ETSI GR PDL 029 V1.1.1 (2025-02)

Permissioned Distributed Ledger (PDL); Digital Autonomous Organization (DAO)

Disclaimer

The present document has been produced and approved by the Permissioned Distributed Ledger (PDL) ETSI Industry Specification Group (ISG) and represents the views of those members who participated in this ISG.
It does not necessarily represent the views of the entire ETSI membership.

Reference
DGR/PDL-0029_Digi_Auto_Orga

Keywords
data interoperability, data models, PDL

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - APE 7112B
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° w061004871

Contents

Intellectual Property Rights 9
Foreword 9
Modal verbs terminology 9
Executive summary 9
Introduction 10
1 Scope 11
1.1 In Scope 11
1.2 Out of Scope 12
2 References 12
2.1 Normative references 12
2.2 Informative references 12
3 Definition of terms, symbols and abbreviations 13
3.1 Terms 13
3.2 Symbols 14
3.3 Abbreviations 14
4 Definition of a DAO 15
4.1 Foreword 15
4.2 Current definition 15
4.3 Current Implementations 16
4.4 Differences between a DAO and traditional organizations 16
5 Recommendations to operate a DAO 16
5.1 Foreword 16
5.2 Key components of a DAO 17
5.2.1 Implementation Agreements 17
5.2.2 Governing Entity 17
5.2.3 Smart Contracts 17
5.2.4 Data Models 17
5.2.5 Processes 17
5.2.6 Ledgers 17
5.2.7 Codebase 18
5.2.7.1 Definition of a codebase 18
5.2.7.2 Open Source 18
5.2.7.3 Proprietary code 18
5.2.8 Ledger type 18
5.2.8.1 Blockchain attributes 18
5.2.8.2 Functional division of Blockchain 18
5.2.8.3 Multi-Blockchain scenarios 19
5.2.9 Validating nodes 19
5.2.10 Non-validating nodes 19
6 Operating a DAO 19
6.1 Foreword 19
6.1.1 Current DAO Implementations 19
6.1.2 Challenges of Single-Software/Code-development Implementations 19
6.1.3 Composite DAOs 19
6.1.4 DAO Types Defined in this Document 20
6.1.5 Document Structure 20
6.2 Single Ledger DAO environment 20
6.2.1 Single Ledger considerations 20
6.2.1.1 Codebase 20
6.2.1.1.1 Components of the Codebase 20
6.2.1.1.2 Sources of the Codebase 20
6.2.1.1.3 Implications of Single Codebase 21

6.2.1.2 Nodes 21
6.2.1.3 Governance 21
6.2.1.3.1 Concept of Autonomy 21
6.2.1.3.2 Governance Mechanisms 21
6.2.1.3.3 Evolution of the Governing Entity 21
6.2.1.3.4 Key Tasks of the Governing Entity 21
6.2.1.3.5 Further Discussion 22
6.3 Hybrid DAO environment 22
6.3.1 Hybrid DAO environment considerations 22
6.3.2 Types of Hybrid environments 22
6.3.2.1 Multi Ledger 22
6.3.2.2 Multi Vendor 22
6.3.2.3 Multi codebase 22
6.3.3 Multi-Ledger Interoperability 23
6.3.3.1 Challenges of interoperability 23
6.3.3.2 Data Interoperability 23
6.3.3.3 Consensus interoperability 23
6.3.3.4 Smart-Contract interoperability 23
6.3.4 Multi Vendor/Codebase 24
6.3.4.1 Aspects of Multiple vendors/codebases 24
6.3.4.1.1 Operational flow of a platform 24
6.3.4.1.2 Data Structure 24
6.3.4.1.3 Platform Architecture 24
6.3.4.1.4 Processes 24
6.3.4.2 Data Model alignment 25
6.3.4.3 Architectural alignment 25
6.3.4.3.1 Functional Blocks 25
6.3.4.3.2 Interface Reference Points 25
6.3.4.4 Process alignment 26
6.3.4.4.1 Parallel Processing 26
6.3.4.4.2 Process Expectations 26
6.3.4.4.3 Process Perspectives 26
6.3.4.4.4 Inter-ledger Communication 26
6.3.4.4.5 Challenges and Considerations 26
6.3.4.4.5.1 Common Challenges 26
6.3.4.4.5.2 Potential solutions for developers to consider 26
6.3.4.4.5.2.1 Synchronization Mechanisms 26
6.3.4.4.5.2.2 Standardized Data Exchange 26
6.3.4.4.5.2.3 Adaptive Processing 27
6.3.4.4.5.3 Additional Considerations 27
6.3.4.4.5.3.1 Scalability 27
6.3.4.4.5.3.2 Security 27
6.3.4.4.5.3.3 Auditability 27
6.3.4.4.5.3.4 Fault Tolerance 27
6.3.4.4.5.3.5 Interoperability 27
6.3.4.4.5.3.6 Performance Optimization 28
6.3.4.4.5.3.7 Regulatory Compliance 28
6.3.4.4.6 Testing and Verification 28
6.4 Tokens 28
6.4.1 Understanding Tokens 28
6.4.2 Governance Tokens (GTs) 28
6.4.3 Membership Tokens (MTs) 29
6.4.4 Reward Tokens (RTs) 29
6.4.5 Key considerations for token economics (tokenomics) 29

7 Governance 29
7.1 Definition of governance 29
7.1.1 Key principles of Governance 29
7.1.1.1 Architectural concepts pf governance 29
7.1.1.2 Decentralized Decision-Making 30
7.1.1.3 Token-Based Voting 30
7.1.1.4 Proposal Mechanisms 30

7.1.1.4.1 The purpose of proposals....30
7.1.1.4.2 Proposal process....30
7.1.1.5 Smart Contract Execution....31
7.1.1.6 Transparency and Immutability....31
7.1.1.7 Incentive Alignment....31
7.1.1.8 Adaptability....31
7.1.2 Governance Models....31
7.1.2.1 Vertical Governance....31
7.1.2.2 Horizontal Governance....31
7.1.2.3 Balancing Governance Models....32
7.1.3 Purpose of Governance....32
7.1.4 Governance Structure Objectives....32
7.1.4.1 Primary Aim....32
7.1.4.2 Achievement Methods and Mechanisms....32
7.1.4.2.1 Risk Management....32
7.1.4.2.2 Operational Control....32
7.1.4.2.3 Stakeholder Protection....32
7.1.4.3 Balancing Priorities....33
7.1.4.4 Continuous Improvement....33
7.1.5 Governance Functionality Aspects....33
7.1.5.1 Managerial Governance....33
7.1.5.2 Operational Governance....33
7.1.5.3 Hybrid Governance....33
7.1.6 Architectural elements....33
7.1.6.1 Implementation agreements....33
7.1.6.2 Governing Entity....34
7.1.6.2.1 Core Responsibilities....34
7.1.6.2.2 Additional responsibilities related to Establishment and Operation....34
7.1.6.2.3 Types of Governing Entities....34
7.1.6.2.4 Governance Flexibility....35
7.1.6.2.5 Transparency and Accountability....35
7.1.6.3 DAO governance....35
7.1.6.3.1 Basic Principles and aspects....35
7.1.6.3.2 DAO component abstraction....36
7.1.6.3.3 Challenges in Consensus Abstraction....36
7.1.7 Challenges and Considerations....36
7.1.7.1 Voter Apathy and Participation....36
7.1.7.1.1 Challenges....36
7.1.7.1.2 Potential solutions....36
7.1.7.2 Regulatory Compliance....37
7.1.7.2.1 Challenges....37
7.1.7.2.2 Considerations and solutions....37
7.1.7.3 Security and Smart Contract Vulnerabilities....37
7.1.7.3.1 Challenges....37
7.1.7.3.2 Potential solutions....37
7.1.7.4 Scalability of Decision-Making....37
7.1.7.4.1 Challenges....37
7.1.7.4.2 Potential solutions....37
7.1.7.5 Plutocracy and Concentration of Power....37
7.1.7.5.1 Challenges....37
7.1.7.5.2 Potential solutions....38
7.1.7.6 Governance Attacks and Game Theory....38
7.1.7.6.1 Challenges....38
7.1.7.6.2 Potential solutions....38
7.1.7.7 Cross-ledger Governance and Interoperability....38
7.1.7.7.1 Challenges....38
7.1.7.7.2 Potential solutions....38
7.2 Governance in a Single ledger DAO environment....38
7.3 Governance in a Hybrid DAO environment....39
7.3.1 Introduction....39
7.3.2 Alignment of data across ledgers....39
7.3.3 Alignment of consensus protocols across ledgers....40

7.3.3.1 Basic Principles 40
7.3.3.1.1 Introduction 40
7.3.3.1.2 Variations in Quorum and Majority Rules 40
7.3.3.1.3 Population Differences Across Ledgers 40
7.3.3.2 Superset ledgers 42
7.3.3.3 Majority and Quorum in superset ledgers 43
7.3.4 Alignment of Smart Contracts across ledgers 43
7.3.4.1 Introduction 43
7.3.4.2 Smart Contract Abstraction 43
7.3.4.3 Testing and Certification 43
7.3.4.3.1 Introduction to Testing and Certification 43
7.3.4.3.2 Testing 43
7.3.4.3.3 Certification 44
7.3.4.3.4 Entities performing Testing and Certification Testing 45
7.4 Governance through consensus 45
7.4.1 On-ledger consensus 45
7.4.2 Off-ledger consensus 46
7.4.3 Multi-ledger consensus 46
7.4.3.1 Definition 46
7.4.3.2 Voting and majority in a multi ledger environment 47
7.5 Governance without consensus 47
7.5.1 Scenarios 47
7.5.2 Delegated governance 47
7.5.2.1 Liquid Democracy 47
7.5.2.1.1 The need for delegates 47
7.5.2.1.2 Key Principles 48
7.5.2.1.3 Benefits in DAO Governance 48
7.5.2.2 Human delegates 48
7.5.2.3 Machines as delegates 48
7.5.3 Expert-driven governance 49
7.6 Automated governance 49
7.6.1 Introduction to Automated Governance 49
7.6.2 Recommendations 49
7.6.2.1 The components and minimal requirements 49
7.6.2.2 Unified data models 49
7.6.2.3 Unified processes 50
7.6.2.4 Unified IRPs 50
7.6.3 Algorithmic governance 50
7.6.4 Codebase alignment 50
7.6.4.1 The need for alignment 50
7.6.4.2 Standardized development practices 51
7.6.4.3 Modular architecture 51
7.6.4.4 Shared libraries and frameworks 51
7.6.4.5 Cross-ledger compatibility 51
7.6.4.6 Continuous Integration and Deployment (CI/CD) 51
7.6.4.7 Documentation and knowledge sharing 51
7.6.4.8 Governance of code changes 51
7.6.4.9 Security alignment 52
7.6.4.10 Performance optimization 52
7.6.4.11 Interoperability testing 52
7.7 Hybrid governance 52
7.7.1 The basics of hybrid governance 52
7.7.2 Evolution of hybrid governance 52
7.7.3 Initiation 53
7.7.4 Scale 53
7.7.5 Self sustained 53
7.7.6 Change management 53
7.8 Tokens in a Hybrid environment 53
7.8.1 The need for tokens in a hybrid environment 53
7.8.2 Multi-token ecosystem 54
7.8.3 Cross-ledger token compatibility 54
7.8.4 Weighted voting 54

7.8.4.1 The need for weighted voting....54
7.8.4.2 Token-Weighted Voting....54
7.8.4.3 Quadratic Voting....54
7.8.4.4 Reputation-Based Voting....55
7.8.4.4.1 Description of Reputation....55
7.8.4.4.2 Reputation Metrics....55
7.8.4.4.3 Benefits....55
7.8.4.4.4 Implementation Considerations....56
7.8.5 Liquidity provision....56
7.8.6 Smart contract interoperability....56
7.8.7 Adaptive issuance and burning....56
7.8.8 Governance-controlled parameters....56
7.8.9 Privacy considerations....57
7.8.10 Emergency measures....57
7.8.11 Token upgradability....57
8 Final Thoughts....57
8.1 Key Takeaways....57
8.2 Moving forward....58
Annex A: List of Recommendations for DAOs....59
History....61

List of tables

Table 1: Comparison between a DAO and traditional organizations 16
Table 2: Consensus scenarios 42

List of figures

Figure 1: Elements of Governance 30
Figure 2: Data alignment across ledgers 40
Figure 3: Population division across ledgers 41
Figure 4: Merged populations 42
Figure 5: Ingress/Egress Data Testing Process 44
Figure 6: Condition/Behaviour Testing Process 44

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI IPR online database.

Pursuant to the ETSI Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™**, **LTE™** and **5G™** logo are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM**® and the GSM logo are trademarks registered and owned by the GSM Association.

Foreword

This Group Report (GR) has been produced by ETSI Industry Specification Group (ISG) Permissioned Distributed Ledger (PDL).

Modal verbs terminology

In the present document "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

Executive summary

The present document provides a comprehensive exploration of Digital Autonomous Organizations (DAOs) in the context of Permissioned Distributed Ledger (PDL) systems. It addresses the definition, implementation, and governance of DAOs, with a particular focus on the challenges and solutions in both single-ledger and hybrid multi-ledger environments.

The present document aims to provide a thorough understanding of DAO implementation and governance, offering insights and potential solutions to the unique challenges posed by decentralized autonomous systems, especially in complex multi-ledger environments. It serves as a guide for developers, policymakers, and organizations interested in implementing or participating in DAOs within PDL systems.

By addressing both the technical and governance aspects of DAOs, the present document provides a holistic view of their potential to revolutionize organizational structures in the digital age, particularly in the context of the telecommunications and ICT industries.

Introduction

Blockchain technology has introduced a new era of decentralized systems, with Digital Autonomous Organizations (DAOs) emerging as a revolutionary concept in organizational structure and governance. The present document explores the intricate world of DAOs within the context of Permissioned Distributed Ledger (PDL) systems, providing a comprehensive analysis of their definition, implementation, and governance.

DAOs represent a paradigm shift from traditional centralized organizations, offering a model where decision-making and management are distributed among participants. By leveraging smart contracts and distributed ledger technology, DAOs aim to create transparent, efficient, and autonomous systems that can operate with minimal human intervention.

The implementation of DAOs, particularly in complex environments involving multiple ledgers and codebases, presents unique challenges. The present document aims to address these challenges, offering insights and potential solutions for both single-ledger and hybrid multi-ledger DAO environments.

The present document seeks to provide a comprehensive resource for developers, policymakers, and organizations looking to understand, implement, or participate in DAOs within PDL systems. It aims to bridge the gap between theoretical concepts and practical implementation, offering a roadmap for navigating the complex landscape of decentralized autonomous organizations.

The present document pays particular attention to the challenges and opportunities presented by hybrid environments, where multiple ledgers, codebases, or governance models coexist. It provides insights into how DAOs can leverage these complex structures to create more robust, flexible, and inclusive governance systems.

Furthermore, the present document explores how DAOs can adapt to the specific needs of the telecommunications and ICT industries, potentially revolutionizing how partners in these spaces collaborate, innovate and deliver services globally.

As the field of distributed ledger technology continues to evolve, the insights provided in the present document will serve as a foundation for future developments in DAO implementation and governance. It paves the way for more efficient, transparent, and decentralized organizational structures that can adapt to the rapidly changing technological landscape while maintaining the principles of collective ownership and decision-making that are at the core of the DAO concept.

1 Scope

1.1 In Scope

The present document defines and specifies the implementation and governance of Digital Autonomous Organizations (DAOs) within Permissioned Distributed Ledger (PDL) systems. It encompasses:

1) A comprehensive definition of DAOs, including their key characteristics and differentiating factors from traditional organizations.

2) Detailed specifications of the recommendations to operate a DAO, covering implementation agreements, governing entities, smart contracts, data models, processes, ledgers, and codebases.

3) In-depth analysis of DAO operations in both single-ledger and hybrid multi-ledger environments, addressing the complexities and challenges unique to each scenario.

4) Extensive examination of governance structures and mechanisms in DAOs, including:

    a) Managerial and operational governance.

    b) Consensus-based governance models.

    c) Delegated and automated governance approaches.

    d) Hybrid governance structures.

5) Strategies and methodologies for achieving interoperability in multi-ledger DAO environments, specifically:

    a) Data alignment across different ledgers.

    b) Harmonisation of consensus protocols.

    c) Abstraction and alignment of smart contracts.

6) Comprehensive analysis of token economics in hybrid DAO environments, including:

    a) Multi-token ecosystems and their roles.

    b) Cross-ledger token compatibility and transfer mechanisms.

    c) Innovative voting mechanisms, including weighted and reputation-based voting.

    d) Tokenized reputation systems and their impact on governance.

7) Detailed exploration of automated governance, including:

    a) Recommendations for effective implementation.

    b) Strategies for codebase alignment in multi-vendor environments.

    c) Algorithmic governance mechanisms.

8) Guidelines for the evolution of DAO governance, from initiation to self-sustained operation, with a focus on adaptability and scalability.

9) Consideration of the specific applications and implications of DAOs in the telecommunications and ICT industries.

The present document is intended for use by developers, system architects, policymakers, and organizations involved in the design, implementation, or participation in DAOs within PDL systems. It aims to provide a standardized framework for understanding and implementing DAOs, with a particular focus on addressing the challenges posed by complex, multi-ledger environments and the unique needs of the telecommunications sector.

1.2 Out of Scope

The scope of the present document does not extend to:

1) Specific implementation details of underlying Blockchain or distributed ledger technologies.

2) Legal and regulatory considerations for DAO operations in different jurisdictions, although general compliance challenges are discussed.

3) Detailed economic models and tokenomics beyond their direct application to governance mechanisms.

4) Comprehensive security protocols for DAO operations, although security considerations are mentioned where relevant to governance and implementation.

5) Specific programming languages or development frameworks for implementing DAOs, focusing instead on abstract principles and requirements.

6) Detailed technical specifications for cross-ledger interoperability solutions, though the present document discusses the need for and challenges of such solutions.

While these aspects are crucial for the overall functioning of DAOs, they are beyond the scope of the present document and may be addressed in separate, specialized publications.

2 References

2.1 Normative references

Normative references are not applicable in the present document.

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] Bitcoin whitepaper: "Bitcoin: A Peer-to-Peer Electronic Cash System", Nakamoto, S., 2008.

[i.2] ETSI GR PDL 004 (V1.1.1) (2021-02): "Permissioned Distributed Ledgers (PDL); Smart Contracts; System Architecture and Functional Specification".

[i.3] DNB Working Paper 718: "Governance in systems based on distributed ledger technology (DLT): a comparative study", Ellen Naudts (DNB), Timothy Aerts (DNB), Leonard Franken (AFM), Aimo Pieterse (AFM), AFM -- Publiek, June 28, 2021.

[i.4] ETSI GS PDL 012 (V1.2.1) (2023-06): "Permissioned Distributed Ledger (PDL); Reference Architecture".

[i.5] ETSI GS PDL 015 (V1.1.1) (2023-01): "Permissioned Distributed Ledger (PDL); Reputation Management".

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the following terms apply:

algorithmic governance: governance approach that uses predefined rules and algorithms to automate decision-making processes within a DAO

automated governance: government approach that leverages technology to streamline decision-making processes, reduce human intervention, and increase efficiency within the DAO

autonomous: system/platform that operates independently of involvement from external entities or powers, while still potentially requiring certain external resources

codebase: code used to develop smart contracts, applications, user interfaces and machine interfaces in a DAO

cross-ledger compatibility: ability of tokens, smart contracts, or other Blockchain elements to function across different Blockchain networks

data models: models that organize elements of data and standardize how they relate to one another and to properties of real-world objects

decentralized: managed through decentralized governance using tools that allow it to function without a centralized authority

Decentralized Autonomous Organization (DAO): system that enables distributed decision making, management and ownership of assets, typically operated on a Blockchain or distributed ledger technology

delegated governance: governance model where token holders can delegate their voting power to trusted representatives

expert-driven governance: governance approach that relies on subject matter experts for decision-making in specialized areas

Governance Tokens (GTs): tokens used for voting on DAO proposals and decisions

governing entity: entity performing governance tasks by defining rules and Implementation Agreements, as well as ensuring compliance and resolving conflicts

hybrid DAO: DAO that uses more than one ledger type, codebase, data model or process

hybrid governance: governance approach that combines different governance models to create a flexible, robust and adaptable system

implementation agreements: collections of rules and agreements that define how the DAO is implemented

ledgers: blocks of data cryptographically linked to each other, forming the basis of Blockchain technology

liquidity pool: smart contract-based mechanism that holds reserves of two or more tokens, allowing for trading directly against these reserves

Membership Tokens (MTs): tokens used to grant access to DAO resources and services

multi-ledger consensus: process of achieving agreement across multiple PDLs within a DAO

non-validating nodes: nodes that do not participate in validation or consensus management but are required for proper Blockchain operation, such as aligning data across multiple ledgers

off-ledger consensus: decision-making processes that occur outside of the DAO ledgers but are still part of the DAO's governance structure

on-ledger consensus: process of reaching agreement through mechanisms implemented directly on the DAO ledgers

reputation-based voting: voting mechanism where a participant's voting power is influenced by their reputation score, which is based on their contributions and behaviour within the DAO

Reward Tokens (RTs): tokens used to incentivize contributions and participation in the DAO

smart contracts: executable code embedded into the Blockchain that operates on certain conditions

staking: process of locking up tokens as a form of commitment or collateral within a DAO, often in exchange for voting rights or rewards

tokenomics: study and analysis of the economic aspects of a cryptocurrency or Blockchain project

validating nodes: nodes that take an active part in consensus management, validating data format and ensuring hash integrity

weighted voting: voting system where votes are given different weights based on predefined criteria, such as token holdings or reputation scores

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

NOTE: While some of these abbreviations (like GPS, EU, EEA) are commonly known, they are included as they appear in the context of the present document.

AI	Artificial Intelligence
API	Application Programming Interface
BPMN	Business Process Model and Notation
CI/CD	Continuous Integration and Continuous Deployment
DAO	Decentralized Autonomous Organization
DASH	Digital cash
DLT	Distributed Ledger Technology
EEA	European Economic Area
ETSI	European Telecommunications Standards Institute
EU	European Union
GDPR	General Data Protection Regulation
GPS	Global Positioning System
GT	Governance Token
GUI	Graphical User Interface
IA	Implementation Agreement
ICT	Information & Communication Technology
IPR	Intellectual Property Rights
IRP	Interface Reference Point
ISG	Industry Specification Group
MT	Membership Token
PDL	Permissioned Distributed Ledger
RT	Reward Token
SLA	Service Level Agreement
UI	User Interface
UX	User eXperience
VM	Virtual Machine

4 Definition of a DAO

4.1 Foreword

The acronym DAO stands for Decentralized Autonomous Organization. Each component of this term carries significant meaning:

Decentralized: The organization is managed through distributed governance mechanisms, enabling it to function without a central authority. Blockchain, Distributed Ledger Technology (DLT), and Permissioned Distributed Ledger (PDL) technologies often serve as the foundation for this decentralized governance.

Autonomous: The DAO operates independently, with minimal external intervention. While it may require external resources (e.g. power supply, communication infrastructure), it performs its core functions autonomously, resistant to undue external influence.

Organization: A DAO is a collaborative entity working towards goals defined collectively by its constituents, which may include individuals, groups, companies, or institutional bodies.

Key characteristics of a DAO include:

1) **Collective Ownership:** Multiple entities own the DAO. Ownership distribution may be equal or differentiated based on criteria such as investment amount, usage, or asset/token holdings.

2) **Shared Mission:** The DAO operates towards objectives collectively defined by its owners.

3) **Blockchain-Enabled Features:**

    a) **Automated Functionality:** Enabled through smart contracts.

    b) **Data Immutability:** Information is resistant to unauthorized changes and non-repudiable.

    c) **Trust Mechanism:** Facilitates trust between otherwise non-trusting entities.

While Bitcoin [i.1] is often cited as the first DAO due to meeting these criteria, the concept has evolved to encompass a broader range of applications beyond cryptocurrency.

4.2 Current definition

In its most abstract form, a DAO is a system enabling distributed decision-making, management, and ownership of assets.

Contemporary DAO implementations typically exhibit the following characteristics:

Single Ledger Technology: While any ledger type can be used (distributed or centralized (see note 1)), it has to support smart contracts - code executed automatically when specific conditions are met. Current implementations generally use a single ledger technology.

NOTE 1: The term "Centralized" in the context of this clause refers to the form of governance ("governed centrally") rather than to the technology of data storage/distribution ("distributed ledger" vs. "centralized ledger").

Non-proprietary Codebase: The code may be fully open-source (publicly accessible) or partially open (accessible to consortium members). Critically, no elements of the codebase should be proprietary, meaning executable by all but visible only to select entities.

NOTE 2: Some DAOs do operate with proprietary code, though this deviates from the ideal model.

4.3 Current Implementations

DAOs currently serve a diverse range of purposes. While an exhaustive discussion is beyond the scope of the present document, the following non-exhaustive list illustrates the variety of use-cases where DAOs offer decentralized alternatives to traditionally centralized services:

- Legal services (e.g. LexDAO).
- Venture capital (e.g. "The DAO", BitDAO).
- Crowdfunding (e.g. UkraineDAO, ConstitutionDAO, MakerDAO).
- Shared interests and collaboration (e.g. Decentraland, "Friends with Benefits", Aragon, PleasrDAO).
- Cryptocurrency trading (e.g. Uniswap, "Curve DAO", DASH).
- Investment (e.g. Aave, Compound, HeadDAO, GnosisDAO).
- Governance (e.g. Aragon).
- Entertainment and Media (e.g. Flufworld).
- Social Networking (e.g. Blockster).
- DAO-as-a-service (e.g. DAOstack, Aragon).

4.4 Differences between a DAO and traditional organizations

Table 1 outlines the key differences between DAOs and traditional organizations.

Table 1: Comparison between a DAO and traditional organizations

Aspect	DAO	Traditional Organization
Hierarchy and structure	Flat, controlled through democratic vote of all participants	Centralized hierarchy controlled by select entities
Speed of decision-making	Fast. Flat hierarchy allows parallel decision-making by all participants	Slow, sequential approvals following hierarchical structure
Voting	Any change requires agreement through democratic vote	Changes may be made by a single entity or subset without broader consent
Governance	The entire community governs DAO behaviour	Select entities govern organizational behaviour
Transparency	All actions and decisions are transparent and visible to all participants (in public DAOs: visible to everyone)	Limited visibility and transparency. Decisions may be confidential
Operation and event handling	Automated using smart contracts	Often requires manual processes

5 Recommendations to operate a DAO

5.1 Foreword

A DAO typically comprises several mandatory core components, which form its operational foundation. These components should include:

1) **Implementation Agreements:** Rules and protocols accepted and followed by all DAO participants.
2) **Governing Entity:** Responsible for executing governance tasks.
3) **Smart Contracts:** Control DAO behaviour by implementing and enforcing the Implementation Agreements.

4) **Data Models:** Define the structure of objects managed within the DAO.

5) **Processes:** Outline the actions and transitions of objects throughout their lifecycle.

6) **Ledgers:** One or more Blockchain systems capable of executing smart contracts.

7) **Codebase:** Used to configure smart contracts and develop applications within the DAO.

8) **Nodes:** Store and validate data, participating in consensus and governance activities.

The term "Objects" in this context may refer to physical items, virtual goods, tokens, cryptocurrencies, services, products or any combination thereof.

5.2 Key components of a DAO

5.2.1 Implementation Agreements

Implementation Agreements are collections of rules and protocols defining DAO implementation. These are detailed in clause 4.6.3.6.1(a) of ETSI GS PDL 012 [i.4].

5.2.2 Governing Entity

The Governing Entity performs governance tasks by defining rules and Implementation Agreements, ensuring compliance, and resolving conflicts when necessary. It is defined in clause 4.6.3.6.1(b) of ETSI GS PDL 012 [i.4].

5.2.3 Smart Contracts

Smart Contracts are executable code embedded in the Blockchain, operating under specific conditions. They are defined in ETSI GR PDL 004 [i.2].

5.2.4 Data Models

Data Models organize data elements and standardize their relationships to one another and to real-world object properties. They are defined in clause 4.6.3.14.3 of ETSI GS PDL 012 [i.4].

5.2.5 Processes

Processes are sequences of actions or events performed to achieve specific results. In a multi-entity environment, multiple processes may run in parallel within a single entity or across multiple entities. Entities may operate:

1) **Internal processes**: Occurring within an entity.

2) **Externally-facing processes**: Relying on input from or providing output to other processes, either internal or external.

While internal processes may not require adherence to standard data models or event sequences, externally-facing processes require alignment with other externally facing processes to ensure meaningful data exchange and consumption.

5.2.6 Ledgers

Ledgers form the core of the Blockchain paradigm. They consist of data blocks cryptographically linked, creating a dependency between each link and all previous links. This structure allows identification of data modification attempts in previous links, rendering the data in such ledgers highly resistant to tampering.

While terms like "immutable" or "tamper-proof" may not be absolutely accurate, and some definitions prefer "tamper-resistant," this approach generally considers Blockchain data more reliable and trustworthy than data stored in traditional, centrally-managed repositories.

NOTE: While the original Blockchain systems were structured as ledgers of blocks, hence the original term "Blockchain", the present document uses the term "Ledgers" as modern implementations of DLT and PDL are not limited to chain structures and can also be implemented using other topologies such as trees, matrices, graphs and others.

5.2.7 Codebase

5.2.7.1 Definition of a codebase

A codebase encompasses the code used to develop smart contracts, applications utilizing the Blockchain, and user (GUI) and machine interfaces (e.g. APIs) through which various objects and entities interact. A codebase typically employs one or more computer programming languages.

5.2.7.2 Open Source

Open-source codebases are available for select groups to view, copy, and use. These groups may include:

1) **"Everyone"** (stored in a repository allowing unrestricted access).

2) **Select groups meeting specific criteria** (e.g. project or DAO participants).

Most open-source projects allow participants to contribute code, typically managed through version control systems like Git.

Open-source code may be subject to licensing, which may impose certain restrictions on distribution, use, and modification. While such licenses may not necessarily involve costs, using licensed open-source code implies acceptance of the rules and procedures defined in the license.

Even if the open-source code has associated Intellectual Property Rights (IPR), the owners typically do not charge for its use. However, entities (such as developers or consultants) may charge for integrating such open-source code in environments where the owners/administrators lack the necessary development resources.

5.2.7.3 Proprietary code

Proprietary code is developed by individuals or groups who maintain IPR and do not expose it to users. In this case, all development is performed by the IPR owners, and no other contributions are permitted. The IPR owner may charge for the development, integration and use of such code.

5.2.8 Ledger type

5.2.8.1 Blockchain attributes

Blockchains typically differ in several key attributes, including:

1) Consensus protocol for verifying and agreeing on new blocks.

2) Algorithm for generating and appending block hashes.

3) Programming language for writing embedded smart contracts.

4) Programming language for Blockchain applications.

5.2.8.2 Functional division of Blockchain

Currently, most Blockchain projects, including those managing DAOs, use a single Blockchain type. Despite the common use of a single Blockchain, Blockchain functions can be conceptually divided into two main parts:

1) Consensus management: This aspect may differ between ledgers.

2) Data storage: Ideally, this should be ledger-independent and consistent across Blockchain types.

5.2.8.3 Multi-Blockchain scenarios

Clause 6.3 of the present document discusses scenarios involving multiple Blockchain types in a DAO. It suggests methods to:

1) Ensure data consistency and harmonisation across different ledgers.

2) Unify smart contract functionality despite the use of different programming languages.

5.2.9 Validating nodes

Validating nodes actively participate in consensus management by validating new block data and ensuring previous block hashes match their data. These functions maintain ledger data integrity. In a typical DAO, each participant operates a validating node or, at minimum, delegates voting rights to another participant's node to prevent voting power accumulation.

In multi-Blockchain DAOs, while each participant should operate or be associated with a validating node on at least one ledger, they need not operate nodes on all ledgers. Conversely, participants may operate validating nodes on multiple ledgers.

5.2.10 Non-validating nodes

Some Blockchains require additional node types for proper operation. In multi-ledger DAO scenarios, a crucial function of such non-validating nodes may be to align data across all DAO-operated ledgers. While a comprehensive list of such node types and functions is beyond this document's scope, DAOs may incorporate these nodes as needed.

6 Operating a DAO

6.1 Foreword

6.1.1 Current DAO Implementations

DAOs can take various forms. Currently, the most common DAOs are based on a single ledger and often utilize a single software/code development team.

6.1.2 Challenges of Single-Software/Code-development Implementations

DAOs based on a single ledger utilizing utilize a single software/code development team present several challenges:

1) They are dependent on a single development team.

2) If the code is proprietary and not accessible to DAO participants or the public, it potentially undermines the decentralized and autonomous nature of the DAO.

3) When code development and implementation are centralized and lack transparency, the organization may not truly function as a DAO.

6.1.3 Composite DAOs

Some DAOs consist of multiple stand-alone projects grouped together for efficiency and interoperability. This grouping can result in incompatibilities between:

1) Data models.

2) Ledger types.

3) Smart contracts.

4) Processes.

Such DAOs may need to develop interoperability between these diverse elements.

6.1.4 DAO Types Defined in this Document

The present document defines two generic types of DAOs:

1) **Single Ledger DAO:** Utilizes a single ledger (and its associated smart contracts) used by all participants and a single codebase.

2) **Hybrid DAO:** May involve multiple ledgers (and associated smart contracts), multiple codebases, multiple data models, and multiple processes.

6.1.5 Document Structure

The following clauses discuss these two generic types of DAOs in detail:

1) Clause 6.2 explores Single Ledger DAOs.

2) Clause 6.3 examines Hybrid DAOs.

6.2 Single Ledger DAO environment

6.2.1 Single Ledger considerations

6.2.1.1 Codebase

6.2.1.1.1 Components of the Codebase

In a single ledger environment, the codebase is used to develop various software elements:

1) Ledger software running on nodes, including consensus mechanism, data storage, and distribution.

2) Smart contracts embedded in the ledger and executing on each node.

3) Applications using the ledger and smart contracts, including program logic and UI/UX.

4) APIs for data exchange between system elements.

5) Data Models representing information types exchanged and stored.

6) Data model brokers enabling transcription between data models.

6.2.1.1.2 Sources of the Codebase

While there's only one ledger in a single ledger environment, the codebase may come from multiple sources. For example:

1) Ledger software might be open-sourced and generated by a consortium or a foundation or proprietary code from a commercial vendor.

2) Smart contracts could be developed by different developers using tools defined by the ledger developers.

3) Applications might be developed by yet another developer using different programming languages and APIs to interface with the ledger and users.

Clause 5.2.7 discusses the differences between open-source and proprietary code.

6.2.1.1.3 Implications of Single Codebase

The use of a single codebase results in:

1) All nodes behaving similarly.
2) Consistent UX/UI across the system.
3) Uniform data models.
4) Standardized process flows.

6.2.1.2 Nodes

In a single ledger environment, while all nodes run the same codebase, they do not need identical hardware and software configurations. At minimum, they should have sufficient resources to run the codebase and perform resulting communications and storage tasks. It is common to see single-ledger DAOs where some nodes are implemented on dedicated hardware in an office/data-centre while others run on VMs in public clouds. Codebase developers typically determine and may periodically update the minimum requirements.

6.2.1.3 Governance

6.2.1.3.1 Concept of Autonomy

The word "Autonomous" in DAO indicates that governance tasks are delegated to participants/owners rather than handled by a centralized entity.

6.2.1.3.2 Governance Mechanisms

As defined in clauses 5.2.1 and 5.2.2, governance is performed through:

1) Implementation agreements agreed upon by all participants.
2) A governing entity that handles:
    a) Development of implementation agreements
    b) Development and modifications of the codebase
    c) Consensus mechanisms for reaching agreement between participants

6.2.1.3.3 Evolution of the Governing Entity

Initially, the governing entity might be a group of elected founders developing initial rules and overseeing codebase development. Over time, it may become virtual, with tasks either automated or implemented through consensus mechanisms giving all participants a share in decision-making.

6.2.1.3.4 Key Tasks of the Governing Entity

Key tasks of the governing entity include:

1) Managing changes to policy or process through consensus.
2) Ensuring codebase modifications implement agreed changes.
3) Balancing fair, consensus-based decision-making with governance integrity by:
    a) Preventing hostile take-overs by single parties or destructive coalitions.
    b) Ensuring high participation levels in votes and contribution to the proceedings of the DAO to prevent meaningless consensus through rubberstamping.

6.2.1.3.5 Further Discussion

Governance is discussed in further detail in clause 7.

6.3 Hybrid DAO environment

6.3.1 Hybrid DAO environment considerations

Hybrid environments exhibit plurality in one or more aspects:

1) Multiple vendors are contributing to the codebase.

2) Multiple ledgers are used simultaneously.

3) Multiple platforms grouped and governed together using unified rules.

Such environments are more complex than single ledger environments, requiring sophisticated approaches to governance and adherence to mutually agreed architecture, technical and process specifications.

6.3.2 Types of Hybrid environments

6.3.2.1 Multi Ledger

A multi-ledger environment uses more than one ledger, possibly ledgers of different types. Reasons for this may include:

1) Regulatory or commercial limitations on certain ledger types for some participants.

2) Performance variations of ledger types in different network topologies, benefiting from multiple ledgers accommodating various scenarios.

3) Amalgamation of software elements developed by separate vendors through investment by different participants who wish to retain their investment.

4) Data.

6.3.2.2 Multi Vendor

A multi-vendor environment includes code developed by multiple vendors. The reasons for inclusion of multiple vendors may include:

1) Avoiding vendor lock-in by opening the development to multiple vendors, often accompanied by open-source principles, though such openness does not necessarily imply that the code is open to the general public and certain access and contribution limitations may still exist (e.g. code may only be accessible to consortium members).

2) Amalgamation and integration of software modules that were developed by different vendors (including open-source projects) into a cohesive package.

6.3.2.3 Multi codebase

While multiple vendors often imply multiple codebases, in this context, multiple codebases refer to different code elements performing similar functionality. Examples of such plurality may include:

1) Smart contracts in multi-ledger environments, where different ledger types may use different programming languages.

2) Software modules from different vendors/groups performing the same basic functionality (defined as minimum requirements) while offering unique added features.

Participants in multi codebase environments can choose from various codebases addressing specific needs while retaining minimal required functionality.

6.3.3 Multi-Ledger Interoperability

6.3.3.1 Challenges of interoperability

The primary challenge in a multi-ledger environment is achieving interoperability. Given that each ledger operates in a somewhat unique manner, software developers contributing code to a multi-ledger environment need to ensure the environment maintains functionality across its entirety.

Interoperability should thus be maintained at several layers.

6.3.3.2 Data Interoperability

The core functions of a distributed ledger involve data storage and computational tasks performed on that data. While the methods of data storage and computation may vary, the resulting data should remain consistent across the entire environment, regardless of the ledgers used. To achieve this, several objectives are important:

1) Utilization of a unified data model across all ledgers. If data models differ, they should be fully translatable bidirectionally between each pair of ledgers, potentially through a data model broker (often referred to as an "API Gateway").

2) Smart contracts or any other code manipulating data should perform computational tasks in a manner that produces consistent results across ledgers, regardless of their types, when operating on the same data.

6.3.3.3 Consensus interoperability

One of the most notable differences between ledger types is the consensus protocol. While consensus algorithms can be complex, their fundamental purpose is straightforward: reaching a Yes/No decision based on given information. In a multi-ledger environment, the key interoperability requirement is that different ledgers arrive at the same decision given the same data/information.

Several factors may cause consensus operations on different ledgers to produce differing results even with the same data/information:

1) Population differences across ledgers: As not all participants in an environment necessarily have nodes on all ledgers, certain ledgers may have a majority supporting a proposed decision while others have a majority opposing it.

2) Different quorum/delegate rules used to build consensus.

3) Different definitions of majority in different ledgers (e.g. one ledger may require > 50 % while another may require > 2/3^{rds} or > 75 %).

Multiple solutions may address these challenges. The governance of each environment should define a process to resolve such conflicts in a manner that represents all participants, regardless of the ledgers they use/operate.

One potential solution involves creating delegates of participants across all ledgers, ensuring all ledgers have voters representing all participants. In this scenario, when a participant lacks a node on a certain ledger, a representative would vote on their behalf on that ledger. Other solutions may exist or be developed in the future.

6.3.3.4 Smart-Contract interoperability

The operational requirements [i.2] of smart contracts include:

1) Triggering by an event (time-related, data-related, or external request).

2) Acting on data within the ledger and possibly additional external data available to them.

3) Producing specific results, either related to data (e.g. new data is produced and stored on the ledger) or related to other operations (e.g. sending a warning message or a node extracting itself from the consensus process).

The above operational requirements are realized through computer code using a programming language supported by the respective ledger.

These operational requirements are implemented through computer code using a programming language supported by the respective ledger.

As different ledgers use different programming languages and somewhat different methods to store and retrieve data, smart contracts cannot be directly ported from one ledger to another. They need to be tailored to each ledger specifically.

Defining the operational requirements in a neutral manner using clear specifications (rather than sample code) may assist developers in porting smart contracts across ledgers. Test suites may help ensuring the different permutations of smart contracts across ledgers yield the expected results given the same conditions (triggers, data).

In a multi-ledger environment, multiple versions of smart contracts may exist, triggered by the same events/conditions and performing the same functionality across different ledgers. Additional smart contracts may be necessary to ensure proper operations and interoperability through comparison of the resulting data produced on different ledgers.

6.3.4 Multi Vendor/Codebase

6.3.4.1 Aspects of Multiple vendors/codebases

6.3.4.1.1 Operational flow of a platform

The operational flow of a platform can be defined by three key vectors: data structure, platform architecture and processes.

6.3.4.1.2 Data Structure

The structure of data flowing through the platform is defined by a data model. This model outlines the attributes describing the data and the relationships between these attributes.

6.3.4.1.3 Platform Architecture

The architecture of the platform and its components/subcomponents describes:

1) The functions the platform delivers.

2) The division of these functions into components (functional blocks).

3) The interfaces through which these components/blocks interact with each other.

6.3.4.1.4 Processes

The processes that the components follow to fulfil the different functional requirements of the platform are sequences of events/actions occurring at different functional blocks under certain conditions. These processes can result in:

1) Generation of new data.

2) Transportation of data from one place to another.

3) Storage of data.

The following clauses discuss how these three vectors are affected in a multi-vendor/codebase environment.

6.3.4.2 Data Model alignment

Data models provide a disciplined method of describing data structure and relationships between different data components. Often, the same object and its characteristics can be described in multiple ways. For example, a location can be defined in absolute terms (such as GPS coordinates) or in relative terms (direction and distance from another point).

Different developers may choose different, and sometimes incompatible, ways to define certain data elements. This can result in data models that are challenging to align.

EXAMPLE: When one data model describes distance using the metric system and another uses the statute (or imperial) system, the value "1" of a unit of measurement may define a different distance, as one mile is longer than one kilometre. Such differences can be aligned relatively simply using a data-model broker (also known as an "API Gateway") that translates miles to kilometres through multiplication or division by a coefficient.

However, some data types may be more challenging to align across data models. For instance, location data might be represented as:

- A street address ("Number 1 main street").
- GPS coordinates.
- Relative locations ("650 metres north east of the previous location").

Furthermore, certain data models may omit details while others offer more granular information (such as floor, apartment, room, equipment rack number, elevation in the rack, physical port number, etc.).

In such cases, developers will need to agree on a representation that can be translated across data models. Without this, meaningful information may be lost when transferring information between ledgers using different data models. For lossless transfer, the data models do not need to be identical (same attributes in the same sequence), but it is expected to have all attributes on both data models compatible with a mapping mechanism (data model broker) that can re-order the attributes and perform necessary calculations (e.g. convert metric to statute) bidirectionally.

6.3.4.3 Architectural alignment

6.3.4.3.1 Functional Blocks

Functional blocks, as described in the introduction to this clause and in greater detail in ETSI GS PDL 012 [i.4], are elements that perform specific actions contributing to the proper function of the platform. In a multi-vendor/codebase platform, there may be more than one block performing the same action, each on a specific ledger/part of the platform.

While eliminating such duplicates and using a single functional block serving all ledgers and codebases might be desirable, it may not be practical or easy to implement. Consequently, it may be simpler to focus on aligning functionality rather than eliminating duplicates.

Alignment in functionality means that given the same ingress data and operational conditions, functional blocks with the same designated functionality will produce the same results, whether those are data, actions, or both. This requires agreement among developers on the definitions of the functionality of each functional block and the algorithms to be implemented. The actual implementation, in terms of code, may differ due to the specifics of the programming language used on the different codebases.

6.3.4.3.2 Interface Reference Points

Interface Reference Points ("IRPs") are defined in ETSI GS PDL 012 [i.4] and convey information among functional blocks and between functional blocks and other internal and external architectural elements. Such information exchange may be implemented using APIs or other means.

In a multi-ledger/codebase environment, these reference points would still exist, regardless of the respective codebases. However, some additional reference points, not currently defined in ETSI GS PDL 012 [i.4], may need to be defined to handle data exchange and alignment between ledgers and codebases.

6.3.4.4 Process alignment

6.3.4.4.1 Parallel Processing

In a multi-ledger/codebase environment, processes may run in parallel on different ledgers.

6.3.4.4.2 Process Expectations

It is expected that such processes:

1) Follow the same sequence of events/actions given the same ingress data.
2) Align in a manner that when one process depends on data or action/decision from a parallel process, it waits for such data to become available.

6.3.4.4.3 Process Perspectives

It is anticipated that a process is viewed from two angles:

1) A holistic view of the overall process incorporating all sub-processes and ledger/codebase specific processes.
2) A per-ledger/codebase view of the ledger/codebase specific process.

6.3.4.4.4 Inter-ledger Communication

Such alignment may require the development of signalling methods between ledgers to ensure:

1) Dependencies are handled properly.
2) The proper sequence of events is maintained overall.

6.3.4.4.5 Challenges and Considerations

6.3.4.4.5.1 Common Challenges

When implementing process alignment in a multi-ledger/codebase environment, several challenges may arise:

1) Timing discrepancies between ledgers.
2) Data format inconsistencies.
3) Varying processing speeds across different ledgers.

6.3.4.4.5.2 Potential solutions for developers to consider

####### 6.3.4.4.5.2.1 Synchronization Mechanisms

Implementing synchronization mechanisms to ensure consistency across ledgers. This could involve:

1) Time-stamping events.
2) Using distributed clock synchronization protocols.
3) Implementing wait states or holding mechanisms.

####### 6.3.4.4.5.2.2 Standardized Data Exchange

Establishing standardized data exchange formats to ensure consistency across ledgers. This might include:

1) Defining common data models.
2) Implementing data translation layers.

3) Using intermediary data formats for cross-ledger communication.

6.3.4.4.5.2.3 Adaptive Processing

Developing adaptive processing algorithms that can accommodate varying ledger speeds. This could involve:

1) Dynamic load balancing.
2) Prioritization of critical processes.
3) Implementing catch-up mechanisms for slower ledgers.

6.3.4.4.5.3 Additional Considerations

6.3.4.4.5.3.1 Scalability

Ensuring that the alignment solutions can scale as the number of ledgers or the volume of transactions increases. This involves:

1) Designing flexible architectures that can accommodate growth.
2) Implementing efficient data management strategies.
3) Considering horizontal scaling capabilities.

6.3.4.4.5.3.2 Security

Maintaining the security of data and processes across multiple ledgers and codebases. Key aspects include:

1) Implementing robust encryption methods for cross-ledger communication.
2) Ensuring consistent access control mechanisms across all ledgers.
3) Regularly auditing security measures to identify and address vulnerabilities.

6.3.4.4.5.3.3 Auditability

Implementing mechanisms to track and audit processes across different ledgers for compliance and troubleshooting purposes. This encompasses:

1) Creating comprehensive logging systems.
2) Developing cross-ledger audit trails.
3) Ensuring data immutability for audit records.

6.3.4.4.5.3.4 Fault Tolerance

Designing the system to handle failures in individual ledgers without compromising the overall process alignment. This includes:

1) Implementing redundancy mechanisms.
2) Developing fallback processes for ledger failures.
3) Creating automatic recovery procedures.

6.3.4.4.5.3.5 Interoperability

Ensuring seamless interaction between different ledgers and codebases. This involves:

1) Developing standardized APIs for cross-ledger communication.
2) Creating common protocols for data exchange.

3) Implementing translation layers for different data formats.

6.3.4.4.5.3.6 Performance Optimization

Optimizing the performance of cross-ledger processes. This includes:

1) Minimizing latency in cross-ledger communications.
2) Balancing loads across different ledgers.
3) Implementing caching mechanisms where appropriate.

6.3.4.4.5.3.7 Regulatory Compliance

Ensuring that the multi-ledger/codebase environment adheres to relevant regulations. This involves:

1) Implementing mechanisms to enforce regulatory requirements across all ledgers.
2) Ensuring data privacy and protection across the entire system.
3) Developing capabilities for regulatory reporting and compliance audits.

6.3.4.4.6 Testing and Verification

To ensure proper process alignment:

1) Comprehensive testing scenarios should be developed to verify alignment across all ledgers and codebases.
2) Regular audits of process execution may be necessary to identify and correct any misalignments.
3) Simulation of various load conditions can help identify potential bottlenecks or misalignments under stress.

6.4 Tokens

6.4.1 Understanding Tokens

Tokens play a crucial role in the governance and incentive structure of DAOs.

A token model aims to balance the interests of founding members, existing participants, and new entrants while ensuring the long-term sustainability and value of a DAO. As the DAO evolves, token economics may be adjusted through governance decisions to best serve the needs of the ecosystem.

DAOs typically utilize three main types of tokens:

6.4.2 Governance Tokens (GTs)

GTs are used for voting on DAO proposals and decisions.

GTs are allocated to founding members of the DAO, initial token offering participants, existing members based on a key defined by the governance at the design phases of the DAO. A certain number of GTs are reserved for future growth to be allocated to new members.

The key for distribution may depend on seniority of participants, the amount of investment made by participants, the amount of development resources participants have allocated for the DAO, and other factors.

Voting power is proportional to the number of tokens held by a participant.

GTs may be used as stake in DAOs that allow such use. Such stake may lead to earning of Reward Tokens but losing such staked GTs may result in the participant losing some or all of its voting power.

The key for distribution of GTs to new participants may be revised by the governance through consensus.

6.4.3 Membership Tokens (MTs)

MTs are used to grant access to DAO resources and services.

MTs are allocated to existing DAO members, sold to new members, and distributed through partnerships.

MTs are required to access certain DAO functionalities and services.

GTs may be used as stake in DAOs that allow such use. Such stake may lead to earning of Reward Tokens but losing such staked GTs may result in the participant losing access to DAO resources and services.

6.4.4 Reward Tokens (RTs)

RTs are used to incentivize contributions and participation in the DAO.

RTs are earned through ecosystem contributions, staking GTs/MTs, and participating in DAO activities.

RTs can be exchanged for services or converted to other token types.

6.4.5 Key considerations for token economics (tokenomics)

When initiating a DAO the governance allocates an initial supply of tokens of each type. E.g. A DAO may allocate a total supply of 100 million tokens of which 40 % are GTs, 40 % are MTs, and 20 % are RTs.

Initial monetary value (pricing) of tokens should reflect the priorities defined by the governance and may vary by token type. E.g. GTs may be priced higher to reflect governance value. MTs and RTs may be priced lower to encourage participation.

Inflationary/deflationary mechanisms may be implemented to accommodate market trends and to drive growth and preserve long-term value.

The governance should perform regular assessments and perform dynamic adjustments to optimize token distribution and incentives.

Governance participation may be limited to participants holding a minimum GT threshold.

7 Governance

7.1 Definition of governance

7.1.1 Key principles of Governance

7.1.1.1 Architectural concepts pf governance

Governance, as defined in ETSI GS PDL 012 [i.4], encompasses a comprehensive set of rules and tools that regulate the behaviour and function of a PDL Platform.

The implementation and enforcement of these rules occur through various Platform Services. The Governance Platform Services are depicted in Figure 1 herewith.

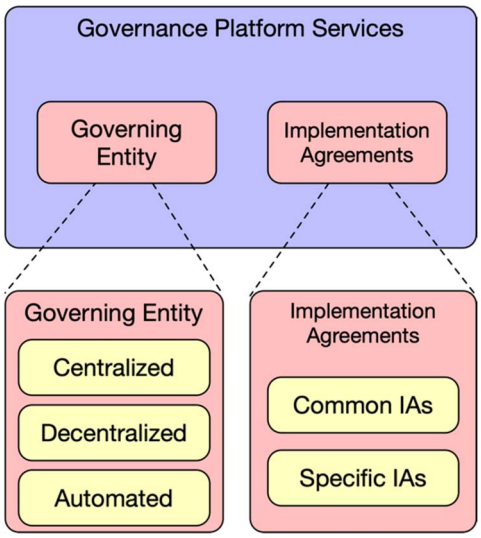

Figure 1: Elements of Governance

7.1.1.2 Decentralized Decision-Making

When a DAO reaches maturity it aims to decentralize decision making (see "Horizontal Governance" in clause 7.1.2.2 herewith).

- Power is distributed among members rather than concentrated in a central authority.
- Aims to reduce single points of failure and potential for abuse.
- Encourages diverse input and collective intelligence.
- Challenges include coordination and reaching consensus.

7.1.1.3 Token-Based Voting

- Governance tokens represent voting power.
- Can be acquired through various means (e.g. purchase, contribution, staking).
- Enables proportional representation based on stake or involvement.
- Raises questions of fairness and potential concentration of power.

7.1.1.4 Proposal Mechanisms

7.1.1.4.1 The purpose of proposals

- Allow any member to suggest changes or actions.
- Often include thresholds to prevent spam (e.g. minimum token holding).
- May involve multi-stage processes (e.g. informal discussion, formal proposal, voting).
- Critical for enabling bottom-up governance and innovation.

7.1.1.4.2 Proposal process

a) Submission: Members submit proposals for changes or actions.

b) Discussion: Community debates and refines proposals.

c) Voting: Token holders vote on proposals.

d) Execution: Approved proposals are implemented via smart contracts.

7.1.1.5 Smart Contract Execution

- Automated enforcement of voting outcomes.
- Reduces need for trust in human executors.
- Ensures transparency and immutability of decisions.
- Requires careful design to prevent exploits and unintended consequences.

7.1.1.6 Transparency and Immutability

- All governance actions are recorded on the Blockchain.
- Provides auditability and accountability.
- Helps build trust among members and with external stakeholders.
- Can pose privacy challenges for sensitive decisions.

7.1.1.7 Incentive Alignment

- Governance mechanisms designed to align individual and collective interests.
- Often involves rewards for participation (e.g. additional tokens).
- Aims to encourage long-term thinking and beneficial behaviours.
- Challenges include preventing short-term profit-seeking and "gaming" the system.

7.1.1.8 Adaptability

- Governance systems often include mechanisms for self-modification.
- Allows the DAO to evolve in response to new challenges or opportunities.
- Requires careful balance to maintain stability while enabling change.
- Often involves higher thresholds for fundamental changes.

7.1.2 Governance Models

7.1.2.1 Vertical Governance

In a single-vendor/codebase platform, governance typically follows a vertical model. Here, a single entity (often the developer or platform owner) defines and implements the operational rules. Even in PDL platforms where decisions may involve distributed consensus or majority voting, if a single entity manages the rules and their code implementation, the governance remains essentially vertical. Consequently, a PDL can maintain a centralized governance structure despite geographical distribution.

7.1.2.2 Horizontal Governance

Conversely, in a fully decentralized, horizontal platform, both the decisions and the decision-making processes are developed collaboratively and accepted by all participants.

7.1.2.3 Balancing Governance Models

A delicate balance exists between horizontal and vertical governance. Even fully decentralized, horizontally governed systems typically initiate from a foundational codebase a nucleus that enables the platform to commence operations and activate governance functionalities [i.3].

7.1.3 Purpose of Governance

The primary purpose of governance is to organize the powers, responsibilities, and liabilities related to achieving the Platform's objectives.

7.1.4 Governance Structure Objectives

7.1.4.1 Primary Aim

The governance structure aims to create a platform that is:

1) Efficient.
2) Secure.
3) Reliable.
4) Adaptable.
5) Sustainable.

7.1.4.2 Achievement Methods and Mechanisms

7.1.4.2.1 Risk Management

Timely identification and management of risks. This involves:

1) Regular risk assessments.
2) Development of risk mitigation strategies.
3) Implementation of risk monitoring systems.

7.1.4.2.2 Operational Control

Facilitation of controlled operational management. This includes:

1) Establishing clear operational procedures.
2) Implementing oversight mechanisms.
3) Ensuring adherence to defined protocols.

7.1.4.2.3 Stakeholder Protection

Protection of all stakeholders' interests. This encompasses:

1) Implementing fair and transparent decision-making processes.
2) Ensuring equitable distribution of benefits and responsibilities.
3) Establishing conflict resolution mechanisms.

7.1.4.3 Balancing Priorities

In pursuing these objectives, the governance structure should:

1) Balance short-term operational needs with long-term strategic goals.
2) Consider the diverse interests of all stakeholders.
3) Adapt to changing technological and regulatory environments.

7.1.4.4 Continuous Improvement

To maintain effectiveness, the governance structure should:

1) Regularly review and assess its performance.
2) Seek feedback from stakeholders.
3) Implement necessary adjustments and improvements.

7.1.5 Governance Functionality Aspects

7.1.5.1 Managerial Governance

This aspect oversees:

1) Platform development.
2) Service and application development (both new and existing).

7.1.5.2 Operational Governance

This aspect manages:

1) Service provision.
2) Participant management.
3) Voting processes.
4) Overall platform operation.

7.1.5.3 Hybrid Governance

Some governance functions may employ a hybrid approach, combining automated tasks with human intervention. To achieve true horizontal governance, human intervention should be minimized over time, although the transition to fully automated governance may be gradual.

7.1.6 Architectural elements

7.1.6.1 Implementation agreements

Implementation Agreements (IAs) serve as a fundamental architectural element of governance. These agreements comprise a collection of rules and protocols that define how Services are implemented and regulate the behaviour of the PDL platform. IAs specify the detailed methods and procedures for service implementation.

In a collaborative environment, participants of the PDL platform typically develop these agreements and rules jointly. Platform, service, and application developers then implement them. In single-vendor environments, the governing entity often prescribes these agreements, with the developer responsible for implementation.

Implementation Agreements fall into three categories:

1) **Common IAs and Rules**: These apply universally to all applications, users, and entities involved with a PDL platform. For instance, a PDL platform might mandate that all entities and applications utilize a specific Identity Service and adhere to particular security protocols.

2) **Common IAs with Specific Rules**: While these agreements are common across the platform, the specific rules may vary depending on the application or jurisdiction. For example, a PDL platform might require all entities and applications to use a specific Location Service, but different geographical regions may employ different methods to define location, and various applications may require different levels of location accuracy.

3) **Specific Implementation Agreements**: These IAs apply only to certain applications or jurisdictions, binding only the entities subject to such jurisdiction or using those applications. For example, EU/EEA entities are obliged to comply with GDPR; consequently, any application operated by an entity subject to EU/EEA jurisdiction has to employ an Implementation Agreement that aligns with GDPR requirements. Another example might be a requirement for all applications involving monetary transactions to use a specific encryption method as prescribed by the governance.

7.1.6.2 Governing Entity

7.1.6.2.1 Core Responsibilities

The Governing Entity plays a crucial role in performing governance tasks. Its primary responsibilities include:

1) Defining rules and Implementation Agreements.

2) Ensuring compliance with established protocols.

3) Resolving conflicts when necessary.

7.1.6.2.2 Additional responsibilities related to Establishment and Operation

Governance establishes:

1) Methods for forming the Governing Entity.

2) The entity's composition.

3) Procedures for defining and accepting rules and Implementation Agreements.

4) Methods for enforcing compliance.

5) Legally binding agreements to be signed by participating entities and individuals.

7.1.6.2.3 Types of Governing Entities

7.1.6.2.3.1 Centralized Governance

In this scenario, PDL participants choose or agree upon the entity or entities performing governance. When Centralized Governance involves multiple entities, decisions are reached through consensus among these entities.

7.1.6.2.3.2 Decentralized Governance

Here, governance comprises all PDL participants or a group of their representatives. The DAO executes governance tasks using PDL consensus mechanisms and established policies.

7.1.6.2.3.3 Automated Governance

This approach involves software (either pre-programmed or employing Artificial Intelligence) making decisions, reaching consensus, and enforcing policies on behalf of PDL participants.

7.1.6.2.3.4 Emerging Governance Structures

As the field evolves, additional governance structures may emerge. While acknowledging their potential existence, detailed exploration of such future structures falls outside the scope of the present document.

7.1.6.2.4 Governance Flexibility

The Governing Entity should maintain flexibility to:

1) Adapt to changing technological landscapes.
2) Respond to evolving regulatory requirements.
3) Accommodate growth in the number of participants or complexity of operations.

7.1.6.2.5 Transparency and Accountability

Regardless of the type of Governing Entity, it should prioritize:

1) Transparency in decision-making processes.
2) Clear communication of rules and policies to all participants.
3) Mechanisms for holding the Governing Entity accountable to the participants.

7.1.6.3 DAO governance

7.1.6.3.1 Basic Principles and aspects

7.1.6.3.1.1 Added complexity

DAO governance builds upon the principles outlined earlier, with the added complexity of plurality. Some key aspects are listed herewith.

7.1.6.3.1.2 Ledger Plurality

Unlike typical PDL platforms that use a single ledger type (mandating a single codebase and consensus protocol), a DAO may incorporate multiple ledger types. Each ledger may utilize a different codebase and consensus protocol. This diversity introduces new governance challenges:

1) Coordination between multiple ledgers and consensus protocols.
2) Alignment of information stored across different ledgers.
3) Achieving consensus-based agreement across all ledgers.

7.1.6.3.1.3 Development Plurality

As a consequence of ledger plurality, a DAO may encompass multiple parallel development efforts. While DAO governance may aspire to unify all development towards a single codebase, this goal may be impractical or even undesirable. Instead, the DAO can:

1) Define desired platform functionality across various use-cases and scenarios.
2) Specify the expected outcomes for each use case.
3) Task developers with creating code for respective ledgers that delivers the desired results.

7.1.6.3.1.4 Testing and Certification

A critical task for DAO governance involves testing and certifying code developed by various developers or vendors. This process ensures that the code yields expected results across all use-cases.

7.1.6.3.2 DAO component abstraction

7.1.6.3.2.1 Data Abstraction

Data may be stored and retrieved through various methods. However, after abstracting ledger-specific elements (such as hash algorithm, headers, and footers), the normative data in a block should remain consistent across different ledgers. This approach necessitates the use of unified (or fully interoperable) data models to represent information. As such data models should be abstract allowing data exchange across ledgers.

7.1.6.3.2.2 Consensus Abstraction

While consensus may be achieved through different mechanisms, the outcomes (in terms of approving or denying a transaction/proposal) should be uniform across ledgers. As such the consensus model should be abstracted separating the consensus protocol from the question and response.

7.1.6.3.2.3 Smart Contracts Abstraction

Smart contracts are typically written using ledger-specific programming languages. As automatic porting of such code between programming languages may not be feasible, smart contracts should be abstracted into universal methods such as:

1) Flow diagrams (e.g. BPMN).
2) State machines.
3) Sequence diagrams.

Developers can then implement these abstracts into ledger-specific smart-contract code. (Smart contract functional abstraction is discussed in depth in clause 7.3.4.2.)

7.1.6.3.3 Challenges in Consensus Abstraction

Abstracting consensus presents unique challenges, as different ledgers may employ varying consensus protocols based on different quorum and majority rules. This diversity introduces the risk of contradictory results under identical conditions. To address this, governance should consider:

1) Adjusting sample/population ratios to ensure consistent voter-to-population ratios across protocols.
2) Addressing population differences across ledgers by assigning cross-ledger delegates to represent participants lacking nodes on specific ledgers.

By abstracting data and consensus to a separate level, the DAO can perform tasks and ensure data uniformity in a multi-ledger environment, thereby maintaining the integrity and effectiveness of its governance structure.

7.1.7 Challenges and Considerations

7.1.7.1 Voter Apathy and Participation

7.1.7.1.1 Challenges

- Low turnout in voting can lead to decisions made by a small minority.
- Causes include complexity of issues, frequency of votes, and time commitment required.

7.1.7.1.2 Potential solutions

- Incentivizing participation through rewards (e.g. reputation scores).
- Implementing delegation systems.
- Simplifying voting processes and improving user interfaces.

7.1.7.2 Regulatory Compliance

7.1.7.2.1 Challenges

- Legal status of DAOs is often unclear or varies by jurisdiction.
- Challenges in compliance with existing corporate governance laws.
- Potential legal liabilities for DAO members.

7.1.7.2.2 Considerations and solutions

- Need for legal wrappers or hybrid structures.
- Engaging with regulators to develop appropriate frameworks.
- Balancing decentralization with legal requirements.

7.1.7.3 Security and Smart Contract Vulnerabilities

7.1.7.3.1 Challenges

- Risk of exploits in smart contract code.
- Potential for malicious proposals or attacks on voting systems.

7.1.7.3.2 Potential solutions

- Rigorous auditing, testing and certification of smart contracts.
- Implementing timelocks and multi-sig requirements for critical actions.
- Developing robust upgrade mechanisms for addressing vulnerabilities.

7.1.7.4 Scalability of Decision-Making

7.1.7.4.1 Challenges

- Difficulty in managing large numbers of proposals and votes.
- Potential for decision fatigue among members.

7.1.7.4.2 Potential solutions

- Implementing hierarchical or delegated decision-making structures.
- Using prediction markets or other mechanisms to filter proposals.
- Balancing between direct democracy and efficient governance.

7.1.7.5 Plutocracy and Concentration of Power

7.1.7.5.1 Challenges

- Risk of wealth concentration leading to disproportionate influence.
- Potential for vote buying or collusion among large token holders.

7.1.7.5.2 Potential solutions

- Implementing quadratic voting or other alternative voting systems.
- Capping voting power or implementing progressive taxation on large holdings.
- Separating governance tokens from economic rights.

7.1.7.6 Governance Attacks and Game Theory

7.1.7.6.1 Challenges

- Vulnerability to various attack vectors (e.g. 51 % attacks, governance takeovers).
- Complex game-theoretic considerations in incentive design.

7.1.7.6.2 Potential solutions

- Implementing time-locks and gradual power accrual mechanisms.
- Designing robust economic incentives aligned with long-term DAO health.
- Continuous modelling and simulation of governance mechanisms.

7.1.7.7 Cross-ledger Governance and Interoperability

7.1.7.7.1 Challenges

- Difficulties in coordinating governance across multiple Blockchains.
- Challenges in managing assets and voting power across different ecosystems.

7.1.7.7.2 Potential solutions

- Developing standards for cross-ledger governance.
- Implementing bridge mechanisms for governance tokens and voting.
- Exploring layer-2 solutions for more efficient cross-ledger coordination.
- See clause 7.3 for further details.

7.2 Governance in a Single ledger DAO environment

As demonstrated in the previous clause 7.1, governance in a single ledger DAO environment does not significantly differ from governance of other, non-DAO, PDL platforms. There may be significant differences compared to governance of non-permissioned DLT platforms, but that is beyond the scope of the present document and study.

The steps of initiating and maintaining a single ledger PDL-based DAO may include:

a) The founders of the DAO appoint a board that decides on the ledger to use, the applications to be included, the tokens to be used (where applicable) and a code development strategy (opensource, proprietary code, third party vendor, multiple vendors). Such board will typically include representatives of the founding members as well as subject-matter experts that may or may not have voting rights.

b) The DAO Board decides on membership policies, distribution of tokens, penalty policies.

c) The DAO gradually introduces automated membership management and reduces dependency on the board. This is the migration from *Vertical Governance* to *Horizontal Governance*.

d) The DAO maintains a software development strategy ensuring it addresses the needs of the participants. When multiple vendors are working in parallel the DAO will also need to define or adopt standard specifications and clearly defined processes and data models that the code generated by the vendors adheres to, to ensure interoperability between software modules developed by different vendors.

e) The DAO may have to develop a testing and certification strategy to ensure software interoperability across vendors/modules.

f) The DAO maintains a board that may now include members from non-founding entities. The DAO may define the election process and the eligibility of candidates.

g) The DAO implements the security and identity policies defined by the board, including changes thereof.

h) The DAO implements operational monitoring and control processes as defined by the board.

i) The board delegates certain decisions to the DAO membership through consensus votes. Governance through consensus is discussed in further details in clause 7.4 herewith.

7.3 Governance in a Hybrid DAO environment

7.3.1 Introduction

Governance in a hybrid DAO environment that includes multiple ledgers is more complex than governance of a DAO based on a single ledger because it has to include provisions for alignment of data across the different ledgers, alignment of voting across different consensus protocols and a software development strategy that abstracts the functionality of software modules from the underlying ledger and programming language.

7.3.2 Alignment of data across ledgers

As mentioned in clause 7.1.6.3.2 when examining a block of data in a ledger, it is important to distinguish between the normative payload (the actual application-related data stored in a block, also referred to as the body of the block) and the envelope data (hash, timestamp, Nonce, Merkle root and other ledger-specific data that is not part of the application data). This normative data has to be identical across the ledgers.

It is recommended that the processes and applications generating and processing data are abstracted from the underlying ledgers. Considering the PDL reference architecture defined in ETSI GS PDL 012 [i.4] the data should be generated and processed in the Application Layer or Platform Services Layer, translated to the ledger-specific format in the DLT-Abstraction Layer and finally stored in the DLT Layer.

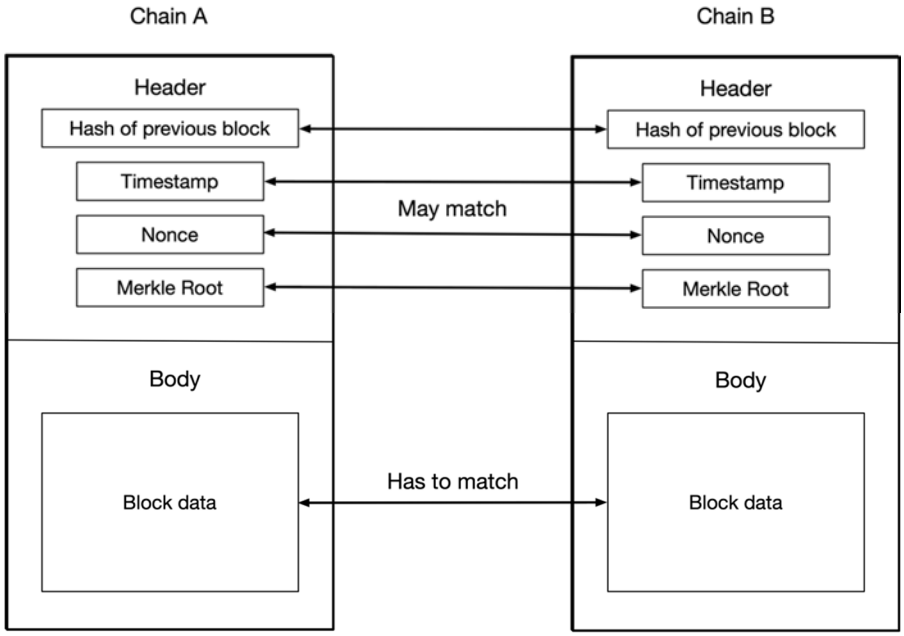

Figure 2: Data alignment across ledgers

7.3.3 Alignment of consensus protocols across ledgers

7.3.3.1 Basic Principles

7.3.3.1.1 Introduction

As mentioned in clause 7.1.6.3.2 consensus protocols vary in several ways. The following clauses explore these variations in detail.

7.3.3.1.2 Variations in Quorum and Majority Rules

Consensus protocols often use different quorum and majority rules. This may result in a scenario where two votes handled in parallel using two different consensus protocols yield different results. One vote may pass while the other might not.

EXAMPLE 1: One ledger might require a minimum quorum of two-thirds of the participants to vote for a vote to be valid, and a simple majority of > 50 % from within that quorum for a vote to pass. Other ledgers might require a smaller quorum of 20 % of the participants for the vote to be valid, but a majority of > 75 % for the vote to pass.

EXAMPLE 2: Certain consensus protocols assign voting rights to a group of delegates rather than requesting all members to vote.

7.3.3.1.3 Population Differences Across Ledgers

7.3.3.1.3.1 The cause for population differences

Consensus protocols are often based on different populations. In a multi-ledger DAO environment, not all participants participate in all ledgers. There can be several reasons for this.

7.3.3.1.3.2 Regulatory and Policy Constraints

Certain participants cannot use certain ledger types due to regulatory or internal policy reasons.

7.3.3.1.3.3 Varied Interest in Applications

Certain applications implemented using a specific ledger are not of interest to some DAO participants, who would thus refrain from installing a node on such a ledger.

7.3.3.1.3.4 Impact on Voting Patterns

Different populations may result in different voting patterns. Considering that a consensus in a DAO is not just about "is it true or false?" but may also be "do you approve this change or not?". While a true/false resolution is objective and is expected to offer the same result regardless of the population, a subjective resolution of whether a participant agrees to a proposed change may generate different responses among different populations.

Figure 3 below presents a hypothetical scenario where the general population is divided between those who like their tea with sugar, those who like their tea without sugar and those who do not like tea at all. Three groups are sampled from the general population, which vary in their composition. Each group is represented by a different ledger.

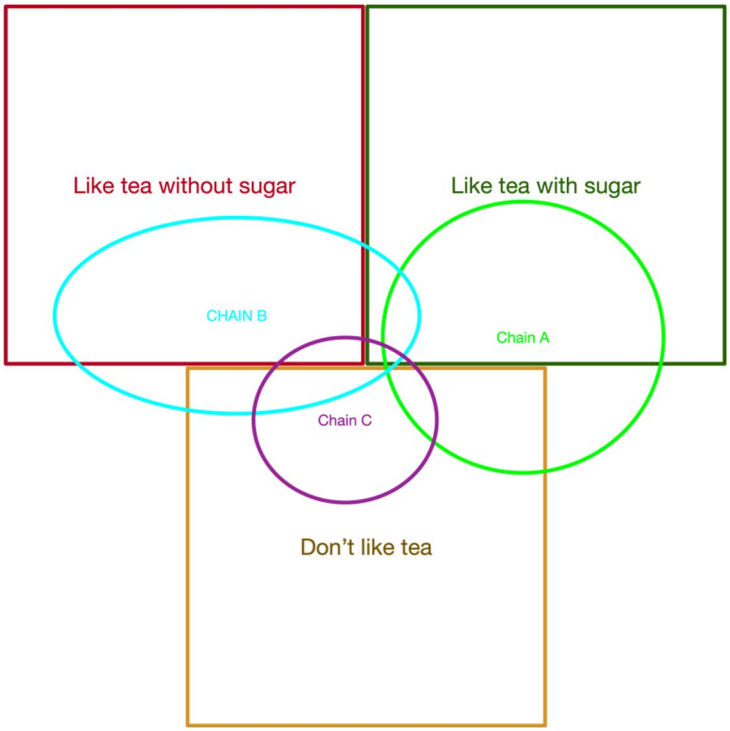

Figure 3: Population division across ledgers

- Group A (Ledger A) is divided between those who like their tea with sugar, those who do not like tea at all and those who do not have an opinion (do not belong to any of the rectangles).
- Group B (Ledger B) is divided between those who like their tea without sugar, those who like their tea with sugar, those who do not like tea and those who do not have an opinion.
- Group C is divided between those who likes their tea with sugar, those who like their tea without sugar and those who do not like tea at all.

For the purpose of this demonstration, it is assumed that the density of the population is equal in all parts of the diagram, meaning - a larger space represents more participants.

Table 2 presents several scenarios where the populations in Groups A, B and C are asked to participate in polls and respond based on their preferences. Note that group members who do not have a preference in certain polls are likely to abstain.

Table 2: Consensus scenarios

Scenario	Poll question	Group A	Group B	Group C
1	Should we add sugar to tea served at the office?	More than 60 % vote in favour. Those who do not like tea at all and those who do not have an opinion - abstain. Motion passed.	About 70 % vote against. Those who do not like tea at all and those who do not have an opinion - abstain. Motion failed.	Only than 10 % have an opinion about sugar (either for or against). The rest do not have an opinion or do not like tea. 90 % abstain. Vote invalid (no quorum).
2	Should we serve unsweetened tea at the office?	Off of the 60 % of the group's population who like sweet tea some may vote against serving unsweetened tea, but some may choose to be considerate and vote "for" on this poll considering that they can add sugar to their tea on their own. Others may just abstain. Motion in Limbo (may pass or fail or vote may be invalid).	About 70 % vote in favour. 10 % may vote against. 20 % do not have an opinion or do not like tea and will abstain. Motion passed.	Only than 10 % have an opinion about sugar (either for or against). The rest do not have an opinion or do not like tea. 90 % abstain. Vote invalid (no quorum).
3	Should we serve espresso at the office?	There is no information available about coffee preferences of the population so a prediction of the results cannot be made.		

7.3.3.2 Superset ledgers

Table 2 presented two scenarios where a subjective consensus vote across separate groups with a different blend of population yielded different results. A possible solution to such a scenario would be defining a superset ledger such that Groups A, B and C are merged into a new group D.

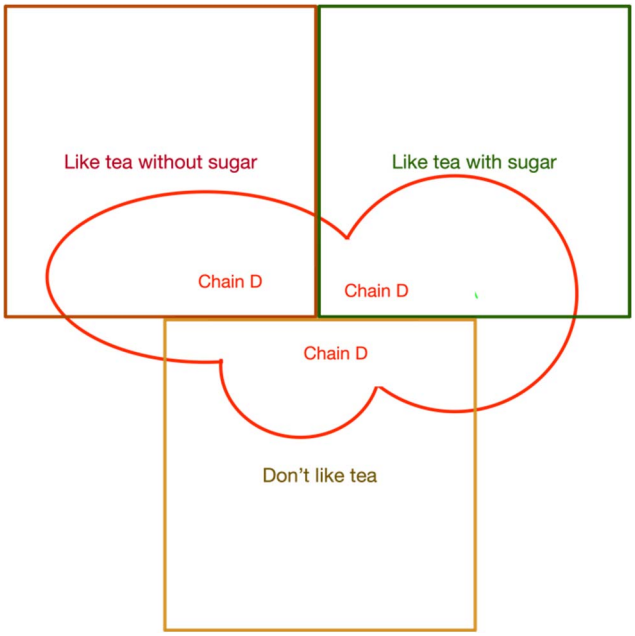

Figure 4: Merged populations

Had the poll questions be raised to the merged population (Ledger D) the results of the first two question (serve sweet or unsweetened tea) would probably be tied. There seems to be a slightly more participants who prefer sweet tea, so if the required majority is > 50 % they may win the vote, but their majority may not suffice if the consensus protocol requires a majority of > 66 % or > 75 %. In any both events - about 40 % of the potential voters will abstain so the validity of the vote depends on the number of participating voters so that they exceed the quorum requirements.

Merging ledgers into a superset is impractical in the real world as each ledger is operated using a different codebase on different nodes, thus a possible practical solution would be to create delegates of each participant in one ledger on ledgers where such participant does not have a voting node. In such manner Participant A(i) in Ledger A who does not have a node on Ledger B, will have a delegate Participant B(i) on Ledger B who will vote on their behalf. This approach generates equal presentation of all participants in all ledgers.

7.3.3.3 Majority and Quorum in superset ledgers

Creating a superset ledger may still not suffice to achieve equal representation as the majority and quorum requirements of different consensus protocols may differ. The present document does not offer a solution other than aligning those parameters through modifications to the code. It should be the role of the governance to reach agreement between all participants, likely through traditional vote rather than PDL consensus, on the required quorum for a vote to be valid and required majority for a vote to pass. The developers will then need to implement such agreement into the logic of their respective ledgers even if the original quorum and majority requirements differ.

7.3.4 Alignment of Smart Contracts across ledgers

7.3.4.1 Introduction

Smart Contracts are written using ledger-specific programming languages. Applying a smart contract written in a specific programming language in a ledger that uses another programming language will generate errors. Certain tools exist that can port code written in one language to code written in a different language, but such tools have not yet reached sufficient maturity to allow them to automatically port such code and put it into production without prior testing, verification and certification that the ported code operates as expected.

7.3.4.2 Smart Contract Abstraction

In a multi ledger environment, smart contracts should be abstracted into methods such as flow diagrams (e.g. BPMN), state machines or sequence diagrams. The developers then implement such abstracts into ledger-specific smart-contract code.

Flow charts, BPMN charts, state machines and sequence diagrams provide a visual representation of the functionality of a system/module. They describe the flow of information and the sequence of events in different scenarios without writing a single line of code. They use agreed upon symbols and graphical representation of objects and actions in a manner that is abstracted from any underlying code.

Software tools exist that facilitate the generation of such diagrams. Some such tools even have the ability to turn a BPMN diagram into executable code (that may or may not be compatible with the programming language used by a ledger).

Such diagrams, combined with clearly defined data models, reference architecture, product and process specifications, provide developers the information they need to write code that implements the requirements.

7.3.4.3 Testing and Certification

7.3.4.3.1 Introduction to Testing and Certification

To ensure the code written by developers indeed meets the requirements and delivers the required functionality a test-suite should be developed, preferably combined with a certification program, to allow both developers and users to ensure the code meets all the requirements and performs as expected in all operational scenarios.

7.3.4.3.2 Testing

Testing should be performed against a test-suite designed to ensure the performance of a tested element meets defined criteria.

Such criteria should define the desired egress data/behaviour to be generated/demonstrated by the tested element for a given ingress data/condition.

Figure 5 below depicts the process of testing that a tested element generates the desired results given specific ingress data. At least one set of data pairs (ingress data vs. desired egress data) should be made available for the testing environment to be able to produce a "passed" or "failed" result.

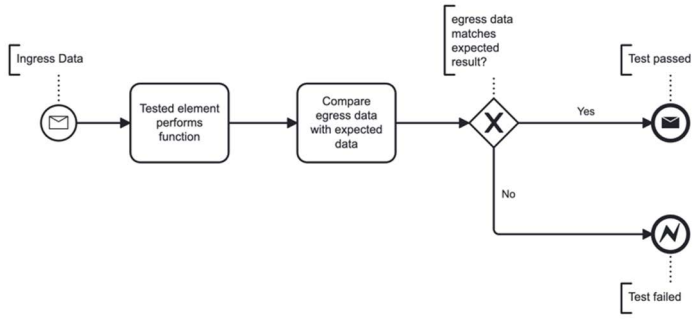

Figure 5: Ingress/Egress Data Testing Process

Figure 6 below depicts the process of testing that a tested element demonstrates the desired behaviour given specific condition. At least one set of data pairs (condition vs. behaviour) should be made available for the testing environment to be able to produce a "passed" or "failed" result.

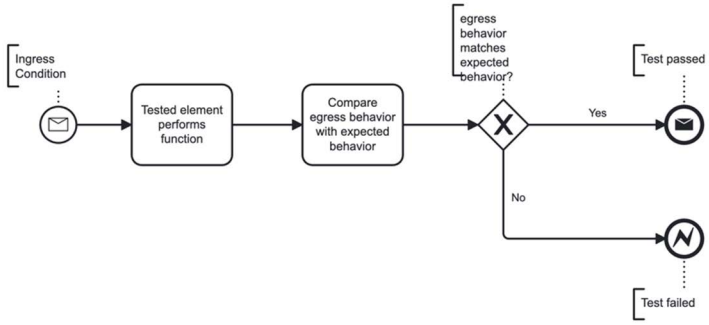

Figure 6: Condition/Behaviour Testing Process

7.3.4.3.3 Certification

7.3.4.3.3.1 Certification Process

a) Certification is a process where testing of one or more tested elements is performed based on a test suite that includes:

   1) A list of tests that should be performed.

   2) At least one Ingress/Egress pair for each data testing to be performed.

   3) At least one Condition/Behaviour pair for each behaviour test to be performed.

b) The test suite may define which of the tests are mandatory and which are optional.

c) The test suite may define which of the tests mandates a "passed" condition for the certification to be granted.

d) During the certification process all mandatory tests and any number of optional tests are performed.

e) The results of such tests are analysed and should all mandatory tests that also require a "passed" result have indeed passed - the tested element is considered "Certified".

f) Two or more tested elements may be bundled into a bundled element and tested as a bundle.

g) Should such bundle pass all mandatory tests - each of its underlying elements is considered "Certified".

7.3.4.3.4 Entities performing Testing and Certification Testing

7.3.4.3.4.1 Self Testing

If permitted by the governance, an entity may perform self-tests and publish the results in a location and a format defined by the governance.

7.3.4.3.4.2 Testing by a designated test entity

Should the governance so require, an entity seeking to test its tested elements has to approach a designated testing entity, approved by the governance, that will perform the tests. The testing entity may charge a fee for such tests.

7.3.4.3.4.3 Certification by a designated certification entity

Should the governance so require, an entity seeking to certify its tested elements has to approach a designated certification entity, approved by the governance, that will perform the certification tests. The certification entity may charge a fee for such certification tests.

7.3.4.3.4.4 Self Certification

If permitted by the governance entities may certify themselves by performing the test suits and certification process on their own and publishing the results in a location and a format defined by the governance. In such an event it is recommended that such certification process involves additional participants in the DAO (e.g. immediate network neighbours that can attest that the results reported by the self-certifying entity are indeed visible by them) and is subject to PDL consensus to avoid fraudulent self-declarations.

7.4 Governance through consensus

7.4.1 On-ledger consensus

On-ledger consensus refers to the process of reaching agreement through mechanisms implemented on the DAO ledgers. On-ledger consensus is a core component of the DAO's governance structure, enabling a transparent, efficient, and decentralized decision-making processes.

In the context of a DAO, on-ledger consensus has the following key characteristics:

1) **Automation:** Consensus decisions are executed automatically through smart contracts on the DAO ledgers.

2) **Transparency:** All consensus activities are recorded on the DAO ledgers, making them visible and verifiable to all participants.

3) **Immutability:** Once consensus is reached and recorded on-ledger, it cannot be altered without leaving a trace.

4) **Voting Mechanisms:** The DAO uses token-based voting for on-ledger consensus. GT holders can propose and vote on decisions directly on the DAO ledgers.

5) **Smart Contract Execution:** The results of consensus voting trigger automatic execution of corresponding smart contracts, implementing the agreed-upon decisions.

6) **Decentralized Decision Making:** On-ledger consensus allows for decentralized governance where all token holders can participate in decision-making processes.

7) **Quick Implementation:** Decisions made through on-ledger consensus can be implemented rapidly once the voting period ends and the consensus is reached.

8) **Auditability:** The entire consensus process, from proposal to implementation, is recorded on-ledger, allowing for easy auditing and verification.

7.4.2 Off-ledger consensus

Off-ledger consensus refers to decision-making processes that occur outside of the DAO ledgers but are still part of the DAO's governance structure. Off-ledger consensus complements on-ledger processes, allowing the DAO to handle a wider range of governance scenarios and adapt to various operational needs. It is particularly useful for aspects of governance that require human judgment, extensive deliberation, or need to operate within existing legal and business frameworks.

In the context of a DAO, off-ledger consensus has the following key characteristics:

1) **Manual Processes:** Unlike on-ledger consensus, off-ledger processes may be handled manually through traditional methods.

2) **Flexibility:** Off-ledger consensus allows for more flexible and nuanced discussions that may be difficult to encode in smart contracts.

3) **Lower Cost:** It avoids Blockchain transaction fees, making it more cost-effective for certain scenarios.

4) **Confidentiality:** Sensitive matters and privileged information can be discussed/stored off-ledger, maintaining privacy and compliance with regulations where necessary.

5) **Broader Participation:** It can involve stakeholders who may not hold tokens or have direct on-ledger representation.

6) **Preparatory Discussions:** Often used for preliminary discussions before formal on-ledger voting.

7) **Legal and Regulatory Compliance:** Certain legal or regulatory requirements may necessitate off-ledger processes.

8) **Complex Decision Making:** Suitable for decisions that require extensive debate, negotiation, or input from external experts.

9) **Governance Bodies:** The DAO may have off-ledger governance bodies like committees for specific functions (e.g. onboarding, technical, approvals).

10) **Integration with On-ledger Processes:** Results of off-ledger consensus may be brought on-ledger for formal ratification or implementation.

7.4.3 Multi-ledger consensus

7.4.3.1 Definition

Multi-ledger consensus refers to the process of achieving agreement across multiple PDLs within a DAO. This approach is crucial for ensuring interoperability and coherent decision-making in a diverse, multi-ledger environment. Some of the aspects related to this topic are discussed in the previous clause 7.3.

Key aspects of multi-ledger consensus include:

1) **Cross-ledger communication:** As defined in clause 7.3.2, implementing secure and efficient protocols for information exchange between different ledgers is crucial for the functionality of a multi ledger environment. It is suggested that in a multi ledger environment each block on each ledger has a replica block on all other ledgers containing the same block data. The ledger specific headers may differ.

2) **Consensus alignment:** As defined in clause 7.3.3, ensuring that consensus mechanisms on different ledgers can be reconciled or integrated is key for multi-ledger consensus to succeed.

3) **Interoperability standards:** Developing and maintaining standards that allow for seamless interaction between diverse Blockchain networks will enable the use of brokers and gateways as a means of inter-ledger interoperability.

4) **Conflict resolution:** Establishing mechanisms to resolve discrepancies or conflicts that may arise between ledgers.

5) **Scalability:** Designing the multi-ledger consensus process so it can accommodate growth in the number of participating ledgers and transaction volume.

6) **Security:** Implementing robust security measures to protect the integrity of cross-ledger consensus processes.

7.4.3.2 Voting and majority in a multi ledger environment

As discussed in clause 7.3.3 voting and establishing majority in a multi-ledger environment presents unique challenges and opportunities:

1) **Token representation:** Determining how governance tokens on different ledgers are represented and weighted in cross-ledger voting processes.

2) **Quorum requirements:** Establishing appropriate quorum thresholds that account for participation across multiple ledgers.

3) **Vote aggregation:** Developing mechanisms to accurately aggregate and tally votes from different ledgers.

4) **Supermajority considerations:** Implementing supermajority requirements for critical decisions that affect multiple ledgers.

5) **Ledger-specific vs. global voting:** Differentiating between decisions that affect specific ledgers and those that impact the entire multi-ledger ecosystem.

6) **Voting periods:** Coordinating voting periods across ledgers to ensure fair participation and timely decision-making.

7) **Delegate systems:** Implementing cross-ledger delegation mechanisms to enhance participation and representation.

8) **Verification and audit:** Ensuring transparency and verifiability of voting processes across all participating ledgers.

By carefully designing these voting and majority mechanisms, a DAO can maintain effective governance while leveraging the strengths of a multi-ledger architecture.

7.5 Governance without consensus

7.5.1 Scenarios

While consensus-based decision-making is a cornerstone of most DAO operations, there are scenarios where governance without consensus becomes necessary or beneficial. This clause explores alternative governance models that can complement or, in some cases, replace consensus-based approaches. By incorporating non-consensus governance models, a DAO can ensure efficient decision-making across a wide range of scenarios, from routine operations to complex, specialized issues. These approaches provide flexibility and resilience, complementing traditional consensus-based governance structures.

7.5.2 Delegated governance

7.5.2.1 Liquid Democracy

7.5.2.1.1 The need for delegates

Delegated governance (often also referred to as "Liquid Democracy") is a form of governance that combines elements of direct democracy and representative democracy. In the context of DAOs, it allows token holders to either vote directly on proposals (for votes taking place in a ledger they are part of) or delegate their voting power to other members that participate in ledgers they are not part of. It thus involves the appointment of trusted representatives to make decisions on behalf of the broader DAO community.

7.5.2.1.2 Key Principles

7.5.2.1.2.1 Topic-based Delegation

- Members can delegate differently for different types of decisions.
- Allows for specialization and expertise-based voting.

7.5.2.1.2.2 Instant Accountability

- Delegators can revoke delegation at any time.
- Encourages delegates to act in the best interest of their delegators.

7.5.2.1.3 Benefits in DAO Governance

7.5.2.1.3.1 Cross-ledger Participation

- Allows a participant to present their interests in a ledger they are not part of.
- Offers representation of all participants in all ledgers.

7.5.2.1.3.2 Expertise Utilization

- Allows members to delegate to those with more knowledge in specific areas.
- Can lead to more informed decision-making.

7.5.2.1.3.3 Scalability

- Helps manage large numbers of proposals and decisions.
- Reduces decision fatigue for individual members.

7.5.2.1.3.4 Dynamic Representation

- More flexible than traditional representative systems.
- Adapts to changing member preferences and DAO needs.

7.5.2.2 Human delegates

In this model, DAO members elect human representatives to make decisions on their behalf. The key aspects of the human delegate model are:

1) **Delegate election process:** Regular elections are held to choose delegates.
2) **Delegate Responsibilities:** Delegates are tasked with making informed decisions on complex issues.
3) **Delegate Accountability:** Mechanisms are in place to recall delegates who do not perform satisfactorily and hold delegates involved in malicious behaviour accountable to same.
4) **Delegate Transparency:** Delegates are required to be able to explain their decision-making processes and rationale.

7.5.2.3 Machines as delegates

As an alternative or complement to human delegates, machine learning algorithms or AI systems can be employed as decision-making agents. Such machines would typically offer the following characteristics:

1) **Data-driven decisions:** Machines analyse vast amounts of data to make decisions more objective and less susceptible to human bias.

2) **Rapid response:** Automated systems can make decisions quickly in time-sensitive situations.

3) **Bias mitigation:** Properly designed systems can help reduce human biases in decision-making.

4) **Continuous learning:** Machine delegates can improve their decision-making capabilities over time.

7.5.3 Expert-driven governance

In scenarios requiring specialized knowledge, governance can be delegated to subject matter experts in one or more forms as follows:

1) **Expert panels:** Formation of panels comprising industry experts and thought leaders.

2) **Advisory roles:** Experts provide recommendations to guide DAO decision-making.

3) **Peer review:** Implementation of peer review processes for critical decisions.

4) **Knowledge transfer:** Mechanisms to share expert knowledge with the broader DAO community.

7.6 Automated governance

7.6.1 Introduction to Automated Governance

Automated governance leverages technology to streamline decision-making processes, reduce human intervention, and increase efficiency within the Connectivity DAO. This approach relies on smart contracts, algorithms and predefined rules to execute governance functions automatically.

7.6.2 Recommendations

7.6.2.1 The components and minimal requirements

For automated governance to function effectively, certain foundational elements need to be in place. These recommendations ensure consistency, interoperability, and seamless execution of automated processes across the DAO ecosystem.

By establishing these minimum requirements, a DAO can create a solid foundation for automated governance. This infrastructure enables efficient, consistent, and secure execution of governance functions across the entire ecosystem, paving the way for more advanced automation and scalability in the future.

7.6.2.2 Unified data models

Unified data models are essential for automated governance to interpret and process information consistently. They should offer the following features:

1) **Standardized data structures:** Implement common data formats and schemas across all DAO operations.

2) **Semantic interoperability:** Ensure consistent interpretation of data across different systems and ledgers.

3) **Versioning system:** Maintain a robust versioning system for data models to manage updates and backwards compatibility.

4) **Data validation:** Implement automated checks to ensure data adheres to the unified model.

5) **Cross-ledger compatibility:** Design data models that can be translated across different Blockchain environments.

7.6.2.3 Unified processes

Unified processes ensure that automated governance actions are executed consistently and predictably. They should offer the following features:

1) **Standardized workflows:** Define common processes for recurring governance tasks.
2) **Process templates:** Create reusable templates for governance processes that can be customized as needed.
3) **Event-driven automation:** Implement triggers that automatically initiate relevant processes based on predefined events.
4) **Audit trails:** Maintain comprehensive logs of all automated processes for transparency and troubleshooting.
5) **Escalation pathways:** Define clear procedures for when automated processes require human intervention.

7.6.2.4 Unified IRPs

Unified Interface Reference Points (IRPs) are crucial for seamless interaction between different components of the automated governance system. They should offer the following characteristics:

1) **Standardized API assignment:** Develop and maintain a set of standard APIs for all governance-related interactions and define which IRPs are used by which API.
2) **Protocol-agnostic interfaces:** Design IRPs that can function across different Blockchain protocols and networks.
3) **Security standards:** Implement robust security measures for all IRPs to protect against unauthorized access or manipulation.
4) **Version control:** Maintain strict version control for IRPs to ensure compatibility and manage updates.
5) **Documentation:** Provide comprehensive, up-to-date documentation for all IRPs to facilitate integration and troubleshooting.
6) **Testing framework:** Develop a rigorous testing framework to validate the functionality and security of IRPs.

7.6.3 Algorithmic governance

Certain governance functions can be automated through algorithms and smart contracts. Such functions may operate without requiring prior consensus process:

1) **Predefined rules:** Establishment of clear, codified rules for routine decisions.
2) **Threshold-based actions:** Automatic execution of actions when predefined conditions are met.
3) **Dynamic parameter adjustment:** Algorithms that adjust system parameters based on real-time data.
4) **Fail-safes:** Implementation of checks and balances to prevent algorithmic errors or exploitation.

7.6.4 Codebase alignment

7.6.4.1 The need for alignment

Codebase alignment is crucial for ensuring the consistency, reliability, and interoperability of automated governance systems in a hybrid DAO ecosystem. This alignment facilitates easier implementation, maintenance, updates, and scalability of the DAO's technical infrastructure.

By focusing on these aspects of codebase alignment, a DAO can ensure that its automated governance systems operate cohesively and efficiently across its entire ecosystem. This alignment not only improves the current functionality but also provides a solid foundation for future growth and innovation in the DAO's technical infrastructure.

While implementing such alignment across diverse teams with different ownership and internal procedures may be challenging, alignment and cooperation are key for the success of a DAO and participants are expected to act accordingly.

Key aspects of codebase alignment are listed herewith.

7.6.4.2 Standardized development practices

- Adopting consistent coding standards and style guides across all development teams.
- Implementing uniform version control practices using tools like Git.
- Establishing common code review processes to maintain quality and consistency.

7.6.4.3 Modular architecture

- Designing the codebase with a modular structure to enhance reusability and maintainability.
- Developing standardized interfaces between modules to ensure smooth integration. Following the ETSI GS PDL 012 [i.4] reference architecture is key to the success of such effort.
- Implementing a plugin system to allow for easy extension of functionality without modifying core code.

7.6.4.4 Shared libraries and frameworks

- Creating and maintaining a set of shared libraries for common functionalities.
- Adopting consistent frameworks across different components of the DAO ecosystem.
- Implementing a centralized package management system for easy distribution and updates of shared code.

7.6.4.5 Cross-ledger compatibility

- Developing abstraction layers to ensure code can run on different Blockchain environments.
- Implementing adapters, bridges and brokers to facilitate interaction between different ledgers.
- Standardizing data serialization and deserialization methods across ledgers.

7.6.4.6 Continuous Integration and Deployment (CI/CD)

- Implementing automated testing and certification pipelines to ensure code quality and compatibility.
- Adopting consistent deployment processes across all environments.
- Utilizing containerization technologies like Docker for consistent runtime environments.

7.6.4.7 Documentation and knowledge sharing

- Maintaining comprehensive, up-to-date documentation for all code components.
- Implementing a knowledge base or wiki for sharing best practices and solutions.
- Conducting regular code walkthroughs and training sessions to ensure alignment across development teams.

7.6.4.8 Governance of code changes

- Establishing a clear process for proposing and approving changes to shared codebases.
- Implementing a voting mechanism for major code changes that affect multiple components.
- Creating a roadmap for code evolution and deprecation of outdated components.

7.6.4.9 Security alignment

- Adopting consistent security practices and protocols across all codebases.
- Implementing regular security audits and penetration testing.
- Establishing a uniform process for handling and patching security vulnerabilities.

7.6.4.10 Performance optimization

- Setting standardized performance benchmarks across different components.
- Implementing consistent profiling and optimization techniques.
- Establishing guidelines for resource usage and efficiency.

7.6.4.11 Interoperability testing

- Developing comprehensive test suites to ensure interoperability between different components and ledgers.
- Implementing integration testing environments that mirror the production ecosystem.
- Conducting regular cross-team testing sessions to identify and resolve compatibility issues.

7.7 Hybrid governance

7.7.1 The basics of hybrid governance

By combining different governance approaches a DAO can offer a balanced and flexible approach featuring one or more of the below:

1) **Tiered decision-making:** Using different governance models for various types of decisions.
2) **Escalation mechanisms:** Processes to elevate decisions from automated systems to human review when necessary.
3) **Consensus override:** Allowing for overriding consensus-based decisions in emergency situations.
4) **Adaptive governance:** Frameworks that can shift between governance models based on the DAO's evolving needs.

Hybrid governance in a DAO combines various governance models to create a flexible, robust, and adaptable system. This approach allows the DAO to leverage the strengths of different governance mechanisms while mitigating their individual weaknesses.

By implementing this hybrid approach, a DAO can create a governance system that is both efficient and adaptable, capable of evolving with the needs of the participants while maintaining the core principles of decentralization and community involvement.

7.7.2 Evolution of hybrid governance

Hybrid governance evolves over time, adapting to the changing needs of the DAO:

- **Initial phase:** Combines centralized, and often manual, decision-making with limited community input.
- **Growth phase:** Gradually introduces more automated decentralized elements and community participation.
- **Maturity phase:** Achieves a balance between automated, delegated, and community-driven governance.

7.7.3 Initiation

The initiation phase of hybrid governance focuses on establishing the foundational elements:

- **Core team:** A small group of founders or experts makes initial decisions.
- **Basic voting:** Introduction of simple voting mechanisms for key decisions.
- **Transparency measures:** Implementing systems for clear communication of governance processes.

7.7.4 Scale

As the DAO grows, the governance model scales to accommodate increased participation and complexity:

- **Automated voting:** Introduction of consensus based automated voting for select aspects of the operation of the DAO.
- **Delegated voting:** Introduction of delegate systems to manage increased member base and multitude of ledgers.
- **Specialized committees:** Formation of expert groups for specific domains (e.g. technical, financial).
- **Multi-tiered decision making:** Different governance mechanisms for various types of decisions.

7.7.5 Self sustained

The goal is to reach a self-sustaining governance model:

- **Automated routine decisions:** Use of smart contracts for day-to-day governance tasks.
- **Community-driven initiatives:** Empowering members to propose and lead new projects.
- **Incentive alignment:** Ensuring governance participants are rewarded for beneficial contributions.
- **Consensus based evolution:** Evolving the DAO based on consensus rather than centralized leadership.

7.7.6 Change management

Effective change management is crucial for the evolution of hybrid governance. Changes to consensus-based decision needs to be carefully managed to ensure it receives sufficient support from the membership. In certain scenarios a supermajority may be required.

Changes may require:

- **Feedback loops:** Regular assessment and adjustment of governance mechanisms.
- **Gradual transitions:** Phased implementation of significant changes to governance structures.
- **Education and onboarding:** Continuous efforts to inform and engage the community about governance processes.

7.8 Tokens in a Hybrid environment

7.8.1 The need for tokens in a hybrid environment

In a hybrid governance environment, tokens play a crucial role in aligning incentives, distributing power, and facilitating various governance mechanisms across different Blockchain networks and governance models. The Connectivity DAO leverages a multi-token system to support its hybrid governance structure.

By carefully designing and implementing these token mechanisms, a DAO can create a robust, flexible, and inclusive governance system that operates efficiently across its hybrid environment. This token ecosystem not only facilitates governance but also drives engagement, aligns incentives, and supports the overall mission of the DAO in a multi-ledger context.

7.8.2 Multi-token ecosystem

A hybrid DAO environment may seek to deploy multiple token types such as:

- **Governance Tokens (GTs):** Used for voting and proposal submission across all ledgers.
- **Membership Tokens (MTs):** Grant access to DAO resources and services.
- **Reward Tokens (RTs):** Incentivize participation and contributions.

Such token types are discussed in further details in clause 6.4.

7.8.3 Cross-ledger token compatibility

To ensure tokens can be used across different ledgers a DAO should:

- Implement bridge protocols to enable token transfers between different Blockchain networks.
- Develop standardized token interfaces to ensure consistent functionality across ledgers.

7.8.4 Weighted voting

7.8.4.1 The need for weighted voting

To ensure consensus votes represent a fair weight for each participant based on agreed-upon weighting criteria a DAO should:

- Implement a system where tokens from different ledgers may have different voting weights based on predefined criteria.
- Allow for dynamic adjustment of voting weights to balance power across the ecosystem.

7.8.4.2 Token-Weighted Voting

One approach to weighted voting, which is also the simplest and most trivial to implement, is "one token per vote". Meaning - a participant that owns a single token can make one vote while a participant with ten tokens can make ten votes. This voting approach has the following pros and cons:

- Pros: Aligns voting power with financial stake.
- Cons: Can lead to plutocracy.

7.8.4.3 Quadratic Voting

One approach to preventing plutocracy is Quadratic voting.

- Voting power is the square root of tokens held.
- Aims to balance influence between small and large token holders.

7.8.4.4 Reputation-Based Voting

7.8.4.4.1 Description of Reputation

Reputation-based voting is a mechanism that incorporates participants' contributions and behaviour into their voting power, rather than relying solely on token ownership. This approach aims to prevent participants with large size/turnover from controlling votes simply by staking large amounts of tokens.

7.8.4.4.2 Reputation Metrics

7.8.4.4.2.1 Factors to consider

####### 7.8.4.4.2.1.1 Behaviour

- SLA performance reputation.
- Voting reputation.
- Fraud reputation.

NOTE: See ETSI GS PDL 015 [i.5] for further details.

####### 7.8.4.4.2.1.2 Contributions

- Contribution of code.
- Development of processes.
- Making proposals for the development of the DAO.
- Allocation of development resources.

####### 7.8.4.4.2.1.3 Participation

- In votes.
- In discussions.
- In events (media, marketing).
- In industry groups (consortia, SDOs).

####### 7.8.4.4.2.1.4 Effect on voting power

These factors can be combined into a single reputation score or used as separate metrics affecting voting power.

7.8.4.4.2.2 Voting Power Calculation

- The number of Governance Tokens (GTs) is multiplied by a reputation score derived from the metrics defined above.
- This approach allows reputation scores to influence voting power or access to certain DAO functions.

7.8.4.4.3 Benefits

- By incorporating reputation into the voting mechanism, DAOs can create a more dynamic and engaged governance system that rewards ongoing contributions and responsible behaviour.
- Encourages active involvement in the DAO and reduces dormancy.
- Discourages fraudulent behaviour.

- Creates a more nuanced representation of a participant's stake in the DAO beyond simple token ownership.
- Allows reputation to influence a member's ability to attract new business (e.g. through improved SLA scores or financial ratings). See ETSI GS PDL 015 [i.5] for further details on this approach.

7.8.4.4.4 Implementation Considerations

To implement an effective reputation-based voting system, a DAO should:

- Develop a system where governance participation and positive contributions increase a member's reputation score.
- Implement transparent and fair mechanisms for calculating and updating reputation scores.
- Ensure that the reputation system is resistant to manipulation and gaming.
- Regularly review and adjust the reputation calculation algorithm to maintain its effectiveness and fairness.
- Consider implementing a decay factor for reputation to encourage continued active participation.

7.8.5 Liquidity provision

A token liquidity pool is a smart contract-based mechanism that holds reserves of two or more tokens, allowing users to trade directly against these reserves. Mobilizing tokens across ledgers may require lengthy processes. It is thus advised that a DAO follows the following guidelines:

- Create liquidity pools for token pairs across different ledgers to ensure token mobility.
- Implement incentives for liquidity providers (users who deposit tokens into the pool and earn fees from trades in proportion to their share of the pool) to maintain healthy token economics.

Liquidity providers may be DAO participants or external entities (e.g. banks).

7.8.6 Smart contract interoperability

In the context of tokenomics a DAO should:

- Design smart contracts that can interpret and act on token-related data from multiple ledgers.
- Implement oracle services to facilitate accurate cross-ledger token data exchange.

7.8.7 Adaptive issuance and burning

As the usage of tokens may evolve with time, a DAO should:

- Implement mechanisms to adjust token supply based on the needs of the hybrid governance system.
- Allow for ledger-specific token minting or burning while maintaining overall ecosystem balance.

This can be achieved by reserving sufficient tokens of each type in the initiation phase and managing the price of tokens going forward.

7.8.8 Governance-controlled parameters

To ensure tokens are managed in a fair and transparent manner the DAO should:

- Enable votes on key token parameters such as issuance rates, distribution, and utility across ledgers.
- Implement time-locks and multi-sig requirements for sensitive token-related decisions.

7.8.9 Privacy considerations

To ensure privacy is maintained across the token ecosystem the DAO should:

- Implement zero-knowledge proof systems (or similar) for token-related transactions where privacy is crucial.
- Allow for optional privacy features in token transfers and governance participation.

7.8.10 Emergency measures

Since Blockchain transactions cannot be easily reversed a DAO should minimize exposure to risks resulting from vulnerabilities and emergencies in a manner that will prevent or minimize their effects to turn into irreplaceable blocks in the DAO ledgers:

- Design circuit breakers and pause functions for token contracts in case of detected vulnerabilities.
- Implement a multi-ledger alert system for coordinated responses to token-related emergencies.

7.8.11 Token upgradability

To accommodate for future growth and market dynamics a DAO should:

- Design token contracts with upgrade paths to allow for future improvements.
- Implement a governance process for approving and executing token contract upgrades across ledgers.

8 Final Thoughts

8.1 Key Takeaways

Decentralized Autonomous Organizations (DAOs) represent a significant paradigm shift in organizational structure and governance, particularly for the telecommunications industry and broader ICT sector. By leveraging Blockchain technology, smart contracts, and innovative governance models, DAOs offer a revolutionary approach to collaboration, innovation, and service delivery on a global scale.

Key takeaways from this exploration of DAOs include:

1) **Adaptability:** The hybrid governance model, combining on-ledger and off-ledger processes, allows DAOs to evolve and adapt to changing industry needs and technological advancements. This flexibility is crucial in the rapidly changing telecommunications landscape.

2) **Inclusivity:** By incorporating various governance mechanisms, including consensus-based, delegated, and expert-driven approaches, DAOs encourage participation from a diverse range of stakeholders. This inclusivity can lead to more robust and innovative solutions.

3) **Efficiency:** Automated governance mechanisms and cross-ledger interoperability streamline decision-making processes and resource allocation, potentially reducing operational overhead and accelerating innovation cycles.

4) **Innovation Catalyst:** The open, collaborative nature of DAOs fosters innovation in both technical solutions and governance models. This is particularly valuable in an industry that requires constant technological advancement.

5) **Scalability:** The proposed multi-ledger and hybrid governance structures are designed to accommodate growth in both the number of participants and the complexity of operations, making DAOs suitable for large-scale industry collaborations.

6) **Regulatory Agility:** The flexible architecture allows for adaptation to varying regulatory requirements across different jurisdictions, a crucial feature for global telecommunications entities.

7) **Enhanced Security and Trust:** Through the use of distributed ledger technology and smart contracts, DAOs can provide increased transparency and immutability of decisions and transactions, fostering trust among participants.

8) **Economic Alignment:** The multi-token ecosystems and innovative voting mechanisms, such as reputation-based voting, create sophisticated incentive structures that can align the interests of diverse stakeholders.

9) **Interoperability Focus:** The emphasis on cross-ledger compatibility and data alignment positions DAOs as potential facilitators of industry-wide interoperability standards.

10) **Automated Efficiency:** The exploration of algorithmic governance and automated processes points towards a future of highly efficient, low-friction organizational operations.

8.2 Moving forward

As the ICT industry continues to evolve, DAOs provide a framework for collective progress, enabling participants to pool resources, share risks, and collectively address industry-wide challenges while maintaining individual competitive advantages. The success of such initiatives depends on active participation, ongoing refinement of governance processes, and a commitment to the shared vision of a more interconnected and efficient global ICT ecosystem.

Moving forward, it is crucial for industry players to:

1) Remain agile and open to feedback, continuously refining DAO structures and processes.

2) Invest in education and skill development to fully leverage the potential of DAOs.

3) Engage with regulators and policymakers to create supportive legal frameworks for DAO operations.

4) Prioritize security and privacy considerations in DAO implementations.

5) Foster a culture of collaboration and open innovation within the constraints of competitive markets.

DAOs are not just a technological solution, but a new paradigm for industry cooperation that has the potential to reshape the future of the global ICT ecosystem. By embracing this model, the ICT sector can drive unprecedented levels of collaboration, efficiency, and innovation, ultimately delivering enhanced value to customers and stakeholders worldwide.

As the industry embraces this transformative shift, the journey of implementing and refining DAOs in the ICT sector promises to be as challenging as it is exciting. The potential rewards - in terms of accelerated innovation, improved resource allocation, and more responsive industry structures - make this a journey well worth undertaking.

Annex A:
List of Recommendations for DAOs

Clause 5: Recommendations to operate a DAO

[O1] Processes **MAY** be running in parallel on different ledgers.

[D1] Processes **SHOULD** follow the same sequence of events/actions given the same ingress data.

[D2] Processes **SHOULD** be aligned in a manner that when one process is dependent on data or action/decision from a parallel process it waits for such data to become available.

Clause 6: Operating a DAO

[O2] The DAO **MAY** define the election process and the eligibility of candidates.

[O3] The DAO **MAY** have to develop a testing and certification strategy to ensure software interoperability across vendors/modules.

[D3] Smart contracts or any other code that manipulates data **SHOULD** perform computational tasks in a manner that produces the same results across ledgers, regardless of their types, containing the same data.

[O4] Additional smart contracts **MAY** need to be developed to ensure proper operations and interoperability through comparison of the resulting data produced on the different ledgers.

Clause 7: Governance

[O5] Certain governance functions **MAY** be governed through a hybrid approach involving both automated tasks and human tasks.

[O6] The DAO **MAY** define the desired functionality of the platform in as many use-cases and scenarios and the desired outcome of the platform for each such use case.

[D4] Smart contracts **SHOULD** be abstracted into methods such as flow diagrams (e.g. BPMN), state machines or sequence diagrams.

[O7] The test suite **MAY** define which of the tests are mandatory and which are optional.

[O8] The test suite **MAY** define which of the tests mandates a "passed" condition for the certification to be granted.

[O9] If permitted by the governance, an entity **MAY** perform self-tests and publish the results in a location and a format defined by the governance.

[O10] If permitted by the governance entities **MAY** certify themselves by performing the test suits and certification process on their own and publishing the results in a location and a format defined by the governance.

Clause 7.6: Automated governance

[D5] Unified data models **SHOULD** offer standardized data structures, semantic interoperability, versioning system, data validation, and cross-ledger compatibility.

[D6] Unified processes **SHOULD** offer standardized workflows, process templates, event-driven automation, audit trails, and escalation pathways.

[D7] Unified IRPs **SHOULD** offer standardized API assignment, protocol-agnostic interfaces, security standards, version control, documentation, and testing framework.

Clause 7.8: Tokens in a Hybrid environment

[D8] A DAO **SHOULD** implement bridge protocols to enable token transfers between different Blockchain networks.

[D9] A DAO **SHOULD** develop standardized token interfaces to ensure consistent functionality across ledgers.

[D10] A DAO **SHOULD** implement a system where tokens from different ledgers MAY have different voting weights based on predefined criteria.

[D11] A DAO **SHOULD** allow for dynamic adjustment of voting weights to balance power across the ecosystem.

[D12] A DAO **SHOULD** enable token staking across multiple ledgers to participate in governance.

[D13] A DAO **SHOULD** implement cross-ledger staking rewards to encourage long-term commitment.

[D14] A DAO **SHOULD** implement checks and balances to prevent a hostile takeover or fraudulent activity.

[D15] A DAO **SHOULD** create liquidity pools for token pairs across different ledgers to ensure token mobility.

[D16] A DAO **SHOULD** implement incentives for liquidity providers to maintain healthy token economics.

[D17] A DAO **SHOULD** develop a system where governance participation and positive contributions increase a member's reputation score.

[D18] A DAO **SHOULD** allow reputation scores to influence voting power or access to certain DAO functions.

[D19] A DAO **SHOULD** design smart contracts that can interpret and act on token-related data from multiple ledgers.

[D20] A DAO **SHOULD** implement oracle services to facilitate accurate cross-ledger token data exchange.

[D21] A DAO **SHOULD** implement mechanisms to adjust token supply based on the needs of the hybrid governance system.

[D22] A DAO **SHOULD** allow for ledger-specific token minting or burning while maintaining overall ecosystem balance.

[D23] A DAO **SHOULD** enable votes on key token parameters such as issuance rates, distribution, and utility across ledgers.

[D24] A DAO **SHOULD** implement time-locks and multi-sig requirements for sensitive token-related decisions.

[D25] A DAO **SHOULD** implement zero-knowledge proof systems (or similar) for token-related transactions where privacy is crucial.

[D26] A DAO **SHOULD** allow for optional privacy features in token transfers and governance participation.

[D27] A DAO **SHOULD** design circuit breakers and pause functions for token contracts in case of detected vulnerabilities.

[D28] A DAO **SHOULD** implement a multi-ledger alert system for coordinated responses to token-related emergencies.

[D29] A DAO **SHOULD** design token contracts with upgrade paths to allow for future improvements.

[D30] A DAO **SHOULD** implement a governance process for approving and executing token contract upgrades across ledgers.

History

Document history		
V1.1.1	February 2025	Publication