Authorization Object Network

A Distributed Object Propagation Network for
Authorized State Transitions

Mats Heming Julner

2026-06-06

Abstract

Modern coordination systems rely upon retained permission. Assets are deposited before trading, balances are maintained before settlement, and permissions are stored before future execution. In each case, coordination is achieved by allowing permission to persist beyond the specific action for which it was originally granted. Over time, retained permission accumulates into authority.

This paper explores an alternative model. Rather than embedding authorization within institutions, applications, custodians, or intermediaries, authorization is externalized into independently addressable objects. Participants publish explicit Authorization Objects describing permitted state transitions, execution constraints, validity requirements, and expiration conditions. Execution becomes the process of discovering, verifying, and consuming valid authorization relationships. The resulting architecture is termed an Authorization Object Network (AON): a distributed object propagation network for authorized state transitions. The network does not maintain balances, custody assets, establish global consensus, or determine canonical truth. Instead, it propagates authorization-related objects that allow independent participants to discover executable relationships across heterogeneous systems.

AON reduces coordination to four primitive object classes: Authorization Objects, Condition Objects, Proof Objects, and Receipt Objects. Complex workflows emerge through composition of these objects into executable authorization graphs. Coordination is achieved through discoverable relationships between explicit permissions rather than through institutions that retain permission on behalf of participants. The central claim of this paper is that authority is not the primitive underlying coordination. Authorization is. Authority emerges when authorization persists independently of the action for which it was granted. By externalizing authorization into independently addressable objects, coordination may occur without requiring permission to accumulate within the coordinating system.

1. Introduction

Many forms of coordination depend upon actions that cannot occur immediately. Settlement may depend upon delivery. Transfers may depend upon verification. Execution may depend upon

information that does not yet exist. Systems therefore require a mechanism through which permission can survive until the conditions required for execution have been satisfied. Historically, systems have addressed this problem by retaining permission until execution becomes possible (Lamport, 1978). As digital infrastructure has evolved, this pattern has become increasingly common. Institutions, applications, and protocols routinely maintain permissions on behalf of participants in order to coordinate future actions. The resulting systems differ substantially in implementation, but they share a common property: authorization persists beyond the action for which it was originally granted. This paper asks a structural question. Can coordination occur without requiring permission to accumulate?

We present Authorization Object Network (AON) as a minimal affirmative answer. Rather than embedding authorization within institutions, applications, or intermediaries, AON externalizes authorization into independently addressable objects. These objects may be propagated, discovered, verified, and consumed by independent participants without requiring any coordinating system to retain permission on their behalf. The protocol is built around four primitive object classes: Authorization Objects, Condition Objects, Proof Objects, and Receipt Objects. Together, these objects form authorization relationships from which execution may emerge. Coordination arises not from institutions that retain permission, but from the discovery and execution of valid authorization relationships.

AON does not determine truth, maintain ownership, or coordinate shared state. Truth remains local to the systems that produce it. State remains local to the systems that maintain it. The network propagates authorization-related objects and the relationships between them, allowing coordination to emerge without requiring a coordinating authority.

2. Authorization and Authority

Authorization and authority are often treated as interchangeable concepts. In practice, they frequently appear together. Systems that possess authority generally operate through authorization, while systems that manage authorization often accumulate authority over time. This paper treats the two concepts as distinct.

Authorization is permission for a bounded action. It identifies what may occur and the conditions under which it may occur. Authorization is specific rather than general. It applies to a particular action or class of actions and derives its legitimacy from the participant granting permission. Authority emerges whenever authorization persists independently of the action for which it was granted. Permission remains available for future use, allowing actions to occur without requiring a new authorization event. Execution becomes possible because authorization continues to exist. Authority is therefore not treated as a legal, institutional, or political concept. It is treated as a structural consequence of retained authorization.

This paper adopts the following definition:

Proposition: Authority emerges whenever authorization persists independently of the action for which it was granted.

Definition: Authority is authorization retained by the system.

Coordination systems frequently retain authorization because future execution depends upon conditions that may only become satisfiable later. Authority emerges because authority simplifies coordination. The question addressed by this paper is not whether authority exists, nor whether authority is beneficial or harmful. The question is whether coordination requires authorization to be retained at all. If authorization can remain attached to the action being authorized rather than becoming embedded within the coordinating system, then future execution may derive directly from authorization itself. Coordination would no longer require permission to accumulate within the institution performing coordination.

3. Authorization as an Addressable Object

Historically, authorization has existed as application state. Permission is stored within the systems coordinating activity and therefore inherits the properties of those systems. Permission becomes embedded within the coordinating institution rather than remaining attached to the action being authorized. As authorization persists, authority emerges. If authority is authorization retained by the system, then reducing authority requires more than distributing the institution that retains permission. Authorization itself must become independent of the systems that store it.

AON adopts this approach by externalizing authorization into independently addressable objects. An Authorization Object is a portable representation of explicit permission. It identifies the participant granting permission, the action being authorized, the conditions under which execution may occur, and the circumstances under which authorization ceases to be valid. The object does not execute actions, maintain balances, determine ownership, or enforce outcomes. It exists solely as a description of permission. For authorization to exist independently of institutions, it must possess an identity that does not depend upon those institutions. Authorization therefore becomes an addressable artifact capable of being propagated, referenced, verified, and consumed across systems without becoming embedded within them.

This transition follows a broader pattern within digital infrastructure. Concepts that were once embedded within systems gradually became independently addressable artifacts (Fielding, 2000). Documents became addressable resources. Digital assets became addressable artifacts. Authorization remains one of the few coordination primitives that is still primarily embedded within systems (Berners-Lee et al., 1994; Nakamoto, 2008). AON explores the possibility that authorization may undergo the same transition.

4. The Minimal Object Model

If authorization is externalized into independently addressable objects, a natural question follows: what information is required to express an authorized state transition?

The objective of AON is not to model every possible application directly. Exchanges, settlement systems, identity frameworks, and workflow engines all maintain application-specific state and logic. AON instead seeks the smallest set of primitives capable of expressing authorization

itself. Every authorized state transition depends upon four irreducible elements. Permission must exist for the transition to occur. Requirements may exist that determine whether execution is valid. Evidence may be required to demonstrate that those requirements have been satisfied. Finally, some record must exist indicating that execution occurred.

These elements correspond to four primitive object classes: Authorization Objects, Condition Objects, Proof Objects, and Receipt Objects. Authorization Objects establish permission. Condition Objects establish requirements. Proof Objects provide evidence. Receipt Objects record completed execution. Together, these primitives form the minimal structure required to express authorized state transitions. Additional abstractions may emerge through composition, but they derive from these objects rather than replacing them.

This minimality serves a structural purpose. Every additional primitive introduces new assumptions regarding how coordination should occur. The protocol therefore seeks to reduce coordination to permission, requirements, evidence, and history while avoiding application-specific assumptions about execution itself.

5. Object Identity and Content Addressing

For authorization to exist independently of institutions, authorization objects must possess identities that do not depend upon institutions. Traditional systems typically identify information through location. Identity therefore depends upon the system within which information resides.

This dependency is incompatible with portable authorization. If authorization is to become an independently addressable artifact, identity must derive from the authorization itself rather than from the institution storing it. AON therefore adopts a content-addressed model (Merkle, 1980; Benet, 2014). Each object possesses an identity derived from a canonical representation of its contents. The identity is not assigned by a network, application, or authority. It emerges directly from the object itself. Identical objects produce identical identities. Different objects produce different identities. The consequence is that authorization becomes independent of location. An Authorization Object may be created by one participant, propagated by another, indexed by a third, and consumed by a fourth while retaining the same identity throughout its lifecycle. The object does not belong to the systems through which it passes. Those systems merely transport it.

Content addressing also makes objects immutable. If an object’s contents change, its identity changes. The modified object therefore becomes a new object rather than a modified version of an existing one. Authorization remains stable, inspectable, and attributable throughout its lifetime. Objects rarely exist in isolation. Authorizations reference conditions. Conditions reference proofs. Receipts reference completed execution. Relationships between objects are therefore expressed through references to object identities. Coordination emerges not from shared state, but from relationships between independently addressable objects. The resulting structure forms a graph.

This graph is not assigned by a central authority. It emerges from independently created objects referencing one another through shared identities. Participants need only agree upon object construction and identity derivation. The coordination structure arises naturally from the resulting

relationships. Once authorization, requirements, evidence, and history exist as content-addressed objects, execution no longer depends upon a system possessing permission. Execution depends upon the existence of relationships between objects that collectively satisfy the requirements of authorization. The next section develops this idea and introduces Executable Authorization Graphs.

6. Executable Authorization Graphs

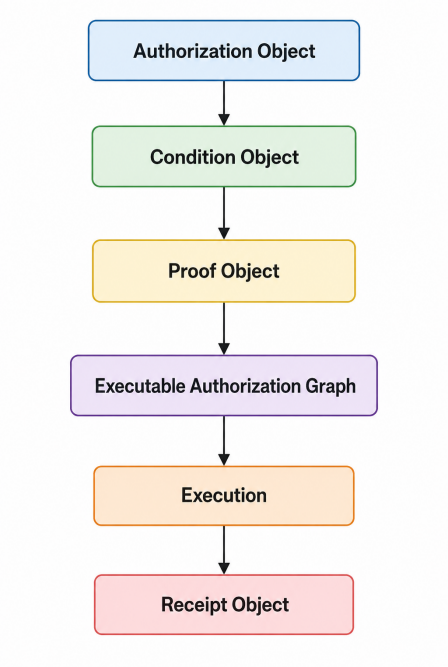

Figure 1. Minimal execution path within an Authorization Object Network.
Execution emerges when authorization, conditions, and proofs collectively form a satisfiable authorization graph. Receipt Objects record completed execution.

Authorization alone does not produce execution. An Authorization Object may describe a permitted state transition, but permission by itself is insufficient to determine whether execution should occur. Most authorizations depend upon requirements that must first be satisfied. Those requirements depend upon evidence. Execution therefore emerges not from individual objects, but from the relationships between them. AON treats execution as a property of a graph. Execution is therefore derived rather than assigned. Individually, these objects are insufficient to justify execution. Execution emerges through composition. An Executable Authorization Graph consists of an Authorization Object and the conditions and proofs required to satisfy it. When sufficient evidence exists to satisfy all required conditions, the graph becomes executable. Permission already exists. Requirements have been satisfied. Execution becomes valid.

This distinction is fundamental. Execution does not create legitimacy. Legitimacy already exists within the authorization. Execution merely consumes valid authorization whose requirements have been satisfied. Traditional coordination systems typically assign execution authority to institutions. AON adopts a different model. The network does not determine whether execution

should occur. It merely makes authorization relationships discoverable. Executability is a property of the graph itself. When sufficient objects exist to satisfy the requirements of authorization, execution becomes possible regardless of which participant discovers the opportunity. Coordination therefore shifts from assignment to discovery. Participants may independently observe the object graph and search for executable authorization relationships. These participants are referred to as executors. Their role is limited to discovering and exercising valid authorization relationships. Successful execution transforms the graph. Prior to execution, authorization represents executable potential. Following execution, a Receipt Object records completed execution. The graph has moved from possibility to history. Authorization, conditions, and proofs collectively represent the conditions under which execution may occur. Receipts represent completed execution.

The consequence is a coordination model in which participants need not surrender permission to a coordinating institution. They need only publish authorization. The network propagates authorization-related objects. Independent participants discover executable relationships. Execution emerges whenever sufficient evidence exists to satisfy the requirements of permission. Because authorization relationships ultimately depend upon facts originating outside the network itself, AON must distinguish between object propagation and truth production. This leads to the concepts of namespaces and local truth.

7. Namespaces and Local Truth

The previous section established that execution emerges when authorization relationships become satisfiable. This immediately raises a further question: who determines whether the proofs within those relationships are valid?

Many distributed systems answer this question through consensus. Participants collectively determine which facts are accepted and which version of history should be treated as authoritative. AON adopts a different approach. The network does not determine truth. Truth already exists. Proof Objects do not create the facts to which they refer. The underlying facts originate within external systems that remain responsible for determining their own validity. AON neither replaces nor supersedes these sources of truth. The protocol therefore treats truth as local.

A namespace identifies the environment within which a proof, condition, or authorization may be interpreted (Saltzer, Reed, and Clark, 1984). Meaning derives from the system that produced the underlying fact rather than from the network propagating the proof. Namespaces provide context, not authority. This distinction allows the network to remain neutral with respect to participating systems. Each namespace remains responsible for determining its own truths according to its own rules. Viewed structurally, AON coordinates relationships between state machines rather than state itself. Once truth remains local to namespaces, a further consequence follows. The network no longer requires global agreement regarding the facts to which authorization refers.

8. Why Consensus Is Unnecessary

Consensus occupies a central position within many distributed systems because those systems maintain shared mutable state. Ownership, balances, account histories, and state transitions require participants to converge upon a common view of reality. Consensus exists to resolve disagreement regarding state (Lamport, 1998; Nakamoto, 2008). AON operates under different assumptions. The protocol does not maintain ownership, balances, accounts, or application state. Truth remains local to namespaces. Authorization remains attached to Authorization Objects. Conditions, proofs, and receipts remain independently verifiable regardless of where they are observed.

As a consequence, the protocol does not possess the category of shared mutable state that consensus traditionally exists to protect. The network propagates objects and the relationships between them. It does not transform those objects into a globally authoritative state machine. Object validity does not depend upon global ordering. Participants may therefore possess different views of the object graph without creating conflicting realities. Visibility affects discoverability, not validity. State coordination seeks agreement regarding reality. Object propagation seeks discoverability of independently verifiable artifacts. AON belongs to the latter category.

The network does not establish canonical state. It makes authorization relationships discoverable. Consensus becomes unnecessary not because agreement is undesirable, but because the protocol does not coordinate the category of state that requires agreement.

9. Security Properties

Security requirements derive from the responsibilities of a system. AON operates under different constraints. The protocol does not maintain ownership, determine balances, execute settlement, or establish canonical truth. Those responsibilities remain local to the systems that produce them. As a consequence, many attack surfaces common to state-bearing systems are absent by design.

The primary objective of AON is to preserve the integrity of authorization relationships. The relevant security question is therefore not whether an attacker can alter ownership or seize assets through the network, but whether an attacker can create authorization relationships that appear legitimate without possessing legitimate authorization. The architecture addresses this concern through content-addressed identity and independent verification. Modification changes identity. Authorization remains directly attributable. Truth remains local to namespaces.

The network shifts verification from institutional possession of permission toward independent verification of authorization artifacts. Participants need not trust the systems propagating objects. They need only verify the objects themselves and the relationships those objects express. This does not prevent invalid information from existing within the network. Participants may publish invalid authorizations, misleading proofs, or arbitrary relationships between objects. The protocol does not attempt to prevent such artifacts from being created. Instead, it ensures that validity remains independently verifiable.

AON does not determine legitimacy. It propagates the artifacts through which legitimacy may be evaluated. The resulting security model follows directly from the architectural boundaries of

the protocol. Truth remains local to namespaces. Authorization remains explicit. Execution remains bounded by authorization.

10. Relation to Existing Systems

Authorization Object Network occupies a distinct position within the landscape of distributed systems. Traditional coordination systems operate by retaining permission on behalf of participants. These systems solve the problem of future execution by embedding authorization within the coordinating institution itself. Authority emerges because retained permission simplifies coordination. AON addresses a different question. Rather than coordinating through retained permission, the protocol explores whether coordination can emerge from explicit authorization that remains external to the coordinating system.

The objective is not to eliminate execution, settlement, or coordination, but to separate those functions from the accumulation of authority. Blockchains address a different problem. Systems such as Bitcoin and Ethereum maintain globally replicated state through consensus. Ownership, balances, and state transitions derive legitimacy from agreement regarding canonical state. Consensus is required because the network itself becomes the source of truth regarding those states. AON does not maintain ownership, balances, or application state. Truth remains local to namespaces. Authorization remains explicit. The network coordinates authorization relationships between state machines rather than maintaining a state machine of its own. Intent-based systems provide a closer comparison (ERC-7683; Flashbots, 2023). In such systems, participants express desired outcomes while independent actors compete to discover valid execution paths. AON shares this emphasis on discovery rather than centralized execution assignment. The distinction is that intent systems are primarily concerned with expressing what participants wish to occur, whereas AON is concerned with expressing what participants permit to occur. It introduces a coordination layer concerned specifically with authorization and the relationships that emerge from authorization.

11. Limitations

The objective of AON is deliberately narrow. The protocol seeks to coordinate authorization relationships between independent systems without requiring permission to accumulate within the coordinating layer. Many problems commonly addressed by distributed systems fall outside this scope. Most importantly, AON does not determine truth. Proof Objects may reference facts originating within external systems, but the network does not establish whether those facts are correct. Truth remains the responsibility of the namespaces that produce it. AON also does not maintain state. The protocol does not track balances, ownership records, account histories, positions, obligations, or application-specific state. Participating systems remain responsible for maintaining their own state and determining their own truths.

As a consequence, AON does not eliminate trust in external systems. Authorization relationships may depend upon proofs originating within blockchains, settlement systems, identity

frameworks, or other namespaces. The protocol may verify that a proof conforms to the rules of a namespace, but it cannot guarantee the correctness of the namespace itself. The protocol similarly does not guarantee execution. Valid authorization may remain unused. Executable authorization graphs may remain undiscovered. AON also does not prevent authority from emerging outside the protocol. Institutions may continue to retain permission. Participants may continue to delegate authority to intermediaries. The protocol demonstrates that authority is not structurally required for coordination within AON. It does not prevent authority from reappearing through arrangements external to the network.

Finally, AON does not attempt to replace the systems upon which it depends. The protocol is not a blockchain, settlement network, identity framework, custody system, or execution environment. It assumes the continued existence of systems capable of maintaining state, producing proofs, and determining truth within their respective domains. These limitations follow directly from the architectural boundaries established throughout this paper. AON externalizes authorization while leaving truth, state, and execution within the environments where they already exist.

12. Conclusion

This paper began with a simple question. Can coordination occur without requiring permission to accumulate? Modern coordination systems typically answer this question by retaining authorization. Assets are deposited before trading. Permissions are stored before future execution. Authority emerges because authorization persists beyond the action for which it was originally granted. This paper developed a different possibility.

Rather than embedding authorization within institutions, applications, or intermediaries, AON externalizes authorization into independently addressable objects. Historically, authorization has existed primarily as application state embedded within larger systems. AON explores the possibility that authorization itself may become an independently addressable network primitive. If authorization can remain attached to the action being coordinated, then independent systems may coordinate without requiring control to be transferred to the coordinating layer. Whether this model ultimately proves useful in practice remains an empirical question. The contribution of this paper is narrower.

Authority is authorization retained by the system.

If this observation is correct, then coordination and authority need not be treated as synonymous. AON explores the implications of that possibility.

References

Benet, J. (2014). *IPFS: Content Addressed, Versioned, P2P File System.*

Berners-Lee, T., Cailliau, R., Luotonen, A., Nielsen, H. F., & Secret, A. (1994). *The World-Wide Web.*

ERC-7683. (2024). *Cross-Chain Intents Standard.*

Fielding, R. T. (2000). *Architectural Styles and the Design of Network-based Software Architectures.*

Flashbots. (2023). *SUAVE: Single Unifying Auction for Value Expression.*

Lamport, L. (1978). *Time, Clocks, and the Ordering of Events in a Distributed System.* Communications of the ACM, 21(7), 558–565.

Lamport, L. (1998). *The Part-Time Parliament.* ACM Transactions on Computer Systems, 16(2), 133–169.

Merkle, R. C. (1980). *Protocols for Public Key Cryptosystems.* IEEE Symposium on Security and Privacy.

Nakamoto, S. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System.*

Saltzer, J. H., Reed, D. P., & Clark, D. D. (1984). *End-to-End Arguments in System Design.* ACM Transactions on Computer Systems, 2(4), 277–288.