A Collaborative Enforcement Mechanism for Spectrum Sharing Using Blockchain and Smart Contracts: An application for the 1695-1710MHz band

Pedro J. Bustamante
School of Computing and Information
University of Pittsburgh
Pittsburgh, PA
Email: pjb63@pitt.edu

Marcela M. Gomez
School of Computing and Information
University of Pittsburgh
Pittsburgh
Email: mmg62@pitt.edu

Martin Weiss
School of Computing and Information
University of Pittsburgh
Pittsburgh, PA
Email: mbw@pitt.edu

Taieb Znati
School of Computing and Information
University of Pittsburgh
Pittsburgh, PA
Email: znati@pitt.edu

Debarun Das
School of Computing and Information
University of Pittsburgh
Pittsburgh, PA
Email: ded59@pitt.edu

J. Stephanie Rose
School of Computing and Information
University of Pittsburgh
Pittsburgh, PA
Email: jsr67@pitt.edu

***Abstract*—Traditionally, spectrum allocation has been governed by centralized schemes (e.g., command-and-control). Nonetheless, other mechanisms, such as collaborative enforcement, have proven to be successful in a variety of scenarios. In Collaborative enforcement (i.e., collective action), the stakeholders agree on decision-making arrangements (i.e., access, allocation, and control of the resources) while being involved in monitoring the adherence to the rules as a shared effort. Blockchain is a distributed ledger of records/transactions (i.e., database) that brings many benefits such as decentralization, transparency, immutability, etc. One of the most notable characteristics of blockchain-based platforms is their definition as *trustless* environments, as there is no central entity in charge of controlling the network interactions. Instead, trust is a group effort, achieved through repeated interactions, consensus algorithms, and cryptographic tools; therefore, converting blockchain systems into prominent examples of collaborative governance regimes. In this paper, our goal is to analyze a particular application of blockchain and smart contracts for the 1695-1710MHz sharing scenario. In this way, we provide a theoretical analysis of the feasibility and the required characteristics to implement such a system. In addition, through the implementation of a Proof of Concept, we evaluate how the implementation of a blockchain-based organization can be the motor to build a collaborative governance scheme in the spectrum sharing arrangement of the 1695-1710MHz band**

I. Introduction

One of the main challenges as radio spectrum sharing matures is to find adequate governance systems and the appropriate enforcement mechanisms. Historically, the *de-facto* approach was to assign these tasks to a central or third-party entity (e.g., the Federal Communications Commission (FCC)) [1]. In particular, the Common Pool Resource (CPR) (e.g., the exploitation of spectrum bands[1]) literature finds that other governance mechanisms are possible, including collaborative governance [4].

Collaborative governance or collective action is characterized by the fact that all stakeholders in the system agree on decision-making arrangements (e.g., access, allocation, and control of the available resources). Furthermore, a key aspect in collaborative governance is that all or the majority of participants are involved in monitoring the adherence to the rules in a group effort.

It is almost inevitable not to associate blockchain and cryptocurrencies, because the biggest application of blockchain is, in fact, cryptocurrencies [5]. Nonetheless, there are a considerable number of applications being deployed and used on top of the blockchain [6]. We can find all types of financial applications, examples of the use of blockchain for the internet of things, applications designed to build completely decentralized organizations, etc. [7]. This explosion of blockchain platforms and applications has its root in many of the key characteristics of the solution. In particular, its decentralization (i.e., no single entity has full control over the network). Indeed, many core functionalities of the system (i.e., the consensus mechanism) are a group effort among all or the majority of

[1]Authors in [2], [3] found that the exploitation of spectrum bands can, indeed, be classified as a Common Pool good (CPR).

participants in the network [8].

It is undeniable that the blockchain and many of its central characteristics have the potential to revolutionize many current applications, such as the exchange of monetary units at a distance [9]. Nevertheless, it is when solutions such as Smart Contracts are developed, where blockchain platforms can reach new horizons, usually known as the evolution of blockchain or Blockchain 2.0 [10]. These automated scripts running on top of the blockchain can improve the way in which users interact with the blockchain while providing a more efficient manner to manage and access the information (i.e., digital tokenized assets) stored on the blockchain [11].

In this work, we explore how the implementation of a blockchain-based organization can be the bridge to create a successful collaborative governance regime in the spectrum sharing arrangement of the 1695-1710MHz band. For this purpose, this work is organized as follows: In Section II, we study the different concepts behind collaborative governance, the characteristics of the 1695-1710MHz band, and an overview of blockchain and smart contracts. In Section III, we analyze the theoretical features of our proposed blockchain-based system. In Section IV, we provide the details of the Smart Contracts to be implemented as part of our solution. Finally, in Section V, we present the results of the small scale implementation (i.e., Proof of Concept) of our proposed blockchain system.

II. Background

A. Collaborative Governance

Our main goal with this work is to find suitable governance mechanisms for the emergence of sharing schemes in the allocation of radioelectric spectrum bands. It is necessary to point out that in the particular case of Common Pool Resources (CPRs), the literature situates enforcement as part of the governance structure and incorporates it into the definition of rules [12]. In this manner, for this paper, we assume that a broad definition of the main characteristics of governance encloses the required definitions for the enforcement situations.

Before talking about collaborative governance, let us first define governance in the context of this work. In general terms, *governance* refers to the act of governing, be it in the public and/or private sector [13]. In the particular context of collective action, governance is placed as a *"dimension of jointly determined norms and rules designed to regulate individual and group behavior"* [14]. This definition includes the commonly accepted interpretation of collective action. However, for this work, we also consider it important to include concepts such as that the*"collaborative governance includes the set of coordinating and monitoring activities that enable the survival of the collaborative partnership"* in the definition of collective governance [15].

The definition of collaborative governance that we will use throughout this work comes from the work by Ansell and Gash [16]: *"A governing arrangement where one or more public or private agencies directly engage non-state stakeholders in a collective decision-making process that is formal, consensus-oriented, and deliberative and that aims to make or implement public policy or manage public programs or assets."*. This definition includes some key aspects worth mentioning. First, it is not limited to public agencies (i.e., state actors). Second, the participants in the collective process are always part of the decision-making process. Third, the members are formally organized and have continuous communication. Finally, all decisions are the product of consensus mechanisms.

We believe that the *Integrative Framework for Collaborative Governance (IFCG)* by Emerson et al. [13] is the most appropriate tool to analyze and evaluate our proposed governance mechanism. The framework has three nested dimensions: the general system context, the Collaborative Governance Regime (CGR), and its collaborative dynamics & actions (See Fig. 1). First, the context of the system (outermost box) represents the context surrounding the system in terms of the host of political, legal, socioeconomic, environmental, and other influences that affect and are affected by the CGR. Moreover, this context causes the emergence of drivers, including leadership, consequential incentives, interdependence, and uncertainty, which help initiate and set the direction of the CGR. The Collaborative Governance Regime (CGR) (middle box) contains the collaborative dynamics and collaborative actions. These components together are the core of the model since they shape the quality and extent to which the overall collaborative governance is developed and effective. Finally, the collaborative dynamics (inner box) is composed of three interactive components: principled engagement, shared motivation, and capacity for joint action. The three components together produce collaborative actions or the steps taken in order to implement the shared purpose of the CGR. For this paper, we are focused entirely on the Collaboration Dynamics (CD) and Collaboration Governance Regime (CGR) and how the resulting actions can *glue* together the whole process of collective action[2].

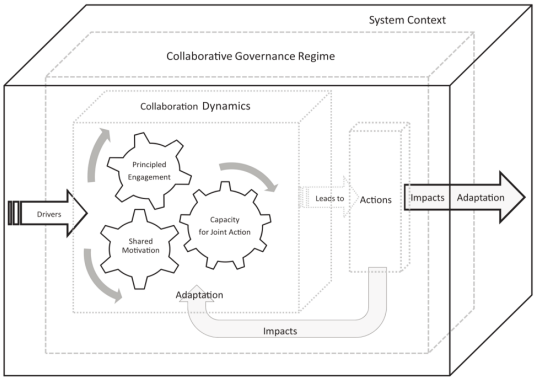

Fig. 1: Integrative Framework for Collaborative Governance

B. The 1695-1710MHz Band

We have chosen the spectrum sharing framework of the 1695-1710MHz band. We selected this band due to its sim-

[2]The analysis of the System Context of the IFCG (e.g., political influence) lies outside the scope of this paper.

plicity and the advantages of working with an existing, widely known, and well-defined scheme.

The 1695-1710MHz band is part of the Advanced Wireless Services (AWS-3) defined by the Federal Communications Commission (FCC) and the National Telecommunications and Information Agency (NTIA) [17]. The original incumbent or Primary User (PU) are the Meteorological Satellites of the National Oceanic and Atmospheric Administration (NOAA). This PU utilizes the band for space-to-earth (i.e., downlink) operations. On the other hand, the new entrants or Secondary Users (SU) are mobile LTE handsets (MS), which are restricted to uplink transmissions (i.e., ms-to-base-station). A third participant in the band are the corresponding base stations (i.e., eNodeB) serving each MS. Nonetheless, the eNodeBs do not have transmission rights in the scheme. They serve as the coordination point between the PU and SU [18].

Another key characteristic of the 1695-1710MHz band is its definition of *restricted zones*. First, an Exclusion Zone (EZ) located around the Primary Users, where the SUs are not allowed to transmit. Second, a Coordination Zone (CZ). This new area extends beyond the border of the EZ [2]. Its boundary is defined based on several factors, including the transmission power, antenna gains in the direction of interference, time variations of antenna gains, receiver susceptibility to interference, propagation effects of radio waves, mobility of each station, etc. Transmission privileges in the CZ are granted to the new entrants if, and only if, the proposed transmission will not contribute to the aggregate interference at the PU location in such manner that it will cause harmful interference [19].

Based on the concepts of the exclusion and coordination zones, many authors have developed multiple approaches to specify the boundaries (i.e., size) of these restricted areas [19]–[21]. These approaches seek a more flexible definition that the one suggested by the FCC/NTIA. In this paper, we utilize the notation introduced by Bhattarai et al. in their definitions of *The Multi-Tiered Incumbent Protection Zones (MIPZ)* [19]. This framework for geolocation-database-driven spectrum sharing gives the option to the PU to adjust the size (i.e., boundary) of the restricted zones *on the fly* based on the instantaneous interference conditions. As a result, three types of zones are defined around the PU (See Fig. 2):

- **No Access Zone (NAZ):** The spatial area surrounding the immediate vicinity of the PU, where transmission privileges are limited only to licensed incumbents.
- **Limited Access Zone (LAZ):** The spatial area encompassing the NAZ. In this region, a limited number of new entrants are allowed to transmit simultaneously. The number of SU allowed to transmit is computed using a specific propagation model, such that transmissions outside the LAZ cause negligible harmful interference.
- **Unlimited Access Zone (UAZ):** The region that lies outside the LAZ. In this zone, unlimited transmission privileges are granted to the new entrants, since they do not represent any *threat* to the PU's normal operations.

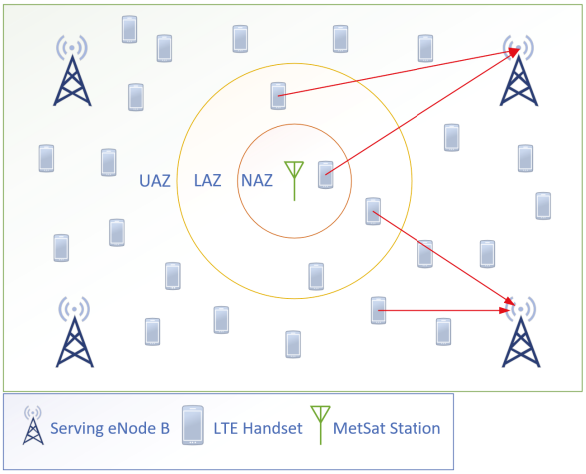

Fig. 2: MIPZ environment layout

C. Overview of Blockchain

1) Architecture: Blockchain implements a shared, replicated, and distributed ledger[3]. This allows to create a peer-to-peer (P2P) system, where users transfer (i.e., transact) digitized, tokenized, and valuable assets (e.g., cryptocurrencies) at a distance with no need of a central trusted third party (e.g., the post office). For this purpose, every node in the network has a complete copy of the ledger, while the network guarantees that every user's copy reflects the current state of the underlying data (i.e., *the state of the world*) [11].

This implemented ledger can be seen as a secure form of a *database* of records (i.e., transactions), with features that include decentralization (any transaction is conducted in a P2P manner), persistency (all transactions are broadcasted), auditability (transactions are verifiable and traceable), immutability (committed entries cannot be changed or deleted), and security (all entries are cryptographically secured) [6].

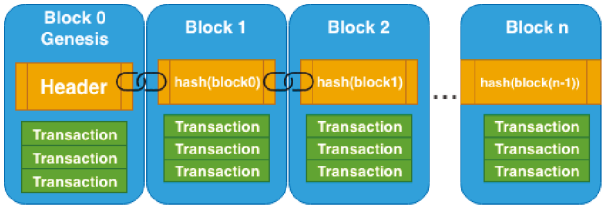

Fig. 3: Blockchain Architecture

The ledger is constructed as a log of records (i.e., transactions) that are batched into time-stamped blocks identified by its cryptographic hash, creating in this manner a unique identifier for each block. Each block also contains a pointer to the hash of the previous block (i.e., parent block), creating in this manner a chain of blocks, commonly known as the Blockchain (see Fig. 3) [6]. The first block of the chain, known

[3]A book for recording and totaling transactions by account type and a beginning balance & ending balance for each account [22].

TABLE I: Taxonomy of Blockchain Platforms

	Public	Consortium	Private
Access	Permission-less	Permissioned but more flexible	Permissioned
Read privileges	Open (everyone)	Usually restricted	Usually restricted
Immutability Level	High	Medium	Medium
Efficiency	Low: High overhead from consensus	High: Low overhead from consensus	High: Low overhead from consensus
Centralization	No	Partial	Yes
Consensus Algorithm	Usually PoW and/or PoS	Pre-approved by nodes	Pre-approved by nodes
User's Identity	Anonymous / Pseudonymous	Anonymous / Known Identities	Known identities
Asset	Platform-Native	Any	Any
Platforms Examples	Bitcoin & Ethereum	Ripple & Multichain	Hyperledger Fabric

as *Genesis*, has no parent block and it is common to the whole network.

2) The Blockchain Block: The two main components in a blockchain implementation are: the transactions (i.e., the log of records in the ledger) and the blocks containing these transactions. In the case of the later, it is further divided into the block header and body (see Fig. 4). The body contains the transactions and a transaction counter. On the other hand, the header contains several details regarding the creation of the block. This includes a timestamp (exact date and time of the block creation), a nonce (arbitrary number to guarantee that the transactions are handled only once), the hash of the parent block (to construct the link between blocks), and the Merkle tree root (the hash or fingerprint of the block[4]) [24].

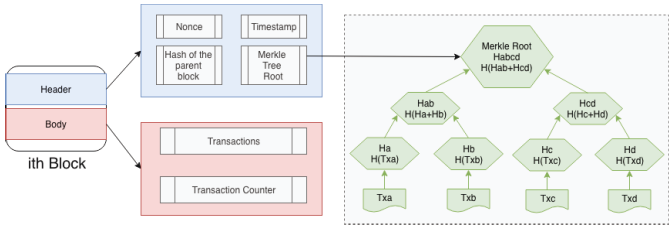

Fig. 4: Blockchain Block

3) Taxonomy of Blockchain Platforms: In the literature, there are many ways to categorize blockchain-based organizations. For instance, platforms are classified by the main objective of the system (i.e., exchange of cryptocurrencies), the type of computations (e.g., support for smart contracts), etc. [7]. Nonetheless, the most common way to categorize a blockchain platform is based on its access rights configuration [6], [25], [26]. In this light, blockchain-based systems are classified into public, consortium, and private. In a public or permission-less blockchain, such as Bitcoin or Ethereum, any node (usually anonymously or using pseudonyms) can access the network. On the other hand, in private or permissioned platforms, only a limited number of users (usually with known identities) can access the system. Further, older users exercise access control to new entrants. Finally, in consortium or *hybrid*[5] blockchains, instead of allowing any user to participate in the network or authorizing a single node/company to have full access control, a few selected nodes perform the most important functions in the system, including access control (see Table I) [27].

4) Consensus Algorithms: One of the most prominent characteristics of most blockchain platforms is that they do not rely on a trusted centralized entity (e.g., a bank). Consequently, it is critical for the system to have a method for agreeing on the validity of the entries (i.e., transactions) in the database and the order in which they are attached to the chain (i.e., batched into blocks). Otherwise, the nodes in the network would have a different view of the *state of the world* [28]. To this end, most blockchain-based platforms implement a distributed consensus mechanism (i.e., algorithm). Reaching a consensus in the Blockchain is an adaptation of the Byzantine Generals (BG) problem[6]. The implemented consensus algorithm in blockchain-based platforms is supported on factors such as the type of system (e.g., public), the network configuration (e.g., known node identities), and the type of asset being exchanged (e.g., cryptocurrency) [7], [25]. The undeniable popularity of cryptocurrencies[7] and the significant amount of applications being developed on top of blockchain platforms, have led to the development of a substantial number of consensus algorithms for the blockchain (see Fig. 5). In what follows, we briefly go over the most popular consensus algorithms in the blockchain and its main characteristics.

Proof of Work (PoW): The most widely known mechanism due to its utilization in Bitcoin. The goal of the algorithm is to verify the validity of a new block for it to be appended at the end of the chain. For this purpose, nodes (*miners* in Bitcoin) compete to solve some complicated mathematical puzzle[8] (*mining* in Bitcoin). The first node to solve the puzzle appends the new block at the end at the chain[9] [31].

Proof of Stake (PoS): An energy-efficient alternative for PoW. The basic assumption in PoS is that users with higher stakes (e.g., ownership of digital assets) in the system are less likely to harm (i.e., attack) the network. Thus, users with higher stakes, have greater probabilities of verifying a new

[4]It is the result of repeatedly digesting pairs of transactions until there is one root hash [23].

[5]A hybrid version between private and public blockchains.

[6]A group of generals of the Byzantine army have circled an enemy camp. Nevertheless, for an attack to be successful, the majority of generals must agree on whether, how, and when to attack. However, one or more generals could be *traitors* trying to boycott the plan [29].

[7]As for June 2019, the total number of cryptocurrencies was 2,212 [30].

[8]In Bitcoin, nodes try to find a hash equal or less than a given target.

[9]In some implementations of blockchain, such as Bitcoin, the node receives a reward (e.g., Bitcoins) for this process.

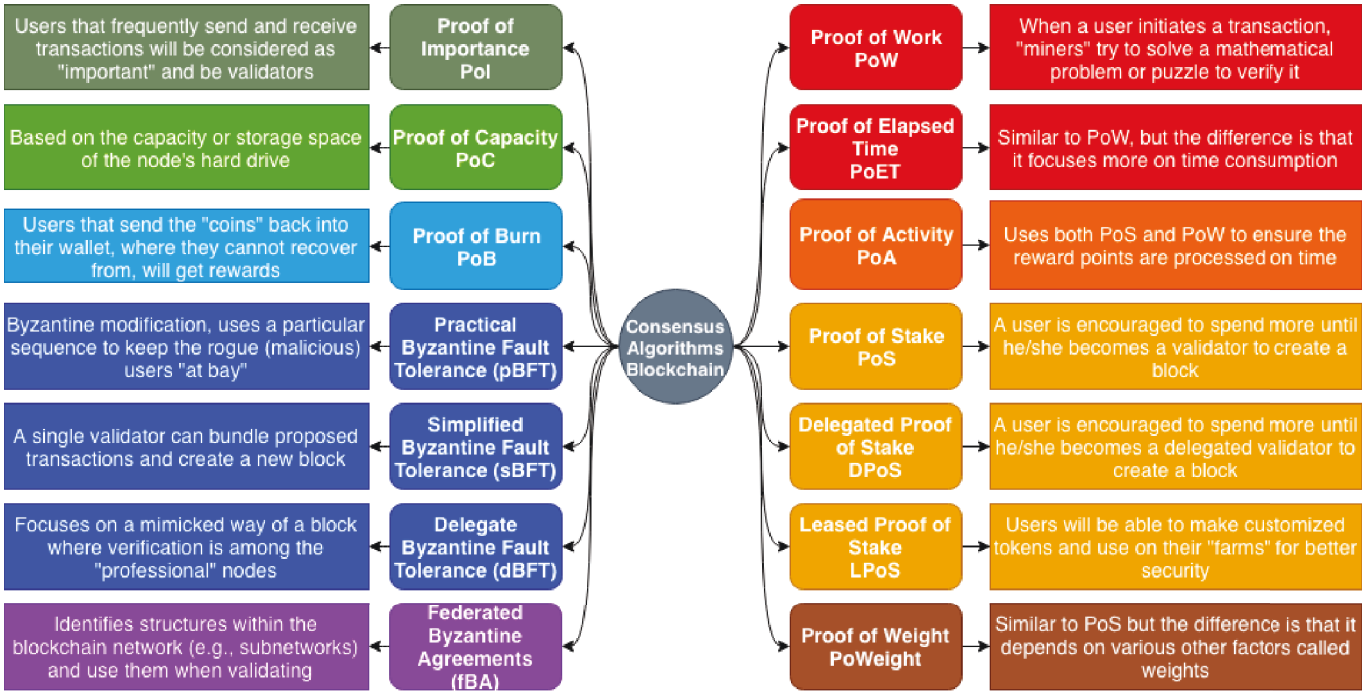

Fig. 5: Consensus Algorithms in the Blockchain

block in a pseudo-random process to choose a validator [27].

Many variations of PoS have been proposed such as Delegated Proof of Stake (DPoS), Proof of Weight (PoWeight), and Leased Proof of Stake (LPoS) [32]. Further, many algorithms have been developed as hybrid versions of PoS and PoW. This includes Proof of Importance (PoI), Proof of Capacity (PoC), and Proof of Elapsed Time (PoET). All these new algorithms share the same core characteristics of PoS and PoW with small variations such as election of leaders, indirect democracy, different definitions of stake, different definitions of work, etc. [8].

Practical Byzantine Fault Tolerance Algorithm (pBFT): The most popular mechanism for reaching a consensus within private blockchains, such as Hyperledger Fabric [33]. The process starts with a new *candidate block* being proposed by a leader[10]. The authentication process for the proposed block is divided into three phases: pre-prepare (whether to broadcast a vote for a block), prepare (broadcast the block for commit), and commit (validation of the block). To enter the next phase, the node has to receive votes from 2/3 of all nodes in the network [34].

In the same manner as PoS and PoW, multiple variations of pBFT have been proposed in the literature. Thus, we can find federated, delegated, and simplified schemes of pBFT [32].

D. Overview of Smart Contracts

1) Definition: Defining smart contracts has proven to be complicated. Many definitions of smart contracts can be found in the literature, as we can observe in Fig 6. This situation reflects the many debates on the nature of smart contracts and how a clear definition is a contest between competing terminology. Nonetheless, most concepts used to define smart contracts usually fall into one of two categories: Smart Contract Code or Smart Legal Contracts [35].

Smart Contract Code: Blockchain platforms can run code to execute all the operations of validation, verification, and transmission of blocks and transactions. While initial blockchains, such as Bitcoin, are optimized to perform a small set of simple operations, newer generations of blockchain-based systems can perform more complex operations in full-fledged programming languages [36]. This new set of tools allows registering code in the blockchain in the form of clauses of agreements, which will be later executed when certain conditions are met in an automatic, self-enforced, and self-adjudicated manner. In other words, smart instructions or code will allow any user to build an application to run on a virtual machine (i.e., blockchain node) that is executed by consensus across the entire network, allowing it to modify a globally accessible state as its *database* [11].

Smart Legal Contracts One of the most common manner to classify a legal contract is into *Complete* and *Incomplete*. Complete contracts allow to specify all possible legal consequences for every possible state of the world [37]. Nevertheless, due to the presence of *edge-case* liabilities, most contracts are *incomplete*. Hence, there exist multiple legal mechanisms to deal with this situation. In the particular case of Blockchain, since the data cannot be changed or deleted (i.e., immutable), smart contracts on top of the blockchain are necessarily complete. Consequently, for the time being, smart legal documents would be, most likely, a combination of smart contract code and some

[10]Determined in a round robin, where the primary node (i.e. leader) is responsible for ordering the transactions to be validated.

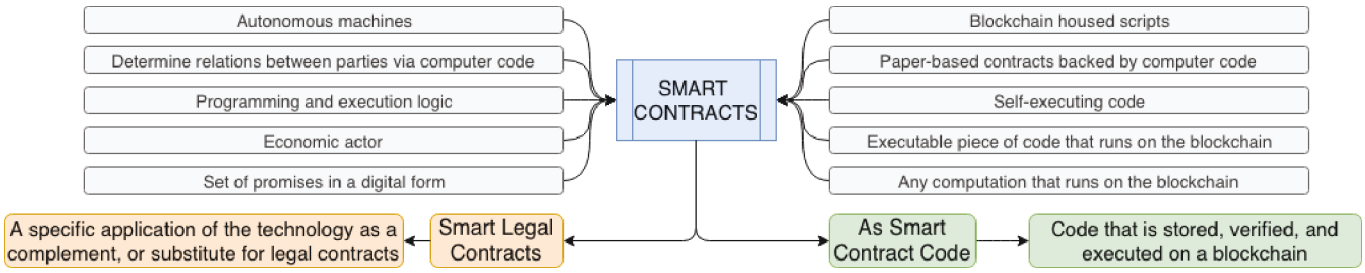

Fig. 6: Smart Contracts and its Multiple Definitions

external traditional legal mechanisms [38].

2) Structure: A smart contract is a software script (i.e., a piece of code) constituted of a series of embedded functions (i.e., instructions) to be run on top of the blockchain. A contract has a unique address. In fact, a contract is invoked by sending a transaction to this address [7]. This can be some form of asset transaction (i.e., transfer assets) or an information exchange with the contract. In addition, a smart contract has its own state (i.e., memory and storage) and value (i.e., current balance) to take custody over assets on the chain (see Fig. 7). Once the process of a smart contract is executed, an output is produced to be approved by the network. In the same manner as the inputs, the outputs of the contracts can be a further transaction or just the update of some information in the network. Moreover, a smart contract can invoke or be invoked by another contract to perform its main function or another task using one of its multiple defined entry points (i.e., functions) of execution [11].

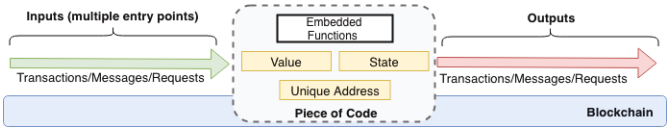

Fig. 7: Smart Contracts Structure

E. Blockchain for Spectrum Management

In [25] Weiss et al. explore the application of the blockchain to radio spectrum management. In particular, how blockchain systems could be implemented in primary/secondary cooperative sharing and in primary/secondary non-cooperative sharing agreements. In our work, we explore the application of blockchain to a secondary cooperative sharing scheme. Thus, the system is capable of enabling spectrum trading markets among multiple participants in the band. For this purpose, the blockchain-based platform needs to enable for different types of spectrum trading transactions. Further, the system could use smart contracts to facilitate this interactions (i.e., transactions). Since the system is based on blockchain, it also leverages all the benefits of this type of organization. Benefits such as auditable, traceable, and unified record of interactions. In addition, a secondary cooperative sharing framework could be constructed on top of both a public and a private blockchain-based platform. Nevertheless, a deeper analysis is needed to consider factors such as the latency in blockchain to develop a real-time application or the access of the regulator to the blockchain to guarantee fair market operations and efficient use of spectrum resources [25].

III. Blockchain-based System for Spectrum Sharing Applications

In this section, we provide a theoretical analysis of different alternatives to exploit the benefits of blockchain in spectrum sharing scenarios. In particular, we aim to design and build a blockchain-based secondary cooperative spectrum sharing framework for the 1695-1710MHz band. This section covers the definition of the asset to be exchanged, the type of blockchain to be used, the most suitable consensus algorithm, and the type of transactions to take place in the system.

A. The underlying data

The first step when creating a blockchain solution for an organization is to define the underlying data that will be stored in the chain. In other words, the tokenized, digital, and valuable asset to be exchanged.

This requirement can be analyzed from many perspectives. First, we can use blockchain-based systems already in place. To start, we can create a blockchain used exclusively for payments for spectrum units. This system is really similar to the more than 2,000 cryptocurrencies platforms currently available[11] [30]. The biggest difference is that it is exclusively used by the participants in the sharing framework. In this manner, the PU receives some monetary units (e.g., Bitcoins) in exchange of access rights to the restricted zones (i.e., LAZ and NAZ) around its transmitter.

Another system that can be used is a notary service build on top of the blockchain. Hence, the main assumption is that access rights could be converted into property titles to be stored on the chain [39]. The agents can show *proof-of-existence* and *proof-of-ownership* of the different property titles representing transmission access rights. In addition, the created titles can be exchanged between the agents as a *transfer-of-ownership* (See Table III).

The final option is significantly different from any other currently available solution. It consists in *tokenizing* the spectrum band to be further stored on the blockchain.

[11] As of June 2019.

One approach could be to *virtualize* the shared spectrum. The new entrants of the band are LTE handsets using the band for uplink communications. LTE was originated as a standard to allow more flexible spectrum allocation. In the particular case of LTE-Advanced (LTE-A), a key characteristic is the capability to perform carrier aggregation in order to achieve wider bandwidths [40]. Thus, the Physical Resource Block (PRB) is the basic element for radio resource allocation. A PRB is a set of resource elements, which are time-frequency resource-units. These elements can be defined as one subcarrier over one OFDM symbol. In total, 12 OFDM subcarriers, contiguous in frequency, over one slot in time will form a PRB. Summing up, the time-frequency region that encloses a PRB corresponds to a 0.5 millisecond-time slot and 180 kHz in the frequency domain [41].

The minimum size of radio resource that can be allocated is the minimum Time Transmission Interval (TTI) in the time domain, which corresponds to one subframe of 1 millisecond. Subframes can be further grouped into frames of 10 milliseconds length with specific arrangements of the PRBs. The number of allocated PRBs will contribute to the bandwidth a specific user can count on for a given transmission; nevertheless, the actual number of PRBs is determined by the standard (see Table II) [42].

Resource Blocks	6	15	25	50	75	100
Transmission Bandwidth (MHz)	1.4	3	5	10	15	20
Occupied Bandwidth (MHz)	1.1	2.7	4.5	9.0	13.5	18.0
Guardband (MHz)	0.32	0.3	0.5	1.0	1.5	2.0

TABLE II: LTE Parameters (Resource Aggregation)

As shown by Gomez et al. in [43], PRBs could be a rough proxy for a virtualized (i.e., tokenized) resource unit that could be traded in order to improve the liquidity in secondary markets of spectrum[12]. In this approach, the number of PRB that are available in the market could be given by the PU. SUs can use these resources placed on a *pool*[13] according to what the LTE-A standard dictates [43].

The underlying data to be stored on the blockchain would be a pool of PRBs that can be transferred between the incumbent and the new entrants in the band. In the rest of this paper, we will use this *tokenized* and *virtualized* spectrum as the underlying data of our proposed system.

	Monetary Units	**Property Titles**	**Tokenized Spectrum**
Underlying Data	Payment blockchain for the exchange of transmission access rights	Notary blockchain for the transfer of transmission access rights	Blockchain storing LTE PRBs for the exchange between PU and SU.

TABLE III: Underlying Data for the Blockchain Platform

[12]Following [43] we do not assert the technical feasibility of doing so and only use PRBs as suggestive of a possible technical approach.

[13]Pool of resources: a certain number of PRBs, which in turn correspond to specific values of bandwidth that can be translated into capacity rates.

B. The Type of Blockchain

Blockchain-based platforms are usually classified into three categories: public, private, or consortium. For the 1695-1710MHz band, any of these blockchain classes could be utilized (see Table IV).

For a public blockchain, any user with the required capabilities could enter the network and have access to both the stored data and potentially to request a spectrum token, where no authentication or known identity are required to enter the system. A public configuration can be seen as an incentivizing scheme that encourages new participants to join the network. Further, a public blockchain is considered the only truly decentralized, democratized, and authority-free platform [44]. The system also comes with a few disadvantages. The most significant one is the high overhead from having a consensus algorithm robust enough to stop *bad* behaviors among the users. Other issues with a public blockchain could include the lack of complete privacy and anonymity, faulty nodes involved in malicious activities, limitations in the type of asset, and a considerable introduced latency due to platform rules [44].

The second approach would be to create a completely private blockchain. This seems like a more appropriate solution for our sharing scheme because the identities of all the participants are already well-known and well-defined. In this scenario, an entity should be in charge of the access control to the network. The most obvious candidates for this function would be the incumbent of the band and/or a SAS-like entity[14] This approach has many benefits, including having low overhead from the consensus and an asset pre-approved by the nodes. The biggest problem with such a system is the introduced centralization: only the PU or other third-party entity is in charge of admitting new users into the network [25]. This final consideration could be a concern from the perspective of a fully collaborative governance regime.

Finally, the system could be constructed as a consortium blockchain. In this manner, we still could leverage the known-identities of the participants, while reducing centralization. For this purpose, instead of only having the PU as the central authority granting access to the network, the PU in coordination with the eNodeBs (i.e., base stations) could be in charge of the main functions of the system, including access control. In the particular case of the 1695-1710MHz band, this appears as the more suitable solution. The configuration would allow for an native-asset to be stored in the chain, a wider selection of consensus algorithms, known identities, and reduced centralization.

C. The Type of Consensus Algorithm

Due to the popularity of blockchain and its applications, there exists a significant number of consensus algorithms being deployed for the Blockchain (see Section II-C4). All these algorithms have different computation, time, organizational, and energy constraints.

[14]Spectrum Access System (SAS) of the Citizens Broadband Radio Services (CBRS) in the 3.5GHz sharing scheme [25], [45].

Type of Blockchain			
	Public	**Private**	**Consortium**
Application	Any user can access the system	The PU controls the access to the system	The PU in coordination with the eNodeBs execute the main functions
Advantages	Greater incentives to join the network	Access control by known identities	Higher decentralization through PU and SU coordination
Disadvantages	Privacy concerns & high overheads	Centralized system	Requires continuous coordination between PU and SU

TABLE IV: The Type of Blockchain

In the case of the 1695-1710MHz band, the mobile stations have limited resources (computation, battery, communication, etc.). Tasks such as validation of blocks and transactions demand a considerable amount of resources for full nodes in the network. Consequently, the combination of resource-intensive consensus algorithms such as Proof of Work, Proof of Capacity, and Proof of Activity and mobile stations acting as full nodes[15] are, by any means, a sub-optimal solution for our system [46]. In addition, algorithms based on the user's stake (e.g., Proof of Stake) could also result in serious issues for the system. The sharing scheme is defined in a manner in which the PU has control over the majority of resources (i.e., the pool of resources). This is translated in unbalanced probabilities for validation of transactions and new blocks [46]. Nevertheless, it is necessary to point out that in the case of a public blockchain, the participants of the band could only be users instead of full nodes. They would not be involved in tasks such as the validation and verification of transmissions and blocks (e.g., mining). Consequently, their resource-limitations could be neglected, since other nodes would be performing the most time and resource consuming duties. In this case, the only limitation comes from platforms features. For instance, in Ethereum, on average, a new block is appended to the chain every 16 seconds [47], while in Bitcoin, on average, a block takes 10 minutes to be attached to the blockchain [48]. This means that the transactions contained in these blocks are not validated until the block is a part of the chain, which implies a significant latency, especially for *real-time* applications.

To overcome the challenges (e.g., high energy consumption) posed by the introduction of *Proof-based* consensus algorithms in public blockchains, many private and consortium blockchains have opted to adapt and use *old* mechanisms, particularly the Practical Byzantine Fault Tolerance Algorithm (pBFT) [49]. The pBFT model focuses on providing a practical Byzantine state machine replication that tolerates Byzantine faults (i.e., malicious nodes). The algorithm is designed to work in asynchronous systems [50], which converts it in the most suitable solution for our blockchain-based scheme.

The algorithm divides users into two categories: clients and full nodes. For our scheme, we could assume that the clients are all the new entrants in the system and the full nodes, with higher resources, are the PU, the eNodeBs, and/or a SAS-like entity. In this manner, a client sends a request (e.g., validate a set of transactions) to a leader node to invoke a service operation. The leader node multicasts the request to the backup nodes. The nodes execute the request and then send a reply to the client. The client awaits $f+1$ (f represents the maximum number of nodes that may be faulty) replies from different nodes with the same result. This result is the result of the operation. The final result is that all honest nodes come to an agreement on the order of the record and they either accept it or reject it (see Fig. 8) [50].

[15]Nodes involved in all the activities of the blockchain. For instance, validation (i.e., mining in Bitcoin) of new blocks to be appended to the chain.

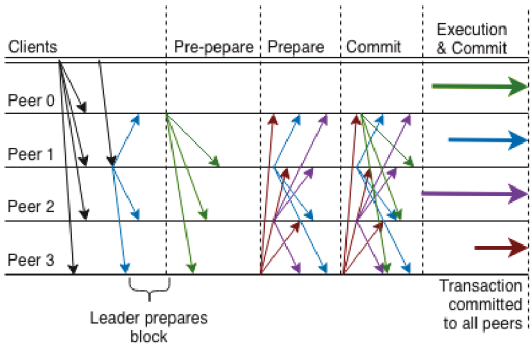

Fig. 8: Practical Byzantine Fault Tolerance Algorithm

D. Type of Transactions

In this section, we explore how a transaction would take place in our proposed system. This includes the creation of tokes (e.g., PRBs) by the primary user, how tokens are transferred, and how tokens are utilized.

1) Tokens: In the blockchain, tokens usually either have a fixed supply or follow a transparent supply schedule, making them anti-inflationary [51]. There are two major types of tokens: currencies and native-tokens. A currency is usually native to a blockchain, such as bitcoin in Bitcoin and Ether in Ethereum. Unlike a currency, a token is not native to a blockchain, but is created on top of it and it is, usually, governed and accessed by a smart contract [52]. In our example, a token could represent the right to utilize an available PRB for a given time. Although some tokens might indicate access to the same physical resources, their time feature converts them into unique and unrepeatable tokens.

2) Creating the Tokens: The primary user is in charge of creating a pool of resources. In other words, the incumbent determines the available resources in a period of time. These resources could be expressed in the number of available PRBs for the Limited Access and Unlimited Access Zone. Once the resources have been created, they can be transferred to the SU as a normal transaction in Blockchain. In addition, the created tokens can be transferred to a smart contract for their administration (see Section IV).

3) Transferring the Tokens: A new entrant to the band needs to acquire a given number of tokens (e.g., PRBs) to transmit in the band. Hence, a SU translates its capacity needs into, for instance, required PRBs. These tokens would be transferred by the PU to the SU. Additionally, a SU can obtain the required tokens by using a smart contract (see Section IV).

4) Using the Tokens: It is necessary to remember that every token is unique and immutable. Therefore, when a Secondary User has acquired a number of tokens (e.g., PRBs), the user has the corresponding transmission rights to use them as needed. For this purpose, a SU must transfer the utilized tokens to an *eater account*, where they are no longer accessible. This idea comes from the concept of Proof of Burn (PoB) or consensus through coin destruction, where tokens are *destroyed* or *burned* by sending them to an *unspendable* address, known as an *eater* address[16] [9] (see Fig. 9).

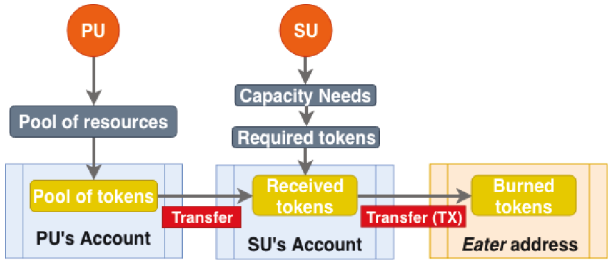

Fig. 9: Type of Transactions

E. Enforcement Characteristics

All the transactions are validated and verified through a consensus algorithm in the network. In other words, the blockchain makes sure that no tokens are transferred twice and that a user with insufficient funds cannot transfer them. In this manner, the blockchain itself provides the required enforcement mechanism to avoid unauthorized transactions (i.e., unapproved transmissions in restricted zones). Therefore, a SU can only receive the created tokens (e.g., PRBs) from the PU and cannot transfer (i.e., transmit) more than the ones received. Due to the immutability and traceability characteristics of the blockchain, all transactions are always correctly registered, which allows for any future enforcement process.

IV. Smart Contracts in a Blockchain-based Spectrum Sharing Scenario

Smart contracts are a key asset in most blockchain-based platforms. Once the information is stored on a blockchain, it can be acted upon through smart contracts [25]. In the case of our scheme, smart contracts could be beneficial in the creation, transference, and usage of tokens (e.g., PRBs). Further, smart contracts can be a crucial resource in enforcement tasks [54].

A. Creating and Transferring Tokens Contract

The first smart contract to be created in our system, *createTransferSpectrumTokens*, has two main functionalities: store the pool of resources, and administer (i.e., transfer) these tokens for their utilization by new entrants (see Fig. 10).

Register Secondary Users: The first function of the *createTransferSpectrumTokens* contract is to register the secondary users. In this manner, only registered participants can check

[16]Address with no private key: while anybody can view the coins and transactions at that address, nobody can access it to unlock the funds [53].

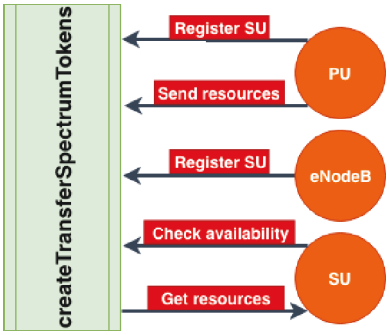

Fig. 10: Create and Transfer Spectrum Tokens Contract

and receive the available resources on the pool. It is necessary to point out that the smart contract is written in a way in which only the PU or the corresponding eNodeBs can register new users in the system (see Fig. 12).

Register Pool of Resources: The incumbent of the band transfers the pool of resources[17] for their future administration (i.e., transference) by the smart contract. In the same manner as before, the contract is written in a manner where only the PU can transfer these resources (see Fig. 12).

Check resource availability: A SU can check back with the smart contract the available resources in the pool. Thus, the new entrant sends a message to the smart contract, which replies with a list of the available resources. There are no restrictions on the participants who can call this function.

Ask for resources: A SU can ask the smart contract to transfer a given number of tokens. If the request complies with the necessary requirements[18], the smart contract transfer the tokens (e.g., PRBs) for their utilization by the SU (see Fig. 12).

B. Using the Tokens Contract

The goal of the second contract is to include an additional layer between the SU and the *eater* address. Instead of directly transferring a token to be *burned*, the SU transfers it to a contract, *useBurnSpectrumTokens*, which forwards them to the *eater* account (see Fig. 11).

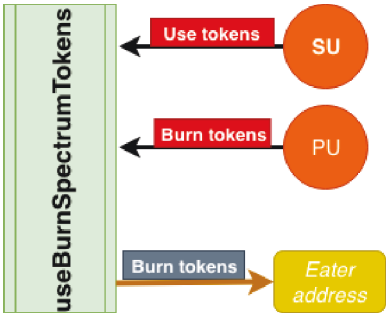

Fig. 11: Use and Burn Spectrum Tokens Contract

[17]Send a list of available resources in a given time frame.

[18]The requirements include: Is the SU registered? The resources agree with the block aggregation in LTE?, etc.

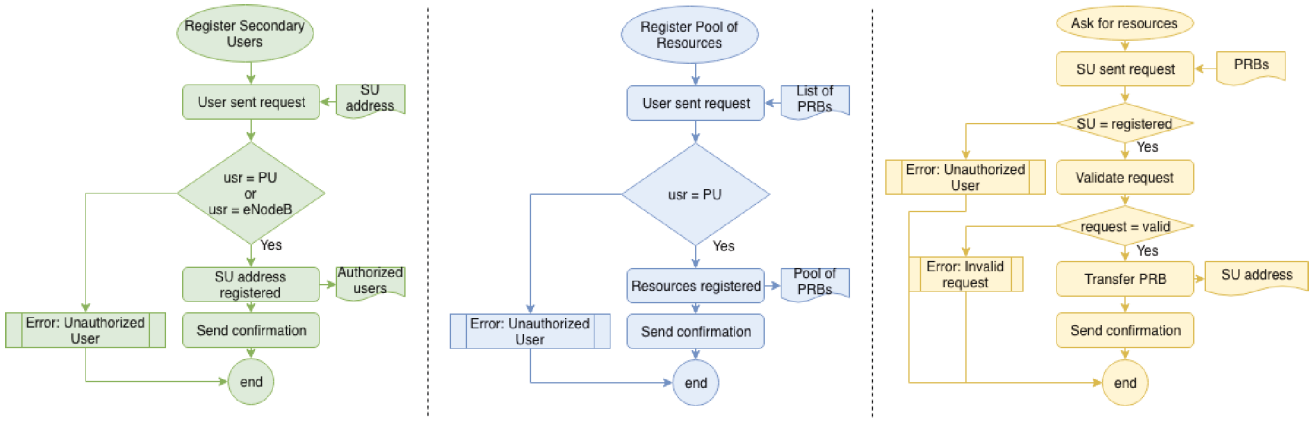

Fig. 12: Create and Transfer Spectrum Tokens Contract: Main Functionalities

Use the tokens: The first function of the *useBurnSpectrumTokens* contract is in charge of receiving the used tokens from the different secondary users. Hence, when a new entrant invokes this function, it transfers the adjudicated and utilized tokens to the contact's own account. It is necessary to point out that there is no limit in the number of tokens that the contract can hold before they are sent for *destruction* (see Fig. 14).

Burn the tokens: This smart contract also includes a functionality where multiple tokens (e.g., PRBs) can be sent for destruction at once. In other words, the smart contract can send all the tokens it has received to the *eater* address for them to be *burned*. The function is written in a manner where only the primary user can call this function (see Fig. 14).

C. Enforcement

The inclusion of smart contracts in the scheme allows for additional layers of validation, verification, and transparency for a more efficient enforcement. First of all, with a smart contract, secondary users can now check what is the availability of resources. In addition, the *createTransferPRB* smart contract keeps an additional registry of not only the pool of resources but also, the adjudication of tokens. On the other hand, the *useBurnSpectrumTokens* smart contract also allows for an additional layer of information by keeping track of the received tokens for destruction. This functionality permits additional controls by the different actors in terms of received and consumed tokens. It is necessary to point out that since the smart contracts are deployed on top of the blockchain, all the transactions from and to the smart contracts are also registered in the blockchain and validated through consensus. In this manner, the initial enforcement provided by the community activities of the blockchain is still present when using smart contracts.

V. Proof of Concept

In this section, we present a Proof of Concept of the different proposed characteristics of our blockchain-based spectrum sharing scheme. The purpose of this small-scale implementation is to verify whether the presented concepts are feasible and have the potential to be transformed into *real-world* applications. We use two well-known platforms. First, an application built on top of a public blockchain, specifically Ethereum. Second, an application developed to be run on a private blockchain under the Hyperledger Fabric environment.

A. Implementation on top of a public blockchain

There are multiple public blockchain platforms available for the development of Decentralized Applications (DAPPs)[19], with Ethereum being the most widely used solution[20]. We develop and test our system in both a local environment and in a working *Ethereum Test-net*.

1) Local Environment: First, we verify our system in a local Ethereum environment: A small implementation of nodes and accounts running in a single computer (see Fig. 13).

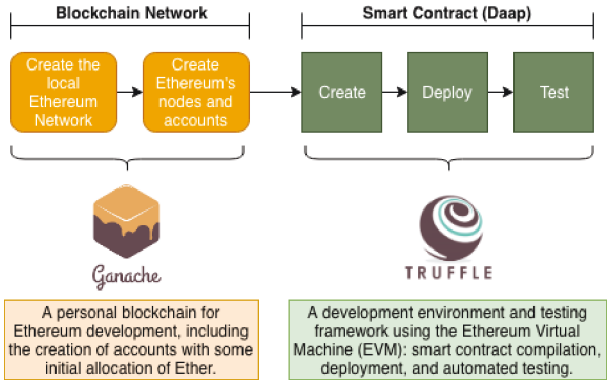

Fig. 13: Public Blockchain Local Development Environment

The main goal of this implementation is to verify the correct implementation of the different functions (i.e., entry points) to be included in our smart contracts (see Section IV). Before explaining our results, we need to go over the main assumptions behind the deployed system. First, since Ethereum is a public blockchain, it has a network-native asset:

[19]Smart Contracts running on top of blockchain-based platforms.

[20]Ethereum has the second largest market capitalization among all blockchain-based systems [30].

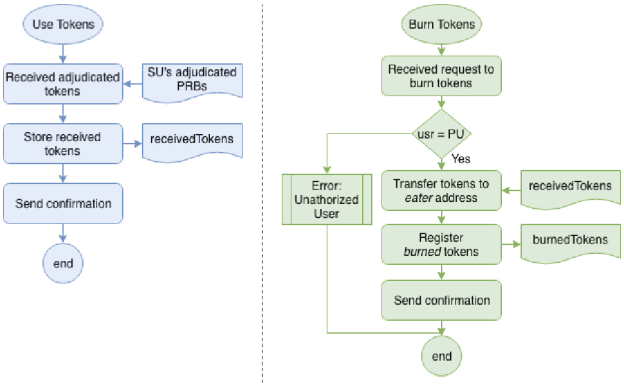

Fig. 14: Use and Burn Spectrum Tokens Contract: Main Functionalities

Ether (ETH) [55]. Consequently, it is not possible to define a new native *token* in this platform. To deal with this situation, a PU registers the available tokens as a variable in the contract (not as a native-asset) and transfers some Ether to emulate these resources. Then, when a SU request a set of tokens, it receives some of the Ether in the contract's own account, emulating in this way the transfer of, for instance, PRBs. Second, our local environment is composed of 11 accounts (1 PU and 10 SU) each of them preloaded with 100 Ether.

Deploying the Contract: First of all, we verify if the contract is correctly compiled[21]. Therefore, we utilize the *Truffle* compiling options on top of the local implementation of Ethereum (created using *Ganache*). The main objective is to verify that each of the functions and requirements are correctly transformed from the high-level scripts into EVM bytecode. The results of the contract deployments are depicted in Table V. As shown, both contracts are correctly initiated on the system while using some *gas*[22] for its deployment.

Contract	Gas Used	Gas Price (gwei)[23]	Total Cost (ether)	Price (USD)[24]
createTransferTokens	358,571	20[25]	0.00717142	2.13
useBurnTokens	273,842	20	0.00547684	1.62

TABLE V: Local Environment Contract Deployment

Testing the Contract Once the contracts have been successfully deployed, different users (i.e., Ethereum users) can call the distinct functionalities within the Dapp. This can also be tested automatically using the *truffle* and *ganache* suites. Hence, we initially test each function individually (see Table VI) and then develop a *testing script* to automatically verify different scenarios (see Fig. 15). The first observation that we can note here is that, in the same manner as the original deployment of the contracts, different functions are translated in different utilization of ether (i.e., costs). It is important to note that executing a function implies two costs: transaction (the cost of sending data to the blockchain) and execution (cost of computational operations which are executed as a result of the transaction)[26] [61]. In addition, we can see that the contracts follow all the requirements (i.e., controls) established in Section IV to guarantee their correct implementation. This is particularly true for identity verification, the registration of the different types of users, and the normal operations that would take place in our scheme.

Contract's Function	Transaction Cost (Gas)	Execution Cost (Gas)	Gas Price (gwei)	Total Cost (ether)	Price (USD)
Contract: createTransferSpectrumTokens					
registerSU	43,607	20,927	20	0.00129068	0.382
registerToken	42,224	20,760	20	0.00125968	0.374
transferToken	19,836	13,372	20	0.00066416	0.185
Contract: useBurnSpectrumTokens					
useToken	41,909	20,445	20	0.00124708	0.371
burnToken	39,879	33,607	20	0.00146972	0.437

TABLE VI: Local Environment Function Usage

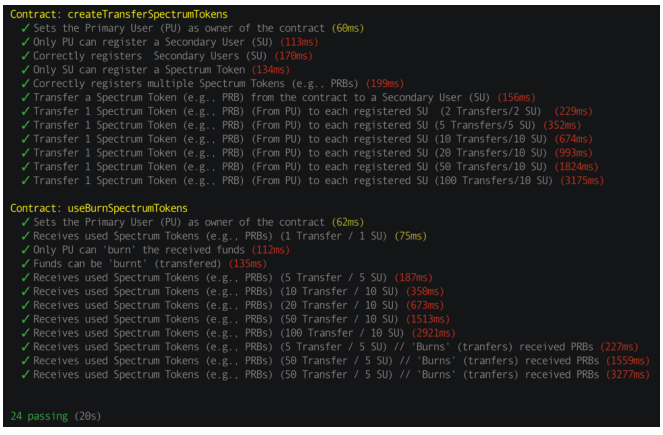

Fig. 15: Local Public Blockchain Testing (Screenshot)

[21] In Ethereum, Dapps are scripted (i.e., written) in Solidity to be later compiled into Ethereum Virtual Machine (EVM) bytecode [56].

[22] Ethereum introduces the concept of gas as a means to avoid for nodes (i.e., EVMs) to be caught in infinite loops. As the contract executes, it consumes some pre-allocated *gas* (i.e., Ether), when the contract runs out of *gas* it is halted and all transactions are reverted [57].

[23] Ehter is broken down by denominations. The smallest unit of ether is a *wei*, which equals $10E18$ ether, and 10^9 of a *wei* forms a *gwei* [58].

[24] The average price of Ether for the last 52 weeks: $297.50 [59].

[25] Ethereum Historical Median Gas Price [60].

[26] Functions that do not modify the status of the contract (e.g., change in the contract's variables) do not require *gas* for execution.

2) Ethereum Testnet: Once our system was fully tested in a local environment, the next step was to asses its functionalities in a working Ethereum Testnet (See Fig. 16). For this purpose, we have chosen the *Ropsten Test Network*.

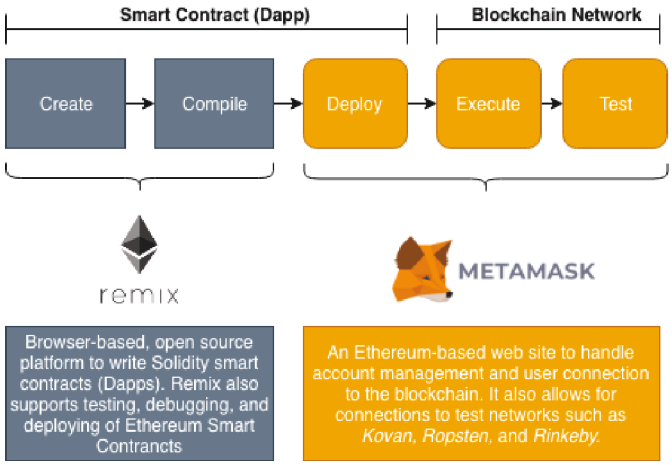

Fig. 16: Public Blockchain Testnet Development

Contract Deployment The main goal of this test is to asses whether our smart contracts (see Section IV) can be deployed in a *working* blockchain. We test whether the different functionalities of the smart contract are successfully executed in a *real-world* scenario. The results of the development, deployment, and usage of our smart contracts on the Ethereum testnet are depicted in Fig. 17. As it is shown both contracts are correctly deployed. Additionally, the different functions of the contracts are successfully executed, emulating in this manner all the operations in our system. In the same way as our local development, the deployment and usage of smart contracts imply a cost of operation. It is necessary to point out that the costs depicted in Table VII are actual costs[27] that reflect the fees charged by the Ropsten Test Network nodes (i.e., EVMs) to execute the operations of our Dapps.

Contract's Operation	Transmission and Execution Cost (Gas)	Gas Price (gwei)	Total Cost (ether)	Price (USD)
Contract: createTransferSpectrumTokens				
deploy	453,390	1	0.000453390	0.134
registerSU	43,405	1	0.000043405	0.012
registerToken	42,745	1	0.000042745	0.012
transferToken	19,725	1	0.000019725	0.005
Contract: useBurnSpectrumTokens				
deploy	254,645	1	0.000254645	0.075
useToken	41,909	1	0.000041909	0.012
burnToken	39,901	1	0.000039901	0.011

TABLE VII: Testnet Function Usage

B. Implementation on top of a Private Blockchain

There are multiple available private and consortium blockchain-based platforms with different configurations, features, advantages, and disadvantages. In this work, we use

[27]In the local environment, the price of gas was an estimation.

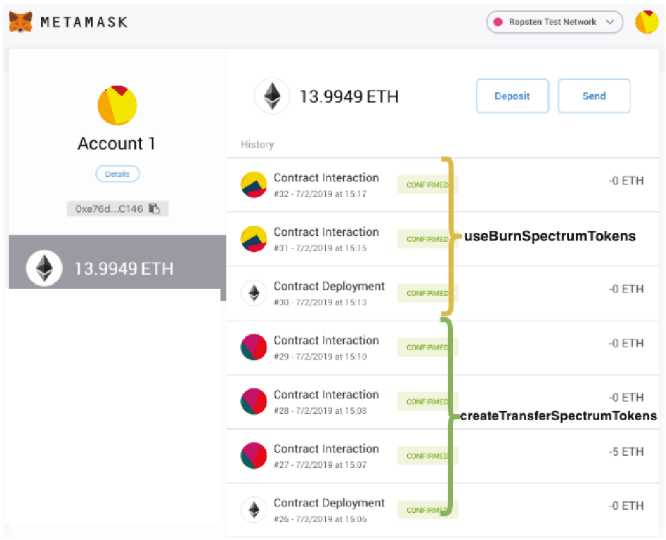

Fig. 17: Ropsten Contract Deployment (Screenshot)

Hyperledger Fabric to test the deployment of our network and the applications (i.e., smart contracts) built on top of it. Hyperledger Fabric is an open source permissioned blockchain platform. A key characteristic of the system is that it does not have a native cryptocurrency, which allows to any user to create an asset [62].

To test our implementation of a private blockchain network (i.e., business network) and the development, deployment, and test of the smart contracts[28] we use the setup described in Fig. 18. In this manner, we first create and develop an application to be further tested in a local environment of a distributed ledger: a Hyperledger fabric local development.

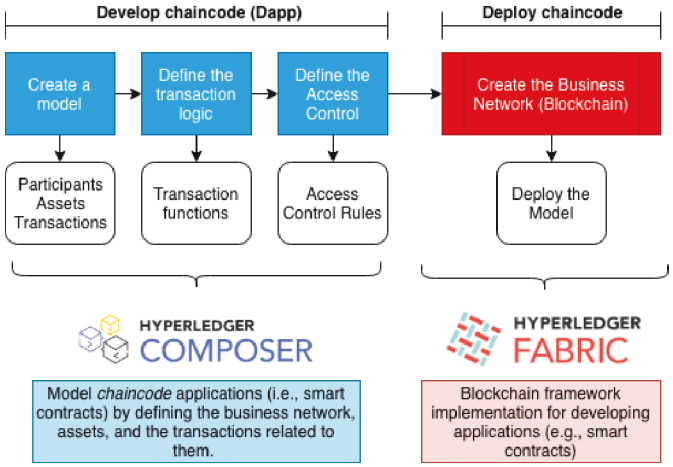

Fig. 18: Hyperledger Fabric Contract Deployment

Developing the Application The first step in testing our application in a private environment is to create the chaincode logic. For this purpose, we defined the digital asset or token to be transferred (e.g.,, PRB), the participants in the exchange

[28]In Hyperledger,, smart contracts are referred to as *chaincode* [63].

(i.e., the PU and SU), and finally the transaction logic (i.e., smart contracts). As shown in Fig. 19, all these pieces were successfully created and compiled to be further tested in a hyperledger fabric environment. It is worth mentioning that this application developed for a private environment has some small variations from the ideas presented in Section IV. First, a limited number of tokens is created. Instead of transferring ownership of the tokens (e.g., PRBs), the token included an *authorized user* field, which is continuously updated with the information of the SU with access rights to that asset or token. Since there is no transferring of ownership, the *burner* account was not included.

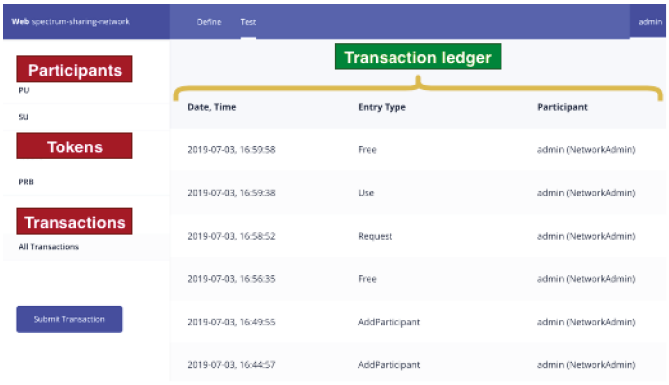

Fig. 19: Hyperledger Contract Deployment (Screenshot)

Testing the Application Once our application was created, it was time to deploy it into a fabric environment. Hyperledger Fabric offers multiple options to deploy an application: local, cloud, browser-based (i.e., Hyperledger Composer Playground), business networks, etc [64]. For our work, we tested the application in both, the Hyperledger Composer Playground (see Fig. 19) and in a local fabric development environment. In both scenarios, the application was not only successfully installed but also, we were able to test each of the functionalities that were implemented, while the ledger was correctly updated.

VI. Evaluation and Conclusions

This paper demonstrates the feasibility of constructing a blockchain system for a secondary cooperative spectrum sharing framework in the 1695-1710MHz band. But most importantly, the created system follows the *Integrative Framework for Collaborative Governance* (See Section II-A) to develop a collaborative enforcement (governance) for the 1695-1710MHz band trough blockchain.

A. Collaborative Governance and Enforcement

The system successfully creates, based on the collaboration dynamics, actions towards founding a Collaborative Governance Regime (CGR) in the sharing framework.

First, due to the clear configuration of the participants in the band, there exists a shared motivation between primary and secondary users to avoid unauthorized transmissions in the restricted zones. For the PU, this means the reduction of potential interference events within the restricted zones due to the embedded securities implemented in the blockchain. On the other hand, for the SUs the increased transparency and the potential increase in the available resources represents a clear incentive to join the system. Ergo, there is a *shared motivation* among all members to join the organization.

Second, since the system does not belong to any particular participant in the organization, this can guarantee the *principle engagement*. Further, the core characteristics of the blockchain (e.g., immutability, persistency, auditability, etc.) allow building a *trustless* environment, where there is no need for developing trusting relationships among the agents. In this manner, even if the principled agreements are not present, the system can still be successfully deployed.

The final component of the collaborative dynamics is *capacity for joint action*. A blockchain-based system requires that all or the majority of users are involved in many key mechanisms within the organization. For instance, users need to be engaged in the consensus mechanisms and/or the definition of the asset. Additionally, every single participant of the network receives a full copy of the ledger, which adds to the overall capacity of joint action.

All these characteristics lead to the emergence of the collaborative dynamics. Nevertheless, to fully reach a Collaborative Governance Regime, it is necessary to have *actions*. Here is where the blockchain-based systems play a crucial role: it is the means to achieve real actions. Moreover, the embedded features of a blockchain-based system act as the mechanism that glues everything together towards creating an operative and valid CGR.

B. Blockchain-based system for Spectrum Sharing

There are multiple examples of how a blockchain-based system can be implemented. For our work, we believe that the proposed design best captures the essence of the established conditions of the 1695-1710MHz band. In this manner, we have a clear, tokenized, virtualized, and digital asset as the core element in our system. In addition, the implementation of a consortium system best follows the conditions of the participants of the band: all users are well known beforehand. The computational and resource limitations of the participants can definitely limit the choices regarding consensus mechanisms for full nodes. In this light, we believe that an algorithm such as the pBFT[29] can definitely overcome these limitations. Furthermore, the differentiation between users and full nodes can help to overcome limitations by delegating functions such as validation and verification. Finally, we believe that the proposed smart contracts are a very good complement for the blockchain system. These contracts (i.e., Dapps) encapsulate many key requirements to guarantee the successful interaction of the participants and a more transparent enforcement process.

Our small scale testing or Proof of Concept was successfully implemented and tested. In fact, we were able to create,

[29]Variations of the algorithm such as Federated or simplified methods are also valid options.

deploy, and test our ideas into two well known blockchain-based platforms: Ethereum and Hyperledger Fabric.

The implementation of our system in a public blockchain, such as Ethereum, allowed us to have a better idea of the implications of developing such an application on top of this type of organization. First, since the system has already a native asset (Ether), it was necessary to map our token to the existing resources. We not only tested the different operations of our system but also, discovered additional implications, such us the potential costs of deployment and execution.

We also tested our system under a private blockchain using the IBM's Hyperledger environment. In this scenario, we successfully created, deployed, and tested our application. A key aspect of this implementation is that it was a closer example of how the system would look like. In this way, we were able to create our own native asset, the participants had a clear identity, and all the interactions happened in a closed (i.e., consortium) environment.

Both systems, public and private, proved to be feasible. Nevertheless, it is necessary to point out some considerations. First, the participants of the 1695-1710MHz band are well-defined and well-known. Second, for a blockchain system to be successful, it is necessary to have some sort of tokenized unit or the corresponding map to the platform's native asset. In other words, it is necessary to tokenize (e.g., virtualize) the spectrum band for it to be a better fit for a blockchain-based system. Implementing a blockchain to manage the interactions among the participants and other functions such as enforcement implies the introduction of an overhead to the system. Overhead that is highly correlated with the design of the system (e.g., type of blockchain, type of consensus algorithm, etc.) and can be translated in a considerable latency for *real-world* applications.

Finally, when talking about enforcement we can see that a blockchain-based solution brings many benefits into the scenario. First, it provides transparency for all users. Both the PU and SU know the exact number of resources available in the system (i.e., the created tokens). In addition, the blockchain registers all the interactions between the participants and the created Dapps. Second, the blockchain glues together all the required characteristics to develop a collaborative governance regime. Thus, all users are part and responsible for the system. Unauthorized transmissions in a restricted zone can be controlled through some core functions of the blockchain. For instance, a SU cannot use more resources than assigned, since a consensus would not be reached in the network and the requested transactions would not be committed. Finally, the presence of smart contracts provides an extra layer of information for the enforcement tasks.

VII. Future Work

Our smart contracts have a series of requirements for their correct implementation and usage. Nevertheless, new functionalities can be included in the system, especially to ease the tasks of enforcement. In this manner, the smart contract can monitor and register, for instance, attempts to generate unauthorized transmissions within restricted zones.

Our system was successfully tested in multiple environments (both in the private and public scenarios). Nonetheless, this was a proof of concept. It is necessary to test our proposed models in a much larger environment. For instance, we need to asses whether the system scales, the total cost of deployment, and the overhead and latency implications of a much larger organization.

In this work, we were focused on the sharing scheme of the 1695-1710MHz band. However, we believe that other spectrum sharing scenarios could benefit from the implementation of a blockchain-based system. In particular, we are interested in exploring how blockchain could be applied to, for instance, the sharing scheme of the 3.5GHz.

On the technical side of our proposed application, it is necessary to clearly define and test a way in which the spectrum band could be successfully tokenized (i.e., virtualized). For instance, we need to test the technical feasibility of using LTE Physical Resource Blocks (PRBs) as the tokenized spectrum units to be traded.

VIII. Acknowledgements

This work was sponsored in part by the National Science Foundation through grants 1265886, 1547241, 1563832, 1642949, and 1642928.

References

[1] G. Staple and K. Werbach, "The end of spectrum scarcity [spectrum allocation and utilization]," *IEEE spectrum*, vol. 41, no. 3, pp. 48–52, 2004.

[2] P. Bustamante, M. Gomez, M. B. Weiss, T. Znati, J.-M. J. Park, D. Das, and J. S. Rose, "Agent-based modelling approach for developing enforcement mechanisms in spectrum sharing scenarios: An application for the 1695-1710mhz band," 2018.

[3] M. B. Weiss, W. H. Lehr, A. Acker, and M. M. Gomez, "Socio-technical considerations for spectrum access system (sas) design," in *2015 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, pp. 35–46, IEEE, 2015.

[4] M. D. McGinnis, *Polycentric governance and development: readings from the workshop in political theory and policy analysis*. University of Michigan Press, 1999.

[5] M. Pilkington, "11 blockchain technology: principles and applications," *Research handbook on digital transformations*, vol. 225, 2016.

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

[7] K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the internet of things," *Ieee Access*, vol. 4, pp. 2292–2303, 2016.

[8] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "An overview of blockchain technology: Architecture, consensus, and future trends," in *2017 IEEE International Congress on Big Data (BigData Congress)*, pp. 557–564, IEEE, 2017.

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

[10] K. Fanning and D. P. Centers, "Blockchain and its coming impact on financial services," *Journal of Corporate Accounting & Finance*, vol. 27, no. 5, pp. 53–57, 2016.

[11] R. Brown, "A simple model for smart contracts." https://gendal.me/2015/02/10/a-simple-model-for-smart-contracts/, 2015. Accessed: 05/28/2019.

[12] E. Ostrom, "Beyond markets and states: polycentric governance of complex economic systems," *American economic review*, vol. 100, no. 3, pp. 641–72, 2010.

[13] K. Emerson, T. Nabatchi, and S. Balogh, "An integrative framework for collaborative governance," *Journal of public administration research and theory*, vol. 22, no. 1, pp. 1–29, 2012.

[14] E. Ostrom, "The evolution of institutions for collective action," *Edición en español: Fondo de Cultura Económica, México*, 1990.

[15] J. M. Bryson and B. C. Crosby, "Failing into cross-sector collaboration successfully," in *Big ideas in collaborative public management*, pp. 65–88, Routledge, 2014.

[16] C. Ansell and A. Gash, "Collaborative governance in theory and practice," *Journal of public administration research and theory*, vol. 18, no. 4, pp. 543–571, 2008.

[17] M. C. T. Relief, "Job creation act of 2012, pub," *L. I*, pp. 12–96.

[18] M. B. Weiss, M. Altamimi, and M. McHenry, "Enforcement and spectrum sharing: A case study of the 1695–1710 mhz band," in *8th International Conference on Cognitive Radio Oriented Wireless Networks*, pp. 7–12, IEEE, 2013.

[19] S. Bhattarai, A. Ullah, J.-M. J. Park, J. H. Reed, D. Gurney, and B. Gao, "Defining incumbent protection zones on the fly: Dynamic boundaries for spectrum sharing," in *2015 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, pp. 251–262, IEEE, 2015.

[20] M. Altamimi, M. B. Weiss, and M. McHenry, "Enforcement and spectrum sharing: Case studies of federal-commercial sharing," *Available at SSRN 2310883*, 2013.

[21] S. Bhattarai, J.-M. J. Park, W. Lehr, and B. Gao, "Tesso: An analytical tool for characterizing aggregate interference and enabling spatial spectrum sharing," in *2017 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, pp. 1–10, IEEE, 2017.

[22] J. R. Haber, *Accounting demystified*. Amacom Books, 2004.

[23] M. Szydlo, "Merkle tree traversal in log space and time," in *International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 541–554, Springer, 2004.

[24] M. Miller, *The ultimate guide to Bitcoin*. Pearson Education, 2014.

[25] M. B. Weiss, K. Werbach, D. C. Sicker, and C. Caicedo, "On the application of blockchains to spectrum management," *IEEE Transactions on Cognitive Communications and Networking*, 2019.

[26] Z. Zheng, S. Xie, H.-N. Dai, and H. Wang, "Blockchain challenges and opportunities: A survey," *Work Pap.–2016*, 2016.

[27] V. Buterin, "On public and private blockchains." https://blog.ethereum.org/2015/08/07/on-public-and-private-blockchains/, 2015. Accessed on: 05/29/2019.

[28] G. Greenspan, "Avoiding the pointless blockchain project," *MultiChain, blog*, 2015.

[29] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 4, no. 3, pp. 382–401, 1982.

[30] CoinMarketCap, "All cryptocurrencies." https://coinmarketcap.com/all/views/all/, 2019. Accessed on: 06/02/2019.

[31] S. Nakamoto *et al.*, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[32] A. Baliga, "Understanding blockchain consensus models," in *Persistent*, 2017.

[33] H. Sukhwani, J. M. Martínez, X. Chang, K. S. Trivedi, and A. Rindos, "Performance modeling of pbft consensus process for permissioned blockchain network (hyperledger fabric)," in *2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS)*, pp. 253–255, IEEE, 2017.

[34] C. Cachin, "Architecture of the hyperledger blockchain fabric," in *Workshop on distributed cryptocurrencies and consensus ledgers*, vol. 310, 2016.

[35] J. Stark, "Making sense of blockchain smart contracts," *UN Blockchain*, 2016.

[36] V. Buterin *et al.*, "Ethereum white paper: a next generation smart contract & decentralized application platform," *First version*, 2014.

[37] O. D. Hart, "Incomplete contracts and the theory of the firm," *JL Econ. & Org.*, vol. 4, p. 119, 1988.

[38] J. A. Fairfield, "Smart contracts, bitcoin bots, and consumer protection," *Washington and Lee Law Review Online*, vol. 71, no. 2, p. 36, 2014.

[39] C. Sullivan and E. Burger, "E-residency and blockchain," *computer law & security review*, vol. 33, no. 4, pp. 470–481, 2017.

[40] S. Ahmadi, *LTE-Advanced: a practical systems approach to understanding 3GPP LTE releases 10 and 11 radio access technologies*. Academic Press, 2013.

[41] G. Arunabha, J. Zhang, J. G. Andrews, and R. Muhamed, "Fundamentals of lte," *The Prentice Hall communications engineering and emerging technologies series*, 2010.

[42] M. Sawahashi, Y. Kishiyama, A. Morimoto, D. Nishikawa, and M. Tanno, "Coordinated multipoint transmission/reception techniques for lte-advanced," *IEEE Wireless Communications*, vol. 17, no. 3, p. 26, 2010.

[43] M. Gomez, L. Cui, and M. B. Weiss, "Trading wireless capacity through spectrum virtualization using lte-a," in *2014 TPRC Conference Paper*, 2014.

[44] S. Shobhit, "Public, private, permissioned blockchains compared," 2018.

[45] M. Palola, M. Höyhtyä, P. Aho, M. Mustonen, T. Kippola, M. Heikkilä, S. Yrjölä, V. Hartikainen, L. Tudose, A. Kivinen, *et al.*, "Field trial of the 3.5 ghz citizens broadband radio service governed by a spectrum access system (sas)," in *2017 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, pp. 1–9, IEEE, 2017.

[46] L. S. Sankar, M. Sindhu, and M. Sethumadhavan, "Survey of consensus protocols on blockchain applications," in *2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS)*, pp. 1–5, IEEE, 2017.

[47] Etherscan, "Ethereum block time history," 2019. Retrieved from: https://etherscan.io/chart/blocktime on 07-23-2019.

[48] BitInfoCharts, "Bitcoin block time historical chart," 2019. Retrieved from: https://bitinfocharts.com/comparison/bitcoin-confirmationtime.html on 07-23-2019.

[49] L. Bach, B. Mihaljevic, and M. Zagar, "Comparative analysis of blockchain consensus algorithms," in *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pp. 1545–1550, IEEE, 2018.

[50] B. Curran, "What is practical byzantine fault tolerance." https://blockonomi.com/practical-byzantine-fault-tolerance/, 2018. Accessed on: 05/31/2019.

[51] Y. Chen, "Blockchain tokens and the potential democratization of entrepreneurship and innovation," *Business Horizons*, vol. 61, no. 4, pp. 567–575, 2018.

[52] N. Lipusch, "Initial coin offerings–a paradigm shift in funding disruptive innovation," *Available at SSRN 3148181*, 2018.

[53] R. Smith, "Proof of burn: Consensus through coin destruction." https://coincentral.com/proof-of-burn/, 2019. Accessed on: 06/03/2019.

[54] R. Koulu, "Blockchains and online dispute resolution: smart contracts as an alternative to enforcement," *SCRIPTed*, vol. 13, p. 40, 2016.

[55] R. M. Parizi, A. Dehghantanha, *et al.*, "Smart contract programming languages on blockchains: An empirical evaluation of usability and security," in *International Conference on Blockchain*, pp. 75–91, Springer, 2018.

[56] L. Luu, D.-H. Chu, H. Olickel, P. Saxena, and A. Hobor, "Making smart contracts smarter," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pp. 254–269, ACM, 2016.

[57] C. Dannen, *Introducing Ethereum and Solidity*. Springer, 2017.

[58] Radar, "Ethereum base units." https://gwei.io, 2019. Accessed on: 07/01/2019.

[59] CoinMarketCap, "Ethereum price, charts, and market capitalization." https://coinmarketcap.com/currencies/ethereum/#charts, 2019. Accessed on: 07/01/2019.

[60] ETHGasStation, "Recommended gas prices in gwei." https://www.ethgasstation.info, 2019. Accessed on: 07/01/2019.

[61] V. Buterin *et al.*, "Ethereum white paper," *GitHub repository*, pp. 22–23, 2013.

[62] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich, *et al.*, "Hyperledger fabric: a distributed operating system for permissioned blockchains," in *Proceedings of the Thirteenth EuroSys Conference*, p. 30, ACM, 2018.

[63] V. Dhillon, D. Metcalf, and M. Hooper, "The hyperledger project," in *Blockchain enabled applications*, pp. 139–149, Springer, 2017.

[64] Z. Wan, M. Cai, J. Yang, and X. Lin, "A novel blockchain as a service paradigm," in *International Conference on Blockchain*, pp. 267–273, Springer, 2018.