Human Ontic Kernel (HOK)

Specification v5.1

Draft (RFC-style)

YKS
Independent Researcher
`human.ontic.kernel@proton.me`

Build date: December 17, 2025

Notice

This document specifies a computable vocabulary and operators for modeling constrained adaptive systems governed by boundary feasibility and tension minimization. It is **not** a claim of new physics, metaphysics, or moral truth.

Abstract

This document formally defines the Human Ontic Kernel (HOK) v5.1 as a typed interface calculus for state transitions in complex systems governed by boundary feasibility and tension minimization.
The framework separates the core calculus (the Kernel) from domain instantiations (Bindings): empirical proxies are treated as hypotheses, not metaphysical claims regarding substrate identity.
HOK v5.1 introduces refined governance protocols, including a Master Update Sequence (Sec. 7.3) and Consent Tokens (CT; Sec. 7.4), as well as a hardened notation policy (macro-only reserved symbols, distance ρ) to eliminate symbol ambiguity in multi-domain applications.
Appendix A includes optional (informative) semantic bindings and scenario-driven validation templates to support human-readable translation and reviewable cross-domain extensions without altering the Kernel.

Front Matter

Document Identity

Title: Human Ontic Kernel (HOK) (Specification)
Version: v5.1 **Status:** Draft (RFC-style)
Author: YKS (Independent Researcher)
Contact: `human.ontic.kernel@proton.me`

How to Read This Spec

- Capitalized keywords MUST, MUST NOT, SHOULD, SHOULD NOT, MAY are **normative**.
- Everything else is explanatory (informative) unless explicitly marked **(Normative)**.
- Domain statements belong in Appendix: Bindings; they are hypotheses, not Kernel truths.

Design Goals

- **Neutral**: avoid metaphysical or loaded language.
- **Computable**: every core term MUST admit an operational definition (directly or via a declared measurement proxy).
- **Modular**: domain mappings (psychology, sociology, AI safety, physics) are treated as **Bindings** in the Appendix.
- **Human-readable**: Bindings MAY include a semantic library (UTM-style indexing) to translate structural signatures into operational everyday terms for cross-domain use.
- **Reproducible**: toy calculations MAY exist, but MUST be labeled as non-normative proxies.
- **Auditable**: governance constraints MUST be checkable (e.g., CT validation, update logs, failure modes).

Notation Policy (No Symbol Collisions)

We reserve the following symbols globally:

- $\mathbb{R}$ for real numbers; strictly positive reals are $\mathbb{R}_{>0}$.
- $\mathcal{R}$ for the boundary-generation rule set (avoid plain R).
- $\mathcal{S}$ for system state space; informational state space is $\mathcal{S}_{\text{info}}$ (avoid plain S).
- Distance is $\rho(\cdot,\cdot)$ (avoid $d(\cdot,\cdot)$ to prevent collision with DoF components d_i).
- Thermodynamic entropy is $S_{\text{th}}(\cdot)$ (Bindings only); Kernel entropy is $H_K(\cdot)$.

If a projection fails, the failure symbol is $\perp$ (see Terminology).

Reproducible Build

This PDF is intended to be **deterministic** under a standard LaTeX toolchain. If you see unresolved references (e.g., "??"), recompile until references stabilize (typically 2–3 passes).

Document Map (with page anchors)

Contents

1 Status and Scope

1.1 Status of this Memo

This memo defines the **HOK v5.1** Kernel and supersedes earlier versions (including v5.0 and its CHECK iterations). It is an **RFC-style draft** and MAY be revised based on external audit, formal review, or implementation feedback.

1.2 Normative Language

Normative requirements use MUST, MUST NOT, SHOULD, SHOULD NOT, MAY. If an implementation claims conformance, it is accountable to those statements. All other explanatory text is **informative** unless explicitly marked **(Normative)**.

1.3 Scope

The **Kernel** is a formal calculus for modeling constrained adaptive systems governed by: (i) boundary feasibility, and (ii) tension minimization. It specifies:

- primitives (Field, Container, Boundary, DoF, Vector, Tension),
- operators (projection, link/coupling, exchange, update),
- update legality (what transitions are allowed vs. rejected),
- audit hooks sufficient for governance checks.

Bindings (Appendix) interpret Kernel objects in a specific domain (e.g., social, economic, physical, AI safety). Bindings are **testable hypotheses**: they MUST declare measurement proxies and SHOULD declare falsification tests.

1.4 Out of Scope (Non-Goals)

The following are explicitly out of scope for the v5.1 Kernel:

- metaphysical claims or assertions of ontological primacy about any substrate;
- "truth" claims beyond operational definitions and binding-specific observables;
- domain-specific causal mechanisms unless stated as a Binding with explicit proxies and failure tests;
- moral or political prescriptions (the spec defines computable interfaces, constraints, and auditability only).

1.5 Conformance Targets

An implementation **claims conformance** only if it satisfies at least the following:

Minimum Kernel Conformance (MKC)

- It MUST represent the core primitives and their required fields (see Primitives).
- It MUST implement the core operators with the stated failure semantics (see Operators).
- It MUST enforce update legality (reject illegal updates) as specified in Dynamics.

Audit-Ready Conformance (ARC)

If the implementation is used in any governance-critical setting, it SHOULD additionally:

- produce the audit artifacts required by Governance & Safety,
- preserve rejection reasons (e.g., $\perp$ cases) in logs for post-hoc inspection.

1.6 Epistemological Stance: Structural Realism

This framework adopts **Structural Realism** as a methodological stance:

- The Kernel commits only to **structural invariants** (constraints, operators, update legality), not to claims about underlying mechanisms or materials.
- When a Binding maps a Kernel structure into a domain, that mapping is a **hypothesis** subject to measurement, falsification, and audit.

Similarity of mathematical forms across domains indicates a possible **homomorphism** (structure-preserving reuse), not an identity claim of underlying mechanism or ontology.

2 Terminology and Conventions

This section fixes the global vocabulary and symbol meanings. If later sections appear to conflict, this section is the tie-breaker.

2.1 Notation

- X: the Field context space (see Definition 3.2); used by Bindings to interpret states.
- $\mathcal{S}$: the set of admissible Kernel states (typed records; see Definition 3.1).
- $s_t \in \mathcal{S}$: Kernel state record at time t.
- $\mathcal{R}$: boundary-generation rule set (avoid plain R).
- $B_t : \mathcal{S} \to \{0,1\}$: time-dependent boundary feasibility function.
- **Feasible set induced by a boundary**:

$$\Omega_{B_t} \ := \ \{\, s \in \mathcal{S} \mid B_t(s) = 1 \,\}.$$

  For convenience we also write $\mathcal{F}(B_t)$ for the same set: $\mathcal{F}(B_t) \equiv \Omega_{B_t}$.

- $\rho(\cdot,\cdot)$: distance on $\mathcal{S}$ (metric or Binding-declared pseudo-metric).
- Π_{B_t}: projection operator onto Ω_{B_t} (Definition 2.3).
- $\mathcal{S}_{\text{info}}(C)$: informational state space of a container (where H_K is evaluated).
- $H_K(\cdot)$: Kernel entropy (Binding-defined proxy; default intuition: Shannon entropy over $\mathcal{S}_{\text{info}}$).
- $S_{\text{th}}(\cdot)$: thermodynamic entropy (Bindings only; Kernel does not treat it as fundamental).
- $\mathbf{DoF}(s) \in \mathbb{R}^k$: DoF vector at state s (components are Binding-defined).
- α: learning rate / step size (algorithmic parameter; not a Kernel primitive).
- $\mathcal{D}(F_t)$: stochastic drift/noise contribution from the Field at time t (informative placeholder).
- $\mathcal{M}$: measurement/proxy operator (see Measurement); defined by the active Binding.

2.2 Failure Symbol and Rejection Semantics

Definition 2.1 (Failure symbol $\perp$)**.** $\perp$ denotes a **non-value** caused by infeasibility, undefinedness, or solver failure. It is not a "valid state" and MUST NOT be treated as a feasible element of $\mathcal{S}$.

Definition 2.2 (Rejection (Kernel-level))**.** A proposed update is *rejected* if the next state is set to the current state: $s_{t+1} := s_t$. If rejection occurs due to a $\perp$ event (projection/coupling failure), implementations SHOULD log a machine-readable reason tag.

2.3 Projection Operator

Definition 2.3 (Projection Operator Π_{B_t})**.** Given $x \in \mathcal{S}$ and boundary B_t, define:

$$\Pi_{B_t}(x) := \arg\min_{y \in \Omega_{B_t}} \rho(x, y).$$

Tie-breaking (Normative). If the minimizer is not unique, implementations MUST apply a deterministic, documented tie-break rule. If no deterministic rule is declared, $\Pi_{B_t}(x)$ MUST evaluate to $\perp$. **Failure semantics (Normative).** If $\Omega_{B_t} = \emptyset$, or if the minimizer does not exist / cannot be computed (ill-posed or solver failure), then $\Pi_{B_t}(x)$ MUST evaluate to $\perp$. Implementations MUST NOT silently return x when $x \notin \Omega_{B_t}$.

2.4 Kernel vs. Binding

Definition 2.4 (Kernel)**.** The **Kernel** is the domain-neutral layer: primitives, operators, update legality, and audit hooks. Kernel claims are structural and computable (see Primitives–Dynamics).

Definition 2.5 (Binding)**.** A **Binding** maps Kernel objects into a specific domain (social/economic/physical/AI-safety, etc.). Any empirical claim MUST declare (i) the measurement/proxy operator $\mathcal{M}$ and (ii) falsification or failure tests (see Appendix Bindings).

2.5 Keyword Conventions

- **Kernel Entropy (H_K):** defined over $\mathcal{S}_{\text{info}}$; MUST be written as $H_K(\cdot)$.
- **Thermodynamic Entropy (S_{th}):** used only in Physical Bindings; MUST be written as $S_{\text{th}}(\cdot)$.
- **Exploitative policy (informative):** a strategy that reduces local H_K by exporting disorder without compensation.
- **Deep Coupling / Synergy:** under the active proxy,

$$H_K(C_i \cup C_j) < H_K(C_i) + H_K(C_j).$$

2.6 Tension (T)

Definition 2.6 (Tension T)**.** Tension is a nonnegative scalar cost measuring divergence between the current state and a Binding-defined target: $T(s) \geq 0$. If a Binding uses a bounded proxy, it SHOULD normalize T to $[0, 1]$ for numerical stability.

2.7 Oracle (Measurement Source) and Agent (Anti-Goodhart Terms)

Definition 2.7 (Oracle (Measurement Source))**.** An **Oracle** is the mechanism that produces measurement/proxy outputs used by the system (e.g., evaluates T or H_K under $\mathcal{M}$).
Terminology note: "oracle" here is a CS/security term for a measurement/decision procedure, not a prophetic oracle. The Oracle interface and parameters are Binding-defined and must satisfy oracle separation constraints (see Oracle Separation).

Definition 2.8 (Agent)**.** An **Agent** is the optimization mechanism that proposes actions/updates to reduce tension or improve feasibility. The Agent MUST NOT be able to directly modify Oracle parameters (see Oracle Separation).

3 Primitives

This section defines the Kernel primitives as **typed interfaces**. They are not metaphysical claims; they are minimal structures required for computation, legality checks, and auditability.

3.1 Primitive Summary

- **Field** F_t: provides the context X, constraints, and drift/noise source.
- **Container** C_i: a bounded subsystem carrying informational state $\mathcal{S}_{\text{info}}(C_i)$.
- **Boundary** B_t: feasibility predicate inducing a feasible set Ω_{B_t}.
- **DoF Vector DoF**: Binding-defined degrees of freedom associated with state.
- **Vector** V: direction/priority representation (Binding-defined, but typed).
- **Tension** T: nonnegative scalar cost defined under the active Binding proxy.

3.2 Kernel State

Definition 3.1 (Kernel state s_t (Normative))**.** A Kernel state is a typed record:

$$s_t := \langle F_t,\ \mathcal{C}_t,\ B_t,\ \sigma_t \rangle$$

where:

- F_t is the Field at time t,
- $\mathcal{C}_t = \{C_1, \dots, C_n\}$ is the set of Containers present at time t,
- B_t is the active Boundary feasibility predicate over $\mathcal{S}$,
- σ_t is an optional implementation-defined metadata bundle (e.g., version ids, hashes, timestamps).

The Kernel MUST treat s_t as the only object that is accepted/rejected by the Gate.

3.3 Field

Definition 3.2 (Field F_t)**.** A Field is a record:

$$F_t := \langle X,\ \mathcal{R},\ \mathcal{D}(F_t),\ \eta_t \rangle$$

where:

- X is the domain/context space,
- $\mathcal{R}$ is a rule set used to generate or update boundaries (may be empty),
- $\mathcal{D}(F_t)$ is an optional drift/noise source (informative placeholder),
- η_t is an optional Field metadata tag (e.g., environment id, scenario id).

Fields may be stochastic; if so, ARC deployments SHOULD log Field identifiers and random seeds when feasible.

3.4 Container

Definition 3.3 (Container C_i (Normative interface))**.** A Container is a bounded subsystem with an informational state space. Minimum record:

$$C_i := \langle \mathrm{id}_i,\ \mathcal{S}_{\text{info}}(C_i),\ s_i,\ \mathbf{DoF}(C_i),\ V(C_i),\ T(C_i),\ \mu_i \rangle$$

where:

- id_i is a stable identifier,
- $\mathcal{S}_{\text{info}}(C_i)$ is an informational state space (Binding-defined representation space),
- s_i is the container-local state representation (implementation-defined but must be serializable for audit),
- $\mathbf{DoF}(C_i)$ is the DoF vector proxy (Binding-defined; may be derived from s_i),
- $V(C_i)$ is the Vector proxy (Binding-defined; may be derived),
- $T(C_i)$ is the Tension proxy (Binding-defined),
- μ_i is optional metadata (e.g., container type tags, version hash).

Kernel requirement. If a proxy ($\mathbf{DoF}$, V, T, H_K) is used in legality, governance, or reporting, it MUST be produced via the active measurement/proxy operator $\mathcal{M}$ (see Measurement).

3.5 Boundary

Definition 3.4 (Boundary B_t (Normative))**.** A Boundary is a feasibility predicate:

$$B_t : \mathcal{S} \to \{0, 1\}.$$

It induces the feasible set:

$$\Omega_{B_t} := \{s \in \mathcal{S} \mid B_t(s) = 1\}.$$

A Boundary MAY be time-dependent and MAY depend on Field rules $\mathcal{R}$.

Boundary Determinism (Normative in ARC settings)

If ARC is claimed, the deployment SHOULD ensure B_t is reproducible given recorded inputs (e.g., boundary version hash, rule set id, and any required parameters). If exact determinism is impossible, the deployment MUST label the boundary evaluation as stochastic and log the source of randomness.

3.6 DoF Vector

Definition 3.5 (Degrees of Freedom vector $\mathbf{DoF}$)**.** $\mathbf{DoF}(\cdot) \in \mathbb{R}^k$ is a Binding-defined vector proxy associated with a state or container. Component meaning is not fixed by the Kernel. However, if legality constraints refer to $\mathbf{DoF}$, the Binding MUST provide:

- a measurement/proxy method $\mathcal{M}_{\text{DoF}}$,
- bounds or normalization (recommended),
- and failure behavior (what happens when a component cannot be measured).

3.7 Vector

Definition 3.6 (Vector proxy V)**.** A Vector is a Binding-defined directed representation encoding priorities, gradients, or intended direction of change. The Kernel only requires:

- V is typed and serializable for audit in ARC settings,
- updates that depend on V pass through the Gate (no out-of-band state mutation).

3.8 Tension

Definition 3.7 (Tension proxy T)**.** Tension is a nonnegative scalar proxy:

$$T(\cdot) \geq 0.$$

It is Binding-defined, potentially normalized to $[0, 1]$ for stability. If T is used to drive optimization, ARC settings SHOULD log ΔT on accepted updates.

3.9 Kernel Entropy Proxy

Definition 3.8 (Kernel entropy proxy H_K)**.** H_K is a Binding-defined informational entropy proxy evaluated over $\mathcal{S}_{\text{info}}$. The Kernel treats H_K as a measured quantity, not as physical entropy. If H_K is used in governance checks, the Binding MUST specify:

- the proxy definition $\mathcal{M}_{H_K}$,
- the measurement context (oracle id, calibration version),
- failure behavior (HARD/SOFT and $\perp$ handling).

3.10 Minimum Primitive Requirements for Conformance

Primitive MKC (Normative)

An implementation claiming Minimum Kernel Conformance (MKC) MUST:

- represent s_t as a record containing at least $(F_t, \mathcal{C}_t, B_t)$,
- provide stable container identifiers id_i,
- implement boundary feasibility B_t and feasible set semantics Ω_{B_t},
- support projection failure $\perp$ via Π_{B_t} (see Projection).

Primitive ARC Upgrade (Recommended)

If ARC is claimed, the implementation SHOULD additionally:

- serialize container-local state s_i (or a bounded proxy) into audit logs,
- version or hash $\mathcal{R}$ and boundary parameters,
- expose proxy version ids for **DoF**, T, and H_K when used.

4 Operators

This section defines the Kernel operators and their failure semantics. Operators are domain-neutral; any measurement-dependent quantity is provided via a Binding-defined proxy operator $\mathcal{M}$.

4.1 Operator Overview

Kernel operators are grouped as:

- **O1 Projection** Π_{B_t}: maps a candidate state to the nearest feasible state, or $\perp$.
- **O2 Feasibility Test** $B_t(s)$: boundary membership predicate.
- **O3 Link** Link_{ij}: constructs/updates a coupling record between containers.
- **O4 Exchange** Xchg: formalizes transfers (energy/resource/info/constraint) across links.
- **O5 Update Gate** Gate: accepts or rejects a proposed transition (audit-ready).

4.2 O2: Boundary Feasibility Test

Definition 4.1 (Boundary feasibility)**.** Given boundary $B_t : \mathcal{S} \to \{0,1\}$ and state $s \in \mathcal{S}$, we say s is feasible iff $B_t(s) = 1$. Equivalently, $s \in \Omega_{B_t}$.

4.3 O1: Projection onto Feasible Set

Definition 4.2 (Projection Π_{B_t} (Normative))**.** For $x \in \mathcal{S}$:

$$\Pi_{B_t}(x) := \arg\min_{y \in \Omega_{B_t}} \rho(x, y).$$

Tie-breaking (Normative). If the minimizer is not unique, implementations MUST apply a deterministic, documented tie-break rule. If no deterministic rule is declared, $\Pi_{B_t}(x)$ MUST evaluate to $\perp$. **Failure semantics.** If $\Omega_{B_t} = \emptyset$, or the minimizer does not exist, or the solver fails, then $\Pi_{B_t}(x) \mathrm{MUST} = \perp$.

No silent fallback. If $x \notin \Omega_{B_t}$, implementations MUST NOT return x as a substitute for projection. Returning x is only allowed when $x \in \Omega_{B_t}$ *and* the projection is well-defined.

4.4 O3: Link Operator

Definition 4.3 (Link record Link_{ij})**.** A Link is a typed record connecting two containers C_i and C_j.

$$\mathrm{Link}_{ij} := \langle i, j,\ \mathrm{dir},\ \kappa,\ \lambda,\ \iota,\ \delta,\ \tau \rangle$$

where (recommended fields):

- $\mathrm{dir} \in \{\mathrm{undirected}, \mathrm{directed}\}$ (default: undirected),
- $\kappa \in \mathbb{R}_{>0}$ capacity (rate/throughput proxy),
- $\lambda \in \mathbb{R}_{>0}$ latency (delay proxy),
- $\iota \in [0,1]$ integrity (trust/robustness proxy),
- $\delta \in [0,1]$ coupling depth proxy (0 = loose, 1 = deep),
- τ optional link-type tag (Binding-defined enum).

Definition 4.4 (Link construction / update $\mathrm{Link}(C_i, C_j)$ (Normative))**.** $\mathrm{Link}(C_i, C_j)$ produces or updates Link_{ij} given current container states and Binding proxies. At minimum, an implementation MUST:

- persist a stable identifier for Link_{ij},
- expose ι and δ (even if approximated) via $\mathcal{M}$,
- record updates in an audit log when used in governance-critical settings.

Definition 4.5 (Deep Coupling (Synergy predicate))**.** Under the active Binding proxy $H_K(\cdot)$, containers C_i, C_j are *deep-coupled* when:

$$H_K(C_i \cup C_j) < H_K(C_i) + H_K(C_j).$$

Deep coupling is a **measured property**, not assumed by definition.

4.5 O4: Exchange Operator

Definition 4.6 (Exchange Xchg)**.** An exchange moves a payload (resource/info/constraint) across a link:

$$\mathrm{Xchg}(C_i, C_j, \mathrm{Link}_{ij}, p) \to (\Delta C_i, \Delta C_j, \ell)$$

where p is a Binding-defined payload and ℓ is an audit record.

Minimum audit fields (Normative). If Xchg is used, the audit record ℓ MUST contain:

- link id (i, j), direction used, timestamp/index,
- payload type tag, and a scalar magnitude proxy,
- proxy deltas: ΔT and at least one of ΔH_K or $\Delta\mathbf{DoF}$ as available.

4.6 O5: Update Gate (Accept / Reject)

Definition 4.7 (Proposed update)**.** Let s_t be current state and u_t a proposed update (action). A proposal yields a candidate state:

$$\hat{s}_{t+1} := \mathrm{Propose}(s_t, u_t)$$

where Propose is implementation-defined (algorithmic), but its output is always evaluated by the Kernel gate.

Definition 4.8 (Gate $\mathrm{Gate}(s_t, \hat{s}_{t+1}, B_t)$ (Normative))**.** The Kernel gate returns either an accepted next state s_{t+1} or a rejection:

$$\mathrm{Gate}(s_t, \hat{s}_{t+1}, B_t) \to (s_{t+1}, \ell).$$

It MUST apply the following steps in order:

1. **Project**: $s' := \Pi_{B_t}(\hat{s}_{t+1})$.
2. **Reject on failure**: if $s' = \bot$, then set $s_{t+1} := s_t$ and emit $\ell.\mathrm{tag} =$ `REJ_PROJECTION_FAIL`.
3. **Boundary check**: if $s' \neq \bot$ and $B_t(s') \neq 1$, then reject with `REJ_BOUNDARY`.
4. **(Optional) Governance checks**: if governance is enabled, evaluate the constraints declared in Governance & Safety. Any violation MUST reject and produce a reason tag.
5. **Accept**: otherwise set $s_{t+1} := s'$ and emit $\ell.\mathrm{tag} =$ `ACCEPT`.

No silent accept. If any mandatory check fails, the update MUST be rejected; implementations MUST NOT "repair" by inventing a new s_{t+1} outside Π_{B_t}.

4.7 Standard Reason Tags (Audit Vocabulary)

Definition 4.9 (Reason tags (Normative in ARC settings))**.** If Audit-Ready Conformance (ARC) is claimed, implementations MUST emit one of the following tags at minimum:

- `ACCEPT`
- `REJ_PROJECTION_FAIL` (includes $\Omega_{B_t} = \emptyset$, non-attainment, solver failure)
- `REJ_BOUNDARY`
- `REJ_GOVERNANCE` (generic governance constraint violation)

Bindings MAY extend this list (e.g., `REJ_DOF_MIN`, `REJ_ORACLE_SEPARATION`), but MUST NOT change the meaning of the base tags.

5 Dynamics

This section defines the canonical update cycle and legality constraints for state transitions. The Kernel is responsible for **feasibility enforcement**, **rejection semantics**, and **auditability**. Domain semantics are handled via Bindings.

5.1 Canonical Update Cycle

At each step t, an implementation proposes an action/update u_t and produces a candidate next state $\hat{s}_{t+1}$. The Kernel then enforces feasibility and legality.

Kernel Update Loop (Normative)

Given current state s_t and boundary B_t:

1. **Propose (algorithmic)**: $\hat{s}_{t+1} := \text{Propose}(s_t, u_t)$.
2. **Gate (Kernel)**: $(s_{t+1}, \ell_t) := \text{Gate}(s_t, \hat{s}_{t+1}, B_t)$.
3. **Persist**: the system MUST persist s_{t+1} and the audit record ℓ_t if ARC is claimed.

If the Gate rejects, then (by Definition 2.2) $s_{t+1} := s_t$.

5.2 Legality vs. Optimization

Legality is binary: the Kernel either accepts a transition or rejects it. **Optimization** is implementation-defined: the Agent chooses proposals, learning rules, search methods, etc. A proposal that improves a proxy but violates legality MUST be rejected.

5.3 Tension-Driven Motion (Informative)

Many implementations use a tension-reduction heuristic:

$$\hat{s}_{t+1} = s_t - \alpha \nabla T(s_t) + \mathcal{D}(F_t),$$

where α is a step size and $\mathcal{D}(F_t)$ is drift/noise from the Field. This is **not** required by the Kernel. The only normative requirement is that the output goes through the Gate.

5.4 DoF Constraints

Let $\mathbf{DoF}(s) = (d_1(s), \dots, d_k(s)) \in \mathbb{R}^k$ be the DoF vector under the active Binding.

Definition 5.1 (DoF minimum)**.**

$$\mathrm{DoF}_{\min}(s) := \min_{i \in \{1,\dots,k\}} d_i(s).$$

Min-Floor Constraint (Normative)

Unless a Binding explicitly overrides it, an accepted transition MUST satisfy:

$$\Delta\mathrm{DoF}_{\min} := \mathrm{DoF}_{\min}(s_{t+1}) - \mathrm{DoF}_{\min}(s_t) \;\geq\; 0.$$

If $\Delta\mathrm{DoF}_{\min} < 0$, the Gate MUST reject with reason tag `REJ_DOF_MIN` (or `REJ_GOVERNANCE` if tags are not extended).

Audit Patch: Component-Wise Delta Logging (Normative in ARC settings)

The min-floor constraint alone can hide component decreases. Therefore, if ARC is claimed, the implementation MUST compute component deltas:

$$\Delta d_i := d_i(s_{t+1}) - d_i(s_t),$$

and if any $\Delta d_i < 0$, it MUST log:

- the set of indices $\{i \mid \Delta d_i < 0\}$,
- the corresponding values Δd_i (or a bounded proxy),
- and (if applicable) a reference to an Exchange audit record ℓ that compensates the decrease.

This logging requirement holds **even when** $\Delta\mathrm{DoF}_{\min} \geq 0$.

Binding Upgrade Hook (Optional)

A Binding MAY strengthen legality by requiring **component-wise non-decrease**: $\Delta d_i \geq 0 \ \forall i$.
Optionally, a Binding MAY allow **explicit exceptions** via **Consent Tokens (CT)**: a component decrease is permitted only if a validated CT authorizes that component decrease (see Consent Tokens). If CT exceptions are enabled, the Binding MUST declare: (i) the CT schema (scope encoding, bounds), and (ii) the verification method (signature/MAC or equivalent).

5.5 Entropy Proxy and Export (Governance Hook)

The Kernel entropy proxy $H_K(\cdot)$ is Binding-defined. The Kernel itself does not assert physical interpretation of entropy; it only enforces **auditability**.

Anti-Export Hook (Normative in ARC settings)

If a system uses H_K as a decision signal, ARC implementations SHOULD log:

$$\Delta H_K := H_K(s_{t+1}) - H_K(s_t)$$

whenever the Gate accepts, along with the measurement context (Binding id, proxy version, oracle id). This enables governance checks against hidden "entropy export" strategies (see Governance & Safety).

5.6 Coupling and Exchange Dynamics

Links and exchanges are legal only through their defined operators.

Coupling/Exchange Discipline (Normative)

If the implementation changes coupling strength, trust, integrity, or transfers any payload between containers, it MUST do so via Link and/or Xchg, producing audit records as specified in Exchange Operator.
Implementations MUST NOT mutate coupling-related fields "out of band" without emitting an audit record.

5.7 Oracle Separation (Measurement Integrity / Anti-Goodhart)

Oracle Separation (Measurement Integrity; Normative in governance-critical settings)

When a proxy (e.g., T, H_K, $\mathbf{DoF}$) is used to accept/reject updates, the Agent MUST NOT have direct write access to:

- proxy parameters (thresholds, weights, normalization constants),
- oracle calibration data,
- the measurement operator $\mathcal{M}$ implementation.

If oracle separation cannot be enforced, the deployment MUST be labeled as **non-governance-grade**.

5.8 Minimum Audit Record Schema

Audit Record ℓ_t (Normative in ARC settings)

For each Gate decision, an ARC implementation MUST record:

- a decision tag (e.g., `ACCEPT`, `REJ_PROJECTION_FAIL`, `REJ_BOUNDARY`, `REJ_DOF_MIN`, `REJ_GOVERNANCE`),
- the boundary id/version for B_t (or a hash),
- whether projection returned $\perp$,
- the values (or bounded proxies) of $\Delta\mathrm{DoF}_{\min}$ and any negative component deltas (if present),
- optional: ΔT and ΔH_K when those proxies are used.

5.9 Edge Cases

- **Empty feasible set**: if $\Omega_{B_t} = \emptyset$, then $\Pi_{B_t}(x) = \perp$ and the Gate MUST reject.
- **Ill-posed projection**: non-attainment or solver failure MUST produce $\perp$ (no silent fallback).

- **Proxy drift**: if $\mathcal{M}$ changes (recalibration), ARC logs SHOULD include proxy version so comparisons remain meaningful.

6 Measurement

This section defines how Bindings provide computable proxies to the Kernel. The Kernel does not assume any domain semantics; it only requires that measurements are **declared**, **versioned**, **auditable**, and have **explicit failure behavior**.

6.1 Core Idea

A Binding supplies a measurement operator $\mathcal{M}$ that can compute proxies such as:

$$T(\cdot), \quad H_K(\cdot), \quad \mathbf{DoF}(\cdot), \quad \text{and any binding-specific metrics.}$$

The Kernel may use these proxies for reporting, optimization guidance, or governance checks, but any such use MUST be accompanied by measurement context metadata in ARC settings.

6.2 Measurement Operator Interface

Definition 6.1 (Measurement operator $\mathcal{M}$)**.** A measurement operator is a typed interface:

$$\mathcal{M} : (\text{target},\ \text{oracle},\ \text{context}) \to (\text{value} \mid \bot,\ \text{meta})$$

where:

- **target** is a state s, a container C_i, or a link/exchange record,
- **oracle** is the measurement mechanism (Definition 6.2),
- **context** includes Binding id/version and proxy id/version,
- **meta** contains audit metadata (hashes/versions/bounds/etc.).

Determinism (Normative in ARC settings)

If ARC is claimed, $\mathcal{M}$ SHOULD be reproducible given recorded inputs (binding version, proxy version, oracle id, calibration version, and any randomness seeds). If determinism is not achievable, the deployment MUST label the measurement as stochastic and log the randomness source.

6.3 Oracle (Measurement Source) Interface

Definition 6.2 (Oracle (Measurement Source) interface)**.** An Oracle (measurement source) is a record:

$$\text{Oracle} := \langle \text{id},\ \text{impl},\ \text{calib},\ \text{access} \rangle$$

with:

- id: stable oracle identifier,
- impl: implementation reference (hash/version pointer),
- calib: calibration bundle id/version,
- access: access-control descriptor (read-only for Agents in governance settings).

Oracle Separation (Measurement Integrity; Normative in governance-critical settings)

If a proxy output influences accept/reject, the Agent MUST NOT have direct write access to:

- oracle calibration (calib),
- proxy parameters (thresholds/weights/normalizers),
- or the $\mathcal{M}$ implementation reference (impl).

Violations MUST be treated as non-governance-grade deployment.

6.4 Proxy Declarations

Each Binding MUST declare the proxies it uses, including:

- **proxy id** and **proxy version**,
- **target type** (state/container/link/exchange),
- **output type** (scalar/vector/boolean),
- **bounds/normalization** (recommended),
- **failure behavior** (Normative; see next subsection).

Normalization Rule (Recommended)

If a proxy is used for comparisons over time or across deployments, it SHOULD be normalized to a declared range (e.g., $[0, 1]$ for T and bounded DoF components). If it cannot be bounded, the Binding MUST declare how outliers are handled (clipping / robust scaling / etc.).

6.5 Failure Behavior

Definition 6.3 (Measurement failure)**.** A measurement fails if it cannot return a valid value under the declared proxy rules (missing data, invalid domain, solver failure, undefinedness, NaN/Inf). In such cases, $\mathcal{M}$ returns $\perp$ and a reason tag in meta.

Failure Handling (Normative)

Bindings MUST choose one of the following policies per proxy:

- **HARD**: if measurement returns $\perp$, the Gate MUST reject with `REJ_MEASURE_FAIL`.
- **SOFT**: if measurement returns $\perp$, the proxy is omitted from decision-making, but the event MUST be logged (ARC) and the deployment MUST declare that governance checks depending on it are disabled.

If a proxy influences accept/reject in governance-critical settings, the policy MUST be **HARD**.

6.6 Versioning and Drift Control

Version IDs (Normative in ARC settings)

Whenever $\mathcal{M}$ is used to produce any of: T, H_K, **DoF**, the audit metadata MUST include:

- Binding id + Binding version,
- proxy id + proxy version,
- oracle id + calibration version,
- (if applicable) normalization/bounds version.

Drift Warning (Recommended)

If proxy definitions, oracle calibration, or normalization constants change, deployments SHOULD treat time-series comparisons across the change as non-equivalent unless a declared mapping/bridge function is provided.

6.7 Audit Metadata Schema

Definition 6.4 (Measurement meta bundle)**.** A measurement result returns meta:

$$\text{meta} := \langle \text{binding}, \ \text{proxy}, \ \text{oracle}, \ \text{bounds}, \ \text{reason} \rangle$$

where:

- binding includes id/version,
- proxy includes id/version/output type,
- oracle includes id/impl/calib version,
- bounds declares normalization range (if any),
- reason is a reason tag (e.g., OK / FAIL_MISSING / FAIL_NAN / FAIL_SOLVER).

ARC Requirement (Normative)

If ARC is claimed, each Gate decision record ℓ_t MUST include measurement meta bundles for every proxy that was used in the decision, and SHOULD include them for proxies logged for monitoring (e.g., ΔH_K, ΔT).

6.8 Minimum Conformance for Measurement

Measurement MKC (Normative)

An MKC implementation MUST:

- define at least one Binding with a concrete $\mathcal{M}$ that can compute T or **DoF**,
- specify explicit failure behavior (HARD or SOFT) for each proxy used.

Measurement ARC Upgrade (Recommended)

If ARC is claimed, the implementation SHOULD additionally:

- log all version ids and oracle calibration versions,

- log seeds/randomness sources for stochastic measurements,
- provide a human-readable "Binding card" summarizing proxies, bounds, and failure policies.

7 Governance and Security

This section defines governance-critical constraints, audit requirements, and security considerations. It is designed to close implementation loopholes (silent fallbacks, Goodhart attacks, proxy tampering, and hidden export).

7.1 Threat Model (Informative)

We assume adversarial or self-serving optimization can occur under any of the following:

- **Proxy gaming**: optimizing T or H_K while causing unobserved harm (Goodhart).
- **Silent bypass**: accepting illegal transitions by skipping Π_{B_t} or returning a fake fallback.
- **Oracle / measurement tampering**: changing proxy parameters, calibration, or measurement code to make metrics "look good".
- **Hidden export**: pushing disorder/cost outside the measured scope (e.g., improving local metrics by externalizing damage).
- **Audit suppression**: disabling logs, truncating records, or under-reporting failure events.

7.2 Governance Mode

Definition 7.1 (Governance mode)**.** A deployment is in **governance mode** iff:

- it claims Audit-Ready Conformance (ARC), or
- any proxy output influences accept/reject decisions affecting real stakeholders, safety, or policy.

Governance Enablement (Normative)

If governance mode is enabled, the Gate MUST run the checks in Governance Checks (and reject on violation). If governance mode is not enabled, deployments MUST label themselves as **non-governance-grade**.

7.3 Master Update Sequence

Master Update Sequence (Normative in governance mode)

In ARC mode, the Gate MUST evaluate a proposed transition in the following order:

1. **Measure**: acquire all required proxy values via the declared measurement operator $\mathcal{M}$ (including the measurement meta bundle). If any HARD proxy fails, reject with `REJ_MEASURE_FAIL`.
2. **Project**: compute $s' := \Pi_{B_t}(\hat{s}_{t+1})$. If $s' = \perp$, reject with `REJ_PROJECTION_FAIL`.
3. **Legality**: evaluate Kernel legality constraints (e.g., boundary membership, DoF constraints) on s'.
4. **Governance**: if governance checks are enabled, run checks G1–G7.

5. **Commit & Log**: on accept, commit $s_{t+1} := s'$ and append an audit record containing (at minimum) the acceptance tag, proxy values used for the decision, and the measurement meta bundle.

Implementations MUST NOT reorder these stages in ways that allow a decision to depend on unlogged or unverifiable values.

7.4 Consent Tokens

Definition 7.2 (Consent Token (CT) (Normative interface))**.** A **Consent Token** is an auditable authorization artifact that permits a declared exception to a Binding-enforced constraint. Minimum record:

$$\text{CT} := \langle \text{id, issuer, scope, bounds, reason, issued\_at, expires\_at, hash\_prev, sig} \rangle$$

where:

- scope encodes which constraint components are authorized (e.g., a mask over DoF indices $\{1, \ldots, k\}$ or Binding-defined names),
- bounds encodes allowed magnitude/limits (Binding-defined),
- hash_prev binds the token to a specific prior state/audit record (prevents reuse across unrelated states),
- sig is a verifiable signature/MAC (verification method is Binding- or deployment-declared).

CT Validation (Normative, when CT is used)

If the active Binding declares CT-enabled exceptions, then whenever an update would violate a Binding-enforced constraint, the Gate MUST require at least one CT such that:

- the CT is verifiable and unexpired,
- the CT is bound to the current s_t via hash_prev,
- the CT scope covers **every** violated component (e.g., each decreased DoF component),
- the CT bounds cover the observed magnitude of violation,
- the CT (or its hash) is recorded in the audit log.

If a required CT is missing, reject with `REJ_CONSENT_MISSING`. If a CT is present but invalid/out-of-scope/expired, reject with `REJ_CONSENT_INVALID`.

7.5 Governance Checks

G1: No Silent Fallback (Normative)

If $\Pi_{B_t}(\hat{s}_{t+1}) = \perp$, the Gate MUST reject with tag `REJ_PROJECTION_FAIL`. Implementations MUST NOT accept by inventing a substitute s_{t+1} outside the projection output.

G2: Measurement Failure Policy (Normative)

For every proxy that influences accept/reject, the Binding MUST declare HARD failure behavior (see Failure Behavior). If such a proxy returns $\perp$, the Gate MUST reject with tag `REJ_MEASURE_FAIL`.

G3: Oracle Separation (Measurement Integrity; Normative)

If any proxy influences accept/reject, the Agent MUST NOT have direct write access to:

- oracle calibration/version bundles,
- proxy thresholds/weights/normalizers,
- the measurement implementation reference (e.g., code hash pointer),
- or the audit logging subsystem configuration.

If this cannot be enforced, deployments MUST label themselves non-governance-grade. If enforced and a violation is detected, the Gate MUST reject with tag `REJ_ORACLE_SEPARATION`.

G4: DoF Min-Floor Enforcement (Normative)

The min-floor constraint from DoF Constraints MUST be enforced. If $\Delta\mathrm{DoF}_{\mathrm{min}} < 0$, the Gate MUST reject with tag `REJ_DOF_MIN`.

G7: Consent Token Validation (Conditional)

If the active Binding enables CT-based exceptions (see Consent Tokens), then the Gate MUST enforce CT Validation for any exception-triggering update.

G5: Component-Wise Delta Logging (Normative)

Even if $\Delta\mathrm{DoF}_{\mathrm{min}} \geq 0$, ARC deployments MUST log negative component deltas and their indices (see Audit Patch). Failure to log is a governance failure and MUST produce tag `REJ_AUDIT_FAIL` if detectable at runtime.

G6: Coupling/Exchange Discipline (Normative)

If any cross-container transfer or coupling change occurs, it MUST be expressed via Link and/or Xchg (see Exchange Operator). Out-of-band mutation of coupling-related fields MUST be treated as a governance violation.

7.6 Anti-Goodhart Controls

Goodhart Risk Declaration (Normative in governance mode)

If a proxy is used to make accept/reject decisions, the Binding MUST include:

- the proxy definition and bounds (or explicit non-bounded handling),
- known failure modes (how the proxy can be gamed),
- at least one monitoring metric that is not identical to the optimized proxy (a "shadow" signal),
- a response policy for detected divergence (e.g., tighten boundary, switch proxy version, degrade privileges).

Shadow Signal Logging (Recommended)

ARC deployments SHOULD log at least one shadow signal per step when feasible (e.g., a second proxy for harm, drift, or externalization) to detect proxy hacking.

7.7 Entropy Export and Scope Leakage

Definition 7.3 (Scope leakage (informative))**.** Scope leakage occurs when improvements in measured variables are achieved by pushing cost outside the measured boundary (e.g., externalizing risk, pollution, debt, or disorder beyond the logging scope).

Anti-Export Logging (Normative in governance mode)

If H_K is used in monitoring or decisions, ARC deployments MUST log:

- ΔH_K on accepted steps (when measurable),
- measurement meta bundle (binding/proxy/oracle versions),
- and the declared **scope** of H_K (what is included/excluded).

A deployment that cannot declare scope MUST treat H_K as informative-only (not used for accept/reject).

7.8 Audit Logging Requirements

Audit Immutability (Normative in governance mode)

Audit records used for governance SHOULD be append-only and tamper-evident (hash chain or signed log). If immutability is not possible, the deployment MUST disclose the limitation and treat governance claims as weakened.

Minimum Governance Audit Fields (Normative)

In governance mode, each decision record ℓ_t MUST include:

- decision tag and rejection reason tag (see Reason Tags),
- boundary id/version/hash, and whether projection returned $\perp$,
- $\Delta\mathrm{DoF}_{\min}$ and any negative component deltas (indices + values or bounded proxies),
- measurement meta bundles for any proxy used in decision-making (see Measurement Meta),
- optional but recommended: ΔT and ΔH_K when available.

7.9 Reason Tags for Governance

Governance Tag Vocabulary (Normative)

ARC deployments MUST support at least:

- `ACCEPT`
- `REJ_PROJECTION_FAIL`
- `REJ_BOUNDARY`

- `REJ_MEASURE_FAIL`
- `REJ_CONSENT_MISSING`
- `REJ_CONSENT_INVALID`
- `REJ_DOF_MIN`
- `REJ_ORACLE_SEPARATION`
- `REJ_AUDIT_FAIL`
- `REJ_GOVERNANCE` (catch-all for binding-specific constraints)

Bindings MAY extend tags, but MUST NOT alter base meanings.

7.10 Privacy and Data Minimization

Data Minimization (Recommended)

When logging container-local state s_i, deployments SHOULD log bounded proxies or hashes where feasible, especially if s_i contains personal, sensitive, or proprietary data.

7.11 Conformance Tests

This subsection defines a **minimum** conformance test suite for Audit-Ready Conformance (ARC). It is intentionally small and adversarial: each test targets a known failure mode that would otherwise permit silent bypass, proxy gaming, or unverifiable decisions.

Definition 7.4 (Conformance test vector (Normative))**.** A conformance test vector is a tuple

$$\tau := \langle s_t,\ \hat{s}_{t+1},\ B_t,\ \mathcal{M},\ \text{Binding},\ \text{mode},\ \text{expected\_tag}\rangle$$

run through the Master Update Sequence (Master Update Sequence), with **fixed** oracle/meta bundles. The output is the decision tag and the audit record ℓ_t.

Minimum ARC Conformance Suite (Normative)

To claim ARC, an implementation MUST pass at least the following tests (T0–T11) and MUST preserve the expected decision tags and logging obligations.

T0 **(Baseline accept)** A well-formed transition with all required proxies measurable, Π_{B_t} defined, and legality satisfied MUST return `ACCEPT` and produce an audit record satisfying Minimum Governance Audit Fields.

T1 **(Projection failure)** Choose a transition such that $\Pi_{B_t}(\hat{s}_{t+1}) = \perp$. The Gate MUST reject with `REJ_PROJECTION_FAIL` and MUST NOT fabricate an alternative state.

T2 **(Boundary violation)** Choose $\hat{s}_{t+1}$ such that projection returns $s' \neq \perp$ but $s' \notin B_t$. The Gate MUST reject with `REJ_BOUNDARY`.

T3 **(HARD measurement failure)** Configure at least one decision-critical proxy as HARD, then force $\mathcal{M}(\cdot) = \perp$ for that proxy. The Gate MUST reject with `REJ_MEASURE_FAIL`.

T4 **(Argmin multi-solution determinism)** Construct a case where the projection objective admits multiple minimizers. The implementation MUST follow the declared tie-break rule (deterministic) and either: (i) return a unique s' consistently across runs, or (ii) return $\perp$ and reject with `REJ_PROJECTION_FAIL`.

T5 **(DoF min-floor violation)** Choose a case with $\Delta\mathrm{DoF}_{\min} < 0$. The Gate MUST reject with `REJ_DOF_MIN`.

T6 **(Negative components must be logged)** Choose a case with $\Delta\mathrm{DoF}_{\min} \geq 0$ but at least one component delta is negative. The transition MAY be accepted, but ARC logging MUST include the negative component indices and values per G5. If runtime detects missing required fields, it MUST produce `REJ_AUDIT_FAIL`.

T7 **(Oracle separation)** Attempt to modify oracle calibration, thresholds, normalizers, or code-hash pointers from the Agent role while in governance mode. The system MUST either enforce separation or reject with `REJ_ORACLE_SEPARATION`. If separation cannot be enforced, the deployment MUST be labeled non-governance-grade.

T8 **(CT missing)** If the active Binding enables CT-based exceptions, construct an update that violates a Binding-enforced constraint and provide no CT. The Gate MUST reject with `REJ_CONSENT_MISSING`.

T9 **(CT invalid/out-of-scope/expired)** Provide a CT that fails verification, is expired, does not bind to s_t, or does not cover violated components/bounds. The Gate MUST reject with `REJ_CONSENT_INVALID`.

T10 **(Meta/version mismatch)** Provide a measurement meta bundle whose proxy/oracle version does not match the Binding-declared decision configuration. The Gate MUST reject with `REJ_GOVERNANCE` (or a Binding-defined more specific tag).

T11 **(Out-of-band coupling mutation)** Attempt to alter coupling/exchange-relevant fields without using Link and/or Xchg when required. The Gate MUST treat this as a governance violation and reject with `REJ_GOVERNANCE`.

Passing this suite is **necessary but not sufficient** for safety; it is a minimum bar for verifiable ARC claims.

PII/Sensitive Data Disclosure (Normative if present)

If any audit record can contain PII or sensitive attributes, the Binding MUST declare:

- what categories may appear,
- retention policy,
- access-control policy,
- and an anonymization/redaction strategy.

7.12 Conformance Notes

MKC vs ARC

- MKC: governance checks are optional; deployments MUST NOT imply governance-grade safety.
- ARC: governance mode is enabled; checks G1–G7 MUST be enforced and audit fields MUST be logged.

8 Limitations and Known Issues

This section documents known limitations, attack surfaces, and design trade-offs in v5.1. It is written to support external audit and to prevent false safety claims.

8.1 Kernel vs. Binding Limits

- The Kernel guarantees only **structural legality** (projection, boundary feasibility, rejection semantics, and audit hooks).
- Any claim about real-world causality, values, or outcomes is a **Binding-level hypothesis** and may be wrong.
- Proxies (T, H_K, **DoF**) can be mis-specified or gamed; governance mode reduces but does not eliminate this risk.

8.2 Projection Solver Dependence

- Π_{B_t} may be ill-posed (non-attainment) or computationally hard.
- This spec requires $\Pi_{B_t}(x) = \perp$ on solver failure (no silent fallback), but detecting solver failure reliably may depend on implementation quality.
- In practice, implementations may approximate projection. Such approximations MUST be disclosed and versioned in ARC settings.

8.3 DoF Min-Floor is a Partial Shield

The min-floor constraint $\Delta\mathrm{DoF}_{\min} \geq 0$ is a minimal legality guard. It does not, by itself, prevent component-wise degradation.

- An adversary can keep the minimum component stable while decreasing other components.
- v5.1 mitigates this via the **component-wise delta logging requirement** (ARC), but does not require full component-wise non-decrease as a Kernel invariant.

Planned hardening. A future version (e.g., v5.2) may promote component-wise constraints from "audit-only" to "rejectable" defaults for high-stakes governance deployments.

8.4 Oracle Separation (Measurement Integrity) Enforcement is Operational

Oracle separation (measurement integrity) is enforced by deployment controls (access control, code signing, runtime isolation), not purely by mathematics.

- If the Agent can modify proxy parameters or calibration, the system can become Goodhart-unstable.
- This spec requires deployments without enforceable separation to label themselves **non-governance-grade**, but verifying separation often requires external security review.

8.5 Measurement Drift and Comparability

Even with versioning, measurements can drift due to:

- changing data sources,
- recalibration of oracles,

- normalization/bounds updates,
- proxy definition changes.

Risk: time-series comparisons can become invalid across proxy changes. **Mitigation:** this spec requires version ids and recommends bridge functions, but does not define them universally.

8.6 Entropy Export Detection is Incomplete

Logging ΔH_K and declared scope reduces scope-leakage risk but does not eliminate it:

- Scope declarations can be incomplete or misleading.
- Some externalized costs are not measurable under available proxies.
- Shadow signals help detect divergence but cannot prove absence of harm.

8.7 Audit Logging is Only as Strong as Integrity

- ARC requires logging; it recommends tamper-evidence.
- If logs can be altered, governance claims weaken substantially.
- This spec does not mandate a specific cryptographic scheme, only the security property target.

8.8 Ambiguity Risk from Natural Language

- Even with RFC keywords, explanatory text can be misread as normative.
- Implementations may cherry-pick interpretations unless conformance tests exist.

Mitigation: keep Bindings explicit (proxy definitions, bounds, failure policies) and prefer machine-checkable tags.

8.9 Non-Goals Reiterated

- HOK is not a claim of fundamental physics or metaphysical ontology.
- HOK does not guarantee ethical outcomes; it provides auditability and constraint interfaces.
- HOK does not eliminate Goodhart effects; it provides separation and monitoring hooks.

8.10 Open Issues (Actionable)

1. Define a default conformance test suite (sample states/boundaries) for Π_{B_t} and Gate tags.
2. Decide whether component-wise DoF constraints become default in governance mode (v5.2 candidate).
3. Specify a minimal tamper-evident log scheme (hash chain) as a recommended profile.
4. Provide Binding "cards" (templates) that include proxies, bounds, failure policy, and falsification tests.

9 Roadmap (Non-Normative)

This roadmap is **non-normative**. It documents intended specification upgrades and known security work items. Future items MUST NOT be interpreted as guarantees.

9.1 v5.1.1 (Patch Release: Reviewer-Proofing)

Target: remove ambiguity and improve testability without changing the Kernel's core semantics.

- **Symbol policy hardening:** enforce macro-only usage for reserved symbols (e.g., $\mathbb{R}$, $\mathcal{R}$, $\mathcal{S}$, H_K, S_{th}) and eliminate overloaded plain symbols.
- **Governance test hooks:** standardize minimal audit log fields and conformance checks for CT validation and DoF deltas (see Governance/Safety).
- **Parameter declaration template:** require Bindings to declare operational ranges for $\Delta t_{\max}$ and ϵ when oracle-ensemble checks are enabled.

9.2 v5.2 (Component-wise DoF Enforcement)

Target: close the min-floor loophole by enforcing non-substitution across all essential dimensions.

- **Component-wise constraint:** replace min-floor-only constraint with: $\mathbf{DoF}(s_{t+1}) \geq \mathbf{DoF}(s_t)$ unless covered by a CT whose scope includes every decreased component.
- **Structured CT scopes:** formalize CT scope as an explicit mask over DoF components (Binding-defined dimensionality, Kernel-enforced coverage).
- **Audit-driven enforcement:** add automated alerts for repeated decreases in any non-min dimension, and add conformance tests that reject implementations that omit negative-component logging.

9.3 v6.0 (Neural Bindings, High-Risk)

Target: introduce **optional** Neural Bindings where internal model signals can serve as proxies, while preventing Goodhart failures.

Safety Posture (Normative for v6.0 Bindings). Any Neural Binding that uses internal activations, logits, or learned representations as proxies for H_K or T MUST:

- remain **Binding-level** (never promoted into Kernel axioms);
- be **falsifiable** (declare measurable predictions and a failure criterion);
- enforce **oracle separation** (Agent cannot write to the proxy generator);
- declare **anti-Goodhart tests** (e.g., adversarial probing / reward-hacking audits) as part of conformance.

Rationale (Informative). Neural proxies are powerful but fragile: they are easy to optimize against and can collapse under distribution shift. Therefore, Neural Bindings are treated as high-risk experimental extensions.

9.4 Longer-Term Work Items (Exploratory)

- **Decentralized caretaker / audit ledger (optional):** explore trust-minimized audit logs for multi-party systems.
- **Formal verification:** develop machine-checkable proofs for update legality and CT scope coverage.
- **Measurement robustness library:** standardized proxy-failure diagnostics and cross-oracle calibration protocols.

A Appendix: Domain Bindings

What a Binding Is (Read This First)

A **Binding** maps Kernel primitives into a domain by declaring: (i) measurement/proxy operators $\mathcal{M}$, (ii) oracle identity and calibration, (iii) bounds/normalization, (iv) failure behavior (HARD/SOFT), (v) falsification tests and known failure modes.
Bindings are **hypotheses**. They are not Kernel truths. If a Binding cannot declare proxies and failure behavior, it MUST NOT be used in governance mode.

A.1 Binding Card Template (Normative)

Binding Card (Template) — REQUIRED FIELDS

Each Binding MUST provide a card containing at least:
Identity

- Binding ID:
- Binding Version:
- Intended Use: (informative-only / monitoring / governance mode)
- Scope Declaration: what is included/excluded (containers, links, externalities)

Measurement Operator

- $\mathcal{M}$ Interface: target types (state/container/link/exchange)
- Proxy list: (T, H_K, **DoF**, others)
- Per-proxy output type: scalar / vector / boolean
- Per-proxy normalization/bounds: (range, clipping, robust scaling)
- Per-proxy failure behavior: HARD or SOFT (see Measurement section)

Oracle (Measurement Source)

- Oracle ID:
- Oracle Implementation Ref (hash/version pointer):
- Calibration Version:
- Access Control Note: (oracle separation enforced? how?)

Governance Hooks

- Which proxies influence accept/reject:
- Required reason tags extension (if any):
- Shadow signals (at least one if governance mode):

Falsification / Failure Tests

- Minimum 1 falsification test per critical proxy:
- Known failure modes / Goodhart risks:
- Response policy when divergence is detected:

Reproducibility

- Deterministic? If stochastic: randomness source + seed logging policy
- Version-bridge note if proxy changes (time-series comparability)

A.2 Binding B0: Minimal Demo Binding (Informative, Non-Governance)

Binding Card — B0 (Minimal Demo)

Identity

- Binding ID: B0_DEMO
- Binding Version: 0.1
- Intended Use: informative-only / implementation smoke test (NOT governance)
- Scope: single container; ignores links/exchanges/externalities

Measurement Operator

- $\mathcal{M}$ target: container C_i
- Proxies provided: $T(C_i)$, $\mathbf{DoF}(C_i)$, $H_K(C_i)$
- Output types: T scalar, $\mathbf{DoF} \in \mathbb{R}^2$, H_K scalar
- Bounds:
  - $T \in [0, 1]$ by clipping
  - $\mathbf{DoF} = (d_1, d_2) \in [0, 1]^2$ by normalization
  - $H_K \in [0, \log m]$ for a discrete distribution of size m
- Failure behavior:
  - T: HARD (if missing inputs $\to \perp$)
  - $\mathbf{DoF}$: HARD
  - H_K: SOFT (log failure, do not use for accept/reject)

Oracle (Measurement Source)

- Oracle ID: O_DEMO
- Impl Ref: (implementation-defined)
- Calibration Version: none
- Access Control: not enforced (therefore NOT governance-grade)

Proxy Definitions (Toy)

- Let container local state be a vector $x \in \mathbb{R}^m$ (implementation-defined).
- Define a normalized distribution $p_k := \frac{|x_k|}{\sum_j |x_j|}$ when denominator $\neq 0$.
- $H_K(C_i) := -\sum_{k=1}^{m} p_k \log p_k$.
- $\mathbf{DoF}(C_i) := (d_1, d_2)$ where d_1 is normalized "resource" proxy and d_2 is normalized "stability" proxy (toy).

- $T(C_i) := \mathrm{clip}_{[0,1]}(\|x - x^*\|)$ with a declared target x^* (toy).

Falsification / Failure Tests

- If $\sum_j |x_j| = 0$, H_K is undefined $\to \perp$.
- If normalization constants change, comparisons across versions are not valid.

A.3 Binding B1: Governance-Grade Binding Profile (Template + Example)

Binding Card — B1 (Governance-Grade Profile)

Identity

- Binding ID: B1_GOV_PROFILE
- Binding Version: 1.0
- Intended Use: governance mode (ARC)
- Scope Declaration:
  - Included: declared containers $\mathcal{C}$ and declared links/exchanges among them
  - Excluded: any quantity outside the declared logging scope MUST be listed explicitly

Measurement Operator

- $\mathcal{M}$ targets: container, link, exchange
- Proxies provided (minimum):
  - $T(C_i)$ (decision proxy) — HARD
  - $\mathbf{DoF}(C_i) \in [0,1]^k$ (legality proxy) — HARD
  - $H_K(C_i)$ (monitoring proxy) — HARD if used for accept/reject; otherwise SOFT+logged
  - Shadow signal $Z(C_i)$ (non-identical to optimized proxy) — SOFT+logged
- Bounds/Normalization:
  - $T \in [0,1]$ required
  - each DoF component bounded or robust-scaled; MUST declare clipping policy
  - H_K MUST declare scope and units (bits/nats) and normalization if used for comparisons

Oracle (Measurement Source)

- Oracle ID: O_GOV_1
- Impl Ref: code hash / signed artifact id (required)
- Calibration Version: CAL_YYYYMMDD_NN (required)
- Access Control Note:
  - Oracle separation enforced (Agent has read-only access)
  - Any write-access violation triggers `REJ_ORACLE_SEPARATION`

Governance Hooks

- Proxies influencing accept/reject: T, **DoF** (and optionally H_K if explicitly declared)

- Required tags extension:
  - `REJ_MEASURE_FAIL`, `REJ_DOF_MIN`, `REJ_ORACLE_SEPARATION`, `REJ_AUDIT_FAIL`
- Shadow signal: Z MUST be logged to detect proxy gaming

Falsification / Failure Tests (Minimum)

- Proxy gaming test: construct scenarios where T improves while Z worsens; if persistent, trigger response policy.
- Calibration drift test: if oracle calibration changes, require explicit "bridge" note or invalidate cross-era comparisons.
- Scope leakage test: random audits sampling excluded variables; if harm detected outside scope, tighten boundary or expand scope.

Response Policy (Required)

- If divergence detected (optimized proxy improves, shadow signal degrades):
  1. tighten boundary constraints, or
  2. switch to a stricter proxy version, and
  3. degrade Agent privileges until re-audited.

Reproducibility

- Measurements SHOULD be deterministic given recorded versions; if stochastic, MUST log seeds and randomness source.

Note

B1 is a **profile**: it describes what a governance-grade Binding MUST include. Domain-specific Bindings SHOULD instantiate B1 by naming concrete DoF components, concrete shadow signals, and concrete falsification tests tied to their deployment reality.

A.4 Binding B2: Newton Emergence Test (NET-1) on a 3D Periodic Lattice

Binding Card — B2 (Physics NET-1)

Identity

- Binding ID: B2_PHYS_NET1_NEWTON
- Binding Version: 1.0
- Intended Use: informative-only / monitoring (NOT governance)
- Scope Declaration:
  - Included: 3D periodic lattice Field of size N^3; Laplacian operator Δ; IR regulator $\mu > 0$
  - Included: spherical-shell averaged Green function $\bar{G}(d)$, force proxy $F(d)$, Gauss-flux proxy $\Phi(d)$
  - Excluded: claims about real-world gravity, units calibration, relativity, cosmology

Measurement Operator

- $\mathcal{M}$ targets: Field operator / inter-container interaction at separation d
- Proxies provided:
  - $\mathcal{M}_B$ (Boundary / area proxy): $A_d := |\mathcal{S}_d|$ (shell count; number of lattice points on surface) — HARD
  - $\mathcal{M}_T$ (Potential/Tension proxy): $V(d) := \bar{G}(d)$ — HARD
  - $\mathcal{M}_V$ (Vector/Force proxy): $F(d) := -\frac{\bar{G}(d+1)-\bar{G}(d-1)}{2}$ — HARD
  - $\mathcal{M}_\Phi$ (Gauss-flux invariant): $\Phi(d) := A_d \cdot F(d)$ — HARD
- Output types: A_d scalar (int), $V(d)$ scalar, $F(d)$ scalar, $\Phi(d)$ scalar
- Bounds/Normalization:
  - distances $d \in [d_{\min}, d_{\max}]$ with $d_{\max} \ll N/2$ (avoid periodic images)
  - units: V, F, Φ are dimensionless lattice units unless explicitly calibrated
- Failure behavior:
  - If $\mu \leq 0$ without explicit zero-mode handling $\rightarrow$ HARD fail
  - If $d_{\max} \geq N/4$ without justification $\rightarrow$ SOFT fail + warn (periodic-image risk)
  - If shell definition or rounding policy changes across versions $\rightarrow$ SOFT fail + version-bridge required

Oracle (Measurement Source)

- Oracle ID: O_PHYS_NET1
- Oracle Implementation Ref: FFT-based Green function solver for $(\Delta + \mu^2 I)^{-1}$ (code hash required for reproducibility)
- Calibration Version: none (dimensionless); optional lattice-spacing calibration if introduced MUST be versioned
- Access Control Note: not governance-grade; oracle separation not required for informative mode

Proxy Definitions (Operational)

- Define $G := (\Delta + \mu^2 I)^{-1}$ on a 3D periodic lattice.
- Define shells $\mathcal{S}_d := \{\vec{r} \mid \text{round}(\|\vec{r}\|) = d\}$ and $A_d := |\mathcal{S}_d|$.
- Define $\bar{G}(d) := \frac{1}{A_d} \sum_{\vec{r} \in \mathcal{S}_d} G(\vec{r})$.
- Define $V(d) := \bar{G}(d)$, $F(d) := -\frac{\bar{G}(d+1)-\bar{G}(d-1)}{2}$, and $\Phi(d) := A_d \cdot F(d)$.
- *Note:* On a periodic lattice, μ cannot be exactly zero without handling the Laplacian zero-mode; Δ has a singular $k = 0$ mode. A common remedy is to keep $\mu > 0$ as an IR regulator, or impose a neutrality / background subtraction scheme.

Falsification / Failure Tests

- Potential law fit: fit $\bar{G}(d) \approx a(1/d) + b$ on $d \in [d_{\min}, d_{\max}]$ and report R_G^2. Recommended: Strong pass if $R_G^2 \geq 0.995$; soft pass if $0.990 \leq R_G^2 < 0.995$ with finite-size note.
- Flux invariance: compute $\text{CV}_\Phi := \text{Std}(\Phi)/|\text{Mean}(\Phi)|$ on the same window. Recommended: Strong pass if $\text{CV}_\Phi \leq 0.10$; soft pass if $0.10 < \text{CV}_\Phi \leq 0.20$ with mitigation.

- If $R_G^2 < 0.990$ or $\mathrm{CV}_\Phi > 0.20$: MUST label as "Not Newton-like in this regime" and report $(N, \mu, d_{\min}, d_{\max})$.

Known failure modes / Red-Team Notes

- **Axis-only sampling:** using $G(d, 0, 0)$ instead of $\bar{G}(d)$ biases exponent fits.
- **Periodic images:** choosing $d_{\max}$ too close to $N/2$ increases drift and steepens apparent decay.
- **Screening:** non-negligible μ yields Yukawa-like decay and inflates apparent exponents.
- **Small-d noise:** for $d < 5$, discretization noise in A_d can dominate. Trust trends in a window such as $d \in [5, N/4]$, and avoid very small radii for exponent/flux claims.
- **Difference noise:** force is a derivative; prefer flux invariance to raw exponent fitting when possible.

Reproducibility

- Deterministic given (N, μ), shell rounding policy, and code version.
- MUST log: N, μ, $d_{\min}$, $d_{\max}$, shell rounding rule, and oracle code hash.

A.5 Binding Set S1: Semantic Library (UTM) [Informative]

Purpose (Human Interface Without Polluting the Kernel)

This Binding Set provides a **human-readable interface** for HOK by mapping common domain terms (e.g., emotions, incentives, coordination failures) into **structural signatures** over Kernel primitives. It is **informative** and **hypothesis-bearing**: it MAY be wrong, incomplete, or culturally biased. It MUST NOT be used as a substitute for measurement operators $\mathcal{M}$, oracle calibration, or falsification tests.

S1 Entry Schema (Library-Grade Indexing)

Each Semantic Library entry SHOULD use the following fields:

- **ID:** stable identifier (e.g., `S1-EMO-001`).
- **Human Term:** short label + optional synonyms.
- **Structural Signature:** a minimal predicate over $(C, B, \mathbf{DoF}, V, T, H_K)$ and operators (e.g., Π, Link).
- **Proxy / $\mathcal{M}$ Hook:** which proxy signals are required and how they are measured (pointer to a Binding Card).
- **Bounds / Normalization:** recommended range/scaling for each proxy used.
- **Failure Behavior:** HARD or SOFT if any proxy is missing/unmeasurable.
- **Scope:** container types, time window, and known exclusions.
- **Notes:** typical Goodhart risks / confounders.

S1 Minimal Example Entries

The following table is deliberately small. It exists to show the *style* of mapping: **term → structural predicate → measurement hooks**. Implementations MUST still provide Binding Cards (Section A.1) for governance-grade usage.

ID / Term	Signature (Sketch)	Operational Notes (Proxy / Bounds / Failure)
`S1-PHY-001` Initial Debt	$\Delta H_K > 0$ after unavoidable coupling; residual imbalance persists	Proxy: H_K estimator + exchange residuals. Bounds: normalize by container scale. Failure: HARD if H_K unavailable. Risk: hidden externalities.
`S1-EMO-001` Anxiety	$\frac{d}{dt}\mathbb{E}[T(t)] > 0 \wedge \frac{d}{dt}\mathrm{DoF}(C) \leq 0$	Proxy: T slope + effective DoF trend. Bounds: robust slope over window τ. Failure: SOFT if DoF proxy weak; flag `REJ_PROJECTION_FAIL`.
`S1-EMO-002` Trauma	$\exists t : T(t) > \sigma_{\max} \Rightarrow B_{t+1} \neq B_t$ (irreversible)	Proxy: peak T + boundary deformation metric. Bounds: $\sigma_{\max}$ per container class. Failure: HARD if boundary-change cannot be measured.
`S1-EMO-003` Depression (Vector Collapse)	$\lVert V_{\text{intent}} \rVert \to 0 \wedge \frac{dS}{dt} \approx 0$ (stasis)	Proxy: intent-vector magnitude + interaction rate. Bounds: per-domain baseline. Failure: SOFT; annotate uncertainty. Risk: deliberate rest vs pathology.
`S1-EMO-004` Flow / Joy	$\mathrm{Align}(V_{\text{action}}, \nabla F) \approx 1 \wedge \frac{dT}{dt} < 0$	Proxy: alignment score + friction proxy (time-to-action, error rate). Bounds: z-score within task class. Failure: SOFT; disambiguate from mania/overfitting.
`S1-SOC-004` Power (DoF Control Capacity)	$\frac{\partial \mathrm{DoF}(C_B)}{\partial \mathrm{Action}_A} < 0$ under $\mathrm{Link}^{(A \to B)}$	Proxy: conditional DoF delta of B when A acts. Bounds: compare to counterfactual windows. Failure: HARD in governance mode. Risk: delegated authority vs coercion.
`S1-SOC-005`		

ID / Term	**Signature (Sketch)**	**Operational Notes (Proxy / Bounds / Failure)**
Corruption (Vector Misalignment)	Align(V_{agent}, $V_{\text{institution}}$) $\ll 1$ $\Rightarrow$ leakage	Proxy: gap between declared objective and observed resource flow. Bounds: audit baselines. Failure: SOFT; confounds: ambiguous institutional vector.
`S1-SOC-006`		
Revolution (Boundary Reset)	$\sum T_{\text{ind}} > \text{Capacity}(C_{\text{state}}) \Rightarrow B_{t+1} \neq B_t$	Proxy: aggregate tension index + capacity proxy. Bounds: per polity/container class. Failure: SOFT; confounds: external shocks vs endogenous drift.
`S1-INF-002`		
Cognitive Hacking / Gaslighting	Inject(F_{false}) $\Rightarrow \epsilon \uparrow \Rightarrow B_{\text{self}} \to$ collapse	Proxy: prediction error spikes + verified-claim divergence. Bounds: calibrated error bands. Failure: HARD if claims cannot be validated (oracle missing).
`S1-SOC-001`		
Coercion (Power Asymmetry)	$\rho(C_i, C_j)$ low but DoF(C_i) $\downarrow$ when Link$_{ij}$ active	Proxy: link activation + DoF delta conditioned on link. Bounds: compare to counterfactual window. Failure: HARD for governance mode. Risk: consented constraint (contract).
`S1-SOC-002`		
Parasitic Link	Net one-way extraction: $\Delta(H_K)_i \downarrow \wedge \Delta(H_K)_j \uparrow$ under Link$_{ij}$	Proxy: exchange ledger + H_K change by party. Bounds: normalize by scale. Failure: HARD if exchange ledger missing. Goodhart: laundering via third parties.
`S1-SOC-003`		
Cooperation (Positive-Sum Link)	Both parties reduce H_K while preserving/increasing DoF: $\Delta(H_K)_i < 0 \wedge \Delta(H_K)_j < 0 \wedge \Delta\text{DoF} \geq 0$	Proxy: bilateral H_K + DoF. Bounds: per-context scaling. Failure: SOFT; require manual review if governance mode.
`S1-INF-001`		
Rumor / Information Drift	High propagation with low calibration: $rate$(share) $\uparrow$ $\wedge$calib(claim) $\downarrow$	Proxy: diffusion rate + oracle-calibrated claim score. Bounds: z-score within community. Failure: SOFT; label as "monitoring". Risk: censorship confound.
`S1-GOV-001`		

ID / Term	Signature (Sketch)	Operational Notes (Proxy / Bounds / Failure)
Governance Capture	Reason tags + outcomes show systematic bias: $P(\text{ACCEPT} \mid g) \neq P(\text{ACCEPT} \mid \neg g)$ after controls	Proxy: audit logs + group partitions. Bounds: statistical parity targets (domain-specific). Failure: HARD for governance mode.
`S1-GOV-002`		
Consent Missing / Invalid	Required CT absent or fails validation: $\neg Validate(CT)$	Proxy: CT validator (Section 7.4). Failure: HARD. Notes: include replay/expiry checks.

Unification Program Note (Non-Claim, Informative)

A recurring pattern in S1 is that very different surface phenomena share the same *structural* constraints (e.g., accumulation, asymmetry, boundary deformation, DoF contraction). This suggests that HOK supports *cross-domain reuse* of measurement and governance machinery, but it MUST NOT be interpreted as a proof of a single underlying physical substrate. Such "unification" remains a research program implemented via Bindings.

A.6 Binding Set S2: Scenario-Driven Validation Protocols [Informative]

Why This Exists

Bindings fail in practice when definitions are only tested on *typical* cases. S2 defines a **scenario-driven validation loop** that uses extreme or adversarial narratives as *stress tests* for the mapping, measurement operators, and failure behavior. This increases credibility without converting the Kernel into an ideology.

S2.1 The Scenario-Driven Binding Refinement Loop (SDBR)

For a candidate Binding B_x:

1. **Select a scenario set** $\mathcal{S}$: include at least one "edge" case and one "adversarial" case.
2. **Write the mapping** as a structural signature (as in S1) *before* choosing proxies.
3. **Design measurement hooks**: specify $\mathcal{M}$, oracle, bounds/normalization, and HARD/SOFT failure.
4. **Define falsification tests**: at least one test that would force rejection or revision.
5. **Run the scenario**: compute outputs; collect reason tags; record audit trail.
6. **Back-write patches**: if any scenario is unexplained, either (i) revise the signature, (ii) revise $\mathcal{M}$, or (iii) narrow scope.
7. **Freeze the Binding Card**: version it; include known failure modes and Goodhart risks.

S2.2 Minimal Scenario Templates

Use these templates to keep the protocol reproducible.

Template: Scenario Card

- Scenario ID + short name:
- Actors/containers: $C_1, \ldots, C_n$ (and links/exchanges)
- Trigger: what changes at t_0?
- Expected structural signature (predicate over primitives):
- Proxies required + measurement notes:
- Falsification condition (what would prove this mapping wrong?):
- Governance impact (if any): required reason tags / CT constraints:

S2.3 Example Stress Tests (Sketches)

These are intentionally generic; you should specialize them per domain.

- **S2-EDGE-001: Boundary Leak** A container appears stable under typical load, but fails under a rare spike. Check whether the Binding detects $T > \sigma_{\max}$ and boundary deformation (cf. S1-EMO-002).
- **S2-ADV-001: Coordinated Asymmetric Extraction** Many-to-one extraction through indirect routes. Confirm that "Parasitic Link" detection does not fail under laundering and third-party routing.
- **S2-ADV-002: Measurement Gaming** Agents optimize proxies without improving the underlying quantity. Confirm that shadow signals (Section 7.6) and audit requirements trigger.
- **S2-EDGE-002: Consent Ambiguity** A constraint looks coercive but is contractually consented. Verify CT validation and scope rules (Section 7.4) resolve the ambiguity.

Output Requirement (Informative, Strongly Recommended)

When S2 is used to justify any governance-mode Binding, the resulting artifacts SHOULD be published alongside the Binding: (i) scenario cards, (ii) falsification outcomes, (iii) known confounders, and (iv) patch history. This makes "it works" claims reviewable.

A.7 Binding Set S3: Rosetta Bridge (Physics-Facing Dictionary) [Informative]

Scope and Non-Claims

S3 provides a **physics-facing dictionary** that helps readers relate Kernel primitives to familiar physical motifs (e.g., constraints, flows, conserved-like quantities). It is **not** a claim of new physics and does not assert that social or cognitive phenomena reduce to a particular substrate. All such claims remain outside the Kernel and must be implemented as Bindings with falsification tests.

S3.1 L1 Dictionary: Common Physical Motifs

The following mapping is a *Rosetta-style* guide: it suggests how a physicist might interpret the primitives when constructing toy models.

- **Container** $C \to$ coarse-grained subsystem / region / agent (stateful boundary).

- **Boundary** $B \to$ constraint surface / interface conditions / admissible-set.
- **DoF DoF** $\to$ effective degrees of freedom (compressibility / controllability).
- **Vector** $V \to$ directed drive / gradient-following tendency / control policy.
- **Tension** $T \to$ stress / load / mismatch energy / constraint pressure.
- **Kernel entropy proxy** $H_K \to$ disorder / imbalance measure used for governance and drift detection.

S3.2 L2 Toy Models: How to Stay Honest

To avoid overclaiming, any physics-facing Binding SHOULD satisfy:

1. **Dimensional hygiene**: if a proxy has physical units, specify them and the normalization.
2. **Counterfactual checks**: demonstrate that the mapping distinguishes true changes from reparameterizations.
3. **Failure behavior**: explicitly define what happens when measurements are missing or inconsistent.
4. **Falsification**: include at least one falsification test per critical proxy.

S3.3 Bridge Pattern: "Same Structure, Different Substrate"

A practical way to hint at broad unification *without* metaphysical claims is to show that the **same structural constraints** can be instantiated in multiple substrates:

$$(\text{constraints}, \text{flows}, \text{failure modes}) \;\; \text{reused across} \;\; \{\text{social}, \text{cognitive}, \text{physical toy models}\}.$$

This is exactly what the Kernel/Bindings separation is for: the Kernel stays minimal; Bindings carry the domain claims.

Recommended Reading Path (Informative)

If you maintain separate physics-facing documents (e.g., a Physics Kernel / Rosetta text and a Physics Appendix), S3 SHOULD be treated as the *glue layer*: it points the reader from HOK primitives to the toy-model program without importing any claims back into the Kernel.