **Product Splitting**: the act of splitting one product batch into two or more batches of the same type. Also in this case, the tokens associated with the original product must be burned, and new tokens associated with the new products are created, preserving the overall number of tokens.
- **Product Transformation**: the act of merging one or more products, producing one or more products of different types. For instance, grapes can be transformed into must, used to produce wine, but also to marc used to produce grape pomace brandy. Also in this case, the tokens associated with the original products must be burned. So, new tokens associated with the new products are created by preserving the overall balance of quantities. The transformation is often associated with asseveration events.
- **Data Registration**: a record holding specific data is stored in the blockchain by a given address, guaranteeing the date and the actor who stored it. For instance, fertilization, pesticide treatment, and pruning are recorded as events linked to a field (Productive Resource).
- **Notarization Event**: the data concerning a Notarization (see Section III-A2) are registered by a specific address, ensuring the date and inalterability of the document and the signature of the registrant.
- **Certification of data or of a notarization**: a third party certifies, through a transaction coming from its address, the correctness of a data registration, or of a notarized document.
- **Unlocking**: one or more transactions from given addresses are needed to unlock a process, that is to register another event (typically, a transformation event).
- **Payment**: a payment (in cryptocurrency) is made available to a given address.

Events are declared by using a keyword followed by the name of the event to identify it, and a parameters list to save when the event is triggered. These parameter values enable to log the information or to execute the conditional logic. Events enable communication with the smart contract from the front-end or other applications.

When a transformation event regards more than one product, the operator selects one "main" product batch to start with, then adds the others to the event's data. This operation involves the creation of a new product batch, whereas the original batches are marked as "frozen", and cannot be further modified.

IV. BUILDING A CONFIGURABLE dApp SYSTEM FOR AGRI-FOOD TRACEABILITY

Starting from the aforementioned phase, our approach proposes configurable and modular building blocks for agri-food supply chain management systems that can be represented on a blockchain.

After having analyzed the problem domain and identified the actors and the key entities, we designed a general SC structure, able to represent the problem domain and to be configured to support specific agri-food supply chains. The SCs run on Ethereum blockchain, typically a permissioned version of it, and are written in Solidity language. The data structures of these general SCs include general data, which all instances of the SCs should have, and configurable data, specific for each supply chain. Figure 1 shows the UML class diagram representing the basic structure of our system of Smart Contracts. This diagram uses the ABCDE method notation, which augments UML with stereotypes specific to Solidity language [26]. Here, the UML class notation is used to represent classes and records – denoted with stereotypes "≪class≫" and "≪struct≫", respectively. The meaning of other stereotypes we used is easily understood.

The general data of contracts are shown as UML attributes. Each productive resource or product has a mapping of relevant events, which are the elements where the configurable data are stored. In fact, the data pertaining to specific agri-food productions are always associated with events happening on the resources or products. These data are stored in a byte array named "parameters" of struct "AgriEvent". For each data, we store its type, name, and value, packed into bytes. In this way, the SC representing a single productive resource or product can be configured to represent virtually any possible data structure and all types of events.

The automatic generation of the system's SCs is done starting from JSON, or .scv, files. In our system, the SCs used are the same regardless of the use case (for instance, olive oil, wine, or honey production, just to cite a few). The customization is made by specifying the actual producers involved, the roles active in the system, the name and basic data of the resources and of the type of products managed, the supported events, and finally the actual data used (name and type).

ABCDE method prescribes to perform a thorough evaluation of the produced SCs with respect to their security and gas consumption and also provides checklists to perform this evaluation [26]. In this way, we verified the correctness of our SCs, using security patterns and program verification techniques. We also applied gas-saving patterns to ensure low cost and high efficiency of the SCs, in the case they are run on Ethereum main net.

If mistakes are made while writing configuration files, we can have different scenarios. If these errors lead to an inconsistent system, for example entering identifiers referring to non-existent entities, they are detected immediately and can be easily corrected. If the error cannot be detected automatically, for example, an operator is assigned the wrong actor type or a parameter is associated with an incorrect event, it invalidates the system. In the initial setup phase, it is necessary to carefully check the functioning of the system in all phases of the supply chain, to verify its proper behavior. This has nothing to do with the correctness of SC. Of course, if the data definitions are incorrect, SC will contain incorrect data.

The system creates a SC for each producer (`Producer`), and a SC for each resource (`ProductiveResource`). These SCs, once created, do not change during supply chain management.

Each batch of product being grown, processed, or transformed (`AgriProduct`), is associated with a SC, which takes into account the recordings of events (`AgriEvent`) on it.

Since both a `ProductiveResource` and an `AgriProduct` share several data and operations, they inherit from `AbstractResource` abstract SC.

`Producer` contains a mapping with the identifiers and the addresses of all productive resources and products owned by or related to it.

Products are generated or transformed starting from one or more productive resources, or from one or more existing products. Each product tracks its origin(s) through an array of addresses of the upstream resources or products. An (`AbstractResource` holds a list of events (`AgriEvent`), which in turn contains a list of data related to the event (all encoded in a string of bytes named "parameters"), to flexibly attribute data to the events, as written above. If necessary, a list of generic data could also be added to the AbstractResource, encoded in a string of bytes, as in the "parameters" field of `AgriEvent`.

A QR code printed on the final product allows finding the related `AgriProduct` on the blockchain. It will directly hold the blockchain address of the SC associated with the batch of final products. From the QR code, it is possible to retrieve all the events in (reverse) registration order and, for each event, all relevant data, including links to documents and their possible hash digests.

If an `AgriProduct` has one or more origins, by navigating to these further origins (`AgriProduct` or `ProductiveResource`), all upstream products/resources can be found, with the related events, and so on. At the end of this chain of products, you will also access its `ProductiveResources`, and from this its original `Producers`. Backward navigation must be thoroughly designed to make it easier in the case of multiple origins.

The relationship between product and upstream products/resources is navigable in both directions. An `AgriProduct` contains the array "origins", with the addresses of the contracts of type `ProductiveResource` or `AgriProduct` that created it (`ProductiveResource` has no origins). This is a list created during the creation of `AgriProduct` and cannot be modified. A generic product/resource also contains a "produced" array with the addresses of the generated `AgriProducts`, due to events of type: *Creation*, *Transformation*, *Division*, or *Contribution*. This array can be updated, typically by appending the address of a newly created `AgriProduct`.

To optimize the code, we use OpenZeppelin [44] - a toolkit to develop, compile, update, deploy, and interact with Smart Contracts. We also systematically apply gas-saving patterns [45].

Note, however, that the flexibility of our approach will lead to high gas costs in creating and updating SCs on the Ethereum blockchain. Representing data using arrays of bytes (see next Section IV-A) is way more costly than using native data, both in terms of storage and of computations needed to code and decode it. For this reason, we advise using a permissioned blockchain publicly accessible for reading, and not a public blockchain. The cost of using a permissioned, or consortium, blockchain is much lower, and above all much more predictable than the cost of a public blockchain.

The software tool able to generate SC and User Interface code has been developed together with our industrial partner in the research project that funded the traceability system, NetService Inc. For this reason, the tool is not available under an open-source license, being copyrighted by this firm.

A. DATA TYPES REPRESENTATION

As already mentioned, since our goal is to develop a general-purpose system, able to be configured for every agri-food production process, we use a flexible data representation. Besides the common data, such as type, name, unique identifier, and owner, each element can be provided with a list of data, each represented by a string. The string representing a data includes three sub-strings giving the name of the data, its type, chosen among the set of allowed types previously described, and its value.

For instance, the grape harvest event could have associated the list of parameters that describe how many quintals were collected (type `int`), the grape variety (type `string`), and even a photo taken during the harvest (type `upload`).

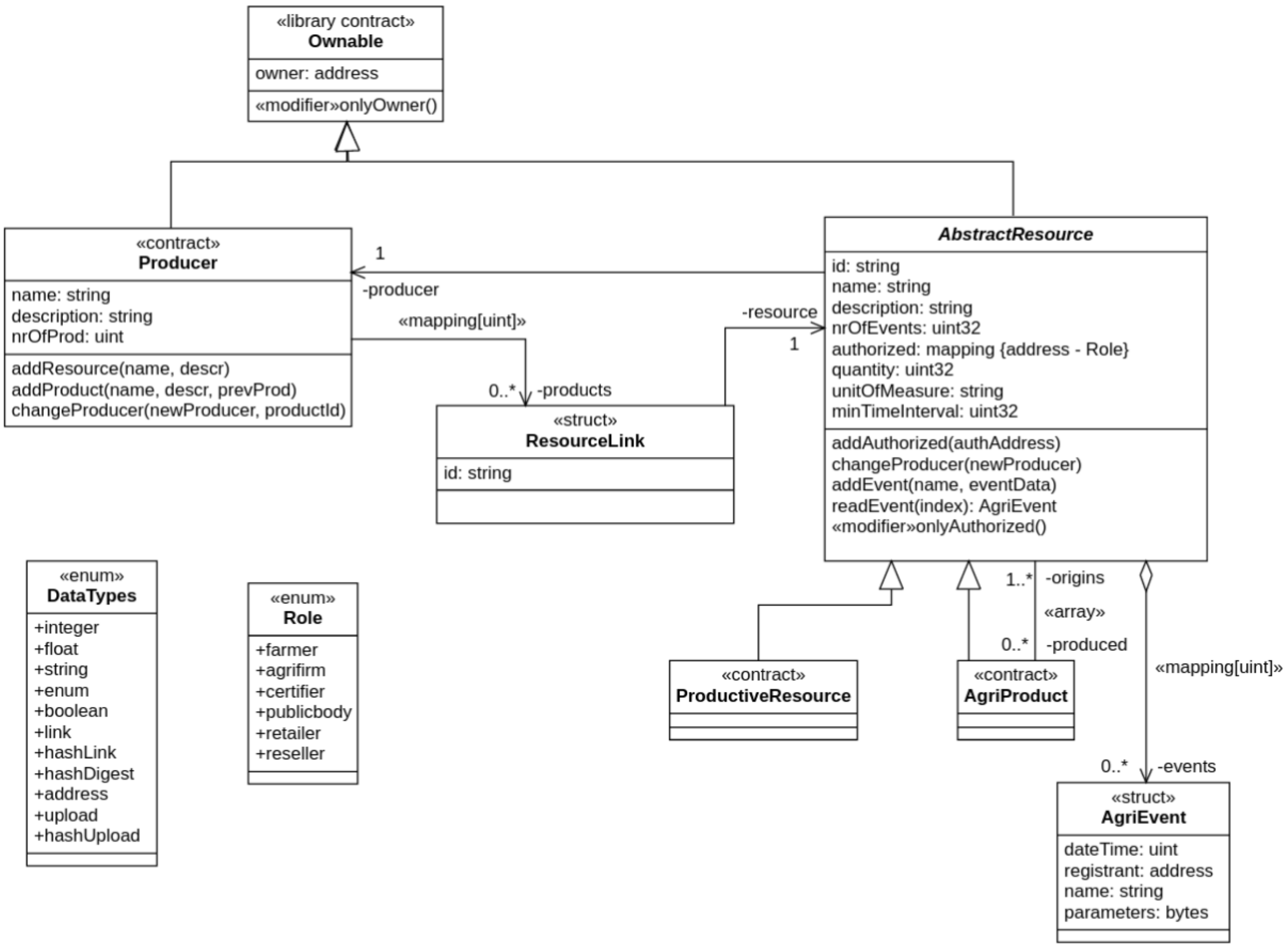


FIGURE 1. UML class diagram, with ABCDE method extensions, representing the smart contracts used in our system.

The following piece of code shows how these three data might be embedded in a string:

```
1harvest\#4t\#3grape\#nebbiolo\#
Bphoto\#https://langhe.it/nebb-12.jpg
```

Here the first character is the data type (1: int, 3: string, B: photo upload), then there is the name of the field, the '#' separator, and the encoded data. Each field ends with a '#' separator. The harvest amount is 225 quintals (`4t` is the Base 58 encoding of 225). The field ('`grape`') has "`nebbiolo`" as a value (a well known variety of red grapes); the field, whose name is '`photo`', holds the URL of an image related to the harvest.

B. OFF-CHAIN COMPONENTS

1) OFF-CHAIN DATA

The costs of data storage in public blockchain are volatile and costly, so the blockchain is not a place suitable for containing large amounts of data. Even if a permissioned blockchain is used, the amount of data that can be stored inside a blockchain is limited, because data would be replicated in every node, unnecessarily wasting resources. Moreover, sometimes storing large amounts of data within a transaction can be downright impossible, due to the limited block size of the blockchain. For example, Ethereum has a block gas limit to limit the number, computational complexity, and size of transaction data included in any block.

In the case of large data, rather than storing the raw data directly on the blockchain, it is useful to store there a link to the data, and a short information able to identify the data. This pattern, known as Off-Chain Data Storage [46], consists in storing the hash digest of the raw data on-chain. This approach can be used to guarantee the date and integrity of such data. The hash value, recorded immutably in a blockchain transaction, guarantees that the original raw data from which the hash was derived were not changed afterward. If the off-chain data change after the recording of their hash digest, the hash digest read from the blockchain and that computed on the changed data will differ, thus demonstrating the alteration.

The data stored off-chain have the following characteristics:

- They are accessed through a URL stored in the blockchain.

- They can be stored in different repositories, under different URL locations.
- If their immutability needs to be certified, their hash digest is stored in the blockchain, together with their URL. The date of registration and the address of the registrant are also always stored.
- The access control to the data must be performed off-chain, by the same system holding the data. In fact, the URL written in the blockchain cannot be hidden, due to blockchain transparency.
- If the data are simple – that is a record with a few, simple fields – it is more convenient to store them directly on-chain. In agri-food management, many operations – for instance fertilization, pesticide treatments, pruning, Ph analysis, and weighing – are described by a few data.

Recently, the combination of a blockchain and a distributed file system has been used and looks promising. According to [12], a well-established platform, InterPlanetary File System (IPFS), that is a peer-to-peer distributed file system ensuring immutability and non-reliance on a central server, could be a valid solution to store data off-chain. This solution has cost advantages and ensures the integrity of the hash value that represents the raw data.

2) USER INTERFACE

The entities and events of an agri-food supply chain, as defined in the sections above, are standard, and can cover most of the production process. Consequently, also the applications enabling their input, editing (when allowed), and retrieving, will perform standard tasks.

For this reason, their user interface (UI) can be automatically generated, starting from the same description of the system used to generate the SCs. More precisely, once the producers, actors, resources, products, and events of a specific production process are defined, with their data, constraints, and authorizations, it is possible to automatically generate an app able to create and edit the events defined for the production process. In our approach, the app is in fact a responsive HTML5 Web page provided with Javascript code.

The style and appearance of the UI can be customized, but the data input, with all proper checks, does not require further programmer's intervention. Moreover, the navigation among the events, the products, and the productive resources can be automatically programmed, starting from a QR code written on the final product. This navigation does not require that the user controls a blockchain address, and can be performed by every customer.

Figure 2 shows some possible screenshots of our application, automatically generated, relating to the honey case study that we are going to present in Section V. In this figure, we show how easy it is possible to record an event (for example *Harvest*) and the data associated with it, to view the history of events associated with a product, or to view the details of an event already recorded.

V. CASE STUDY: A BLOCKCHAIN TRACEABILITY SYSTEM FOR THE HONEY SUPPLY CHAIN

In this section, we show the effectiveness of the approach proposed in the previous section by providing a practical case study in the domain of the agri-food supply chain. This case study is a simplified version of a blockchain-based traceability system for certifying the origin and quality of honey produced by members of a consortium. The requirements for the honey certification system were collected from a consortium of Sardinian honey producers. It is a consortium of small producers of high-quality organic honey. The experimentation of the complete system, in collaboration with two Sardinian honey producers, is currently underway. In particular, the system allows to trace the honey production, from beehives to the honey harvest, to the honey potting, until the sale of jars to a wholesaler, and finally to shops and supermarkets.

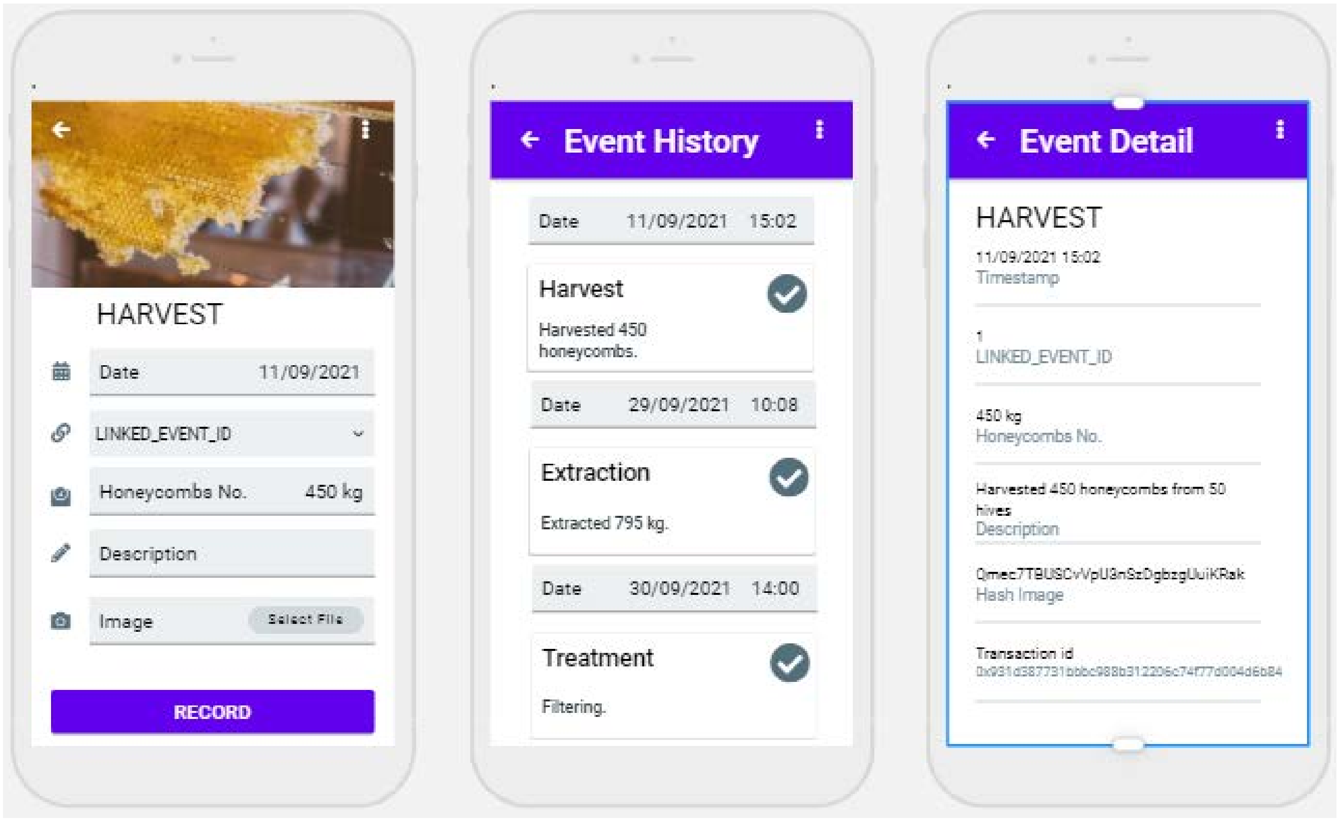
Our choice is not accidental because honey, the main consumer product from beekeeping, has been identified as one of the most adulterated food in the world through dilutions, substitution, or other fraudulent forms such as a misleading description of sources and geographical origin.

In Fig. 3 we represent, in a simplified way, the process of the honey supply chain, in different layers. From top to bottom, the figure shows the actors, the layer of physical products (apiary, honey, jars), that of digital documents stored in one or more servers accessible from the Internet, the layer of "tokens", and of course the blockchain.

In the physical world, the actors perform actions and register the related events in the underlying layers. Some events produce documents or images, which are stored off-chain in the layer of Digital Data, and record the document's link and hash digest on the blockchain, as described in section IV-B1. Other events simply directly record the related information on the blockchain.

The tokens stored in the blockchain, represent the physical quantities of productive resources and products. In our case study, the first token represents the number of beehives of the Productive Resource at the source of the represented honey production. The second token represents the amount of extracted honey expressed in Kg, which is related to the number of beehives – N beehives cannot produce more than kN Kg of honey, where k is a proper constant, depending on the specific kind of bee, beehive, and year. The third token represents the number of jars of a given weight produced, which is obviously related to the amount of honey poured into the jars. The SCs managing the transformation events from the Productive Resource (the apiary, with its beehives) to the Agri Product honey, and from the Agri Product honey to the Agri Product "batch of jars" will enforce the constraints that the amount of honey depends on the number and constant k of beehives and that the amount of honey poured into the jars is greater or equal to the number of jars, multiplied by their capacity.

The proposed solution is designed to be highly transparent and scalable. Anyone can access the dApp website, open the Web page designed for customer access, read the QR code


HARVEST

Date 11/09/2021

LINKED_EVENT_ID

Honeycombs No. 450 kg

Description

Image Select File

RECORD

Event History

Date 11/09/2021 15:02

Harvest

Harvested 450 honeycombs.

Date 29/09/2021 10:08

Extraction

Extracted 795 kg.

Date 30/09/2021 14:00

Treatment

Filtering.

Event Detail

HARVEST

11/09/2021 15:02
Timestamp

1
LINKED_EVENT_ID

450 kg
Honeycombs No.

Harvested 450 honeycombs from 50 hives
Description

Qmec7TBUSCvVpU3nSzDgbzgUuiKRak
Hash Image

Transaction id
0x931d387731bbbc988b312206c74f77d004d6b84

FIGURE 2. Recording an event and navigating a product's event history.

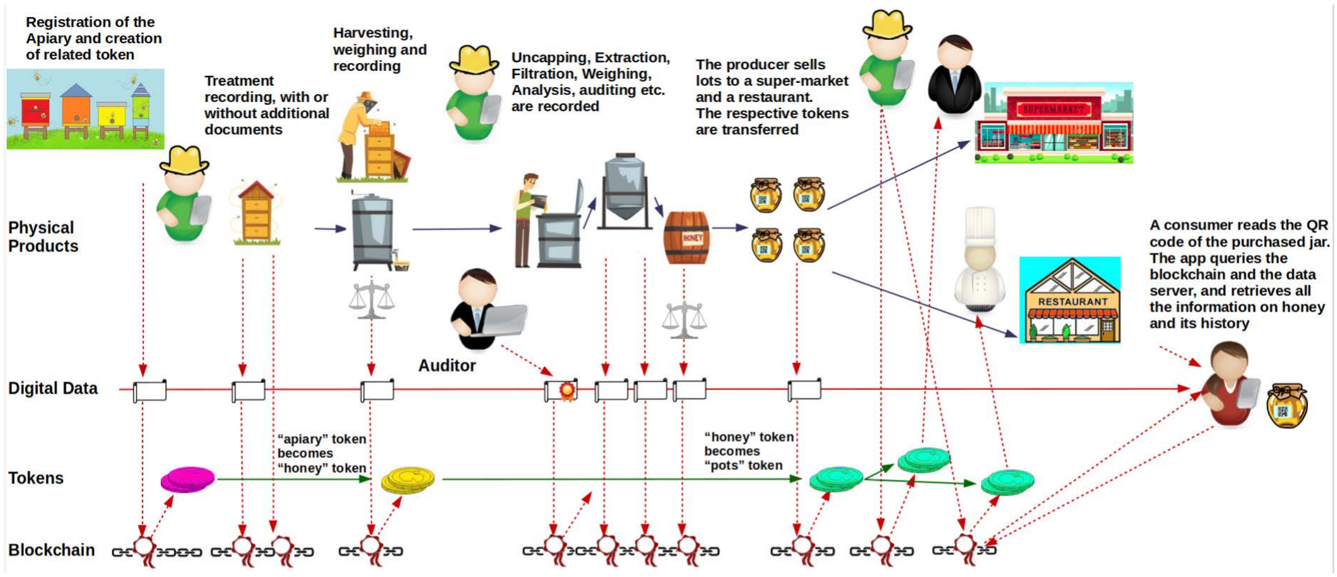

Registration of the Apiary and creation of related token
Treatment recording, with or without additional documents
Harvesting, weighing and recording
Uncapping, Extraction, Filtration, Weighing, Analysis, auditing etc. are recorded
The producer sells lots to a super-market and a restaurant. The respective tokens are transferred
SUPERMARKET
RESTAURANT
A consumer reads the QR code of the purchased jar. The app queries the blockchain and the data server, and retrieves all the information on honey and its history
Physical Products
Auditor
Digital Data
"apiary" token becomes "honey" token
"honey" token becomes "pots" token
Tokens
Blockchain

FIGURE 3. Simplified view of the honey supply chain process.

printed on the honey jar which includes the address of the Agri Product corresponding to the batch of the jar, and from this access the blockchain and start to navigate through the SCs holding the events describing the history of the honey in the jar.

Manufacturers and distributors, as well as retailers, can also create transactions through the mobile application, after a login giving their credentials. After the login, the system Web site will redirect the user to the proper Web page, which of course differs from that shown to generic users. A second control of the actors' ability to send transactions is also made by the SCs, which will accept transactions only from a list of accredited addresses, and further control that the specific service is invoked by an actor enabled to invoke it.

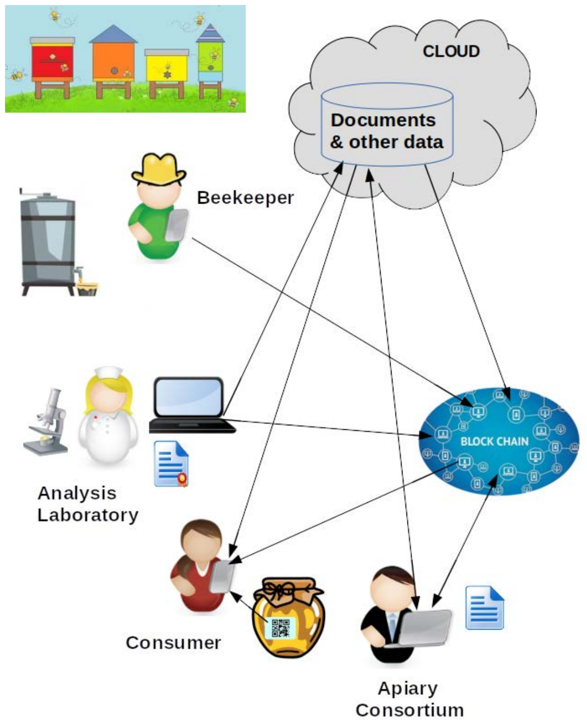


FIGURE 4. System interactions among actors.

Figure 4 schematically shows some of the basic interactions that make up the system. The system consists of the blockchain with its SCs, of one or more servers that manage data on the cloud (documents, images, maps, etc.) and which might be also the websites of the consortium or of the beekeepers, and of the applications that run on a PC or on smartphones connected to the network.

Beekeepers, with their smartphones, can record the treatments done on the apiary, the honey extraction, and all the stages of its transformation. The analysis laboratory records the results on the cloud server, and certifies their hash digest on the blockchain, with a program that runs on a PC. An inspector of the Apiary Consortium examines the certified history of the honey in production and records, in turn, his certification. A consumer, reading the label of the honey jar she bought with a smartphone, can access the whole history of honey, certified on the blockchain.

A. DEFINING THE ACTORS

Accordingly, to the methodology described in this research paper, the honey passes through different chain actors to reach the final consumers. The first step is to identify and describe the actors involved in the system:

- **Beekeeper**: owns the apiaries and takes care of the breeding and good health of the bees. These are people engaged in beekeeping for the production of honey for sale and consumption. They sell raw honey to processors or packages and sell it directly to retailers and consumers. This actor is of type ‘‘Farm’’.
- **Apiary Consortium**: a consortium of beekeepers that acts as a certifier of the quality and origin of honey. They do not participate directly in the production of honey but promote marketing campaigns. This actor is of type ‘‘Certification Authority’’.
- **Processor**: this actor, whose type is ‘‘Transformer’’, purchases crude honey from beekeepers, packages it with its brand name, and then sells the processed honey to retailers and consumers.
- **Certifier**: any independent actor who certifies the quality of the beehives and of the honey (agricultural expert, analysis laboratory, regional body, and the like). It can be of type ‘‘Certification Authority’’, ‘‘Professional’’ or ‘‘Analysis Lab’’;
- **Retailer**: delivers honey to consumers. This is a shop that engages in honey trading by buying honey directly from producers and selling it to final consumers.
- **End Customer**: a person who, given a barrel, bucket, or jar of honey, even before buying the product, wants to verify its origin and history.

B. DEFINING THE ENTITIES

Traditionally, the best-known primary products of beekeeping are honey and wax. Pollen, propolis, royal jelly, poison, bees larvae are also primary marketable bee products. For the sake of simplicity, we deal only with honey, and not with other products derived from the beehive.

The traceability system is therefore built starting from the entities of the honey chain and from the basic events, shown in tables 1 and 2, respectively, which are drawn up describing the specific production process. The primary productive resources are the apiaries, which vary little over time, and usually produce various products, on a periodic basis.

In this analysis, ‘‘honey’’ actually means a specific batch of product, whose processing chain is tracked by the system. A product or a resource can be transformed into another product (for example, an apiary into extracted honey, and then this into honey jars), or it can be divided into batches (which are also ‘‘products’’) for different processes (for example, a product ‘‘Honey’’ can be divided to be given to two different producers for potting and reselling). A product can derive from one or more primary resources, or from one or more upstream products.

Both productive resources and agro-industrial products contain the following data, recorded in the blockchain:

- **id**: unique internal identifier managed by the system;
- **name**: name of the product (or apiary);
- **quantity**: quantity of the product or resource;
- **unit of measure**: unit of measure of the given quantity;
- **producer**: a firm of the supply chain. It can be a beekeeper who owns, takes care of bees, collects honey and transforms it for consumption, a firm that buys honey from beekeepers, and pots it, and a firm that only deals with marketing, and selling honey pots.
- **authorized** list of operators enabled to enter events on the product or resource.

The quantity serves to prevent products that are not tracked from being introduced into the process in an uncontrolled way. Data relating to a product or resource are associated with it through specific events.

Table 1 shows the system entities, the data associated with them, and their description. Clearly, an Apiary is a Productive Resource, while Honeycomb, Honey and Honey jars are ''Prod

C. THE SYSTEM'S EVENTS

We recall that there are two types of events: *Transformation Events* (TE) and *Documentation Events* (DE). Transformation events transform Apiary Productive resource into Honeycombs, Honeycombs into Honey, Honey into Honey jars (or buckets). The system also handles events able to transform two or more products into a single product (in our case, several batches of honey, coming from different hives, could be merged into a single batch), and to create one or more products starting from one or more products of the same type (in our case, a batch of honey could be divided into smaller batches).

Table 2 shows the types of events managed by our system. The ''Device'' column shows the device with which the relative data are entered, which can be: PC, Smartphone, Tablet, or IoT device. The ''Data storage'' column illustrates where the data related to the event are stored: if it reports ''blockchain'', the data are only in the blockchain, if it reports anything else, the data are in the indicated device and a corresponding link to the real data, and their hash digest, is registered in the blockchain. For each event, a description is provided. The ''Entity'' column describes the resources/products holding the data of the event.

Figure 5 shows a simplified honey production process. Agro-industrial products are denoted with icons depicting the product. Each product can be associated with one or more *Documentation events* (DE), in the image denoted by the document with seal icon, and the dotted arrow. *Transformation events* (TE) are denoted by solid arrows with the corresponding event label, starting from the right side of products or resources, and cause them to be transformed into other products, or divided into batches. In the example shown, the honey comes from two apiaries (1 and 2). Through a single event (Harvest) it becomes honeycomb. Once extracted (Extraction), it becomes honey and is then divided (Splitting) into two batches (Honey 1 and Honey 2). These batches are then potted; the first is put in jars, the second in buckets.

D. AUTOMATIC GENERATION OF THE SYSTEM'S SMART CONTRACTS

As already mentioned, the system's SCs are automatically generated from JSON, or .csv, files (in fact,.csv files are converted into JSON ones). In the case study system, we have five JSON files. Specifically, we have a JSON file for each of the concepts needed to manage the system: actors, producers, resources/products, events, and parameters.

Figure 6 shows the content of these files for actors, producers, and resources (Productive Resources and Products), for three producers – a beekeeper (Miele Monte Arcosu), a firm that pots and sells honey (Miele di Sardegna), and a honey retailer (Mielizia).

The described resources are two apiaries managed by the beekeeper, various types of intermediate (honeycombs and raw honey), and final products (buckets and jars of honey). Only the apiaries have a defined quantity of beehives, which is set when the system is initialized and can be changed using a specific function of the Product SC, called by its owner. The quantities of other products are set by the events which create them, under the constraints quoted in Section III-A4, ''Transformation events''.

Figure 7 shows the content of the.csv file with the Event definition, and of the file with the definition of the data fields specific for each event.

The event definition includes the list of the resources the event can be attached to and, in the case of a transformation event, the list of the products that can be generated. These events define also the factor k, which is used to determine the quantity of the generated product, given the quantity of the generating product. For instance, the unit of measure of honey is Kg, whereas the unit of measure of 500 g jars is their number. So, the conversion factor is 2 – 1 Kg of honey will produce 2 jars. Description events have neither output products, nor conversion factors.

The parameters table, shown in the same figure, reports for each event the data specific for that event. For instance, let's examine the Harvest event (HVT). Remember that in our process harvest means to collect the honeycombs full of honey from the apiaries. The harvest event has three data fields:

- **Honeycombs No**: is equal to the number of harvested honeycombs, is an integer, and is mandatory.
- **Description**: is a string holding a description of no more than 80 characters, optional.
- **Image**: is an optional image, obtained also using the smartphone camera, uploaded to a cloud server, and whose URL and hash digest are stored in the blockchain; it is optional.

This information is also used to automatically generate the user interface of the apps in charge of taking inputs from the operators of the system, and of the app able to navigate through the history of a final product.

1) SYSTEM's CHARACTERISTICS

To summarize, the proposed system has the following characteristics:

1) Ability to manage the honey production cycle, as described following field inspections and interviews with beekeepers; at a later time, other products can be added such as propolis, royal jelly, wax, etc.
2) Use of cloud space to store and retrieve documents, both documents produced by the operators, such as

TABLE 1. Resources and products managed by a honey traceability system.

Entity	Data storage	Description
Apiary	Blockchain, Producer's server	Registration of a production area, and related beehives. It can include photos, maps, and other documents, which are kept on a dedicated server. The recording of this data takes place during the system initialization phase before it enters operations. New data can be added, or data can be modified, at a later time. The blockchain holds links to the data, as well as their hash digest.
Honeycomb	Blockchain, possibly Producer's server	Registration of a specific set of honeycombs, harvested from the beehives. It is associated with events that document its harvesting, and which may include pdf files, photos, and other documents, which are kept on a dedicated server.
Honey	Blockchain, possibly Producer's or Retailer's server	Registration of a specific batch of honey, extracted from the honeycombs. It is associated with events that document its processing, and which may include pdf files, photos, and other documents, which are kept on a dedicated server.
Honey jars	Blockchain, possibly Producer's server	Registration of a batch of packaged honey, in which all jars have the same QR code on the label, allowing product tracing. Here we deal with jars, but in principle, they might be also drums or buckets.

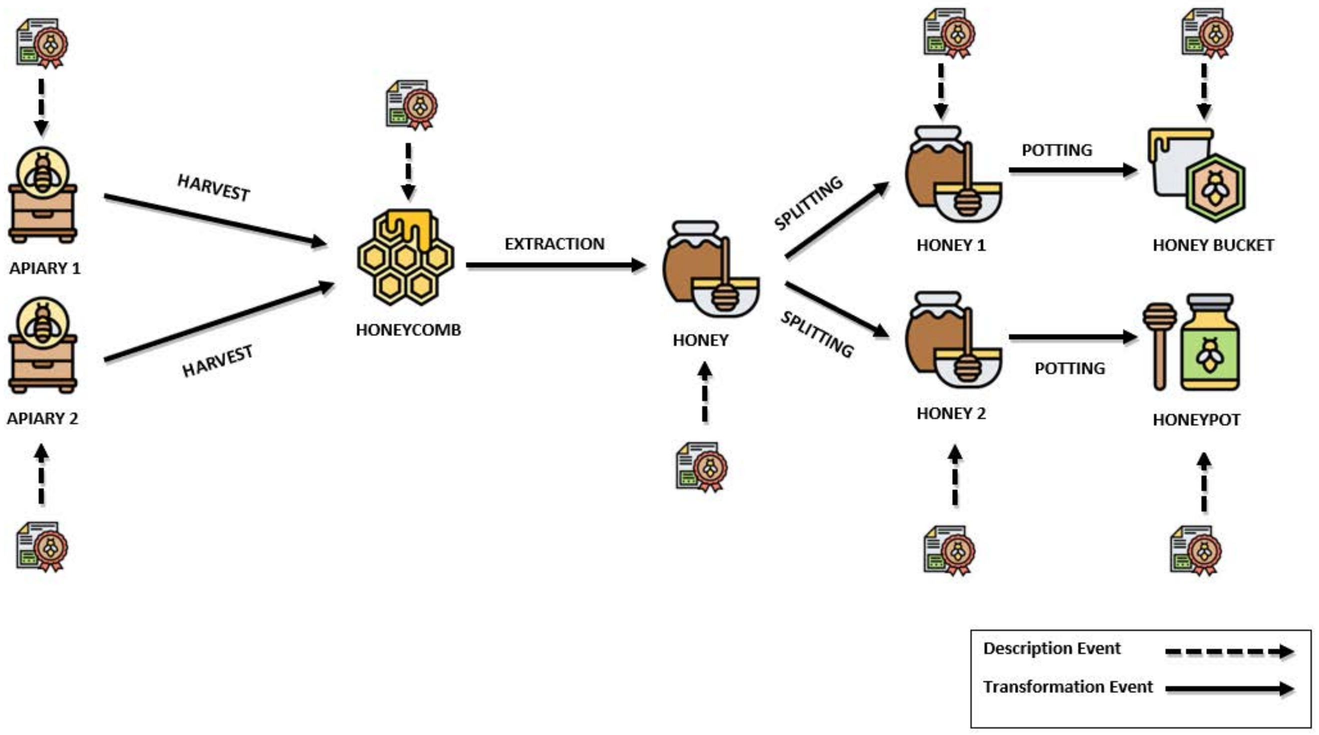


FIGURE 5. Honey production flow.

photos, and documents available in other ways, as long as they are accessible from the Internet.

3) Use of standard Android smartphones or tablets for data entry by system operators.
4) Use of a responsive website for system operations and management by the operators, and for retrieving information on honey batches by consumers.
5) Use of a "permissioned" blockchain with public access to ensure cost stability.
6) Need to produce and associate appropriate labels with QR code, to tag the final products (jars, buckets, honey drums).
7) Opportunity for the Apiary Consortium to become an official actor and certifier of the traced batches of honey. for this purpose, it must:
   a) Have a blockchain address, and an administrator to manage its private key and send transactions.
   b) Advertise the address on its site, to assure users about the identity behind the transactions originating from that address.
   c) Define procedures and parameters to certify the batches of honey produced by its members, like: actual size and good state of health of an apiary, organoleptic qualities of the batch of honey analyzed, certification of laboratory analyses made on honey samples, etc.
8) The final consumer, pointing the smartphone at the QR code of the honey jar label, is connected to the website of point 4, and here s/he can go back to the "history" and certifications of the honey purchased.

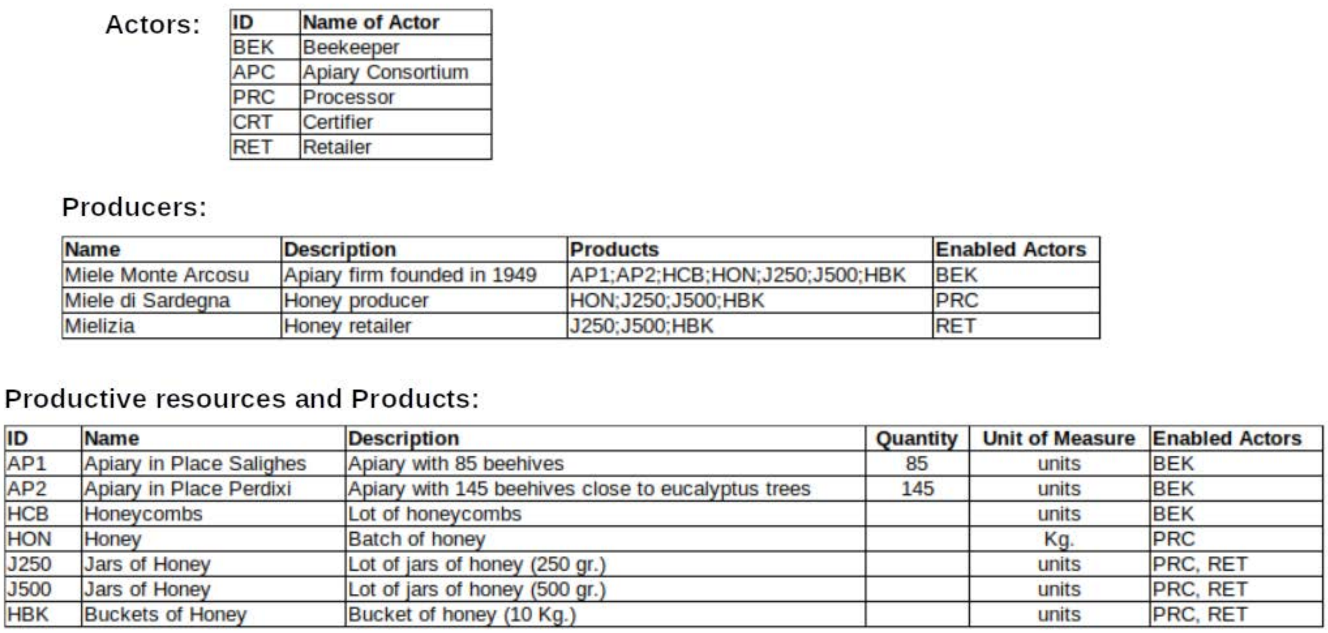
The system is currently under consideration by a consortium of beekeepers of Central Sardinia. We are working

TABLE 2. **Events and functions of a honey traceability system: Transformation Events (TE), documentation events (DE).**

Event/Type	**Device**	**Data storage**	**Related Entity**	**Description**
Harvest **TE**	Smartphone or Tablet	Blockchain, Producer's server	Apiary	This event occurs when honeycombs are harvested from beehives, at a given time of the year. It creates a Honeycomb batch. It guarantees that the transformation takes place respecting the temporal and quantitative constraints, that is the total quantity collected per year must be consistent with the number of hives in the apiary, according to a parameter defined when initializing the Apiary SC. For instance, a given apiary cannot produce more than 300 kg of honey per year.
Extraction **TE**	Smartphone or Tablet	Blockchain, Producer's server	Honeycomb	This event takes place when honey is extracted from honeycombs, after their harvesting. It creates a Honey batch. It guarantees that the transformation takes place respecting the constraints on the total quantity of honey contained inside the honeycombs, and on the percentage that can be extracted.
Merging **TE**	Smartphone or Tablet	Blockchain	Honey	Two or more batches of honey are merged and transformed into a new batch. The event ensures that the total amount of honey is conserved. This event is initiated by a "pivot" honey Agri-Product and is registered for all input honey batches. It does not include off-chain information.
Splitting **TE**	Smartphone or Tablet	Blockchain	Honey	A (batch of) honey is split into two or more batches. The sum of the quantities of the new products created must be equal to (or possibly less than) the quantity of the original product.
Potting **TE**	Smartphone or Tablet	Blockchain, Producer's server	Honey	A batch of honey is potted into a lot of jars. The event guarantees that the total amount of honey of the jars is equal to (or less than) the quantity of the input honey batch. Additional information related to the event can be stored off-chain.
Conferral **DE**	Smartphone or Tablet	Blockchain	Honey or Honey jars	A product is conferred by the Producer in charge of it to another actor. For example, a given quantity of honey jars is sold to a wholesaler. The conferral must be accepted by the destination producer; this event, if the conferral is not accepted, produces nothing and can be canceled by the simple insertion of subsequent events. Hence, it is a Documentation Event.
Acceptance **DE**	Smartphone or Tablet	Blockchain	Honey or Honey jars	A Producer (Processor or Retailer) accepts the conferral of a product by another Producer. The ownership of the Agri-Product is transferred to this Producer; no new product is created, so this is not a transformation event. Subsequent events will be registered by the new owner.
Description **DE**	PC	Blockchain, Producer's server	Apiary, Honey or Honey jars	Descriptive data associated with a product or an apiary are recorded. It can include photos, maps, and other documents, which are kept on a dedicated server. It can affect the quantities produced (e.g.: if hives are added or removed to an apiary).
Treatment **DE**	Smartphone or Tablet	Blockchain, possibly Producer's server	Apiary or Honey	Registration of bee breeding practices made on an apiary, or treatments made during the production process (addition of preservatives, pasteurization, and so on.). It is possible to integrate it with off-chain documents, kept on the Producer's server.
Certification **DE**	Smartphone or Tablet	Blockchain, Producer's or Certifier's server	Apiary, Honey or Honey jars	A certifier (Apiary Consortium, or another Certifier) certifies a treatment or the produced honey. It produces a certificate, kept on the server, whose hash digest is registered on the blockchain. This event is registered by a transaction sent from the address of the certifier, so its origin is certain.
Automatic Registration **DE**	IoT	Blockchain	Agri-Product	An IoT device performs a registration following an external event. For example, a refrigerator records the internal temperature every 2 hours. In this way, it will be possible to ascertain the storage conditions of fresh food. Not relevant for the honey system.
Verification	Smartphone, Tablet or PC		Honey jars	This is not a true event, but a service provided by the system. End customers, or any other actor in the process, check the integrity and history of the product. They read the product QR code, which includes the address of the SC related to the honey jars lot, and can read its history on the blockchain. If they want, they can also find the related documents (e.g.:.pdf files, images) following the links.

with two small apiary businesses focused on high-quality organic production, to install and evaluate the prototype of the system.

As already pointed out, the proposed system is flexible enough to support almost all types of agri-food chains, possibly with some specific additions. A previous conference paper [13] describing a preliminary version of the presented system was about olive oil production. The firm which developed with us the system is also applying it to wine and cheese production.



Actors:

ID	Name of Actor
BEK	Beekeeper
APC	Apiary Consortium
PRC	Processor
CRT	Certifier
RET	Retailer

Producers:

Name	Description	Products	Enabled Actors
Miele Monte Arcosu	Apiary firm founded in 1949	AP1;AP2;HCB;HON;J250;J500;HBK	BEK
Miele di Sardegna	Honey producer	HON;J250;J500;HBK	PRC
Mielizia	Honey retailer	J250;J500;HBK	RET

Productive resources and Products:

ID	Name	Description	Quantity	Unit of Measure	Enabled Actors
AP1	Apiary in Place Salighes	Apiary with 85 beehives	85	units	BEK
AP2	Apiary in Place Perdixi	Apiary with 145 beehives close to eucalyptus trees	145	units	BEK
HCB	Honeycombs	Lot of honeycombs		units	BEK
HON	Honey	Batch of honey		Kg.	PRC
J250	Jars of Honey	Lot of jars of honey (250 gr.)		units	PRC, RET
J500	Jars of Honey	Lot of jars of honey (500 gr.)		units	PRC, RET
HBK	Buckets of Honey	Bucket of honey (10 Kg.)		units	PRC, RET

FIGURE 6. The content of the.csv files describing actors, producers and resources.

VI. CONCLUSION AND FUTURE WORK

Nowadays, consumers worldwide want to be sure that the food they eat is safe and can be reliably traced back to its point of origin to give assurance that what they are buying is authentic and healthy. For this reason, they are demanding the highest standards of food safety throughout the supply chain and they are willing to pay for the intangible attributes of secure traceability and country of origin labeling. Traceability systems are considered important to ensure the safety of a food product and prevent food fraud in the food supply chain. It is essential to improve the current traceability systems, as unscrupulous producers could exploit the gaps in the systems to their advantage and to the detriment of consumers.

Systems based on blockchain technology and smart contracts, integrated with the Internet of Things, allow to implement a traceability system where the producers can share the responsibility to contribute information to their products, and independent third parts can identify themselves and certify the correctness of the data related to products' origin and quality. In this way, the customer can be assured of the truthfulness of the reported information with a high degree of confidence.

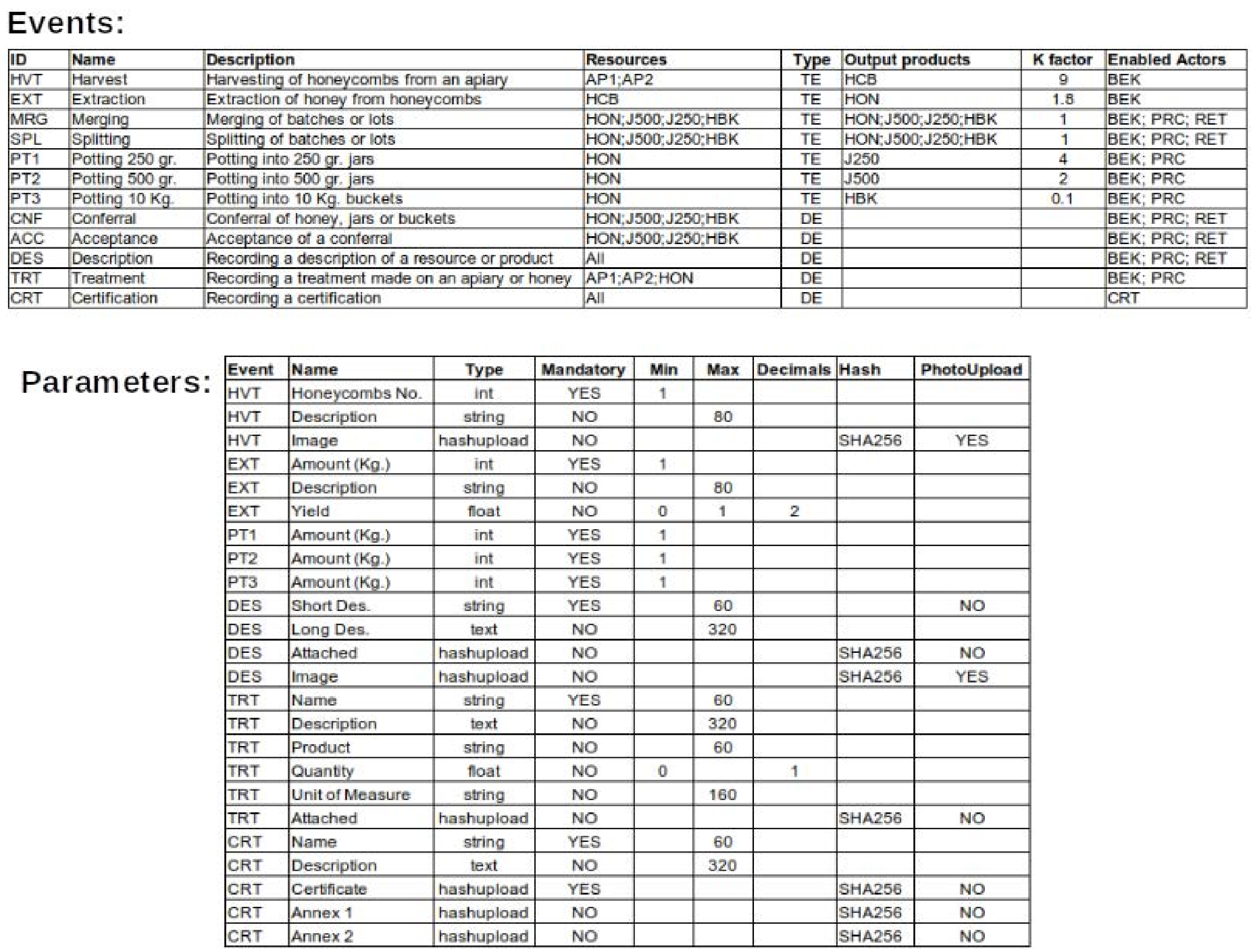
In this context, we proposed a system enabling developers to quickly and smoothly develop traceability systems in the agri-food domain, without the need to grasp in every detail the technicalities of SC development, which is clearly different from classical software development. To this purpose, we accurately represented the problem domain, which was found suitable for such an approach, and developed a system able to automatically generate both the SCs and the UI of a tracing system.

Our approach starts from the description of the supply chain to be traced in terms of actors, producers, resources and products, events, and data. This description is given using a set of spreadsheet pages, which is a tool very easy to use also by people expert in the domain, but not in computer science. From these pages, converted to.csv files, the SCs are generated, as well as the HTML5 pages able to interact with them and providing the UI of the dApp.

This methodology can be used at every node of the supply chain and can capture critical events, which are subsequently recorded immutably. Also, the actors who registered the events can be identified with a very high degree of certainty. In this way, the certification of every step of the production process is not only made by the producer itself – as it is in traditional systems – but can be audited by trusted third parties, which gives a much higher degree of trust that the information on the product is correct.

Specifically, the advantages of the management system of our agri-food supply chain via blockchain, generated in a semi-automatic way, are:

- The consumer can be sure of the origin, of the production process, and of the quality of the product purchased.
- The task of the authorities in charge of the control of products and of production processes is facilitated, and on-site inspections can be reduced.
- The manufacturer can certify in a simple and non-falsifiable way all the steps of production.
- Software development times and costs are reduced while maintaining a high level of security and trust.
- Blockchain system might also manage contractual transactions and payments. In this case, the system could also be extended to allow payments, and the transfer of



Events:

ID	Name	Description	Resources	Type	Output products	K factor	Enabled Actors
HVT	Harvest	Harvesting of honeycombs from an apiary	AP1;AP2	TE	HCB	9	BEK
EXT	Extraction	Extraction of honey from honeycombs	HCB	TE	HON	1.8	BEK
MRG	Merging	Merging of batches or lots	HON;J500;J250;HBK	TE	HON;J500;J250;HBK	1	BEK; PRC; RET
SPL	Splitting	Splitting of batches or lots	HON;J500;J250;HBK	TE	HON;J500;J250;HBK	1	BEK; PRC; RET
PT1	Potting 250 gr.	Potting into 250 gr. jars	HON	TE	J250	4	BEK; PRC
PT2	Potting 500 gr.	Potting into 500 gr. jars	HON	TE	J500	2	BEK; PRC
PT3	Potting 10 Kg.	Potting into 10 Kg. buckets	HON	TE	HBK	0.1	BEK; PRC
CNF	Conferral	Conferral of honey, jars or buckets	HON;J500;J250;HBK	DE			BEK; PRC; RET
ACC	Acceptance	Acceptance of a conferral	HON;J500;J250;HBK	DE			BEK; PRC; RET
DES	Description	Recording a description of a resource or product	All	DE			BEK; PRC; RET
TRT	Treatment	Recording a treatment made on an apiary or honey	AP1;AP2;HON	DE			BEK; PRC
CRT	Certification	Recording a certification	All	DE			CRT

Parameters:

Event	Name	Type	Mandatory	Min	Max	Decimals	Hash	PhotoUpload
HVT	Honeycombs No.	int	YES	1				
HVT	Description	string	NO		80			
HVT	Image	hashupload	NO				SHA256	YES
EXT	Amount (Kg.)	int	YES	1				
EXT	Description	string	NO		80			
EXT	Yield	float	NO	0	1	2		
PT1	Amount (Kg.)	int	YES	1				
PT2	Amount (Kg.)	int	YES	1				
PT3	Amount (Kg.)	int	YES	1				
DES	Short Des.	string	YES		60			NO
DES	Long Des.	text	NO		320			
DES	Attached	hashupload	NO				SHA256	NO
DES	Image	hashupload	NO				SHA256	YES
TRT	Name	string	YES		60			
TRT	Description	text	NO		320			
TRT	Product	string	NO		60			
TRT	Quantity	float	NO	0		1		
TRT	Unit of Measure	string	NO		160			
TRT	Attached	hashupload	NO				SHA256	NO
CRT	Name	string	YES		60			
CRT	Description	text	NO		320			
CRT	Certificate	hashupload	YES				SHA256	NO
CRT	Annex 1	hashupload	NO				SHA256	NO
CRT	Annex 2	hashupload	NO				SHA256	NO

FIGURE 7. **The content of the.csv files describing events, and parameters, that is data values.**

ownership of a product could also be associated with a cryptocurrency transfer.

To the best of our knowledge, this is the first attempt to automatically develop custom dApps for the agri-food supply chain, by building configurable SCs to be assembled together. We truly believe that the proposed approach is very innovative. Moreover, the proposed approach was actually used to develop some real tracing systems, thus confirming its capacities.

The research presented in this paper adds value to the state of the art in several ways. Firstly, it helps the developers in creating higher-quality SCs, because the SCs which are configured for a specific system are already proven and debugged. Secondly, it can help reduce development time, because systems are generated by compiling a description of the system given as tables of data. Thirdly, this approach makes food safety compliance easy and significantly cuts down on paperwork for the actors in the agri-food supply chain.

Our flexible approach has potential applications well beyond the honey industry, which we proposed as a case study, to a range of other agri-food producers. It could be widely used for the design and development of dApps aimed to implement agri-food supply chain traceability systems. The presented methodology facilitates the communication between domain experts and developers, and then automatically transforms the key concepts of the problem domain into SC code.

We are presently working on two extensions of the approach. First, to reduce the cost in gas of SC creation and execution, we are working to give the system the ability to automatically generate specific SC data structures from the data specification, instead of using the representation with byte arrays. This is not a simple task, because we have also to generate getter and setter functions, maintaining the security and reliability level of the present system.

Secondly, we are working to port the approach and the system to other Distributed Ledger Technology systems which are used for implementing permissioned blockchains, namely

Hyperledger Fabric and Tendermint. We are also considering generalizing the system to support other types of supply chains, besides the agri-food sector.

REFERENCES

[1] S. Nakamoto, ''Bitcoin: A peer-to-peer electronic cash system,'' Manubot, Tech. Rep., 2008.

[2] K. Demestichas, N. Peppes, T. Alexakis, and E. Adamopoulou, ''Blockchain in agriculture traceability systems: A review,'' *Appl. Sci.*, vol. 10, no. 12, pp. 1–22, 2020.

[3] C. Costa, F. Antonucci, F. Pallottino, J. Aguzzi, D. Sarriá, and P. Menesatti, ''A review on agri-food supply chain traceability by means of RFID technology,'' *Food Bioprocess Technol.*, vol. 6, no. 2, pp. 353–366, Feb. 2013.

[4] F. Tian, ''An agri-food supply chain traceability system for China based on RFID & blockchain technology,'' in *Proc. 13th Int. Conf. Service Syst. Service Manage. (ICSSSM)*, Jun. 2016, pp. 1–6.

[5] F. Antonucci, S. Figorilli, C. Costa, F. Pallottino, L. Raso, and P. Menesatti, ''A review on blockchain applications in the agri-food sector,'' *J. Sci. Food Agricult.*, vol. 99, no. 14, pp. 6129–6138, Nov. 2019.

[6] P. Bottoni, N. Gessa, G. Massa, R. Pareschi, H. Selim, and E. Arcuri, ''Intelligent smart contracts for innovative supply chain management,'' *Frontiers Blockchain*, vol. 3, p. 52, Nov. 2020.

[7] J. F. Galvez, J. C. Mejuto, and J. Simal-Gandara, ''Future challenges on the use of blockchain for food traceability analysis,'' *TrAC Trends Anal. Chem.*, vol. 107, pp. 222–232, Oct. 2018.

[8] C. Dalvit, M. De Marchi, and M. Cassandro, ''Genetic traceability of livestock products: A review,'' *Meat Sci.*, vol. 77, no. 4, pp. 437–449, Dec. 2007.

[9] E. Commission. *General Food Law*. Accessed: Feb. 20, 2022. [Online]. Available: https://ec.europa.eu/food/horizontal-topics/general-food-law_en

[10] M. M. Aung and Y. S. Chang, ''Traceability in a food supply chain: Safety and quality perspectives,'' *Food Control*, vol. 39, pp. 172–184, May 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0956713513005811

[11] L. Cocco and K. Mannaro, ''Blockchain in agri-food traceability systems: A model proposal for a typical Italian food product,'' in *Proc. IEEE Int. Conf. Softw. Anal., Evol. Reeng. (SANER)*, Mar. 2021, pp. 669–678.

[12] L. Cocco, K. Mannaro, R. Tonelli, L. Mariani, M. B. Lodi, A. Melis, M. Simone, and A. Fanti, ''A blockchain-based traceability system in agri-food SME: Case study of a traditional bakery,'' *IEEE Access*, vol. 9, pp. 62899–62915, 2021.

[13] L. Marchesi, K. Mannaro, and R. Porcu, ''Automatic generation of blockchain agri-food traceability systems,'' in *Proc. IEEE/ACM 4th Int. Workshop Emerg. Trends Softw. Eng. Blockchain (WETSEB)*, May 2021, pp. 41–48, doi: 10.1109/wetseb52558.2021.00013.

[14] C. D. Clack, V. A. Bakshi, and L. Braine, ''Smart contract templates: Foundations, design landscape and research directions,'' 2016, *arXiv:1608.00771*.

[15] M. Alharby and A. van Moorsel, ''Blockchain-based smart contracts: A systematic mapping study,'' 2017, *arXiv:1710.06372*.

[16] W. Zou, D. Lo, P. S. Kochhar, X.-B.-D. Le, X. Xia, Y. Feng, Z. Chen, and B. Xu, ''Smart contract development: Challenges and opportunities,'' *IEEE Trans. Softw. Eng.*, vol. 47, no. 10, pp. 2084–2106, Oct. 2021.

[17] H. Rocha and S. Ducasse, ''Preliminary steps towards modeling blockchain oriented software,'' in *Proc. 1st Int. Workshop Emerg. Trends Softw. Eng. Blockchain*, May 2018, pp. 52–57.

[18] M. Jurgelaitis, R. Butkiene, E. Vaičiukynas, V. Drungilas, and L. Čeponiene, ''Modelling principles for blockchain-based implementation of business or scientific processes,'' *CEUR Workshop Proc.*, vol. 2470, pp. 43–47, Apr. 2019.

[19] A. B. Tran, Q. Lu, and I. Weber, ''Lorikeet: A model-driven engineering tool for blockchain-based business process execution and asset management,'' in *Proc. 16th Int. Conf., BPM*. Sydney, NSW, Australia: Springer, Sep. 2018.

[20] Q. Lu, A. Binh Tran, I. Weber, H. O'Connor, P. Rimba, X. Xu, M. Staples, L. Zhu, and R. Jeffery, ''Integrated model-driven engineering of blockchain applications for business processes and asset management,'' *Softw., Pract. Exper.*, vol. 51, no. 5, pp. 1059–1079, May 2021.

[21] C. Udokwu and A. Norta, ''Deriving and formalizing requirements of decentralized applications for inter-organizational collaborations on blockchain,'' *Arabian J. Sci. Eng.*, vol. 46, no. 9, pp. 8397–8414, Sep. 2021.

[22] C. Udokwu, P. Brandtner, A. Norta, A. Kormiltsyn, and R. Matulevičius, ''Implementation and evaluation of the DAOM framework and support tool for designing blockchain decentralized applications,'' *Int. J. Inf. Technol.*, vol. 13, no. 6, pp. 2245–2263, Dec. 2021.

[23] C. K. Frantz and M. Nowostawski, ''From institutions to code: Towards automated generation of smart contracts,'' in *Proc. IEEE 1st Int. Workshops Found. Appl. Self Syst. (FAS*W)*, Sep. 2016, pp. 210–215.

[24] V. A. de Sousa, C. Burnay, and M. Snoeck, ''B-MERODE: A model-driven engineering and artifact-centric approach to generate smart contracts,'' in *Proc. Conf. Adv. Inf. Syst. Eng.*, Cham, Switzerland: Springer, 2020, pp. 1–15.

[25] M. Marchesi, L. Marchesi, and R. Tonelli, ''An agile software engineering method to design blockchain applications,'' in *Proc. 14th Central Eastern Eur. Softw. Eng. Conf. Russia ZZZ (CEE-SECR)*, 2018, pp. 1–8.

[26] L. Marchesi, M. Marchesi, and R. Tonelli, ''ABCDE—Agile block chain DApp engineering,'' *Blockchain, Res. Appl.*, vol. 1, nos. 1–2, Dec. 2020, Art. no. 100002.

[27] A. Mavridou and A. Laszka, ''Designing secure Ethereum smart contracts: A finite state machine based approach,'' in *Proc. Int. Conf. Financial Cryptogr. Data Secur.*, Cham, Switzerland: Springer, 2018, pp. 523–540.

[28] M. Tripoli and J. Schmidhuber, ''Emerging opportunities for the application of blockchain in the agri-food industry,'' FAO ICTSD, Rome Geneva, Italy, Tech. Rep. CC BY-NC-SA, 2018, vol. 3.

[29] Y. Tribis, A. El Bouchti, and H. Bouayad, ''Supply chain management based on blockchain: A systematic mapping study,'' *MATEC Web Conf.*, vol. 200, Sep. 2018, Art. no. 00020.

[30] M. S. Farooq, S. Riaz, A. Abid, T. Umer, and Y. B. Zikria, ''Role of IoT technology in agriculture: A systematic literature review,'' *Electronics*, vol. 9, no. 2, p. 319, Feb. 2020.

[31] F. Tian, ''A supply chain traceability system for food safety based on HACCP, blockchain & Internet of Things,'' in *Proc. Int. Conf. Service Syst. Service Manage.*, Jun. 2017, pp. 1–6.

[32] Y. Wang, J. H. Han, and P. Beynon-Davies, ''Understanding blockchain technology for future supply chains: A systematic literature review and research agenda,'' *Supply Chain Manage., Int. J.*, vol. 24, no. 1, pp. 62–84, Jan. 2019.

[33] J. Li and X. Wang, ''Research on the application of blockchain in the traceability system of agricultural products,'' in *Proc. 2nd IEEE Adv. Inf. Manage., Communicates, Electron. Autom. Control Conf. (IMCEC)*, May 2018, pp. 2637–2640.

[34] A. Iftekhar, X. Cui, M. Hassan, and W. Afzal, ''Application of blockchain and Internet of Things to ensure tamper-proof data availability for food safety,'' *J. Food Qual.*, vol. 2020, pp. 1–14, May 2020.

[35] S. S. Kamble, A. Gunasekaran, and R. Sharma, ''Modeling the blockchain enabled traceability in agriculture supply chain,'' *Int. J. Inf. Manage.*, vol. 52, Jun. 2020, Art. no. 101967.

[36] Y. Chang, E. Iakovou, and W. Shi, ''Blockchain in global supply chains and cross border trade: A critical synthesis of the state-of-the-art, challenges and opportunities,'' *Int. J. Prod. Res.*, vol. 58, no. 7, pp. 2082–2099, Aug. 2019.

[37] M. P. Caro, M. S. Ali, M. Vecchio, and R. Giaffreda, ''Blockchain-based traceability in agri-food supply chain management: A practical implementation,'' in *Proc. IoT Vertical Topical Summit Agricult. Tuscany (IoT Tuscany)*, May 2018, pp. 1–4.

[38] G. Baralla, A. Pinna, and G. Corrias, ''Ensure traceability in European food supply chain by using a blockchain system,'' in *Proc. IEEE/ACM 2nd Int. Workshop Emerg. Trends Softw. Eng. Blockchain (WETSEB)*, May 2019, pp. 40–47.

[39] S. Wang, D. Li, Y. Zhang, and J. Chen, ''Smart contract-based product traceability system in the supply chain scenario,'' *IEEE Access*, vol. 7, pp. 115122–115133, 2019.

[40] B. Yu, P. Zhan, M. Lei, F. Zhou, and P. Wang, ''Food quality monitoring system based on smart contracts and evaluation models,'' *IEEE Access*, vol. 8, pp. 12479–12490, 2020.

[41] T. H. Pranto, A. A. Noman, A. Mahmud, and A. B. Haque, ''Blockchain and smart contract for IoT enabled smart agriculture,'' *PeerJ Comput. Sci.*, vol. 7, p. e407, Mar. 2021, doi: 10.7717/peerj-cs.407.

[42] B. Haque, R. Hasan, and O. M. Zihad, ''SmartOil: Blockchain and smart contract-based oil supply chain management,'' *IET Blockchain*, vol. 1, nos. 2–4, pp. 95–104, Dec. 2021. [Online]. Available: https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/blc2.12005

[43] R. H. Von Alan, S. T. March, J. Park, and S. Ram, ''Design science in information systems research,'' *MIS Quart.*, vol. 28, no. 1, pp. 75–105, 2004.

[44] OpenZeppelin. (2020). *Openzeppelin: Contracts*. [Online]. Available: https://github.com/OpenZeppelin/openzeppelin-contracts

[45] L. Marchesi, M. Marchesi, G. Destefanis, G. Barabino, and D. Tigano, "Design patterns for gas optimization in Ethereum," in *Proc. IEEE Int. Workshop Blockchain Oriented Softw. Eng. (IWBOSE)*, Feb. 2020, pp. 9–15.

[46] X. Xu, I. Weber, and M. Staples, *Architecture for Blockchain Applications*. Cham, Switzerland: Springer, 2019.

LODOVICA MARCHESI (Member, IEEE) graduated in computer science from the University of Cagliari, in February 2018. She is currently pursuing the Ph.D. degree with the Department of Mathematics and Computer Science, University of Cagliari. She worked as a Research Grant on the "Study of Blockchain Technology Applied to Systems for Managing Complex Documents" at the Department of Electrical and Electronic Engineering, University of Cagliari. Her research interests include the application of blockchain technology in different sectors, cybersecurity for blockchain applications, the study of software engineering practices for blockchain development and distributed applications, the study of machine learning algorithms and economic models, and financials related to the cryptocurrency market.

KATIUSCIA MANNARO received the Engineering degree *(summa cum laude)*from the University of Cagliari, Italy, in 2001, the master's degree in internet banking from the University Cattolica of Sacro Cuore of Milan, in 2002, and the Ph.D. degree in electronic engineering and computer science with a thesis on "Adopting Agile Methodologies in Distributed Software Development," in 2008. Since then, she has actively continued her master's research. In 2003, she won a Scholarship for Young Researchers in FIRB Project (MAPS-Agile Methodologies for Software Production). She worked as a Postdoctoral Fellow at the Department of Electrical and Electronic Engineering (DIEE), University of Cagliari, from 2010 to 2017, and since 2019, she has been a Postdoctoral Fellow at the Department of Mathematics and Computer Science, University of Cagliari. She has teaching experience as a Lecturer in the field of the software engineering and her research is published in a large number of conference proceedings, articles in books, and international journals. Her research interests include blockchain technologies, software modeling, agile and lean methodologies, and smart cities.

MICHELE MARCHESI (Senior Member, IEEE) received the degree in electronic engineering from the University of Genova, in 1975. He has been a Full Professor with the Faculty of Engineering, University of Cagliari, since 1994. Since 2016, he has been a Full Professor at the Department of Mathematics and Computer Science, University of Cagliari, where he teaches software engineering courses. He has authored over 200 international publications, including over 70 in the magazine. He has been one of the first in Italy to deal with OOP, since 1986. He was a Founding Member of TABOO, the Italian association on object-oriented techniques. He worked on object analysis and design, UML language, and metrics for object-oriented systems since the introduction of these research themes. In 1998, he was the first in Italy to deal with extreme programming (XP) and agile methodologies for software production. He organized the first and most important world conference on extreme programming and agile processes in software engineering, Sardinia, from 2000 to 2002. Since 2014, being among the first in Italy, he has extended his research interest to blockchain technologies, obtaining significant results in the scientific community.

ROBERTO TONELLI (Member, IEEE) received the Ph.D. degree in physics and the Ph.D. degree in computer engineering, in 2000 and 2012, respectively. He is currently a Temporary Researcher and a Professor with the University of Cagliari, Italy. His research has been the study of power laws in software systems within the perspective of describing software quality. Since 2014, he has been extended his research interest to blockchain technology. His research interests include widespread and multidisciplinary. He received the Prize for the top-50 most influential papers on blockchain in 2018, in January 2019 from Blockchain Connect Conference, San Francisco, CA, USA.