BCaR: Blockchain in Car Registration

Tiago Rosado
Instituto Superior Técnico, Lisboa, Portugal
November 2018

Abstract—Blockchain technology enables the development of decentralized business models but centralized organizations, as governments, can adopt this technology to improve safety and security of critical data. In this document we present a solution, based on Hyperledger Fabric, to implement a car registration system which can improve interoperability between government entities and might be extended to a cross-borders system, extending the cooperation efforts in the European Union. The system handles the processes related to car registration, from registering a vehicle, changing ownership status or registering a leasing contract between a lessor and a lessee. This system can ease the information exchange among multiple states as the car registration information is distributed to each government entity in a single decentralized system. We latter analyze the benefits and implications of a blockchain technology application and present an overview of the system's performance.

Index Terms—blockchain, car registration, critical data storage, distributed ledger

1 Introduction

Blockchain technology has reached the mainstream by numerous criptocurrencies implementing this technology and shook the concept of currency. Bitcoin, as the most known application of blockchain technology, implements a digital currency not relying on any central authority to be managed [1]. The decentralization provided by blockchain technology can be considered a direct competitor to organizations relying on a centralized business model, such as banks and governments.

However, centralized organizations can also look at blockchain technology as an innovation with the ability to improve their efficiency. Currently some of these organizations are researching and implementing blockchain technology to the benefit of business efficiency and government's transparency. One of such example is the Estonian government who implemented some of its services using this technology around 2012 [2].

1.1 Objectives

Considering the potential of blockchain technology, the main objective of this work is to implement a car registration infrastructure based on blockchain. A car registration system based on blockchain can decentralize this kind of registries and as a consequence improve data availability and resilience to faults. As a set of entities, ranging from leasing companies to government offices, rely on the car registration system, a decentralized system based on blockchain can improve its performance and safety when compared with a centralized solution.

Given the decentralization inherent to blockchain it is crucial to understand the role of an entity such as the national registry. As it will be discussed throughout this document, a blockchain application for car registration can still take in consideration the authority of the national registry entity. With this requirement, a blockchain based car registration system can benefit from decentralization but still maintain a centralized authority to partially manage and control the system.

This technology can also represent an effort to improve government efficiency and transparency [2]. Therefore we propose an implementation of blockchain technology for car registration using smart contracts and analyze its impact on car registration business processes.

2 Background

In this chapter we go over a set of blockchain and distributed consensus concepts. We start by taking an overview of a centralized car registration system. Then e define what are smart contracts and how blockchain networks reach consensus for electing blocks. We go over the concept of proof-of-work and later introduce the use of Byzantine Fault Tolerant based consensus in blockchain technology, comparing both approaches. As an application of a BFT consensus algorithm to the blockchain technology we introduce Hyperledger Fabric and later define the differences between permissioned and permissionless blockchains as well as the differences between private and public blockchains. Finally we go over some of the applications currently using blockchain technology.

2.1 Current car registration system

A car registration system, as implemented by most government entities around the world, is usually a centralized information system. This information system handles every information related to car registration and is managed by a national registry entity, although other governmental and non-governmental entities have access to services handling car registration information.

Citizens are able to ask for information related to vehicles, as most of the information stored in the car registration system is publicly available. Government entities responsible for controlling motorized vehicles are able to interact with the car registration system to issue and cancel registration plates and to change the official characteristics of a vehicle.

Tax authorities within the country have system access to declare imported and exported vehicles, as well as to get information about vehicle ownership changes. Regarding seizure orders, lawyers, courts and solicitors are allowed to interact with the car registration system to issue those orders. Finally external entities, such as leasing companies, are also allowed to connect to the car registration system to consult car records.

A car registration system may be composed of several servers responsible for different operations, such as having some servers responsible for providing web services to external entities relying on car registration information, as tax authorities, vehicle regulation authorities or even some companies directly related to vehicle registrations, such as leasing companies. Another component of such a system is the use of a client interface for employees handling car registry data, which can be presented in a client side application or a website providing a mean to execute the existing operations over a car registry.

All of the system components described interact with the core of the information system which in most of the cases is a relational database hosted on a different server, responsible for managing and storing the information required to handle car registries. Each of the components described might vary on its complexity by having backup servers to tolerate system fault or attacks from malicious parties, preventing data losses and system failures on each of the components constituting the car registration system.

As within European Union (EU) there is a cooperation among member states to exchange car registration information, as to issue transit violations or simply consult registry information regarding a vehicle in a different EU country. Thus, some of the described system components are exposed to cross-borders access to provide vehicle information to different state members. However, as opposed to using blockchain technology state members rely on the variability of each other's car registration systems to read cross-borders vehicle's information.

2.2 Smart contracts

Smart Contracts are programs written to form agreements between users in the blockchain. Using smart contracts it is possible to ensure the clauses of a contract are accomplished and that breaching the contract is expensive or even prohibitive [3].

Blockchain establishes a consensus based on minimal trust between network nodes to execute smart contracts. When a node receives a transaction, contract functions are ran to ensure the validity of the transaction and the conditions stated in the contract are met. In case of failure the transaction is discarded by the network nodes.

Extending the capabilities of smart contracts, it is possible to run decentralized applications based on the blockchain technology. Therefore, we can create applications such a car registry platform completely based on smart contracts.

2.3 Consensus

In order for the blockchain network to agree on the next block to add to the blockchain there is a need to decide who should be the author of the block. Bitcoin uses a proof-of-work to decide this matter [1]. Given the concurrency between nodes it is possible that two nodes can broadcast different verisons of the next block at the same time. In this situation, other nodes start working over the first block they reveive but still save the other branch, this represents a fork. A fork is removed once the next proof-of-work is found [1].

2.3.1 Proof-of-work

A proof-of-work is hard to produce and easy to verify, created by nodes in the blockchain network. This implies the need for a computational intensive problem to be solved. The node responsible for solving the problem is the potential author of the next block to be added to the blockchain [4]. In case of Bitcoin, a node needs to scan for a nonce value for a block so that the hash of the block starts with a defined number of zero bits.

However, this type of proof has the potential side effect of slowing the transaction processing within the network as major computational work is wasted in the proof-of-work to create a block instead of transaction processing. The approach used by proof-of-work increases the difficulty of successful attacks, given the need for a great computational power to create a proof-of-work and use it to tamper blockchain data.

2.3.2 BFT approach

Initial blockchains such as Bitcoin and even the improved version implemented in Ethereum still have disadvantages related to the methods used for node consensus, as distributed consensus algorithms have scalability constraints.

Considering distributed consensus, it is important to compare traditional Byzantine Fault Tolerant (BFT) algorithms [5], [6], [7] to consensus mechanisms implemented in traditional blockchains, namely proof-of-work approaches. Considering a BFT based blockchain is expected to follow the path of standard BFT algorithms, this kind of blockchain would have poor scalability compared with standard blochains using proof-of-work approach. However, BFT algorithms have a performance advantage [8]. On the other hand, proof-of-work is easier to implement in public blockchains as nodes only need to know about one other node to perform the proof-of-work and this approach enables decentralized identity management. BFT algorithms require each node to know every other node in the network so that consensus is reached, normally requires a trusted central authority, therefore this approaches are more likely to be implemented in permissioned or private blockchains [8].

Regarding consensus conflicts, such as the creation of temporary forks in the blockchain, BFT based algorithms manage them with greater efficiency compared with proof-of-work. BFT algorithms, contrary to proof-of-work, achieve this as they grant the property of consensus finality, as in Definition 1. Considering the realization of this property, block addition to the blockchain is immediately confirmed.

Definition 1. (Consensus Finality) If a correct node p appends block b to its copy of the blockchain before appending block b′, then no correct node q appends block b′ before b to its copy of the blockchain [8].

Resilience to attacks is also a matter of analysis as one of the key advantages of blockchain is assurance of immutable data, once the data is stored in the blockchain. Proof-of-work approach in fact supports up to 50% of faulty nodes in the network, however regarding Bitcoin's approach to fault mitigation, tolerance drops to 25% [9], contrasting with BFT algorithms that support up to $\frac{n}{3}$ faulty nodes, where n is the number of nodes on a totally asynchronous network.

Given the possibility of forks, Bitcoin network nodes only take a block as final once 6 or more blocks have been appended to the block. On the other hand, this approach can be circumvented by timestamp manipulation [8]. BFT algorithms, as complying with Definition 1, support long asynchronous periods and global outages. Latency in BFT algorithms usually matches network latency, contrary to proof-of-work approach where rising block size translates in higher throughput with the downside of higher latency. As latency of the system increases the number of possible blockchain forks increases, as mentioned by Vukolič [8].

2.4 Hyperledger Fabric

Hyperledger Fabric is a blockchain infrastructure sponsored by Linux Foundation and IBM. Hyperledger Fabric, as a private blockchain, restricts the nodes participating in the system to trusted and identifiable ones. Restricting the nodes of the system enables performance improvements over consensus mechanisms and can reduce the power consumption of such system. As the nodes are known to every element in the system, Byzantine Fault Tolerant (BFT) algorithms can replace the typical power hungry mechanism of Proof-of-Work. Permissioned blockchains can usually be implemented within a group of entities who may not completely trust each other [10].

Analyzing Byzantine Fault Tolerant State Machine Replication (BFT-SMaRt), it is noticeable that they differ from blockchain networks as: Multiple applications can run concurrently; Applications can be installed on demand by anyone and application code is untrusted code. Thus, most blockchain systems use an hard coded consensus mechanism and a Order-Execute architecture, requiring transactions to be ordered, propagated and later sequentially executed by the nodes. Hyperledger Fabric uses a Execute-order-validate architecture [10], [11], based on a modular consensus mechanism that can be adjusted to the specific need of smart contracts running on the blockchain.

Through Execute-order-validate architecture, a transaction is divided into three different phases. During execution phase, each transaction is executed and its correctness verified by a restricted set of peers called endorsement peers. During this phase it is possible for transactions to be executed in parallel. The second phase is assigned to an ordering service, responsible for establishing total order of transactions received and already signed by endorsement peers, using the consensus mechanism defined and fabricating blocks accordingly. Ordering service nodes are also responsible for updating the state of the blockchain to all peers using atomic broadcast. For any of the operations, ordering service nodes do not need to execute smart contracts, know about the current application state nor validate the smart contracts. On validate phase, transactions are validated by the remaining peers of the network. If there is no issue in this last step, the block is then appended to the blockchain on each peer's local copy.

Regarding Ordering Services' operations related to consensus mechanisms, Hyperledger Fabric currently has three consensus mechanisms implemented [10]. A one node consensus, requiring only one node to establish total order of transactions, a method normally used to speed up the development environment of smart contracts, a Crash Fault Tolerant (CFT) based ordering service ran on cluster, and a BFT-SMaRt [11] consensus mechanism. On the matter of block formation by the Ordering Service, a block is formed when one of the following conditions occurs: (1) the maximum number of transactions per block is reached, (2) the maximum block size is reached or (3) a certain time has passed since the first transaction was added to the block.

2.5 Permissionless versus Permissioned Blockchains

Public blockchains rely on public nodes, meaning that any node can contribute to the network by running a program designed to keep a local blockchain copy, contribute to network consensus, contribute to process transactions and create blocks. As public blockchains enable any node to contribute to the network, information stored in this type of blockchain setup needs to be public. Considering public blockchains, the requirement of making public any data stored in the blockchain has the advantage of increasing data transparency but sacrifices privacy [2].

Permissioned blockchains emerged as a necessity to restrict blockchain network participants to identifiable and explicitly authorized nodes with specific permissions [11]. Therefore private blockchains provide mechanisms, for the organizations managing the blockchain, to restrict the nodes accessing and contributing to the blockchain. As a result, private blockchains increase the confidentiality of the data they store. However,the fact of relying on a central authority and a restrict number of nodes may provide less resilience and sturdiness.

2.6 Applications

Although cryptocurrencies, as Bitcoin, are the most popular application of blockchain technology, there is a large set of applications based on blockchain backed by governments, banks and private companies. As stated in [2], this technology also improves the privacy of citizens and transparency of government work.

Applying blockchain technology to business may lower operations costs and coordination efforts, as the system automaticaly handles possible conflicts [2]. Distributed ledgers can be used to better secure data itself contained in the blockchain and easily share citizen's data between government entities.

The Estonian government implemented a blockchain solution in 2012 applied to registries of national health system (e-Health Records [12]), judicial and citizen registry system. A concrete application of the blockchain technology by the Estonian government is e-Residence [13]. Through e-Residence application, the Republic of Estonia issues digital identities (together with a digital ID smart card) for people and organizations around the world who want to develop

a location independent business. Using this application, e-Residents can start a company, access business banking, sign and securely send documents and declare taxes online [13].

3 Business Processes of Car Registration

In this chapter we take an overview of the business processes involved in the operation of the current car registration system. A new car registration system based on blockchain technology, as proposed in this thesis, may bring innovation over a centralized car registration system and change some of the business processes currently in place. An analysis of the current business processes, of the Portuguese national registry entity, will be conducted and a set of changes over the current business processes will be suggested as a way to improve those processes. Finally, we will go over a scenario using a blockchain based car registration system and model some of the business processes associated with this new system.

3.1 Current Business Processes

Analyzing the current car registration system, a user can interact with the system mainly through two methods. A user can interact, directly or indirectly, with the system by going to a national registry office and require information regarding a vehicle or require to change a vehicle's registry. Then a national registry employee consults the car registration system and provide the information to the user or update the vehicle's registry. Information changes, for registered vehicles, mostly require the owner or an authorized party to fill in a form, such as [14].

On the other hand, a vehicle owner can have a narrower but more direct access to the car registration system by using an online platform, although most of the information updates still require approval from a registry employee. A simplified version of the required actions to interact with the current car registration system, is shown as a business process model diagram in Figure

However, a back-office process, is still needed to fully process the citizen's request. The back-office process starts when a registry employee looks for the list of pending requests in the car registration system. As a registry employee selects a pending request, he can reject the request and provide a brief textual explanation on the reason leading to that rejection. On the other hand, the registry employee can complete the pending request by inserting the necessary documentation for the request to proceed. After this step, document's verification is executed by the registry employee. Finally, the request result is issued and the process is finished.

3.2 Business processes on a blockchain based system

Given the properties of a blockchain based car registration system, business processes presented in Section 3.1 can be further modified to benefit from blockchain properties. Thus we propose a blockchain based car registration system on which change requests over vehicle information, can be firstly registered to the system, as soon as the request is issued by the respective participant and the request's payment is confirmed. A blockchain based car registration system would then work following the principle that information updates over vehicle's data are correct unless this information is latter proven to be wrong. This approach to updates in a information system takes advantage of blockchain technology and its immutability to simplify business processes.

3.2.1 Register a new vehicle

Taking into account, the registration of a newly fabricated vehicle into a blockchain based car registration system, we assume the intervention of a registry employee is recommended. This recommendation is based on the fact that this operation might require several authorizations from authorities, such as the Direction for Motorized Vehicles.

We propose a registration process for a new vehicle in the car registration system based on blockchain. A request is made through a front-office process or through a online portal and it is latter processed by a national registry employee. The employee will verify the information available in the request and possibly complete some of the information. On a second step, the employee will verify that all request's information is according to the documents provided and might add new documentation before completing the process.

3.2.2 Request a change of ownership

Regarding ownership change process we propose a two phase process, based on the principle stated in the beginning of this section. The current owner of a vehicle wishing to pass his ownership position to another entity is required to fill a online form with all necessary information and documentation. Once this form is submitted and the necessary payment is confirmed by the system, a pending ownership change is submitted to the blockchain based car registration system which is latter confirmed by the prospect owner.

As the vehicle's owner issues a change of ownership request, with information about the new owner and its share of ownership, the vehicle ownership is registered as a pending ownership and the prospect new owners is required to take action. The prospect owner is then informed of this pending operation and is able to accept or reject the ownership change. Only after the prospect owner confirms the ownership change, the new owner is effectively registered as owner of the vehicle in the blockchain.

3.2.3 Register as guarantee

Once a vehicle owner wants to register a vehicle as guarantee for a loan, he is required to issue a request with the intended documentation. Once the payment for the request is confirmed, the blockchain is updated with new information regarding the guarantee tied to the vehicle.

3.2.4 Associate lease contract

When owning a vehicle, the owner is able to issue a lease contract signed by the owner, as a lessor and a third-party, as a lessee, with a specific duration. A lease contract expects the lessee to make regular payments for a specified number of months. In exchange, the lessor gives permissions for the lessee to use the vehicle.

Therefor we propose the lease contract association in a blockchain based car registration system is divided into

two phases. The lease registration process is started by a vehicle owner, submitting lease information. Then the lessee implied in the lease contract is required to confirm or deny his involvement in the contract. Considering the lessee accepts the lease contract the contract is validated in the system. Otherwise, the lease operation is reverted once the prospect lessee rejects it.

3.2.5 Execute a vehicle seizure

A vehicle seizure occurs when, by judicial order, the ownership of an asset, in this case a vehicle, is forcibly changed in order to liquidate owner's debt. A vehicle seizure request is currently made by a judicial officer and latter executed by national registry employees. On a blockchain based car registration system, we propose the process to be executed by a judicial officer. As soon as a judicial officer submits the required information for seizure of the vehicle, the action is submitted to the blockchain and the ownership of the vehicle is changed.

4 BCar System

In this chapter we propose a car registration system based on blockchain technology. At first we consider the requirements of a car registration system, based on this analysis we propose to implement a blockchain based information system with granular access control. A set of use cases is defined as part of the requirements for the proposed system.

A data model is defined, taking into account vehicle information required by European law, as well as the participants of the proposed system. A light consideration on the permissions of each participant is presented. Then a set of operations available in the system are presented.

4.1 Requirements

Currently, car registration systems in Europe are controlled and maintained by each member state. Therefore, a car registration system based on blockchain technology can distribute most of the maintenance effort and even some of the system's control by network nodes detained by government entities.

As part of a car registration system's requirements, at least the following use case scenarios:

A. Main use cases:
   1. A car seller wishes to transfer car ownership to a car buyer.
   2. A leasing company provides a contract to a client with the clauses of the client using the car for a defined period of time on which the leasing company is responsible for paying maintenance and insurance expenses in exchange for a defined monthly payment by the client. As the contract reaches its ending, the client can buy the car from the leasing company for a previously defined amount.
   3. A car owner submits a request to give the car ownership as a guarantee to a creditor in case the car owner does not fulfill the contract.

B. Secondary use cases:
   1. A car manufacturer submits a request to create a car registry entry for a newly produced car.
   2. Given a judicial order and in order to liquidate a car owner debt, the car ownership is transferred to the entity responsible for collecting the debt payments. This use case should be executed by a judicial officer.

4.2 Solution

Considering the granular access control required for a government service, such as the car registration system, it was necessary to select a permissioned blockchain. As we intend to provide most of the blockchain control to a set of entities such as the national registry institution of each European state member. This configuration ensures the safety and confidentiality of data.

4.2.1 Participant definition

As the car registration system requires access control to registry information, a set of participants were defined and their specific permissions.

Person A Person type participant encompasses the simplest entity able to take ownership of a vehicle. A person is able to execute the following operations, when owning a vehicle:

- Start a change of ownership of a vehicle
- Start a lease contract
- Request a lease contract cancellation
- Reject a lease contract cancellation
- Accept a lease contract cancellation
- Add the vehicle as guarantee for a loan
- Request for a vehicle guarantee to be canceled

Company A collective entity, is represented as a Company type participant on the proposed car registration system. A Company type participant, requires a list of entities which are Person type participants. This entities, identified in a Company participant, are the Company owners or entities responsible for this Company's actions on the car registration system. Car registry operations on behalf of a Company participant are required to be performed by the singular entities identified as Company owners or Person type participants in charge of the Company's actions on the car registration system.

Judicial Officer Operations executed to fulfill judicial orders, as vehicle seizures, are issued by judicial entities. Thus, judicial entities are represented as a Judicial Officer type participant.

Registry Employee A Registry Employee type is destined to users of the car registration system, working for national registry entities. When compared to the other participant types, registry employees have unrestricted permissions to car registries in the information system. Registry Employee participants are able to solve conflicts if they arise on the system.

External Entity An additional participant was required to be defined in order to include operation's permissions within the car registration system which were executed by external entities, as tax authorities, to the vehicle registry and were not included in the above mentioned participant types.

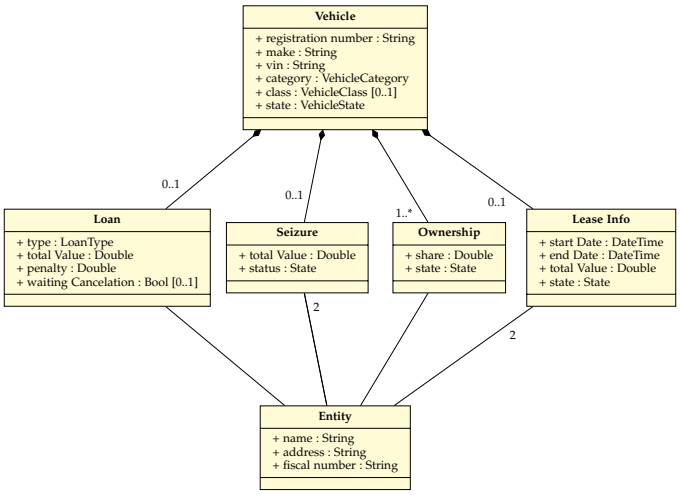

Fig. 1. Proposed vehicle data model

4.3 Data model

Based on European regulations for motorized vehicles and the vehicle registration modification form [14], a simplified data model was built. A vehicle is registered in the car registration system with the following information, a registration number, a make, a vehicle identification number and category. Vehicle owners are identified and their shares are registered in a Ownership object. A Ownership object for each owner is associated with the vehicle, as presented on Figure 1. A vehicle can be registered in the car registration system in several conditions, the different available states a vehicle can belong to are *Active*,*Inactive*,*Destructed*,*Suspended* and *Stolen*.

Lease information related to a certain vehicle is registered, on the blockchain based car registration system, when a vehicle is subject to a lease contract. This information,as presented in Figure 1, is stored in the Lease Info object associated with the vehicle. Each vehicle must be tied to a maximum of one loan at a time.

4.4 Car registry operations

4.4.1 Create a vehicle

Registering a vehicle on the car registrations system is usually an operation done by a national registry entity. However most of the vehicle information is given by an external entity responsible for managing motorized vehicle regulation.

In order to simplify interactions over the blockchain-based car registration system we assume this information is registered manually by national registry personal. Therefor when a vehicle needs to be registered, the following information is required: registration number (number plate), VIN, make, model, vehicle category and the owners information along with their respective shares.

4.4.2 Change Ownership

Changing ownership of a vehicles is separated in two steps and is specific for each owner, as vehicles can have multiple owners with different shares each. As expected, this operation is only allowed to be executed by the owners of the vehicle subject to ownership change.

Thus the hole ownership change needs to be initialized by the current owner wishing to give up his position on a certain vehicle, as proposed in Section 3.2.2. This step requires the owner to issue a *Change Owner* transaction, specifying the vehicle identification number, the registration number and make of the vehicle as well as the list of new owners to which the current owner will give his share of the vehicle.

As the first step is concluded, the vehicle ownership changes to the new owners. However the new ownership is still required to be approved by the new owners, ensuring the old owner and new owners are registered and associated with the vehicle, providing the necessary information in case of a judicial dispute.

In order to complete the ownership change, only the new owner registered in the initial *Change Owner* transaction is able to confirm this operation issuing a *Confirm Ownership* transaction specifying the vehicle's VIN, registration number and make as well as the ownership share intended to be given to him. Only after this step the ownership information is considered valid.

4.4.3 Lease

As proposed in the business process presented in Section 3.2.4, the process for registering a lease contract is separate into a two steps operation. The owner is proposed to issue a *Create Lease* transaction which can be canceled through a *Cancel Lease* transaction by the lessor, the lessee or by a registry employee. On the other hand, the lease registration can be accepted either by a registry employee or by the prospect lessee issuing a *Confirm Lease* transaction.

As part of the *Create Lease* transaction, the vehicle identification number, registration number as well as the make of the vehicle are required. Considering lease contract information, the *Create Lease* transaction will also refer the start and end dates of the contract, the expected total value of the lease and the third-party to which the permission to use the vehicle is given.

Once the *Create Lease* transaction is correctly executed, the vehicle enters a *Waiting lease confirmation* state. In order to successfully register a lease contract, the lessee registered in the initial *Create Lease* transaction is required to issue a *Confirm Lease* transaction containing the vehicle identification number, the vehicle's make and registration number as well as the total value accorded in the lease contract. In case the transaction issued by the lessee matches with the information passed by the lessor, in the *Create Lease* transaction, the system registers the lease as associated with the vehicle and the vehicle enters the *Active lease* state.

A *Cancel Lease* transaction can be executed by any of the parties involved, either the lessor, the lessee or a national registry employee. *Cancel Lease* transaction must contain the vehicle identification number, registration number, make, the total value contracted on the lease, as well as the respective parties involved, the lessee and the lessor.

For a lease contract termination to happen, it is necessary to issue a *Cancel Lease* transaction. Considering the *Cancel Lease* operation is issued by a judicial officer, this action takes effect immediately, thus the lease contract is revoked.

In case the *Cancel Lease* transaction is issued by a lessee or a lessor and the vehicle is in *Active lease* state, the process

takes two steps, as both parties need to agree on canceling the contract. For the lease contract to be actually terminated, after a *Cancel Lease* transaction, it is necessary for the other party to issue a *Confirm Lease Termination* transaction with the same arguments as *Cancel Lease* transaction. On the other hand, it is possible for the lessor or the lessee to reject or cancel the lease termination process by issuing *Cancel Lease Termination* transaction.

4.4.4 Seize vehicle

Given a process on which a vehicle owner is subject to a court to liquidate his debts, judicial officers can seize a vehicle or issue a pending seizure. Thus, if a pending seizure is in place, the asset can not be sold or change ownership until the process is solved. Latter the same process can also result in a seizure and the ownership of the asset can only be changed according to court order.

In order to issue a pending seizure for a vehicle, a judicial officer is required to submit a *Issue Pending Seizure* transaction. Considering a case on which the *Pending seizure* state needs to be reverted, a *Cancel Seizure* transaction should be emitted. The *Cancel Seizure* transaction can only be issued by a judicial officer.

In order to effectively seize the asset, a *Issue Seizure* transaction is required to be issued. When a vehicle is not tied to any pending seizure , it is possible for the judicial officer to directly emit a *Issue Seizure* transaction. During *Issue Seizure* transaction execution, the ownership of the vehicle is changed so the owner implied in the seizure is removed as owner and the creditor becomes the new owner.

4.4.5 Register as guarantee

In order to register a vehicle as guarantee, the vehicle's owner is required to order a *Register Guarantee* transaction mentioning the creditor, the type of loan (Collateral or Mortgage), total value of the loan given to the vehicle owner and the penalty to which the debtor needs to pay in case the loan is paid ahead of time. During the *Register Guarantee* transaction execution, a set of rules is verified in order for the transaction to be successful. The transaction can only be issued by the owner and it is not possible to register a vehicle as a guarantee to a loan if the vehicle is already tied as a guarantee to a loan.

It is possible to cancel the guarantee by issuing a *Cancel Guarantee* transaction. A creditor can issue a *Cancel Guarantee* transaction with immediate effects, requiring to identify the vehicle though the vehicle identification number, registration number and make.

On the other hand, if *Cancel Guarantee* transaction is issued by the vehicle owner it requires the creditor to confirm or deny the operation using accordingly the transactions *Confirm Guarantee Cancellation* or *Reject Guarantee Cancellation*.

4.4.6 Change Vehicle State

A change in a registered vehicle's state is usually presented by an external entity such as the Direction for Motorized Vehicles, however as simplification, the national registry employee is in charge of this update on the car registration system. Thus, as a national registry employee receives a request for updating the state of a vehicle in the system, he issues a *Change State* transaction. The vehicle states which a vehicle can have were already described in Section 4.3.

5 Implementation

Based on the data model described on Section 4.3 a similar data model was created using a domain specific language of Hyperledger Composer which latter was deployed to a Hyperledger Fabric version 1.1 based network. Hyperledger Composer Modeling Language is an object-oriented modeling language designed to define the domain model for a business network defined for the Hyperledger Fabric. Each transaction described through out Section 4.4 was also modeled in Hyperledger Composer Modeling Language. Every information regarding the created data model was stored on the Hyperledger Fabric's blockchain with no off-chain database handling car registry records.

Transactions tied to smart contracts were also defined as resources using the domain specific modeling language. Furthermore, all the developed smart contract functions where developed using Javascript and Hyperledger Composer API version 0.19.7. Hyperledger Composer provides an interface adaptable to any language to interact with Hyperledger Fabric blockchain, currently there is also an implementation of this API for Golang.

Specific access control rules where defined using Hyperledger Composer access control language. Hyperledger Composer access control is define trough a set of rules which can detail CRUD access control considering the participant executing the transaction or reading asset information.

5.1 Implemented access control rules

Regarding the vehicle creation, as presented in Section 3.2.1 and latter described in Section 4.4, this operation can only be executed by a Registry Employee participant type. Thus, a rule allowing Registry Employee type participants to issue *Create Vehicle* transactions was created.

Considering *Change Ownership* transactions, all Entity participants, thus all participants of the system are entitled to issue this transaction. However, the verification of this rules is done through the smart contract's code, as verifying this rules through the access control mechanism leads to performance issues. As the change of ownership flow described in Section 4.4.2 includes a second step, through *Confirm Ownership* or *Cancel Ownership* transactions, a set of rules where also modeled for this transactions.

In the process of creating a lease, as issuing a *Create Lease* transaction, only the Registry Employee participants and vehicle owners are entitled to execute this transaction. As presented earlier in this Section, given performance issues, the access control verification when issuing a *Create Lease* transaction is provided through the smart contract. Regarding lease transaction flow, as mentioned in Section 4.4, the same rules, as described for *Create Lease* transaction, are defined for *Confirm Lease*, *Cancel Lease*, *Confirm Lease Termination* and *Cancel Lease Termination* transactions.

All transactions regarding seizure flow, thus *Issue Pending Seizure* , *Issue Seizure* and *Cancel Seizure* transactions will have the same rules as following. Hyperledger Composer

access control mechanism will have a rule defined allowing Judicial Officer and Registry Employee participant types to create each of the vehicle seizure flow transactions.

Registering a vehicle as guarantee (Register as Guarante and Cancel Guarantee transactions), requires that only vehicle owners or registry employees are allowed to execute the transaction. Thus, the verification of ownership and verifying that the transaction issuer is a registry employee is made trough the smart contract's code. Regarding *Cancel Guarnatee*, *Confirm Cancelation* and *Reject Cancelation* transactions, when issued by a creditor, a verification that the issuer is a creditor for the loan is also made through the smart contract's code. Finally, only Registry Employee participant types are allowed to create *Change Vehicle State* transaction.

5.2 Hyperledger Fabric configuration

In this section we will go over each system's components, explaining their contribution for the an Hyperledger Fabric infrastructure. Then we present how a client is able to interact with the Hyperledger Fabric's blockchain and describe each step required to execute a transaction, from the moment the transaction is issued to the final steps taken to add the transaction's updates to the blockchain.

5.2.1 System components

As part of the Hyperledger Fabric infrastructure, a network peer can be a endorsement peer, an anchor peer or a normal peer. As each peer type is not mutually exclusive, it is possible for a peer to be both a endorsement peer and an anchor peer.

Smart contracts' data is stored as key-value pairs in this database. Thus, when executing a transaction, the state database is used to make chaincode execution more efficient. As alternative, CouchDB[1] can be used as an external state database, providing additional query support and richer queries when compared with the default state database.

Regarding the different peer types, an endorsement peer is responsible for validating transactions by executing transactions' chaincode (functions of the smart contract required by the transaction). On the other hand an anchor peer is responsible for communicating with the Hyperledger Fabric network exterior to the organization on which it belongs. Finally, a normal peer is only responsible for receiving a block of transactions issued by the ordering service and updating its local blockchain with the corresponding block.

The ordering service is a core component of any Hyperledger Fabric infrastructure. The service is composed by ordering nodes, responsible for reaching a consensus on building a block of transactions to update the blockchain. As part of the infrastructure, ordering nodes gather transactions submitted by clients, verify the endorsement signatures of those transactions and build a block containing a batch of signed and verified transactions. The endorsement policies in place may vary according to the configuration setup for a specific Hyperledger Fabric network, defined by network's administrators.

A total three organizations is presented and each of them is composed by four peers. An endorsement organization (which can be a National Registry Entity) is responsible for endorsing transactions, thus is composed by two endorsement peers and two normal peers. In addition, an endorsement peer and a normal peer also act as anchor peers, handling Fabric's communications outside of the organization. The remaining organizations are composed by four peers each and two of those peers act as anchor peers for the organizations. Finally we consider a client application responsible for providing a platform for the end-users to use the proposed system through an API provided by Hyperledger Fabric.

1. CouchDB - http://couchdb.apache.org

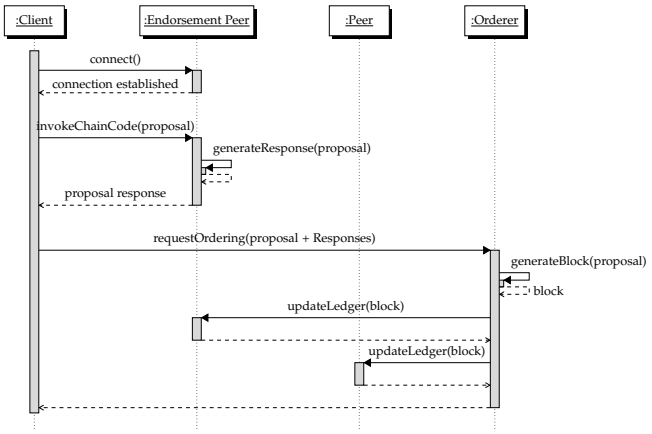

Fig. 2. Block proposal and addition mechanism for Hyperledger Fabric.

In order for the different components to interact and to maintain access control to information stored in the blockchain, Hyperledger Fabric provides channels. In case of the BCar system, a unique channel is used in the Hyperledger Fabric infrastructure.

5.2.2 System behavior

In order to use the proposed system, a client connects to an Hyperledger Fabric network running the described smart contract functions. This connection is established using Hyperledger Fabric API.

On the other hand, transactions with intent to update blockchain's state require to be submitted through Fabric's consensus mechanism, as shown in Figure 2. Then he submits the blockchain's update proposal to a sub-set of peers responsible for verifying and signing transaction proposals, named endorsement peers. According to a customizable policy, a specific number of valid proposal signatures are required in order for the transaction proposal to be accepted as valid by the network peers.

As the client collects the required proposal signatures, the list of signatures along with the proposal are sent to a ordering peer (or orderer), responsible for grouping signed transactions into a block. Once transactions are ordered into a block, the block is sent to every peer on the network and the blockchain is updated.

6 Results

To assess the performance of the blockchain based car registration system proposed we had at our disposal a single Virtual Machine (VM) with 8vCPU, 32 GB of RAM and 40

GB of HDD space, running Ubuntu Xenial (16.04.5 LTS). Regarding the software setup, tests were performed on top of Docker v18.06.0-ce. Each peer, certified authority node and orderer was running on the base image of Hyperledger Fabric x86 64 v1.1.0. As state database, instances of Hyperledger Fabric CouchDB image version x86 64 v0.4.6 were used.

In order to measure system's performance, Hyperledger Caliper [2] software was used. Given Hyperledger Caliper is still in development phase, a version based on commit *c37860b042*[3] was adapted for the Bcar registration system to be tested.

Hyperledger Caliper allows to measure the performance of multiple blockchain implementations, one of them is Hyperledger Fabric, given a set of use cases. This tool can produce reports containing various performance indicators, such as transactions per second,transaction latency and resource utilization.

Considering Hyperledger Fabric nodes setup, we configured the system with a total of 9 containers with a total of 2 Hyperledger Fabric peers. The system was configured using solo consensus mechanism. Each Hyperledger peer was tied to a different organization, so 2 organizations where configured in total, requiring a certified authority running on a separate container for each organization.

Organization's peers used in the experimental setup were configured as endorsement peers. Thus an additional container (Chaincode Peer) is used to execute the transaction's required chaincode.

6.1 Methodology

A set of functions were selected to sample the system's overall performance, based on the principal transaction flows described in Section 4. Thus, we evaluated the performance of *Create Vehicle*, *Change Vehicle State*, *Change Ownership*, *Issue Seizure* and *Register as Guarantee* functions.

Each function was tested three times with a varying number of fixed throughput issued, for each block size configuration. Firstly, a set of 100 transactions where issued against the BCar registration system with a fixed send rate of 50 transactions per second (TPS). Then, a set of 200 transactions where issued with a fixed send rate of 100 TPS. Finally, a set of 400 transactions where issued with a fixed send rate of 200 TPS against the system. As each set of transactions was sent, the throughput was calculated dividing the number of transactions successfully executed by the time taken to execute such transactions.

As each component of the Hyperledger Fabric system is running on a single Virtual Machine, we took advantage of Hyperledger Caliper functionalities by tracking the RAM and CPU usage of each Docker container. As a remark it is expected for the CPU usage percentage to surpass 100% usage as docker statistics aggregate CPU usage as the sum of each core's percentage of use.

Regarding block size and maximum time for block formation, three experiments were conducted. Each function's throughput and latency information was measured against a block size of 1MB, with 250ms timeout, a 2MB block size with 500ms timeout and 4MB block size and a 1 second timeout configuration.

2. Hyperledger Caliper - https://www.hyperledger.org/projects/caliper
3. Hyperledger Caliper Repository - https://github.com/hyperledger/caliper/tree/c37860b042

	1MB			2MB			4MB		
Send Rate	50 tps	100 tps	200 tps	50 tps	100 tps	200 tps	50 tps	100 tps	200 tps
Create Vehicle	5	6	timeout	6	6	5	6	6	5
Change Ownership	6	7	7	7	7	7	7	8	7
Issue Seizure	6	8	8	7	8	7	8	8	7
Register as Guarantee	6	7	6	7	8	6	7	8	6
Change State	7	7	8	6	8	8	7	8	8

TABLE 1
Throughput (tps) considering the variation of block size.

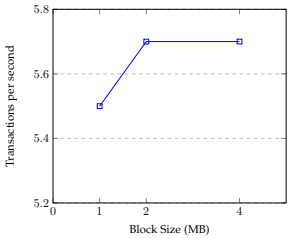

Fig. 3. Throughput of Create Vehicle function

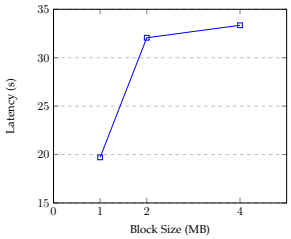

Fig. 4. Latency of Create Vehicle function

6.2 Evaluation

Considering the five functions selected to evaluate the system we averaged the throughput of each function given a certain block size, based on the information of Table 1. Thus, we can conclude on average a block size of 2 MB or a block size of 4 MB can achieve higher throughput that a 1 MB block size. A higher block size is expected to provide an higher throughput to a block chain based system. As we can see by analyzing the throughput and latency graphics for the same function, as in Figures 3 and 4.

The maximum throughput achieved in the tests was around 7.67 transactions per second, when the system was configured to a block size of 4 MB and a maximum timeout of 1 second, when issuing either a *Issue Seizure* or a *Change State* transaction. However this same transactions presented a latency of 23.53 seconds for *Issue Seizure* and 22.42 seconds for *Change State* transaction. Analizing the graphics, it seams the best compromise to optimize both throughput performance and lower latency is achieved using a 2 MB block size.

6.3 System bottlenecks

Considering Table 2, we notice that across all tested block sizes, the latency suffers a major increase when comparing a 100 TPS send rate with the 200 TPS send rate. The overall latency is roughly doubled when the send rate is around 200 TPS, with no major increase in throughput. Analyzing the system statistics for 1 MB block size during a 100 TPS send rate test, as in Table 3, it is clear the ordering peer (orderer.example.com) is not requiring above average CPU or memory usage. Furthermore, the ordering peer presents low memory and CPU usage during the test. Regarding all components used in the test, CouchDB instances reveal to be the most resource hungry containers.

On the other hand, the use of Hyperledger Composer framework might have a significant impact on Hyperledger Fabric overall performance. It is possible that the use of

	1MB			2MB			4MB		
Send Rate	50 tps	100 tps	200 tps	50 tps	100 tps	200 tps	50 tps	100 tps	200 tps
Create Vehicle	13.74	25.68	timeout	12.68	25.62	57.87	11.65	24.15	64.25
Change Ownership	12.28	21.04	40.37	11.43	21.29	42.24	10.05	19.26	47.03
Issue Seizure	12.05	11.09	19.61	11.35	19.82	36.82	9.15	18.9	42.54
Register as Guarantee	39.73	11.75	20.74	9.75	18.66	46.85	10.05	20.63	45.36
Change State	10.79	20.29	35.38	11.56	18.32	36.85	11.18	19.73	36.39

TABLE 2
Latency (seconds) considering the variation of block size

Name	Memory(max)	Memory(avg)	CPU(max)	CPU(avg)
dev-peer0.org2.example.co...0.0.1	124.7MB	116.8MB	90.68%	25.30%
dev-peer0.org1.example.co...0.0.1	128.9MB	120.9MB	82.95%	25.42%
peer0.org2.example.com	77.5MB	65.9MB	96.47%	38.99%
peer0.org1.example.com	74.1MB	62.9MB	91.21%	40.05%
orderer.example.com	23.2MB	19.1MB	22.63%	3.91%
couchdb.org2.example.com	134.4MB	124.7MB	195.93%	82.82%
ca.org1.example.com	7.2MB	7.2MB	0.00%	0.00%
ca.org2.example.com	7.0MB	7.0MB	0.01%	0.00%
couchdb.org1.example.com	144.9MB	127.0MB	211.25%	83.66%

TABLE 3
Register Guarantee function with 100.0 tps send rate (1MB block size)

native Hyperledger Fabric chaincode to develop the car registration system's smart contracts may lead to performance improvements. Considering the complexity of a car registration system, as the BCar system, it is plausible the quantity of information stored in the blockchain and the complexity of available operations might provide a reason for such latency and throughput results. Thus, the implementation of access control rules embedded in the smart contact, as opposed to using solely the Hyperledger Composer's access control mechanisms, improved the latency of such transaction's execution. However, it is still noticeable that the access control mechanisms used in BCar system contribute to lower performance regarding throughput and latency.

7 Conclusion

Through this thesis, we present a use case for blockchain technology in public registry. Blockchain technology as of its decentralized nature can provide a different approach to registry storage. As of this fact, we presented a blockchain based car registration system taking advantage of decentralization capabilities of the technology.

A data model was build considering the operations identified for a car registration system as well as the participants of the system. The set of available operations for the Bcar registration system was then described according to their effects on a car registry.

Considering Bcar system participants, 5 participant types where defined with different permissions over registry operations. A person type participant, a company type participant, a judicial officer participant and a registry employee participant. Finally, an external entity type was presented in order to allow entities, such as tax authorities, to read vehicle registry information.

Regarding vehicle registry's operations we presented two step transaction flows as *Change Ownership* transaction flow, allowing to take advantage of a blockchain based car registration system. Those operations enabled for registry employees to lower their intervention in those transaction flow.

Finally we conducted a set of performance tests over a simple configuration of the Hyperledger Fabric system and conducted an analysis over the collected data. As of the test results we analyzed the throughput and latency results over a set of system's functions varying the block size of the system.

7.1 Future work

As the Bcar registration system was designed, it encompasses most of the operations over car registries. However, the system focuses only on car registry data. A broader system could be built based on the proposed Bcar system, including information regarding vehicle characteristics and legal inspection registries. Considering such a system, blockchain technology could be used to unify the car registry system with the vehicle registry maintained by the Direction for Motorized Vehicles. Furthermore the unified system could be designed as a unique system for all European Union state members.

References

[1] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," *Www.Bitcoin.Org*, p. 9, 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[2] M. Walport, "Distributed ledger technology: Beyond block chain," *Government Office for Science*, pp. 1–88, 2015.

[3] N. Szabo, "The idea of smart contracts," *Nick Szabos Papers and Concise Tutorials*, 1997.

[4] X. Xu, C. Pautasso, L. Zhu, V. Gramoli, A. Ponomarev, A. B. Tran, and S. Chen, "The blockchain as a software connector," *Proceedings - 2016 13th Working IEEE/IFIP Conference on Software Architecture, WICSA 2016*, pp. 182–191, 2016.

[5] M. Castro and B. Liskov, "Practical Byzantine fault tolerance," in *Proceedings of the 3rd USENIX Symposium on Operating Systems Design and Implementation*, Feb. 1999, pp. 173–186.

[6] M. Correia, G. S. Veronese, N. F. Neves, and P. Veríssimo, "Byzantine consensus in asynchronous message-passing systems: a survey," *International Journal of Critical Computer-Based Systems*, vol. 2, no. 2, pp. 141–161, 2011.

[7] G. S. Veronese, M. Correia, A. N. Bessani, L. C. Lung, and P. Verissimo, "Efficient Byzantine fault tolerance," *IEEE Transactions on Computers*, vol. 62, no. 1, pp. 16–30, 2013.

[8] M. Vukolić, "The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9591, pp. 112–125, 2016.

[9] I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8437, pp. 436–454, 2014.

[10] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich, S. Muralidharan, C. Murthy, B. Nguyen, M. Sethi, G. Singh, K. Smith, A. Sorniotti, C. Stathakopoulou, M. Vukolić, S. W. Cocco, and J. Yellick, "Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains," 2018. [Online]. Available: http://arxiv.org/abs/1801.10228

[11] M. Vukolić, "Rethinking Permissioned Blockchains," *Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts - BCC '17*, pp. 3–7, 2017. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3055518.3055526

[12] "e-health records." [Online]. Available: https://e-estonia.com/solutions/healthcare/e-health-record

[13] "e-residency." [Online]. Available: https://e-resident.gov.ee/

[14] "Documento nico automvel," Oct 2005. [Online]. Available: http://www.irn.mj.pt/sections/irn/a$_r egistral/servicos - externos - docs/impressos/automovel/requerimento - de - registo/downloadFile/file/DUA_m odelo_u nico.pdf?nocache =$ 1217232046.45