2020 IEEE International Conference on Software Architecture Companion (ICSA-C)

Building a prototype based on Microservices and Blockchain technologies for notary's office: An academic experience report

Pamella Soares de Sousa, Nataniel Parente Nogueira, Rayane Celestino dos Santos,
Paulo Henrique M. Maia, Jerffeson Teixeira de Souza
State University of Ceará
Fortaleza, Ceará
{pamella.soares, nataniel.parente, rayane.santos}@aluno.uece.br, {pauloh.maia, jerffeson.souza}@uece.br

***Abstract*—The problem of lack of trust in data sharing between different parties can bureaucratize processes carried out by entities such as notary's office. Blockchain technology can circumvent this problem by providing a distributed and unchanging base. By merging this technology with the microservice architecture, secure and robust systems can be created due to the independent deployments and development of microservices, thus easing the application maintenance and evolution. Considering the advantages in the combination of the mentioned technologies, the present work aims to propose an approach that allows the integration between notary's offices and other institutions, ensuring security and celerity in the exchange of information between the parties. In this paper we report on our academic experience in the creation of the proposed approach, a pilot prototype, the development process and the tools used in the implementation, and the lessons learned.**

***Index Terms*—Microservices, Blockchain, Notary's Office, Smart Contracts, DevOps**

I. Introduction

Despite the technological immersion we are witnessing, with increasingly powerful smartphones and more robust, reliable and distributed systems, we still face systems and services from public and private agencies which are lacking in the quality of service delivery to the population. The way these types of services are offered usually generate long queues and discomfort in those establishments [1].

Birth registration of a child in Brazil, for example, is usually done in a civil registry notary's office, in person. As a rule, the child's father, preferably, goes to the notary's office holding an identity card and statement of live birth issued by the hospital[1]. Brazil recorded an average of 7945 live births registered per day in the year 2018[2]. This statistic gets more complex when we include the other services provided by the notary's office.

Nowadays the technology is an ally for institutions that want to improve their processes and bring convenience to their customers. One of the possible solutions to the previously mentioned problem is the creation of applications that offer trust between the involved parties, such as notary's offices and hospitals. The blockchain technology arises as a potential solution to this issue. [2].

A blockchain is essentially a distributed database of records that have been executed and shared among participating parties [3]. The five basic principles of blockchain are: distributed database, peer-to-peer transmission, transparency with pseudonymity, irreversibility of records and computational logic [4]. Therefore, all parties have access to include a new information on the network, but never to erase or to update data.

Since different systems are integrated in a distributed manner, it is necessary to develop complex, robust and resilient applications in order to be able to deal with scalability, decentralized operations, intercommunicating services, availability, etc. In this realm, the microservice-based architecture becomes a prominent solution due to its way of implementing applications as a collection of small and independent services that can communicate to each other through well-defined interfaces using lightweight protocols [5], [6]. For this reason, the microservices architecture is maturing as an architectural style for developing distributed software systems with high requirements for scalability and adaptability in companies such as Amazon, Netflix, and LinkedIn [7].

Microservices make the integration of new entities into the system easier and faster, since modularization and scalability are some their key features [8]. Therefore, by combining blockchain technology with the microservice architecture, it is possible to encapsulate contract functions in a microservice [9] and create a robust and secure system, as this architecture can optimize the scalability and deployment of an application.

In this context, our goal is to propose an approach, based on microservices and blockchain, that allows the integration between notary's offices and other institutions, ensuring security and speed in the exchange of information between the parties. Such approach makes it possible to encapsulate smart contracts in specific microservices depending on the functionalities determined for the service. Therefore, the main contribution of this paper is threefold: (i) an approach for the proposed business model integrating blockchain and microservices; (ii) a prototype implementation that can generate a birth certificate

[1]https://guiadocumentos.com.br/certidao-de-nascimento/
[2]https://sidra.ibge.gov.br/tabela/2679

978-1-7281-4659-1/20/$31.00 ©2020 IEEE
DOI 10.1109/ICSA-C50368.2020.00031

and register it on a blockchain; and (iii) an experience report on the design decisions and lessons learned that can direct new students in the field.

This paper is organized into nine sections as follows: Section 2 explains the blockchain and microservice architecture style, while Section 3 discusses the main related work. Section 4 describes the functional and quality requirements for the prototype application, as well as the chosen microservices. Section 5 details the main flow of the application, while Section 6 reports the development process, including the organization of sprints, the DevOps tools used, and implementation pipeline. Section 7 shows the design decisions and lessons learned in the project development. Section 8 exposes the threats to validity of the work. Finally, section 9 draws the conclusions and future work.

II. Background

A. Blockchain

Currently, most of the digital economies rely mainly on third parties to validate financial or operational transactions, such as the services offered by the notary's offices in Brazil, the use case addressed in this paper. However, this type of system is vulnerable to human failure, intrusion, or can be maliciously managed. The blockchain technology is an alternative to circumvent the aforementioned problems. By using of cryptography techniques, it offers greater confidence and facility in transactions, thus creating a decentralized system without the need for third parties [10].

According to Cyran [11], a blockchain can be characterized as a distributed data structure, a public ledger, with all transactions executed in the system, in which each transaction in the public ledger is unchanged and verified by consensus of most participants in the system, avoiding failures and ensuring data reliability. Such agreement is achieved through consensus mechanisms which are sets of steps taken by all or most of the nodes to agree on a state or value [12]. This mechanism is what helps the blockchain system to be secure, preventing its users from sending wrong or fraudulent information.

Blockchain, introduced by Satoshi Nakamoto through the Bitcoin, was initially proposed to compose an electronic payments system, a peer-to-peer online communication protocol that facilitates the use of cryptocurrencies [13]. There are now new business models proposed through a "programmable blockchain". This is possible due to the use of smart contracts that have demonstrated their effective use when integrated with the blockchain technology.

In short, a smart contract can be understood as an executable script stored on the blockchain that is capable of automatically executing the terms of an agreement in a transaction [14]. After the contract is deployed to the blockchain, users can execute them by sending transactions to the contract address. Then, transactions will be executed on all consensus nodes. The contract can, according to the transaction, read, write on their private storage and even create new contracts.

The blockchain has important properties, such as the immutability of data when recorded in the public ledger. Data update is only possible through a new transaction and a new consensus. In addition, data integrity is ensured by replicating data and transactions across different nodes, keeping the system available and secure. Transactions in the public ledger are passive of verification and auditability. Furthermore, the implementation technology is often open and verifiable [15].

B. Microservice-based Architecture

Microservices are an architectural style to develop a single application as a collection of independent, well-defined, and intercommunicating services, each running in its own process and communicating through lightweight mechanisms [5]. Microservices are built around business capabilities using a concept from the DDD (Domain Driven Design) [16], named bounded context, in order to delimit their business functionalities and associated data. In particular, a microservice can be understood as a single responsibility application that can be independently deployed, scaled, and tested [7]. By adopting the microservices architecture, developers can engineer applications that are composed of multiple, self-contained, and portable components deployed across numerous distributed servers [17].

The microservice architecture provides benefits such as independent deployments and development, small and focused teams, fault isolation, decentralized governance, and decentralized data management [6]. In particular, it proposes a solution for efficiently scaling computational resources. Since microservices can be individually scaled, they provide an efficient manner to allocate computational resources, enabling flexible horizontal scaling in cloud environments.

III. Related Work

Despite that the literature involving blockchain and microservices is still in its beginning stage, there are some approaches that combine the benefits of both technologies to create new systems and, in some cases, making analogies between smart contracts and microservices. Some of these papers are summarized below.

Da Silva *et al.* [18] implemented a Proof of Concept (PoC) based on emerging technologies such as blockchain Hyperledger, microservices, and big data. Such system aims to provide adequate assistance to patients by ensuring an appropriate emergency care. Consequently, it is possible to reduce the waiting queue at hospitals and to make a better use of resources. In that work, while blockchain was used to create two networks (one for Patient and other for Attendance), microservices formed the underlying architecture of the system. In addition, Scrum was used to ensure that the system specifications were achieved in a period of 17 academic weeks.

Dai *et al.* [19] propose TrialChain, a platform that integrates private and public blockchain, a web system and a data science platform from the National Center for Cardiovascular Diseases (CNDC) from China. The proposal aims to increase the integrity and forms of data validation in laboratory information and clinical trial management to reduce the risk of data manipulation and to increase confidence in the results.

Microservices were used to create a multi-host architecture that runs independently and replicates blockchain data to each host to protect the system against data loss in the event of node failure. A web interface was created to allow query and data validation in the public blockchain.

Cyran [11] presents a solution that aims to protect sensitive health data in hospital environments. The system architecture consists of the use of container Docker that enhances system deployment. The container contains the web application layer, the key store service, the cache service, and the blockchain service, all implemented using the microservices architecture. Protection of data is achieved by using cryptographic layers and public and private keys to be encrypted before being inserted into the blockchain and decrypted when sharing data. The solution also uses smart contracts, which assist in data discovery, retrieval, and decryption.

Nagothu *et al.* [20] suggest the development of a microservice surveillance monitoring system. To overcome architectural security vulnerabilities due to the use of distributed data, it is proposed the use blockchain in the application. The authors implemented the system with *Facial Recognition*, *Audio Analysis, License Plate Recognition*, and *Behavior Analysis* microservices, each of them with a dedicated database whose information is synchronized and stored in the blockchain. In addition, smart contracts record and sign certain surveillance data according to what is bound to nodes.

Tonelli *et al.* [15] discuss that microservices and smart contracts share many similarities, such as decentralization. The main point in common is that smart contracts can communicate with each other, just like traditional microservices. Furthermore, each smart contract provides its service, such as changes, updates, login, and general transaction types. The authors suggest a use case whose the primary goal is to enable doctors to keep track of their patients' disease diagnoses. Three smart contracts were implemented: *DoctorPseudoRest*, *PatientPseudoRest*, *DiagnosisPseudoRest*.

IV. Requirements and Microservices

This section presents the functional and quality requirements of the proposed system that were identified based on the main services provided by a notary's office. From this analysis, the microservices were delimited according to the business rules and data modeling of the identified requirements.

A. Functional Requirements

As mentioned, the requirements were raised from the main flow of a notary's employee performing his/her service, ranging from logging on the system to realizing a customer-requested service. To elicitate the requirements, we used local observation, brainstorming and scenarios. Those requirements were represented in the form of short, abstract and high-level descriptions through user stories [21]. Initially, only the notary's employee will interact with the system, as presented in Table I.

TABLE I
User Stories

ID	User Story - Notary Employee
R1	I am able to log in to the notary's system.
R2	I can list the services provided by the notary.
R3	I can list the customers registered in the notary.
R4	I can select the customer, its requested service, and start an order.
R5	I am able to register a customer with the system.
R6	I can register a certificate record in the system.
R7	I can perform a digital signature on behalf of the notary.
R8	I can generate a payment slip for the customer.

B. Quality Requirements

We specify quality requirements considering the demands of the proposed approach and how microservices and blockchain can contribute, both together and individually, to achieve the quality attributed, as shown in Figure 1.

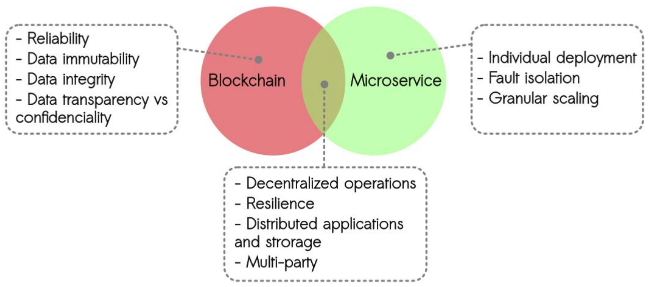

Fig. 1. Advantages of microservice and blockchain technologies

In general, the use of blockchain technology, along with the microservice architecture, can generate benefits on scalability, security, and information sharing, among others. The proposed approach uses the terminology "distributed" in two perspectives. On the application side, a microservice architecture is comprised of several independent and distributed sub-applications that can provide several benefits as detailed in Section II. Regarding the blockchain, the fact that information is stored in a distributed manner brings other advantages such as resilience, decentralized and multi-party operations, for example.

Considering resilience of the application side, a problem in a specific service would only affect it. Other services would continue to handle requests typically. In a monolithic system, the whole system would be damaged if one component misbehaves. The resilience of blockchain technology is due to the replication of the information on each node of the network, allowing them to be quickly recovered if any node loses it. In both situations, the problem of a single point of failure can be mitigated.

One of the key features of blockchain is enabling a shared infrastructure in which there is no control by the participating organizations. This technology is suitable for multi-part scenarios in which intermediaries are acting within the current systems. Since there are multiple parties, the same microservice may be requested by different entities, depending on what the microservice is designed to do. Besides, decentralized operations are possible since no party controls the system, but each user can manage his/her own data and assets [22].

C. Microservices

From the user stories mentioned in Table I, we delimited the microservices of the system. Hence, the approach back-end is being formed by six microservices:

- **Login microservice:** allows notary's employees to log in to the system so that they can perform the services requested by the client.
- **Client microservice:** maintains the registry of the notary's clients. It is possible to register information such as name, ID, and date of birth, among others. The microservice also allows modification and removal of this information.
- **Service microservice:** maintains the registry of the notary's services. A notary's office performs civil and other services, and each service has a description and its price.
- **Order microservice:** maintains all customer orders and their services requested on a given date.
- **Certificate registration microservice:** records all certificates that have been requested by notary's customers (e.g., birth, marriage, death).
- **Signature validation microservice:** the registry must validate and record the veracity of the certificate before it can be included in any database. Thus, we designed a microservice that allows the notary's digital signature so that the document is registered only after this procedure.

V. The Proposed Approach

Figure 2 represents a general flow of activities that should be followed to allow the development of applications that use our proposed approach. We consider two types of participating entities that will be involved in the approach:

- **Notary's office:** company or institution, public or private, in which the issuance, analysis, authentication, registration, and filling of notes and documents takes place, giving public faith to the presented documents .
- **External institution (EI):** organization that the notary needs to consult to continue the activities related to the respective service. Each service provided by the notary may depend on authenticated and validated information from a specific EI. In addition, those entities may also consult and record information on the blockchain in accordance with their respective services.

In the current approach, the flow described in detail will be related to notary services, since this paper intends to present the development of the microservice architecture for this domain. We illustrate the proposed approach with a scenario based on the real process of issuing a birth certificate in Brazil. **Scenario:** A child was born in a certain hospital that recorded his/her Statement of Live Birth ("*Declaração de Nascido Vivo*" in portuguese - DNV) on the blockchain. The parents went to the notary's office to apply for certificate registration, and the notary's office consulted whether the child's DNV was already contained in the shared blockchain records. After verifying that the declaration was valid, the notary's office begin the procedures for registration and issuance of the child's birth certificate. After the notary's office completed the process of the requested service, the child's information and certificate are recorded on the blockchain. Since it is necessary that the newborn child already has its Individual Taxpayer Registration ("*Cadastro de Pessoa Física*" in portuguese - CPF), the parents requested the CPF to the Post Office, the place of issue of such document in Brazil. In order to carry out the necessary checks, the Post Office's employee verified on the blockchain whether the child was already registered in a notary's office. After that verification was positive, the CPF request carries on.

Given this real scenario, the steps presented in Figure 2 will be described as follows, which will be detailed to present the microservices involved:

1) **Blockchain registration:** the EI referring to the hospital where the child was born records her DVN on the blockchain so that it is available to the notary's office network that has access to the blockchain.
2) **Initial procedures:** the client, father or mother of the child, arrives at a notary's office and requests a service. In this case, the present paper assumes that the registry is of the type "Civil notary's office", which can perform registration and issuance of documents such as birth, marriage and death certificates. For example, a notary employee serves the client and logs in to the system to start the requested service. After logged in, the employee checks whether the client is already registered in the notary's office. If so, the client and his/her requested service are selected and included in an order. Note that the Login, Client, Order, and Service microservices were requested.
3) **Blockchain query:** at this time, the clerk will make a request to authenticate information necessary to continue the requested service. This step will be performed if the service needs data shared by an EI. At this time, the employee can check the newborn's DNV to validate it.
4) **Notary service procedures:** the employee will request all necessary data and documents to fill in some forms. Since the client chose the birth certificate registration and issuance service, for example, then the information to include will be: ID and CPF, as well as a birth certificate or parental certificate. DNV would not be required as it can be accessed through the blockchain. Having the necessary information, the employee will fill in a form and attach a copy of the documents. He/she finishes the service by generating the certificate and digitally signing it. For this step, to sign the document, the Signature Validation microservice is required. Before the blockchain registration is effective, an invoice for payment is generated, delivered to the client, who in turn must make the payment in some notary's cashier department. Note that the Order microservice will also be solicited.
5) **Blockchain registration:** after completion of the pro-

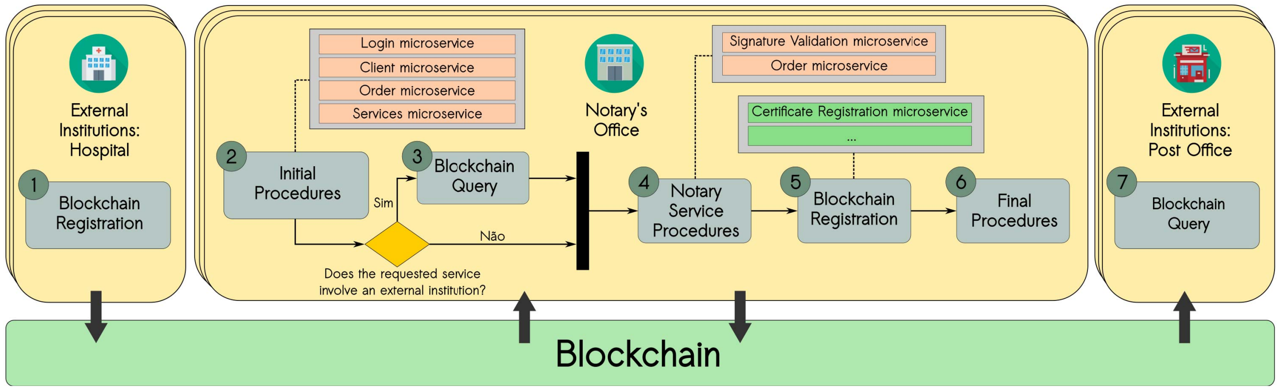

Fig. 2. Execution flow of a notary service in our proposed approach.

cedure and having proof of payment for the service, all information collected, along with the notary's digital signature, will be recorded on the blockchain. Here, the microservice involving the use of the blockchain is requested (Certificate Registration microservice).

6) **Final procedures:** as each client may have several orders, the current order will be finalized. Here again the Order microservice will be requested for its completion.
7) **Blockchain query:** this step occurs in a process similar to step 3. However, the the Post Office's employee is the one who consults the registered certificate at the notary's office.

VI. System Implementation

In this section, we present the development tools and processes used, as well as the description of the smart contract functions related to the Certificate Registration microservice. As discussed, we implemented a pilot of the proposed approach, i.e, a simplified version of the flow shown in Figure 2 considering a notary system. The complete prototype is available on GitHub[3].

A. Tools

The microservices from Section IV-C were listed as tasks in the online tool "Trello". This allowed developers to have a better understanding of the development process evolution. The development environment consisted of the components and tools listed in Table II.

B. The DevOps Development

According to Dyck et al. [23], DevOps is an organizational approach that emphasizes empathy and collaboration between IT development and operations teams in systems development organizations to produce resilient systems and promoting continuous delivery of changes. We adopted this approach to improve the quality of development taking into account team communication, process optimization, speed of production, continuous delivery, proper use of tools, and the other benefits

[3]https://github.com/Notary-BlockMS

TABLE II
Implementation Tools

Software	Version	Purpose
NodeJs	10.16.3	Provides a development framework that allows the use of NodeJs package manager NPM to install JavaScript dependencies.
Java	13.0.1	Programming language and object-oriented computing plataform used to develop Service Discovery in this project.
Solidity	5.6.0	It is an object-oriented programming language used in this paper to write smart contracts on a blockchain platform, Ethereum.
Etherjs	4.0.0	A compact and complete JavaScript library that allows interaction with the Ethereum network.
Web3	1.2.4	A library that allows a JavaScript application to interact with an Ethereum network node using an HTTP or IPC connection.
Mongodb	4.2.0	NoSQL database used at this work to store system-generated data.
React	16.12.0	Declarative, efficient and flexible JavaScript library for creating user interfaces (UI).

that this culture provides to the team. The DevOps techniques used in this work are listed below by categories:

- **Agile methodology:** the Scrum methodology was widely used to perform software project management and planning. The sprints were well defined and a Sprint Review Meeting was held at the end of each sprint.
- **Collaboration:** as mentioned previously, Trello was used by the team to manage the project. In addition, we adopted Slack as a collaborative channel for communication between team members and integration with other tools.
- **Continuous Integration (CI):** during the project development, we used the "Travis CI" tool to automate the testing and building of the source code of each commit.
- **Testing:** unit tests were used in the system functions to ensure that the system is working as specified. We used the JS Tape as testing tool.
- **Deployment and Cloud**: Docker was used to create a microservice image and to deploy the container on the cloud. The microservices were deployed on cloud platforms after being containerized: the Google Cloud

hosted the Service Discovery application, while Heroku hosted all other microservices.

C. Development Pipeline

We used agile methodology to organize project execution due to team characteristics and the need for constant delivery. Our team consisted of three first-year students of a master's degree in computer science and each delivery was evaluated by a professor who represented the client of the application. We followed the sprint concepts of the Scrum method, which were executed as follows:

- *Sprint Zero*: All developers had their first contact with microservices development and created a mock microservice to be consumed in the cloud.
- *Sprint 1*: Firstly, the Login microservice was developed. The user was included by the team using a collaboration platform for API development. Then, it was possible log in the web application typing the registered email and password. The CRUD of the Client microservice was implemented. Thus, the functions of listing all clients registered in the notary's office, searching them through an id, including them in the database, updating the data, and the possibility of removing it from the client list by searching their respective id were implemented. The CRUD of the Service microservice was also implemented.
- *Sprint 2*: At this moment, we focused on the development of the front-end. The Login, Home and Register Client pages were developed in the web application.
- *Sprint 3*: The Register Certificate and View Document pages were developed and the API Gateway was build.
- *Sprint 4*: Finally, Eureka and Zuul were used to implement service discovery. Both services were deployed to the Google cloud platform. The Digital Signature pages were developed in the web application. A CRUD of the Certificate Registration microservice was developed. We implemented a smart contract that allows a certificate hash to be entered and viewed when a user searches for it using its ID, as this microservice communicates with the Ethereum blockchain. We used a storage technique when adding a conventional database so that the document is stored off-chain (see Section VII-A).

At the end of each sprint, functional microservices were deployed at the Heroku platform after local tests and executions. A mock service was also developed to simulate the operation of the signature validation microservice. However, in future work, it is necessary that the signature validation microservice is integrated with a Certifying Authority (CA).

D. Smart Contract Implementation

As already explained, the system was implemented by using Ethereum and its dependencies. In particular, this section details the smart contract used by the Certificate Registration microservice, giving an overview of its functions, which is presented by its pseudo-code in Algorithm 1.

Algorithm 1: *RegisterCertificate* smart contract

```
function includeCertificateBlockchain(id, certificate)
    if msg.sender != owner || DocumentsList[id].used then
        Abort;
    insert Document in DocumentsList;
    return sucess;
end_function
function getAllCertificatesBlockchain()
    return DocumentsList;
end_function
function getCertificateBlockchain(id)
    return Document;
end_function
function changeOwner(newOwnerAdress)
    if msg.sender != owner then
        Abort;
    update current owner with new owner adress;
    return sucess;
end_function
```

The Certificate Registration microservice can perform the registration of the certificate requested by the client by communicating with the *RegisterCertificate* smart contract, which consists of four functions that (i) add a certificate, (ii) return all blockchain certificates, (iii) return a registered certificate by its ID and (iv) change contract owner.

The first function of the Algorithm 1 *includeCertificateBlockchain* registers the certificate on the blockchain when given two parameters: id and certificate. Only the contract owner can execute this transaction, so a verification is performed. After the necessary checks have been carried out, the *timestamp* and *used* (boolean variable), along with the input parameters, are stored in a struct that is stored in a certificate list. The *used* variable is adopted to prevent a certificate with the same id from being inserted into the blockchain. The second *getAllCertificateBlockchain* function returns the list of all inserted document IDs. Thus, since blockchain data is append only, one can get the history of the certificates.

The third function of the smart contract is *getCertificateBlockchain*, which returns a Certificate structure with the specific ID informed as parameter. Finally, the *changeOwner* function allows the contract owner's address to be changed, but only the current owner is allowed to modify it. This function is necessary to enable the change of the user who will perform certain smart contract operations. In this paper, we implemented only the *RegisterCertificate* contract. However, as new microservices are added to meet new system functionalities, other specific contracts can be planned and encapsulated according to the new microservices funcionalities.

VII. Discussion

A. Design Decisions

Requirement specification. Initially, we conducted a survey of the mainstream requirements to be implemented. Activity, sequence, and use case diagrams have been created for better understanding and specification of the system. We used abstract descriptions of user stories using non-technical language to define required system behavior. Table 1 contains the list of user stories from the perspective of the notary's employee.

Microservices decomposition. We delimited microservices from the business rules and data model to which each raised requirement was related. For this work, we identified the need for six microservices to compose the proposed approach, as detailed in Section IV-C.

Data management. Regarding data storage, each microservice has its own database. This decision allows each service to manage its own database, either through different instances using the same database technology, or even using different database systems, since the idea of microservices is that they are independent and uncoupled.

Use of the blockchain. The goal of the approach is to provide a safer and more agile system for the Brazilian notary's offices. There is an excess of bureaucracy in those places and the way processes are executed is still quite rustic, causing a waste of time, favouring the chances of fraud and always needing an intermediary in the services provided. To solve these problems, it was necessary a way to ensure the immutability of the information and, at the same time, bringing agility to the services provided by the Brazilian notary's offices. Therefore, we used blockchain as a data storage component for the microservices related to the notary's services. In order to mitigate the storage scalability problem due to the large data volume, we used an off-chain technique in which only the document hash issued by the notary's office is stored in the blockchain, while the document itself is stored in a traditional database.

Communication style. The communication between the microservices was accomplished through the HTTP protocol methods (GET, PUT, POST, and DELETE), using the JSON format to data exchange. Thus, the communication followed a RESTful style.

Service discovery and API gateway. To mitigate the challenge of working with URLs from different microservices, we used a service discovery (Eureka) and dynamic routing (Zuul) solution in the approach. Thus, the requests were centralized in one place and a proxy forwarded them to the correct microservice.

B. Lessons Learned

Creating an application upon a microservice architecture using blockchain is not such a simple task, specially for inexperienced groups. During the development, we found several challenges, most of which were overcome. We describe some lessons learned in the development process that may be useful to others with little experience in the area.

Inexperience in microservices can be overcome. At first, the implementation of microservices can be a little confusing to be understanding. Our team had no practice in developing microservices. To begin the development, some books and tutorials taken from the Internet have been selected to aid the comprehension of the subject. Before implementing the main application, each team member created a simple example microservice to practice the studied concepts. With a little practice and theoretical background, our team achieved good results.

The choice of microservices. The process of choosing which microservices would constitute the application of this paper caused discussions among the members. To separate the microservices, we based our decision on some previous work that carried out the decomposition of monolithic systems in a microservice-based architecture. We also used an approach of separating microservices according to their requirements, taking care that the failure of one microservice does not affect the functionality of another.

Choice of microservices that would use blockchain. The choice of which microservices would use a blockchain technology also raised many questions in the team. In some cases, the technology is only optional, but in others it is essential. To guide our team in this choice, we used similar academic studies and a flowchart [22] with questions that indicated in the use or not of the blockchain technology. We applied this flowchart to each microservice rather than to the general use case, as done [22].

DevOps improves production. In the beginning the team faced resistance to perform continuous delivery. Using DevOps tools has improved implementation progress and feedback documentation. We could better measure the activities we would deliver on each sprint and upcoming deliveries.

VIII. Threats to Validity

This paper has threats to external, construction, completion, and internal validity [24], as follows:

- *Restrictive*: this paper mainly introduces an approach for notary's offices, without expecting to extend this proposal to solve problems of hospitals, police stations or other similar entities. This constitutes a threat to the external validity of the paper. However, the approach has the potential to be generalized to other areas of knowledge.
- *Limited knowledge of notary business rules*: despite dedicating this solution to notary's offices, we do not try to be exhaustively strict about the business definitions already delimited by notary's offices, thus allowing a threat to the construction validity and likely changes to execute this solution in a real environment (production version). However, we conducted informal interviews with legal specialists and conducted a preliminary survey of the registers related to the notary's office in Brazil.
- *Limited knowledge of technologies*: we consider the threat of completion validity to be limited experience in blockchain technology, microservices architecture, and the merging of the two. Despite the short deadline for the project execution, we strove to assimilate as much content as necessary for the system to perform well.
- *Decisions made during the development*: due to the lack of knowledge of the notary's technologies and business rules, a threat to internal validity should be considered, as the choices we made may have influenced the state of the application we present. To alleviate this problem we had constant feedback with the course's supervisor.

IX. Final Considerations

This paper presented an academic experience based on microservice and blockchain technologies for notary's offices. We detailed a an approach for the proposed business model and then showed a prototype implementation for issuing a birth certificate and registering that certificate on a blockchain using our approach. Finally, we reported on and discussed a set of design decisions and lessons learned that can help other students and beginners in the technologied addressed in this paper. We developed six microservices, one of them capable to connect to Ethereum, a public Blockchain, to record information from a certificate. We also developed a web application as a visual layer to manage those microservices. We used the following DevOps techniques to build the application: Scrum, CI with Travis, unit tests, Docker, Google Cloud, Heroku, and Trello. Our development pipeline was divided into five sprints of twenty days each.

As future work, we plan improvements in both blockchain technology and microservices. We intend to improve the example application to receive other external entities, as well as blockchain technology related enhancements. Regarding microservices, we can extend the application to allow customers to interact with the system themselves, which will demand the implementation of new services. Those enhancements are capable of providing secure communication between entities that currently have few or no communication and greater celerity and convenience to the end client of the application. Furthermore, we intend to carry out an analysis of performance metrics of the developed application. At last, we plan to generalize the proposed approach to formalize a reference architecture to the secure and scalable integration of services provided by different entities.

Acknowledgments

This work is partially supported by the National Council for Scientific and Technological Development – CNPq/Brazil under grant Universal 438783/2018-2 and by the Coordination for the Improvement of Higher Education Personnel - CAPES/Brazil, under grant 88882.447675/2019-01. In addition, the authors are grateful for the assistance given by FFIT (Innovation and Technology Services).

References

[1] D. Chuen and R. Deng, *Handbook of Blockchain, Digital Finance, and Inclusion, Volume 1: Cryptocurrency, FinTech, InsurTech, and Regulation*. Elsevier Science, 2017.

[2] V. Gatteschi, F. Lamberti, C. Demartini, C. Pranteda, and V. Santamaria, "To blockchain or not to blockchain: That is the question," *IT Professional*, vol. 20, no. 2, pp. 62–74, 2018.

[3] M. Crosby, P. Pattanayak, S. Verma, V. Kalyanaraman *et al.*, "Blockchain technology: Beyond bitcoin," *Applied Innovation*, vol. 2, no. 6-10, p. 71, 2016.

[4] M. Iansiti and K. R. Lakhani, "The truth about blockchain," *Harvard Business Review*, vol. 95, no. 1, pp. 118–127, 2017.

[5] M. Fowler, "Microservices," 2014.

[6] P. Jamshidi, C. Pahl, N. C. Mendonça, J. Lewis, and S. Tilkov, "Microservices: The journey so far and challenges ahead," *IEEE Software*, vol. 35, no. 3, pp. 24–35, May 2018.

[7] S. Newman, *Building Microservices*, 1st ed. O'Reilly Media, Inc., 2015.

[8] N. Dragoni, S. Giallorenzo, A. L. Lafuente, M. Mazzara, F. Montesi, R. Mustafin, and L. Safina, "Microservices: yesterday, today, and tomorrow," in *Present and ulterior software engineering*. Springer, 2017, pp. 195–216.

[9] A. Taherkordi and P. Herrmann, "Pervasive smart contracts for blockchains in iot systems," in *Proceedings of the 2018 International Conference on Blockchain Technology and Application*, 2018, pp. 6–11.

[10] M. Atzori, "Blockchain technology and decentralized governance: Is the state still necessary?" *SSRN*, 2016.

[11] M. A. Cyran, "Blockchain as a foundation for sharing healthcare data," *Blockchain in Healthcare Today*, 2018.

[12] W. Wang, D. T. Hoang, P. Hu, Z. Xiong, D. Niyato, P. Wang, Y. Wen, and D. I. Kim, "A Survey on Consensus Mechanisms and Mining Strategy Management in Blockchain Networks," IEEE, 2019.

[13] R. Böhme, N. Christin, B. Edelman, and T. Moore, "Bitcoin: Economics, technology, and governance †," *Journal of Economic Perspectives*, vol. 29, pp. 213–238, 05 2015.

[14] M. Alharby and A. van Moorsel, "Blockchain-based smart contracts: A systematic mapping study," *Computer Science Information Technology*, 2017.

[15] R. Tonelli, M. I. Lunesu, A. Pinna, D. Taibi, and M. Marchesi, "Implementing a microservices system with blockchain smart contracts," in *2019 IEEE International Workshop on Blockchain Oriented Software Engineering (IWBOSE)*. IEEE, 2019, pp. 22–31.

[16] E. Evans, *Domain-driven design : tackling complexity in the heart of software*, 1st ed. Addison-Wesley, 2014.

[17] H. Vural, M. Koyuncu, and S. Guney, "A systematic literature review on microservices," *Computational Science and Its Applications – ICCSA*, 2017.

[18] D. A. da Silva, F. Kfouri, S. C. dos Santos, L. H. Coura, W. Cristoni, G. S. Goncalves, L. G. dos Santos, J. C. O. Junior, B. M. R. da Fonseca, J. C. L. Costa *et al.*, "Urgent and emergency care: an academic application system case study," in *16th International Conference on Information Technology-New Generations (ITNG 2019)*. Springer, 2019, pp. 143–152.

[19] H. Dai, H. P. Young, T. J. Durant, G. Gong, M. Kang, H. M. Krumholz, W. L. Schulz, and L. Jiang, "Trialchain: A blockchain-based platform to validate data integrity in large, biomedical research studies," *arXiv preprint arXiv:1807.03662*, 2018.

[20] D. Nagothu, R. Xu, S. Y. Nikouei, and Y. Chen, "A microservice-enabled architecture for smart surveillance using blockchain technology," in *2018 IEEE International Smart Cities Conference (ISC2)*. IEEE, 2018, pp. 1–4.

[21] B. Ramesh, L. Cao, and R. Baskerville, "Agile requirements engineering practices and challenges: an empirical study," *Information Systems Journal*, vol. 20, no. 5, pp. 449–480, 2010.

[22] S. K. Lo, X. Xu, Y. K. Chiam, and Q. Lu, "Evaluating suitability of applying blockchain," in *2017 22nd International Conference on Engineering of Complex Computer Systems (ICECCS)*. IEEE, 2017, pp. 158–161.

[23] A. Dyck, R. Penners, and H. Lichter, "Towards definitions for release engineering and devops," in *2015 IEEE/ACM 3rd International Workshop on Release Engineering*. IEEE, 2015, pp. 3–3.

[24] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in software engineering*. Springer Science & Business Media, 2012.