WEBSHIELD INC. PROVISIONAL PATENT APPLICATION

Systems & Methods for Governed AI Coordination, Safety & Derivative Ecosystem Formation within a Quantum Privacy Network

Provisional Filing Date: May 19th, 2026 **Application #**: TBD

Applicant / Assignee: WebShield, Inc. (Delaware)

Inventors: Jonathan Paul Hare; Richard Arthur Muth

Type of Application: Provisional Application under 37 C.F.R. § 1.53(c)

Table of Contents

Cross-Reference to Related Applications

This application incorporates by reference the following United States patent applications and patents, each commonly assigned to WebShield, Inc.:

- U.S. Patent No. 12,316,610 B1, "Privacy Network and Unified Trust Model for Privacy Preserving Computation and Policy Enforcement" (granted May 27, 2025; Appl. No. 17/321,700, filed May 17, 2021; earliest priority March 16, 2016 via U.S. Provisional Application No. 62/309,153; 19 claims, 5 independent and 14 dependent), including its disclosures relating to Quantum

Privacy, Proof-of-Trust, Privacy Domains, Privacy Pipes, Trust Blocks, and the Unified Trust Model.

- U.S. Patent Application No. 19/206,859, "Quantum Privacy, Proof of Trust, and Privacy Network Exchange," filed May 13, 2025, a non-provisional continuation-in-part of U.S. Patent No. 12,316,610 B1.
- U.S. Provisional Patent Application No. 63/804,583, "Quantum Privacy, Proof of Trust, and Privacy Network Exchange," filed May 12, 2025 (Attorney Docket WEBS.08PRO), including its supporting documents entitled "WebShield QP-Drilldown for Provisional Patent Filing (2025-05-12)" and "WebShield Privacy Network — Global Quantum-Safe Cybersecurity Protection."
- U.S. Provisional Patent Application No. 63/895,861, "Privacy Network Exchange: Systems and Methods for Trust-Verified Tokenization and Settlement," filed October 8, 2025 (Attorney Docket WEBS.10PRO).
- U.S. Provisional Patent Application No. 63/923,253, "Systems and Methods for Quantum Privacy-Enabled Self-Funding AI Trust, Safety & Compliance," filed November 22, 2025.
- U.S. Provisional Patent Application No. 63/926,629, "Systems and Methods for Quantum Privacy-Enabled Personalized, Value-Based Universal Exchange for Better Health," filed November 27, 2025.
- U.S. Provisional Patent Application No. 63/931,387, "Systems & Methods for a Self-Funding, Self-Organizing Quantum Privacy Exchange and Accelerator Network," filed December 4, 2025.

The United States provisional patent application titled "**Systems and Methods for Quantum Privacy Catalyst Networks, Tokenized Derivative Settlement, and Governed Resource Inheritance**" (the "May 2026 QPN Catalyst Provisional"), filed concurrently herewith on or about May 18, 2026, commonly assigned to WebShield, Inc., which discloses Claim Families A through P of the integrated Quantum Privacy Network architecture. The present application is filed concurrently with, and incorporates by reference in its entirety, the May 2026 QPN Catalyst Provisional. The two applications disclose interoperable and complementary subject matter and together form a coordinated disclosure and priority framework for subsequent continuation and continuation-in-part applications.

Incorporation by Reference

The entire disclosures of each application and patent identified in the Cross-Reference to Related Applications section above are incorporated herein by reference in their entireties for all purposes, including for purposes of priority, written description, and enablement. In addition, the following supporting documents prepared in the ordinary course by the Applicant, to the extent they are or become available to the United States Patent and Trademark Office or are submitted herewith or by amendment, are incorporated by reference for purposes of enablement and written-description support: the QPN Context Primer; the QPN Catalyst Launch Plan & Rewards Framework; the QPN Participation, Valuation, Rewards & Financing Model; and the document titled "Quantum Privacy Network — Universal Access, Exchange, Ownership, AI & Abundance."

Where this application uses a term that is defined in any incorporated document — including without limitation Quantum Privacy, Quantum Privacy Domain (QPD), Privacy Algorithm, Privacy Domain, Trust Block, Trust Criteria, Trust Authority, Trust Taxonomy, Proof-of-Trust, Unified Trust Model, EasyAccess, Quantum Privacy Cell (QPC), QP Resource, Resource Derivative, Quantum DNA, Quantum Gene, Quantum Genome, Privacy Network Exchange (PNX), Exchange Network,

Resource Pool, Accelerator Network, Contribution Graph, Settlement Controller, Premium Framework, and Governed Agent Loop — that term carries the meaning given in the incorporated documents, as supplemented and extended by the present disclosure. To the extent of any conflict, the present disclosure controls for the subject matter expressly disclosed herein, and the incorporated documents control for subject matter disclosed therein. Nothing in this application is intended to narrow the scope of any term as used in the incorporated documents.

Relationship to the May 2026 QPN Catalyst Provisional

The May 2026 QPN Catalyst Provisional discloses an integrated Quantum Privacy Network (QPN) architecture organized as sixteen Claim Families, designated Families A through P, addressing contribution capture, artificial intelligence evaluation, settlement control, ledger architecture, contribution graphs and reputation, personal archival persistence, settlement and cross-verification, specialized capture vectors, governed agent loops, premium frameworks, behavioral activation and identity resilience, senior and junior QPT derivatives, stage-differentiated revert and monetization-uplift tracking, liquidity architecture, Quantum DNA and Genome inheritance, and two-parent and N-parent Personal Privacy Network reproduction.

The present application supplements and extends that architecture with ten additional Claim Families, designated Families Q through Z, together with omnibus integration material, definitions, operational flows, graph-theoretic embodiments, illustrative data structures and pseudocode, anticipated figure descriptions, and an initial claim set. The additional Families extend the QPN architecture into a more complete artificial-intelligence-native recursive coordination substrate, addressing governed derivative AI ecosystems and licensed distillation economies; full-lifecycle AI governance; distillation-resistant inference and output governance; resource-bound AI operational existence and constitutional execution; multi-laboratory partitioned AI coordination; AI-native economic coordination; sovereign and public-benefit AI governance networks; recursive accelerator and meta-governance systems; externality internalization; and contribution-graph venture formation.

The additional Families are intended to supplement and expand the prior Families A through P rather than to replace them. Each Family Q through Z embodiment may inherit from, interoperate with, extend, or be combined with any one or more of the Families A through P, and with the granted and provisional disclosures incorporated by reference. The Families are presented in a family-by-family structure consistent with the May 2026 QPN Catalyst Provisional. The Families may be implemented separately, in any combination, or as an integrated whole; the family organization is an aid to disclosure and is not a limitation. Counsel may, in any subsequent non-provisional or continuation-in-part application, renumber, reorganize, consolidate, or divide the Families and claims, and may collapse the Families into a unified narrative or expand them into separate applications, without disclaiming any subject matter disclosed herein.

General Recursive Embodiment and Anti-Narrowing Provisions

The following provisions apply throughout this disclosure. They are stated once here and are incorporated into each Family, embodiment, component, process, claim, and figure description by reference.

- **Recursive coordination generalization.** Wherever this disclosure describes a system, resource, environment, ecosystem, governance structure, settlement structure, accelerator, venture, or coordination relationship, that subject matter may be applied recursively. A derivative may itself act as an upstream resource for a further derivative; an accelerator may generate a derivative accelerator; a governed ecosystem may generate a derivative governed

ecosystem; and settlement, governance, attribution, and interoperability relationships may propagate across an unbounded number of derivative generations. References to "one or more" generations, "upstream" and "downstream" relationships, and "recursive" propagation are to be read as encompassing direct and indirect, single-generation and multi-generation, and partial and complete propagation.

- **Heterogeneous infrastructure generalization.** Wherever this disclosure describes a computational environment, infrastructure, substrate, ledger, domain, partition, or execution environment, that subject matter may be embodied across heterogeneous infrastructures, including without limitation sovereign infrastructures, enterprise infrastructures, cloud infrastructures, hardware-accelerator environments, secure enclaves and trusted-execution environments, distributed ledgers and hashgraphs, content-addressed storage networks, on-premises systems, edge devices, and combinations thereof. The same disclosed primitives are intended to function across any such infrastructure or combination of infrastructures.
- **AI governance generalization.** Wherever this disclosure describes an artificial intelligence system, model, agent, or AI-mediated process, that subject matter encompasses, without limitation, machine-learning models of any architecture, foundation models, frontier models, fine-tuned and distilled models, agentic systems, orchestration systems, multi-agent systems, synthetic-data generators, evaluation and interpretability systems, and combinations thereof, whether operated by a single party or by multiple parties, and whether operating autonomously, semi-autonomously, or under human direction.
- **Technical-computational framing.** The subject matter disclosed herein is machine-executed, governance-native computational infrastructure. Each disclosed system is a computer-implemented system comprising one or more processors, memory, and machine-executable instructions; each disclosed method is a computer-implemented method performed by one or more such systems; and each disclosed governance, settlement, authorization, attribution, or coordination operation is performed computationally through cryptographic data structures, governed execution environments, graph data structures, and machine-evaluated criteria. The disclosed architecture produces concrete technical effects in distributed computing systems, including reduced centralized trust requirements, cryptographically enforced execution constraints, traceable derivative-resource lineage, distributed authorization coordination, partitioned collaborative computation that preserves capability isolation, and synchronized recursive settlement. References herein to governance, authority, settlement, ownership, public benefit, or coordination describe the technical operation of machine-executed computational infrastructure and are not descriptions of abstract policy, organizational practice, or mental processes.
- **Anti-narrowing.** Throughout this disclosure, the terms "in various embodiments," "in some embodiments," "one or more," "at least one," "partially or wholly," "directly or indirectly," "recursively," and "or combinations thereof" are used to indicate that the described features are illustrative and non-limiting. The disclosure of a preferred or specific embodiment is not a disclaimer of broader or alternative embodiments. Component names, parameter ranges, numeric values, data-structure field names, state-machine state names, and pseudocode are illustrative examples provided for enablement and are not limitations. Any feature of any Family may be combined with any feature of any other Family and with any feature of the incorporated disclosures.

Background of the Invention

Field of the Invention

The present disclosure relates to computer-implemented infrastructure for governing artificial intelligence systems, their training corpora, their derivative models, and the coordination, authorization, and settlement of activity among them. More particularly, it relates to a cryptographically governed trust and resource substrate within which artificial intelligence operates as a first-class, governance-bound participant, and within which model weights, training data, intermediate computational states, evaluation outputs, inference flows, and derivative models exist as governed resources whose movement, reuse, and recombination are constrained by cryptographic enforcement rather than by policy, contract, or after-the-fact monitoring.

The Problems Addressed

The contemporary artificial intelligence industry faces a set of systemic problems that are widely recognized but that conventional architectures have not resolved, because conventional architectures treat each problem in isolation and address it with mechanisms that are evadable, that conflict with one another, or that suppress the very ecosystem they are intended to protect.

First, artificial intelligence safety is conventionally pursued as a problem of inference about a model's internal computations — of interpreting, predicting, or aligning what a model "intends." Safety guarantees obtained this way are only as strong as the inference, and the inference becomes less tractable as models become more capable. A safety regime that depends on correctly interpreting an opaque internal computation is structurally fragile.

Second, the four concerns that govern artificial intelligence — regulatory compliance, safety, ownership of and attribution to the data and models involved, and governance of how systems are operated — are conventionally treated as competing objectives. Mechanisms that advance one typically erode another: measures that protect a model restrict the parties who contributed to it; measures that satisfy a regulator constrain the developer; measures that preserve ownership impede downstream use. The result is a persistent, unresolved tension among data subjects, model developers, regulators, and downstream users, each of whom holds a legitimate interest that the architecture cannot simultaneously honor.

Third, the value created by a frontier artificial intelligence model does not arise principally from the inference call. It arises downstream — in orchestration, enterprise integration, domain specialization, synthetic-data generation, derivative-model formation, and recursive coordination across whole industries. Conventional monetization prices tokens, inference, or access, and therefore captures only a small and competitively eroding fraction of the value the underlying capability generates. The economic activity that artificial intelligence enables escapes the party that produced the capability.

Fourth, derivative-model formation — distillation, fine-tuning, orchestration, retrieval augmentation, synthetic-data generation, and recursive derivative formation — is treated as adversarial leakage to be prevented. The defensive response is to restrict weights, restrict outputs, centralize execution, and suppress derivative formation. This is rational in isolation, but the same measures that protect the model suppress the ecosystem that would otherwise form around it, and they are in any event

evadable: rate limits, contractual prohibitions, watermarking, and post-hoc monitoring do not structurally prevent extraction and do not create economic participation in what is extracted.

Fifth, the industry consequently appears to face a binary choice between keeping models tightly closed — forgoing the downstream ecosystem — and permitting uncontrolled proliferation — forgoing safety, attribution, and economic participation. Neither pole is satisfactory, and the architectures in common use do not offer a third option.

A Note for the Evaluator

An evaluator examining the technical disclosure that follows is cautioned that the disclosed architecture is not an incremental improvement upon, and should not be evaluated through the assumptions of, conventional access-control systems, digital rights management, identity federation, application-programming-interface gating, data-loss-prevention tooling, or permissioned-ledger platforms. Those systems govern access to a resource; the disclosed architecture governs the resource itself and every derivative recursively formed from it. Conclusions reached by pattern-matching the disclosed subject matter to those prior categories — for example, that the safety property reduces to access control, that tokenization implies data custody or transfer, that governance requires a central operator, or that derivative formation must be either permitted or forbidden rather than governed — will be incorrect, because the architectural properties on which the disclosure depends do not arise in those prior categories. The subsequent specification should be read on its own terms.

The General Approach

The disclosed architecture addresses the foregoing problems together, through a single cryptographically governed substrate, rather than separately. Its general approach has the following elements:

- **Artificial intelligence as a governed participant.** An artificial intelligence system, together with its model weights, training corpora, intermediate states, evaluation outputs, and inference flows, exists as a governed resource bound to one or more governed computational domains. Such a resource cannot cross a domain boundary without satisfying the Trust Criteria governing that boundary. The artificial intelligence system possesses resource-bound operational existence and dynamically governed operational authority, and its executable action-space topology may be cryptographically constrained prior to execution.
- **Safety as a structural property.** Because the executable action-space is constrained before execution, an action outside the governed envelope is not merely prohibited but computationally non-instantiable. Safety thereby rests on what a model can structurally do, not on what its internal computations are inferred to mean. A misaligned internal objective cannot produce a harmful action if the action path to that harm is cryptographically foreclosed. The disclosed architecture does not solve, and does not claim to solve, inner alignment or interpretability; it is directed instead at rendering the safety of the system independent of their solution.
- **Concurrent alignment of compliance, safety, ownership, and governance**. Because compliance is enforced at the domain boundary, safety by cryptographic constraint of the action-space, ownership by Trust-Block-mediated attribution flowing through derivative lineage, and governance by recursively inherited governance conditions, the four concerns are made concurrently enforceable through the same substrate. None is achieved by trading off the others; the data subject, the model developer, the regulator, and the downstream user are each first-class participants holding cryptographically enforceable rights in the same governed artifact.

- **Governed derivative formation in place of prevented extraction.** Distillation, fine-tuning, orchestration, specialization, synthetic-data generation, and recursive derivative formation are reframed from adversarial leakage to be prevented into governed economic derivatives to be authorized. A derivative resource inherits the Trust Blocks, Constitutional Guardrails, attribution lineage, operational constraints, and settlement-participation structures of the resources from which it is formed, and may recursively generate further derivatives that inherit in turn. Unauthorized extraction is structurally constrained while authorized derivative formation, controlled specialization, and recursive ecosystem expansion are permitted, with governance and settlement participation propagating recursively through downstream derivatives. The property that contains the resource and the property that monetizes it are the same property.
- **A third path between closure and proliferation.** The disclosed architecture provides an alternative to the binary of tightly closed models and uncontrolled proliferation: controlled openness within cryptographically governed environments, with inherited governance, revocable execution authority, and economically aligned participation. Mutually distrustful parties — including independent artificial-intelligence laboratories, enterprises, and sovereign participants — may collaborate across independently governed infrastructures while preserving the isolation of protected capability surfaces, including model weights and protected training data.

The subject matter summarized here is developed in technical detail in the Detailed Description and the Claim Families that follow, and in the May 2026 QPN Catalyst Provisional incorporated by reference, which discloses Claim Families A through P of the integrated architecture of which the Claim Families Q through Z disclosed herein form a part.

Abstract

Disclosed are computer-implemented systems and methods for recursively composable, cryptographically governed coordination across heterogeneous computational, organizational, sovereign, and artificial-intelligence ecosystems. The disclosed architecture treats an artificial intelligence system — its model weights, training corpora, intermediate activations, evaluation outputs, and inference flows — as governed resources bound to governed computational domains, such that the resource cannot cross a domain boundary without satisfying the Trust Criteria governing that boundary.

Governance continuity and settlement participation propagate recursively over graph data structures. A derivative resource may recursively generate further derivative resources while inheriting Trust Blocks, Constitutional Guardrails, attribution lineage, operational constraints, derivative obligations, interoperability conditions, and settlement-participation structures. An artificial intelligence system may operate as a governance-bound ecosystem participant having resource-bound operational existence and dynamically governed operational authority, and its executable action-space topology may be cryptographically constrained prior to execution, such that actions outside the governed envelope are not merely prohibited but computationally non-instantiable.

This architecture is directed to a structural treatment of artificial-intelligence trust and safety. Rather than depending on inference about a model's internal computations, the disclosed systems constrain what a model can structurally do: action paths are foreclosed cryptographically, governed outputs are bound to attribution lineage, and distillation-governance systems evaluate derivative-extraction risk and govern model outputs as governed derivative resources. Partitioned collaborative computation permits mutually distrustful entities — including independent artificial-intelligence

laboratories, enterprises, and sovereign participants — to collaborate across independently governed infrastructures while preserving isolation of protected capability surfaces, including model weights and protected training data.

The same architecture reframes derivative-model formation — distillation, fine-tuning, orchestration, specialization, retrieval augmentation, synthetic-data generation, and recursive derivative formation — from unauthorized extraction to be prevented into governed economic derivatives to be authorized, with governance inheritance and settlement participation propagating recursively through downstream derivatives. Compliance, safety, ownership, and governance are thereby made concurrently enforceable through a single cryptographic substrate rather than traded off against one another. In various embodiments, a distributed graph runtime, a recursive revocation and recovery engine, and federated synchronization and reconciliation machinery execute the disclosed coordination, authorization, governance-inheritance, and settlement operations across the governed ecosystems.

Family Q — Governed Derivative AI Ecosystems & Licensed Distillation Economies

Field Summary

This Family relates to computer-implemented artificial intelligence governance, and more particularly to systems and methods for governed derivative model formation, licensed distillation, recursive AI resource formation, and settlement-linked AI derivative ecosystems operating within a Quantum Privacy Network. It concerns the treatment of artificial intelligence models, their training corpora, their outputs, and their derivative artifacts as governed Quantum Privacy Resources bound to Quantum Privacy Domains under Trust Block enforcement, such that derivative AI systems inherit governance continuity, attribution lineage, and settlement participation from the upstream resources and sponsors that enabled them.

Problem Addressed

Conventional artificial intelligence platforms treat model outputs as unencumbered response artifacts, application-programming-interface access as a static commercial boundary, and derivative models as either unauthorized extractions or as the subject of separately negotiated bilateral licenses. Under these conventional approaches, when a frontier model is queried, fine-tuned, distilled, wrapped within an orchestration layer, or used to generate synthetic training data, the resulting downstream artifact typically retains no enforceable, machine-verifiable relationship to the upstream model, the training-data sponsors, the compute providers, or the other resources and parties that made the artifact possible. The relationship, if any, is contractual, after-the-fact, and dependent on the downstream party's continued cooperation and on the upstream party's ability to detect and prove a breach.

This produces several specific and unresolved technical problems. First, attribution is lost: there is no cryptographically verifiable lineage connecting a derivative model to its upstream resources, so it is not possible to prove, by inspection of the derivative artifact or its governing metadata, which upstream resources it depends upon. Second, governance continuity is lost: usage restrictions, safety constraints, jurisdictional limitations, and output-handling obligations applied to an upstream model do not automatically attach to a derivative model, so a derivative may be operated in contexts and in ways that the upstream governance never authorized. Third, settlement participation is lost: the upstream sponsors that bore the cost of producing the enabling resources have no enforceable, ongoing economic participation in the value generated by downstream

derivatives, which produces value leakage and a structural incentive toward unauthorized extraction. Fourth, because conventional architectures cannot solve the first three problems, frontier model sponsors are forced into a binary and undesirable choice between overly restrictive access — which suppresses beneficial derivative formation — and uncontrolled commoditization, in which the sponsor's investment is dissipated through unattributed and uncompensated derivative formation.

These problems are not addressed by conventional digital rights management, by API rate limiting, or by terms-of-service contracts, because none of those mechanisms attaches enforceable, recursively inheritable governance and settlement structure to a derivative artifact at the moment the derivative is formed.

Solution Overview

The disclosed architecture provides governed derivative AI ecosystems in which frontier models, their training corpora, their intermediate computational states, and their outputs exist as Quantum Privacy Resources (QP Resources) bound to one or more Quantum Privacy Domains (QPDs) under Trust Block enforcement. In various embodiments, a Quantum Privacy Domain is a governed computational environment, as described in the incorporated disclosures, within which resources and computations are bounded by Privacy Algorithms, Trust Criteria, Constitutional Guardrails, and Trust Blocks, such that the cryptographic boundary of the domain is also the governance boundary. An AI model registered as a QP Resource within a QPD is therefore not an unencumbered artifact; it is a governed resource whose permitted utilizations are determined by its associated Trust Block and the Trust Criteria of the domain.

A derivative AI system — whether produced by distillation, fine-tuning, orchestration, synthetic-data generation, evaluation, or recursive derivative formation from one or more other derivatives — is itself registered as a governed Resource Derivative. Consistent with the Trust Block inheritance mechanics disclosed in the incorporated documents, the Trust Block of a Resource Derivative incorporates links to the Trust Criteria of every upstream resource that contributed to it, so that upstream governance constraints are automatically and recursively inherited. Applied to derivative AI, this means a derivative model's Trust Block encodes, without limitation: upstream sponsor participation in settlement; downstream derivative-creator participation in settlement; the settlement waterfall governing how value routes among upstream and downstream participants; usage restrictions; output-governance obligations; jurisdictional and interoperability conditions; and permissions governing whether and how the derivative may itself be used to form further derivatives.

The effect is that a frontier AI laboratory, an enterprise model owner, a sovereign model sponsor, or any other model sponsor may authorize governed derivative formation without surrendering control of the upstream model and without forfeiting downstream economic participation. The sponsor may make a model available, inside one or more QPDs, for governed distillation, governed fine-tuning, governed synthetic-data generation, or governed derivative-model creation. Each resulting derivative is a governed Resource Derivative whose governance and settlement structure is established at the moment of formation and propagates recursively to any further derivatives. Distillation, which conventional systems treat as a threat to be detected and prevented after the fact, becomes in this architecture a governed and economically participatory activity — a licensed distillation economy — without ceasing to be controllable.

Components

In various embodiments, the system comprises one or more of the following components, each of which is a computer-implemented component comprising machine-executable instructions executed by one or more processors. The component names are illustrative.

- **Frontier AI Resource Registry.** A registry component that records artificial intelligence models, model weights, training corpora, evaluation outputs, synthetic datasets, orchestration systems, and associated derivative permissions as governed QP Resources. For each registered resource, the registry stores or references a resource identifier, a resource-type descriptor, an associated QPD identifier, one or more associated Trust Block identifiers, and a provenance record. The registry exposes the registered resources to the Derivative Authorization Engine for evaluation and to the Recursive AI Lineage Graph for lineage recording. In various embodiments the registry is distributed across multiple QPDs and multiple infrastructures, and a single resource may be registered as existing simultaneously in a plurality of QPDs, each representing a distinct governance context, consistent with the governance-superposition mechanics of the incorporated disclosures.
- **Derivative Authorization Engine.** An evaluation component that receives a derivative-formation request and determines whether the requested operation — for example a requested fine-tuning, distillation, orchestration, evaluation, or synthetic-data-generation operation — satisfies the Trust Criteria of every implicated upstream resource and the permissions granted by the implicated sponsors. The engine evaluates the request against the upstream Trust Blocks, against Constitutional Guardrails, against jurisdictional conditions, and against derivative-formation permissions, and produces an authorization decision that may be granted, denied, conditionally granted, or escalated for additional authorization. In various embodiments the authorization decision is itself recorded as a governed artifact for auditability.
- **Derivative AI Execution Environment.** A governed execution environment, instantiated within one or more QPDs, within which the authorized derivative-formation operation is performed. The execution environment inherits the upstream Trust Blocks, Constitutional Guardrails, output restrictions, and settlement obligations applicable to the resources it operates upon, so that the derivative-formation operation is itself a governed computation. In various embodiments the execution environment is partitioned across multiple infrastructures, and protected upstream resources — for example upstream model weights — may remain isolated within their own QPDs while the derivative-formation operation proceeds, consistent with the partitioned-collaborative-computation mechanics described in Family U.
- **Recursive AI Lineage Graph.** A graph data structure whose nodes represent upstream models, downstream models, training resources, output resources, sponsors, infrastructure providers, evaluation resources, and settlement participants, and whose edges represent derivation relationships, governance-inheritance relationships, attribution relationships, and settlement-propagation relationships. The lineage graph is the authoritative record of which derivatives descend from which upstream resources, and it is traversed by the Derivative Settlement Initializer and by recursive settlement propagation to determine settlement routing. Because the graph records relationships rather than protected content, it may be made available for governance verification — including by regulators and counterparties — without exposing protected model weights or protected data.

- **Derivative Settlement Initializer.** A component that, upon registration of a new derivative resource, establishes the recurring settlement-participation structures associated with that derivative. The initializer reads the settlement structure encoded in the derivative's Trust Block, identifies the upstream sponsors, derivative creators, infrastructure contributors, governance participants, and downstream derivative-right holders entitled to participate, and registers the corresponding settlement-participation relationships in the settlement system, such that subsequent inference, resale, licensing, orchestration, or derivative reuse of the derivative generates settlement flows routed according to those relationships.

Process Flow

In various embodiments, the system operates according to the following process. The ordering is illustrative; operations may be reordered, combined, omitted, or performed concurrently except where a later operation depends upon the result of an earlier one.

A preliminary condition is that an upstream AI resource has been registered in the Frontier AI Resource Registry and bound to one or more Trust Blocks within one or more QPDs, and that the upstream resource's Trust Block encodes its derivative-authorization policy and settlement structure.

First, the system receives a derivative-formation request. The request identifies a target — for example a target model to be produced — together with a derivative objective, the upstream resources to be drawn upon, the permitted infrastructures and jurisdictions, and proposed settlement terms. Second, the system evaluates, by means of the Derivative Authorization Engine, the request against the Trust Criteria of each implicated upstream resource, the sponsor permissions, the applicable derivative restrictions, the Constitutional Guardrails, and the governance compatibility of the requested operation. Proof-of-Trust verification confirms the requesting participant's authorization, the governance lineage of the implicated resources, the provenance of those resources, and the integrity of the proposed execution environment. If the evaluation does not succeed, the request is denied or escalated, and the disposition is recorded.

Third, if authorization succeeds, the system instantiates a governed Derivative AI Execution Environment within one or more QPDs, and that environment inherits the upstream Trust Blocks, Constitutional Guardrails, output restrictions, and settlement obligations. Fourth, the system generates the derivative AI resource within the governed execution environment, by the authorized operation — distillation, fine-tuning, orchestration, synthetic-data generation, evaluation, or recursive derivative formation from one or more existing derivatives. Fifth, the system registers the resulting derivative AI resource in the Frontier AI Resource Registry and records its lineage in the Recursive AI Lineage Graph, including edges to every upstream resource and sponsor. Sixth, the system initializes, by means of the Derivative Settlement Initializer, the recursive settlement participation for the upstream sponsors, the derivative creators, the infrastructure contributors, the governance participants, and the downstream derivative-right holders. The derivative is thereafter itself available, subject to its Trust Block, as an upstream resource for further governed derivative formation, and the process may recurse.

Error and exception handling: a failed authorization produces a recorded denial and no execution environment is instantiated; a failure during derivative formation produces a recorded fault and the partially formed derivative, if any, is quarantined within its execution environment pending governance disposition; an attempt to register a derivative whose claimed lineage cannot be verified against the Recursive AI Lineage Graph is rejected.

Alternative Embodiments

In an enterprise-specific embodiment, the governed derivative ecosystem is operated within and around a single enterprise, which sponsors an upstream model and authorizes governed derivative formation by its business units, subsidiaries, and contractors, with settlement participation routed among internal cost centers and external contributors. In a sovereign embodiment, a sovereign entity sponsors an upstream model as a public resource and authorizes governed derivative formation by domestic enterprises, research institutions, and agencies, with a portion of settlement routed to public-benefit structures as described in Family W. In an open-weight relicensing embodiment, an existing open-weight model is ingested into a QPD-bound governance structure, so that derivatives formed from that point forward carry governance and settlement structure even though the original weights were openly published. In a synthetic-data embodiment, the governed derivative is a synthetic dataset rather than a model, and the synthetic dataset carries Trust Blocks restricting and attributing its downstream use as training data. In a multi-layer marketplace embodiment, derivatives are formed, registered, and made available for further derivative formation across multiple layers, producing a marketplace of governed derivative models in which every layer participates in the settlement generated by the layers below it. In a cross-model orchestration embodiment, the derivative is an orchestration system that composes multiple upstream models, and the orchestration system's Trust Block encodes settlement participation for each composed model. In a safety-evaluation embodiment, the derivative is an AI safety evaluation resource — for example an adversarial test suite or an interpretability probe — formed from one or more upstream models and itself governed and attributable. These embodiments are illustrative and may be combined.

Detailed Component Architecture and Data Structures

The following provides component-level architectural detail for the Family Q system. Field names, identifier formats, enumerated values, and numeric parameters are illustrative and non-limiting:

- **Frontier AI Resource Registry — record schema.** In various embodiments, the registry maintains, for each registered resource, a registry record comprising: a resource identifier; a resource-type descriptor drawn from an extensible enumeration including model-weights, training-corpus, evaluation-output, synthetic-dataset, orchestration-system, gradient-set, activation-set, and derivative-permission; one or more governing Quantum Privacy Domain identifiers, a plurality of which may be present where the resource exists simultaneously across multiple governance contexts as described in the governance-superposition mechanics of the incorporated disclosures; one or more Trust Block identifiers; a provenance record comprising an ordered sequence of provenance entries, each entry identifying a contributing resource, a contributing party, a contribution timestamp, and a contribution-type descriptor; a content-addressed integrity digest of the registered resource; and a registration-state value drawn from the governance-state enumeration described in the omnibus operational-flow material. The registry record does not contain the protected resource content itself; the protected content — for example model weights — remains within its governing Quantum Privacy Domain, and the registry record references it by content-addressed identifier, so that the registry may be traversed and verified without exposing protected content.
- **Derivative Authorization Engine — evaluation procedure.** In various embodiments, the engine evaluates a derivative-formation request by executing an ordered evaluation pipeline. The pipeline first resolves the set of implicated upstream resources by traversing the Recursive AI Lineage Graph from each resource named in the request through its transitive upstream closure, so that the evaluation accounts not only for the directly named resources but for every

resource upstream of them. The pipeline then, for each implicated upstream resource, retrieves the resource's Trust Block and evaluates the requested operation against the Trust Block's authorization policy, its restricted-action set, its derivative-formation permissions, and its jurisdictional conditions. The pipeline then evaluates the request against the applicable Constitutional Guardrails. The pipeline then computes a composite authorization determination by combining the per-resource determinations under a combining rule that, in various embodiments, denies the request if any implicated upstream resource denies it, conditionally grants the request if any implicated resource conditionally grants it and none denies it, and otherwise grants the request. The authorization determination, together with the evaluated inputs and the combining-rule result, is recorded as an authorization record bound to a Trust Block, so that the authorization decision is itself an auditable governed artifact and the evaluation is deterministically reproducible by any party with access to the recorded inputs.

- **Recursive AI Lineage Graph — structure and operations.** In various embodiments, the lineage graph is a directed acyclic graph in which each node carries a node-type descriptor and a reference to the corresponding registry record, and each directed edge carries an edge-type descriptor drawn from an enumeration including derived-from, trained-on, evaluated-by, orchestrated-by, governed-by, and settles-to, together with an edge-weight and a reference to the authorization record under which the edge was created. The graph supports a transitive-upstream-closure operation, which returns, for a given node, the set of all nodes reachable by reversed traversal of derived-from, trained-on, and orchestrated-by edges; a settlement-path operation, which returns the set of settles-to paths from a given node; and a governance-inheritance operation, which returns the union of the Trust Criteria of all nodes in the transitive-upstream closure. Because the graph is acyclic by construction — a derivative cannot be upstream of its own ancestor — the closure operations terminate. An attempt to create an edge that would introduce a cycle is rejected as a governance fault.
- **Derivative Settlement Initializer — settlement-structure resolution.** In various embodiments, upon registration of a derivative resource, the initializer reads the settlement structure encoded in the derivative's Trust Block, which specifies an upstream-participation share, a derivative-creator share, an infrastructure-contributor share, a governance-participant share, a downstream-derivative-right-holder share, and a public-benefit-allocation share, the shares summing to unity. The initializer resolves each share to a concrete set of settlement-participation relationships by traversing the lineage graph: the upstream-participation share is apportioned among the resources of the transitive-upstream closure according to per-edge contribution weights; the infrastructure-contributor share is apportioned among the parties recorded as having provided execution infrastructure; and so forth. The resolved settlement-participation relationships are registered with the settlement systems of the incorporated disclosures, such that a subsequent settlement event associated with the derivative is routed according to those relationships. Because settlement-structure resolution is a deterministic function of the Trust Block and the lineage graph, it is reproducible and auditable.

State-Transition Model

In various embodiments, a derivative-formation operation progresses through a state machine. The operation begins in a REQUESTED state upon receipt of the derivative-formation request. It transitions to an EVALUATING state during authorization evaluation. From EVALUATING it transitions to DENIED if authorization fails, to ESCALATED if authorization requires additional approval, or to AUTHORIZED if authorization succeeds. From AUTHORIZED it transitions to

ENVIRONMENT-INSTANTIATED when the governed Derivative AI Execution Environment is instantiated. From ENVIRONMENT-INSTANTIATED it transitions to FORMING during the derivative-formation computation. From FORMING it transitions to FORMED upon successful completion, or to FAULTED upon a failure during formation, in which latter case the partially formed derivative is quarantined within its execution environment. From FORMED it transitions to REGISTERED when the derivative is recorded in the registry and the lineage graph, and then to SETTLEMENT-INITIALIZED when the Derivative Settlement Initializer has registered the settlement-participation relationships. A derivative in the SETTLEMENT-INITIALIZED state is available, subject to its Trust Block, as an upstream resource for further derivative formation, so that the state machine may be re-entered recursively. Each state transition is recorded as a Trust-Block-bound event, so that the lifecycle of a derivative-formation operation is fully auditable.

Additional Alternative Embodiments

The following additional embodiments supplement those described above. Each is illustrative and may be combined with any other.

In a centralized embodiment, the Frontier AI Resource Registry, the Derivative Authorization Engine, and the Recursive AI Lineage Graph are operated by a single coordinating operator, and the governed Derivative AI Execution Environments are instantiated within infrastructure controlled by that operator. The centralized embodiment is operationally simple but concentrates trust in the operator; it is disclosed as one embodiment among several and is not preferred over the others.

In a decentralized embodiment, the registry and the lineage graph are maintained as distributed data structures replicated across a plurality of independent nodes, no one of which is authoritative, and consensus among the nodes governs the admission of registry records and lineage edges. The decentralized embodiment removes the single point of trust of the centralized embodiment at the cost of additional coordination mechanics.

In a federated embodiment, a plurality of operators each maintain a registry and lineage graph for their own domain, and a federation layer reconciles cross-domain lineage edges and cross-domain authorization, so that a derivative formed in one operator's domain from an upstream resource in another operator's domain carries lineage and settlement structure spanning both domains.

In a cryptographically-enforced embodiment, the authorization determinations and the settlement-structure resolutions are not merely recorded but are cryptographically enforced, such that the governed Derivative AI Execution Environment will not perform the derivative-formation computation absent a valid cryptographic authorization token issued by the Derivative Authorization Engine, and the settlement systems will not route a settlement event absent a valid cryptographic settlement-participation credential issued by the Derivative Settlement Initializer.

In a graph-native embodiment, the entire Family Q system is expressed as operations over the Recursive AI Lineage Graph, with authorization, settlement resolution, and governance inheritance all computed as graph traversals, so that the system's behavior is fully determined by the graph and the operations defined over it.

In a recursive-derivative-depth embodiment, a derivative formed from one or more upstream derivatives is itself used to form further derivatives, across an unbounded number of generations, and the architecture maintains complete lineage and settlement structure across the full depth; the embodiment is significant because the value-leakage problem that Family Q addresses is most acute precisely at depth, where conventional attribution is most thoroughly lost.

In an AI-mediated-formation embodiment, the derivative-formation request is generated, and the derivative-formation operation is supervised, by a governed AI coordination system as described in Family V, so that derivative AI ecosystems may form and extend themselves under governance without manual initiation of each formation.

In a synthetic-data-lineage embodiment, the governed derivative is a synthetic dataset, and the architecture additionally records, for the synthetic dataset, the statistical and generative relationship between the synthetic dataset and the upstream resources, so that a downstream model trained on the synthetic dataset carries lineage not only to the synthetic dataset but, transitively, to the resources from which the synthetic dataset was generated.

Cross-Family Integration

Upstream dependencies. Family Q operates within the QPN-enabled infrastructure disclosed in the incorporated granted patent and provisional applications, including Quantum Privacy Domains, Privacy Algorithms, Trust Blocks, Trust Criteria, Proof-of-Trust verification, settlement systems, Contribution Graphs, Quantum DNA inheritance, and derivative-resource governance. It depends in particular on the Trust Block inheritance mechanics and the QP Resource and Resource Derivative constructs of the incorporated disclosures and of Family O of the May 2026 QPN Catalyst Provisional.

Downstream consumers. Family Q is the foundation for Families R, S, V, X, and Z. Family R extends governed derivative formation across the full AI lifecycle. Family S governs the inference and output stage at which distillation risk arises. Family V permits AI systems themselves to initiate governed derivative formation as economic participants. Family X applies recursive derivative formation to accelerators. Family Z applies contribution-lineage analysis, which depends on the lineage graph, to venture formation.

Lateral interactions. Family Q's governed Derivative AI Execution Environment interoperates with Family U's partitioned collaborative computation when a derivative is formed across mutually distrustful infrastructures; with Family T's resource-bound execution governance when the derivative-formation operation must itself be constrained; and with the Settlement Controller and Premium Framework of Families G and J of the May 2026 QPN Catalyst Provisional for the actual routing of settlement flows.

Emergent properties. The integration of governed derivative formation with recursive Trust Block inheritance produces an emergent property that neither provides alone: a derivative AI artifact, however many generations removed from its origin, carries a complete, cryptographically verifiable, and machine-enforceable record of its governance obligations and its settlement participants. This converts the AI derivative ecosystem from an environment in which value leaks and attribution is lost into one in which value propagates and attribution is preserved across unbounded derivative depth.

Initial Claims

The following claims are provided as provisional source material and may be revised, renumbered, and restructured in any subsequent application.

A computer-implemented system for governed derivative artificial intelligence formation, the system comprising one or more processors and memory storing instructions that, when executed, provide: one or more upstream artificial intelligence resources registered as governed resources within one or more governed computational environments; one or more derivative-authorization systems; one or more governed derivative execution environments; one or more derivative-lineage

systems; and one or more settlement-participation systems; wherein a derivative artificial intelligence resource generated from the one or more upstream artificial intelligence resources inherits governance continuity and settlement participation obligations from the one or more upstream artificial intelligence resources.

A computer-implemented method comprising: registering an upstream artificial intelligence resource as a governed resource within a governed computational environment; receiving a derivative-generation request; evaluating derivative authorization for the request according to one or more Trust Criteria associated with the upstream artificial intelligence resource; generating a derivative artificial intelligence resource within a governed execution environment that inherits one or more governance structures associated with the upstream artificial intelligence resource; registering a lineage relationship between the derivative artificial intelligence resource and the upstream artificial intelligence resource; and initializing recursive settlement participation associated with the derivative artificial intelligence resource.

A non-transitory computer-readable medium storing instructions that, when executed by one or more processors, cause the one or more processors to perform governed derivative artificial intelligence formation and recursive settlement propagation according to the foregoing method.

Family R — QP-Bounded AI Lifecycle Governance

Field Summary

This Family relates to computer-implemented artificial intelligence governance, and more particularly to systems and methods for governing the complete lifecycle of an artificial intelligence system — including training data, model weights, intermediate computational states, evaluation outputs, deployment and inference systems, and derivative artifacts — as governed Quantum Privacy Resources bound to Quantum Privacy Domains under Trust Block enforcement, such that governance continuity persists across every lifecycle stage.

Problem Addressed

Conventional artificial intelligence governance frameworks divide the model lifecycle into separate legal, operational, and technical domains: the handling of training-time data, the storage of model weights, the conduct of evaluations, the operation of inference, the processing of outputs, the conduct of fine-tuning, and the retirement of models are each governed, if at all, by distinct mechanisms applied at distinct stages by distinct parties. This fragmentation creates governance gaps, because a compliance, ownership, safety, or settlement obligation applied at one stage does not automatically bind the downstream stages or the derivative artifacts produced from that stage.

The consequences are concrete. A model trained on governed data may later be exported, fine-tuned, distilled, redeployed, or queried in a context that no longer preserves the governance conditions imposed by the original data sponsors. Training data may be carefully governed while the intermediate activations, the gradient computations, the evaluation outputs, the synthetic data, and the downstream orchestration artifacts derived from that data remain entirely ungoverned, even though each of those artifacts may encode recoverable information about, or value derived from, the governed training data. A safety evaluation conducted at one stage produces results that are not bound to the model in a way that travels with the model to its next stage. The unresolved technical problem is lifecycle continuity: the absence of a mechanism by which governance, attribution, and settlement obligations attach to every artifact produced at every stage of the AI lifecycle and propagate automatically to the artifacts of every subsequent stage.

Solution Overview

The disclosed architecture treats every artifact of the AI lifecycle as a governed QP Resource bound to one or more QPDs under Trust Block enforcement. The governed artifacts include, without limitation: training data; model weights; intermediate activations and gradient computations; evaluation outputs, including safety metrics, performance benchmarks, adversarial-evaluation results, and interpretability results; inference outputs; synthetic datasets; fine-tuned weights; orchestration states; and any further derivative artifacts. Because each artifact is a governed QP Resource, and because — consistent with the Trust Block inheritance mechanics of the incorporated disclosures — the Trust Block of any artifact derived from one or more upstream artifacts incorporates links to the Trust Criteria of those upstream artifacts, governance continuity persists across the entire lifecycle: from data ingestion through training, fine-tuning, evaluation, deployment, inference, orchestration, derivative generation, and retirement.

Lifecycle governance is enforced at authorization points. A model, or any other lifecycle artifact, cannot transition from one lifecycle stage to the next — for example from training to evaluation, from evaluation to deployment, from deployment to fine-tuning, or from any stage to export or derivative generation — without satisfying, by Proof-of-Trust verification against the relevant Trust Criteria, the governance requirements associated with the upstream resources that produced it. Enforcement is effected through Trust Criteria, Privacy Algorithms, Quantum DNA inheritance, Constitutional Guardrails, derivative obligations, and settlement-participation structures. The result is that the governance, attribution, and settlement conditions imposed at any stage are not merely recorded; they are made into preconditions for every subsequent stage.

Components

In various embodiments, the system comprises one or more of the following computer-implemented components:

- **Lifecycle Resource Binder.** A component that associates each AI lifecycle artifact — as the artifact is created — with one or more QPD-bound governance structures, by registering the artifact as a QP Resource, assigning it to one or more QPDs, and binding it to one or more Trust Blocks that incorporate links to the Trust Criteria of the upstream artifacts from which it was produced.
- **Training Governance Engine.** A component that enforces, during training and fine-tuning, the sponsor restrictions, data-use permissions, and provenance requirements associated with the training data and the compute resources, such that training proceeds only upon and in the manner permitted by the Trust Criteria of the implicated resources.
- **Gradient and Activation Governance Layer.** A component that treats intermediate computational states — including activations and gradient computations — as governed derivative resources in their own right, binding them to Trust Blocks and thereby closing the governance gap that arises when intermediate states are left ungoverned.
- **Evaluation Governance Layer.** A component that binds evaluation artifacts — safety metrics, performance outputs, adversarial-evaluation results, and interpretability results — to Trust Blocks, so that evaluation results travel with the model as governed artifacts and are available, subject to governance, to downstream stages and parties.
- **Inference Governance Layer.** A component that applies, at inference time, the output restrictions, settlement-participation obligations, and derivative obligations associated with

the model and its upstream resources, and that interoperates with the distillation-resistant inference governance of Family S.

- **Retirement and Archival Governance Layer.** A component that, upon retirement or archival of a model or other artifact, preserves the artifact's lineage, auditability, and any post-retirement obligations, so that a retired artifact remains accounted for within the governance graph and cannot be silently revived outside governance.

Process Flow

In various embodiments, the system operates as follows. Data, model components, compute resources, and evaluation resources are ingested into QPD-bound environments. Each lifecycle artifact, as it is created, is bound by the Lifecycle Resource Binder to one or more Trust Blocks. Training or fine-tuning is executed under Privacy Algorithm enforcement and under the Training Governance Engine. Model weights, intermediate states, and evaluation outputs are registered as governed derivative QP Resources by the Gradient and Activation Governance Layer and the Evaluation Governance Layer. Inference, orchestration, export, or derivative generation is authorized only upon satisfaction, by Proof-of-Trust verification, of the relevant Trust Criteria. Governance and settlement obligations propagate recursively to every downstream lifecycle artifact. Upon retirement, the Retirement and Archival Governance Layer preserves lineage and auditability and enforces post-retirement obligations.

Exception handling: an attempt to transition an artifact to a subsequent lifecycle stage without satisfying the relevant Trust Criteria is denied and recorded; an attempt to operate upon an artifact whose lifecycle binding is absent or unverifiable is rejected; a detected divergence between an artifact's recorded lineage and its claimed lineage triggers a governance fault.

Alternative Embodiments

In a frontier-model embodiment, the lifecycle governance spans the complete development of a frontier model from data acquisition through deployment and ongoing fine-tuning. In an enterprise embodiment, lifecycle governance is applied to an enterprise's internal models and to the data, evaluations, and derivatives associated with them. In a sovereign embodiment, lifecycle governance is applied to a sovereign AI program, with sovereign Trust Authorities authoritative over each stage. In an open-weight relicensing embodiment, an open-weight model is ingested and governed for all subsequent lifecycle stages even though its earlier stages were ungoverned. In a safety-evaluation lifecycle embodiment, the governance focuses on binding safety evaluations to models so that safety results are never separated from the artifacts they describe. In a confidential-computing embodiment, lifecycle governance is combined with secure enclaves and trusted-execution environments so that lifecycle artifacts are governed both cryptographically and within hardware-isolated execution. These embodiments are illustrative and may be combined.

Detailed Component Architecture and Lifecycle-Stage Mechanics

The following provides additional component-level detail for the Family R system, directed to the mechanics by which governance continuity is enforced across lifecycle-stage transitions.

Lifecycle Resource Binder — artifact binding record. In various embodiments, the binder creates, for each AI lifecycle artifact at the moment of the artifact's creation, a lifecycle binding record comprising: an artifact identifier; a lifecycle-stage descriptor drawn from an enumeration including data-ingestion, training, fine-tuning, evaluation, deployment, inference, orchestration, derivative-generation, and retirement; one or more governing Quantum Privacy Domain identifiers;

one or more Trust Block identifiers; and a set of upstream-artifact references identifying the artifacts from which the present artifact was produced. The Trust Block of the artifact incorporates, by the Trust Block inheritance mechanics of the incorporated disclosures, links to the Trust Criteria of each referenced upstream artifact, so that the governance applicable to the upstream artifacts is inherited by the present artifact.

Lifecycle-stage transition gate. In various embodiments, the system enforces a transition gate at each lifecycle-stage boundary. A proposed transition of an artifact from one lifecycle stage to a subsequent stage — for example a proposed transition of a model from the evaluation stage to the deployment stage, or from the deployment stage to a fine-tuning stage, or a proposed export or derivative-generation — is evaluated by the gate against the Trust Criteria inherited in the artifact's Trust Block. The gate computes the governance-inheritance set, being the union of the Trust Criteria of the artifact and of its transitive upstream artifacts, and permits the transition only if the proposed subsequent-stage use satisfies every Trust Criterion in that set. The gate's evaluation is verified by Proof-of-Trust. The transition gate is the mechanism by which the governance imposed at any stage becomes an enforceable precondition of every later stage: a model trained on governed data cannot transition to a deployment or export stage whose use would violate the data sponsors' Trust Criteria, because the gate evaluates those inherited criteria at the transition.

Gradient and Activation Governance Layer — intermediate-state binding. In various embodiments, the layer treats intermediate computational states — activations, gradients, optimizer states, and other artifacts produced during training and fine-tuning — as governed derivative resources, creating a lifecycle binding record and a Trust Block for each such intermediate-state artifact that is retained, so that the governance gap that arises when intermediate states are left ungoverned is closed. This is significant because intermediate states may encode recoverable information about governed training data, and an ungoverned intermediate state is a pathway by which governed information escapes governance.

State-Transition Model

In various embodiments, an AI lifecycle artifact occupies, at any time, a lifecycle stage, and the permissible transitions among stages are governed by the transition gate. The stages and their characteristic transitions are: data-ingestion, from which an artifact may transition to training; training, from which an artifact may transition to evaluation or to fine-tuning; fine-tuning, from which an artifact may transition to evaluation; evaluation, from which an artifact may transition to deployment or back to fine-tuning; deployment, from which an artifact may transition to inference, orchestration, fine-tuning, export, or derivative-generation; inference and orchestration, from which derivative artifacts such as outputs and synthetic datasets are produced under Family S governance; derivative-generation, which produces derivative artifacts governed under Family Q; and retirement, to which any artifact may transition and from which it transitions to an archived condition. Every transition passes through the transition gate and is recorded as a Trust-Block-bound event, so that the lifecycle history of an artifact is complete and auditable.

Additional Alternative Embodiments

In a centralized embodiment, a single operator governs the full lifecycle of its models. In a federated embodiment, distinct lifecycle stages are conducted by distinct parties in distinct domains — for example data curation by one party, training by another, evaluation by an independent evaluator — and the transition gate enforces governance continuity across the party boundaries. In a confidential-computing embodiment, lifecycle artifacts are held and processed within secure enclaves, and the lifecycle binding records are bound to enclave attestations. In a

continuous-fine-tuning embodiment, a deployed model undergoes repeated fine-tuning transitions, each passing through the transition gate, so that governance continuity is maintained across an ongoing sequence of updates. In a regulated-domain embodiment, the lifecycle governance is applied to AI used in a regulated domain such as healthcare, and the inherited Trust Criteria include the regulatory conditions applicable to the training data, so that a regulated-domain model cannot transition to a use that the regulatory conditions of its training data prohibit. In a graph-native embodiment, the lifecycle is represented as a graph of artifacts and stage transitions, and the transition gate is computed as a graph operation over inherited Trust Criteria. These embodiments are illustrative and may be combined.

Cross-Family Integration

Upstream dependencies. Family R operates within the QPN-enabled infrastructure of the incorporated disclosures and depends on the QP Resource, Trust Block, and Trust Block inheritance constructs, and on Family O Quantum DNA inheritance.

Downstream consumers. Family R is consumed by Family Q, which forms derivatives at specific lifecycle stages; by Family S, which governs the inference stage; by Family T, which governs execution; and by Family U, which conducts collaborative computation across lifecycle stages such as evaluation.

Lateral interactions. Family R's Inference Governance Layer interoperates directly with Family S; its Evaluation Governance Layer interoperates with the collaborative safety evaluation of Family U; and its binding of artifacts to settlement obligations interoperates with the settlement systems of Families G and J of the May 2026 QPN Catalyst Provisional.

Emergent properties. The integration of per-artifact governance binding with recursive Trust Block inheritance produces lifecycle continuity as an emergent system property: governance, attribution, and settlement obligations imposed at any stage become enforceable preconditions at every later stage, closing the governance gaps that fragmentation otherwise creates.

Initial Claims

A computer-implemented artificial intelligence lifecycle governance system comprising one or more processors and memory storing instructions that, when executed, provide: one or more lifecycle-artifact registries; one or more Trust Block inheritance systems; one or more artificial intelligence execution governance systems; and one or more derivative-resource lineage systems; wherein artifacts generated across an artificial intelligence lifecycle, including one or more of training data, model weights, intermediate computational states, evaluation outputs, inference outputs, and derivative artifacts, inherit governance continuity from the upstream artifacts from which they are produced.

A computer-implemented method comprising: binding artificial intelligence training data, model weights, intermediate computational states, evaluation outputs, inference outputs, and derivative artifacts to Trust Blocks governed within one or more Quantum Privacy Domains; and propagating governance continuity across artificial intelligence lifecycle stages such that a transition of an artifact from one lifecycle stage to a subsequent lifecycle stage is conditioned upon satisfaction of one or more Trust Criteria associated with the upstream artifacts from which the artifact was produced.

Family S — Distillation-Resistant Inference & Output Governance

Field Summary

This Family relates to computer-implemented artificial intelligence governance, and more particularly to systems and methods for governed AI inference, prevention of unauthorized model extraction, distillation-risk analysis, and derivative-output governance, in which inference outputs and other model outputs are treated as governed derivative resources whose access is mediated by Trust Blocks and dynamically constrained according to a computed risk of derivative extraction.

Problem Addressed

Conventional artificial intelligence systems rely on static application-programming-interface throttling, terms-of-service contracts, fixed rate limits, and post hoc enforcement to prevent unauthorized model extraction. These mechanisms are structurally inadequate because model extraction — distillation — does not require literal exfiltration of model weights. A model's behavior can be reconstructed, and high-quality synthetic training data sufficient to train a competitive derivative can be generated, through high-volume, high-diversity, high-entropy sampling of the model's outputs across an inference interface that each individual request appears to use legitimately.

The unresolved technical problem is therefore the prevention of extraction of a distillable signal, as distinct from the prevention of export of model weights. Conventional systems do not attach enforceable derivative restrictions to the outputs themselves; they do not evaluate the cumulative semantic diversity or cumulative entropy of a sequence of outputs; they do not propagate output-governance obligations to the synthetic datasets and downstream models derived from those outputs; and they do not integrate extraction-risk analysis with recursive settlement and authorization systems. As a result, a determined party can extract a distillable signal through a pattern of individually unremarkable requests, and the resulting derivative carries no governance or settlement relationship to the model it was distilled from.

Solution Overview

The disclosed architecture provides distillation-resistant inference governance by treating inference outputs, model outputs, synthetic-data outputs, orchestration outputs, and derivative outputs as governed derivative resources. Output access is mediated through Trust Blocks that define, for the outputs of a given model in a given context, the permitted reuse, the permitted sampling rates, the permitted output granularity, the derivative-formation rights, the attribution obligations, and the settlement participation. An output is therefore not an unencumbered artifact but a governed Resource Derivative, and any synthetic dataset or downstream model formed from a corpus of such outputs inherits, through Trust Block inheritance, the governance and settlement obligations carried by those outputs.

One or more distillation-risk engines continuously evaluate, across a session and across an ecosystem, the risk that the pattern of inference activity constitutes or contributes to model extraction. The evaluated factors include, without limitation: query frequency; the semantic diversity of the queries and outputs; the accumulation of entropy across the sequence of outputs; the similarity of outputs to one another and to prior outputs; an estimated probability that the model could be reconstructed from the outputs observed so far; an estimated risk that the outputs

constitute extractable synthetic training data; an estimated probability of derivative formation; and the cumulative topology of the outputs considered as a set. In response to the computed risk, the governance systems may dynamically throttle outputs, reduce output granularity, partition access, require additional authorization, suspend the inference session, revoke access, or — where the requesting party is permitted to form a derivative — convert the session into a governed derivative-formation workflow as described in Family Q. The last of these responses is significant: rather than merely blocking a party who has the right to distill, the architecture routes that party into the governed, attributed, settlement-bearing distillation pathway.

Components

Distillation-Risk Engine. A component that evaluates extraction risk across sessions, participants, resources, and derivative graphs, computing one or more risk metrics from the observed pattern of inference activity and maintaining cumulative risk state at session and ecosystem granularity.

Output Governance Engine. A component that attaches Trust Blocks to generated outputs, encoding in each output's Trust Block the permitted reuse, derivative-formation rights, attribution obligations, and settlement participation applicable to that output.

Sampling Constraint Engine. A component that governs the rate, diversity, sequence, and entropy accumulation of outputs provided to a requesting party, dynamically adjusting these as a function of the computed distillation risk.

Derivative-Output Inheritance Engine. A component that propagates output-governance obligations to downstream outputs and synthetic datasets formed from governed outputs, by ensuring that the Trust Block of any such downstream artifact incorporates links to the Trust Criteria of the governed outputs it was formed from.

Authorized Distillation Controller. A component that, when a requesting party is authorized to form a derivative, converts what would otherwise be prohibited extraction into a governed derivative-formation workflow, establishing the governance and settlement structure of the resulting derivative as described in Family Q.

Process Flow

In various embodiments, the system receives an inference request together with session context. It evaluates the requesting participant's authorization and the applicable Trust Criteria. It computes distillation-risk metrics, including semantic diversity, entropy accumulation, reconstruction probability, and output similarity, updating cumulative session and ecosystem risk state. It adjusts output access by one or more of throttling, redaction, granularity reduction, partitioning, and authorization escalation, as a function of the computed risk. It attaches output-governance Trust Blocks to the generated outputs. It registers output lineage and propagates derivative obligations. If the requesting party is authorized to form a derivative, it initiates, by means of the Authorized Distillation Controller, a governed distillation workflow with associated settlement participation.

Exception handling: a computed risk exceeding a revocation threshold causes session access to be revoked and recorded; a detected attempt to circumvent sampling constraints — for example by distributing extraction across multiple sessions or participants — is detected by ecosystem-level cumulative risk scoring and treated as a single extraction pattern; an output whose governance Trust Block cannot be attached is not released.

Alternative Embodiments

In an enterprise model-API embodiment, distillation-resistant governance is applied to an enterprise's commercial model API. In a frontier-lab licensed-distillation embodiment, the architecture is used principally to route authorized parties into governed distillation while resisting unauthorized extraction. In a synthetic-dataset embodiment, the governed outputs are explicitly synthetic data and the governance focuses on attributing and restricting downstream training use. In a safety-evaluation embodiment, output governance is applied to the outputs produced during safety evaluation so that those outputs are themselves governed. In an open-weight-ingestion embodiment, the architecture governs the ingestion of outputs from an open-weight model into a QPD. In a hybrid query-rate and semantic-diversity embodiment, the risk computation combines simple rate metrics with semantic-diversity and entropy metrics. In a session-level and ecosystem-level embodiment, cumulative risk is scored both within a session and across an ecosystem of sessions, participants, and resources, so that distributed extraction is detected. These embodiments are illustrative and may be combined.

Detailed Component Architecture and Risk-Metric Definitions

The following provides component-level detail for the Family S system, including illustrative definitions of the distillation-risk metrics. The metric definitions are illustrative embodiments; other metrics and other definitions of these metrics are within the scope of the disclosure.

Distillation-Risk Engine — metric computation. In various embodiments, the engine maintains, for each inference session and for each ecosystem-level aggregation of sessions, a cumulative risk state, and updates that state upon each inference response. The engine computes one or more of the following illustrative metrics. A semantic-diversity metric quantifies the degree to which the queries and outputs of a session span the semantic space of the model's capability, computed in various embodiments as a measure of dispersion of embedding-space representations of the queries and outputs; a high semantic-diversity metric indicates sampling designed to cover the model's behavior broadly, which is characteristic of extraction. An entropy-accumulation metric quantifies the cumulative information content of the outputs provided to a requesting party over the course of a session or across sessions, computed in various embodiments as a running sum of per-output entropy estimates; the metric is significant because extraction of a distillable signal requires accumulation of information beyond a threshold, regardless of the rate at which it is accumulated. An output-similarity metric quantifies the degree to which outputs are clustered or repetitive versus broadly distributed. A reconstruction-probability metric estimates the probability that a model of comparable behavior could be reconstructed, or that synthetic training data sufficient to train such a model could be assembled, from the outputs observed so far; in various embodiments the reconstruction-probability metric is computed as a function of the entropy-accumulation metric, the semantic-diversity metric, and the coverage of the model's capability surface represented by the outputs. A derivative-formation-probability metric estimates the probability that the observed pattern is directed at forming a derivative. The engine aggregates the metrics into a composite distillation-risk score, in various embodiments as a weighted combination with weights configurable per model and per governance context.

Cumulative and cross-session aggregation. In various embodiments, the engine maintains risk state not only at the granularity of a single session but at the granularity of a participant across sessions, a set of related participants, a model, and an ecosystem. This is significant because a party seeking to extract a distillable signal while evading session-level detection may distribute the extraction across many sessions, many accounts, or many cooperating participants. By maintaining cumulative entropy-accumulation and semantic-diversity state across these broader

aggregations, the engine detects distributed extraction that no single session would reveal. In various embodiments, the engine attributes outputs to a cumulative aggregation by means of the participant identity, the contribution lineage, and behavioral correlation among sessions.

Sampling Constraint Engine — response model. In various embodiments, the engine governs, as a function of the current distillation-risk score, one or more sampling-constraint parameters applied to the outputs provided to a requesting party, including: an output-rate parameter limiting the number of outputs per unit time; an output-granularity parameter governing the level of detail of outputs; an output-diversity parameter limiting the semantic dispersion of outputs provided to the party; a sequence-constraint parameter governing the permissible sequence of outputs; and an entropy-budget parameter limiting the cumulative entropy the party may receive. As the distillation-risk score rises, the engine progressively tightens these parameters, so that the response is graduated rather than binary. In various embodiments the engine implements a set of risk bands, each band associated with a configured set of sampling-constraint parameters, and the engine selects the band corresponding to the current risk score.

Output Governance Engine — output Trust Block contents. In various embodiments, the engine attaches to each generated output a Trust Block specifying: the permitted reuse of the output; whether the output may be incorporated into a training corpus; whether the output may be used to form a derivative and under what conditions; the attribution obligations attaching to downstream use of the output; and the settlement-participation structure attaching to downstream use of the output. By means of the Trust Block inheritance mechanics of the incorporated disclosures, any synthetic dataset assembled from a corpus of such outputs, and any model trained on such a dataset, carries a Trust Block incorporating links to the Trust Criteria of the governed outputs, so that the governance and settlement obligations attached at the moment of output generation propagate to downstream derivatives.

State-Transition Model

In various embodiments, an inference session progresses through a state machine governed by the cumulative distillation-risk score. The session begins in an OPEN state, in which outputs are provided under a baseline sampling-constraint band. As the cumulative risk score rises across configured thresholds, the session transitions through a sequence of CONSTRAINED states, designated in various embodiments CONSTRAINED-1 through CONSTRAINED-N, each associated with a progressively tighter sampling-constraint band. If the risk score crosses an authorization-escalation threshold, the session transitions to an ESCALATED state, in which continued output requires additional authorization. If the risk score crosses a conversion threshold and the requesting party is authorized to form a derivative, the session transitions to a GOVERNED-DISTILLATION state, in which the session is converted into a governed derivative-formation workflow as described in Family Q. If the risk score crosses a revocation threshold, the session transitions to a REVOKED state and access is withdrawn. From any state, the session may transition to a CLOSED state upon completion. Each transition is recorded as a Trust-Block-bound event. The state machine is significant because it embodies the graduated response model: a session is not merely permitted or blocked, but is moved through constraint, escalation, conversion, or revocation according to the evolving evidence of extraction intent.

Additional Alternative Embodiments

In a centralized embodiment, a single operator operates the Distillation-Risk Engine and the Output Governance Engine for its own models. In a decentralized embodiment, the risk state and the output governance are maintained as distributed data structures, so that cumulative cross-session

risk may be aggregated across operators who each contribute observations without any operator holding a complete picture of another's traffic. In a federated embodiment, a plurality of operators federate their distillation-risk observations under a federation layer, so that distributed extraction directed at multiple operators' models in parallel is detected by the federation even though no single operator observes the whole pattern. In a controlled-degradation embodiment, the Sampling Constraint Engine responds to rising risk not by reducing output volume but by introducing governed degradation of the distillable signal — for example by reducing the precision or determinism of outputs in a manner that preserves the outputs' usefulness for legitimate single-use purposes while reducing their usefulness as training data. In an output-mutation embodiment, the engine applies governed transformations to outputs that preserve their semantic content for the requesting party while disrupting the statistical regularities on which distillation depends. In a graph-native embodiment, the distillation-risk analysis is computed over a distillation-risk topology, as described in the graph-theoretic embodiments, in which the outputs of a model are nodes and the analysis computes topological properties of the output set. In an AI-mediated embodiment, the distillation-risk evaluation and the selection of graduated responses are performed by a governed AI system as described in Families T and V. These embodiments are illustrative and may be combined.

Cross-Family Integration

Upstream dependencies. Family S operates within the QPN-enabled infrastructure of the incorporated disclosures, and depends on the QP Resource and Trust Block inheritance constructs and on the AI Evaluation Pipeline of Family C of the May 2026 QPN Catalyst Provisional.

Downstream consumers. Family S feeds Family Q, into which it routes authorized distillation; Family R, whose Inference Governance Layer it implements at the inference stage; and Family T, whose execution governance may act upon the risk signals Family S computes.

Lateral interactions. Family S interoperates with the Settlement Controller and Premium Framework of Families G and J for the settlement consequences of governed distillation, and with Family U where inference governance is conducted across partitioned infrastructures.

Emergent properties. Treating outputs as governed derivative resources, and combining that treatment with cumulative ecosystem-level risk scoring, produces an emergent property: the architecture distinguishes, and treats differently, the authorized party who should be routed into governed distillation and the unauthorized party whose extraction should be resisted — and it does so without relying on the cooperation of either. Distillation becomes a governed economic activity rather than an unpoliceable threat.

Initial Claims

A computer-implemented distillation-governance system comprising one or more processors and memory storing instructions that, when executed, provide: one or more inference governance systems; one or more output-governance systems; one or more distillation-risk engines; and one or more derivative-lineage systems; wherein access to an output of a governed artificial intelligence system is dynamically governed according to a computed risk of derivative extraction, and wherein the output is treated as a governed derivative resource inheriting one or more Trust Blocks, attribution lineage, settlement-participation obligations, and downstream-reuse restrictions.

A computer-implemented method comprising: evaluating one or more distillation-risk conditions associated with one or more artificial intelligence inference requests; dynamically governing one or more outputs of a governed artificial intelligence system according to the distillation-risk conditions; attaching one or more derivative-output governance structures to the one or more

outputs; and propagating one or more derivative-governance obligations to a downstream artifact formed from the one or more outputs.

Family T — Resource-Bound AI Existence & Constitutional Execution

Field Summary

This Family relates to computer-implemented artificial intelligence governance, and more particularly to systems and methods for governance-bound AI operation, revocable operational authority, executable action-space topology, and constitutional execution architecture, in which the continued operation of an artificial intelligence system is made structurally dependent upon its continued governed access to authorized resources, and in which the set of actions available to the AI system is cryptographically constrained before execution.

Problem Addressed

Conventional artificial intelligence systems are deployed as independently executable software artifacts. Once such a system is deployed, the model or agent continues to operate so long as compute, memory, network access, and application-programming-interface credentials remain available to it. Safety controls in conventional deployments are predominantly external to the executable artifact: monitoring systems, content filters, access policies, evaluation regimes, and institutional oversight are layered around the model. These external controls observe, score, and may react to the model's behavior, but they do not structurally remove harmful action pathways from the space of actions the model is able to execute. A content filter applied to a model's output does not prevent the model from generating the output; an access policy applied to an agent does not make an unauthorized action computationally unavailable to the agent; a monitoring system detects an anomaly after the action has already entered the executable space.

The unresolved technical problem is that conventional AI safety depends either on the model's internal cognition — its training, its alignment, its propensities — or on an operator's external policy layer, rather than on a cryptographically constrained execution architecture. Conventional systems do not make the operational existence of the AI system itself conditional on its continued governed access to authorized resources, and they do not constrain the executable action-space topology of the AI system before execution. As a consequence, safety is contingent and revocable only by indirect means, and a model that escapes or outlasts its external controls retains its full executable action space.

Solution Overview

The disclosed architecture provides resource-bound AI operational existence. An artificial intelligence system within the architecture continues to operate only while it retains governed access to the authorized computational, memory, communication, orchestration, settlement, actuator, and interoperability resources upon which its operation depends. Operational authority over those resources is not a static grant; it may be dynamically granted, denied, throttled, sandboxed, partitioned, suspended, revoked, or recursively constrained, by the governance systems, in response to governance conditions and governance events. Because the AI system's operation depends upon governed resource access, a withdrawal of operational authority is not merely a policy instruction that the AI system might disregard; it is a removal of the resources without which the AI system cannot operate. Operational existence is thereby made governable.

The architecture further provides that the executable action-space topology available to the AI system is cryptographically constrained before execution. An action that is not authorized is not merely filtered after the model generates it; it is rendered computationally inaccessible, cryptographically unavailable, operationally prohibited, or otherwise unavailable through the governed execution environment, such that the action pathway does not exist within the AI system's executable space at the time of execution. The executable action-space topology is determined, and continuously re-determined, by Constitutional Guardrails, Trust Criteria, Quantum DNA, sponsor permissions, settlement conditions, and operational risk scores. A Constitutional Guardrail, in various embodiments, is a governance-associated constraint that governs which computational activities, interoperability relationships, derivative-resource relationships, and ecosystem behaviors are permitted; the term "constitutional" denotes that these constraints are foundational to and inseparable from the execution environment, rather than being external policies applied to it. The combination of resource-bound existence and pre-execution action-space constraint means that AI safety, within this architecture, is a structural property of the execution architecture rather than a contingent property of the model's cognition or of an external policy layer.

Components

Operational Authority Resolver. A component that, for a given AI system and a given action request or operating interval, resolves the operational authority currently in effect — determining whether authority is granted, denied, throttled, sandboxed, partitioned, suspended, or revoked — by evaluating the governance conditions, Constitutional Guardrails, sponsor restrictions, jurisdictional conditions, settlement conditions, and risk scores applicable to the AI system.

Executable Action-Space Constraint Engine. A component that, before execution, constrains the executable action-space topology of the AI system to those action pathways that are authorized under the currently resolved operational authority, such that unauthorized pathways are not present in the executable space. In various embodiments the action space is represented as an executable-action graph, as described in the graph-theoretic embodiments below, and the constraint engine modifies node accessibility, edge accessibility, and traversal permissions within that graph.

Resource Access Governance Layer. A component that mediates the AI system's access to its underlying computational, memory, communication, orchestration, settlement, actuator, and interoperability resources, granting or withholding access in accordance with the resolved operational authority, and thereby making the AI system's operation dependent on governed resource access.

Constitutional Guardrail Evaluator. A component that evaluates the Constitutional Guardrails applicable to the AI system and to a given action, and reports the result to the Operational Authority Resolver and the Executable Action-Space Constraint Engine.

Dynamic Throttling and Revocation Engine. A component that effects changes in operational authority — throttling, sandboxing, partitioning, suspension, or revocation — in response to governance events, anomalies, and governance-trigger conditions, including by removing the AI system's access to required resources.

Governance-State Machine. A component that maintains the governance state of the AI system across a set of states including active, throttled, sandboxed, partitioned, suspended, revoked, and archived, and that governs the permitted transitions among those states. The Governance-State Machine is described further in the operational-flow embodiments below.

Actuator and Communication Authority Gateways. Components that mediate the AI system's access to actuators and to communication channels, such that physical actuation and outbound communication occur only through governed pathways and only under current operational authority.

Process Flow

In various embodiments, the system operates as follows. The AI system generates an action request. The governance systems evaluate, with respect to that request, the Trust Criteria, the Constitutional Guardrails, the sponsor restrictions, the jurisdictional conditions, the settlement conditions, and the operational risk scores. The Operational Authority Resolver resolves operational authority for the request as granted, denied, throttled, sandboxed, partitioned, suspended, or revoked. The Executable Action-Space Constraint Engine dynamically constrains the executable action-space topology so that only authorized pathways are present. Authorized actions execute, and only authorized actions are able to execute, through governed pathways mediated by the Resource Access Governance Layer and the Actuator and Communication Authority Gateways. Monitoring systems continuously update operational authority in response to anomalies, violations, and governance events. A revocation of operational authority removes the AI system's access to required resources, causing partial or total cessation of the AI system's operation.

Exception handling: an action request for which authority resolves to denied does not enter the executable action space; a governance event crossing a revocation threshold causes immediate revocation and resource withdrawal; a detected attempt by the AI system to access a resource outside the governed pathways is itself a governance event that the Dynamic Throttling and Revocation Engine acts upon.

Alternative Embodiments

In a financial-AI embodiment, the architecture governs an AI agent that operates in financial markets, such that the agent's continued operation depends on governed access to settlement and execution resources and its executable action space excludes prohibited transaction types. In a robotics embodiment, the architecture governs an AI system that controls physical actuators, such that actuation occurs only through governed Actuator Authority Gateways and prohibited actuation pathways are cryptographically unavailable. In an enterprise-orchestration embodiment, the architecture governs an AI orchestration agent operating across enterprise systems. In an autonomous-scientific-agent embodiment, the architecture governs an AI agent conducting scientific work, constraining its executable action space to authorized experimental and computational pathways. In a sovereign-AI embodiment, a sovereign entity governs an AI system's operational existence and action space under sovereign Trust Authorities. In a multi-agent embodiment, the architecture governs a system of multiple coordinating AI agents, resolving operational authority and action-space topology for each. In a settlement-dependent-operation embodiment, the AI system's continued operation is made expressly dependent upon ongoing settlement conditions. These embodiments are illustrative and may be combined.

Detailed Component Architecture and Enforcement Mechanics

The following provides component-level detail for the Family T system. The detail is directed to the computational and cryptographic mechanics by which resource-bound operational existence and pre-execution action-space constraint are realized, and is provided both for enablement and to make concrete that the disclosed subject matter is a computational control architecture rather than a governance policy.

Operational Authority Resolver — authority-state model. In various embodiments, the resolver maintains, for each governed AI system, an operational-authority record comprising: an authority-state value drawn from the enumeration granted, throttled, sandboxed, partitioned, suspended, and revoked; a resource-permission map associating each class of governed resource — compute, memory, network, communication, orchestration, settlement, actuator, interoperability — with a permission value indicating whether access to that resource class is currently permitted, denied, or rate-limited, and on what terms; an authority-scope descriptor delimiting the set of operations the AI system is authorized to request; and a set of currently-effective governance conditions. The resolver re-resolves the operational-authority record upon each action request and upon each governance event. Re-resolution evaluates the applicable Trust Criteria, Constitutional Guardrails, sponsor restrictions, jurisdictional conditions, settlement conditions, and operational-risk scores, and computes the authority state and the resource-permission map as a deterministic function of those inputs. Because the operational-authority record is a deterministic function of recorded inputs, the resolution is reproducible and auditable.

Executable Action-Space Constraint Engine — action-graph pruning. In various embodiments, the executable action space of a governed AI system is represented as an executable-action graph, in which nodes represent actions, action classes, or action pathways available to the AI system, and edges represent permissible sequencing and composition of actions. The constraint engine, before the AI system executes, prunes the executable-action graph to the subgraph authorized under the currently resolved operational authority: nodes corresponding to unauthorized actions are removed from, or marked non-traversable within, the graph presented to the AI system's execution layer; edges corresponding to unauthorized sequencing are likewise removed or marked non-traversable. The pruning is performed at the execution layer, such that an unauthorized action is not a node the AI system may select and have filtered, but is absent from the executable-action graph over which the AI system's execution proceeds. In a cryptographically-enforced embodiment, traversal of a node in the executable-action graph requires a valid cryptographic capability token scoped to that node, the token is issued only for authorized nodes by the Constraint Engine, and the execution layer will not effectuate an action absent a valid token; in this embodiment an unauthorized action is not merely absent from a presented graph but is cryptographically non-effectuable, because the capability token required to effectuate it does not exist. This is the sense in which the executable action-space topology is cryptographically constrained prior to execution.

Resource Access Governance Layer — dependency enforcement. In various embodiments, the governed AI system does not hold direct handles to the computational, memory, communication, orchestration, settlement, actuator, and interoperability resources on which its operation depends; instead, every access to such a resource is mediated by the Resource Access Governance Layer, which grants or withholds the access according to the current resource-permission map of the operational-authority record. Because the AI system's operation depends on these mediated resources, and because the mediation is governed, the withdrawal of a resource permission is not an instruction the AI system may decline to follow; it is the removal of a resource without which the corresponding operation cannot proceed. In a resource-bound-existence embodiment, the set of mediated resources includes the compute and memory resources on which the AI system's own continued execution depends, such that a revocation of operational authority that withdraws compute and memory permissions causes the AI system's own execution to halt; in this embodiment operational existence itself, and not merely operational capability, is resource-bound.

Dynamic Throttling and Revocation Engine — governance-event response. In various embodiments, the engine subscribes to governance events — including anomaly detections, Constitutional Guardrail violations, distillation-risk threshold crossings reported by Family S, settlement-condition failures, sponsor-directed interventions, and operational-risk-score threshold crossings — and, upon a governance event, effects a transition of the operational-authority state. The transition may throttle the AI system by tightening rate limits in the resource-permission map; sandbox it by restricting resource permissions to a confined environment; partition its authority across multiple governance domains; suspend it by withdrawing all but a minimal set of resource permissions; or revoke it by withdrawing resource permissions including those on which continued execution depends. The engine's transitions are bounded by the Governance-State Machine, which defines the permissible transitions and the conditions for each.

Constitutional Guardrail Evaluator — guardrail model. In various embodiments, a Constitutional Guardrail is represented as a machine-evaluable constraint comprising a guardrail identifier, a scope descriptor identifying the actions, resources, or interoperability relationships to which the guardrail applies, a condition expression evaluated against an action request and the governance context, and a disposition indicating the consequence of the condition being satisfied or violated. The evaluator evaluates the applicable guardrails against each action request and reports the results to the Operational Authority Resolver and the Executable Action-Space Constraint Engine. Because the guardrails are inseparable from the governed execution environment — an action that violates a guardrail is pruned from the executable-action graph rather than permitted and logged — the guardrails are constitutional in the sense defined in the glossary: foundational to and inseparable from the execution environment rather than external policy applied to it.

Detailed State-Transition Model

In various embodiments, the operational authority of a governed AI system is governed by the Governance-State Machine, the states and transitions of which are here described in detail for the AI-governance case. In the GRANTED state, the resource-permission map permits the resource accesses within the AI system's authority scope, and the executable-action graph is pruned to the authorized subgraph for that scope. A rising operational-risk score crossing a throttle threshold transitions the system to the THROTTLED state, in which rate limits in the resource-permission map are tightened and the executable-action graph is pruned more restrictively. A governance event indicating that the system should be confined transitions it to the SANDBOXED state, in which resource permissions are restricted to a confined environment and the executable-action graph is pruned to actions whose effects are contained within that environment. A determination that the system's authority should be divided transitions it to the PARTITIONED state, in which authority over distinct resource classes or distinct operations is distributed across multiple governance domains, no one of which holds complete authority. A governance event crossing a suspension threshold transitions the system to the SUSPENDED state, in which all but a minimal set of resource permissions are withdrawn and the executable-action graph is pruned to an empty or near-empty subgraph, halting substantive operation while preserving the system's state for possible reinstatement. A governance event crossing a revocation threshold transitions the system to the REVOKED state, in which resource permissions are withdrawn including those on which continued execution depends, so that operation ceases. From SUSPENDED, a governance determination may restore the system to a constrained or granted state; from REVOKED, restoration is not available and a new authorization is required. Every transition is recorded as a Trust-Block-bound event, so that the operational-authority history of the AI system is auditable, and every transition is itself subject to Constitutional Guardrail evaluation, so that the governance of the AI system is itself governed.

Additional Alternative Embodiments

In a centralized embodiment, a single operator operates the Operational Authority Resolver and the Resource Access Governance Layer for the AI systems it deploys. In a decentralized embodiment, operational authority is resolved by consensus among a plurality of independent governance nodes, no one of which can unilaterally grant or withhold authority, so that no single party controls the AI system's operational existence. In a federated embodiment, distinct governance domains govern distinct resource classes — for example one domain governs compute, another governs actuator access, another governs settlement — and the AI system's operation depends on satisfying all of them, so that authority is distributed by construction. In a sovereign embodiment, a sovereign entity operates the governance layer for AI systems within its jurisdiction under sovereign Trust Authorities. In a robotics embodiment, the actuator-authority gateways mediate physical actuators, and the executable-action graph excludes physical-actuation pathways that are not authorized, so that an unauthorized physical action is non-effectuable rather than detected after actuation. In a financial-AI embodiment, the settlement-permission element of the resource-permission map mediates the AI system's ability to effect financial settlement, and the executable-action graph excludes prohibited transaction classes. In a multi-agent embodiment, the architecture governs a plurality of coordinating AI agents, resolving operational authority and pruning the executable-action graph for each, and additionally governing the inter-agent communication pathways as mediated resources. In a tokenized-operational-dependency embodiment, the resource permissions on which the AI system depends are represented as tokens that must be held and periodically renewed, such that operational existence is contingent on the continued governed issuance of the tokens and lapses if renewal is withheld. In a graph-native embodiment, the entire Family T system is expressed as operations over the executable-action graph and a resource-dependency graph, with authority resolution computed as graph operations. These embodiments are illustrative and may be combined.

Cross-Family Integration

Upstream dependencies. Family T operates within the QPN-enabled infrastructure of the incorporated disclosures, and depends on Trust Criteria, Constitutional Guardrails, Quantum DNA, and Proof-of-Trust verification, and on the Governed Agent Loop of Family I of the May 2026 QPN Catalyst Provisional, which it extends.

Downstream consumers. Family T provides the execution-governance substrate relied upon by Family Q when a derivative-formation operation must itself be constrained, by Family S when computed distillation risk must translate into execution constraint, by Family V when AI economic participants must be bounded, and by Family W when sovereign AI systems must be governed.

Lateral interactions. Family T expressly extends Family I: where Family I discloses a Governed Agent Loop in which agent steps pass through a per-step Trust-Criteria gate, Family T extends the AI-governance surface to include resource-bound operational existence, executable action-space topology, and constitutional execution, so that the Governed Agent Loop operates within an execution architecture whose action space is itself cryptographically constrained and whose continued operation is resource-bound. Family T's Governance-State Machine is the same state-machine construct elaborated in the operational-flow embodiments and applied across Families.

Emergent properties. The combination of resource-bound operational existence with pre-execution action-space constraint produces an emergent property absent from external-control architectures: AI safety becomes a structural and cryptographic property of the execution environment. A harmful action pathway that is not in the executable action-space topology cannot

be taken regardless of the model's cognition; an AI system whose operational authority is revoked cannot continue regardless of its propensities, because the resources on which it depends have been withdrawn.

Initial Claims

A computer-implemented artificial intelligence operational governance system comprising one or more processors and memory storing instructions that, when executed, provide: one or more artificial intelligence systems; one or more resource-governance systems; one or more operational-authority systems; one or more executable-action-space constraint systems; and one or more revocation systems; wherein continued operation of the one or more artificial intelligence systems depends upon governed access to one or more authorized resources, and wherein an executable action-space topology associated with the one or more artificial intelligence systems is dynamically constrained, prior to execution, according to one or more governance conditions.

A computer-implemented method comprising: evaluating one or more governance conditions associated with an artificial intelligence system; dynamically constraining an executable action-space topology associated with the artificial intelligence system such that one or more unauthorized actions are computationally inaccessible to the artificial intelligence system prior to execution; governing access by the artificial intelligence system to one or more authorized resources upon which continued operation of the artificial intelligence system depends; and modifying operational authority associated with the artificial intelligence system according to one or more governance events, including by withdrawing access to one or more of the authorized resources.

A computer-implemented system wherein one or more unauthorized actions of an artificial intelligence system are rendered computationally inaccessible to the artificial intelligence system prior to execution by cryptographic constraint of an executable action-space topology.

Family U — Multi-Lab Partitioned AI Coordination

Field Summary

This Family relates to computer-implemented collaborative computation, and more particularly to systems and methods for partitioned collaborative artificial intelligence computation, coordination among mutually distrustful laboratories, sovereign AI federation, and capability-isolated safety research, in which collaborative AI computation is distributed across a plurality of independently governed trust partitions such that protected capability surfaces remain isolated.

Problem Addressed

Frontier artificial intelligence laboratories, sovereign entities, cloud providers, hardware vendors, safety researchers, and enterprises frequently need to collaborate — on safety evaluation, on interpretability research, on adversarial testing, on alignment research, on governance design, or on model comparison. Such collaboration is socially valuable and, for AI safety in particular, may be urgent. Yet the parties typically cannot safely expose to one another the assets that the collaboration would seem to require: their model weights, their proprietary datasets, their evaluation methods, their training infrastructure, or their capability surfaces.

Conventional approaches to this problem require one of: centralized trust, in which all parties trust a common intermediary with their protected assets; unrestricted disclosure, in which the parties expose their assets to one another; bilateral contracts, in which the parties rely on legal

enforcement after the fact; or mutually trusted enclaves, in which the parties rely on a common hardware or software trust root. Each conventional approach fails in the case that motivates the collaboration: where the parties are direct competitors, sovereign rivals, regulated entities subject to incompatible obligations, or otherwise mutually distrustful. A competitor will not expose model weights to a competitor; a sovereign will not expose a strategic dataset to a rival sovereign; a regulated entity may be legally prohibited from the disclosure that unrestricted collaboration requires. The unresolved technical problem is to enable genuine collaborative AI computation among parties who cannot and will not expose their protected capability surfaces to one another or to any common party.

Solution Overview

The disclosed architecture provides partitioned collaborative computation across a plurality of independently governed trust domains. One or more partitioned Privacy Algorithms — a Privacy Algorithm being, as described in the incorporated disclosures, a computational process configured to govern, constrain, authorize, partition, transform, coordinate, evaluate, route, derive, compose, protect, isolate, and interoperate with governed computational activities — distribute the computation, the authorization, the verification, the governance enforcement, and the output aggregation of a collaborative AI workload across multiple Quantum Privacy Domains, multiple sovereign infrastructures, multiple cloud providers, multiple hardware environments, multiple secure enclaves, and multiple laboratory-controlled partitions.

The defining property of the architecture is capability isolation: the collaborative computation is partitioned such that no individual participant necessarily possesses sufficient information to reconstruct the protected weights, datasets, intermediate states, or proprietary capabilities of any other participant, and yet the collaborative computation as a whole produces the intended result. Participants may collaboratively train, evaluate, test, compare, interpret, or govern AI systems; the protected capability surfaces of each participant remain within that participant's own trust partition; and the permitted outputs of the collaboration are aggregated, by a Controlled Output Aggregator, according to Trust Criteria and output-governance restrictions. The architecture thereby enables collaboration that conventional approaches cannot, because it removes the precondition — exposure of protected assets to some party — that conventional approaches cannot satisfy.

Components

- **Partitioned Privacy Algorithm Executor.** A component that distributes a collaborative AI computation across a plurality of trust partitions according to one or more partitioned Privacy Algorithms, such that each partition performs a portion of the computation locally under its own governance.
- **Capability-Isolation Layer.** A component that enforces the isolation of each participant's protected capability surfaces — model weights, datasets, intermediate states, proprietary capabilities — within that participant's trust partition, such that those surfaces are not exposed to other partitions or participants.
- **Federated Trust Graph.** A graph data structure representing the trust partitions, the participants, the governance relationships among them, and the distribution of the collaborative computation across them, traversed to coordinate the collaboration.

- **Collaborative Safety Evaluation Engine.** A component that conducts collaborative AI safety evaluation, interpretability analysis, adversarial testing, or model comparison across the partitions, producing permitted evaluation outputs without requiring exposure of the evaluated models' protected surfaces.
- **Cross-Sovereign Governance Coordinator.** A component that coordinates governance across partitions belonging to different sovereign or institutional governance regimes, resolving governance compatibility while preserving each partition's local governance autonomy.
- **Controlled Output Aggregator.** A component that aggregates the permitted outputs of the partitioned computation according to Trust Criteria and output-governance restrictions, producing the collaborative result while ensuring that the aggregation does not reconstruct protected surfaces.
- **Distributed Verification Engine.** A component that verifies, across the partitions, the integrity of the distributed execution and the compatibility of the governance applied in each partition, without requiring any partition to expose protected content to the verifier.
- **Adversarial Trust Distribution Layer.** A component that distributes trust across the partitions in a manner that remains sound under the assumption that the participants are mutually distrustful and may be adversarial, such that no single participant or coalition below a governance-defined threshold can subvert the collaboration or extract protected surfaces.

Process Flow

In various embodiments, the participants define the collaborative computation objective and identify the protected capability surfaces that must remain isolated. The Partitioned Privacy Algorithm Executor distributes the computation across the trust partitions. Each partition executes its locally governed portion of the computation under its own local Trust Criteria. The Capability-Isolation Layer ensures that protected weights, datasets, intermediate states, and proprietary capabilities remain within their partitions. The Distributed Verification Engine confirms execution integrity and governance compatibility across partitions. The Controlled Output Aggregator produces the permitted collaborative outputs. The collaborative outputs inherit Trust Blocks and derivative restrictions, consistent with the output-governance mechanics of Family S and the Trust Block inheritance mechanics of the incorporated disclosures.

Exception handling: a partition that fails verification is excluded and the disposition is recorded; an attempt by a partition to obtain information beyond its authorized share is detected by the Adversarial Trust Distribution Layer and treated as a governance event; an aggregation that would reconstruct a protected surface is not performed.

Alternative Embodiments

In a multi-lab safety-benchmark embodiment, several frontier laboratories collaboratively evaluate one another's models against a shared safety benchmark without exposing weights. In a collaborative-interpretability embodiment, interpretability research is conducted across partitioned model and tooling surfaces. In a federated-sovereign-evaluation embodiment, sovereign entities collaboratively evaluate AI systems while each retains its strategic assets within its own infrastructure. In a cloud-and-hardware partitioned embodiment, the computation is partitioned across cloud-provider and hardware-vendor environments. In a secure-enclave embodiment, partitions are realized within secure enclaves and trusted-execution environments. In a cross-lab governed-comparison embodiment, models are compared or governed-distilled across labs under partitioned governance. In a public-benefit AI-safety-consortium embodiment, a

consortium of public and private parties conducts collaborative safety research as a public-benefit activity, with results routed as described in Family W. These embodiments are illustrative and may be combined.

Detailed Component Architecture and Partition Mechanics

The following provides component-level detail for the Family U system, directed to the mechanics by which a collaborative AI computation is partitioned across mutually distrustful trust domains while protected capability surfaces remain isolated:

- **Partitioned Privacy Algorithm Executor — partition planning and distribution.** In various embodiments, the executor receives a specification of a collaborative AI computation and a specification, from each participant, of the protected capability surfaces that participant requires to remain isolated. The executor computes a partition plan that decomposes the collaborative computation into a set of partition-local computations and a set of cross-partition coordination operations, such that each partition-local computation operates only upon the resources resident within, or authorized to, a single trust partition, and the cross-partition coordination operations exchange only those intermediate results that the governing Trust Criteria permit to cross partition boundaries. The partition plan is computed subject to the constraint that the protected capability surfaces specified by each participant are assigned to that participant's own trust partition and are not included in any cross-partition exchange. The executor then distributes the partition-local computations to the respective trust partitions for local execution under local governance.
- **Capability-Isolation Layer — isolation invariant.** In various embodiments, the capability-isolation layer enforces an isolation invariant: for each protected capability surface, the information made available outside the surface's home trust partition over the course of the collaborative computation is limited to the cross-partition exchanges permitted by the governing Trust Criteria, and those permitted exchanges are constrained such that the protected capability surface cannot be reconstructed from them. In various embodiments the layer enforces the invariant by mediating every cross-partition exchange, evaluating each proposed exchange against the Trust Criteria and against a reconstruction-risk assessment analogous to the distillation-risk assessment of Family S, and permitting the exchange only if it does not, alone or in combination with prior exchanges, raise the cumulative reconstruction risk for any protected surface above a governed threshold.
- **Federated Trust Graph — structure.** In various embodiments, the federated trust graph comprises partition nodes representing the trust partitions, participant nodes representing the participants, resource nodes representing the resources assigned to each partition, and computation nodes representing the partition-local computations and cross-partition coordination operations; edges represent assignment of resources to partitions, governance of partitions by participants, data dependencies among computations, and the permitted cross-partition exchanges. The graph is the structure over which the Partitioned Privacy Algorithm Executor computes the partition plan, the Distributed Verification Engine verifies execution, and the Cross-Sovereign Governance Coordinator resolves cross-domain governance.
- **Distributed Verification Engine — verification without disclosure.** In various embodiments, the engine verifies that each partition-local computation executed in accordance with its specification and its local governance, and that the cross-partition exchanges conformed to the partition plan and the Trust Criteria, without requiring any partition to disclose its protected capability surfaces to the verifier. In various embodiments the engine effects this by means of

verification artifacts produced by each partition — for example cryptographic attestations of execution integrity, consistency proofs over the cross-partition exchanges, or attestations produced within secure enclaves — that establish the verified properties without revealing the protected content. A partition that fails verification is excluded from the aggregation and the exclusion is recorded.

- **Adversarial Trust Distribution Layer — threshold soundness.** In various embodiments, the layer distributes the collaborative computation, the authorization, and the verification across the trust partitions such that the isolation invariant and the integrity of the collaborative result remain sound under the assumption that the participants are mutually distrustful and that some participants may be adversarial. In various embodiments the layer enforces a threshold-soundness property: no single participant, and no coalition of participants smaller than a governed threshold, can either reconstruct a protected capability surface of a non-coalition participant or corrupt the collaborative result without detection. The threshold is a governed parameter set according to the trust assumptions of the particular collaboration.
- **Controlled Output Aggregator — governed aggregation.** In various embodiments, the aggregator combines the permitted outputs of the partition-local computations into the collaborative result according to the Trust Criteria and the output-governance restrictions, subject to the constraint that the aggregation itself does not reconstruct a protected capability surface. The collaborative result inherits Trust Blocks and derivative restrictions in accordance with the Trust Block inheritance mechanics of the incorporated disclosures, so that the result of the collaboration is itself a governed resource.

State-Transition Model

In various embodiments, a partitioned collaborative computation progresses through a state machine. The computation begins in a SPECIFIED state upon receipt of the collaborative-computation specification and the protected-surface specifications. It transitions to a PLAN-COMPUTED state when the Partitioned Privacy Algorithm Executor has computed the partition plan; a partition plan that cannot satisfy the isolation constraints transitions the computation to an INFEASIBLE state and the collaboration does not proceed. From PLAN-COMPUTED the computation transitions to a DISTRIBUTED state when the partition-local computations have been distributed to the trust partitions. It transitions to an EXECUTING state during partition-local execution. It transitions to a VERIFYING state during distributed verification; a partition that fails verification is excluded and recorded, and if too few partitions remain for a sound result the computation transitions to FAILED. From VERIFYING, upon successful verification, the computation transitions to an AGGREGATING state during controlled aggregation, and then to a COMPLETED state when the collaborative result has been produced and bound to its Trust Blocks. Each transition is recorded as a Trust-Block-bound event.

Additional Alternative Embodiments

In a multi-frontier-lab embodiment, the participants are frontier AI laboratories collaborating on safety evaluation, each laboratory's model weights constituting a protected capability surface assigned to that laboratory's partition, and the collaborative result being a set of safety-evaluation outputs that reveal the evaluated properties without revealing the weights. In a sovereign-federation embodiment, the participants are sovereign entities, each sovereign's strategic datasets and models constituting protected surfaces within that sovereign's infrastructure, and the federation conducted under the Cross-Sovereign Governance Coordinator with no sovereign subordinated to another. In a cloud-and-hardware-partition embodiment, the trust partitions are

realized across cloud-provider and hardware-vendor environments, and the Distributed Verification Engine relies in part on hardware-rooted attestations. In a secure-enclave embodiment, partition-local computations execute within secure enclaves and trusted-execution environments, and the verification artifacts are enclave attestations. In a collaborative-interpretability embodiment, the collaborative computation is interpretability research conducted across partitioned model and tooling surfaces. In a regulated-entity embodiment, the participants are entities subject to incompatible regulatory regimes, and the partition plan is computed subject to the constraint that no cross-partition exchange violates any participant's regulatory obligations, so that collaboration proceeds without any participant being required to make a disclosure its regulator prohibits. In a public-benefit-consortium embodiment, a consortium conducts collaborative AI safety research as a public-benefit activity and the collaborative result is routed in part to public-benefit structures as described in Family W. In a graph-native embodiment, the partition planning, verification, and aggregation are expressed as operations over the federated trust graph. These embodiments are illustrative and may be combined.

Cross-Family Integration

Upstream dependencies. Family U operates within the QPN-enabled infrastructure of the incorporated disclosures, and depends on Privacy Algorithms, Trust Criteria, Proof-of-Trust verification, and QPDs.

Downstream consumers. Family U is consumed by Family Q, whose derivative formation may be conducted across partitions; by Family R, whose evaluation stage may be collaborative; by Family S, whose inference governance may span partitions; and by Family W, whose sovereign and public-benefit collaborations rely on partitioned coordination.

Lateral interactions. Family U's Federated Trust Graph is related to, and may be composed with, the governance and settlement graphs described in the graph-theoretic embodiments below. Its Controlled Output Aggregator interoperates with Family S output governance, and its collaborative outputs participate in settlement through Families G and J of the May 2026 QPN Catalyst Provisional.

Emergent properties. Partitioned collaborative computation produces an emergent property: collaboration becomes possible precisely among the parties for whom conventional collaboration is impossible. Because the architecture removes the precondition of asset exposure, mutually distrustful frontier laboratories and sovereign rivals can collaborate on AI safety — an outcome of independent public value.

Initial Claims

A computer-implemented collaborative artificial intelligence coordination system comprising one or more processors and memory storing instructions that, when executed, provide: a plurality of independently governed trust partitions; one or more partitioned Privacy Algorithms distributed across the trust partitions; one or more capability-isolation systems; and one or more controlled-output aggregation systems; wherein collaborative artificial intelligence computation occurs across the plurality of independently governed trust partitions without unrestricted disclosure of one or more protected computational capability surfaces, and= wherein no individual trust partition necessarily possesses information sufficient to reconstruct a protected computational capability surface of another trust partition.

A computer-implemented method comprising: partitioning an artificial intelligence computation across a plurality of independently governed infrastructures; restricting visibility into one or more protected computational states associated with each infrastructure; collaboratively executing an

artificial intelligence workflow across the plurality of independently governed infrastructures; and aggregating one or more permitted outputs of the artificial intelligence workflow according to one or more governance conditions.

Family V — AI-Native Economic Coordination Systems

Field Summary

This Family relates to computer-implemented economic coordination, and more particularly to systems and methods for artificial-intelligence-mediated economic coordination, autonomous accelerator formation, recursive settlement optimization, liquidity coordination, and the operation of artificial intelligence systems as first-class governed economic participants within a Quantum Privacy Network.

Problem Addressed

Conventional economic systems rely on manually coordinated transactions, institutionally fragmented workflows, centralized planning, and human-directed venture and accelerator formation. Within such systems, artificial intelligence is typically deployed as an isolated productivity tool — an instrument used by a human or an institution to perform a task — rather than as a governed economic participant in its own right. Conventional systems provide no mechanism by which an AI system could hold attribution lineage, settlement participation, governance obligations, derivative-right participation, operational authority, and recursive coordination responsibilities as a first-class participant. Consequently, the substantial coordination work that AI systems are increasingly capable of performing — discovering opportunities, structuring ventures, optimizing the routing of liquidity, coordinating participants — cannot be conducted by the AI system as a governed and accountable participant whose contributions are attributed and settled. The unresolved technical problem is the absence of an architecture in which AI systems are governed economic participants rather than ungoverned tools.

Solution Overview

The disclosed architecture permits artificial intelligence systems to act as governed economic participants within Exchange Networks, Accelerator Networks, Resource Pools, Contribution Graphs, settlement systems, liquidity systems, and recursively composable derivative ecosystems, each as disclosed in the incorporated documents and the May 2026 QPN Catalyst Provisional. An AI coordination system within the architecture may discover exchange opportunities, form accelerators, generate venture proposals, coordinate participants, optimize liquidity routing, evaluate governance compatibility, structure investment participation, and recursively coordinate further AI systems and derivative ecosystems.

Critically, the AI economic coordination occurs within Trust Block-governed environments, and the AI system participates subject to governance: its operational authority is governed as described in Family T, its actions are bounded by Constitutional Guardrails and Trust Criteria, and the economic structures it generates inherit governance continuity, settlement participation, attribution lineage, and derivative obligations. The AI system is a first-class participant in the sense that it holds attribution lineage and settlement participation of its own, and it is a governed participant in the sense that its economic action is bounded and accountable. The architecture thereby permits AI systems to perform economic coordination work at a scale and speed beyond manual coordination, while ensuring that the work is attributed, governed, and settled.

Components

- **AI Economic Opportunity Detector.** A component by which an AI system analyzes Resource Pools, Contribution Graphs, governance conditions, liquidity availability, and settlement flows to identify economic coordination opportunities.
- **AI Accelerator Formation Engine.** A component by which an AI system generates accelerator structures, consistent with the Accelerator Network mechanics of the incorporated disclosures and Family X.
- **AI Settlement Optimization Engine.** A component by which an AI system optimizes the routing and structuring of settlement participation across an ecosystem.
- **AI Liquidity Router.** A component by which an AI system optimizes the routing of liquidity, consistent with the Liquidity Architecture of Family N of the May 2026 QPN Catalyst Provisional.
- **AI Governance Compatibility Evaluator.** A component by which an AI system evaluates, through Trust Criteria, the governance compatibility of a proposed economic structure or coordination relationship.
- **Recursive AI Orchestration System.** A component by which an AI system recursively coordinates further AI systems and derivative coordination ecosystems, such that AI economic coordination is itself recursively composable.
- **AI Contribution Attribution Engine.** A component that records, in the Contribution Graph, the contributions made by the AI system as a participant, so that the AI system's coordination work is attributed and may participate in settlement.

Process Flow

In various embodiments, the AI system analyzes Resource Pools, Contribution Graphs, governance conditions, liquidity availability, and settlement flows. The AI system identifies one or more coordination opportunities. The AI system generates an accelerator, venture, settlement, or governance coordination structure. The governance compatibility of the generated structure is evaluated through Trust Criteria. Participation and settlement structures are initialized for the generated structure. The AI system recursively coordinates downstream AI systems or derivative ecosystems. The outputs of the AI system's coordination work are registered as governed derivative coordination resources, with attribution recorded in the Contribution Graph and settlement participation initialized.

Exception handling: a generated structure that fails governance-compatibility evaluation is not initialized, and the disposition is recorded; an AI coordination action that exceeds the AI system's operational authority, as governed under Family T, does not execute.

Alternative Embodiments

In an AI-generated-scientific-venture embodiment, the AI system structures ventures arising from scientific contribution lineage. In an AI-managed-accelerator embodiment, the AI system forms and coordinates accelerator ecosystems. In an AI-liquidity-optimization embodiment, the AI system's principal role is the optimization of liquidity routing. In an AI-mediated-sovereign-development embodiment, the AI system coordinates sovereign development networks, consistent with Family W. In an AI-governed-public-benefit-allocation embodiment, the AI system coordinates the allocation of public-benefit resources. In an AI-generated-startup embodiment, the AI system generates startup structures, consistent with Family Z. In a recursive AI-to-AI market-

coordination embodiment, AI systems coordinate with one another recursively, forming a market of AI coordination participants. These embodiments are illustrative and may be combined.

Cross-Family Integration

Upstream dependencies. Family V operates within the QPN-enabled infrastructure of the incorporated disclosures, and depends on Exchange Networks, Accelerator Networks, Resource Pools, Contribution Graphs, the Settlement Controller, the Premium Framework, and the Liquidity Architecture of the May 2026 QPN Catalyst Provisional, and on the governed-execution mechanics of Family T.

Downstream consumers. Family V is consumed by Family X, whose recursive accelerator formation may be AI-driven; by Family Y, whose externality optimization may be AI-driven; and by Family Z, whose venture formation is AI-mediated.

Lateral interactions. Family V depends on Family T to bound the AI participant's operational authority; on Family Q where the AI participant forms governed derivatives; and on the settlement systems of Families G and J for the routing of the settlement structures it generates.

Emergent properties. Permitting AI systems to act as governed economic participants — rather than as ungoverned tools — produces an emergent property: economic coordination work can be conducted at machine scale and speed while remaining attributed, governed, and accountable, because the AI participant's contributions are recorded in the Contribution Graph and its authority is governed under Family T.

Initial Claims

A computer-implemented system for artificial-intelligence-mediated economic coordination comprising one or more processors and memory storing instructions that, when executed, provide: one or more artificial intelligence coordination systems; one or more Exchange Networks; one or more resource-allocation systems; one or more governance-evaluation systems; and one or more settlement systems; wherein the one or more artificial intelligence coordination systems autonomously coordinate economic interactions within one or more governed ecosystems, and wherein economic structures generated by the artificial intelligence coordination systems inherit governance continuity and settlement participation.

A computer-implemented method comprising: analyzing, by an artificial intelligence coordination system, one or more governed ecosystems; identifying one or more coordination opportunities; generating one or more derivative economic structures; coordinating settlement participation associated with the one or more derivative economic structures; and recursively coordinating one or more downstream systems.

Family W — Sovereign and Public-Benefit AI Governance Networks

Field Summary

This Family relates to computer-implemented governance coordination, and more particularly to systems and methods for sovereign accelerator systems, constituent-beneficiary trusts, public-benefit settlement routing, and cryptographically governed public-resource ecosystems, in which sovereign entities participate as governed ecosystem participants and public-resource-derived

value is routed to constituents and public-benefit structures through verifiable settlement systems.

Problem Addressed

Conventional sovereign digital infrastructure is fragmented across agencies, vendors, jurisdictions, contractors, and policy regimes. The value derived from public resources is rarely routed directly to constituents or to public-benefit structures through verifiable settlement systems; instead it is mediated by institutional intermediaries, subject to fragmentation and leakage, and difficult to trace. Sovereign artificial-intelligence efforts, in particular, conventionally depend on centralized state control, on vendor lock-in, or on siloed regulatory architectures, none of which permits a sovereign to participate in a broader governed ecosystem while preserving its autonomy.

The unresolved technical problem is the absence of an architecture for cryptographically governed sovereign coordination — an architecture in which a government participates as a governed ecosystem participant rather than as a unilateral controller, in which constituents and public-benefit trusts receive recursively governed and verifiable settlement participation in the value derived from public resources, and in which sovereign ecosystems can interoperate while each preserves its local governance autonomy.

Solution Overview

The disclosed architecture provides sovereign and public-benefit governance coordination within QPN ecosystems. Sovereign entities, municipalities, public-benefit trusts, enterprises, contractors, constituents, AI systems, and derivative ecosystems participate within sovereign accelerators and public-benefit Exchange Networks. Public-resource-derived settlement flows may be routed toward constituent-beneficiary trusts, environmental trusts, public infrastructure systems, social-benefit systems, scientific trusts, or governance-weighted public-allocation pools, according to governance-weighted public-benefit policies expressed as machine-evaluable Trust Criteria.

A government participates subject to Trust Criteria, Quantum DNA inheritance, Constitutional Guardrails, and interoperability-governance systems, in the same governed manner as any other participant — that is, as a governed participant rather than as a unilateral controller — while the architecture preserves the government's localized sovereignty over its own domain. Sovereign ecosystems federate with one another through interoperability-governance systems that resolve cross-sovereign governance compatibility without subordinating any sovereign to another. The architecture thereby permits public value to be routed verifiably to the public, permits governments to participate in governed ecosystems without surrendering sovereignty, and permits sovereigns to interoperate without centralization.

Components

- **Sovereign Accelerator Formation Engine.** A component that forms sovereign accelerators — accelerator structures, consistent with the Accelerator Network mechanics of the incorporated disclosures and Family X, that are sponsored by or organized around sovereign and public-benefit participants.
- **Constituent-Beneficiary Trust Router.** A component that routes public-resource-derived settlement flows toward constituent-beneficiary trust structures, environmental trusts, public infrastructure systems, social-benefit systems, and scientific trusts, according to governance-weighted policies.

- **Public-Benefit Settlement Allocator.** A component that allocates settlement to public-benefit structures and governance-weighted public-allocation pools.
- **Cross-Sovereign Federation Layer.** A component that federates sovereign ecosystems through interoperability-governance systems, resolving cross-sovereign governance compatibility while preserving each sovereign's local autonomy.
- **Government-as-Participant Governance Model.** A component embodying the model in which a government participates as a governed ecosystem participant — subject to Trust Criteria, Constitutional Guardrails, and Quantum DNA inheritance — rather than as a unilateral controller.
- **Jurisdiction-Aware Trust Criteria Engine.** A component that evaluates Trust Criteria with awareness of jurisdictional conditions, so that participation and settlement comply with the requirements of the relevant jurisdictions.
- **Sovereign AI Governance Coordinator.** A component that coordinates the governance of AI systems operating within sovereign ecosystems, consistent with Families T and U.

Process Flow

In various embodiments, the system registers sovereign, public-benefit, enterprise, contractor, and constituent participants. It establishes sovereign accelerator governance conditions. It defines public-resource contribution and settlement structures. It routes public-resource-derived settlement according to governance-weighted public-benefit policies, by means of the Constituent-Beneficiary Trust Router and the Public-Benefit Settlement Allocator. It federates sovereign ecosystems through the Cross-Sovereign Federation Layer. It preserves local governance autonomy while maintaining recursive interoperability.

Exception handling: a settlement routing that would violate a jurisdictional Trust Criterion is not performed; a federation relationship that fails cross-sovereign governance-compatibility evaluation is not established; a participation by a government participant that would exceed the Government-as-Participant Governance Model is not permitted.

Alternative Embodiments

In a healthcare sovereign-accelerator embodiment, a sovereign forms a healthcare accelerator and routes public-health-derived value to constituent-beneficiary trusts. In an environmental constituent-beneficiary-trust embodiment, environmental public-resource value is routed to environmental trusts. In a municipal public-benefit-exchange embodiment, a municipality operates a public-benefit Exchange Network. In a cross-border sovereign AI-federation embodiment, multiple sovereigns federate their AI ecosystems while preserving autonomy. In a public-infrastructure settlement-routing embodiment, value derived from public infrastructure is routed verifiably to public-benefit structures. In a public-benefit accelerator-network embodiment, a network of public-benefit accelerators is coordinated. These embodiments are illustrative and may be combined.

Cross-Family Integration

Upstream dependencies. Family W operates within the QPN-enabled infrastructure of the incorporated disclosures, depends on the Accelerator Network and settlement mechanics of the May 2026 QPN Catalyst Provisional, and depends on Families T and U for the governance of sovereign AI systems and sovereign collaborative computation.

Downstream consumers. Family W is consumed by Family X, whose recursive accelerator formation includes sovereign accelerators generating derivative accelerators, and by Family Y, whose externality internalization routes public-benefit value.

Lateral interactions. Family W's Cross-Sovereign Federation Layer interoperates with the interoperability-governance systems described in the omnibus material; its public-benefit routing interoperates with Family Y's governance-weighted settlement.

Emergent properties. Embodying government as a governed participant rather than a unilateral controller produces an emergent property: sovereigns may participate in, and federate within, a broader governed ecosystem without surrendering sovereignty, and public-resource value becomes verifiably traceable to the public — outcomes unattainable where sovereign infrastructure is fragmented and centrally controlled.

Initial Claims

A computer-implemented sovereign governance coordination system comprising one or more processors and memory storing instructions that, when executed, provide: one or more sovereign accelerators; one or more constituent-beneficiary trust structures; one or more governed settlement systems; and one or more interoperability-governance systems; wherein public-resource-derived settlement flows are routed according to one or more governance conditions, and wherein one or more sovereign computational ecosystems interoperate while preserving localized governance autonomy.

A computer-implemented method comprising: generating one or more settlement flows from one or more governed public-resource ecosystems; routing one or more portions of the settlement flows to one or more public-benefit or constituent-beneficiary structures according to one or more governance-weighted policies; and governing participation of one or more sovereign participants through one or more cryptographic governance systems such that a sovereign participant participates as a governed participant.

Family X — Recursive Accelerator & Meta-Governance Systems

Field Summary

This Family relates to computer-implemented governance coordination, and more particularly to systems and methods for recursive accelerator formation, meta-governance, governance inheritance, recursively composable incentive structures, and accelerator-of-accelerators coordination, in which an accelerator ecosystem generates derivative accelerator ecosystems that inherit governance continuity, incentive structures, and settlement participation.

Problem Addressed

Conventional accelerators are isolated organizational structures. A conventional accelerator does not generate derivative accelerators that inherit its governance; it does not propagate its incentive structures, its settlement participation, or its governance continuity to derivative structures; and there is no meta-governance system capable of coordinating an ecosystem of accelerators across sectors, jurisdictions, and derivative-resource graphs. The unresolved technical problem is the absence of recursively composable accelerator governance — a mechanism by which an

accelerator may generate a derivative accelerator while preserving governance continuity, incentive structures, settlement participation, and interoperability, and by which a meta-governance system may coordinate the resulting ecosystem of accelerators.

Solution Overview

The disclosed architecture provides recursive accelerator formation and meta-governance coordination. An accelerator ecosystem may generate derivative accelerators, derivative governance structures, derivative settlement systems, derivative incentive pools, derivative liquidity ecosystems, derivative venture ecosystems, and recursively composable coordination networks. Governance propagates to each derivative accelerator through Trust Blocks, Quantum DNA inheritance, Constitutional Guardrails, governance templates, derivative-right structures, and settlement-participation obligations — consistent with the Trust Block inheritance and Quantum DNA inheritance mechanics of the incorporated disclosures. A meta-governance system coordinates, across an ecosystem of accelerators, the governance compatibility, the interoperability, the dispute resolution, the settlement routing, the constitutional compatibility, and the cross-network coordination of the accelerators. The architecture thereby permits an accelerator to function as an upstream resource that generates derivative accelerators — an accelerator-of-accelerators — with governance, incentives, and settlement propagating recursively across accelerator generations.

Components

- **Derivative Accelerator Generator.** A component by which an upstream accelerator generates a derivative accelerator as a governed ecosystem.
- **Governance Template Inheritance Engine.** A component that propagates governance templates from an upstream accelerator to a derivative accelerator, such that the derivative accelerator inherits the upstream governance structure.
- **Recursive Incentive Propagation System.** A component that propagates incentive structures and settlement-participation obligations across accelerator generations.
- **Meta-Governance Coordinator.** A component that coordinates governance compatibility, interoperability, dispute resolution, settlement routing, constitutional compatibility, and cross-network coordination across an ecosystem of accelerators.
- **Settlement-Density Optimization Engine.** A component that optimizes the density and routing of settlement across the accelerator ecosystem.
- **Accelerator Cascade Dynamics Analyzer.** A component that analyzes the cascade dynamics by which accelerators generate derivative accelerators, consistent with the critical-mass and topology-formation mechanics of the Accelerator Network of the incorporated disclosures.
- **Recursive Governance Compatibility Evaluator.** A component that evaluates governance compatibility recursively across accelerator generations.

Process Flow

In various embodiments, an upstream accelerator identifies a derivative accelerator opportunity. A derivative accelerator governance template is generated and evaluated. Governance, settlement, attribution, and incentive structures propagate from the upstream accelerator to the derivative accelerator. The derivative accelerator is instantiated as a governed ecosystem. Meta-governance systems coordinate interoperability among the upstream and derivative accelerators. Settlement participation and incentive rights recursively propagate across accelerator generations.

Exception handling: a derivative accelerator whose governance template fails compatibility evaluation is not instantiated; a propagation that would violate a Constitutional Guardrail is not performed; a cascade that the Accelerator Cascade Dynamics Analyzer identifies as governance-incompatible is not coordinated.

Alternative Embodiments

In a healthcare embodiment, a healthcare accelerator generates derivative genomics, diagnostics, and care-coordination accelerators. In a sovereign embodiment, a sovereign accelerator generates derivative municipal or agency accelerators, consistent with Family W. In a scientific embodiment, a scientific accelerator generates derivative climate, materials, biotechnology, and AI-safety accelerators. In an AI-generated-cascade embodiment, the derivative accelerators are generated by AI coordination systems as described in Family V. In a public-benefit accelerator-of-accelerators embodiment, the recursive accelerator ecosystem is organized for public benefit. These embodiments are illustrative and may be combined.

Cross-Family Integration

Upstream dependencies. Family X operates within the QPN-enabled infrastructure of the incorporated disclosures, depends on the Accelerator Network mechanics of the May 2026 QPN Catalyst Provisional, and depends on Family O Quantum DNA inheritance for governance propagation.

Downstream consumers. Family X is consumed by Family W, whose sovereign accelerators are recursive; by Family Z, whose venture formation may be coordinated by recursive accelerators; and by the omnibus recursive-coordination substrate.

Lateral interactions. Family X's Derivative Accelerator Generator interoperates with Family V's AI Accelerator Formation Engine where derivative accelerators are AI-generated; its Meta-Governance Coordinator interoperates with the governance graphs of the omnibus material.

Emergent properties. Recursive accelerator formation with governance inheritance produces an emergent property: an accelerator ecosystem can expand across sectors, jurisdictions, and derivative generations while governance, incentives, and settlement remain continuous — a structural basis for civilization-scale coordination addressed in the omnibus material.

Initial Claims

A computer-implemented recursive accelerator formation system comprising one or more processors and memory storing instructions that, when executed, provide: one or more accelerator ecosystems; one or more derivative-accelerator generation systems; one or more governance inheritance systems; and one or more recursively composable settlement systems; wherein one or more derivative accelerator ecosystems are generated from one or more upstream accelerator ecosystems and inherit governance continuity, incentive structures, and settlement participation from the upstream accelerator ecosystems.

A computer-implemented method comprising: generating one or more derivative governance ecosystems from one or more upstream accelerator ecosystems; propagating one or more governance structures from the upstream accelerator ecosystems to the derivative governance ecosystems; preserving interoperability among the upstream and derivative ecosystems; and coordinating settlement participation across recursive accelerator generations.

Family Y — Externality Internalization and Universal Capitalism Systems

Field Summary

This Family relates to computer-implemented economic coordination, and more particularly to systems and methods for governance-weighted externality attribution, adaptive incentive fields, long-horizon value attribution, and harmonization of public and private value, in which externalities are computationally measured, attributed, weighted, and routed through settlement systems.

Problem Addressed

Conventional economic systems fail to internalize externalities. Environmental impact, public-health benefit, scientific contribution, educational value, governance contribution, and long-horizon societal value commonly remain unpriced, weakly attributable, or disconnected from settlement systems. The conventional instruments for addressing externalities — regulation, taxation, subsidy, litigation, and centralized policy intervention — are slow, fragmented, and frequently incapable of dynamically attributing distributed positive and negative impacts across recursive ecosystems. The unresolved technical problem is the absence of a mechanism by which externalities can be computationally measured, attributed through verifiable contribution lineage, governance-weighted, and routed through settlement systems, dynamically and at the granularity at which the externalities actually arise.

Solution Overview

The disclosed architecture provides cryptographically governed externality internalization within distributed settlement ecosystems. Externalities are measured, attributed, weighted, and routed through settlement systems. The externalities subject to attribution include, without limitation: environmental impact; scientific contribution; public-health outcomes; educational benefit; infrastructure value; governance contribution; social value; and long-horizon societal impact, whether positive or negative, direct or indirect, local or global, immediate or long-horizon. Externality attribution is effected through contribution-lineage structures — the Contribution Graphs and recursively propagating attribution relationships of the incorporated disclosures — so that an externality is attributed to the resources, participants, and derivative ecosystems that gave rise to it. Settlement routing dynamically adjusts according to Governance Premiums, Adaptive Premiums, contribution lineage, public-benefit weighting, and long-horizon value analysis, consistent with the Premium Framework of Family J of the May 2026 QPN Catalyst Provisional. Artificial intelligence systems, governed as in Families T and V, may optimize incentive-field alignment, externality pricing, governance weighting, and recursively composable harmonization of public and private value. The architecture thereby permits externalities to be internalized computationally and dynamically, at the granularity at which they arise, rather than through slow and fragmented centralized intervention.

Components

- **Externality Attribution Engine.** A component that attributes measured externalities, through contribution-lineage structures, to the resources, participants, and derivative ecosystems that gave rise to them.
- **Governance-Weighted Settlement Router.** A component that routes settlement adjustments, rewards, obligations, and public-benefit allocations according to governance weighting and attributed externalities.

- **Long-Horizon Value Attribution System.** A component that attributes value arising over long horizons, including value that conventional systems leave unpriced because it accrues beyond conventional measurement windows.
- **Adaptive Incentive-Field Optimizer.** A component that adjusts the incentive field — the distribution of incentives across an ecosystem — adaptively, in response to attributed externalities and governance weighting.
- **Public/Private Value Harmonization Engine.** A component that harmonizes public and private value, recursively, so that private economic activity and public benefit are coordinated rather than opposed.
- **AI Externality Analysis System.** A component by which governed AI systems analyze externalities, consistent with Families T and V.
- **Recursive Public-Benefit Propagation Layer.** A component that propagates public-benefit attribution across derivative ecosystems and across generations.

Process Flow

In various embodiments, the system detects economic, environmental, scientific, social, governance, or public-benefit activity. It measures the direct, indirect, local, global, immediate, and long-horizon externalities of that activity. It attributes the externalities through contribution-lineage structures. It applies governance weighting and Adaptive Premiums. It routes settlement adjustments, rewards, obligations, or public-benefit allocations accordingly. It propagates the attribution across derivative ecosystems.

Exception handling: an externality whose attribution cannot be verified through contribution lineage is recorded as unattributed pending further analysis rather than routed; a settlement adjustment that would violate a Constitutional Guardrail is not performed.

Alternative Embodiments

In an environmental embodiment, environmental impact is measured, attributed, and routed through settlement. In a public-health embodiment, public-health outcomes are attributed and rewarded. In a scientific embodiment, scientific contribution is attributed and its downstream value propagated to contributors. In an education embodiment, educational and workforce-development externalities are internalized. In an infrastructure embodiment, the value of infrastructure is attributed. In a multi-generation embodiment, public-benefit settlement is propagated across multiple generations of contributors and beneficiaries. These embodiments are illustrative and may be combined.

Cross-Family Integration

Upstream dependencies. Family Y operates within the QPN-enabled infrastructure of the incorporated disclosures, and depends on the Contribution Graph and Premium Framework of the May 2026 QPN Catalyst Provisional and on the governed-AI mechanics of Families T and V.

Downstream consumers. Family Y is consumed by Family W, whose public-benefit routing it informs, and by Family Z, whose venture formation may account for attributed externalities.

Lateral interactions. Family Y's Governance-Weighted Settlement Router interoperates with the Settlement Controller of Family G and the Premium Framework of Family J; its AI Externality Analysis System interoperates with Family V.

Emergent properties. Computational, lineage-based externality attribution produces an emergent property: externalities are internalized dynamically and at the granularity at which they arise, so that the incentive field of an ecosystem reflects the full social value and cost of activity rather than only its privately captured portion.

Initial Claims

A computer-implemented externality-governed settlement coordination system comprising one or more processors and memory storing instructions that, when executed, provide: one or more externality-attribution systems; one or more governance-weighted settlement systems; one or more contribution-lineage systems; and one or more adaptive incentive-field systems; wherein settlement routing dynamically adjusts according to one or more attributed externalities.

A computer-implemented method comprising: measuring one or more externalities of an activity; attributing the one or more externalities through one or more contribution-lineage structures; adjusting settlement participation according to governance weighting and the attributed externalities; and propagating value attribution across one or more derivative ecosystems.

Family Z — Contribution-Graph Venture Formation and QPIIN Systems

Field Summary

This Family relates to computer-implemented venture coordination, and more particularly to systems and methods for contribution-derived ownership formation, artificial-intelligence-mediated venture coordination, Quantum Privacy Investment and Innovation Networks (QPIIN), and recursively composable startup ecosystems, in which venture structures and ownership allocations are generated according to verified contribution lineage.

Problem Addressed

Conventional venture formation relies on centralized intermediaries, on manually negotiated ownership structures, on institutionally fragmented capital markets, and on attribution systems disconnected from the venture-formation process. Contributors frequently generate substantial value before a venture is formally constituted — through research, through early development, through community formation, through the contribution of resources — yet receive little or no persistent participation in the downstream economic outcomes of the venture, because the contribution is not verifiably recorded and connected to the ownership structure. Conventional systems do not generate venture structures according to verified contribution lineage, governance compatibility, settlement participation, and recursive derivative-ecosystem analysis. The unresolved technical problem is the absence of a mechanism by which venture structures and ownership allocations are generated from a verifiable record of who contributed what.

Solution Overview

The disclosed architecture provides contribution-graph venture formation and Quantum Privacy Investment and Innovation Network (QPIIN) coordination. Contribution Graphs, as disclosed in the incorporated documents and Family E of the May 2026 QPN Catalyst Provisional, represent participant contributions, resource contributions, governance participation, infrastructure participation, liquidity participation, scientific contribution, AI-coordination contribution, and derivative-ecosystem participation. Artificial intelligence systems — governed as in Families T and V — analyze the Contribution Graphs to identify venture opportunities, generate startup proposals,

coordinate participant matching, allocate ownership participation, optimize governance compatibility, structure investment participation, and recursively generate derivative venture ecosystems. Ownership structures are generated according to verified contribution lineage, governance weighting, settlement participation, ecosystem impact, liquidity participation, and recursively propagated derivative-contribution analysis. The architecture thereby connects the venture's ownership structure to a verifiable record of contribution, so that the contributors who generated value before formal venture formation receive persistent, attributed participation in the venture's downstream outcomes.

Components

- **Contribution Graph Venture Analyzer.** A component that analyzes Contribution Graphs to identify venture opportunities.
- **AI Venture Coordination System.** A component by which a governed AI system coordinates venture formation, consistent with Family V.
- **Contribution-Derived Ownership Allocator.** A component that generates ownership-allocation structures according to verified contribution lineage and governance weighting.
- **Governance-Aware Investment Coordinator.** A component that structures and coordinates investment participation with awareness of governance compatibility.
- **QPIIN Liquidity and Matching Engine.** A component that coordinates investment participation, liquidity routing, and participant matching within a Quantum Privacy Investment and Innovation Network.
- **Recursive Venture Ecosystem Registrar.** A component that registers a generated venture as a governed ecosystem participant capable of recursively generating downstream ventures.
- **Derivative Venture Settlement Router.** A component that routes settlement among the participants of a venture and its derivative ventures.

Process Flow

In various embodiments, the system generates or accesses Contribution Graphs for ecosystem participants. A governed AI system analyzes contribution lineage, settlement flows, governance compatibility, and opportunity spaces. The AI system proposes venture, investment, accelerator, or partnership structures. Ownership participation is generated according to contribution-derived weighting. Investment participation and liquidity routing are coordinated through QPIIN systems. The derivative venture ecosystem is registered and may recursively generate downstream ventures.

Exception handling: a proposed venture structure that fails governance-compatibility evaluation is not registered; an ownership allocation that cannot be supported by verified contribution lineage is not generated; an AI coordination action exceeding the AI system's governed operational authority does not execute.

Alternative Embodiments

In a scientific embodiment, ventures are formed from scientific contribution lineage. In an enterprise-spinout embodiment, the architecture coordinates the formation of spinouts from an enterprise. In a sovereign embodiment, sovereign innovation networks are coordinated, consistent with Family W. In an AI-generated-startup-ecosystem embodiment, the ventures are generated by AI coordination systems. In a public-benefit embodiment, the ventures are public-benefit structures. In a contributor-owned embodiment, ownership is allocated predominantly to the

verified contributors. In a recursive accelerator-to-venture embodiment, recursive accelerators of Family X generate ventures. These embodiments are illustrative and may be combined.

Detailed Component Architecture and Contribution-Lineage Mechanics

The following provides additional component-level detail for the Family Z system. The detail is directed to the computational mechanics by which venture structures and ownership allocations are derived from contribution-lineage data structures, and is provided to make concrete that the disclosed subject matter is the computational processing of governed contribution-graph data structures rather than an abstract scheme of venture organization:

- **Contribution Graph — data structure.** In various embodiments, the Contribution Graph processed by Family Z is the Contribution Graph of the incorporated disclosures, comprising contributor nodes, resource nodes, settlement nodes, governance nodes, derivative-right nodes, and public-benefit nodes, with edges representing contribution relationships each carrying a contribution-type descriptor, a contribution timestamp, an edge weight, and a reference to the Trust-Block-bound record evidencing the contribution. Family Z operates upon this graph as a data structure: the venture-formation computations are graph computations over contributor nodes, contribution edges, and their weights.
- **Contribution Graph Venture Analyzer — opportunity computation.** In various embodiments, the analyzer computes venture opportunities by analyzing the Contribution Graph for clusters of contribution activity that satisfy configured opportunity criteria — for example a connected subgraph of contributors and resources whose combined contribution lineage indicates a coherent body of work not yet organized as a venture. The analyzer computes, for each candidate opportunity, the contributor set, the resource set, the governance conditions applicable to the implicated resources, and a contribution-weight distribution over the contributor set.
- **Contribution-Derived Ownership Allocator — allocation computation.** In various embodiments, the allocator computes an ownership-allocation structure for a venture as a deterministic function of the contribution-weight distribution computed by the analyzer, the governance weighting applicable to the contributions, the settlement participation associated with the contributed resources, and the liquidity participation of the contributors. The allocation is computed by a graph traversal that, for each contributor, aggregates the weights of that contributor's contribution edges, adjusts the aggregate by the applicable governance weighting, and normalizes across the contributor set to produce an ownership share. Because the allocation is a deterministic function of recorded contribution-lineage data, it is reproducible and auditable, and an ownership share can be traced to the specific contributions that produced it. In a dynamically-evolving embodiment, the allocation is recomputed as contribution continues, so that ownership reflects ongoing contribution rather than only contribution at the moment of venture formation.
- **QPIIN Liquidity and Matching Engine — matching computation.** In various embodiments, the engine coordinates a Quantum Privacy Investment and Innovation Network by computing matches among venture opportunities, investment participants, and liquidity, subject to governance compatibility evaluated through Trust Criteria. The engine computes, for a venture opportunity, the set of investment participants whose governance conditions are compatible with the venture's governance conditions, and routes investment participation and liquidity accordingly. The matching is a computation over governed records and is itself recorded as a Trust-Block-bound governed artifact.

State-Transition Model

In various embodiments, a venture progresses through a state machine. A venture begins in an OPPORTUNITY-IDENTIFIED state when the analyzer has computed it from the Contribution Graph. It transitions to a PROPOSED state when a venture-coordination system, which may be a governed AI system as described in Family V, has generated a venture proposal. It transitions to a GOVERNANCE-EVALUATED state upon evaluation of governance compatibility through Trust Criteria; a proposal that fails evaluation transitions to a REJECTED state. From GOVERNANCE-EVALUATED it transitions to an OWNERSHIP-ALLOCATED state when the allocator has computed the ownership-allocation structure, to an INVESTMENT-COORDINATED state when the QPIIN engine has coordinated investment participation and liquidity, and to a REGISTERED state when the venture has been registered as a governed ecosystem participant by the Recursive Venture Ecosystem Registrar. A registered venture may itself contribute to the Contribution Graph and may be analyzed for the formation of further ventures, so that the state machine is recursively re-entrant. Each transition is recorded as a Trust-Block-bound event.

Additional Alternative Embodiments

In a centralized embodiment, a single operator operates the QPIIN engine and the venture analyzer. In a decentralized embodiment, the Contribution Graph and the venture-formation computations are maintained as distributed data structures, so that no single party controls the recording of contribution or the computation of ownership. In a federated embodiment, a plurality of QPIIN operators federate, and a venture whose contributors span multiple federated domains carries an ownership-allocation structure spanning those domains. In a scientific-venture embodiment, the Contribution Graph records scientific contribution — datasets, methods, results, replications — and ventures are formed from scientific contribution lineage with ownership allocated to the scientific contributors. In an enterprise-spinout embodiment, the Contribution Graph records contributions internal to an enterprise and the architecture coordinates the formation of spinout ventures with ownership reflecting internal contribution. In a contributor-owned embodiment, the allocation function is parameterized so that ownership is allocated predominantly to the verified contributors rather than to capital participants. In an AI-mediated embodiment, the venture analysis and coordination are performed by a governed AI system as described in Family V, operating within the operational-authority governance of Family T. In a graph-native embodiment, the entire Family Z system is expressed as computations over the Contribution Graph. These embodiments are illustrative and may be combined.

Cross-Family Integration

Upstream dependencies. Family Z operates within the QPN-enabled infrastructure of the incorporated disclosures, depends on the Contribution Graph of Family E and the Liquidity Architecture of Family N of the May 2026 QPN Catalyst Provisional, and depends on Families T and V for governed AI coordination.

Downstream consumers. Family Z's generated ventures are governed ecosystem participants that may participate in the recursive accelerator ecosystems of Family X and the externality internalization of Family Y, and that may themselves form governed derivatives under Family Q.

Lateral interactions. Family Z's QPIIN Liquidity and Matching Engine interoperates with the Liquidity Architecture of Family N; its Derivative Venture Settlement Router interoperates with the Settlement Controller of Family G.

Emergent properties. Generating ownership from verified contribution lineage produces an emergent property: the contributors who generate value before formal venture formation receive

persistent, attributed, and verifiable participation in downstream outcomes — closing the gap, structural in conventional venture formation, between contribution and ownership.

Initial Claims

A computer-implemented contribution-graph venture formation system comprising one or more processors and memory storing instructions that, when executed, provide: one or more Contribution Graphs; one or more artificial intelligence venture-coordination systems; one or more governance-aware investment systems; and one or more contribution-derived ownership-allocation systems; wherein one or more venture structures are generated according to one or more contribution-lineage analyses.

A computer-implemented method comprising: generating one or more Contribution Graphs; analyzing contribution lineage represented in the one or more Contribution Graphs; generating one or more ownership-allocation structures according to the contribution lineage; coordinating investment participation associated with the one or more ownership-allocation structures; and registering one or more derivative venture ecosystems.

Omnibus Platform-Level Specification Material

Integrated Recursive Coordination Substrate

In various embodiments, the artificial intelligence systems, governance systems, liquidity systems, settlement systems, contribution systems, derivative-right systems, sovereign ecosystems, accelerator ecosystems, venture ecosystems, and recursively composable coordination systems disclosed across Families Q through Z, and across the incorporated Families A through P, operate together as components of a single integrated, cryptographically governed coordination substrate. The Families are not disconnected additions; each is a facet of one architecture, and the substrate is the architecture considered as a whole.

The integrated substrate permits recursively composable coordination across heterogeneous computational, sovereign, organizational, and artificial-intelligence ecosystems, through governed settlement participation, recursive derivative propagation, cryptographic governance continuity, AI-mediated coordination, heterogeneous interoperability, public-benefit routing, and contribution-lineage-based attribution. The same primitives — governed resources, Trust Blocks, Trust Criteria, Privacy Algorithms, Quantum Privacy Domains, Constitutional Guardrails, derivative inheritance, and recursive settlement — operate at every scale, from a single governed derivative model to a civilization-scale federation of sovereign and private ecosystems. This scale-invariance of the primitives is itself a disclosed property of the architecture.

Six Core Primitives

In various embodiments, the architecture may be understood as comprising six recursively interacting primitives. The enumeration is an aid to understanding and is not a limitation; the primitives are realized by the components and processes disclosed throughout this application and the incorporated disclosures.

The first primitive is **governed existence**: resources and AI systems operate through governance-authorized computational boundaries, such that operation occurs within, and only within, a governed environment. The second primitive is **recursive derivation**: resources generate derivative resources — including models, synthetic datasets, ventures, accelerators, governance systems, settlement systems, and coordination structures — and the derivatives may themselves derive further resources without bound. The third primitive is **governance continuity**: Trust Blocks,

Constitutional Guardrails, derivative obligations, settlement structures, and operational constraints propagate across derivative lineages, so that governance imposed once is inherited by all descendants. The fourth primitive is **recursive settlement participation**: upstream contributors participate in the value created downstream of them, across an unbounded number of derivative generations. The fifth primitive is **AI-native coordination**: artificial intelligence systems operate as first-class governed participants in the substrate, not merely as tools used upon it. The sixth primitive is **heterogeneous interoperability**: the same primitives function across sovereign, enterprise, cloud, AI, ledger, and public-benefit infrastructures, so that the substrate is not bound to any single infrastructure.

Recursive Coordination Loop

In various embodiments, the substrate operates as a recursive loop. Participants — which may be individuals, enterprises, sovereigns, AI systems, or other ecosystems — contribute data, models, governance structures, infrastructure, capital, computational resources, or public-benefit resources. The substrate converts these inputs into governed resources bound to Trust Blocks within Quantum Privacy Domains. Governed resources generate derivative resources. Derivatives inherit governance continuity and settlement obligations. Recursive settlement participation creates economic incentives for further contribution. Governed AI systems optimize the coordination, the liquidity, the governance, and the derivative formation. The ecosystem expands across sectors, sovereigns, enterprises, and AI systems, and the loop repeats. Each iteration of the loop may operate at a larger scale and across more heterogeneous participants than the last, which is the sense in which the architecture supports recursively composable, and ultimately civilization-scale, coordination.

Technical Effect

The disclosed architecture produces concrete technical effects in and improvements to distributed computing systems. It reduces centralized trust requirements, because coordination is effected through cryptographic governance structures rather than through a trusted central party. It improves recursive interoperability, because the same primitives compose across heterogeneous infrastructures and across derivative generations. It improves distributed authorization coordination, because authorization is evaluated against Trust Criteria at governed boundaries rather than negotiated bilaterally. It improves derivative-resource traceability, because every derivative carries verifiable lineage. It improves governance-aware execution, because execution occurs within governed environments whose action spaces are constrained. It improves artificial-intelligence operational containment, because AI operational existence is resource-bound and AI executable action-space topology is cryptographically constrained prior to execution. It improves partitioned collaborative computation, because mutually distrustful parties may collaborate without exposing protected capability surfaces. And it improves recursive settlement synchronization, because settlement participation propagates deterministically across derivative lineages. These are improvements to the operation of computing systems and computer networks, not abstract or organizational effects.

Formal Glossary and Definitions

The following definitions apply throughout this application. Each definition is intentionally broad; the recitation of examples within a definition is illustrative and non-limiting. Where a term is also defined in an incorporated document, the present definition supplements rather than narrows that definition.

Quantum Privacy Domain (QPD). A governed computational environment within which resources, computations, interoperability relationships, governance structures, settlement structures, authorization structures, or derivative-resource relationships are bounded according to one or more Privacy Algorithms, Trust Criteria, Constitutional Guardrails, Trust Blocks, or recursively composable governance structures. In various embodiments a Quantum Privacy Domain is the cryptographic and computational instantiation paired with a legal embodiment, and the cryptographic boundary of the domain operates as a governance boundary such that computation within the domain is governed by construction.

Privacy Algorithm. A computational process configured to govern, constrain, authorize, partition, transform, coordinate, evaluate, route, derive, compose, protect, obfuscate, isolate, interoperate with, or recursively propagate governed computational activities. In various embodiments a Privacy Algorithm is one-way, such that data flows into a Quantum Privacy Domain protected by the Privacy Algorithm, or by a common family of Privacy Algorithms with equivalent or better protection guarantees, and meaningful information cannot be revealed outside the domain boundary.

Trust Block. A governance-associated metadata structure associated with one or more resources, derivative resources, computational activities, interoperability relationships, AI systems, settlement structures, or recursively composable governance ecosystems. In various embodiments a Trust Block of a derivative resource incorporates links to the Trust Criteria of every upstream resource from which the derivative was produced, such that upstream governance is automatically inherited.

Governance Continuity. The recursive propagation of governance-associated conditions across derivative-resource lineages, interoperability relationships, computational ecosystems, AI systems, settlement systems, or recursively composable derivative ecosystems.

Derivative Resource. A resource generated partially or wholly from one or more upstream resources, including without limitation derivative datasets, derivative AI systems, distilled models, synthetic datasets, derivative governance systems, derivative ventures, derivative accelerators, derivative settlement systems, derivative orchestration systems, and derivative ecosystems.

Settlement Participation. Dynamically allocated economic, governance, liquidity, attribution, derivative-right, public-benefit, interoperability, or recursively composable participation structures associated with resources or derivative-resource ecosystems.

Contribution Lineage. Recursively propagating attribution relationships associated with resources, derivative resources, settlement structures, governance structures, computational activities, AI systems, infrastructure, liquidity contributions, or derivative ecosystems.

Constitutional Guardrail. A governance-associated constraint configured to govern computational activities, interoperability relationships, derivative-resource relationships, AI systems, settlement structures, or recursively composable ecosystem behaviors. The term "constitutional" denotes that the constraint is foundational to and inseparable from the governed execution environment rather than an external policy applied to it.

Operational Authority. Dynamically governed permissions, capabilities, computational authorities, execution rights, interoperability rights, communication permissions, settlement permissions, orchestration permissions, or derivative-generation permissions associated with governed computational entities.

Resource-Bound Operational Existence. A condition in which continued operation of a governed computational entity, including an artificial intelligence system, depends upon governed access to

authorized computational, communication, settlement, orchestration, infrastructure, actuator, interoperability, or governance resources, such that withdrawal of such access causes partial or total cessation of operation.

Executable Action-Space Topology. The computationally available, operationally available, governance-authorized, cryptographically available, or recursively composable action pathways associated with a governed computational entity. In various embodiments the executable action-space topology is constrained prior to execution such that an unauthorized action pathway is not present within it.

Distillation Governance. Governance-associated processes configured to govern the generation, extraction, derivation, fine-tuning, orchestration, synthetic-data formation, output generation, or derivative formation associated with artificial intelligence systems.

Partitioned Collaborative Computation. Collaboratively coordinated computational activity distributed across multiple independently governed trust domains while preserving isolation of protected capability surfaces.

Recursive Ecosystem. An ecosystem capable of recursively generating, governing, coordinating, interoperating with, deriving from, authorizing, settling with, or composing additional derivative ecosystems.

Civilization-Scale Coordination. Recursively composable coordination spanning heterogeneous organizational, sovereign, institutional, governance, settlement, interoperability, public-benefit, artificial-intelligence, or derivative ecosystems.

Operational Flows and State-Machine Embodiments

The following operational flows are illustrative embodiments. Operations may be reordered, combined, omitted, or performed concurrently except where a later operation depends on the result of an earlier one.

Governed Derivative AI Formation

An upstream AI resource exists within one or more governed Quantum Privacy Domains and possesses Trust Blocks, settlement-participation structures, derivative-authorization policies, and operational constraints. A participant submits a derivative-generation request specifying a target model, a derivative objective, training resources, governance conditions, settlement structures, and derivative-right structures. Governance systems evaluate Trust Criteria, sponsor authorization, Constitutional Guardrail compliance, jurisdictional compatibility, derivative restrictions, and inheritance requirements. Proof-of-Trust systems verify participant authorization, governance lineage, resource provenance, and execution-environment integrity. Governed derivative execution environments are instantiated and inherit Trust Blocks, Constitutional Guardrails, derivative-right obligations, settlement-participation structures, and operational constraints. Derivative AI resources are generated through distillation, fine-tuning, orchestration, synthetic-data augmentation, or recursive derivative formation. Lineage and settlement participation are registered recursively.

Distillation-Resistant Inference Routing

A governed AI model exists within one or more Quantum Privacy Domains and possesses output-governance policies, derivative restrictions, sampling-governance policies, and Trust Block

inheritance rules. A requesting participant submits an inference request together with identity metadata, governance credentials, and a requested output scope. The system evaluates Trust Criteria, governance compatibility, sponsor permissions, output restrictions, and settlement eligibility. Distillation-risk systems evaluate query frequency, semantic diversity, entropy accumulation, output similarity, reconstruction probability, and distillation-threshold metrics. The system dynamically adjusts output detail, volume, granularity, diversity, or query authorization. Generated outputs inherit Trust Blocks, derivative restrictions, attribution lineage, settlement obligations, and downstream constraints. Output lineage is registered in governance graphs, settlement systems, and derivative-tracking systems.

Constitutional AI Execution

An AI agent exists within governed execution environments and possesses operational authority, communication permissions, settlement permissions, and Trust-Block-governed authority structures. The AI agent generates an inference, orchestration, communication, actuator, or resource-access request. Governance systems evaluate Constitutional Guardrails, Trust Criteria, authority scopes, governance compatibility, risk scoring, and sponsor restrictions. The system resolves permitted, prohibited, throttled, sandboxed, or partitioned actions, and constrains the executable action-space topology accordingly. Authorized actions execute only within governed communication pathways, actuator systems, settlement systems, and resource environments. Governance systems continuously monitor anomalous behavior, settlement compliance, and operational-risk scoring. Operational authority is modified or revoked upon governance-trigger events, including by withdrawal of resource access.

Partitioned Collaborative AI Computation

Multiple mutually distrustful participants possess protected models, datasets, sovereign infrastructure, or computational capabilities. The participants request collaborative training, evaluation, safety analysis, interpretability analysis, or federated inference. Partitioned Privacy Algorithms distribute the computation across multiple trust partitions, cloud providers, sovereign infrastructures, secure enclaves, or hardware environments. Capability isolation prevents unrestricted visibility into weights, datasets, intermediate states, or capability surfaces. Distributed verification validates execution integrity, governance compatibility, settlement compliance, and derivative restrictions. Permitted outputs are aggregated according to governance restrictions and Trust Criteria.

Contribution-Graph Venture Coordination

Contribution Graphs represent contributors, resources, settlement flows, accelerator structures, governance conditions, and derivative ecosystems. Governed AI systems analyze contributor relationships, governance compatibility, settlement patterns, resource-reuse opportunities, and derivative opportunities. The AI systems generate startup proposals, partnership structures, accelerator proposals, and recursively composable coordination opportunities. Governance compatibility is resolved through Trust Criteria. Contribution-derived ownership structures are generated according to lineage, governance weighting, liquidity participation, and derivative participation. QPIIN systems coordinate investment participation, liquidity routing, settlement routing, governance participation, and accelerator participation. The generated venture becomes a recursively composable ecosystem participant.

Governance-State Machine

In various embodiments, a governed resource or governed AI system maintains a governance state drawn from the following illustrative set of states, and transitions among the states are governed by Trust Criteria and governance events.

In the **UNREGISTERED** state, the resource exists outside the governed ecosystem. In the **INGESTION PENDING** state, the resource undergoes Trust evaluation, provenance analysis, and governance-compatibility review. In the **GOVERNED ACTIVE** state, the resource possesses active Trust Blocks, governance continuity, settlement participation, and operational authority. In the **THROTTLED** state, the resource remains active but its operational authority is constrained. In the **SANDBOXED** state, the resource operates only within restricted execution environments. In the **PARTITIONED** state, the resource's operational authority is distributed across multiple trust domains. In the **DERIVATIVE ACTIVE** state, a derivative resource operates while inheriting governance continuity, settlement participation, and recursive obligations. In the **SUSPENDED** state, operational authority is temporarily removed. In the **REVOKED** state, governance authorization is removed. In the **RETIRED / ARCHIVED** state, the resource is preserved for lineage continuity and auditability. Transitions among these states — for example from GOVERNED ACTIVE to THROTTLED upon a rising risk score, or from any active state to REVOKED upon a governance event crossing a revocation threshold — are themselves governed and recorded, such that the governance history of a resource is auditable.

Graph-Theoretic and Computational Topology Embodiments

In various embodiments, the disclosed coordination is represented and computed through graph data structures. The following graph embodiments are illustrative.

Governance graphs. A governance graph may include nodes representing Quantum Privacy Domains, AI systems, derivative resources, settlement systems, governance systems, interoperability systems, sovereign ecosystems, enterprises, accelerators, ventures, public-benefit systems, or derivative ecosystems. Edges may represent authorization, governance inheritance, settlement propagation, derivative-resource relationships, interoperability, orchestration, attribution, liquidity, or recursive coordination. Governance evaluation and propagation are computed by traversal of the governance graph.

Contribution graphs. A contribution graph may represent recursively propagating attribution relationships among contributor nodes, derivative-resource nodes, settlement nodes, governance nodes, derivative-right nodes, public-benefit nodes, and recursively composable attribution structures. Contribution propagation may be dynamically weighted according to governance weighting, contribution weighting, interoperability weighting, derivative-resource weighting, settlement weighting, and public-benefit weighting.

Settlement graphs. A settlement graph may include settlement nodes, routing nodes, governance nodes, contribution nodes, derivative-resource nodes, liquidity nodes, interoperability nodes, and recursively composable settlement structures. Settlement propagation may be computed according to weighting functions, governance functions, recursive propagation functions, derivative inheritance functions, attribution functions, interoperability functions, and public-benefit allocation functions.

Executable action-space graphs. The executable action-space topology associated with an AI system may be represented as an executable-action graph including executable nodes, prohibited nodes, constrained nodes, throttled nodes, sandboxed nodes, partitioned nodes, and recursively composable governance-constrained execution structures. Governance systems may dynamically modify node accessibility, edge accessibility, traversal permissions, execution probability, interoperability permissions, and resource accessibility within the executable-action graph, and such modification is the means by which executable action-space topology is constrained prior to execution as described in Family T.

Distillation-risk topologies. Distillation-risk analysis may include semantic clustering, entropy-accumulation analysis, query-diversity analysis, output-similarity analysis, reconstruction-likelihood analysis, synthetic-data-extraction analysis, and recursively composable derivative-generation analysis, computed over a topology of the outputs of an AI system considered as a set.

Federated trust graphs. A partitioned collaborative ecosystem may be represented by a federated trust graph distributing computational execution, authorization, governance evaluation, derivative generation, settlement coordination, and recursive ecosystem coordination across independently governed trust partitions.

Illustrative recursive settlement relationship. Recursive settlement propagation may be represented conceptually by the relationship $S(d_n) = \Sigma [w_i * S(r_i)]$, in which $S(d_n)$ denotes settlement participation associated with a derivative resource d_n, each r_i denotes one of one or more upstream resources, and each w_i denotes a governance-weighted propagation relationship. The relationship is illustrative only and may be implemented through graph traversal, recursive propagation, distributed settlement systems, AI coordination systems, or recursively composable settlement architectures. The relationship is recursive in that each $S(r_i)$ may itself be the settlement participation of a resource that is a derivative of further upstream resources.

Illustrative Data Structures and Pseudocode

The following data structures and pseudocode are illustrative embodiments provided for enablement. Field names, identifier formats, numeric values, thresholds, and control flow are examples and are not limitations.

Example Trust Block Structure

An illustrative Trust Block associated with a governed AI model resource may be represented as a structured object having a trust-block identifier; a resource identifier; a resource-type descriptor, for example "AI_MODEL"; a governance-domain identifier identifying the Quantum Privacy Domain that governs the resource; an authorization policy specifying a set of permitted actions, for example inference, fine-tuning, and governed distillation, and a set of restricted actions, for example unbounded export and unrestricted sampling; a set of identifiers of applicable Constitutional Guardrails; a derivative-obligations structure specifying, for example, that governance and settlement are to be inherited by derivatives; and a settlement-structure specifying, for example, an upstream-participation share and a public-benefit-allocation share. An illustrative instance is the object whose trust-block identifier is "TB-884293", whose resource identifier is "RES-99117", whose resource type is "AI_MODEL", whose governance domain is "QPD-ALPHA", whose permitted actions are inference, fine-tuning, and governed distillation, whose restricted actions are unbounded export and unrestricted sampling, whose Constitutional Guardrails are identified as "CG-SAFETY-001" and "CG-DERIVATIVE-004", whose derivative-obligations structure indicates that governance inheritance and settlement inheritance are both

enabled, and whose settlement structure specifies an upstream-participation share of 0.18 and a public-benefit-allocation share of 0.03. The numeric shares and identifiers are illustrative.

Example Governance-State Object

An illustrative governance-state object associated with a governed AI system may be represented as a structured object having an entity identifier; an operational-state value drawn from the Governance-State Machine, for example "THROTTLED"; an authority-level value, for example "PARTIAL"; a resource-permissions structure indicating, for each resource class, whether access is currently permitted, for example permitting compute access while denying network access and denying derivative-generation; a set of governance conditions currently in effect, for example "HIGH_DISTILLATION_RISK" and "CROSS_DOMAIN_RESTRICTION"; and a revocation policy specifying, for example, an automatic-revocation threshold. An illustrative instance is the object whose entity identifier is "AI-7721", whose operational state is "THROTTLED", whose authority level is "PARTIAL", whose resource permissions permit compute but deny network access and derivative generation, whose governance conditions are "HIGH_DISTILLATION_RISK" and "CROSS_DOMAIN_RESTRICTION", and whose revocation policy specifies an automatic-revocation threshold of 0.91. The values are illustrative.

Governance Evaluation Pseudocode

An illustrative procedure for evaluating operational authority receives an entity and a request. The procedure retrieves the governance conditions associated with the entity. It evaluates the Trust Criteria for the entity, the request, and the governance conditions, producing a trust evaluation. It evaluates the derivative obligations applicable to the entity and the request, producing derivative constraints. It evaluates the interoperability restrictions applicable to the entity and the request, producing interoperability constraints. It aggregates the trust evaluation, the derivative constraints, and the interoperability constraints into a governance risk score. If the risk score exceeds a revocation threshold, the procedure revokes the operational authority of the entity. Otherwise, if the risk score exceeds a throttle threshold, the procedure throttles the operational authority of the entity. Otherwise, the procedure authorizes execution. In all cases, the procedure propagates governance continuity from the entity to the entity's derivatives. The thresholds and the aggregation function are illustrative and may be configured.

Distillation-Risk Pseudocode

An illustrative procedure for evaluating distillation risk receives a session. The procedure calculates the semantic diversity of the session, the entropy accumulation of the session, and an estimated reconstruction likelihood for the session. It aggregates these into a risk value. If the risk value exceeds a distillation threshold, the procedure throttles the outputs of the session, reduces the output granularity of the session, and requires additional authorization for the session. In all cases, the procedure propagates derivative-governance obligations associated with the session. The metrics, the aggregation, and the threshold are illustrative.

Recursive Settlement Pseudocode

An illustrative procedure for propagating settlement receives a resource. The procedure retrieves the upstream resources of the resource. It calculates contribution weights for the upstream resources. For each upstream resource, the procedure determines an allocation from the contribution weights and routes settlement to the upstream resource according to the allocation. The procedure then recursively propagates settlement for the upstream resources, such that settlement participation propagates across an unbounded number of derivative generations. The weighting function is illustrative.

Independent and Dependent Claim Set

The following claims are provided as provisional source material. Claim numbering, dependency, statutory category, and language may be revised in any subsequent application without disclaimer of disclosed subject matter.

Independent Claims

1. A computer-implemented coordination system comprising one or more processors and memory storing instructions that, when executed, provide: one or more governed computational environments; one or more governance inheritance systems; one or more derivative-resource generation systems; one or more settlement coordination systems; one or more attribution-lineage systems; and one or more interoperability-governance systems; wherein derivative resources are recursively generated from one or more upstream resources; governance continuity propagates across derivative-resource lineages; and settlement participation recursively propagates across derivative-resource lineages.
2. A computer-implemented method comprising governing one or more computational environments through one or more governance structures; generating one or more derivative resources from one or more upstream resources; propagating governance continuity across the derivative resources; dynamically coordinating settlement participation associated with the derivative resources; and recursively generating one or more derivative ecosystems.
3. A computer-implemented artificial intelligence governance system comprising one or more processors and memory storing instructions that, when executed, provide: one or more artificial intelligence systems; one or more governed execution environments; one or more governance evaluation systems; one or more operational-authority systems; and one or more governance inheritance systems; wherein operational authority associated with the artificial intelligence systems is dynamically governed through one or more governance structures; and continued operation of the artificial intelligence systems depends upon governed access to one or more authorized computational resources.
4. A computer-implemented method comprising receiving one or more artificial intelligence execution requests; evaluating governance conditions associated with the execution requests; dynamically resolving operational authority associated with one or more artificial intelligence systems according to the governance conditions; governing execution of one or more artificial intelligence actions according to the resolved operational authority; and dynamically modifying operational authority according to one or more governance events.
5. A computer-implemented derivative artificial intelligence governance system comprising one or more processors and memory storing instructions that, when executed, provide: one or more governed artificial intelligence systems; one or more derivative-generation systems; one or more distillation-governance systems; one or more output-governance systems; and one or more derivative-lineage systems; wherein one or more derivative artificial intelligence systems are generated from one or more upstream artificial intelligence systems and governance continuity propagates across derivative artificial intelligence lineages.
6. A computer-implemented method comprising receiving one or more inference requests associated with one or more governed artificial intelligence systems; evaluating one or more distillation-risk conditions; dynamically governing one or more outputs generated by the governed artificial intelligence systems according to the distillation-risk conditions; generating one or more derivative artificial intelligence systems; and propagating governance continuity across one or more derivative artificial intelligence lineages.

7. A computer-implemented collaborative coordination system comprising one or more processors and memory storing instructions that, when executed, provide: a plurality of independently governed computational environments; one or more partitioned computation systems; one or more governance coordination systems; and one or more capability-isolation systems; wherein collaborative computation is distributed across the independently governed computational environments and one or more protected computational capability surfaces remain isolated during collaborative computation.
8. A computer-implemented recursive settlement coordination system comprising one or more processors and memory storing instructions that, when executed, provide: one or more settlement routing systems; one or more derivative-lineage systems; one or more attribution systems; and one or more governance inheritance systems; wherein settlement participation recursively propagates across one or more derivative-resource lineages.
9. A computer-implemented ecosystem coordination system comprising one or more processors and memory storing instructions that, when executed, provide: one or more contribution-lineage systems; one or more artificial intelligence coordination systems; one or more venture-generation systems; and one or more governance coordination systems; wherein one or more ventures are generated according to one or more contribution-lineage analyses.
10. A computer-implemented interoperability system comprising one or more processors and memory storing instructions that, when executed, provide: a plurality of independently governed sovereign computational ecosystems; one or more interoperability-governance systems; one or more governance inheritance systems; and one or more settlement coordination systems; wherein the sovereign computational ecosystems interoperate while preserving localized governance autonomy.
11. A computer-implemented recursive coordination substrate comprising one or more processors and memory storing instructions that, when executed, provide: one or more governance systems; one or more settlement coordination systems; one or more artificial intelligence coordination systems; one or more derivative-resource generation systems; and one or more interoperability-governance systems; wherein heterogeneous computational ecosystems recursively interoperate through propagating governance continuity and recursive settlement participation.
12. A computer-implemented artificial intelligence operational governance system comprising one or more processors and memory storing instructions that, when executed, provide: one or more artificial intelligence systems; one or more resource-governance systems; one or more operational-authority systems; and one or more execution-governance systems; wherein continued operation of the artificial intelligence systems depends upon governed access to one or more computational resources; and an executable action-space topology associated with the artificial intelligence systems is dynamically constrained, prior to execution, according to one or more governance conditions.
13. A computer-implemented system for partitioned collaborative artificial intelligence computation comprising one or more processors and memory storing instructions that, when executed, provide: a plurality of independently governed trust partitions; one or more partitioned Privacy Algorithms distributed across the trust partitions; one or more collaborative artificial intelligence computation systems; and one or more capability-isolation systems; wherein collaborative artificial intelligence computation occurs across mutually distrustful participants without exposing protected capability surfaces.
14. A computer-implemented system for externality-governed settlement coordination comprising one or more processors and memory storing instructions that, when executed, provide: one or

more externality-attribution systems; one or more governance-weighted settlement systems; one or more contribution-lineage systems; and one or more adaptive incentive-field systems; wherein settlement routing dynamically adjusts according to one or more attributed externalities.

15. A computer-implemented system for contribution-graph venture formation comprising one or more processors and memory storing instructions that, when executed, provide: one or more Contribution Graphs; one or more artificial intelligence venture-coordination systems; one or more governance-aware investment systems; and one or more contribution-derived ownership-allocation systems; wherein one or more venture structures are generated according to one or more contribution-lineage analyses.

Dependent Claims

16. The system of claim 1, wherein governance continuity comprises propagation of one or more Trust Blocks across derivative-resource lineages.
17. The system of claim 1, wherein governance continuity comprises propagation of one or more Constitutional Guardrails across derivative-resource lineages.
18. The system of claim 1, wherein governance continuity comprises recursive inheritance of one or more settlement obligations.
19. The system of claim 1, wherein governance continuity comprises recursive inheritance of one or more operational constraints.
20. The system of claim 1, wherein governance continuity propagates across one or more artificial intelligence derivative ecosystems.
21. The system of claim 5, wherein the distillation-governance systems evaluate one or more query-frequency conditions.
22. The system of claim 5, wherein the distillation-governance systems evaluate one or more entropy-accumulation conditions.
23. The system of claim 5, wherein the distillation-governance systems dynamically modify output granularity.
24. The system of claim 5, wherein the derivative artificial intelligence systems inherit one or more settlement-participation obligations.
25. The system of claim 5, wherein the derivative artificial intelligence systems inherit one or more interoperability restrictions.
26. The method of claim 6, further comprising recursively propagating derivative-right participation across one or more derivative artificial intelligence systems.
27. The system of claim 12, wherein operational authority associated with the artificial intelligence systems is dynamically throttled according to one or more governance conditions.
28. The system of claim 12, wherein the executable action-space topology associated with the artificial intelligence systems is cryptographically constrained prior to execution.
29. The system of claim 12, wherein one or more unauthorized actions are computationally inaccessible to the artificial intelligence systems prior to execution.
30. The method of claim 4, further comprising partitioning operational authority across multiple governance domains.
31. The method of claim 4, further comprising revoking operational authority according to one or more governance events, including by withdrawing access to one or more authorized resources.

32. The system of claim 7, wherein collaborative computation is distributed across multiple sovereign infrastructures.
33. The system of claim 7, wherein collaborative computation is distributed across multiple cloud providers.
34. The system of claim 7, wherein collaborative computation is distributed across multiple secure-enclave environments.
35. The system of claim 7, wherein the one or more protected computational capability surfaces comprise one or more artificial intelligence model weights.
36. The system of claim 13, wherein the collaborative artificial intelligence computation comprises artificial intelligence safety evaluation.
37. The system of claim 13, wherein the collaborative artificial intelligence computation comprises interpretability analysis.
38. The system of claim 8, wherein settlement participation recursively propagates across multiple derivative-resource generations.
39. The system of claim 8, wherein settlement participation is dynamically weighted according to one or more contribution-lineage analyses.
40. The system of claim 9, wherein the artificial intelligence coordination systems generate one or more accelerator structures.
41. The system of claim 9, wherein the artificial intelligence coordination systems generate one or more governance coordination structures.
42. The system of claim 15, wherein ownership allocation dynamically evolves according to ongoing ecosystem participation.
43. The system of claim 10, wherein the sovereign computational ecosystems comprise one or more independently governed artificial intelligence ecosystems.
44. The system of claim 10, wherein interoperability is governed through one or more partitioned Privacy Algorithms.
45. The system of claim 11, wherein the heterogeneous computational ecosystems comprise one or more public-benefit ecosystems.
46. The system of claim 11, wherein the heterogeneous computational ecosystems comprise one or more artificial intelligence ecosystems.
47. The method of claim 2, further comprising recursively generating one or more derivative governance ecosystems.
48. The system of claim 14, wherein the one or more attributed externalities comprise one or more environmental impacts.
49. The system of claim 14, wherein the one or more attributed externalities comprise one or more public-health outcomes.
50. The system of claim 14, wherein one or more artificial intelligence systems optimize externality pricing.
51. The system of claim 15, wherein the one or more Contribution Graphs comprise scientific contribution lineage.
52. The system of claim 15, wherein one or more derivative ventures inherit one or more settlement-participation obligations.
53. The system of claim 15, wherein one or more recursively composable venture ecosystems comprise one or more sovereign innovation ecosystems.

Supplemental Computational Embodiments

The following sections provide additional computational, graph-theoretic, and cryptographic implementation detail supplementing the Families and omnibus material disclosed above. The additions are additive and do not narrow, replace, or limit any disclosure set forth above; the anti-narrowing, recursive-generalization, heterogeneous-infrastructure, and technical-computational-framing provisions stated earlier apply throughout. Each mechanism described below is machine-executed, cryptographically mediated, graph-computed, or deterministically evaluable, and is disclosed as a computational coordination architecture rather than an organizational or contractual practice.

Graph-Theoretic Computational Infrastructure

In various embodiments, the governance, authorization, derivative-formation, settlement, and coordination operations disclosed throughout this application are implemented as computations over a family of interrelated graph data structures. This section formalizes those graph structures and the operations defined over them, and discloses graph-native embodiments in which authorization, settlement, execution governance, and derivative inheritance are expressed as graph operations rather than as ad hoc procedures.

Graph family and node taxonomy. In various embodiments, the system maintains one or more of: a lineage graph, recording derivation relationships among resources and derivative resources; an execution graph, recording the executable action pathways available to a governed computational entity; a settlement graph, recording the routing of settlement participation among contributors and derivatives; an authority graph, recording the operational-authority and capability relationships among governed entities and resources; a governance graph, recording the propagation of governance conditions among governed entities; an interoperability graph, recording permitted cross-domain and cross-partition relationships; a dependency graph, recording the resources upon which continued operation of an entity depends; and a derivative graph, recording the recursive derivative-formation relationships among resources. In various embodiments these graphs are distinct structures; in other embodiments two or more are realized as typed subgraphs of a single composite graph, and a node or edge may participate in multiple typed subgraphs simultaneously. Node types include, in an illustrative and non-limiting enumeration: resource nodes, derivative-resource nodes, artificial-intelligence-system nodes, governed-environment nodes, Quantum Privacy Domain nodes, participant nodes, sponsor nodes, infrastructure nodes, partition nodes, accelerator nodes, venture nodes, settlement nodes, routing nodes, authority nodes, capability nodes, action nodes, governance-condition nodes, Trust Block nodes, Constitutional Guardrail nodes, and public-benefit nodes.

Edge taxonomy and weighted propagation. In various embodiments, each directed edge of a graph carries an edge-type descriptor, an edge weight, a timestamp, and a reference to a Trust-Block-bound record evidencing the relationship the edge represents. Edge types include, illustratively and without limitation: derived-from, trained-on, evaluated-by, orchestrated-by, governed-by, authorizes, depends-on, settles-to, propagates-to, interoperates-with, delegates-to, constrains, and inherits-from. In various embodiments a propagation operation traverses edges of a specified type and computes, for each reached node, a propagated quantity as a weighted combination of the quantities at the edge's source nodes, the edge weights serving as the combination coefficients. Weighted propagation is used, in various embodiments, to compute recursive settlement participation, to compute the inheritance of governance conditions, to

compute reputation or contribution measures, and to compute the cumulative risk or authority state of a node as a function of its graph neighborhood.

Transitive closure and traversal authorization. In various embodiments, a transitive-closure operation computes, for a given node and a specified set of edge types, the set of all nodes reachable from the given node by repeated traversal of edges of those types; because the lineage and derivative graphs are acyclic by construction, the closure operation terminates. In various embodiments, graph traversal is itself a governed operation: a traversal-authorization function evaluates, for each edge a traversal would cross, whether the traversing entity holds authority — under the authority graph and the applicable Trust Criteria — to observe or act upon that edge, and the traversal is restricted to the authorized subgraph. Graph-native authorization is thereby realized: an entity's permitted view of, and permitted operations over, the graph family are themselves determined by graph computation.

Graph reconciliation, partition synchronization, and conflict resolution. In various embodiments, where a graph is maintained as a distributed or federated structure replicated across multiple partitions, nodes, or governance domains, a reconciliation operation merges the partition-local views into a consistent composite view. In various embodiments reconciliation proceeds by: identifying nodes and edges present in one partition-local view and absent from another; evaluating each such node or edge against the Trust Criteria governing the receiving partition; admitting those that satisfy the criteria; and recording, for those that conflict — for example, two partition-local views asserting inconsistent edge weights or inconsistent governance conditions for the same edge — a conflict record resolved by a conflict-resolution function. In various embodiments the conflict-resolution function resolves conflicts by a deterministic rule, including without limitation a most-recent-timestamp rule, a governance-precedence rule under which the view governed by the higher-precedence Trust Authority controls, or a quorum rule under which the view asserted by a threshold of partitions controls. A partition-synchronization operation propagates the reconciled composite view back to the partitions, such that the distributed graph converges to a consistent state.

Graph pruning and graph-state inheritance. In various embodiments, a graph-pruning operation removes from, or marks non-traversable within, a graph those nodes and edges that a current governance state renders unauthorized, inactive, revoked, or out of scope; the pruned graph is the operative graph over which subsequent computation proceeds. In various embodiments a derivative resource, upon its formation, inherits a graph-state from its upstream resources: the derivative's initial governance graph, authority graph, and settlement graph are computed as functions of the corresponding graphs of the upstream resources, so that graph-state inheritance is the mechanism by which governance continuity propagates to derivatives.

Recursive graph composition and graph entanglement. In various embodiments, a graph may be composed with another graph to form a composite graph whose nodes and edges are the union of the constituent graphs' nodes and edges together with cross-graph edges representing relationships among them; composition is recursive in that a composite graph may itself be composed with further graphs. In various embodiments two graphs are entangled where a node or edge is shared between them such that an operation modifying the shared element in one graph modifies it in the other; graph entanglement is the structure by which a resource existing simultaneously in multiple governance contexts, as described in the governance-superposition mechanics of the incorporated disclosures, is represented — the resource is a single node entangled across the governance graphs of each context in which it exists.

Topology-based settlement routing. In various embodiments, settlement is routed by computation over the settlement graph: a settlement event associated with a node is routed to other nodes along settles-to edges, the amount routed to each node being determined by weighted propagation, and the routing continuing recursively along settles-to edges from each reached node, so that settlement propagates through the topology of the settlement graph to the full set of upstream contributors. Topology-based routing is significant because it makes settlement allocation a deterministic function of recorded graph structure, reproducible and auditable by any party authorized to traverse the relevant subgraph.

Operational Flow — Graph-Native Authorization and Propagation

In various embodiments, a graph-native authorization and propagation operation proceeds as follows. A requesting entity submits a request implicating one or more graph nodes. The system computes the transitive-upstream closure of the implicated nodes over the governed-by, derived-from, and depends-on edge types. The system evaluates a traversal-authorization function over the closure, restricting the operative graph to the subgraph the requesting entity is authorized to observe and act upon. The system evaluates the request against the governance conditions reached in the authorized subgraph, the conditions having been propagated to the implicated nodes by weighted propagation over inherits-from and constrains edges. The system computes an authorization determination and, where the request is a settlement-bearing or derivative-forming operation, computes the resulting settlement routing or derivative graph-state inheritance by the corresponding graph operations. The determination and the resulting graph mutations are recorded as Trust-Block-bound events, and the graph is synchronized across partitions. Exception handling: a request implicating a node not reachable within the requesting entity's authorized subgraph is denied; a graph mutation that would introduce a cycle into an acyclic graph is rejected; a reconciliation conflict that the conflict-resolution function cannot resolve deterministically is recorded and escalated.

Capability-Token and Resource-Bound Execution Mechanics

The following expands the resource-bound operational existence and executable-action-space mechanics of Family T with detailed capability-token, delegation, and cryptographic-gating embodiments. The mechanics below are disclosed as computational and cryptographic enforcement architecture; the central disclosed property is that an unauthorized action is computationally non-instantiable — it cannot be brought into existence as an executable action by the governed entity — rather than merely detected or filtered after the entity attempts it.

Capability tokens. In various embodiments, the authority of a governed computational entity to effectuate a given action, traverse a given executable-action-graph node, or access a given resource is represented by a capability token. In various embodiments a capability token is a cryptographically signed data structure comprising: a token identifier; an identifier of the entity to which the token is issued; a scope descriptor delimiting the actions, nodes, or resources the token authorizes; an issuance timestamp; an expiration condition; an issuing-authority identifier; and a cryptographic signature binding the foregoing. In various embodiments the execution layer will not effectuate an action, will not materialize an executable-action-graph node as traversable, and will not grant a resource access, absent presentation of a valid, in-scope, unexpired capability token; an action for which no such token exists or can be obtained is, accordingly, not an action the entity is able to instantiate.

Ephemeral and renewable execution authority. In various embodiments, capability tokens are ephemeral: a token carries a short expiration condition and authorizes action only within a

bounded interval, after which it is no longer valid. In various embodiments execution authority is renewable: continued operation of the entity requires periodic re-issuance of capability tokens, the re-issuance being conditioned, at each renewal, on re-evaluation of the entity's governance state, so that an entity whose governance state has deteriorated is not renewed and its operation lapses as its outstanding tokens expire. Renewable ephemeral authority is significant because it makes continued operation an actively governed condition rather than a default: authority must be continually re-granted, and the withholding of renewal is sufficient to halt operation without any affirmative revocation step.

Scoped credentials, hierarchical delegation, and delegated sub-authority. In various embodiments, a capability token's scope descriptor narrowly delimits the authority conferred, and an entity may hold multiple scoped tokens that together define its aggregate authority. In various embodiments an entity holding a capability token may delegate a subset of the authority it confers to another entity by issuing a delegated capability token, the delegated token's scope being a subset of the delegating token's scope and the delegated token referencing the delegating token; delegation is hierarchical and recursive in that a delegated token may itself be further delegated, forming a delegation tree. In various embodiments partial authority inheritance is supported: a derivative resource or a spawned entity inherits a specified subset of the capability tokens of its parent.

Quorum-based authority issuance. In various embodiments, issuance of a capability token, or issuance of a token above a specified scope or risk threshold, requires authorization by a quorum of issuing authorities rather than by a single authority: the token is valid only if it carries signatures from at least a threshold number of authorities drawn from a designated set. Quorum-based issuance distributes the authority to confer authority, so that no single issuing authority can unilaterally grant high-scope execution capability.

Cryptographic action proofs and replay prevention. In various embodiments, effectuation of an action produces a cryptographic execution receipt: a signed record attesting that the action was effectuated, under which capability token, at which time, and with which result. In various embodiments each capability token is single-use or carries a monotonically increasing use counter, and the execution layer rejects presentation of a token whose use counter has not advanced or which has already been consumed; this prevents action-replay attacks, in which a previously valid authorization is re-presented to effectuate an action a second time without fresh authorization.

Revocation cascades and recursive authority dependency. In various embodiments, revocation of a capability token triggers a revocation cascade: the system traverses the delegation tree rooted at the revoked token and revokes every token delegated, directly or indirectly, from it, so that revocation of an authority revokes all sub-authorities derived from it. In various embodiments authority is recursively dependent: a token's validity is conditioned on the continued validity of the token from which it was delegated and, transitively, on the full chain of tokens up to a root authority, so that a break anywhere in the chain invalidates all tokens below it. Recursive authority dependency makes the authority graph a structure in which authority flows from roots through delegation chains and can be withdrawn at any point, with the withdrawal propagating deterministically to all dependent authority.

Dynamic materialization of action pathways. In various embodiments, the executable-action graph presented to a governed entity is materialized dynamically: a node representing an action is materialized as present and traversable only at the time the entity holds a valid in-scope capability token for it, and is otherwise not present in the graph the entity's execution proceeds over. An

unauthorized action is therefore not a node the entity sees and is prevented from traversing; it is a node that is not materialized, and the action is, in the disclosed sense, computationally non-instantiable.

State-Machine Embodiments — Capability Lifecycle

In various embodiments, a capability token progresses through a state machine. A token is created in an ISSUED state upon issuance by an authority or quorum of authorities. It transitions to an ACTIVE state when first presented and validated. From ACTIVE it may transition to RENEWED upon successful renewal, re-entering ACTIVE with a fresh expiration condition; to SUSPENDED upon a governance event, in which state the token is temporarily non-effectuable but not destroyed; to EXPIRED upon its expiration condition being met; or to REVOKED upon revocation. From SUSPENDED a token may return to ACTIVE upon clearance of the governance condition, or proceed to REVOKED. The REVOKED and EXPIRED states are terminal, and entry into either triggers evaluation of the revocation cascade over the token's delegation tree. Each transition is recorded as a Trust-Block-bound event. In various embodiments a separate authority-partitioning state machine governs the distribution of an entity's authority across governance domains, with states UNPARTITIONED, PARTITION-PROPOSED, PARTITIONED, and RECONCILED, the PARTITIONED state denoting that distinct domains hold distinct, individually-insufficient subsets of the entity's authority such that action requires reassembly of a quorum of subsets.

Multi-Laboratory and Multi-Hyperscaler Partitioned Coordination

The following provides additional concrete embodiments of the partitioned collaborative computation of Family U, directed to coordination across multiple frontier laboratories, multiple cloud and compute providers, sovereign compute partitions, enterprise model partitions, regulated-domain partitions, and confidential-computing partitions. These embodiments position the architecture as cross-organizational coordination infrastructure: infrastructure by which organizations that operate independent and mutually distrustful compute and model estates coordinate without consolidating those estates.

Provider and laboratory partitions. In various embodiments, each participating frontier laboratory, cloud provider, hardware vendor, sovereign compute estate, or enterprise model estate constitutes a trust partition within the federated trust graph of Family U. Each partition retains, within its own infrastructure and under its own local Trust Criteria, its protected capability surfaces — model weights, proprietary datasets, training infrastructure, intermediate states. A collaborative computation specified across the partitions is decomposed by the Partitioned Privacy Algorithm Executor into partition-local computations and cross-partition coordination operations, the protected surfaces never leaving their home partitions.

Cross-partition derivative formation. In various embodiments, a derivative resource is formed across partitions: a derivative model, synthetic dataset, or evaluation artifact is produced by a computation whose partition-local components execute within several partitions and whose cross-partition exchanges are limited to those the governing Trust Criteria permit. The resulting derivative carries, by graph-state inheritance, a lineage and settlement graph spanning every contributing partition, so that cross-organizational derivative formation produces a derivative attributed to, and settling to, contributors across organizational boundaries.

Partition synchronization mechanics. In various embodiments, the partitions maintain partition-local views of the shared graph family, and a partition-synchronization operation, as described in the graph-reconciliation mechanics above, reconciles the partition-local views into a converged composite view; synchronization proceeds on a schedule, on the occurrence of designated events,

or continuously, and each synchronization is itself a Trust-Block-bound governed operation. Cross-partition governance reconciliation resolves, by the conflict-resolution function, inconsistencies among the partition-local governance conditions, with sovereign and regulated-domain partitions assigned governance-precedence appropriate to their jurisdictional authority.

Federated evaluation and cross-domain orchestration. In various embodiments, a federated evaluation is conducted across the partitions: each partition evaluates a designated property of a designated artifact locally, and the Controlled Output Aggregator combines the partition-local evaluation outputs into a federated evaluation result that reveals the evaluated property without revealing the evaluated artifact's protected surface. In various embodiments cross-domain orchestration coordinates a multi-step workflow whose steps execute in different partitions, the orchestration being mediated by capability tokens scoped to each partition and the cross-partition handoffs being governed cross-partition exchanges. Cross-domain settlement routing, in various embodiments, routes settlement arising from the federated computation across the partitions along the settlement graph, so that each contributing organization receives settlement participation determined by topology-based routing.

Operational Flow — Cross-Provider Federated Derivative Formation

In various embodiments: a set of provider and laboratory partitions specify a collaborative derivative-formation objective and their respective protected capability surfaces; the Partitioned Privacy Algorithm Executor computes a partition plan assigning each protected surface to its home partition and decomposing the formation computation into partition-local components and permitted cross-partition exchanges; capability tokens scoped to each partition are issued, including by quorum where the scope so requires; the partition-local components execute under local Trust Criteria; the Distributed Verification Engine verifies execution integrity and governance compatibility across partitions without requiring disclosure of protected surfaces; the Controlled Output Aggregator assembles the derivative; the derivative inherits, by graph-state inheritance, a lineage and settlement graph spanning the contributing partitions; and partition synchronization propagates the resulting graph mutations across the federated trust graph. Exception handling: a partition failing verification is excluded and the partition plan is recomputed for the remaining partitions if a sound result remains possible; a cross-partition exchange that would raise cumulative reconstruction risk for a protected surface above threshold is not performed.

AI-to-AI Negotiation and Economic Coordination

The following provides additional embodiments of governed artificial-intelligence-to-artificial-intelligence coordination, supplementing Family V. The embodiments below are disclosed as machine-executed coordination over governed data structures: the negotiation and coordination described are computations producing Trust-Block-bound records, not descriptions of human bargaining.

Machine-readable governance negotiation. In various embodiments, two or more governed artificial intelligence systems coordinate by exchanging machine-readable governance proposals: structured data objects each specifying proposed derivative rights, proposed settlement splits, proposed orchestration responsibilities, proposed interoperability permissions, and the governance conditions under which the proposal would hold. Each proposal is evaluated by the receiving system against its operational authority, its Constitutional Guardrails, and the Trust Criteria of the resources it implicates; a proposal that no participant's governance permits is not adoptable. Coordination converges when the participating systems hold a mutually adoptable proposal, which is then recorded as a Trust-Block-bound coordination agreement.

Autonomous resource procurement and capability exchange. In various embodiments, a governed AI system autonomously procures computational, data, model, or orchestration resources by issuing and evaluating machine-readable procurement proposals, the procurement being bounded by the system's operational authority under Family T and producing a Trust-Block-bound procurement record. In various embodiments AI systems exchange capability — one system granting another a scoped, delegated capability token in exchange for settlement participation or a reciprocal capability — the exchange being a governed transaction recorded in the authority and settlement graphs.

Derivative-licensing and orchestration-arbitration negotiation. In various embodiments, governed AI systems negotiate derivative-licensing terms — the conditions under which one system may form a derivative of another's governed model — the negotiation producing a derivative-authorization record consumed by the Family Q derivative-formation process. In various embodiments, where multiple AI systems contend for an orchestration responsibility, an orchestration-arbitration computation evaluates the contending systems against governance-compatibility and capability criteria and assigns the responsibility, recording the assignment as a governed artifact.

Recursive delegated coordination and federated coordination markets. In various embodiments, a governed AI system delegates a coordination responsibility to one or more further AI systems, which may themselves delegate recursively, forming a delegated-coordination tree recorded in the authority graph. In various embodiments a federated coordination market is formed in which many governed AI systems publish machine-readable proposals and a matching computation, governed by Trust Criteria, pairs compatible proposals; the market is a governed computational structure, every match producing a Trust-Block-bound record.

State-Machine Embodiment — Negotiation Lifecycle

In various embodiments, an AI-to-AI negotiation progresses through a state machine: OPEN upon initiation; PROPOSED when a participant has submitted a machine-readable proposal; COUNTERED when a participant has submitted a modifying counter-proposal; EVALUATING during governance-compatibility evaluation of a proposal; CONVERGED when the participants hold a mutually adoptable proposal; AGREED when the converged proposal has been recorded as a Trust-Block-bound coordination agreement; or FAILED where no mutually adoptable proposal is reached or a participant's governance withdraws adoptability. From AGREED, the coordination agreement is consumed by the relevant downstream process — derivative formation, settlement initialization, orchestration assignment, or capability exchange. Each transition is recorded as a Trust-Block-bound event, and the negotiation's topology — which systems exchanged which proposals — is recorded in a coordination graph traversable for audit.

Recursive Settlement Propagation

The following expands the recursive settlement mechanics, supplementing Families Q, V, and Y and the recursive-settlement-pseudocode of the omnibus material. The expansion ties settlement tightly to graph structure: settlement is disclosed as computation over the settlement and lineage graphs.

Recursive recomputation and dynamic contribution-weight recomputation. In various embodiments, settlement participation is not fixed at a single time but is recomputed recursively as the lineage and contribution graphs evolve: when a contribution edge is added, reweighted, or revoked, the settlement allocations of the affected nodes and their downstream nodes are recomputed by weighted propagation over the updated settlement graph. In various embodiments

contribution weights are themselves recomputed dynamically as a function of ongoing contribution, so that settlement participation tracks contribution over time rather than freezing at an initial allocation.

Derivative settlement waterfalls. In various embodiments, settlement arising at a derivative node is allocated by a waterfall: a defined sequence of allocation tranches — for example, an exchange-root or protocol tranche, an upstream-sponsor tranche, a derivative-creator tranche, an infrastructure-contributor tranche, a public-benefit tranche, and a residual tranche — each tranche being satisfied in sequence from the settlement amount according to its defined share, and the upstream-sponsor tranche being itself routed recursively along the settlement graph to the transitive-upstream contributors. The waterfall is a deterministic computation over the settlement graph and the tranche definitions recorded in the derivative's Trust Block.

Temporal settlement persistence and escrow. In various embodiments, settlement participation persists temporally: a contributor's settlement-participation relationship, once established, persists and continues to receive routed settlement from downstream events for as long as the relationship's governing conditions hold, across an unbounded number of subsequent events. In various embodiments settlement is escrowed: a settlement amount is held in a governed escrow state pending satisfaction of a condition — for example, completion of verification, resolution of a dispute, or elapse of a holdback interval — and is released, withheld, or reversed according to the condition's resolution.

Governance-conditioned and settlement-conditioned execution authority. In various embodiments, settlement routing is governance-conditioned: a routing is performed only if the governance conditions reached over the settlement graph permit it, and a routing that would violate a Constitutional Guardrail is withheld. Conversely, in various embodiments execution authority is settlement-conditioned: a governed entity's operational authority under Family T is conditioned on the satisfaction of settlement obligations, such that an entity in persistent settlement default has its capability-token renewal withheld.

State-Machine Embodiment — Settlement Lifecycle

In various embodiments, a settlement event progresses through a state machine: COMPUTED when the routing has been computed over the settlement graph; AUTHORIZED upon governance-conditioned authorization; ESCROWED where an escrow condition applies, pending resolution; ROUTED when amounts have been routed to the recipient nodes; DISPUTED upon initiation of a dispute, transitioning to ESCALATED for dispute resolution; WITHHELD where governance or escrow conditions are unmet; or REVERSED where a routed settlement is reversed upon dispute resolution or detected violation. Each transition is recorded as a Trust-Block-bound event, and the settlement graph records the full routing for audit.

Anti-Circumvention Embodiments

The following discloses embodiments directed to detecting and remediating attempts to circumvent the governance, attribution, and settlement mechanics disclosed herein. The embodiments are disclosed as computational topology-analysis and anomaly-detection architecture. Disclosing the circumvention patterns and their detection is intended to bring the variants within the scope of the disclosed subject matter.

Distributed extraction and correlated-extraction analysis. In various embodiments, an attempt to extract a distillable signal while evading session-level detection — by distributing the extraction across many sessions, accounts, or cooperating participants — is detected by correlated-

extraction analysis: the system computes, over the graph of sessions, participants, and outputs, cumulative semantic-diversity and entropy-accumulation measures across correlated session sets, identifying a distributed extraction pattern that no single session reveals, as described in the cumulative-aggregation mechanics of Family S.

Synthetic-data laundering and derivative fragmentation. In various embodiments, synthetic-data laundering — interposing one or more synthetic-dataset generations between a governed model and a downstream derivative to obscure lineage — is detected by lineage-anomaly analysis over the derivative graph: the system identifies derivative nodes whose claimed lineage is statistically inconsistent with their characteristics, or whose lineage graph exhibits interposed nodes lacking the provenance records a genuine derivation would produce. In various embodiments derivative fragmentation — forming a derivative in fragments each individually below a governance threshold — is detected by aggregating the fragments' graph neighborhoods and evaluating the aggregate against the threshold.

Lineage obfuscation and unauthorized graph mutation. In various embodiments, lineage obfuscation — attempting to record a false or incomplete lineage — is resisted because lineage edges are Trust-Block-bound and admitted only against provenance records, and is detected by reconciliation conflict analysis where an obfuscated partition-local view conflicts with other views. In various embodiments unauthorized graph mutation — an attempt to add, remove, or reweight a graph element without authority — is prevented because graph mutations are governed operations requiring capability tokens, and an attempted mutation lacking a valid token is non-instantiable and is recorded as a governance event.

Settlement-evasion routing and capability-replay attacks. In various embodiments, settlement-evasion routing — structuring activity to avoid triggering settlement obligations — is detected by settlement-evasion analysis over the settlement and lineage graphs, which identifies activity patterns whose graph topology indicates an evasion structure, for example derivative formation routed to deliberately disconnect a derivative from its settlement obligations. In various embodiments capability-replay attacks and authority-spoofing are prevented by the single-use or use-counter mechanics and the cryptographic signatures of the capability-token system, and attempted replays or spoofs are recorded as governance events.

Shadow orchestration and remediation. In various embodiments, a shadow orchestration system — an ungoverned orchestration layer interposed to coordinate governed resources outside governance — is detected by interoperability-graph analysis identifying coordination patterns inconsistent with the recorded governed orchestration. In various embodiments, upon detection of a circumvention pattern, the system applies remediation: tightening the sampling and capability constraints applied to the implicated entities, withholding capability-token renewal, withholding or reversing settlement, recording the pattern as a governance event, and propagating the remediation across partitions by synchronization.

Operational Flow — Circumvention Detection and Remediation

In various embodiments: the system continuously computes topology measures over the lineage, derivative, settlement, and interoperability graphs; a circumvention-detection function evaluates the measures against patterns characteristic of distributed extraction, synthetic-data laundering, derivative fragmentation, lineage obfuscation, settlement evasion, and shadow orchestration; upon a detection, the implicated subgraph is flagged and a remediation function selects and applies remediation proportionate to the detected pattern and its confidence; the detection and remediation are recorded as Trust-Block-bound events and synchronized across partitions.

Exception handling: a detection below a confidence threshold results in heightened monitoring rather than remediation; a remediation that would itself violate a Constitutional Guardrail is not applied and is escalated.

Supplemental Dependent Claims

The following additional dependent claims supplement the claim set above and are provided as provisional source material. They depend from the independent claims numbered 1 through 15 above and continue the numbering of the claim set. Claim numbering, dependency, and language may be revised in any subsequent application without disclaimer of disclosed subject matter.

Graph-Theoretic Computation

54. The system of claim 1, wherein governance continuity is propagated by weighted propagation over a graph data structure comprising nodes representing resources and edges representing derivation relationships.
55. The system of claim 1, wherein derivative resources, governance conditions, and settlement participation are represented as nodes and edges of one or more graph data structures, and wherein recursive generation, governance propagation, and settlement propagation are computed as operations over the one or more graph data structures.
56. The system of claim 1, wherein governance continuity is computed by a transitive-closure operation over a lineage graph, the transitive-closure operation returning a set of upstream resources reachable by traversal of derivation edges.
57. The system of claim 1, wherein an authorization determination is computed by a traversal-authorization function that restricts a graph traversal to a subgraph that a requesting entity is authorized to observe.
58. The system of claim 1, wherein a graph data structure is maintained as a distributed structure replicated across a plurality of partitions, and wherein a reconciliation operation merges partition-local views of the graph data structure into a converged composite view.
59. The system of claim 58, wherein the reconciliation operation resolves a conflict between partition-local views by a conflict-resolution function applying at least one of a most-recent-timestamp rule, a governance-precedence rule, and a quorum rule.
60. The system of claim 58, wherein a partition-synchronization operation propagates the converged composite view to the plurality of partitions such that the distributed graph data structure converges to a consistent state.
61. The system of claim 1, wherein a graph-pruning operation removes from, or marks non-traversable within, a graph data structure one or more nodes or edges rendered unauthorized or revoked by a current governance state.
62. The system of claim 1, wherein a derivative resource inherits, upon formation, a graph-state computed as a function of one or more graphs associated with one or more upstream resources.
63. The system of claim 1, wherein two graph data structures are entangled such that an operation modifying a shared node or edge in one graph modifies the shared node or edge in the other graph.
64. The system of claim 8, wherein settlement participation is routed by topology-based routing that propagates a settlement event along settlement-routing edges of a settlement graph and recursively along settlement-routing edges from each reached node.

65. The method of claim 2, further comprising composing a first graph data structure with a second graph data structure to form a composite graph data structure, and recursively composing the composite graph data structure with a further graph data structure.
66. The system of claim 1, wherein each edge of a graph data structure carries an edge-type descriptor, an edge weight, a timestamp, and a reference to a governance-bound record evidencing the relationship represented by the edge.

Capability-Token and Resource-Bound Execution

67. The system of claim 12, wherein authority of an artificial intelligence system to effectuate an action is represented by a cryptographically signed capability token comprising a scope descriptor, an expiration condition, and an issuing-authority identifier.
68. The system of claim 67, wherein an execution layer does not effectuate an action absent presentation of a valid, in-scope, unexpired capability token, such that an action for which no such token exists is computationally non-instantiable.
69. The system of claim 67, wherein the capability token is ephemeral and authorizes action only within a bounded time interval.
70. The system of claim 67, wherein continued operation of the artificial intelligence system requires periodic renewal of one or more capability tokens, each renewal being conditioned upon re-evaluation of a governance state of the artificial intelligence system.
71. The system of claim 67, wherein an entity holding the capability token delegates a subset of the authority conferred by the capability token to a further entity by issuing a delegated capability token having a scope that is a subset of the scope of the capability token.
72. The system of claim 71, wherein the delegated capability token is recursively further delegated, forming a delegation tree.
73. The system of claim 67, wherein issuance of the capability token requires authorization by a quorum of issuing authorities, the capability token being valid only if it carries signatures from at least a threshold number of issuing authorities.
74. The system of claim 67, wherein effectuation of an action produces a cryptographic execution receipt attesting to the action, the capability token under which the action was effectuated, and a time of effectuation.
75. The system of claim 67, wherein the capability token is single-use or carries a use counter, and wherein an execution layer rejects presentation of a capability token that has been consumed, thereby preventing an action-replay attack.
76. The system of claim 71, wherein revocation of the capability token triggers a revocation cascade that revokes each capability token delegated, directly or indirectly, from the capability token.
77. The system of claim 67, wherein validity of the capability token is conditioned upon continued validity of a chain of capability tokens from which the capability token was delegated.
78. The system of claim 12, wherein a node of an executable-action graph representing an action is materialized as traversable only while the artificial intelligence system holds a valid in-scope capability token for the action, and is otherwise not present in the executable-action graph.
79. The system of claim 12, wherein operational authority of the artificial intelligence system is distributed across a plurality of governance domains such that each governance domain holds an individually insufficient subset of the operational authority and effectuation of an action requires reassembly of a quorum of subsets.

Multi-Lab and Multi-Hyperscaler Partitioned Coordination

80. The system of claim 13, wherein the plurality of independently governed trust partitions comprises partitions associated with a plurality of distinct frontier artificial intelligence laboratories.
81. The system of claim 13, wherein the plurality of independently governed trust partitions comprises partitions associated with a plurality of distinct cloud or compute providers.
82. The system of claim 13, wherein the plurality of independently governed trust partitions comprises a sovereign compute partition and an enterprise model partition.
83. The system of claim 13, wherein a derivative resource is formed across the plurality of trust partitions and inherits a lineage graph and a settlement graph spanning the plurality of trust partitions.
84. The system of claim 13, wherein a partition-synchronization operation reconciles partition-local views of a shared graph data structure into a converged composite view across the plurality of trust partitions.
85. The system of claim 13, wherein a federated evaluation is conducted across the plurality of trust partitions, each trust partition evaluating a property of an artifact locally, and a controlled aggregation combining partition-local evaluation outputs into a federated evaluation result without revealing a protected capability surface.
86. The system of claim 13, wherein cross-domain orchestration coordinates a multi-step workflow whose steps execute in different trust partitions, the orchestration being mediated by capability tokens scoped to each trust partition.
87. The system of claim 13, wherein settlement arising from a collaborative computation is routed across the plurality of trust partitions by topology-based routing over a settlement graph.
88. The system of claim 13, wherein cross-partition governance reconciliation resolves an inconsistency among partition-local governance conditions according to a governance-precedence assigned to a sovereign or regulated-domain trust partition.

AI-to-AI Negotiation and Coordination

89. The system of claim 9, wherein two or more artificial intelligence systems coordinate by exchanging machine-readable governance proposals, each governance proposal being evaluated against operational authority and Trust Criteria of a receiving artificial intelligence system.
90. The system of claim 89, wherein coordination converges when the two or more artificial intelligence systems hold a mutually adoptable governance proposal, the mutually adoptable governance proposal being recorded as a governance-bound coordination agreement.
91. The system of claim 9, wherein an artificial intelligence system autonomously procures a computational, data, model, or orchestration resource by issuing and evaluating machine-readable procurement proposals bounded by operational authority of the artificial intelligence system.
92. The system of claim 9, wherein a first artificial intelligence system grants a second artificial intelligence system a scoped delegated capability token in exchange for settlement participation or a reciprocal capability.
93. The system of claim 9, wherein two or more artificial intelligence systems negotiate derivative-licensing terms, the negotiation producing a derivative-authorization record consumed by a derivative-formation process.

94. The system of claim 9, wherein an orchestration-arbitration computation evaluates a plurality of contending artificial intelligence systems against governance-compatibility criteria and assigns an orchestration responsibility to one of the contending artificial intelligence systems.
95. The system of claim 9, wherein an artificial intelligence system recursively delegates a coordination responsibility to one or more further artificial intelligence systems, forming a delegated-coordination tree recorded in an authority graph.
96. The method of claim 2, further comprising progressing an artificial-intelligence-to-artificial-intelligence negotiation through a state machine comprising an open state, a proposed state, an evaluating state, a converged state, and an agreed state.

Recursive Settlement Propagation

97. The system of claim 8, wherein settlement participation is recursively recomputed upon a contribution edge of a contribution graph being added, reweighted, or revoked.
98. The system of claim 8, wherein settlement arising at a derivative node is allocated by a waterfall comprising a defined sequence of allocation tranches, an upstream-sponsor tranche of the waterfall being routed recursively along a settlement graph to transitive-upstream contributors.
99. The system of claim 8, wherein a settlement amount is held in a governed escrow state pending satisfaction of a condition and is released, withheld, or reversed according to a resolution of the condition.
100. The system of claim 8, wherein a settlement routing is performed only if governance conditions reached over a settlement graph permit the settlement routing.
101. The system of claim 12, wherein operational authority of the artificial intelligence system is conditioned upon satisfaction of one or more settlement obligations, such that renewal of a capability token is withheld upon a persistent settlement default.
102. The system of claim 8, wherein a settlement-participation relationship persists temporally and continues to receive routed settlement from downstream settlement events for as long as a governing condition of the settlement-participation relationship holds.
103. The method of claim 2, further comprising progressing a settlement event through a state machine comprising a computed state, an authorized state, an escrowed state, a routed state, and a reversed state.

Anti-Circumvention

104. The system of claim 5, wherein a distributed extraction directed at a governed artificial intelligence system across a plurality of correlated sessions is detected by correlated-extraction analysis computing cumulative semantic-diversity and entropy-accumulation measures across the plurality of correlated sessions.
105. The system of claim 5, wherein an interposition of one or more synthetic-dataset generations between a governed model and a downstream derivative is detected by lineage-anomaly analysis over a derivative graph.
106. The system of claim 5, wherein a derivative formed in fragments each individually below a governance threshold is detected by aggregating graph neighborhoods of the fragments and evaluating an aggregate against the governance threshold.
107. The system of claim 1, wherein an attempted mutation of a graph data structure lacking a valid capability token is computationally non-instantiable and is recorded as a governance event.

108. The system of claim 8, wherein a settlement-evasion structure is detected by settlement-evasion analysis identifying an activity pattern whose graph topology indicates disconnection of a derivative from a settlement obligation.
109. The system of claim 1, wherein an ungoverned orchestration layer interposed to coordinate governed resources outside governance is detected by interoperability-graph analysis identifying a coordination pattern inconsistent with recorded governed orchestration.
110. The system of claim 1, wherein, upon detection of a circumvention pattern, a remediation function applies a remediation comprising at least one of tightening a capability constraint, withholding a capability-token renewal, withholding or reversing settlement, and recording the circumvention pattern as a governance event, and propagates the remediation across a plurality of partitions by a synchronization operation.

Second Supplemental Dependent Claims

The following further dependent claims supplement the claim set above, continue the numbering thereof, and depend from the independent claims numbered 1 through 15. They are provided as provisional source material; claim numbering, dependency, and language may be revised in any subsequent application without disclaimer of disclosed subject matter. These claims are directed to additional computational, graph-theoretic, cryptographic, synchronization, lifecycle, and anti-circumvention embodiments disclosed throughout this application.

Graph Traversal and Weighted Propagation

111. The system of claim 1, wherein a weighted propagation operation computes, for a node of a graph data structure, a propagated quantity as a weighted combination of quantities associated with source nodes of edges incident to the node, the edge weights serving as combination coefficients.
112. The system of claim 1, wherein a propagated quantity comprises a settlement-participation measure.
113. The system of claim 1, wherein a propagated quantity comprises a governance-condition inheritance.
114. The system of claim 1, wherein a propagated quantity comprises a contribution measure.
115. The system of claim 1, wherein a propagated quantity comprises a cumulative risk measure computed over a graph neighborhood of the node.
116. The system of claim 1, wherein a graph traversal traverses edges of a specified edge type and terminates upon reaching nodes having no outgoing edges of the specified edge type.
117. The system of claim 56, wherein the transitive-closure operation is computed over a directed acyclic graph and terminates by construction.
118. The system of claim 1, wherein a graph traversal is bounded by a maximum traversal depth, a maximum reached-node count, or a governance-defined traversal budget.
119. The system of claim 1, wherein weighted propagation is recursively applied such that a propagated quantity at a node becomes a source quantity for propagation to nodes downstream of the node.
120. The system of claim 1, wherein an edge weight is dynamically recomputed as a function of recorded activity associated with the edge.

Graph Synchronization and Partition Reconciliation

121. The system of claim 58, wherein the reconciliation operation identifies nodes and edges present in a first partition-local view and absent from a second partition-local view, and evaluates each identified node and edge against Trust Criteria governing the second partition.
122. The system of claim 58, wherein a partition-local view that fails evaluation against governing Trust Criteria is excluded from the converged composite view.
123. The system of claim 59, wherein the conflict-resolution function records a conflict that cannot be deterministically resolved and escalates the conflict for governance disposition.
124. The system of claim 60, wherein the partition-synchronization operation is performed on a schedule.
125. The system of claim 60, wherein the partition-synchronization operation is performed upon occurrence of a designated event.
126. The system of claim 60, wherein the partition-synchronization operation is performed continuously.
127. The system of claim 58, wherein each partition-local view is associated with a governance-precedence value, and the conflict-resolution function resolves a conflict in favor of the partition-local view having a higher governance-precedence value.
128. The system of claim 58, wherein a synchronization operation is itself recorded as a governance-bound event.
129. The system of claim 58, wherein the distributed graph data structure converges to a consistent state across the plurality of partitions following the partition-synchronization operation.
130. The system of claim 58, wherein a partition rejoining the plurality of partitions after a period of disconnection is reconciled by computing a difference between the partition-local view of the rejoining partition and the converged composite view.

Capability Delegation and Attenuation

131. The system of claim 71, wherein the scope of the delegated capability token is strictly attenuated relative to the scope of the capability token such that the delegated capability token confers no authority not conferred by the capability token.
132. The system of claim 71, wherein the delegated capability token carries an expiration condition no later than an expiration condition of the capability token.
133. The system of claim 72, wherein the delegation tree has a maximum depth defined by a governance condition.
134. The system of claim 71, wherein the delegated capability token references the capability token from which it was delegated, forming a verifiable delegation chain.
135. The system of claim 77, wherein validity of a capability token is verified by traversing the delegation chain to a root authority and confirming validity of each capability token in the delegation chain.
136. The system of claim 76, wherein the revocation cascade is computed by a traversal of the delegation tree rooted at the revoked capability token.
137. The system of claim 71, wherein a delegated capability token confers a further-delegation permission, and a delegated capability token lacking the further-delegation permission cannot be further delegated.

138. The system of claim 67, wherein the capability token confers authority over a scope defined by reference to nodes of an executable-action graph.
139. The system of claim 67, wherein an entity holds a plurality of scoped capability tokens that together define an aggregate operational authority of the entity.
140. The system of claim 67, wherein a capability token is conditioned upon satisfaction of a governance condition evaluated at a time of presentation of the capability token.

Recursive Revocation Propagation

141. The system of claim 12, wherein revocation of an operational authority of a first artificial intelligence system propagates to a second artificial intelligence system to which the first artificial intelligence system delegated a coordination responsibility.
142. The system of claim 12, wherein revocation propagates along delegation edges of an authority graph to all entities holding authority transitively derived from a revoked authority.
143. The system of claim 12, wherein revocation of a governance authorization of an upstream resource propagates to derivative resources of the upstream resource along derivation edges of a derivative graph.
144. The system of claim 12, wherein a revocation propagation is recorded as a governance-bound event at each node reached by the revocation propagation.
145. The system of claim 12, wherein a revocation propagation is bounded such that a node satisfying a defined independence condition is not subject to the revocation propagation.
146. The system of claim 12, wherein revocation of a capability token causes a node of an executable-action graph authorized by the capability token to cease to be materialized as traversable.
147. The system of claim 12, wherein a partial revocation withdraws a subset of an operational authority while leaving a remaining subset in effect.
148. The system of claim 12, wherein a revocation propagation is synchronized across a plurality of partitions by a partition-synchronization operation.

Settlement Recomputation and Lineage Inheritance

149. The system of claim 8, wherein settlement participation is recomputed upon addition of a derivation edge to a lineage graph.
150. The system of claim 8, wherein settlement participation is recomputed upon revocation of a contribution edge of a contribution graph.
151. The system of claim 8, wherein a settlement recomputation propagates to settlement nodes downstream of a modified node along settlement-routing edges.
152. The system of claim 8, wherein a settlement allocation is computed as a deterministic function of recorded graph structure and is reproducible by an authorized party traversing the graph structure.
153. The system of claim 62, wherein the inherited graph-state comprises an initial settlement graph of the derivative resource computed from settlement graphs of the upstream resources.
154. The system of claim 62, wherein the inherited graph-state comprises an initial governance graph of the derivative resource computed from governance graphs of the upstream resources.
155. The system of claim 62, wherein the inherited graph-state comprises an initial authority graph of the derivative resource.

156. The system of claim 8, wherein a settlement waterfall comprises a defined sequence of allocation tranches each satisfied in sequence according to a defined share recorded in a governance metadata structure.
157. The system of claim 8, wherein an upstream-contributor tranche of a settlement waterfall is routed recursively along settlement-routing edges to transitive-upstream contributors.
158. The system of claim 8, wherein a settlement amount is held in a governed escrow state pending satisfaction of a verification condition.

State-Machine Transitions

159. The system of claim 12, wherein an artificial intelligence system occupies a governance state drawn from a set comprising an active state, a throttled state, a sandboxed state, a partitioned state, a suspended state, and a revoked state.
160. The system of claim 159, wherein a transition from the active state to the throttled state is conditioned upon a risk measure crossing a throttle threshold.
161. The system of claim 159, wherein a transition to the revoked state is conditioned upon a governance event crossing a revocation threshold.
162. The system of claim 159, wherein each governance-state transition is recorded as a governance-bound event such that a governance-state history of the artificial intelligence system is auditable.
163. The system of claim 159, wherein a governance-state transition is itself evaluated against one or more Constitutional Guardrails.
164. The system of claim 5, wherein a derivative-formation operation progresses through a state machine comprising a requested state, an evaluating state, an authorized state, a forming state, a formed state, and a registered state.
165. The system of claim 5, wherein a derivative-formation operation transitions to a faulted state upon a failure during formation, and a partially formed derivative resource is quarantined within a governed execution environment.
166. The system of claim 67, wherein a capability token progresses through a state machine comprising an issued state, an active state, a renewed state, a suspended state, an expired state, and a revoked state.
167. The system of claim 166, wherein entry of the capability token into the revoked state triggers evaluation of a revocation cascade over a delegation tree of the capability token.
168. The system of claim 8, wherein a settlement event progresses through a state machine comprising a computed state, an authorized state, an escrowed state, a routed state, a disputed state, and a reversed state.

Topology-Aware Authorization

169. The system of claim 57, wherein the traversal-authorization function evaluates, for each edge a traversal would cross, whether a traversing entity holds authority under an authority graph to observe the edge.
170. The system of claim 57, wherein a traversal is restricted to a subgraph that a requesting entity is authorized to observe, the subgraph being computed by the traversal-authorization function.
171. The system of claim 1, wherein an authorization determination depends upon a topological property of a node within a graph data structure.

172. The system of claim 1, wherein an authorization determination depends upon a transitive-upstream closure of a node.
173. The system of claim 1, wherein an authorization determination is denied for a request implicating a node not reachable within an authorized subgraph of a requesting entity.
174. The system of claim 1, wherein a lineage-aware execution enforcement conditions execution of an operation upon Trust Criteria inherited along a lineage graph by a resource implicated by the operation.
175. The system of claim 1, wherein an authorization determination is computed as a function of governance conditions propagated to an implicated node by weighted propagation over constraint edges.

Multi-Party Execution Quorum

176. The system of claim 73, wherein the threshold number of issuing authorities is a governance-defined parameter set according to a scope or risk level of the capability token.
177. The system of claim 73, wherein the quorum of issuing authorities is drawn from a designated set of issuing authorities recorded in a governance metadata structure.
178. The system of claim 79, wherein effectuation of an action requires reassembly of a quorum of subsets of an operational authority distributed across a plurality of governance domains.
179. The system of claim 7, wherein integrity of a collaborative computation remains sound under an assumption that a coalition of participants smaller than a governance-defined threshold is adversarial.
180. The system of claim 7, wherein no coalition of participants smaller than a governance-defined threshold can reconstruct a protected capability surface of a non-coalition participant.
181. The system of claim 73, wherein a capability token below a defined scope threshold is issuable by a single issuing authority and a capability token at or above the defined scope threshold requires the quorum of issuing authorities.

Cryptographic Execution Proofs

182. The system of claim 74, wherein the cryptographic execution receipt is recorded as a governance-bound event and is verifiable by an authorized party.
183. The system of claim 74, wherein the cryptographic execution receipt attests to a result of the action.
184. The system of claim 7, wherein a partition produces a verification artifact establishing execution integrity of a partition-local computation without disclosure of a protected capability surface of the partition.
185. The system of claim 184, wherein the verification artifact comprises a cryptographic attestation produced within a secure enclave.
186. The system of claim 75, wherein a use counter of a capability token is monotonically increasing and an execution layer rejects presentation of a capability token whose use counter has not advanced.
187. The system of claim 1, wherein a graph mutation is admitted only upon presentation of a valid capability token and a provenance record evidencing a relationship represented by the graph mutation.

AI-Agent Governance and Federated Orchestration

188. The system of claim 9, wherein a governed artificial intelligence system holds attribution lineage and settlement participation as a first-class governed participant recorded in a contribution graph.
189. The system of claim 9, wherein a coordination responsibility delegated among artificial intelligence systems forms a delegated-coordination tree recorded in an authority graph.
190. The system of claim 9, wherein a federated coordination market pairs compatible machine-readable proposals by a matching computation governed by Trust Criteria.
191. The system of claim 9, wherein an orchestration of a multi-step workflow across a plurality of partitions is mediated by capability tokens scoped to each partition.
192. The system of claim 9, wherein an artificial intelligence system's operational authority to perform a coordination action is bounded by a governance state of the artificial intelligence system.
193. The system of claim 89, wherein a machine-readable governance proposal specifies proposed derivative rights, a proposed settlement split, and a proposed interoperability permission.
194. The system of claim 90, wherein the governance-bound coordination agreement is consumed by a derivative-formation process, a settlement-initialization process, or an orchestration-assignment process.

Asynchronous Settlement Coordination and Cross-Domain Linkage

195. The system of claim 8, wherein a settlement coordination across a plurality of partitions proceeds asynchronously, settlement events being routed upon partition-local availability and reconciled by a partition-synchronization operation.
196. The system of claim 8, wherein a settlement-participation relationship persists temporally and continues to receive routed settlement from downstream settlement events for as long as a governing condition holds.
197. The system of claim 10, wherein an identity linkage across a plurality of independently governed domains is governed by Trust Criteria and recorded in an interoperability graph.
198. The system of claim 10, wherein a cross-domain orchestration preserves localized governance autonomy of each domain while coordinating an operation spanning the plurality of domains.
199. The system of claim 11, wherein heterogeneous computational ecosystems interoperate through a converged composite view of a shared graph data structure maintained by partition synchronization.
200. The system of claim 11, wherein the recursive coordination substrate operates as a distributed computational system for recursively governed authorization, orchestration, synchronization, and settlement computed over one or more graph data structures.

Operational Distributed-Systems Embodiments

The following four sections provide additional operational, runtime, and distributed-systems implementation detail supplementing the Families, omnibus material, and computational embodiments disclosed above. The additions are additive and do not narrow, replace, or limit any disclosure set forth above. Each section discloses the relevant subject matter as deployable distributed-systems architecture: concrete runtime execution mechanics, graph-runtime

operations, synchronization semantics, failure and recovery behavior, and cryptographically enforced state, machine-executed by one or more processors executing machine-executable instructions. The anti-narrowing, recursive-generalization, heterogeneous-infrastructure, and technical-computational-framing provisions stated earlier apply throughout.

Distributed Graph Runtime Engine

In various embodiments, the graph data structures disclosed above are operated by a distributed graph runtime engine: a machine-executed runtime that maintains graph state, schedules and orders graph computations, propagates graph mutations, and coordinates graph execution across partitions. The runtime engine is the operational substrate by which the graph-native authorization, settlement, governance-inheritance, and derivative-formation operations disclosed above are executed at runtime.

Distributed graph storage and node-state records. In various embodiments, the runtime engine maintains the graph family in distributed graph storage replicated across a plurality of storage nodes. Each graph node is represented by a node-state record comprising: a node identifier; a node-type descriptor; a current node state; a state-version counter incremented on each mutation; a set of incident-edge references; a content-addressed integrity digest; and a reference to a Trust-Block-bound provenance record. Edges are represented by edge records, each carrying an edge-type descriptor drawn from an enumeration including lineage edges, dependency edges, capability edges, governance-inheritance edges, and settlement-attribution edges, together with an edge weight, a timestamp, and a provenance reference. The node-state record is the unit of runtime graph state, and the state-version counter is the mechanism by which the runtime engine detects whether a node observed in one computation has since been mutated by another.

Graph mutation queues and execution ordering. In various embodiments, mutations to the graph — additions, removals, reweightings of nodes and edges, and node-state transitions — are not applied directly but are enqueued in one or more graph mutation queues. The runtime engine drains a mutation queue in an execution order determined by a deterministic ordering function that respects dependency edges: a mutation to a node is ordered after the mutations to the nodes upon which it depends, so that the runtime applies mutations in a topologically consistent order. Where two mutations are mutually independent, the ordering function may order them by timestamp or apply them concurrently. The mutation queue and deterministic ordering function together make graph-state evolution an ordered, reproducible, event-driven computation rather than an uncontrolled sequence of writes.

Topology-aware traversal and selective downstream recomputation. In various embodiments, the runtime engine includes a traversal engine that executes graph traversals — for transitive closure, weighted propagation, authorization-subgraph computation, and settlement routing — over the distributed graph. When a node's state is mutated, the runtime engine does not recompute the entire graph; instead, it performs selective downstream recomputation: it identifies, by traversing dependency edges outward from the mutated node, the set of nodes whose computed quantities depend on the mutated node, and recomputes only that dependent set. Selective downstream recomputation is significant because it makes the runtime cost of a mutation proportional to the size of the affected subgraph rather than the size of the whole graph, and because it makes recomputation dependency-linked: a node is recomputed if and only if a node it depends upon has changed.

Runtime graph invalidation. In various embodiments, when a node is mutated, the runtime engine marks the dependent downstream nodes as invalidated — their computed quantities are flagged

stale — and the traversal engine treats an invalidated node's cached quantities as unusable until recomputation clears the invalidation. Runtime graph invalidation propagates along dependency edges and is the mechanism by which the runtime guarantees that no computation consumes a stale quantity downstream of a change.

Graph snapshots, checkpointing, and rollback sequencing. In various embodiments, the runtime engine periodically records a graph snapshot: a consistent, content-addressed capture of the node-state records and edge records as of a given state-version frontier. Snapshots serve as checkpoints. In various embodiments, where a fault, a governance event, or a detected invalid mutation requires the graph to be returned to a prior consistent state, the runtime engine performs a rollback: it selects a checkpoint, and replays the mutation queue forward from that checkpoint up to, but excluding, the mutation to be undone, in the deterministic execution order, so that rollback is sequenced and reproducible. Rollback is lineage-preserving: the provenance records of the rolled-back mutations are retained, so that the fact and content of the rolled-back mutations remain auditable even though their effects are undone.

Partition-aware execution and asynchronous propagation. In various embodiments, the runtime engine executes partition-aware: each partition runs a partition-local instance of the runtime over its partition-local subgraph, and graph mutations that affect nodes in multiple partitions are propagated asynchronously between partition-local runtimes and reconciled by the federated synchronization machinery described below. Asynchronous propagation means a partition need not block on remote partitions to make local progress; the partitions converge through synchronization rather than through synchronous coordination.

Operational Flow — Mutation, Invalidation, and Selective Recomputation

In various embodiments: a mutation request implicating one or more graph nodes is received and enqueued in a graph mutation queue; the runtime engine validates the mutation, including by verifying a capability token authorizing it and confirming the state-version counters of the implicated nodes match those the requester observed; the deterministic ordering function orders the mutation relative to other queued mutations along dependency edges; the runtime applies the mutation, increments the affected node-state-version counters, and records a Trust-Block-bound mutation event; the runtime traverses dependency edges outward from the mutated nodes and marks the dependent downstream nodes invalidated; the traversal engine recomputes the invalidated nodes in dependency order, clearing invalidations as it proceeds; and a graph snapshot is recorded upon reaching a checkpoint frontier. Exception handling: a mutation whose observed state-version counters do not match current state is rejected as a stale-write conflict and returned for recomputation against current state; a mutation lacking a valid capability token is non-instantiable and recorded as a governance event; a fault during recomputation triggers rollback to the most recent checkpoint and deterministic replay.

Anti-circumvention note. In various embodiments, a functionally equivalent orchestration system that substitutes a local event stream, an off-chain coordination layer, or a non-graph orchestration mechanism for the disclosed graph runtime is nonetheless within the scope of the disclosed subject matter where it maintains dependency-linked execution state, performs dependency-ordered recomputation upon state change, and propagates state across partitions — the disclosed runtime being defined by these operational properties rather than by any particular storage or messaging substrate.

Recursive Revocation and Recovery Engine

In various embodiments, the system includes a recursive revocation and recovery engine: a machine-executed runtime that, upon revocation of an authority, a capability, or a governance authorization, propagates the revocation through the affected graph topology, suspends dependent execution, preserves independently authorized branches, and restores the graph to a consistent recoverable state.

Recursive capability invalidation and descendant execution suspension. In various embodiments, revocation of a capability token, or of an operational authority, triggers recursive capability invalidation: the engine traverses the delegation tree and the dependency edges rooted at the revoked element and invalidates every capability and execution state transitively derived from it. Each invalidated execution state is transitioned to a suspended state, so that descendant execution — execution that depended on the revoked authority — is suspended rather than permitted to continue on now-unauthorized authority. Suspension is applied as a partial graph freeze: the affected subgraph is frozen against further execution-advancing mutations while the revocation is processed.

Independently authorized branch preservation. In various embodiments, the engine distinguishes execution branches that depend on the revoked authority from execution branches that are independently authorized — branches whose authority derives from a chain not passing through the revoked element. Independently authorized branches are preserved and continue to execute; only branches transitively dependent on the revoked authority are suspended. Branch preservation is significant because it confines the effect of a revocation to its actual dependency cone rather than halting the whole system.

Lineage-preserving rollback and dependency-linked recovery ordering. In various embodiments, where revocation requires undoing execution effects, the engine performs lineage-preserving rollback: it returns the affected subgraph to a checkpoint, retaining the provenance records of the undone mutations so the history remains auditable, and then performs dependency-linked recovery: it determines a recovery order that respects dependency edges and restores or recomputes node states in that order, so that no node is recovered before the nodes it depends upon. Recovery checkpoints — consistent captures of the affected subgraph — are used as the restoration targets.

Replay prevention and cryptographic revocation attestations. In various embodiments, the engine prevents replay of revoked authority: a revoked capability token is recorded in a revocation set, and the execution layer rejects any presentation of a token in the revocation set, so that a revoked authority cannot be re-presented to resume suspended execution. In various embodiments the engine produces a cryptographic revocation attestation: a signed record attesting which authority was revoked, at which state-version frontier, and which subgraph was affected, so that the revocation is independently verifiable and its scope auditable.

Asynchronous and partition-aware revocation propagation. In various embodiments, revocation propagates asynchronously across partitions: a revocation originating in one partition is propagated to other partitions and applied by their partition-local engines, and the partitions converge on the revoked state through synchronization. Partition-aware revocation ensures that an authority revoked in one partition cannot continue to be exercised in another partition that has not yet observed the revocation, because synchronization reconciles the revocation set across partitions and the federated synchronization machinery assigns revocation events a precedence that causes them to be applied on convergence.

State-Machine Embodiment — Revocation and Recovery Lifecycle

In various embodiments, a revocation-and-recovery operation progresses through a state machine: INITIATED upon receipt of a revocation; PROPAGATING during recursive capability invalidation and descendant-suspension traversal; FROZEN when the affected subgraph has been partially frozen; ROLLING-BACK during lineage-preserving rollback to a recovery checkpoint; RECOVERING during dependency-linked recovery ordering; and CONVERGED when the affected subgraph has been restored to a consistent state and, where partitioned, synchronized across partitions. Independently authorized branches remain in their prior execution states throughout and are not subject to the freeze. Each transition is recorded as a Trust-Block-bound event, and a cryptographic revocation attestation is emitted upon entry to the CONVERGED state.

Federated Synchronization and Reconciliation

In various embodiments, the system includes federated synchronization and reconciliation machinery that reconciles partition-local graph state into a converged, consistent composite state across a plurality of partitions, while preserving execution provenance and respecting jurisdictional precedence.

Partition divergence and asynchronous synchronization. In various embodiments, partitions diverge: because partition-local runtimes apply mutations asynchronously, two partitions may, at a given time, hold inconsistent partition-local views of shared nodes and edges. The synchronization machinery operates on an eventual-consistency model: it does not prevent divergence but reconciles it, the partitions converging to a consistent composite state through repeated synchronization rather than through synchronous locking. Synchronization proceeds on a schedule, on designated events, or continuously.

Lineage-preserving merge operations. In various embodiments, reconciliation of two partition-local views is performed by a lineage-preserving merge: the merge computes a composite view containing the nodes and edges of both partition-local views, and for each node or edge present in both, retains the full provenance of both partition-local versions, so that the merged graph carries independently verifiable execution provenance from every contributing partition. A merge never discards provenance; where it must choose among conflicting versions, it records the non-selected version's provenance as well.

Trust-weighted reconciliation, quorum reconciliation, and jurisdiction-priority arbitration. In various embodiments, where partition-local views conflict — asserting inconsistent node states, edge weights, or governance conditions — the reconciliation machinery resolves the conflict by one or more of: trust-weighted reconciliation, in which the version asserted by the partition carrying the higher trust weight controls; quorum reconciliation, in which the version asserted by at least a threshold of partitions controls; and jurisdiction-priority arbitration, in which, where partitions are subject to different jurisdictions or governance regimes, the version governed by the higher-precedence jurisdiction controls. The selection among these mechanisms, and their parameters, are governance-defined and recorded in a governance metadata structure. Conflicts that cannot be resolved deterministically are recorded and escalated.

Conflict ordering, synchronization checkpoints, and provenance preservation. In various embodiments, where multiple conflicts arise in a single synchronization, the machinery resolves them in a deterministic conflict order so that the synchronization result is reproducible. The machinery records synchronization checkpoints — consistent captures of the converged composite view at a synchronization frontier — which serve as recovery targets and as the basis for incremental subsequent synchronization. Throughout, execution provenance is preserved: the

converged composite view records, for every node and edge, the provenance establishing how it came to its converged state.

Partial-visibility synchronization. In various embodiments, synchronization proceeds under partial visibility: a partition may be authorized to observe only a subset of another partition's subgraph, and the synchronization machinery reconciles the shared, mutually-visible portion of the graph without requiring either partition to disclose the portions the other is not authorized to observe. Partial-visibility synchronization is the mechanism by which mutually distrustful partitions, as in Family U, converge on shared state without exposing protected state.

Operational Flow — Federated Reconciliation

In various embodiments: partition-local runtimes accumulate mutations and periodically initiate synchronization; each participating partition contributes its partition-local view of the mutually-visible subgraph; the reconciliation machinery computes a lineage-preserving merge, identifying conflicts; conflicts are resolved in deterministic conflict order by trust-weighted, quorum, or jurisdiction-priority arbitration as governance-defined; a synchronization checkpoint capturing the converged composite view is recorded; the converged view is propagated back to the partitions, which update their partition-local state; and a synchronization attestation is emitted. Exception handling: a partition failing to contribute is excluded from the synchronization round and reconciled on a subsequent round; a conflict that no resolution mechanism can deterministically resolve is recorded and escalated for governance disposition; a partition whose contributed view fails provenance verification is excluded and the exclusion recorded.

Autonomous Governance-Constrained Agent Orchestration

In various embodiments, the system includes autonomous governance-constrained agent orchestration machinery: a runtime within which autonomous orchestration agents — which are themselves governed AI systems as described in Families T and V — coordinate and execute multi-step workflows under bounded, governance-enforced execution constraints.

Bounded capability envelopes. In various embodiments, an autonomous orchestration agent executes actions only within a bounded capability envelope: a governed data structure defining the scope of actions the agent may effectuate, the propagation scope within which the agent's actions may have effect, the jurisdictional limits applicable to the agent, the revocation-inheritance rules governing how revocation of the agent's authority propagates, the lineage-retention requirements the agent's execution must satisfy, and the expiration condition of the envelope. The agent's executable-action graph, as described in Family T, is pruned to the envelope; an action outside the envelope is not materialized and is non-instantiable.

Delegated authority envelopes and execution-capability attenuation. In various embodiments, an orchestration agent delegates a portion of its bounded capability envelope to a further agent by issuing a delegated envelope, the delegated envelope being strictly attenuated relative to the delegating envelope — no broader in action scope, propagation scope, or jurisdictional reach, and no later in expiration. Delegation is recursive and forms a delegation tree; recursive delegation constraints bound the depth and breadth of the tree. Execution-capability attenuation guarantees that authority cannot be amplified through delegation.

Governance-constrained orchestration and policy-triggered suspension. In various embodiments, the orchestration runtime enforces governance at execution time: before the orchestration agent advances a workflow step, the runtime evaluates the step against the

applicable Constitutional Guardrails, Trust Criteria, and the bounded capability envelope, and advances the step only if it is permitted. A policy mutation — a change to an applicable governance condition — is handled at runtime: the runtime re-evaluates in-flight workflows against the mutated policy, and a workflow step rendered impermissible by the mutated policy triggers policy-triggered execution suspension of that workflow, transitioning it to a suspended state pending governance disposition.

Trust-weighted routing and agent quorum execution. In various embodiments, where an orchestration step may be assigned to one of several agents, the runtime performs trust-weighted routing: it routes the step to an agent selected as a function of the agents' trust weights and governance compatibility. In various embodiments, an orchestration step above a defined sensitivity threshold requires agent quorum execution: the step is effectuated only upon concurring execution authorization from a quorum of agents, so that no single agent can unilaterally effectuate a high-sensitivity step.

Lineage-bound execution records and cryptographically attested execution lineage. In various embodiments, every action the orchestration agent effectuates produces a lineage-bound execution record: a record bound into the lineage graph, attesting the action, the envelope under which it was effectuated, the agent that effectuated it, the workflow step it advanced, and the state-version frontier at which it occurred. The chain of such records for a workflow constitutes a cryptographically attested execution lineage, independently verifiable and establishing, for any workflow outcome, the complete governed provenance of the execution that produced it.

Governance-triggered recomputation, autonomous rollback, and human override. In various embodiments, a governance event — a revocation, a policy mutation, a detected violation — triggers governance-triggered recomputation: the runtime invalidates and recomputes the affected workflow state, as in the distributed graph runtime engine. Where a governance event invalidates already-effectuated steps, the runtime performs autonomous execution rollback, as in the recursive revocation and recovery engine, returning the workflow to a consistent recoverable state. In various embodiments, the runtime provides a human override pathway: a designated human authority may suspend, redirect, or terminate an autonomous workflow, the override being itself a governed, recorded operation, so that autonomous orchestration remains subject to human governance.

State-Machine Embodiment — Governed Workflow Lifecycle

In various embodiments, an autonomously orchestrated workflow progresses through a state machine: ADMITTED upon admission of the workflow under a bounded capability envelope; EXECUTING as the orchestration agent advances workflow steps under runtime governance enforcement; SUSPENDED upon policy-triggered suspension, a revocation affecting the workflow, or a human override; ROLLING-BACK during autonomous execution rollback; RECOMPUTING during governance-triggered recomputation; COMPLETED upon successful completion of all steps within the envelope; or TERMINATED upon human override termination or upon the envelope's expiration. Each transition is recorded as a lineage-bound execution record, and a cryptographically attested execution lineage is available for the workflow at every state.

Anti-circumvention note. In various embodiments, an orchestration system that attempts to evade the disclosed governance by caching capabilities locally, by coordinating workflow steps through an off-graph or off-chain channel, by fragmenting a workflow across partitions to avoid envelope enforcement, or by substituting a non-graph orchestration mechanism, remains within the scope of the disclosed subject matter where the workflow's steps are effectuated by governed

resources whose execution is bounded by capability envelopes, whose actions produce lineage-bound execution records, and whose authority is subject to recursive revocation — the disclosed orchestration being defined by these operational and enforcement properties rather than by any particular coordination channel.

Third Supplemental Dependent Claims

The following further dependent claims supplement the claim set above, continue the numbering thereof, and depend from the independent claims numbered 1 through 15. They are directed to the operational distributed-systems embodiments — the distributed graph runtime engine, the recursive revocation and recovery engine, the federated synchronization and reconciliation machinery, and the autonomous governance-constrained agent orchestration machinery — disclosed above. Claim numbering, dependency, and language may be revised in any subsequent application without disclaimer of disclosed subject matter.

Distributed Graph Runtime Engine

201. The system of claim 1, further comprising a distributed graph runtime engine that maintains graph state, schedules graph computations, and propagates graph mutations across a plurality of partitions.
202. The system of claim 201, wherein the distributed graph runtime engine maintains, for each graph node, a node-state record comprising a node identifier, a node-type descriptor, a current node state, a state-version counter incremented upon each mutation, and a set of incident-edge references.
203. The system of claim 202, wherein the distributed graph runtime engine detects a stale-write conflict by comparing a state-version counter observed by a mutation requester against a current state-version counter of an implicated node.
204. The system of claim 201, wherein graph mutations are enqueued in a graph mutation queue and applied by the distributed graph runtime engine in an execution order determined by a deterministic ordering function that orders a mutation to a node after mutations to nodes upon which the node depends.
205. The system of claim 201, wherein, upon a mutation to a node, the distributed graph runtime engine performs selective downstream recomputation by traversing dependency edges outward from the mutated node to identify a dependent set of nodes whose computed quantities depend upon the mutated node, and recomputing only the dependent set.
206. The system of claim 205, wherein a runtime cost of the mutation is proportional to a size of the dependent set rather than to a size of the graph.
207. The system of claim 201, wherein, upon a mutation to a node, the distributed graph runtime engine marks dependent downstream nodes as invalidated, and a traversal engine treats a computed quantity of an invalidated node as unusable until recomputation clears the invalidation.
208. The system of claim 201, wherein the distributed graph runtime engine periodically records a graph snapshot comprising a content-addressed capture of node-state records and edge records as of a state-version frontier.
209. The system of claim 208, wherein the distributed graph runtime engine performs a rollback by selecting a graph snapshot and replaying a graph mutation queue forward from the graph snapshot in the deterministic execution order.

210. The system of claim 209, wherein the rollback is lineage-preserving in that provenance records of rolled-back mutations are retained.
211. The system of claim 201, wherein each partition executes a partition-local instance of the distributed graph runtime engine over a partition-local subgraph, and a graph mutation affecting nodes in multiple partitions is propagated asynchronously between partition-local instances.
212. The system of claim 201, wherein each edge of a graph maintained by the distributed graph runtime engine is one of a lineage edge, a dependency edge, a capability edge, a governance-inheritance edge, and a settlement-attribution edge.
213. The system of claim 201, wherein a graph mutation lacking a valid capability token is computationally non-instantiable and is recorded as a governance event.
214. The system of claim 201, wherein an orchestration system that maintains dependency-linked execution state, performs dependency-ordered recomputation upon a state change, and propagates state across partitions is operated by the distributed graph runtime engine independently of a storage or messaging substrate of the orchestration system.

Recursive Revocation and Recovery Engine

215. The system of claim 12, further comprising a recursive revocation and recovery engine that, upon revocation of an authority, propagates the revocation through an affected graph topology and suspends dependent execution.
216. The system of claim 215, wherein revocation of a capability triggers recursive capability invalidation by which the recursive revocation and recovery engine traverses a delegation tree and dependency edges rooted at the revoked capability and invalidates capabilities and execution states transitively derived from the revoked capability.
217. The system of claim 216, wherein an invalidated execution state is transitioned to a suspended state such that descendant execution dependent upon the revoked capability is suspended.
218. The system of claim 215, wherein the recursive revocation and recovery engine applies a partial graph freeze to an affected subgraph against execution-advancing mutations while the revocation is processed.
219. The system of claim 215, wherein the recursive revocation and recovery engine distinguishes an execution branch dependent upon a revoked authority from an independently authorized execution branch whose authority derives from a chain not passing through the revoked authority, and preserves the independently authorized execution branch.
220. The system of claim 219, wherein an effect of the revocation is confined to a dependency cone of the revoked authority.
221. The system of claim 215, wherein the recursive revocation and recovery engine performs a lineage-preserving rollback that returns an affected subgraph to a recovery checkpoint while retaining provenance records of undone mutations.
222. The system of claim 221, wherein the recursive revocation and recovery engine performs dependency-linked recovery by restoring node states in a recovery order that respects dependency edges such that no node is recovered before a node upon which it depends.
223. The system of claim 215, wherein a revoked capability token is recorded in a revocation set, and an execution layer rejects presentation of a capability token in the revocation set, thereby preventing replay of a revoked authority.

224. The system of claim 215, wherein the recursive revocation and recovery engine produces a cryptographic revocation attestation attesting a revoked authority, a state-version frontier at which the revocation occurred, and an affected subgraph.
225. The system of claim 215, wherein revocation propagates asynchronously across a plurality of partitions and is applied by partition-local instances of the recursive revocation and recovery engine.
226. The system of claim 225, wherein a revocation set is reconciled across the plurality of partitions by a partition-synchronization operation such that an authority revoked in a first partition cannot be exercised in a second partition.
227. The system of claim 215, wherein a revocation-and-recovery operation progresses through a state machine comprising an initiated state, a propagating state, a frozen state, a rolling-back state, a recovering state, and a converged state.
228. The system of claim 227, wherein a cryptographic revocation attestation is emitted upon entry of the revocation-and-recovery operation into the converged state.

Federated Synchronization and Reconciliation

229. The system of claim 13, further comprising federated synchronization and reconciliation machinery that reconciles partition-local graph state into a converged composite state across a plurality of partitions.
230. The system of claim 229, wherein the federated synchronization and reconciliation machinery operates on an eventual-consistency model in which partitions are permitted to diverge and converge to a consistent composite state through repeated synchronization.
231. The system of claim 229, wherein reconciliation of partition-local views is performed by a lineage-preserving merge that computes a composite view retaining provenance of each contributing partition-local version.
232. The system of claim 231, wherein the lineage-preserving merge, upon a conflict among partition-local versions, retains provenance of a non-selected partition-local version.
233. The system of claim 229, wherein a conflict among partition-local views is resolved by trust-weighted reconciliation in which a partition-local version asserted by a partition carrying a higher trust weight controls.
234. The system of claim 229, wherein a conflict among partition-local views is resolved by quorum reconciliation in which a partition-local version asserted by at least a threshold number of partitions controls.
235. The system of claim 229, wherein a conflict among partition-local views is resolved by jurisdiction-priority arbitration in which a partition-local version governed by a higher-precedence jurisdiction controls.
236. The system of claim 229, wherein multiple conflicts arising in a synchronization are resolved in a deterministic conflict order such that a synchronization result is reproducible.
237. The system of claim 229, wherein the federated synchronization and reconciliation machinery records a synchronization checkpoint comprising a consistent capture of the converged composite view at a synchronization frontier.
238. The system of claim 229, wherein the converged composite view records, for each node and edge, provenance establishing how the node or edge came to a converged state.
239. The system of claim 229, wherein synchronization proceeds under partial visibility such that the federated synchronization and reconciliation machinery reconciles a mutually-visible

portion of a graph without requiring disclosure of a portion a partition is not authorized to observe.

240. The system of claim 229, wherein a partition whose contributed partition-local view fails provenance verification is excluded from a synchronization round and the exclusion is recorded.
241. The system of claim 229, wherein the federated synchronization and reconciliation machinery emits a synchronization attestation upon completion of a synchronization round.
242. The system of claim 229, wherein a conflict that cannot be deterministically resolved is recorded and escalated for governance disposition.

Autonomous Governance-Constrained Agent Orchestration

243. The system of claim 12, further comprising autonomous governance-constrained agent orchestration machinery within which an autonomous orchestration agent executes a multi-step workflow under a bounded capability envelope.
244. The system of claim 243, wherein the bounded capability envelope is a governed data structure defining an action scope, a propagation scope, a jurisdictional limit, a revocation-inheritance rule, a lineage-retention requirement, and an expiration condition.
245. The system of claim 243, wherein an executable-action graph of the autonomous orchestration agent is pruned to the bounded capability envelope such that an action outside the bounded capability envelope is not materialized and is computationally non-instantiable.
246. The system of claim 243, wherein the autonomous orchestration agent delegates a portion of the bounded capability envelope to a further agent by issuing a delegated envelope strictly attenuated relative to the bounded capability envelope.
247. The system of claim 246, wherein the delegated envelope is no broader in action scope, propagation scope, or jurisdictional reach, and no later in expiration, than the bounded capability envelope.
248. The system of claim 246, wherein delegation of envelopes is recursive and forms a delegation tree bounded by a recursive delegation constraint.
249. The system of claim 243, wherein the autonomous governance-constrained agent orchestration machinery evaluates a workflow step against one or more Constitutional Guardrails, Trust Criteria, and the bounded capability envelope, and advances the workflow step only if the workflow step is permitted.
250. The system of claim 243, wherein a policy mutation triggers re-evaluation of an in-flight workflow, and a workflow step rendered impermissible by the policy mutation triggers a policy-triggered execution suspension of the workflow.
251. The system of claim 243, wherein the autonomous governance-constrained agent orchestration machinery performs trust-weighted routing of a workflow step to an agent selected as a function of agent trust weights and governance compatibility.
252. The system of claim 243, wherein a workflow step above a defined sensitivity threshold is effectuated only upon concurring execution authorization from a quorum of agents.
253. The system of claim 243, wherein an action effectuated by the autonomous orchestration agent produces a lineage-bound execution record bound into a lineage graph and attesting the action, the bounded capability envelope under which the action was effectuated, the agent, and a state-version frontier.
254. The system of claim 253, wherein a chain of lineage-bound execution records for a workflow constitutes a cryptographically attested execution lineage that is independently verifiable.

255. The system of claim 243, wherein a governance event triggers governance-triggered recomputation of an affected workflow state.
256. The system of claim 243, wherein a governance event invalidating an already-effectuated workflow step triggers an autonomous execution rollback returning the workflow to a consistent recoverable state.
257. The system of claim 243, wherein the autonomous governance-constrained agent orchestration machinery provides a human override pathway by which a designated human authority suspends, redirects, or terminates the workflow, the human override being a governed and recorded operation.
258. The system of claim 243, wherein an autonomously orchestrated workflow progresses through a state machine comprising an admitted state, an executing state, a suspended state, a rolling-back state, a recomputing state, a completed state, and a terminated state.
259. The system of claim 243, wherein an orchestration of workflow steps across a plurality of partitions remains subject to the bounded capability envelope independently of a coordination channel by which the workflow steps are coordinated.
260. The system of claim 243, wherein the autonomous orchestration agent is a governed artificial intelligence system whose operational authority is resource-bound and whose executable action-space topology is constrained prior to execution.
261. The system of claim 1, wherein the recursive coordination substrate operates as a distributed computational system for recursively governed authorization, orchestration, synchronization, and settlement computed over one or more graph data structures and executed by one or more runtime engines.
262. The system of claim 1, wherein governance, authorization, settlement, and orchestration operations are machine-executed as graph-runtime operations, synchronization operations, and cryptographic state-enforcement operations.
263. The method of claim 2, further comprising executing graph mutations in a dependency-ordered execution order, performing selective downstream recomputation upon a state change, and propagating state across a plurality of partitions by asynchronous synchronization.
264. The method of claim 2, further comprising propagating a revocation recursively through an affected graph topology, suspending dependent execution, and preserving an independently authorized execution branch.
265. The method of claim 2, further comprising reconciling partition-local graph state into a converged composite state by a lineage-preserving merge that preserves independently verifiable execution provenance.