Journal of Integrated Design and Process Science 26 (2022) 103–129
DOI 10.3233/JID-210021
IOS Press

Blockchain-based digital trust mechanism: a use case of cloud manufacturing of LDS syringes for COVID-19 vaccination

Trupti Rane and Jingwei Huang*
Department of Engineering Management and Systems Engineering, Old Dominion University, 2101 Engineering Systems Building, Norfolk, VA, USA

Abstract. Trust is essential in the digital world. It is a critical task to build digital trust for the ongoing digital engineering transformation. Aiming at developing a blockchain-based digital trust mechanism for Cloud Manufacturing or Manufacturing-as-a-Service (MaaS), in this paper, we use the manufacturing of low dead space (LDS) medical syringes through Cloud Manufacturing as a motivating scenario to develop a basic framework. To meet the need of optimally saving COVID-19 vaccine doses to save more lives, the medical device manufacturing community needs to make a swift move to meet the surged need for LDS syringes. Cloud Manufacturing is a form of emerging Digital Manufacturing facilitated with Cloud/Edge Computing, the Internet of Things, and other digital technologies. Cloud manufacturing allows quickly establishing a digital virtual enterprise that pools together various manufacturing resources worldwide to meet the surged needs of products and save cost and time. Trusting the product quality and safety is a significant challenge when using Cloud Manufacturing to manufacture the products. This paper proposes a schema of blockchain-based digital trust mechanisms with examples of using Cloud Manufacturing of medical LDS syringes for the urgent needs of catering COVID-19 vaccination.

Keywords: Digital trust, blockchain, smart contract, ethereum, internet of things, trust mechanisms, manufacturing-as-a-service, cloud manufacturing, cyber manufacturing, digital engineering, digitalization, COVID-19, medical syringes

1. Introduction

The fast progress in technologies such as the Internet of Things (IoT), Cloud Computing, Big Data, Blockchain, AI&ML, 3D Printing, Cyber-Physical Systems, and others have been changing the technological landscape of engineering processes from design to manufacturing, maintenance till retirement. The streams of those technological changes have been converged, leading to revolutionary Digital Engineering Transformation with "digitalization" at its core (Huang et al., 2020). As a representative manifestation of Digital Engineering, Cloud Manufacturing (CMfg), with other similar names such as manufacturing as a Service (MaaS), Cyber Manufacturing (NSF, 2020), and others, is leading the way towards digitalizing and virtualizing manufacturing service, products, manufacturing processes, and manufacturing resources. With concepts and models inspired from Cloud Computing, "Cloud manufacturing is a customer-centric model that exploits on-demand access to diversified and distributed manufacturing resources to form temporary, reconfigurable production lines with enhanced efficiency, reduced product lifecycle costs, and allow for optimal resource loading in response to variable-demand customer-generated tasks (Wu et al., 2013)." CMfg represents an evolution of networked and service-

*Corresponding author: Jingwei Huang. Department of Engineering Management and Systems Engineering, Old Dominion University, 2101 Engineering Systems Building, Norfolk, VA, USA. E-mail: j2huang@odu.edu.

oriented manufacturing models that comprise a pool of shop floor reconfigurable and interchangeable items and may access a shared pool of computing devices according to cloud computing (CC) principles. Integrating high-level SaaS cloud models with CMfg models at the production level allows for service-oriented product development and mass customization. Customers can order, configure, select, and use customized resources and services, ranging from computer-aided engineering software tools to after-sales services. Resource virtualization also stimulates a product-centric approach, in which the product directly requests processing, assembling, and handling from available providers while it is in the execution, delivery, and use stages. It helps dynamically balance the optimized control with a long-term global view and the rapid local reaction abilities to unexpected events. Among all the technologies, cloud computing and IoT deeply influence the development of cloud manufacturing. With the advent and wide acceptance of IoT in technologies like Smart Homes, smart cities, health care, security surveillance, IoT has also found its way into manufacturing through smart technology in the form of RFIDs and various sensors. IoT has had a vital role in enabling real-time machine status monitoring for both confirming resource availability and monitoring resources using sensors for critical failure notification as well as preventive servicing.

As one of the oldest industrial sectors, manufacturing is typically physical. So, when it goes to cyberspace, trust becomes a critical issue. There is an obvious need for concrete trust in cloud-based manufacturing processes such as manufacturing the goods, finances flow, and conceptualization of connections. Traditionally, a client interacts with manufacturers which have a physical presence of manufacturing facilities and have centralized management of the manufacturing process of a product. Depending on the types of products, if necessary, a client may physically visit the manufacturing facilities or send a representative to monitor the manufacturing process. With Cloud Manufacturing, the physical interactions between a client and a manufacturer become limited; the manufacturing process becomes distributed and dynamically configured. Therefore, trust issues concerning product quality, product security, the intellectual property right management of product design, and others become significant concerns. Like the transition from in-house computing to cloud computing, people will gradually build trust in Cloud Manufacturing and accept the new concept and technology. New digital mechanisms of trust are needed to facilitate building trust. The research question is: What are the digital mechanisms of trust and solutions that would foster trust in Cloud Manufacturing?

Towards this direction, we start with a specific motivating use case to identify the main problems and develop concepts and the solution framework. The motivating scenario is about the surged shortage of Low-Dead Space (LDS) medical syringes, which can save an extra dose of vaccine in each vial, thus vaccinating more people. It is crucial in the current situation with a shortage of vaccines worldwide. More details about the motivating scenario will be presented later in section 3. From the motivating scenario, we identify major trust concerns, significant parties involved, identity management, and information flow between them that increases trust. With the advance of disruptive digital technologies of Blockchain and the Internet of Things in mind, our research goal is to develop a blockchain-based digital trust mechanism to enable archiving and auditing IoT monitored manufacturing processes and environment in Cloud Manufacturing.

We use Ethereum smart contract as the mechanism for business transactions and for information transparency, which provides immutable manufacturing process data as evidence to support trust judgment. We present a prototype implemented with Remix to demonstrate how the proposed approach works in the motivating use case.

We have organized the content of this paper as follows. After this introduction, we discuss the related research in section 2. Section 3 presents our motivating scenario of cloud manufacturing of LDS medical syringes for meeting urgent needs of COVID-19 vaccination. Section 4 presents our technical approach to blockchain-based trust mechanisms. Then, section 5 presents our experiments with Remix. Finally, in section 6, we discuss what we learned from the case and conclude this paper.

2. Related research

2.1. Cloud manufacturing

Many researchers have made valuable contrition to facilitate and enable the understanding of Cloud manufacturing (CMfg). The paper (Molina et al., 2007) reviews different traditional manufacturing models, highlights the gaps in these models, and sets the context for implementing the "Build to Order"(BTO) paradigm to address these gaps. The paper introduces the concept of "Virtual enterprise," with the capability to collectively manufacture for a customer with individual manufacturers who only own some of its competencies. It set the stage for CMfg. The papers (Li et al., 2010), (Tao et al., 2011), (Wu et al., 2013) contribute significantly towards the emergence of CMfg as a strategic manufacturing model in the early 2010 s.

In the paper (Mell et al., 2011), the authors provide the NIST's definition of Cloud Computing(CC). Following this definition, paper (Xu, 2012) defines CMfg as "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable manufacturing resources (e.g., manufacturing software tools, manufacturing equipment, and manufacturing capabilities) that can be rapidly provisioned with minimal management effort or service provider interaction." The author focuses on how CC is part of the CMfg model on the IaaS (Infrastructure as a Service) abstraction level. The paper emphasizes how adopting CMfg in the IaaS model at the production layer of enterprises (with high production volumes, variable batch sizes, and frequently changing product types) is necessary for the smart digital factory of the future.

In the recent past, many studies have focused on the digital transformation of manufacturing through the cloud. Paper (Borangiu et al., 2019) explores how digital transformation through cloud services and resource virtualization has changed traditional manufacturing and can facilitate the sustainability and maintainability of manufacturing. The authors attribute two crucial drivers that speed up the Digital transformation in manufacturing:

- **Resource virtualization:** Resource virtualization relates to the capability of creating and managing virtual machines (VM). Virtualization allows decoupling a set of physical computing resources or manufacturing resources from its use, thus permitting easy migration of a workload to another resource during its execution.
- **Cloud services:** Cloud services are a crucial attribute of digital manufacturing because they facilitate Direct Digital Manufacturing (DDM). DDM is rapid manufacturing using additive manufacturing techniques such as 3D printing.

Another paper (Barbhuiya et al., 2019) that highlights exciting research in the CMfg domain is related to the SmartMaaS framework. The paper demonstrates SmartMaaS via a prototype that can accept customers' product requests in design genes and manufactures 3D printed products using an actor-based system. It highlights how SmartMaaS can enable Smart Manufacturing by facilitating rapid turnaround time, product quality, and innovative design.

The paper (Zhang et al., 2014) highlights terms like Agile Manufacturing, Concurrent Engineering, Networked Manufacturing, Manufactured Grid, and Crowdsourcing. These terms collectively give way to what we know today as cloud manufacturing which leverages these concepts to provision manufacturing resources through the cloud. Research work in papers (Fisher et al., 2018), (Li et al., 2010) and (Zhu et al., 2020) point to a variety of advanced manufacturing systems (AMS) such as Computer Integrated Manufacturing (CIM), Flexible Manufacturing (FM), and networked manufacturing (NM). The paper reinforces how CMfg is an amalgamation of the concepts from several of these advanced manufactured systems with addressing the bottlenecks of each of these systems through cloud computing and IoT.

The papers (Caggiano et al., 2016) and (Zhong et al., 2017) introduced Smart monitoring as one of the critical factors for the success of a cloud-based manufacturing system. The paper (He & Xu, 2015) provides the leading influential technologies like cloud computing and IoT, which depend on high-performing computing (HPC) solutions that use supercomputers and computer clusters to handle multiple tasks at high speed. It also highlights a Service-Oriented-Architecture (SOA) for on-demand resource allocation. In papers (Caggiano et al., 2016) and (Lee et al., 2016), the authors focus on Internet-of-things-based sensors that help identify issues and provide operational data that can be used in predictive analytics to prevent issues occurring. IoT sensor-based monitoring is interchangeably also known as Smart Monitoring. The paper (Wu et al., 2013) provides the strategic vision for cloud manufacturing across the consumers/users, application providers, physical resource providers, aka manufacturers, and provides further insights into a Cloud-manufacturing approach's fundamental characteristics. The paper highlights the key issues that Cloud manufacturing as an industrial practice faces and how digital trust is one of the central focus areas where further research is necessary.

2.2. *Trust mechanisms*

Digital trust is one of the driving criteria for the success of a system, the components of which interact through the cloud. It is also one of the significant challenges faced by such systems. The paper (Liu et al., 2019) deep dives into some of the significant challenges of digital trust in Cloud Computing(CC). One of the main challenges that the paper highlights are that most enterprises have reservations to put their critical data and application in the cloud. Paper (Xu, 2012) highlights the trust and security concerns due to the anxiety of having sensitive data outside the physical perimeter of the organization. The key takeaway from the paper is that when CC is involved, establishing digital trust is an added responsibility of the service provider along with providing the actual service. Along with the service provider, the Original Equipment Manufacturers (OEM) and other stakeholders also need to have a digital trust established.

The paper (Yan et al., 2016) provides the criteria for trust evaluation on how the digital/cloud manufacturing space conducts direct and indirect trust evaluations. It provides a framework for quantifying the trust and formulates a model of comprehensive trust evaluation with the help of a case study. The paper (Wu et al., 2018) discusses the cybersecurity-related challenges cloud manufacturers and service providers face. How do we ensure that the integrated systems are secure when we introduce advanced technology into a legacy system and retrofit it with sensors and IoT devices? The paper identifies Intrusion detection, Authentication, Encryption, and Access control as the four control mechanisms to counter cybersecurity-related issues and concerns in the domain. With virtualization, the primary security issues include data leakage because multiple third-party manufacturers share physical resources, identity and access management, the physical protection of virtual resources, and the prevention of cross-virtual machine channel attacks.

In (Xu, 2012), the author presents how cloud computing provides a framework to connect distributed resources shared across enterprises. When manufacturing uses Cloud computing, it generates a very high volume of operational data due to the simulation, scheduling, monitoring, and optimization of cloud manufacturing services. However, with increased operational data comes issues with data governance and analytics of big data such a prediction analysis, analytics-based decision making, and edge computing. The paper (Y. Lu & Xu, 2019) presents how big data analytics can use a digital twin of manufacturing equipment connected via the cloud.

Paper (Umeda et al., 2015) discusses requirements and strategies to deal with intellectual property protection and management in cloud manufacturing. With a cloud model, the customers may not trust the service provider or third-party manufacturers unless precise protocols and regulations are in place to manage stakeholders' intellectual property. There are many challenges with the lack of central governance due to geographically distributed stakeholders, such as how conflicts are regulated and how all stakeholders involved in the system adhere to regulatory compliance (Helo et al., 2014), (Moghaddam et al., 2019), and (Buterin, 2014) (Yadekar, Yaser; Shehab, 2013).

Provenance is an important type of evidence to gain trust. In recent years, research on blockchain-based product provenance for improving product traceability in the supply chain has received much attention (Q. Lu & Xu, 2017; Suhail et al., 2020). A single consignment may be sent to a customer through multiple shippers or manufactured by multiple OEMs and consolidated and shipped together. So, it is crucial to establish where each item is coming from, which plant produces them, the raw material, and whether the manufacturer followed suitable standards.

In (Huang & Nicol, 2013), the authors examined the existing trust mechanisms used in Cloud Computing practice. It proposes a framework that integrates a broad range of trust mechanisms like formal mechanisms (such as accreditation, audit, and attribute certification & validation) and informal mechanisms (interaction experience-based, reputation and recommendation-based, self-assessment and revealing, and transparency-based). The transparency-based trust mechanism demonstrated in this paper is particularly relevant for trust concerns in cloud manufacturing. As addressed in (Huang & Nicol, 2013), trust is a mental state comprising (1) expectancy: the trustor expects a specific behavior of the trustee such as providing valid information or effectively performing cooperative actions; (2) belief: the trustor believes that the expected behavior will occur, based on the evidence of the trustee's competence, good intention, and integrity; (3) willingness to take a risk: the trustor is willing to take the risk for (or be vulnerable) that belief in a specific context, where there is an expectation for the specific behavior of the trustee. Cloud Security Alliance (CSA) operates the STAR (Security, Trust & Assurance Registry) program (CSA, n.d.), which allows free public access to cloud service providers' self-assessment about their security control against the most frequently asked questions and the best practice of cloud services. CSC.com proposed, and CSA adopted the Cloud Trust Protocol (CTP) (CSA, 2011), which is a representative transparency-based trust mechanism, allowing cloud users to request the response of a cloud service provider on some specific information about configuration, vulnerability, audit log, service management, service statistics, and others, collectively called "elements of transparency." "The primary purpose of the CTP and the elements of transparency is to generate evidence-based confidence that everything claimed to be happening in the cloud is indeed happening as described,..., and nothing else" (Knode & Egan, 2010). Cloud Manufacturing can borrow those ideas and create transparency-based trust mechanisms for this field.

2.3. *Blockchain for cloud manufacturing*

Since the mysterious "Nakamoto" published the white paper Bitcoin (S. Nakamoto, 2008), blockchain has gone much beyond final services and found applications in many fields (Wang et al. 2019), such as Internet of Things, supply-chain, and cloud manufacturing, for its strong capability to conserve data integrity. The concept of smart contracts was first proposed by Nick Szabo (N. Szabo, 1996). Ethereum is perhaps the first or the most popular blockchain system running smart contracts. In Ethereum, smart contracts are expressed with language Solidity, which is a Turing-complete programming language. The powerful implementation of smart contracts in Ethereum makes it be widely used in many areas (Zheng et al., 2020; K. Christidis & M. Devetsikiotis, 2016, Mohanta et al., 2018). In the following, we discuss some research on incorporating blockchain into the cloud manufacturing industry.

The papers (A. Vatankhah Barenji et al., 2018) and (Rožman et al., 2021) highlight how blockchain can implement the cloud manufacturing framework at the shop floor and machine level. It provides an alternate approach to cloud computing and capacity allocation via blockchain. The papers attempt to project the existing gaps in cloud manufacturing around security along with information centralization and evaluate blockchain technology as a two-level peer-to-peer network to implement cloud manufacturing. The paper (Zhu et al., 2020) coins the term BBCM, which stands for blockchain-based Cloud manufacturing, and proposes an operation model for CMfg based on blockchain. The authors propose a shift in the operational model for cloud manufacturing from the traditional internet and service-based model to a consensus-based blockchain model.

The paper (R. Vatankhah Barenji, 2021) coins a new term for blockchain-based trust called Blocktrust to address the trust problem in cloud manufacturing. In Blocktrust based cloud manufacturing, trust in a private peer-to-peer network is maintained by having trust scores for each node. The core idea of this study is to establish trust using a feedback value assigned to every provider and requester in the network using blockchain. While this works very well in a private blockchain, addressing some challenges is necessary to implement it for a public blockchain.

The paper (Liang et al., 2017) provides valuable insights on how blockchain can provide data provenance on cloud data objects. Even though it does not directly refer to cloud manufacturing, the virtualized manufacturing artifacts are cloud data objects. A data provenance architecture named ProvChain is proposed to provide a tamper-proof environment for cloud data objects and uses blockchain receipt to validate every provenance data entry. This framework can work in a cloud-based application with the standard number of data reads but storing all the movements in a blockchain with large datasets might get challenging.

There is other exciting research around diverse use cases of blockchain in cloud manufacturing. The paper (Zhu et al., 2020) highlights concepts from blockchain such as virtual currency, smart contract, consensus algorithm, and HASH256 encrypted technology to realize automatic consensus-driven services. Furthermore, it demonstrates the concepts with a case study on 3D printing using blockchain-based cloud manufacturing.

3. Motivating Scenario: Shortage of low dead space syringes for saving COVID-19 vaccination doses and a cloud manufacturing solution

3.1. Scenario description

We propose the potential use of cloud manufacturing of LDS syringes to save extra vaccine doses in fighting the COVID-19 pandemic as our motivating scenario. Vaccine suppliers deliver vaccines with at least one extra dose in each vial. However, LDS syringes are needed to get that extra dose in each vial. While administering the vaccines, healthcare providers realized that the supplied vials came with at least one extra dose, which the standard syringes were incompetent to extract because of design limitations. The standard syringe that came with the vials holds 3 ml of vaccine after injecting, but a thinner, 1-ml syringe called “low dead-volume” or “low dead-space” syringe works well for small doses of vaccines, like that of Pfizer and Moderna. The supply of these 1-ml syringes is currently way lower than the demand (Hufford, Austen; Hopkins, 2021). A few hospitals upgraded their syringes but not all, leading to wastage of the precious vaccine doses. While the Pfizer kits are now coming with a suitable syringe, the Moderna kits still have the standard ones. Per stats, since the US has bought about 300 million Moderna doses so far, there is a high possibility of wastage of around 30 million doses by not enabling vaccinators with the suitable syringe. Also, this issue is not limited to the US. Several countries like Japan and other European countries have reported similar supply chain problems with the low dead volume syringes (Moutinho, 2021).

With so many countries struggling with vaccine supplies for mass vaccinations, the syringe shortage is a significant problem that needs immediate attention. Thus, bringing forth the significance of advanced manufacturing and further strengthening the need for digital transformation in manufacturing with modern practices like Cloud Manufacturing. With Cloud manufacturing, addressing the supply shortages of the syringes needs effective collaboration between multiple manufacturers who can cater their manufacturing artifacts and resources by virtualization and consumption on-Cloud. A cloud manufacturing system can tap into any available third-party manufacturing resources to address the manufacturing bottlenecks for low dead volume syringes. Multiple manufacturers can collaborate and manufacture syringes for a single service provider to speed up the process. However, for cloud manufacturing to work out and thrive, digital trust is essential. We aim to develop a digital trust mechanism by using smart contracts in an Ethereum blockchain.

This paper demonstrates the implementation of trust in cloud manufacturing with an example of LDS syringes. We will be using a smart contract leveraging the Ethereum blockchain. It will be a public blockchain with a smart contract between the manufacturer operating in a cloud manufacturing model. The potential customers will be service providers and end customers like hospitals and regulatory bodies such as the FDA (who regulate the medical devices' manufacture).

Let us assume a scenario of cloud manufacturing to meet the surged needs of LDS medical syringes for the COVID-19 vaccination. To focus on the valuable insights, we use a simplified supply-chain relation as illustrated in Fig. 1. From the figure, we can identify several vital roles, such as

Regulatory agency: e.g., FDA(G): Regulatory agencies oversee the manufacturing of medical devices such that their consumption is safe for the general public. E.g., In the USA, the FDA's Center for Devices and Radiological Health (CDRH) regulates all firms that manufacture, repackage, relabel, or import medical devices sold in the country. The FDA protects the public health by having regulations for human and animal drugs and biologics, medical devices, tobacco products, food including animal food, cosmetics, and electronic products that emit radiation. FDA classifies both the LDS (low dead space) piston syringes and LDS hypodermic needles as Class II Medical Devices. In Class II devices, general controls alone cannot provide considerable safety assurance and effectiveness. Class II devices have a moderate to high risk to the patient or user.

Medical syringe supplier/distributor(S): The manufacturer distributes products to medical facilities. He will assemble the individual parts before distributing them to its end users.

Third-party manufacturers (B, N): e.g., Syringe barrel manufacturer, syringe needle manufacturer. These are the participant manufacturers in cloud manufacturing who collaborate towards the manufacturing of the LDS syringes.

Raw material supplier (Mi): These are the suppliers that supply the raw materials such as heat-treatable stainless steel or carbon steel for the needles and flexible synthetic rubber for the plunger heads, and biocompatible plastics or glass for the syringe body

Healthcare service provider(H): These are the direct consumers of the syringes who administer mediation through these syringes to the patients.

3.2. Potential trust issues in cloud manufacturing of LDS Syringes

3.2.1. Identity management

In the traditional manufacturing of medical devices, the identity establishment and validation are relatively straightforward because the manufacturer is the central entity who procures supplies, manufacturers goods, and then hands over to its consumers, who could be distributors or medical facilities. The manufacturers will have contractual agreements with suppliers that they trust either through previous agreements or quality inspections of samples, and the manufacturers know that they are interacting with the said supplier. When they interact with potential consumers, be it distributors or medical facilities, they can physically or electronically validate the identification information relatively quickly. When there is an inspection from a regulatory body like FDA, they know for sure through easy validation that they are dealing with the correct entity. It is not the case with cloud manufacturing. The

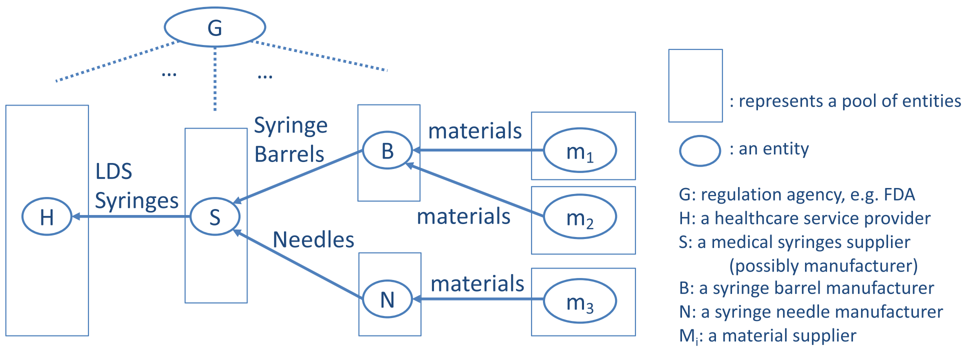

G
...
...
H
LDS Syringes
S
Syringe Barrels
Needles
B
N
materials
materials
materials
m_1
m_2
m_3
: represents a pool of entities
: an entity
G: regulation agency, e.g. FDA
H: a healthcare service provider
S: a medical syringes supplier (possibly manufacturer)
B: a syringe barrel manufacturer
N: a syringe needle manufacturer
M_i: a material supplier

Fig. 1. Scenario of LDS medical syringe manufacturing.

manufacturer working on a consignment may not be in the country as the product consumption country. When a distributor from that country requests a consignment, how does the manufacturer ensure that he communicates with the same entity they are claiming to be? When a regulatory body requests information, how do they validate the identity and access permissions?

3.2.2. *Material used for manufacturing*

The raw materials of the syringe should be compatible with injection fluids. ISO/FDA does not have specific material requirements, and the material depends on manufacturers' design, process, and sterility processes. However, a particular manufacturer may have specific standards per contractual agreement with customers or maintain standardization across its consignment manufactured in a distributed setting. In this case, how does the service provider guarantee that all manufacturing partners use the same material for all the products/parts, and they comply with the dimensions, shape, consistency of the manufactured devices? Since the manufacturing of thousands of parts happens daily, a complete inspection is impossible. Line inspectors can randomly check components at fixed time intervals to ensure they meet material size, shape, and consistency specifications but not the entire consignment.

The manufacturers in traditional manufacturing, in most cases, have contractual agreements with suppliers that they trust either through previous agreements or quality inspections of samples. The manufacturers themselves are distributors who ensure that their desired quality raw materials go into the manufacture of their products, so they are directly responsible for the quality of the product, in our case, the LDS syringes. It is not the case in cloud manufacturing. The manufacturer is not the distributor of the end-product, and even though they are responsible for the quality of the end-product, their end customers, who are the distributors or medical facilities, may have trust issues on raw materials used for manufacturing.

3.2.3. *Transparency and provenance*

Transparency and Provenance are critical in the manufacturing of medical devices. In traditional manufacturing, establishing transparency and Provenance is relatively simple as there is a single source for all units of products distributed by this manufacturer. These dynamic changes in cloud manufacturing as multiple manufacturers are involved in fulfilling a consignment, giving rise to an urgent need to establish a provenance mechanism for cloud manufacturing, especially medical devices.

Class II devices have mandatory performance standards, labelling requirements, and post-market surveillance. Post Market surveillance is an integral part of the science of pharmacovigilance that involves monitoring a medical device after its release to the market. As part of this surveillance, consumers can report adverse reactions to drugs and medical devices. The FDA may also carry out active surveillance and monitor the safety and efficiency of medical devices through a Post-Approval Study. The service provider must trace any post-market surveillance findings to the third-party manufacturers who sourced the product in cloud manufacturing. How can the service provider trace the devices back to individual partners who manufactured them?

3.2.4. Adhere to current good manufacturing practices (CGMPs) from FDA

CGMP assures the quality of products by having manufacturers efficiently and suitably control their operations. The manufacturers of LDS Syringes should adhere to class II Current Good manufacturing practices from FDA and should be able to prove the adherence when inspected. It includes procuring the correct quality raw materials, instituting robust quality management systems, operating procedures, identifying and investigating product quality deviations, and maintaining reliable testing laboratories. All parts of the LDS Syringes should be manufactured, assembled, and sterilized by recognized national or international codes of good manufacturing practice for medical devices.

In traditional manufacturing, manufacturers produce the products in their premises in manufacturing units that they have complete autonomy of, both physically and for controlling the operations. So, complying with CCMP is an easy task. However, it is not the case in cloud manufacturing. How can service providers trust third-party manufacturers, some of whom could be outside the geographic boundaries of FDA regulations, to adhere to these regulations? Some relevant parts of CGMP include the following aspects.

(1) Sterility

The sterility of LDS syringes is vital since they need to be free from disease-causing agents. They are usually to be packaged individually in airtight plastic. Several syringes are subsequently packed into boxes, stacked on pallets, and shipped to distributors. For manufacturing medical devices like LDS syringes, sterility is a vital part of the manufacturing process. The products need to be free from disease-causing agents. As part of regulations in most countries, it is compulsory to demonstrate sterility in manufacturing conditions if inspected. For cloud manufacturing to work for this use case, equipping the service providers to easily verify that sterility was maintained during the manufacturing process by all the third-party manufacturers involved is necessary.

Sterility should also be demonstrated to the regulatory bodies when requested or inspected. This aspect is crucial for establishing digital trust. Few examples for established sterilization methods for Category A and B devices are:

Established category A sterilization methods: Dry heat, EO with devices in a fixed, rigid chamber, Moist heat or steam and Radiation (e.g., gamma, electron beam)

Established category B sterilization methods: Hydrogen peroxide (H2O2), Ozone (O3), Flexible bag systems (e.g., EO in a flexible bag system, diffusion method, injection method)

Novel Sterilization method: A Novel Sterilization Method is a method that FDA has not reviewed and determined to be adequate to sterilize the device for its intended use effectively.

Examples of novel sterilization methods: Vaporized peracetic acid, High-intensity light or pulse light, Microwave radiation, Sound waves, and Ultraviolet light

(2) Submission of 510(k) to demonstrate substantial equivalence

In specific scenarios in the manufacturing of Class II devices, A 510(k) submission is necessary as per protocol. It is necessary for the demonstration of substantial equivalence to another legally US marketed device. Substantial equivalence means that the new device is as safe and effective as the predicate. A device is substantially equivalent if, in comparison to a predicate, it

- has the same intended use as the predicate; and
- has the same technological characteristics as the predicate.

or it

- has the same intended use as the predicate; and
- has different technological characteristics and does not raise different questions of safety and effectiveness; and
- the information submitted to FDA demonstrates that the device is as safe and effective as the legally marketed device.

A device may not be marketed in the US until the submitter receives a letter finding the device substantially equivalent. Class II Syringes and Needles are - Not eligible for a third-party review of 510K (FDA, 2020a). For medical devices, FDA classifies them according to their assessed risk. The highest-risk devices (Class III) require the FDA's premarket approval application before marketing. Manufacturers have to demonstrate a reasonable assurance that the devices are safe to use and effective to get the device's approval. For moderate-risk medical devices (Class II), marketing is cleared by the FDA once the manufacturer demonstrates that it is substantially equivalent to a legally marketed predicate device that does not require premarket approval. Devices that present a low risk of harm to the user (Class I) are subject to general controls only, and most are exempt from premarket notification requirements.

FDA can inspect manufacturing facilities worldwide, including facilities that manufacture active ingredients and medical devices. In most cases, it relies upon reports of potentially defective drug products from the public and the industry to identify sites for which an inspection or investigation is needed (FDA, 2017).

Establishment registration: When FDA registers a business, it will not issue a Registration Certificate to the business. FDA only issues a registration number (FDA, 2021). FDA registration means a registration of the business rather than the registration of the devices (FDA, 2020b). Owners or operators of places of business (also called establishments or facilities) involved in the production and distribution of medical devices intended for use in the United States (US) are required to register annually with the FDA. This process is known as establishment registration.

3.2.5. Service provider patent

The service provider may hold the patent for the syringe design. In which case, others cannot produce it. In a distributed scenario with individual manufacturing of parts, how does this work? How does a service provider with a patent protect its intellectual property and trust third-party manufacturers who may be in a different country? How to work around the difficulty in detecting infringement in patents in a cloud manufacturing scenario?

3.3. System concept of CMfg for LDS syringes

As a potential solution focusing on the data provenance and adherence to the current good manufacturing practices (CGMP) in cloud manufacturing, we propose an IoT-enabled cloud manufacturing system setting that deploys a smart contract to record the readings of the IoT devices into the immutable blockchain. The IoT devices monitor the manufacturing conditions and record them so that no tampering is possible. Anyone requiring access to this information can traverse through the blockchain to get it and consume it as required. In the Cloud Manufacturing environment, we have the following assumptions for our motivating scenario.

- IoT devices equipped in the manufacturing facility of medical syringes, or their parts monitor the environmental status of concern in real-time.
- Edge computing is deployed at the manufacturing site to preprocess the real-time streaming sensor data.

- Cloud services perform the following tasks:
  - Provide cloud-based manufacturing computing services.
  - Archive the data processed by edge nodes into a cloud repository.
  - Generate a hash for each file of archived data.
  - Produce the digest (represented as a statistical summary) of real-time manufacturing status.
  - Store the status digest and associated hashes of data files into a distributed ledger– blockchain.

Each IoT device identifies by a specific code and connects to a computer connected to the blockchain. The IoT device sends the sensor data to the computer that prepares and logs the transactions into the blockchain. It also enables demonstrating them to potential customers who can now confidently order products such as medical devices from this manufacturer once this manufacturer demonstrates the sterility in its manufacturing process.

4. Approach

4.1. Identity management

For Identity management, an Identity Trust Fabric (ITF) blockchain, which will act as a distributed ledger for cryptographic proofs for decentralized identities, can be built. This ITF blockchain will provide verifiable credentials GTID (Global transaction ID) and facilitate decentralized trust between the entities involved, including the credential issuers, holders, service providers, and consumers, thus eliminating the need for a centralized authority. The ITF Blockchain will use Decentralized Publickey Infrastructure (DPKI) for identity management. The idea is to build a Self-Sovereign Identity (SSI) where individual stakeholders can own and manage their digital identities. Upon legally establishing a decentralized identity, we can verify the enrollment of service providers within the ecosystem. (Hewett et al., 2019) present the concepts of Digital Identity and their implementation using blockchain for the supply chain industry. The same concept extends in our use case for identity management.

Authentication: In one simplistic example, a person creates a pair of private and public keys in an identity wallet. An ITF stores the public key (identifier) hash immutably. While adding the transaction, the entity in the block transactions gets assigned an auto-generated identifier or Smart-ID by the smart contract. This ID is saved in the blockchain with the attributes and serves as a GTID.

The ITF also stores the certification record. If the user wants to access a service, it is enough to present its identifier/Smart-ID as a QR code or within a token. The service provider verifies the identity by comparing the hash values of identifiers with their corresponding hash records in the ITF. If they match, ITF grants the access. It can be a separate smart contract that service providers can query, partners to validate identity, or integrated with the transaction management with independent, smart-contract functions implementation.

4.2. Manufacturing transparency and provenance management

In manufacturing medical devices, especially Class I and Class II, sterility is of utmost necessity. The idea here is to use IoT sensors to monitor the sterility determinants and store these manufacturing conditions in the blockchain to demonstrate adherence to sterile manufacturing. Once the blockchain stores the data, it is immutable and can be easily validated after the fact. It enables demonstrating these conditions to the authorities like the FDA when requested or in an inspection.

IoT sensors are used in various applications to record diverse readings such as temperature, humidity, and chemical concentration, among others. However, with the current scheme of things, it might not be possible to connect all IoT devices directly to blockchain since they are not as large as conventional computers, and there have been challenges with establishing a consensus with IoT devices in a

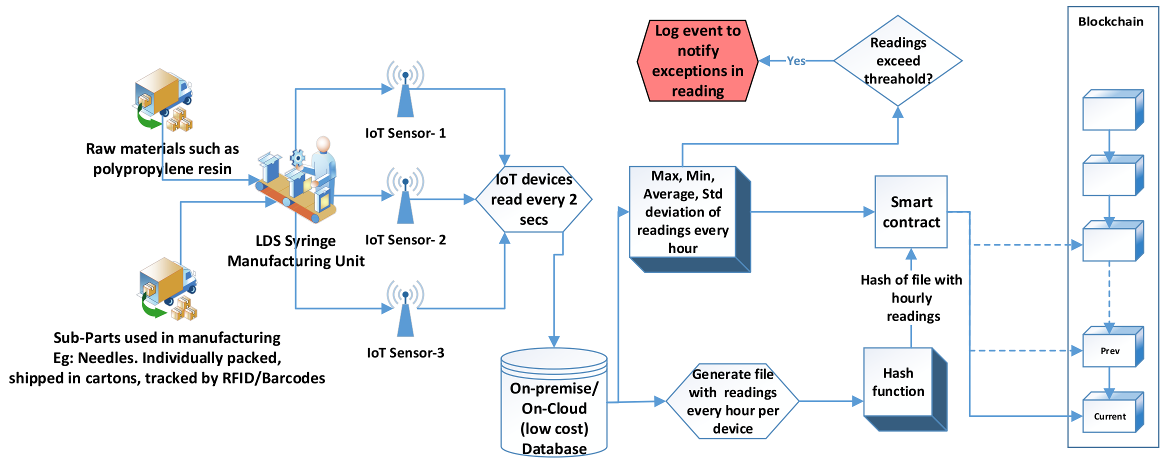

Raw materials such as polypropylene resin
LDS Syringe Manufacturing Unit
Sub-Parts used in manufacturing Eg: Needles. Individually packed, shipped in cartons, tracked by RFID/Barcodes
IoT Sensor- 1
IoT Sensor- 2
IoT Sensor-3
IoT devices read every 2 secs
On-premise/ On-Cloud (low cost) Database
Max, Min, Average, Std deviation of readings every hour
Log event to notify exceptions in reading
Yes
Readings exceed threahold?
Smart contract
Hash of file with hourly readings
Generate file with readings every hour per device
Hash function
Blockchain
Prev
Current

Fig. 2. Data flow; cloud manufacturing of LDS syringes with blockchain.

blockchain. The consensus mechanisms we have today are comprehensive for IoT devices that have limited computing. IoT devices are heterogeneous, and Scalability is also one of the challenges as the size of blockchain proliferates, especially in IoT systems. Also, not all data generated by IoT devices can be stored on an Ethereum blockchain mainly because generating these many transactions is infeasible both from a storage standpoint and economically. It could make the maintenance of blockchain expensive and extremely slow.

Hence, we first need to establish what information the transaction management blockchain should store such that the data integrity can be confirmed without mirroring all the IoT readings into the blockchain storage. It is worth reiterating that blockchain is not a database but can be conceptualized more like a ledger.

- Let us assume that the IoT device in consideration sends out the reading every 2 seconds for simplicity and demonstration purposes. This device is placed in the manufacturing unit to get an accurate reading of the measure conveniently. This measure could be a reading of temperature, humidity, or concentration of a particular chemical that the manufactured devices use during the sterilization process. If some data need access control, use Role-based Access Control (RBAC) as conditions of the smart contract.
- One reading every 2 seconds implies having 30 readings per minute and 1800 readings per hour.
- The manufacturer has a dedicated private database, either on-premises or on-cloud, that gets fed this data. The data will be the operational data from the manufacturer's IoT devices stored on a low-cost database. The retention period of this data can be configurable based on the average lifespan of the manufactured product. For example, for an LDS Syringe, if the expiry date could be five years from the manufacturing date, data up to 5-6 years will warrant enough coverage for FDA inspections until the last unit from that batch is either consumed or invalidated due to expiry.
- The database logs the stream of readings between the start and end time in a file. It could be a simple flat file with 1800 entries of timestamp and the corresponding reading per device. The blockchain then stores the hash of this file.
- Along with this hash, a single blockchain transaction also stores the max, min, average, and standard deviation of all the 1800 data points. The database also maintains the maximum, minimum, and average reading thresholds. If readings breach these thresholds, the smart contract triggers an event to record an exception in the sterility condition on the blockchain.

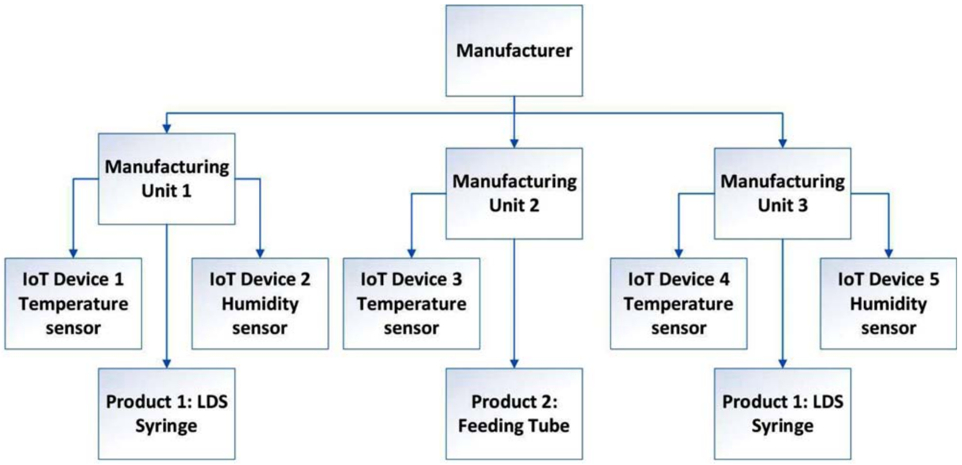

Manufacturer
Manufacturing Unit 1
Manufacturing Unit 2
Manufacturing Unit 3
IoT Device 1 Temperature sensor
IoT Device 2 Humidity sensor
IoT Device 3 Temperature sensor
IoT Device 4 Temperature sensor
IoT Device 5 Humidity sensor
Product 1: LDS Syringe
Product 2: Feeding Tube
Product 1: LDS Syringe

Fig. 3. A brief view of deployment of IoT devices in LDS syringe manufacturing.

4.3. Validation of data

During an inspection, the readings from the database can demonstrate adherence and compliance with sterility for a duration of time. The hash is calculated on the dataset and compared to that hash saved in the Ethereum transaction blockchain. If it matches, it proves that the database is not tampered with, and the IoT device indeed captures the readings.

Validation of the Maximum, Minimum, Average readings for the configurable duration is possible without a site visit by traversing through the blockchain and querying the blockchain to check for exceptions logged. Additional validation and access control are in place in the smart contract for managing the access to functions of the smart contract.

Figure 3 shows a simple view of how manufacturing units place IoT devices. Each manufacturing unit manufactures one or more products. Based on the sterility requirements in manufacturing these products, each unit may have one or many IoT devices. Each could measure any reading, such as temperature, humidity, or chemical content. A manufacturer may have more than one unit to manufacture the same goods or a series of products.

5. Blockchain experiments with ethereum smart contract

Some of the key characteristics of blockchain that make it broadly applicable to work across different types of applications and across diverse industries are:

- Distributed/Decentralized architecture: It does not involve the traditional client-server, centralized architecture. Information is stored in a cryptographically secured distributed environment, duplicating data on each machine/node part of the blockchain. It helps in multiple ways, such as.
  - By preventing a single point of failure
  - By eliminating the traditional cybersecurity risks such as DOS, spoofing attacks that usually target the server
- Immutable data storage: Once data gets a blockchain, this information becomes noneditable. Simplistically stating, each block stores the hash of the previous block. Any slight change in the previous block's data will result in the change in its hash value resulting in a mismatch of hashes

with the next block, thus invalidating the transaction. It helps with maintaining the data integrity of the blockchain.

- Eliminating the third-party/middleman: Blockchain-based on a smart contract eliminates the need for an escrow/mediator to oversee transactions between two parties.

The proposed solution can be deployed on the public Ethereum blockchain. Any node that is part of the blockchain will be able to interact with the manufacturer. Regulatory agencies, Syringe distributors, and healthcare service providers interact with the manufacturer as customers through the blockchain as long as they know where (address of the contract) the contract is located. Once deployed, this smart contract will not be altered or destroyed. The blockchain will store the manufacturing conditions to demonstrate sterility. This data is relevant for both data provenance and inspections. This data can be verified offsite as well and can give a snapshot of conditions in the past.

5.1. Smart contract

A smart contract consists of snippets of executable code that run on top of the blockchain. When executed, it facilitates an agreement between two or more parties who may not trust each other without a trusted third party (Buterin, 2014). Smart contracts enable users to embed their agreements and trust relations into a code and enable the execution of automated transactions without the supervision of a central authority. Smart contracts help automate paper contracts and agreements that otherwise require a third party by converting them into digital contracts.

The proposed smart contract will be between a manufacturer, regulatory bodies (e.g., FDA), and the manufacturer's potential customers. These customers could be hospitals that directly procure goods (E.g., LDS syringes) from this manufacturer or intermittent cloud manufacturing service providers who will procure and then distribute to medical centers and hospitals.

Each deployment of the smart contract will be associated with one manufacturer. Different manufacturers can deploy different instances of the smart contract, and these will be independent of one another hosted on different smart contract addresses on the Ethereum blockchain. It is in line with deploying an off-the-shelf application by different users where the same application will deploy independently. With the approach, the IoT devices are proposed to monitor the manufacturing facilities and act as sensors that capture the manufacturing conditions. A manufacturer can have one or more IoT devices placed in different manufacturing units associated with the syringes (or other products) manufacturing facilities. One smart contract implementation will hold information on all devices and readings in all the manufacturing units associated with this manufacturer. Each IoT device registered by the manufacturer will be assigned a threshold for low and high readings beyond which they will be outside its bound. If all readings are within the set low and high threshold, the environment in the manufacturing unit can be considered ideal for sterility and manufacturing the product. The smart contract will also hold a list of available products for customers to order and the inventory available through the manufacturer.

One of the advantages of cloud manufacturing is the cost efficiency it brings it. Hence, it would be counterproductive to have a cost overhead of a country-independent trusted third-party bank for regulating payments from service providers to manufacturers. A trusted payment mechanism that assures that the service provider will make payments on goods receipt is equally crucial for digital trust between the stakeholders involved. The proposed smart contract also provides ways of carrying out trusted transactions between a manufacturer and its customer (supplier/distributors or medical facilities).

5.2. *Smart contract template for CMfg*

A transaction to a smart contract can be triggered by referencing the unique addresses assigned to it by blockchain technology. Solidity is a high-level language used to implement smart contracts. Like an object-oriented programming artifact such as a class, smart contracts have attributes or variables that are either private or public, along with methods that operate on these attributes. The private attributes of a smart contract can be accessed through these operations/methods of the smart contract, whereas the public attributes are accessible by reference outside the methods defined in that smart contract. Even though an attribute may be private, it does not mean the values of this variable are a secret. If a smart contract developer knows the exact transaction where the smart contract referenced this attribute on a blockchain, the private attributes may be viewed by referencing this transaction using a block explorer. However, the private attributes are nonmodifiable from outside the smart contract. In addition to attributes and methods, the smart contract also has events. Events are very crucial for smart contracts. Because smart contracts' transactions are asynchronous, when a miner mines the transaction, the return code is not available synchronously to the frontend (JavaScript or any other UI of the blockchain). Here events come to the rescue as the smart contract's address stores these events. The blockchain frontend can configure to listen/subscribe to these events and act accordingly. They can serve as transaction log as well.

The smart contract developed to ensure the sterility constraints in cloud manufacturing can have the attributes, events, and operations below.

5.2.1. *Attributes*

(1) Single EOA (Externally owned address) for Manufacturer.
(2) Registration ID
(3) Product Inventory. A mapping list of all products with their inventory available for selling on this blockchain.
(4) A list of all devices deployed by manufacturer. The information is stored in a structure which has the device identifier, the manufacturing unit id, its status (ACTIVE/INACTIVE) which indicates if it is actively monitoring and logging into the blockchain, and its low and high threshold.
(5) Any address may request for access to the readings of an IoT sensor device. This means FDA account; any service provider or end-customer can attempt to access the readings. Any data logged in Ethereum is public and available to any address subscribed to this blockchain. So, the data is immutable but not private.

The smart contract maintains a list of access requests made by any address through its get methods. It enables the manufacturer to have visibility on which accounts accessed the readings. The visibility of these attributes is private, so the access is only through the smart contract. In solidity, every public attribute has default getter methods for access. The private visibility keeps control on provisioning access only through the methods defined in the smart contract.

The Shipments list maintains the list of customers, the quantity to be sent out to each customer, and the amount exchanged for the product sale. The customer's default mapping maintains any defaults for specific customers. For example, if a service provider states by default, auto-send 20 LDH syringe cartons if all readings are within bound, this default is saved in an attribute named Customer_Qty.

Figure 4 above represents a process workflow of possible interactions between a Manufacturer, two customers Customer-1 and Customer-2, the Regulatory body (FDA), and the smart contract.

Figure 4 depicts how the smart contract template for CMfg works, capturing a few daily operating scenarios. Let us assume that the initial balance in the manufacturer's account and two customer accounts are 100 ETH, respectively. The smart contract address does not have any amount locked

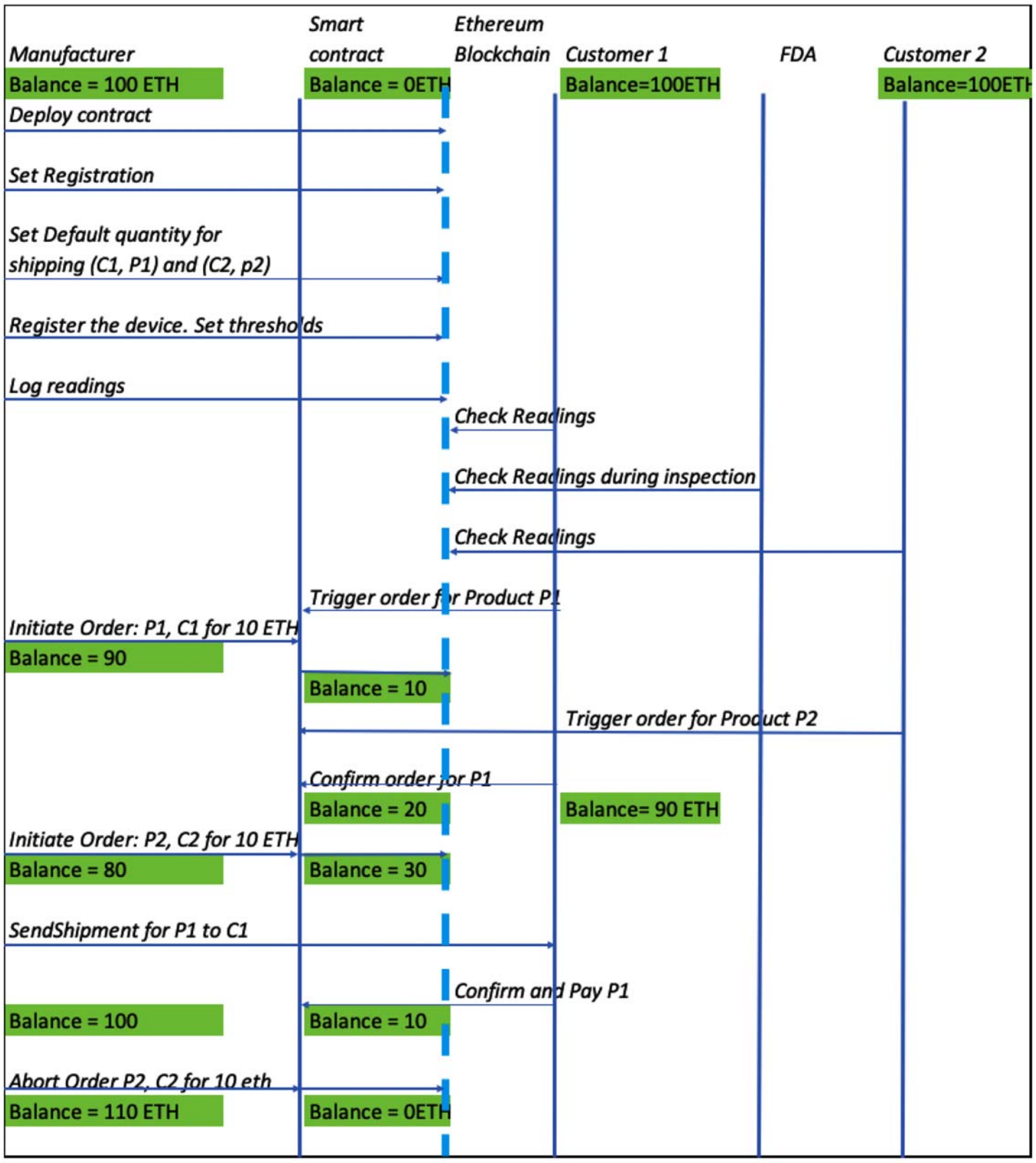


Fig. 4. Process workflow diagram of the smart contract functions.

against it. As the first step, the manufacturer deploys the smart contract. He then sets its registration id and sets default quantities for the two regular customers. Setting the default quantity implies that when these customers validate the IoT readings posted, get convinced that the current good manufacturing processes are adhered to, and approve to trigger an order from the manufacturer, the dispatch of shipments with respective default quantities happens. Ethereum deducts a small fraction of processing charges for these transactions. This amount of processing transactions depends on the code in the smart contract for these operations/methods.

Next, the manufacturer registers the IoT devices in their manufacturing units and sets the thresholds for these devices. Subsequently, the IoT devices will collect the sensor data and post the readings to the blockchain via the logging method of the smart contract. Once the readings are in the blockchain, the customers can check these readings to make informed decisions on purchases from the manufacturer. The regulatory body can also validate these readings in the event of an inspection.

Upon validation, Customer-1 triggers an order for product P1. Upon receiving this trigger for intent to purchase, the manufacturer initiates the order for P1 for the default quantity Q1 for Customer-1. The manufacturer also locks 10 ETH into the smart contract. This amount is the selling price of product P1 for quantity Q1. At this point, the manufacturer's balance reduces to approximately 90 ETH. The smart contract has locked 10 ETH. The manufacturer's balance is an approximation here as we are not tracing the fractional processing fee that gets deducted by each of the other operations performed by the manufacturer using the smart contract. Upon receiving the initiatial order event, Customer-1 confirms the order by locking 10 ETH into the smart contract's address. The balance for Customer-1 is now 90 ETH, and the smart contract has a balance of 20 ETH in total. Subsequently, the manufacturer also initiates an order for Customer-2. Assuming the transaction amount is again 10 ETH, 30 ETH is now locked in the smart contract. The manufacturer now ships out to Customer-1. Upon receipt, Customer-1 confirms and pays the manufacturer. At this point, Customer-1 sends 20 ETH to the manufacturer's account. It includes the 10 ETH locked by the manufacturer originally and the 10 ETH sent by the customer. Suppose the manufacturer aborts Customer-2's order, on the other hand, for reasons better known to him. In this case, Customer-2's 10 ETH locked with the smart contract will return to its account.

At the end of these interactions, the manufacturer gets a net of 10 ETH for his transaction with Customer-1. He gets nothing from Customer-2 as their transaction is in abort status. Hence final balance with the manufacturer is 110 ETH. Customer-1's balance is less by 10 ETH, and Customer-2 is as-is. The residual balance locked on the smart contract is also 0 since no "in process" transactions are now pending.

5.2.2. Operations/methods

(1) Manufacturer attributes maintenance:

- The *set_RegistrationID()* and *updateInventory()* functions are used by the manufacturer to set its registration ID and product inventory.
- The *getInventory()* function is an internal function used within the smart contract to get the available inventory of a product.
- The *setQty_customer()* will set the default quantity for a customer for a particular product. It means when this customer with this quantity set calls a *trigger_order()* function, the default qty is shipped out.
- Modifier *onlyManufacturer(),* when used in a function, ensures that it only executes when the condition in the modifier is valid. In this case, the contract accepts only the transactions from the manufacturer's.

(2) Device maintenance:

- Only the Manufacturer's account can call functions to activate/deactivate a device.
- *activateDevice():* When the below parameters are passed as input, the *activateDevice()* function will activate the IoT device to post readings.
  - Manufacturing unit
  - IoT Device ID
  - Low and high threshold within which the readings are considered normal to maintain sterility.
- Similarly, functions to Deactivate/Reactivate IoT device are also part of the smart contract.

(3) Manufacturing transparency and provenance management:

Post Reading:

- Each IoT reading on the Ethereum blockchain will have the following:
  - IoT Device ID
  - Start and End timestamp
  - Min, Max, Avg, Std Deviation
  - The hash of all the readings being aggregated that fall under the start and end timestamp.
- *postReading()* logs a consolidated reading from the IoT device. The manufacturer's account logs this through the computer on which the database that the IoT device readings reside is. Consolidation of max_reading, min_reading, average_reading, the standard deviation for all readings between the start and end timestamp, and the readings' hash is a single transaction.

Read readings:

- *comparethresholds()* below compares the min, max, and average reading for a start and end timestamp to the threshold values and returns a true if these are within this threshold.
- It means that during this time frame, the manufacturing conditions complied with the expected behavior. *getReading()* returns the reading logged by the device passed with the start and end time. The reading is returned to the UI through an event as well.
- The *checkProduct()* checks whether a passed productid exists in the blockchain and has inventory against it.
- *checkAccessRequest()* is used by the manufacturer if he needs to know if a particular address has ever requested to access a device's reading.

5.2.3. Operations overview

- Any customer can request a reading from a device using the *getReading()* function of the smart contract. It is possible to request any number of readings logged in the blockchain by the manufacturer.
- Once convinced that manufacturing has complied with the necessary sterility practices, they can send a *trigger_order()* function.
- It sends an event that the manufacturer is actively listening to. Once received, the manufacturer sends an *initiate_purchase().* This transaction value is the amount the manufacturer is quoting to receive in return for the product's sale.
- The customer then sends *Confirm_purchase()* to confirm the purchase. The value of the transaction delivers to the smart contract. At this point, a smart contract has amounts from both parties, but the transaction state is locked.
- Once *confirm_purchase()* is received, the manufacturer sends the shipment.
- The customer confirms and pays by sending a *confirm_and_pay().* At this point, the manufacturer gets his money and customer his goods—all without a third party involved.
- If the manufacturer wants to abort the transaction before a *confirm_purchase()* is sent, he can do so. His money will be sent back to his account, and the transaction will be reverted.

5.3. Examples of operations with smart contract

Once the smart contract is created and deployed on a blockchain, the smart contract code becomes immutable. No changes to the smart contract code are possible at this point. It ensures the integrity of the smart contract, and its terms are maintained. This section highlights the output for sample transactions to provide an outlook of the working of the smart contract created in the preceding section.

Example 1: Manufacturer posts readings from the IoT device into the blockchain

Once the Ethereum address deploys the smart contract, the address of the account through which it is deployed becomes the manufacturer. The manufacturer subsequently sets his registration ID, registers the devices in its manufacturing units, and sets the device thresholds. Only the manufacturer will be authorized to log the readings from the IoT sensors. The log_readings() function within the smart contract can register a manufacturer, add products to its list, activate a sample IoT device, set its thresholds, and log readings to the blockchain.

Below is the output of log_readings() once it executed successfully:

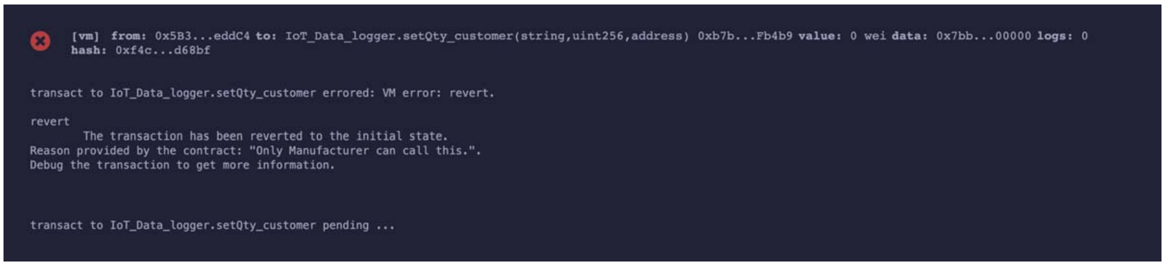
status	true *Transaction mined, and execution succeed*
transaction hash	0xc9c76972d915395218cda9fa... <*The transaction hash generated by Ethereum using the data of the transaction such as sender, gas, nonce and value of the transaction*>
from	0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 <*The address of the transaction initiator/sender. The from address in this case will be the manufacturer's address*> .
to	IoT_Data_logger.log_readings() 0xd9145CCE52D386f254917e481eB44e9943F39138 <*The "to address" is the address of the smart contract. Each smart contract deployed on the Ethereum blockchain has its own contract address. The function that is invoked by the sender is also logged in the output*>
gas	80000000 gas <*max gas allocated after spending which, if the transaction needs more gas, it will be reverted.*>
transaction cost	80000000 gas <*The costs for sending the contract code to the Ethereum blockchain. It depends on the size of the transaction and contract.*>
execution cost	953976 gas <***EXECUTION COST** is the actual cost of the executing the transaction on Ethereum. In this case, the log_readings function registers a IoT device, log the 4 transactions for the IoT sensor The cost is for all these operations.*>
hash	0xc9c76972d915395218cda9fa... <*The transaction hash*>
input	0x2a8...674a9 <*Hash of inputs passed*>
decoded input	{} lt;*We have not passed any inputs to the smart contract function*>
decoded output	<***Return value***> { "0": "bool: true" }
logs	[{ <***Event 1: Change notification to activate device***> "from": "0xd9145CCE52D386f254917e481eB44e9943F39138", "topic": "0xe5527ca694f4246807eeef... ", "event": "ChangeNotification", "args": { "0": "0x5B38Da6a701c5... ", "1": "IoT Device activated", "sender": "0x5B38Da6a701c5... .",

	"notificationMsg": "IoT Device activated" }}, {
	<Event 2: Logger to set thresholds> "from": "0xd9145CCE52D386f254917e481eB44e9943F39138", "topic": "0xfbd815649c4d5f4a928ff248d912d..", "event": "logger", "args": { "0": "Thresholds for readings set" }}, {***<Event 3: Logger to Post transactions>*** "from": "0xd9145CCE52D386f254917e481eB44e9943F39138", "topic": "0xfbd815649c4d5f4a928ff248d912dbf5..", "event": "logger", "args": { "0": "Transactions posted" }}] *<The first argument is true since the transaction executed successfully. Next, the 3 events are logged indicating activation of an IoT device, setting of its thresholds, and posting the readings. The events posted provide information on sender of the events, what topics they are associated to, the name of the events and the arguments associated with it.* *These events can be listened to by the blockchains user interface. Listeners can be for all events or for a specific topic.>*
value	0 wei *<The actual amount being transferred from sender to receiver, if any associated with this transaction>*

Example 2: An external account other than a manufacturer attempts to set default quantity for a customer.

Any account that has the address at which the smart contract is deployed can invoke a smart contract operation. The smart contract should have in-built security policy and restrictions to secure access management. In this example, an external entity uses its account that is not the manufacturer and attempts to set a default quantity for a customer. The smart contract is set in a way that only manufacturer can set up default quantity This operation should therefore not be successful.

The snapshot below demonstrated the same where the transaction is reverted to the initial state when invoked by any entity that is not the manufacturer.



```
[vm] from: 0x5B3...eddC4 to: IoT_Data_logger.setQty_customer(string,uint256,address) 0xb7b...Fb4b9 value: 0 wei data: 0x7bb...00000 logs: 0
hash: 0xf4c...d68bf

transact to IoT_Data_logger.setQty_customer errored: VM error: revert.

revert
       The transaction has been reverted to the initial state.
Reason provided by the contract: "Only Manufacturer can call this.".
Debug the transaction to get more information.



transact to IoT_Data_logger.setQty_customer pending ...
```

Example 3: Manufacturer sets default quantity for a customer.
For setting the default quantity, The manufacturer calls the setQty_customer() function. The output of this transaction is shown as below.

status	true Transaction mined and execution succeed
transaction hash	0xb277e9c190a964352ce75a1a91 . . .
from	0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2
to	IoT_Data_logger.setQty_customer(string,uint256,address) 0xb7bb1792BBfabbA361c46DC5860940e0E1bFb4b9
gas	80000000 gas
transaction cost	80000000 gas
execution cost	93171 gas
hash	0xb277e9c190a964352ce75a1a91 . . .
input	0x7bb...00000
decoded input	{ "string _Product_id": "P1", "uint256 _Quantity": "10", "address _Customer": "0x5B38Da6a701c568545dCfcB03FcB875f56beddC4" } *<In this function, we are passing the product ID, the default quantity, and the customer's address for which we set the quantity>*
decoded output	{ "0": "bool: true" }
logs	[]
value	0 wei

Example 4: Customer triggers an order request to the manufacturer
Once the customer confirms that the logged IoT readings are within the threshold to make sure that the manufacturing conditions are sterile, the customer can trigger a request to the manufacturer for ordering the product. Two possibilities exist. Either a customer who already has an established relationship with manufacturer and for whom a default quantity is already set calls the trigger_order_scheduled() function or a new customer triggers an order request using trigger_order_adhoc() function by specifying a custom quantity they need. In the example below, Customer C1 triggers the trigger_order_scheduled() function to order product P1, for the default quantity Q1 set for the customer.

status	true Transaction mined and execution succeed
transaction hash	0xacb0240a24c079f70650abe866f2b770 . . .
from	0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to	IoT_Data_logger.trigger_order_scheduled(string) 0xb7bb1792BBfabbA361c46DC5860940e0E1bFb4b9
gas	80000000 gas
transaction cost	80000000 gas
execution cost	276521 gas
hash	0xacb0240a24c079f70650abe866f2b770 . . .
input	0xc58...00000
decoded input	{ "string _Product_ID": "P1" }
decoded output	{ "0": "bool: true" }

logs	[{ *<Event 1>*: ”from”: “0xb7bb1792BBfabbA361c46DC5860940e0E1bFb4b9”, ”topic”: “0x9be3c99bd5619cb25afeaf47343305 . . . ”, ”event”: “ReturnQty”, ”args”: { “0”: “0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2”, “1”: “10”, “sender”: “0xAb8483F64d9C6d1EcF9b849Ae..”, “message”: “DEFAULT_QTY” } }, *<Event 2>*: { “from”: “0xb7bb1792BBfabbA361c46DC5860940e0E1bFb4b9”, ”topic”: “0x0face0a73c45f51afeeaafd9ffbacc83a . . . ”, ”event”: “ReturnInventory”, “args”: { “0”: “0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2”, “1”: “100”, “sender”: “0xAb8483F64d9C6d1EcF9b849Ae . . . ”, “message”: “INVENTORY” } }, *<Event 3>*: { “from”: “0xb7bb1792BBfabbA361c46DC5860940e0E1bFb4b9”, ”topic”: “0xfbd815649c4d5f4a928ff248d912dbf50e6eb2 . . . ”, ”event”: “logger”, ”args”: { “0”: “trigger - scheduled order “ } }] *<Three events are logged for setting default quantity, inventory and subsequently triggering a scheduled order>*
value	0 wei

The manufacturer then sends a request for initiate purchase. The value specifies how many ether coins the manufacturer expects from the customer. With this manufacturer sends 10 ethers to the smart contract’s address. This is like an escrow account now. If manufacturer does any foul play, he will end up losing this money.

from	*<Manufacturer*’*s address>*
to	IoT_Data_logger.initiatePurchase(address,string,uint256)
gas	80000000 gas
transaction cost	80000000 gas
execution cost	64904 gas
hash	0x781dc9efa925785c . . .
input	0x157...00000
decoded input	{ ”address _Customer”: “0x5B38Da6a701c568545dCfcB03FcB875f56beddC4”, “string _Product_id”: “P1”, ”uint256 _Qty”: “10” }

decoded output	{ "0": "bool: true" }
logs	[{ "from": "0xb7bb1792BBfabbA3 ... ", "topic": "0xf31e622e914905950 ... ", "event": "Purchaseinitiate", "args": { "0": "P1", "1": "10", "2": "0x5B38Da6a701c568545dCfcB03FcB875f56beddC4" } }]
value	**10000000000000000000 wei**

Next Customer confirms the transaction:

status	true Transaction mined and execution succeed
transaction hash	0x6ad312e97bcc1e800b8fde48012c3c4 ...
from	0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to	IoT_Data_logger.confirmPurchase(string,uint256)
gas	80000000 gas
transaction cost	80000000 gas
execution cost	61906 gas
hash	0x6ad312e97bcc1e800b8fde48012c3c4 ...
input	0x8d6...00000
decoded input	{ "string _Product_id": "P1", "uint256 _Qty": "10" }
decoded output	{ "0": "bool: true" }
logs	[{ "from": "0xb7bb1792BBfabbA361c46DC5860940e0E1bFb4b9", "topic": "0xd5d55c8a68912e9a110618df8d5..", "event": "PurchaseConfirmed", "args": {}]
value	10000000000000000000 wei

sendShipment() is called by manufacturer to confirm that goods are shipped. The tracing number is provided

status	true Transaction mined and execution succeed
transaction hash	0x204f664689ae75d86799a5 ...
from	0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2
to	IoT_Data_logger.sentShipment(address,string,uint256,string)
gas	80000000 gas
transaction cost	80000000 gas
execution cost	28425 gas
input	0x87b...00000
decoded input	{ "address _Customer": "0x5B38Da6a701c568545dCfcB03FcB875f56beddC4", "string _Product_id": "P1", "uint256 _Qty": "10", "string _tracking_no": "12345" }

decoded output	{ “0”: “bool: true” }
Logs	[{ ”from”: “0xb7bb1792BBfabbA361c46DC5860940e0E1bFb4b9”, ”topic”: “0x0ef0840f374c2a220a129c3093 . . . ”, ”event”: “Shipmentsent”, ”args”: { “0”: “P1”, “1”: “10”, “2”: “12345” } }]
Value	0 wei

Next, Customer confirms the purchase and pays. At this point the balances get transferred to the manufacturer.

Key insights:

The above experiment implemented with Ethereum Remix shows the feasibility of the proposed framework. By leveraging Blockchain, the framework could ensure the data integrity of the manufacturing provenance and achieve the goal of transparency for trust.

Placing IoT sensors at the operational layer generates humongous data. On the other hand, logging everything in a blockchain is not just expensive but infeasible. The smart contract above addresses these issues by leveraging edge computing and logging statistics (max, min, average, and standard deviation of readings) on the blockchain and potentially storing detailed records on a lower-cost local database on the manufacturer’s premise or on-cloud. Storing the hash of the records on the blockchain ensures that the detailed dataset has not been altered, providing a data-integrity-based digital trust mechanism to the stakeholders.

On operations, we have also illustrated by example how a manufacturer can potentially carry out financial transactions with customers using the smart contract without any other third party. The manufacturer can efficiently maintain product inventory and auto-trigger shipments of a set default quantity to their customers upon demonstrating sterile manufacturing conditions. The smart contract can also facilitate an online, over-the-blockchain manufacturing site inspection by regulatory authorities like the FDA.

6. Conclusion

We created a basic framework of a blockchain-based digital trust mechanism for Cloud Manufacturing and implemented it with Ethereum blockchain and smart contract. This framework is constructed by developing a use case of cloud manufacturing of low dead space (LDS) medical syringes to overcome the urgent supply shortage of this medical device for the COVID-19 vaccination. The problem addressed by this use case itself is a significant issue in fighting the COVID-19 pandemic, and it is a representative problem that appears in today’s manufacturing and supply chain. Cloud manufacturing appears a promising solution that allows quickly establishing a digital virtual enterprise that pools together various manufacturing resources worldwide to meet the surged needs of products and save cost and time. However, a major barrier in this new distributed manufacturing model is the lack of trust with respect to cross-domain identity management, standard/regulation compliance, and manufacturing conditions for product quality and sterility to ensure the safety of the later medical product use. We proposed a basic framework to mitigate this trust problem by integrating evidence-based and transparency-based trust mechanisms together with IoT and blockchain technologies. Through the LDS syringe manufacturing use case, we demonstrate the technical feasibility of this solution.

References

Barbhuiya, S., Nikolopoulos, D., Price, M., Robinson, T., Nolan, D., Zhang, W., & Kyle, S. (2019). SmartMaaS: A Framework for Smart Manufacturing-as-a-Service. *17th International Conference on Manufacturing Research ICMR 2019*, *Proceeding*, 1–7.

Borangiu, T., Trentesaux, D., Thomas, A., Leitão, P., & Barata, J. (2019). Digital transformation of manufacturing through cloud services and resource virtualization. *Computers in Industry*, *108*, 150–162. https://doi.org/10.1016/j.compind.2019.01.006

Buterin, V. (2014). A next-generation smart contract and decentralized application platform. *Etherum*, *January*, 1–36. https://buyxpr.com/build/pdfs/EthereumWhitePaper.pdf

Caggiano, A., Segreto, T., & Teti, R. (2016). Cloud Manufacturing Framework for Smart Monitoring of Machining. *Procedia CIRP*, *55*, 248–253. https://doi.org/10.1016/j.procir.2016.08.049

Christidis, K., & Devetsikiotis, M. (2016). Blockchains and Smart Contracts for the Internet of Things. *IEEE Access*, *4*, 2292–2303. https://doi.org/10.1109/ACCESS.2016.2566339

CSA. (n.d.). *STAR (Security, Trust and Assurance Registry).*

CSA. (2011). *CloudTrust Protocol* {*(CTP)*}.

FDA. (2017). *Is It Really* "*FDA Approved?*" | *FDA*. Official Website. https://www.fda.gov/consumers/consumer-updates/it-really-fda-approved

FDA. (2020a). *Premarket Notification 510(k)* | *FDA*. Official Website. https://www.fda.gov/medical-devices/premarket-submissions/premarket-notification-510k#se

FDA. (2020b). *Registration and Listing*. Official Website. http://www.fda.gov/MedicalDevices/DeviceRegulationandGuidance/HowtoMarketYourDevice/RegistrationandListing/default.htm

FDA. (2021). *FDA in Brief: FDA Calls on Certain Firms to Stop Producing and Issuing Misleading* "*FDA Registration Certificates*" | *FDA*. Official Website. https://www.fda.gov/news-events/fda-brief/fda-brief-fda-calls-certain-firms-stop-producing-and-issuing-misleading-fda-registration

Fisher, O., Watson, N., Porcu, L., Bacon, D., Rigley, M., & Gomes, R. L. (2018). Cloud manufacturing as a sustainable process manufacturing route. *Journal of Manufacturing Systems*, *47*, 53–68.

He, W., & Xu, L. (2015). A state-of-the-art survey of cloud manufacturing. *International Journal of Computer Integrated Manufacturing*, *28*(3), 239–250. https://doi.org/10.1080/0951192X.2013.874595

Helo, P., Suorsa, M., Hao, Y., & Anussornnitisarn, P. (2014). Toward a cloud-based manufacturing execution system for distributed manufacturing. *Computers in Industry*, *65*(4), 646–656. https://doi.org/10.1016/j.compind.2014.01.015

Hewett, N., Lehmacher, W., & Wang, Y. (2019). Inclusive Deployment of Blockchain for Supply Chains: Part 1 – Introduction. *World Economic Forum*, *March*, 26.

Huang, J., Gheorghe, A., Handley, H., Pazos, P., Pinto, A., Kovacic, S., Collins, A., Keating, C., Sousa-Poza, A., Rabadi, G., Unal, R., Cotter, T., Landaeta, R., & Daniels, C. (2020). Towards digital engineering: the advent of digital systems engineering. *Int. J. System of Systems Engineering*, *10*(3), 234–261. https://doi.org/10.1504/IJSSE.2020.109737

Huang, J., & Nicol, D. M. (2013). Trust mechanisms for cloud computing. *Journal of Cloud Computing*, *2*(1), 1–14. https://doi.org/10.1186/2192-113X-2-9

Hufford, Austen; Hopkins, J. S. (2021). *Covid-19 Vaccines Are Wasted as Special Syringes Run Short - WSJ*. The Wall Street Journal. https://www.wsj.com/articles/covid-19-vaccines-are-wasted-as-special-syringes-run-short-11620379801

Knode, R., & Egan, D. (2010, July). *Digital Trust in the Cloud – A Precis for the CloudTrust Protocol (V2.0).* CSC.COM.

Lee, J., Bagheri, B., & Jin, C. (2016). Introduction to cyber manufacturing. *Manufacturing Letters*, *8*, 11–15. https://doi.org/10.1016/j.mfglet.2016.05.002

Li, B.-H., Zhang, L., Wang, S.-L., Tao, F., Cao, J. W., Jiang, X. D., Song, X., & Chai, X. D. (2010). Cloud manufacturing: a new service-oriented networked manufacturing model. *Computer Integrated Manufacturing Systems*, *16*(1), 1–7.

Liang, X., Shetty, S., Tosh, D., Kamhoua, C., Kwiat, K., & Njilla, L. (2017). ProvChain: A Blockchain-Based Data Provenance Architecture in Cloud Environment with Enhanced Privacy and Availability. *Proceedings - 2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, CCGRID 2017*, *May*, 468–477. https://doi.org/10.1109/CCGRID.2017.8

Liu, Y., Wang, L., Wang, X. V., Xu, X., & Jiang, P. (2019). Cloud manufacturing: key issues and future perspectives. *International Journal of Computer Integrated Manufacturing*, *32*(9), 858–874.

Lu, Q., & Xu, X. (2017). Adaptable blockchain-based systems: A case study for product traceability. *IEEE Software*, *34*(6), 21–27.

Lu, Y., & Xu, X. (2019). Cloud-based manufacturing equipment and big data analytics to enable on-demand manufacturing services. In *Robotics and Computer-Integrated Manufacturing* (Vol. 57, Issue November). https://doi.org/10.1016/j.rcim.2018.11.006

Mell, P., Grance, T., & others. (2011). *The NIST definition of cloud computing*. https://doi.org/https://doi.org/10.6028/NIST.SP.800-145

Moghaddam, M., Jones, A., & Wuest, T. (2019). Design of marketplaces for smart manufacturing services. *Procedia Manufacturing*, *39*(2019), 194–201. https://doi.org/10.1016/j.promfg.2020.01.312

Mohanta, B. K., Panda, S. S., & Jena, D. (2018). An Overview of Smart Contract and Use Cases in Blockchain Technology. *2018 9th International Conference on Computing, Communication and Networking Technologies, ICCCNT 2018*, *November*, 1–4. https://doi.org/10.1109/ICCCNT.2018.8494045

Molina, A., Velandia, M., & Galeano, N. (2007). Virtual enterprise brokerage: a structure-driven strategy to achieve build to order supply chains. *International Journal of Production Research*, *45*(17), 3853–3880. https://doi.org/10.1080/00207540600818161

Moutinho, S. (2021). *Syringe size and supply issues continue to waste COVID-19 vaccine doses in United States* | *Science* | *AAAS*. Scienceinsider. https://www.sciencemag.org/news/2021/03/syringe-size-and-supply-issues-continue-waste-covid-19-vaccine-doses-united-states

Nakamoto, S. (2008). *Bitcoin: A peer-to-peer electronic cash system,*" https://bitcoin. org/bitcoin. pdf.

NSF. (2020). *Future Manufacturing (FM)*. NSF. https://www.nsf.gov/pubs/2020/nsf20552/nsf20552.htm

Rožman, N., Diaci, J., & Corn, M. (2021). Scalable framework for blockchain-based shared manufacturing. *Robotics and Computer-Integrated Manufacturing*, *71*, 102139. https://doi.org/10.1016/j.rcim.2021.102139

Suhail, S., Hussain, R., Khan, A., & Hong, C. S. (2020). Orchestrating product provenance story: when IOTA ecosystem meets electronics supply chain space. *Computers in Industry*, *123*, 103334.

Szabo, N. (1996). Smart Contracts: Building Blocks for Digital Markets. Retrieved May 1, 2020, from https://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart_contracts_2.html

Tao, F., Zhang, L., Venkatesh, V. C., Luo, Y., & Cheng, Y. (2011). Cloud manufacturing: a computing and service-oriented manufacturing model. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, *225*(10), 1969–1976.

Umeda, S., Nakano, M., Mizuyama, H., Hibino, H., Kiritsis, D., & von Cieminski, G. (2015). Protecting Intellectual Property in a Cloud Manufacturing Environment: Requirements and Strategies. *IFIP Advances in Information and Communication Technology*, *460*, V–VI. https://doi.org/10.1007/978-3-319-22759-7

Vatankhah Barenji, A., Li, Z., & Wang, W. M. (2018). Blockchain Cloud Manufacturing: Shop Floor and Machine Level. *Smart SysTech 2018; European Conference on Smart Objects, Systems and Technologies*, 1–6.

Vatankhah Barenji, R. (2021). A blockchain technology based trust system for cloud manufacturing. *Journal of Intelligent Manufacturing*, *April 2020*. https://doi.org/10.1007/s10845-020-01735-2

Wang, X., Zha, X., Ni, W., Liu, R. P., Guo, Y. J., Niu, X., & Zheng, K. (2019). Survey on blockchain for Internet of Things. *Computer Communications*.

Wu, D., Greer, M. J., Rosen, D. W., & Schaefer, D. (2013). Cloud manufacturing: Strategic vision and state-of-the-art. *Journal of Manufacturing Systems*, *32*(4), 564–579.

Wu, D., Ren, A., Zhang, W., Fan, F., Liu, P., Fu, X., & Terpenny, J. (2018). Cybersecurity for digital manufacturing. *Journal of Manufacturing Systems*, *48*, 3–12. https://doi.org/10.1016/j.jmsy.2018.03.006

Xu, X. (2012). From cloud computing to cloud manufacturing. *Robotics and Computer-Integrated Manufacturing*, *28*(1), 75–86.

Yadekar, Yaser; Shehab, E. M. J. (2013). *Challenges of Cloud Technology in Manufacturing Environment*. *1*(September), 337–342.

Yan, K., Cheng, Y., & Tao, F. (2016). A trust evaluation model towards cloud manufacturing. *International Journal of Advanced Manufacturing Technology*, *84*(1–4), 133–146. https://doi.org/10.1007/s00170-015-8002-5

Zhang, L., Luo, Y., Tao, F., Li, B. H., Ren, L., Zhang, X., Guo, H., Cheng, Y., Hu, A., & Liu, Y. (2014). Cloud manufacturing: a new manufacturing paradigm. *Enterprise Information Systems*, *8*(2), 167–187.

Zheng, Z., Xie, S., Dai, H.-N., Chen, W., Chen, X., Weng, J., & Imran, M. (2020). An overview on smart contracts: Challenges, advances and platforms. *Future Generation Computer Systems*, *105*, 475–491. https://doi.org/https://doi.org/10.1016/j.future.2019.12.019

Zhong, R. Y., Wang, L., & Xu, X. (2017). An IoT-enabled real-time machine status monitoring approach for cloud manufacturing. *Procedia CIRP*, *63*, 709–714.

Zhu, X., Shi, J., Huang, S., & Zhang, B. (2020). Consensus-oriented cloud manufacturing based on blockchain technology: An exploratory study. *Pervasive and Mobile Computing*, *62*. https://doi.org/10.1016/j.pmcj.2020.101113.

Author Biographies

Trupti Rane is a Solution Architect in the Business Intelligence and Analytics domain at Canadian Pacific Railway. She is pursuing her Ph.D. in Engineering, specializing in Engineering Management and Systems Engineering at Old Dominion University, Norfolk, Virginia, USA. She has completed her Master of Science in Information Technology from the University of Cincinnati, Ohio, USA, and Bachelor of Engineering in Computer Engineering from Goa University, India. Her research interests include Cloud Manufacturing, Digital Trust, Blockchain, Data provenance and Cybersecurity. Contact her at trane002@odu.edu.

Jingwei Huang is an Associate Professor in the Department of Engineering Management and Systems Engineering at Old Dominion University, Norfolk, Virginia, USA. He received Ph.D. in Information Engineering from University of Toronto, Toronto, Ontario, Canada. His research is in the areas of Digital Transformation, Digital Engineering Transformation, Digitalization, Formal semantics of trust, Digital trust, Digital mechanisms of trust and security, Provenance modeling, Trustworthy AI Systems, Big Data & Machine Learning. Contact him at j2huang@odu.edu.