Scuola universitaria professionale della Svizzera italiana
Dipartimento tecnologie innovative

SUPSI

DAO on ICP: A new paradigm for corporate governance

Student

Lorenzo Ronzani

Supervisor

Giuliano Gremlich

Co-Supervisor

Roberto Guidi

Customer

Ticino Blockchain Technologies Association

Degree course

MSE Computer Science

Module

MP30_0001 Thesis project

Year

2025

Date

August 23, 2025

Contents

List of Figures

List of Tables

Abstract

This project explores the design and implementation, entirely based on the Internet Computer Protocol (ICP) ecosystem, of a fully decentralized platform for creating and managing Decentralized Autonomous Organizations (DAOs). The work addresses the limitations of existing DAO platforms by combining ICP's unique features, such as on-chain hosting of frontend, backend, and governance logic, with legal compliance under Swiss laws. The platform adopts a modular architecture, enabling the integration of new legal forms and jurisdictions without impacting core components. The development process combined a legal and architectural analysis of DAO governance models with a technical implementation based on ICP canisters. The system was tested in a simulated environment using dfx, the ICP command-line execution environment, and through automated cargo test suites, with additional informal validation from early adopters of the Proof of Concept (PoC). Key results include the realization of an ICP-native platform with no centralized components, support for fully automated action execution with verifiable results, and a legally compliant implementation of Swiss associations. The architecture's extensibility allows adaptation to other jurisdictions and legal forms, expanding potential adoption. The project demonstrates the feasibility of legally aware, fully decentralized governance systems, paving the way for future developments such as the integration of Verifiable Credentials (VCs), advanced privacy mechanisms, and real-world deployment in collaboration with institutional partners.

Chapter 1

Introduction

This thesis explores the design and development of a fully decentralized platform for creating and managing Decentralized Autonomous Organizations (DAOs) [1]. The objective is to offer an innovative alternative to existing DAO platforms by leveraging the unique capabilities of Internet Computer Protocol (ICP) [2] technology to build a fully decentralized, flexible, and legally aware system.

A DAO, sometimes referred to as a Decentralized Autonomous Corporation (DAC), is a community-driven entity governed by decentralized computer programs (smart contracts), where decisions and financial operations are executed through decentralized ledger technologies such as blockchains. For an organization to be considered a DAO, its core processes must be transparent, autonomous and executed decentralized. DAOs generally function as member-owned collectives without centralized leadership. However, their legal status remains uncertain, as traditional regulatory frameworks struggle to accommodate such novel organizational models.

Despite the growing interest in DAOs, most existing solutions have significant limitations. Some services are entirely paper-based, relying on traditional legal intermediaries to establish compliant DAOs without offering any technological foundation. Others provide digital interfaces, but retain centralized components in their architecture, such as web frontends or backend services, which undermines the decentralized ethos. Furthermore, many tools require substantial manual intervention due to their lack of jurisdictional focus, making them hard to automate or adapt to specific national legal frameworks. To date, there is no publicly available system that effectively addresses all of these challenges in a cohesive, fully decentralized manner.

This project aims to fill this gap. It introduces a DAO creation and management tool built

entirely on ICP technology, where all components, including frontend, backend, and logic, are deployed as on-chain smart contracts. The architecture is designed to be modular and extensible, allowing developers or users to tailor the system to a specific country and legal form. For the purpose of this thesis, the implementation focuses on the creation and operation of an association under Swiss law, serving as a concrete and legally grounded use case.

The ICP provides a unique foundation for such a project. As a general purpose blockchain designed to host complete applications directly on-chain, it removes the need for traditional IT infrastructure. It enables the development of Web 3.0 [3] services that are secure, composable, and truly decentralized. Launched in May 2021 after years of intensive research and development, the ICP introduces a fundamentally new blockchain architecture based on advanced cryptographic principles, enabling smart contracts, known as canisters, to autonomously manage computation, storage, and inter-canister communication.

The motivation behind this thesis is to explore how organizational workflows traditionally shaped by bureaucratic, centralized, and often inefficient processes can be reimagined through a purely decentralized and programmable model. By implementing a DAO platform natively on ICP, this work explores the feasibility, limitations, and legal considerations of replacing conventional structures with blockchain-native alternatives, while offering a focused solution tailored to a real-world legal context.

1.1 Project goals

The primary goal of this project is to design and implement a fully decentralized platform, for the creation and management of DAOs, that is legally compliant and practically useful within real-world contexts. The platform is built entirely on the ICP, leveraging its native features to eliminate centralized components and maximize decentralization, automation, and trust. By aligning with legal constraints and targeting specific jurisdictional use cases, the platform aims to serve as a valuable tool for communities, developers, and institutions seeking to adopt DAO-based governance structures.

This project blends software engineering and research, and its objectives can be summarized along the following axes:

- **Legal and theoretical analysis**
    - Conduct a comprehensive study of DAO governance models and evaluate their applicability to Swiss corporate entities, with a particular focus on three legal

forms [4]: Associations, Limited Liability Companies (LLC) or Società Anonima a Garanzia Limitata (SAGL throughout the document), and Public Limited Company (PLC) or Società Anonima (SA throughout the document).

- Define a legal integration strategy that ensures compliance with Swiss law while preserving the core advantages of decentralized governance.

- **Evaluation of existing solutions**
  - Analyze current DAO platforms to understand their benefits, limitations, and typical architectural patterns.
  - Identify areas where current solutions fall short, particularly in terms of decentralization, legal compliance, and automation.

- **Exploration of improvements enabled by ICP**
  - Investigate how ICP can enhance the decentralization, scalability, and usability of DAO platforms.
  - Explore integration opportunities with modern identity verification mechanisms, including the use of Verifiable Credentials (VCs) [5] for Know Your Customer (KYC) [6] procedures.
  - Identify and validate the extent to which automation can replace traditional organizational workflows.
  - Assess the potential and current limitations of integrating legal logic directly into platform operations.

- **Platform development**
  - Define a modular and extensible architecture that allows customization based on jurisdiction-specific requirements.
  - Analyze end-user needs to inform user-centric design choices.
  - Develop and deploy the platform entirely on the ICP blockchain, ensuring all components, including frontend, backend, and governance logic, are executed on-chain.
  - Test the platform in a real-world environment to assess usability, stability, and performance.

- **Real-world validation**
  - Implement a case study in collaboration with the Ticino Blockchain Technologies Association (TBTA) [7], demonstrating the practical viability of the solution.

- Engage with institutional stakeholders, including the Municipality of Lugano, to evaluate the feasibility and interest in broader adoption of DAO-based governance frameworks.

1.2 Document overview

This thesis documentation is organized into six chapters, each addressing a specific dimension of the project, from foundational research to real-world validation and future perspectives:

- **Chapter 2**: Establishes the theoretical and legal foundation for DAO development, with a particular focus on Swiss jurisdiction. It includes an evaluation of existing DAO solutions, highlighting their limitations, and introduces the core technologies relevant to this work, such as the ICP and VCs. The chapter also discusses the importance of KYC processes and explores how identity verification can be integrated into decentralized systems.
- **Chapter 3**: Describes the architectural design of the proposed platform, emphasizing its flexibility, modularity, and jurisdictional adaptability. The chapter analyzes key workflows, identifies potential points of failure, and discusses strategies for ensuring scalability, robustness, and interoperability with external systems or other blockchains.
- **Chapter 4**: Details the technology stack used to develop the platform, including the tools and frameworks chosen to build and deploy the system on ICP. It describes how the various components were implemented and how the system was iteratively tested throughout the development process.
- **Chapter 5**: Presents the outcomes of the project, including a discussion of the platform's strengths, weaknesses, and the current limitations, both technological and legal, of fully automating DAO creation and management. The chapter also highlights the public release of the platform's source code, the deployment of a working version in a real environment, the maintainability of it and the collection of early feedback from stakeholders, domain experts, and community users. A comparative evaluation with existing DAO platforms is also provided to position the developed solution within the current landscape.
- **Chapter 6**: Summarizes the entire project, reflecting on the challenges encountered and lessons learned. It outlines the major benefits and trade-offs of the proposed approach and suggests concrete directions for future improvements, including the addition of new legal-form templates and deeper institutional integration.

Chapter 2

Analysis

This chapter provides the foundational knowledge necessary to understand the legal, technical, and architectural decisions discussed in the subsequent chapters. It begins with an analysis of the legal frameworks governing three common Swiss organizational structures, Associations, SAGL, and SA, focusing on the minimum legal requirements for establishing and managing these entities in compliance with Swiss law. The chapter then evaluates existing DAO platforms, specifically Aragon [8], Tally [9] and Juicebox [10], identifying their strengths and limitations in relation to decentralization and legal enforceability. To contextualize the chosen implementation stack, a brief overview of the ICP and its identity system, Internet Identity [11], is also presented. Moreover, the chapter introduces the core concepts of VCs and KYC processes, which, although not implemented in the current prototype, are essential for understanding the broader regulatory and operational landscape in which decentralized governance systems operate. Finally, a short overview of the methodology applied in the project is provided.

2.1 Swiss company forms

To establish a legally compliant DAO within the Swiss legal framework, it is essential to understand the foundational requirements of recognized organizational structures. This section presents a high-level overview of Switzerland's three main legal forms under federal law: the Association, the Limited Liability Company (SAGL / GmbH / SARL), and the Public Limited Company (SA / AG). For the most commonly used form in business contexts relevant references to the Swiss Code of Obligations (CO) [12] and to the Swiss Civil Code (CC) [13] are included to highlight key compliance requirements.

For each legal form, the analysis covers the following dimensions:

- Legal personality and permissible purpose
- Minimum number of founders and formation rules
- Liability regimes and capital requirements
- Governance structures and representation authority
- Registration and public disclosure obligations
- Accounting and audit standards
- Critical legal constraints relevant to DAO-like automation

This overview provides the legal baseline necessary to assess the feasibility of replicating or automating these models within a decentralized, blockchain-based infrastructure.

2.1.1 Association

The Swiss Association represents one of the most accessible and flexible legal forms for organizing non-commercial initiatives. Governed by articles 60 to 79 of the Swiss Civil Code, it acquires legal personality upon the adoption of written statutes and the convening of a constituent general meeting (art. 60(1) [14] CC). Its purpose must be non-commercial in nature, typically cultural, scientific, or charitable. Should the association engage in commercial activities, it is required to register in the Commercial Register [15] and becomes subject to the CO, including accounting, audit, and transparency obligations.

Founding an association requires no minimum capital, and both natural and legal persons may serve as founders. The legal formation process involves the creation of statutes and the formal establishment of the members' general assembly, which acts as the association's supreme decision-making body. The statute must define the association's internal governance structure, including the composition and responsibilities of the executive committee (or management board), which typically handles day-to-day operations. If desired, the statutes may delegate authority to a council or board with broader decision-making power.

From a liability standpoint, members are generally not personally responsible for the association's debts, unless otherwise stated in the statutes (art. 75a [16] CC). However, committee members may be held liable for damages caused by intentional or negligent acts, under the general tort principles of the CO (arts. 41 – 61 [17] CO).

Registration in the Commercial Register is not mandatory unless the association operates commercially or opts for registration voluntarily to enhance its credibility. In such cases, it becomes subject to the standard public disclosure and transparency requirements laid out in the CO. Regarding financial obligations, associations that are not registered and generate annual revenues below CHF 500'000 may adopt simplified, cash-based accounting (art. 957(2) [18] CO). However, registered associations or those engaged in commercial activities must comply with full accounting standards and, depending on their size, may be subject to statutory audit requirements (arts. 957 – 963b [18] CO).

There are several legal constraints particularly relevant when considering DAO-like automation. First, the statutes must conform to specific legal formats and may not override mandatory provisions established in the CC. Profit distribution is prohibited; any surplus must be reinvested in the association's stated purpose. Commercial activity without registration is not allowed, and managing bodies are subject to fiduciary duties, any negligent breach of duty may result in personal liability. These elements highlight key areas where DAO automation must be carefully balanced against legal requirements for accountability and governance in the Swiss context.

2.1.2 Limited Liability Company

The Swiss Limited Liability Company (SAGL), represents a hybrid legal structure that combines elements of a partnership and a corporation. Governed by articles 772 to 827 of the Swiss Code of Obligations, the SAGL is a fully recognized legal entity that acquires its legal personality upon registration in the Commercial Register (art. 779(1) [19] CO). It is designed primarily for small to medium-sized businesses and offers a flexible but formal governance model, making it a relevant structure for DAO-related use cases that require commercial capabilities and limited liability.

The company may be founded by one or more individuals or legal entities, with no requirement for Swiss nationality or residency among the founders. However, Swiss law mandates that at least one individual authorized to represent the company (manager with formal signing authority) must reside in Switzerland or hold an EU/EFTA residence permit (art. 806 [20] CO). The minimum capital requirement is CHF 20'000, which must be fully paid in at the time of incorporation (art. 773 [21] and 777c [22] CO). Membership interests, referred to as quotas, can be denominated in Swiss francs or the equivalent in foreign currency. Transfers of quotas require written form, board approval, and must be recorded in the company's internal quota register (arts. 784 – 788 [23] CO). Liability is limited to the company's capital unless additional obligations are defined in the statutes (art. 800 [24] CO).

The governance structure is composed of two primary bodies: the General Meeting of shareholders and the Management Board (or one or more Managing Directors). The Articles of Association [25] must clearly define voting rights, profit distribution rules (art. 825 [26] CO), and quorum requirements for decision-making. The management board holds full operational and legal authority to represent and bind the company.

Registration in the Swiss Commercial Register is mandatory for the formation of the SAGL as well as for any statutory modifications, changes in capital, or updates to quota holders. The names of all shareholders and their respective capital contributions must be publicly disclosed, ensuring a high degree of transparency. From an accounting standpoint, SAGLs are required to maintain full double-entry bookkeeping and produce annual financial statements in accordance with CO articles 957 – 963b. The company may be subject to a statutory audit depending on its size. An ordinary audit is required if the company exceeds two of the following thresholds for two consecutive years: CHF 20 million in total assets, CHF 40 million in annual revenue, or 250 full-time employees. Smaller companies may request an exemption or undergo a limited audit only (art. 727 [27] CO).

In the context of DAO-like automation, several compliance constraints must be considered. The founding documents, including the Articles of Association and capital deposit certificate, must be notarized before registration. Capital may not be reduced below the CHF 20'000 legal minimum, and strict rules apply to quota transfers, requiring administrative involvement and board approval. Every change in statutory, ownership or capital structure must be registered in the Commercial Register, and a local representative with legal authority is mandatory. These regulatory obligations pose clear limitations to full automation and must be addressed through governance mechanisms that integrate human oversight and accountability into DAO workflows modeled on this legal form.

2.1.3 Public Limited Company

The Swiss Public Limited Company (SA), is a capital-based company designed primarily for commercial enterprises that require formal governance, investment scalability, and limited shareholder liability. Governed by articles 620 to 763 of the Swiss Code of Obligations, the SA becomes a legal entity upon registration in the Commercial Register (art. 640 [28] CO). Shareholder liability is limited to the value of their capital contribution, and the company can issue either registered or bearer shares (arts. 634 – 635 [29] CO).

An SA may be founded by a single natural or legal person. There is no Swiss residency requirement for shareholders, but at least one member of the board of directors must reside in Switzerland or hold an EU/EFTA residence permit (art. 715(2) [30] and 718 [31] CO).

The minimum share capital is CHF 100'000 (art. 624 [32] CO), of which at least CHF 50'000, or 20% of the total share capital, whichever is higher, must be paid in at the time of incorporation. This amount must remain blocked in a bank account until the company is officially registered (art. 652(2) [33] CO). Capital contributions in kind must be independently valued and verified by an auditor (art. 652a [34] and 653 [35] CO). Share capital is divided into shares with a minimum nominal value of CHF 0.01 each (arts. 625 [36] and 682 [37] CO), and statutes may define restrictions or preemptive rights on share transfers (art. 685 [38] CO).

The governance structure of an SA is more formalized than other company types and must include three statutory bodies: the General Meeting of Shareholders, the Board of Directors, and the Statutory Auditor (arts. 698 – 706 [39] and 716 – 729 [40] CO). The board may delegate executive functions to specific members or external officers, but remains ultimately responsible for oversight. The statutes must clearly define the company's internal rules, including quorum requirements, dividend policies, and shareholder rights.

The SA must be registered in the Commercial Register at the time of incorporation and for any structural or capital-related changes. It is subject to extensive transparency obligations. Annual financial statements and auditor reports must be published in the Swiss Official Gazette of Commerce or submitted electronically to any person who requests the same within one year of their approval (art. 958e [41] CO). Information such as shareholder meeting minutes, auditor appointments, and capital structure is publicly accessible.

All SAs are required to maintain full double-entry bookkeeping and prepare annual financial reports in accordance with CO articles 957 – 963b. An ordinary audit is mandatory if the company exceeds two of the following thresholds in two consecutive fiscal years: total assets of CHF 20 million, annual revenue of CHF 40 million, or more than 250 full-time employees (art. 727 [27] CO). In addition, listed or public-interest companies must prepare non-financial Environmental, Social and Governance (ESG) [42] reports covering environmental, human rights, and anti-corruption topics starting from the 2023 fiscal year, as required by art. 964a [43] CO.

From the perspective of DAO automation, several strict compliance requirements impose clear constraints. These include the notarization and capital blocking required at incorporation, the need for Swiss-resident board members, and the structured functioning of corporate bodies such as the general assembly and the board of directors. Any issuance of new shares, payment of dividends, or capital restructuring must follow rigorous procedures that include reserve validation and creditor safeguards. These legal formalities require human oversight and legal accountability, posing limitations for fully automated governance and decision-making processes. However, the SA remains the most robust and legally

recognized structure for large-scale, investment-driven projects, making it a relevant, albeit complex, candidate for DAO-based models operating in regulated environments.

2.1.4 Summary

Table 2.1 provides a comparative overview of the three main Swiss legal forms analyzed in this section: association, Limited Liability Company (SAGL), and Public Limited Company (SA). It summarizes the core legal, operational, and compliance characteristics of each structure, highlighting how these aspects affect their potential compatibility with automated or decentralized governance systems.

Aspect	**Association**	**SAGL**	**SA**
Legal basis	Art. 60 – 79 CC	Art. 772 – 827 CO	Art. 620 – 763 CO
Founders - capital	No capital required; any number of members	Min. CHF 20'000 fully paid in	Min. CHF 100'000 ($\geq$ CHF 50'000 or 20% of the total share capital paid in)
Liability	No personal liability by default; statute may override	Limited to capital	Limited to capital
Management bodies	Members' meeting; executive committee	General meeting; managing director(s)/board	Shareholders' meeting; board; statutory auditor
Registration - transparency	Optional registration unless commercial activity	Mandatory registration; shareholder info public	Mandatory registration; high disclosure obligations
Accounting & audit requirements	Simplified accounting if small and unregistered; full accounting and audit if registered (art. 957 CO)	Full accounting required; statutory audit if thresholds met (art. 727 CO)	Full accounting; mandatory audit; non-financial reporting required for large/public companies (art. 964a CO)

Table 2.1: Overview of company forms in Switzerland

This comparison reveals a spectrum of complexity and formality among the legal forms. The Association offers the greatest flexibility, with minimal founding requirements, optional registration, and simplified accounting, making it suitable for community-based or low-risk DAO initiatives. However, its inability to pursue profit, combined with limits on automation due to statutory constraints and fiduciary duties, may restrict its use in commercial DAO scenarios.

The SAGL presents a more structured governance model with capital requirements and formal transparency obligations. It allows for profit-making and provides limited liability, making it a better fit for DAO projects aiming for operational autonomy within legal boundaries. However, the mandatory registration and manual controls over share (quota) transfers and capital management may limit automation potential unless carefully integrated into the platform design.

The SA, while the most robust and investment-oriented structure, is also the most demanding in terms of capital, governance, and transparency. Its strict procedural requirements and the presence of multiple statutory bodies introduce significant friction for DAO automation. Still, its legal recognition, scalability, and audit-ready structure make it the most viable model for high-value or public-facing DAO projects that aim to integrate deeply into the existing regulatory and financial ecosystem.

Overall, each form offers trade-offs between flexibility, legal complexity, and automation feasibility. These considerations will guide the platform's legal modeling and technical design choices in the following chapters.

2.2 Existing DAO management tools

To contextualize the design choices of the proposed ICP-based platform, this section analyzes three widely adopted DAO tooling solutions: Aragon, Tally, and Juicebox. Each of these platforms addresses different aspects of decentralized governance and organizational management, offering valuable insights into the current state of DAO infrastructure.

Aragon provides a comprehensive framework for creating and managing DAOs on Ethereum [44], with features such as customizable governance models, treasury control, and modular smart contracts. Tally focuses on governance coordination, offering user-friendly interfaces for proposal creation, token-based voting, and contributor transparency, while integrating off-chain data to support on-chain decisions. Juicebox, in contrast, specializes in treasury and funding mechanisms, enabling DAOs to manage capital, revenue streams, and funding rounds through programmable financial rules.

By examining the basic idea, strengths, and limitations of these platforms, this analysis highlights both best practices and critical shortcomings. Particular attention is paid to decentralization, automation potential, and legal alignment dimensions essential to the design of a DAO system that aspires to operate within a compliant and fully decentralized environment. The findings presented here directly inform the design considerations introduced in Chapter 3.

2.2.1 Aragon

Aragon is one of the earliest and most established DAO frameworks, introduced in 2016 – 2017 to enable users to deploy fully operational DAOs on Ethereum and, more recently, on Polygon [45]. Its architecture combines Aragon OSx [46], a modular smart contract kernel with plugin support, and the Aragon App [47], a user-friendly, no-code interface for DAO deployment and governance. Through this framework, organizations can create token-based governance systems, manage treasuries, execute on-chain voting, and even handle dispute resolution, without the need to write smart contract code. To date, Aragon has supported the creation of over 6'000 DAOs, collectively governing assets in excess of $30 – 35 billion.

2.2.1.1 Strengths

A notable strength of Aragon lies in its full on-chain execution model. Proposals, once approved through token voting, are automatically executed via smart contracts, eliminating the need for a trusted intermediary or off-chain coordination. This level of automation supports advanced workflows such as decentralized payroll, staking, and contributor onboarding. Additionally, the system is modular, allowing DAOs to extend their governance capabilities through plugins and sub-DAOs, adapting to increasingly complex organizational needs.

Legal alignment is another key feature of the Aragon ecosystem. The organization behind the platform is based in Zug, Switzerland, and offers a predefined legal wrapper allowing DAOs to be formally structured as Swiss associations. This approach enables organizations to operate within a recognized legal framework, offering limited liability to members and compatibility with real-world business practices such as signing contracts or opening bank accounts.

2.2.1.2 Limitations

Despite these benefits, Aragon also presents some significant limitations. Although the core logic is decentralized, the Aragon App, the main interface for users, is centrally hosted and maintained by Aragon X AG. This creates a single point of dependence for access, upgrades, and documentation. Moreover, while the platform supports customizable governance templates, it is not designed to accommodate new legal forms or jurisdiction-specific requirements beyond the Swiss association model. This lack of extensibility makes it difficult to adapt the tool for broader legal automation across multiple countries or entity types.

The Ethereum-centric nature of Aragon introduces further constraints. Its infrastructure cannot be hosted in a fully decentralized manner, and integration with systems outside the Ethereum ecosystem, such as traditional IT systems, legal databases, or multi-chain governance layers, requires the use of centralized bridges or third-party oracles. These dependencies reintroduce trust assumptions that may conflict with the platform's decentralization goals.

In addition, Aragon has faced internal governance challenges. In late 2023, disputes emerged over the control of the platform's $160 million treasury, sparking debate about the role and accountability of the Aragon association. This case illustrated the difficulty of balancing community-driven governance with legal responsibility and operational security, especially in high-value ecosystems.

2.2.1.3 Adoption

Real-world adoption of Aragon has been substantial. Thousands of DAOs have been created using its infrastructure, including charitable organizations, research collectives, and funding pools. As of 2023, more than 3'800 DAOs remained active. The Aragon Network DAO itself governs its operations using Aragon Voice [48] for proposals and Aragon Court [49] for subjective dispute resolution. While the platform's legal wrapper offers a rare bridge between on-chain activity and real-world institutions, its architectural and jurisdictional rigidity presents challenges for projects seeking customizable legal automation across multiple contexts.

2.2.2 Tally

Tally is a governance-focused platform, based on Ethereum, that provides DAO interfaces and infrastructure centered entirely around on-chain proposal creation, voting, and execu-

tion. Built on the widely adopted Governor smart contract framework from OpenZeppelin [50], Tally positions itself not as a full DAO creation toolkit, but as a high-performance governance interface for DAOs that already exist. The platform is especially popular among high-traffic protocol DAOs, including Uniswap [51], Arbitrum [52], Eigenlayer [53], Wormhole [54], and others. As of early 2025, Tally secured an $8 million Series A funding round, further solidifying its role in Ethereum's DAO governance landscape.

2.2.2.1 Strengths

One of Tally's defining characteristics is its strict commitment to on-chain governance. Unlike platforms such as Snapshot [55], which rely on off-chain signature-based voting, Tally ensures that all governance tokens vote directly on-chain, and that proposals, once approved, are executed automatically through standard Governor logic. This provides verifiable, tamper-resistant governance flows that closely align with the DAO ideal of trustless automation.

In addition to this technical foundation, Tally is notable for its strong focus on user experience and interoperability. It features a modern, intuitive UI designed to lower the barrier to governance participation, offering real-time proposal dashboards, voting analytics, and delegation tools. The platform also supports domain name resolution via Ethereum Name System (ENS) [57], cross-DAO delegation, and a proprietary system called MultiGov™, which enables DAOs across multiple chains, including Ethereum, Solana [57], and various Layer-2 networks, to be managed through a unified interface. This multi-chain support has made Tally particularly appealing to protocols operating across complex ecosystems.

To further boost voter participation, Tally introduces a staking mechanism for governance tokens, allowing users to receive liquid staking derivatives (stLSTs) while continuing to delegate voting power. This approach helps address voter apathy by aligning financial incentives with active participation. Detailed analytics, such as token distribution, proposal frequency, and historical participation trends, offer valuable insights for DAO administrators and make Tally suitable for real-world institutions that require transparent, auditable governance processes.

2.2.2.2 Limitations

However, Tally's functional scope is intentionally narrow. Unlike platforms such as Aragon, Tally does not provide tooling for DAO creation, treasury management, or legal structuring. It assumes that the DAO already exists and is looking for an interface to manage its gov-

ernance lifecycle. Consequently, users who need to wrap their DAO in a legal entity, such as a Swiss association, or set up modular DAO logic with plug-and-play modules must seek those capabilities elsewhere. Furthermore, while Tally's frontend is widely praised for usability, it is centrally hosted and maintained by the platform's core team. There is currently no decentralized or open-source alternative UI, which introduces an element of vendor lock-in and reliance on centralized infrastructure.

2.2.2.3 Adoption

Despite these constraints, Tally has become the de facto governance interface for several of Ethereum's most active DAOs. For example, approximately 70% of Uniswap DAO's total token voting activity flows through Tally. Its infrastructure supports tens of billions of dollars in protocol-controlled assets across multiple chains, including Arbitrum, ZKsync [58], and Solana. While it does not aim to cover the entire DAO lifecycle, Tally excels as a focused, battle-tested tool for post-launch governance, particularly in environments that require high transparency, legal auditability, and support for large-scale stakeholder coordination.

2.2.3 Juicebox

Juicebox is a DAO infrastructure platform focused primarily on programmable fundraising and treasury management. Originally designed to support public goods, community protocols, and campaign-style DAOs, Juicebox enables creators to collect ETH, mint governance tokens, and configure how funds are released and redeemed, all through on-chain smart contract logic. The platform operates as a flexible treasury engine, offering fine-grained control over contribution rules, token distribution, payout schedules, and contributor redemptions. As of 2025, Juicebox has powered over 750 fundraising campaigns, including high-profile projects such as ConstitutionDAO [59], AssangeDAO [60], and SharkDAO [61], collectively raising hundreds of millions of dollars in ETH.

2.2.3.1 Strengths

A defining strength of Juicebox lies in its automation of treasury flows. Funding cycles, typically weekly or monthly, determine how much ETH is collected, how many tokens are minted per contribution, and how funds can be spent or refunded. These parameters are encoded on-chain, offering a transparent and trust-minimized framework where contributors can verify in advance how funds will be used. This on-chain rule enforcement distinguishes Juicebox from traditional crowdfunding platforms and aligns closely with DAO principles of

programmable finance.

Juicebox also supports extensive flexibility in funding mechanics, including bonding curves, contributor tiers, time-locked releases, and refundability. This makes it well-suited for dynamic, community-driven use cases such as grants programs, NFT [62] acquisition pools, social collectives, and public goods funding. Contributors receive ERC-20 [63] tokens representing governance rights, which can be integrated with off-chain tools like Snapshot for voting. Although Juicebox does not provide its own governance interface, it is commonly used alongside Discord [64], GitBook [65], and other coordination layers to support community decision-making.

From an operational standpoint, Juicebox emphasizes transparency. All funding rules, proposals, contributor histories, and fund distributions are recorded on-chain and publicly auditable. This open-data model strengthens accountability and enables DAO members and external observers to monitor treasury operations in real time. The open-source nature of the core protocol reinforces the project's commitment to verifiability and modular use.

2.2.3.2 Limitations

However, Juicebox also comes with several important limitations. It is not a full governance stack: there is no built-in voting system, legal wrapper, or on-chain dispute resolution. Governance must be implemented separately, often using off-chain tools that introduce centralized dependencies. In addition, Juicebox's infrastructure does not include native support for DAO formation, membership identity, or asset management beyond the fundraising logic. DAOs that wish to expand into protocol governance, cross-chain coordination, or legal structuring must integrate third-party components manually.

Furthermore, Juicebox does not currently offer any jurisdictional legal alignment, such as the ability to wrap a DAO into a Swiss association or similar legal entity. As a result, DAO operators seeking real-world legal compliance must incorporate externally or rely on legal wrappers provided by other platforms. The platform is also Ethereum-based, and while its contracts are fully verifiable, the Juicebox frontend is centrally hosted and maintained. This introduces some degree of vendor dependence for user interaction and interface reliability.

2.2.3.3 Adoption

Despite these constraints, Juicebox has proven to be an extremely effective tool for community fundraising and campaign coordination. A leading example is ConstitutionDAO, which raised over $47 million in ETH through Juicebox to bid on a rare copy of the U.S. Constitu-

tion. Similarly, SharkDAO raised more than 1'000 ETH to acquire a portfolio of Nouns NFTs, coordinating contributions from over 900 individuals using Juicebox's treasury automation tools. These cases illustrate the platform's capacity to facilitate transparent, large-scale, and decentralized fundraising without relying on centralized custodians or intermediaries.

2.2.4 Summary

Table 2.2 offers a comparative summary of the three analyzed DAO tooling platforms, Aragon, Tally, and Juicebox, highlighting their respective strengths, limitations, and intended use cases. This comparison provides the necessary context to better understand the architectural and design decisions discussed in Chapter 3.

Feature / criterion	**Aragon**	**Tally**	**Juicebox**
Governance	On-chain Governor with modular plugins	Native on-chain Governor (OpenZeppelin-based)	Token-based voting (no built-in proposal system)
Treasury management	Built-in finance module with permission control	External integration only (wallet-based)	Fully programmable, cycle-based on-chain treasury
Automation potential	High: vote-execute pipelines + plugin system	Medium–high: execution follows Governor logic	High: token minting, fund releases, redemptions all on-chain
Legal wrapping / compliance	Swiss association wrapper supported	No legal support offered	No legal support offered
Transparency & verifiability	Partial: frontend centralized, backend on-chain	Partial: frontend centralized, votes on-chain	Strong for treasury rules, frontend centralized
Interoperability	Ethereum-only, bridges needed for cross-chain	Multi-chain (via MultiGov™, e.g. Solana, Arbitrum)	Ethereum-native; integrates with Snapshot/GitBook
Flexibility / extensibility	Medium: plugin-based, but not easily jurisdiction-aware	Low: fixed governance UI and logic	Medium: customizable treasury logic, but no full DAO support

Table 2.2: Comparison of DAO frameworks

Feature / criterion	Aragon	Tally	Juicebox
Organizational scope	Full-stack: governance, treasury, legal wrapper	Governance-focused: not for DAO creation	Fundraising-focused: lacks governance or membership modules
Scalability & adoption	Thousands of DAOs; mature ecosystem	Used by top protocols (e.g. Uniswap, Wormhole, etc.)	Widely adopted for large-scale campaigns (e.g. ConstitutionDAO)
User experience & wallet model	Token-based UX; requires Ethereum wallet	Token-based UX with delegation tools	Wallet required; mints ERC-20 tokens to contributors
Best-fit use case	Launching structured DAOs with legal clarity	Delegated governance for mature, protocol-level DAOs	Crowdfunding, grants, NFT pools, community treasuries

Table 2.2: (continued)

This comparative analysis highlights how current DAO tools often specialize in specific operational domains. Aragon providing a full-stack environment with legal structuring, Tally focusing on governance after launch, and Juicebox targeting campaign-style treasury flows. However, each solution exhibits trade-offs across interoperability, decentralized frontend availability, automation coverage, and compliance alignment, particularly when considered from a Swiss legal perspective.

These observations form the analytical basis for Chapter 3, where a new platform architecture is introduced aiming to address several of the limitations outlined above through a more holistic, interoperable, and legally compliant approach to DAO creation and management.

2.3 Internet Computer Protocol (ICP)

To fully understand the architectural and design choices presented in Chapter 3, it is essential to introduce the ICP, the foundational technology on which this project is built. Unlike tra-

ditional blockchain platforms limited to decentralized logic execution, ICP offers a full-stack decentralized environment where both backend services (smart contracts) and frontend interfaces can be hosted directly on-chain, ensuring unmatched transparency and trust.

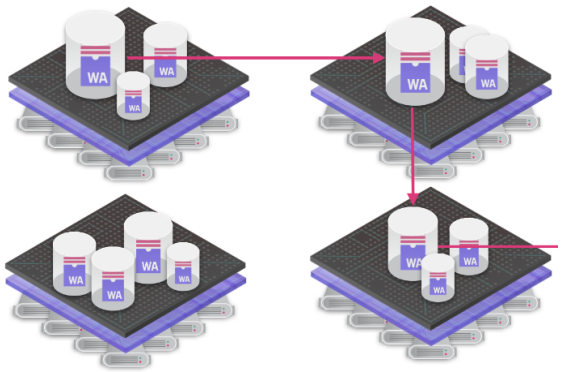
Designed for scalability, ICP employs a novel architecture that allows applications to run in parallel across independent computational units called canisters (more info in Section 2.3.1.1), enabling the system to scale horizontally while maintaining consistent performance. The smart contract model on ICP is natively supported through programming languages such as Rust [66] and TypeScript [67], allowing developers to build complex systems with modern tooling and strong memory safety guarantees.

The protocol also introduces a reverse-gas model (more details in Section 2.3.1.2), where the cost of computation is covered by the application itself rather than its users, a significant step toward improving user experience and reducing friction in decentralized services. Authentication and access control are further simplified through Internet Identity, a privacy preserving, WebAuthn-based authentication system that eliminates the need for traditional wallets or passwords (see Section 2.3.2 for more details).

ICP's ability to host all application layers on-chain, interact seamlessly with both off-chain APIs and on-chain services, and preserve data and logic in tamper-proof canisters offers a compelling foundation for legally compliant and operationally transparent DAO platforms. The following subsections provide an overview of the core components of ICP most relevant to this project's objectives.

2.3.1 Architecture

The ICP introduces a novel blockchain architecture designed to address the limitations of monolithic chains. Rather than relying on a single, global state machine, ICP is structured as a network of interconnected blockchains (Figure 2.1), called subnets, which collectively support large-scale, decentralized computation.



Source: ICP subnet architecture [68]
Figure 2.1: ICP subnets architecture

Each subnet operates independently and is composed of multiple nodes, achieving consensus through a Byzantine Fault Tolerant (BFT) [69] algorithm enhanced by threshold cryptography. This modular architecture enables horizontal scalability, as new subnets can be seamlessly added to the network without disrupting existing workloads. In this model, computation and state are distributed across the network in a way that preserves security, isolation, and composability.

The core building block of this system is the canister, ICP's enhanced version of a smart contract. Canisters contain both code and state and are deployed on subnets. These can interact with each other asynchronously, even across different subnets, using a cryptographically secure routing mechanism based on Chain Key Technology [70].

The architecture is purpose-built to support fully decentralized applications, offering a foundation that enables developers to build both backend and frontend logic directly on-chain.

2.3.1.1 Canister

At the core of the Internet Computer's architecture lies the canister, a powerful evolution of the traditional smart contract. Unlike contracts on other blockchains, which are often stateless or limited in scope, canisters are stateful WebAssembly (Wasm) [71] modules that encapsulate both executable logic and persistent memory. They behave more like decentralized microservices than conventional contracts, enabling rich application behavior across time and interaction cycles.

Each canister exposes one or more public entry points, can store and mutate its internal state, initiate asynchronous calls to other canisters, and interact directly with users. This model allows developers to create robust, modular systems where components communi-

cate securely across the network, including across different subnets.

Crucially, canisters can also serve frontend code, such as HTML [72], CSS [73], and JavaScript [74], directly from the blockchain. This makes it possible to deploy fully self-contained applications where both the user interface and backend logic reside on-chain, without the need for centralized infrastructure providers like AWS [75], IPFS [76], or GitHub Pages [77].

Developers can choose between several supported languages, depending on the nature of the component:

- **Rust**: Offers fine-grained control, high performance, and safety guarantees, ideal for logic, heavy or critical components.
- **Motoko** [78]: A higher-level language created by DFINITY [79] to simplify actor-based programming on Wasm, especially suitable for those new to low-level systems programming.
- **TypeScript**: Enables the use of familiar JavaScript tooling while compiling to Wasm for canister deployment, well-suited for interfaces or business logic where rapid development matters.

This flexible and language-agnostic development model allows building a truly full-stack decentralized application, where the backend (Rust) and frontend (TypeScript) components coexist securely on the same protocol, and users interact directly with the blockchain without intermediaries.

In the context of DAO infrastructure, canisters provide a foundational layer for building and automating governance processes, managing user interactions, and enforcing compliance logic, all while preserving the decentralization, auditability, and transparency guarantees of the blockchain.

2.3.1.2 Reverse gas model

A key innovation of the ICP is its reverse gas model, which fundamentally changes how computational resources are paid for on the blockchain. On traditional networks like Ethereum, users must hold tokens to pay for gas fees when interacting with smart contracts. This imposes a barrier to entry, especially for users unfamiliar with crypto tools or for organizations with compliance constraints around holding digital assets.

ICP inverts this logic: canisters (not users) pay for computation, using a unit of measurement called cycles. Cycles are purchased by converting ICP tokens and are consumed

proportionally to the computational resources a canister uses, such as CPU time, memory, and network bandwidth.

By preloading canisters with cycles, developers can build applications where users interact freely without ever needing to manage wallets, buy tokens, or sign gas transactions. This model dramatically enhances usability and enables seamless, Web2-like experiences while preserving full decentralization.

For a DAO management platform, this model is particularly impactful. It allows public administrations, enterprises, or casual users to interact with governance tools, vote, or submit proposals without any blockchain-specific friction.

The reverse gas model therefore plays a critical role in lowering the adoption barrier, making decentralized governance accessible to a broader audience and opening the door to real-world use cases beyond the typical crypto-native crowd.

2.3.1.3 Decentralized hosting

One of the most powerful differentiators of the ICP is its ability to host entire web applications, frontend and backend, directly on-chain. Unlike most blockchain platforms, where smart contracts live on-chain but web interfaces are served off-chain (e.g., via IPFS, Cloudflare [80], or GitHub Pages), ICP allows developers to deploy and serve static assets like HTML, CSS, and JavaScript directly from canisters.

This means a full-stack application, whether it's a voting interface, DAO dashboard, or admin panel, can be entirely decentralized, censorship-resistant, and verifiable. When a user visits a decentralized Application (dApp) built on ICP, every interaction, from fetching the web interface to executing business logic, is handled by the same trustless infrastructure.

This eliminates a common security risk in Web3: frontend–backend mismatch. On Ethereum or Solana, even if the smart contract is secure, the frontend may be hosted off-chain and vulnerable to tampering or version drift. On ICP, however, the frontend is versioned, immutable, and served from the same canister that contains the logic, ensuring consistency and full auditability.

For a DAO management platform, this capability guarantees that governance users always interact with a trustworthy and transparent interface, delivered securely by the same system that enforces the underlying rules. There's no need for reliance on centralized web services, the entire stack lives on-chain, setting a new standard for end-to-end decentralization.

2.3.1.4 Data persistence and privacy

The ICP provides a unique model for storing and managing application data: everything is persisted directly on-chain within the canister's WebAssembly memory, removing the need for off-chain storage solutions like IPFS or centralized databases. This enables users and dApps to operate with a fully integrated and verifiable state model, where logic and data coexist securely inside the same infrastructure.

Data within a canister is replicated across all nodes in the hosting subnet and validated via consensus, ensuring strong integrity guarantees. Unlike typical blockchain contracts that rely on external tools to handle complex state, canisters offer long-lived, mutable, and directly accessible storage, ideal for rich, dynamic applications like DAO platforms.

Crucially, the ICP also provides cryptographic capabilities that enable fine-grained privacy controls. Using vetKeys [81] (a form of distributed key generation) and Threshold ECDSA [82], canisters can encrypt and securely share data, maintaining confidentiality without relying on external key management services. These tools allow for secure, on-chain access control mechanisms, with private keys never existing on a single node.

For a DAO system, this means sensitive information, such as user identities, voting results, role permissions, or internal governance configurations, can be stored and accessed entirely on-chain, while still enforcing privacy policies. This unlocks use cases that require both transparent governance and data protection, a combination rarely supported in traditional smart contract platforms.

2.3.1.5 Interoperability

One of the standout features of the ICP is its built-in support for both off-chain HTTPS requests and native blockchain integrations, enabling canisters to seamlessly communicate with the external world, whether centralized APIs or other decentralized networks.

Thanks to HTTPS outcalls, canisters can directly initiate requests to external web services, APIs, data feeds, and even webhook-based systems, without requiring oracles or bridges. These outcalls are made verifiable and tamper-resistant through the use of threshold signatures, cryptographically signed by the subnet itself. This ensures that off-chain communication remains consistent with the chain's security model.

In addition to off-chain capabilities, ICP supports native multi-chain interoperability. Through Chain-Key Bitcoin (ckBTC) [83] and Chain-Key Ethereum (ckETH) [84], canisters can hold and transfer Bitcoin [85] and Ethereum assets directly. These wrapped tokens are issued

natively on ICP and maintain a 1:1 peg with their original counterparts, while preserving programmability via canisters.

This dual interoperability layer provides two powerful capabilities for dApp developers:

- **Off-chain integration**, such as pulling verified market data, communicating with legal registries, or triggering workflows in external systems.
- **Cross-chain governance and treasury**, where a dApp can hold BTC, ETH, or ERC-20 assets and use them in programmable logic, enabling hybrid dApp scenarios that operate across multiple blockchain ecosystems.

Together, these features allow a DAO platform to remain open and extensible while maintaining decentralization and auditability, critical for real-world applicability.

2.3.1.6 Performance

The ICP achieves low-latency performance by introducing a clear distinction between two types of canister interactions:

- **Query calls** are read-only operations that execute locally on a single node without involving consensus. These return results in milliseconds and are ideal for fast user interactions such as loading dashboards, proposal overviews, or member lists.
- **Update calls**, by contrast, modify the state of the blockchain and require full consensus among nodes in the subnet. These operations are persisted and finalized with a typical latency of 1 – 2 seconds.

This dual call model enables applications to deliver real-time interactivity without compromising security. For a DAO management platform, this means users can instantly view data, while still relying on secure and verifiable update flows for governance votes, proposal execution, and role changes.

Under the hood, every node in the ICP network runs a comprehensive replica stack that includes consensus, execution, message routing, and state management layers. This replica architecture ensures consistent behavior across nodes, supports high throughput, and contributes to the platform's ability to serve dynamic applications at web speed.

Combined with the Internet Computer's fast finality (typically 2 seconds) and its ability to serve frontend assets natively, this performance model is uniquely tailored for complex, decentralized systems where responsiveness is key, such as DAO platforms operating at scale.

2.3.1.7 Scalability

The ICP achieves horizontal scalability through its subnet-based architecture. Unlike traditional monolithic blockchains, where all smart contracts execute within a single, shared runtime, the ICP network is composed of many independently functioning subnets, each capable of hosting thousands of canisters.

As usage grows, new subnets can be added dynamically by onboarding new node providers. This allows the network to scale organically, distributing workloads across multiple blockchains without requiring forks, protocol upgrades, or changes to existing deployments.

For a DAO management platform, this scalability model ensures that system responsiveness and reliability are preserved even under heavy user load or during governance events involving large communities. It also guarantees long-term growth: the infrastructure can support an increasing number of DAOs, users, and automated services without congestion or performance degradation.

This dynamic and modular architecture provides a clear advantage over traditional blockchains like Ethereum, where all smart contracts compete for the same global blockspace, leading to bottlenecks and rising fees. On ICP, compute and storage capacity grows with demand, enabling sustainable and predictable scaling for complex applications.

2.3.2 Internet Identity

Internet Identity is the native authentication system of the ICP. Developed by the DFINITY Foundation, it offers a secure and privacy-preserving alternative to traditional login methods, without requiring usernames, passwords, or crypto wallets. Instead, it uses WebAuthn compatible devices, such as fingerprint readers, Face ID, or hardware keys like YubiKey.

At a high level, each user creates a pseudonymous identity anchor, which is bound to one or more devices. Authentication is then performed locally on the device via cryptographic signatures. Importantly, the user's identity is abstracted through a domain-specific principal, meaning that services on ICP can recognize and authenticate users without being able to track their activity across other dApps.

This architecture brings several advantages for decentralized platforms:

- **Simplified onboarding**: Users can access dApps features immediately without managing wallets, passwords, or email-based registration flows.
- **Privacy protection**: No personally identifiable information is required or exposed dur-

ing authentication.

- **Security by design**: Authentication keys never leave the user's hardware device, minimizing risk of phishing or key reuse.
- **Decentralized by nature**: Internet Identity itself runs as a canister on ICP, eliminating reliance on third-party services like OAuth [86] or Firebase [87].

From a DAO perspective, this model is particularly powerful. It aligns seamlessly with the reverse gas model, allowing users to participate in governance actions without needing tokens or worrying about transaction fees. At the same time, it delivers a user experience that is accessible even to non-crypto-native participants, an important consideration when targeting broader audiences such as institutions, public entities, or civil society groups.

Although Internet Identity is currently exclusive to ICP and requires device-specific registration, its privacy guarantees, usability, and native integration make it an ideal choice for building decentralized applications with real-world reach and minimal user friction.

2.3.3 Governance and Network Nervous System (NNS)

The ICP is not only a platform for decentralized applications, it is itself governed by a decentralized autonomous organization known as the Network Nervous System (NNS) [88]. This system governs the entire protocol and infrastructure of the ICP, making it a foundational example of DAO governance at scale.

The NNS controls the network's core system canisters, including those responsible for node registry, subnet management, token economics, and software upgrades. All fundamental changes to the protocol, such as adding new nodes, modifying consensus rules, or adjusting staking parameters, must be proposed and approved through this DAO.

Governance is enacted through neurons [89], which are created by staking ICP tokens. Each neuron can submit proposals and vote, with voting power determined by the amount staked and the time commitment (dissolve delay). Voting is fully on-chain, and all decisions are recorded immutably and executed automatically.

As of 2025, the NNS manages billions of USD in staked ICP, oversees dozens of subnets, and has processed over 30'000 proposals since its launch. This scale of decentralized control is unmatched among layer-1 blockchains and showcases the operational viability of protocol-level DAOs.

For this thesis, the NNS represents more than just a governance model, it is a real-world

demonstration that complex systems can be run entirely through decentralized governance mechanisms. Its existence reinforces two key points:

- The ICP infrastructure is decentralized by design, with no single point of control, a foundational requirement for fully decentralize DAO platforms targeting legal, financial, and operational autonomy.
- DAO-based governance is already operational at protocol level, validating ICP as a credible and capable environment for managing DAOs that deal with significant assets, permissions, and stakeholder interactions.

The idea of building on ICP allows a platform to be anchored to a system that not only supports DAOs, it is itself governed as one.

2.3.4 Summary

The ICP introduces a unique paradigm for decentralized application development by combining compute, storage, identity, and frontend delivery into a single blockchain environment. Its architecture, based on scalable subnets and threshold cryptography, enables canisters, stateful, WebAssembly-based smart contracts, to execute with high performance and full-stack determinism. Key features like the reverse gas model, decentralized hosting, and HTTPS outcalls lower both technical and UX barriers, making ICP especially attractive for complex, user-facing applications that demand high availability, verifiability, and secure interoperability.

From a DAO management perspective, these capabilities are critical. ICP allows not only for automating governance logic and treasury operations, but also for deploying fully verifiable user interfaces, handling sensitive data with privacy guarantees, and integrating identity and compliance mechanisms all on-chain. Furthermore, the existence of the NNS as a DAO governing the protocol itself showcases ICP's suitability for decentralized governance at scale. These advantages form the technological foundation upon which the proposed platform is designed, as detailed in Chapter 3.

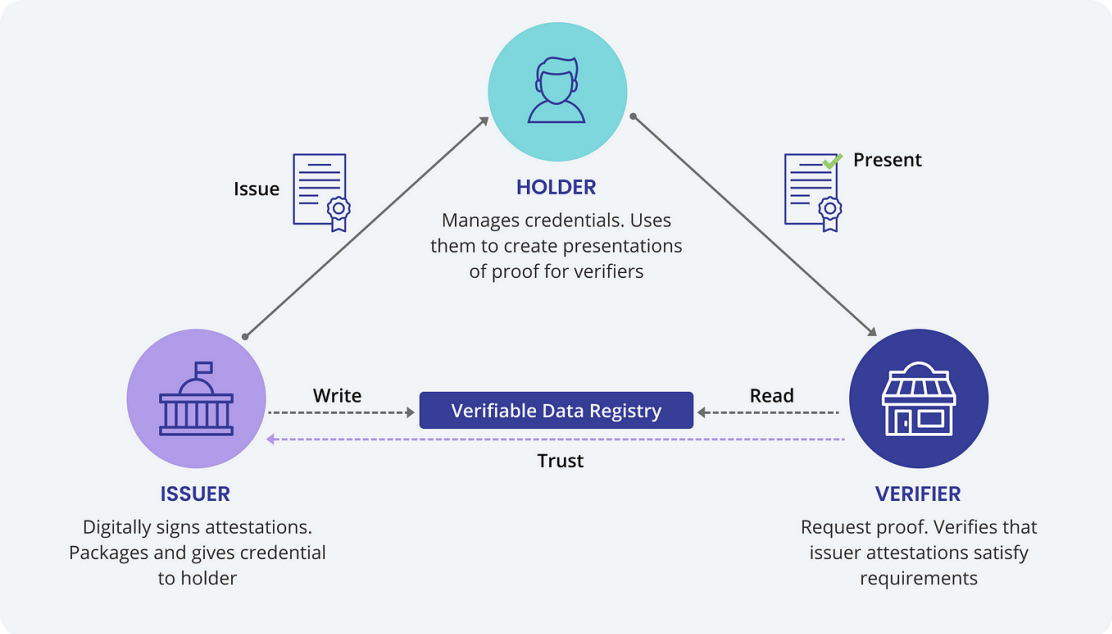
2.4 Verifiable Credentials (VCs)

Verifiable Credentials (VCs) are cryptographically signed digital attestations based on W3C [91] standards that allow users to prove identity-related attributes in a privacy preserving, decentralized manner. Within the context of DAO management, VCs provide an essential

building block for enabling Know Your Customer (KYC) processes without depending on centralized identity providers or exposing sensitive user data unnecessarily.

DAOs that aim to operate in legally compliant settings, whether interfacing with financial services, corporate entities, or government registries, often need to verify user identity at key interaction points. These include onboarding, voting rights allocation, treasury permissions, and jurisdictional restrictions. By integrating VCs, DAOs can meet these regulatory demands while preserving the values of decentralization and user autonomy.

The VC model is built around a "trust triangle" consisting of three roles: the Issuer, the Holder, and the Verifier (Figure 2.2). The Issuer, such as a government, corporate registrar, or compliance authority, creates and signs a credential and provides it to the Holder. The Holder stores this credential and later presents it to a Verifier (e.g., the DAO platform), who checks its validity without needing to contact the Issuer directly. Verification is performed using a decentralized Verifiable Data Registry, ensuring that the process remains both trustable and interoperable.



Source: What Links Identity and VCs Together Across Applications? [90]
Figure 2.2: VC trust triangle

This model introduces several capabilities that are particularly relevant in DAO governance:

- **Decentralization and privacy**: VCs allow individuals or legal entities to retain full control of their identity data. Users can selectively disclose only what is necessary, for

example, confirming jurisdiction without revealing a full passport or address.

- **Legal readiness**: VCs align with emerging standards such as the European Union's eIDAS 2.0 regulation [92] and the Swiss e-ID [93] framework, making them a strong candidate for integration into compliance-ready DAOs.
- **Smart contract compatibility**: Credentials can be automatically verified by canisters, enabling KYC gating mechanisms for sensitive actions like proposal creation, fund access, or delegate assignment.
- **Flexible representation**: VCs can encode a wide range of identity types, from individual citizen credentials to organizational proofs like the Verifiable Legal Entity Identifier (vLEI) [94].

This architecture supports scalable and privacy-preserving compliance processes that respect user autonomy while aligning with institutional requirements, making VCs an increasingly critical component in the evolution of legally operable DAO infrastructure.

2.5 Know Your Customer (KYC)

In the context of legally compliant DAO operations, KYC procedures play a central role. They ensure that participants can be reliably identified when necessary, particularly during onboarding, voting eligibility checks, role assignment, and financial interactions. By enforcing these verifications, DAO platforms can align with regulatory expectations and maintain credibility when interacting with traditional institutions such as banks or supervisory authorities.

A promising approach to enable KYC in decentralized environments is the use of VCs. Rather than requiring centralized storage of sensitive user data, VCs allow identity attributes to be issued, digitally signed, and managed directly by the user. When requested by the DAO, participants can present these credentials to prove their eligibility without exposing more information than required. This model supports principles that are essential in DAOs, such as decentralization, automation, and user sovereignty.

Integrating VC-based KYC mechanisms offers several benefits for DAO platforms:

- Reliable verification of identity and eligibility without retaining sensitive data.
- Automation of checks through cryptographic credential validation.
- Support for both individuals and organizations through tailored credential types.

- Interoperability with evolving legal ecosystems, including the Swiss e-ID and the EU's eIDAS 2.0 regulation.

By adopting such an approach, DAO platforms can meet compliance requirements while preserving the decentralized and user-centric ethos that defines them.

2.6 Methodology

The methodology adopted in this project is multifaceted, combining research, stakeholder engagement, iterative development, and public validation. The goal was to approach the problem from legal, technical, and practical perspectives to ensure a well grounded and impactful solution. The process can be categorized into three main phases:

- **Research and stakeholder analysis**
  - **Stakeholder identification and involvement**: The first step involved identifying the main stakeholders of the project, including legal experts, blockchain developers, institutional actors, and DAO practitioners.
  - **Institutional input**: Indirect communication with the Municipality of Lugano, via one of the project advisors, allowed the collection of preliminary insights and expectations regarding DAO adoption at the city level.
  - **DAO community engagement**: The project was presented during the annual symposium organized by DAO Suisse [95], engaging with experts in the field and gathering critical feedback on the proposed approach.
  - **Expert interviews**: A key part of the research involved both listening to and speaking directly with Lara Schmidt, a specialist at DFINITY who works on the NNS, one of the largest DAOs in operation. Her insights on on-chain governance, DAO tooling, and voting mechanisms helped inform the technical and governance aspects of the platform.
  - **Legal research**: A detailed investigation was conducted using official Swiss government sources [96] to analyze the legal requirements and constraints of different types of associations under Swiss law.
  - **Literature review**: Existing DAO platforms and tools were reviewed to identify prevailing trends, technical architectures, benefits, and limitations.
- **Platform design and development**

- **Agile development**: The platform was developed following Agile principles [97], with biweekly sprints and a formal review every four weeks. Feedback was regularly collected from a founding member of the TBTA, who acted as a domain advisor throughout the project.
- **Iterative prototyping**: A working prototype was incrementally designed and implemented using ICP. Special attention was given to flexibility and extensibility, allowing the future addition of other country and legal form combinations.
- **Open source publication**: The codebase was made publicly available on GitHub [98] to ensure transparency, encourage external review, and support long-term extensibility by the broader community.

- **Deployment and validation**
  - **Public deployment and feedback collection**: The platform was deployed in a publicly accessible environment [99], allowing stakeholders and interested individuals to test the system and provide feedback.
  - **Continuous improvement**: Feedback loops were maintained throughout the project to refine both the platform features and the legal integration logic.

2.7 Summary

This chapter provided a structured analysis of the foundational dimensions necessary to design and implement a legally compliant, decentralized DAO management platform. It began with an in-depth examination of the Swiss legal landscape, identifying the association as the most suitable legal wrapper for a DAO. The comparative analysis of Aragon, Tally, and Juicebox clarified the current state of DAO tooling and highlighted gaps in stack completeness, legal compliance, and deployment transparency especially for projects seeking institutional-grade robustness and regulatory alignment.

The analysis then shifted to the Internet Computer Protocol, the core technological foundation of the project. Through a breakdown of its architecture, canister model, performance features, and decentralized hosting capabilities, the ICP was shown to offer a unique full-stack blockchain environment. Additional components such as Internet Identity and the Network Nervous System demonstrated how authentication and governance are natively supported on-chain, aligning with the goals of a fully decentralized and user-friendly DAO platform. These features make it possible to design applications where both frontend and backend operate on-chain, without compromising usability or scalability.

Lastly, the chapter touched upon Verifiable Credentials and their role in enabling privacy-

preserving identity verification and compliance workflows such as Know Your Customer. While not part of the implementation phase, these components frame the legal and regulatory demands that DAOs increasingly face and suggest future extensions for real-world deployments. Altogether, the analysis sets a comprehensive foundation for the design decisions and system architecture that is detailed in Chapter 3.

Chapter 3

Design

Following the analysis of both existing DAO management solutions and the underlying ICP infrastructure, this chapter presents the architecture and core design decisions of the platform. The goal was not only to build a functional prototype, but to define a system that could realistically be adopted in the real world by institutions, communities, and individuals seeking to manage organizations transparently, securely, and in compliance with legal requirements.

Key design priorities include full-stack decentralization, verifiability of actions, seamless interoperability with off-chain and multi-chain environments, and legal alignment with Swiss regulations. The platform was designed to support scalable user bases and enable a high degree of automation, while offering a privacy-preserving approach to data management. Particular attention was given to flexibility. The system is structured to allow future additions of new legal wrappers, governance templates, or country-specific compliance flows without major reengineering.

The design was developed around a strong separation of concerns, especially in the backend logic, to improve maintainability, clarity, and upgradeability. This approach also reduces centralization risks inherent in ICP's canister controller model, by minimizing privileges and clearly isolating responsibilities across components. The result is a modular system that can be maintained by public institutions and open-source contributors, while remaining accessible and intuitive for end-users interacting through DAO-based organizations. In the following sections, the main components of the design are presented, including DAO lifecycle management, system architecture, and core interaction flows, along with other critical aspects that emerged as necessary to meet the platform's goals.

3.1 Platform's architecture

The architecture of the platform has been carefully designed to support decentralized, transparent, and extensible DAO operations, while remaining compliant with real-world legal requirements. Built entirely on the ICP, the whole system is composed of a set of modular and upgradeable canisters, each encapsulating specific responsibilities. This modularity, combined with a strong separation of concerns, enables the platform to maintain high standards of maintainability, extensibility, and scalability.

The system is designed to work seamlessly even as new legal forms, jurisdictions, or components are added. Most importantly, each canister is independently deployed and replicated across the ICP network, benefiting from the platform's decentralized infrastructure. This guarantees resilience, verifiability, and high availability across all components.

As illustrated in Figure 3.1, the architecture is composed of the following canisters:

- **Agency**: Responsible for the creation of DAOs by deploying legal-form-specific canisters (currently only the "Association" form). When a DAO is created, the Agency initializes the association canister and performs all necessary setup (see Section 3.2.1 for more details). After deployment, the Agency remains inactive unless the DAO needs to migrate to a different legal form, in which case it deploys a new canister and coordinates the data transfer. The rest of the system functions independently of this component during regular operations.
- **Company**: This canister represents the DAO/company instance. It encapsulates all operations, roles, and data related to the specific legal form. The current implementation supports the Swiss association form. In the future, to support other legal structures, new canisters can be introduced to replicate the responsibilities of this one. All other components in the system are designed to interoperate seamlessly with any legal-form-specific canister following the expected interfaces and patterns.
- **SOGC publication**: A shared canister that stores references to official publications from the Swiss Official Gazette of Commerce (SOGC) [100] across all DAOs. This allows DAOs to retain historical continuity and compliance even when transitioning between legal forms. The data model is append-only to ensure traceability and auditability.
- **Document storage**: Similar to the SOGC publication canister, this component provides shared, append-only storage for documents linked to each DAO. It supports decentralized document handling and is designed to eventually integrate features for document verification to ensure immutability and integrity.

- **Discovery**: A shared directory service that maps users to their associated DAOs and provides a discovery mechanism for users to explore other DAOs. While each company canister manages its internal list of members and roles, this canister serves as a cross-platform registry. It improves platform navigability and supports user experience features such as the "random DAO" discovery page (more details in Section 3.2.2).
- **Voting**: A central governance engine that handles the entire lifecycle of DAO voting. It supports vote creation, configuration of on-chain and off-chain actions tied to voting outcomes, tracking of the voting process, and validation of results. This shared canister interacts with other components to execute the result of successful proposals, making it a critical pillar of decentralized governance on the platform.
- **Network**: A utility component that wraps ICP's HTTPS outcall functionality. It allows secure and verifiable outbound HTTPS requests to external services. This canister is shared across the system and currently used by the Voting canister for off-chain interactions, with the potential to be extended for additional use cases.
- **Platform**: Hosts the frontend of the application and serves the web interface directly from a canister. This enables a fully decentralized full-stack application where the frontend is aligned with the backend smart contract logic and protected from tampering. The Platform canister communicates with all backend components to execute user interactions.
- **Internet Identity**: A standard canister provided by the ICP ecosystem for user authentication. Although not developed within this project, it plays an essential role in the platform. It allows secure, decentralized, and privacy-preserving login using WebAuthn-compatible devices, without requiring passwords or wallet extensions.

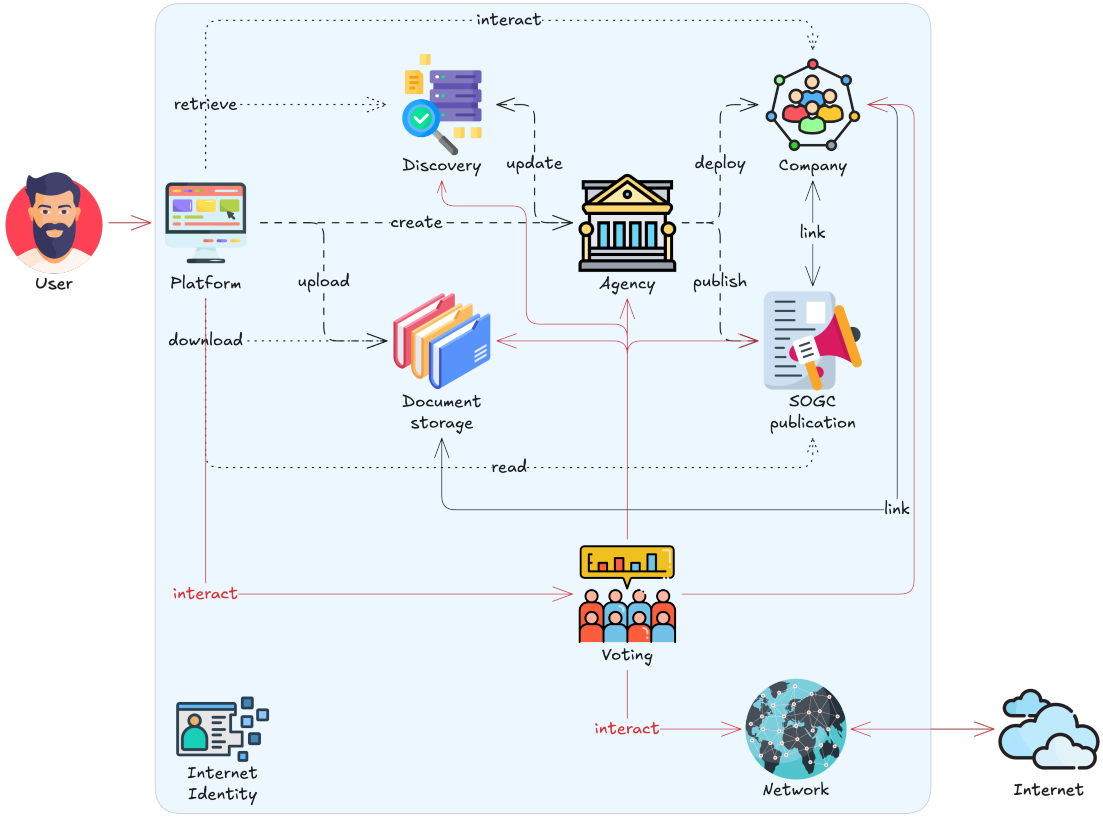


Figure 3.1: Platform's architecture

Together, these components form a robust and flexible system architecture that is aligned with the principles of decentralization and legal compliance. Each canister operates independently, yet cooperatively, through clearly defined interfaces and communication flows. The design ensures that the platform can scale in terms of functionality, users, and regulatory coverage without compromising its decentralized nature.

In the next sections, we will detail how these components interact through core user workflows (Section 3.2) and governance operations (Section 3.3 and 3.4).

3.2 Core interaction workflows

Having introduced the architectural structure of the system (Section 3.1), this section focuses on the key interaction workflows that bring the platform to life. These workflows represent the real actions and sequences that occur within the current implementation, showcasing how users engage with the platform and how the various canisters collaborate to support those interactions.

Each workflow is described through a concrete use case that highlights the responsibilities of the components involved, the logic that governs their interactions, and the user perspective throughout the process. By walking through these scenarios, the goal is not only to document the system's behavior but also to demonstrate how the design principles discussed earlier, such as modularity, decentralization, compliance, and usability, translate into practical operations.

The following subsections cover the most relevant workflows, including DAO creation, discovery, interaction, voting, external integrations, and authentication. Together, they offer a comprehensive overview of how the system operates in real-world conditions, setting the stage for understanding its value and extensibility.

3.2.1 DAO creation workflow

The DAO creation workflow (Figure 3.2) represents the legal-compliant process of establishing a decentralized association on the platform. It begins when a user accesses the platform and authenticates through Internet Identity, ensuring a secure, passwordless login. Once authenticated, the user can initiate the creation of a DAO by completing a form that collects the essential legal and organizational details (e.g. name, purpose, members) (more details on Section 4.6.2). The platform validates the submitted data against basic requirements, such as correct field types and the presence of all mandatory values, before proceeding.

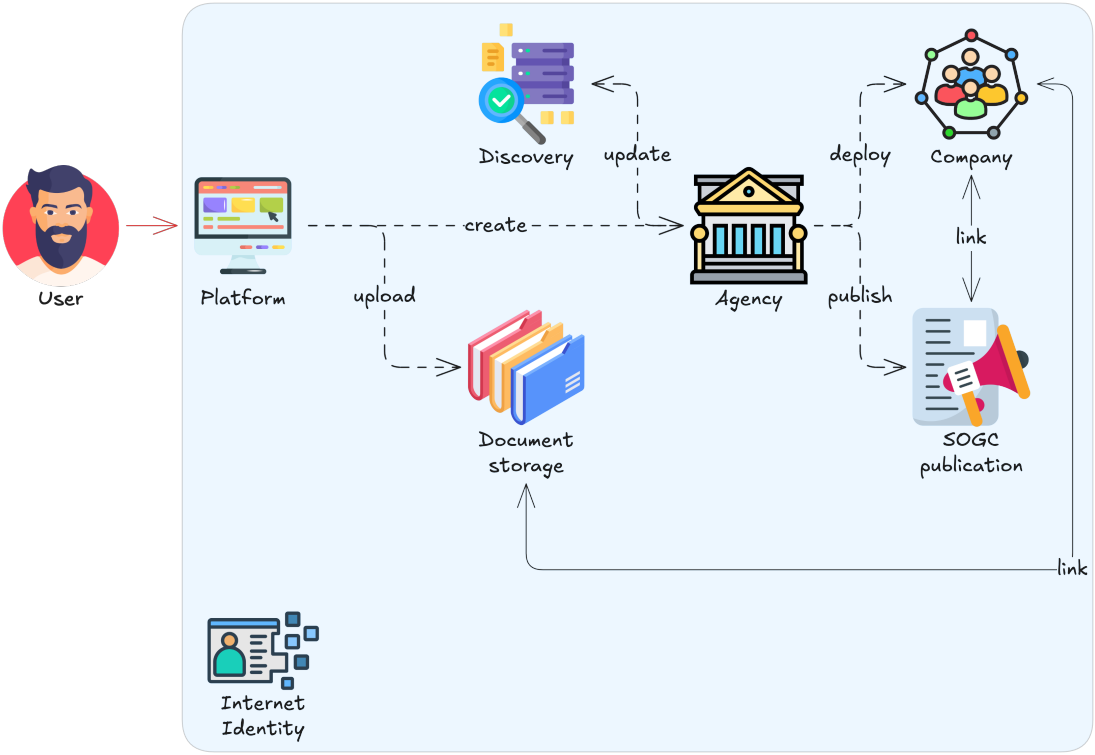


Figure 3.2: DAO creation workflow

Based on the provided information, the platform generates the necessary legal documents off-chain and immediately stores them in the Document storage canister. Although document generation currently occurs off-chain to reduce costs, the process remains transparent because the platform is open source and hosted on-chain, allowing independent verification of its behavior.

After document storage, the platform sends all relevant information to the Agency canister. The Agency, acting as the deployment authority for legal forms, creates a new Company canister corresponding to the chosen form. During the initialization phase, the Company canister receives all required data directly from the Agency. This is the only point in the Company canister's lifecycle where its internal state can be written without going through a voting process, a necessary exception to allow proper setup before governance takes over.

Following the successful deployment, the Agency publishes a legal-compliant entry in the SOGC publication canister to record the company's creation. Finally, the Discovery canister is updated to map the DAO members to the newly created company, enabling the platform to act as a "phone book" for users to quickly find their organizations. From this point onward, any changes to the company canister's data must pass through the platform's governance and voting mechanisms, ensuring that all modifications are transparent, verifiable, and fully

decentralized.

3.2.2 DAO discovery workflow

Figure 3.3 illustrates the role of the Discovery canister within the platform's architecture. Its primary purpose is to act as a decentralized "phone book" for DAO companies, enabling the platform to quickly map authenticated users to the principals (i.e. canister IDs) of the Company canisters they belong to. This is not a functional requirement for the Company canisters themselves, they already maintain their own member lists and role data for operational checks, but it greatly improves lookup efficiency on the platform side.

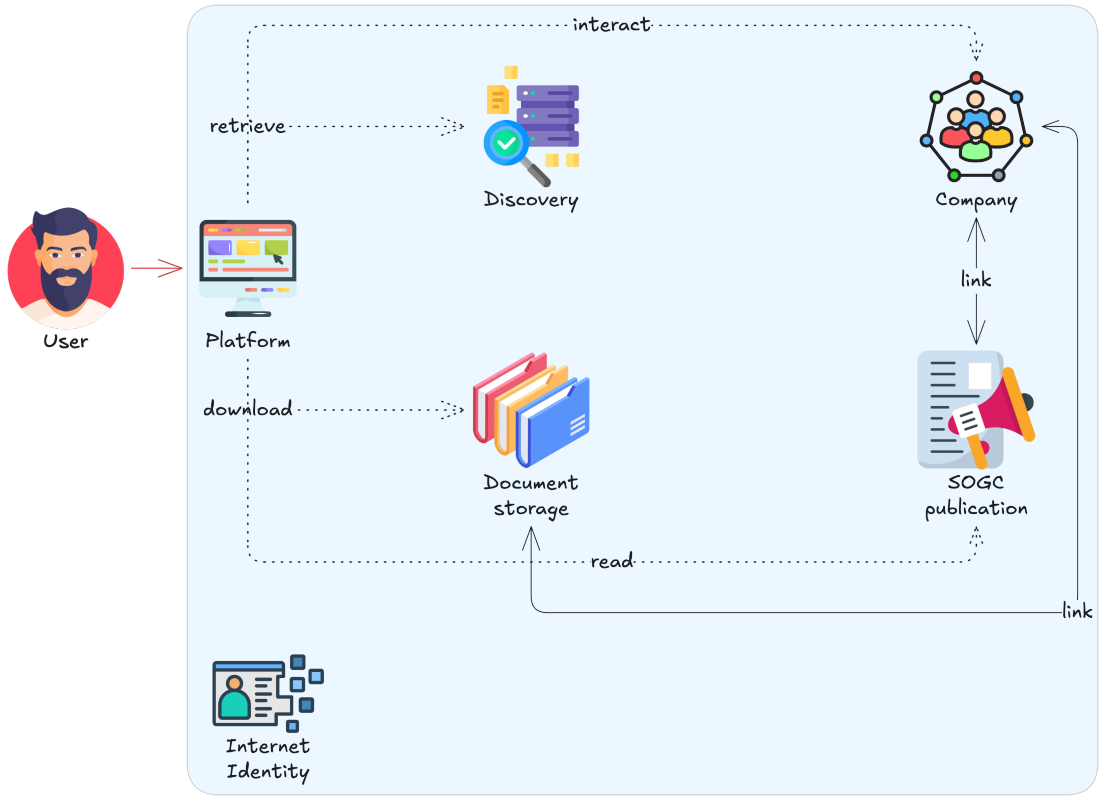


Figure 3.3: DAO discovery workflow

When a user logs into the platform via Internet Identity, the Discovery canister can be queried to return the list of company canisters associated with that user. This enables the platform to fetch relevant metadata directly from each Company canister and, for more complex content such as official documents or SOGC publications, retrieve the corresponding entries from their respective dedicated canisters.

Beyond user-specific lookups, the Discovery canister also supports returning a specified

number of random company canisters, functioning as a DAO exploration tool. In the current proof of concept, these results are selected purely at random from all registered DAOs. However, the architecture leaves room for future enhancements, such as filtering by activity status, applying recommendation algorithms, or surfacing DAOs based on thematic or legal attributes.

3.2.3 DAO lifecycle

The Company canister represents a company instance for a specific legal form, embedding all the rules and operations necessary to ensure legal compliance. In this proof of concept, the starting point is the association legal form, identified during the analysis phase (Section 2.1.1) as the most suitable candidate for integrating DAO processes: it is legally recognized yet has fewer constraints compared to more complex structures involving treasury management, single ownership, or heavy regulatory oversight.

The lifecycle of a Company canister (Figure 3.1) can be described in the following phases:

- **Initialization**: The canister is deployed exclusively by the Agency canister, which receives all required company data from the platform during the creation process. This initialization phase is the only moment when company data is written directly in the canister memory, without passing through a voting process.
- **Data retrieval**: During normal operations, the platform interacts directly with the Company canister to read information. Basic metadata (such as member lists and role definitions) are stored inside the Company canister itself, while more complex data, like documents or SOGC publications, are stored in dedicated shared canisters, with the Company canister holding references to them.
- **Data editing**: Any modification to the Company canister's internal state must go through a voting process. Once a vote is concluded, the Voting canister is responsible for invoking the appropriate methods and applying the changes (more details on Section 4.4.2), ensuring that no individual can unilaterally alter the company's state.
- **Maintenance and resource management**: Each canister on ICP requires cycles to operate. The Agency provides an initial allocation (generally 1 TC) at creation, covering a predefined operational period. After this, the responsibility shifts entirely to the Company owners, who must top up the canister via the platform to keep it alive, similar in principle to paying ongoing operational fees. This mechanism enforces decentralized accountability and prevents excessive central control.

- **Termination / migration**: A DAO can be dissolved, in which case its status is updated to reflect inactivity. The canister can remain online for historical reference until its cycles are depleted, after which it will be automatically removed by the network. Data inside reference canisters are persisted. When changing legal form, the canister is marked as inactive, and all its relevant data is transferred to a new Company canister deployed by the Agency for the new legal form.

This lifecycle model ensures that DAO operations remain transparent, verifiable, and legally compliant, while also supporting future adaptability for new legal forms or evolving regulatory requirements.

3.2.4 Change legal form workflow

During the lifecycle of a DAO, there may arise situations where a change of legal form becomes necessary, whether for strategic, operational, or regulatory reasons. In the implemented ecosystem, each legal form is represented by its own dedicated canister (currently only Association is supported). As such, a change in legal form requires the deployment of a new canister specifically designed to handle the governance and operational rules of that legal form.

The system's design, supported by the Agency canister, makes this process straightforward (Figure 3.4). When a change is initiated, potentially as the result of a proposal approved via the Voting canister in a future production-ready version, the Agency reads all relevant metadata from the current Company canister (Association in the example), marks it as inactive, and deploys a new Company canister (SAGL in the example) for the target legal form. This initialization phase automatically migrates members and their roles to the new canister, ensuring operational continuity.

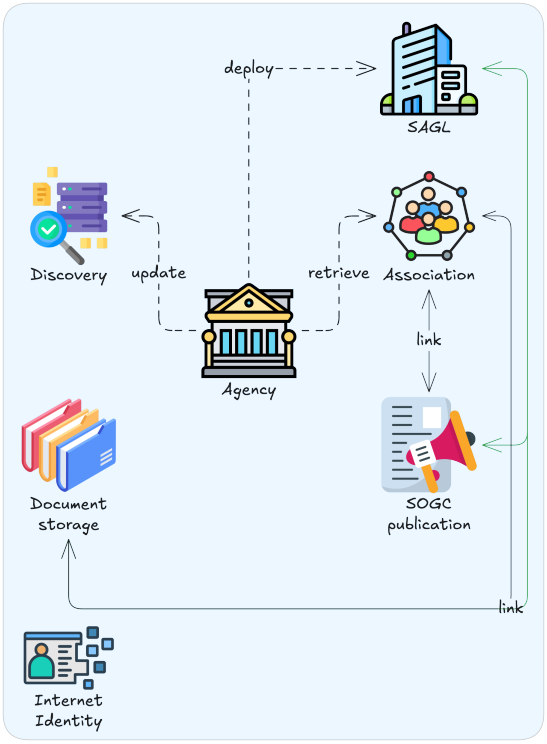


Figure 3.4: DAO change legal form workflow

The Agency then updates the Discovery canister to reference the new Company canister. The discovery mapping is designed to be efficient: instead of storing direct user - principal relationships, it uses a user - canister ID mapping and a separate canister ID - principal mapping. This way, a legal form change requires updating only a single mapping entry, rather than modifying multiple user records.

A key advantage of this design is that historical records are preserved. Since complex data structures such as SOGC publications, documents, and voting records are stored in dedicated shared registries, they remain intact and associated with the company across legal form changes. The old canister remains queryable (directly with its principal), potentially in a restricted read-only mode, ensuring that historical inspection and legal verification remain possible until the canister naturally expires due to lack of cycles.

This approach not only simplifies the operational transition between legal forms but also preserves the legal and historical integrity of the company within the DAO ecosystem.

3.2.5 Voting process workflow

The voting process (Figure 3.5) is at the core of the DAO governance model implemented in the platform, and it is by far the most frequently executed workflow in the daily operation of a Company canister. Every decision that alters the state of the DAO, whether it is a minor operational update or a legally significant change, must pass through this process to ensure transparency, fairness, and compliance. The workflow begins when a user with the necessary permissions accesses the platform and initiates the creation of a new poll.

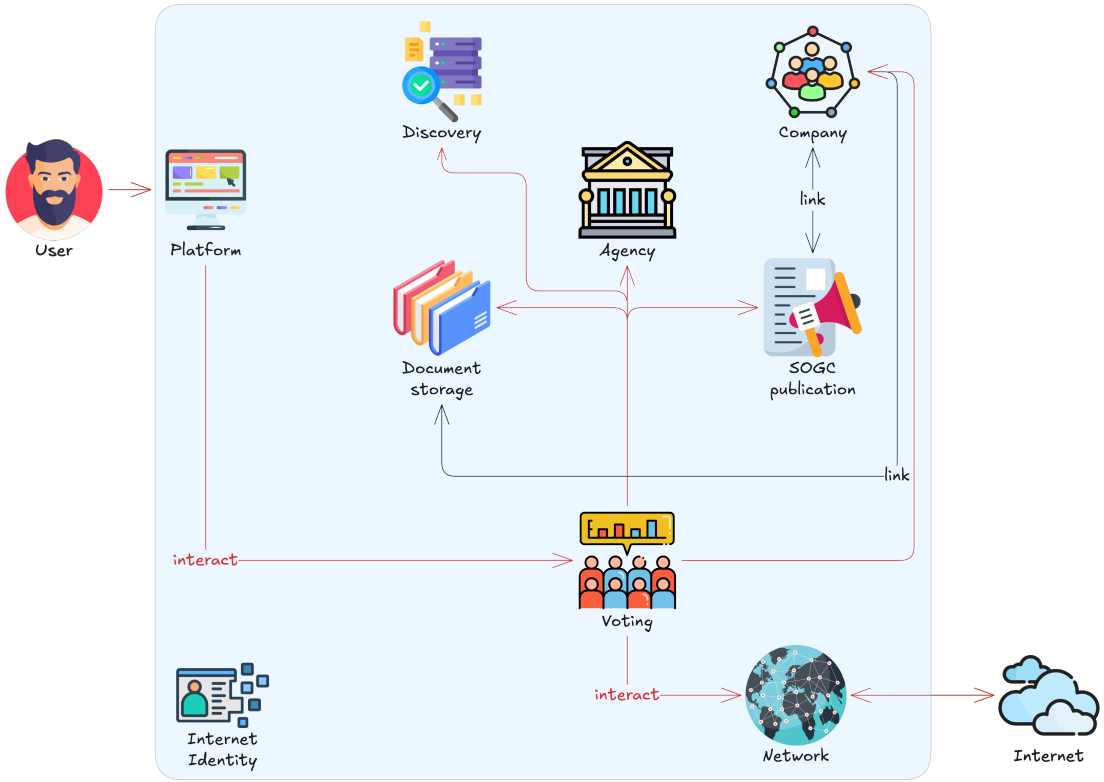


Figure 3.5: Voting process workflow

When configuring a poll, the creator can define multiple parameters to tailor the decision-making process to the specific situation. These include the poll title, a detailed description, the available voting options, the end date, quorum requirements, and the set of eligible voters (see more details in Section 4.6.5). The system enforces strict voting rules, ensuring that each eligible participant can vote only once. In the current proof of concept, all participants have equal voting power, but the design supports future extensions for weighted voting schemes, such as assigning different vote weights based on token holdings, share ownership, or other criteria, without altering the underlying architecture.

One of the most distinctive and powerful aspects of the system is the automatically executed

action mechanism based on the voting result (see Section 4.4.2 for more details). During poll creation, the system dynamically inspects the available canisters (e.g., Company or Network canisters in the current implementation) to detect callable methods and their expected parameters. This process allows the poll creator to select a target canister, choose the method to invoke, and supply the required arguments. This design is inherently flexible, enabling the execution of a wide variety of operations, ranging from updating company metadata in the Company canister to performing complex interactions with external systems via the Network canister. For external operations, the Network canister serves as the bridge to the outside world, handling HTTPS outcalls to any accessible API. This effectively means that, given the right API, the DAO can interact with virtually any off-chain service as part of its governance process.

Poll creators can also decide whether the voting should trigger an official notification or produce formal records in the Document storage and SOGC publication canisters. This distinction allows the system to differentiate between purely internal operational decisions, such as modifying an internal policy, and legally binding changes, like amending the company's registered name. Optional fields in the poll creation process allow the initiator to provide custom messages and additional context for these publications, enhancing clarity and traceability.

Once a poll is launched, eligible voters can view the proposal details, monitor the ongoing vote, and submit their decision. All votes and related metadata are stored on-chain, ensuring complete transparency and auditability. Participants can track the evolving results in real time, fostering community engagement and oversight. The poll closes either when the defined end date is reached or when all eligible members have voted. At that point, the system evaluates the results against the defined quorum and decision rules. If the outcome meets the specified conditions, the corresponding action is executed automatically.

For on-chain actions, verification is straightforward: the results can be validated directly against the blockchain records. For off-chain actions, execution depends on the target service's API. Currently failures are only logged but the platform design allows for advanced handling mechanisms, such as retry strategies, alternative execution paths, or multi-step confirmation workflows. These advanced handling mechanism can be easily added in future releases.

By combining a rigorous, fully transparent voting process with the capacity for automated action execution, the platform creates a governance model that is both technically versatile and legally compliant. This approach not only ensures that every significant decision is subject to collective agreement but also minimizes administrative delays by eliminating the need for manual follow-up actions once a vote has concluded. It empowers DAOs to

operate with the efficiency of traditional organizations while retaining the decentralization, auditability, transparency and flexibility that define blockchain-based governance.

3.2.6 Off-chain network calls workflow

The ICP provides native support for outbound HTTPS requests through its HTTPS outcall canister, enabling smart contracts to interact directly with services outside the blockchain. While this functionality is extremely powerful, the native interface is intentionally generic and requires the caller to specify multiple low-level parameters. From a user experience perspective, exposing such raw complexity to the frontend would be cumbersome and error-prone.

To address this, the developed platform introduces a Network canister that acts as a dedicated wrapper around the native HTTPS outcall canister. This wrapper abstracts the underlying complexity and exposes a simplified interface tailored to the platform's operational needs. By encapsulating common configuration parameters, the Network canister allows other canisters to trigger external requests without having to repeatedly configure or understand the full native API. In the proof of concept, this mechanism is primarily used by the Voting canister to send emails via an external mailing service. This service supports idempotent keys, ensuring that even when the request is executed multiple times (once per replica in the subnet), the external system processes it only once.

This design choice is not only for convenience, it also addresses a critical property of the ICP's replicated execution model. Because each canister instance in a subnet executes the same code independently, an outbound request is performed once per replica. For consensus to be reached, all replicas must receive functionally identical responses. Without idempotency or deterministic response handling, inconsistent results would break consensus and cause the request to fail. When working with services that do not natively support idempotent keys, developers may need to implement a response normalization layer in the Network canister, reducing the payload to only the essential, stable information.

Security is also a consideration when interacting with off-chain services. In this proof of concept, authentication is handled through API keys stored in the Network canister's configuration. While this is sufficient for demonstration purposes, it is important to note that once a request leaves the blockchain, the default trust guarantees provided by the ICP no longer apply, the reliability and integrity of the data depend on the external service. Future enhancements could include additional verification mechanisms, support for multiple authentication methods, request retries, and integration with more complex protocols beyond simple HTTPS calls.

The Network canister thus serves as both a usability enhancement and a consensus-safety layer, enabling seamless and reliable off-chain interactions without compromising the determinism required by the ICP's execution model.

3.2.7 Authentication with Internet Identity

Authentication in the proposed DAO management platform is powered by Internet Identity, a native authentication service provided by the ICP. This mechanism enables users to authenticate using secure, privacy-preserving credentials without the need for traditional usernames and passwords. Each authenticated session is bound to a unique principal that serves as the user's identity within the ecosystem.

As illustrated in Figure 3.1, Internet Identity is integrated into every interaction with the platform. From the initial access to the frontend, through DAO creation, participation in governance, document access, and even off-chain calls, all actions are authenticated at the canister level. While the current proof of concept implements only minimal checks, mainly to verify user identity and ensure required parameters are provided, the architecture is designed to support a more robust, fine-grained authentication and authorization model in the future.

This foundation allows for secure enforcement of rules across all system components. For example, Internet Identity principals could be cross-referenced with membership roles stored in the Company canister to validate voting eligibility, confirm authorization for document submissions, or enforce operational restrictions. By embedding this mechanism across the architecture, the system can maintain full decentralization while ensuring that only authorized actors perform critical actions.

Internet Identity integrated VCs in July 2025. This enables seamless KYC processes directly within the system, associating each user with a legally recognized identity. Such an enhancement not only strengthens compliance with regulatory requirements but also opens the door to a wide range of legally binding operations. By bridging decentralized identity with verified legal status, the platform now supports an extensive range of advanced DAO governance and operational models.

The use of Internet Identity not only strengthens security but also aligns with the platform's vision of complete on-chain DAO lifecycle management. It removes dependencies on centralized authentication providers, reinforcing the trustless nature of the ecosystem.

3.3 Canisters controllers management

The controller of a canister holds complete operational authority over it. This includes starting and stopping execution, installing or upgrading code, deleting the canister, retrieving its status, and adjusting configuration parameters such as the freezing threshold, resource allocation, or the list of controllers itself. Given this level of power, canisters' controller management is a critical aspect of maintaining a secure and decentralized architecture. A poorly managed canisters' controller assignment can undermine the entire trust model of the system, regardless of the robustness of the underlying technology. For this reason, the system design incorporates clear rules and separation of responsibilities, combined with technical safeguards, to minimize the risk of unilateral or malicious actions.

3.3.1 Technical governance of canisters' controllers

From a technical perspective, canisters' controllers are assigned to each canister in the architecture in a way that aligns with canister's role and criticality:

- **Agency** canister is controlled by the service provider. This canister is responsible for creating DAOs and changing their legal form. If disabled, new companies cannot be created, but existing ones can continue to operate normally.
- **SOGC publication** canister is controlled by the service provider. It is used to manage official publications. Its absence does not prevent DAOs from operating, but removes an official communication channel.
- **Document storage** canister is controlled by the service provider. It stores and manages official documents, often for government purposes. If disabled, the legal process behind the platform loses its sense, but daily DAO operations can continue.
- **Discovery** canister is controlled by the service provider. This canister functions as a registry for the platform and is not essential for the DAO itself. Daily DAO operation can continue.
- **Voting** canister is controlled by the service provider, with a multisignature wallet [101] recommended to prevent unilateral shutdowns, since loss of this canister would halt governance operations.
- **Network** canister is controlled by the service provider, with limited impact if unavailable, as it only manages external HTTPS calls.

- **Company** canister is controlled by a multisignature wallet involving all board members, ensuring that it cannot be disabled without board consensus.
- **Internet Identity** canister is managed directly by ICP, removing the need for internal governance.

While the service provider appears to hold significant control, it is important to note that in most real-world deployments this role would likely be fulfilled by a governmental entity. Such an entity is inherently large and complex, with power distributed across multiple departments and regulatory processes, meaning that ownership and decision-making are naturally fragmented rather than concentrated in a single individual or small group.

3.3.2 Real-world and system-level decentralization

While ICP provides a strong technological foundation for decentralization, true resilience comes from combining these technical guarantees with deliberate system-level governance choices. The architecture avoids single points of failure by distributing control among different actors, service providers, governmental entities, multisignature boards, and the ICP infrastructure itself.

This layered approach ensures that no single party can unilaterally disrupt the ecosystem. In some cases, even if a component is technically disabled, its real-world relevance is already compromised if trust in its controller is lost (for example, a SOGC publication canister controlled by an untrusted authority). By designing with both technical safeguards (multisignature, role separation, open-source redeployability) and real-world governance considerations in mind, the system ensures that decentralization is preserved not just in theory, but also in practice throughout its entire operational lifecycle.

3.4 Integration of new company legal form

One of the strengths of the proposed architecture is its high degree of modularity, which makes it straightforward to introduce a new company legal form into the system. The design ensures that most components are agnostic to the specific legal form, allowing them to interact seamlessly with any properly designed Company canister.

The process of integrating a new company form can be summarized in three main steps:

- **Implement the new canister**: Use the existing Association canister as a reference template, but adapt it to fully comply with the legal requirements of the new company

legal form. This includes defining its operational logic, governance rules, and data structures in a legally valid way. The new created canister should extend the base data structure representing a DAO available in the Common knowledge library (see Section 4.2).

- **Update the Agency canister**: Extend the Agency canister so it can deploy the new company legal form by including its compiled code. This ensures that when a user requests to create a company of this type, the Agency has the correct deployment logic and configuration to correctly deploy the new canister.
- **Update the platform**: Modify the frontend to display, as a creation option, the new company legal form. Update the voting creation logic so it can query and interact with the methods of the new canister type during governance processes.

Adding a new company legal form requires no changes to the Discovery, Document storage, SOGC publication, Voting, or Network canisters, as they are already designed to operate generically with any Company canister that follows the agreed interface.

This modular approach ensures that introducing a new company legal form does not require extensive architectural changes. The result is a future-proof design where legal and business requirements can evolve over time without forcing disruptive system-wide rewrites.

3.5 Summary

The design of the platform is guided by the need to combine technical decentralization, legal compliance, and practical usability. The architecture relies on a network of specialized canisters, each responsible for a distinct concern, which ensures modularity and scalability. This structure leverages the native properties of the ICP while facilitating future extensions without requiring disruptive changes to the core system.

The interaction workflows define how users and organizations operate within the platform. DAO creation, discovery, lifecycle management, voting, and legal form transitions are modeled to reflect real organizational processes while exploiting automation and verifiability. The voting workflow introduces automatic execution of actions with on-chain guarantees, while off-chain calls extend interoperability beyond the ICP ecosystem. Authentication through Internet Identity provides a secure and user-friendly access point, with role-based governance logic prepared for future expansion.

Controllers' management is addressed both from a technical and systemic perspective to support real-world decentralization. Responsibilities are structured in a way that prepares

the transition from a proof of concept deployment toward a fully decentralized governance framework. Finally, the platform is designed with extensibility in mind, while the current focus is on Swiss associations, the architecture accommodates additional legal forms and jurisdictions with minimal adaptation.

Chapter 4

Implementation

The Implementation chapter translates the conceptual design, described in Chapter 3, into a working proof of concept, developed as a first case study to demonstrate the potential of managing a DAO entirely on the ICP. While not intended as a production-ready system, the implementation aims to showcase the feasibility of this approach and the opportunities it unlocks if further developed. The system has been built to be modular, easily extendable, and capable of handling the complete DAO lifecycle in a fully decentralized manner. The following sections present the main technical aspects of the implementation, including the chosen technology stack, the architectural details with type management strategies, data persistency on ICP, the key development challenges encountered, the testing approach adopted to validate the solution, and an overview of the platform's key pages.

4.1 Technological stack

The proof of concept was designed to run entirely on the ICP, taking full advantage of its decentralized architecture and unique features (described in Section 2.3). Both backend and frontend are deployed on ICP, ensuring that the entire platform, from logic execution to user interaction, operates in a trustless, on-chain environment. The system's modular design is complemented by a modern and well-documented technology stack, chosen not only for performance and reliability but also to maximize learning and maintainability.

On the backend side, all canisters are written in Rust, the language officially maintained by the DFINITY Foundation for ICP development. Rust was selected for its performance, memory safety, and strong typing, as well as for the opportunity to deepen knowledge of a language that is increasingly relevant in blockchain contexts. Canisters expose their func-

tionality through Candid interfaces [102], which is the ICP standard for serialization and type exchange, ensuring strongly typed and consistent communication between components.

The frontend is implemented as a React [103] web application written in TypeScript, deployed directly on ICP. It uses the shadcn/ui library [104] (built on top of Tailwind CSS [105]) for UI components, offering a modern, customizable, and accessible design system. This combination was chosen for its large community, excellent documentation, and widespread adoption, making it a robust choice for long-term maintainability.

To support development and deployment, several tools and services are integrated into the workflow:

- **dfx** [106]: the official ICP CLI for local development, building, and deploying canisters.
- **Custom deployment script**: automates building and deploying canisters in the correct order. For example, the Company canister is only built (to be later deployed by the Agency), while others are built and deployed immediately.
- **Cargo** [107]: dependency management and compilation for Rust code.
- **npm** [108]: dependency management for TypeScript and frontend build scripts.
- **GitHub** [109]: version control and collaborative code hosting.

External services play an important role as well:

- **Courier** [110]: an external messaging service for sending emails through the Network canister. It was chosen for its free tier, external hosting (minimal maintenance), and support for idempotency keys, which are essential in ICP to ensure consensus across replicated canisters (as explained in the Section 3.2.6).
- **CycleOps** [111]: a community service for automated cycle monitoring and top-ups, reducing the operational risk of canisters running out of resources.

Development is carried out locally using an ICP replica to allow for fast iteration and testing. Once a proof of concept is ready, the platform is deployed on the mainnet to evaluate real-world performance, validate assumptions, and confirm the system's reliability under true decentralized conditions.

4.2 Codebase architecture

The overall architecture of the platform is organized to ensure modularity, maintainability, and scalability, while taking full advantage of the ICP's decentralized environment. The codebase is divided into two main parts, backend and frontend, each with a clear internal structure and responsibilities (Figure 4.1).

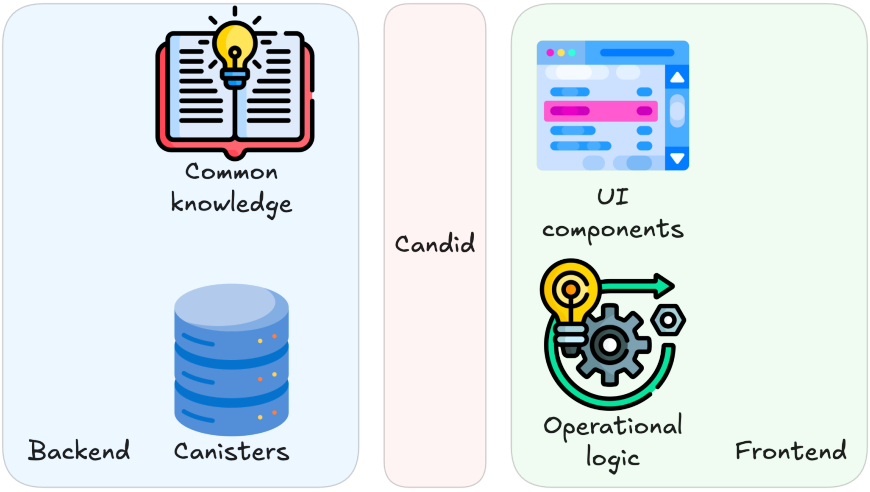


Figure 4.1: Codebase architecture

The backend is fully deployed on ICP and is written in Rust. It is structured into two main layers as shown in Figure 4.1:

- **Common knowledge**: Implemented as a Cargo library shared across all canisters. This module contains the system's common domain knowledge. It includes data models, repositories, services, inter-canister communication abstractions, and shared utilities such as validation and formatting helpers. Centralizing these elements ensures type safety, avoids duplication, and minimizes inter-canister dependencies, while enabling consistent logic reuse across the system.
- **Canisters**: Each canister is implemented as a separate Cargo module with its own Candid interface specification. The internal architecture of every canister follows a controller – service – repository pattern, which keeps business logic separated from persistence and operational logic. This structure promotes clean code organization and simplifies upgrades or feature extensions in the future.

The frontend is a decentralized React application written in TypeScript and deployed directly on ICP. It uses shadcn/ui (based on Tailwind CSS) for UI components, enabling global theme

management and consistent design. Its structure is organized into two key parts as depicted in Figure 4.1:

- **Operational logic**: This layer is divided into:
  - **Types**, TypeScript equivalents of Rust types generated from Candid bindings. While these bindings act as Data Transfer Objects (DTOs) [112], an abstraction layer ensures that data is handled correctly at runtime (more details on Section 4.2.1).
  - **Data management**, Services handle communication with Rust canisters, and Providers act as centralized controllers that manage and cache state using react-query [113], with periodic refreshes to keep data up to date.
- **UI components**: Includes reusable components, views, guards, loaders, and other interface elements, maintaining a clean separation between presentation and logic.

Although there is no fully automated synchronization between Rust types and their TypeScript counterparts, the TypeScript type system ensures that any changes in Candid bindings trigger compilation errors, making inconsistencies visible during development. The architecture also follows standard naming and structuring conventions, which simplifies onboarding for new contributors and ensures consistency when integrating new canisters or components.

4.2.1 Type managements

Type consistency between the backend canisters and the frontend platform is a critical aspect of this project. The implemented approach ensures that data exchanged across the system maintains strict typing guarantees, avoiding runtime mismatches and reducing potential integration errors.

The development process begins in the backend, where the developer edits or adds types in the Rust codebase and ensures that these are correctly reflected in the Candid interface. During the build process, TypeScript binding types are automatically generated from the Candid interface. These generated types act as DTOs and are used in the frontend as the base for creating domain-specific abstractions.

In the platform, each generated type is wrapped inside a TypeScript class. This additional layer ensures that the final object used at runtime not only respects the backend-defined type structure but also provides a concrete class where domain logic or utility methods can be implemented in the future. Even if the wrapping process from generated DTOs

to final classes is manual, the TypeScript compiler enforces compatibility: any changes in the generated bindings that break the abstraction triggers compile-time errors, alerting the developer immediately.

This layered approach, Rust type to Candid type to generated TypeScript DTOs to runtime safe classes, ensures a strong contract between backend and frontend while maintaining flexibility for future extensions. During the early stages of the project, several attempts were made to implement a fully automated Rust to final TypeScript class conversion pipeline. However, these were unsuccessful due to technical constraints, both from Rust library limitations and from the fact that some dependencies cannot be integrated into ICP executable code.

4.2.2 Security and access control

Security within the system is built on top of ICP's Internet Identity authentication mechanism, which is natively integrated across the architecture (Figure 3.1). In the proof of concept, authentication is enforced on canister methods that require user-specific actions, while other endpoints remain open for easier demonstration and testing. However, the design already provides the foundation to extend authentication checks across all interactions, pending further analysis of role-based access policies.

Authorization is managed at the Company canister level. Each Company canister maintains its own registry of members, represented as a list of pairs containing the user's principal and assigned role. In the current implementation, roles are primarily used to determine voting eligibility, but the system is designed to support more granular permission models, allowing specific operations to be restricted to particular roles in future iterations.

From a governance standpoint, all canisters in the PoC are deployed with a controller managed directly by the author, to maintain operational control for testing purposes. In a production setting, this approach could be replaced with a multisig governance model or distributed controller ownership to ensure decentralization and minimize single points of failure.

Keys and credentials for external services, such as those used by the Network canister, are securely stored as environment variables and injected during canister deployment, avoiding hardcoding within the codebase.

Finally, logging is implemented in critical areas, especially for inter-canister calls and outbound network interactions. At this stage, logs are accessed directly from the canister's output, but the system could be expanded to include structured, verifiable audit trails for enhanced accountability.

This minimal yet extensible setup demonstrates the feasibility of a decentralized, role-aware access control framework within the ICP ecosystem, while leaving space for more robust and production-ready mechanisms in future developments.

4.3 Data persistency

On the ICP, each canister combines both logic and memory, making it possible to store application data directly on-chain. This characteristic allows the system to remain entirely decentralized without relying on off-chain databases. However, a critical detail is that when a canister is upgraded, its in-memory data is wiped unless it is stored in stable structures [114] (e.g. BTreeMap, Vec, Cell). The disadvantage of using stable structures is that they are slower compared to keeping everything in memory, much like the difference between accessing RAM [115] and writing to permanent storage.

For this reason, all important data in the platform, such as company registries, membership lists, voting records, and SOGC publications, are stored in stable structures to ensure persistence across canister upgrades. While practically all data within the system are important, it is necessary to distinguish between sensitive and non-sensitive information to define appropriate protection levels.

At the time this proof of concept was developed, all data stored in canisters was inherently public. Early in the project, DFINITY announced work on vetKeys, a feature enabling private data management inside canisters. A few weeks before the project's conclusion, vetKeys was released, so the future development roadmap now includes integrating this functionality. This will make it possible to handle sensitive data in an encrypted form directly on-chain, without altering the general architecture but requiring an analysis of where and how to apply it effectively.

In future iterations, the combination of vetKeys with the existing role-based access model will allow fine-grained, selective data sharing. This means that certain sensitive data, such as identity-linked company records or confidential documents, could be encrypted and made accessible only to authorized principals, while still benefiting from the decentralization, persistence, and verifiability of the ICP.

Communication between backend and frontend is managed through the Candid interface. While Candid ensures consistency between Rust and TypeScript, it also introduces a strong coupling between these environments, especially in terms of type definitions, an aspect already discussed in the Types management subsection (4.2.1).

4.4 Development challenges

Building this proof of concept means working across two different languages, Rust for the backend and TypeScript for the frontend, while deploying on the emerging ICP ecosystem. This combination brings a number of practical challenges, from managing configurations to automatically executing actions and handling local canister IDs.

Rather than obstacles, these become opportunities to design tailored solutions and refine the workflow, providing insights that could guide future iterations. The following subsections outline these key challenges and the approaches used to address them.

4.4.1 Configuration management

While in a language like TypeScript configuring environment variables is straightforward, in Rust the process is less direct due to its compiled nature. This creates a kind of chicken-and-egg problem [116]: to specify certain environment variables, such as the IDs of other canisters that need to communicate with each other, the canisters must first be deployed to obtain these IDs. However, the canisters cannot be deployed without first compiling the code, which would normally require the IDs to be already defined.

To address this, a pattern commonly used in the ICP ecosystem is implemented, where configuration values are stored in canister memory as standard metadata. In this approach, the canister code is written to reference values stored in memory rather than compile-time constants. This allows the code to be compiled and deployed without knowing the final configuration values. Once deployed, the configuration can be set or updated dynamically, ensuring that the canisters can interact correctly without requiring a full redeployment.

This design not only solves the circular dependency issue but also decouples deployment from configuration management, improving maintainability and flexibility. In a production environment, the same approach could be extended to securely update configurations at runtime, such as API keys, integration endpoints, or canister interconnection parameters, without altering or redeploying the canister code.

4.4.2 Automatically executed action management

In this proof of concept, the execution of post-vote actions is designed to be fully automated through inter-canister calls. When a poll is created, the system stores three essential elements: the target canister ID, the method to invoke, and the method's arguments. Argu-

ments are stored as a list of strings to maintain flexibility and allow for a wide variety of use cases. At runtime, the Voting canister convert these string arguments to the correct type and invokes the target method.

Implementing this in Rust is challenging due to its strong type system. The parsing logic has to handle different Candid data representations:

- **Basic types** such as `text`, `nat`, `bool` and so on.
- **Records** which are equivalent to objects. If the argument string contains an object-like structure (using the syntax `{ ... }`) the system recursively processes its fields.
- **Variants** which are Candid's representation of enums, typically expressed as (using the syntax `{ X: null }`), where `X` is the selected enum value and `null` indicates that this value is selected.

The most complex part is ensuring that the reconstructed argument matches exactly the expected signature type of the target method, not only in content but also in structure. After significant trial and error, a parsing and type-mapping process is implemented to reliably support these cases.

The system also needed to account for conditional execution. Actions are executed only when the poll result matches predefined conditions, such as an affirmative outcome or a specific custom option. To support this, each possible voting option is paired with its corresponding action at poll creation time. For example, standard options such as "Accept" or "Reject" can map to distinct calls, while custom options ("Option 1", "Option 2", etc.) allow for more granular workflows. During result evaluation, the Voting canister simply identifies the winning option and triggers the associated action, ensuring a clean and predictable execution flow.

4.4.3 Local canister IDs management

This challenge emerged exclusively during local development, where it is common practice in ICP to assign fixed local IDs to canisters deployed via dfx. This approach ensures a stable, reproducible environment in which canister references remain constant between deployments. While effective in most scenarios, it introduces a subtle problem when deploying new canisters programmatically from other canisters, such as when the Agency canister creates a new Company canister, where the ID must be assigned dynamically.

To understand why, it is important to recall how canister ID allocation works. Normally, when deploying a new canister, a request is made to the Management canister [117] (part of ICP

itself), which assigns the next available ID in an incremental sequence. When fixed IDs are specified, this allocation step is skipped, meaning the Management canister's internal counter is not updated. Consequently, when attempting to deploy a new canister without a predefined ID, the Management canister may issue an ID that is already in use, causing the deployment to fail.

The only viable solution is to avoid occupying the earliest IDs in the allocation sequence, leaving them available for the Management canister's incremental assignments. However, canister IDs are not entirely random, their generation follows specific patterns influenced by multiple factors. To address this, a small JavaScript utility is developed to precompute valid canister IDs positioned further along the allocation chain. This ensures that initial IDs remain available for automatic allocation, preventing conflicts while preserving the stability of local deployments.

4.4.4 Summary

The challenges described in Section 4.4 are not the only ones encountered during development, but they represent the most distinctive and instructive cases. Each reflects a specific intersection between the constraints of the ICP ecosystem, the dual-language nature of the codebase, and the architectural choices made for this proof of concept. Rather than providing an exhaustive list, the section presents concise "pills" of knowledge, insights, and solutions that emerge directly from hands-on experimentation. These lessons not only shape the current implementation but also serve as practical references for future work on similar projects.

4.5 Tests

Given that this project is a proof of concept, the testing strategy aims to strike a balance between coverage and development efficiency, ensuring a stable foundation without overinvesting in production-grade quality assurance. The focus is primarily on the backend, where the core business logic resides, to guarantee that the underlying design remains solid and maintainable before opening the codebase to contributions from others.

Two main categories of automated tests are implemented:

- **Unit tests**: Developed with Cargo test. Focus on validating core business logic inside the canisters, such as voting mechanisms, DAO state transitions, and other critical

backend operations. Utility functions are tested only when they directly impacted these processes.

- **Integration tests**: Executed in a local simulated ICP environment using dfx. Verify that canisters could correctly interact with each other and that the overall system behaves as expected when executing key workflows.

In addition to automated testing, a public PoC was deployed before the project's conclusion, enabling real users to experiment with the system. While feedbacks, from PoC and development versions, are mostly collected informally, the GitHub repository is left open for public issue submissions, providing a channel for more structured bug reporting and suggestions.

These efforts provided valuable technical and usability insights, helping to identify both functional gaps and areas where the user experience could be improved. However, it is important to note that current test coverage is intentionally limited to the scope of a proof of concept. For a production-ready release, additional tests, including performance, security, and broader end-to-end scenarios, are essential.

4.6 Platform

This section presents the user-facing components of the developed platform, illustrated through a series of screenshots. The goal is not to provide a complete evaluation of the user interface, but rather to demonstrate the functional flow of the proof of concept and highlight the main features available to users. The structure follows a typical user journey, starting from the homepage and progressing through DAO creation, management, and participation in governance processes.

The showcased pages include the homepage, where users can discover existing organizations and access the creation form for new DAOs; the DAO dashboard, which aggregates key information and actions; the voting modules, covering both proposal creation and participation; and the dedicated spaces for document management and official publications. Finally, the platform integrates an email notification mechanism, designed to enhance user awareness of governance activities.

Together, these elements illustrate how the proposed architecture translates into a concrete, operational environment, making visible the core workflows discussed in the design section (Chapter 3) and implemented entirely on ICP canisters.

4.6.1 Homepage

The homepage of the platform (Figure 4.2) serves as the primary entry point for authenticated users, offering a clear overview of their participation in DAOs. At the top of the interface, the authenticated principal is displayed, ensuring transparency about the active identity in use. The navigation area provides two main options:

- **My DAOs**, which lists the organizations in which the user is a member.
- **Explore**, which allows users to browse a randomly generated set of other DAOs available on the platform.

This structure supports both personal engagement and discovery, guiding users through their first interactions with the system.

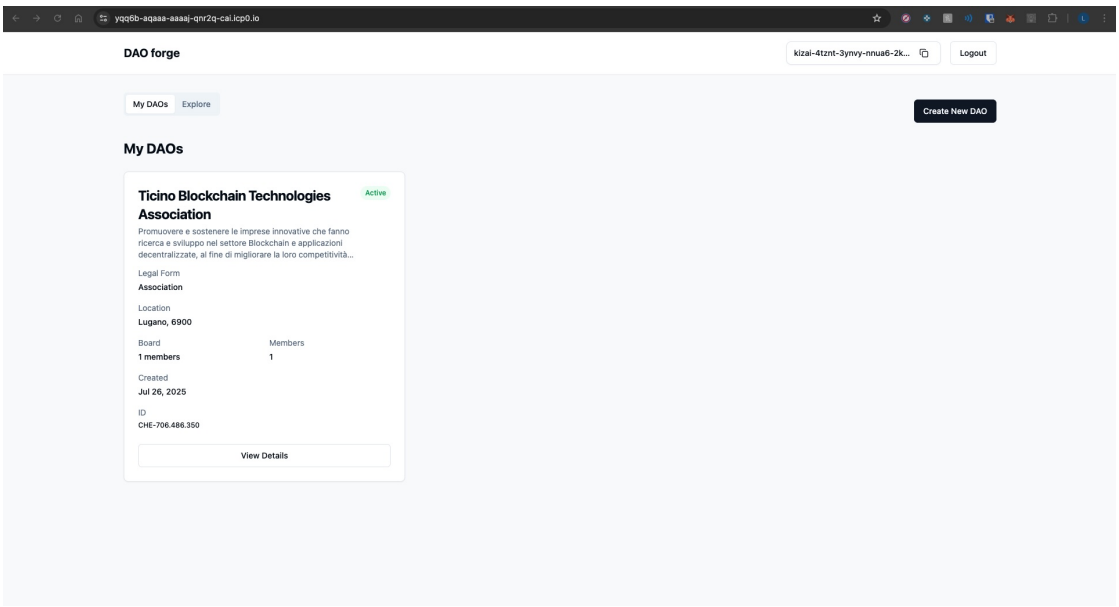


Figure 4.2: Platform's homepage

Below the navigation, the interface presents individual DAO cards that summarize key information about each organization, including its legal form, location, board composition, membership size, creation date, and unique identifier. At this stage, the card is primarily informational, with the main interaction being the ability to access the DAO dashboard by selecting "View Details" button. This design choice reflects the proof of concept nature of the platform, prioritizing clarity and accessibility over complex interactions, while already demonstrating the potential for a seamless user journey from discovery to detailed governance activities.

4.6.2 DAO creation form

The DAO creation form (Figure 4.3) provides the interface through which a new organization is instantiated on the platform. It guides the user through the input of essential company information, including the name, purpose, legal form, and location details. The interface also allows the specification of members, currently identified by their principal, along with their respective roles within the organization. By design, the creator of the DAO is automatically added to the member list, ensuring that the entity always begins with a valid and functional governance structure.

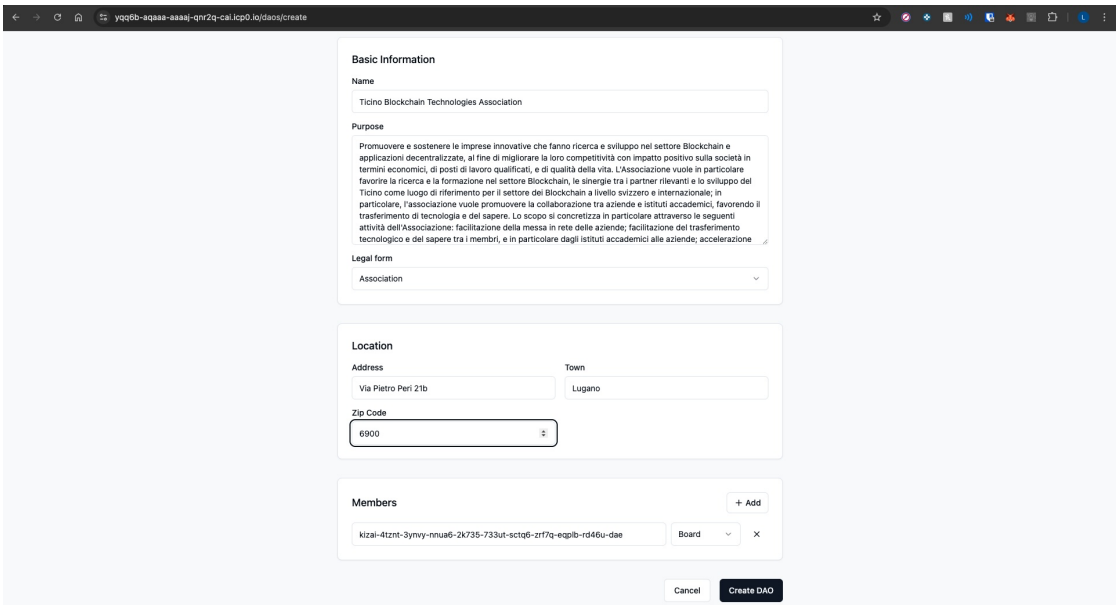


Figure 4.3: DAO creation form

Figure 4.3 illustrates a section of the DAO creation form, which is designed to balance legal compliance requirements with usability. Once the information is provided, the user can finalize the process by submitting the form through the “Create DAO” button. At this stage, the platform automatically deploys the corresponding canister, generates the required documents and SOGC publications, and links them to the newly created DAO. This workflow ensures that the organization becomes immediately operational and seamlessly integrated into the broader system of the platform.

4.6.3 DAO dashboard

Figure 4.4 illustrates the DAO dashboard, which serves as the central point for accessing and managing all information related to a specific organization. At the top, the dashboard

provides a concise overview including the name, legal form, address, number of members, and creation date. This summary ensures that essential details are immediately visible to the user, reinforcing both clarity and traceability.

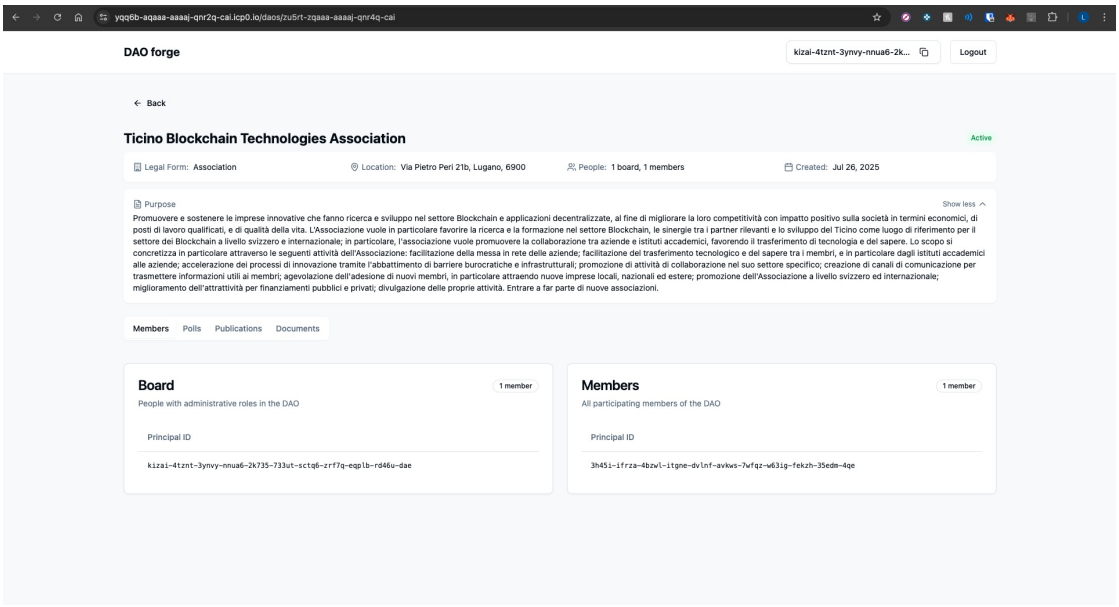


Figure 4.4: DAO dashboard

Under the overview, the dashboard organizes the DAO's resources into tabbed sections, covering members, polls, publications, and documents. The example in the figure displays the members' section, where participants are listed according to their role, distinguishing between board members and standard members. This structure highlights the internal organization of the DAO and facilitates the navigation between different aspects of its governance and activities.

4.6.4 DAO documents and publications

The platform provides dedicated sections for managing both publications and documents, accessible through the tabbed navigation interface (Figures 4.5 and 4.6). SOGC publications represent official records automatically generated when specific changes occur, such as modifications to the organization's name or address. These records represents the current one of the Swiss Official Gazette of Commerce and are displayed in read-only mode to preserve their integrity.

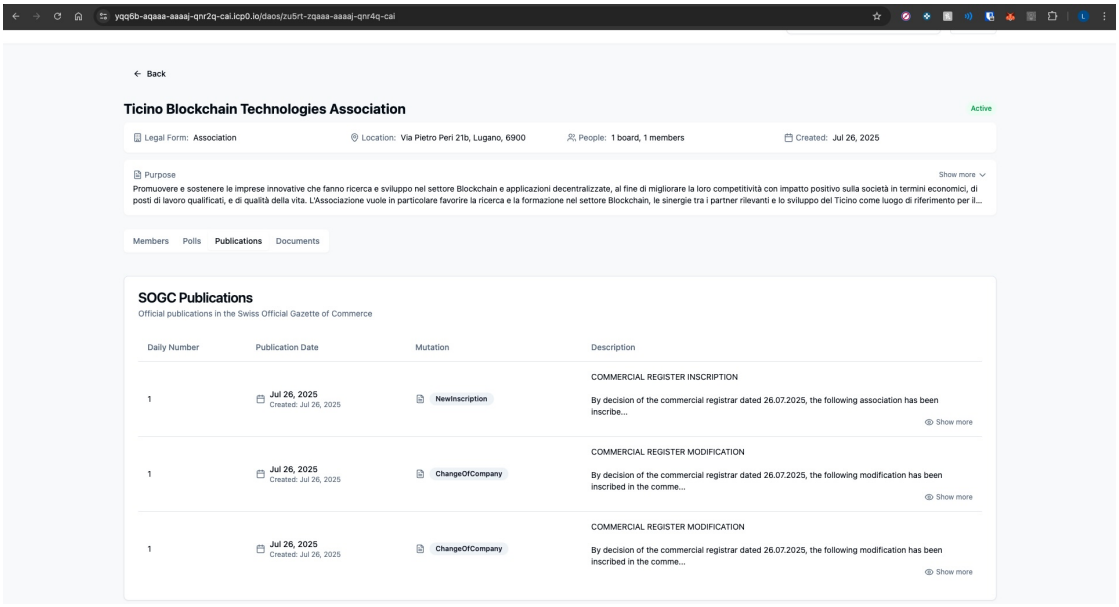


Figure 4.5: DAO SOGC publications

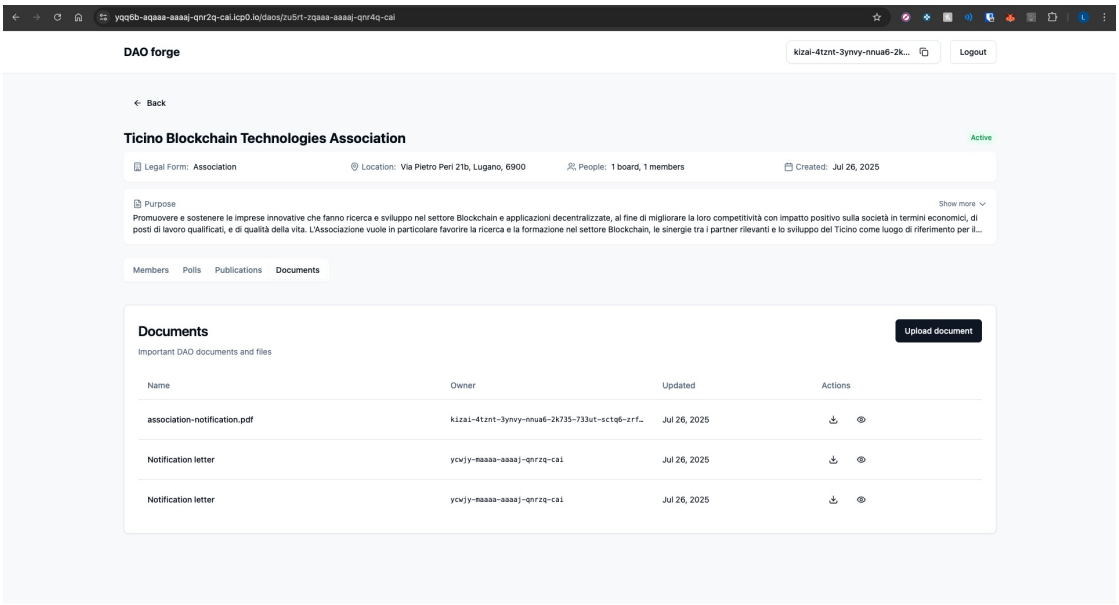


Figure 4.6: DAO documents

In contrast, the documents section offers greater flexibility. It includes system-generated files, such as those associated with SOGC publications, which often require manual signing and submission to comply with legal constraints. At the same time, it allows users to upload additional material, such as historical documents, files created during in-person procedures, or digitally signed contracts. This combination ensures that the platform maintains a comprehensive and legally consistent repository of organizational records while supporting both

automated and user-driven inputs.

4.6.5 Voting creation

Figure 4.7 shows the initial stage of the voting creation process, where the user defines the voting options and configures the action to be executed if the proposal passes. The platform supports both standard voting (Accept / Reject) and custom voting, where any number of tailored options can be defined. Once the options are set, the user selects the target canister, specifies the method to be called, and provides the required parameters. In this example, the parameter is dynamically linked to the winning option of the vote, allowing the DAO to update its name directly based on the community's decision.

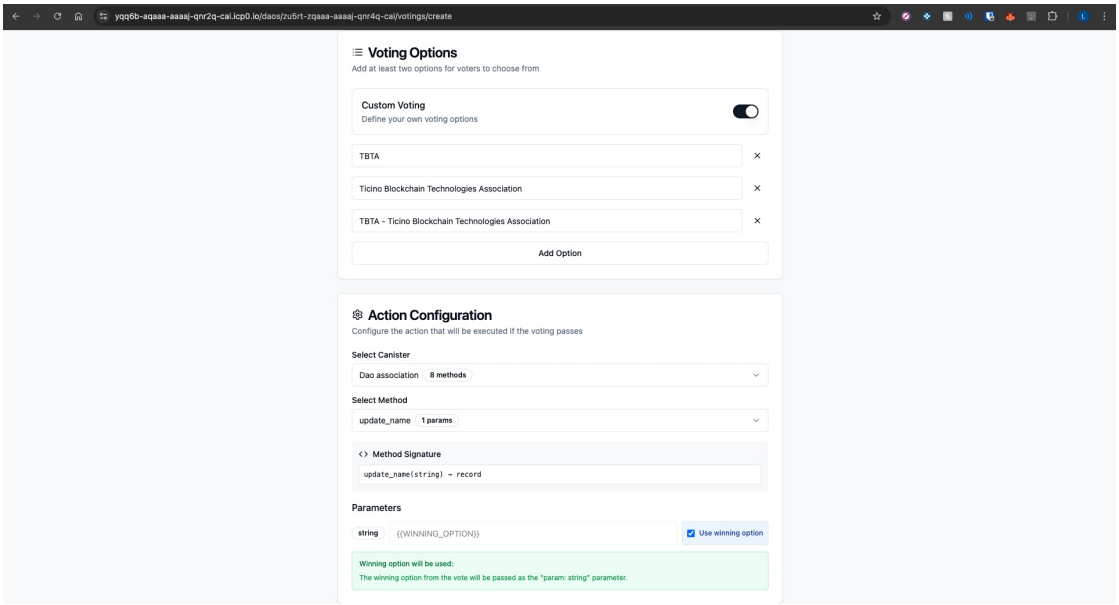


Figure 4.7: Voting creation, action specification

Figure 4.8 illustrates the configuration of voting rules, including the quorum required for validity, the approval threshold, and the selection of eligible voters according to their roles in the DAO. An additional feature is the optional notification letter, which allows the initiator to draft a message that will be sent by email once the vote ends. This message acts as an official communication of the decision and can be archived as part of the DAO's records.

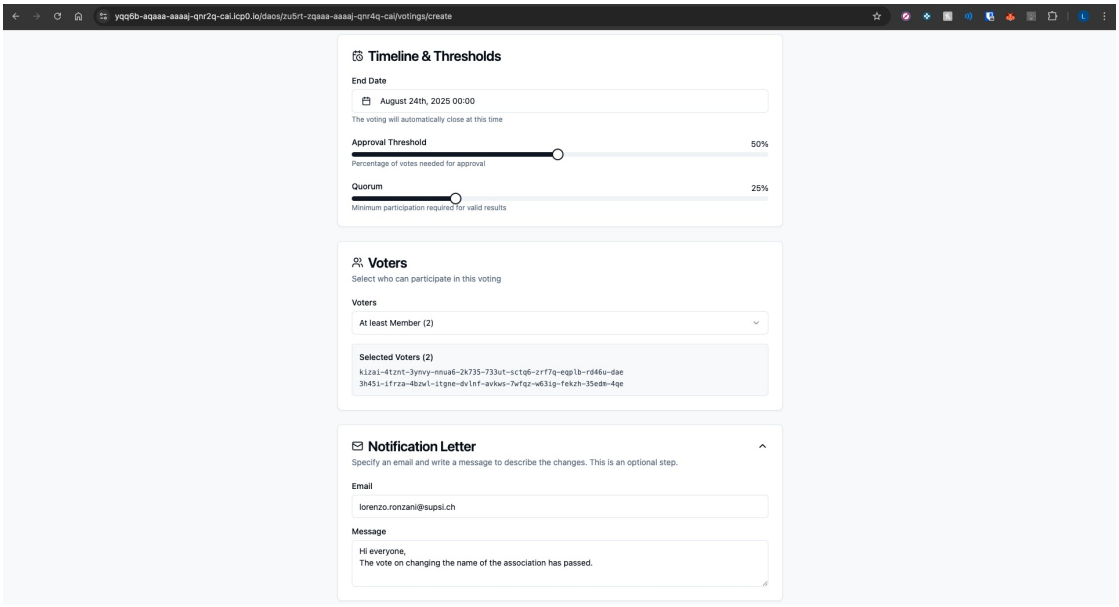


Figure 4.8: Voting creation, survey rules

A key aspect of this process is that the entire execution is automated and occurs directly on-chain. When a vote passes, the specified canister method is called without requiring any manual intervention, ensuring both integrity and efficiency. This design demonstrates how governance decisions translate seamlessly into concrete organizational changes, strengthening the link between community participation and operational execution.

4.6.6 Voting participation

Figure 4.9 illustrates the interface for voting participation once a user has already cast their vote. The modal provides a comprehensive overview of the poll, including its description, applicable rules such as quorum and approval threshold, and the choice made by the user. It also clearly displays the action configured to be executed if the proposal passes, ensuring transparency on the practical outcome of the decision.

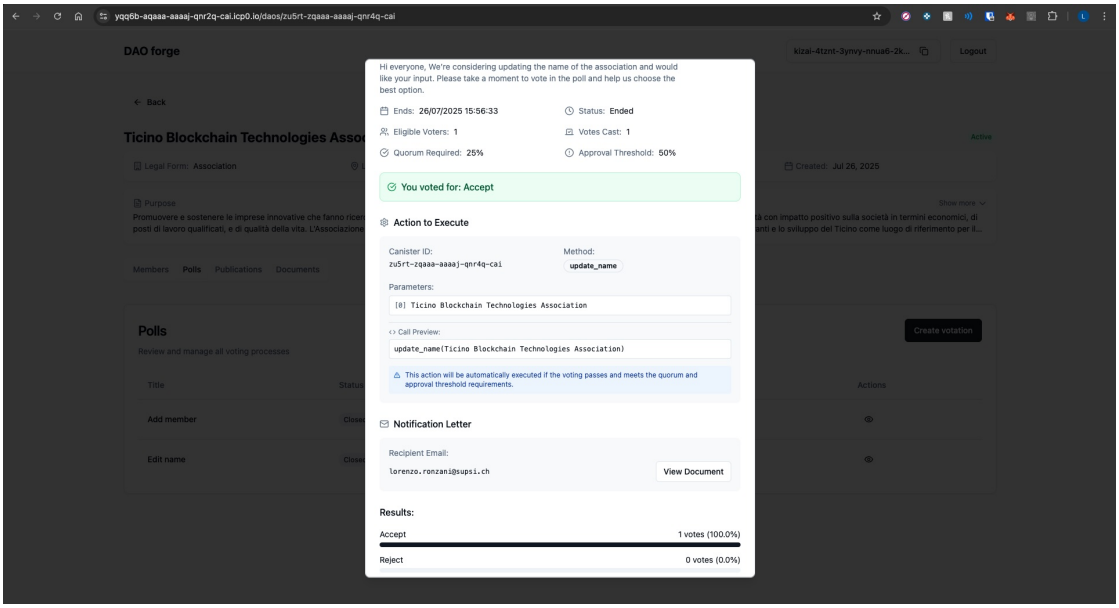


Figure 4.9: Voting participation

In addition, the modal highlights the presence of a notification letter, showing the recipient and offering the option to view the prepared document directly. At the bottom of the interface, the current voting results are summarized, allowing participants to track the progress of the decision-making process in real time. This design ensures clarity for users by consolidating all essential information about the vote in a single view.

4.6.7 Notification email

Figure 4.10 presents an example of a notification email automatically generated and sent by the Voting canister. The structure of the message includes customizable elements such as the subject, the type of notification and the main body text, which in this case is labeled as an Update. The email also contains an action button that can be configured to redirect the recipient directly to the relevant DAO page, ensuring that users can immediately review or follow up on the update.

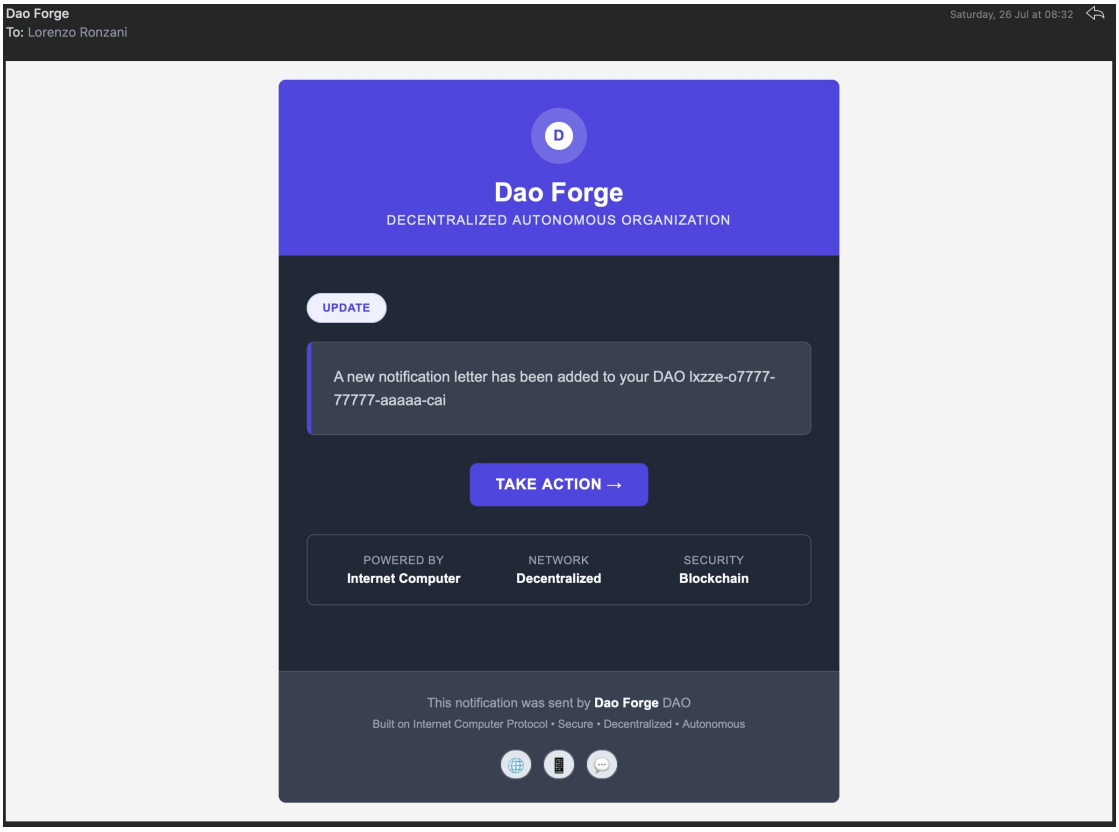


Figure 4.10: Notification email

This mechanism enhances communication between the platform and its users by providing timely notifications about important events, such as the outcome of a vote or the creation of related documents. By integrating this functionality into the voting process, the platform extends transparency and accessibility beyond the interface itself, ensuring that critical updates reach stakeholders even outside the application.

4.7 Summary

The implementation chapter translates the design choices into a functioning system, combining the technological stack, the codebase architecture, and the user-facing platform. The project relies on ICP canisters, Rust, and Typescript to provide decentralized execution, strong type management, and secure access control. Data persistence is handled directly on-chain, ensuring consistency and verifiability of organizational records.

During development, a number of challenges emerge, particularly in configuration management, automatic execution of actions, and handling of local canister identifiers. These issues

are resolved through pragmatic solutions that maintain the integrity of the architecture while simplifying future extensibility. A set of targeted tests validates the correctness of the system and supports iterative improvements.

The platform consolidates these technical foundations into a coherent user experience. Core functionalities such as DAO creation, dashboard management, document and publication handling, and the complete voting lifecycle are integrated within a single environment. Screenshots illustrate how these features are exposed to the user, highlighting the balance between legal compliance requirements and usability. Automated processes, such as the generation of documents and notification emails, reinforce the principle of automation, reducing manual intervention.

The implementation demonstrates the feasibility of managing legally aligned DAOs entirely on the ICP technology. The resulting platform, while still a proof of concept, provides a functional basis that showcases the potential of the proposed architecture and sets the stage for evaluation in Chapter 5.

Chapter 5

Results

The development of the platform for creating and managing DAOs entirely on the ICP was guided by a central question: is it possible to design a fully decentralized, legally compliant, and practically usable DAO platform that operates entirely on-chain? This chapter presents the results of the work, assessing how the implemented platform answers the objectives and hypotheses introduced in the early stages of this thesis. The outcomes are evaluated along three main axes: legal feasibility, technical achievements, and practical applicability. The implementation culminated in a functional PoC deployed on the ICP mainnet [99], enabling users to create and manage organizations, conduct votes, and execute automated on-chain actions within the Swiss legal framework for associations. The results discussed here are based on the technical performance of the system, its alignment with the legal constraints identified in the analysis chapter (Chapter 2), and the feedback gathered from initial real-world interactions.

5.1 Legal feasibility and scope

The results demonstrate that a fully on-chain DAO management platform is feasible within certain legal boundaries. For relatively simple organizational forms, such as Swiss associations, it is possible to implement governance and operational processes in a way that is both legally compliant and technologically sound. This is largely because Swiss associations have fewer bureaucratic requirements compared to more complex entities such as SAGLs or SAs. The decision to focus the first implementation on associations proved advantageous for two reasons:

- From the analysis in Chapter 2, it emerged as the most straightforward legal form

to translate into a decentralized model, with minimal incompatibilities between legal requirements and automated processes.

- Collaborating with the TBTA provided an opportunity to validate the concept in a real-world context, adding significant value to the platform's proof of concept.

While the Swiss association use case fits well with the current capabilities of ICP and the platform's architecture, more complex forms, particularly those requiring KYC procedures, complex treasury management, or jurisdiction-specific audits, remain challenging. These limitations are not due to ICP's technical capacity, but rather to the legal environment's readiness to recognize and accommodate DAOs.

5.2 Technical achievements

One of the most important outcomes is the demonstration that a complete DAO management workflow, from creation, to governance, to execution of actions, can be implemented entirely on-chain using ICP canisters. The platform operates without relying on any centralized infrastructure, including the frontend, which is hosted in a canister alongside the backend logic. This architecture, detailed in Chapter 3 and Chapter 4, delivers the following technical results:

- **Complete decentralization**: All components run as canister smart contracts, ensuring that no part of the system depends on off-chain servers.
- **Automation of governance**: Votes can trigger on-chain actions automatically, with the system storing the canister ID, method name, and arguments during poll creation and executing them after the voting period ends.
- **Type reconstruction in Rust**: The platform successfully reconstructs Candid encoded arguments from string representations at runtime, handling basic types, records (objects), and variants (enums). This was a significant technical challenge due to Rust's strict typing model.
- **Horizontal scalability by design**: ICP's native canister model enables the platform to scale by deploying additional canisters without introducing centralized load balancing mechanisms.
- **Interoperability and extensibility**: The modular design allows additional legal forms or jurisdictions to be integrated without altering unrelated components.

5.3 Automation within legal boundaries

While the platform's technical capabilities allow for full automation of many governance processes, practical application is constrained by legal requirements that still demand human intervention. A clear example is the creation of an association, which under Swiss law requires the completion and physical signing of specific documents. In the implemented system, these documents are generated automatically, pre-filled with the correct data, and stored in the document registry canister for the organization. However, the final step, printing, signing, and mailing, remains manual. This represents the maximum achievable automation for such processes under current regulations. Despite these boundaries, the platform optimizes every possible step within the legal framework. This approach ensures that automation is meaningful and compliant, avoiding the pitfall of creating a technically advanced system that fails to produce legally valid outcomes.

5.4 Comparison with existing solutions

The platform's results can also be understood in the context of existing DAO management tools. A comparative analysis was conducted with three representative platforms: Aragon, Tally, and JuiceBox.

Feature / criterion	**This work**	**Aragon**	**Tally**	**Juicebox**
Backend decentralization	Yes (ICP canisters)	Yes (Ethereum smart contracts)	Yes	Yes
Frontend hosting	On-chain	Off-chain	Off-chain	Off-chain
Legal tailoring	Swiss association model (modular for other jurisdictions or legal forms)	Limited (generic model)	None	None

Table 5.1: Implemented platform vs currently available platforms

Feature / criterion	This work	Aragon	Tally	Juicebox
Automation scope	Creation, Governance and automated action execution	Governance, limited operations	Governance only	Treasury & funding

Table 5.1: (continued)

As shown in Table 5.1, unlike the compared solutions, the platform implemented during this thesis achieves complete decentralization by hosting both backend and frontend on-chain. The legal tailoring to Swiss associations is unique, and the modular architecture opens the way for other jurisdictions and legal forms to be integrated. In terms of automation, the system can progress from DAO creation to voting, to action execution, and even to verifying results, whereas the other platforms typically stop at governance or treasury management.

5.5 User testing and feedback

The implemented PoC is deployed publicly on the ICP mainnet, making it possible for anyone to create and manage organizations, conduct votes, and explore the platform's features. This open availability turned the system into a practical testing ground, where informal feedback was collected through user interactions, direct communication, and GitHub issues. Although a formal usability study was not conducted, the feedback received so far has confirmed the platform's stability and functional correctness. In some cases, user comments also provided insights into potential improvements in user experience, particularly around the onboarding process and the clarity of certain workflows. The PoC's availability also served a strategic purpose, it allowed stakeholders such as TBTA to follow the project's evolution and verify that the implementation aligned with their operational needs.

5.6 Extensibility and future potential

One of the most valuable results is the confirmation that the architecture is modular, flexible, and extensible. The system's design makes it possible to add, without requiring major rewrites to existing components:

- New legal forms (e.g., SAGL, SA),

- Country-specific compliance modules,
- Additional governance mechanisms or voting models.

This adaptability is essential for scaling the platform beyond the Swiss association model and for future-proofing it against changes in technology or regulation. A particularly promising direction is the integration of VCs to automate KYC processes. With the release of VC support in Internet Identity in mid of July 2025, the platform can now rely on a natively supported mechanism for secure and privacy-preserving identity verification. Incorporating this feature would enable a higher degree of automation for operations that are currently constrained by legal identity requirements. In parallel, the adoption of vetKeys would further strengthen the platform's security and confidentiality guarantees by ensuring selective disclosure and privacy-preserving access to sensitive data. Together, these enhancements pave the way for a more compliant, efficient, and user-friendly system.

5.7 Technical challenges overcome

The development process revealed and resolved several technical challenges:

- **Inter-canister action execution**: Managing stored canister IDs, method names, and arguments for dynamic execution required careful handling of types and serialization.
- **Rust type flexibility**: Overcoming Rust's strict typing model to reconstruct complex Candid types from strings at runtime was a non-trivial achievement.
- **Configuration management**: The system supports flexible configuration for different environments, which was particularly important for local development and testing.

These results, discussed in detail in Chapter 4, confirm that the chosen technologies and architectural decisions were appropriate and robust.

5.8 Readiness for real-world adoption

The platform has not yet undergone a full-scale real-world deployment with TBTA or the Municipality of Lugano due to timing and procedural delays. However, the close involvement of TBTA representatives during development ensures that many features already align with the daily operations of a real association. The PoC status allows for immediate experimentation, and the current phase focuses on gathering user feedback, refining workflows, and exploring the integration of additional automation features where legally possible.

5.9 Summary

The results of this work show that it is possible to design and implement a fully decentralized DAO management platform that is both technically sound and legally compliant, at least for certain organizational forms like Swiss associations. The implemented PoC proves the architectural concept, validates the use of ICP as a foundation for such systems, and opens a clear path toward future expansion. While legal boundaries remain a limiting factor for full automation, the platform achieves maximum possible automation under current regulations and demonstrates how legal and technical constraints can be reconciled. The combination of full decentralization, legal tailoring, and modular extensibility positions the platform as a unique contribution to the field, bridging the gap between blockchain-native governance and real-world legal compliance.

Chapter 6

Conclusions

This thesis has explored the end-to-end development of a platform for creating and managing DAOs entirely on the ICP. The goal was ambitious, to combine blockchain-native decentralization with the legal requirements of a real-world jurisdiction, in this case, Switzerland.

The result is a working PoC that runs fully on-chain, with all frontend, backend, and governance logic deployed as ICP canisters. The chosen legal form for the initial implementation is the Swiss association, selected for its lower legal complexity and the feasibility of mapping its requirements into automated, decentralized workflows. This approach not only allows for a realistic proof of concept but also facilitates collaboration with the TBTA as a real-world testing partner.

Throughout the project, emphasis was placed on building a modular, extensible architecture that can be easily adapted to new legal forms, jurisdictions, and governance models without disrupting existing components. As a result, while the current platform is tailored to Swiss associations, it also serves as a flexible foundation for future expansions.

6.1 Key engineering achievements

From a technical standpoint, the project achieves several milestones that are worth highlighting. One of the most significant is the fully decentralized on-chain deployment of all components. Unlike many existing DAO platforms that rely on centralized off-chain hosting for their frontend, this solution ensures decentralization across the entire stack.

Another achievement lies in the modular architecture, where each core function, such as voting, document storage, or discovery, is encapsulated in its own canister. This design

makes it possible to evolve or replace individual modules without affecting the rest of the system, ensuring both maintainability and scalability.

The project also introduces an innovative mechanism for automatic action execution with verifiable results. This feature allows the platform to go beyond simply recording governance decisions: once a vote is passed, the corresponding action can be executed automatically on-chain, and its outcome can be cryptographically verified. This closes the loop from proposal creation to execution and validation, enabling a higher degree of automation and trust compared to many existing DAO platforms, where execution often remains a manual or partially off-chain process.

Legal compliance is not treated as an afterthought. The platform integrates Swiss legal logic directly into its workflows, balancing automation with the constraints of existing law. This is complemented by automated document generation, where required legal forms are pre-filled with DAO data, reducing manual work while respecting manual submission requirements.

Finally, the testing strategy combines unit tests for core business logic, integration tests in a simulated dfx environment, and early-stage user feedback collected through the public PoC. This multifaceted approach ensures that both technical correctness and usability considerations are addressed.

6.2 Limitations and lessons learned

The development process made it clear that the greatest barriers to fully automated DAO management are not technological, but legal. While ICP provides the infrastructure for complete decentralization and secure data handling, many organizational procedures, such as signing incorporation documents or conducting certain treasury operations, still require physical presence or analog methods.

On the technical side, the most notable challenges were related to type handling across languages. The interaction between Rust, Candid interfaces, and ICP's execution model required careful design to ensure interoperability. There were also specific implementation shifts needed for features like automatic action execution and configuration management, which would have been straightforward in more dynamic languages.

From a usability standpoint, the platform's current UI is functional for its intended purpose as a proof of concept, but a more polished user experience would be necessary for a production-ready product aimed at wider adoption.

6.3 Future developments

From a technical perspective, several developments are planned to further enhance the platform:

1. Implementation of tamper proofing for documents will ensure that once a document is added to the platform, it cannot be modified, providing verifiable proof of authenticity. This feature will be particularly important for legal and governance transparency.
2. Integration of vetKeys will allow for privacy-preserving data management, enabling selective disclosure of sensitive information through threshold cryptography. This will complement the platform's current transparency model with robust privacy capabilities.
3. Adoption of VCs within Internet Identity. This will make it possible to automate KYC processes and expand the range of legal operations that can be conducted fully on-chain.
4. Moving from proof of concept to product will require a structured process of security audits, performance optimization, and scalability testing, ensuring that the platform can meet the reliability requirements of real-world deployments.

In parallel to technical improvements, legal and operational developments will focus on testing the platform in real-world environments. Collaborations with TBTA and the Municipality of Lugano will serve as critical validation stages, allowing the system to be evaluated against actual governance processes.

Based on feedback from these trials, identified issues will be addressed, and a market launch strategy will be developed. Long-term goals also include working with government bodies to adapt legal frameworks and enable greater automation within compliant boundaries, ultimately increasing the platform's adoption potential.

6.4 Potential impact

If fully realized, this platform could significantly influence the future of decentralized governance. By integrating blockchain-native automation with jurisdiction-specific legal compliance, it paves the way for organizations that are both autonomous and legally recognized.

In the Swiss context, it could serve as a model for compliant DAOs, opening the door for adoption by non-profits, civic initiatives, and potentially small and medium-sized en-

terprises. Internationally, its modular structure allows adaptation to other legal systems, enabling cross-border collaboration and experimentation with new governance models.

This project began with the question: can a legally aware, fully decentralized DAO platform be built without compromising usability? The answer is yes, within the limits imposed by current laws. The architecture is solid, the technical foundations are in place, and the proof of concept demonstrates both feasibility and value.

The next challenge lies in closing the gap between technological possibility and legal acceptance. As blockchain infrastructure matures and legal frameworks evolve, the opportunities for deeper automation and broader adoption will expand. The current platform is not the endpoint but the starting foundation for future innovation in decentralized organizational systems.

6.5 Acknowledgements

I would like to express my sincere gratitude to those who supported this work. Special thanks to Eng. Giuliano Gremlich and Eng. Roberto Guidi for their guidance and feedback throughout the entire project, from the initial concept to the final implementation. My appreciation also goes to Eng. Giacomo Poretti, contact point for the TBTA, for his time and continuous evaluation of the platform's features. Finally, I am grateful to Eng. Loris Grossi for reviewing the documentation and ensuring clarity and quality from an external perspective.

Bibliography

[1] Decentralized Autonomous Organization, https://en.wikipedia.org/wiki/Decentralized_autonomous_organization. Last accessed August 2025

[2] Internet Computer Protocol, https://internetcomputer.org/. Last accessed August 2025

[3] Web 3.0, https://en.wikipedia.org/wiki/Web3. Last accessed August 2025

[4] Swiss company legal forms, https://www.kmu.admin.ch/kmu/en/home/concrete-know-how/setting-up-sme/les-differentes-formes-juridiques.html. Last accessed August 2025

[5] Verifiable Credential, https://www.w3.org/TR/vc-data-model-2.0/#what-is-a-verifiable-credential. Last accessed August 2025

[6] Know Your Customer, https://en.wikipedia.org/wiki/Know_your_customer. Last accessed August 2025

[7] Ticino Blockchain Technologies Association, https://www.tbta.ch/. Last accessed August 2025

[8] Aragon, https://www.aragon.org/. Last accessed August 2025

[9] Tally, https://www.tally.xyz/. Last accessed August 2025

[10] Juicebox, https://juicebox.money/. Last accessed August 2025

[11] Internet Identity, https://internetcomputer.org/internet-identity. Last accessed August 2025

[12] Swiss Code of Obligations, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en. Last accessed August 2025

[13] Swiss Civil Code, https://www.fedlex.admin.ch/eli/cc/24/233_245_233/en. Last accessed August 2025

[14] Art. 60 CC, https://www.fedlex.admin.ch/eli/cc/24/233_245_233/en#art_60. Last accessed August 2025

[15] Commercial Register, https://www.kmu.admin.ch/kmu/en/home/concrete-know-how/setting-up-sme/starting-business/commercial-register%20.html. Last accessed August 2025

[16] Art. 75a CC, https://www.fedlex.admin.ch/eli/cc/24/233_245_233/en#art_75_a. Last accessed August 2025

[17] Art. 41 CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_41. Last accessed August 2025

[18] Art. 957 CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_957. Last accessed August 2025

[19] Art. 779 CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_779. Last accessed August 2025

[20] Art. 806 CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_806. Last accessed August 2025

[21] Art. 773 CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_773. Last accessed August 2025

[22] Art. 777c CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_777_c. Last accessed August 2025

[23] Art. 784 CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_784. Last accessed August 2025

[24] Art. 800 CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_800. Last accessed August 2025

[25] Articles of association, https://www.kmu.admin.ch/kmu/en/home/concrete-know-how/setting-up-sme/starting-business/choosing-legal-structure/limited-company/articles-of-association.html. Last accessed August 2025

[26] Art. 825 CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_825. Last accessed August 2025

[27] Art. 727 CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_727. Last accessed August 2025

[28] Art. 640 CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_640. Last accessed August 2025

[29] Art. 634 CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_634. Last accessed August 2025

[30] Art. 715 CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_715. Last accessed August 2025

[31] Art. 718 CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_718. Last accessed August 2025

[32] Art. 624 CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_624. Last accessed August 2025

[33] Art. 652 CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_652. Last accessed August 2025

[34] Art. 652a CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_652_a. Last accessed August 2025

[35] Art. 653 CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_653. Last accessed August 2025

[36] Art. 625 CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_625. Last accessed August 2025

[37] Art. 682 CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_682. Last accessed August 2025

[38] Art. 685 CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_685. Last accessed August 2025

[39] Art. 698 CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_698. Last accessed August 2025

[40] Art. 716 CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_716. Last accessed August 2025

[41] Art. 958e CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_958_e. Last accessed August 2025

[42] Environmental, Social and Governance, https://en.wikipedia.org/wiki/Environmental,_social,_and_governance. Last accessed August 2025

[43] Art. 964a CO, https://www.fedlex.admin.ch/eli/cc/27/317_321_377/en#art_964_a. Last accessed August 2025

[44] Ethereum, https://ethereum.org/en/. Last accessed August 2025

[45] Polygon, https://polygon.technology/. Last accessed August 2025

[46] Aragon OSx, https://www.aragon.org/osx. Last accessed August 2025

[47] Aragon App, https://www.aragon.org/aragon-app. Last accessed August 2025

[48] Aragon Voice, https://blog.aragon.org/introducing-aragon-voice/. Last accessed August 2025

[49] Aragon Court, https://legacy-docs.aragon.org/products/aragon-court/aragon-court. Last accessed August 2025

[50] Governance Open Zeppelin, https://docs.openzeppelin.com/contracts/5.x/governance. Last accessed August 2025

[51] Uniswap, https://app.uniswap.org/. Last accessed August 2025

[52] Arbitrum, https://arbitrum.io/. Last accessed August 2025

[53] Eigenlayer, https://app.eigenlayer.xyz/. Last accessed August 2025

[54] Wormhole, https://wormhole.com/. Last accessed August 2025

[55] Snapshot, https://docs.snapshot.box/. Last accessed August 2025

[56] Solana, https://solana.com/. Last accessed August 2025

[57] Ethereum Name System, https://ens.domains/. Last accessed August 2025

[58] ZKsync, https://www.zksync.io/. Last accessed August 2025

[59] ConstitutionDAO, https://www.constitutiondao.com/. Last accessed August 2025

[60] AssangeDAO, https://assangedao.org/. Last accessed August 2025

[61] SharkDAO, https://sharks.wtf/. Last accessed August 2025

[62] Non Fungible Token, https://en.wikipedia.org/wiki/Non-fungible_token. Last accessed August 2025

[63] ERC-20, https://ethereum.org/en/developers/docs/standards/tokens/erc-20/. Last accessed August 2025

[64] Juicebox discord, https://discord.com/invite/wFTh4QnDzk. Last accessed August 2025

[65] GitBook, https://www.gitbook.com/. Last accessed August 2025

[66] Rust, https://www.rust-lang.org/. Last accessed August 2025

[67] Typescript, https://www.typescriptlang.org/. Last accessed August 2025

[68] ICP subnet architecture, https://internetcomputer.org/how-it-works/architecture-of-the-internet-computer/. Last accessed August 2025

[69] Byzantine Fault Tolerant, https://en.wikipedia.org/wiki/Byzantine_fault. Last accessed August 2025

[70] Chain Key Cryptography, https://learn.internetcomputer.org/hc/en-us/articles/34209486239252-Chain-Key-Cryptography. Last accessed August 2025

[71] WebAssembly, https://webassembly.org/. Last accessed August 2025

[72] HTML, https://developer.mozilla.org/en-US/docs/Web/HTML. Last accessed August 2025

[73] CSS, https://developer.mozilla.org/en-US/docs/Web/CSS. Last accessed August 2025

[74] Javascript, https://developer.mozilla.org/en-US/docs/Web/JavaScript. Last accessed August 2025

[75] Amazon Web Services, https://aws.amazon.com/. Last accessed August 2025

[76] IPFS, https://ipfs.tech/. Last accessed August 2025

[77] GitHub pages, https://docs.github.com/en/pages. Last accessed August 2025

[78] Motoko, https://internetcomputer.org/docs/motoko/home. Last accessed August 2025

[79] DFINITY Foundation, https://dfinity.org/. Last accessed August 2025

[80] Cloudflare, https://www.cloudflare.com/. Last accessed August 2025

[81] VetKeys, https://internetcomputer.org/docs/building-apps/network-features/vetkeys/introduction. Last accessed August 2025

[82] Threshold ECDSA, https://internetcomputer.org/docs/building-apps/network-features/signatures/t-ecdsa/. Last accessed August 2025

[83] Chain-Key Bitcoin, https://internetcomputer.org/docs/defi/chain-key-tokens/ckbtc/overview. Last accessed August 2025

[84] Chain-Key Ethereum, https://internetcomputer.org/docs/defi/chain-key-tokens/cketh/overview. Last accessed August 2025

[85] Bitcoin, https://bitcoin.org/en/. Last accessed August 2025

[86] OAuth, https://oauth.net/2/. Last accessed August 2025

[87] Firebase, https://firebase.google.com/. Last accessed August 2025

[88] NNS, https://internetcomputer.org/nns. Last accessed August 2025

[89] Neurons, https://learn.internetcomputer.org/hc/en-us/articles/34084120668692-Neurons. Last accessed August 2025

[90] What Links Identity and VCs Together Across Applications?, https://affinidi.medium.com/what-links-identity-and-vcs-together-across-applications-9523af3884a9. Last accessed August 2025

[91] W3C, https://www.w3.org/. Last accessed August 2025

[92] eIDAS, https://digital-strategy.ec.europa.eu/it/policies/eidas-regulation. Last accessed August 2025

[93] Swiss e-ID, https://www.eid.admin.ch/en. Last accessed August 2025

[94] Verifiable Legal Entity Identifier, https://vlei.com/. Last accessed August 2025

[95] DAO Suisse, https://www.daosuisse.com/. Last accessed August 2025

[96] Admin.ch, https://www.admin.ch/gov/en/start.html. Last accessed August 2025

[97] Agile manifesto, https://agilemanifesto.org/principles.html. Last accessed August 2025

[98] DAO Forge codebase, https://github.com/lorenzoronzani/dao-forge. Last accessed August 2025

[99] DAO Forge platform, https://yqq6b-aqaaa-aaaaj-qnr2q-cai.icp0.io/. Last accessed August 2025

[100] Swiss Official Gazette of Commerce, https://www.seco.admin.ch/seco/en/home/Publikationen_Dienstleistungen/Publikationen_und_Formulare/handelsamtsblatt.html. Last accessed August 2025

[101] MultiSig controller, https://internetcomputer.org/docs/building-apps/canister-management/control#multisig. Last accessed August 2025

[102] Candid, https://internetcomputer.org/docs/building-apps/interact-with-canisters/candid/candid-concepts. Last accessed August 2025

[103] React, https://react.dev/. Last accessed August 2025

[104] Shadcn, https://ui.shadcn.com/. Last accessed August 2025

[105] Tailwind CSS, https://tailwindcss.com/. Last accessed August 2025

[106] Dfx, https://internetcomputer.org/docs/tutorials/developer-liftoff/level-1/1.3-intro-dfx. Last accessed August 2025

[107] Cargo, https://doc.rust-lang.org/cargo/. Last accessed August 2025

[108] Npm, https://www.npmjs.com/. Last accessed August 2025

[109] GitHub, https://github.com/. Last accessed August 2025

[110] Courier, https://app.courier.com/. Last accessed August 2025

[111] CycleOps, https://cycleops.dev/. Last accessed August 2025

[112] Data Transfer Object, https://en.wikipedia.org/wiki/Data_transfer_object. Last accessed August 2025

[113] Tanstack react-query, https://tanstack.com/query/latest/docs/framework/react/overview. Last accessed August 2025

[114] Stable structures, https://internetcomputer.org/docs/building-apps/developer-tools/cdks/rust/stable-structures. Last accessed August 2025

[115] Random Access Memory, https://en.wikipedia.org/wiki/Random-access_memory. Last accessed August 2025

[116] Chicken and egg, https://en.wikipedia.org/wiki/Chicken_or_the_egg. Last accessed August 2025

[117] Management canister, https://internetcomputer.org/docs/references/system-canisters/management-canister. Last accessed August 2025