Electronic Communications of the EASST
Volume 84 Year 2025

12th International Symposium on Leveraging Applications of Formal Methods, Verification and Validation / 2nd AISoLA - Doctoral Symposium, 2024

Edited by: Sven Jörges, Salim Saay, Steven Smyth

DAOMod: A Modeling Method for Decentralized Autonomous Organization Development

Sowelu Avanzo, Daniele Pautasso, Irene Domenicale, Alex Norta, Julio Linares, Claudio Schifanella, Marie Hattingh, Daniel Sami Mitwalli

DOI: 10.14279/eceasst.v84.2675

Electronic Communications of the EASST (https://eceasst.org).
Published by **Berlin Universities Publishing**
(https://www.berlin-universities-publishing.de/)

DAOMod: A Modeling Method for Decentralized Autonomous Organization Development

Sowelu Avanzo[1], Daniele Pautasso [1], Irene Domenicale [1], Alex Norta [2 3 5], Julio Linares[4], Claudio Schifanella[1], Marie Hattingh[5], Daniel Sami Mitwalli[6]

University of Torino, Torino, Italy (e-mail: soweluelios.avanzo@unito.it; daniele.pautasso@unito.it; irene.domenicale@unito.it; claudio.schifanella@unito.it)[1], Tallinn University, Tallinn, Estonia (e-mail: alex.norta.phd@ieee.org)[2], Dymaxion OÜ, Tallinn, Estonia [3], Circles Coop eG (e-mail: julio@joincircles.net)[4], University of Pretoria, Pretoria, South Africa (e-mail: marie.hattingh@up.ac.za)[5], University of Limerick, Limerick, Ireland (e-mail: mitwalli.daniel@ul.ie) [6]

Abstract: Decentralized Autonomous Organizations (DAOs) are a type of Decentralized Applications (DApps) that utilize smart contracts to support governance processes. To achieve a high degree of utility of the system, stakeholders need to identify a suitable organizational structure in the early stages of design. While Model-Driven Development (MDD) methods are established for DApp and smart contract design, they lack specialization for modeling the organizational structures of DAOs. To address this gap, we propose a modeling language and a method which support crucial DAO design and development phases. The method is evaluated through an *in vivo* case study. Unlike existing solutions, comprehensive stepwise guidance is provided by our method for both technical and non-technical stakeholders involved in DAO development from the initial stages of the project.

Keywords: Blockchain, DAO, Governance, Model-Driven Design

1 Introduction

Decentralized applications (DApps) are systems that incorporate smart contracts as a component of their architecture. These are computer programs running in a blockchain-based environment. Decentralized Autonomous Organizations (DAOs) represent a special class of DApps that use smart contracts to enable decentralized governance processes [RJK23]. In recent years a wide range of real-world applications of DAO have been developed in most diverse fields [BT24]. Particularly interesting are novel applications of DAOs that promise to catalyze positive societal impact by supporting economic models oriented to the commons [VJ23].

However, the development of DAOs involves a significant degree of complexity, and recent attention has been paid to the main issues that limit the utility of DAOs [ANS23]. First, various empirical studies, including [SKP+23, WYS+22] reveal the high degree of centralization in decision-making power. Second, relevant issues for DAOs are the lack of scalability in voting systems and voter apathy, that is, the lack of participation in voting, as [WYS+22] discusses. Third, the study in [WYS+22] highlights the issue of disunity in the communities of DAO members, which restricts the efficiency of decision making.

These issues relate to the complexity of designing DAOs with organizational structures to ensure that all members contribute to specific governance areas and that their actions align with the goals of the system. DAO governance demands novel requirements compared to traditional organizations [QDL+22]. In DAOs, roles and permissions are dynamically assigned based on tokenization, allowing for a more flexible and decentralized organizational form, as discussed in [QDL+22]. Furthermore, DAO communities can in some cases define the organization structure itself by upgrading the smart contracts of the DAO [QDL+22, BFJ23]. Therefore, it is essential to correctly assign roles and permissions to avoid unintended power increases [DMH+21].

Recently, various solutions supporting DAO development were proposed. Several platforms enable the deployment of DAOs as a service based on a no-code approach, also allowing users to customize DAOs without the need for coding [BT24]. Users can do so by choosing among a set of predetermined functionalities those that best meet the requirements of the system. However, this approach results in users having to choose among a large number of platform-specific features. The decision model in [BFJ23] reduces this complexity. Still, constraining factors for this decision model are the dependency on platform availability and the lack of interoperability of existing solutions [BFJ23, ANS23]. On the other hand, platform-independent enterprise modeling approaches, such as Archimate[1] or Organization Models [ST] lack suitability for decentralized governance. The modeling of decentralized governance processes is only supported by some languages [KG21, DINM23]. Moreover, the study in [QDL+22] comprises an extensive conceptual model of DAO architecture, and another study [VP24] includes a Web3 DAO Ontology that defines a detailed model of DAO voting systems.

However, although some of the prior works offer stepwise guidance or DAO models in isolation, none of the approaches for decentralized governance design integrate a suitable visual modeling notation and methodological guidelines in a cohesive framework. This limits their applicability and practical utility, particularly for non-technical users. Moreover, they lack suitability for modeling the permission management logic of DAOs, which shapes their organizational structure.

Consequently, we address this gap in the state-of-the-art with a method to develop DAOs with suitable organizational structures. We do so by responding to the following research question: *How to develop a method for modeling DAOs with suitable organizational structures?* To establish a separation of concerns, we address the following sub-RQs.

- **RQ1**: *What is the ontology incorporating suitability properties of organizational structures of DAOs?*
- **RQ2**: *What is a suitable modeling language for specifying platform-independent design of DAOs?*
- **RQ3**: *What is the stepwise method to model DAOs adopting the modeling language developed?*

We answer the research questions outlined by making the contributions described below. We first generate categories of reference requirements for DAOs based on existing specifications and literature. This allows us to develop an ontology, based on such concepts, that incorporates

[1] https://www.opengroup.org/archimate-forum/archimate-overview

suitable classes and properties to model DAOs (suitability properties), hence responding to RQ1. This formal representation of the domain constitutes the ontological metamodel of the modeling language.

Subsequently, based on the suitability properties of the DAOs modeled in the previous step, we extended the graphical notation and syntax of an agent-oriented modeling language [ST] representing the Organization model diagrams to incorporate concepts relevant to the DAO design. These form the Decentralized Organization diagrams, part of the DAOMod modeling language (RQ2).

We then present the step-by-step guidelines that complete the framework (RQ3) by allowing one to design DAOs using the DAOMod modeling language. The method steps facilitate the selection of suitable design features based on non-functional requirements. Thus, by means of these contributions, we accomplish the research objective of proposing a method that enables the platform-independent design of DAOs. Finally, an evaluation is performed following a case study approach, described in [RHRR12] to ensure relevance to concrete organizational problems.

The remainder of the paper is structured as follows. Section 2 discusses the case study, methodology, related works, and background literature. In Section 3, we respond to RQ1 by presenting a model of reference requirements of DAOs and their implementation in the DAOMod ontology. In Section 4, we respond to RQ2 by presenting the notation and syntax of the DAOMod modeling language. Subsequently, in Section 5, we present the steps of the DAOMod modeling method (RQ3). The evaluation is presented in Section 6, and Section 7 provides a discussion of the results and concludes this paper by summarizing the findings, discussing limitations, and future work.

2 Preliminaries

In this section, we first discuss the research methodology (Section 2.1). Subsequently (Section 2.2), we discuss the background literature necessary for the reader to understand the contributions of this article. In Section 2.3, we present related work.

2.1 Methodology

This study adopts the design science research methodology (DSR), as defined in [HMPR08]. DSR is a structured approach that aims to generate new design theories through an iterative process of justification, development, and evaluation of information system (IS) artifacts that can be methods, models, and constructs [HMPR08].

DSR is especially suited to the context of DAO development, because of the inherently sociotechnical nature of DAOs and their rapidly evolving technology. These aspects make the DSR framework indispensable to address the practical challenges unique to real-world DAO development projects and develop new design theories for this class of systems based on empirical evidence.

To ensure *relevance*, IS artifacts are designed to address unresolved problems within organizational contexts. Moreover, *rigor* is maintained by grounding their development in validated theoretical foundations and methodologies, which form the study's knowledge base. In this research, four DSR artifacts have been developed: a reference model outlining the functional and non-functional requirements of DAOs (RQ1), an ontology that formalizes this model, which is DAOMod, a modeling language designed for specifying DAOs (RQ2), and a method detailing the steps for specifying DAOMod models (RQ3).

The case study research strategy, defined in [RHRR12], is adopted to assess the usefulness of DAOMod and to ensure the relevance of the artifacts developed. A *case study* analysis [RHRR12] is carried out, involving a DAO development project, which relates to the CommonsHood system in Italy, presented in [VAC+, VAB+23].

We conduct the case study evaluation based on use case *demonstration* and subsequent *usefulness evaluation* through semi-structured interviews with the case study participants. In the former activity, we model DAOs to address concrete problems faced by the systems and organizations on which the case is focused. In the latter case, we interview members of the target organization about the usefulness of the method adopted.

The proposed modeling method is useful for specifying DAO requirements and is widely applicable. We select a DAO development case that faces particular software engineering challenges for which existing solutions are not adequate. In particular, the reliance on real-world socio-economic interactions that take place in local communities of users, discussed, for instance, in [VAC+].

To evaluate the proposed modeling method, a case study was conducted following a structured process. First, a **requirement elicitation** phase involved preliminary meetings with stakeholders and members of the CommonsHood project to gather the necessary specifications for system extensions. Next, a **DAO use-case demonstration** was performed, applying the method to specify the DAO systems according to the project requirements, with active participation of CommonsHood members in model development and model validation. Finally, the **DAOMod utility evaluation** was carried out through semi-structured interviews with case study participants. The evaluation focused on the semantic and pragmatic utility of the method, using the questions reported in [APD+25], which address the semantic and pragmatic utility of the modeling language. The questions are adapted to our case from the study in [DINM23]. Participants were introduced to the method and modeling syntax before assessing models related to their organization's use case. Thematic analysis was used to rigorously interpret the interview data, providing insight into the strengths and areas of improvement of the method.

2.1.1 CommonsHood

The CommonsHood project uses a blockchain-based wallet app to support urban communities through local token circulation. The project is agnostic to the economic model adopted by the local communities, as it allows each participant to potentially deploy their own token type, without the need for coding, through a graphical interface. Common use cases for tokens generated through the platform are digitized coupons, event tickets, local currencies, and NFTs that support the traceability of object ownership [VAB+23]. CommonsHood currently also supports the deployment of DAOs as a service, which is used by associations for simple governance processes. A crucial characteristic of CommonsHood is the integration of DAOs with FirstLife [VAB+23]. The latter is a social network characterized by the focus on the local dimension, which leverages geo-referenced posts and entities to facilitate neighborhood-level economies by connecting local resources with community needs

This use case concerns in particular the extension of the capabilities of CommonsHood DAOs to facilitate the management of an urban garden and other collective spaces as a commons by the citizens of Turin, Italy[2]. The urban garden and the surrounding spaces are located on the campus of the School of Management and Economics of the city. A network of associations bootstrapped the project to re-purpose the spaces and allow citizens to organize outdoors activities and cultivate the land plot. CommonsHood is currently used to support tokenized crowdfunding campaigns by means of the *Crowdsale* and *Exchange* smart contracts[3]. Furthermore, citizens can also rent tools for gardening and other resources whose ownership is tracked by means of NFTs.

2.2 Background

The background literature necessary for the reader to understand the contributions of the article is presented below. This discusses DAO architectures and the original notation of the modeling languages that we extend in this study to incorporate the suitability properties of DAOs ([ST, Udo22]).

2.2.1 Reference model of DAO architectures

The theoretical model proposed in the study [QDL+22] divides the DAO architectures into five layers. Three of these are specific to DAO systems and characterize them with respect to the perspectives of *organization*, *coordination*, and *execution*. The Organization layer establishes the governance structure of the DAO, aligning it with its long-term objectives. This study focuses primarily on this layer, while future work will integrate the Coordination and Execution layers, which will address decision-making protocols and token economy mechanisms.

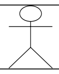
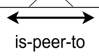
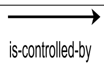
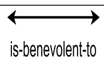
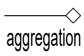
2.2.2 Organization Models

We present here Organization Models, whose notation is defined in Table 1. *Roles* embody distinct functions and responsibilities in an organization. They are represented by the actor icon used, for example, in UML diagrams, and a textual description of the role is included. The relationships used in organizational model diagrams include *peer collaboration* (*is-peer-to*), *control relations* (is-controlled-by), *benevolence relations* (is-benevolent-to), and role *aggregation relations* (*aggregate*), representing the relation between more general and more specialized roles. Directed association links are used to represent all such relations, and a description of the type of relation is included. This notation is suitable for representing simple governance structures in organizations, but it lacks suitability for the complex decentralized governance structures of DAOs.

[2] https://crowdusg.net/commonshood-in-the-garden-social-economies-civic-blockchain-and-urban-gardening-in-turin/

[3] https://gitlab.di.unito.it/commonshood/smart-contracts

Table 1: Organization model diagram notation.

Element	Notation	Description
Role		Roles specify the functions performed by agents in an organization.
is-peer-to	 is-peer-to	The *is-peer-to* relation indicates that two roles are at the same level of the organization hierarchy.
is-controlled-by	 is-controlled-by	The *is-controlled-by* relation indicates the subordination of a role to a superior.
is-benevolent-to	 is-benevolent-to	The *is-benevolent-to* relation indicates a mutually beneficial relationship between roles.
aggregation	 aggregation	The *aggregation* relation indicates that one role is the sub-role or specialization of another.

2.3 Related Work

The study in [PJBO21] proposes a method focused on the design of the governance of blockchain systems. Although DAOs can be adopted to manage the governance of a blockchain network, they are applicable to a wider range of use cases[BT24]. This limits the suitability of the method for the design and development of DAOs. The theoretical model of DAO architectures described in [QDL+22] is crucial to the DAO domain, as it consolidates knowledge on this class of systems, but lacks specific development support. Several ontologies and modeling languages provide a formal representation of aspects that are relevant to DAO development. However, these fail to integrate different perspectives on these systems in a holistic model focused on the design of decentralized organizational structures. For example, the ontology in [BLSA19] models is described in detail by role-based access control (RBAC) enabled by smart contracts in the context of smart buildings. RBAC is relevant to role assignment, permission management, and resource allocation in DAOs. Still, the ontology lacks general applicability to the comprehensive domain of DAOs. Other studies that specifically address RBAC in DAOs, such as those in [CLB+22, MC23], focus on platform-dependent design.

The DECENT ontology and language, presented in [KG21], models decentralized governance in the context of the renewable energy sector and provides a conceptual model of rules and policies. The Smart Legal Contract Markup Language (SLCML) and its ontology [DINM23] allow users to model legally relevant smart contracts and cross-organizational collaborations between DAOs, while lacking suitability for modeling the internal governance processes and functions of DAOs. The Web3 DAO Ontology [VP24] models the domain of decision-making systems associated with DAOs. The ontology includes a comprehensive model of smart contracts and off-chain functions that implement these systems. However, neither of the three ontologies mentioned nor the respective languages models the membership and permission handling logic, crucial to the concrete implementation of organizational structures of DAOs. Furthermore, these approaches lack stepwise guidance and visual notation, which aspects limit the usability of the mentioned approaches.

3 Suitability Properties of DAOs

In this section, we address RQ1 by presenting the ontology that models the Organization Layer of DAO architectures, which captures the worldview underlying the modeling method. Firstly, we introduce the conceptual model that is grounded in the functional and non-functional requirements of DAOs. Secondly, in the subsequent Section (3.2), we present the implementation of the ontology derived from the conceptual model.

3.1 Organization Layer Model

As described in [QDL+22], the Organization layer of DAOs focuses on the organizational form of the system and its governance structure. These features are influenced by the on-chain permission management logic, since this shapes the power dynamics among DAO members and is critical to DAO operations [DMH+21]. We provide a conceptual model of this architectural layer in Figure 1. In our model, the assignment of roles and permissions is enabled by a *permission manager* construct. This serves to specify which *roles* have the authority to grant a set of *permissions* or delegate power to other *roles* [MC23, ZW22, BFJ23]. Furthermore, roles can be assigned to autonomous or human agents, and, as discussed in [ZW22], both peer-to-peer and hierarchical governance structures are in place in current DAO implementations. The *permission_manager* class associates a set of *permission* elements with the appropriate *roles* by means of the *implements* relationship. Each *permission_manager* is *included_in* exactly one DAO, for which it implements the access-control policy. Our conceptualization is also based on the model introduced in the DECENT ontology [KG21], in which each *mechanism* enforces a specific *rule* in the domain of decentralized governance, and *policies consist_of* multiple *mechanisms*. In our model, the *permission_manager* construct consists of two main *mechanisms*: *permission_assignment* and *role_assignment*. The former mechanism allows the granting and revocation of specified *permissions* and determines which permissions can be revoked or granted by higher roles in a hierarchical organizational structure [ZW22, BFJ23]. The latter defines which roles can be assigned or revoked by other roles.

Different approaches exist in current DAO implementations to assign membership and roles. Among these, the study in [ZW22] mentions the *purchase* or *staking* of governance tokens. Other mentioned methods involve *election* of members or *invitation* to participate in specific DAO committees. In our model, these are specified by the *assignment_method* attribute. As discussed in [CLB+22] DAOs can also support hierarchical inheritance of permissions. This aspect of access control enables the creation of complex hierarchical structures, and is specified by means of the *hierarchical_inheritance* attribute, assigned to the *permission_manager* class. This Boolean attribute is set to true if the hierarchical inheritance is enabled in the DAO modeled, and false otherwise.

Agents, which are an abstraction that models both human actors and autonomous computer programs (software agents)[ST], can play roles within DAOs and may also take part in *committees* [MC23]. *Committees* represent communities or sub-communities of DAO members involved in decision-making. In the proposed model, both *DAOs* and *committees* can themselves *play* roles, as they extend the *organizational_unit* class. This characteristic is supported by work

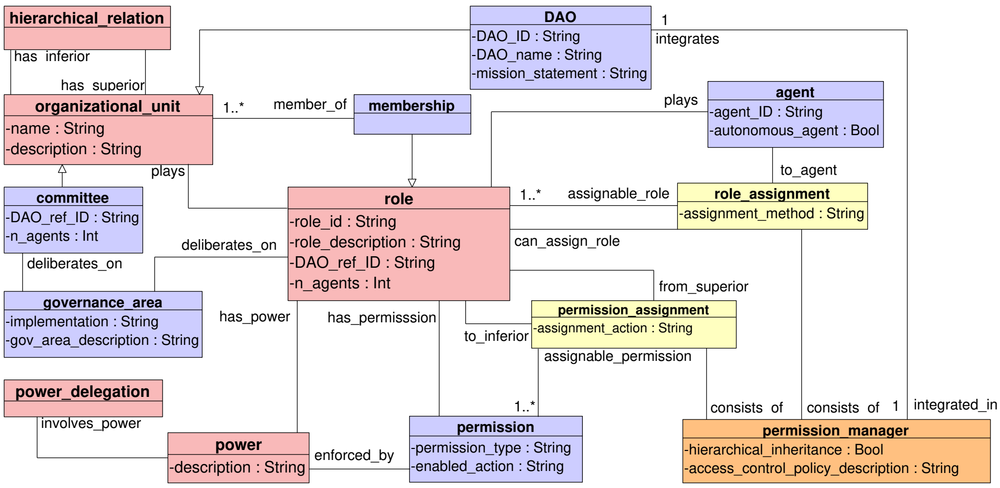


Figure 1: Organization tier of the DAOMod Ontology. Classes colored in red are also present in the SLCML ontology [DINM23], classes in yellow extend the *Mechanism* class and the one in orange extends *Policy* in DECENT [KG21].

highlighting the importance of autonomous execution of proposals in DAOs [SKP+23][4].

The Organization layer also shapes the definition of *governance areas* and their assignment to particular *committees* or *roles*, who are responsible for deliberating on a set of decisions [QDL+22]. To satisfy scalability demands, DAOs, certain governance areas do not utilize *on-chain* execution (*governance_area_type*) for the implementation of the decisions made. Beyond supporting smart contracts, DAOs also depend on *off-chain* and *hybrid* governance mechanisms for areas with lower security requirements [ZAL+22, WYS+22, QDL+22].

We classify *permissions* into three types according to the expected impact on the functionality of the system (*permission_type*). This classification is supported by [ZAL+22, QDL+22, BFJ23]. As highlighted in [QDL+22, BFJ23], a key characteristic of DAOs is the capability to handle the upgradability of smart contracts of the DAO itself. Actions that modify access control policies are considered critical to safety as they can modify the organizational structure of the DAO. These are associated with the *structural permission_type*.

The study in [ZAL+22] distinguishes between *strategic* and *operational* decision-making in DAOs. We adopt this distinction in our classification. *Permissions* are classified as *strategic* if they influence the policies and mechanisms that manage the token economy of the DAO or its decision-making processes, but do not alter the structure of the organization or update the primary smart contracts of the system. For example, the power to issue a specific type of token, transact tokens stored in the DAO's treasury, participate in voting or make proposals is classified as *strategic* due to the remarkable impact these actions have on the coordination of the DAO. In contrast, *permissions* exerting less influence on DAO operations are considered as *operational*. These typically provide access to resources or enable task execution. In Table 2 we introduce a set of reference non-functional requirements (NFRs) for the organizational structures of DAOs, derived from existing literature. NFRs serve as criteria for evaluating the utility [Udo22] of a

[4] https://aragon.org/agent

Table 2: Reference Non-Functional Requirements of DAO Governance Structures

NFR	Description
Decentralized at the infrastructure level	Is based on a (permissionless) blockchain [RJK23].
Decentralized at the governance level	Distributes control across diverse committees, roles, and agents to reduce risks associated with centralization [QDL+22, RJK23].
Secure	Ensures that only authorized parties can deliberate and act on behalf of the DAO [DMH+21].
Scalable	Manages the increase in the number of users, roles and interactions effectively [BFJ23, Udo22].
Upgradable	Allows for seamless integration of new features and improvements [BFJ23, RJK23].

system. The study in [RJK23] highlights the importance of decentralization in terms of *infrastructure* and *governance* in DAOs. Furthermore, it is crucial that the organizational structure of DAOs is defined to be *secure* from privilege increases and other governance-related threats, as highlighted in [DMH+21]. Furthermore, the ability of the governance structure to expand as the DAO grows being (*scalability*), is essential for this architectural layer. Finally, since the governance structures of DAOs may require frequent upgrades to adapt to evolving necessities of the community, *upgradeability* is a relevant NFR.

3.2 DAOMod Ontology Implementation

In this section, we present the implementation of the DAOMod ontology, which models classes, object, and data properties corresponding to the concepts shown in Figures 1 in Web Ontology Language (OWL) 2.0 [5]. As we aim to achieve extensibility of the proposed approach, we integrate and extend the DECENT and SLCML ontologies [DINM23, KG21]. The newly added classes and properties enable users to define governance structures that also benefit from the perspectives contained in SCLML and DECENT. Since, due to lack of space, we cannot provide a complete discussion of the DECENT and SLCML ontologies, we refer the reader to [DINM23] and [KG21] for more details.

We first translated the semi-formal representation of the DECENT ontology [KG21] into OWL 2.0 for compatibility with SLCML [DINM23]. Subsequently, we merge DECENT into SLCML.

We then define additional classes, new object properties, and new data properties, corresponding to the constructs shown in Figure 1 that were not initially included in the knowledge base. Furthermore, we modify the domain and range restrictions of the *consists_of*, *plays* and *implements* to allow users to define relations among newly included constructs.

The DAOMod Ontology is part of a broader study presented in [APD+25], which develops all the layers mentioned in Section 2.2.1, including the organizational, coordination and execution layers. The ontology follows a tiered structure, where each tier corresponds to one of these three architectural layers. It serves as a framework for modeling DAOs by defining individuals that

[5] https://www.w3.org/TR/owl2-overview/

represent the desired design features.

The present work focuses on the most abstract tier, concerning the *Organization* perspective, core aspects of the DAO are defined, including its name and *mission_statement*, defining its long-term goal. In addition, design decisions are taken that inform the organizational structure by defining *roles*, *permissions*, *committees* that the DAO should implement, as well as the desired permission and role assignment logic and control relations between the roles.

We utilize the Protégé editor[6] to develop the ontology. Moreover, the ontology is checked for soundness using the HermiT reasoner[7]. HermiT is a Protegè plugin designed to perform consistency checks on OWL ontologies. This serves to determine whether an ontology is logically consistent and to prevent undesirable inferences from occurring. The complete implementation of the ontology can be found on GitHub[8]. The significant extension of the ontology underscores the need to define a simplified graphical syntax that captures crucial aspects of DAO design to facilitate modeling, which is the subject of the subsequent section.

4 DAOMod Syntax

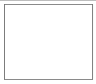
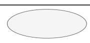
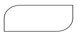
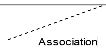
Visual modeling languages are a crucial component of agent-oriented methods that provide the foundation for structured design processes. In this Section, we present the modeling language for DAO development that is part of the DAOMod method, which extends the basic syntax of Organization Models (Table 1) to facilitate the specification of the Organization Layer of DAOs. This is crucial for defining the permission-assignment logic of the DAO and, therefore, achieving focused deliberation and execution.

We first provide a description in Table 3 of the extended graphical notation of Decentralized Organization Diagrams. Secondly, the syntax of Decentralized Organization Models is formalized in Figure 2.

The Decentralized Organization models maintain similar notations for the *Role* element, and for *is-controlled-by* and *aggregation* relations that are present in the original syntax (Table 1). In addition to those, the *Governance Area*, *Committee*, *Permission* and *DAO* elements are included. The DAO element specifies the DAOs being modeled in the diagram, symbolized by a white square. DAO elements contain roles and committees that indicate the roles played by individual agents and collective decision-making protocols involved in the governance of that particular DAO. Furthermore, the *association* relation is used to connect *Governance Area* elements, symbolized by green rectangles, with *roles* and *committees*. By this relation, the long-term goals defined in the Governance Area elements are assigned to these entities by the DAO. Metadata concerning the implementation of such governance areas are provided by means of the *implementation* attribute, which can assume the values of *on-chain*,*off-chain* or *hybrid*, depending on the execution type required for the tasks the governance area is concerned with. *Permission* elements refine such goals by associating to each role and committee the specific actions they are tasked with and authorized to perform within the DAO. Permissions are represented by rectangles with rounded edges in the upper left and lower right corners. The control relation

[6] https://protege.stanford.edu/
[7] http://www.hermit-reasoner.com/
[8] https://github.com/SoweluAvanzo/DAOMod

Table 3: Decentralized Organization Model Elements and Properties

Elements	Notation	Description	Properties
DAO		DAO system involving roles, committees, governance areas, and permissions included in the square.	DAO_ID; DAO_name; *mission_statement*
Committee		Organized group of agents deliberating on a set of governance_areas.	committee_ID; committee_description; n_agent_min; n_agent_max; appointment_method
Governance Area		Domain of interest under the consideration of a given *committee* or *role* in a DAO.	gov_area_ID; gov_area_description; implementation
Permission		Authorization to perform a given action enabled by the organization.	allowed_action; permission_type
Association		Assignment of a *governance_area* or *permission* to a committee or a role.	source_ID: role; committee; target_ID: permission; governance_area
Federation		Indicates participation in a target committee by a role or committee.	source_ID: role; committee; target_ID: committee

(*is-controlled-by*) is also included in the extended syntax. In the DAO domain, this can connect *roles* and *committees* to other *roles* or *committee* elements. The relation indicates the ability of the target role to assign or revoke the source role to agents. Furthermore, the control relation can also indicate the capability of the target role to delegate their own permissions to agents playing the source role or to revoke the permissions of the source role. The *aggregation* relation that was present in the original notation maintains its meaning of specifying the relation between a *role* or a *committee* and its and its specialization. In addition, a *federates-into* relation is defined (Table 3) to indicate the participation of a *role* or *committee* in a *committee*. This specifies the right of the source role or committee to express their opinion on matters decided by the target community of DAO members. To avoid redundancy, Table 3 only shows the newly included elements, with respective properties. The *is-peer-to* and *is-benevolent-to* relations are removed to avoid excessive complexity in the extended syntax. The former can be specified as the absence of control relations; the latter is not relevant to the implementation of access control policies in DAOs.

In the following, the Decentralized Organization Diagram syntax is formalized by means of a grammar-like notation. We write $\mathbb{N}$ for the set of natural numbers and $\mathbb{B}$ for the set of booleans. We denote a (possibly empty) list of elements using square brackets; the writing $[E]_n$ is short for a list of $n \in \mathbb{N}$ elements. Moreover, we assume pairwise disjoint sets of identifiers, one for each element type: these sets are ranged over by ID_X (for $X \in \{Diag, DAO, R, C, P, G\}$), and are such that $ID_X \cap ID_Y = \emptyset$ whenever $X \neq Y$. Given a diagram element e, the writing $e \in X$ means that e

Diagram	::=	$(id, diag\_name, [DAO]_n)$	$id \in ID_{Diag}, n \geq 1$
DAO	::=	$(id, dao\_name, cb\_trans, [R]_{n_1}, [C]_{n_2}, [P]_{n_3}, [G]_{n_4})$	$id \in ID_{DAO}, n_i \in \mathbb{N}$
R	::=	$(id, r\_name, aggr\_lvl, fed\_lvl, agent\_type, assign\_method, n\_ag\_min,$ $n\_ag\_max, ControlledBy, AssociatedTo, AggregatesInto_R, FederatesInto)$	$id \in ID_R$
agent_type	::=	*human* | *autonomous*	
C	::=	$(id, c\_name, aggr\_lvl, fed\_lvl, appoint\_method, n\_ag\_min, n\_ag\_max,$ $ControlledBy, AssociatedTo, AggregatesInto_C, FederatesInto)$	$id \in ID_C$
P	::=	$(id, allowed\_action, permission\_type)$	$id \in ID_P$
permission_type	::=	*structural* | *strategic* | *operational*	
G	::=	$(id, gov\_area\_description, implementation)$	$id \in ID_G$
implementation	::=	*on_chain* | *hybrid* | *off_chain*	

Figure 2: DAOMod Decentralized Organization Diagram grammar.

is of type X. By extension, $e \in X_1 \cup \cdots \cup X_n$ means that e has type X_i for some $1 \leq i \leq n$.

With respect to Figure 2, we impose the following constraints.

- $aggr\_lvl, fed\_lvl \in \mathbb{N}$ are natural numbers specifying the hierarchic level of a role (resp. committee) element in terms of the *aggregates-into* and *federates-into* relations. This field is used to guarantee the correctness of the relations, formalized in the next point.

- *ControlledBy*, *AssociatedTo*, $AggregatesInto_R$, $AggregatesInto_C$ and *FederatesInto* are sets of identifiers, used to express the relations between elements of the model: each role (respectively, committee) element specifies the entities it is related to by means of a suitable set of identifiers, as described below.

  – Both roles and committees can be controlled by other roles and/or committees, that is, $ControlledBy \subseteq ID_R \cup ID_C$. The boolean field $cb\_trans \in \mathbb{B}$ of any given DAO indicates whether the *controlled-by* relation is to be understood as transitive or not. Note that the directed graph induced by the *controlled-by* relation is not required to be acyclic; thus, it is possible to specify circular dependencies. Self-loops in this context specify roles or committees that can alter their own permissions or assignment logic.

  – Both roles and committees can be associated to permissions and/or governance areas, that is, $AssociatedTo \subseteq ID_P \cup ID_G$.

  – Roles and committees differ in the way they aggregate into other entities. On the one hand, roles can aggregate into different roles, that is, $AggregatesInto_R \subseteq ID_R$, indicating that the role is a specialization of the role it aggregates into. On the other hand, committees only aggregate into other committees, that is, $AggregatesInto_C \subseteq ID_C$, indicating the committee is a sub-committee of the one it aggregates into. Moreover, the *aggregates-into* relation must be consistent with the hierarchic level specified by the various roles and committees. Formally, given two distinct roles r_1 and r_2 of aggregation levels $r_1.aggr\_lvl$ and $r_2.aggr\_lvl$ respectively, the role r_1 can aggregate into

r_2 only if $r_1.aggr\_lvl < r_2.aggr\_lvl$. An analogous constraint exists between committees, so that a committee c_1 of level $c_1.aggr\_lvl$ can aggregate into a committee c_2 of level $c_2.aggr\_lvl$ only if $c_1.aggr\_lvl < c_2.aggr\_lvl$.

 – Given a role or committee element e and a distinct committee c, of federation levels $e.fed\_lvl$ and $c.fed\_lvl$ respectively, e can federate into (that is, become member of) c only if $e.fed\_lvl < c.fed\_lvl$.

- $n\_ag\_min, n\_ag\_max \in \mathbb{N}$ are natural numbers such that $n\_ag\_min \leq n\_ag\_max$. These two fields specify, respectively, the minimum and maximum number of agents that can be member of a given committee, or that can be assigned a given role.

- All the remaining fields, such as *name*, *assign_method* and *appoint_method*, are strings providing additional information about a given instance. In particular, *assign_method* and *appoint_method* refer to the methods used to assign a given role and to enable participation in a committee.

The XML schema available on the GitHub repository[9] is used to define a structure of XML files that store information on the Decentralized Organization diagram. We also defined Xpath queries that enforce the conditions formalized above which concern relations among elements, and cannot be enforced directly by means of XSD.

We enforce the conditions through a Python script that first validates the conformance to the grammar of the XML files passed as input, using the Lxml library[10] for the XML parsing and the XML schema for validation[11]. Subsequently, for each condition imposed, the script compares the subsets of IDs retrieved with the Xpath queries.

5 DAO Modeling Method

In this section, we provide a description of the stylistic guidelines for using the modeling language presented to model DAOs with suitable organizational structures. In the following, we describe the method steps to specify a Decentralized Organization Model.

Step 1: DAO Specification Developers should, first, specify DAOs in the diagram, and assign to them their *governance area* elements. For each DAO, a *mission statement* should be indicated, defining at a high level the main organizational goal of the DAO. The type of implementation is specified for each governance area, including *on-chain* governance areas first, followed by the *hybrid* and *off-chain*.

Step 2: Role and Committee Specification. Subsequently, the roles and relative metadata are specified. Then these are *federated into* suitable *committees*. *Committee* and *role* elements are associated to governance areas with which they are concerned. If the system requires a high degree of *governance scalability*, committees will aggregate a high number of agents, and, in order to limit governance complexity, the average aggregation level of the DAO should be contained as much as possible.

[9] https://github.com/SoweluAvanzo/DAOMod
[10] https://lxml.de/
[11] https://pypi.org/project/xmlschema/

Step 3: Control Relation Definition. Subsequently, for each role and committee entity, control relations are specified, considering the need for roles or committees to grant or revoke permissions. Control relations should also be defined in case permission delegation relations are needed.

Step 4: Permission Specification. Finally, each role and committee can be assigned specific permissions. The assignment order follows the priority rank mentioned in Section 3, based on *permission_type*. This allows developers to specify the functions that are most critical to DAO security first, followed by those that are less critical.

Step 5: Model Review and Validation. Finally, developers should review the model and check that permissions are assigned correctly. This involves in particular ensuring that each permission presents the right priority degree, based on the *permission type* attribute, and checking whether unintended power escalations are possible in the defined model. Such an event may occur in the case that a role or committee, by means of the control relations that connect it to other entities, can be granted or revoked a permission that is not supposed to. This may imply that, under certain conditions, critical DAO operations can be hindered.

In the following, we discuss the trade-offs faced by developers in modeling the governance structure of a DAO. A first trade-off exists between the need to ensure security through the immutability of governance rules and the necessity to enable dynamic adaptability to unforeseen conditions (Upgradability). Therefore, if the former requirement is prominent, users shall define governance structures where few structural permissions are defined, and control relations should avoid self-loops, which indicate control over a role's own permissions. If the latter is more important, by contrast, the user shall define a governance structure with control relations enabling the upgrade of structural permissions. This guarantees more granular upgradability and adaptability to new conditions.

There is also a trade-off in the selection of the *implementation* parameter of governance areas. If most governance areas are handled *on-chain*, the DAO will face scalability constraints. In contrast, if *off-chain* governance areas are more prominent, the DAO will face security constraints, and undermine its *infrastructure decentralization*, as this characteristic depends on the *on-chain* execution of system functions.

Governance decentralization, instead, in this early stage of design, is influenced by the power distribution between multiple committees and roles, and by the extent to which roles and committees are included in (or can be excluded from) decision making. Furthermore, the *hierarchical_inheritance* parameter provides increased security and control over the organizational structure, but may hinder its decentralization, as more power is provided to controllers to exclude the controlled from DAO operations.

Furthermore, the more *structural* and *strategic* permissions are distributed between all stakeholders, the more decentralized the governance of the DAO.

6 Evaluation

In Section 6.1 we evaluate the modeling method by discussing the use case models that respond to the needs of the systems outlined in 2.1.1. Subsequently, in Section 6.2, we analyze the qualitative usefulness evaluation conducted on the CommonsHood case study.

6.1 CommonsHood Use Case Model

We present here the diagrams of CommonsHood's DAO, modeled by applying the method steps outlined in Section 5. Due to the need to ensure syntactic correctness, the model was transformed into an XML file that reports the corresponding metadata of elements and relations, following the approach used in [Udo22]. Each of them was, subsequently, validated using the provided Python script, which checks that all conditions defined in Section 4 are met, thus ensuring the correctness of the model based on the defined syntax. The model met the syntactic correctness requirements. The XML files, schemas reporting the model data, the script and results of the syntactic correctness checks, which are not reported here due to lack of space, can be found in the GitHub repository [12].

In order to meet the needs of the stakeholders involved in the CommonsHood in the Garden project, described in Section 2.1.1, two types of DAO were modeled. Figure 4 specifies the organizational structure of the former type of DAO, which will be used by associations and collectives that participate in the project to automate and decentralize their decision-making processes, as evidenced in the *mission_statement*. As multiple associations are currently present in the ecosystem and more are expected to join it, multiple DAOs of this type shall be deployed. Two *governance area* elements are defined in this DAO type, concerning *Association Management* and *Association Membership*. The DAO presents a flat organizational structure in which only one role exists (*Association Member*). This takes part in and *is-controlled-by* the *Association Board committee*, which signifies that *Association Members* can collectively evaluate the *application* of new participants, and also exclude existing members through voting.

Proposals may involve arbitrary token transactions from the DAO treasury or calls to other smart contract functions on behalf of the DAO. In particular, the DAO can also call a dedicated smart contract function to propose tasks to the community that can be performed voluntarily by members according to collaborative principles. The smart contract implements an escrow account and is used by *Association DAOs* to distribute *Reward Tokens* (RT) to volunteers upon successful completion of the tasks assigned and the distribution of the tokenized completion certification to the volunteer. To start the task proposal process, *Association members* first propose a tokenized reward for the completion of the task through a dedicated function. If this proposal is approved by the Board, the chosen amount of tokens is transferred to the escrow account. The tokens are time-locked for a predetermined period. Violations can also be reported by *Association Members* if it is found that the task has not been completed successfully. Unless *violation reports* are submitted for the assigned task, the *Association DAO* transfers an NFT-based certificate as a confirmation of task execution to the smart contract of the escrow account, which distributes the rewards tokens to the volunteer. Otherwise, the funds are returned to the Association DAO treasury after a predetermined period has passed.

Permissions assigned to the *committee* of the DAO also include setting up token-based crowdfunding campaigns of the DAO through the Crowdsale smart contract. Since the project ecosystem requires a coordination unit, a DAO instance with a different governance structure and features named *Community DAO* was designed to coordinate and interact with *Association DAOs*. The governance structure of this type of DAO is modeled in Figure 3. Two *committees* are specified, named *Operations Committee* and *Policy Committee*. The former coordinates the daily

[12] https://github.com/SoweluAvanzo/DAOMod/tree/master/DAOMod Schemas and Models

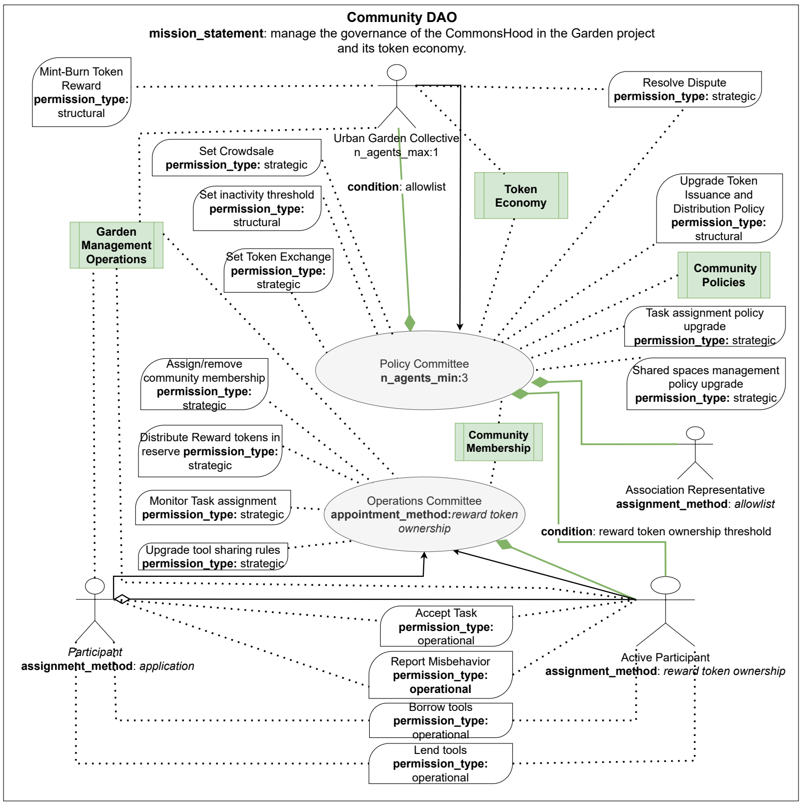


Figure 3: Decentralized Organization model of the Community DAO.

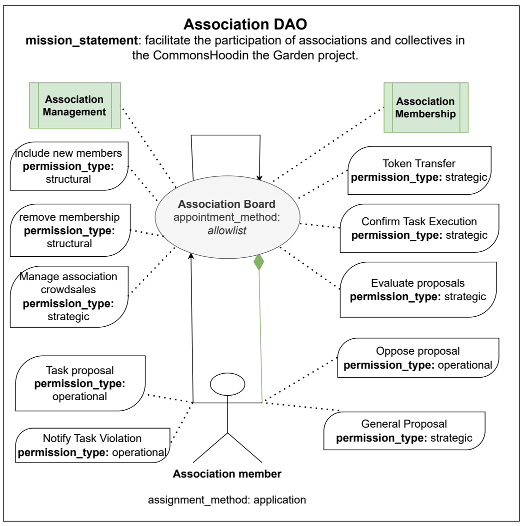


Figure 4: Decentralized Organization model of the Association DAO.

operations of the community for which collective deliberation is needed. These include transferring tokens from the community reserve, onboarding new *Association DAOs* in the project, and evaluating applications of individual *participants*, which are by default approved.

The *Policy Committee*, by contrast, is tasked with defining the general policies and rules of the project. In addition, it is responsible for controlling the token economy of the whole community. This control is based on authorizations to upgrade the smart contracts that execute crowdsales, token exchanges and the token issuance policies and tokens. Four *roles* are specified: *Urban Garden Collective*, *Participant*, *Active Participant*, *Association Representative*. The *Urban Garden Collective* and *Association Representative* roles are played by accounts of *Association DAOs*, and therefore are designed for *autonomous agents*. Having dedicated roles in the Community DAO ensures their focused involvement in the governance of the community.

The *Urban Garden Collective* role represents the urban gardening collective currently in charge of coordinating and bootstrapping the project. Acting as a project coordinator, this entity is initially provided with a high degree of control over the *Community DAO*, symbolized by the crucial *structural* and *strategic* permissions assigned to it.

The *Participant* role is assigned to volunteers when they join the Community DAO. This role entitles them to accept and execute the tasks proposed by the associations, borrow and lend tools in exchange for tokens. In addition, they can report misbehavior of other users to the DAO, which triggers a dispute resolution process. The ownership of *reward tokens* enables users to transition to the *Active Participant* role. The *Active Participant* role can vote as part of the Operations Committee, and, upon reaching a given threshold of Reward Tokens, also in the *Policy Committee*. Also the *Association Representatives* and the *Urban Garden Collective* roles participate in this committee by default.

6.2 Semi-Structured Interview Results

Four interviews were conducted. The purposive sample includes two participants in the CommonsHood project and two external industry experts. The former group includes a senior project manager, who managed IT projects focused on blockchain-based civic technologies for the Computer Science Department of the University of Torino for 4 years and a research fellow in Computer Science with a master's degree in Economics, who has worked on the CommonsHood in The Garden project for one year. He conducted requirement elicitation, communication with project stakeholders, and testing activities for the project. Both external industry experts are senior full-stack developers specialized in Web-3 projects. The former holds a master's degree in computer engineering and the latter in Multimedia Engineering.

Following a common approach for the evaluation of modeling languages, we examined two key dimensions: semantic usefulness and pragmatic usefulness [DNW+21]. Below is a brief description of each. The *semantic usefulness* refers to the ability of the modeling method to capture suitable design features of the DAOs and provide a correct representation. The metrics associated with this category include *correctness*, *relevance*, *completeness* and *authenticity* adapted from the study in [DINM23]. The *pragmatic usefulness* of the modeling method refers to its perceived utility and practical applicability in organizational contexts. In the following, we describe the properties considered to measure the pragmatic aspects of the modeling language and method, which we adapted from the study in [DINM23]. The *subjective norm* property indicates the extent to which important positions in organizations will support the adoption of the method according to the interviewees. The respondents commented on the perceived *relevance* of mastering DAOMod for their type of job and the perceived *output quality*. Furthermore, we investigated the *demonstrability* of the benefits of using DAOMod, the perceived impact on *performance* and *productivity* according to the interviewees. Finally, the *perceived usefulness* metric was investigated based on responses from case study participants. The interview track, along with the metrics assessed per each question, is provided in the Appendix A.

The interviewees confirmed the *correctness* of the DAO models generated using the method steps, stating that the use case models exhibited no significant flaws. They also noted that the CommonsHood DAO model advances the initial specification work of the DAOs that the team had previously conducted. Furthermore, all interviewees provided positive feedback on the *completeness* of the syntax chosen to represent the DAO domain. They emphasized its richness as a valuable feature for users and highlighted that the syntax includes elements capable of representing any DAO. Regarding the *authenticity*, and precision of the models, some concerns emerged. An interviewee pointed out that since different blockchains and smart contract languages could be used for development, multiple translation logics should be defined. In the case study, both CommonsHood participants anticipated challenges for users in adapting to such complex systems once implemented, particularly because the CommonsHood in the Garden project is still in its early stages. However, one of the interviewees noted that this decentralized governance infrastructure will be essential in the long run to support the project's scaling and decentralization.

The interviewees recognized DAOMod as a valuable standard for formalizing and communicating DAO specifications. An interviewee emphasized its usefulness in providing a shared language to define interactions within DAO platforms, such as CommonsHood. Moreover, interviewees confirmed the remarkable simplicity of the proposed diagrams. However, a key limitation identified concerns its limited accessibility to non-technical stakeholders, who may require additional guidance to familiarize themselves with the modeling method, which is perceived as more suited to a technical audience with prior knowledge of the DAO domain. Furthermore, DAOMod was noted to be an effective tool for onboarding new team members, particularly in the context of development projects focused on complex systems such as DAOs. A major limitation highlighted by multiple interviewees was the lack of code generation support, with all evidencing the need for automated smart contract code generation to enhance the practicality of the method for developers.

The novelty of the method is also associated with its lack of maturity compared to existing alternatives. This, as highlighted by one respondent, may limit the support received from relevant positions in organizations. Finally, the interviews showed that, in terms of *output quality*, impact on *productivity* and *performance*, the method may be more suitable for large teams and organizations than for smaller ones. Moreover, most participants agree that the approach may have a positive impact on such metrics in the long run, but in the short run it may reduce them, due to the need for users to familiarize with it and with the DAO domain.

7 Discussion and Conclusion

This article presents the development and evaluation of a modeling method for DAOs. This is developed by first investigating reference functional and non-functional requirements of DAOs discussed in the literature. Based on such reference requirement categories, we developed the DAOMod ontology. Subsequently, we propose step-by-step guidelines for modeling DAOs using the developed language. The method proposed guides developers in selecting suitable design features by means of prioritized non-functional requirements of the project.

The modeling method is evaluated through an *in vivo* case study. The use case demonstration confirms the real-world applicability of DAOMod and illustrates how it can address the key governance challenges outlined in the introduction. Specifically, the instantiation of the Commons-Hood system governance model provides an example of how organizations can adopt DAOMod to align the intended organizational structure design with their project requirements.

The DAO models instantiated using DAOMod facilitate establishing a separation of concerns among roles and committees, which allows members to engage in parallel deliberation and focused task execution in the context of their respective domains (symbolized by Governance Areas and Permission elements). This separation of concerns can mitigate decision-making bottlenecks and increase participation, thus countering the issue of voter apathy and limited voting scalability. Moreover, the control relations specified in the two DAOs are intended to provide the committees of the DAOs with granular control of individual members, increasing the security of the system by mitigating the impact of malicious behavior on governance. The method proved to be effective in supporting stakeholders in the specification and refinement of the main goals of the system (Governance Areas and Permission assignment), which aspect is useful for the implementation of suitable incentive mechanisms. These are critical in DAOs, as they foster alignment around shared objectives, counteract community fragmentation, and sustain long-term collective engagement.

The semantic and pragmatic utility of DAOMod were also positively evaluated by the semi-structured interviews conducted with the case study participants. Concerning semantic qualities, the case study participants positively commented on the precision, completeness, and relevance of the models. Moreover, the richness of the syntax was remarked to be necessary to model DAOs due to their complexity. One main limitation was mentioned with respect to the semantic usefulness of DAOMod models: the need to fill the semantic gap between the modeling language models and their translation into code. Given the focus on platform-independent design, the development of a translator of DAOMod models into smart contract code is not the focus of this article. However, this can be achieved on the basis of the foundation provided by the context-free grammar formalized in this work.

DAOMod was also positively evaluated with respect to its pragmatic usefulness. In this regard, the main strengths that were evidenced included the novelty of the approach and its utility to standardize DAO specification processes, and for documentation purposes.

One major limitation in this respect is that non-technical stakeholders have to first familiarize themselves with the DAO domain to benefit from the modeling method. In addition, development teams must make an initial investment in time and resources to learn the method, which, according to respondents, ultimately improves their productivity and performance in the long run. These remarks are consistent with the inherent complexity of the DAO domain, and therefore of DAO development projects.

Future work will concern the implementation of dedicated tool support for the modeling language to increase the usability of the approach and, therefore, its maturity. Moreover, while comprehensively addressing the early stages of design, the method lacks support for code generation. Cinco Cloud provides a sophisticated environment for developing tool support for graphical languages such as DAOMod. Among other features, it provides a modeling canvas, views to manipulate the model and it's properties, the ability to support custom file codecs, and the ability to integrate code generators. Third, since the modeling method is focused on static modeling of DAOs, we foresee the extension of the method to incorporate dynamic aspects needed to capture the evolving nature of DAOs.

A further research line that stems from this work will concern the implementation and evaluation of the DAOs modeled as part of the case study. This could be combined with an evaluation of the system architecture using scenario-based evaluation methodologies. In particular, the methodology presented in [SN18] can be used, as it is particularly suitable for large-scale e-governance systems.

Finally, we also recognize the limitations of the case study-based evaluation conducted in terms of generalizability. More specifically, although the interviews enabled us to collect feedback from both technical and non-technical stakeholders, we acknowledge that, since all interviewees have prior knowledge of blockchain technology, a potential bias is introduced. For this reason, a more comprehensive evaluation of the approach is needed. In particular, this may include quantitative evaluation by means of surveys with a larger sample of industry experts and developers. In this subsequent evaluation step, the sample should include also developers and non-technical users lacking prior knowledge of the Web3 domain. Furthermore, several alternative methods supporting governance design are well established in the industry, although not tailored for DAO specificities. Therefore, we plan to extend our evaluation to include a comparative analysis of the perceived utility of DAOMod compared to alternatives not specifically focused on Web3 design, such as Archimate[13], the original Organization Models [ST], and governance-oriented business process modeling languages like eSML [NMD+15].

Acknowledgement

This research is funded by the PON REACT-EU program, DM 1061/2021, Action IV.5, the Estonian PRG1641 project, the Estonian MKM-POL21-2025 project on the blockchain-based implementation of a decentralized knowledge graph, and Circular Health EDIH (CHEDIH) under the Digital Europe Program GA 101083745. The authors acknowledge collaboration with Circles Coop, Bitspossessed collective, Elena San Miguel Perez, and Enrique Alcaraz Perez.

A Semi-Structured Interview Track

We report below the interview track followed. Each question is followed by the relative metric of semantic and pragmatic usefulness aimed, according to the study [DINM23].

- **Question 1:** Does the DAOMod method correctly specify DAOs? Do you see any particular flaw? (Correctness)

[13] https://www.opengroup.org/archimate-forum/archimate-overview

- **Question 2:** Are elements, relations and properties relevant to the DAO domain? (Relevance)
- **Question 3:** Does DAOMod contain all elements and properties needed to define DAOs effectively in the early stages of development? (Completeness)
- **Question 4:** Does DAOMod provide a realistic representation of a DAO? (Authenticity)
- **Question 5:** Do you find the creation of DAOMod models is useful in your job? (Usefulness)
- **Question 6:** Would relevant positions in your organization support the adoption of DAOMod for shaping DAOs in the early stages of development? (Subjective Norm)
- **Question 7:** Will DAOMod usage or application be relevant in jobs like yours? (Job relevance)
- **Question 8:** Will the quality of output I get from DAOMod be high? (Output Quality)
- **Question 9:** Assuming that you are using DAOMod. Do you think you could easily explain the benefits of using DAOMod to others? (Demonstrability)
- **Question 10:** Would your or your team's performance improve if you use DAOMod? (Performance)
- **Question 11:** Would you be more productive if you use the DAOMod in your job, provided that you need to develop a DAO? (Productivity)

Bibliography

[ANS23] S. Avanzo, A. Norta, C. Schifanella. A Modelling Approach for a High Utility Decentralized Autonomous Organization Development. In *5th International Congress on Blockchain and Applications (BLOCKCHAIN'23)*. Guimarães, Portugal, July 12-14 2023.

[APD+25] S. Avanzo, D. Pautasso, I. Domenicale, A. Norta, J. Linares, C. Schifanella, M. Hattingh. DAOMod: A Modeling Method for Decentralized Autonomous Organization Development. Technical report, 2025. Available at https://www.researchgate.net/publication/386019556_DAOMod_A_Modeling_Method_for_Decentralized_Autonomous_Organization_Development.

[BFJ23] E. Baninemeh, S. Farshidi, S. Jansen. A decision model for decentralized autonomous organization platform selection: Three industry case studies. *Blockchain: Research and Applications* 4(2):100127, 2023.

[BLSA19] L. Bindra, C. Lin, E. Stroulia, O. Ardakanian. Decentralized access control for smart buildings using metadata and smart contracts. In *2019 IEEE/ACM 5th International Workshop on Software Engineering for Smart Cyber-Physical Systems (SEsCPS)*. Pp. 32–38. 2019.

[BT24] S. Bonnet, F. Teuteberg. Decentralized autonomous organizations: A systematic literature review and research agenda. *International Journal of Innovation and Technology Management* 21(04):2450026, 2024.

[CLB+22] S. Craß, A. Lackner, N. Begic, S. A. M. Mirhosseini, N. Kirchmayr. Collaborative Administration of Role-Based Access Control in Smart Contracts. In *2022 4th Conference on Blockchain Research & Applications for Innovative Networks and Services (BRAINS)*. Pp. 87–94. 2022.

[DINM23] V. K. Dwivedi, M. Iqbal, A. Norta, R. Matulevičius. Evaluation of a legally binding smart-contract language for blockchain applications. *Journal of Universal Computer Science* 29(7):691–717, 2023.

[DMH+21] V. Dhillon, D. Metcalf, M. Hooper, V. Dhillon, D. Metcalf, M. Hooper. Decentralized organizations. *Blockchain enabled applications: Understand the blockchain ecosystem and how to make it work for you*, pp. 73–111, 2021.

[DNW+21] V. Dwivedi, A. Norta, A. Wulf, B. Leiding, S. Saxena, C. Udokwu. A formal specification smart-contract language for legally binding decentralized autonomous organizations. *IEEE access* 9:76069–76082, 2021.

[HMPR08] A. R. Hevner, S. T. March, J. Park, S. Ram. Design science in information systems research. *Management Information Systems Quarterly* 28(1):6, 2008.

[KG21] F. Kaya, J. Gordijn. Decent: An ontology for decentralized governance in the renewable energy sector. In *2021 IEEE 23rd conference on business informatics (CBI)*. Volume 1, pp. 11–20. 2021.

[MC23] F. Monteiro, M. Correia. Decentralised Autonomous Organisations for Public Procurement. In *Proceedings of the 27th International Conference on Evaluation and Assessment in Software Engineering*. Pp. 378–385. 2023.

[NMD+15] A. Norta, L. Ma, Y. Duan, A. Rull, M. Kõlvart, K. Taveter. eContractual choreography-language properties towards cross-organizational business collaboration. *Journal of Internet Services and Applications* 6(1):1–23, 2015.

[PJBO21] R. v. Pelt, S. Jansen, D. Baars, S. Overbeek. Defining blockchain governance: A framework for analysis and comparison. *Information Systems Management* 38(1):21–41, 2021.

[QDL+22] R. Qin, W. Ding, J. Li, S. Guan, G. Wang, Y. Ren, Z. Qu. Web3-based decentralized autonomous organizations and operations: Architectures, models, and mechanisms. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 53(4):2073–2082, 2022.

[RHRR12] P. Runeson, M. Host, A. Rainer, B. Regnell. *Case study research in software engineering: Guidelines and examples*. John Wiley & Sons, 2012.

[RJK23] O. Rikken, M. Janssen, Z. Kwee. The ins and outs of decentralized autonomous organizations (DAOs) unraveling the definitions, characteristics, and emerging developments of DAOs. *Blockchain: Research and Applications*, p. 100143, 2023.

[SKP+23] T. Sharma, Y. Kwon, K. Pongmala, H. Wang, A. Miller, D. Song, Y. Wang. Unpacking How Decentralized Autonomous Organizations (DAOs) Work in Practice. *arXiv preprint arXiv:2304.09822*, 2023.

[SN18] S. Saay, A. Norta. Designing a Scalable Socio-Technical Method for Evaluating Large e-Governance Systems. In *Advanced Computational and Communication Paradigms*. Pp. 571–580. Springer, 2018.

[ST] L. S. Sterling, K. Taveter. *The Art of Agent-Oriented Modeling*. The MIT Press. OCLC: 1178943942.

[Udo22] C. J. Udokwu. A modelling approach for building blockchain applications that enables trustable inter-organizational collaborations. 2022.

[VAB+23] C. Viano, S. Avanzo, G. Boella, C. Schifanella, V. Giorgino. Civic Blockchain: Making blockchains accessible for social collaborative economies. *Journal of Responsible Technology* 15:100066, 2023.

[VAC+] C. Viano, S. Avanzo, M. Cerutti, A. Cordero, C. Schifanella, G. Boella. Blockchain tools for socio-economic interactions in local communities. 41(3):373–385.
doi:10.1093/polsoc/puac007
https://academic.oup.com/policyandsociety/article/41/3/373/6540164

[VJ23] P. Van Vulpen, S. Jansen. Decentralized autonomous organization design for the commons and the common good. *Available at SSRN 4418782*, 2023.

[VP24] M.-C. Valiente, J. Pavón. Web3-DAO: An ontology for decentralized autonomous organizations. *Journal of Web Semantics* 82:100830, 2024.
doi:https://doi.org/10.1016/j.websem.2024.100830
https://www.sciencedirect.com/science/article/pii/S1570826824000167

[WYS+22] Q. Wang, G. Yu, Y. Sai, C. Sun, L. D. Nguyen, S. Xu, S. Chen. An Empirical Study on Snapshot DAOs. *arXiv preprint arXiv:2211.15993*, 2022.

[ZAL+22] X. Zhao, P. Ai, F. Lai, X. R. Luo, J. Benitez. Task management in decentralized autonomous organization. *Journal of Operations Management* 68(6):649–674, 2022.
doi:10.1002/joom.1179
https://onlinelibrary.wiley.com/doi/10.1002/joom.1179

[ZW22] C. Ziegler, I. M. Welpe. A Taxonomy of Decentralized Autonomous Organizations. 2022.