Blockchain-Based Decentralised Autonomous Organisations

Felix Saraiva
felix.saraiva@tecnico.ulisboa.pt

Instituto Superior Técnico
Advisor: Prof. Miguel Nuno Dias Alves Pupo Correia

Abstract. Blockchain technology has been recognised as a prominent breakthrough in the information and economic sectors, presenting many possible successful applications, enabling unexplored opportunities for new systems based on decentralisation and security. Such characteristics allowed communities to use this technology to revolutionise the idea of decentralised organisations and create decentralised autonomous organisations. This study presents the current applications of such DAOs and their benefits. In order to further demonstrate the promising advantages of a DAO, this investigation looks at the currently applied public contests system that allows public institutions to recruit private services from external entities. Such contests lack the transparency, security, and democratic representation that a DAO could provide. Therefore, this study proposes the implementation of a permissionless-based DAO in the *Ethereum* blockchain, supported by a consensus algorithm that replaces the intermediary authority in public hiring contests.

Keywords: blockchain, public contests, decentralised autonomous organisations, smart contracts

Table of Contents

List of Figures

1 Introduction

From its early stages, blockchain technology has been solving several problems in our society regarding transparency, privacy, and immutability [2]. Its most recognised application, Bitcoin, has led to groundbreaking changes in our economic systems and currencies in the digital world [3].

More recently, developers and researchers are exploring the advantages of developing decentralised web applications directly on the blockchain through smart contracts, hoping that these projects get recognised by the market and reshape the way web applications are developed into a futuristic idea called Web3. The Web3 universe surrounds itself with qualities such as decentralisation and automatisation, originating the concept of a Decentralized Autonomous Organization or DAO [4].

DAOs are collectively owned and managed organisations conceived in the blockchain. In order to make decisions or use their built-in treasuries, approval from the members is required. There are no CEOs in a DAO, therefore, it does not have a central point of authority capable of abusing its decision power in the organisation [5].

1.1 The Problem

Several strategies are devoted to solving trustworthiness, transparency, and traceability problems in economics. Problems such as the Principal-Agent Problem [6], or business interactions where intermediary individuals can alter the course of an arrangement, are problems where the blockchain technology, and particularly DAOs, could be the solution. These problems are particularly challenging when the exchange is between public and private organisations.

Let us consider the example of a public institution that requires to hire outside services from a private corporation. In most democratic countries, a public contest, organised by a public institution, is arranged to decide between several candidates which one is more qualified to provide the service. These contests should be as transparent and immutable as possible, but unfortunately, the way they are applied raises several reasons for improvement.

Most public institutions employ the services of a trusted third party that operates as the bridge between the public institution and the contestants, introducing a new layer of trust between the three entities. *Vortal* is an example of an entity that provides such services (*https://www.vortal.biz/*). The contestants interact directly with this trusted third party by submitting their proposals on an electronic platform. Then the proposals are sent to the private institution for validation and approval by the jury members, resulting in a final winning proposal that the trusted third party declares to all the contest members. All the members have access to this information and are well informed about why their proposal was not chosen.

This study proposes a Decentralized Autonomous Organization to be implemented instead of a trusted third party that acts as the middleman between the public institution and the contestants. This way, both the contestants and the public organisation communicate directly with the DAO, which inherits characteristics of blockchain technology such as decentralisation, security, immutability, traceability, transparency, privacy and interoperability.

In fact, *Vortal* itself is already considering implementing blockchain technologies in its electronic platforms, as it can be verified on its "Research & Innovation" page [7]. Moreover, the company and academics published a recent paper related to that idea [8]. However,

their implementation insists that decisions are exclusively made within the organisation in a permissioned blockchain, different from the implementation presented by this study.

1.2 Document Structure

The remainder of the document is structured as follows. Section 2 presents the related work that highlights the most important topics of the research, beginning with the explanation of the blockchain technology, its characteristics and components, such as the consensus algorithm. Then the research moves to the application of smart contracts in the blockchain; this subsection describes smart contracts' Ethereum structure, development and lifecycle. With the required knowledge from blockchain technologies and smart contracts, the study moves to the main topic of Decentralised Autonomous Organisations, starting with the differences between decentralised organisations, decentralised applications and DAOs. The study then proceeds to explain one of the main problems solved by DAOs, the principal-agent problem, followed by the core requirements of a DAO, the many voting mechanisms and the several classes of DAOs.

Section 3 holds into consideration the problem presented previously in this section and structures a possible solution to be implemented. It starts by describing the possible use cases of the system and its functional and non-functional requirements. The remainder of the solution's explanation is divided into the main components of a DAO, suggesting possible implementations for a blockchain infrastructure, a network of stakeholders, DAO tokens and governance processes.

Section 4 describes the evaluation methodology of the proposed solution. This section is divided into possible evaluation methods to be applied against the existing solution, followed by the smart contract's code evaluation and the system's evaluation as a DAO.

To conclude, this study presents a schedule of future work divided into the several months that follow. The final section presents the final thoughts and conclusions from the entire research.

2 Related Work

This section presents the work that has been done around the components of the proposed solution. First, blockchain technology is explained along with the essential aspects surrounding it. The characteristics of Consensus algorithms such as Proof-of-Work and Proof-of-Authority are required for a solid decision on what would be the best implementation for the solution. Following the initial blockchain research, the Ethereum implementation of smart contracts, their lifecycles, and structure is presented as an essential foundation for the solution's architecture. Finally, the main topic, Decentralized Autonomous Organisations, is analysed and compared against other types of organisations. The different classes of DAOs are presented, along with different voting mechanisms and the requirements that differentiate a Decentralised Application from a DAO.

2.1 Blockchains

In order to first understand the notion of blockchain, one must grasp the abstraction of a distributed ledger with the purpose of appending timestamped digital transactions/documents in a way that it becomes impossible to tamper with them. This distributed ledger is

distributed in a peer-to-peer network where each member has a copy of the entire chain of blocks and can therefore verify the authenticity of the blockchain. This verification method is possible because each block stores a *hash* of the previous block. A *cryptographic hash function* H holds the property of being very easy to compute $H(a)$, but highly unfeasible to find an $a' \neq a$ such that $H(a') = H(a)$ [9]. Therefore, if any byzantine member (a malicious actor) of the network attempts to tamper with any of the early records of the chain, that record and all that follow will become invalid because they no longer hold a valid hash of the previous block. Although very powerful, hash functions alone are not enough to assure the complete security of a blockchain. Thanks to the evolution of technology, computers are getting faster at calculating hashes, driving brute force techniques to be more reliable. Therefore more protocols and algorithms are required to ensure all the qualities of this technology. One important aspect of blockchains is that there can be two types of blockchain networks, *permissionless* or public blockchain and *permissioned* or private blockchain.

Permissionless Blockchains. In blockchains such as Bitcoin or Ethereum, anyone can access the ledger, issue a transaction, run a node, or publish a smart contract. These permissionless public networks carry anonymity and full transparency, but slow performance and complex scaling possibilities [8]. In these chains, the nodes receive monetary incentives through Proof-of-Work, a protocol later discussed in this section that consumes large amounts of energy.

Permissioned Blockchains. Although the first vision of blockchain was destined for building public and trustless networks with no central authority, the enterprise and organisational fields are evolving towards a more private and permissioned application of blockchains. Permissioned blockchains are very much like public blockchains, with the same level of traceability and immutability, with the difference that nodes need permission to access the network. Quorum and Hyperledger are some examples of private blockchains which use different protocols than PoW, such as the Proof-of-Authority protocol [10]. By providing an additional level of security that can be seen as an access control layer, members of permissioned blockchains require to identify themselves, often through the use of contracts, and can only perform specific actions granted to them by the ledger administrators.

Consensus. In a centralised system, it is straightforward to reach an agreement; a central authority holds a master copy of the ledger and makes sure everyone agrees on that copy. The same cannot be obtained when addressing a system where trust in a central authority is undesirable. A consensus algorithm addresses this problem by allowing all nodes of a blockchain network to reach a joint agreement about the current state of the distributed ledger.

Consensus algorithms are also present in the solution of problems such as the Double-Spending Problem and Byzantine Generals Problem. In the double-spending problem, the same currency is reused on two transactions simultaneously. Blockchains address this problem by making several distributed nodes verify the transactions. Furthermore, because blockchains are a distributed approach, there is the possibility that some nodes might be malicious, which leads to changes in the communication contents. Such a situation presents the Byzantine Generals Problem, where normal nodes need to identify tampered information and obtain consistent results with other normal nodes. Such solutions are present in the design of the corresponding consensus algorithm [11].

There are several consensus algorithms currently applied in distributed systems, and this study presents a detailed description of the most commonly used in blockchains.

Proof-of-work. In order to achieve consensus, permissionless distributed ledgers, such as *Ethereum* and *Bitcoin*, use a consensus protocol called *Proof-of-Work*. *Proof-of-Work* is directly related to mining a block in a blockchain. Mining is the act of going through an intense race of trial and error to complete a proof-of-work challenge, which ends when a miner finds the correct nonce that meets the condition of the challenge. Only blocks with a valid nonce can be added to the chain. Once a miner solves the complex computational puzzle, the blockchain awards him a reward, introducing the idea that it is more profitable to look after the network than attack it. Also, solving the puzzle can effectively be seen as a vote, preventing Sybil attacks. Because the entire PoW is based on a hash, a block cannot be counterfeit without breaking the *Hash Function*, providing a tamper-proof quality. One of the intents of Consensus is to extend the chain; the more extensive the chain, the harder it is to counterfeit blocks. Rewriting a part of the chain requires solving all the associated puzzles; however, a malicious node would require over 51% of the network's mining power to achieve this successfully. Although robust and secure, a PoW algorithm is very slow and consumes too much energy and resources. Hence the energy spent by the attacker would probably outweigh the profits [12].

Proof-of-stake. In the current reality, blockchains such as Bitcoin use enormous amounts of energy. According to Digiconomist, annually, it consumes an estimate of 204.50 TWh per year, comparable to the power consumption of Thailand [13]. Alternative approaches started being considered to reduce the scalability and environmental sustainability surrounding the use of the PoW protocol. Proof-of-Stake uses the validator's coins (stake) as the non-counterfeit resource. This way, nodes vote with their coins rather than with computational power and the chance to select the next block is proportional to the stake size. In Proof-of-stake, nodes can become Validators if they meet the required conditions; usually, a certain amount of stake must be deposited. Validators are then chosen at random to create blocks and are responsible for checking and validating the blocks they do not create. When a majority of validators approve a block, it gets added to the blockchain, and the creator receives the fees associated with the transaction inside this block as a reward. If a validator validates a malicious block, it loses its pre-deposited stake, making it more profitable to protect the network than to attack it [14].

Proof-of-Authority. Proposed as part of the *Ethereum* network, PoA is another consensus protocol, from the family of *Byzantine fault-tolerant* (BFT) algorithms, designed for permissioned blockchains and implemented into the clients *Aura* and *Clique*. It is based on a mining rotation schema relying on a set of N trusted nodes called authorities. Each authority is uniquely identified, and a majority of at least $N/2 + 1$ is considered honest. In order to obtain consensus in PoA, time is divided into rounds, and each round has a leader who elects a new block to be added to the chain. In *Clique*, the leader's proposal is absolute and immediately committed by the other authorities, whereas in *Aura*, the other nodes must first accept the proposal and only then the network will accept the new block [15].

Forks. Blockchain forks are directly related to the appearance of new software versions within the community. When a change to the blockchain is required, the blockchain diverges into two potential paths, when this happens, the existing blocks are not affected, assuring the verifiability and integrity of the blockchain. These changes can be forced to alter the blockchain's protocol or some basic set of rules depending on the origin of the conflict. Short-lived forks result from the difficulty of reaching consensus within the system. Considering

the example of accidental forks, when two blocks are found simultaneously, the blockchain is forked into two chains where subsequent blocks are added, eventually resulting in one chain being longer than the other. The blockchain then abandons the blocks in the shorter chain, referred to as orphaned blocks, returning to a single chain [16]. The second type of forks is the intentional forks, representing intentional changes or updates to the blockchain. Inside intentional forks, we have:

1. **Hard Forks:** In this case, blockchain nodes need to decide whether they stay in the old version or install and upgrade to the new software with new rules. If they stay in the old version, they will see the new blocks from the new version as invalid. A permanent split can be forced if one group of nodes refuses to use the new software. There are many reasons to execute a hard fork, such as creating an entirely new cryptocurrency, fixing some major flaw in the system, or even undoing the consequences of a hack, just like what happened to the Ethereum organization TheDao.
2. **Soft Forks:** Like in hard forks, rules are changed in the blockchain, and both the new and old versions are present to the community. The significant difference is that with the addition of new rules, old version users stop processing the blocks in hard forks, and in soft forks, the two versions of the software typically remain compatible. It can also be seen as a hard fork creating two blockchains, while a soft fork only results in one [17].

2.2 Smart Contracts

Blockchain technology introduced qualities desired by numerous applications far beyond cryptocurrencies and monetary transactions, leading to the idea of *smart contracts*. Abstractly, smart contracts represent settlements between untrusted parties automatically enforced by the consensus algorithm without the need of trusted third party authorities [18]. Although the name might point to it, smart contracts are not necessarily legal documents. In more technical terms, smart contracts are computer programs written in code in a blockchain, executed when predetermined conditions are met and verified.

Ethereum smart contracts. In Ethereum, smart contracts are seen as immutable computer programs that run deterministically in an *Ethereum Virtual Machine* on the Ethereum blockchain. They are immutable because smart contracts do not present the chance of being changed or updated like regular software; the only way to change a smart contract is to deploy a new instance of the same; deterministic because the outcome of the execution will always be the same regardless of the user. Because they run inside an EVM, they can only access their state, the context of the transaction that called them and some information about the most recent blocks [19].

When developing a smart contract in the Ethereum blockchain, programmers typically use high-level languages, like *Solidity*, that compile their code into bytecode executed in the EVM. Creating a smart contract begins with a contract creation transaction sent to a unique destination address called the *zero address* (address 0x0); this transaction requires only a data payload that contains the compiled bytecode, which creates the contract. This address can never spend Ether or initiate a transaction; it is merely a destination address that creates a contract, although it can be used to destroy Ether deliberately by receiving an amount of Ether that can never be spent again [19].

Lifecycle. Every contract is identified by an Ethereum address derived from the contract creation transaction; this address is then used to send funds to the contract and to execute one of the contract's functionalities. So, deploying a smart contract is, in a way, a transaction; therefore, it requires the creator to pay the fees of that same transaction. It is significant to mention that the contract's creator does not hold any privileges at the protocol level; they do not hold a private key of the contract because such keys do not exist. In other terms, contracts own themselves [19].

Contracts in Ethereum remain in a standby state until executed by initiating a transaction from an *Externally Owned Account* (EOA). This execution can be triggered directly by a transaction or indirectly when a contract calls another contract. A contract can even deploy another contract; though it is not possible to execute contracts in parallel, the execution is seen as a *Single-Threaded Machine*. If an execution fails, all changes are undone, the attempt gets recorded as a failed transaction, and the Ether spent is deducted from the originating account. Otherwise, if an execution is successful, the program is executed without errors and reaches the end of its execution [19].

Smart contracts can be a powerful technology, but they also have limits. By design, they are isolated from the rest of the internet since a contract cannot get external information via HTTP requests. This happens deliberately due to the early explained characteristics of the *consensus* algorithm, where relying on information external to the blockchain could compromise the protocol. This boundary can be surpassed using *Oracles* [20]. Oracles are a bridge between outside data and a blockchain. They connect smart-contracts with external APIs to retrieve and verify external data for blockchains. An Oracle is as reliable as the information it provides, and of course, using such a tool does not fix the problem of relying on a single external piece of information that can be manipulated. Therefore when applied, an oracle should be decentralised, meaning it should pull data from multiple sources, preventing the possibility that one source can be malicious or fail and consequently compromise the contract's execution [21]. Another limitation is the maximum size of a smart contract, which is limited to 24KB; otherwise, the contract runs out of gas. In order to avoid such a situation, developers use the *Diamond Pattern*. A *diamond* is a contract with external functionalities provided by other contracts called *facets*. *Facets* are contracts that share functionalities, libraries, and variables independently [22].

Although it is stated earlier in this chapter that smart contracts are immutable, they can still be deleted. A contract becomes a blank account by removing its code and the internal storage from its address. After such deletions, any attempts to execute the code from that "contract" will fail. In order to have the option to delete a contract, the developers must include the functionality in the contract's code. In an EVM, the operation is called *SELFDESTRUCT*. The developers are incentivised to include such an operation by receiving a *gas* refund after releasing network resources from deleting the contract's stored state [19]. In Ethereum, "gas refers to the cost necessary to perform a transaction on the network" [23]. Deleting a contract does not clear its transaction history since the blockchain is immutable.

Development. Designing and developing smart contracts begins with the selection of the platform on which the smart contract is written, followed by the choice of a language the platform supports. Script-based blockchains, such as Bitcoin, use a simple language that supports only limited operation expressions; this facilitates the flexibility of its primary purpose, simple financial transactions. In order to surpass the limitations of script languages, Ethereum introduces the Ethereum Virtual Machine that can execute bytecode, support

Turing-complete programming languages and theoretically execute any computer program [24].

Although there exist several programming languages for smart contracts currently being supported, in Ethereum, the most popular is *Solidity*. *Wood* created Solidity with the intent of programming smart contracts; it has a syntax similar to *JavaScript*, and thanks to its success and general attributes, it is applied to several other blockchain platforms. The Solidity project also contains an application binary interface (ABI) responsible for encoding the contract calls for the EVM and reading data out of transactions. In computer science, an ABI is an interface between two program modules, usually between the OS and an application. In Ethereum, the responsibility of the ABI is to declare the functions of the contract and make the connection between a function's arguments and its return values [19].

Structure. In order to have some practical illustration of the theory presented so far, the following code represents a simple example of a smart contract. (in Solidity)

```
1
2    // SPDX-License-Identifier: GPL-3.0
3    pragma solidity >=0.4.16 <0.9.0; // Solidity compiler version
4
5    contract SimpleStorage { // Contract beginning
6        uint storedData; // Declaring a state variable of type uint
7        function set(uint x) public { // Stores a supplied number
8            storedData = x;
9            }
10       function get() public view returns (uint) { // Returns the
   stored number
11           return storedData;
12           }
13   }// Contract end
```

Listing 1.1: Simple Storage Example. Extracted from [4]

Following the line that encodes the used version of Solidity, we can witness the beginning of the contract, which looks very similar to the initiation of a class in Object-Oriented Programming. Then we have the declaration of a state variable called *storedData* and two functions, *set()* and *get()*. The first receives a parameter of type *uint* that is then stored in the variable created earlier; the last function returns the previously stored value inside *storedData*. As we can conclude, this is a straightforward contract with two functions that anyone can access without any restrictions on who can alter the number stored in *storedData*.

2.3 DAOs Decentralised Organisations

In general, the hierarchical structure of organisations has maintained itself somewhat regularly over the years. Although it differs from corporation to corporation, humans usually hold the responsibility of separating individuals into several classes and establishing the conditions of those classes, therefore setting the powers and limits of each member of the organisation, and as we all know, humans can be biased.

Decentralised Organisations. Centralised Organisations focus their management and decision-making on the top of the organisational hierarchy. For example, coordinating financial and human resources, talent deployment and marketing are conducted by a single

senior team or individual. Centralisation is a good preference if scalability or standardisation carries a significant financial concern to the organisation [25]. On the other hand, decentralised organisations switch the human management of property via the legal system by a direct interaction between people according to a protocol specified in code or electronically [5]. Decentralisation is more appropriate when the organisation is looking to provide different services for different markets, requiring the company to respond more quickly to changes in customer needs [25]. Centralisation versus Decentralisation is a trade-off between extensive scaling and stable standardisation against fast response and flexibility to change.

Decentralised Applications. Standard web applications such as *Twitter* and *Uber* reside on centralised servers, controlled by the software's organisation, that provides a Client-Server architecture. Although very scalable and agile, centralised applications do not provide the desired characteristics of decentralised systems.

Decentralised Applications or DApps are built using smart contracts as their *backend* code which runs on decentralized peer-to-peer networks. Their *frontend* can be written in any language, just like a standard app. DApps often use off-chain data storage services to manage their *data storage* due to the high gas cost and current low block gas limit, which causes smart contracts to be inappropriate for storing large amounts of data [19]. DApps are distinguished by being open source, respecting the nature of blockchains where third party audits must be possible, supporting cryptocurrency, using tokens to quantify all credits and transactions among participants, assuring integrity and transparency between nodes using the *Decentralised Consensus* algorithm, and not having a central point of failure, since the system is itself executed in a blockchain [26] and is therefore distributed. Although DApps promote privacy and resistance to censorship, their development is still in its early stages, with a few disadvantages. So, it is important to note that DApps might be hard to scale and modify since smart contracts are built to be immutable. Also, it is a technology where the creation of user-friendly frontends is quite challenging, decreasing its ease of use.

Decentralised Autonomous Organisations. A DAO is a self-organised organisation, with no centralised hierarchy, controlled by smart contracts running on the blockchain. The difference between a DAO and a Decentralised Organisation is that a DAO has internal capital or some internal property that has value somehow. For example, the DO BitTorrent has no internal property; Bitcoin, on the other hand, does. However, we can argue that Decentralised Organisations do have internal property, the difference here lies in the word "Autonomous" while in a DO, the organisation's members are the ones that make the decisions, in a Decentralised Autonomous Organisation, the DAO itself makes decisions [5].

Like DApps, the creation of a DAO is highly based on a smart contract launched on the blockchain. When the DAO is created and deployed, a poll is the only way to alter its regulations. After its launch, a DAO can start to collect funds for the purpose or project it was created. These funds come in the form of native tokens, a currency that is linked to the DAO's smart contract.

Once members start to invest in the DAO, they can earn rights such as voting in decision polls, providing feedback or even setting future ideas for the organisation [27]. Some DAOs have exclusively one type of token that performs different functions, and other DAOs can have multiple types of tokens that are used for different functionalities. For example, one token can be used as a payment mechanism and the other as a governance token, separating investing members from those who use the last token that can be used for voting. However, not all token holders necessarily have a right to vote. In certain DAOs, such as *Dash*, only

masternodes are allowed to vote, but any member who holds 1000 dash tokens becomes a *masternode* [28]. In *MakerDAO* only those who hold a unique type of token, MKR, can vote for the risk management and business logic of the Maker system [29]. It is also possible to distribute non-transferable tokens based on a member's reputation in the DAO. Reputation can be earned and lost based on the consequences of a member's actions.

In a DAO, the decision making process is one of the core features that separates it from the more standard organisations. Decisions must be recorded on-chain and must be executed automatically. Meaning that if a public blockchain is used, anyone can verify the voting, not just the token holders, and once a proposal pool comes to a successful decision, that same decision must be applied automatically, without the need for human interference [28].

With these improvements to decentralised organisations, DAOs also address security and economic problems, benefiting from the qualities of blockchains and the interactions end users have with such organisations. One of those problems is the Principal-Agent Problem.

The Principal-Agent Problem. In the business world, circumstances often occur when an "agent", is hired by another entity, the "principal", to make decisions on behalf of the principal's interests. However, the agent can decide to act in their own best interests contrary to their principal's. Such a situation happens due to a conflict in priorities, as shown in Figure 1 below, and *asymmetric information*, with the agent retaining more information about the assignment than the principal. Therefore the principal cannot trust the agent to work in its best interest [6]. In economics and political sciences, this problem is known as the *principal-agent problem*, also known as the *agency dilemma*.

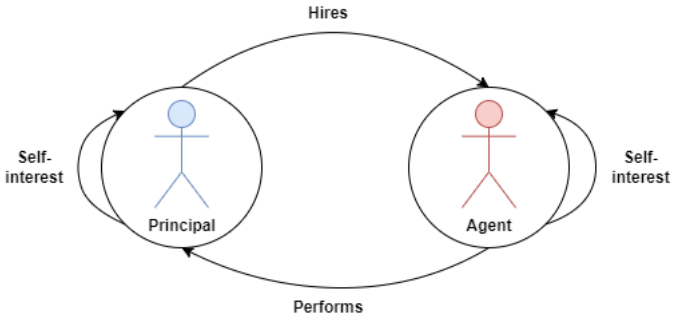

Fig. 1: Agency Theory basic idea [1]

This problem represents a complex challenge to both the public and private sectors and makes the interactions between the two difficult. DAOs address this problem by ensuring that an organisation's incentives and information flow are appropriately aligned in a codified format. The alignment of incentives is a defining concept of blockchain protocols and is therefore inherited by a DAO.

A properly executed DAO aligns stakeholders' incentives from its founders, token holders, users, and the general community to govern a decentralised organisation and, therefore, solve the Principal-Agent Problem [30].

Requirements. When studying the requirements of a DAO, it must be clear that DAOs are similar to traditional organisations but built around a new and unique model. Therefore,

most DAOs will inherit several financial and determination goals from traditional organisations. Hence, among other requirements, a DAO should have:

1. **A Purpose:** A DAO starts with a shared goal or idea that has the potential to grow around a community or organization. Therefore, a DAO without a good and solid project or motivation will hardly succeed.
2. **A Community:** With a good idea settled, there must be a way for the community to get together and discuss the topics and future paths of the DAO. So, DAOs must have a way of spreading the project across a large number of stakeholders to organise the organisation's decentralisation so that members can decide the direction the DAO will take [31]. There are several ways of doing this; the most famous ones are Discord, Telegram and Snapshot.
3. **A Voting Mechanism:** Organisations that are best implemented as DAOs need to establish a method for participants to engage in decision-making. Voting is the primary mechanism that allows the DAO to make changes and decentralised decisions. This mechanism can be created within the DAO's smart contract, or it could use an external third-party provider.
4. **A Governance Token:** Although DAOs are inclusive and transparent, members are still required to prove their right to an opinion. Some DAOs employ permissionless and mintable governance tokens. Other protocols issue tokens when users provide market liquidity or network security, and some DAOs even use Non-Fungible Tokens (NFTs) to manage governance. The objective of these digital possessions is to define that only holders of such tokens have voting rights [32].
5. **Funds Management:** A DAO's funds are managed in a decentralized way. Therefore, most will use a collective treasury usually held in a multi-signature wallet, which can only be spent if all the main participants agree [31].

Concerning the order of these requirements, there exist numerous discussions about the suitable models for starting a DAO. The sequence displayed above is merely a proposal.

Voting Mechanisms. Because governance protocols are still in their early development, many voting mechanisms are still being developed and tested. Voting has a significant role in DAOs, influencing member participation, security and community culture. The selected type of mechanism can play a critical role in the long term success of projects. Voting mechanisms are divided into two categories:

- **On-Chain Voting:** Submits votes directly on the blockchain as transactions, requiring voters to pay transaction fees for each vote. It enables smart contracts to execute proposals automatically based on the result of the on-chain pool, and therefore, there is no need to use a trusted third party to count or enact votes [33].
- **Off-Chain Voting:** In token weighted voting schemes, voters have to sign messages with their crypto wallet to vote, and that data is then recorded in a decentralized file storage system such as IPFS, reducing the risk of tampering. However, a trusted third party is still needed to accurately count the votes and enact the results on-chain using escalated privileges [33]. The currently most used off-chain voting system is *Snapshot*. When using non-token weight voting, polls are done in forums or chat servers, giving each user only one vote, regardless of the number of tokens each user holds. This method is vulnerable to manipulation, but it stimulates community discussion and allows early feedback from the voters before a more official token voting. In these approaches, transaction fees are not required to vote.

In order to better understand the characteristics of voting protocols, the list below represents some of the most popular voting mechanisms applied in current DAOs. Warning that there is no 'correct' voting protocol, therefore, the focus should be on each protocol's trade-offs.

1. **Token-based Quorum Voting:** The most commonly used protocol in DAOs, Quorum voting, requires a certain threshold of votes for a proposal to pass. Once this percentage or number of votes is satisfied, the decision with the most votes goes forward [34]. If the required Quorum is never reached, the proposal fails.
   - **Pros:**
     - Highly implemented and therefore tested.
     - Straightforward for voters.
   - **Cons:**
     - Selecting the right quorum requirement is not trivial. A low quorum makes a proposal easy to pass a high quorum makes it harder.
     - Requires a high amount of participation from members to pass proposals. It can be expensive and time consuming.
     - Token-based voting is sensitive to certain kinds of attacks such as flash governance attack.
2. **Holographic Consensus:** In this mechanism, proposals are associated with predictions. Members can stake funds *for* or *against* a proposal they believe will pass or fail. If correct, predictors get financial rewards and the proposals with higher passing predictions are "*boosted*". In those proposals, the voting quorum switches from an absolute (for example, 60%) to a relative majority, where the number of "*For*" and "*Against*" votes are compared to reach a decision. In order to vote, participants require *Reputation*, which is earned, not bought [35].
   - **Pros:**
     - Protects projects from malicious proposals, since payment is required to affect the mechanism (via staking funds on proposals).
     - Allows DAOs to process proposals faster.
     - It is a solid proposal for the problems of scale and resilience in DAOs.
   - **Cons:**
     - It is a much more complex and expensive solution to implement.
3. **Conviction Voting:** Based on a decision-making mechanism, that funds proposals and allows voters to split their token weight through several proposals. Voters can change their opinions and move their votes from one proposal to another. The longer a user holds a preference for a specific proposal, the more its 'conviction' grows and that provides more weight to that preference over time. This new conviction factor gives consistent preferences more influence than short term participants that merely try to influence a vote [36]. In other words, voters are continuously asserting their preferences for which proposal they think should be approved.
   - **Pros:**
     - Provides collusion resistance, avoids Sybil attacks and mitigates several attack vectors of time-boxed voting mechanisms.
     - Prevents users with large stakes from suppressing minority voters.
     - There is no need for a majority consensus in every proposal; instead, voters can hold on to the proposals they support [34].
   - **Cons:**

- Complex and difficult to implement.
- Usually used alongside other voting mechanisms better suited for approved/disapproved proposals.

4. **Quadratic Voting:** Seeking to address the problems of the voting paradox and the majority rule, Quadratic Voting focuses on the allocation of votes that express the degree of preferences of the voters. It works by assigning total voting budget per member where voters can pay for more votes on a specific proposal emphasizing their preference [37]. The cost of each additional vote grows exponentially:

$$cost\_to\_the\_voter = (number\_of\_votes)^2$$

It can be implemented by assigning a budget of votes per member, increasing democratic equality from the determined allocation of votes. Alternatively, it can apply costs for votes using currency, such as the native DAO governance token, thereby decreasing the wealth influence in the voting process by the increasing exponential cost of buying more votes [37].

   - **Pros:**
     - Minorities can buy additional votes, which helps increase total social welfare.
     - The model can be applied to funding mechanisms for public goods
     - Its a robust and efficient mechanism.
   - **Cons:**
     - The direct involvement of money can benefit the wealthy, who can afford to buy more votes than the rest of the population.
     - Lacks moderation when dealing with Sybil Attacks.

Crypto Airdrops. One way to promote a new launch of a blockchain startup project, such as a DAO, is by issuing tokens for free. Airdrop is a method to distribute tokens, help projects reach a broader audience, and incentivise the increase of awareness in token holders. In a DAO, if a person holds governance tokens, they are more likely to participate in the organisation's decisions, resulting in a stronger DAO [38]. Airdrops can be executed in several ways depending on the motives, but the most common are:

- **Bounty airdrop:** In order to receive the airdrop of digital tokens, the user must complete a set of promotional tasks, for example, sharing a social media post or signing up for a newsletter.
- **Holder airdrop:** Snapshots in airdrops are usually done before an airdrop round to record the balance of each token holder at that specific point in time. In holder airdrops, Snapshots are used to distribute the tokens based on each token holder's balance [39].
- **Standard airdrop:** In this option, projects distribute digital assets evenly through all the wallets associated with their community.

Classes of DAOs. Due to the flexibility that smart contracts introduced to blockchain technology, it is possible to assemble DAOs to achieve several business goals. Nonetheless, the following list shows the most common categories that a DAO can fall into, depending on its structure, modus operandi, and technology:

1. **Protocol DAOs -** Also known as *AMM* (automated market maker) DAOs, Protocol DAOs are a type of DAO where its smart contracts provide decentralised financial services, also known as DeFi, to its users [40]. This type of DAO uses tokens as voting metrics to enforce the protocol and propose financial changes. The first Protocol DAO was *MakerDao* and is currently one of the most successful projects in the Ethereum universe. The *MakerDAO* is an Ethereum DAO that allows users to lend and borrow cryptocurrencies. Thanks to the high level of volatility on cryptocurrencies, *MakerDao* uses a stablecoin known as *Dai* to determine lending rates and repayable amounts. Stablecoins are cryptocurrencies pegged to another currency such as the U.S. dollar or to the price of gold.
2. **Investment DAOs -** DAOs that focus on democratize investments by supporting pooling for several DeFi investments [41]. Contrary to investment firms that can be elitist and non-inclusive, investment DAOs are transparent and inclusive with the purpose of being open to anyone, anywhere at anytime [40].
3. **Grant DAOs -** One of the first applications for DAOs, where the members donate funds into a grant pool and, democratically, decide the allocation of funds between several projects. These DAOs aim to help fund new DeFi and Web3 projects by building decentralized communities that operate more flexibly than traditional centralized institutions [40].
4. **Collector DAOs -** These DAOs bring together art collectors that join their funds so that the community can invest in very valuable art collectives. This way, each member of the DAO owns their share of the items [40]. Ideas like this allow small investors to come together and play an important role in the art industry.
5. **Social DAOs -** This type of DAO focuses on bringing people together by building large networks of individuals who share similar interests on a social platform. Rather than using centralized servers and storing our data for targeted Ads, Social DAOs look for a decentralized way of connecting people. One of the most famous Social DAOs is FWB, which aims to unify artists and cultural thinkers with shared values and incentives [40].

3 Architecture

With a focus on the problem presented in section 1.1, regarding public contests, this study proposes the implementation of a DAO that should follow a set of well-defined requirements in order for the system to be beneficial and to allow government institutions to see its advantages compared with the currently implemented solutions.

This section presents the solution's architecture by suggesting the technologies that can be implemented and the vision that will result in a working system. First, the presentation of the requirements allows to narrow down the several features of the system along with the user-system interactions that are greatly described by their use cases. Then the ContestDAO is divided into its main components: a blockchain infrastructure, a network of stakeholders, the Native tokens and the governance processes. Each of these subsections presents the component in greater detail by proposing several implementation options for each one.

3.1 Requirements

The current model for launching public contests starts with the publication of a "*Book of responsibilities and procedure program*" by the public organisation. In that document

are presented the rules and bylaws of the contest, which shall respect the rules defined in the "*Public Contracts Code*", Decree-Law No. 18/2008, in the Portuguese law [42]. In that first publication stands the information on how the contestants can submit their proposals, the respective deadlines of the contest, and the service's requirements. Some of these rules might not be fair to all the contestants. However, thanks to the applied centralised solution, they have no voice in the matter. These are strict and well-defined laws that will not be followed in the proposed solution, directing the focus of this study to the benefits of applying blockchain technology, respectively a DAO, to the problem presented.

The main objective of this solution is to eliminate the need for a trusted third party that works as a bridge between the two main entities of a public contest. Instead, a DAO shall take its place and eliminate any chance of mutability from that same third entity. The interactions of such a contest can be visualised in Figure 2 for a more abstract point of view. Consequently, by implementing such a solution, the contestants would be sure that the final decision, made by the public institution, is made democratically and would always be able to verify the integrity of the proposals of other contestants. Such aspects shall be backed up by non-functional requirements such as safety and integrity of data, and the system should be designed with fault tolerance, resilience to attacks and high availability.

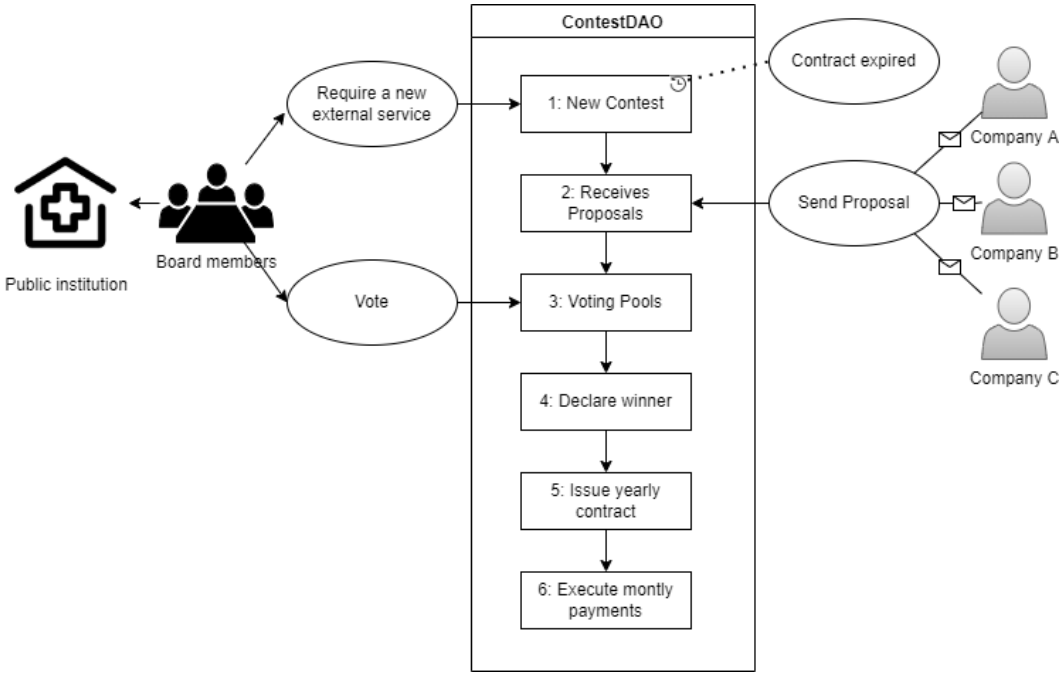

Fig. 2: Abstract ContestDAO interaction

3.2 Stakeholders

One of the main characteristics of a DAO is the importance of the stakeholders that form the community within the DAO. In the case of this solution the stakeholders are:

1. **Public institution:** Represented by their Board Members (Contests Jury), the public institution is one of the main stakeholders of this DAO. It represents the Principal on the Principal-Agent Problem, presented in section 2.3, and is responsible for requesting a service to the public. This stakeholder has the power to create proposals and vote on those proposals representing one of the decision-making members within the DAO. They are also responsible for adding new members to the DAO and choosing the winners of the contests.
2. **Contestants:** Represent a group of members with limited actions within the DAO. When a corporation desires to provide a service to fulfil a determined request published by the DAO, it must submit a proposal with its service details and cost. These stakeholders do not have a vote on the major decisions of the DAO and only benefit from it as an unbiased, transparent, immutable and secure bridge between them and the public institution.
3. **Senior Contestants:** These are contestants that have participated in several contests and therefore have won a determined amount of good reputation within the DAO community. They represent a similar type of stakeholder as the Contestants but with a higher power in the DAO. They can submit proposals and cast votes on non-contest matters.
4. **Verifiers:** In this permissionless DAO, not only members of the DAO can verify the transactions and decisions made by it. Any user can visit this information publicly for legal or other purposes. With the exception that some documents or data might be sensitive regarding the Public institution and therefore must be securely stored, a more formal request should be made to the public institution in these circumstances.
5. **ContestDAO:** The DAO itself represents a stakeholder because it can take autonomous decisions like upgrading a Contest to a Senior Contest, declaring the winner of a contest, executing monthly payments and taking other actions depending on a proposal's decision, with the responsibility of storing those decisions on the blockchain.

In the DAO, board members and Contestants' stakeholders will be differentiated by roles using role-based access control. When applying access control to smart contracts, one can define multiple roles with different permissions, allowing the potential to accomplish the previously mentioned requirements.

3.3 Use Cases

With the previous stakeholders in mind and with the intent of getting closer to more concrete functional requirements, it is essential to establish the possible user interactions with the system through the following use cases:

Public Institution Board Members Use Cases:

1. **Create Administrative Proposal -** A board member wishes to create a new proposal related to administrative affairs, such as deciding on allocating DAOs funds, adding or removing a board member, starting a new contest, or other proposals. (See Figure 3 for UML use case diagram)
2. **Vote -** When a proposal is submitted, a board member can vote on a proposal submitted by them or another member of the DAO, such as a Contestant. (See Figure 3 for UML use case diagram)

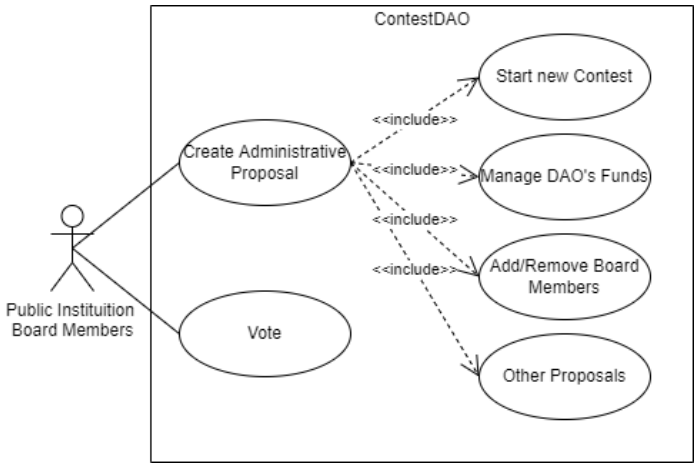

Fig. 3: Public Institution Board Members Use Case Diagram

Contestants Use Cases:

1. **Submit Proposals -** After the board members have requested a new contest, contestants can submit their proposals within a determined period. Each contestant can only submit one proposal per contest. (See Figure 4 for UML use case diagram)
2. **Verify Results -** At any moment, contestants can verify the results of a past contest that contains the classifications of the contest. (See Figure 4 for UML use case diagram)

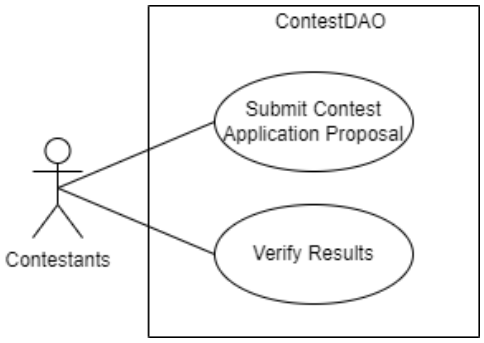

Fig. 4: Contestants Use Case Diagram

Senior Contestants Use Cases:

After earning the required reputation, a Contestant member becomes a Senior Contestant, earning new functionalities and responsibilities within the DAO.

1. **Submit Proposals -** A Senior Contestant can submit Proposals not only with the application for a specific contest, but also related with the DAO. They cannot however submit proposals to start new contests.(See Figure 5 for UML use case diagram)

2. **Vote -** Senior Contestants can vote in decision making and DAO related proposals but not on Contest Proposals.(See Figure 5 for UML use case diagram)
3. **Verify Results -** At any moment, contestants can verify the results of a past contest that contains the classifications of the contest. (See Figure 5 for UML use case diagram)

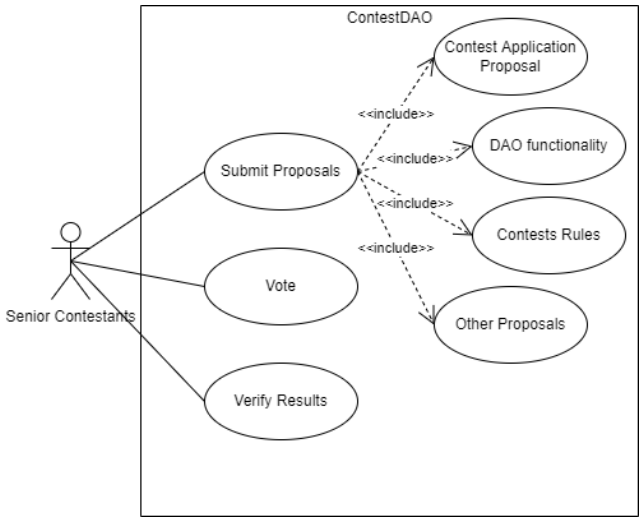

Fig. 5: Senior Contestants Use Case Diagram

3.4 Functional Requirements

Having these use cases in mind it is possible to describe a more solid representation of the following functional requirements:

1. **FR1-New Contest:** Shall start a voting pool regarding the start of a new contest for a specific service with the board members' predetermined set of requirements. Only board members shall be able to create these proposals.
2. **FR2-Submit Contest Proposal:** After the board members approve the contest, the option to submit proposals shall open for the contestants; the DAO receives all proposals and starts a voting pool on each one. The deadline for the submission after the announcement of the contest is five days.
3. **FR3-Submit Proposal:** Board and Senior Contestant members shall also be able to submit non-contest related proposals. Where only these two types of members shall be able to vote.
4. **FR4-Voting Pools:** Each Board and Senior Contestant member shall have five days to analyse a proposal and vote; otherwise, their voice will not be considered. For an administrative proposal to be valid and where N represents the total number of board members, at least N/2 + 1 must vote. In the case of Contest Proposals, only Board members can vote and the approved contest proposals win the contest.

5. **FR5-Voting:** Board members shall possess a determined quantity of governance token allowing them to cast votes on open proposals. Contestants can only receive these tokens until they become Senior members, using a share-based membership approach based on reputation. When they reach the determined level of reputation to become Senior they can cast votes on non-contest proposals.
6. **FR6-Declare Winner:** The DAO shall declare the winner, and the Board members shall publish the contest results, holding the required justifications regarding the contest's outcome.
7. **FR7-Issue Yearly contract:** The DAO shall craft and issue a yearly contract to the winner.
8. **FR8-Execute monthly payments:** The DAO shall execute monthly payments for the services provided by the winner until the end of the contract.
9. **FR9-Connect Wallet:** Each member shall be able to connect their crypto wallet to the DAO, which then identifies the member's role based on the address of the respective wallet.
10. **FR10-Verify Results:** Each member shall be able to verify the results of any past contest or transaction made by the DAO.

These requirements shall be implemented under the characteristics of a DAO. These are a blockchain infrastructure, having at least one governance token or a method to decentralize governance, a network of multiple stakeholders and a set of governance processes. The following subsections will describe these aspects individually and in more depth.

3.5 Governance Processes

Although this approach will operate in a permissionless blockchain, memberships will be managed in a more permissioned way. Members will have to submit a proposal to join the DAO, and the current board members must then approve this proposal. There will be three types of members in this solution: board members, contestants and senior contestants.

Board members and Senior Contestants will operate according to a share-based membership where shares represent direct voting power, similar to the approach applied by the *MolochDAO*, where in order to enter the MolochDAO, each new joiner is required to submit a proposal so that the group can assess if it has the necessary expertise and funds to join the community. On the other hand, Contestants will not have any voting power; they will merely be able to purchase utility tokens to submit proposals. This brings up a problem that only Senior Contestants or Board members can submit proposals. In other words, only a current member can submit a proposal. Therefore, to become a new contestant, a prospective member must convince an existing Board member to submit a proposal on his behalf. A Contestant can also submit a proposal for a new member, but since the objective is to participate in contests, other contestants might not see any advantages of adding new adversaries to the DAO. Each Contestant membership proposal must specify:

1. Organization and services.
2. Type of Contests that wishes to participate.
3. A contestant membership, to be awarded to the prospective member in exchange for the tribute.

The member that submits the proposal for a prospective member must pay a deposit in advance, of which 80% is returned regardless of the outcome of the vote. This refund method

reduces spam as the other 20% is only returned if the member is accepted. Voting remains open for five days. If the proposal reaches a majority of positive votes, then a board member calls a smart contract's function that will issue a role and membership to the candidate.

The reputation protocol of this solution will be the mechanism used to upgrade the membership of a Contestant. In order to become a Senior Contestant, a member must earn reputation. In this solution, there are two ways of winning reputation. The first is participating in contests, and the second is winning contests. Senior Contestants can also be downgraded to regular Contestants by losing reputation. This can occur if a Senior contestant submits an indecent proposal or by not casting votes on open proposals during several proposals. Indecent proposals do not have any votes from other members and are therefore cancelled.

Reputation cannot be bought and can only be given or taken by the DAO. When a Contestant meets any of the requirements above, the DAO will keep track of the member's reputation, and when a required level of reputation is reached, it will provide the member with a membership upgrade. If a member loses reputation below the required level, the reputation level goes to zero, and his membership goes back to a regular Contestant. A Senior Contestant member can never become a Board member by winning reputation. Only board members can propose and vote to add or remove another board member.

3.6 DAO Tokens

In order to apply on-chain voting, the use of governance tokens is required. This solution will implement two types of tokens: governance and utility tokens.

Governance tokens represent voting power in the DAO. These tokens will be distributed through a share-based approach between Board members and a reputation-based approach between Senior members. During the development of these tokens, Ethereum standards will be followed, and the required mechanisms to protect from double voting and other voting problems will be applied.

On the other hand, utility tokens are cryptocurrencies that have some form of utility, and in this case, they will used by all members to submit proposals; these tokens can be fungible. In terms of application, the difference is that members who only hold utility tokens can exclusively propose applications in contests.

In order to apply these two types of tokens, ERC 1155: Multi Token Standard will be applied.

3.7 Blockchain Infrastructure

During the research stage, it was possible to conclude that several systems that implement blockchain technology in business corporations or E-Procurement Platforms defend the idea that a private network such as a permissioned blockchain is the best approach. A permissioned blockchain provides better scalability and efficiency, allows access control over the nodes through valid certificates and does not require the use of governance tokens to manage the network. It looks like a suitable choice, but in this solution, the information shared in public contests is itself public, so it is unnecessary to prevent the public from seeing it. Using a private network would also introduce the problem that the organisation should control most nodes to ensure security, eliminating one of the characteristics desired by the proposed solution, decentralisation. Therefore, this solution will be constructed using a permissionless blockchain, Ethereum, with a consensus PoW protocol, employing Solidity

smart contracts that define the back-end logic, executed by an *Ethereum Virtual Machine* and a *Web3.JS* front-end framework that defines the UI logic.

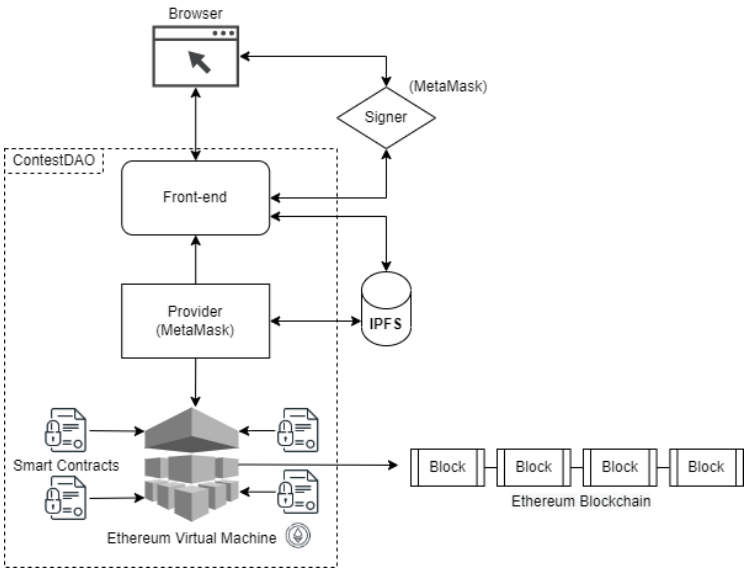

Fig. 6: ContestDAO Architecture

When the front-end is required to interact with the blockchain, it will use a provider, such as *MetaMask*, that implements a *JSON-RPC* specification. A provider will allow the client to execute read operations on the blockchain, but in order to write to the state, the transaction must be signed using a private key. For example, when a contestant queries the results of a previous contest, it executes a read transaction, but when submitting a new proposal for a contest, the DAO will require the user to sign the transaction using their private key. Only then the transaction would be passed to the blockchain and accepted by the other nodes. *MetaMask* can also be used as a signing tool by storing a user's private key in the browser, as an extension, and it is called when a signature is needed.

Alternative decentralised off-chain storage can be used to reduce the cost of storing all of the smart contract's data on the Ethereum blockchain. IPFS is a suitable option for such an outcome. It is a distributed file system for storing and accessing data. This way, it is possible to keep such a database distributed in a peer-to-peer network. In case legal needs require the DAO to store more confidential information IPFS shall be implemented with the option of encrypting such information in a secure way. IPFS can also be helpful as a replacement for centralised providers that store the solution's front-end code.

This solution will also require a development environment such as *Hardhat* to compile, test, deploy, and debug smart contracts. *Hardhat* uses *mainnet forking* to simulate having the same state as the Ethereum mainnet, but as a local development network. The lifecycle of the solution's smart contracts development shall follow these three steps:

1. **Creation:** The practical process of coding the designed contracts according with the requirements.

2. **Testing:** This step shall be done in parallel with the last. *Hardhat* facilitates this task by having multiple functional testing plugins.
3. **Deployment:** In this final step the code is compiled from Solidity to bytecode in order to be executed by the *EVM*.

4 Evaluation

The evaluation methodology describes how the proposed solution will be tested in a way that it can be validated.

The start of the methodology will be an evaluation against the current applied solution of Public Contests. First, aspects such as the performance and fault tolerance shall be considered between the two solutions. Also, all the considerations between a centralised and decentralised approach shall be evaluated, along with the benefits that implementing blockchain technology would bring by implementing the proposed solution. To better compare the two systems, metrics of throughput, latency, fault tolerance and security need to be collected.

The proposed solution shall also suffer an evaluation of a future real-world implementation to demonstrate the challenges of implementing such a solution, justified with the positives and negatives aspects of such an implementation. This evaluation can include the legal barriers that such an implementation would have to face.

Another important aspect of this methodology is the smart contract's code quality. When a smart contract's code is deployed into the blockchain, it is immutable and stays there permanently. There are several tools for auditing smart contracts that should be used to assure their security and quality before deployment.

As a static analysis tool, a tool such as Slither could be applied. Static analysis implies an automated source code analysis without executing the application. For symbolic execution, Manticore can be applied. Symbolic execution is a way to test the smart contract by analysing the consequences that inputs cause each part of a program to execute. Also, fuzz testing is essential since these tests attempt to find vulnerable software bugs by feeding random data inputs into their tests. Echidna can be an excellent tool for fuzz testing. All these tools are from the Trial of Bits group but other tools might be used to accomplish the verification of these tests.

Monitoring the code will also be necessary, so Tenderly could be an adequate tool to set up alerts and control the monitoring of the smart contracts.

Finally, this solution shall also be evaluated against the framework of a DAO. In order to prove that such a DApp is indeed a DAO, a formal evaluation of the system's features and characteristics shall identify it as a DAO. In order to do so, not only the technological aspects shall be considered but also aspects such as the target community, the governance methods, its purposes and legal structure. To better accomplish this final step, it is essential to compare our solution against currently implemented examples of DAOs.

5 Schedule of Future Work

Future work is scheduled as follows:

Work/Month	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Solidity & Tools	X	X					
DAOs Framework Analyses	X	X					
Implementation		X	X	X	X		
Testing					X	X	
Thesis	X	X	X	X	X	X	X

6 Conclusion

With the analysis of a practice used by most democratic governments to select the best private institutions for a service and the research around the benefits of blockchain technology, this study ultimately found a solution that could improve such a practice.

With the understatement of the present problems from both sides of these public contests and following deep research into blockchain and smart contract technologies, this study started to find solutions to those problems in the characteristics of DAOs.

The research of several blockchains and smart contract development topics led to a more detailed architecture of a possible solution that could be implemented to such a problem.

Such an architecture proposes an Ethereum permissionless blockchain as an infrastructure, a list of stakeholders composed by the Public institution, the Contestants, Senior Contestants with escalated privileges, outside Verifiers and the DAO itself, proposes the use of governance tokens as a way of expressing a vote and reputation, and a set of governance processes that represent the core of a DAO. Each of these components shall be well evaluated according to the methods presented in section 4.

To visualise this solution, it was essential to understand the current applied system, where such contests require a third private entity to serve as a connection point at both ends of the contest. In this stage, the study focused on the contests done by public hospitals where in particular, Vortal is an example of such a third entity. Important documents such as "CONCURSO PÚBLICO N.º 1-2.0095/22" and the respective electronic platform: VortalHEALTH, were significant contributions to the problem's analysis.

In conclusion, this study intends to demonstrate that essential records of public affairs and, in particular, public contests are matters that genuinely benefit from the use of blockchain technologies. These contests are public and, therefore, shall be available to the public in the most secure and transparent way possible. Furthermore, this study defends that the entities participating in such contests shall have a voice in how the contests are organised—concluding that decentralised autonomous organisations provide the security, transparency and decentralised governance to fulfil such changes.

References

1. Wikipedia, "Principal–agent problem," *Wikipedia*, 2022, last accessed 12 April 2022. [Online]. Available: https://en.wikipedia.org/wiki/Principal-agent_problem
2. M. E. Peck, "Blockchains: How they work and why they'll change the world," *IEEE Spectrum*, vol. 54, no. 10, pp. 26–35, 2017.
3. S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *bitcoin.org*, pp. 3–4, 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf
4. Ethereum, "Introduction to smart contracts," *soliditylang*, 2021, last accessed 29 March 2022. [Online]. Available: https://docs.soliditylang.org/en/v0.8.13/introduction-to-smart-contracts.html#storage-example
5. V. Buterin, "Daos, dacs, das and more: An incomplete terminology guide," *Ethereum Foundation Blog*, 2014, last accessed 29 March 2022. [Online]. Available: https://blog.ethereum.org/category/research-and-development/
6. The Investopedia Team, "What is the principal-agent problem?" *INVESTOPEDIA*, April 2021, last accessed 12 April 2022. [Online]. Available: https://www.investopedia.com/terms/p/principal-agent-problem.asp
7. Vortal, "Research & innovation," 2022, last accessed 17 April 2022. [Online]. Available: https://www.vortal.biz/innovation/
8. T. Nodehi, A. Zutshi, and A. Grilo, "A blockchain based architecture for fulfilling the needs of an e-procurement platform," *Proceedings of the 5th NA International Conference on Industrial Engineering and Operations Management Detroit, Michigan, USA*, August 2020.
9. M. Herlihy, "Blockchains from a distributed computing perspective," *Communications of the ACM*, vol. 62, no. 2, pp. 80–85, Feb. 2019.
10. J. Polge, J. Robert, and Y. L. Traon, "Permissioned blockchain frameworks in the industry: A comparison," *ICT Express*, vol. 7, pp. 229–233, June 2021, last accessed 18 April 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405959520301909#tbl1
11. D. Mingxiao, M. Xiaofeng, Z. Zhe, W. Xiangwei, and C. Qijun, "A review on consensus algorithm of blockchain," in *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2017, pp. 2567–2572.
12. A. Agarwal, "Proof-of-work (pow)," 2022, last accessed 21 March 2022. [Online]. Available: https://ethereum.org/en/developers/docs/consensus-mechanisms/pow/
13. digiconomist, "Bitcoin energy consumption index," 2022, last accessed 21 March 2022. [Online]. Available: https://digiconomist.net/bitcoin-energy-consumption
14. P. Wackerow, "Proof-of-stake (pos)," Jan. 2022, last accessed 21 March 2022. [Online]. Available: https://ethereum.org/en/developers/docs/consensus-mechanisms/pos/
15. S. D. Angelis, L. Aniello1, R. Baldoni, F. Lombardi1, A. Margheri, and V. Sassone, "PBFT vs proof-of-authority: Applying the CAP theorem to permissioned blockchain," *University of Southampton*, 2018, last accessed 19 April 2022. [Online]. Available: https://iris.uniroma1.it/retrieve/handle/11573/1337256/1302941/DeAngelis_PBFT_2018.pdf
16. Coinbase, "What is a fork?" *Coinbase*, March 2022, last accessed 16 May 2022. [Online]. Available: https://www.coinbase.com/learn/crypto-basics/what-is-a-fork
17. C. F. Institute, "Hard fork," *Corporate Finance Institute*, 2022, last accessed 15 April 2022. [Online]. Available: https://corporatefinanceinstitute.com/resources/knowledge/other/hard-fork/
18. M. B. Nicola Atzei and T. Cimoli, "A survey of attacks on ethereum smart contracts," *Cryptology ePrint Archive*, p. 1, 2017. [Online]. Available: https://eprint.iacr.org/2016/1007.pdf
19. D. G. W. Andreas M. Antonopoulos, *Mastering Ethereum*, 1st ed. CA: O'Reilly Media Inc, 2019.
20. Ethereum.org, "Introduction to smart contracts," *Ethereum.org*, 2022, last accessed 29 March 2022. [Online]. Available: https://ethereum.org/en/developers/docs/smart-contracts/
21. E. Staff, "Oracles," *Ethereum.org*, 2022, last accessed 29 March 2022. [Online]. Available: https://ethereum.org/en/developers/docs/oracles/

22. N. Mudge, "Eip-2535: Diamonds, multi-facet proxy," *Ethereum Improvement Proposals*, 2020, last accessed 29 March 2022. [Online]. Available: https://eips.ethereum.org/EIPS/eip-2535#motivation
23. J. Frankenfield, "Gas (ethereum)," 2021, last accessed 24 March 2022. [Online]. Available: https://www.investopedia.com/terms/g/gas-ethereum.asp
24. B. Hu, Z. Zhang, J. Liu, Y. Liu, J. Yin, R. Lu, and X. Lin, "A comprehensive survey on smart contract construction and execution: paradigms, tools, and systems," *Patterns*, vol. 2, 2021. [Online]. Available: https://doi.org/10.1016/j.patter.2020.100179
25. T. Bililies, "Centralization versus decentralization: what's right for you?" *AlixPartners*, 2016, last accessed 8 April 2022. [Online]. Available: https://www.alixpartners.com/media/14446/ap_centralization_versus_decentralization_apr_2016.pdf
26. W. Cai, Z. Wang, J. B.Ernst, Z. Hong, C. Feng, and V. C. M. Leung, "Decentralized applications: The blockchain-empowered software system," *IEEE*, vol. 6, 2018, last accessed 05 March 2022. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=\&arnumber=8466786
27. A. Ivanovs, "What is a dao? examples of dao crypto projects," *Geekflare*, March 2022, last accessed 09 April 2022. [Online]. Available: https://geekflare.com/finance/dao-crypto-projects/
28. A. Sims, "Blockchain and decentralised autonomous organisations (daos): The evolution of companies?" *University of Auckland Business School*, November 2019, last accessed 14 April 2022. [Online]. Available: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3524674
29. MakerDAO, "What is mkr?" *Medium*, September 2015, last accessed 14 April 2022. [Online]. Available: https://medium.com/@MakerDAO/what-is-mkr-e6915d5ca1b3
30. Cryptopedia Staff, "What is a decentralized autonomous organization (dao)?" *Cryptopedia*, March 2022, last accessed 12 April 2022. [Online]. Available: https://www.gemini.com/cryptopedia/decentralized-autonomous-organization-dao
31. Binance Academy, "How to create a dao?" *Binance Academy*, February 2022, last accessed 11 April 2022. [Online]. Available: https://academy.binance.com/en/articles/how-to-create-a-dao
32. Cryptopedia Staff, "How dao frameworks can facilitate defi governance," *Cryptopedia*, November 2021, last accessed 12 April 2022. [Online]. Available: https://www.gemini.com/cryptopedia/dao-crypto-decentralized-governance-blockchain-governance
33. Tally Contributors, "On chain vs. off chain voting," *Tally*, June 2021, last accessed 13 April 2022. [Online]. Available: https://wiki.tally.xyz/docs/on-chain-vs-off-chain-voting
34. E. Arsenault, "Voting options in daos," *Medium*, December 2020, last accessed 12 April 2022. [Online]. Available: https://medium.com/daostack/voting-options-in-daos-b86e5c69a3e3
35. M. Field, "Holographic consensus—part 1," *Medium*, November 2018, last accessed 12 April 2022. [Online]. Available: https://medium.com/daostack/holographic-consensus-part-1-116a73ba1e1c
36. J. Emmett, "Conviction voting: A novel continuous decision making alternative to governance," *Medium*, November 2021, last accessed 13 April 2022. [Online]. Available: https://medium.com/commonsstack/conviction-voting-a-novel-continuous-decision-making-alternative-to-governance-62e215ad2b3d
37. J. Potts, E. Rennie, and J. Goldenfein, "Blockchains and the crypto city," *it-Information Technology*, vol. 59, no. 6, 2017.
38. M. Graham, "Crypto airdrops explained," *Boardroom*, March 2022, last accessed 14 April 2022. [Online]. Available: https://boardroom.tv/airdrop-crypto-nft-explained/
39. J. Ma, "Snapshot," *Binance Academy*, 2022, last accessed 14 April 2022. [Online]. Available: https://academy.binance.com/en/glossary/snapshot
40. Ledger, "Your DAO guide - the most important DAO categories defining the space," *Ledger*, December 2021, last accessed 11 April 2022. [Online]. Available: https://www.ledger.com/academy/your-dao-guide
41. The Economic Times, "DAO explained - types, key characteristics & flipsides," *The Economic Times*, January 2022, last accessed 11 April 2022. [Online]. Available: https://economictimes.indiatimes.com/markets/cryptocurrency/dao-explained-types-key-characteristics-flipsides/articleshow/89040222.cms

42. Ministério das Obras Públicas, Transportes e Comunicações, "Decreto-lei n.º 18/2008," *Diário da República n.º 20/2008*, 2008. [Online]. Available: https://dre.pt/dre/detalhe/decreto-lei/18-2008-248178