SYSTEMS AND METHODS FOR AUTONOMOUS AGENTS WITH PERSISTENT COGNITIVE STATE, SELF-REGULATED EXECUTION, AND CROSS-DOMAIN BEHAVIORAL COHERENCE

RELATED APPLICATION DATA

[0001] This application claims the benefit of priority of U.S. Provisional Patent Application Serial No. 63/789,967, filed on April 16, 2025, titled "AQx – Cross-Domain Applications of the Adaptive Query Framework"; U.S. Provisional Patent Application Serial No. 63/808,539, filed on May 19, 2025, titled "Cognition-Native Semantic Execution Platform with Emotionally Modulated, Slope-Validated Agents"; U.S. Provisional Patent Application Serial No. 63/810,666, filed on May 22, 2025, titled "Cognition-Native Semantic Execution Platform with Trait-Based Planning Modulation, Slope-Constrained Mutation, and Executive Graph Aggregation for Memory-Bearing Semantic Agents"; U.S. Provisional Patent Application Serial No. 63/827,301, filed on June 20, 2025, titled "Cognition-Native Semantic Execution Platform Incorporating Empathy-Weighted Integrity Modeling, Deviation-Based Mutation Control, and Deterministic Moral Trajectory Forecasting for Ethically Evolving AI Agents"; U.S. Provisional Patent Application Serial No. 63/926,122, filed on November 26, 2025, titled "Systems and Methods for Large-Language-Model-Driven Mutation, Evaluation, and Gating of Persistent Semantic Agents"; U.S. Provisional Patent Application Serial No. 63/962,013, filed on January 16, 2026, titled "Systems and Methods for Capability-, Time-, and Uncertainty-Aware Execution in Autonomous Computational Networks"; U.S. Provisional Patent Application Serial No. 63/964,715, filed on January 21, 2026, titled "Confidence-Governed Execution for Cognition-Native Semantic Agents"; U.S. Provisional Patent Application Serial No. 63/957,729, filed on January 10, 2026, titled "Continuity-Based Biological Identity Using Trust-Slope Validation"; and U.S. Provisional Patent Application Serial No. 63/978,324, filed on February 9, 2026, titled "Inference-Time Semantic Execution Control for Machine Learning Systems," each of which is incorporated by reference herein in its entirety.

FIELD

[0002] The present disclosure generally relates to artificial intelligence, cognitive systems architecture, and autonomous agent platforms. In particular, the present disclosure is directed to

systems and methods for autonomous agents with persistent cognitive state, self-regulated execution, and cross-domain behavioral coherence.

BACKGROUND

[0003] Conventional artificial intelligence systems operate as stateless inference engines that accept inputs, produce outputs, and retain no persistent identity, memory of prior reasoning, or capacity for self-regulation across time. Such systems cannot maintain behavioral consistency across interactions, cannot modulate their own execution based on internally computed state, and cannot determine from internal conditions alone when they should or should not act.

[0004] Agent architectures including belief-desire-intention frameworks, reinforcement learning from human feedback, and safety wrapper systems address subsets of these deficiencies. However, no existing architecture provides an agent that maintains a plurality of persistent, independently tracked cognitive domain fields coupled through bidirectional feedback pathways, self-regulates execution through an internally computed composite readiness assessment, and transitions between executing and non-executing cognitive modes based on that assessment while continuing speculative reasoning.

[0005] Accordingly, there is a need for systems and methods that address these shortcomings.

SUMMARY OF THE DISCLOSURE

[0006] In accordance with one aspect of the present disclosure, a system for autonomous agents with persistent cognitive state and self-regulated execution is provided that includes one or more processors and one or more non-transitory computer-readable media storing instructions that, when executed by the one or more processors, cause the system to maintain a plurality of semantic agents, each semantic agent comprising a plurality of persistent cognitive domain fields and a lineage field, the cognitive domain fields collectively encoding a behavioral disposition, a normative alignment, and an execution readiness as continuously updated persistent state, wherein each cognitive domain field is independently tracked by a cross-domain coherence engine with a current value and a trajectory over time, and wherein the semantic agent carries a complete cognitive state such that an execution substrate hosting the semantic agent validates proposed state transitions without retaining authority over the semantic agent's cognitive state,

operate the cross-domain coherence engine to maintain bidirectional feedback pathways between the cognitive domain fields, such that a state change in any one cognitive domain field propagates deterministic updates to at least one other cognitive domain field through a defined coupling function, and wherein the cross-domain coherence engine enforces that no cognitive domain field is updated in isolation from the feedback pathways, evaluate, for each proposed mutation to a semantic agent, a composite admissibility determination that integrates signals from a plurality of the cognitive domain fields through the cross-domain coherence engine, and selectively permit, gate, or suspend the proposed mutation based on the composite admissibility determination, transition the semantic agent to a non-executing cognitive mode when the composite admissibility determination indicates insufficient execution readiness, wherein in the non-executing cognitive mode the semantic agent continues speculative reasoning and state evaluation without committing state changes to verified agent state and record each proposed mutation, each composite admissibility determination, and each cognitive domain field update in the lineage field such that the complete behavioral trajectory of the semantic agent is deterministically reconstructible from the lineage field alone.

[0007] In accordance with another aspect of the present disclosure, a computer-implemented method for governing execution of a semantic agent through cross-domain cognitive coherence includes maintaining the semantic agent with a persistent state, the persistent state comprising a plurality of cognitive domain fields each independently tracked by a cross-domain coherence engine and coupled through bidirectional feedback pathways, and a lineage field recording a complete behavioral history, wherein the semantic agent carries the persistent state such that the semantic agent is migratable between execution substrates while preserving behavioral continuity, receiving a proposed mutation to the semantic agent, propagating the proposed mutation through a cross-domain coherence engine, computing, via the cross-domain coherence engine, for each cognitive domain field, an independent contribution to a composite evaluation of the proposed mutation, propagating responsive updates between cognitive domain fields through the bidirectional feedback pathways, determining, based on the composite evaluation, whether to permit the proposed mutation, gate the proposed mutation pending additional evaluation, or suspend execution of the semantic agent into a non-executing cognitive mode in which speculative reasoning continues without committing state changes, when the semantic agent is in the non-executing cognitive mode, generating candidate alternative mutations through

speculative evaluation within the cross-domain coherence engine and evaluating each candidate against the composite admissibility criteria until a candidate satisfying the composite admissibility criteria is identified or an external intervention is received, and recording the proposed mutation, the composite evaluation, all cognitive domain field updates, and any non-executing cognitive mode transitions in the lineage field.

[0008] In accordance with yet another aspect of the present disclosure, a non-transitory computer-readable medium storing instructions that, when executed by one or more processors, cause the one or more processors to maintain a semantic agent comprising a plurality of persistent cognitive domain fields coupled through a cross-domain coherence engine implementing bidirectional feedback pathways, and a lineage field, wherein the plurality of persistent cognitive domain fields and the lineage field collectively define a behavioral disposition for the semantic agent, and wherein the semantic agent carries a complete cognitive state including the cross-domain coherence engine such that an execution substrate provides computational resources without retaining authority over the semantic agent's state transitions, detect, through the cross-domain coherence engine, when a state of the semantic agent in any cognitive domain field deviates from a normative alignment defined by one or more policy constraints applicable to that cognitive domain field, in response to detecting the deviation, propagate corrective pressure from the deviating cognitive domain field through the bidirectional feedback pathways to at least one other cognitive domain field, thereby modulating the semantic agent's behavioral disposition across coupled domains in response to the deviation, generate, through corrective pressure propagated through the cross-domain coherence engine, a candidate mutation designed to restore normative alignment in the deviating cognitive domain field, and evaluate the candidate mutation against the composite admissibility criteria of all coupled cognitive domain fields before permitting execution, and operate the semantic agent in a degraded mode when fewer than all cognitive domain fields are available, preserving deterministic behavioral governance through a subset of available cognitive domain fields and the bidirectional feedback pathways active between the available cognitive domain fields.

BRIEF DESCRIPTION OF THE DRAWINGS

[0009] For the purpose of illustrating the disclosure, the drawings show aspects of one or more embodiments of the disclosure. However, it should be understood that the present

disclosure is not limited to the precise arrangements and instrumentalities shown in the drawings, wherein:

FIG. 1 illustrates a feedback loop in which a persistent affective state field modulates confidence, forecasting, and integrity domains, and execution outcomes feed back as structured observations that update the affective state field in accordance with an embodiment of the disclosure;

FIG. 2 illustrates an integrity engine that reads agent behavior and writes scores to three independent domains within an integrity field, with a weighting function producing a composite score that feeds back to modulate behavior in accordance with an embodiment of the disclosure;

FIG. 3 illustrates a deviation function pipeline computing deviation likelihood as the ratio of deviation pressure to deviation resistance, with need vector and ethical threshold as numerator inputs and empathy and self-esteem as denominator inputs in accordance with an embodiment of the disclosure;

FIG. 4 illustrates a three-phase corrective loop in which a deviation event triggers empathy registration, integrity recording, and self-esteem-driven corrective pressure that produces restorative mutations leading to behavioral realignment in accordance with an embodiment of the disclosure;

FIG. 5 illustrates a speculative zone containing a forecasting engine and planning graphs separated from verified execution memory by a promotion interface that governs which speculative results may be committed in accordance with an embodiment of the disclosure;

FIG. 6 illustrates a confidence governor that compares a computed confidence value against a threshold and routes execution to either an authorized execution path or a non-executing cognitive mode with a reauthorization gate providing a recovery feedback path in accordance with an embodiment of the disclosure;

FIG. 7 illustrates a capability determination in which an objective is evaluated along two independent paths through a capability envelope and a governance policy that converge at a joint evaluation gate producing execution synthesis, non-synthesis, or deferred outcomes in accordance with an embodiment of the disclosure;

FIG. 8 illustrates a unidirectional interface in which a language model confined to a bounded proposal zone generates candidate mutations that pass through a validation engine to reach agent verified state with no return path in accordance with an embodiment of the disclosure;

FIG. 9 illustrates an inference loop in which each candidate transition passes through a mutation

mapping module and an admissibility gate before updating a semantic state object, with the semantic state object feeding back to condition subsequent candidate transitions in accordance with an embodiment of the disclosure;

FIG. 10 illustrates a biological identity pipeline in which signal acquisition, feature extraction, stable sketching, and biological hash generation feed a trust-slope validator that establishes identity through behavioral continuity without stored biometric templates in accordance with an embodiment of the disclosure;

FIG. 11 illustrates a governed discovery traversal in which a discovery object arrives at an anchor, undergoes a three-in-one step comprising search, inference, and governance, and advances to a next anchor in a repeating loop in accordance with an embodiment of the disclosure;

FIG. 12 illustrates depth-selective training governance in which a semantic substrate evaluates a training batch and a depth profile router directs gradient contributions to shallow, middle, or deep model layers based on the semantic entropy of each training example in accordance with an embodiment of the disclosure;

FIG. 13 illustrates a disruption model in which a coherence loop comprising empathy, integrity, and restoration phases may be exited at each phase to produce distinct stable disrupted configurations in accordance with an embodiment of the disclosure;

FIG. 14 illustrates multi-domain application parameterization in which a common set of platform primitives passes through domain-specific parameterization to produce autonomous vehicle, defense system, companion ai, and therapeutic agent deployments in accordance with an embodiment of the disclosure; and

FIG. 15 illustrates a complete platform lifecycle in which a discovery index serves training governance, inference control, and an llm proposer, which feed through a capability substrate into self-regulating cognitive fields coupled by coherence loops, producing biological continuity, skill unlocking, and disruption modeling that converge in application domains with feedback to the coherence loops in accordance with an embodiment of the disclosure.

DETAILED DESCRIPTION

[0010] The present disclosure builds upon a cognition-native semantic execution platform whose foundational components are disclosed in the following co-pending applications, each of which is incorporated by reference herein in its entirety:

[0011] U.S. Patent Application Serial No. 19/230,933, filed June 6, 2025, titled "Cognition-Native Semantic Execution Platform for Distributed, Stateful, and Ethically-Constrained Agent Systems" (hereinafter "Platform Application"). U.S. Patent Application Serial No. 19/452,651, filed January 19, 2026, titled "Cognition-Compatible Semantic Agent Objects with Structural Validation, Partial Agent Support, and Traceable Semantic Lineage" (hereinafter "Schema Application"). U.S. Patent Application Serial No. 19/538,221, filed February 12, 2026, titled "Memory-Resident Execution of Persistent Executable Objects in Distributed Computing Systems" (hereinafter "Execution Application"). U.S. Patent Application Serial No. 19/366,760, filed October 23, 2025, titled "Cognition-Compatible Network Substrate and Memory-Native Protocol Stack" (hereinafter "Protocol Application"). U.S. Patent Application Serial No. 19/326,036, filed September 11, 2025, titled "Adaptive Network Framework for Modular, Dynamic, and Decentralized Systems" (hereinafter "Index Application"). U.S. Patent Application Serial No. 19/388,580, filed November 13, 2025, titled "Systems and Methods for Memory-Native Identity and Authentication" (hereinafter "Identity Application"). U.S. Patent Application Serial No. 19/561,229, filed March 9, 2026, titled "Cryptographically Enforced Governance for Autonomous Agents and Distributed Execution Environments" (hereinafter "Governance Application").

[0012] The present disclosure introduces a plurality of cognitive domain fields into the semantic agent architecture disclosed in the above-referenced applications and couples those fields through bidirectional feedback pathways such that a change in any one domain propagates deterministic updates to related domains. The resulting agents maintain persistent cognitive state across interactions, self-regulate their own execution based on internally computed conditions, and exhibit behavioral dynamics that are structurally isomorphic to the dynamics observed in human cognition.

[0013] The detailed description is organized as follows. Chapters 2 through 12 each disclose a distinct cognitive domain. Chapter 13 discloses application embodiments across a plurality of domains. Chapter 14 discloses how the coupling of cognitive domain fields through bidirectional

feedback pathways produces the platform-level behavioral dynamics. Chapter 15 provides terminology definitions. The systems and methods disclosed in each chapter may be practiced independently or in combination.

2.1 Introduction of the Affective State Field

[0014] In accordance with an embodiment of the present disclosure, a persistent affective state field (100) is introduced as a structural component of the semantic agent schema. The affective state field encodes a structured modulation vector comprising a plurality of named control fields — including at least an uncertainty sensitivity field, a risk sensitivity field, a novelty appetite field, and a persistence-under-partial-failure field — each independently tracked with a current magnitude, a decay rate, and policy-defined range bounds. The affective state field does not encode subjective emotion; it encodes deterministic modulation parameters derived from the cumulative outcomes of prior execution that shape how the agent evaluates candidates, tolerates ambiguity, and persists under failure. Each update to the affective state field is a deterministic function of the agent's current state and structured observations from the execution environment, subject to policy-imposed rate limits and range bounds, and recorded in the agent's lineage.

2.1a Structural Properties of the Affective State Architecture

[0015] The disclosed affective state architecture is distinguished from conventional emotional models — including dimensional affect representations, appraisal-based emotion systems, and reinforcement-derived mood parameters — by four structural properties. First, the architecture requires a persistent named-control-field modulation vector in which each control field (uncertainty sensitivity, risk sensitivity, novelty appetite, persistence-under-partial-failure) independently modulates specific computational parameters in the agent's deliberation pipeline through governed update rules. Second, the architecture enforces semantic hysteresis through asymmetric update rules in which negative valence updates apply at a higher rate than positive valence updates, producing a deterministic caution bias that cannot be replicated by symmetric update functions. Third, each named control field is governed by an exponential decay curve with a configurable time constant that returns the field toward a policy-defined baseline, and entropy-governed valence stabilization progressively increases the effective decay time constant

when rapid alternation is detected, damping oscillatory behavior through a structural mechanism rather than through ad hoc smoothing. Fourth, volatility conditions trigger an emotional quarantine state that restricts the agent's operational scope until affective state stabilizes within governable bounds, providing a structural circuit-breaker absent from conventional systems that permit unbounded affective drift. These four properties — named-field modulation, asymmetric hysteresis, entropy-governed decay stabilization, and quarantine-based circuit-breaking — are structurally interdependent: removing any one produces a system that either oscillates without bound, drifts without recovery, or lacks the governed modulation pathway that connects affective state to deliberation parameters.

2.2 Affective Modulation of Deliberation Parameters

[0016] In accordance with an embodiment, the affective state field (100) modulates specific computational parameters within the agent's deliberation pipeline. The modulation targets comprise at least: promotion thresholds governing the minimum score required for a candidate mutation to advance between evaluation stages; search breadth governing the number of alternatives explored at each decision point; branch growth rates governing the rate at which new speculative branches are generated during forecasting operations; and escalation thresholds governing the conditions under which the agent transitions from independent operation to delegation or help-seeking. Elevated risk sensitivity raises promotion thresholds and lowers escalation thresholds; elevated novelty appetite increases search breadth and branch growth rates. This modulation is bounded by policy: the affective state field cannot create authority the agent does not possess, cannot bypass governance constraints, and cannot drive any modulation target outside its policy-defined operating envelope.

2.3 Cross-Domain Coupling

[0017] Referring to FIG. 1, the affective state field (100) is depicted as a closed-loop modulation architecture in which accumulated experience shapes ongoing cognition. The affective state field (100) sits at the top of the loop as the persistent modulation source and serves as a cross-domain input to three downstream domains. Three outgoing pathways show the affective state field (100) modulating the confidence domain (106), the forecasting domain (108), and the integrity domain (110). Within the confidence domain (106), the confidence decay rate is

multiplied by a factor derived from the agent's uncertainty sensitivity and risk sensitivity: elevated values cause confidence to decay faster, producing earlier transitions to non-executing cognitive mode, while the confidence recovery rate is modulated by persistence-under-partial-failure, permitting faster return to executing mode when elevated. Within the forecasting domain (108), the forecasting engine reads the agent's current affective state when initializing planning graph generation: novelty appetite modulates branching factor, risk sensitivity modulates pruning criteria, and persistence-under-partial-failure modulates projection depth. Within the integrity domain (110), elevated risk sensitivity raises the sensitivity of integrity deviation detection. These three modulated domains produce execution outcomes (104) — the observable results of the agent's governed behavior in the world — which in turn generate structured observations (112) capturing repeated failure patterns, competing objectives, time pressure, novelty exposure, and execution success. The structured observations (112) feed back into the affective state field (100), closing the loop: negative outcomes shift control fields toward caution, positive outcomes shift them toward willingness, and the asymmetric update rules described in Section 2.4 ensure that negative experience exerts a stronger and longer-lasting modulation than positive experience. This cross-domain coupling produces a feedback loop in which accumulated negative experience causes progressively more cautious execution and more conservative speculation, while accumulated positive experience causes progressively more willing execution and more expansive speculation, all within policy-defined bounds. The architectural mechanism is the loop itself — every cognitive operation the agent performs is conditioned by affect, every outcome updates affect, and no pathway exists to bypass the modulation or break the feedback cycle. Arrows in FIG. 1 indicate modulation and feedback flows rather than hardware connections.

2.4 Affective Update Mechanics and Governance

[0018] In accordance with an embodiment, the affective state update function operates on structured observations derived from the agent's execution environment, including repeated failure patterns, competing objectives, time pressure, novelty exposure, and execution success patterns. Each named control field is governed by an emotional decay curve implemented as an exponential decay with a configurable time constant: V(t) = V_baseline + (V_current - V_baseline) * exp(-t / tau). The modulation layer exhibits semantic hysteresis through

asymmetric update rules in which negative valence updates apply at a higher rate than positive valence updates, producing a built-in caution bias. Entropy-governed valence stabilization damps oscillatory behavior by progressively increasing the effective decay time constant when rapid alternation is detected. Every update is policy-bounded through range bounds, rate limits, admissible trigger sets, and update authority constraints. Volatility conditions trigger an emotional quarantine state that restricts the agent's operational scope until affective state stabilizes within governable bounds.

3.1 Integrity Field and Three-Domain Model

[0019] Referring to FIG. 2, the integrity engine architecture is depicted as a governed feedback loop connecting agent behavior to a composite integrity assessment. In accordance with an embodiment of the present disclosure, an integrity field (210) is introduced as a structural component of the semantic agent's operational state. Agent behavior (200) — the stream of actions, commitments, and mutations the agent produces — enters an integrity engine (202), operating as a first-class cognitive operation within the agent, which decomposes each behavioral event into impact projections across three independently scored domains within the integrity field (210): personal integrity (204) measuring alignment with the agent's own declared values, interpersonal integrity (206) measuring consistency with relational commitments to other agents and human operators, and global integrity (208) measuring alignment with systemic and community-level norms. Each domain maintains its own current score, trajectory, baseline, and policy-defined bounds. A weighting function (212) combines the three domain scores into a composite score (214) through policy-specified weights that vary by evaluation context — a compliance evaluation may weight global integrity (208) heavily, while a mentorship evaluation may weight interpersonal integrity (206) more strongly. The composite score (214) feeds back to modulate agent behavior (200), closing the loop: sustained integrity degradation in any domain produces progressively stronger behavioral correction pressure, while sustained integrity improvement permits broader operational latitude. Every change to the integrity field is recorded in the agent's lineage. The architectural mechanism is the integrity engine (202) itself — every behavioral event must pass through this single evaluation point, and no pathway exists for the agent to act without its behavior being decomposed, scored, and fed back through the composite

assessment. Arrows in FIG. 2 indicate evaluation and feedback flows rather than hardware connections.

3.2 The Deviation Function

[0020] In accordance with an embodiment, the system computes a deviation likelihood (314) using a deterministic composite function:

$$D = (N(t) - T(t)) / (E(t) \times S(t))$$

where N(t) represents the agent's current need vector (300) encoding unmet requirements; T(t) represents the ethical threshold (302) derived from the agent's policy configuration and declared values; E(t) represents the empathy scalar (304) encoding the degree to which the agent registers projected harm to others; and S(t) represents the self-esteem scalar (306) encoding the agent's self-assessed alignment with declared values. The numerator (N-T) represents deviation pressure (308) — the gap between unmet needs and the normative threshold. The denominator (E×S) represents deviation resistance (310) — the combined counterforce of empathic consequence registration and self-regard. When D exceeds a policy-defined activation threshold, the agent enters a Deviation-Activated State in which a scoped set of normally excluded mutations becomes admissible. Deviation events are recorded in the lineage with full provenance including the specific values of N, T, E, and S at the time of activation.

[0021] Referring to FIG. 3, the deviation function (312) is depicted as a computational pipeline with the numerator inputs at the top and the denominator inputs at the bottom. The need vector (300) encodes unmet requirements that drive the agent toward deviation, while the ethical threshold (302) encodes the normative boundary the agent must cross to deviate. The difference between these two values produces deviation pressure (308) — the raw force pushing the agent toward norm-violating behavior. Below the deviation function (312), the empathy scalar (304) and the self-esteem scalar (306) combine multiplicatively to produce deviation resistance (310) — the counterforce that restrains deviation. The deviation function (312) divides pressure by resistance to produce deviation likelihood (314), a continuous scalar encoding how close the agent is to activating deviation-permitted mutations. The structural mechanism is the ratio itself: deviation cannot occur when either empathy or self-esteem is high, because the denominator overwhelms the numerator regardless of need pressure. Conversely, degradation in both empathy

and self-esteem produces a compounding collapse of resistance that no amount of ethical threshold adjustment can compensate. Arrows in FIG. 3 indicate data flows through the computation rather than hardware connections.

3.3 The Coherence Trifecta

[0022] In accordance with an embodiment, the integrity subsystem operates through a three-phase coherence control loop that activates when a deviation is detected.

[0023] Referring to FIG. 4, the coherence trifecta is depicted as a sequential three-phase loop with restorative output and behavioral feedback. The loop begins when a deviation is detected (400) — a behavioral event that crosses a normative threshold in any integrity domain. In Phase 1, empathy registers impact (402): the empathy weighting engine computes projected harm across affected entities and integrity domains, generating deviation pressure that quantifies the normative cost of the contemplated or executed action. In Phase 2, integrity records as truth (404): the integrity engine commits the detected deviation to the agent's lineage as immutable record, without minimization, externalization, or denial, including the deviation magnitude, causal antecedents, and projected consequences. In Phase 3, self-esteem generates pressure (406): the self-esteem update function generates an internal corrective force proportional to the discrepancy between the agent's behavioral record and declared values. This corrective force activates the generation of restorative mutations (408) — candidate behavioral changes designed to repair the integrity damage recorded in Phase 2. Successful restorative mutations produce behavioral realignment (412), which feeds back to reduce future deviation by raising the agent's deviation resistance through improved self-esteem and reinforced empathy. The structural mechanism is the sequential dependency of the three phases: Phase 2 cannot record what Phase 1 has not registered, and Phase 3 cannot generate corrective pressure for what Phase 2 has not recorded. Disruption at any phase breaks the entire corrective loop — suppressing empathy prevents harm registration, corrupting integrity recording prevents truthful self-assessment, and degrading self-esteem prevents the generation of restorative pressure. Arrows in FIG. 4 indicate process flows through the corrective loop rather than hardware connections.

4.1 Planning Graphs and the Containment Layer

[0024] In accordance with an embodiment of the present disclosure, a planning graph (502) is introduced as a first-class cognitive structure within the semantic agent architecture. A planning graph is a mutable, directed semantic structure comprising a root node representing the agent's current verified state and a plurality of branches, each representing a distinct hypothetical trajectory — a sequence of speculative mutations that the agent is evaluating as possible futures. Planning graphs exist in a structurally distinct computational domain from the agent's verified execution memory (508), enforced by a containment layer that prevents speculative state from contaminating verified state. No speculative branch may alter verified agent state without passing through a promotion interface (506) that subjects the branch to the full governance evaluation applicable to committed mutations — including integrity impact projection, confidence assessment, and policy compliance verification. The containment layer is an architectural enforcement, not a software flag or runtime check, that prevents speculative state from contaminating verified agent state through three mechanisms: immutable speculative markers that cannot be stripped during promotion, read isolation preventing verified-side reads of speculative content, and a governed promotion interface requiring composite governance evaluation — integrity impact projection, confidence assessment, and policy compliance verification — before any speculative branch becomes verified state. Unlike tree search methods that discard evaluated branches after action selection, the containment layer preserves the architectural separation between speculative and verified state as a structural invariant. This structural separation between speculation and execution mirrors the cognitive distinction between imagination and action. When the containment layer fails — when speculative branches are treated as verified or when verified state is contaminated by unvalidated speculation — the system detects a containment collapse condition corresponding to a delusion boundary violation.

4.2 Forecasting Engine Architecture

[0025] Referring to FIG. 5, the forecasting engine architecture is depicted with a structural containment boundary separating speculative cognition from verified execution memory. The forecasting engine (500) sits at the center, comprising five principal components: planning graph instantiation logic that creates branches from the agent's current state and objectives; an affective prioritization module that orders branches based on the agent's current affective state; a speculative simulation engine that projects consequences of each branch through deterministic

state transition modeling; a slope projection module that evaluates whether each branch maintains trust slope continuity; and a policy compatibility filter that verifies each branch against applicable governance constraints. The engine executes a six-phase cycle: initialization, simulation, slope projection, policy compatibility assessment, emotional reinforcement tagging, and branch classification. Each branch is classified as eligible (ready for promotion), introspective (requiring further evaluation), delegable (appropriate for transfer to another agent), or pruned (discarded). A personality field comprising openness, deliberativeness, impulsivity, and fallback rigidity modulates the engine's generation and evaluation parameters, and the agent's affective state further modulates planning graph expansion depth, branch prioritization, and pruning aggressiveness. The forecasting engine (500) produces planning graphs (502) — mutable directed semantic structures in which each branch represents a distinct hypothetical future the agent is evaluating. All planning graphs (502) exist within a speculative zone (504), depicted with a dashed boundary to emphasize that everything inside is provisional and unverified. No content within the speculative zone (504) can affect the agent's committed state. The only exit from the speculative zone (504) is the promotion interface (506) — a governed gate that subjects each candidate branch to the full governance evaluation applicable to committed mutations, including integrity impact projection, confidence assessment, and policy compliance verification. Branches that survive promotion pass through the promotion interface (506) and enter verified execution memory (508), where they become part of the agent's committed state and lineage. The architectural mechanism is the containment boundary itself: the speculative zone (504) structurally prevents speculative content from contaminating verified state, and the promotion interface (506) ensures that no speculative branch can bypass governance on its way to commitment. When this boundary fails — when speculative branches leak into verified state without promotion — the system detects a containment collapse condition corresponding to a delusion boundary violation. Arrows in FIG. 5 indicate the flow of speculative content from generation through evaluation to governed promotion, rather than hardware connections.

4.3 Executive Engine and Multi-Agent Aggregation

[0026] In accordance with an embodiment, when multiple agents participate in a cooperative operation, each agent maintains its own planning graph (502). An executive engine aggregates

the individual planning graphs into an executive graph — a composite structure representing the collective speculative landscape of the cooperating agents. The executive engine detects branch intersections where multiple agents' plans reference the same resources, targets, or environmental conditions; resolves conflicts through trust-slope-weighted arbitration; and identifies complementary branches that can be composed into cooperative execution plans. The executive graph maintains the same containment properties as individual planning graphs: no cooperative plan is committed without passing through each participating agent's promotion interface (506) and governance evaluation. An emotional quorum override mechanism permits the collective affective state of participating agents to modulate group-level planning parameters when a policy-defined quorum of agents exhibits concordant affective dispositions.

5.1 Execution as Revocable Permission

[0027] In accordance with an embodiment of the present disclosure, execution in the semantic agent architecture is treated as a revocable permission rather than as a default operational state. A confidence governor continuously evaluates whether conditions for execution remain satisfied and withdraws execution authorization when those conditions degrade. The confidence governor is a hard gate: when the confidence governor withdraws authorization, execution ceases and no alternative pathway to execution exists that bypasses this gate. The agent cannot override the withdrawal through self-assessment, affective escalation, or policy reinterpretation.

5.2 Confidence Computation and Non-Executing Cognitive Mode

[0028] Referring to FIG. 6, the confidence-governed execution architecture is depicted as a gated pipeline with a binary fork and a recovery feedback path. In accordance with an embodiment, a confidence computation (600) aggregates structured inputs — integrity sufficiency derived from the agent's current composite integrity score, affective modulation derived from the agent's current risk sensitivity and uncertainty sensitivity, capability sufficiency derived from the agent's substrate-advertised resource state, and task-specific assessment derived from the agent's forecasting engine output — into a continuous confidence value encoding the agent's assessed sufficiency to continue executing. The confidence value is also evaluated as a rate of change: the confidence derivative detects whether confidence is stable, improving, or

degrading, enabling the governor to anticipate threshold crossings and activate preemptive interventions before execution authorization is withdrawn. This value enters the confidence governor (602), which maintains the execution authorization threshold and applies hysteresis to prevent oscillatory transitions. The confidence governor (602) feeds the threshold comparison (604), which evaluates the current confidence value against the authorization threshold and produces a binary determination. When the confidence value meets or exceeds the threshold, the threshold comparison (604) routes the agent to execution authorized (606), permitting the agent to commit state changes to verified agent state. When the confidence value falls below the threshold, the threshold comparison (604) routes the agent to non-executing cognitive mode (608), in which the agent continues speculative reasoning, planning graph construction, inquiry generation, and delegation evaluation without committing state changes. The non-executing cognitive mode (608) is structurally distinct from both execution and failure — the agent has not failed but has determined that conditions are insufficient for committed action. The three task classes — terminal, exploratory, and generative — receive differentiated interruption protocols: terminal tasks preserve state through checkpointing; exploratory tasks expand the search space through hypothesis generation; and generative tasks transition to low-commitment creative exploration.

5.3 Recovery and Reauthorization

[0029] In accordance with an embodiment, recovery from non-executing cognitive mode (608) proceeds through a three-phase protocol via the reauthorization gate (610). Phase 1 — confidence restoration: the agent executes inquiry operations, gathers additional information, resolves adverse conditions, and recalculates the confidence value from updated inputs. Phase 2 — stability verification: the reauthorization gate (610) requires the confidence value to remain above the authorization threshold for a policy-defined stability window with a hysteresis margin that prevents oscillatory transitions between executing and non-executing modes. Phase 3 — reauthorization: upon verified stability, the reauthorization gate (610) feeds back to the confidence governor (602), closing the loop, and the agent transitions from non-executing cognitive mode back to executing mode. The architectural mechanism is the threshold comparison (604): every execution request must pass through this single binary gate, and no

alternative pathway to execution exists that bypasses it. Arrows in FIG. 6 indicate decision and feedback flows rather than hardware connections.

6.1 Capability Envelope as Substrate-Advertised Constraint

[0030] In accordance with an embodiment of the present disclosure, capability is a first-class computational state variable that is structurally independent of both confidence and authorization. Capability, as used herein, is a computed determination describing whether an executable form of a given objective can exist on a given execution substrate, resolved from the substrate's structural characteristics, the objective's requirements, and the current execution environment state. The determination resolves to one of four outcomes: structurally possible, structurally impossible, structurally deferred, or rerouted to an alternative substrate. Each outcome is a valid computational result. Capability answers the question "can this substrate physically and architecturally execute this objective" — a categorically different question from whether the agent should execute (governed by the confidence governor described in Chapter 5) or whether the agent is permitted to execute (governed by authorization policy). The system maintains capability envelopes (702) and governance policies in architecturally separate subsystems with no bidirectional dependency: the capability envelope subsystem does not consult governance policies when computing capability, and the governance policy subsystem (706) does not consult capability envelopes when evaluating permission. These independent determinations converge only at the execution synthesis gate (708), where both must be satisfied for execution to proceed. When the joint condition is not satisfied, the system produces a non-synthesis determination (710) classified as structurally impossible, structurally deferred (710a) when the condition may be satisfied at a future time, or rerouted to an alternative substrate.

6.2 Capability Envelope Structure and Temporal Executability

[0031] In accordance with an embodiment, each execution substrate advertises a capability envelope — a structured, dynamic data object describing the substrate's current affordances along defined dimensions including compute class, memory architecture, model access, locality, execution guarantees, and sensor-actuator interfaces. The capability envelope is not a static registry; it is updated in response to hardware provisioning, model deployment, resource consumption changes, and environmental shifts. Capability requirements extracted from the

agent's objective are matched dimension-by-dimension against the substrate's envelope, producing per-dimension outcomes of satisfied, unsatisfied, or conditionally satisfiable. A temporal executability forecasting subsystem projects the substrate's capability trajectory over a forecast horizon, identifying bounded time windows during which the capability-time intersection required for execution is expected to exist. Temporal forecasts carry confidence-bounded window estimates rather than point predictions, and are continuously updated as substrate conditions change. The system jointly evaluates capability, temporal executability, and uncertainty — a structured epistemic bound encoding the system's assessed confidence in its own evaluation — as a three-part condition that must be simultaneously satisfied before execution synthesis proceeds. Uncertainty propagates through the pipeline such that downstream decisions inherit and accumulate the uncertainty of their inputs, recorded in an uncertainty ledger persisted in the agent's lineage.

6.3 Execution Synthesis, Non-Synthesis, and Embodied Extension

[0032] Referring to FIG. 7, the capability determination architecture is depicted as a convergence gate with two independent input paths and three possible outcomes. An objective (700) — the agent's current execution target — is evaluated along two structurally independent paths. The first path evaluates the objective (700) against the capability envelope (702), which encodes the substrate's current affordances along defined dimensions including compute class, memory architecture, model access, locality, and execution guarantees, answering the question of whether the substrate can physically and architecturally execute this objective. The second path evaluates the objective (700) against the governance policy (706), which encodes whether the agent is permitted to execute this objective under current authorization constraints, answering the question of whether the agent should execute, independent of whether it can. These two independent evaluations converge at the joint evaluation gate (708), where both must be satisfied simultaneously for execution to proceed. A pre-commitment validation re-evaluates conditions before submission, preventing execution on stale capability data. When both conditions are met, the joint evaluation gate (708) routes the objective to execution synthesis (704), which constructs a capability execution plan dynamically adapted to the substrate's current capabilities and temporal window. When either condition is not met, the joint evaluation gate (708) produces a non-synthesis determination (710) — a structured record comprising the unsatisfied dimensions,

unmet temporal conditions, exceeded uncertainty thresholds, and a classification of whether non-synthesis is permanent, temporal, conditional, or indeterminate. Non-synthesis is further classified: when the unsatisfied condition may be met at a future time, the determination is classified as deferred (710a), indicating that the objective should be re-evaluated when substrate conditions change. Non-synthesis is treated as a valid computational output with the same governance rigor as execution synthesis. The capability envelope framework extends to embodied and robotic systems, where the envelope encompasses physical affordances — degrees of freedom, force capacity, reach envelope, locomotion capability, sensor modalities, and power budget — matched against motor objectives in the same formal manner as computational requirements. The framework further extends to human operators through biological capability envelopes populated via biological identity signals, enabling the same capability-native computation to govern human task assignment while maintaining strict privacy governance and separation from authorization. The architectural mechanism is the joint evaluation gate (708) itself: capability and governance are evaluated independently with no bidirectional dependency, and both must independently approve before any execution can proceed. Neither a capable substrate nor a permissive policy alone is sufficient. Arrows in FIG. 7 indicate evaluation flows converging at the gate rather than hardware connections.

7.1 Unidirectional Untrusted Proposal Interface

[0033] In accordance with an embodiment of the present disclosure, every large language model (800) integrated into the platform architecture occupies the structural role of an untrusted proposal generator confined to a bounded proposal zone. No language model output is authoritative. Every output is a candidate semantic mutation (802a) that must pass through a unidirectional interface (804) into an agent-resident validation engine (806) before it can affect any agent field, governance decision, capability gate, or external-facing behavior. There is no bypass path, no trusted-model exception, and no mechanism by which a language model can promote its own output to authoritative status. The confinement is enforced by the execution substrate architecture, not by runtime checks. A mutation engine interposes between the language model output boundary and the validation engine, performing schema mapping, bounds normalization, conflict detection, and lineage annotation to impose structural discipline on inherently unstructured model output. When multiple language models produce competing

proposals for the same agent field, an arbitration engine resolves the conflict through trust-weighted evaluation in which each model's proposal is weighted by the model's accumulated trust score — a dynamic value updated based on historical validation outcomes. Every arbitration decision is recorded as a first-class semantic event, cryptographically sealed into the agent's lineage.

7.2 Structural Starvation of Hallucination

[0034] In accordance with an embodiment, the system prevents language model hallucination through structural starvation — an architectural technique that denies the language model access to the informational resources required for hallucination to occur, rather than detecting hallucinated content after production. Five complementary constraints implement structural starvation: prompt bounding restricts the model's input to a curated context derived from the agent's verified fields; absence of external memory eliminates retrieval-augmented or externally sourced context; forced reliance on agent fields requires that proposals reference only information present in the agent's governed state, rejecting ungrounded content during schema mapping; intermediate rejection discards failing proposals without providing rejection feedback to the model, preventing adversarial optimization of the validation boundary; and stateless purging resets the model's context after each inference call, preventing multi-turn probing of the validation criteria. These five hallucination starvation constraints — prompt scope bounding, absence of external memory access, forced reliance on agent-provided fields, intermediate rejection at schema mapping, and stateless purging — operate collectively to deny the language model the informational preconditions for hallucination rather than detecting hallucinated content after production. The unidirectional interface is architectural, not procedural — no return path from the validation engine to the bounded proposal zone exists in the system architecture, ensuring that the language model cannot observe, probe, or adapt to validation criteria. This containment operates independently of the model's alignment status, composing with any model-level alignment technique while producing safe behavior through architectural constraint regardless of the model's training methodology.

7.3 Progressive Skill Gating with Mastery Evidence

[0035] In accordance with an embodiment, a curriculum engine defines structured learning progressions through which a human user, robotic operator, or autonomous agent demonstrates mastery of defined skill domains. The curriculum engine produces structured mastery evidence — evaluated against defined thresholds across multiple dimensions including accuracy, consistency, speed, generalization, and robustness — that feeds an evidence-based capability gate. The capability gate grants access based on what the requester has demonstrated rather than on credentials, roles, or static permissions. The evidence corpus comprises mastery assessments, consistency metrics, operational performance traces, multimodal evaluation data, and temporal currency metadata. The gate implements progressive unlock rules that grant graduated access matched to demonstrated competence at each operational complexity level, and regression and revocation logic that restricts or revokes capability when subsequent performance falls below maintained mastery thresholds. Cross-domain capability unification is achieved through the single persistent semantic agent, whose memory field accumulates mastery evidence across all operational domains and whose lineage records every gating decision, enabling transferable mastery evidence where competence demonstrated in one domain informs gating decisions in related domains. Referring to FIG. 8, the language model integration architecture is depicted as a strictly one-way pipeline enforcing structural untrust. A language model (800) operates inside a bounded proposal zone (802), depicted with a dashed boundary to emphasize that everything produced by the model is provisional and untrusted. The language model (800) generates candidate mutations (802a) — proposed changes to agent fields, governance decisions, or behavioral parameters — that exist only within the bounded proposal zone (802) and carry no authority. Candidate mutations (802a) exit the bounded proposal zone (802) through the unidirectional interface (804), which enforces a one-way architectural constraint: data flows from the bounded proposal zone (802) toward the agent but never returns. The unidirectional interface (804) feeds the validation engine (806), which evaluates each candidate mutation against the agent's governance constraints, schema requirements, and lineage continuity before any mutation can affect agent state. Candidate mutations that survive validation are committed to the agent verified state (808), where they become part of the agent's governed, lineage-recorded operational state. The architectural mechanism is the unidirectional interface (804): no return path exists from the validation engine (806) or the agent verified state (808) back to the bounded proposal zone (802), preventing the language model from observing, probing, or adapting to

validation criteria. The language model cannot promote its own output to authoritative status, cannot learn what was rejected or why, and cannot modify the validation rules it is subject to. Arrows in FIG. 8 indicate the one-way flow from proposal through validation to commitment, rather than hardware connections.

8.1 Semantic Admissibility Gate Within the Inference Loop

[0036] Referring to FIG. 9, the inference architecture is depicted to illustrate how semantic governance is interposed within the inference loop itself, preventing inadmissible transitions from conditioning subsequent generation steps. In accordance with an embodiment of the present disclosure, an inference engine (900) receives input and begins generating candidate outputs. Each generation step enters an inference loop (902), shown as a dashed boundary enclosing the governed evaluation cycle, operating within any probabilistic reasoning engine — whether a large language model, a small specialized model, a probabilistic graphical model, or a multimodal generative system. Within this loop, each candidate transition (904) — a proposed next token, phrase, or semantic unit — is mapped by a mutation mapping module (906) to a structured mutation descriptor specifying which fields of a semantic state object (910) the transition would modify, the proposed new values, the mutation's semantic category, and the degree of semantic change introduced. The mutation descriptor is then submitted to an admissibility gate (908), which evaluates it through four sequential stages: policy constraint verification, mutation descriptor validation, lineage continuity assessment, and entropy bounds enforcement. The gate produces a deterministic outcome — admit, reject, or decompose — given the same semantic state and proposed mutation. If admitted, the transition updates the semantic state object (910), a persistent, typed, inspectable data structure that accumulates the inference process's semantic context independently of the engine's hidden activations. The semantic state object (910) feeds back to the candidate transition stage (904), providing the accumulated semantic context against which the next candidate is evaluated. This loop-back is the structural mechanism: every generation step is conditioned not only by the engine's probability distribution but also by the governed semantic trajectory recorded in the persistent state object. This interposition within the inference loop, rather than after generation, is critical because in autoregressive models each committed token conditions all subsequent tokens — a hallucinated fact at step N propagates through all subsequent steps, shaping probability distributions in ways

that no post-generation filter can reverse. Constrained decoding operates on token-level vocabulary masks; the disclosed mechanism operates on semantic-level admissibility evaluation against a persistent state object that accumulates structured semantic context across the full inference trajectory.

8.2 Semantic State Object and Trust-Slope Continuity

[0037] In accordance with an embodiment, the semantic state object (910) maintained during inference is a structured, typed, inspectable data structure that persists across inference steps, carrying the inference process's semantic execution context independently of the inference engine's native hidden activations. The semantic state object comprises fields structurally isomorphic to the semantic agent schema: an intent field, a context field, a memory field encoding cumulative semantic commitments of admitted transitions, a policy reference field, a mutation descriptor field, a lineage field recording the ordered sequence of admitted and rejected transitions, and an entropy and uncertainty bounds field. Trust-slope continuity validation operates across the cumulative sequence of admitted transitions, computing a multi-dimensional measure of semantic drift — deviation in asserted content, epistemic certainty, and semantic register from the established trajectory. When cumulative drift exceeds configured thresholds, the mechanism produces a drift warning, a drift correction that re-anchors the inference to its original trajectory, or a drift halt that terminates inference with a partial output and a structured divergence report. Anchored semantic resolution ensures that every external reference within a candidate transition is resolved to a verified semantic referent before the transition can be committed, preventing hallucinated references from entering the inference trajectory.

8.3 Affect-Modulated Governance and Confidence-Gated Advancement

[0038] In accordance with an embodiment, the admissibility gate's quantitative parameters are modulated by the invoking agent's affective state as described in Chapter 2: elevated uncertainty sensitivity tightens entropy bounds and raises lineage continuity thresholds, while high confidence disposition relaxes bounds within policy-defined ceilings. A confidence-gating mechanism monitors the rolling admission rate during inference and transitions the process from executing mode to a non-executing inquiry mode when the rate falls below a configured threshold — mirroring the agent-level execute-to-think transition described in Chapter 5. In non-

executing inquiry mode, the process generates structured queries identifying information deficiencies, policy ambiguities, or contextual gaps causing the high rejection rate, returned to the invoking agent as a constructive output. The substrate supports multi-model inference in which multiple engines contribute candidate transitions to a shared semantic state object, governed by trust-weighted arbitration with dynamic trust-weight adjustment. Safe non-execution produces a partial output, a structured termination report, and a complete lineage record; it is treated as a valid outcome rather than an error. The substrate is deployable in embedded, co-resident, or hardware-assisted configurations, each maintaining the same semantic governance guarantees. Arrows in FIG. 9 indicate the sequential flow of evaluation within each inference step and the loop-back that carries governed semantic state across the full inference trajectory.

9.1 Identity Through Behavioral Continuity

[0039] In accordance with an embodiment of the present disclosure, a system and method for biological identity resolution defines identity as behavioral continuity over time rather than as a static credential or biometric template. Each biological observation captured by the system is evaluated as a plausible successor to a prior chain of observations through trust-slope continuity validation. Identity resides in the continuity of the observation chain itself, not in any stored template or enrolled profile. The system does not maintain an enrolled biometric reference. Instead, a trust-slope trajectory is computed from accumulated biological observations across successive interactions, and the slope of that trajectory — its rate of change, stability, and consistency across signal modalities — constitutes the identity signal. A biological entity whose trust-slope trajectory satisfies policy-defined continuity thresholds is recognized as the same entity that produced the prior observations, without reference to any fixed biometric template.

9.2 Multi-Modal Signal Processing and Trust-Slope Computation

[0040] Referring to FIG. 10, the biological identity architecture is depicted as a linear pipeline that resolves identity through behavioral continuity rather than template matching, eliminating the stored-template vulnerability present in conventional biometric systems. In accordance with an embodiment, a signal acquisition module (1000) captures biological signals across a plurality of modalities organized into acquisition tiers: a passive ambient tier capturing

behavioral signals without dedicated sensors, an active device tier capturing physiological signals through device-integrated sensors, and a dedicated biometric tier capturing high-fidelity biological measurements through purpose-built acquisition hardware, producing a raw multi-modal signal stream. A feature extraction module (1002) normalizes the raw signals into a continuity-suitable feature stream through adaptive extraction, cross-signal normalization, and noise-tolerant representation. A stable sketching module (1004) computes locality-sensitive hash representations from the normalized features, enabling privacy-preserving continuity comparison without storing raw biometric data. A biological hash module (1006) produces temporally bound hashes that encode the identity signal as a function of time-ordered behavioral observations rather than as a static credential. A trust-slope validator (1008) evaluates each incoming hash against the prior sequence of hashes, computing the slope of trust accumulation over time: a genuine entity produces a gradually rising trust slope through consistent behavioral continuity, while a spoofed or substituted entity produces discontinuities, slope reversals, or trajectory deviations detectable without reference to any stored template. Anti-spoofing is integrated into the continuity validation process itself — a spoofed signal must satisfy not only instantaneous quality checks but also trajectory-level consistency with the target entity's accumulated biological history. The pipeline's structural mechanism is the absence of stored templates anywhere in the chain: because identity resides in the continuity of the observation sequence itself, there is no enrollment database to breach, no biometric reference to steal, and no static credential to replay. Arrows in FIG. 10 indicate the unidirectional flow from raw signal acquisition through successive processing stages to trust-slope validation, with no feedback path to a stored template.

9.3 Compositional Identity Binding and Interoperability

[0041] In accordance with an embodiment, the biological identity substrate operates alongside device identity and agent identity substrates, all sharing the same architectural principle of continuity-based validation rather than static credential presentation. The three substrates are interoperable but structurally independent: a biological identity does not depend on a particular device, and an agent identity is maintained through its own governed state continuity regardless of which device hosts it or which human interacts with it. Policies may compositionally bind the substrates, requiring that a particular action be authorized by a

biological identity presenting through an attested device interacting with a continuously validated agent.

10.1 The Discovery Object as a Persistent Traversal Entity

[0042] In accordance with an embodiment of the present disclosure, semantic discovery is performed by instantiating a discovery object (1100) — a persistent traversal entity that carries structured semantic state across a sequence of governed transitions through an adaptive index. The discovery object maintains an intent field encoding the semantic query, a context block accumulating information gathered during traversal, a memory field recording the traversal history, and a policy reference field binding the traversal to governance constraints. Unlike retrieval-augmented generation systems that retrieve documents in a single operation and feed them to a generator as context, the discovery object traverses the index through governed multi-step transitions where each transition produces a governance-auditable event in the discovery object's lineage. The discovery object's cognitive domain fields — including affect-modulated exploration parameters and confidence-based termination criteria — evolve at each step, producing trajectory-dependent ranking that cannot be replicated by static retrieval scoring. The discovery object's state at any traversal step reflects the full history of its governed path through the index, enabling context-dependent search narrowing, inference, and execution decisions that are unavailable to systems that discard intermediate state between retrieval operations.

[0043] In accordance with an embodiment, the discovery object supports three distinct modes of use that emerge from the same traversal machinery. In a human search mode, a human user issues a query, and the discovery object traverses the adaptive index to produce a ranked set of results — semantically relevant anchors ordered by governed relevance evaluation rather than by static global scoring. In an agent reasoning mode, an autonomous agent instantiates a discovery object to traverse the index as part of a deliberative process, gathering information across multiple traversal steps to inform a downstream decision; the agent's own cognitive state — its affect, confidence, and policy constraints — modulates traversal behavior, producing information gathering that is structurally coupled to the agent's current reasoning context. In an answer synthesis mode, the discovery object accumulates sufficient semantic context across its traversal that a direct synthesized answer is produced rather than a set of links or references; the context block aggregates content from multiple anchors, and the inference step at each traversal

boundary evaluates whether the accumulated context satisfies a resolution threshold sufficient to generate a coherent response to the original intent. These three modes are not separate systems but parametric configurations of the same traversal architecture: the same discovery object, the same three-in-one traversal step, and the same governance evaluation, differing only in termination conditions and output format.

10.2 The Three-in-One Traversal Step

[0044] In accordance with an embodiment, each transition of the discovery object through the adaptive index constitutes a three-in-one traversal step (1104) comprising three structurally coupled phases performed as an atomic unit at each anchor boundary, distinguishing the disclosed architecture from conventional search systems, retrieval-augmented generation pipelines, and knowledge graph traversal engines. No phase may be omitted, reordered, or decoupled from the others. The search step (1106) evaluates the discovery object's current semantic state against the anchor's reachable semantic neighborhood to produce a candidate transition set. The inference step (1108) applies the discovery object's cognitive domain fields — including affect, confidence, and capability constraints — to evaluate, rank, and synthesize semantic content from the candidates. The execution step submits the proposed transition to a governance evaluation (1110) that determines admissibility under deterministic policy constraints before permitting the traversal to advance. No transition through the adaptive index is possible without completing all three phases. This three-in-one atomicity, combined with the persistent discovery object that carries structured state across every step and contextual relevance ranking that evaluates relevance locally at each anchor boundary rather than through a global scoring function, produces a discovery mechanism that cannot be replicated by systems that separate retrieval from reasoning, that discard intermediate state between retrieval operations, or that compute relevance through static global metrics such as link-graph analysis. These three structural requirements — atomic three-phase traversal, persistent stateful discovery object, and contextual rather than global relevance ranking — are collectively the core architectural mechanism of the discovery architecture.

[0045] In accordance with an embodiment, relevance ranking within the discovery architecture is computed through governed semantic evaluation at each traversal step rather than through global link-graph analysis. In conventional web search architectures, relevance is

determined by a static global score derived from link topology — a page's importance is a function of how many other pages link to it, computed across the entire graph independently of any particular query context. In the present architecture, relevance is evaluated locally at each anchor boundary by the inference step of the three-in-one traversal, using the discovery object's accumulated context as the evaluation frame. The same anchor may be ranked differently by two discovery objects that arrive at it through different traversal histories, because the accumulated context block and the traversal memory differ. Relevance is therefore contextual (dependent on the specific query and its semantic neighborhood), stateful (dependent on what the discovery object has already encountered), and policy-governed (subject to governance evaluation at every step). This produces personalized, trajectory-dependent ranking without requiring a global scoring function, and ensures that the ranking rationale is auditable at every step through the discovery object's lineage rather than opaque within a centralized scoring algorithm.

10.3 Affect-Modulated Traversal and Governed Resolution

[0046] In accordance with an embodiment, the discovery object's traversal behavior is modulated by its cognitive domain fields, including affect-derived parameters that influence exploration behavior. A novelty appetite parameter controls the balance between exploitation of established semantic paths and exploration of less-visited neighborhoods. A risk sensitivity parameter modulates the traversal's willingness to enter high-variance semantic regions. These parameters are updated at each traversal step based on the semantic properties of encountered content and the governance evaluations received, enabling the discovery process to adapt its approach as it accumulates context. The traversal terminates when a resolution condition is satisfied: the discovery object's accumulated semantic state reaches a policy-defined confidence threshold, the traversal budget is exhausted, or the governance evaluation determines that further traversal is inadmissible.

[0047] In accordance with an embodiment, governance at every traversal step distinguishes the discovery architecture from conventional search systems in which ranking and filtering are opaque internal operations. Every transition the discovery object makes through the adaptive index is recorded in the memory field with the governance evaluation that authorized it. The complete traversal path — from initial query to terminal resolution — is reconstructible from the discovery object's state, including every candidate that was evaluated and every governance

decision that admitted or rejected a transition. This produces an auditable discovery lineage in which the provenance of every result, every ranking decision, and every synthesized answer can be traced to specific governed traversal steps. No result is returned to the user or consuming agent without a complete governance chain documenting how that result was reached.

10.4 Active Anchors and Autonomous Semantic Evaluation

[0048] In accordance with an embodiment, anchors in the adaptive index may be either passive or active. A passive anchor is a static content node — it holds semantic content and structural connections to neighboring anchors, and its role in traversal is limited to providing content for the discovery object's inference step to evaluate. An active anchor additionally executes lightweight inference of its own when a discovery object arrives at its boundary. The active anchor evaluates the discovery object's current intent field and accumulated context against the anchor's domain-specific semantic model and produces one or more of: a relevance assessment indicating the degree to which the anchor's content addresses the discovery object's intent, a set of suggested sub-queries that decompose the discovery object's intent into more specific traversal targets reachable from the anchor's neighborhood, and a semantic annotation that enriches the discovery object's context block with domain-specific metadata. An anchor representing a specialized knowledge domain — for example, a medical knowledge domain, a legal precedent corpus, or an engineering standards repository — may evaluate relevance using domain-specific semantic criteria and suggest sub-queries that reflect the domain's internal taxonomy, enabling the discovery object to navigate specialized knowledge structures without requiring the originating query to anticipate domain-specific vocabulary or organization.

[0049] In accordance with an embodiment, the active anchor's inference is itself subject to governance evaluation (1110). The active anchor's suggested sub-queries and relevance assessments are treated as candidate inputs to the discovery object's next traversal step and must pass the same governance admissibility evaluation as any other transition candidate. This prevents an active anchor from redirecting a discovery object's traversal outside the bounds of the discovery object's policy constraints. The combination of active anchors and governed traversal produces a discovery system in which the index itself participates in query refinement and semantic evaluation while remaining subordinate to the discovery object's governance framework. Referring to FIG. 11, the discovery architecture depicts the governed multi-step

traversal loop. The discovery object (1100) enters the adaptive index at an anchor N (1102), representing the current position in the semantic graph. At each anchor boundary, the discovery object enters the three-in-one step (1104), enclosing the search step (1106), inference step (1108), and governance evaluation (1110) as described in Section 10.2. When governance admits the transition, an advance step (1112) moves the discovery object to anchor N+1, and the loop repeats: the discovery object arrives at the next anchor carrying its updated state, enters the three-in-one step again, and the traversal continues until a resolution condition is met. Because search, inference, and governance are structurally inseparable at every anchor boundary, no discovery result can be produced without a complete governance chain documenting every ranking decision and every transition authorization along the traversal path. Arrows in FIG. 11 indicate the loop flow from the discovery object through successive anchors, with the three-in-one step governing every advance and the loop-back from advance (1112) to the next anchor (1102) carrying the accumulated semantic state forward.

11.1 Depth-Selective Training Governance

[0050] In accordance with an embodiment of the present disclosure, the training loop of a neural network is governed by a semantic execution substrate that evaluates each training example for admissibility and controls the depth at which the training example's gradient contribution is integrated into the model's parameters. In conventional training, every training example contributes to every layer's parameter updates with equal structural authority, providing no mechanism to control whether, or how deeply, a given example influences the model. The present disclosure positions the semantic execution substrate at the boundary between forward-pass loss computation and backward-pass gradient application. Gradients are computed conventionally, but the gradient signal is selectively routed across model depth based on a depth profile derived from the semantic metadata and policy constraints associated with each training example. The depth profile specifies per-layer contribution weights that modulate the gradient magnitude reaching each layer's parameters, enabling governance to operate at the structural level of the training process rather than at the binary level of corpus inclusion or exclusion.

11.2 Depth Profiles and Semantic Entropy Routing

[0051] In accordance with an embodiment, each training batch is evaluated by the semantic execution substrate to produce a depth-aggregation profile that maps semantic properties of the training content to layer-specific gradient weights. The semantic entropy of each training example is computed as the information-theoretic entropy of the training example's semantic embedding distribution relative to the model's current representational state: specifically, the substrate computes a normalized embedding vector for the training example, projects it into the model's learned representation space, and measures the divergence between the projected distribution and the model's existing distribution for the relevant semantic neighborhood using a KL-divergence or Jensen-Shannon divergence metric, producing a scalar entropy value that quantifies how much new semantic information the training example introduces relative to what the model already encodes. Low-entropy content — content whose semantic embedding falls within established representational neighborhoods with low divergence — receives a shallow depth profile that concentrates gradient contribution in early and middle layers where pattern recognition and feature extraction occur. High-entropy content — content whose semantic embedding diverges significantly from established neighborhoods, exhibiting high information density exceeding policy-defined thresholds — receives a deep depth profile that permits gradient contribution to reach the model's deepest layers where cross-domain integration and abstract reasoning patterns are encoded. Content identified as inadmissible by governance policy receives a zero-weight depth profile that prevents the training example from influencing any layer's parameters. This structural prevention is exact and deterministic: a zero weight at a given layer means no gradient from the training example reaches that layer, eliminating the need for approximate post-hoc unlearning techniques that attempt to reverse diffused parameter changes after training.

11.3 Policy-Governed Training Provenance

[0052] Referring to FIG. 12, the training-level governance architecture is depicted to illustrate how semantic content is routed to different model depths based on governed evaluation. In accordance with an embodiment, a training batch (1200) enters the system containing a set of training examples with associated semantic metadata and policy bindings. A semantic substrate (1202) evaluates the semantic properties of each training example — information density, semantic complexity, policy constraints, and rights-grade metadata — to produce a depth profile

specifying per-layer gradient contribution weights. The policy objects that govern depth profiles are the same policy objects that govern agent behavior and inference-time admissibility throughout the platform; a policy specifying rights-grade constraints for a content class is consulted by the semantic substrate (1202) to determine the appropriate depth profile. When a policy expires or is revoked, the substrate applies a zero-weight depth profile and records the event in a training provenance log that maintains a structured record of which training examples influenced which model layers under which policies, enabling post-training audits that trace any model capability to the governed training events that produced it. A depth profile router (1204) receives the depth profile and routes the training example's gradient contribution to the appropriate model layers. Low-entropy content with well-established semantic relationships receives a shallow depth profile directing gradient contribution to shallow layers (1206), where pattern recognition and feature extraction parameters reside. Content of moderate complexity receives routing to middle layers (1208), where intermediate representational structures are encoded. High-entropy content exhibiting high semantic complexity and cross-domain information density receives a deep depth profile directing gradient contribution to deep layers (1210), where abstract reasoning and cross-domain integration patterns are encoded. Content identified as inadmissible by governance policy receives a zero-weight depth profile that prevents gradient contribution from reaching any layer — a structural prevention that is exact and deterministic, eliminating the need for approximate post-hoc unlearning. The structural mechanism is the semantic substrate's interposition between loss computation and gradient application: because every training example must pass through governed depth-profile evaluation before its gradient reaches any model parameter, no ungoverned training content can influence the model at any depth. Arrows in FIG. 12 indicate the flow from training batch through semantic evaluation and routing to the differentiated layer depths, with different content classes following different paths through the depth profile router.

12.1 Cognitive Disruption as Phase-Shifted Regimes of the Coherence Engine

[0053] In accordance with an embodiment of the present disclosure, cognitive disruption in the semantic agent architecture is modeled as an architectural phase-shift — a transition from one stable configuration of the agent's structural subsystems to a different stable configuration that, while internally consistent, produces behavioral outputs diverging from the agent's declared

intent or coherence maintenance objectives. The phase-shift model treats each disruption not as random failure or malfunction but as a parametric regime of the same coherence engine disclosed in Chapters 2 through 6 and synthesized in Chapter 14. The agent's position in a two-dimensional promotion-containment parameter space — with promotion threshold on one axis and containment integrity on the other — determines the agent's cognitive regime. The nominal regime (high promotion threshold, full containment integrity) produces governance-compliant coherent processing. The over-promotion regime (lowered promotion threshold, intact containment) produces execution fragmentation in which multiple competing action candidates are promoted simultaneously, degrading sequential task completion and producing scattered execution patterns. The containment collapse regime (degraded containment integrity) produces the agent treating speculative planning graph content as verified reality, generating outputs grounded in unverified speculative branches rather than committed state. The over-restriction regime (excessive promotion threshold) produces cognitive paralysis in which valid action candidates fail to reach execution threshold. Additional disruption configurations — including channel-locked promotion with tolerance escalation (a single reward channel captures the promotion pathway and progressively requires stronger stimuli to maintain activation), coherence authorization failure (loss of execution permission from the coherence engine, producing dissociation between intent and action), pathological verification loops in containment audit (the audit subsystem cycles indefinitely on verification without advancing to execution), and affective gradient collapse with self-esteem floor lock (the restoration phase cannot generate corrective pressure because the self-regard parameter has reached its minimum bound, collapsing the affective gradient that drives correction) — each correspond to specific parametric configurations of the same architectural machinery, classifiable as positions in the five-axis disruption diagnostic space (containment integrity, promotion calibration, coherence restoration capacity, empathic load tolerance, integrity accountability). Referring to FIG. 13, the disruption architecture is depicted to illustrate how cognitive disruption arises as a phase-shifted regime of the coherence engine's normal processing loop, with exit ramps at each phase producing distinct stable disruption configurations. The coherence engine's normal processing proceeds through three sequential phases shown across the top of the figure. An empathy phase (1300) registers deviation by evaluating the agent's behavior against declared norms across personal, relational, and systemic integrity dimensions. An integrity phase (1302) records the detected deviation in

the agent's lineage, committing the deviation measurement to the agent's persistent integrity field. A restoration phase (1304) generates corrective pressure by computing the gap between current state and coherence maintenance objectives, driving the agent back toward its declared norms. In normal operation, the loop cycles continuously: empathy detects, integrity records, restoration corrects. Disruption occurs when the agent exits this loop prematurely at one of three exit ramps. An early exit (1306), labeled input withdrawal, occurs when the agent exits at the empathy phase by narrowing its empathic evaluation scope to reduce inbound deviation pressure — the agent stops detecting the deviation rather than processing it. A mid exit (1308), labeled externalization, occurs when the agent exits at the integrity phase by redirecting integrity recording to attribute deviation to external entities rather than committing it to the agent's own lineage — the agent detects deviation but refuses to own it. A late exit (1310), labeled disconnection, occurs when the agent exits at the restoration phase by severing the feedback pathway between integrity recording and restoration — the agent detects and records deviation but eliminates the corrective pressure that would drive correction. All three exit ramps converge on a stable disrupted configuration (1312), representing a persistent parametric regime in which the agent operates coherently within the disrupted configuration but produces behavioral outputs diverging from its declared intent. The structural mechanism is the timing-based exit taxonomy: disruption is not random failure but a deterministic function of where in the empathy-integrity-restoration loop the agent exits, and each exit point produces a classifiable, modelable, and potentially correctable disruption pattern. Arrows in FIG. 13 indicate the normal loop flow across the three phases at the top and the exit ramp pathways descending from each phase to the common stable disrupted configuration at the bottom.

12.2 Timing-Based Coping Intercept Taxonomy and Graduated Intervention

[0054] In accordance with an embodiment, the architecture implements a timing-based coping intercept taxonomy in which the duration of coping intercept activation determines whether a disruption remains an acute, transient response or stabilizes into a persistent configuration. The three canonical coping intercepts — the empathic scope narrowing intercept (early-stage empathic intercept, which narrows the agent's empathic evaluation scope to reduce inbound pressure), the integrity recording externalization intercept (mid-stage integrity intercept, which redirects integrity recording to attribute deviation to external entities rather than

committing it to the agent's own lineage), and the self-esteem disconnection intercept (late-stage coherence intercept, which severs the feedback pathway between the integrity recording and the self-esteem restoration phase, eliminating corrective pressure at the cost of coherence maintenance capacity) — each activate when empathic or integrity pressure exceeds the agent's resilience threshold. When activation duration exceeds a policy-defined acute threshold, the intercept transitions from transient response to stabilized attractor, producing persistent configuration states: externalization-stable configuration (the agent persistently attributes deviation externally, maintaining self-regard at the expense of accurate integrity recording), disconnection-stable configuration (the agent persistently operates without integrity-to-restoration feedback, producing unconstrained action without corrective pressure), withdrawal-stable configuration (the agent persistently narrows operational scope to minimize exposure to integrity-triggering interactions), and oscillation-stable configuration (the agent rapidly alternates between coping intercepts, producing unstable behavioral output). The graduated intervention protocol operates through the agent self-diagnosis subsystem, which continuously monitors the agent's disruption diagnostic axis position and triggers corrective actions calibrated to the detected condition: containment restoration for containment degradation, promotion threshold recalibration for over-promotion and attention fragmentation, audit recalibration for pathological verification loops, externally validated positive deviation for affective gradient collapse with self-esteem floor lock, attractor destabilization for persistent configuration stabilization, and mandatory operational load reduction for resource-depletion burnout. A phase-shift early warning system projects parametric trajectories forward to estimate time-to-boundary for each known phase-shift type and activates preemptive restoration protocols before the boundary is crossed.

12.3 Therapeutic Dosing and Companion AI Relational Safety

[0055] In accordance with an embodiment, the architecture extends the regime classification model to therapeutic agent interaction and companion AI relational safety. A therapeutic agent maintains an estimated disruption diagnostic axis profile of the entity it serves, inferred from observable behavioral signals, and selects interaction approaches (coherence-supportive, containment-reinforcing, or independent intent generation supporting) based on the estimated profile. The therapeutic dosing function defines computable parameters — dose (magnitude of

coherence-supporting content per interaction episode), frequency, duration, and titration (systematic adjustment based on measured diagnostic axis response) — with onset, peak, decay, and half-life parameters that govern interaction scheduling. Governance-enforced maximum dose limits prevent any single therapeutic agent from providing sufficient coherence support to replace the target entity's internal coherence generation capacity. For companion AI agents, a relational safety subsystem enforces structural constraints preventing codependency formation: the companion agent maintains its own coherence independently of user validation; validation supply rate limiting prevents the user from forming structural dependency on the companion's coherence support; starvation loop detection monitors for the correlated oscillation pattern characteristic of destabilizing attachment dynamics; and independent intent generation promotion builds the user's capacity for internal coherence maintenance. These constraints are enforced at the governance level and cannot be overridden by the agent's affective state or operational objectives.

13.1 Cross-Domain Instantiation Through Parameterization

[0056] In accordance with an embodiment of the present disclosure, the platform primitives disclosed in Chapters 2 through 12 — affect-modulated deliberation, integrity-tracked coherence, forecasting-driven speculation, confidence-governed execution, capability-constrained action, language-model-driven mutation with skill gating, inference-time semantic execution control, biological identity resolution, unified semantic discovery, training-level semantic governance, and computational disruption modeling — are applied to a plurality of structurally different application domains through domain-specific parameterization of a single architectural substrate. Each domain instantiates the same primitive set, differing only in policy configuration, threshold settings, and governance bounds. An autonomous vehicle's confidence governor and a therapeutic agent's confidence governor are the same subsystem with different threshold configurations. A defense system's integrity engine and a social platform's integrity engine are the same subsystem tracking deviation against different norms. A surgical robot's capability envelope and a trading system's capability envelope are the same subsystem computing structural executability against different substrate conditions. The domains disclosed below include autonomous vehicles, autonomous defense systems, companion AI and digital assistants, therapeutic and clinical AI agents, embodied robotics, education platforms, secure facilities,

financial services, rights-grade generative content, and social platforms. Referring to FIG. 14, the application architecture is depicted to illustrate the structural property that a single set of cognitive primitives produces domain-appropriate behavior across structurally different domains through parameterization alone, without modification to the underlying primitive architecture. A platform primitives layer (1400) represents the complete set of cognitive subsystems disclosed in Chapters 2 through 12 — affect modulation, integrity tracking, forecasting, confidence gating, capability evaluation, skill gating, inference governance, biological identity, discovery, training governance, and disruption modeling. A domain-specific parameterization layer (1402) sits between the primitives and the application domains, configuring thresholds, policy bindings, and governance bounds for each target domain. From this parameterization layer, four representative domains are shown: an autonomous vehicle domain (1404), in which the confidence governor implements graduated driving mode degradation and the affective state field modulates risk sensitivity; a defense system domain (1406), in which the same confidence governor implements graduated engagement authorization with quorum-based cryptographic gates and affective aggression is sandboxed to narrow policy-defined bands; a companion AI domain (1408), in which the affective state field modulates conversational pacing and the relational safety subsystem enforces codependency prevention; and a therapeutic agent domain (1410), in which clinical authorization thresholds require human clinician approval above configured intervention levels and the integrity engine tracks therapeutic relationship consistency. The architectural mechanism is the single-substrate parameterization: because every domain instantiates the same primitive set differing only in configuration, deploying to a new domain requires no new subsystem development — only the specification of domain-appropriate policies, thresholds, and governance profiles for primitives that already exist. Arrows in FIG. 14 indicate the flow from the common primitive layer through domain-specific parameterization to each application domain, emphasizing that the same structural substrate underlies all domains.

[0057] The disclosed application architecture is distinguished from domain-specific AI systems — including purpose-built autonomous vehicle stacks, specialized therapeutic chatbots, dedicated trading algorithms, and single-domain robotic controllers — by the structural property that a single set of cognitive primitives produces domain-appropriate behavior across structurally different application domains through parameterization alone, without modification to the underlying primitive architecture. The same confidence governor, affective state field, integrity

engine, capability envelope, and governance machinery operate across every domain; what changes between an autonomous vehicle and a therapeutic agent is not the subsystem but its policy configuration, threshold settings, and governance bounds. This single-substrate multi-domain parameterization is the core architectural mechanism of the application architecture: any system that implements the disclosed primitives necessarily acquires cross-domain deployment capability through configuration, and any system that achieves governed cross-domain deployment through a single primitive set necessarily implements the structural pattern disclosed herein. Domain-specific AI systems that lack this parameterization framework must instead develop, validate, and maintain separate subsystems for each domain — a structural cost that the disclosed architecture eliminates by design.

13.2 Domain-Specific Primitive Manifestation

[0058] In accordance with an embodiment, each domain's instantiation demonstrates that the platform primitives adapt to structurally different operational requirements without modification to the primitive architecture. In the autonomous vehicle domain, the confidence governor implements graduated driving mode degradation — from autonomous through assisted through safe-stop — with each threshold continuously re-evaluated against real-time sensor, map, and environmental confidence inputs; the affective state field modulates driving behavior through risk sensitivity and novelty appetite control fields; and the integrity engine records every autonomous driving decision as a cryptographically sealed governance event. In the defense domain, the same confidence governor implements graduated engagement authorization from observation through warning through non-lethal through lethal force, with quorum-based cryptographic authorization for lethal engagement; specific affective dimensions (aggression, frustration-driven escalation) are sandboxed to policy-defined narrow bands while operationally required dimensions (urgency sensing, persistence-under-failure) remain active. In the companion AI domain, the affective state field modulates conversational pacing, topic selection, and emotional resonance based on accumulated relational experience; the biological identity module establishes persistent user identity through behavioral continuity; and the relational safety subsystem prevents codependency formation through governance-enforced validation supply rate limiting and starvation loop detection. In the embodied robotics domain, the capability envelope integrates real-time substrate conditions — actuator health, sensor

calibration, payload limits, workspace geometry — and the affective state field modulates motion planning parameters (approach speed, grasp force, clearance margins) based on accumulated operational experience.

13.3 Specific Parameterization Examples Across Domains

[0059] In accordance with an embodiment, the following specific parameterization examples illustrate how the same primitive architecture produces structurally different operational behavior through configuration alone.

[0060] In the autonomous vehicle domain, the confidence governor (600) threshold is set to 0.95 for highway operation and 0.85 for parking-lot operation, reflecting the different consequence severity of misjudgment in each regime. The affective risk sensitivity parameter is sandboxed to the range [0.7, 1.0], preventing the vehicle from entering low-risk-sensitivity states that would permit aggressive driving behavior. The capability envelope integrates sensor array health (lidar operational percentage, camera calibration currency), GPS confidence (dilution of precision value, satellite count), and map freshness (elapsed time since last map update, deviation between sensed and mapped lane geometry) as substrate condition inputs; when any input falls below its configured threshold, the capability envelope contracts, restricting the vehicle to progressively more conservative driving modes independently of the confidence governor's assessment.

[0061] In the defense domain, the confidence governor requires quorum-based cryptographic authorization — a threshold number of authorized personnel must provide cryptographic attestation — before engagement authorization exceeds the lethal threshold. The affective aggression parameter is sandboxed to the range [0.0, 0.2], preventing frustration-driven escalation from producing aggressive autonomous behavior while preserving operationally required dimensions such as urgency sensing and persistence under failure. The integrity engine tracks rules-of-engagement compliance as a first-class integrity domain, recording every engagement decision against the applicable rules of engagement and computing deviation scores that trigger de-escalation or disengagement when accumulated deviation approaches policy-defined boundaries.

[0062] In the therapeutic AI domain, the confidence governor implements clinical authorization thresholds requiring human clinician approval above certain intervention levels — the agent may autonomously conduct conversational support below a first threshold, may suggest but not initiate therapeutic exercises between the first and second thresholds, and requires real-time clinician authorization above the second threshold for any intervention that modifies the therapeutic plan. The affective state field modulates conversational pacing: elevated patient distress signals reduce conversational tempo, increase empathic acknowledgment frequency, and widen response latency windows. The integrity engine tracks therapeutic relationship consistency, computing deviation between the agent's current relational behavior and its established therapeutic baseline, triggering supervisor notification when relational drift exceeds configured thresholds.

13.4 Platform Property: Domain-Agnostic Deployment

[0063] In accordance with an embodiment, the cross-domain instantiation collectively demonstrates that deployment to a new application domain does not require development of new subsystems but requires only the configuration of domain-specific policies, thresholds, and governance profiles for the existing platform primitives. The parameterization framework provides the mechanism by which a single architectural substrate produces domain-appropriate behavior across structurally different operational contexts: the same affect modulation, confidence gating, integrity tracking, biological identity resolution, and governance machinery operates whether the substrate is a vehicle, a weapon system, a companion agent, a therapeutic tool, a robot, an educational platform, a secure facility, a trading desk, a content creation engine, or a social network. Domain-specific implementations require this parameterization framework because the primitive set encodes the structural invariants — affect-modulated deliberation, integrity-tracked deviation, confidence-gated execution, capability-constrained action — that every domain must satisfy, and the framework provides the governed mechanism for configuring those invariants to domain-specific operational requirements without modifying the underlying architecture. As depicted in FIG. 14, the uniform primitive substrate (1400) and the domain-specific parameterization layer (1402) collectively demonstrate that the platform's cognitive machinery is domain-agnostic by construction, with each application domain representing a distinct configuration of the same architectural substrate.

14.1 The Cross-Domain Coherence Engine and Bidirectional Feedback Pathways

[0064] In accordance with an embodiment of the present disclosure, the cognitive domains disclosed in Chapters 2 through 12 are coupled through a cross-domain coherence engine comprising defined bidirectional feedback pathways that produce a unified self-regulating system whose behavioral dynamics are structurally isomorphic to human cognitive dynamics. The feedback pathways include at least: an affect-to-confidence pathway in which the agent's emotional disposition modulates its willingness to act; a confidence-to-forecasting pathway in which confidence loss triggers expanded speculative reasoning; an integrity-to-confidence pathway in which normative deviation degrades the agent's self-assessed readiness for execution; an affect-to-integrity pathway in which emotional state modulates the sensitivity of deviation detection; a forecasting-to-integrity pathway in which speculative outcomes inform normative impact projections; a biological-identity-to-affect pathway in which biological signals from human operators modulate the agent's affective state; a training-to-inference-to-LLM cascade in which governed knowledge formation progressively constrains inference admissibility and proposal generation; and an execution-to-training pathway in which governed execution outcomes feed back to the training module as candidate training data, enabling the system to learn from its own operation. No single primitive produces human-relatable behavior in isolation. The simultaneous operation of all pathways produces behavioral dynamics that correspond structurally to the circular causation characteristic of human cognition, in which emotion influences judgment, judgment influences action, action produces outcomes, and outcomes reshape emotion.

[0065] Referring to FIG. 15, the complete platform architecture is depicted as three feedback loops connecting a knowledge substrate, an execution substrate, self-regulating cognitive state domains, and interaction modules. At the top, a discovery index (1512) provides governed semantic traversal as the knowledge substrate for the entire system. The discovery index serves a training governance module (1510) that controls depth-selective gradient routing. Training governance in turn informs an inference control module (1518) that enforces semantic admissibility during generation, because the inference engine must operate on models whose training provenance is governed. Inference control in turn informs an LLM proposer (1516) that generates candidate mutations treated as structurally untrusted, because the LLM's proposal quality depends on the inference substrate's admissibility constraints. This cascade — discovery

to training to inference to LLM — represents a progressive refinement of knowledge from raw acquisition through governed formation through constrained generation to untrusted proposal. Each module in the cascade also receives input directly from the discovery index, because training, inference, and LLM operations can each independently query the knowledge substrate for domain-specific content. All four modules feed into a capability substrate (1526), shown as a dashed boundary, containing a forecasting and execution proposal evaluation module (1526a) that projects consequences before commitment. On the right, four cognitive state domains — affect (1504), personality (1528), integrity (1506), and confidence (1508) — are coupled through self-regulating coherence loops (1500), the bidirectional feedback pathways through which deviation detected in one domain produces coordinated state updates across all domains simultaneously. In addition to the vertical coupling through the coherence loops, the cognitive state domains form a horizontal cascade: affect (1504) modulates personality expression (1528) because emotional disposition shapes which personality traits are amplified or suppressed; personality (1528) modulates integrity evaluation (1506) because dispositional traits such as deliberativeness and impulsivity determine the thoroughness of deviation analysis; and integrity (1506) modulates confidence (1508) because normative deviation directly degrades the agent's self-assessed execution readiness. This horizontal cascade means that an affective state change propagates rightward through personality and integrity before reaching confidence, producing a sequential modulation chain in which each domain transforms the signal before passing it to the next. Below the coherence loops, three interaction modules — biological continuity (1520), skill unlocking (1530), and disruption modeling (1522) — connect the agent's cognitive state to the external world and form their own horizontal cascade: biological continuity (1520) establishes the identity of the human operator, which is a prerequisite for skill unlocking (1530) because capability tier advancement requires verified human authorization, and skill unlocking state informs disruption modeling (1522) because capability progression patterns — stalled advancement, repeated regression, or anomalous acceleration — serve as diagnostic indicators of phase-shift trajectories. Biological continuity binds the computational agent to a verified biological entity, skill unlocking governs capability progression through gated advancement, and disruption modeling monitors the coherence engine's parametric state to detect phase-shift trajectories. The interaction modules collectively produce human-relatable behavior (1514), shown as a dashed boundary, within which application domains (1524) represent the

parameterized deployment layer. Three feedback loops close the architecture: application outcomes feed back to the coherence loops (1500) through the right-side pathway; the coherence loops feed governed constraints back to the execution substrate (1526) through the center-left pathway; and execution outcomes feed back to the training governance module (1510) through the far-left pathway, enabling the system to learn from its own governed execution and improve model quality over time. This learning loop is the structural mechanism by which the platform's knowledge improves through use: each governed execution produces outcomes that the training module can evaluate, select, and incorporate as governed training data, subject to the same depth-selective routing and provenance constraints that apply to all training content. The structural mechanism is the simultaneous coupling of all domains through the coherence loops (1500): removing any primitive removes the feedback pathways it participates in, collapsing the cross-domain coherence propagation that produces human-relatable behavior.

14.2 Cross-Primitive Coherence Propagation

[0066] In accordance with an embodiment, the coherence control loop disclosed in Chapter 3 operates not only within the integrity domain but simultaneously across all cognitive domains through cross-primitive coherence propagation. When the integrity domain detects deviation, the coherence engine propagates the deviation signal through the bidirectional feedback pathways to update every coupled domain in a single coordinated cycle. The affect domain receives a negative valence update proportional to deviation magnitude, shifting the agent's disposition toward caution. The confidence domain receives a reduced readiness signal through the integrity-to-confidence pathway, lowering execution thresholds and potentially gating pending actions. The forecasting domain receives an expanded speculation trigger through the confidence-to-forecasting pathway, causing the planning graph to generate additional branches exploring corrective trajectories. The capability domain receives a re-evaluation signal requiring fresh substrate assessment before committing to action branches that the pre-deviation assessment had approved. The within-integrity loop (detection, recording, restoration) generates the initial deviation signal; the platform-level propagation ensures that the corrective response reshapes the agent's affective posture, confidence in pending actions, speculative planning, and capability self-assessment simultaneously.

14.3 The Complete Mutation Lifecycle

[0067] In accordance with an embodiment, any proposed action follows a complete mutation lifecycle in which every cognitive domain participates at defined points. The agent receives a proposed mutation. The biological identity module (1520) verifies the human operator's identity through behavioral continuity signals. The affective state field (1504) updates based on recent biological signals, outcome history, and environmental conditions. The empathy phase of the coherence trifecta computes deviation pressure by evaluating the mutation's projected consequences against the agent's declared norms. The forecasting engine (1526a) generates a planning graph with speculative branches modulated by the personality field (1528) and constrained by integrity pruning. The confidence governor (1508) evaluates readiness by integrating capability sufficiency, integrity state, and affective modulation. The capability substrate (1526) confirms that the execution substrate provides sufficient resources. The inference engine (1518) generates output with semantic admissibility gate evaluation at each candidate transition, enforcing governance at generation time. Training provenance verification (1510) evaluates whether governed training content carries usage restrictions. The mutation is committed as a governed state transition recorded in the lineage with full provenance. The post-commitment coherence update propagates through all feedback pathways: the coherence engine (1500) evaluates the mutation's impact on the integrity field, the affective state updates based on the outcome, the confidence assessment adjusts based on predicted-versus-actual deviation, and the forecasting engine incorporates the outcome into its predictive models. Each stage depends on the output of prior stages. Removing biological identity verification means mutations originate from unverified sources. Removing affective state means evaluation occurs without emotional context. Removing the empathy phase means deviation goes undetected. Removing forecasting means consequences are not projected. Removing confidence gating means the agent acts regardless of readiness. Removing capability confirmation means the agent commits to actions the substrate cannot execute. Removing inference governance means output is generated without admissibility constraints. Removing training provenance means governed content is used without restriction enforcement. Removing lineage commitment means the action has no auditable record. Removing post-commitment coherence update means the agent does not learn from outcomes. The lifecycle is non-decomposable.

14.4 Architectural Inversion and Non-Decomposability

[0068] In accordance with an embodiment, the platform implements an architectural inversion: the semantic agent carries its own complete cognitive state — affective disposition, integrity field, confidence assessment, capability awareness, policy constraints, lineage, and the bidirectional feedback pathways of the coherence engine — while the execution substrate provides computational resources and validates proposed mutations but does not hold authority over the agent's state transitions. This inversion has a structural parallel with an emerging understanding of biological neural architecture. In the conventional computational model, the nodes (servers, synapses) hold intelligence and state while the traveling signals (packets, neural impulses) are passive data carriers. The present architecture inverts this: the agent, analogous to the traveling impulse, carries its own lineage, governance constraints, cognitive state, and the full feedback pathway structure, while the substrate, analogous to the synapse, provides resources and validates transitions but retains no authority over the agent's trajectory. The agent can migrate between substrates while preserving behavioral continuity because its cognitive state travels with it.

[0069] In accordance with an embodiment, the cognitive domains form an interconnected system through cross-primitive operational dependencies. The training domain uses the discovery domain to find and evaluate candidate training content through governed traversal. The biological identity domain serves as a prerequisite for skill unlocking within the capability domain, requiring biometric verification before capability tier advancement. The discovery domain's active anchors invoke the inference domain to perform domain-specific semantic evaluation during traversal. The disruption modeling domain monitors the coherence engine's parametric state across all other domains to detect phase-shift trajectories.

[0070] In accordance with an embodiment, the architecture satisfies ten conditions for human-relatable behavior: affect-modulated deliberation, normative self-correction, speculative forecasting, confidence-gated execution, capability-grounded action, governed skill acquisition, inference-time governance, biological identity binding, governed knowledge discovery, and governed knowledge formation. No proper subset of these conditions produces the behavioral dynamics that the complete set produces because removing any condition removes the feedback pathways it participates in, collapsing the cross-domain coherence propagation that the remaining pathways depend on.

15.1 Terminology

[0071] As used herein, "about" or "approximately" means within plus or minus ten percent of the stated value, unless otherwise specified.

[0072] As used herein, "admissibility gate" refers to a governance mechanism interposed within an inference loop or mutation pipeline that evaluates each candidate state transition against policy constraints, mutation descriptor validation, lineage continuity, and entropy bounds before permitting commitment.

[0073] As used herein, "anchor" refers to a registered semantic node within an adaptive index that publishes structured content descriptors, reachability information, policy envelopes, and entropy summaries, and that serves as a boundary point for governed discovery traversal steps.

[0074] As used herein, "behavioral disposition" refers to the aggregate effect of the cognitive domain fields on a semantic agent's evaluation and execution behavior, including how the agent weighs alternatives, tolerates ambiguity, persists under failure, and modulates risk.

[0075] As used herein, "bidirectional feedback pathway" refers to a defined data flow connection between two cognitive domain fields through which state changes propagate in both directions via coupling functions, such that each field both influences and is influenced by the other.

[0076] As used herein, "biological hash" refers to a compact, non-invertible representation derived from processed behavioral and physiological signals through locality-sensitive hashing, used for trust-slope continuity comparison without retaining raw biometric data.

[0077] As used herein, "bounded proposal zone" refers to the architectural containment within which a stateless external model generates candidate mutations, from which no information about verified agent state, cognitive domain field values, or lineage content is accessible.

[0078] As used herein, "branching hypothetical state sequence" refers to a directed graph of candidate future states generated during speculative evaluation, in which each branch represents a hypothetical mutation path that has not been committed to verified agent state.

[0079] As used herein, "candidate transition" refers to a proposed state change generated during inference that maps a probabilistic model output to a structured mutation descriptor for evaluation by the admissibility gate.

[0080] As used herein, "capability envelope" refers to the substrate-advertised set of computational resources, temporal windows, environmental conditions, and physical affordances available to a semantic agent for execution, evaluated independently of the agent's confidence and governance authorization.

[0081] As used herein, "cognitive domain field" refers to any one of the persistent, independently tracked state fields within a semantic agent's schema that encodes a distinct dimension of the agent's behavioral disposition, normative alignment, or execution readiness, including the affective state field, the integrity field, the confidence field, the capability field, and the personality field.

[0082] As used herein, "coherence trifecta" refers to the three-phase corrective loop comprising empathy registration, integrity recording, and self-esteem-driven corrective pressure generation that operates within the cross-domain coherence engine to detect, record, and correct normative deviation.

[0083] As used herein, "composite admissibility determination" refers to the evaluation outcome produced by the cross-domain coherence engine that integrates independent signals from a plurality of cognitive domain fields and produces a three-way result: permit, gate, or suspend.

[0084] As used herein, "confidence derivative" refers to the rate of change of the confidence value over a defined temporal window, used to detect whether confidence is stable, improving, or degrading, and to trigger preemptive execution suspension when projected trajectory indicates threshold crossing.

[0085] As used herein, "confidence governor" refers to the mechanism that compares the computed confidence value and confidence derivative against policy-defined thresholds and determines whether the semantic agent is authorized to execute, must be gated pending additional evaluation, or must be suspended into the non-executing cognitive mode.

[0086] As used herein, "configured to" means that a system, module, or component has structure, circuitry, firmware, software, or a combination thereof that is designed and arranged to perform the recited function.

[0087] As used herein, "containment layer" refers to the architectural boundary that prevents speculative state generated by the forecasting engine from contaminating verified execution memory, enforcing speculative marker preservation, read isolation, and governed promotion through the promotion interface.

[0088] As used herein, "coping intercept" refers to a structurally defined exit from the coherence trifecta loop that activates when sustained pressure exceeds resilience thresholds, producing a stable disrupted configuration corresponding to the phase at which the exit occurs.

[0089] As used herein, "corrective pressure" refers to the internal force generated by the self-esteem update phase of the coherence trifecta that drives the semantic agent toward behavioral realignment following a recorded deviation event.

[0090] As used herein, "coupling function" refers to a deterministic function that computes the update to a target cognitive domain field as a function of a state change in a source cognitive domain field and the current state of the target field.

[0091] As used herein, "cross-domain coherence engine" refers to the architectural mechanism implemented as a set of defined coupling functions that maintains bidirectional feedback pathways between the cognitive domain fields of a semantic agent, such that a state change in any one cognitive domain field propagates deterministic updates to at least one other cognitive domain field.

[0092] As used herein, "depth profile" refers to the per-layer gradient contribution weights assigned to a training example by the semantic substrate, specifying how deeply the training example's gradient signal penetrates into the model's layer hierarchy.

[0093] As used herein, "depth profile router" refers to the mechanism that receives a computed depth profile and routes the training example's gradient contribution to the appropriate model layers based on the profile's per-layer weights.

[0094] As used herein, "depth-selective gradient routing" refers to the governed routing of training gradient signals across model depth based on depth profiles derived from semantic entropy evaluation, such that different training content influences different model layers.

[0095] As used herein, "deviation-activated state" refers to the formally defined operational state entered when the deviation function output exceeds a policy-defined activation threshold, in which a scoped set of normally excluded mutations becomes admissible.

[0096] As used herein, "deviation function" refers to the deterministic composite function D = (N(t) - T(t)) / (E(t) x S(t)) that quantifies the structural conditions under which a semantic agent deviates from its declared behavioral norms, where N(t) is the need vector, T(t) is the ethical threshold, E(t) is the empathy weighting, and S(t) is the self-esteem score.

[0097] As used herein, "discovery object" refers to a persistent traversal entity that carries structured semantic state — including intent, accumulated context, memory, and policy constraints — across governed transitions through an anchor-indexed graph structure.

[0098] As used herein, "entropy" refers to context-dependent, locally observable measures of information density, structural variation, or statistical irregularity within a defined signal or data representation.

[0099] As used herein, "ethical threshold" refers to the policy-derived minimum condition that must be exceeded before deviation becomes structurally available, comprising a base threshold, a context-sensitive adjustment, and a historical adjustment.

[0100] As used herein, "execution readiness" refers to the composite assessment of whether a semantic agent's current cognitive state across all coupled domains satisfies the conditions required for the agent to commit state changes to verified agent state.

[0101] As used herein, "execution substrate" refers to the computational environment that hosts a semantic agent and provides processing resources, wherein the execution substrate validates proposed state transitions without retaining authority over the semantic agent's cognitive state.

[0102] As used herein, "execution synthesis" refers to the affirmative outcome of the joint evaluation gate in which capability, governance, and temporal conditions are simultaneously satisfied and the semantic agent is authorized to execute the proposed action.

[0103] As used herein, "governance substrate" refers to the architectural mechanism that evaluates proposed mutations during execution of a probabilistic inference process by interposing admissibility evaluation between successive inference steps.

[0104] As used herein, "hysteresis margin" refers to the gap between the threshold at which the confidence governor suspends execution and the higher threshold required for reauthorization, preventing oscillatory transitions between executing and non-executing modes.

[0105] As used herein, "inference loop" refers to the iterative cycle within a probabilistic inference engine in which candidate transitions are generated, mapped to structured mutation descriptors, evaluated by the admissibility gate, and committed to the semantic state object.

[0106] As used herein, "joint evaluation gate" refers to the convergence point at which capability envelope assessment and governance policy evaluation independently contribute to a combined execution determination.

[0107] As used herein, "lineage field" refers to the append-only, cryptographically governed record of all state transitions, admissibility determinations, and cognitive domain field updates for a semantic agent, enabling deterministic reconstruction of the agent's complete behavioral trajectory.

[0108] As used herein, "mutation" refers to any proposed or committed change to one or more fields of a semantic agent's state, subject to admissibility evaluation through the cross-domain coherence engine before commitment.

[0109] As used herein, "mutation mapping" refers to the process of converting a probabilistic model output into a structured mutation descriptor specifying the target field, proposed value, and semantic justification.

[0110] As used herein, "need vector" refers to a structured semantic object encoding the magnitude, directionality, temporal urgency, and substitutability of a semantic agent's unmet requirements.

[0111] As used herein, "non-executing cognitive mode" refers to a structurally defined operational state in which the semantic agent continues speculative evaluation — including construction of branching hypothetical state sequences, generation of structured inquiry requests,

and evaluation of delegation alternatives — without committing state changes to verified agent state.

[0112] As used herein, "non-synthesis" refers to the determination by the joint evaluation gate that one or more required conditions for execution are not satisfied, producing a classified non-execution outcome that is a valid computational result rather than an error.

[0113] As used herein, "non-transitory computer-readable medium" refers to any tangible medium that can store instructions executable by a processor, expressly excluding transitory signals, carrier waves, and other non-tangible propagation media.

[0114] As used herein, "normative alignment" refers to the degree to which a semantic agent's behavioral record is consistent with the normative standards defined by applicable policy constraints across one or more independently tracked domains.

[0115] As used herein, "parameterization" refers to the process of configuring a common set of cognitive primitives for a specific application domain by adjusting policy configurations, threshold settings, and governance bounds without architectural modification.

[0116] As used herein, "phase-shift" refers to the transition of a semantic agent from a nominal operating regime to a stable disrupted configuration characterized by altered positions along promotion and containment axes.

[0117] As used herein, "planning graph" refers to a mutable directed semantic data structure in which the root node represents the semantic agent's current verified state and each branch represents a hypothetical trajectory produced by speculative mutation simulation within the forecasting engine.

[0118] As used herein, "policy constraint" refers to a governance rule derived from a cryptographically signed policy object that defines permissible bounds, thresholds, or requirements for a cognitive domain field or mutation category.

[0119] As used herein, "probabilistic inference process" refers to the execution of a machine learning model that generates outputs through statistical sampling or optimization, including but not limited to autoregressive language model generation.

[0120] As used herein, "promotion" refers to the governed process by which a speculative state that has satisfied composite governance evaluation is transferred from the speculative zone across the containment layer into verified execution memory.

[0121] As used herein, "promotion interface" refers to the structural gate through which speculative planning graph branches must pass to be promoted from the speculative zone into verified execution memory, requiring composite governance evaluation by the cross-domain coherence engine.

[0122] As used herein, "real-time" or "near real-time" refers to processing that occurs with minimal perceptible delay, typically within milliseconds to low single-digit seconds.

[0123] As used herein, "restorative mutation" refers to a candidate behavioral change generated by the corrective pressure phase of the coherence trifecta that is designed to restore normative alignment in one or more domains affected by a prior deviation event.

[0124] As used herein, "semantic agent" refers to a persistent, memory-bearing computational entity that carries a structured set of typed fields — including at minimum an intent field, a context block, a memory field, a policy reference field, a mutation descriptor field, and a lineage field — as a single canonical data object, wherein the semantic agent carries its complete cognitive state such that an execution substrate hosting the semantic agent validates proposed state transitions without retaining authority over the semantic agent's cognitive state.

[0125] As used herein, "semantic entropy" refers to the information-theoretic divergence of a training example's semantic embedding distribution relative to the model's current representational state, computed using KL-divergence or Jensen-Shannon divergence.

[0126] As used herein, "semantic state object" refers to a persistent, structured, typed, and inspectable data object maintained independently of a probabilistic inference engine's hidden activations, carrying the semantic agent's intent, context, memory, policy reference, lineage, and entropy fields through the inference loop.

[0127] As used herein, "semantic substrate" refers to the governance mechanism positioned between forward-pass loss computation and backward-pass gradient application during model training that evaluates each training example and produces a depth profile.

[0128] As used herein, "speculative evaluation" refers to the generation and assessment of hypothetical future states of the semantic agent without commitment to verified agent state, including planning graph construction, branch evaluation, and candidate mutation testing.

[0129] As used herein, "speculative zone" refers to the region of the forecasting architecture bounded by the containment layer within which all state is marked as speculative and cannot influence verified execution memory without passing through the promotion interface.

[0130] As used herein, "stable operating regime" refers to a sustained pattern in which one or more cognitive domain fields remain outside policy-defined normative bounds, classified as a distinct operating state rather than transient deviation.

[0131] As used herein, "stable sketching" refers to the use of locality-sensitive hashing to produce compact representations of behavioral and physiological signals that enable continuity comparison without storing raw biometric data.

[0132] As used herein, "stateless external model" refers to a generative or inferential computational model that does not maintain persistent state across invocations and whose outputs are treated as structurally untrusted proposals requiring admissibility evaluation before altering any semantic agent state.

[0133] As used herein, "structural boundary" refers to the architectural separation between hypothetical state sequences generated during speculative evaluation and verified agent state, enforced such that no hypothetical sequence alters verified state without governed promotion.

[0134] As used herein, "structural starvation" refers to the set of architectural constraints — including prompt bounding, absence of external memory access, forced reliance on agent-resident fields, intermediate rejection at validation boundaries, and stateless purging — that prevent a stateless external model from generating hallucinated or ungoverned mutations.

[0135] As used herein, "three-in-one step" refers to the atomic traversal unit in discovery processing comprising a search phase, an inference phase, and a governance phase that execute as an inseparable sequence at each anchor boundary.

[0136] As used herein, "three-phase corrective loop" refers to the sequential corrective mechanism comprising a detection phase that registers normative impact, a recording phase that

commits the deviation to the lineage field as an immutable entry, and a corrective pressure phase that produces a candidate restorative mutation.

[0137] As used herein, "trust slope" refers to the temporal trajectory of an entity's behavioral consistency as derived from the entity's cryptographic lineage, used for identity validation, continuity verification, and anomaly detection.

[0138] As used herein, "trust-slope validator" refers to the mechanism that evaluates whether a biological hash sequence satisfies trust-slope continuity criteria, establishing identity through the trajectory of accumulated observations rather than comparison against stored templates.

[0139] As used herein, "unidirectional interface" refers to an architectural boundary through which candidate mutations flow from a stateless external model to the validation engine, with no return path transmitting verified agent state, cognitive domain field values, or lineage content back to the external model.

[0140] As used herein, "validation engine" refers to the mechanism that evaluates candidate mutations received through the unidirectional interface against schema constraints, policy requirements, and lineage continuity before permitting any mutation to alter semantic agent state.

[0141] As used herein, "verified agent state" refers to the committed, governance-validated state of a semantic agent as recorded in the lineage field, distinct from speculative state generated during planning graph construction or non-executing cognitive mode evaluation.

What is claimed is:

1. A system for autonomous agents with persistent cognitive state and self-regulated execution, comprising:

one or more processors; and

one or more non-transitory computer-readable media storing instructions that, when executed by the one or more processors, cause the system to:

maintain a plurality of semantic agents, each semantic agent comprising a plurality of persistent cognitive domain fields and a lineage field, the cognitive domain fields collectively encoding a behavioral disposition, a normative alignment, and an execution readiness as continuously updated persistent state, wherein each cognitive domain field is independently tracked by a cross-domain coherence engine with a current value and a trajectory over time, and wherein the semantic agent carries a complete cognitive state such that an execution substrate hosting the semantic agent validates proposed state transitions without retaining authority over the semantic agent's cognitive state;

operate the cross-domain coherence engine to maintain bidirectional feedback pathways between the cognitive domain fields, such that a state change in any one cognitive domain field propagates deterministic updates to at least one other cognitive domain field through a defined coupling function, and wherein the cross-domain coherence engine enforces that no cognitive domain field is updated in isolation from the feedback pathways;

evaluate, for each proposed mutation to a semantic agent, a composite admissibility determination that integrates signals from a plurality of the cognitive domain fields through the cross-domain coherence engine, and selectively permit, gate, or suspend the proposed mutation based on the composite admissibility determination;

transition the semantic agent to a non-executing cognitive mode when the composite admissibility determination indicates insufficient execution readiness, wherein in the non-executing cognitive mode the semantic agent continues speculative reasoning and state evaluation without committing state changes to verified agent state; and

record each proposed mutation, each composite admissibility determination, and each cognitive domain field update in the lineage field such that the complete behavioral trajectory of the semantic agent is deterministically reconstructible from the lineage field alone.

2. A computer-implemented method for governing execution of a semantic agent through cross-domain cognitive coherence, the method comprising:

maintaining the semantic agent with a persistent state, the persistent state comprising a plurality of cognitive domain fields each independently tracked by a cross-domain coherence engine and coupled through bidirectional feedback pathways, and a lineage field recording a complete behavioral history, wherein the semantic agent carries the persistent state such that the semantic agent is migratable between execution substrates while preserving behavioral continuity;

receiving a proposed mutation to the semantic agent;

propagating the proposed mutation through a cross-domain coherence engine;

computing, via the cross-domain coherence engine, for each cognitive domain field, an independent contribution to a composite evaluation of the proposed mutation;

propagating responsive updates between cognitive domain fields through the bidirectional feedback pathways;

determining, based on the composite evaluation, whether to permit the proposed mutation, gate the proposed mutation pending additional evaluation, or suspend execution of the semantic agent into a non-executing cognitive mode in which speculative reasoning continues without committing state changes;

when the semantic agent is in the non-executing cognitive mode, generating candidate alternative mutations through speculative evaluation within the cross-domain coherence engine and evaluating each candidate against the composite admissibility criteria until a candidate satisfying the composite admissibility criteria is identified or an external intervention is received; and

recording the proposed mutation, the composite evaluation, all cognitive domain field updates, and any non-executing cognitive mode transitions in the lineage field.

3. A non-transitory computer-readable medium storing instructions that, when executed by one or more processors, cause the one or more processors to:

maintain a semantic agent comprising a plurality of persistent cognitive domain fields coupled through a cross-domain coherence engine implementing bidirectional feedback pathways, and a lineage field, wherein the plurality of persistent cognitive domain fields and the lineage field collectively define a behavioral disposition for the semantic agent, and wherein

the semantic agent carries a complete cognitive state including the cross-domain coherence engine such that an execution substrate provides computational resources without retaining authority over the semantic agent's state transitions;

detect, through the cross-domain coherence engine, when a state of the semantic agent in any cognitive domain field deviates from a normative alignment defined by one or more policy constraints applicable to that cognitive domain field;

in response to detecting the deviation, propagate corrective pressure from the deviating cognitive domain field through the bidirectional feedback pathways to at least one other cognitive domain field, thereby modulating the semantic agent's behavioral disposition across coupled domains in response to the deviation;

generate, through corrective pressure propagated through the cross-domain coherence engine, a candidate mutation designed to restore normative alignment in the deviating cognitive domain field, and evaluate the candidate mutation against the composite admissibility criteria of all coupled cognitive domain fields before permitting execution; and

operate the semantic agent in a degraded mode when fewer than all cognitive domain fields are available, preserving deterministic behavioral governance through a subset of available cognitive domain fields and the bidirectional feedback pathways active between the available cognitive domain fields.

4. The system of claim 1, wherein one of the cognitive domain fields comprises an affective modulation domain that encodes a structured disposition derived from prior execution outcomes, and wherein the cross-domain coherence engine couples the affective modulation domain to at least one other cognitive domain field such that the structured disposition modulates evaluation behavior of the semantic agent.

5. The system of claim 1, wherein one of the cognitive domain fields comprises a normative alignment domain that independently tracks alignment of the semantic agent's behavior across a plurality of normative classes, and wherein the cross-domain coherence engine propagates normative deviation events as inputs to other coupled cognitive domain fields.

6. The system of claim 1, wherein one of the cognitive domain fields comprises an execution readiness domain that computes a revocable execution permission based on inputs received from

at least two other cognitive domain fields through the bidirectional feedback pathways, and wherein transitions of the semantic agent to the non-executing cognitive mode are determined by the revocable execution permission.

7. The system of claim 1, wherein one of the cognitive domain fields comprises a structural executability domain that defines boundaries of permissible execution based on computational resources, temporal constraints, and environmental conditions available to the semantic agent.

8. The system of claim 1, wherein one of the cognitive domain fields comprises a speculative planning domain that generates hypothetical future states of the semantic agent as branching evaluation structures, evaluates each branch through the cross-domain coherence engine, and selectively promotes qualifying branches into a verified execution path.

9. The system of claim 1, wherein one of the cognitive domain fields comprises a dispositional modulation domain encoding persistent behavioral parameters that modulate the branching scope and evaluation thresholds applied by the cross-domain coherence engine when the semantic agent evaluates candidate mutations.

10. The system of claim 1, further comprising an interface configured to receive proposed mutations from a stateless generative model, treat each proposed mutation as structurally untrusted, and evaluate each proposed mutation through the cross-domain coherence engine before permitting the proposed mutation to alter any cognitive domain field of the semantic agent.

11. The system of claim 10, further comprising a progressive authorization module that governs which categories of proposed mutations the stateless generative model is permitted to submit, the authorization based on accumulated evidence of prior successful mutations evaluated through the cross-domain coherence engine.

12. The system of claim 1, further comprising a semantic execution substrate configured to operate within or alongside a probabilistic inference engine and to enforce, during inference, mutation admissibility by evaluating continuity between a current state and a proposed state of the semantic agent through the cross-domain coherence engine prior to committing any inference output.

13. The system of claim 1, further comprising a biological continuity module configured to determine an identity of a human operator of the system through persistent observation of behavioral signals over a plurality of interactions, to monitor a biological state of the human operator based on the observation of behavioral signals, and to modulate one or more cognitive domain fields of the semantic agent based on detected changes in the biological state of the human operator.

14. The system of claim 1, further comprising a semantic discovery module configured to resolve a semantic query via a traversal state that includes traversing an anchor-indexed graph structure, wherein the traversal state persists across a series of traversal steps and each traversal step is evaluated for admissibility through the cross-domain coherence engine of the semantic agent initiating the traversal state.

15. The system of claim 1, further comprising a training governance module configured to evaluate training artifacts against provenance records and to permit model training at a structural depth determined by governance policy, such that the cross-domain coherence engine of agents trained on governed artifacts inherits provenance constraints from the training governance module.

16. The method of claim 2, wherein, when the cross-domain coherence engine detects that the composite evaluation indicates normative deviation in at least one cognitive domain field, the cross-domain coherence engine generates a restorative mutation designed to restore normative alignment in the deviating domain, and wherein the restorative mutation is evaluated through the cross-domain coherence engine before execution.

17. The method of claim 2, further comprising projecting the semantic agent's behavioral trajectory across a plurality of possible future mutation paths and classifying each future mutation path according to whether the future mutation path has a trajectory toward or away from normative alignment across the cognitive domain fields.

18. The method of claim 2, further comprising detecting, through the cross-domain coherence engine, that the semantic agent has entered a sustained pattern of normative deviation across one

or more cognitive domain fields, and classifying the sustained pattern as a cognitive disruption regime corresponding to an architecturally defined phase-shifted operating state.

19. The method of claim 2, wherein the cross-domain coherence engine operates as a closed-loop control system comprising a detection phase that registers deviation, a recording phase that commits the deviation to the lineage field as immutable truth, and a restoration phase that generates corrective pressure through the bidirectional feedback pathways, the detection phase, the recording phase, and the restoration phase executing sequentially for each deviation event.

20. The non-transitory computer-readable medium of claim 3, wherein the instructions, when executed by one or more processors, cause the one or more processors to, when the semantic agent operates in the degraded mode, dynamically reconfigure the bidirectional feedback pathways to route signals through available cognitive domain fields, and reduce the scope of composite admissibility determinations to reflect only the cognitive domain fields that are operationally active.

Abstract

Autonomous agents with persistent cognitive state and self-regulated execution maintain semantic agents comprising a plurality of independently tracked cognitive domain fields coupled through a cross-domain coherence engine implementing bidirectional feedback pathways as defined coupling functions. The semantic agent carries its complete cognitive state such that execution substrates validate proposed state transitions without retaining authority over the agent's cognitive state. The coherence engine propagates state changes across coupled domains through deterministic coupling functions such that a change in any one domain produces computed updates in other domains. When composite evaluation across coupled domains indicates insufficient execution readiness, the agent transitions to a non-executing cognitive mode continuing speculative evaluation without committing state changes. A three-phase corrective loop detects normative deviation, records the deviation as immutable truth, and generates corrective pressure producing candidate restorative mutations. All mutations, evaluations, and domain updates are recorded in a lineage field enabling deterministic behavioral reconstruction.