


Article

Decentralizing Supply Chain Anti-Counterfeiting and Traceability Systems Using Blockchain Technology

Neo C. K. Yiu 

Department of Computer Science, University of Oxford, Oxford OX1 3QD, UK;
neo-chungkit.yiu@kellogg.ox.ac.uk

Abstract: An interesting research problem in the supply chain industry is evaluating and determining the provenance of physical goods—demonstrating the authenticity of luxury goods such as bottled wine. However, many supply chain systems and networks have been built and implemented with centralized system architecture, relying on centralized authorities or any form of intermediary, and leading to issues such as single-point processing, storage and failure, which could be susceptible to malicious modifications to product records or various potential attacks to system components by dishonest participant nodes traversing along the supply chain. Blockchain technology has evolved from merely being a decentralized, distributed and immutable ledger of cryptocurrency transactions to a programmable interactive environment for building decentralized and reliable applications addressing different use-cases and existing problems in the world. In this research, with a chosen research method of proof-by-demonstration, the Decentralized NFC-Enabled Anti-Counterfeiting System (dNAS) is proposed and developed, decentralizing a legacy anti-counterfeiting system of the supply-chain industry using Blockchain technology to facilitate trustworthy data provenance retrieval, verification and management, as well as strengthening the capability of the product's anti-counterfeiting and traceability qualities in the wine industry, with the capacity to further extend this to the supply chain industry as a whole. The proposed dNAS utilizes a decentralized blockchain network with a consensus protocol compatible with the concept of enterprise blockchain, programmable smart contracts and a distributed file storage system to develop a secure and immutable scientific-data provenance tracking and management platform on which provenance records, providing compelling properties of the data integrity of luxurious goods, are recorded, verified and validated automatically.



Citation: Yiu, N.C.K. Decentralizing Supply Chain Anti-Counterfeiting and Traceability Systems Using Blockchain Technology. *Future Internet* **2021**, *13*, 84. <https://doi.org/10.3390/fi13040084>

Academic Editor: Spyros Makridakis

Received: 14 February 2021

Accepted: 20 March 2021

Published: 25 March 2021

Keywords: blockchain; anti-counterfeiting; smart contracts; product authenticity; end-to-end traceability; supply chain integrity; supply chain provenance; decentralization; enterprise blockchain; NFC-Enabled Anti-Counterfeiting System; near-field communication; internet-of-things

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In this section, the motivation of this research, and a list of research questions with the research methodology, are pinpointed and explained by going through the growing challenges of improving counterfeit product trading. The current solutions, which have already been implemented in the supply chain industry against the challenges, along with the ineffectiveness and inefficiency of current solutions against counterfeit product trading, are discussed to better address the need for more innovative and decentralized solutions, such as dNAS, which is proposed in this research alongside a set of research objectives.

1.1. The Existing Challenges in Supply Chain Anti-Counterfeiting

The problem of counterfeit product trading, such as luxurious goods or pharmaceutical products, has been one of the major challenges the supply chain industry has been facing in an innovation-driven global economy. The situation has been exacerbated by an

exponential growth in counterfeit and pirated goods worldwide, which has also plagued companies which have multinational supply chain networks for decades and longer.

The analytical study—[1], published by the Organization for Economic Cooperation and Development (OECD) and European Union Intellectual Property Office (EUIPO) in 2016 has further suggested that the volume of counterfeit product trading has been alarming, reaching as much as \$509 billion, representing 3.3% of world trade and 6.8% of imports from non-EU countries. Counterfeit trading activities not only infringe on trademarks and copyrights, and negatively impact on the sales and profits of industries, they also generate profits for organized crime at the expense of the affected companies and governments, as well as having broader adverse effects on the economy, public health, safety and security of the wider community.

In response to the growing concern, innovative, fully-functional, integrable and affordable product anti-counterfeiting solutions, utilizing cutting-edge technologies, have been widely and urgently demanded so as to ensure the provenance and traceability of genuine products throughout supply chain counterfeiting, and suggested solutions should be widely adopted regardless of the industry, the size of the company and its supply chain systems.

1.2. Current Alternative Resolutions of Product Anti-Counterfeiting

Given the growing concern and worsening situation in the trading activities of counterfeit products, anti-counterfeiting solutions have been developed and implemented in the supply chain systems of different industries, as listed and explained in [2]. Wireless communication technologies, such as Radio-Frequency Identification (RFID) or near-field communication (NFC), which is a subset of RFID, powering the Internet-of-Things (IoT) are mostly the existing solutions with centralized architectures currently based on. The tags with wireless communication capabilities are packaged on the packets of goods or the product itself for purposes of identification, anti-counterfeiting and traceability.

One of the solutions to the growing concerns regarding product counterfeiting in supply chain systems, and specifically for the wine industry, was the *NFC-enabled Anti-Counterfeiting System (NAS)*, as depicted in Appendix A.1 [3], developed and implemented back in 2014, which aimed at providing an innovative and fully functional alternative for supply chain anti-counterfeiting and traceability, based on near-field communication technology and cloud-based microservice architecture with centralized storage structure, hosted by any winemaker, to help improve the worsening situation of product counterfeiting, especially for the wine industry.

It has come to a point where, even though the implementation of NAS itself is already more secured than most of the typical supply chain systems, with original wine records being validated at any node along the supply chain, centralized architectures of NAS could still pose a risk in terms of data integrity, as any node, not only winemakers, along the supply chain have full control of wine records stored in their own database architectures. In case different nodes along the supply chain are untrusting of each other, there is still a possibility that duplicated wine records lead to an adverse situation where wine consumers can still purchase bottled wine counterfeits with fabricated wine records retrieved from NAS.

As explained by the research findings of security analyses performed in [2], amongst NAS and other existing anti-counterfeiting alternatives with a centralized architecture, utilizing wireless tag communication technologies, there are three common counterfeit attacks in the existing centralized anti-counterfeiting and traceability solutions: (1) modification of product records stored in tags, such as fabricating product identifiers or metadata for any wine product, (2) cloning of tag metadata such as those genuine wine records to any counterfeit product tag, and (3) removal of legitimate tags from genuine wine products with reapplications to other counterfeit wine products.

The typical architecture of centralized supply chains creates several concerns, as discussed in [2]. First, there is a tremendous processing burden on servers, since significant

numbers of products are flooded through multiple supply chain nodes. Second, substantial storage is required to store authentication records for every single processed wine product. Third, with existing solutions based on a centralized architecture, traditional supply chains have the inherent problem of single-point failure, and so potential service downtime and data loss could be expected. The legacy of NAS relies on centralized authorities, such as winemakers, to combat counterfeit products which could, in all likelihood, result in *single-point processing, storage, and failure*.

To overcome the concerns of software solutions which are developed with centralized architecture, Blockchain Technology (or other Distributed Ledger Technologies built with decentralized peer-to-peer networks) stands out as a potential framework to establish a modernized, decentralized, trustworthy, accountable, transparent, and secured supply chain innovation against counterfeiting attacks, compared to those built with a centralized architecture, with a comparison between the two detailed in Figure 1. In this research, the aim is to develop and implement a novel prototype with a decentralized architecture, based on the legacy NAS, to reinforce the innovative idea of product anti-counterfeiting in a decentralized fashion, namely, the *Decentralized NFC-based Anti-Counterfeiting System (dNAS)*.

Decentralized Architecture with Blockchain	Centralized Architecture
New data is only added when consensus is reached	New data is added via administrator without any consensus reached
Can only insert new data, old data is immutable	No restriction on any data modifications
Distributed in nature	Single point of failure
Decentralized in nature	Single point of control
Peer-to-peer structure with cloud instances	Client-server architecture
Cryptographic verification	Server performs actions on users' behalf
Resiliency and availability increased with number of peers	Backups and contingency plans are manually implemented
Cryptographic authentication and authorization	Cryptography implemented separately as add-ons

Figure 1. Comparison of Decentralized and Centralized Architecture.

1.3. Research Objectives

The open research questions, and the difficulties inherent in addressing them, should be detailed before actually diving into the system design of the proposed dNAS. It will then be followed by an outline of a preliminary design of the system model and an effective blockchain-based system architecture running on a cloud-production environment. Given the variety of advantages, such as the prevention of single-point failure, as well as the better resilience and availability of its being applied to the supply chain participants, that are brought by the blockchain technology and the concept of decentralized application, which lead to a more secured and sophisticated supply chain system against counterfeiting attacks, the main research question should be:

“How is it feasible to decentralize an anti-counterfeiting application, already developed and implemented in the supply chain industry, to better combat the rampant counterfeiting attacks?”

In order to progress the research with a prototype of decentralizing the aforementioned legacy NAS, the main question is then addressed by answering the following sub-questions throughout the research:

1. How are the chosen blockchain and its consensus protocol being implemented in the proposed dNAS?

2. How are the current operations of wine record management and product provenance verification reengineered to be a more decentralized, secured, autonomous and economical alternative?
3. How are wine record data being stored in the proposed decentralized and distributed storage architecture?

Given the main research question and the identified set of derived sub-questions, we aim to find an organized way of exploring them step-wise in this research. The best approach is to follow research techniques in computing, namely *proof-by-demonstration* and *implementation-driven* research techniques, under which a working prototype of dNAS will be introduced and validated with the idea of the decentralized supply chain solution, addressing the main research question and the derived sub-questions. It is important that, during development, the next best steps are always taken into account and ultimately offer a better analysis of the research questions with detailed documentation of the development of such an innovative and decentralized solution, using blockchain technology and other useful technical concepts, to improve the current situation of product counterfeits in the supply chain industry.

2. Background

In this section, following the advantages brought by the blockchain technology, a variety of current blockchain implementations applied to different categories of the supply chain industry are also mentioned, with reasons why the proposed dNAS could be a novel and feasible innovation to improve product counterfeiting as a primary motive for developing and implementing dNAS. The idea of developing decentralized applications, acting as a contribution of this research, which the development of dNAS is based on, will further be elaborated.

2.1. Related Work of Blockchain Implementations

With the advancement of blockchain development in recent years, there have been some existing blockchain innovations developed on different domains and in combination with other emerging technologies for different purposes. A blockchain-based digital certificate system and blockchain-enabled system for personal data protection were proposed in [4,5] respectively. Refs. [6,7] have also given overviews of blockchain-based applications developed on different domains where a variety of examples of blockchain systems and use cases could be found in healthcare domain as depicted in [8,9] for decentralized health record management and storage, and energy domain [10] for electric e-mobility, grid management and Peer-to-Peer energy trading.

Blockchain technology has also been utilized for integration with other technologies, such as *Internet-of-Things* (IoT) and *Artificial Intelligence* (AI), on use cases, such as [11,12], to enable the interoperability of an IoT device in a decentralized environment with *Fog computing* coming into play as the orchestrating layer, ref. [13] on a blockchain-based access control layer to the IoT data storage, and [14] on decentralized AI applications for enhanced data security, improved trust in robotic decisions, and decentralized intelligence.

Blockchain innovations have also been implemented across the supply chain industry, and for the use case of improving the product traceability and anti-counterfeiting aspects of the industry. The authors of [15] have proposed a blockchain system concept to enhance the transparency, traceability and process integrity of manufacturing supply chains, while an Ethereum-based fully decentralized traceability system for the Agri-food supply chain management, named AgriBlockIoT, was developed in [16]. Furthermore, a novel blockchain-based product-ownership management system, a blockchain-based anti-counterfeiting system coupled with chemical signatures for additive manufacturing and ontology-driven blockchain design for supply chain provenance, are also detailed in [17–19], respectively.

However, the aforementioned blockchain implementations tend to only support the findings with conceptual designs of how blockchain technology could be utilized to achieve

better provenance, traceability and process integrity, but few have really been practically implemented, with insightful evaluations of the prototypes and strategies regarding the actual implementation and integration in any category of the supply chain industry. Issues such as system availability, system integration model, security considerations, data integrity with the actual decentralized data management operations and system functionalities beyond retailer points remain to be addressed.

Some implementations are conceptualized and developed basing on computation-extensive permission-less blockchain networks and consensus algorithms, aiming at full decentralization over the scalability and stability of the developed decentralized systems. Instead of developing blockchain implementations based on conceptual design, decentralizing a legacy anti-counterfeiting system already implemented in the supply chain industry, with blockchain innovations integrated with, *would be a more pragmatic way to begin* so as to provide timely support to improve the current situation of product counterfeits of the supply chain industry, with a novel dNAS being developed and implemented.

2.2. Contribution—Building the Decentralized Application (DApp)

The research aims to investigate the development of decentralised applications (DApp), utilizing the available Enterprise Blockchain frameworks, enabling distributed applications to function with unprecedented versatility defined in smart contracts, with a distributed file storage system, such as *InterPlanetary File System* (IPFS) and *Swarm*, suggested in [20,21], respectively, and how it could bring advantages over the legacy solutions in supply chain anti-counterfeiting and traceability. It focuses on the idea of being provably honest under which users of the decentralized application should be able to automatically verify the actions requested by the server instances of other nodes along the supply chain whenever they would like to have the server instances checking with the decentralized storage and decentralized networks with the validation results returned.

dNAS aims to deliver a more secure and quality approach to verifying the authenticity and provenance of luxurious products such as bottled wine, with the use of a peer-to-peer Blockchain Network of Blockchain 2.0 implementations, as explained in [2], based on the concept of enterprise consortium, and a peer-to-peer distributed file-storage system, such as IPFS, which has the potential to eliminate an absurd amount of costs for on-chain storage and provide enhanced privacy, reliability and quality of service compared with the legacy NAS with centralized architecture.

Contrary to existing conceptual blockchain implementations applied in different industries, this research is about the decentralization of a legacy anti-counterfeiting solution already implemented in the supply chain industry, to improve product counterfeits for the industry. *Practicality* in terms of the system implementation, integration model, data integrity, system scalability and stability, is what this research will aim to achieve. The proposed prototype depicted in this research not only stands as a working DApps prototype, but also defines a framework and practice for different nodes along the supply chain to integrate the low-cost, real-time and immutable blockchain technology into their daily supply chain workflows.

For entities to embrace this budding technology with confidence, dNAS needs to be secured, reliable and flexible for the available integration models, and the registered supply chain nodes adopting it should find dNAS straightforward to be implemented. Similarly, for consumer interaction with the user interface of dNAS, it needs to be user-friendly and transparent. dNAS itself does offer a foundation to explore the strengths and weaknesses of decentralized applications over their centralized counterparts, especially in the supply chain industry. The evaluation of the proposed dNAS against the predefined research objectives, and in areas such as decentralized wine record management, data and process integrity in dNAS, and system security, availability and integration, would also uncover certain limitations and concerns regarding the hosting and deployment of DApps on the Enterprise Blockchain or with other Blockchain technologies and frameworks. The proposed dNAS could further be improved with potential future work inspired by the

findings on the limitations and concerns raised in the system evaluation process of dNAS, which is elaborated in [22].

3. System Model Definition

dNAS is a decentralized and integrated system, tailor-made for the supply chain industry, and specifically the wine industry. While capabilities such as end-to-end traceability, provenance and authenticity were enabled in the legacy NAS, its anti-counterfeiting and traceability model was actually based on winemaker roles, with their products moving downstream to their registered supply chain participants and wine consumers until the purchase points. The anti-counterfeiting model of dNAS is actually based on a decentralized *enterprise consortium* consisting of multiple winemakers and supply chain participants, responsible for operating the web application and the NFC-enabled mobile applications of the legacy NAS with a dedicated blockchain network, via an integrated interface, with blockchain nodes assigned to every consortium member. This section will discuss the use-case analysis and proposed system architecture of dNAS.

3.1. System Requirements of dNAS

According to the potential opportunities and concerns of developing decentralized solutions for supply chain anti-counterfeiting and traceability, explained in the research discussion result of [2], a fundamental set of system requirements for developing dNAS is defined.

There are, in total, 19 fundamental system requirements, as listed in [2], to develop dNAS, with rationales ranging from data integrity, scalability, data privacy, improved authentication and authorization, ease of integration and consensus performance to state transparency, summarizing what the supply chain industry, and even the specific wine industry, expected from the innovative dNAS, with potential opportunities and concerns in the decentralization of the supply chain anti-counterfeiting and traceability, which are also addressed through the development and implementation of dNAS.

3.2. System Roles' Definition

Before explaining the system architecture of dNAS based on the gathered system requirements, the constituent system roles and their corresponding use cases are clarified. In dNAS, the decentralized architecture and anti-counterfeiting mechanism are built around a concept of enterprise consortium consisting of winemakers and the supply chain participant nodes along the supply chain. Similar to those defined in the legacy NAS, there are four main system roles in the proposed system model of dNAS, depicted as follows.

The Winemaker Role: The winemaker is the manufacturer of wine products. The provided functions of winemakers include (1) creating new wine records with a set of wine pedigree data, supply chain data and transaction data, (2) proposing that their supply chain participants are added to or removed from the consortium, as well as their designated blockchain nodes to the blockchain network, (3) setting the quantity of wine products available to be sold, and (4) retrieving information regarding its supply chain participants so that the latest status of sales can be retrieved. The winemaker role can also inquire about any wine product their participants have been marketing to wine consumers, and verify whether specific wine products have been recalled or exchanged, or confirm if the current status of these wine products has been verified with public addresses provided by related wine consumers.

The Supply Chain Participant Role: The supply chain participants could have the role of wholesaler, retailer or reseller, and literally any supply chain node after winemakers and prior to wine consumers, along the supply chain. Participants can use functions of dNAS to encrypt wine record components using private keys, and wine consumers can, therefore, utilize the concept of asymmetric cryptography, which is applied to the system functions of its blockchain service, hosted by the consortium administrator. The participants' public addresses are utilized to verify if the participant is who they claim to be. After performing

each transfer operation, the participant will update the data fields of the transaction data and supply chain data, and the updated content hash is, therefore, pushed on-chain, and available for the registered nodes of winemaker role and supply chain participant role, to obtain the required data.

The Wine Consumer Role: The wine consumers are the end purchasers of wine products. The wine consumer can verify whether a specific supply chain participant is in a business relationship with another winemaker, and also verify whether the supply chain participant has an inventory of specific wine products. Wine consumers can prove that their identities are consistent with their public addresses, and obtain the individual transaction records and wine statuses of wine records via linking the corresponding blockchain nodes to a specific block explorer to review targeted blocks and transactions. As long as a wine product is transferred by a registered node of the wine-consumer role, the authenticity and end-to-end traceability will be assured and legitimate, even beyond any retailer point post-supply-chain.

The Consortium Administrator Role: A *consortium* consists of winemakers and supply-chain participants. The consortium administrator is elected through a democratic process involving every consortium member, who is assigned their own blockchain service instance with a designated blockchain node running on the blockchain network, which is managed by the consortium. The proposed dNAS would assume that there is only one administrator node elected for the *enterprise consortium* in this research, for ease of operation and presentation. The possible approach of having on-chain governance aspects to decentralize the enterprise consortium, or enhance the degree of dNAS decentralization as a whole, should be covered in future work on implementing dNAS.

3.3. Use Case Analysis of dNAS

Given the defined system roles of dNAS, a use case analysis based on these system roles was performed, as demonstrated in Appendix B.1. dNAS is developed basing on a concept of *enterprise consortium*, as depicted in Figure 2, consisting of multiple registered nodes of the winemaker role, supply chain participant role and the consortium administrator, introduced to dNAS, unlike the legacy NAS on which the anti-counterfeiting system and mechanism are based *solely on winemakers*.

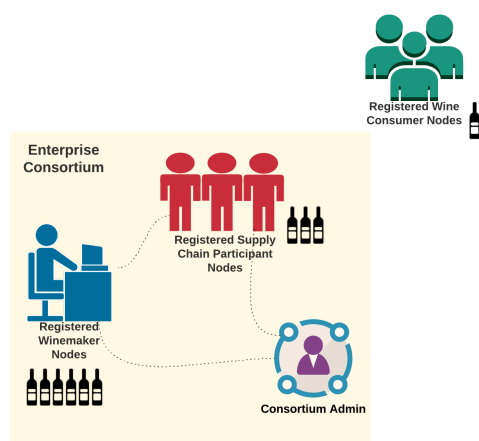


Figure 2. Enterprise Consortium of dNAS.

Getting started with the blockchain system, as depicted in Figure 3, there are use cases only accessible to the consortium administrator, such as the management of smart contracts, which are decentralized software design patterns and methods with access shared across different blockchain nodes running on the network, based on the authorization mechanism defined. The smart contract will need to be developed, deployed or even upgraded to the blockchain network if necessary. The consensus level is indeed the number of votes from consortium members, required for a consensus to be reached to confirm that a registered

node will be added to or removed from the consortium. The consensus level can only be set by the consortium administrator if necessary, especially for a situation where the size of the consortium has changed, drastically affecting the degree of decentralization and fraction needed to reach a consensus. Other consortium members are able to propose the addition of a new node to, or removal an existing node from, the consortium, based on the public address of their designated blockchain node, which is stored in an on-chain registry. Every consortium member would be assigned with a blockchain node, and are therefore eligible to send and validate transactions of the blockchain network to help reach a consensus and state broadcasting.

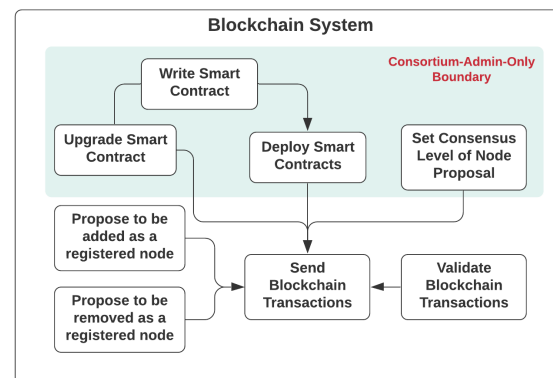


Figure 3. Use Cases of dNAS Blockchain System.

Regarding the use cases of the database-operating web application, as listed in Figure 4, any registered node of the winemaker role and supply chain participant role can obtain access to the functionalities provided by the web application to manage the inventory of wine products they hold, the list of collaborative supply-chain participants, and projects involving different wine products with different supply chain participants. A wine record, which could be created *only* by registered nodes of the winemaker role, consists of four constituent data categories, such as wine pedigree data, transaction data, supply chain data and unsuccessful validation data.

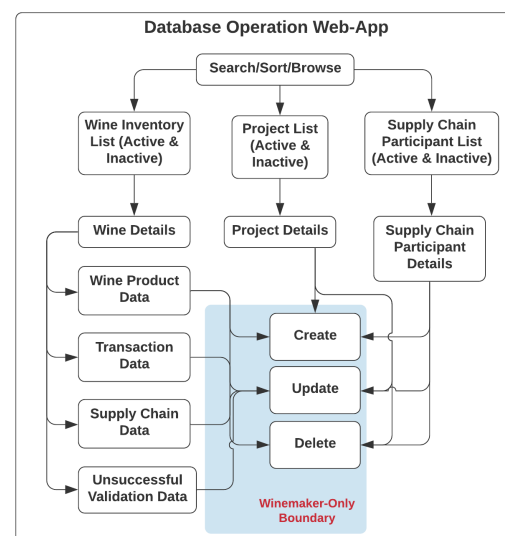


Figure 4. Use Cases of Database Web Application.

Likewise, only registered nodes of the winemaker role could perform operations such as create, update and delete in the web application, while registered supply-chain participant nodes and registered wine-consumer nodes could only update the respective transaction data and supply chain data via *ScanWINE* mobile application at the point of

wine acceptance or purchase, following a series of data validation steps, when scanning the NFC tags on wine products.

Like the legacy NAS, only registered nodes of the winemaker role could access the NFC-enabled tag-writing mobile application-*TagWINE*, with use cases as listed in Figure 5, and perform its functionalities, such as viewing wine records on the mobile application, recording required data fields of the wine record in NFC tags, and checking the history of the previous writing activities for those tags on the wine products they produced.

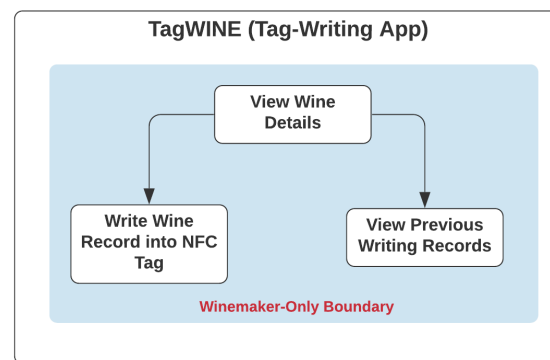


Figure 5. Use Case of Tag-Writing App-TagWINE.

While, for the NFC-based tag-reading mobile application-*ScanWINE*, with use cases as listed in Figure 6, any registered nodes, including those of the wine-consumer role, is accessible to the functionalities, such as scanning NFC tags, which will also trigger the process of validating the provenance and authenticity of wine products by analysing the data stored in NFC tags, a unique identifier of scanned NFC tags and the identity of registered nodes.

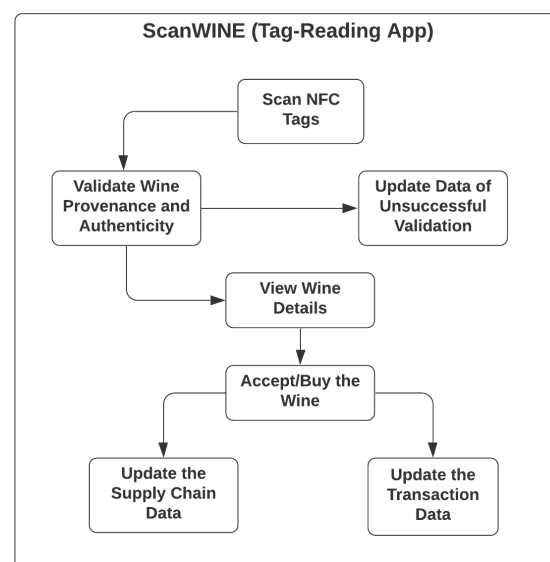


Figure 6. Use Case of Tag-Reading App-ScanWINE.

In case such a validation process fails, the unsuccessful validation data of that wine record will be updated, and related winemakers and supply chain participants would be notified of the new status, shown on their database management web application. After going through a successful validation process, the wine record will be shown on the mobile application, with further options for wine consumers to buy, or supply chain participants to accept, the wine product, with physical wine products at their end. The transaction data and the supply chain data are, therefore, updated following the activities of purchasing or accepting wine products are confirmed.

The account management of registered nodes, with use cases listed in Figure 7, is actually based on the design patterns developed in *AccountController* of the app-backend service in legacy NAS, for which every node could go through registration steps and become registered nodes of either the winemaker role, supply-chain-participant role or wine-consumer role. The on-board process and registration steps of these aforementioned roles will be different depending on how each type of registered node could obtain access to different system components, including the database-operating web application, *TagWINE*, *ScanWINE* and the blockchain network, based on their roles and whether they are members of the *enterprise consortium*. The *AccountController* offers functionalities including validating the role of specific registered nodes, and voting to add or remove registered nodes of the *enterprise consortium*.

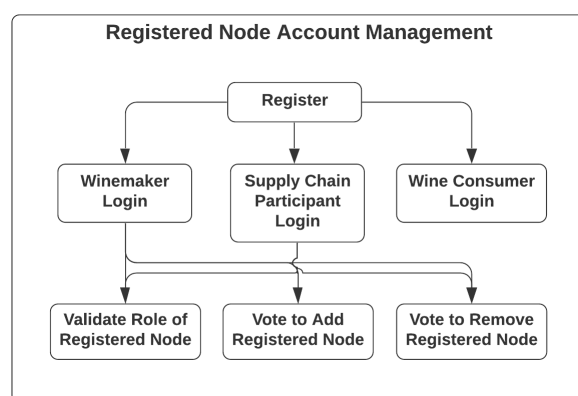


Figure 7. Use Case of Registered Node Management.

4. The Decentralized NFC-Enabled Anti-Counterfeiting System-dNAS

Following the identified system model definition, a system architecture of dNAS is then designed accordingly, in order to answer the main research question of this research. With the proposed system architecture of dNAS as demonstrated in Appendix B.2, how exactly the legacy NAS could be re-engineered and run on the chosen blockchain network of enterprise blockchain implementations, such as Ethereum Proof-of-Authority, is explained, as well as the system components that will be decentralized to enable core functionalities of the novel dNAS. The proposed system architecture of dNAS can largely be explained in three parts, namely (1) the decentralized blockchain network, (2) the blockchain interface, and (3) the system components built in the legacy NAS.

4.1. The Decentralized and Distributed Blockchain Network

The decentralized blockchain network, as demonstrated in Figure 8, is developed based on Enterprise blockchain implementations such as the Quorum blockchain with a consensus algorithm of *Istanbul Byzantine Fault Tolerant* (IBFT), and Ethereum blockchain network with a consensus algorithm of *Proof-of-Authority* (PoA), in line with the concept of enterprise consortium, introduced in the system design and consisting of client nodes which could be either validator nodes (full node) involved in validating transactions and signing a new block in the blockchain, or the listener node (light node), which can only listen to blockchain states and perform non state-transitioned interactions with the smart contracts deployed to the network. Whether a validator node or listener node is assigned depends on the role of authorities in the enterprise consortium. For instance, the winemaker role would be assigned with a validator node of the blockchain network, while some of the supply-chain-participant nodes would be restricted to a listener node, depending on their involvement in the enterprise consortium.

The dNAS blockchain network could be built basing on enterprise blockchain implementations in the Quorum blockchain, Ethereum blockchain, Hyperledger Fabric or Tendermint. If the *Proof-of-Authority* blockchain network of dNAS is developed, it could be

built basing on different Ethereum client implementations, such as Go-Ethereum, which could specifically be termed the Clique Proof-of-Authority network or Aura Proof-of-Authority network. If the blockchain network is developed with the Quorum blockchain, the *Istanbul Byzantine Fault Tolerant* network could be developed with Quorum client implementations, such as Go-Quorum. No matter which enterprise blockchain implementations the blockchain network of dNAS will be based on, it will consist of a set of client nodes managed by different authorities of the enterprise consortium, thus enabling the peer-permissioning functionalities, and so the network is functioned as a permissioned network, unlike public networks such as the Ethereum main net, which is an open network and does not emulate the concept of peer-permissioning, due to the fact that the proposed dNAS is developed for the supply chain industry, and specifically for the wine industry. When starting the client node, individual nodes are connected to each other, via specification of the node identifier of any client nodes already running on the network, so as to form a network.

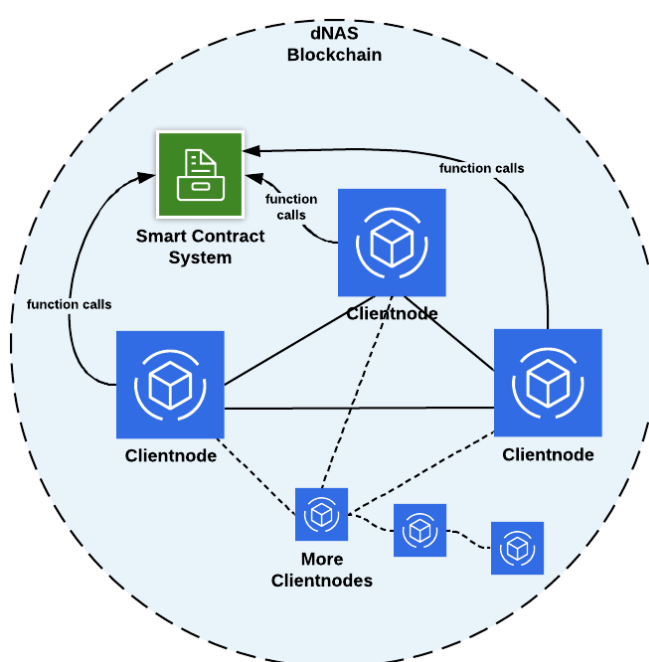


Figure 8. The dNAS Blockchain Network.

Smart contracts are developed in solidity and deployed to the dNAS blockchain network by the consortium administrator, who is also hosting validator nodes on the network on behalf of consortium members. Whenever a transaction request approaches the blockchain network, the transaction-sending node will invoke methods defined in the smart contract accordingly, based on an application binary interface (ABI) specified in the data fields of a transaction to perform state-transitioned changes in smart contract storage as well as the states of the network, with transactions validated and a new block mined. The smart contracts can further be upgraded through the implementation of a smart contract upgradability pattern. The management of smart contracts with standard processes such as deployment or upgrade is handled by the contract deployment suite, which only blockchain nodes of the consortium administrator will have access to, and perform such operations on these smart contracts.

There are also network monitoring components, such as the blockchain explorer and node management suite, that can be deployed and connected to the blockchain nodes running on the network. The former is a user interface connecting blockchain nodes on the network for authorities to perform querying and viewing on validated transactions and mined blocks, including those at pending stage. The latter is a dashboard listing the different performance metrics of every blockchain node running on the network.

The data collected from these network monitoring tools could further be augmented and integrated with any open-source data visualization tools, such as Kibana, with the enhanced monitoring capability of dNAS as a whole.

With the advent of the decentralized blockchain network integrated with system components of the legacy NAS via a tailor-made blockchain interface with other tools, the suggested architecture pattern can then provide a high-throughput and content-addressed block storage model to the architecture, which greatly improves the performance of the proposed dNAS. Similarly, regarding the advantages contributed by adopting Blockchain technology, it has been able to prevent single-point failure, and nodes along the supply chain do not essentially need to trust, but collaborate with, each other in the enterprise consortium.

4.2. The Blockchain Interface

Given that the decentralized blockchain network is regarded as a vital part of the proposed dNAS, a dedicated blockchain interface, as demonstrated in Figure 9, which is essentially another microservice, will need to be designed and developed to help integrate the decentralized bits into system components of the legacy NAS. The blockchain service, acting as an interface between system components of the legacy NAS and the blockchain network, is developed basing on functionalities, offered by open-source Ethereum interface libraries such as *Ethers.js* and *web3.js*, aiming to interact with the designated blockchain node and, in turn, the blockchain network. API endpoints are also developed based on logic patterns in different controllers of the blockchain service.

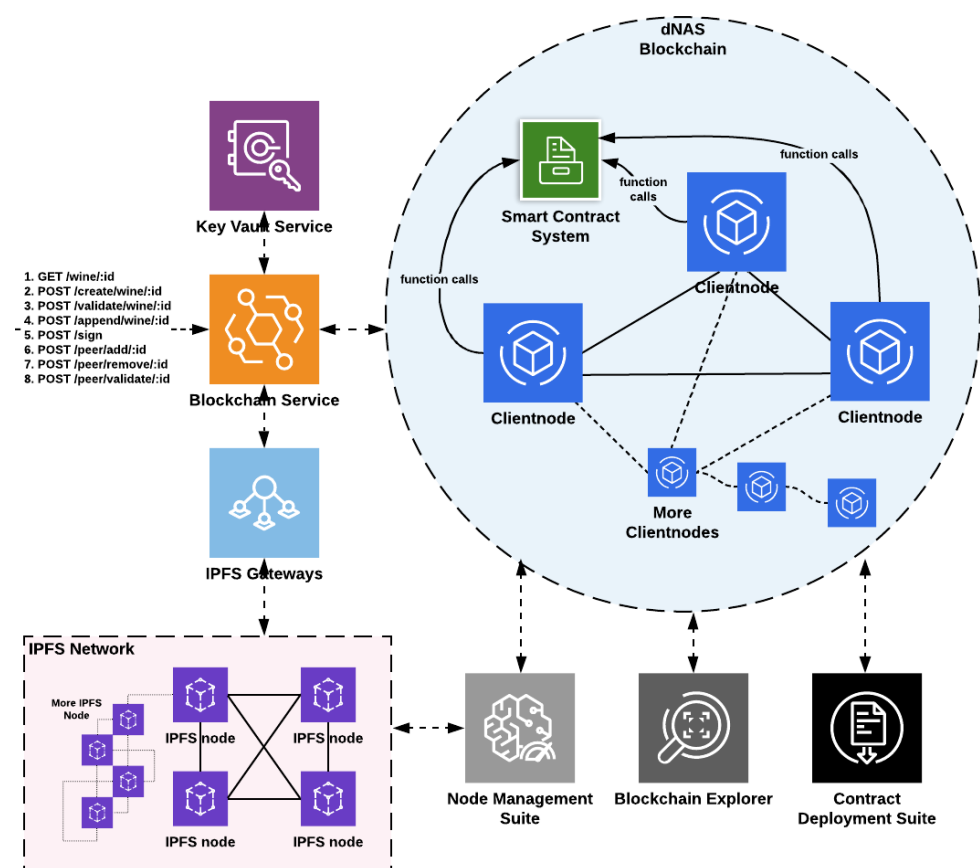


Figure 9. The Proposed Blockchain Interface of dNAS.

The *AppController* of the app-backend service in the legacy NAS could invoke endpoints to perform different functionalities, such as pushing data on-chain, getting or validating data from the deployed smart contracts, executing peer-related operations, including votes to add or remove participating nodes, and validating the roles of registered

nodes in the enterprise consortium. The blockchain network is also utilized as a controller of the enterprise consortium managing access control and the identities of registered nodes by implementing a concept of on-chain peer registry, and serving as a tamper-proof log of transaction events emitted back to the blockchain service.

With different helper functions developed in the blockchain service and functionalities enabled with open-source Ethereum interface libraries, the blockchain service instance can, therefore, direct its designated blockchain node to send transactions so as to perform different operations related to wine records and the peer registry. Sending transactions will require a signature produced with a private key of the node account. Therefore, a dedicated key-management module, such as the key vault service, will need to be in place to store key pairs related to any signing activities involved in any transaction processed by a specific node account, which is also linked with the blockchain service instance, and to providing a secured mechanism for any key pair to be retrieved whenever there is a transaction or a signing process.

Instead of storing the complete version of the wine record with all of its constituent metadata, such as transaction data, wine pedigree data and supply chain data, only the content hash of the wine record along with other payloads for data security and privacy should be stored in the data structure defined in the deployed smart contracts. A distributed data storage system, such as IPFS, is, therefore, introduced and migrated to the proposed dNAS, so that the dedicated and distributed IPFS node could be connected to an instance of the blockchain service whenever there is a request sent to the blockchain service to process. Storing wine records in such a distributed storage system provides a unique content hash which would then return to the blockchain service for further operations before being included in any transaction, and then be signed by dedicated node accounts and sent to the transaction pool of the blockchain network queuing to be processed for validation.

The content hash of a specific wine record is then retrieved from the smart contract whenever there is a validation process to be executed, and further used to retrieve wine record subsets from the distributed storage system off-chain, via the designated IPFS node. IPFS speaks of a permanent web, implying that any data published to the network should be available forever. The node management suite is also connected to IPFS nodes so as to have the same monitoring feature applied to every distributed IPFS node that is assigned to registered nodes of the enterprise consortium.

4.3. The Legacy NAS Components

According to system components of the legacy NAS, as demonstrated in Figure 10, the applications provided to different registered nodes along the supply chain are the database-operating web application, *TagWINE*—the NFC-enable tag-writing mobile application and *ScanWINE*—the NFC-enabled tag-reading mobile application. The mobile applications are designed to interact with the NTAG 203 of NFC tag model.

The functionalities, enabled by those system components of the legacy NAS, substantiating the use cases described in the use case analysis, are actually based on logic patterns in *AppController* of the app-backend service. For instance, Create, Read, Update and Delete (CRUD) operations on wine records, performed on the database-operating web application, actually send a request to *AppController*, invoking respective methods related to such operations, and collaborate with different system components, such as the database system developed in Microsoft SQL, to reflect states transitioned back to the web application. Another example could be using *ScanWINE* to scan NFC tags, and validate the provenance and authenticity of wine products, by validating and comparing respective states of specific wine records stored in the NFC tags, the off-chain database system connected with *AppController*, distributed IPFS storage, and the on-chain data storage of smart contract deployed to the dNAS blockchain network via the integration between the app-backend service and the blockchain service.

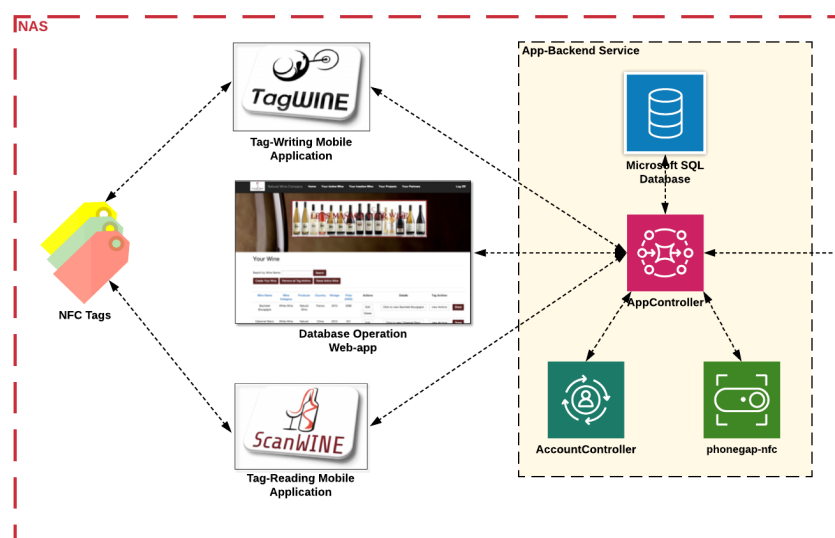


Figure 10. The System Components of Legacy NAS.

Regarding the components of the app-backend service, a database system is integrated with *AppController* to store states and versions of wine records, and *AccountController* processes the account management of registered nodes with states and user sessions stored in the database. The *phonegap-nfc* module is integrated with *AppController* to enable *ScanWINE* and *TagWINE* mobile applications with the functionalities of any NFC-related interaction, with the specific NFC tags packaged on wine products. Further logic patterns are developed and included in *AppController* for integration with the decentralized bits of the dNAS via the blockchain interface by invoking related API endpoints of the proposed blockchain service to perform functionalities, such as pushing data to the blockchain and validating on-chain data of specific wine records with transaction events returned to the app-backend service.

NTAG 203 Ferrite was the selected type of NFC tag for development of the legacy NAS, given its universal compatibility with most of the NFC-enabled devices and its resistance against metallic interference, for which the *NTAG 203 Ferrite Tag* can change the magnetic flux path to avoid interference to the NFC tag, with a high surface resistance and effectiveness in preventing resonance and suppressing coupling from the metallic interference, as NFC tags are likely to be attached to foil packaging materials surrounding wine bottlenecks. However, the *NTAG 203* chip is already out of production and has been replaced by those of the *NTAG21x* series, such as *NTAG 213* or *NTAG 216*, given their increased memory storage and security functions, including a password lock and scan counter. *NTAG 216 Ferrite* is, therefore, the selected type of NFC tag for the development of dNAS, owing to its extensive configuration with 7-byte *UIDs* and 888-byte *memory capacity*, for more flexibility in terms of the data stored in each NFC tag representing every unique wine product.

5. System Implementation

Given the system model design and system architecture elaborated, the actual system implementation steps of dNAS are expected. This section includes the proposed system implementation, with system operation protocols specified for the detailed system functionalities of the novel dNAS, on different wine record management process. The system implementation is categorized into two phases, namely the initialization phase and data-processing phase, covering system implementations, such as consortium formation and decentralized wine record management, on dNAS.

5.1. Initialization Phase

The initialization phase includes all the system implementation steps that need to be in place before dNAS can be fully functional for wine record management when wine products are moving along the supply chain. The initialization phase consists of consortium formation processes, with only the sample on-boarding process of new node to enterprise consortium covered in this research.

On-Boarding of New Node to Enterprise Consortium

After registering for an account to access the system components and applications of dNAS hosted by the consortium, such as the database-operating web application and both NFC-enabled mobile applications according to its defined system role, the registered node would then be given access to every dNAS system component. In order to become a consortium member, an on-boarding operation is needed to execute with a blockchain node instance assigned, which is hosted by the consortium. As detailed in the use-case analysis, only registered nodes of the winemaker role and supply-chain-participant role will be assigned with a blockchain node, an IPFS node and key pairs, which are shared and restrained to their own key vault service. While registered nodes of the wine-consumer role utilize shared instances of these distributed system components hosted by the consortium, for data processing operations, such as the wine record validation and appending process, wine consumers will consequently need to use the *ScanWINE* mobile application to validate a bottled wine with the returned wine record, before purchasing the wine product, with updated transaction and supply chain data updated in the dNAS. The suggested protocol for adding the new node to the consortium is described in Figure 11.

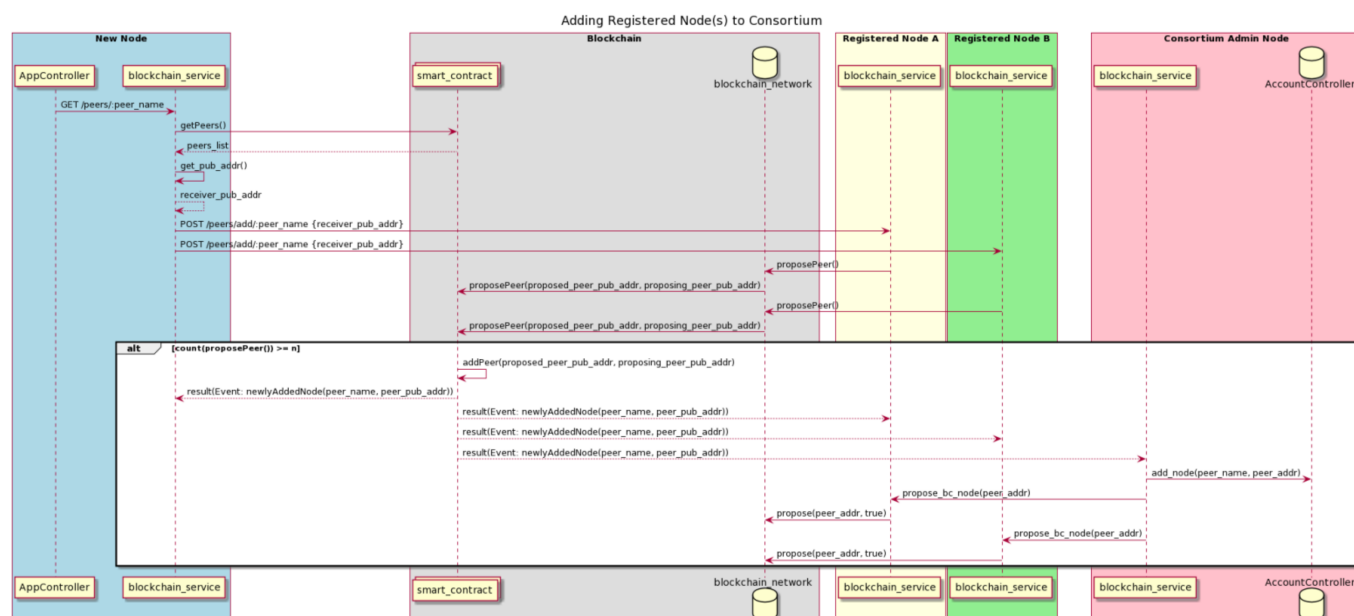


Figure 11. Adding New Registered Nodes to Enterprise Consortium.

To have a newly registered node added to the consortium, a democratic voting process is performed amongst the current consortium members. It is assumed that such an operation is kicked in after a new node is granted access to dNAS as a registered node, approved by the consortium administrator, and assigned the newly registered node of dNAS with instances of distributed components, including a blockchain node and an IPFS node. With the *URL* of the blockchain node instance specified, the blockchain service sends querying request to reach to the “*getPeers*” method (not limited by role restriction and so it could be accessed even if the blockchain node is a light node of the blockchain network) of the deployed smart contract to get a full list of peer details of existing consortium members.

The blockchain service instance of the proposed node then invokes the propose-to-add endpoint of blockchain service instance owned by every consortium member.

Once the votes of adding a proposed node to the consortium have amounted to half the consortium's size in terms of the number of participating members, *two consequences* would then occur: (1) the peer details of the proposed node will be added to the on-chain registry of consortium members, which is, in turn, a *white list* of the consortium members, to enable role-restriction in accessing the deployed smart contract's methods for wine record management, and (2) once the process on-chain operation is complete, the addition of the new node to the on-chain registry will occur, and be broadcast to the blockchain service instance of the consortium administrator based on the listener enabled in the blockchain service, which would, in turn, initialize a proposal to the blockchain network of adding the proposed blockchain node to the blockchain network, as well as sending requests to the counterpart of every existing consortium member to agree to this proposal via their own blockchain nodes, which run on the same blockchain network once it amounts to $N / 2 + 1$, in case the consensus algorithm of Proof-of-Authority is chosen for developing the blockchain network of dNAS.

There is also a so-called *bootstrap stage* applied to the on-boarding process, in which a specific starting number of registered nodes, e.g., five, that were proposed to join the consortium do not have to go through such a democratic voting process. Instead, the proposed nodes will be added to the on-chain registry of peer details directly, by the consortium administrator, as well as having their blockchain nodes added to and run on the network, so as to guarantee a large amount of blockchain nodes running on the blockchain network and enough of the consortium to make democratic decisions. It is also assumed that the voting mechanisms of electing a consortium administrator will not be covered in this research, and would be explained as a potential limitation to dNAS to develop the on-chain governance steps described in future works on dNAS. Similar to the operation of adding a registered node, there is also an operation which removes a registered node from the consortium, which is not covered in the research, given their similarity.

5.2. Data-Processing Phase

The data-processing phase includes all the necessary system operations regarding the decentralized wine record management along the supply chain, such as creation, validation and appending, involving different consortium members, with only the wine record creation process and wine record validation process covered in this research necessarily.

5.2.1. Wine Record Creation Process

Once a registered node of winemaker role is added to the consortium with its blockchain node as part of the blockchain network, the wine-record creation process will be started after the physical bottled wine is produced, packaged and ready to be shipped at the winemaker point, as the wine record creation process could only be performed by registered nodes of the winemaker role, such as the one performed by the Registered Node A, as described in Figure 12.

The wine-record creation process starts with the app-backend service, as described in the proposed dNAS system architecture of the Registered Node A. First, a request is sent to its blockchain service instance, by invoking the endpoint of *"/peer/validate"*, to confirm if Registered Node A is indeed a part of the consortium based on the on-chain validation with a public key, stored in its own instance of key vault service, compared to the version stored in the on-chain registry. Once it is ascertained that a requesting node is part of the consortium, its app-backend service instance would then be notified, and the off-chain creation process of a wine record would commence, including the processes of creating the actual wine record and storing the full wine record to the database of the app-backend service.

Invoking an endpoint of signing operations in the blockchain service would produce and return a signature on a *byte-array* of the wine identifier, tagging the identifier and

device identifier involved in the process. In order to write the required data to an NFC tag, Register Node A will then utilize the NFC-enabled *TagWINE* mobile application, which only the registered nodes of winemaker role could access, according to the use cases' analysis, to write the wine identifier, signature and the value of write counter on the tag itself. The "*wine_status*" and "*supply_chain_data*" are then updated: the former will have updated data fields with latest values, while new entries would be created for the latter whenever there is a new activity of any interaction with the tag. The protection mode, with a randomly predefined password automatically injected by the app-backend service, is also applied to NTAG 216 Ferrite NFC tag once the tag-writing process is completed.

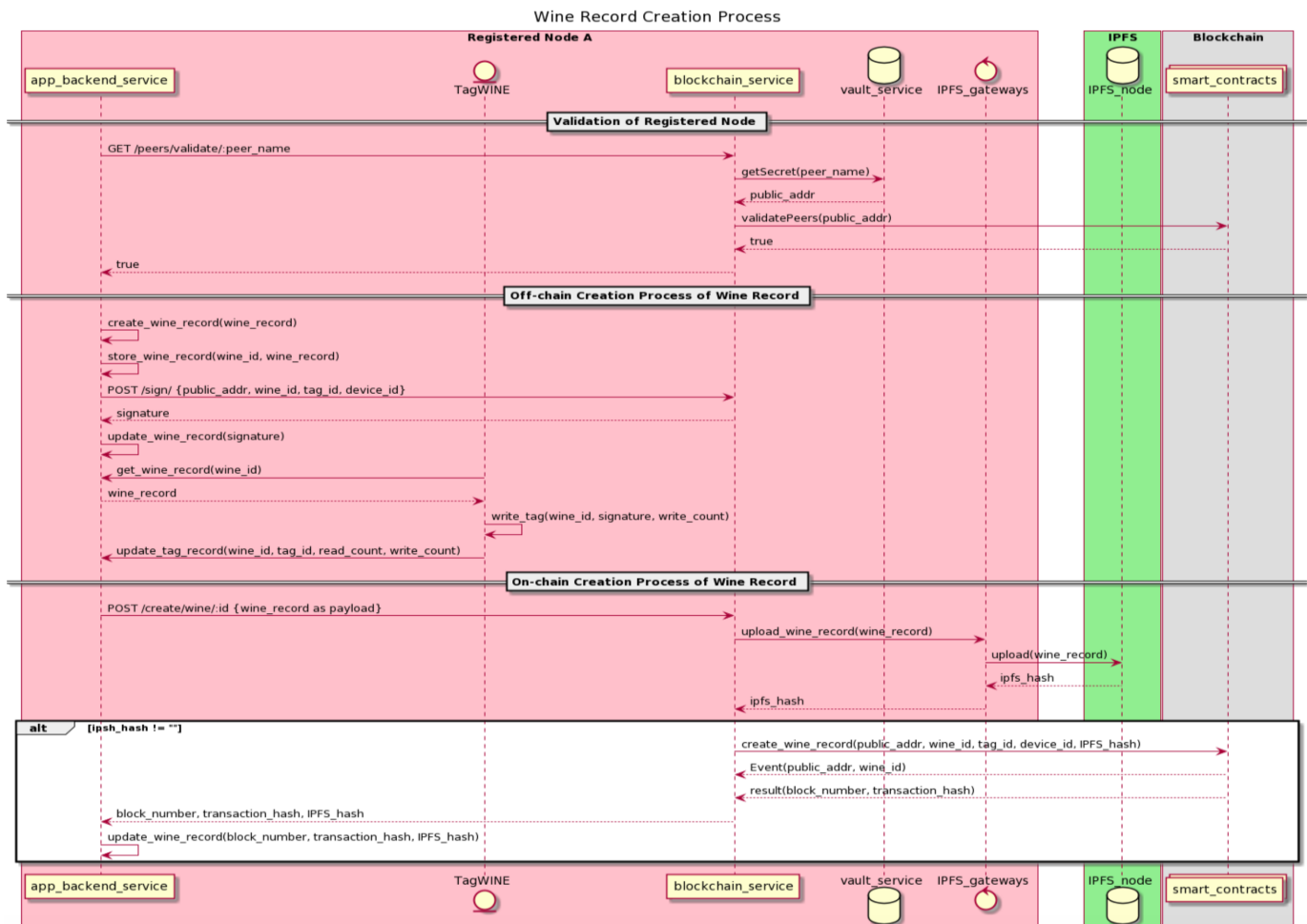


Figure 12. The Wine Record Creation Process.

After completing all the required processes of the off-chain creation operation, the on-chain creation process of a wine record is initiated. The app-backend service will invoke the *create* endpoint of the blockchain service, with the subset version of the wine record, instead of the full wine record, as the payload of the API call, and perform operations required on both the IPFS network and the blockchain network. Once such a wine record subset is uploaded to the IPFS network via a designated IPFS node, a 46-byte content hash based on *SHA256 hash algorithm* is then returned to the blockchain service. The content hash of IPFS, alongside the public address of the registered node and the unique identifiers of the wine product, tag and device, are all included in the payload when invoking the "*create_wine_record*" method of a deployed smart contract with transactions sent to the blockchain network.

The state of the on-chain storage of all these payload fields is updated when the transaction is validated and packaged in the block. The new block is mined by one of the blockchain nodes running on the network, with the respective block number and

transaction hash, and the event of “*create_wine_record*” is emitted on-chain and returned to the blockchain service. Both block number and transaction hash are then returned to the app-backend service, and the supply chain data and transaction data of the full wine record stored in the dedicated database of the app-backend service are updated. The corresponding physical bottled wine product is, therefore, ready to be shipped downstream to the next participant node along the supply chain, which is already registered with dNAS, to become part of the enterprise consortium.

5.2.2. Wine Record Validation Process

Once a wine record is created off-chain and on-chain by the registered node of a winemaker role, with the tag-writing process completed, those bottled wine products are then shipped to the next node along the supply chain. Assuming the next node along the supply chain is a registered node of dNAS as well as a member of the consortium, such as the Registered Node B described in Figure 13, the corresponding wine record of an incoming wine product would then be brought into a validation process, performed on its wine record, before the wine product itself can be accepted or purchased, via the NFC-enabled *ScanWINE* mobile application.

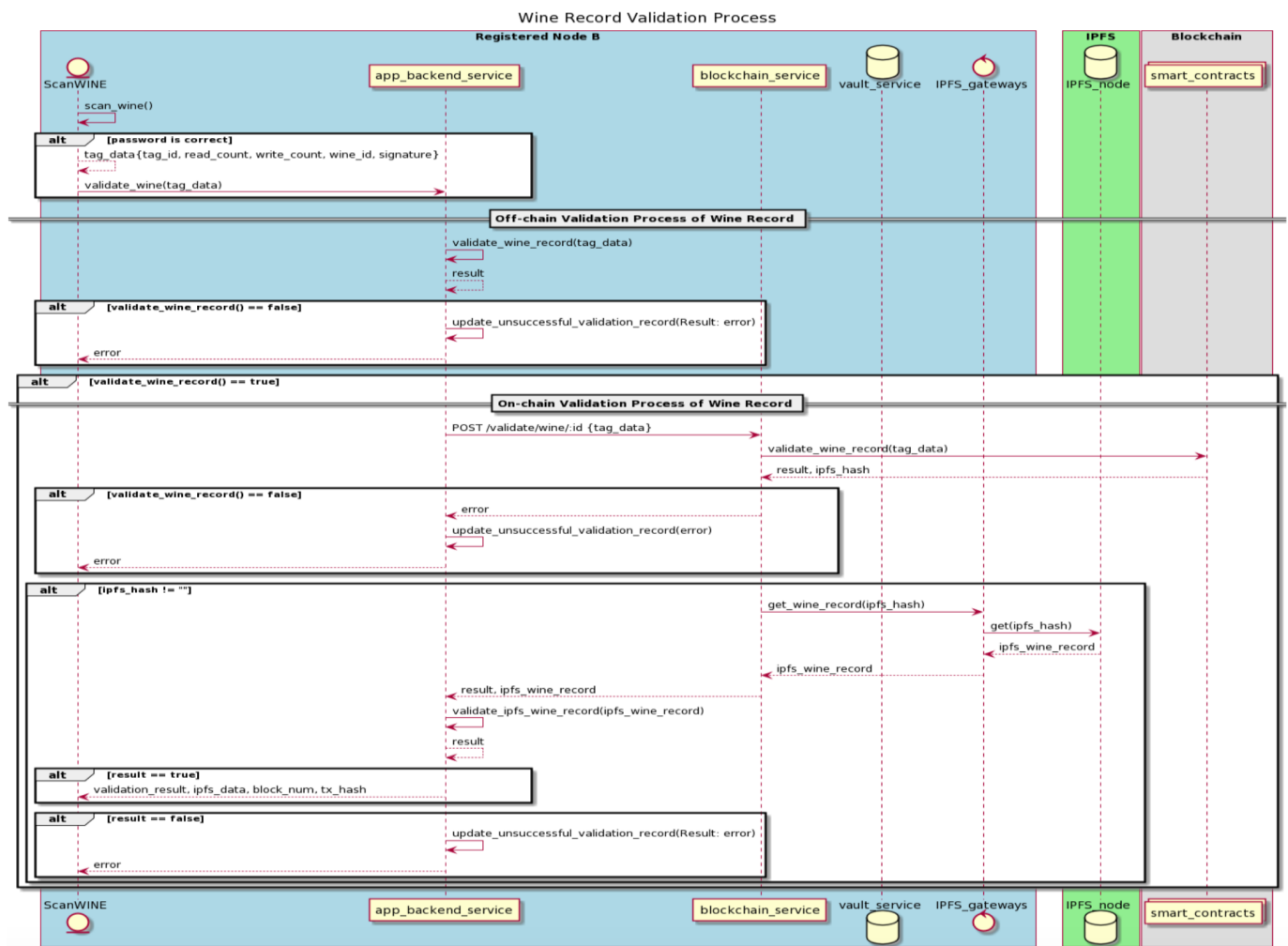


Figure 13. The Wine Record Validation Process.

The registered nodes of the supply-chain-participant role and wine-consumer role will then scan the NFC tags with *ScanWINE* to update the state of that wine record. *ScanWINE* is the only entry point for registered nodes of both roles to transit the state of a wine record, unlike the registered nodes of winemaker role, which could utilize the database-operating

web application to create and manage wine records. Such a wine record validation process is a *three-layered validation process*, against (1) the data stored in the database of app-backend service, (2) the on-chain storage, and (3) the updated wine record subset stored on the IPFS network.

When a registered node logs into *ScanWINE* on its device and scans the NFC tag of an incoming bottled wine product, it will automatically get through the password protection applied to the NFC tag, as the static password is always injected during the tag-reading process. The data written to the tag, mentioned in the creation process, will then return to *ScanWINE*, after which they can validate the wine record based on the data retrieved from the NFC tag. An off-chain validation process against the wine record will then take place to compare the retrieved tag data and those stored in the dedicated database of a shared instance of app-backend service. An error, classified as (1) modification attack, if wine identifier and the signature are inconsistent with the counterparts stored in the database, (2) reapplication attack when either the “*write_counter*” or “*read_counter*” is different from that of the wine record stored in the database, or (3) cloning attack, when an inconsistent tag identifier is found, would be returned to the app-backend service, and the “*unsuccessful_validation_data*” of the corresponding wine record will consequently be updated, with an error message shown on the user interface of *ScanWINE* application.

The on-chain validation processes of that wine record will therefore be initiated, provided that the off-chain validation process is completed successfully. The app-backend service will send a request to the blockchain service via its “*validate*” endpoint, which will further invoke the “*validate_wine_record*” method of the deployed smart contract, named “*wine_data_contract*”, with “*tag_data*” as the payload. The public address of the last registered node handling the wine product, which was involved in producing the signature from “*tag_data*”, is then recovered on-chain with the identifiers of both the wine product and corresponding tag of “*tag_data*”. The recovered public address is then validated against its counterpart, stored on-chain under the same wine identifier in a dedicated mapping storage on the smart contract.

Similar to the errors returned during the off-chain validation process, an error, either classified as (1) modification attack, where both the payload and recovered public address could not be found or matched on-chain, (2) reapplication attack, where the “*write_counter*” or “*read_counter*” are inconsistent with its counterparts stored on-chain, or (3) cloning attack, where an inconsistent tag identifier is found on-chain, would be returned to the app-backend service, and the respective “*unsuccessful_validation_data*” of a corresponding wine record updated with an error message shown on the user interface of *ScanWINE* application. Once the on-chain validation process is successful, the results and latest “*IPFS_hash*” of the specific wine identifier are then returned to the blockchain service, and the latest version of the wine record subset stored in the IPFS network could then be retrieved, with the “*IPFS_hash*” specified via its IPFS node. The wine record subset retrieved from IPFS network is then compared with its counterpart, stored in the database, interacting with the app-backend service. The validation result is then returned to the app-backend service and the user interface of *ScanWINE*.

With the successful validation of the wine record, the overview page of the corresponding wine record is then brought up on the user interface of *ScanWINE*, with wine pedigree data, wine status data and some transaction data fields, such as the unique transaction hash with a corresponding block number with which the registered node can query and confirm the latest block and transaction related to this wine product on a blockchain explorer connected to the blockchain network. The “*read_count*” of the tag is incremented after every tag-reading process, and so its counterparts stored on-chain and off-chain will also be incremented during every validation operation. The faulty and counterfeited wine product, with the error of any classified attack logged in the “*unsuccessful_validation_data*” of its wine record, should, therefore, be rejected and returned to its winemakers or the previously registered nodes of the wine products for further operations.

6. Developing the dNAS Prototype

Given the elaborated system architecture of dNAS, and the system implementation with the operational steps explained, the working prototype of dNAS is developed. The development process of dNAS followed the standard Agile software development methodology, involving software project management tools, software development environments, and configurations utilizing modern concepts, such as containerized applications, service-oriented architecture, source version control, and cloud environment deployment. The tools and concepts applied to the development process of the individual components in dNAS, such as the blockchain network and smart contracts, are also explained.

According to the system architecture of dNAS, decentralized system components of dNAS actually consist of a blockchain network, smart contracts, an IPFS network and a blockchain service. This section includes the implementation details and the design concepts of the individual decentralized system components of dNAS.

6.1. The dNAS Blockchain Network

The Ethereum developer community has developed many different client implementations, such as Go-Ethereum PoA Clique, Parity PoA Aura and Pantheon PoA, with options on consensus protocols provided, as well as other enterprise blockchain implementations, such as Quorum, Hyperledger Fabric and Tendermint, which are available for the development of a permissioned network. There are indeed benefits to having multiple Ethereum clients, such as (1) improved resilience to the network enabled by faster issue detection and correction—with more people interpreting the protocol specification, there is a greater chance of these errors being uncovered and detected—and (2) enhanced security—if there is an attack factor or bug in any of the Ethereum implementations, this means the network is usually working fine, as there is a wider diversity of clients available. *Go-Ethereum nodes* currently form the majority of the Ethereum network, including different implementations with different consensus algorithms, and a dNAS blockchain network is developed basing on the Go-Ethereum PoA implementation.

The network implementation started by creating Ethereum accounts for each of the blockchain nodes with a key pair, and its key-store file was generated during the process. The secret key could be decrypted if required, such as for signing transactions, with the specified password. Ethereum accounts are created and assigned to validator nodes, acting as consortium members, to be included in dNAS blockchain network.

The genesis file essentially defines the genesis block, also known as the first block of any blockchain network. There is some significant information set in the genesis file for starting the network; for instance, the *"chainId"* specifies the chain those validator nodes are connected to, the *"period"* specifies a block time for which a block is mined, regardless of the existence of transactions in the specified time interval, and the public addresses of Ethereum accounts generated are also under *"extraData"* to specify that these accounts are going to run as validator nodes when the network starts, with their public addresses also listed under *"alloc"* so they are allocated funds in *gwei* once the network is started. The genesis file is then initiated by every validator node that will be run on the same dNAS blockchain network.

A *"geth"* data folder, consisting of *"chaindata"*, *"lightchaindata"*, *"node states"* and *"transaction RLP (Recursive Length Prefix)"*, will then be created under each of the validator nodes after being initialized with the specified genesis file. The folder will serve as a local copy of global states, block states of specific chain and individual transaction states, transited in line with the progress of the dNAS blockchain network. The gas limit of the genesis block is also set in the genesis file for the network to begin. However, the gas limit of the block-to-mined is determined by the gas used in its parent block, *which is dynamic and not static in nature*, according to one set for the genesis block, specified as *"gasLimit"* in the genesis file. The mechanism set out in the Go-Ethereum implementation aims to increase the gasLimit if $\text{parentGasUsed} > \text{parentGasLimit} * (2/3)$; otherwise, it is lowered and the

amount increased/decreased is determined by how far away it can be from “ $parentGasLimit * (2/3) parentGasUsed$ ”.

The individual validator nodes are also ready to form a network with the specified Ethereum node identifiers. During the start-up process of a blockchain node, the gas price is set to zero, as it is not considered in any blockchain processes performed in dNAS. The APIs of *remote procedure calls* (RPC) are also available via RPC calls by specifying the internet protocol address of the running node or an *internal process call* (IPC) file under the “*geth*” data folder created for each of the running nodes, to consortium members and the consortium administrator, for any customized node operation. The node operations could propose the addition of new nodes to dNAS blockchain network via the following sample method, invoked by any blockchain node running on the network, as follows:

clique.propose(public_address,true)

With the number of such proposals amounting to more than $N / 2 + 1$, where N is the number of running nodes, the proposed public address, representing a proposed node, is then added and run as part of the dNAS blockchain network. The blockchain node could further be attached to any open-source blockchain explorer, such as *BlockScout*, with the “*ETHEREUM_JSONRPC_HTTP_URL*” specified based on the *provider URL* of the blockchain node.

6.2. The Decentralized Functional Smart Contracts

Smart contracts, hosting functional methods, have the core functional code to become fully decentralized and delegated to all the trusted authorities of the consortium and its blockchain network. Every blockchain node on the dNAS blockchain network must agree on the current state of the smart contract storage. This should be done in accordance with the smart contract mechanism and workflows, such as the source code compiled with the Solidity compiler and run by the Ethereum Virtual Machine of every blockchain node of the network, as described in Figure 14.

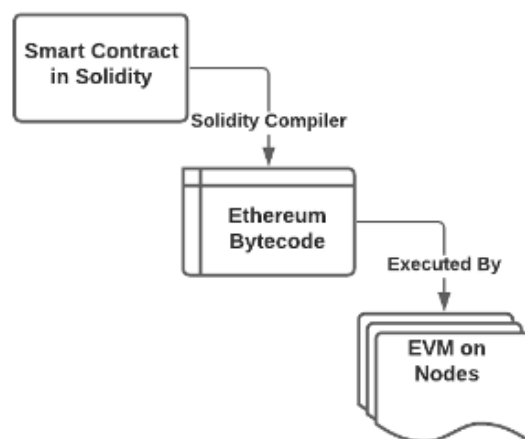


Figure 14. The Workflow of Ethereum Smart Contract Source.

The smart contract source code, deployed to the blockchain network, is implemented in Solidity, with the Truffle framework as its development environment and automated testing library. There are typically three smart contracts developed in dNAS, so as to cover every operation defined in the initialization phase and data process phase, namely (1) the “*WineDataContract*” for wine data operations, (2) the “*PeerRegistryContract*” for the consortium registry management, and (3) the “*Proxy*” which enables a proxy pattern for the purpose of smart contract upgradability.

In order to perform those operations defined under the data processing phase, the design patterns of mapping storage are defined in smart contracts, with individual data types also specified, such as the “*WineDataHash*”, which is the content hash (*Content ID*)

obtained from any IPFS process representing a specific wine record subset mapped to a unique wine identifier, which could also be further mapped per different write counts as “iteration”. Similar design patterns of mapping storage could be found to apply to the public key of the previous processing node, which is also a registered node of dNAS along the supply chain, and to identifiers of different tags and devices. The “currentImplementation” and “initializeCounter” are in place according to the proxy pattern, which is used to check if a contract address has already been initialized previously.

Events are also defined in the “WineDataContract”, and will be emitted when the on-chain process of the wine record creation or appending is completed. The events emitted in either of the operations are then captured by the event listener defined in the blockchain service. The creation process will include both the public address and hashed device identifier creating the wine record in its event, while the appending process will include both previous and current public addresses of the wine record.

Proceeding with the definition of individual methods, according to the wine record creation process, a specified wine identifier is checked and to validate if any records for the identifier have already been logged on-chain with the “require” statement. The individual variables mapped to the wine identifier are then updated with parameters, including a content hash obtained from IPFS node (WineDataHash), the public address, tag identifier and device identifier, of the request payload. The on-chain wine record creation process in Algorithm 1, is completed by incrementing the write count so that it matches the counterpart stored in the NFC tag, with the creation process also being emitted.

Algorithm 1: Creating On-chain Wine Record Entry

Input : wineId, wineDataHash, newPublicAddress, tagId, deviceId

Output: Boolean result of create operation

```

if MapWriteCount(wineId)==0 then
    MapPubAddr(wineId) ← newPublicAddress;
    MapDataHash(wineId) ← wineDataHash;
    MapTagId(wineId) ← tagId;
    MapDeviceId(wineId) ← deviceId;
    MapWriteCount(wineId)+ = 1;
    emit event;
    return true;
else
    return (err,errMessage);
  
```

In case a wine record is stored on-chain for a specific wine identifier with the corresponding physical wine product moved downstream to specific supply chain participants, the wine record must be validated with the wine record validation processes previously detailed. The on-chain validation of a specific wine record could be categorized into two parts, applied to both the wine record hash and the signature, which are also stored in the NFC tag. For the former, Algorithm 2 is a straightforward comparison between the content hash of a specific wine record subset stored in the IPFS network, and its counterpart, stored on-chain. It proves that the content of the wine record was not altered along the supply chain to the point where validation takes place.

The latter is used to validate the signature, with the sample on-chain method demonstrated in Algorithm 3, which was produced by the previous processing node with its private key on data such as the wine identifier, tag identifier and the device identifier, stored in physical NFC tags. The idea of validation on the signature is to recover the public key of the signature on-chain and to validate if the recovered public address is consistent with its counterpart stored under the specific wine identifier on-chain.

Algorithm 2: Validating the Wine Record (IPFS hash)

Input : wineId, wineDataHash
Output: Boolean result of validation
if MapWriteCount(wineId) != 0 **then**
 | return MapDataHash(wineId) == wineDataHash;
else
 | return (err, errMessage);

Algorithm 3: On-chain Signature Validation

Input : wineId, v, r, s
Output: Boolean result of validation
if MapWriteCount(wineId) != 0 **then**
 | $prefix \leftarrow 'Ethereumsignatureprefix';$
 | $encodedWineData \leftarrow encode(wineId, TagId_w, DeviceId_w);$
 | $hashedWineData \leftarrow hash(prefix, encodedWineData);$
 | return recover(hashedWineData, v, r, s) == MapPubAddr(wineId);
else
 | return (err, errMessage);

Due to the fact that both the tag identifier and device identifier are retrieved from the on-chain mapping storage, with the wine identifier as a key, during the on-chain validation process, both tag identifier and device identifier are also validated to ensure they are consistent with the version of the wine record stored in the off-chain database, the IPFS network and in the physical NFC tag as the constituents to produce the signature. Both methods of validation can confirm if a specific wine identifier has already been stored on-chain to make sure the specific wine record of the wine identifier has been created. Adding the *prefix* makes the calculated signature, with the *keccak256* hash function, recognisable as an *Ethereum-specific signature*.

Once the validation steps are completed successfully, the registered nodes are provided with options to accept a wine product, and when the wine product is accepted, its wine record will be updated with new transaction data and the supply chain data appended. The method, named “*appendWineRecord*”, is also defined in the smart contract for the on-chain appending process by updating the mapping variables accordingly.

6.3. Design Patterns of Smart Contract Upgradability

Deployed smart contracts are the shared functional code in the dNAS blockchain network and available for its designated blockchain nodes to perform the different operations requested. Just like any other software code, smart contracts are expected to be upgraded to offer new functionalities, fix software bugs, etc. The most straightforward way to update the smart contract is to deploy another smart contract and update each of the individual blockchain service instances of dNAS with the new address of the upgraded smart contract. However, this would require a new contract address to be specified for every restarting blockchain service instance, and there are multiple blockchain service instances, each owned by individual consortium members. The contract storage will also be lost and needs to perform a state migration with which every state-transitioned operation is performed again in order to sustain on-chain states aligned with the previous version of the smart contract.

Given the above drawbacks of any traditional mechanism of upgrading the deployed smart contract, the proxy upgradability pattern offers a common ground regardless of the rounds of upgrades with the same proxy contract; nevertheless, it will definitely increase trust in the consortium administrator’s ability to perform a smart contract upgrade for the consortium. The proxy contract uses the “*delegatecall*” function of EVM assembly code,

as demonstrated in Figure 15, pointing to the contract address of *WineDataContract v1*, as described in Figure 16, alongside the “*msg.sender*”, which is also embedded, and initiated by the consortium administrator instead of the proxy contract itself.

```
assembly {
    let ptr := mload(0x40)
    calldatacopy(ptr, 0, calldatasize())
    let result := delegatecall(gas(), impl, ptr, calldatasize(), 0, 0)
    let size := returndatasize()
    returndatacopy(ptr, 0, size)
    switch result
    case 0 {
        revert(ptr, size)
    }
    default {
        return(ptr, size)
    }
}
```

Figure 15. EVM Assembly Code with delegatecall.

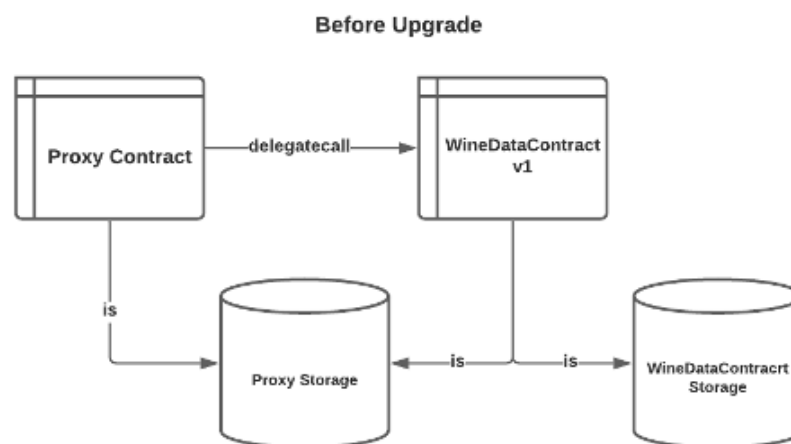


Figure 16. Proxy Pattern Before Smart Contract Upgrade.

The upgrade process is initiated and performed by the consortium administrator, according to the “*upgradeTo*” method defined in the proxy contract. The only thing that the proxy contract needed to operate is to point to a contract address of the upgraded smart contract, which is the *WineDataContract v2*, as described in Figure 17.

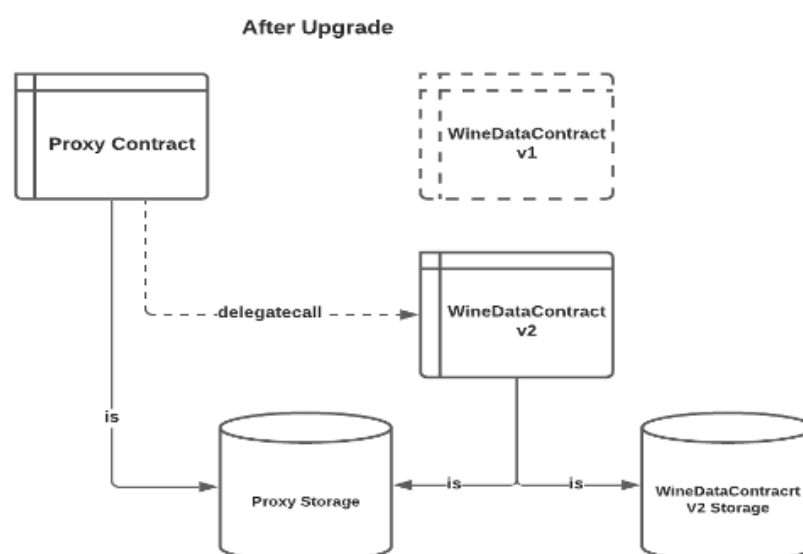


Figure 17. Proxy Pattern After Smart Contract Upgrade.

6.4. The Blockchain Interface

The blockchain service is built as an interface between the app-backend service and the blockchain network, with other service logic related to the key vault service and IPFS nodes embedded into it, when performing different system implementation operations. Given the functionalities of the open-source Ethereum interface libraries, such as *Ethers.js* or *Web3.js*, the blockchain service instance is able to invoke methods of the deployed smart contracts, and to interact with its local blockchain node with an instantiated object of “*ethers*” or “*web3*”, and the contract address specified via injecting environment variables when bringing up the service. Other environment variables represent information, such as the *address* of its IPFS local node, as well as both the *role_id* and *secret_id* of the key vault service for key management purposes.

The individual routers further link to their respective controllers. The controllers would utilize respective helper functions if some processes are repeated in the same controller functions, such as the process of obtaining a dedicated IPFS instance and an instance of key vault service. Some helper functions would also invoke the methods of the deployed smart contracts, as shown in Figure 18, demonstrating how the requests from an app-backend service could flow through the blockchain service, with related functions invoked at different levels.

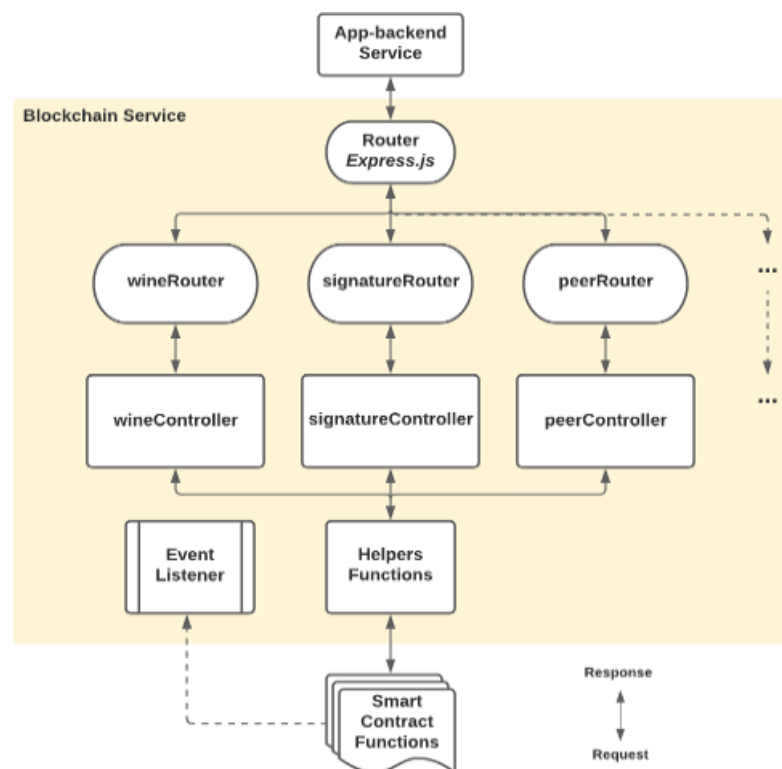


Figure 18. Functional Request Flow of Blockchain Service.

The developed smart contract source codes, such as those of *WineDataContract*, *PeerRegistryContract* and *Proxy*, after being compiled as JSON objects with the Truffle framework, are injected in the blockchain service for further object instantiations and operations.

6.5. The Decentralized Peer-to-Peer Data Storage with InterPlanetary File System (IPFS)

As it is too costly, and therefore not sensible, to store a full version of wine record on-chain, an alternative persistent data source is required, and the IPFS decentralized peer-to-peer data storage is an obvious choice for deployment in dNAS, under which an immutable and permanent content hash would be obtained from the IPFS network. The IPFS hash of a wine record subset would be stored on-chain under its respective wine

identifier, and could be referred to whenever there is a request for further operations, such as wine record validation across different blockchain service instances. From the security perspective, wine record subsets could not be altered or tampered with before being passed to the dNAS blockchain network and the IPFS network, given the fact that every content hash of a specific wine record subset obtained from the IPFS network will be changed if any of the contents of the wine record subset have been updated, which would further lead to an inconsistency with its counterpart stored on-chain when a wine record validation process is performed.

The IPFS functionalities in the blockchain service instance are enabled by “*js-ipfs*”. Before performing further IPFS operations, the IPFS node instance needs to be created in the blockchain service instance. During the wine record creation operation, in which changes are applied to the wine record subsets, the content hash is generated. The generated content hash have *multi-hash format* and *base-58 encoding*, under which the unique multi-hash allows the *exclusivity* of the generated content hash when the updated wine record subsets are added to the IPFS network. Once the IPFS node instance is generated, the wine record subsets are pushed to the IPFS network, with both “*path*” and “*content*” specified in the “*addDataIPFS*” function. The generated content hash of IPFS is then pushed and stored on-chain.

IPFS content hash is retrieved from the on-chain storage during the wine record validation operation, and the retrieved content hash is further sent to the IPFS node to obtain the represented wine record subset from the IPFS network. When looking up wine record subsets in the IPFS network, the network utilizes the *distributed hash table* (DHT), looking for respective IPFS nodes storing the specified wine record subset behind a content hash. The more times a wine record subset is retrieved from the IPFS network, the more copies of that record exist in the network, as the copy will reside in some IPFS nodes of the network, and IPFS nodes with the retrieved data will also act as hosts, accelerating the look-up process from the distributed hash table for other IPFS nodes looking up the same set of wine record subsets.

Object pinning of retrieved data could also be applied to ensure the data is always available on a local IPFS node, if the retrieved data needs to be available for further operations; however, as the retrieved wine record subset is only required for the specific wine record validation operation, the cache of the local node should be freed-up periodically if the data are retrieved with the normal “*cat*” method. IPFS nodes could also be added to or removed from a *private IPFS network*, like the one set used for dNAS, under which only IPFS nodes owned by any of consortium members and the consortium administrator are part of the private IPFS network according to the node management mechanism described in system implementation. Any message regarding the addition of data to, or retrieval of data from, the IPFS private network is shared *only* across the network. IPFS nodes on the private network could manage a list of IPFS nodes they have been communicating with, which are also owned by other consortium members of dNAS for *IPFS node management*.

6.6. The Key Management Module–Key Vault Service

It is important to know that blockchain transactions could only be sent from an account of a blockchain node, owned by any consortium member, with a valid digital signature produced by a private account key, such as the digital signature generated during the data-processing operation. Given how important the key management of Ethereum is to the security model of dNAS, it is vital to keep the private key of any account involved in dNAS safe, with a well-defined key management process and selection of key management modules. The key vault service deployed in dNAS could prevent any malicious player from compromising wine record components to negatively impact the data and processing integrity of the entire dNAS solution, though data stored on-chain is already hashed to obfuscate.

The key vault service of dNAS is developed basing on the open-source *Vault* by HashiCorp, which could further be remodelled with *AWS Secrets Manager* or *Akeyless Vault*

depending on the system requirements, so as to offer functionalities to manage and control access to the secrets of different Ethereum nodes and accounts, respectively, owned by any consortium member, and to manage secrets and contract addresses returned after every smart contract deployment process. Identities need to be verified before a specific blockchain service instance can access and interact with a key vault service instance of dNAS. There are two authentication methods for every distributed key vault service instance, used by any individual consortium member—*token-based* and *approle-based*—as defined in “*VaultAuthMethod*”.

The former is based on a unique access token generated by any vault service instance for a specific blockchain service, which is also owned by the consortium member. The access token generated is limited by the authentication leases, implying that reauthentication is needed after a given lease period to continue accessing the key vault service. The latter is about a set of policies and login constraints, such as the *RoleID* and *SecretID* generated, regarding the specific key vault service instance, which must be met to receive a token with those policies and access the key vault service. The distributed blockchain service instance could access the distributed key vault service instance by specifying the “*VaultAuthMethod*” with necessary metadata, such as the URL of a deployed vault service instance, or the API version or both the role identifier and secret identifier, depending on the authentication method specified, through environment variables, when bringing up the instance of distributed key vault service.

Once the key vault service is instantiated with a given blockchain service instance, both owned by a specific consortium member, the secrets of an Ethereum node and account owned by the same consortium member are then ready to be created and stored in the same key vault service instance with a given “*VAULT_PREFIX_KEY*” and *memberID*. According to the smart contract deployment process, the consortium administrator is the only role that is permitted to develop, deploy and upgrade deployed smart contracts, as depicted in the use-case analysis of dNAS. The contract address of both individual smart contracts and the corresponding proxy contract alongside the public address of the contract owner, which is the consortium administrator for this case, could use a similar design pattern to store the metadata as “*SCDeploymentSecret*” with the corresponding proxy address as a key to the secrets stored.

7. Overall Evaluation

dNAS demonstrated that any state transition on a wine record along the supply chain could only be validated and completed automatically if, (1) the involved registered nodes and its distributed nodes, such as blockchain nodes and the IPFS nodes, are cryptographically authenticated, (2) the related wine record and the transactions for specific process are cryptographically verified, and (3) the consensus of such state transition is reached with the related transactions packed in a mined block, by blockchain nodes owned by other consortium members. The immutable states of transactions in different blocks are also available on the network and can be accessed via different tools developed in dNAS, such as the blockchain explorer. The wine record verification and product provenance abilities of dNAS were confirmed to be strengthened and benefited with decentralization, under which no more single-point failure and control were found in the proposed dNAS. The resiliency and availability of the validated wine records, with the history of transitioned states, are also enhanced with the persistent volume on the states of blockchain and the related transactions, alongside the wine record data stored in dNAS.

Though dNAS was found to be less scalable than its centralized counterparts, as we expected, the advantages brought by decentralization have enabled different nodes along the supply chain of wine industry to work collaboratively with wine records flowing downstream, with state transitions validated automatically in a decentralized and immutable fashion. dNAS proved to function not only as a verification service, but also as a notary service, providing a proof-of-existence for wine records owned by different supply-chain participants. Further qualitative analysis on dNAS as a whole was also performed, with

limitations and risks identified to further improve different aspects, such as the system security, partial decentralization and scalability of dNAS.

In accordance with the research objective, it is confirmed that it is feasible to decentralize an anti-counterfeiting application of the supply chain industry to better improve the situation of counterfeiting attacks in the wine industry, and even the supply chain industry as a whole. With the flexibility of the architecture of decentralized system components, the decentralized solutions and operations proposed in dNAS are indeed industry-agnostic, and should be available for any industry with supply chain systems looking to combat product counterfeit.

8. Conclusions

With the current challenges in the anti-counterfeiting of the supply chain industry, the research was performed based on an implementation-driven research technique, with the main question and individual sub-questions listed in the research objectives and addressed in sections describing the system design, system implementation and the actual development steps detailed in this research. A proposed solution, namely the decentralized NFC-enabled anti-counterfeiting System (dNAS), is therefore developed as an autonomous and decentralized application for supply-chain anti-counterfeiting and traceability, especially for the wine industry, providing a way for registered nodes of the supply chain to verify the legitimacy of wine products, with cryptographically validated wine records. dNAS appears to strengthen the anti-counterfeiting and traceability capacities for the wine industry and the wider supply chain industry, while minimising the involvement of centralized parties, such as winemakers of the centralized NFC-enabled anti-counterfeiting system (NAS), which this research aimed to decentralize.

The literature analyses of this research cover current alternative resolutions to product anti-counterfeiting, and existing blockchain implementations applied in different industries. The idea of decentralized application was also explained and applied to the development of dNAS as this research's contribution to explaining why DApp could exhibit properties, such as its zero downtime, immutability and resistance to collusion. The proposed system model design and system architecture of dNAS were implemented basing on a set of system requirements, defined according to the findings of a security analysis performed against the legacy NAS. How dNAS is actually implemented is explained via system implementation procedures, categorized into two main phases, namely the *initialization phase*, including those operations related to consortium registry management and smart contract deployments, and the *data processing phase*, including the wine record creation and validation operations of dNAS. The technical details of system development, design patterns applied to different system components and the system execution of the dNAS working prototype, were also covered and elaborated.

The developed prototype of dNAS provided a clear demonstration of how a legacy supply-chain anti-counterfeiting and traceability system, for the wine industry in this case, could be decentralized and reengineered from its centralized architecture, which was once hosted and maintained by intermediaries. dNAS is built around the idea of enterprise consortium with registered nodes along the supply chain of the wine industry, balancing the degree of decentralization, scalability, security and the privacy of the proposed solution, with the concept of enterprise blockchain chosen as the preferred development model of the blockchain network, as well as the availability of shared smart contracts with the ability to be upgraded, with versions shared amongst the consortium members. The future directions of this research are also defined, including the evaluation of dNAS with system-testing procedures, such as automated-testing and performance-testing on different system components of dNAS, and the discussion the results of these system-testing activities and a qualitative system analysis of the prototype dNAS. More details on the limitations of, and future opportunities for, dNAS, including the development of a fully distributed model of dNAS, the introduction of possible security and privacy-preserving features, and the concept of product ownership, are discussed and explained in the separate research of [22].

Author Contributions: Conceptualization, N.C.K.Y.; methodology, N.C.K.Y.; software, N.C.K.Y.; validation, N.C.K.Y.; formal analysis, N.C.K.Y.; investigation, N.C.K.Y.; resources, N.C.K.Y.; data curation, N.C.K.Y.; writing—original draft preparation, N.C.K.Y.; writing—review and editing, N.C.K.Y.; visualization, N.C.K.Y.; project administration, N.C.K.Y. The author has read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data of analytical study published by Organization for Economic Cooperation and Development (OECD) and European Union Intellectual Property Office (EUIPO) in 2019, is available at: <http://www.oecd.org/corruption-integrity/reports/trade-in-counterfeit-and-pirated-goods-9789264252653-en.html> accessed on 20 March 2021.

Conflicts of Interest: The author declares no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

dNAS	The Decentralized NFC-Enabled Anti-Counterfeiting System
NAS	The NFC-Enabled Anti-Counterfeiting System
NFC	Near-Field Communication
RFID	Radio-frequency Identification
IoT	Internet-of-Things
IPFS	InterPlanetary File System
PoW	Proof-of-Work Consensus Algorithm
PoA	Proof-of-Authority
IBFT	Istanbul Byzantine Fault Tolerant
EVM	Ethereum Virtual Machine
CRUD	Create, Read, Update and Delete
Geth	Go-Ethereum Client
RPC	Remote procedure calls
IPC	Internal process calls
GHOST	Greedy Heaviest Observed Subtree
DAApps	Decentralized Applications
RLP	Recursive Length Prefix

Appendix A. Background

Appendix A.1. The Overview of the Legacy NAS (NFC-Enabled Anti-Counterfeiting System)

NAS with centralised data architectures which are predominantly based on the typical and familiar cloud-based client-server architecture style is demonstrated in Figure A1.

The whole NAS solution consists of FIVE main components, which are (1) the back-end system with the web-based database management user interface for wine data management, for which the management of data columns of specific wine products in their custody can be performed by winemakers, (2) an NFC-enabled mobile application, *ScanWINE*, for the tag-reading purpose of wine products at retailer points for wine consumers, or supply chain participants of the supply chain, before accepting a wine product, (3) another NFC-enabled mobile application, *TagWINE*, performing tag-writing for wine products at the wine-bottling stage, (4) the NFC tags packaged on the bottleneck for those purposes and actions, and (5) any NFC-enabled smartphones or tablets of supply chain participants and wine consumers along the supply chain.

There are FOUR major categories of data that are processed along the supply chain with NAS, namely, (1) nodal transaction history data, (2) supply chain data, (3) wine pedigree data, which is processed with its dedicated controllers based on their predefined schema and data models, and (4) unsuccessful validation data, returned from any unsuccessful validation at the stage of accepting wine products. The wine products are processed

and handled by different nodes along the supply chain, with the data updated by scanning the NFC tags of the wine product using the tag-reading *ScanWINE* and the wine record updated according to the database. These categories of data are updated along the supply chain until the point of purchase, where wine consumers use the tag-reading *ScanWINE* to scan the NFC tags and retrieve, for example, as wine pedigree data and transaction data for real-time validation to determine if the wine product is counterfeit or not.

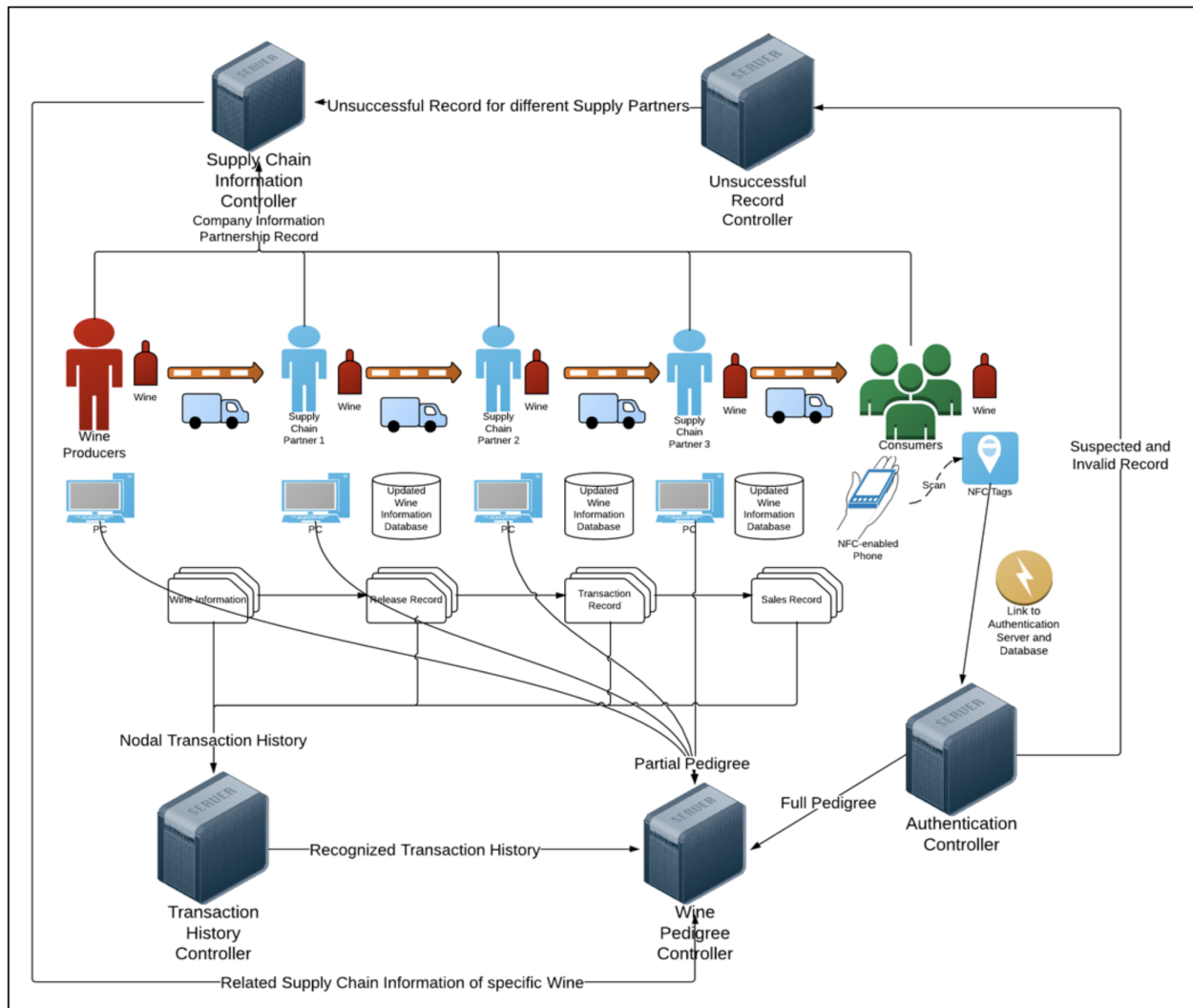


Figure A1. The System Architecture of Legacy NAS (Source: Neo Y.).

A unique identifier is assigned to each wine product and written into the NFC tag. Tag-attached wine products are then shipped from winemakers to different nodes along the supply chain. During the transportation process of the wine products along the supply chain, each involved node could interact with the NFC tags and add the aforementioned four categories of data into the NFC tags. In this way, the next node can check whether or not the wine products have already passed through the legitimate supply chain. If any inconsistency is found at any node, such wine products may be considered counterfeits and should be returned to the winemakers. However, once the wine product reached the post-purchase stage and circulated in any customer-to-customer market, its authenticity is no longer guaranteed, as anyone who has an NFC reader could interfere and clone the tags' data. Therefore, it is important to develop anti-counterfeiting and traceability systems that could work even when the data of the tag are cloned in the post-purchase supply chain,

with the security attacks detecting and preventing any potential adversary state transition from taking place.

Appendix B. System Model Definition

Appendix B.1. Use Case Analysis Diagram of dNAS

The Use Case Analysis Diagram of dNAS is depicted in Figure A2.

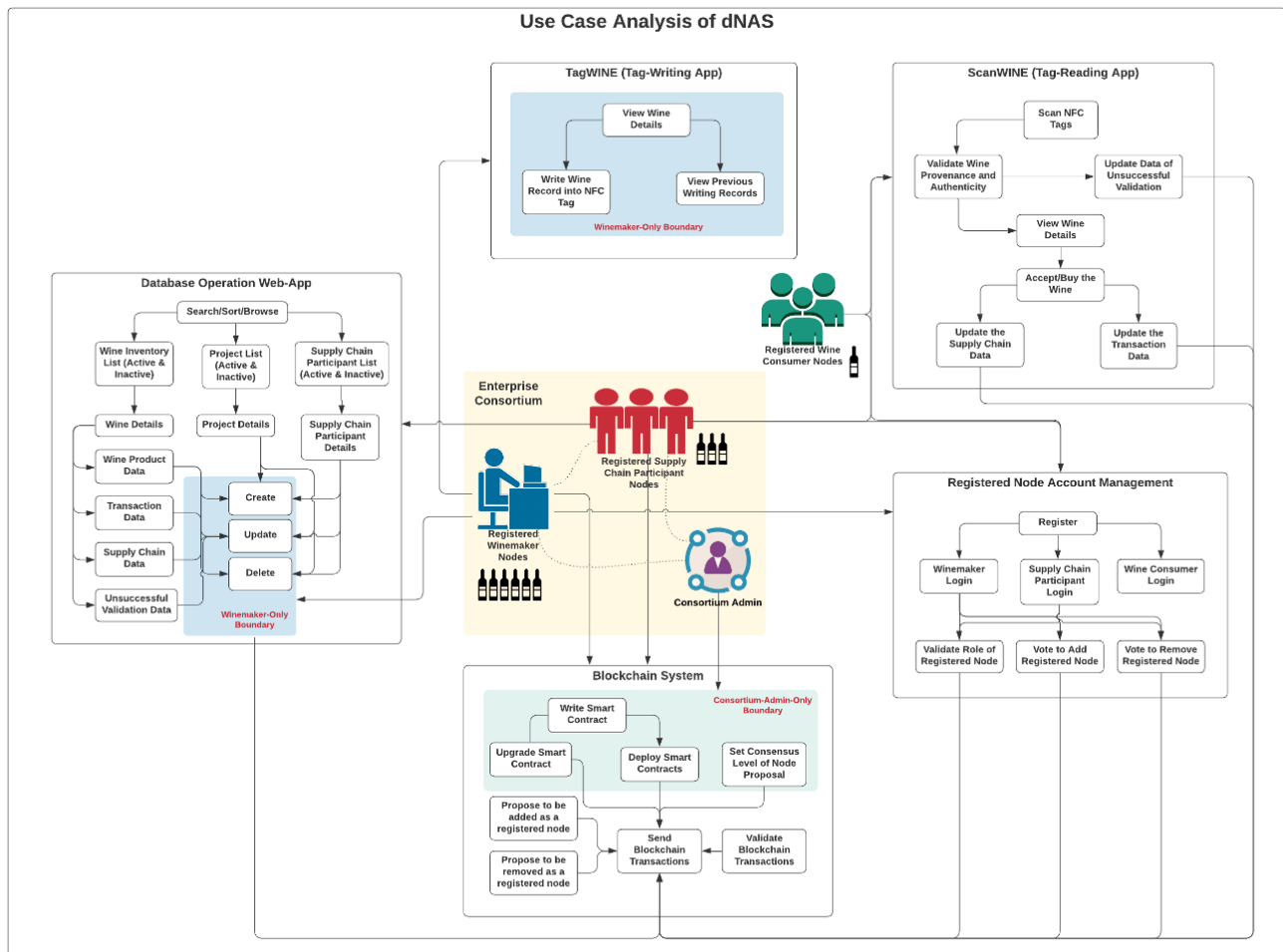


Figure A2. Use Case Analysis Diagram of dNAS.

Appendix B.2. System Architecture Diagram of dNAS

The Full System Architecture Diagram of dNAS is depicted in Figure A3.

19. Kim, H.M.; Laskowski, M. Toward an ontology-driven blockchain design for supply-chain provenance. *Intell. Syst. Account. Financ. Manag.* **2018**, *25*, 18–27. [[CrossRef](#)]
20. Benet, J. Ipfs-content addressed, versioned, p2p file system. *arXiv* **2014**, arXiv:1407.3561.
21. Ozyilmaz, K.R.; Yurdakul, A. Designing a Blockchain-based IoT with Ethereum, swarm, and LoRa: The software solution to create high availability with minimal security risks. *IEEE* **2019**, *8*, 28–34. [[CrossRef](#)]
22. Yiu, N.C.K. An Empirical Analysis of Implementing Enterprise Blockchain Protocols in Supply Chain Anti-Counterfeiting and Traceability. *arXiv* **2021**, arXiv:2102.02601.

Short Biography of Author



Neo C. K. Yiu is a computer scientist and software architect specialized in developing decentralized and distributed software solutions for industries. Neo is currently the Lead Software Architect of Blockchain and Cryptography Development at De Beers Group on their end-to-end traceability projects across different value chains with the Tracr™ initiative. Formerly acting as the Director of Technology Development at Oxford Blockchain Society, Neo is currently a board member of the global blockchain advisory board at EC-Council. Neo received his MSc in Computer Science from University of Oxford and BEng in Logistics Engineering and Global Supply Chain Management from The University of Hong Kong.