OSC-K Kernel Lint Specification v0.2

1. Purpose

This specification defines a lint/check that verifies an Operational Semantics Contract (OSC) contains the minimum semantic kernel required to function as a contract authority artifact.

The lint evaluates semantic skeleton presence and explicitness. It does not evaluate domain correctness.

2. Scope

The OSC-K lint applies to OSC artifacts intended to serve as any of:

- Implementation guide input (for derivation)
- Test oracle (for conformance evaluation)
- Safety gate authority artifact (for refusal at commit boundaries)

The lint does not require a formal grammar and does not transform OSC into a programming language.

3. Inputs

The lint accepts:

- contract_text (required)
- contract_id (optional)
- contract_version (optional)

4. Kernel Requirements (OSC-K)

A contract satisfies OSC-K iff it explicitly provides all four kernel elements:

4.1 Decision Surface

The contract MUST define which outcomes are decision surfaces: outcomes whose interpretation changes externally observable behavior.

The contract MUST include at least one concrete DS category, drawn from:

- halt/proceed (refusal)
- accept/reject
- create/delete
- overwrite/retain
- success/failure classification

4.2 Authority Placement

The contract MUST state where truth lives for deciding desired state.

The contract MUST state at least one non-authoritative category if relevant (e.g., coordination, telemetry, cache, queue), OR explicitly state that no non-authoritative stores/channels exist within scope.

4.3 Failure Model

The contract MUST name failure classes that affect DS outcomes.

The contract MUST distinguish at least two failure causes (e.g., policy vs operational vs representation, or equivalent).

The contract MUST state at least one irreversible side effect that is forbidden on failure (e.g., no delete, no publish, no commit, no ack).

4.4 Identity Semantics

The contract MUST define canonical identity for the primary governed entities.

The contract MUST state any normalization and uniqueness expectations necessary for determinism.

5. Precision & Scope Rule

The OSC-K lint is a minimum semantic skeleton check.

The lint MUST NOT penalize contracts for additional detail, verbosity, or precision beyond the kernel.

Additional clauses, definitions, examples, domain-specific requirements, and implementation notes are permitted.

The lint MAY emit WARN or BLOCK only if additional text:

- contradicts a kernel declaration, OR
- makes a kernel element ambiguous, OR
- omits a kernel element entirely.

6. Evaluation Outcomes

The lint MUST output exactly one of:

- PASS: all four kernel elements present and explicit
- WARN: all four present but one or more kernel elements are ambiguous/vague
- BLOCK: one or more kernel elements missing OR internally contradictory

7. Report Format (Normative)

The lint MUST output a structured report containing:

- result: PASS | WARN | BLOCK
- findings: array of finding objects

Each finding MUST contain:

- kernel_element: DECISION_SURFACE | AUTHORITY_PLACEMENT | FAILURE_MODEL | IDENTITY_SEMANTICS

- status: OK | WARN | MISSING | CONTRADICTORY
- evidence: excerpt(s) from contract_text sufficient to justify the status
- required_fix: directive text describing what must be added or clarified

8. Required Fix Prompts (Templates)

If a kernel element is MISSING or WARN, required_fix MUST be derived from:

Decision Surface:

- “Define which outcomes are decision surfaces (halt/proceed, delete/retain, success/failure classes). Identify which outcomes must be determinate.”

Authority Placement:

- “State where truth lives for deciding desired state. Identify which stores/channels are non-authoritative (coordination/telemetry/cache/queue) if any.”

Failure Model:

- “Name failure classes and state what irreversible actions are forbidden on failure.”

Identity Semantics:

- “Define canonical identity for governed entities and any normalization/uniqueness rules.”

9. Non-Goals

The OSC-K lint MUST NOT:

- Judge whether the contract's design choices are good.
- Enforce specific architectures or patterns.
- Require enumerating every DS exhaustively.
- Require machine-readable clause IDs.
- Replace Attack Types triage discipline.

10. Versioning and Immutability

This specification is immutable for version v0.2.

Any change beyond typos/oopsies requires at least a 0.1 version bump.

Major semantic changes require a +1 bump.