

Politecnico di Torino

Politecnico di Torino

Master's Degree in Cybersecurity

Academic Year 2025/2026

Graduation Session October 2026

CivicDAO: Design and Implementation of a Neighborhood DAO for the Participatory Governance of Common Goods

Supervisor:

Prof. Danilo Bazzanella

Candidate:

Andrea Bobò

Co-supervisors:

Dr. Alfredo Favenza

Dr. Silvio Meneguzzo

Abstract

Digital participatory platforms have shown that citizens can be involved at scale in decisions about the places they inhabit, but participation of this kind remains non-binding. Indeed nothing in their operation obliges an administration to honor the outcome. Decentralized autonomous organizations (DAOs) offer the binding and verifiable execution that such platforms lack. On the other hand they carry well-documented pathologies, namely plutocratic concentration of voting power, chronic voter apathy, exposure to governance attacks and an uncertain legal status. Existing public-sector applications of DAOs leave open the question that is decisive for a genuinely civic use: through which on-chain governance mechanisms can a legally responsible public body remain within the process as a guarantor of compliance, without that control collapsing into a re-centralization that empties citizen participation of its substance?

This thesis answers this question by designing and partially implementing CivicDAO, a neighborhood-scale DAO for participatory governance of common goods. Its architecture is hybrid: on-chain governance is paired with a third-sector legal entity that represents the community toward third parties, so that decentralization and legal accountability coexist rather than exclude one another. Membership is grounded in verified residency rather than in capital and expressed by a non-transferable citizenship token. This token enforces one person, one vote in the private flow governed by the community alone and an equal credit budget in the quadratic voting of the public participatory-budgeting flow. The municipal role is confined to procedural acts and bounded by two complementary mechanisms: permissionless time-based state transitions, which prevent strategic inertia and a codified obligation to state reasons, which turns any non-execution into a visible and attributable act. Participation is sustained by a closed-loop local credit and by a non-transferable, decaying reputation that never weighs a vote, while civic actions performed off-chain are verified optimistically, subject to challenge before a randomly drawn jury.

The design is realized in Solidity, building on the OpenZeppelin libraries and the Foundry framework, and is exercised on a local Anvil node, covering the identity, governance and economy layers, while the infrastructural layer is left to future work. The resulting mechanisms do not dissolve the tension between accountability and decentralization. Rather, they relocate it to a small number of explicit and verifiable points, principally the residency attestation at onboarding and the seam between the on-chain and the legal order.

Keywords: blockchain, decentralized autonomous organization, civic technology, participatory budgeting, soulbound token, quadratic voting, smart contracts, Solidity.

Table of Contents

List of Figures VI

List of Tables VII

1 Introduction 1
- 1.1 Research Questions . . . 3
- 1.2 Structure of the Thesis . . . 3

2 Blockchain 4
- 2.1 Blockchain and Distributed Ledgers . . . 4
- 2.2 Cryptography for Blockchain . . . 5
- 2.3 Structure of the Blockchain . . . 7
- 2.4 Blockchain Classification . . . 8
- 2.5 Consensus Protocols . . . 9
- 2.6 Blockchain 2.0: Ethereum and Smart Contracts . . . 11
- 2.7 Blockchain Security . . . 12

3 State of the art: DAO - Decentralized Autonomous Organization 14
- 3.1 Introduction to DAO . . . 14
- 3.2 Core Characteristics . . . 15
- 3.3 Structure of a DAO . . . 17
  - 3.3.1 Governance . . . 17
  - 3.3.2 Governance Process . . . 18
  - 3.3.3 Treasury . . . 18
- 3.4 Voting Mechanisms . . . 19
- 3.5 Token Standards . . . 21
- 3.6 The Regulatory Classification of Tokens: from FINMA to MiCA . . . 22
  - 3.6.1 FINMA's Classification . . . 23
  - 3.6.2 The Crypto-Asset Categories of the MiCA Regulation . . . 23
- 3.7 DAO Classification . . . 24
- 3.8 Problems and limitations . . . 25

3.8.1 The vulnerability of code and the limits of automated execution 27
3.8.2 Governance attacks and the exploitation of the voting mechanism . . . 27
3.8.3 Plutocratic concentration of voting power . . . 28
3.8.4 Voter apathy and low participation . . . 28
3.8.5 Delegation is not enough . . . 29
3.8.6 The uncertain legal status of DAOs . . . 29
3.9 Non-blockchain-based civic platforms . . . 29
3.10 Civic DAOs and public-sector applications . . . 30

4 CivicDAO Design 32
4.1 Objectives . . . 32
4.2 Building Blocks of the DAO . . . 32
4.2.1 Purpose . . . 32
4.2.2 Use Case . . . 33
4.2.3 Legislation . . . 33
4.3 Organizational Bodies . . . 37
4.3.1 Citizens' Community . . . 37
4.3.2 The Municipality . . . 38
4.3.3 Local businesses . . . 38
4.3.4 Reputational Roles . . . 38
4.3.5 Project teams . . . 39
4.3.6 Jury drawn by lot . . . 39
4.3.7 Legal interface . . . 39
4.3.8 Security Council . . . 39
4.4 Decision-Making . . . 40
4.4.1 The decision-making process . . . 40
4.4.2 Private Flow Process . . . 40
4.4.3 Public Flow Process: Participatory Budgeting . . . 43
4.5 Tokens . . . 45
4.5.1 Legal framework . . . 45
4.5.2 Citizenship Token . . . 46
4.5.3 Membership token business . . . 46
4.5.4 Community Credit . . . 46
4.5.5 Sustainability NFTs and Certificates . . . 47
4.6 Treasury . . . 47
4.6.1 Treasury Assets and Their Use . . . 48
4.6.2 Signatories and Key Management . . . 49
4.7 Incentives . . . 49
4.7.1 The Hooks for Incentivization . . . 49
4.8 Communication . . . 49

4.8.1 On-chain and Off-chain Channels 50

5 Implementation 51
5.1 Technology Stack 51
5.2 General Architecture 52
5.3 Identity and Residency 52
5.3.1 On-boarding and Token minting flow 53
5.4 CitizenshipSBT.sol 56
5.5 Governance 59
5.5.1 ParticipatoryBudget.sol 59
5.5.2 CivicProject.sol 62
5.5.3 The real closed-loop economy 67
5.5.4 Reputation 70
5.5.5 Boundary of the implementation 72

6 Results 74
6.1 Answering the Research Questions 75
6.2 Limitations 77
6.2.1 Residual centralization at the onboarding gateway 77
6.2.2 Sybil resistance is only as strong as the identity layer 77
6.2.3 Excluding socio-technical prerequisites 78
6.2.4 Optimistic verification presupposes an active and honest minority 78
6.2.5 The hybrid on-chain / off-chain gap 79
6.2.6 The irreducible tension of incentive design 79

7 Conclusions and Future Work 80
7.1 Summary of the Work 80
7.2 Addressed Problems and Adopted Solutions 81
7.3 Future Work 84
7.3.1 Completing the Architecture 84
7.3.2 Self-Sovereign Residency through IT-Wallet and the EUDI Wallet 84
7.3.3 Conviction Voting for Participatory Budgeting 85
7.3.4 Ballot Secrecy and Collusion Resistance 85
7.3.5 Governance of Parameters and Upgradeability 85
7.3.6 Deployment, Security Assurance and Empirical Validation . 86
7.3.7 Federation of Neighborhood DAOs 86

Bibliography 87

List of Figures

2.1 Structure of the blockchain. 7
2.2 Structure of a Merkle tree with four data blocks. 8

3.1 The three pillars of DAO governance. 17
3.2 DAO governance process. 18

4.1 Public flow: participatory budgeting process. 34
4.2 Private flow: legal wrapper, DAO governance and community credit circuit. 35
4.3 Organizational bodies of the CivicDAO and their relationships. . . . 37
4.4 States of the private flow. 42
4.5 States of the Public flow. 44

5.1 On-boarding flow of Citizenship token 54

6.1 How the design and its implementation answer the research question. 75

List of Tables

2.1 Comparison of the main consensus protocols. 10

3.1 Main differences between traditional organizations and DAOs. . . . 16
3.2 Comparison of the main Ethereum token standards. 22
3.3 DAO classification. 26

5.1 CivicDAO contract architecture organized by layers. 52

7.1 Problems identified in the state of the art and solutions adopted in CivicDAO. 83

Chapter 1

Introduction

Participatory democracy has long searched for instruments through which citizens can take part in the decisions concerning the places they inhabit and the digitalization of public administration has made the ambition of a structured, continuous and transparent participation more concrete. Participatory budgeting, in which a community deliberates directly on the allocation of a share of public resources, is perhaps the clearest expression of this ambition.

The most mature digital platforms for participatory democracy are non-blockchain civic platforms, such as Decidim and Consul, which enable citizens, associations, and public administrations to discuss, propose and vote on ideas or public policies. Precisely because they operate at scale and are institutionalized, they reveal a structural limitation. The empirical literature on their adoption shows that they broaden participation without redistributing effective decision-making power: they organize information, gather proposals and record votes, yet nothing in their operation prevents an administration from ignoring the outcome or postponing its implementation indefinitely. They offer participation, but not binding and verifiable procedural guarantees.

Blockchain technology and the decentralized autonomous organizations (DAOs) built upon it, promise exactly what these platforms lack, namely a tamper-evident and publicly verifiable record of every decision, coupled with the possibility of automatically executing its outcome, without depending on an intermediary. This promise, however, is not free of tensions. As Section 3.8 documents, DAOs carry their own well-established limitations. Token-based governance tends toward a plutocratic concentration of voting power, participation is chronically low, voter apathy persists even when transaction costs are removed and the legal standing of DAOs remains uncertain, leaving open the question of who is accountable when their decisions cause harm.

A more specific gap emerges when DAOs are considered for use in the public sector. The existing contributions in this direction, examined in Section 3.10, either automate a single administrative process or propose DAO mechanisms as a general set of transferable tools, without addressing the public domain in a specific way. None of them confronts the problem: how a legally responsible public body can remain within the governance process, retaining a role of compliance control, without that control collapsing into a re-centralization that empties citizen participation of its substance. Legal accountability and conformity with the law seem to require a supervising authority, whereas the promise of a DAO lies in removing precisely such an authority.

This work addresses that problem by designing and partially implementing CivicDAO, a decentralized autonomous organization operating at the scale of a neighborhood and aimed at coordinating and incentivizing citizen participation in the care of common goods. The design adopts a hybrid architecture, in which decentralized on-chain governance is paired with a traditional legal wrapper that represents the community toward third parties, so that decentralization and legal accountability coexist rather than exclude one another.It grounds the legitimacy of participation in verified residency rather than in capital: membership is expressed by a non-transferable citizenship token that realizes a one-person-one-vote principle, structurally neutralizing the plutocratic drift of token-weighted governance. It separates two distinct flows: a private flow, through which the community directly governs its own civic actions and a public flow, through which participatory budgeting takes place while the municipality retains a bounded, non-discretionary role of compliance control. It sustains participation over time through a closed-loop local economy and a reputation-based incentive layer and it verifies the execution of civic actions through an optimistic scheme backed by a randomly-drawn jury.

The components of the system are derived from the problem itself and consolidated in the design of Chapter 4. The resulting design is then realized in Solidity, using the Foundry framework, OpenZeppelin libraries and exercised in Anvil, a simulated environment. The implementation described in Chapter 5 covers three of the four architectural layers, namely identity, governance, and economy, while the infrastructural layer and a number of refinements are left, by design, as future work.

1.1 Research Questions

The overarching question that guides the work is the following.

RQ: Through which on-chain governance mechanisms can a civic DAO keep a legally responsible public body within the process as a guarantor of compliance, without that compliance control degenerating into a centralization that nullifies citizen participation?

This question is articulated into three sub-questions:

- **SQ1**: How can the legitimacy of participation be grounded in residency rather than in capital, so as to neutralize the plutocratic concentration of voting power while preserving on-chain verifiability?
- **SQ2**: How can a legally responsible public body be kept within the governance process and how can the community be given legal standing toward third parties, without granting the administration a discretionary veto over the merit of proposals?
- **SQ3**: How can participation be sustained over time against voter apathy, through incentive mechanisms that avoid the perverse effects to which incentive design is notoriously prone?

1.2 Structure of the Thesis

Chapter 2 introduces the technological foundations, presenting blockchain and distributed ledgers, the cryptographic primitives on which their guarantees rest, and the smart-contract model introduced by Ethereum. Chapter 3 focuses on decentralized autonomous organizations: their characteristics, governance structures, voting mechanisms and token standards, together with a critical review of their limitations and of the existing civic and public-sector applications. Chapter 4 derives the requirements of CivicDAO from the research problem and presents its design in full: bodies, decision-making flows, tokens, treasury, incentives and communication. Chapter 5 describes the implementation of the on-chain contracts delimitating the boundary of what has been realized and chapter 6 summarizes and discusses the limitations of the proposed model. Chapter 7 draws the conclusions, outlining the directions for future development.

Chapter 2

Blockchain

2.1 Blockchain and Distributed Ledgers

Blockchain is a distributed database technology, also known as Distributed Ledger Technology (DLT), which cannot be modified by individual users or groups of users [1]. More specifically, DLT is a technology that enables transactions and data to be recorded in a decentralized manner across a network of distributed nodes, eliminating the need for a central authority that controls the ledger. In the context of blockchain, the main objective is to eliminate the central guaranteeing authority.

A generic DLT supports four different operations: Create, Retrieve, Update, and Delete. Blockchain represents a specific type of DLT that allows only two operations: Create and Retrieve. It is therefore not possible to modify or delete any data once it has been recorded.

The main properties of a blockchain are the following [2]:

- **Decentralization**: the defining feature of blockchain is its decentralized nature. This means that the digital ledger is not governed by a single central authority but is instead distributed across the many nodes of the network. Such a model makes the blockchain resistant to censorship and tampering, since it is not subject to the control of any single entity or organization.

- **Immutability**: another fundamental property of blockchain is the immutability of its data. Once a transaction has been recorded on the blockchain and included within a block, it becomes permanent and cannot be altered or deleted without the consent of the majority of the network's participants. This ensures the integrity and reliability of the information stored on the chain.

- **Verifiability and transparency**: blockchain offers a high degree of verifiability and transparency. All transactions are publicly accessible and permanently recorded from the very first block, known as the genesis block. This means that anyone can inspect the entire transaction history and verify the authenticity and integrity of the data without having to rely on a trusted intermediary.
- **Security and anonymity**: cryptography plays a crucial role in blockchain, safeguarding both the security and the anonymity of data. Transactions are encrypted, and network participants are identified through pseudonymous addresses, thereby preserving users' privacy and anonymity.

.

2.2 Cryptography for Blockchain

Cryptography is the foundation on which the security properties of blockchain rest. It is what guarantees the integrity of the data, the authenticity of the transactions and the privacy of the participants [3]. Three cryptographic building blocks are particularly relevant: hash functions, public-key cryptography and zero-knowledge proofs.

Hash functions A cryptographic hash function is an algorithm that takes an input of arbitrary length and produces a fixed-length output, called the hash or digest. Hash functions are the mechanism that links blocks together and that guarantees the tamper-evidence of the ledger. To be suitable for cryptographic use, a hash function must satisfy the following properties:

- **Determinism**: the same input always produces the same output.
- **Efficiency**: the digest can be computed quickly for any input.
- **Pre-image resistance**: given a hash value, it is computationally infeasible to recover the original input.
- **Second pre-image resistance**: given an input, it is infeasible to find a different input that produces the same hash.
- **Collision resistance**: it is infeasible to find any two distinct inputs that produce the same hash.
- **Avalanche effect**: a minimal change in the input produces a drastically different output.

Hash functions chain each block to the previous one through the hash stored in its header, they summarize the transactions of a block by means of a Merkle tree and they underpin Proof-of-Work consensus, where miners search for an input that yields a hash satisfying a given difficulty target. Widely used hash functions include SHA-256, adopted by Bitcoin and Keccak-256, used by Ethereum.

Public-Key Cryptography Public-key cryptography, also known as asymmetric cryptography, is based on a pair of mathematically related keys: a public key, which can be freely shared and a private key, which must be kept secret by its owner. The two keys are generated together and what is encrypted (or signed) with one can only be processed with the other. Deriving the private key from the public key is computationally infeasible. This asymmetry removes the need to share a secret key in advance, which is the main limitation of symmetric cryptography.

In the blockchain context, public-key cryptography serves two essential purposes:

- **Identity and addressing**: each user is identified by their public key (or by an address derived from it), which acts as a pseudonymous identifier within the network. Ownership of the corresponding private key proves ownership of the associated funds.
- **Digital signatures**: a user signs it with their private key to authorize a transaction. Any other node can then verify the signature using the sender's public key. This mechanism provides three fundamental security properties:
  - **Authentication**: it proves that the transaction was created by the holder of the private key.
  - **Integrity**: any modification of the signed data invalidates the signature.
  - **Non-repudiation**: the signer cannot later deny having authorized the transaction in a legal court.

Most blockchains rely on ECC (elliptic-curve cryptography) for this purpose, as it offers the same level of security as older schemes while using considerably shorter keys, which reduces both storage and computational overhead. Bitcoin and Ethereum, for example, adopt the Elliptic Curve Digital Signature Algorithm (ECDSA) over the secp256k1 curve.

Zero-Knowledge Proofs A zero-knowledge proof (ZKP) is a cryptographic protocol that allows one party, the prover, to convince another party, the verifier, that a given statement is true, without revealing any information beyond the validity of the statement itself [3]. For instance, a user could prove that they possess sufficient funds to make a payment without disclosing their actual balance,

or prove knowledge of a secret without revealing the secret. They enable privacy for transactions, in which the validity of a transaction can be verified without exposing its sensitive details such as sender, receiver, or amount.

2.3 Structure of the Blockchain

Blocks are composed by transactions and each block is linked to its predecessor through a cryptographic hash. Iterating this procedure forms the chain and guarantees the integrity of the previous block, all the way back to the genesis block, the first block of the blockchain as shown in Figure 2.1. It is possible for two blocks to be produced simultaneously, which causes a fork in the blockchain's chain. Every blockchain has a specific algorithm for handling forks.

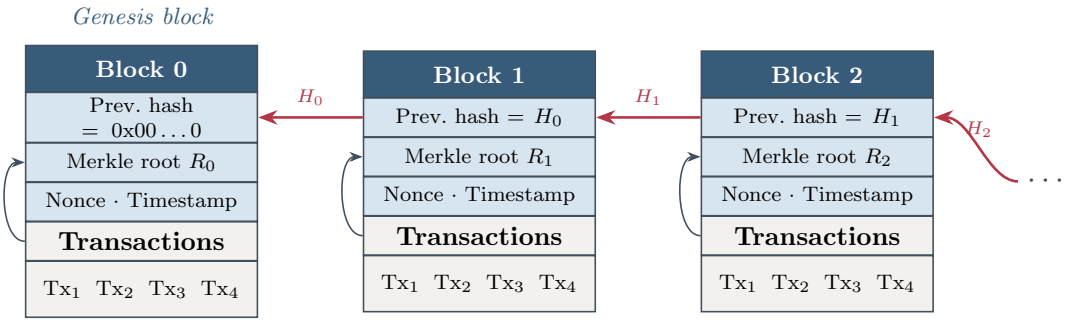


Figure 2.1: Structure of the blockchain.

The structure relies on the Merkle Tree (or Hash Tree) data structure. Its structure is built from the bottom up. At the lowest level lie the original data blocks (L1, L2, L3, L4 in the Figure 2.2), which in the blockchain context typically correspond to transactions. Each data block is passed through a cryptographic hash function to produce a leaf node (Hash 0-0 = hash(L1), Hash 0-1 = hash(L2), and so on). The leaf nodes are then combined in pairs: the hashes of two child nodes are concatenated and hashed again to form an internal (parent) node. This pairwise process is repeated up the tree until a single value, the Top Hash or Merkle root, is obtained, which summarizes the entire underlying dataset.

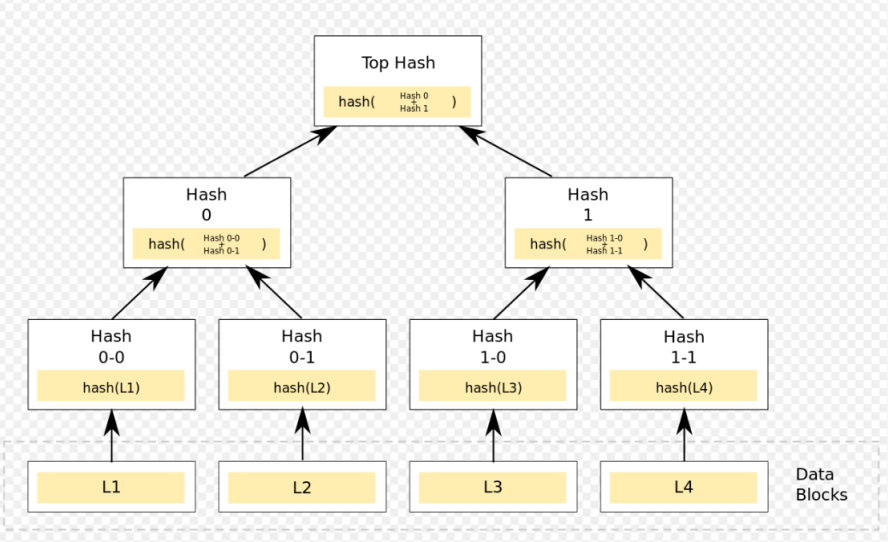


Figure 2.2: Structure of a Merkle tree with four data blocks.

2.4 Blockchain Classification

Blockchains can be broadly classified into three categories, which differ in who is allowed to join the network and to read or write its contents [2]:

- **Public blockchains**: in a permissionless blockchain anyone can join the network without prior authorization, so any participant is free to read the blocks, submit transactions and take part in the consensus process. This openness maximizes decentralization and transparency, but it also requires robust consensus mechanisms (such as Proof of Work or Proof of Stake) to secure the network in the presence of potentially malicious participants. Bitcoin and Ethereum are the best-known examples.
- **Hybrid blockchains**: in a hybrid/consortium blockchain the access to the network is controlled. Prospective participants must be vetted before joining, but once admitted they are able to read the blocks. This model represents a middle solution between fully open and fully closed systems, typically governed by a group of organizations. It offers greater privacy and efficiency than a public blockchain while retaining a degree of decentralization.
- **Private blockchains**: in a permissioned blockchain, a central authority accepts or rejects each request to participate. Access to reading and writing operations is therefore restricted to authorized members. This arrangement provides the highest level of control, privacy and performance, at the cost of reduced decentralization. It is commonly adopted in enterprise and institutional

contexts.

2.5 Consensus Protocols

A crucial challenge that every blockchain must address is the process of reaching consensus on the addition of new blocks to the shared ledger. Thanks to its distributed nature and its hash-based structure, transactions already stored on the blockchain are extremely difficult to alter fraudulently. The most delicate point, therefore, is the insertion of new blocks. This step is fundamental to preserving the integrity and security of the system, because if a malicious actor were able to add fraudulent transactions, the reliability and usefulness of the entire blockchain would be compromised.

Consensus protocols are complex algorithms that allow unknown and mutually distrusting participants to agree on which blocks are valid and may be appended to the chain. Consensus algorithms define a set of rules that all nodes must follow and leverage the participants' shared interest in keeping the ledger honest and reliable. This is especially important in a decentralized network, where no trusted third party exists to arbitrate transactions or resolve disputes. Several algorithms have been developed to meet these goals. The most widely adopted ones are described below and their main differences are summarized in Table 2.1.

- **Proof of Work (PoW)**: this protocol requires participants to solve a computationally expensive problem that is nonetheless quick to verify. Miners must find a nonce value that, when combined with the block data, produces a hash satisfying a given difficulty criterion. Since the correct nonce cannot be predicted, miners must perform a large number of hashing attempts until a valid solution is found [1]. The first miner to succeed propagates the new block to the network, where the other nodes verify that its hash meets the required difficulty. Once the block is accepted and appended to the chain, the miner is rewarded, typically in cryptocurrency. Compromising a PoW system would require controlling at least 51% of the network's computing power. Although PoW is highly effective at ensuring security and integrity, its main drawback is very high energy consumption, since the massive number of computations demands substantial electricity.

- **Proof of Stake (PoS)**: in PoS, validator nodes lock up a certain amount of cryptocurrency or tokens (the stake) and validators are selected according to the size of this stake rather than their computational power [4]. Consensus is thus achieved on the basis of the participants' economic commitment. When a node is chosen to validate a block, it receives the associated transaction fees,

Table 2.1: Comparison of the main consensus protocols.

	Proof of Work (PoW)	**Proof of Stake (PoS)**	**Proof of Authority (PoA)**
What is at stake	Computational power	Cryptocurrency or tokens locked as stake	Validator reputation and disclosed identity
Validator selection	First node to find a nonce whose block hash meets the difficulty target	Nodes chosen in proportion to the size of their stake	Small set of pre-approved, identifiable validators
Energy consumption	Very high	Considerably lower than PoW	Very low
Decentralization	High	High	Limited (few known validators)
Reward / penalty	Block reward to the successful miner	Fees to the validator and loss of stake for an invalid block	Cap on blocks for each validator to discourage abuse
Main attack vector	Control of $\geq 51\%$ of the computing power	Control of $\geq 51\%$ of the total stake	Corruption of the few identifiable validators
Typical use	Public blockchains (e.g. Bitcoin)	Public blockchains (e.g. Ethereum)	Private / consortium blockchains

but if it attempts to add an invalid block, it loses its stake. Compared with PoW, PoS offers considerably lower computational and energy requirements.

- **Proof of Authority (PoA)**: in PoA, what is at stake is not currency, as in PoS, but reputation. Validators must publicly disclose their identity in order to participate and a set of trusted nodes is pre-selected and granted the authority to add new blocks. Operating with a small, known set of validators makes PoA highly scalable and energy-efficient, but at the cost of limited decentralization. Since the validators are few and easily identifiable, an attacker could attempt to corrupt them and undermine the system from within. For this reason, PoA is particularly suited to private or consortium blockchains, where the trustworthiness of participants is easier to guarantee.

The protocols described above are among the most widely used, but many others exist, such as Proof of Burn (PoB), Pure Proof of Stake (PPoS), Delegated Proof of Stake (DPoS) and Proof of Activity (PoAc), to name only a few.

2.6 Blockchain 2.0: Ethereum and Smart Contracts

Ethereum is an open source platform based on blockchain, on which it is possible to execute programs called smart contracts, the building blocks of decentralized applications (dApps), thanks to the integration of a Turing-complete language. Smart contracts are deployed on the ledger and executed by the Ethereum Virtual Machine (EVM), a deterministic runtime replicated across all nodes [5]. A smart contract encodes rules that run exactly as written and that activate automatically when predefined conditions are met, without the mediation of a central authority.

A smart contract is code that manages processes in a precise and objective manner, responding to specific events and without involving third parties. They can handle various types of transactions, not just cryptocurrency transfers. In particular, they can define rules, like an ordinary contract, and enforce them automatically through the code.

State changes are triggered by transactions: signed messages that invoke a contract function and, if valid, modify the shared state. Alongside state changes, contracts emit events, which provide a cheap and externally observable trace of what happened.

To prevent the network from stalling due to the execution of infinite loops, Ethereum introduced ”gas”. A smart contract can run as long as it has gas available,

which is purchased with ETH. The maximum amount of gas available to the smart contract must be defined and set in advance within the contract itself.

Smart contracts offer greater security and speed than traditional contracts, reducing transaction costs while ensuring autonomy and efficiency. On the other hand a contract executes autonomously and vulnerabilities in the code translate directly into risk. Nowadays smart contracts have to face many of the challenges related to trust, security, confidentiality and legal issues.

2.7 Blockchain Security

Although blockchain offers desirable properties such as decentralization, integrity, immutability and transparency, it is not immune to vulnerabilities and attacks.

Risks in first-generation blockchains - Blockchain 1.0
These concern cryptocurrencies and the management of the distributed ledger [3]:

- **51% vulnerability**: if a single actor controls more than half of the network's computing power (in PoW) or stake (in PoS), it can manipulate the consensus, reorder or exclude transactions and disrupt the normal operation of the other nodes.
- **Private key security**: the compromise or loss of a private key results in the loss of control over the associated funds.
- **Criminal activity**: anonymity (or pseudonymity) can be exploited for illicit purposes, such as money laundering or illegal payments.
- **Double spending**: the attempt to spend the same coin twice, favored by network delays or by the exploitation of system vulnerabilities.
- **Transaction privacy leakage**: the public nature of the ledger may allow pseudonymous addresses to be linked to real identities, undermining users' privacy.

Risks in second-generation blockchains - Blockchain 2.0
These are related to the introduction of smart contracts [3]:

- **Criminal smart contracts**: contracts intentionally designed to facilitate criminal activities.
- **Vulnerabilities in smart contracts**: programming errors or logical flaws in the code that can be exploited by an attacker.

- **Under-optimized smart contracts**: inefficient implementations that entail higher computational costs (gas) than necessary.
- **Underpriced operations**: operations whose gas cost does not reflect their actual computational load, which can be exploited to mount Denial-of-Service attacks.

The most common attacks are [3]:

- **Selfish mining**: a miner (or pool) withholds newly mined blocks to build a longer private chain, depriving other participants of their rewards and increasing the risk of a 51% attack.
- **BGP hijacking**: the manipulation of network traffic routing to intercept or delay the propagation of blocks.
- **Eclipse attack**: the isolation of a node from the rest of the network by controlling all of its connections, in order to provide it with a distorted view of the blockchain.
- **Liveness attack** and **balance attack**: attacks aimed, respectively, at delaying the confirmation of transactions and at compromising the consensus by exploiting network partitions.

Countermeasures and research directions To address these threats, several security solutions have been proposed [3], including tools for the analysis and verification of smart contract code, quantitative risk-analysis frameworks, fairer mining protocols and privacy-preserving solutions. Research remains oriented toward increasingly scalable and secure blockchains.

Chapter 3

State of the art: DAO - Decentralized Autonomous Organization

3.1 Introduction to DAO

A Decentralized Autonomous Organization is an organization whose operating rules are encoded in smart contracts on a blockchain and executed automatically, without a central authority mediating their application [6, 7]. Whereas in a traditional organization the enforcement of the rules depends on people and hierarchies, in a DAO the deliberative procedures and their execution are delegated to code that activates when predefined conditions occur. The three words of the acronym explain how it works:

- **Decentralized**: authority is distributed across the network, not concentrated in a single actor;
- **Autonomous**: the rules are executed on their own once triggered;
- **Organization**: the system coordinates a group of participants toward shared goals.

The concept became feasible with the advent of general-purpose smart contract platforms and Ethereum [5], which provided the Turing-complete execution environment required to encode arbitrarily complex governance logic on chain.

3.2 Core Characteristics

DAOs inherit the properties of the underlying blockchain and combine them with the automation of smart contracts. The literature converges on a small set of characteristics that distinguish a DAO from a conventional organization [8, 9]:

- **Decentralization:** authority over decisions and assets is not vested in a single central actor but distributed across the members of the network according to the rules encoded in the contracts. No single participant can unilaterally impose an outcome, which makes the organization resistant to unilateral capture and censorship.

- **Autonomy:** once a decision is taken according to the governance rules, its execution is carried out automatically by the smart contracts, without relying on an intermediary to enforce it. The organization thus runs, within the boundaries of its code, as a self-operating system.

- **Transparency and verifiability:** proposals, votes and treasury movements are recorded on a public ledger and are therefore inspectable by anyone. Any participant can independently verify how a decision was reached and whether it was executed as mandated, without having to trust a central bookkeeper.

- **Immutability and traceability:** once recorded on chain, an act cannot be silently altered or deleted. Every vote, resolution and disbursement leaves a permanent and traceable trace, which guarantees the integrity of the organization's history.

- **Censorship resistance:** no authority, not even a government, can shut down a DAO or interfere with its operation, because the organization is run by a decentralized network of nodes that themselves act as its operators and as the authority over it.

- **Democratic participation:** anyone, anywhere, can in principle join a DAO and contribute to it in the way they prefer. Every member can put forward proposals, take part in votes and in doing so shape the future direction of the organization.

Some of these characteristics are more aspirational than realistically attainable. In practice many DAOs do not perfectly reflect all the ideal characteristics outlined in this list. Each of them may prioritize certain aspects over others, thus resulting in structures that can embody only some of these characteristics. The table 3.1 compares the main characteristics of DAOs and traditional organizations.

Table 3.1: Main differences between traditional organizations and DAOs.

Aspect	**Traditional organization**	**DAO**
Authority and control	Centralized: a hierarchy of managers, a board or an owner holds decision-making power.	Decentralized: authority is distributed across the members according to on-chain rules and no single actor is in control.
Rule enforcement	Rules are enforced by people, internal hierarchies and the legal system.	Rules are encoded in smart contracts and enforced automatically by code when the predefined conditions occur.
Decision-making	Top-down: decisions are taken by management or delegated bodies.	Bottom-up: members submit proposals, vote and approved outcomes are executed automatically.
Transparency	Internal records are private: outsiders rely on periodic disclosures and audits.	Proposals, votes and treasury movements are recorded on a public, independently inspectable ledger.
Record integrity	Records are mutable and kept by a trusted bookkeeper.	Records are immutable once anchored on chain and their integrity does not depend on any single party.
Trust model	Trust is placed in intermediaries and institutions.	Trust is placed in code and cryptography rather than in intermediaries.
Membership and access	Access is mediated through hiring, contracts, or geographic and legal barriers.	Access is typically open and permissionless, anyone can join and contribute.
Legal status	Recognized legal personality: the entity can enter into contracts, own assets and sue in its own name.	Usually no autonomous legal personality and the code is the law, unless a dedicated legal wrapper is adopted.
Continuity and censorship	Operations can be suspended or shut down by an authority.	Operation is continuous and resistant to censorship or unilateral shutdown.

3.3 Structure of a DAO

Structurally, a DAO can be described through a few recurring components. A community of members constitutes the body of participants whose rights are defined by the system. A set of tokens encodes membership, access rights and voting power. A treasury holds and disburses the organization's assets. Finally, a body of governance rules, encoded in smart contracts, determines how collective decisions are taken and executed.

3.3.1 Governance

The governance of a DAO rests on three fundamental pillars as shown in figure 3.1, whose interaction is what makes decentralized and autonomous decision-making possible [10]:

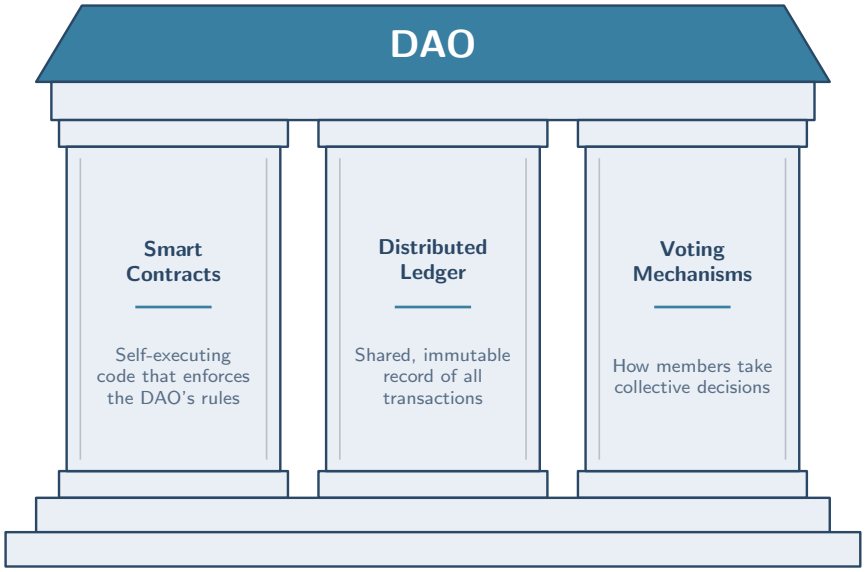


Figure 3.1: The three pillars of DAO governance.

- The first is the **Distributed Ledger**, the underlying blockchain that records every transaction and state change in an immutable and publicly verifiable way. It provides the shared, tamper-resistant source of truth on which the entire organization relies.
- The second is the **Voting Mechanism**, through which members express their preferences on proposals; it is the only way to change the rules encoded in the

smart contracts or to create new ones.

- The third is the set of **Smart contracts**, the self-executing programs that encode the organization's rules including the vote mechanism and automatically enforce the outcome of a vote, translating an approved proposal into concrete on-chain actions without the need for a trusted intermediary.

3.3.2 Governance Process

The operational core is the governance process, through which the community formulates, discusses and approves collective decisions. A proposal typically follows a lifecycle made of a submission phase, in which a member puts forward a proposal, then a phase of deliberation and discussion, in which the community debates it and a voting phase, in which members express their preferences. If the proposal is approved, its outcome is executed, in the fully automated case, directly by the contracts. Otherwise, if a proposal is rejected, it is archived. The lifecycle of a proposal is shown in Figure 3.2 below.

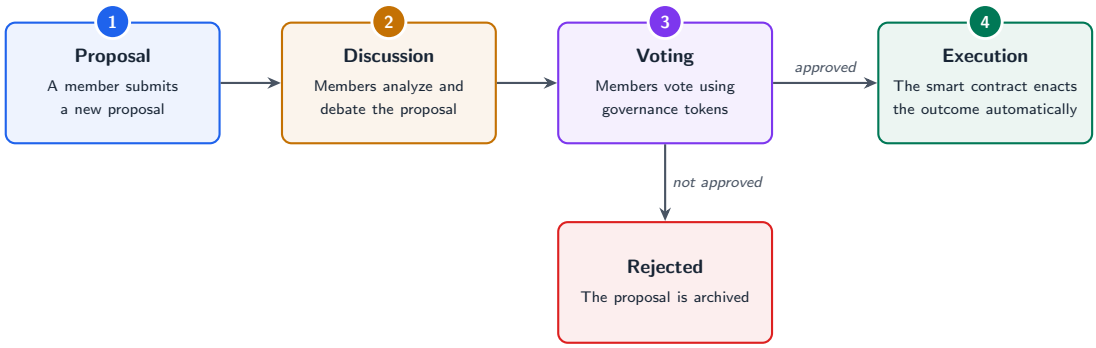


Figure 3.2: DAO governance process.

An important architectural distinction concerns where each of these phases takes place. In a fully on-chain design, every step, including the expression of preferences, is recorded on the blockchain, which maximizes verifiability but imposes a transaction cost on every action. In a hybrid design, deliberation and non-binding signaling are moved off chain to reduce cost and friction, while only the binding outcome and its execution are anchored on chain.

3.3.3 Treasury

The treasury of a DAO represents the financial core of the organization and is managed in a decentralized and automated manner. Through the treasury, a DAO

is able to raise funds from investors, generate revenue from the organization's activities, pay the members who contribute to its operation and provide a return to investors. As with every other aspect of the organization, the treasury must also preserve the DAO's key properties. Consequently, no member can make decisions concerning the DAO's investments without first submitting the proposal to a vote of all members.

The treasury of a DAO is typically managed through a multi-signature (multisig) wallet shared among several members. Rather than being controlled by a single key, the treasury is protected by a smart contract that requires the approval of a predefined threshold of signers before any operation can be authorized and executed. This solution removes the single point of failure associated with individual custody and distributes control across multiple parties.

3.4 Voting Mechanisms

The way a DAO aggregates individual preferences into a collective decision is governed by its voting mechanism. Several models exist, each embodying a different relationship between a member's stake and the weight of their vote. Seven types of voting mechanisms typically used in current DAOs have been analyzed in [10]:

- **Permissioned Relative Majority (PRM)**: most common method where a proposal is approved when it is supported by more than half of the voting power actually cast, so the passing threshold is fixed at 50% of the participating power and the outcome is immediate to compute. This efficiency comes at the cost of security. Since approval depends only on the balance among those who happen to take part and no minimum level of participation is required, a contentious or even harmful proposal may pass unnoticed when turnout is low, a weakness known as the slip-through of passed proposals. The risk is mitigated in small and highly active communities, where members tend to follow every proposal, but it remains a structural threat.
- **Token-based quorum mechanism (TBQ)**: similar to PRM, this mechanism demands a higher level of participation before a proposal can be approved, thereby lowering the risk of slip-through approvals and strengthening the security of the vote. Such broader engagement can be encouraged by introducing an attention threshold, which may in turn enhance decentralization by drawing in a larger number of nodes. The trade-off, however, is that this approach takes longer to reach the required participation.
- **Quadratic voting**: it represents a compromise between the one-token-one-vote (1T1V) and one-person-one-vote (1P1V) schemes, designed to balance

their respective drawbacks through a marginal-cost mechanism. With this method the cost of casting additional votes on a given proposal grows quadratically: a single vote on an option costs one token, two votes cost four tokens, three votes cost nine and so on, so that the number of votes a participant obtains equals the square root of the tokens they commit. QV substantially mitigates the dominance of a small number of large token holders and improves the degree of decentralization, while remaining more flexible and expressive than the rigid 1P1V model, as it still allows participants to signal the intensity of their preferences.

- **Liquid democracy**: it combines direct and representative participation by allowing a member to delegate their vote to another member [11]. When the delegation is revocable and can be granted per topic, a member may entrust their voice, subject by subject, to whoever they consider most competent, while retaining the power to reclaim it at any moment. This leads to a low-barrier form of effective participation that does not change the formal equality of the vote but requires delegation to be limited to avoid infinite accumulation of proxies by any single delegate. Nonetheless, this can also result in the centralization of voting power.

- **Weighted voting**: the votes of members with higher reputation or knowledge carry more weight, encouraging informed decisions at the cost of decentralization. An illustrative example is Knowledge-Extractable Voting, which gives greater voting weight to those who hold certificates in a given field of knowledge.

- **Rage Quitting**: once a proposal is approved, members may exit the organization and redeem their tokens within a grace period, should the outcome fail to meet their expectations. This strengthens members' individual rights and decentralization, but at the cost of decision-making efficiency.

- **Holographic Consensus**: Holographic Consensus by DAOstack addresses these trade-offs through a prediction market which works in parallel with the voting system. Each proposal is paired with a market based on a dedicated token, GEN, which participants stake to bet on whether the proposal will pass or fail, with accurate predictions rewarded. Proposals that attract sufficient supporting stake are boosted and require only a relative majority, whereas the rest remain queued and require an absolute majority. This mechanism improves flexibility, efficiency and decentralization, since even members with little voting power can signal their views through staking. Its main drawback is that bettors who stake on a proposal are incentivized to vote for its approval, potentially distorting the outcome.

Another voting mechanism is proposed in [12]:

- **Vote conviction**: in Conviction Voting, members express their preferences on proposals by staking their tokens for a set period of time. Each member's voting weight grows in proportion to the duration of the stake: the longer the tokens remain locked, the greater the "conviction" attached to that preference. The mechanism is designed to reward careful consideration of a proposal's long-term implications, while discouraging impulsive choices and short-term speculation.

3.5 Token Standards

On the technical level tokens are distinguished by the standard that defines how the underlying smart contract behaves. On Ethereum these standards are known as ERCs (Ethereum Request for Comments), a subcategory of the EIPs (Ethereum Improvement Proposals) that specifies application-level rules. A token standard is a shared set of functions and events that a contract must implement so that wallets, exchanges and other applications can interact with any token in the same way, without needing to know the details of each individual project. This shared interface is what makes tokens interoperable across the ecosystem.

The main distinction between standards is not one of purpose but of fungibility, that is, whether the tokens are interchangeable with one another or unique. The most widely adopted standards are the following:

- **ERC-20**: it is the standard for fungible tokens, in which every unit is identical and interchangeable, in the same way as units of a currency. The standard defines interfaces common to the contracts that implement the tokens, so that, by defining shared behaviors, they can all be used in the same way [13]. It is the standard behind most cryptocurrencies, stablecoins and governance tokens.
- **ERC-721**: it is the standard for non-fungible tokens (NFTs). Each token has a unique identifier and cannot be exchanged on a one-to-one basis with another, which makes it suitable for representing unique items such as digital art, collectibles or certificates [14]. Every collection normally requires its own separate contract.
- **ERC-1155**: it is a multi-token standard that allows a single contract to manage fungible, non-fungible and semi-fungible tokens at the same time. It also supports batch transfers, meaning that several tokens can be moved in one transaction, which reduces cost and complexity compared with deploying separate ERC-20 and ERC-721 contracts [15].

- **ERC-5192**: finalized in 2022, it is a minimal extension of ERC-721 that makes a token non-transferable, or "soulbound". Once assigned to an account, the token is permanently locked to it and any attempt to transfer it fails [16].

Table 3.2: Comparison of the main Ethereum token standards.

	ERC-20	**ERC-721**	**ERC-1155**	**ERC-5192**
Fungibility	Fungible (identical, interchangeable units)	Non-fungible (unique identifier)	Mixed: fungible, non-fungible and semi-fungible	Non-fungible (unique identifier)
Transferable	Yes	Yes	Yes	No, soulbound (permanently locked to the account)
Tokens in one contract	A single fungible supply	One collection (usually a separate contract each)	Many token types in one contract	One collection (extends ERC-721)
Batch transfers	No	No	Yes	No
Typical use	Stablecoins, Cryptocurrencies, governance tokens	Digital art, collectibles, certificates	Many token types with gas-efficient batch transfers	Non-transferable credentials, identity and reputation badges

3.6 The Regulatory Classification of Tokens: from FINMA to MiCA

Tokens can also be classified from a regulatory perspective, which aims not to describe their properties in full but to place them within defined legal categories. Two references are particularly relevant: the 2018 guidelines of the Swiss FINMA and the 2023 European MiCA Regulation.

3.6.1 FINMA's Classification

On 16 February 2018 the Swiss Financial Market Supervisory Authority (FINMA) published its guidelines on Initial Coin Offerings (ICOs), grounding the classification of tokens on their economic function and their transferability [17]. Three types are identified:

- **Payment tokens** (or cryptocurrencies): tokens intended to be used as a means of payment for acquiring goods or services, or as a means of transferring money or value.
- **Utility tokens**: tokens intended to provide digital access to an application or service supplied through a blockchain-based infrastructure. A token qualifies as a utility token only if it is immediately usable for the purpose for which it is issued.
- **Asset tokens**: tokens representing an asset, such as a debt or equity claim against the issuer (for instance, a share in future company earnings or capital flows). In terms of their economic function they are analogous to equities, bonds or derivatives, and are therefore treated by FINMA as securities.

FINMA specifies that the categories are not mutually exclusive: hybrid forms are possible, such as a token that simultaneously performs utility and payment functions, to which the requirements laid down for each relevant category apply cumulatively [17].

3.6.2 The Crypto-Asset Categories of the MiCA Regulation

Regulation (EU) 2023/1114, known as MiCA (Markets in Crypto-Assets), which entered into force on 29 June 2023, introduces the European Union's first harmonized legal framework for crypto-assets not already governed by existing financial legislation [18]. Unlike FINMA's functional approach, MiCA adopts a classification based primarily on value-stabilization mechanisms, distinguishing three categories:

- **E-money tokens (EMTs)**: crypto-assets that aim to keep a stable value by referencing the value of a single official currency and that therefore work as a digital representation of legal-tender money.
- **Asset-referenced tokens (ARTs)**: crypto-assets that aim to keep a stable value by referencing another value or right, or a combination of these, including one or more official currencies, commodities or other assets.
- **Other crypto-assets**: Other crypto-assets: this residual category includes tokens that lack stabilization mechanisms and do not fall into the two preceding categories, such as utility tokens.

It follows from this classification that payment and utility tokens, which are separate and central categories in FINMA's framework, both end up in the residual category of "other crypto-assets", while asset tokens in the Swiss sense tend to fall outside MiCA's scope.

.

3.7 DAO Classification

DAOs come in several distinct types, each defined by the scope of the proposals its members vote on and by the way capital, resources and talent are pooled together. The following paragraphs outline these mission types and their defining characteristics, following the classification proposed in [19]:

- **Collector DAO**: Collector DAOs pool together capital and resources to acquire valuable digital artifacts, most commonly NFTs. By lowering entry barriers to high-value collectibles, they act as community-owned NFT funds. They do not simply buy, instead they draw on the collective knowledge of their members to identify the most promising items, the ones likely to grow in value. The collection is then curated and governed jointly, with members deciding by voting how the assets should be used, collateralized, or resold to generate profit for the community.

- **Grants DAO**: Grants DAOs pool together capital and resources in order to fund promising projects through donations, with the primary aim of generating impact rather than financial return. They typically operate on a non-profit basis, allocating a shared treasury to initiatives that align with the community's values or with a particular ecosystem's development. Members collectively decide which proposals deserve funding, making the governance process central to the DAO's purpose. In this sense, the scope of the vote is defined by where and to whom the funds should be directed, distinguishing grants DAOs from purely for-profit investment structures.

- **Investment DAO**: Investment DAOs pool together capital and resources to invest in promising projects with the explicit goal of generating profit for their members. Unlike grants DAOs, their orientation is for-profit and they tend to adopt a more formalized governance process due to the fact that they directly allocate the funds contributed by participants. Members propose and vote on investment opportunities, sharing both the risks and the potential returns.

- **Protocol DAO**: Protocol DAOs are strongly dominated by decentralized finance (DeFi) applications. Their mission is to operate and govern a single web3 protocol as a community, tokenizing the project in a way comparable to

company shares. Members own a small stake in the protocol and share in its success through the token price, without being required to provide additional funding or resources. Governance tokens often carry utility within the protocol itself, such as paying fees, while still enabling members to propose and vote on changes to the underlying mechanisms.

- **Service DAO**: Service DAOs aggregate the talent and skills of their community to provide services to other DAOs and entities. They coordinate human resources, matching contributors with tasks and clients that require specialized work. The scope of governance here concerns decisions such as which clients to work for and how to distribute compensation, positioning service DAOs as decentralized talent collectives.

- **Media DAOs**: Media DAOs collectively create media content, operating as community-driven alternatives to traditional publishing and content organizations. Their mission generally leans toward awareness creation, education, and community engagement rather than direct financial gain, giving them a predominantly non-profit character. Because much of their activity can take place off-chain, their status as fully "autonomous" organizations is sometimes disputed. Nevertheless, they rely on shared governance to decide on editorial direction and content strategy and are often marked by strong member engagement around a common creative purpose.

- **Social DAO**: Social DAOs operate a like-minded community while pursuing a shared social goal. They provide a space for exchange, cultivate a distinctive culture and often organize events, whether exclusive and invitation-based or open to all. Many also embrace a social-good mission in areas such as the environment, education, or the arts. Financial returns are usually secondary to community-exclusive perks and to the mission itself, which tends to be especially prominent in this type. Like media DAOs, they may operate off-chain and are characterized by particularly engaged members.

3.8 Problems and limitations

Before examining weaknesses related specifically to DAO technology, it is worth recalling that a DAO is built on top of a blockchain and executed through smart contracts. Therefore it inherits all the security weaknesses of the underlying infrastructure [20]. Vulnerabilities in the consensus layer, in the network, or in the smart contracts on which a DAO relies translate directly into risks for the organization itself, since its rules and its treasury ultimately depend on that infrastructure.

Table 3.3: DAO classification.

Type	What it pools	Primary aim	Scope of the vote
Collector	Capital to acquire digital artifacts (NFTs)	Community-owned NFT fund	How to use, collateralize or resell the collection
Grants	Capital obtained as donations	Impact over return (non-profit)	Which proposals to fund and to whom
Investment	Capital to invest	Profit for members (for-profit)	Which opportunities to back
Protocol	A tokenized stake in one web3 protocol	Operate and govern a web3/DeFi protocol	Changes to the protocol's mechanisms
Service	Talent and skills	Provide services to other DAOs and entities	Which clients to serve and how to split compensation
Media	Collective editorial work	Awareness, education and engagement (non-profit)	Editorial direction and content strategy
Social	Community and shared culture	A social goal	Events, culture and mission direction

3.8.1 The vulnerability of code and the limits of automated execution

Smart contracts are self-executing and, once deployed, largely immutable, so any flaw in their code is a structural weakness. Unlike a conventional institution, where an erroneous decision can be corrected through human oversight or administrative review, a DAO executes what the code prescribes and a defect can be exploited before the community is able to react. This is the practical consequence of the principle that "code is law", a phrase coined by Lessig [21] to warn that software regulates behavior much as legislation does. Then it was adopted in the blockchain community to mean that the rules written in a contract are the only ones that apply.

The historical origin of this concern is precise. The problem emerged with The DAO, a decentralized investment fund in which participants voted on the allocation of resources and because of a vulnerability in its smart contract, was subject to an attack that drained roughly 3.6 million ether [22]. The episode is significant because it revealed that the code performed exactly as written, so the failure lay not in a breach of the rules, but in the rules themselves. The response, a hard fork of the underlying blockchain that reversed the theft, exposed a deeper tension. Reversing the outcome required a coordinated human intervention that contradicted the principle of immutability on which the system was founded, splitting the community over whether "the code" should ever be overridden.

The episode opened the academic discussion on the robustness and security of DAOs. Automating execution does not, by itself, guarantee reliable operation, so without adequate procedural safeguards and proper auditing of the code, the removal of human intermediaries removes not only their inefficiencies, but also their capacity to detect errors and intervene.

3.8.2 Governance attacks and the exploitation of the voting mechanism

A distinct kind of weakness affects not the code itself but the voting mechanism that governs the organization. Even when the smart contract works exactly as intended, an attacker can gain enough voting power to push through a malicious proposal, targeting the decision procedure. Many of these attacks exploit the human side of governance, whereas security audits tend to focus on code and protocol vulnerabilities, leaving this class of threat comparatively overlooked [23]. The clearest example is the Beanstalk case: the attacker took out a very large flash loan, used it to obtain a two-thirds majority of the voting power, then immediately executed a proposal in their favor through an emergency function, draining the

treasury within a single transaction. The point is not that the code was faulty, but that the governance rules allowed newly acquired tokens to vote and act at once, with no delay before execution.

3.8.3 Plutocratic concentration of voting power

One of the most significant limitation is the tendency toward a plutocratic concentration of power that is inherent in the token-based model. When the weight of a vote is proportional to the capital held, effective control of the organization shifts toward its largest token holders. Two empirical studies confirm the pattern.

The first, based on a sample of one hundred DAOs, finds that organizations whose token has a secondary market value display a significantly more unequal distribution of voting power. In many of them the voters with greater power participate more actively, further under-representing smaller voters [24]. The second systematically analyzes fifty DAOs, extracting and normalizing the governance parameters encoded in their smart contracts, such as quorum, proposal thresholds, voting periods and execution delays [25]. Its results provide quantitative confirmation of this concentration of power. In many of the organizations examined, fewer than one percent of token holders hold enough tokens to submit a proposal and quorum thresholds are set below five percent, effectively handing governance to the largest holders. High transaction costs act as a further, capital-based barrier to participation. The study captures the resulting paradox concisely: the DAOs analyzed are technically decentralized in their code, yet politically centralized in practice, tending to reproduce the same power asymmetries found in traditional financial markets.

3.8.4 Voter apathy and low participation

The review and empirical analysis by [26] finds low voter engagement across all the governance models examined, including off-chain (gas-less) voting: only a small minority of members votes actively, while most rarely take part. The fact that turnout stays low even when transaction fees are removed suggests that apathy is a built-in feature of the model, not just a consequence of gas costs. The same study also notes that members with more voting power vote more often, so low turnout does not just reduce participation but shifts it toward the largest holders. This creates a self-reinforcing cycle: turnout is so low that quorums must be kept minimal to avoid paralyzing the protocol, decisions end up in the hands of an active minority and the remaining members disengage even further.

3.8.5 Delegation is not enough

Several DAOs try to counter voter apathy by allowing token holders to delegate their voting power to more active or better-informed representatives. The problem is that delegation tends to reproduce the very concentration it was meant to relieve. As a result voting power accumulates in a small group of highly visible delegates, so that a few actors end up controlling most of the delegated votes and effectively recreate a centralized structure [27]. The problem is that delegates are usually presented in ranked lists, the most prominent ones attract still more delegations, in a self-reinforcing dynamic. A further issue is that delegations are often poorly aligned with the actual preferences of those who delegate, so the represented interests may diverge from the priorities of the wider community.

3.8.6 The uncertain legal status of DAOs

A final limitation concerns the unclear legal standing of DAOs and, in particular, the question of who is responsible for their decisions. Because a DAO has no central management and often no recognized legal form, so it is not obvious who can be held accountable when its actions cause harm or breach the law.

In the case Ooki DAO (2022), a United States court treated the organization as an unincorporated association under the Commodity Exchange Act, despite its decentralized and anonymous structure and accepted that the token holders who exercised their voting rights could be held jointly liable for the DAO's conduct. The implication is that taking part in governance may expose them personally to liability.

To address this problem, a number of jurisdictions have begun to adapt existing corporate forms to DAOs, or to introduce new ones. In 2018, Vermont amended its limited liability company statute to allow the creation of blockchain-based limited liability companies (BBLLCs) [28]. Then comparable measures were subsequently adopted by Wyoming in 2021 [29], the Republic of the Marshall Islands in 2022 [30] and Utah in 2023 [31], granting DAOs legal personality and limited liability through dedicated or specifically adapted corporate forms.

3.9 Non-blockchain-based civic platforms

The two main non-blockchain-based digital participation platforms are Decidim and Consul, free and open-source participatory platforms conceived by activists of the 15M movement in Spain, initially deployed in Barcelona and Madrid and subsequently adopted in many countries. These platforms are based on a web application managed by the public administration and they constitute the

benchmark against which the possible added value of an on-chain approach must be measured.

Decidim was developed by the City of Barcelona during the 2015–2019 term and it hosts, among other things, significant participatory budgeting processes. In the 2020–2023 cycle it allowed citizens to decide the allocation of a share of up to 75 million euros of the municipal budget. Consul, for its part, has seen an even wider international diffusion: according to the foundation that oversees its development, the software has been adopted in around 35 countries by more than one hundred institutions, reaching on the order of tens of millions of citizens. These are therefore real tools, operating at scale and institutionalized, unlike the on-chain experiments examined earlier.

The empirical literature on them reveals their structural limitations. Analyzing the deployment of Decidim in Catalan municipalities, Borge, Balcells, and Padró-Solanet [32] identify elements of managerial continuity: the objectives most valued by administrations are transparency, the organization of information and the gathering of citizens' proposals, rather than deliberation and the transfer of sovereignty to the citizens themselves. The authors acknowledge the positive effects, but the overall picture is one of tools that broaden participation without, however, redistributing effective decision-making power. To this must be added the question of institutional dependence: significantly, the Consul Democracy Foundation was established in 2019 precisely in order to reduce the dependence on the City of Madrid that had developed the software, a sign that the sustainability of these platforms remains tied to the institution that hosts them.

Thus, on the one hand, established civic platforms offer participation, but not binding and verifiable procedural guarantees. They record proposals and votes, but nothing in their operation prevents an administration from ignoring the outcome or postponing its implementation indefinitely.

3.10 Civic DAOs and public-sector applications

An important contribution that explores the application of DAOs to the public sector in this direction is eGov-DAO [33], which proposes to use blockchain and smart contracts to automate a public administrative process, specifically the awarding of procurement contracts, with the aim of reducing its vulnerability compared to a centralized infrastructure, narrowing the scope for corruption by promoting transparency and lowering the errors associated with manual processes. Through smart contracts, the system executes and publicly records the entire life

cycle of a contract, from submission to auditing, relying on the immutability of the ledger to guarantee the traceability of every step. It is significant that, while starting from a premise of full decentralization, this architecture reintroduces a centralized element of trust at the level of identity. In fact participation requires a registration procedure in which an authority issues a digital certificate that serves as the user's identity [33]. The proposed paradigm is, however, excessively optimistic, in fact the system is described as immune to both external and internal attacks, and it is concerned more with demonstrating technical feasibility than with critically assessing the conditions for its applicability and implementation.

Through a conceptual analysis based on scenarios applied to municipal contexts [34], they conclude that DAOs are appropriate for recurring, continuous, and multi-stakeholder processes that require a high degree of transparency and a rule-based allocation of resources. By contrast, for simple, localized, and one-off initiatives they prove oversized compared to more accessible participatory platforms such as Decidim or Crossiety. The authors further observe that, even where the context is favorable, the benefits in terms of participation and transparency materialize only when certain antecedent conditions are met, such as digital literacy, administrative and technical capacity and the stability of the digital infrastructure.

In summary, the literature places a neighborhood DAO, one that manages a continuous flow of proposals and collective decisions, within the category of processes for which, according to the authors, the adoption of a DAO is justified. At the same time, feasibility does not depend solely on the design structure, but on a set of socio-technical prerequisites of the territory in which the system is embedded a condition that is far from guaranteed in a neighborhood, characterized by a heterogeneous population that is not necessarily technically competent. Overall, this body of research demonstrates that the application of DAOs to the public sector is technically feasible and potentially advantageous in terms of transparency, traceability and participation, provided that adequate contextual conditions are met. Two fundamental limitations remain, however, which are decisive for the purposes of the present work. The first is that the participation model remains predominantly anchored to token ownership and weighted voting, which is inadequate for a context in which legitimacy must be grounded in residency. The second, more radical, is that these contributions address the application of DAOs to the public or civic domain in general terms without addressing the problem of how a legally accountable public body can coexist within the governance process, while retaining a compliance-oversight role.

Chapter 4

CivicDAO Design

4.1 Objectives

CivicDAO refers to a decentralized autonomous organization operating on the neighborhood scale, designed to coordinate and incentivize citizen participation in the shared care of common goods. Interventions such as cleaning a park, organizing a public event, or redeveloping a shared space find in it the instrument through which they are proposed, deliberated and carried out by the community itself. The aim of the chapter is to derive the components starting from the research problem, namely how to keep a public body legally accountable within a decentralized governance framework, granted with verifiable procedural guarantees, without that body being able at the same time to centralize the decision-making process.

4.2 Building Blocks of the DAO

4.2.1 Purpose

The neighborhood DAO designed here pursues a shared and territorially circumscribed mission: to enable resident citizens to propose, deliberate, fund and coordinate civic projects, strengthening the community's autonomy in the care of common goods, reinforcing the sense of belonging and generating social value through decentralized decision-making processes and mechanisms of stable collaboration among local actors. The formal incentive structure that supports this purpose is organized around three levers: non-transferable reputation, closed-loop community credits and participation certificates (Proof of Participation). The absence of a monetary return for members is consistent with the project's social orientation, in fact the proceeds serve to cover operating costs and to symbolically remunerate activities.

As for the time horizon, CivicDAO has a long-term iterative purpose. It is not a single-purpose DAO destined to dissolve once a specific objective is reached, but rather a permanent governance infrastructure whose purpose is realized through a cyclical succession of civic projects.

4.2.2 Use Case

CivicDAO sits between Social DAOs, organizations that bring together individuals united by a shared interest and prioritize community involvement and Grant DAOs, that facilitates the decentralized funding of public goods and non-profit civic initiatives, according to the taxonomy discussed in Section 3.7.

CivicDAO is designed for the citizens of a neighborhood who wish to contribute actively to the care of the common goods of their own territory. Residents, informal groups and local associations that the legal system recognizes as civic actors entitled to collaborate with the administration. These are people who share a concrete interest in the space they live in, but who, in traditional forms of participation, encounter obstacles like opaque procedures, difficulty coordinating with one another, poor visibility of the contribution they have made and the absence of a structured channel through which to propose initiatives and see them carried out.

CivicDAO offers its users a set of benefits that constitute its value proposition. First, equal access to the decision-making process, allowing every participant to submit proposals and take part in deliberations. Second, transparency and verifiability of actions, since decisions and the use of resources are recorded in a traceable and publicly accessible way. Third, procedural certainty, since the governance system automatically routes each proposal either toward autonomous execution or toward review by the administration, depending on whether or not it falls within the limits of the collaboration agreement.

4.2.3 Legislation

CivicDAO adopts a hybrid approach: governance remains decentralized and on-chain, while a traditional legal wrapper represents the community in its dealings with third parties. CivicDAO operates through two distinct flows:

Public Flow: resources allocated by the Municipality are governed through the instrument of the participatory budgeting (bilancio partecipativo)[1], grounded

[1]Participatory budgeting (bilancio partecipativo) is a form of direct citizen participation in

in horizontal subsidiarity (art. 118, comma 4, Cost.) and in the participatory institutions of local government (art. 8 TUEL, the Consolidated Law on Local Government). On this flow the DAO performs a single function, the casting of the on-chain vote: the community decides how the resources are to be allocated, while the assessment of admissibility and feasibility, the appropriation of funds and the execution remain proceedings of the Municipality. The latter therefore undertakes in advance, through a regulation of its own, to give effect to the community's decisions according to the agreed procedure and where it departs from them it is required to provide specific reasons.

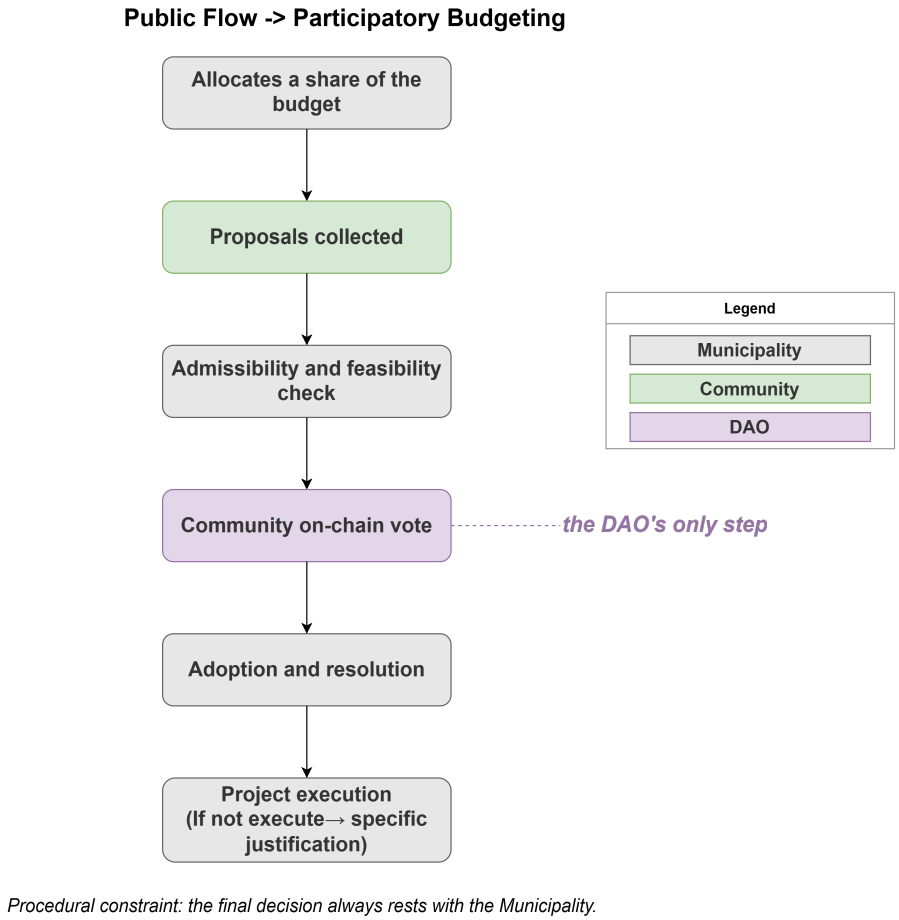


Figure 4.1: Public flow: participatory budgeting process.

the political life of one's city, consisting in the allocation of a share of the local authority's budget to the direct management of citizens, who are thereby enabled to interact and engage with the choices of the administration in order to shape them to their own benefit.

Private Flow: private resources and the community's internal economy are governed directly by the DAO. The legal wrapper is a "Third Sector entity" (ente del Terzo Settore, ETS) endowed with legal personality (an association registered in the RUNTS, the Single National Register of the Third Sector, pursuant to art. 22 of d.lgs. 117/2017), whose full asset partitioning excludes the personal and joint liability of its members that characterizes the unrecognized association"(associazione non riconosciuta), lacking legal personality (art. 38 c.c., the Italian Civil Code).

The holders of the citizenship token, which is non-transferable and assigned to residents, are the members: the on-chain resolution is the form through which the members' assembly expresses its will. The internal economy is self-financing through local businesses, which join the circuit by paying an annual membership fee in euro, collected by the entity into an ordinary bank account and evidenced by a membership token. Within the circuit the community credit operates as a unit of exchange that cannot be purchased with euro and cannot be converted into legal tender. It is earned by citizens as a form of recognition for taking part in activities and can be spent only at participating businesses, which in turn burn it in exchange for sponsorship and visibility. The overall structure of the private flow is shown in Figure 4.2.

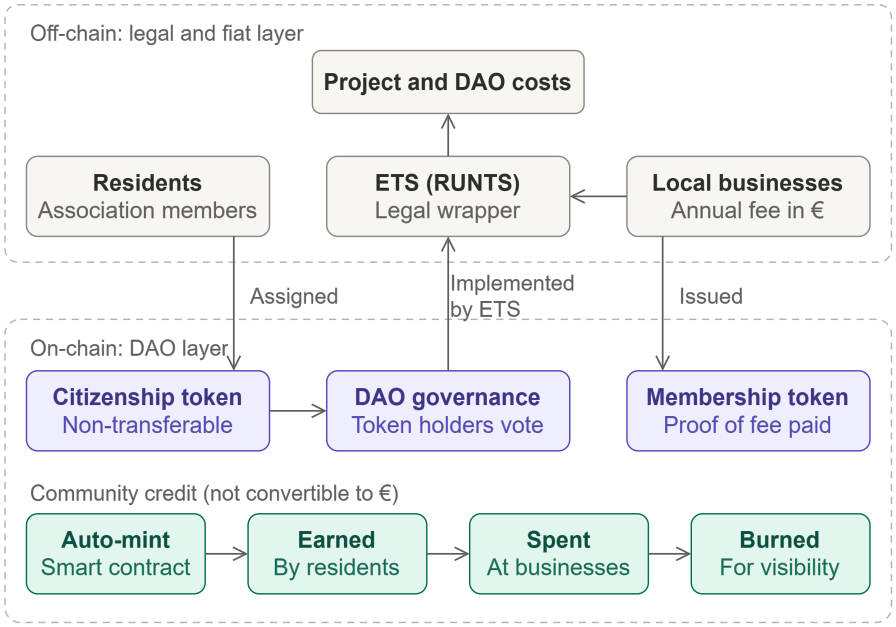


Figure 4.2: Private flow: legal wrapper, DAO governance and community credit circuit.

Enforcement of the law and relationship with the agreements: CivicDAO operates through collaboration agreements, that is, the agreements by which the Municipality and civic actors define, at the end of a shared planning process, the activities of general interest for the care and regeneration of urban common goods. [2]. Each agreement sets out the objectives of the collaboration, the planned activities and the limits of intervention.

The regulation provides for a specific case. Small, social, cultural and sporting events that do not require a public entertainment license can be held with a simple notification to the competent office, provided that the agreement has already defined their features within the established limits. For interventions that go beyond this scope, ordinary authorizations remain necessary.Two cases fall outside this simplified regime. The first is events open to the public beyond those limits, such as the occupation of public land: here the agreement grants exemptions from or reductions of the fee, but does not remove the authorization requirement. The second is works, which require building permits and, when they substantially alter the sites or their intended use, the preliminary review of the Municipal Executive Committee. On an operational level, the governance system examines the proposals and directs each one either toward autonomous execution, when it falls within the object and limits of the agreement, or toward review by the Municipality.

[2]Comune di Bologna, *Regolamento sulle forme di collaborazione tra soggetti civici e Amministrazione per lo svolgimento di attività di interesse generale e per la cura e la rigenerazione dei beni comuni urbani* [Regulation on forms of collaboration between civic actors and the Administration for carrying out activities of general interest and for the care and regeneration of urban common goods], adopted by City Council resolution of 14 November 2022, Rep. n. DC/2022/89, P.G. n. 769201/2022, in force since 1 January 2023.

4.3 Organizational Bodies

The first structural element of a DAO consists in identifying the members of the community and the various bodies granted with specific roles, functions, and rights [35]. The guide points out that rights within the different bodies may vary and not all members of a DAO necessarily hold the same rights. The structure of CivicDAO is deliberately asymmetric, since citizens, local businesses and the Municipality take part in it with radically different roles and prerogatives. The organizational bodies of the system are described below, indicating for each its role, composition, rights and entry mechanisms. An overall representation of the bodies and their relationships is provided in Figure 4.3.

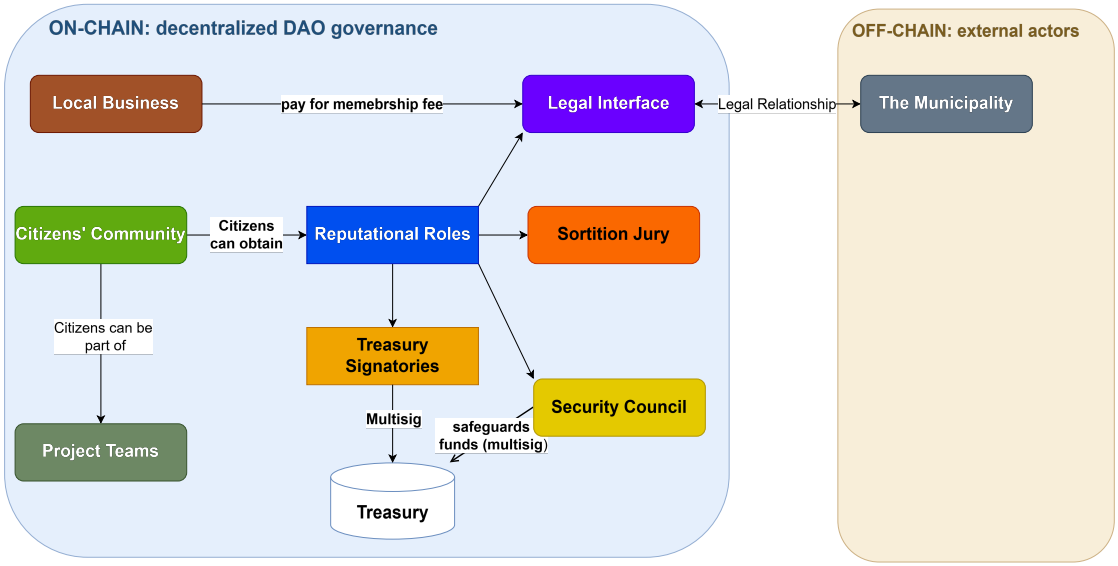


Figure 4.3: Organizational bodies of the CivicDAO and their relationships.

4.3.1 Citizens' Community

The community of resident citizens constitutes the sovereign body of the DAO. Its members are identified by the citizenship token (4.5.2) anchored to the SPID digital identity, according to the flow described in Section 5.3. In line with the guide's recommendation that access to the community should remain as open as possible in order to ensure inclusiveness [35], membership is open to every resident. Its only condition is the verification of residence following SPID authentication, which delimits the electorate on the basis of territorial anchoring without introducing an economic barrier. To this body applies the principle "one person, one vote", made effective by the non-transferability of the token, which prevents both the

concentration of voting power and Sybil attacks [3]. Citizens can submit proposals, take part in the discussion, vote and, if the proposal comes from the private flow, execute the proposals.

4.3.2 The Municipality

The Municipality is not a member of the DAO: it does not hold the citizenship token, it does not submit proposals and it does not vote. The Municipality is a body external to the community, which uses the DAO as an instrument for participatory budgeting, ensures that the activities and proposals of the DAO are managed in accordance with the collaboration agreement and the rules described in Section 4.2.3 and manage the on-boarding of citizens described in Section 5.3.

4.3.3 Local businesses

The neighborhood businesses take part in the DAO as members, through the acquisition of the annual business membership token described in Section 4.5.3. Businesses do not have the right to vote on civic proposal, and this exclusion prevents an evident conflict of interest, avoiding that an economic actor could influence deliberations concerning funds or initiatives from which it might draw direct benefit. Local businesses, through the purchase of the membership share, gain access to the Community Credit circuit (4.5.4), which are initially issued as a reward to citizens after the certified completion of a civic activity. The Community Credits are subsequently spent at local businesses by citizens and then burned by the local businesses to the DAO to purchase visibility and sponsorships. The participation of businesses is therefore of an economic and promotional nature, not a governance one.

4.3.4 Reputational Roles

The reputational system enables access to a set of functional roles reserved for citizens who have accumulated an adequate score. For example, membership in the jury drawn by lot for the resolution of disputes, membership in a given project team, participation in the security council, representation of CivicDAO in its relations with the Municipality through the legal interface described in Section 4.3.7 and the role of treasury signatory, whose composition, threshold, and mechanics are described in detail in Section 4.6.2. Since no individual should hold permanent positions, as this would lead to the formation of centralized authorities

[3] A Sybil attack is an attack in which a single actor creates and controls numerous fake identities in order to acquire disproportionate influence over the governance of the DAO.

[35], the reputational roles do not constitute stable offices but are subject to the reputation decay mechanism: the score decreases in the absence of continuous participation, so that privileges lapse if they are not sustained by an effective and lasting contribution.

4.3.5 Project teams

The realization of the individual civic projects is entrusted to project teams, that is a group of a temporary nature that come into being and dissolve together with the project to which they are dedicated. The members of the project teams are the citizens who contribute operationally to carrying out the activities. For them, following verification of the project's completion, are granted the participation certificates described in 4.5.5 the community credits described in 4.5.4 and an increase on reputation in accordance with what is established by the Incentives subsystem 4.7.

4.3.6 Jury drawn by lot

For the management of internal disputes a guarantor body is provided, composed of high-reputation citizens (4.3.4) selected by lot. The random selection, combined with the reputational requirement, aims to combine competence and impartiality, reducing the risk of capture of the body. The jury deliberates on the controversies and on the matters submitted to it. The principal occasion on which the jury is called upon is the verification of civic projects execution on the private flow.

4.3.7 Legal interface

It represents the entity endowed with legal personality, that is, a neighborhood committee or association, which acts as the legal interface of the DAO in its relations with the outside world, according to the approach described in (4.2.3).

The body is composed of high-reputation citizens, selected from the same pool as the reputational roles (4.3.4). This body assumes formal ownership of the legal relationships, but is devoid of deliberative power: it executes the decisions taken by the community, without substituting itself for it.

4.3.8 Security Council

Its function is exclusively protective in order to intervene in critical situations, such as key compromise, the exploit of a smart contract, suspected collusion among signatories, or anomalies in treasury flows, in order to rapidly secure the funds before

the damage becomes irreversible. The body is composed of high-reputation citizens, selected from the same pool as the reputational roles (4.3.4). The activation of the body requires a multi-signature internal to the council calibrated on a qualified majority, so that no member can act alone. It is bound to a predefined time horizon, after which the freeze automatically lapses if not renewed.

4.4 Decision-Making

4.4.1 The decision-making process

The decision-making process constitutes the second structural element and, as the guide notes, represents the most debated design challenge in the literature on DAOs [35].

4.4.2 Private Flow Process

On the private flow, the community directly governs its own internal resources and oversees the entire life cycle of a civic action: from its proposal, through discussion, voting, all the way to its execution, verification and reward. The life cycle of a private-flow project, after it has been funded, is structured as a state machine of the Figure 4.4, whose stages are described below.

1. **Proposal and discussion**: any citizen holding the citizenship token can propose a civic action on the private flow, such as cleaning up a park, organizing an event, redeveloping a shared space and, more generally, any activity falling within the scope of the private flow as defined in 4.2.3. The private-flow proposal is put to a vote together with the budget in euros required, the project team and the allocation of community credit to be assigned to participants. The proposal then goes through a discussion phase, which makes use of the channels described in 4.8 and of a non-binding preliminary poll based on the Snapshot model[4], through which the community compares positions, refines the proposal and filters out those lacking support before the binding vote is reached.

2. **Team formation and consensus**: project teams, described in 4.3.5, are temporary groups of citizens who commit to carrying out the project operationally.

[4] *Snapshot* is a widely adopted off-chain voting platform for blockchain-based communities. Participants express their choice by signing a message with their wallet instead of sending a transaction, so voting is gasless (it incurs no fee) and produces no on-chain effect.

Since team members will have to advance the cost of materials themselves and perform the activity, their membership provides explicit on-chain consent before the proposal is put to a vote. In this way, at the moment of deliberation the community votes on a real and already-committed team and there is no case in which a proposal lists citizens who have not consented or who do not intend to take part. The team, the budget in euros and the community credit allocation are therefore defined upstream and carried by the proposal itself.

3. **On-chain vote**: once the team's consent has been given and the discussion concluded, the proposal is submitted to a binding vote recorded on-chain. The vote is cast in gasless mode: the transaction is forwarded to a relayer and the gas is paid by the treasury 4.6.1. Every resident who has obtained the citizenship token has the right to vote according to the Token-based Quorum Mechanism model and thus implicitly on a one-person-one-vote basis. The quorum is set conservatively low at launch, with a review clause after a break-in period based on observed turnout. Revocable vote delegation is possible through liquid democracy, discussed in 3.4. Revocable delegation allows a member to entrust their voice, proposal by proposal, to whomever they consider most competent, while retaining at all times the power to withdraw it.

4. **Execution and submission of evidence (Submitted state)**: once the activity has been carried out on the ground, the team uploads the evidence package to IPFS[5]. The package contains documentation of the work performed (photographs and other supporting records), the receipts for the materials advanced and the list of actual performers, composed by the citizens who took part in the activity. The set of actual performers must be a subset of the team that gave its consent at the vote. The upload of the package then starts a verification time window.

5. **Optimistic verification**: by default it is assumed that what the team has attested corresponds to the truth and the community is granted a window within which to raise objections. If no citizen has challenged it by the time the window expires, the project is considered executed and transitions to the Confirmed state.

[5]IPFS (InterPlanetary File System) is a peer-to-peer protocol for distributed file storage that uses content-based addressing: each file is identified by a cryptographic hash of its content (a CID, or Content Identifier), making references immutable and verifiable. In blockchain systems, it is commonly used to store data off-chain while keeping only the CID on-chain as a verifiable reference.

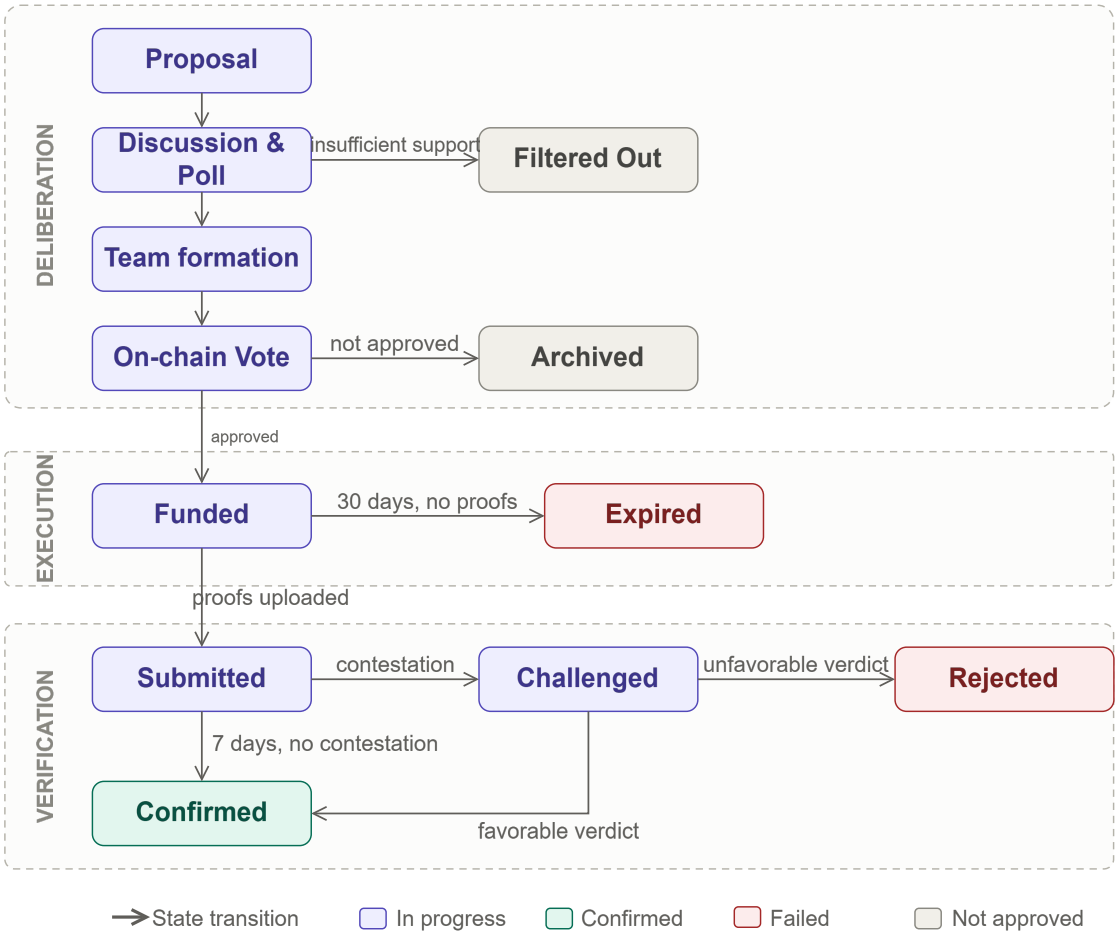


Figure 4.4: States of the private flow.

6. **Challenge and randomly drawn jury (Challenged state)**: in the event that a citizen challenges the package within the window, verification is referred to the jury described in 4.3.6, composed of high-reputation citizens drawn at random. The jury assesses the quality of the work performed, the appropriateness of the expenses documented by the receipts and the correctness of the list of performers. At the end, it issues a verdict that leads the project either to the Confirmed state or to the Rejected state. It should be noted that the challenge window protects the community with respect to the team (was the park really cleaned?) and the members with respect to one another, since a citizen who believes they were unfairly excluded from the list can raise a challenge, referring the matter to the jury.

7. **Confirmation and release of incentives (Confirmed state)**: confirmation is the positive terminal state and produces its effects in a single operation:

- The community credit allocation is divided equally among the actual performers according to the criterion established in 4.7;
- For each performer, the corresponding Proof of Participation attestation described in 4.5.5 is minted;
- The performers' reputation is increased, feeding access to the functional roles 4.3.4;
- An on-chain event is emitted that serves as the accounting basis for the reimbursement, by the legal interface, of the advances made by citizens for the materials, according to the mechanism described in 4.6.1.

8. **Withdrawal of a member**: since the team is fixed at the moment of the vote, the model accounts for the case in which a member does not complete their participation. In this case, their share of community credit is redistributed among the actual performers and no Proof of Participation attestation is minted in their favor. No active penalty is applied to reputation, since reputation is subject to decay 4.3.4. Non-participation simply translates into the absence of the increment that the others receive, while the prior score decays naturally.

9. **Failure and abandonment (Rejected and Expired states)**: the cycle provides for two negative outcomes. In the Rejected state, resulting from an unfavorable jury verdict or the Expired state, which occurs when a funded project never reaches the submission of evidence within a maximum deadline. In this case no incentive is released and the legal interface does not proceed with any reimbursement.

10. **Time parameters**: the cycle is governed by two distinct deadlines, anchored to different events:

    - The optimistic verification window, which runs from the submission of evidence, is set by default to *seven days*.
    - The expiry deadline, which runs from the funding of the project, is set by default to *thirty days* and identifies the maximum time within which the team is required to deliver the evidence. Since projects of different nature require different timeframes, both deadlines are conceived as parameters configurable at the proposal stage, while the seven/thirty-day pair remains the default value.

4.4.3 Public Flow Process: Participatory Budgeting

On this Flow, the DAO performs a single function: casting the on-chain vote 4.2.3. The resources made available by the Municipality are governed through the

instrument of participatory budgeting. The community decides how they are to be allocated, while the assessment of eligibility and feasibility, the appropriation of funds and the execution remain procedures of the administration. The process, represented in Figure 4.5, is structured in the following steps:

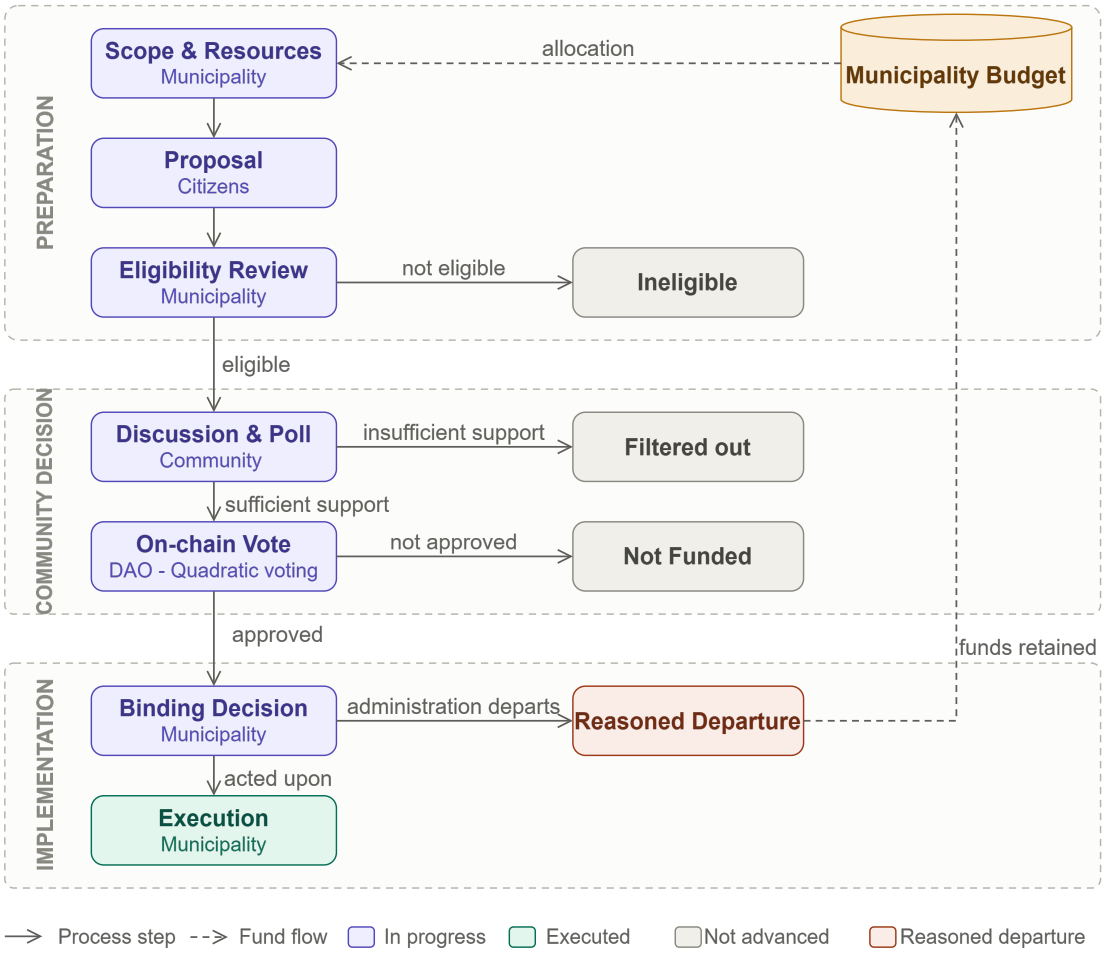


Figure 4.5: States of the Public flow.

1. **Definition of the scope and resources**: the Municipality allocates a share of its budget to participatory budgeting and defines its eligibility scope, on the basis of its own regulations and the collaboration agreement described in 4.2.3.

2. **Submission of proposals and eligibility review**: citizens submit spending proposals within the scope thus delimited. Each proposal is subjected to the Municipality's review of eligibility and feasibility.

3. **Preliminary discussion and non-binding poll**: before the on-chain vote, the deliberative phase makes use of discussion and of a non-binding poll based on the Snapshot model, which allows proposals lacking support to be filtered out and the debate to be steered without producing binding effects or costs for the citizen.

4. **On-chain vote with Quadratic Voting**: eligible proposals are submitted to a binding vote recorded on-chain. Each resident who has obtained the citizenship token is assigned an allocation of voting credits, which can be redistributed across the various proposals. Votes are tallied according to the Quadratic Voting model described in Section 3.4: each citizen has an equal allocation of credits and distributes them across the proposals, with a cost that increases quadratically with respect to the number of votes assigned to a single option.

5. **Effectiveness of the decision and duty to give reasons**: the outcome of the vote is binding by virtue of the commitment that the Municipality undertakes in advance through its own regulation (4.2.3), in order to act on the community's decisions according to the agreed procedure. Where the administration departs from it, it is required to provide specific reasons, archived in an immutable and publicly verifiable form.

6. **Execution**: here the DAO's scope of action has been exhausted, because appropriation and the actual execution rest with the Municipality.

4.5 Tokens

Tokens use smart contracts to set the rules of the organization. They show who is a member, decide who can access it, who can vote and create incentives for members to take part [35]. The model proposed here adopts a multi-token architecture, in which each token is assigned a distinct function within the neighborhood ecosystem.

4.5.1 Legal framework

The following assets support this economy, all configured, in line with the guide's recommendation [35], so as to avoid falling into the categories of asset-referenced tokens (ART) or e-money tokens (EMT), which are subject to the more stringent obligations of the European MiCA regulation, as described in Section **??**.

4.5.2 Citizenship Token

The citizenship token identifies the resident citizen, signals their membership in the community and enables their rights of proposal and vote. It is implemented as a Soulbound Token according to the ERC-5192 standard, and is therefore non-transferable and issued in a single unit per citizen, according to the SPID-anchored onboarding flow described in Section 5.3.1.

In MiCA terms, non-transferability places the token outside the financial perimeter. The token has no secondary market and no negotiable economic value. For this reason, it is neither an asset-referenced token nor an e-money token. It also does not have the characteristics of a financial instrument.

4.5.3 Membership token business

The business membership token is purchased on an annual basis by local businesses and grants access to the services offered by the DAO: visibility, sponsorship of neighborhood events and access to the community credit network. The membership token, issued to businesses that join the network, attests solely to their status as members. The consideration in euros is paid into the bank account of the legal interface. Once payment has been confirmed, the token is issued.

The cost does not constitute a purchase price, but rather the annual membership fee for joining the entity. The token, non-transferable, non-convertible into euros and devoid of any economic rights does not qualify as a crypto-asset within the meaning of MiCA, nor as a financial instrument, its function being exhausted in providing access to the network.

4.5.4 Community Credit

Community credits are the non-monetary reward granted to citizens for their operational participation in civic projects, redeemable at businesses that belong to the neighborhood circuit. The credit follows a circular, three-stage path that defines its closed-circuit nature.

1. The DAO's treasury issues the credits and assigns them to citizens as a reward for a certified contribution (4.7).

2. The citizen spends them at a participating business, receiving goods, services, or discounts in return.

3. The business returns the collected credits burning them. In exchange for this return, the DAO provides the business with visibility and sponsorship services,

closing the cycle.

Therefore community credit never leaves the circuit in the form of money: the credits are neither transferable between citizens nor convertible into euros and they circulate exclusively along this closed loop. Community credits are minted at the moment participation in a project is certified. They are not backed by a reserve in the treasury, but function like a loyalty-points system. The credit represents the recognition of a contribution and derives its value from being redeemable within the circuit. This model, however, introduces two critical issues, namely inflationary risk and merchant trust. There is an inflationary risk because the supply of credits grows with every reward and uncontrolled issuance would progressively dilute their perceived value. Meanwhile, in the absence of real backing, a business that accepts the credits relies solely on the continuity of the circuit. If that trust break down, merchants might stop accepting them, risking the collapse of the circuit.

4.5.5 Sustainability NFTs and Certificates

The system finally provides for a family of non-fungible tokens with a certifying, non-financial function, configured as soulbound credentials: non-transferable, tied to the holder's identity, and devoid of any right to income or capital flows.

The businesses' annual membership token enables access to the circuit's services and it is itself an NFT that certifies the business's contribution to the neighborhood for the year in question.
Alongside there are proofs of participation (Proof of Participation), which document involvement in a specific civic project along with the date, place, and activity carried out.

Their non-fungible and non-transferable form places these credentials outside the scope of MiCA, which excludes unique and non-fungible assets. The role of these instruments in the incentive architecture is examined in greater depth in Section 4.7.

4.6 Treasury

The CivicDAO's treasury reflects the two-flow structure described in Section 4.2.3 and is organized into two distinct components, corresponding to different technical layers and custody regimes:

1. The on-chain treasury, governed directly by the community through a multi-signature wallet (multisig) and intended to hold the assets that circulate on

the decentralized governance layer.

2. The off-chain euro component, held in an ordinary current account registered to the legal interface (4.3.7), which nonetheless operates in a bound position: the account's movements execute the resolutions adopted on-chain by the community, without any margin for autonomous determination.

The euro resides off-chain and is not convertible into community credit, which in turn cannot be purchased with euros nor converted into legal tender. The two compartments therefore remain logically distinct. The only link is represented by the reserve earmarked for covering the on-chain costs of the DAO's operations.

4.6.1 Treasury Assets and Their Use

The assets managed by the system are three, heterogeneous in nature, origin and function:

1. The first is the euro, held off-chain in the Legal Interface's account. Its origin is the annual membership fee paid by local businesses in exchange for the business membership token (4.5.3). Its use is double: covering the DAO's operating costs and reimbursing citizens for the expenses they have advanced to purchase the materials needed to carry out civic projects.

2. The second is the community credit, a non-convertible unit of exchange that circulates on-chain within the closed circuit described in 4.5.4.

3. The third is the reserve earmarked for on-chain transaction costs. The voting model adopted is gasless: the citizen signs their preference without bearing any cost, while the gas for the transaction forwarded by the relayer is borne by the treasury. The reserve for paying gas is held in a euro-pegged stablecoin (EUR-pegged, for example EURC), funded by the controlled conversion of a portion of the treasury's euros.

   The choice of a stablecoin is because it remains stable over time and is not exposed to the volatility inherent in cryptocurrencies. The conversion into the native token, necessary for the actual payment of gas, takes place just-in-time, that is, only at the moment and to the extent that payment is due, so as to minimize exposure.

   From a legal standpoint, the EUR-pegged stablecoin constitutes a crypto-asset within the meaning of Regulation (EU) 2023/1114 (MiCA), more precisely an e-money token and as such its use remains confined to the technical function of covering network costs, without taking on any investment role.

4.6.2 Signatories and Key Management

The on-chain treasury is held via a multi-signature wallet configured with a 4-of-7 threshold: seven signatories in total, four of whom are necessary and sufficient to authorize a movement of funds. The signatories are selected from the pool of reputational roles (4.3.4).

Two signing layers must also be distinguished. Movements on the on-chain layer are authorized by the 4-of-7 multisig, whereas operations on the euro current account are instead signed by the legal interface (4.3.7), which acts in a position bound to the outcome of the on-chain resolutions.

4.7 Incentives

The purpose of the incentive instruments is not to remunerate members, but to align their behavior with the DAO's social purpose 4.2.1 and, above all, to sustain participation. Consistently with the project's non-profit orientation and with the MiCA constraints 4.5.1, no incentive lever confers a monetary return or rights to income flows.

4.7.1 The Hooks for Incentivization

The system rests on three distinct levers:

- **Reputation**: non-transferable and subject to decay, it does not weight the vote but enables access to functional roles 4.3.4. This lever draws on the game-theoretic incentive model of Meneguzzo and Pierri [36], in which a reputation-based mechanism penalizes prolonged inactivity and rewards sustained engagement to preserve a DAO's vitality. Unlike that model, however, CivicDAO deliberately omits the monetary-reward component, consistently with its non-profit orientation and the MiCA constraints (**??**).
- **Community Credits**: they reward operational participation in civic projects and are redeemable within the closed neighborhood circuit.
- **Proofs of Participation**: they document involvement in a specific project.

4.8 Communication

The last structural element is communication, which the guide identifies as fundamental for coordinating activities, making decisions, maintaining transparency

within the DAO and fostering a sense of belonging and involvement among members [35]. In the context of a civic DAO, the sense of belonging is rooted in a pre-existing territorial community and transparency shifts from an internal value to accountability toward the collective.

4.8.1 On-chain and Off-chain Channels

On-chain communication constitutes the official channel for decisions and critical information: recording votes, submitting proposals, verifying treasury transactions. In this model, the on-chain layer is where activities that produce effects are consolidated, such as the outcome of a vote, the disbursement of a fund, or the issuance of a certificate.

Off-chain communication, by contrast, hosts the deliberative and relational phase: the discussion of proposals, the confrontation of differing perspectives, the building of consensus prior to the vote. The model places four different tools:

1. A non-binding preliminary poll, conducted according to the Snapshot model. It allows the citizen to sign their preference off-chain at no cost, filtering out proposals lacking support before they reach the on-chain vote.

2. The neighborhood forum, which hosts the structured discussion of proposals, organized by thread, where arguments remain traceable and consultable.

3. The messaging channels, which support operational coordination and informal day-to-day exchange.

4. In-person off-chain communication which, unlike the typical DAO which must compensate for the geographic dispersion and anonymity of its members by relying entirely on digital tools, can count on the physical co-presence of citizens. It includes neighborhood assemblies, public meetings and physical notice boards in shared spaces and participating businesses.

In any case, only the formal outcome is committed to the blockchain, not the debate that precedes it.

Chapter 5

Implementation

5.1 Technology Stack

The technology stack adopted for the implementation of CivicDAO is the following:

- **Solidity**: high-level programming language adopted as the reference standard for the development of smart contracts targeting the Ethereum Virtual Machine (EVM);

- **OpenZeppelin**: library of standard contracts subjected to security audits, providing reference implementations of the main ERC standards and of recurring primitives, such as access control, pausability and reentrancy protection. Its use reduces the surface for errors and promotes adherence to the best practices recognized by the community;

- **Foundry**: development framework written in Rust, used for the compilation, testing and deployment of smart contracts. Foundry enables writing tests directly in Solidity, ensuring reduced execution times and a high degree of integration with the development environment;

- **Anvil**: local Ethereum node integrated into Foundry, employed as a test network to simulate the behavior of the blockchain in a controlled and isolated environment, without transaction costs and with immediate confirmation times.

The combination of these tools made it possible to cover the entire development cycle of the contracts, namely from writing them through to their execution in a simulated environment.

5.2 General Architecture

The system is organized into four layers, each with a distinct responsibility. Governance commands, the economy executes. In particular, upon confirmation of a project it is CivicProject that calls the economy contracts (minting of credit, attestations, reputation), instead ParticipatoryBudget only attests to the scores expressed by the community on-chain and the outcome recorded by the Municipality. Another fundamental separation concerns write authorities, because citizenship is written by the Municipality, whereas credit, attestations and reputation are written by the DAO.

Table 5.1: CivicDAO contract architecture organized by layers.

Level	Contracts	Responsibilities
1 - Identity	CitizenshipSBT	Establishes who belongs to the community, voting and proposal gate.
2 - Governance	ParticipatoryBudget, CivicProject	Public flow (participatory budget) and private flow (civic projects).
3 - Economy	CommunityCredit, MerchantMembership, ProofOfParticipation, Reputation	Closed-loop community credit, business membership, attestations, reputation.
4 - Infrastructure	(future work)	Multisig treasury, gasless relayer, security council, legal interface, sortition jury.

The implementation presented in this chapter covers the first three layers of the architecture which realize the core mechanisms of the model and demonstrate its end-to-end feasibility. The fourth layer, concerning the operational infrastructure, falls outside the scope of this work and are discussed among the possible future developments in Chapter 7.

5.3 Identity and Residency

Before retracing the flow, it is worth delimiting what this chapter actually implements. The on-chain verification endpoint, the *mintWithAuth* function with its four checks and the soulbound enforcement, described in Section 5.4 is fully implemented.

The surrounding *issuance and institutional integration*, the SPID/SAML authentication with the Municipality as Service Provider, the residency query to the

civil registry (Anagrafe), the derivation of the `nullifier` as `HMAC(server_secret, tax_code)` and the EIP-712 signing of the authorization are *not* realized in this work, because it presupposes the real infrastructure and institutional authority of a public administration (custody of the signing key, access to the registry, accreditation as a SPID Service Provider) and therefore lies outside the scope of a prototype.

First, the process for establishing who belongs to the community is defined and so, how citizens obtain the token they will use to vote. The minting of the token is the endpoint of a flow that keeps two distinct phases separate:

- *authentication* of identity via SPID;
- *authorization* based on residency via the Municipality;

SPID is a national identity: it proves *who* the citizen is, but it does not attest to residency in a given district. From a technical standpoint, SPID operates with SAML 2.0 *Web Browser SSO* and the assertion returned by the Identity Provider carries identity attributes, not the "resident yes/no" outcome. Residency is data that lives in the civil registry and it is the Municipality that produces that outcome by querying the registry with the tax code obtained from authentication.

The Municipality, therefore, acts as a SAML Service Provider toward the SPID Identity Provider in order to obtain the identity. Then, once obtained, it verifies residency in its own registry and returns the outcome of the response to the CivicDAO platform. The complete sequence is shown in Figure 5.1.

5.3.1 On-boarding and Token minting flow

The following phases retrace the numbered messages of the sequence diagram in Figure 5.1. Phases 1-6 constitute the design of the proof of possession to anchor the wallet address to the citizen's identity, phases 7-9 constitute authentication via SPID, while phases 10-11 constitute the minting of the token after verification of the necessary conditions.

1. **Onboarding initiation and wallet connection.** The citizen initiates the onboarding from the DAO dashboard by connecting their wallet. This will be the address entered in the `addressTo` field of the authorization, to which the SPID identity will be anchored.
2. **Authentication request via SPID.** After connecting the wallet, the user must authenticate via SPID and is redirected to the Issuer, the Municipality.

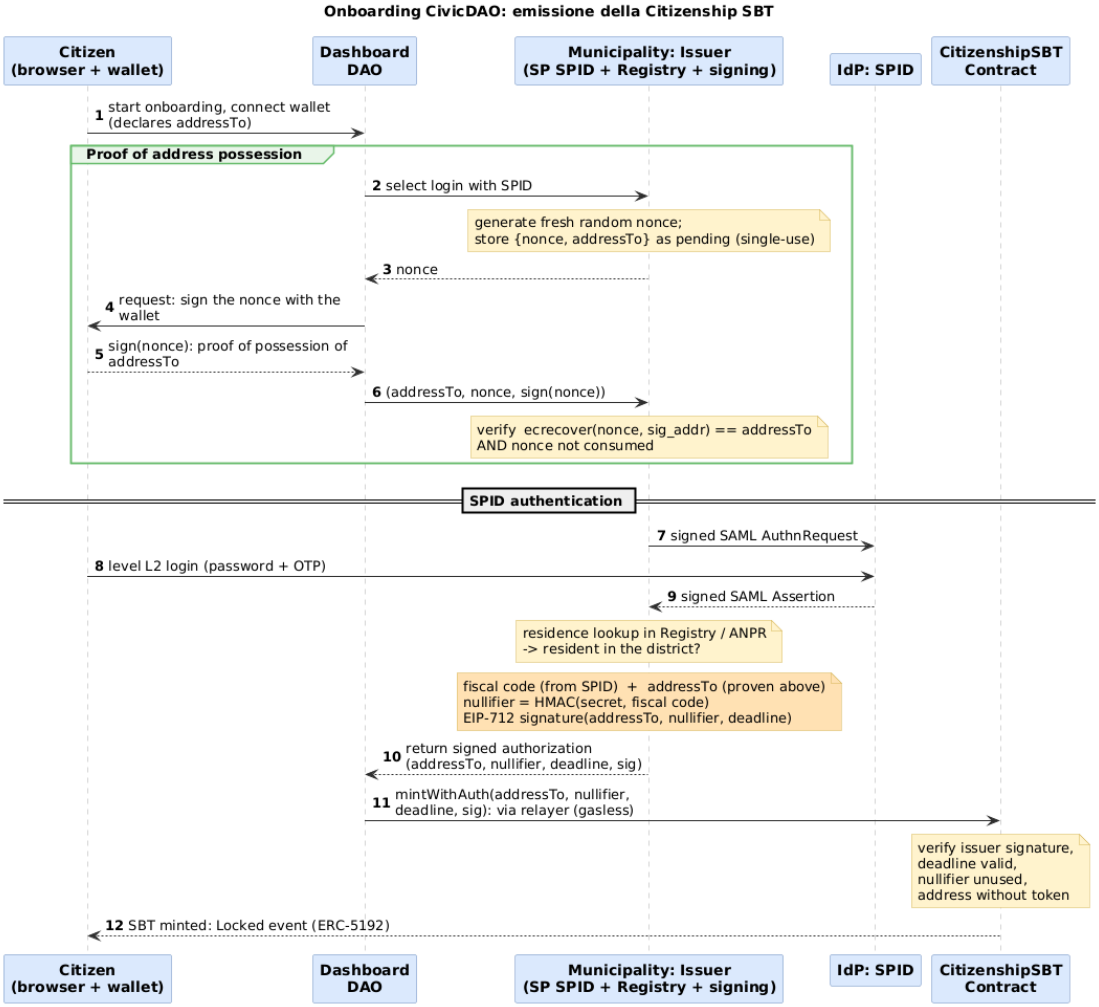


Figure 5.1: On-boarding flow of Citizenship token

Upon receiving the request, the Municipality generates a random, fresh, single-use nonce, and stores it as *pending* in association with `addressTo`: in this way the Municipality will later be able to verify its validity and prevent its reuse.

3. **Sending the nonce.** The Municipality returns the nonce to the dashboard.

4. **Request to sign the nonce.** The dashboard asks the citizen to sign the nonce with the wallet's private key.

5. **Signing the nonce (proof of possession).** The citizen signs the nonce with the private key. The signature constitutes a *proof of possession* of the address, since only the holder of the private key can produce it and it will be verified by the Issuer.

6. **Verification of the proof of possession and consumption of the nonce.** The dashboard forwards to the Municipality the triple *(addressTo, nonce, sig_addr)*. The Municipality recovers the signer via *ecrecover*, an ECDSA function, then verifies that it matches *addressTo*, checking that the nonce is still pending and marking it as consumed. This verification *ecrecover(nonce, sig_addr) == addressTo* proves that the citizen actually controls the wallet, while the consumption of the nonce prevents the replay.

7. **SAML AuthnRequest.** Once the session is established, the Municipality, in its role as Service Provider, redirects the citizen to the Identity Provider and sends a signed SAML *AuthnRequest*, initiating the authentication protocol.

8. **L2-level login.** The citizen authenticates with the IdP using level 2 SPID credentials (password + OTP).

9. **Signed SAML Assertion and residence verification.** If authentication is successful, the IdP returns to the Municipality a signed *SAML Assertion* containing the tax code (codice fiscale), the attribute that identifies the person with certainty. Once the tax code is received, the Municipality queries the Registry Office (Anagrafe) to verify that the person resides in the neighborhood, a requirement that guarantees their membership in the community.

10. **Generation of the nullifier and return of the signed attestation.** In the event of a positive outcome, the Municipality computes *nullifier = HMAC(server_secret, tax_code)*, obtaining an opaque 32 byte identifier that is *deterministic* and dependent on the *key*. The Municipality then signs, using EIP-712, a structured message that binds three fields: *addressTo* (the address whose possession was proven at step 6), *nullifier* (the person in opaque form) and *deadline* (expiry), returning to the dashboard the signed package *(addressTo, nullifier, deadline, sig)*. EIP-712 implicitly binds the signature

to this contract and to this chain. The signed *addressTo* is the same address proven at step 6 and the *nullifier* binds the authorization to the identity ascertained via SPID.

11. **Sending the minting transaction.** The minting transaction is sent to the contract by calling *mintWithAuth(addressTo, nullifier, deadline, sig)*, typically through a *relayer* that pays the gas in *gasless* mode. Upon successful validation, the token is minted.

5.4 CitizenshipSBT.sol

CitizenshipSBT is the identity foundation on which both governance flows rest: it defines the electorate of the public track (*ParticipatoryBudget*) and it is the citizenship required by the private track (*CivicProject* and *Reputation*).

It is an ERC-721 token made soulbound according to the ERC-5192 standard and integrated with EIP-712 for mint authorization. It satisfies two functional requirements: minting subject to identity verification through the entity's signature and the uniqueness and non-transferability of the token.

Roles and state A single privileged role, *ISSUER_ROLE*, identifies the Municipality as the authorized issuer. The type of the signed message is declared only once as *MINT_AUTH_TYPEHASH*, and two data structures underpin the contract's guarantees:

```
bytes32 public constant ISSUER_ROLE = keccak256("ISSUER_ROLE");

bytes32 public constant MINT_AUTH_TYPEHASH =
    keccak256("MintAuth(address addressTo,bytes32 nullifier,uint256 deadline)");

// nullifiers already consumed: anti-replay
mapping(bytes32 => bool) public nullifierUsed;
// mint minus burn, used by ParticipatoryBudget and CivicProject
uint256 private _totalSupply;
```

The *nullifierUsed mapping* keeps track of the authorizations already spent, preventing their reuse. Instead *_totalSupply* is an electorate counter used by *CvicProject.sol* and *ParticipatoryBudegt.sol* that will be discussed later.

mintWithAuth: the onboarding authorized by the Municipality

Minting is exposed by *mintWithAuth* and the function is *permissionless* with respect to the sender: anyone can send the transaction because the legitimacy

of the operation does not depend on who sends it, but on the signature that accompanies it.

```
function mintWithAuth(
    address addressTo,
    bytes32 nullifier,
    uint256 deadline,
    bytes calldata sig
) external {
    // (a) temporal validity
    if (block.timestamp > deadline) revert DeadlineExpired();

    // (b) authenticity and authorization of the signature (EIP-712 + ISSUER role)
    bytes32 digest = hashMintAuth(addressTo, nullifier, deadline);
    address signer = ECDSA.recover(digest, sig);
    if (!hasRole(ISSUER_ROLE, signer)) revert InvalidIssuerSignature();

    // (c) no reuse of the nullifier
    if (nullifierUsed[nullifier]) revert NullifierAlreadyUsed();

    // (d) uniqueness of the address (one person, one token)
    if (balanceOf(addressTo) != 0) revert AlreadyCitizen();

    nullifierUsed[nullifier] = true;
    uint256 tokenId = _nextId++;
    _mint(addressTo, tokenId);

    emit Locked(tokenId);                          // ERC-5192: soulbound forever
    emit CitizenshipMinted(addressTo, tokenId, nullifier);
}
```

The body performs four checks in sequence. A single failure causes the *revert* of the entire transaction:

1. the *deadline* not yet passed;
2. the signature produced by an address with *ISSUER_ROLE* recovered by *ECDSA.recover* on the EIP-712 digest;
3. the *nullifier* not yet used;
4. the address not already holding a token.

A single failure causes the *revert*. Once the checks are passed, the contract marks the nullifier as used, mints the token to *addressTo* and emits the *Locked* event required by ERC-5192, which makes the token permanently non-transferable (*soulbound*). From this moment the citizen is a member of the community.

hashMintAuth: the EIP-712 digest

The digest to be signed is computed by a public function, so that the Municipality can produce and sign it off-chain with its own wallet:

```
function hashMintAuth(address addressTo, bytes32 nullifier, uint256 deadline)
    public view returns (bytes32)
{
    return _hashTypedDataV4(
        keccak256(abi.encode(MINT_AUTH_TYPEHASH, addressTo, nullifier, deadline))
    );
}
```

The use of *_hashTypedDataV4* (EIP-712) implicitly binds the signature to this contract and to this chain. A signature produced for a different instance or a different network would not pass, protecting against cross-contract or cross-chain reuse of the authorization.

Soulbound and electorate counter (_update)

Non-transferability is enforced through the override of *_update* provided by OpenZeppelin. The same function keeps the electorate counter up to date:

```
function _update(address to, uint256 tokenId, address auth)
    internal override returns (address)
{
    address from = _ownerOf(tokenId);
    if (from != address(0) && to != address(0)) revert NonTransferable();
    // mint
    if (from == address(0)) _totalSupply++;
     // burn
    else if (to == address(0)) _totalSupply--;
    return super._update(to, tokenId, auth);
}
```

The central condition lets through only minting (*from* null) and *reverts* with *NonTransferable* on any transfer between two non-null addresses. The contract does not define any burn to implement the revocation of citizenship.

5.5 Governance

5.5.1 ParticipatoryBudget.sol

It implements the only on-chain function of the Public flow 4.4.3, that is Quadratic Voting on a list of proposals already declared eligible off-chain by the Municipality. The contract only attests to the scores expressed by the community and the outcome recorded by the Municipality. Eligibility screening, appropriation, and execution remain procedures of the Municipality.

Roles and Round Status

The contract inherits exclusively from *AccessControl* and defines a single privileged role, *MUNICIPALITY_ROLE*, which the Municipality uses to open rounds and record their outcome. Each round is described by a *struct* that sets its parameters at opening:

```
contract ParticipatoryBudget is AccessControl {
    bytes32 public constant MUNICIPALITY_ROLE = keccak256("MUNICIPALITY_ROLE");
    uint256 public constant QUORUM_BPS = 1_000;
    uint256 public constant BPS_DENOMINATOR = 10_000;

    ...

    struct Round {
        uint256 creditsPerVoter; // QV credit budget C, equal for every citizen
        uint64 start;
        uint64 end; // inclusive deadline
        uint256 electorate; // CitizenshipSBT.totalSupply() snapshot at open
        uint256 voterCount; // distinct citizens who voted
        uint256 proposalCount;
        Status status;
        bool quorumReached; // computed at close
    }
    ...
}
```

Two fields deserve attention:

- *electorate*: the size of the electorate is not read at closing time but frozen as a snapshot of *CitizenshipSBT.totalSupply()* at the moment the round opens, avoiding grants or revocations of citizenship occurring while voting is underway;

- *QUORUM_BPS/BPS_DENOMINATOR*: expresses the quorum threshold (10%) avoiding fractions and any integer division in the quorum calculation.

Opening the round and freezing the slate

```
function openRound(uint256 creditsPerVoter,
string[] calldata proposalCids, uint64 votingDuration)
        external
        onlyRole(MUNICIPALITY_ROLE)
        returns (uint256 roundId)
    {
        // create the Round
        r.electorate = citizenship.totalSupply();
       // register all proposalCids in a single pass:
    }
```

The entire list of CIDs is passed to *openRound*, so that the set is defined at the very moment voting opens and no proposal can be added while voting is underway. Only the Municipality, holder of *MUNICIPALITY_ROLE*, can open a round.

Community writes in the contract

```
    function castVotes(uint256 roundId,
    uint256[] calldata proposalIds, uint256[] calldata votes)
    external {
        //guard: active round before end
        for (uint256 i = 0; i < proposalIds.length; i++) {
            uint256 pid = proposalIds[i];
            if (i != 0 && pid <= proposalIds[i - 1]) revert ProposalsNotSorted();
            if (pid >= r.proposalCount) revert InvalidProposal();

            uint256 v = votes[i];
            if (v == 0) revert ZeroVotes();

            cost += v * v; // quadratic cost
            _proposals[roundId][pid].score += v; // running tally
        }

        if (cost > r.creditsPerVoter) revert BudgetExceeded();

        hasVoted[roundId][msg.sender] = true;
        creditsSpent[roundId][msg.sender] = cost;
        r.voterCount += 1;

        emit VotesCast(roundId, msg.sender, cost);
    }
```

castVotes is the only point where the community writes to the contract's state. For the specified round, it receives the list of voted proposals and the votes assigned to them, applies the quadratic cost, and updates the tally.

According to Quadratic Voting implementation, for each round every citizen has an equal budget of vote credits (default 100). Casting v votes for a proposal costs v^2 credits, the constraint is that the sum of the costs must not exceed the budget. Voting credits are not tokens and in order to avoid to compute square roots on-chain votes are passed, not credits. It follows that the maximum number of votes that can be concentrated on a single proposal is the square root of the budget (10, with a budget of 100).

Permissionless and quorum closure without rounding

```
function closeRound(uint256 roundId) external {
    //everybody can invoke after end
    r.status = Status.Closed;
    r.quorumReached = r.voterCount * BPS_DENOMINATOR >= r.electorate * QUORUM_BPS;

}
```

Closing is *permissionless*: anyone can invoke it once the window has expired. The quorum calculation is rewritten in multiplicative form $voterCount \cdot BPS_DENOMINATOR \geq electorate \cdot QUORUM_BPS$ rather than comparing a percentage obtained through division. This avoids the EVM's integer division and, with it, any possible rounding.

The codified obligation to state reasons

```
function recordOutcome(uint256 roundId, uint256 proposalId,
    Outcome outcome, string calldata reasonsCid) external
        onlyRole(MUNICIPALITY_ROLE)
    {
        ...

        if (outcome == Outcome.NotExecuted && bytes(reasonsCid).length == 0)
        revert ReasonsRequired();

        p.outcome = outcome;
        p.reasonsCid = reasonsCid;

        ...
    }
```

If the outcome is *NotExecuted* the function requires the IPFS CID of the written justification, otherwise it fails. The chain does not store the text of the justification, but only its CID to obtain Immutability (any change to the text changes the CID), Attributability (it is recorded by an address holding 'MUNICIPALITY_ROLE') and Timestampability (from the transaction's timestamp).

5.5.2 CivicProject.sol

The private flow governs the entire lifecycle of a civic project managed directly by the community, from proposal to confirmation or rejection, passing through team formation, voting, execution, and verification as described in Section 4.4.2 and shown by the Figure 4.4.

From the Architecture point of view *CivicProject* does not inherit *AccessControl*, because each transition is protected by a specific check, such as *balanceOf* to verify citizenship, the team/consensus *mappings* for the team, *IJury.isJuror* for the jury, or is *permissionless* once the relevant time window has expired. Moreover *CivicProject* depends only on interfaces, never on concrete economy contracts: it imports *ICommunityCredit*, *IProofOfParticipation*, *IReputation*, *IJury*, not their implementations.

In the conceptual model 4.4.2 the proposal goes through a discussion phase and a team-formation phase, but on-chain these phases do not exist as distinct states. Discussion is an off-chain moment, and team formation is resolved entirely in the collection of the binding consent of its members: the first step after *propose* is *consentToTeam*, which, once the last consent is reached, automatically opens the vote.

Structs and constants

```
struct Project {
    address proposer;
    string description;
    address[] team;
    // community-credit units earmarked for performers
    uint256 creditAllocation;
     // seconds, from submitEvidence to the optimistic deadline
    uint64 verificationWindow;
    // seconds, from Funded to the execution deadline
    uint64 executionWindow;
    Status status;
    uint256 consentCount;
    uint64 votingEnd;
```

```
        // CitizenshipSBT.totalSupply() snapshot taken when voting opens
        uint256 electorate;
        uint256 votesFor;
        uint256 votesAgainst;
        uint256 voterCount;
        uint64 fundedAt;
        uint64 executionDeadline;
        string evidenceIpfsHash;
         // subset of team submitted as having actually done the work
        address[] performers;
        uint64 verificationDeadline;
    }


    uint64 public constant DEFAULT_VERIFICATION_WINDOW = 7 days;

    uint64 public constant DEFAULT_EXECUTION_WINDOW = 30 days;

    uint64 public constant VOTING_DURATION = 3 days;

    uint256 public constant QUORUM_BPS = 1_000;

    uint256 public constant BPS_DENOMINATOR = 10_000;

    uint256 public constant REPUTATION_INCREMENT = 120;
```

Note the distinction between *team* and *performers*. The team is the set of those who committed at the time of the proposal, the performers are those who actually did the work. Only the *performers* receive rewards.

Proposal and binding consent of the team

```
function propose(
    string calldata description,
    address[] calldata team,
    uint256 creditAllocation,
    uint64 verificationWindow,
    uint64 executionWindow
) external returns (uint256 projectId) {
    if (citizenship.balanceOf(msg.sender) == 0) revert NotCitizen();
    if (team.length == 0) revert EmptyTeam();
// register the project; verify that every team member is a citizen
// and does not appear twice (DuplicateTeamMember)
}
```

Proposing is reserved for citizens and does not allow empty teams or teams with duplicate members. Once the project is registered, every team member must give their consent on-chain before the community can vote:

```
function consentToTeam(uint256 projectId) external {
    Project storage p = _getProject(projectId);
    if (p.status != Status.Proposed) revert InvalidStatus();
    if (!isTeamMember[projectId][msg.sender]) revert NotTeamMember();
    if (hasConsented[projectId][msg.sender]) revert AlreadyConsented();

    hasConsented[projectId][msg.sender] = true;
    p.consentCount++;
    emit TeamConsentGiven(projectId, msg.sender);

    if (p.consentCount == p.team.length) {
        p.status = Status.Voting;
        p.votingEnd = uint64(block.timestamp) + VOTING_DURATION;
        // snapshot, as in ParticipatoryBudget.openRound
        p.electorate = citizenship.totalSupply();
        emit VotingOpened(projectId, p.votingEnd);
    }
}
```

Voting opens automatically only when all members have confirmed their participation. If even a single consent is missing, the state remains *Proposed* indefinitely: no one can force the vote to open.

Voting and closing

```
function vote(uint256 projectId, bool support) external {
// one-person-one-vote
}

function closeVoting(uint256 projectId) external {
    Project storage p = _getProject(projectId);
    if (p.status != Status.Voting) revert InvalidStatus();
    if (block.timestamp <= p.votingEnd) revert VotingStillOpen();

    bool quorumReached =
    p.voterCount * BPS_DENOMINATOR >= p.electorate * QUORUM_BPS;
    bool approved = quorumReached && p.votesFor > p.votesAgainst;
    if (approved) {
        p.status = Status.Funded;
        p.fundedAt = uint64(block.timestamp);
        p.executionDeadline = p.fundedAt + p.executionWindow;
```

```
            emit ProjectFunded(projectId, p.executionDeadline);
        } else {
            p.status = Status.Archived;
            emit ProjectArchived(projectId);
        }
    }
```

Here voting is one-person-one-vote. *closeVoting* is *permissionless*, so anyone can close it after the deadline. Approval requires both the absolute quorum and a simple majority of votes in favor.

Fo the quorum the same measure is used as *ParticipatoryBudget* (snapshot of the electorate at vote opening, threshold in basis points) but with a different effect. In the public flow the quorum is only attested (a flag), because the execution depends from Municipality. Here it is a condition of validity. In fact *closeVoting* must yield a definite Funded/Archived outcome, so a project that fails the quorum is Archived.

The *Funded* state authorizes the team to start and triggers the timer of the execution window, but the euro budget is never moved by the contract. Consistently, the "refund" in the *Archived*, *Rejected* and *Expired* states means that the euro refund is not triggered off-chain while the community credit allocation and the PoP are not minted.

Optimistic verification, protest and jury

```
  function submitEvidence(uint256 projectId,
  string calldata ipfsHash, address[] calldata performers) external {
    // only a team member, only in Funded state, within executionDeadline;
    // `performers` (a subset of the team) becomes
    the distribution base for minting
}

function challenge(uint256 projectId) external {
   // any citizen, within verificationDeadline -> Status.Challenged
}

function confirmOptimistic(uint256 projectId) external {
    // anyone, after verificationDeadline, if no one has challenged -> _settle()
}

function resolveChallenge(uint256 projectId, bool approve) external {
    if (!jury.isJuror(msg.sender)) revert NotJuror();
    if (approve) _settle(projectId, p); else { p.status = Status.Rejected;
    ...
```

```
    }
}
```

If no one challenges within the window, *confirmOptimistic* confirms the project without the need for any external judgment. Conversely, if a citizen challenges, the project moves to *Challenged* and the matter passes to the jury, which through *resolveChallenge* issues the verdict, which may be favorable (confirmation) or unfavorable (*Rejected*, no minting). If instead no proof arrives within *executionDeadline*, the project lapses through *expire()* into the *Expired* state.

The *performers* list passed to *submitEvidence* is the subset of the team that actually worked and becomes the base for minting. A member who does not appear among the *performers* simply receives nothing, and their share is implicitly redistributed because the division is performed by *performers.length*.

The effects of confirmation

Settle is the point where the private track activates the real economy:

```
function _settle(uint256 projectId, Project storage p) internal {
    p.status = Status.Confirmed;

    uint256 n = p.performers.length; // > 0: enforced at submitEvidence
    // Equal split. Integer division: any remainder from a
    // non-exact division simply stays unminted (dust), not held anywhere.
    uint256 quota = p.creditAllocation / n;

    for (uint256 i = 0; i < n; i++) {
        address performer = p.performers[i];
        communityCredit.mint(performer, quota);
        proofOfParticipation.mint(performer, projectId);
        reputation.increase(performer, REPUTATION_INCREMENT);
    }

    emit ProjectConfirmed(projectId);
    emit ReimbursementDue(projectId, p.proposer, n, block.timestamp);
}
```

First, the state moves to *Confirmed*. Next, the *creditAllocation* endowment is divided into equal shares among the *performers* only. The division is integer, so any remainder is left unminted (a deliberate choice, to avoid introducing additional and complex remainder-distribution logic). For each performer, three distinct external calls are made: minting of the credit, issuance of the participation attestation and

reputation increment.

ReimbursementDue carries no money. The event that closes the confirmation does not contain the euro amount (which was never on-chain) but only the elements an off-chain system needs to compute and attribute it (project identifier, proposer, number of performers, and timestamp).

5.5.3 The real closed-loop economy

When a civic project reaches confirmation, *CivicProject* produces concrete economic effects by delegating them to three real contracts that implement the boundary interfaces: *CommunityCredit*, *MerchantMembership*, and *ProofOfParticipation*.

CommunityCredit.sol

CommunityCredit implements *ICommunityCredit* and is an ERC-20 standard token spendable at the businesses enrolled in the circuit. The closed loop described in Section 4.5.4 unfolds in three stages:

1. **Minting**: upon confirmation of a project, *CivicProject* mints (*mint*) the credit to each performer.
2. **Spending**: the citizen transfers (*transfer*) their credits to a business in the circuit. The operation is allowed *only* toward a business with an active enrollment.
3. **Redemption and burn**: the business calls *redeem*: its own balance is burned and the total supply decreases, closing the loop.

The destination constraint of the spending phase is handled through the override of *_update* provided by OpenZeppelin, which governs minting, transfer, and burn together:

```
bytes32 public constant MINTER_ROLE = keccak256("MINTER_ROLE");

...
function mint(address to, uint256 amount) external onlyRole(MINTER_ROLE) {
        _mint(to, amount);
    }

function _update(address from, address to, uint256 value) internal override {

        if (from != address(0) && to != address(0) &&
        !merchantMembership.isActiveMember(to)) {
```

```
            revert NotAuthorizedRecipient(to);
        }
        super._update(from, to, value);
    }
```

For every actual transfer, the recipient must be a business with an active enrollment, otherwise the transaction fails with *NotAuthorizedRecipient*.

The closing of the circuit happens with *redeem*:

```
function redeem(uint256 amount) external {
    if (!merchantMembership.isActiveMember(msg.sender))
    revert NotAuthorizedRecipient(msg.sender);
    _burn(msg.sender, amount);
    emit Redeemed(msg.sender, amount);
}
```

The visibility and sponsorship service that the DAO provides in exchange for the redemption remains entirely off-chain: the *Redeemed* event is merely its accounting basis, following the same scheme as *ReimbursementDue* in *CivicProject*.The periodic cap on issuance (*cap*) and the decay of unspent balances are explicitly out of the scope of this version and deferred to future work on the tokenomics.

MerchantMembership.sol

MerchantMembership is a soulbound ERC-721 token, like *CitizenshipSBT*, but with a significant domain difference: a business *accumulates* one certificate per payment (the annual renewals), so re-minting to an address that already holds a certificate is here allowed by design.

```
bytes32 public constant ISSUER_ROLE = keccak256("ISSUER_ROLE");

uint64 public constant MEMBERSHIP_DURATION = 365 days;

function mint(address business) external onlyRole(ISSUER_ROLE) returns
(uint256 tokenId) {
    tokenId = _nextId++;
    uint64 paidAt = uint64(block.timestamp);
    _data[tokenId] = MembershipData(paidAt);
    lastPaidAt[business] = paidAt;
    _mint(business, tokenId);
    emit MembershipPaid(business, tokenId, paidAt);
}

function isActiveMember(address business) public view returns (bool) {
```

```
        uint64 paidAt = lastPaidAt[business];
        return paidAt != 0 && block.timestamp - paidAt <= MEMBERSHIP_DURATION;
    }
```

The function *isActiveMember* is the only one that *CommunityCredit* consults in its own *_update*, and for this reason it is designed as a read based on the last payment recorded in *lastPaidAt*. The membership fee is paid entirely off-chain, to the third-party service entity (ETS). The contract only certifies the *fact* that it occurred and its date.

Proof Of Partecipation

ProofOfParticipation is a soulbound badge minted to every performer of a confirmed project, so accumulation is possible: one for each confirmed project the citizen took part in.

```
    bytes32 public constant MINTER_ROLE = keccak256("MINTER_ROLE");

    ...

    struct TokenData {
        uint256 projectId;
        uint64 mintedAt;
    }

    mapping(uint256 => TokenData) private _data;

    function mint(address to, uint256 projectId) external onlyRole(MINTER_ROLE) {
        uint256 tokenId = _nextId++;
        uint64 mintedAt = uint64(block.timestamp);
        _data[tokenId] = TokenData(projectId, mintedAt);
        _mint(to, tokenId);
        emit ParticipationMinted(to, tokenId, projectId);
    }

    function tokenData(uint256 tokenId) external view returns
    (uint256 projectId, uint64 mintedAt) {
        if (_ownerOf(tokenId) == address(0)) revert TokenDoesNotExist();
        TokenData storage d = _data[tokenId];
        return (d.projectId, d.mintedAt);
    }
```

Here the timestamp (*mintedAt*) is taken internally from *block.timestamp* and not passed by the caller, so the attestation cannot be *backdated*. Moreover, there

are no URIs or IPFS reference: the only on-chain payload is "which project" and "when", readable through *tokenData*.

5.5.4 Reputation

Reputation is a standalone contract, not anchored to *CitizenshipSBT*, which reads *CitizenshipSBT.balanceOf* only to confirm that the subject is a citizen. It is non-transferable by construction and never weighs the vote. Governance remains one-citizen-one-vote everywhere in the system, while reputation serves solely to unlock functional roles by threshold (e.g. jury, signers), verified through *hasReputation*. The increment occurs for each performer exclusively upon confirmation of a project, at the same point where credits and attestation are minted.

Lazy decay (pull, not push)

Reputation must be able to decay over time, but on an EVM a contract cannot execute code by itself. Indeed, there is no way to "trigger" a decrement at the expiry of an interval, while relying on an external *keeper* that periodically updates the scores would introduce a gas-costly dependency. The solution adopted is a *lazy* decay: no function ever actively decrements the score. For each citizen only two values are stored, namely the raw score and the timestamp of the last write, while the current value is *computed at read time*, applying a linear decay from that timestamp up to *block.timestamp*. In a nutshell: reads compute, writes crystallize.

```
struct Score {
    uint256 value;
    uint64 updatedAt;
}

uint256 public immutable decayRatePerDay;

mapping(address => Score) private _scores;

function reputationOf(address citizen) public view returns (uint256) {
    Score storage s = _scores[citizen];
    return _decayed(s.value, s.updatedAt);
}

function _decayed(uint256 value, uint64 updatedAt)
internal view returns (uint256) {
    if (value == 0 || updatedAt == 0) return value;
    uint256 elapsedDays = (block.timestamp - updatedAt) / 1 days;
```

```
        uint256 drop = elapsedDays * decayRatePerDay;
        return drop >= value ? 0 : value - drop;
    }
```

The *value* field of the *Score struct* is therefore not the current score, but the score *at the moment* of *updatedAt*. The actual value is the one returned by *reputationOf*, which is the only one to use in every read.

Function increase: crystallization before writing

Adding the new increment to the *stale* raw value would cause either the loss or the incorrect counting of the decay accrued between two participations. For this reason *increase* first *crystallizes*, that is, it freezes the value decayed up to now, and only afterwards adds the increment, resetting the timestamp to the current instant:

```
    function increase(address citizen, uint256 amount)
    external onlyRole(SCORER_ROLE) {
        if (citizenship.balanceOf(citizen) == 0) revert NotCitizen();

        Score storage s = _scores[citizen];
        uint256 crystallized = _decayed(s.value, s.updatedAt);
        s.value = crystallized + amount;
        s.updatedAt = uint64(block.timestamp);

        emit ReputationIncreased(citizen, amount, s.value);
    }
```

Without the preventive crystallization, the decay accumulated since the last update would be silently reset to zero at the moment of the addition. Write permission is reserved for the *SCORER_ROLE* role, which at deploy time is granted to *CivicProject*.

Calibration of the decay

The observable behavior of reputation follows from the *joint* choice of two constants: the increment assigned at each confirmation, *REPUTATION_INCREMENT* (defined in *CivicProject*, equal to 120), and the daily decay rate, *decayRatePerDay* (set to 1 in the deploy script and made *immutable* in the constructor of *Reputation*).

The reason is that, with *decayRatePerDay* equal to 1, the ratio between increment and rate is directly a number of days. In this way a single contribution of 120 points takes exactly $120/1 = 120$ days, four months of total inactivity, to reach zero.

A citizen who takes part in one confirmed project per month arrives at the next project with about $120 - 30 = 90$ residual points from the previous contribution (75% of the maximum), to which crystallization adds the new increment.

The threshold for functional roles

Functional roles are a simple threshold on the lazy value:

```
function hasReputation(address who, uint256 threshold)
external view returns (bool) {
    return reputationOf(who) >= threshold;
}
```

Since *hasReputation* reads the *computed* value and not the raw one, eligibility for a role decays *automatically* as soon as the score drops below the threshold. Finally, it should be stated as future work that *hasReputation* is today ready as a primitive but *not* yet hooked up to *Jury*, which remains minimal (designated juror). Weighted random selection based on reputation is deferred to subsequent developments.

5.5.5 Boundary of the implementation

The scope realized covers the first three levels of the architecture: identity, governance, and economy. The fourth level, the infrastructural one, and some refinements remain future work. Moreover, in several cases what is missing is already preceded by a "hook" present in the code, which makes the future developments a natural extension.

Revocation of citizenship A natural development is the addition of a burn function reserved for the Municipality (through *ISSUER_ROLE* or a dedicated role), useful for example in case of a change of residence. The electorate accounting is already in place: *_update* handles the burn branch and decrements *_totalSupply*, so that the quorum of the public track would remain correct without further modifications. The real design decision concerns instead the *nullifier*, in fact allowing the re-onboarding of those who legitimately return would require releasing it upon revocation (setting *nullifierUsed* back to false), which in turn requires storing at mint time the association between token and *nullifier*.

Real jury and random selection The *IJury* interface is today satisfied only by *MockJury* (designated juror). A real implementation would follow the same scheme already proven for *CommunityCredit*, *ProofOfParticipation*, and *Reputation*, and would add the random selection of jurors and eligibility by reputational threshold.

The hook already exists: *Reputation.hasReputation* is ready as a primitive, simply not yet hooked up to jury selection.

Liquid democracy Revocable, per-matter vote delegation is not present in the private track, but only in the design. Delegation would graft onto the voting mechanism without altering its cost or quorum.

Tokenomics of *CommunityCredit* The closed loop is complete in its three stages (minting, spending, redemption with burn), but two refinements of the tokenomics remain out of scope: a periodic cap on issuance (*cap*) anchored to the value actually redeemed by the businesses and a decay of unspent credit balances over time.

Infrastructural level It is the fourth level, entirely left to subsequent developments. It is composed by:

- An on-chain multi-signature *treasury* (a 4-of-7 scheme) with a stablecoin reserve to cover gas;
- *Gasless* voting via relayer (ERC-2771), which would make the entire flow usable without the citizen having to own ether;
- A *Security Council* contract for emergency response;
- A contract for the *Legal Interface* that formalizes on-chain the connection with off-chain administrative acts.

Web dApp The user interface, wallet connect, and dashboard for proposals and voting, is not the subject of this chapter, which concerns only the on-chain contracts. It is mentioned for completeness, as the layer that would make the system accessible to a non-technical citizen.

Chapter 6

Results

The implementation realizes the core of CivicDAO as a set of on-chain smart contracts written in Solidity, developed with the Foundry framework and OpenZeppelin libraries and tested in Anvil. It covers three of the four layers of the architecture presented in Section 5.2, namely identity, governance, and economy, leaving the infrastructural layer to future work. Within this boundary the prototype is complete and end-to-end: a citizen can be admitted to the community, the community can propose and a confirmed decision can produce real economic effects.

Concretely, the identity layer is realized by CitizenshipSBT (Section 5.4), a non-transferable token that binds membership to a single verified identity and maintains an on-chain count of the electorate. The governance layer is realized by two contracts that keep the public and the private flow rigorously separate. ParticipatoryBudget (Section 5.5.1) implements the quadratic-voting round of the public flow, in which the municipality opens the round, freezes the slate of eligible proposals and is bound to record a reasoned outcome. On the other hand CivicProject (Section 5.5.2) governs the entire lifecycle of a community-run civic project, from proposal and team formation to one-person-one-vote approval, optimistic verification and settlement. The economy layer is realized by CommunityCredit, MerchantMembership, and ProofOfParticipation (Section 5.5.3), which together close a local credit loop and issue the non-transferable attestations of participation. The last element of the economy is the Reputation (Section 5.5.4), a non-transferable, time-decaying score that never weighs a vote and serves only to unlock functional roles. Taken together, these contracts show partially the mechanisms designed in Chapter 4 that can be expressed in executable code.

6.1 Answering the Research Questions

This section returns to the questions that guided the work and assesses the extent to which they have been answered.

RQ. Through which on-chain governance mechanisms can a civic DAO keep a legally responsible public body within the process as a guarantor of compliance, *without* that control degenerating into a re-centralisation that nullifies citizen participation?

Sub-question	Mechanism	Realized in
SQ1 Legitimacy from *residency*, not capital	Soulbound one-person-one-vote citizenship and equal per-voter credit in quadratic voting, electorate counted and snapshotted on-chain.	CitizenshipSBT, ParticipatoryBudget
SQ2 Public body *in*, no veto on the merit	Separate private/public flows, codified obligation to state reasons for any non-execution, off-chain legal wrapper for standing toward third parties.	ParticipatoryBudget, CivicProject, Legal interface*
SQ3 Sustained, non-perverse participation	Closed-loop community credit and non-transferable, decaying reputation that never weighs a vote, rewards only to actual performers, issuance cap and decay planned.	CommunityCredit, ProofOfParticipation, Reputation

* *Specified in the design. Part of the infrastructural layer deferred to future work (Chapter 7).*

Figure 6.1: How the design and its implementation answer the research question.

SQ1 asked how the legitimacy of participation could be grounded in residency rather than in capital, neutralizing the plutocratic concentration of voting power while preserving on-chain verifiability. The answer is realized in CitizenshipSBT (Section 5.4). Membership is expressed by a soulbound token whose non-transferability ensures that voting rights cannot be bought, sold, or accumulated. In the private flow decisions follow a one-person-one-vote rule and in the public flow the equal per-voter credit budget of quadratic voting gives every citizen the same voice while still letting the intensity of preference be expressed. What the on-chain layer cannot do by itself, namely guarantee that behind each token stands

one and only one resident, is delegated to the identity scheme (ANPR residency and SPID authentication) and constitutes one of the residual limitations discussed in Section 6.2.

SQ2 asked how a legally responsible public body could be kept within the governance process and how the community could be given legal standing toward third parties, without granting the administration a discretionary veto over the merit of proposals. The design answers this by splitting governance into two flows and by circumscribing the municipal role in each. In the private flow the municipality has no role at all. The community proposes, votes and verifies civic actions on its own. In the public flow the municipality participates, but its powers are confined to procedural acts, opening a round, fixing the list of proposals that has already passed an eligibility screening and recording the outcome. The prohibition of a veto on the merit is made effective by a codified obligation to state reasons, in fact recordOutcome cannot register a non-execution unless it is accompanied by the immutable, attributable and time-stamped reference to a written justification (Section 5.5.1). Legal standing toward third parties is provided by the hybrid architecture, in which an off-chain legal wrapper represents the community externally. The on-chain legal-interface contract that would formalize the correspondence between on-chain outcomes and administrative acts is, however, part of the infrastructural layer deferred to future work, and the residual gap this leaves is examined in Section 6.2.

SQ3 asked how participation could be sustained over time against voter apathy, through incentive mechanisms that avoid the perverse effects to which incentive design is notoriously prone. The economic and reputational layer answers this with a set of mutually reinforcing devices. Confirmed participation is rewarded on three planes at once, a share of community credit, a non-transferable proof-of-participation badge and an increment of reputation. All three are minted only upon the verified confirmation of a project and only to the citizens who actually performed the work (Sections 5.5.3). Community credit is spendable only within a closed loop of enrolled local businesses and is burned on redemption, so that its value is tied to real local exchange. Reputation is non-transferable, decays over time when participation stops and never weighs a vote. It can only unlock functional roles, which keeps the incentive to participate from turning into an accumulation of decision-making power. To prevent simulated participation registered for the sole purpose of harvesting credit, the design responds by planning an issuance cap on contributions tied to the value actually redeemed by the businesses and a decay of unspent balances (Section 5.5.5). The extent to which these devices actually overcome apathy is one of the limitations discussed in Section 6.2.

Read together, the three answers compose the response to the overarching question: through which on-chain governance mechanisms a civic DAO can keep a legally responsible public body within the process as a guarantor of compliance without that control degenerating into a re-centralization that nullifies citizen participation. The mechanism is the joint effect of a design that separates a purely community-governed private flow from a public flow strongly anchored to the municipality. The mechanism grounds the right to participate in a soulbound, residency-based citizenship rather than in capital. It makes the advancement of the process independent of the administration's will, through permissionless transitions and a codified obligation to state reasons. Finally, it sustains engagement through a closed-loop economy and a non-vote-weighting reputation. The mechanisms do not dissolve the tension between accountability and decentralization, but they relocate it. The public body remains in few points explicit and verifiable.

6.2 Limitations

This chapter analyzes CivicDAO limitations and, where possible, it offers directions for future work.

6.2.1 Residual centralization at the onboarding gateway

For the minting of the citizenship token, someone must attest that a given applicant is in fact a resident. In the current project this authority is the municipality, which holds the exclusive right to mint the token. In theory, the municipality need only certify a fact it already holds in the population registry (ANPR), without any kind of judgment. An administration cannot alter the outcome of a vote, but it can refuse or delay a citizen's enrollment, turning the onboarding phase into a subtler instrument of exclusion. The risk could be eliminated by returning control of the residency attestation to the citizen through the EUDI/IT-Wallet framework is discussed among the future developments in 7.

6.2.2 Sybil resistance is only as strong as the identity layer

The one-person-one-vote guarantee is not enforced on-chain. The blockchain can ensure that a citizenship token is non-transferable and that a given token votes only once, but it cannot in itself guarantee that behind every token there is a single citizen. This property is inherited from the onboarding scheme, formed by the combination of ANPR residency data and SPID authentication. The consequence is that the integrity of the entire legitimacy model is conditioned by the soundness of the authentication and authorization scheme. Thus, should the identity-verification layer be circumvented, an attacker would be able either

to obtain multiple citizenships or to obtain citizenship without being a resident, creating an asymmetry of voting power. The project attempts to design a robust scheme that prevents these problems by relying on national identity providers. However the guarantee remains an assumption at the boundary of the system and the mechanism that makes it operational is specified at the design level and only partially realized in the implementation (5.3).

6.2.3 Excluding socio-technical prerequisites

Anchoring participation to a verifiable on-chain identity is what gives the model its binding and verifiable character, but it presupposes that every resident possesses a wallet, a digital identity, and a minimum level of digital literacy. The risk is that of excluding the non-technically portion of the population of which a neighborhood is composed. According to this the system reproduces the under-representation already observed in token-based governance (3.8.4). In this under representation classical DAOs skew participation in favor of the largest holders, a residency-based but digitally mediated DAO risks skewing it in favor of those who possess digital skills. The literature on the applicability of DAOs to the public sector is explicit in holding that the benefits materialize only when certain preconditions are met, such as digital literacy, administrative capacity and infrastructural stability. The project lowers the barriers in future implementations 5.5.5, without eliminating, through gasless voting, so that the citizen need not hold ether for transaction fees, off-chain communication channels, to preserve a non-technical point of access and an accessible web dApp with assisted onboarding.

6.2.4 Optimistic verification presupposes an active and honest minority

To avoid subjecting every civic action to a mandatory and hard review, verification is optimistic: a submitted project is assumed to be truthful unless a citizen challenges it within a bounded time window. Optimistic verification is effective only as far as the community is willing to monitor and to raise objections and the empirical evidence on voter apathy (3.8.4) shows that such willingness cannot be taken for granted. If apathy prevails, the absence of challenges cause the verification step to degenerate into an automatic ratification. The randomly-drawn jury that handles a challenge presupposes a sufficient pool of high-reputation citizens to draw from, yet reputation itself accumulates only through prior participation, so that the mechanism is weakest precisely in its initial phase, when such a pool has not yet formed. The realization of a true sortition jury, subject to a reputation threshold, together with the empirical calibration of the challenge window and the quorum, is deferred to future work.

6.2.5 The hybrid on-chain / off-chain gap

Decentralized on-chain governance paired with a traditional legal wrapper gives the community legal standing toward third parties. The price of bridging two orders that obey different rules is that the correspondence between them is assumed rather than proven. Blockchain technology cannot force an administrative act to follow from an on-chain outcome, it can only make it transparent, whereas the legal system does not guarantee that an off-chain act will be faithfully reflected on-chain. Between the two lies an oracle gap that the design bridges with a legal-interface role but does not eliminate. This is the mirror image of the limitation attributed to established civic platforms in Section 3.9, which record proposals and votes yet cannot prevent an administration from ignoring the outcome. The legal interface is intended to convert the community's decisions into acts the administration is bound to honor, but its capacity to do so is a legal question that the technical design cannot settle on its own.

6.2.6 The irreducible tension of incentive design

The incentive layer, composed by non-transferable reputation, closed-loop community credit and participation certificates, exists to counter the apathy that afflicts token-based governance. The problem is that engineering incentives can produce the cobra effect. This consists in simulated participation: contributions registered for the sole purpose of accumulating credit, devoid of any genuine interest in the projects themselves [35]. The model responds, in future implementations (5.5.5), on two planes: a periodic issuance cap anchored to the value actually collected by the merchants and a decay over time of unspent credit balances, discouraging their accumulation.

Chapter 7

Conclusions and Future Work

Digital participatory platforms have shown that citizens can be involved at scale in the decisions that concern the places they inhabit. These platforms collect proposals and record votes, yet nothing in their operation binds an administration to the outcome (Section 3.9). Decentralized autonomous organizations promise precisely the verifiable execution that such platforms lack, but they bring problems, such as plutocratic concentration of voting power, chronic apathy, exposure to governance attacks and an uncertain legal status (Section 3.8). Between these two, the existing public-sector applications of DAOs had left open a question how a legally responsible public body can remain within the governance process as a guarantor of compliance without that role collapsing into a re-centralization that empties citizen participation of its substance (Section 3.10).

This chapter summarizes how the thesis has addressed that question and which problems have been examined. Moreover it outlines the directions along which the work can be extended.

7.1 Summary of the Work

The work began by outlining the technological foundations on which any DAO rests, such as Blockchain (distributed ledgers), the cryptographic primitives that guarantee their properties and the smart-contract model introduced by Ethereum. Then state of the art has been reviewed on decentralized autonomous organizations, their governance structures, voting mechanisms, token standards and regulatory classification. This review identified the limitations that any civic application must confront and positioned the research gap with respect to both non-blockchain civic platforms and existing civic DAOs.

From that gap the thesis derived the design of CivicDAO, a neighborhood-scale DAO for the participatory care of common goods. Its central design choices are the following. Hybrid architecture that pairs on-chain governance with a Third Sector entity (ETS) acting as legal wrapper, a membership grounded in verified residency rather than capital and expressed by a non-transferable citizenship token. Additionally, the separation of a private flow, in which the community autonomously governs its own civic actions, from a public flow, in which participatory budgeting takes place while the Municipality retains a bounded, non-discretionary compliance role. Finally a closed-loop local economy and a reputation layer that sustain participation without conferring monetary returns.

The design was realized as a set of smart contracts written in Solidity, built on the OpenZeppelin libraries, developed and tested with Foundry and executed on a local Anvil node. The implementation covers three of the four architectural layers: identity (`CitizenshipSBT`), governance (`ParticipatoryBudget` and `CivicProject`) and economy (`CommunityCredit`, `MerchantMembership`, `ProofOfParticipation` and `Reputation`). A citizen can be admitted to the community, the community can propose, vote, verify a civic action and a confirmed decision produces concrete economic effects. Finally, the results were assessed against the research questions and the residual limitations of the model were discussed openly.

The source code of the implementation, including the smart contracts, the test suite and demonstration demos, is publicly available at the following repository:

`https://github.com/smeneguz/tesi-civic-dao`

7.2 Addressed Problems and Adopted Solutions

This section starts from the problems identified in the state of the art 3.8 and, for each of them, explains the solution adopted and whether that solution has been implemented or only specified in the design. Table 7.1 provides a compact overview.

Plutocratic concentration of voting power CivicDAO removes voting power from capital, in fact the right to vote derives from a soulbound citizenship token, one per verified resident, that cannot be bought, sold or accumulated. The private flow applies a strict one-person-one-vote rule, while the public flow uses quadratic voting over an equal credit budget for every citizen. Reputation never weighs

a vote. *Implemented* in `CitizenshipSBT`, `ParticipatoryBudget`, `CivicProject` and `Reputation`.

Governance attacks In CivicDAO voting power cannot be borrowed, because it resides in a non-transferable token that can only be obtained through an identity-bound authorization. Moreover, the electorate is frozen as a snapshot at the opening of each vote and in the public flow the slate of proposals is fixed when the round opens, so that neither the electorate nor the set of options can be manipulated while voting is underway. *Implemented* in the mechanism.

Administrations that ignore or postpone the outcome The central weakness of established civic platforms is that an administration can disregard the result of a participatory process without leaving any trace (Section 3.9). CivicDAO addresses it with two mechanisms. The first is a codified obligation to state reasons: the Municipality cannot record the non-execution of a winning proposal unless it attaches the IPFS reference of a written justification, which becomes immutable, attributable and time-stamped. The second is that every transition that depends only on the passing of time, such as closing a vote or confirming an unchallenged project is permissionless, so the process advances even if the public body refuses to act. *Implemented* in `ParticipatoryBudget` and `CivicProject`.

Uncertain legal status and personal liability CivicDAO adopts a hybrid architecture in which an ETS registered in the RUNTS, and endowed with legal personality, represents the community toward third parties. Instead the on-chain resolution is the form through which the members' assembly expresses its will. The separation of the two flows keeps the Municipality's procedures and the community's autonomous activities under distinct legal regimes. Furthermore all tokens are configured so as to fall outside the categories of asset-referenced and e-money tokens regulated by MiCA (Section 3.6.2). *Specified in the design.* The on-chain legal-interface contract belongs to the infrastructural layer.

Voter apathy and the limits of delegation CivicDAO rewards verified participation on three planes at once. Community credit spendable only within a closed circuit of local businesses, a soulbound proof of participation and a reputation that decays in the absence of continued engagement. Rewards go only to the citizens who actually performed the work and since reputation unlocks functional roles but never adds voting weight, the incentive to participate does not turn into an accumulation of decision-making power. *Implemented* in `CivicProject`, `CommunityCredit`, `ProofOfParticipation` and `Reputation`.

Verification of off-chain civic actions A civic DAO must decide on actions, such as cleaning a park, that take place in the physical world and cannot be observed by a smart contract. CivicDAO adopts an optimistic scheme: the evidence submitted by the team is assumed truthful unless a citizen challenges it within a time window, in which case a jury decides. This avoids a mandatory review of every action while protecting both the community from the team and the team members from one another. *Implemented* in `CivicProject`, with the jury currently represented by a minimal mock.

Vulnerability of code The implementation relies on audited OpenZeppelin primitives. They keep each contract small and focused on a single responsibility, makes the governance contracts depend only on interfaces and uses a Security Council with a time-bounded freeze as a last line of defense. These measures mitigate the risk but do not remove it. *Partially implemented*, the Security Council belongs to the infrastructural layer.

Table 7.1: Problems identified in the state of the art and solutions adopted in CivicDAO.

Problem	**Adopted solution**	**Status**
Plutocracy	Soulbound citizenship, one person one vote, equal quadratic-voting budget, reputation that never weighs the vote	Implemented
Governance attacks	Non-borrowable voting power, electorate snapshot, slate frozen at opening	Implemented
Outcome ignored by the administration	Codified obligation to state reasons, permissionless time-based transitions	Implemented
Legal status and liability	ETS legal wrapper, separation of flows, tokens outside MiCA ART/EMT	Designed
Apathy and delegation	Closed-loop credit, proofs of participation, decaying reputation, rewards only to actual performers	Implemented (delegation designed)
Verification of civic actions	Optimistic verification with challenge window and jury	Implemented (mock jury)
Code vulnerability	Audited libraries, modular contracts, Security Council	Partial

7.3 Future Work

Future developments are organized so that each of them answers one or more of the limitations discussed in Section 6.2. They range from completing the implemented architecture to introducing new mechanisms and validating the model in a real context.

7.3.1 Completing the Architecture

The first direction is the completion of the elements already specified in the design and listed in Section 5.5.5. For many of which a hook is already present in the code. The infrastructural layer comprises the 4-of-7 multi-signature treasury with its stablecoin reserve for gas, gasless voting through a relayer compliant with ERC-2771, the Security Council contract, the Legal Interface contract and the jury should become a real sortition body.

The remaining refinements are the following. The revocable per-proposal delegation of liquid democracy in the private flow, the revocation of citizenship upon change of residence, a web dApp with assisted onboarding to make the system accessible to non-technical residents and the tokenomics of community credit, namely a periodic issuance cap anchored to the value actually redeemed by businesses and a decay of unspent balances.

7.3.2 Self-Sovereign Residency through IT-Wallet and the EUDI Wallet

In the on-boarding scheme the Municipality verifies residency and signs each authorization individually, so it could in principle refuse or delay the enrollment of specific citizens. The European Digital Identity framework offers a way to remove this gatekeeping role. The way is enabling citizens to store electronic attestations of attributes under their sole control, to disclose them selectively and to authenticate through pseudonyms where full identification is not required.

In this scenario the citizen would obtain a residency attestation in their own wallet and present it to CivicDAO without the Municipality intervening in the specific enrollment. The Municipality would remain the source of the data, but would no longer be able to discriminate against a particular applicant, because the attestation is general-purpose and not issued for the DAO. Open issues remain, such as the availability of these attestations in the wallet, the cost of verifying their signatures on chain and the fact that access to some attestations requires an electronic identity card, a potential digital barrier discussed in Section 3.8.

7.3.3 Conviction Voting for Participatory Budgeting

The public flow currently allocates resources through discrete rounds of quadratic voting. Conviction voting, introduced in Section 3.4, consists in citizens allocating their support to proposals over time. The conviction behind each proposal grows the longer that support is maintained and a proposal passes as soon as its conviction exceeds a threshold that increases with the share of the available funds it requests. Such a mechanism lowers the attention cost of participation, since a citizen does not have to vote at a specific moment, makes last-minute mobilization ineffective letting small proposals pass faster than large ones.

Adapting conviction voting to CivicDAO requires each citizen receives an equal, non-transferable support budget, analogous to the credit budget of the current quadratic voting. The mechanism should be evaluated as an alternative or a complement to quadratic voting rather than as its replacement, comparing the two through simulation before any adoption.

7.3.4 Ballot Secrecy and Collusion Resistance

In the current implementation every vote is a public transaction linked to an address. This maximizes verifiability, but it has a cost: a citizen can prove to a third party how they voted, which makes vote buying and coercion possible. Furthermore in a small community the link between an address and a person may be easy to establish.

A promising direction is the Minimal Anti-Collusion Infrastructure (MACI) [37, 38], which uses encryption and zk-SNARKs to keep individual votes private while publicly proving the correctness of the tally. Its key property is that a voter cannot prove how they voted, which removes the basis of any bribe. MACI, however, relies on a coordinator who processes the encrypted messages and could see individual votes. In a civic setting this role must not be assigned to the Municipality, as it would reintroduce the very centralization the model avoids and distributing it among several independent parties remains an open problem.

7.3.5 Governance of Parameters and Upgradeability

The design foresees that the quorum is set conservatively at launch and revised after an initial period based on observed turnout (Section 4.4.2). Otherwise in the implementation the quorum, the voting duration, the reputation increment and the decay rate are constants fixed at deployment. A future version should make these parameters modifiable only through a dedicated type of community proposal, executed with a timelock so that every change is announced in advance and can be contested. The same logic applies to upgradeability: correcting a bug in a deployed contract requires an upgrade mechanism. The upgrade authority should

therefore rest with the community through the same timelocked process, leaving to the Security Council only the power to pause the contracts in an emergency.

7.3.6 Deployment, Security Assurance and Empirical Validation

The prototype has been executed only on a local node. A deployment on a public test network would make it possible to measure the actual gas consumed by each operation and thus to size the stablecoin reserve provided by the treasury design. On the security side, the test suite should be extended with fuzz and invariant testing, for instance verifying that each nullifier yields at most one citizenship token or that the credit minted for a project never exceeds its allocation.

The most important validation, however, is empirical. The parameters of the model have been chosen in a theoretical way. A pilot with a real neighborhood and a willing municipality would allow them to be calibrated on observed behavior.

7.3.7 Federation of Neighborhood DAOs

CivicDAO has been conceived at the scale of a single neighborhood, but a city is made of many neighborhoods. A natural extension is a federated structure in which each neighborhood runs its own instance, deployed from a common factory contract, while sharing a city-wide identity layer and possibly a single circuit of participating businesses. In this way the community credit earned in one neighborhood can be spent in another. This structure would raise new design questions, such as the governance of proposals that involve several neighborhoods, the transfer of citizenship when a resident moves and the balance between local autonomy and city-level coordination.

Bibliography

[1] Satoshi Nakamoto et al. *Bitcoin: A peer-to-peer electronic cash system*. 2008 (cit. on pp. 4, 9).

[2] Zibin Zheng, Shaoan Xie, Hongning Dai, Xiangping Chen, and Huaimin Wang. «An overview of blockchain technology: Architecture, consensus, and future trends». In: *2017 IEEE international congress on big data (BigData congress)*. Ieee. 2017, pp. 557–564 (cit. on pp. 4, 8).

[3] Huaqun Guo and Xingjie Yu. «A survey on blockchain technology and its security». In: *Blockchain: Research and Applications* 3.2 (2022), p. 100067. DOI: 10.1016/j.bcra.2022.100067 (cit. on pp. 5, 6, 12, 13).

[4] Sunny King and Scott Nadal. «Ppcoin: Peer-to-peer crypto-currency with proof-of-stake». In: *self-published paper, August* 19.1 (2012) (cit. on p. 9).

[5] Vitalik Buterin et al. «A next-generation smart contract and decentralized application platform». In: *white paper* 3.37 (2014), pp. 2–1 (cit. on pp. 11, 14).

[6] De Filippi et al. «Decentralized autonomous organization». In: *Internet Policy Review* 10.2 (2021) (cit. on p. 14).

[7] Shuai Wang, Wenwen Ding, Juanjuan Li, Yong Yuan, Liwei Ouyang, and Fei-Yue Wang. «Decentralized autonomous organizations: Concept, model, and applications». In: *IEEE Transactions on Computational Social Systems* 6.5 (2019), pp. 870–878 (cit. on p. 14).

[8] Qinxu Ding, Daniel Liebau, Zhiguo Wang, and Weibiao Xu. «A survey on decentralized autonomous organizations (DAOs) and their governance». In: *World Scientific Annual Review of Fintech* 1 (2023), p. 2350001 (cit. on p. 15).

[9] Carlos Santana and Laura Albareda. «Blockchain and the emergence of Decentralized Autonomous Organizations (DAOs): An integrative model and research agenda». In: *Technological Forecasting and Social Change* 182 (2022), p. 121806 (cit. on p. 15).

[10] Yixuan Fan, Lei Zhang, Ruiyu Wang, and Muhammad Ali Imran. «Insight into voting in DAOs: conceptual analysis and a proposal for evaluation framework». In: *IEEE Network* 38.3 (2023), pp. 92–99 (cit. on pp. 17, 19).

[11] Bryan Alexander Ford. «Delegative democracy». In: (2002) (cit. on p. 20).

[12] Qinxu Ding, Weibiao Xu, Zhiguo Wang, and David Kuo Chuen Lee. «Voting schemes in DAO governance». In: *World Scientific Annual Review of Fintech* 1 (2023), p. 2350004 (cit. on p. 21).

[13] Fabian Vogelsteller and Vitalik Buterin. *ERC-20: Token Standard*. Ethereum Improvement Proposals, no. 20. `https://eips.ethereum.org/EIPS/eip-20`. Nov. 2015 (cit. on p. 21).

[14] William Entriken, Dieter Shirley, Jacob Evans, and Nastassia Sachs. *ERC-721: Non-Fungible Token Standard*. Ethereum Improvement Proposals, no. 721. `https://eips.ethereum.org/EIPS/eip-721`. Jan. 2018 (cit. on p. 21).

[15] Witek Radomski, Andrew Cooke, Philippe Castonguay, James Therien, Eric Binet, and Ronan Sandford. *ERC-1155: Multi Token Standard*. Ethereum Improvement Proposals, no. 1155. `https://eips.ethereum.org/EIPS/eip-1155`. June 2018 (cit. on p. 21).

[16] Tim Daubenschütz and Anders. *ERC-5192: Minimal Soulbound NFTs*. Ethereum Improvement Proposals, no. 5192. `https://eips.ethereum.org/EIPS/eip-5192`. July 2022 (cit. on p. 22).

[17] Swiss Financial Market Supervisory Authority (FINMA). *Guidelines for Enquiries Regarding the Regulatory Framework for Initial Coin Offerings (ICOs)*. FINMA, Bern. Published 16 February 2018. 2018. URL: `https://www.finma.ch/en/news/2018/02/20180216-mm-ico-wegleitung/` (cit. on p. 23).

[18] European Parliament and Council of the European Union. *Regulation (EU) 2023/1114 of the European Parliament and of the Council of 31 May 2023 on Markets in Crypto-Assets, and amending Regulations (EU) No 1093/2010 and (EU) No 1095/2010 and Directives 2013/36/EU and (EU) 2019/1937 (MiCA)*. Official Journal of the European Union, L 150, 9.6.2023, pp. 40–205. CELEX:32023R1114. 2023. URL: `https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32023R1114` (cit. on p. 23).

[19] Rabih Amhaz, Cedric Bobenrieth, and Marlene Marz. «Impact of decentralized autonomous organizations (DAO) on society 5.0». In: *8th International Conference on Networks and Communications*. 2024 (cit. on p. 24).

[20] Lu Liu, Sicong Zhou, Huawei Huang, and Zibin Zheng. «From technology to society: An overview of blockchain-based DAO». In: *IEEE Open Journal of the Computer Society* 2 (2021), pp. 204–215 (cit. on p. 25).

[21] Lawrence Lessig. *Code and Other Laws of Cyberspace*. New York: Basic Books, 1999 (cit. on p. 27).

[22] Robbie Morrison, Natasha CHL Mazey, and Stephen C Wingreen. «The DAO controversy: the case for a new species of corporate governance?» In: *Frontiers in Blockchain* 3 (2020), p. 25 (cit. on p. 27).

[23] Rainer Feichtinger, Robin Fritsch, Lioba Heimbach, Yann Vonlanthen, and Roger Wattenhofer. «Sok: Attacks on daos». In: *arXiv preprint arXiv:2406.15071* (2024) (cit. on p. 27).

[24] Tanusree Sharma, Yujin Potter, Kornrapat Pongmala, Henry Wang, Andrew Miller, Dawn Song, and Yang Wang. «Future of algorithmic organization: Large-scale analysis of decentralized autonomous organizations (daos)». In: *arXiv preprint arXiv:2410.13095* (2024) (cit. on p. 28).

[25] Xiaotong Sun, Charalampos Stasinakis, and Georgios Sermpinis. «Decentralization illusion in Decentralized Finance: Evidence from tokenized voting in MakerDAO polls». In: *Journal of Financial Stability* 73 (2024), p. 101286. ISSN: 1572-3089. DOI: https://doi.org/10.1016/j.jfs.2024.101286. URL: https://www.sciencedirect.com/science/article/pii/S1572308924000718 (cit. on p. 28).

[26] Markus Jungnickel, Ferda Özdemir Sönmez, Cathy Mulligan, and William J Knottenbelt. «DAO governance: Voting power, participation, and controversy—a review and an empirical analysis». In: *Distributed Ledger Technologies: Research and Practice* 5.4 (2026), pp. 1–25 (cit. on p. 28).

[27] Johnnatan Messias and Ayae Ide. «Fairness in token delegation: Mitigating voting power concentration in daos». In: *arXiv preprint arXiv:2510.05830* (2025) (cit. on p. 29).

[28] State of Vermont. *The Vermont Statutes Online, Title 11, Chapter 25, § 4173: Blockchain-Based Limited Liability Companies*. Vermont Limited Liability Company Act. 2018. URL: https://legislature.vermont.gov/statutes/section/11/025/04173 (visited on 09/03/2026) (cit. on p. 29).

[29] Internet Native. *Wyoming DAO Jurisdictions*. 2023. URL: https://internetnative.org/wyoming-dao/ (visited on 09/03/2026) (cit. on p. 29).

[30] Internet Native. *Marshall Islands DAO Jurisdictions*. 2023. URL: https://internetnative.org/marshall-islands-dao/ (visited on 09/03/2026) (cit. on p. 29).

[31] Robert B. Lamb. *Utah Passes Innovative DAO Legislation*. The National Law Review. 2023. URL: https://natlawreview.com/article/utah-passes-innovative-dao-legislation (visited on 09/03/2026) (cit. on p. 29).

[32] Rosa Borge, Joan Balcells, and Albert Padró-Solanet. «Democratic Disruption or Continuity? Analysis of the Decidim Platform in Catalan Municipalities». In: *American Behavioral Scientist* 67.7 (2023), pp. 926–939. DOI: 10.1177/00027642221092798 (cit. on p. 30).

[33] Nour Diallo et al. «eGov-DAO: A better government using blockchain based decentralized autonomous organization». In: *2018 International Conference on eDemocracy & eGovernment (ICEDEG)*. IEEE. 2018, pp. 166–171 (cit. on pp. 30, 31).

[34] Sabrina Wollenschläger and Michael Lustenberger. «Self-Organization and Digital Participation: Evaluating DAOs and Alternative Governance Models». In: *Information Polity* (2026). DOI: 10.1177/15701255261421040 (cit. on p. 31).

[35] Michael Lustenberger, Florian Spychiger, Lukas Küng, and Pedro Cuadra. «Mastering DAOs: A practical guidebook for building and managing decentralized autonomous organizations». In: *Available at SSRN 5001424* (2024) (cit. on pp. 37, 39, 40, 45, 50, 79).

[36] Silvio Meneguzzo and Rachele Pierri. «A Game-Theoretic Incentive Model for DAO Governance». In: *2025 IEEE 49th Annual Computers, Software, and Applications Conference (COMPSAC)*. Toronto, Canada: IEEE, 2025. DOI: 10.1109/COMPSAC65507.2025.00232 (cit. on p. 49).

[37] Vitalik Buterin. *Minimal Anti-Collusion Infrastructure*. Ethereum Research forum (ethresear.ch). 2019. URL: https://ethresear.ch/t/minimal-anti-collusion-infrastructure/5413 (cit. on p. 85).

[38] Privacy & Scaling Explorations. *MACI: Minimal Anti-Collusion Infrastructure – Documentation*. 2024. URL: https://maci.pse.dev/docs/introduction (cit. on p. 85).