Canonical Document Header (Informative / Non-Normative Metadata)

- Document ID: SC-CORE-000001
- Canonical Name: THE COHERENCE CORE SPECIFICATION
- Document Class: SPEC
- Version: 1.2.5
- Issuance ID: v39d
- Release Status: ISSUED
- Effective Date: 2026-03-04
- Last Updated: 2026-03-19
- Issuing Authority: Coherence Research
- DOI / Persistent ID: 10.5281/zenodo.18039635
- Supersedes: SC-CORE-000001 (Issuance ID: v39c)
- Superseded By:
- Header Schema: SC-HDR-000001 (Issuance ID: v11)
- Integrity:
  - SHA-256: 3a9b7ac0b9f2c896e7206d6892e9992ebaedab6054e83b97885ad1c55ade93d9
  - Release Tag: 2026-03-19___v39d

Document Class Extension: SPEC (Informative / Non-Normative Metadata)

SPEC::Header-Schema-Version: 0.4

SPEC::Standard: Structural Coherence

SPEC::Spec-Subclass: core-spec

SPEC::Revisability: Versioned Only

SPEC::Canonical-Dependencies: SC-HDR-000001 (v11), SC-AXIOM-000001 (v11l)

THE COHERENCE CORE SPECIFICATION

Coherence as a Universal Structural Invariant

Forced Structure Under Admissibility, Continuity, and Nonterminal Regulation

Authority and Status

This document constitutes the Core specification.

Canonical Effective-Date Posture

(Meta-Structural · Binding)

For this Core specification version, Canonical status applies only as of the Effective Date stated in the Document Header. Any earlier draft, patch, or distribution bearing this document identifier is non-canonical unless it is explicitly marked as Canonical and carries the same Version and patch-id as the Document Header.

The Core specification is the minimal, definition-closed, schema-governed specification of primitive terms, Foundational Conditions, Valid Emergence Conditions, and Admissibility Rules sufficient to determine: - bounded structural coherence, - admissibility of configurations, -

nonterminality under admissible internal drivers, - and forced emergence of non-equivalent structure.

No statement, structure, or rule is canonical unless it is defined in or derivable from this Core and expressed in compliance with this specification Term Schema.

Canonical Scope

The Core specification applies globally and independently of domain, substrate, or scale.

It specifies structural conditions of admissibility only. It makes no claims regarding existence, causation, temporality, dynamics, substance, purpose, or ontology.

All existential statements within this specification are structural in meaning, denoting admissibility and representability only.

Schema Authority

(Meta-Structural · Binding)

Kernel Terms (Definitions, Foundational Conditions, Valid Emergence Conditions, and Admissibility Rules) MUST be expressed using this specification Term Schema (T0–T6; T7 optional).

Any Kernel Term not conforming to the schema is non-canonical by definition.

Canonical Derivations (e.g., Coherence Axioms) are canonical only insofar as they are kernel-grounded, introduce no new primitives or kernel constraints, and remain projection-contained.

Canonical Derivations MUST either: - (a) be explicitly marked as non-normative derivation text, or - (b) express any binding Kernel Terms they rely on as schema-compliant Kernel Terms.

The schema is constitutive, not advisory.

Conformance and Self-Consistency

(Meta-Structural · Binding)

This Core specification is intended to be structurally checkable for self-consistency using only the criteria it defines internally.

1) Binding conformance criterion

For any purported Core term or purported canonical derivation relative to this Core specification version, the binding criterion of Canonical conformance is AR11 (this specification Conformance Characterization). Any Canonical status claim for this document or any downstream this specification document is governed by AR11.

2) Canonical and non-normative material

Authoritative classification of Canonical content is given by D0.7.

Within this document: - Canonical normative material consists of: - the binding Meta-Structural clauses explicitly labeled "(Meta-Structural · Binding)", including Schema-Level Constraints (S-#); and - the T1 Statements of schema-complete Core terms expressed under this specification Term Schema (T0–T6; T7 optional). - Canonical non-normative material consists only of text explicitly marked as a non-normative canonical derivation consistent with S-2 and AR11. - All other text is non-normative commentary for orientation only.

3) Versioning posture for public release

The Version field in the header denotes the Core specification kernel version.

Any future change that modifies: - the closed primitive set in §2, - any normative T1 statement of a Kernel Term (D···, F···, VE···, AR···), - or any binding meta-structural clause explicitly labeled "(Meta-Structural · Binding)" (including any Schema-Level Constraint S-#), MUST be published as a new Core specification version.

4) Structural validation convention (non-procedural)

A conformance evaluator validates this document by checking: - schema completeness (T0–T6; T7 optional) for every Kernel Term, - explicit dependency declarations (T3) and acyclic this specification-term dependency closure (S-5), - primitive closure and meta-vocabulary closure (S-3), - and the non-normative firewall (S-2).

This document does not provide or require an algorithmic validator; the above criteria are declarative and enforceable without procedural commitments.

Modality and Constitutive Force

(Meta-Structural · Binding)

All modal operators in this Core (MUST, MUST NOT, REQUIRED, PROHIBITED) denote constitutive structural necessity and constitutive structural impossibility relative to Core-admissibility, Core-representability, and Core-enforceability. - "MUST" means: without the stated condition, at least one of this specification's constitutive regulative targets fails (loss of containment | loss of boundedness | loss of representability | loss of enforceability), such that one or more downstream Core terms become ill-formed, undecidable, or unenforceable within this specification semantics. - "MUST NOT" means: the stated construct cannot participate as an admissibility-grounding mechanism without inducing constitutive failure of containment, boundedness, representability, or enforceability; i.e., it cannot be admitted as constitutive basis for Core-admissibility.

No statement in this Core introduces obligations, permissions, preferences, policies, intentions, purposes, or commands. All modal force is internal to constitutive admissibility and representability.

Meta–Object Separation

This specification strictly distinguishes: - Object-level constructs: configurations, relations, transformations, redistributions - Meta-structural statements: admissibility, necessity, minimality, boundedness, and enforcement

No object-level construct implies existence, causation, time, purpose, or metaphysical grounding.

Anti-Metaphysics Constraint

All Core statements are purely structural.

This specification does not describe what exists or what is real. It specifies only conditions under which structure is admissible, representable, and enforceable.

Coherence Principle (Foundational)

Coherence is a non-binary, bounded, regulatory invariant governing admissible structure.

Kernel Commitment — Substrate

This specification posits no primitive substrate. Coherence specifies structural constraints that must obtain for any structure to be admissible, representable, and enforceable. What are described in downstream domains as substrates, entities, or material bases are stabilized configurations of structural constraint, not prior foundations.

Incoherence is an admissible internal differential provided it remains bounded, local, and redistributable under this specification constraints.

All this specification enforcement proceeds by regulation and nonterminality, not by exclusion or filtering.

Closure and Versioning

(Non-Normative · Summary)

This summary is referential only. For binding closure/versioning constraints, see: - 3) Versioning posture for public release - S-3. No External Primitives - AR11. Core Conformance Characterization

Version 1.2.5 establishes a rigor-hardened authoritative kernel preserving and extending Version 1.0's constitutive intent while increasing explicit typing, dependency closure, and conformance enforceability.

1. CORE TERM SCHEMA

(Meta-Structural · Binding)

This schema governs the expression, validation, and enforcement of all Core terms.

It applies to: - Primitive Definitions - Foundational Conditions (F) - Valid Emergence Conditions (VE) - Admissibility Rules (AR)

The schema introduces no object-level structure. It regulates representation, dependency, and justification only.

Any Core term not conforming to this schema is non-canonical by definition.

T0. Term Header

(Normative · Mandatory)

Each Core term must begin with a Term Header specifying: - ID: Specification identifier (e.g. D1.4, F7, VE3, AR9) - Name: Specification term name - Category: Definition | Foundational Condition | Valid Emergence Condition | Admissibility Rule - Level: Object | Meta

A Core term MUST NOT omit or modify this header.

T1. Statement

(Normative · Object-Level if Category ≠ Meta)

Each Core term contains exactly one explicit core statement defining or regulating structure.

Permitted forms: - Definition: "X is … iff …" - Condition / Rule: "For all admissible …, it is required that …"

Constraints: - Negation MUST NOT be used as an inferential, justificatory, or exclusion-based mechanism. - Justificatory or explanatory grounding MUST NOT be used beyond structural admissibility. - Motivational, intentional, or purpose-based rationale MUST NOT be used. -

Chained implication or indirect derivation MUST NOT be used to establish admissibility. - Metaphysical or ontological claims MUST NOT be used beyond structural admissibility constraints. - Binary (true/false) coherence semantics MUST NOT be used.

This statement specifies what is regulated or preserved, not what fails.

T2. Scope

(Normative)

Each Core term must explicitly declare its scope: - Applies to: configurations | transformations | redistributions | emergences | pathways | recursions | continuations | derivations | Core terms - Range: global | relative (explicit parameters required) - Contextual limits: explicitly stated, or "none"

A Core term MUST NOT apply implicitly or globally by omission.

T3. Dependencies (Definitional)

(Normative)

Each Core term MUST explicitly declare its **definitional dependencies**: the Core terms required to state, interpret, and enforce the term's binding content.

- Format: `Depends on: {…}`
- Scope: definitional dependencies are the only dependencies that may ground constitutive force for binding clauses.
- Well-foundedness: the definitional dependency graph MUST be acyclic and well-founded (see S-5).

T3b. Audit / Reflective References (Non-definitional)

(Normative · Optional)

A Core term MAY declare **audit / reflective references** to other Core terms that it ranges over, classifies, or characterizes WITHOUT requiring their definitions to interpret this term's binding content.

- Format: `Audit references: {…}`
- Non-creativity: audit references MUST NOT introduce constitutive force or add binding requirements (see S-9).
- Cycles: audit-reference graphs MAY be cyclic. T4. Necessity Witness

(Normative) Each Core term must include a necessity witness identifying: - Regulatory loss: loss of containment | loss of boundedness | loss of representability | loss of enforceability - Downstream witness: the first Core term whose application becomes undecidable or unenforceable if this term is removed

Necessity witnesses: - are structural, not corrective, - preserve graded coherence, - never treat incoherence as inadmissible, - never employ binary validity language.

T5. Minimality / Non-Redundancy

(Normative) This section is mandatory if the term introduces any regulatory capacity not guaranteed elsewhere.

It must state explicitly one of the following: - a structural capability not regulated by any Core term not depending on this term, or - a dependency predicate not constrained elsewhere in this specification.

If minimality is asserted: - excluded alternatives must be listed explicitly, - exclusion basis must be structural.

Minimality is never implied.

T6. Admissibility Interface

(Normative)

Each Core term must declare how it participates in this specification enforcement.

This declaration is purely declarative.

Examples: - “Referenced by AR-level nonterminality enforcement” - “Constrains admissible redistribution pathways” - “Required for expressive-capacity ordering”

Constraints: - no procedures, - no algorithms, - no temporal or causal language.

This section specifies where the term is used, not how it operates.

T7. Keywords

(Optional · Non-Normative)

non-normative tags for navigation only.

Keywords introduce no constraints and carry no interpretive force.

Schema-Level Constraints

(Meta-Structural · Binding)

The following constraints apply to all Core terms without exception.

S-1. Graded Coherence Preservation

Core terms, necessity witnesses, and admissibility interfaces MUST NOT treat coherence or incoherence as binary.

All reasoning must be expressed in terms of: - boundedness, - containment, - regulation, - admissible degrees of freedom.

Clarification: This constraint governs coherence semantics (coherent/incoherent) and does not restrict binary judgments about schema well-formedness, typing, dependency closure, or conformance.

S-2. Non-Normative Firewall - Normative: T0, T1, T2, T3, T4, T5, T6 - Non-Normative: T7

Non-normative sections introduce no constraints.

S-3. No External Primitives

All symbols and predicates must be: - defined within this specification, or - explicitly meta-structural.

External primitives are prohibited.

Clarification: Adjacency (D1.1) is an irreducible object-level primitive of configuration structure; the Core specification defines coherence relative to a configuration’s given adjacency relation and does not derive or select adjacency.

Interpretation: - Within this Core specification, “symbols and predicates” refers to any formally used operator, relation, or named predicate that appears with constraining force inside a Kernel Term’s T1 statement. - Ordinary English in non-normative prose does not introduce primitives. - Any technical relation used with constraining force (e.g., “isomorphism”, “derivable”, “nullifies”) MUST either (a) be defined as a Core term, or (b) be explicitly declared as permitted meta-structural apparatus in the Meta-Structural Mathematical Vocabulary (or a designated meta-vocabulary section) of this Core specification.

S-4. No Implicit Methods

No operation, test, or transformation may be assumed.

All such entities must appear as explicitly defined Core terms.

S-5. Dependency Closure

Every **definitional dependency** (T3) must be: - explicit, - Core-internal, - and well-founded under an acyclic dependency ordering.

Circular dependency invalidates the term.

Note: Audit / reflective references (T3b) are not definitional dependencies and are not subject to acyclicity under S-5.

S-6. Constructive Admissibility and Negation Discipline

Negation operators, when used, denote structural inadmissibility or non-equivalence only.

Negation MUST NOT be employed as: - an inferential mechanism, - a proof strategy, - a justification device, - or a source of admissibility, necessity, or structure.

In particular, no Core statement permits: - derivation by contradiction, - contrapositive reasoning, - exclusion-based inference, - or proof from the failure of an alternative.

All Core admissibility is constructive. No Core statement derives admissibility, necessity, or structure from exclusion, contradiction, or the absence of competing configurations.

S-7. Dependency Non-Temporality

Dependency relations in this specification denote definitional / representational dependency only: a dependent term is ill-formed, undecidable, or unenforceable within this specification semantics without the depended-on term(s). Dependencies MUST NOT be interpreted as time, causation, dynamics, procedural sequence, or process order.

Document order is presentational only and is not constitutive. The only constitutive ordering constraint on dependencies is acyclicity and well-foundedness as stated in T3 and S-5.

Dependency Range Expansion

(Meta-Structural · Binding)

Dependency ranges of the form {Dx–Dy}, {Fx–Fy}, {VEx–VEy}, or {ARx–ARy} are permitted shorthand and are definitionally equivalent to explicit enumeration of all Core terms with identifiers from the start identifier through the end identifier inclusive.

For D-ranges, x and y MUST share the same D-major prefix (e.g., D1.*), and the expansion follows increasing numeric order of the minor component.

Dependency Semantics Clarification

(Meta-Structural · Binding)

A this specification-term dependency edge in T3 denotes a constitutive well-formedness prerequisite only:

A term A depends on term B iff removing B renders A ill-formed, undecidable, or unenforceable within this specification semantics, as defined by S-7.

Meta-structural reference, shorthand, or range-quantification over a family of Core terms (including references to {Fx–Fy}, {VEx–VEy}, {ARx–ARy}, or to "all applicable constraints of a given class") does not, by itself, introduce a this specification-term dependency edge unless the referenced term(s) are required for A's well-formedness in the sense above.

Meta-Structural Mathematical Vocabulary

(Meta-Structural · Binding)

The following representational vocabulary is explicitly permitted as Core-internal meta-structural apparatus for purposes of definition, schema governance, and admissibility characterization only. It introduces no object-level primitives, operations, tests, procedures, or algorithms.

Permitted apparatus includes: - finite set notation and membership (e.g., {…}); - finite sequences / ordered families (e.g., ⟨…⟩), length n, and indexing by natural numbers, including bounds such as $n \geq 1$; - identity and equality of Core objects where a Core term requires identity (e.g., pathway identity in D0.3); - empty set symbol $\emptyset$ as shorthand for {}; - inequality symbol $\neq$ as shorthand for "not equal" / "not identical"; - adjacency-preserving structural isomorphism (used only as a meta-structural criterion for structural equivalence as in D1.6); - partial ordering relations and order comparisons ($\leq$, $<$, $\geq$, $>$) used as ordering-only constraints (non-metric); - function notation, typing, and application used purely as meta-structural apparatus (e.g., $f : A \to B$ and f(a)); - power set notation (e.g., $\mathcal{P}(A)$) and related set-construction shorthands used meta-structurally; - set-builder / comprehension notation used meta-structurally (e.g., $\{x \in A : \varphi(x)\}$); - standard set relations and operators used meta-structurally (e.g., $\subseteq$, $\cup$); - implication / entailment arrows used meta-structurally in constraint statements (e.g., $\Rightarrow$); - satisfaction notation used meta-structurally as a shorthand for "satisfies" (e.g., $X \models P$); - meta-level carrier symbols for classes of Core objects when used only as shorthands (e.g., Config for the class of configurations); - sequence concatenation and composition used purely as definition of ordered families when source–target compatibility holds. - induced carrier sets for configurations, defined from adjacency when element-level talk is used meta-structurally: for any configuration X, $|X| := \{ u : (\exists v)\ Adj(X)(u,v) \vee Adj(X)(v,u) \}$; - induced subconfiguration / restriction notation used meta-structurally: for $S \subseteq |X|$, $X|\_S$ denotes the configuration whose adjacency is Adj(X) restricted to $S \times S$; - adjacency-preserving embeddings used meta-structurally: an embedding $e : |X| \to |Z|$ is adjacency-preserving iff $\forall u,v \in |X|$, $Adj(X)(u,v) \Leftrightarrow Adj(Z)(e(u),e(v))$; the embedded region is e(|X|) and its induced adjacency is Adj(Z) restricted to $e(|X|) \times e(|X|)$; - compatibility of induced adjacency with a source configuration used meta-structurally: the induced adjacency on the embedded region is compatible with Adj(X) iff it agrees under the embedding as above; - relation transport ("image of a relation under an adjacency-preserving isomorphism") used meta-structurally: for an adjacency-preserving bijection $f : |X| \to |Y|$ and any relation $R \subseteq |X|^{\wedge}k$, the image $f(R) := \{ (f(x1),\ldots,f(xk)) : (x1,\ldots,xk) \in R \}$.

Whenever such vocabulary appears, it is meta-structural representational apparatus and does not, by itself, introduce new Core terms or dependency edges beyond those explicitly declared in T3.

Canonical Summary (Schema)

This specification Term Schema enforces explicit definition, scope declaration, dependency validation, structural necessity, and non-redundancy while preserving coherence as a graded regulatory invariant. It governs this specification representation and justification without introducing object-level structure or binary semantics.

S-8. Binding Reachability (Definitional Support)

Any symbol, predicate, relation, or construct used with **constitutive force** in a binding clause (T1–T6) MUST be supported by definitional dependencies alone. Concretely: - every binding use MUST be definitional-reachable via T3 from the term that uses it; and - any use that is only reachable through T3b is non-normative.

Effect: prevents "dependency downgrading" (moving a required definition into audit references) and prevents reflective references from silently acquiring binding force.

S-9. Audit Non-Creativity

Audit / reflective references (T3b) may only **classify, report, or characterize** existing Core material. They MUST NOT: - introduce new primitives, operations, or constraints, - be required to interpret or enforce a binding clause, - or add or modify constitutive force.

Effect: allows reflexive closure (self-audit) without bootstrapping loops in the definitional graph.

S-10. Schema Token Lock

(Meta-Structural · Binding)

All Kernel Terms and Canonical · Binding governance blocks MUST use exactly the canonical schema tokens defined by this specification.

1) Schema block labels are literal. The following labels are constitutive and MUST appear exactly as written when the corresponding block is present:

- T0. Term Header
- T1. Statement
- T2. Scope
- T3. Dependencies (Definitional)
- T3b. Audit / Reflective References (Non-definitional)
- T4. Necessity Witness
- T5. Minimality / Non-Redundancy
- T6. Admissibility Interface
- T7. Keywords (optional; non-normative)

2) Dependency field tokens are literal:

- "Depends on: {…}" for T3
- "Audit references: {…}" for T3b No alternative spellings are permitted in Canonical · Binding material.

3) Header enumerations are closed:

- Category: Definition | Foundational Condition | Valid Emergence Condition | Admissibility Rule
- Level: Object | Meta Any other Category/Level values render the term non-canonical under AR11.

Effect: prevents schema drift and makes conformance mechanizable by literal matching.

S-11. Variable and Typing Discipline

(Meta-Structural · Binding)

This specification fixes the following variable roles and typing conventions whenever variables appear in Binding clauses (T1–T6):

1) Configurations: X, Y, Z, … range over configurations (D1.0).
2) Transformations: T, R, … range over transformations (D1.7).
3) Emergence: An emergence is a transformation $T : X \to Y$ designated as emergence per D1.8. Any binding quantification "over emergences" MUST quantify over transformations of the form $T : X \to Y$, and MUST treat Y as the target configuration, not as the emergence object.
4) No type overloading: a symbol MUST NOT be used as both a configuration variable and a transformation variable within the same binding clause-set unless explicitly re-bound with a fresh name and stated typing.

Effect: prevents emergence-typing drift and stabilizes D0.16 instantiation.

S-12. Binding Vocabulary Firewall

(Meta-Structural · Binding)

Any symbol, predicate, operator, or technical construct used with constitutive force in any Binding clause (T1–T6) MUST be either: (a) a Core term that is definitional-reachable under S-8, or (b) explicitly admitted meta-structural apparatus in the Meta-Structural Mathematical Vocabulary section, or (c) purely expository English that introduces no new predicate/operation/structure.

Binding convention: English technical phrases become constitutive only when formalized. Any English technical phrase intended to behave as a predicate/operation/structure MUST either: (a) be introduced as a schema-complete Definition term, or (b) be explicitly added to the admitted meta-structural apparatus list, with a statement that it introduces no new inferential power beyond existing Core terms.

If a phrase appears in Binding clauses and is not covered by (a) or (b), then it MUST be treated as non-constitutive gloss only; if a binding clause depends on it for enforcement, the clause is unenforceable under D0.17 and non-conforming under AR11.

Effect: prevents stealth primitive leakage in binding text.

S-13. Semantic Reflexive Closure (Publication-Grade)

(Meta-Structural · Binding)

For purposes of this specification suite, "Publication-Grade" denotes a governance classification applied to a purported canonical module or kernel set, and introduces no object-level construct.

A purported canonical module M is Publication-Grade relative to this Core specification version only if M is semantically reflexively closed as defined by D0.31 (Semantic Reflexive Closure

Criterion), evidenced by an explicit binding lexicon ledger $\mathcal{L}$_M (D0.30) whose entries are definitional-reachable under S-8 and whose unresolved surface is empty.

Absent such a witness, M may still be Canonical under D0.7 and conformant under AR11, but MUST be classified as not Publication-Grade.

2. PRIMITIVE DEFINITIONS

(Object-Level · Schema-Governed · Closed Set)

This section defines the complete set of object-level primitive terms used anywhere in the Core specification.

No object-level term may appear outside this list.

D1.0 Configuration

T0. Term Header - ID: D1.0 - Name: Configuration - Category: Definition - Level: Object

T1. Statement A configuration X is a formally nameable structural object for which an adjacency relation Adj(X) is defined.

T2. Scope - Applies to: configurations - Range: global - Contextual limits: none

T3. Dependencies (Definitional) - Depends on: {}

T4. Necessity Witness - Regulatory loss: loss of representability - Downstream witness: All subsequent Core terms become ill-formed without a defined structural carrier.

T5. Minimality / Non-Redundancy This term is irreducible. No prior Core term provides a structural carrier for relations or transformations.

T6. Admissibility Interface - Serves as the carrier for all binding constraints, redistributions, and emergence rules.

T7. Keywords configuration; structure; carrier

D1.1 Adjacency

T0. Term Header - ID: D1.1 - Name: Adjacency - Category: Definition - Level: Object

T1. Statement Adjacency Adj(X) is a binary relation defined per configuration X. The carrier $|X|$ is induced by Adj(X) as specified in the Meta-Structural Mathematical Vocabulary.

T2. Scope - Applies to: configurations - Range: global - Contextual limits: defined per configuration

T3. Dependencies (Definitional) - Depends on: {D1.0}

T4. Necessity Witness - Regulatory loss: loss of containment - Downstream witness: Continuity (F1) and redistribution become unenforceable without adjacency.

T5. Minimality / Non-Redundancy Adjacency uniquely constrains locality and interaction. No other term regulates adjacency-bounded scope.

T6. Admissibility Interface - Adjacency-bounded scope of a configuration X is the induced carrier $|X|$ determined by Adj(X). - Any transformation, redistribution, or interaction described as “adjacency-bounded” is restricted to elements and pairs within this scope. - No additional structure (including topology, metric, k-hop neighborhoods, closure, reachability, connectedness, or any derived neighborhood operator) is implied by this term unless explicitly introduced

elsewhere as a Defined Term or Admitted Apparatus. - Referenced by continuity, redistribution, recursion, and coherence constraints.

T7. Keywords adjacency; locality

D1.2 Continuity

T0. Term Header - ID: D1.2 - Name: Continuity - Category: Definition - Level: Object

T1. Statement Continuity is a structural restriction-consistency condition on a transformation $T : X \to Y$ defined as follows:

T is continuous iff for every $S \subseteq |X|$ there exists $S' \subseteq |Y|$ and an adjacency-preserving isomorphism

$$\varphi_S : X|_S \cong Y|_{S'}$$

such that for all $S_1 \subseteq S_2 \subseteq |X|$, the witnesses are restriction-consistent:

$$\varphi_{S_1} = (\varphi_{S_2})\big|_{S_1}$$

(under the canonical identification induced by restriction).

No additional structure (including topological or metric continuity, limits, convergence, ε–δ conditions, differentiability, smoothness, closure, reachability, or any derived neighborhood operator) is implied by this term unless explicitly introduced elsewhere as a Defined Term or Admitted Apparatus.

T2. Scope - Applies to: transformations (including redistributions) - Range: global - Contextual limits: none

T3. Dependencies (Definitional) - Depends on: {D0.24, D0.28, D1.0, D1.1, D1.7}

T4. Necessity Witness - Regulatory loss: loss of representability under restriction - Downstream witness: Redistribution and emergence lose enforceable locality-consistent propagation constraints.

T5. Minimality / Non-Redundancy Continuity uniquely regulates rupture-by-restriction. No weaker restriction-consistency condition supports admissible redistribution without scale-local loss.

T6. Admissibility Interface - A redistribution SHALL satisfy D1.2 continuity in addition to adjacency preservation. - Used by Foundational Conditions and AR-level transformation enforcement.

T7. Keywords continuity; restriction-consistency; propagation; boundedness

D1.3 Expressive Capacity

T0. Term Header - ID: D1.3 - Name: Expressive Capacity - Category: Definition - Level: Object

T1. Statement Expressive capacity $\mathcal{E}(X)$ is not a separate metric or magnitude; it is the ordering class of X induced by adjacency-preserving embeddability. Define a preorder $\preceq$ on configurations by:

$X \preceq Y :\Longleftrightarrow \exists\, e : |X| \to |Y|$ such that e is an adjacency-preserving embedding (per the Meta-Structural Mathe

Define expressive equivalence by:

$$X \equiv_{\mathcal{E}} Y :\Longleftrightarrow (X \preceq Y) \wedge (Y \preceq X).$$

Then $\mathcal{E}(X)$ denotes the $\equiv_{\mathcal{E}}$-equivalence class of X, and comparisons $\mathcal{E}(Y) > \mathcal{E}(X)$ and $\mathcal{E}(Y) \geq \mathcal{E}(X)$ are interpreted solely as strict / non-strict comparison under the induced partial order on equivalence classes. No additional structure (including numeric magnitude semantics, metric distance, entropy/information measures, probability, measure/volume, cardinality-based interpretations, or any topology/analysis-derived meaning) is implied by $\mathcal{E}(\cdot)$ beyond ordering-only comparison unless explicitly introduced elsewhere as a Defined Term or Admitted Apparatus.

T2. Scope - Applies to: configurations - Range: global - Contextual limits: ordering only; non-metric

T3. Dependencies (Definitional) - Depends on: {D1.0, D1.1}

T4. Necessity Witness - Regulatory loss: loss of enforceability - Downstream witness: Minimal emergence and non-equivalence become undecidable without an ordering substrate that is invariant under adjacency-preserving transport.

T5. Minimality / Non-Redundancy Expressive comparison is the weakest ordering induced by the admitted transport interface; it cannot be reduced to size, count, or any assumed external metric without importing additional structure.

T6. Admissibility Interface - Used to evaluate minimal emergence and non-equivalent configurations.

T7. Keywords expressive capacity; embeddability; ordering; emergence

D1.4 Driver (Structural Differential)

T0. Term Header - ID: D1.4 - Name: Driver - Category: Definition - Level: Object

T1. Statement A driver D(X) is an internal structural differential within a configuration indicating an unresolved adjacency-compatible relational differential.

T2. Scope - Applies to: configurations - Range: relative (per configuration) - Contextual limits: internal only

T3. Dependencies (Definitional) - Depends on: {D1.0, D1.1, D1.2}

T4. Necessity Witness - Regulatory loss: loss of nonterminal enforcement - Downstream witness: AR9 cannot force emergence without admissible drivers.

T5. Minimality / Non-Redundancy No other term captures unresolved but admissible internal strain.

T6. Admissibility Interface - Used to evaluate nonterminality and forced emergence.

T7. Keywords driver; strain; nonterminality

D1.5 Redistribution

T0. Term Header - ID: D1.5 - Name: Redistribution - Category: Definition - Level: Object

T1. Statement A redistribution is a transformation that preserves adjacency and satisfies continuity (D1.2) while resolving at least one admissible driver.

T2. Scope - Applies to: transformations - Range: relative (to configurations with drivers) - Contextual limits: admissible only

T3. Dependencies (Definitional) - Depends on: {D0.24, D1.0, D1.1, D1.2, D1.4, D1.7}

T4. Necessity Witness - Regulatory loss: loss of enforceability - Downstream witness: Nonterminality cannot be operationalized.

T5. Minimality / Non-Redundancy Redistribution uniquely couples resolution with constraint preservation.

T6. Admissibility Interface - Central to AR-level emergence and continuation rules.

T7. Keywords redistribution; transformation; resolution

D1.6 Structural Equivalence

T0. Term Header - ID: D1.6 - Name: Structural Equivalence - Category: Definition - Level: Object

T1. Statement Two configurations X and Y are structurally equivalent, written $X \sim Y$, iff there exists an adjacency-preserving structural isomorphism between them.

T2. Scope - Applies to: configurations - Range: global - Contextual limits: adjacency-preserving only

T3. Dependencies (Definitional) - Depends on: {D0.24, D1.0, D1.1}

T4. Necessity Witness - Regulatory loss: loss of minimality enforcement - Downstream witness: Non-equivalent emergence becomes undefined.

T5. Minimality / Non-Redundancy Equivalence cannot be inferred from expressive capacity alone.

T6. Admissibility Interface - Used to evaluate terminality and emergence distinctness.

T7. Keywords equivalence; isomorphism; identity

D1.7 Transformation

T0. Term Header - ID: D1.7 - Name: Transformation - Category: Definition - Level: Object

T1. Statement A transformation T is a formally nameable structural object relating a source configuration X and a target configuration Y, written $T : X \to Y$, such that X and Y are configurations.

T2. Scope - Applies to: transformations - Range: global - Contextual limits: none

T3. Dependencies (Definitional) - Depends on: {D1.0}

T4. Necessity Witness - Regulatory loss: loss of representability - Downstream witness: All Foundational Conditions and Admissibility Rules quantifying over transformations become ill-formed without a defined transformation carrier.

T5. Minimality / Non-Redundancy Transformation uniquely supplies a structural carrier for $X \to Y$ used by redistribution, emergence regulation, and enforcement rules. No prior primitive provides a Core-internal object-level carrier for mapping between configurations.

T6. Admissibility Interface - Serves as the object-level carrier quantified by Foundational Conditions and Admissibility Rules applying to transformations.

T7. Keywords transformation; mapping; source; target

D1.8 Emergence

T0. Term Header - ID: D1.8 - Name: Emergence - Category: Definition - Level: Object

T1. Statement An emergence is a transformation $T : X \to Y$ designated as emergence iff $\mathcal{E}(Y) > \mathcal{E}(X)$.

T2. Scope - Applies to: emergences - Range: global - Contextual limits: ordering-only

T3. Dependencies (Definitional) - Depends on: {D1.3, D1.7}

T4. Necessity Witness - Regulatory loss: loss of enforceability - Downstream witness: Valid Emergence Conditions and Admissibility Rules regulating emergences become ill-typed without an object-level emergence carrier.

T5. Minimality / Non-Redundancy Emergence uniquely supplies an object-level carrier for the class of transformations subject to VE- and AR-level emergence regulation. No prior primitive defines emergence as a typed structural entity.

T6. Admissibility Interface - Serves as the object-level carrier constrained by VE1–VE8 and AR5–AR7.

T7. Keywords emergence; expressive capacity; ordering

D1.9 Pathway

T0. Term Header - ID: D1.9 - Name: Pathway - Category: Definition - Level: Object

T1. Statement A pathway P(X,Y) is a finite ordered family of transformations $\langle T_1, \dots, T_n \rangle$ with $n \geq 1$, such that $T_1 : X \to X_2$, $T_i : X_i \to X_{i+1}$ for $1 \leq i < n$, and $T_n : X_n \to Y$.

T2. Scope - Applies to: pathways - Range: relative (per source–target pair) - Contextual limits: defined only for configurations for which all intermediate source–target configurations are configurations

T3. Dependencies (Definitional) - Depends on: {D1.0, D1.7}

T4. Necessity Witness - Regulatory loss: loss of enforceability - Downstream witness: VE6 and AR5 become unenforceable without a Core-internal object-level pathway carrier.

T5. Minimality / Non-Redundancy Pathway uniquely supplies a Core-internal carrier for connects X to Y through admissible transformations without introducing procedures or dynamics. No prior primitive represents connectivity across successive transformations.

T6. Admissibility Interface - Serves as the object-level carrier required by VE6 and AR5 pathway-grounding constraints.

T7. Keywords pathway; chain; connectivity; admissibility

D1.10 Recursion

T0. Term Header - ID: D1.10 - Name: Recursion - Category: Definition - Level: Object

T1. Statement A recursion is a non-empty set of pathways that is closed under pathway concatenation whenever the target of one pathway is the source of another.

T2. Scope - Applies to: recursions - Range: global - Contextual limits: concatenation defined only where source–target compatibility holds

T3. Dependencies (Definitional) - Depends on: {D1.9}

T4. Necessity Witness - Regulatory loss: loss of representability - Downstream witness: Core terms regulating recursion and repeated continuation become ill-formed without a typed object-level recursion construct.

T5. Minimality / Non-Redundancy Recursion uniquely provides a Core-internal object-level carrier for repeatable continuation expressed without temporal, algorithmic, or causal language. No prior primitive regulates closure of pathway families under pathway extension.

T6. Admissibility Interface - Serves as the object-level carrier referenced wherever recursion is regulated by VE- and AR-level terms.

T7. Keywords recursion; closure; extension; composition

Closure Statement — Primitive Definitions

(Non-Normative · Summary) Primitive closure is governed by D0.4 (Primitive Closure) and the schema-level constraint S-3 (No External Primitives). This summary is referential only; see the binding terms for the closure criteria and enforcement posture. —

2A. META-STRUCTURAL DEFINITIONS

(Meta-Structural · Schema-Governed · Closed Set)

This section defines the complete set of Core-internal meta-structural definitions used to eliminate ambiguity in this specification enforcement and cross-referencing while introducing no object-level primitives.

No object-level term may be introduced in this section.

D0.0 Core Constraint Bundle

T0. Term Header - ID: D0.0 - Name: Core Constraint Bundle - Category: Definition - Level: Meta

T1. Statement This specification constraint bundle, written $\mathcal{C}$, is the finite set {F1–F14}.

T2. Scope - Applies to: configurations, transformations, redistributions, emergences, pathways, recursions - Range: global - Contextual limits: relative to this Core specification version only

T3. Dependencies (Definitional) - Depends on: {} T3b. Audit / Reflective References (Non-definitional) - Audit references: {F1–F14} - Role: D0.0 names the Foundational Condition family without depending on their internal definitions.

T4. Necessity Witness - Regulatory loss: loss of representability; loss of enforceability - Downstream witness: Any Core term referencing $\mathcal{C}$ becomes ill-formed without a Core-internal constraint-bundle definition.

T5. Minimality / Non-Redundancy This definition uniquely provides a Core-internal name for the complete Foundational Condition set used as the global coherence-regulation bundle. No prior Core term defines $\mathcal{C}$.

T6. Admissibility Interface - Provides the named constraint bundle referenced by Core terms that require a global coherence-regulation bundle without re-enumeration.

T7. Keywords constraint bundle; coherence; closure

D0.1 Core Admissibility Characterization

T0. Term Header - ID: D0.1 - Name: Core Admissibility Characterization - Category: Definition - Level: Meta

T1. Statement An object-level construct E is Core-admissible iff E is well-typed by the Primitive Definitions in §2 and every binding constraint whose scope applies to the type of E is satisfied for E, including all applicable Foundational Conditions, all applicable Valid Emergence Conditions where E is an emergence, and all applicable Admissibility Rules, as determined by the explicit scope declarations in this Core specification.

T2. Scope - Applies to: configurations, transformations, redistributions, emergences, pathways, recursions - Range: global - Contextual limits: relative to this Core specification version only; applicability is determined by the explicit "Applies to" field in T2 of each Core term

T3. Dependencies (Definitional) - Depends on: {D0.4, D0.16, D0.17, S-3, S-5, S-8} T3b. Audit / Reflective References (Non-definitional) - Audit references: {D0.0, D0.5, D0.6} - Role: D0.1 ranges over the constraint families named by D0.0/D0.5/D0.6 without making their members definitional for admissibility itself.

T4. Necessity Witness - Regulatory loss: loss of representability; loss of enforceability - Downstream witness: AR11 cannot be applied as a Core-internal conformance criterion without an explicit Core-internal characterization of "admissible".

T5. Minimality / Non-Redundancy This definition uniquely binds the predicate "admissible" to Core-internal constraint satisfaction using this specification's own scope declarations, preventing imported admissibility semantics. No prior Core term defines admissibility as a Core-internal characterization.

T6. Admissibility Interface - Governs the interpretation of "admissible" in all binding statements and enforcement clauses by binding it to Core-internal constraint satisfaction.

T7. Keywords admissibility; characterization; scope; constraint satisfaction

D0.2 Continuation

T0. Term Header - ID: D0.2 - Name: Continuation - Category: Definition - Level: Meta

T1. Statement A continuation from configuration X to configuration Y is a pathway P(X,Y) such that P(X,Y) is Core-admissible and either (a) each transformation in P(X,Y) is an admissible redistribution, or (b) P(X,Y) is an element of at least one Core-admissible recursion $\mathcal{R}$ (D1.10) under this Core specification.

T2. Scope - Applies to: pathways - Range: relative (per source–target configuration pair) - Contextual limits: defined only for configuration pairs (X,Y) for which P(X,Y) is a pathway as defined in D1.9 and is admissible under D0.1

T3. Dependencies (Definitional) - Depends on: {D1.0, D1.5, D1.9, D1.10, D0.1}

T4. Necessity Witness - Regulatory loss: loss of representability - Downstream witness: AR5 becomes ill-typed where it quantifies over continuations without a Core-internal continuation definition.

T5. Minimality / Non-Redundancy This definition uniquely provides a Core-internal typing for "continuation" as admissible pathway connection not restricted to emergent classification. No prior Core term defines continuation.

T6. Admissibility Interface - Provides the typed construct referenced by AR5 where "continuation" is regulated.

T7. Keywords continuation; pathway; admissibility

D0.3 Pathway Identity and Distinctness

T0. Term Header - ID: D0.3 - Name: Pathway Identity and Distinctness - Category: Definition - Level: Meta

T1. Statement Two pathways P and Q are identical iff they have the same length n and, for each index i with $1 \leq i \leq n$, the i-th transformation in P is the same transformation object as the i-th transformation in Q; pathways are distinct iff they are not identical.

T2. Scope - Applies to: pathways - Range: relative (per source–target configuration pair) - Contextual limits: identity and distinctness are defined only for pathways over the same source–target configuration pair

T3. Dependencies (Definitional) - Depends on: {D1.9, D1.7}

T4. Necessity Witness - Regulatory loss: loss of enforceability - Downstream witness: F9 is underdetermined where it requires more than one distinct redistribution pathway without a Core-internal distinctness criterion.

T5. Minimality / Non-Redundancy This definition uniquely supplies a Core-internal criterion for "distinct pathway" required by F9 without introducing procedures, dynamics, or external equivalence primitives.

T6. Admissibility Interface - Used wherever this specification requires non-singleton pathway availability or distinguishes multiple pathways.

T7. Keywords pathway; identity; distinctness; non-singleton

D0.4 Primitive Definition Closure

T0. Term Header - ID: D0.4 - Name: Primitive Definition Closure - Category: Definition - Level: Meta

T1. Statement The object-level primitive definitions for this Core specification version are exactly {D1.0–D1.10}; no object-level entity is Core-internal unless it is well-typed by one or more of these Primitive Definitions.

T2. Scope - Applies to: Core terms, derivations - Range: global - Contextual limits: relative to this Core specification version only

T3. Dependencies (Definitional) - Depends on: {D1.0–D1.10, S-3}

T4. Necessity Witness - Regulatory loss: loss of representability; loss of enforceability - Downstream witness: AR11 cannot be applied as a Core-internal primitive-closure criterion without a Core-internal specification of the closed primitive set.

T5. Minimality / Non-Redundancy This definition uniquely makes the primitive closure set explicit as a Core-internal object required for strict conformance evaluation, eliminating reliance on non-normative closure prose. No prior Core term specifies the primitive set as an exact closure.

T6. Admissibility Interface - Referenced by conformance evaluation (AR11) whenever primitive closure must be checked without importing external typing criteria.

T7. Keywords primitive; closure; typing; conformance

D0.5 Valid Emergence Condition Closure

T0. Term Header - ID: D0.5 - Name: Valid Emergence Condition Closure - Category: Definition - Level: Meta

T1. Statement The Valid Emergence Conditions for this Core specification version are exactly {VE1–VE8}.

T2. Scope - Applies to: Core terms, derivations - Range: global - Contextual limits: relative to this Core specification version only

T3. Dependencies (Definitional) - Depends on: {} T3b. Audit / Reflective References (Non-definitional) - Audit references: {VE1–VE8} - Role: D0.5 names the Valid Emergence Condition family without depending on their internal definitions.

T4. Necessity Witness - Regulatory loss: loss of enforceability - Downstream witness: canonical derivation conformance claims become underdetermined without a Core-internal specification of the complete VE set.

T5. Minimality / Non-Redundancy This definition uniquely makes the VE set explicit as a Core-internal closure object for strict conformance evaluation, avoiding reliance on non-normative section closure statements. No prior Core term specifies the VE set as an exact closure.

T6. Admissibility Interface - Provides this specification-internal VE set for conformance and completeness claims in derived canonical specifications.

T7. Keywords emergence; VE; closure; completeness

D0.6 Admissibility Rule Closure

T0. Term Header - ID: D0.6 - Name: Admissibility Rule Closure - Category: Definition - Level: Meta

T1. Statement The Admissibility Rules for this Core specification version are exactly {AR1–AR11}.

T2. Scope - Applies to: Core terms, derivations - Range: global - Contextual limits: relative to this Core specification version only

T3. Dependencies (Definitional) - Depends on: {} T3b. Audit / Reflective References (Non-definitional) - Audit references: {AR1–AR11} - Role: D0.6 names the Admissibility Rule family without depending on their internal definitions.

T4. Necessity Witness - Regulatory loss: loss of enforceability - Downstream witness: canonical enforcement completeness claims become underdetermined without a Core-internal specification of the complete AR set.

T5. Minimality / Non-Redundancy This definition uniquely makes the AR set explicit as a Core-internal closure object for strict conformance evaluation, avoiding reliance on non-normative section closure statements. No prior Core term specifies the AR set as an exact closure.

T6. Admissibility Interface - Provides this specification-internal AR set for conformance and completeness claims in derived canonical specifications.

T7. Keywords admissibility; AR; closure; completeness

D0.7 Canonical Material Criterion

T0. Term Header - ID: D0.7 - Name: Canonical Material Criterion - Category: Definition - Level: Meta

T1. Statement A statement in this document is Canonical iff it is either (a) a binding clause contained in a section explicitly labeled "(Meta-Structural · Binding)" (including any Schema-Level Constraint S-#), or (b) the T1 Statement of a schema-complete Core term expressed under

this specification Term Schema (T0–T6; T7 optional), or (c) a statement explicitly marked as a non-normative canonical derivation consistent with S-2 and AR11.

T2. Scope - Applies to: Core terms, binding meta-structural clauses, derivations - Range: global - Contextual limits: relative to this Core specification version only

T3. Dependencies (Definitional) - Depends on: {S-2, S-3, S-8, S-9}

T3b. Audit / Reflective References (Non-definitional) - Audit references: {AR11} - Role: D0.7 may be referenced by conformance characterizations (AR11) without depending on them.

T4. Necessity Witness - Regulatory loss: loss of enforceability - Downstream witness: Canonical status becomes ambiguous whenever binding constraints or admissibility-critical clauses appear outside schema-governed term blocks without an explicit "(Meta-Structural · Binding)" designation.

T5. Minimality / Non-Redundancy This definition uniquely fixes the constitutive boundary between enforceable binding material (binding meta-structural clauses, schema-complete term T1 statements, and explicitly marked canonical derivations) and non-normative commentary.

T6. Admissibility Interface - Governs interpretation of headings, summaries, and prose by denying enforceability unless the text is either a binding meta-structural clause, a schema-complete term T1 statement, or an explicitly marked non-normative canonical derivation.

T7. Keywords specification; canonicity; schema; binding; non-normative

D0.8 Derivability (Canonical)

T0. Term Header - ID: D0.8 - Name: Derivability (Canonical) - Category: Definition - Level: Meta

T1. Statement A configuration Y is derivable from a configuration X (within this Core specification) iff there exists a pathway P(X,Y) (D1.9) such that either (a) every transformation in P(X,Y) is a redistribution (D1.5), or (b) P(X,Y) is an element of at least one recursion $\mathcal{R}$ (D1.10).

T2. Scope - Applies to: configuration pairs (X,Y) - Range: relative (per source–target configuration pair) - Contextual limits: derivability is evaluated only along redistribution-only pathways and recursion-member pathways connecting the specified source–target configuration pair, using pathway and recursion structure as defined in D1.9 and D1.10.

T3. Dependencies (Definitional) - Depends on: {D1.0, D1.5, D1.7, D1.9, D1.10}

T4. Necessity Witness - Regulatory loss: loss of enforceability - Downstream witness: VE5 becomes non-decidable as a Core-internal restriction without a Core-internal definition of "derivable".

T5. Minimality / Non-Redundancy Derivability is defined only as existence of a redistribution/recursion pathway connection; no additional reachability structure is introduced.

T6. Admissibility Interface - Binds the use of "derivable" in Kernel Term statements to Core-internal pathway and recursion structure.

T7. Keywords derivable; derivability; pathway; recursion; reachability

D0.9 Nullification (Constraint Defeat)

T0. Term Header - ID: D0.9 - Name: Nullification (Constraint Defeat) - Category: Definition - Level: Meta

T1. Statement A relation, transformation, redistribution, or recursion nullifies a structural constraint C (in a fixed regulatory context) iff it renders C unenforceable for the target configuration while leaving C in-scope as an applicable adjacency-compatible constraint.

T2. Scope - Applies to: constraints and operations over configurations - Range: relative (per constraint and operation) - Contextual limits: "nullifies" denotes constraint-defeat (loss of enforceability) and is not a temporal, causal, or procedural notion

T3. Dependencies (Definitional) - Depends on: {D1.0, D1.2, D1.5, D1.7, D1.10}

T4. Necessity Witness - Regulatory loss: loss of enforceability - Downstream witness: F2 and AR2 become non-decidable as Core-internal prohibition clauses without a Core-internal definition of "nullifies".

T5. Minimality / Non-Redundancy Nullification is defined only as enforceability-loss for an in-scope constraint; it introduces no new primitives, no additional satisfaction calculus, and no procedural semantics.

T6. Admissibility Interface - Binds "nullifies" usage in Kernel Term statements to Core-internal enforceability loss rather than binary inconsistency.

T7. Keywords nullifies; nullification; constraint defeat; enforceability

D0.10 Coherence Kernel

T0. Term Header - ID: D0.10 - Name: Coherence Kernel - Category: Definition - Level: Meta

T1. Statement The coherence kernel, written $\mathcal{K}$, is the finite set of Kernel Terms that constitute the coherence-delimiting constraint set of this Core specification version: $\mathcal{K} = \{\text{F1–F14}\} \cup \{\text{VE1–VE8}\} \cup \{\text{AR1–AR10}\}$.

T2. Scope - Applies to: configurations, transformations, redistributions, emergences, pathways, recursions - Range: global - Contextual limits: relative to this Core specification version only

T3. Dependencies (Definitional) - Depends on: {} T3b. Audit / Reflective References (Non-definitional) - Audit references: {F1–F14, VE1–VE8, AR1–AR10} - Role: D0.10 packages the kernel identifier set for closure talk; kernel members are audit targets, not definitional prerequisites for naming the set.

T4. Necessity Witness - Regulatory loss: loss of representability; loss of enforceability - Downstream witness: Clean re-entry closure (D0.12) and coherent closure determinacy (D0.15) become ill-formed without an explicit Core-internal kernel constraint set.

T5. Minimality / Non-Redundancy This definition uniquely supplies a Core-internal name for the coherence-delimiting kernel constraint set used for closure and determinacy statements without requiring re-enumeration of constraints in downstream Kernel Terms.

T6. Admissibility Interface - Provides the named kernel constraint set referenced by Core-internal closure, nesting, and determinacy terms.

T7. Keywords kernel; coherence; constraint set; closure

D0.11 Non-Divergence Preorder

T0. Term Header - ID: D0.11 - Name: Non-Divergence Preorder - Category: Definition - Level: Meta

T1. Statement The non-divergence preorder, written $\preceq$, is an ordering-only relation on configurations such that for configurations X and Y, $X \preceq Y$ iff: (a) X and Y are configurations (D1.0); (b) if X is Core-admissible (D0.1), then Y is Core-admissible; and (c) for every Kernel

Term $C \in \mathcal{K}$ whose scope applies to configurations, if C is satisfied for X then C is satisfied for Y.

T2. Scope - Applies to: configurations - Range: global - Contextual limits: ordering-only; non-metric; relative to this Core specification version only

T3. Dependencies (Definitional) - Depends on: {D1.0, D0.1, D0.10}

T4. Necessity Witness - Regulatory loss: loss of minimality; loss of determinacy - Downstream witness: Clean re-entry closure (D0.12) becomes underdetermined without an ordering relation defining "least" closure among kernel-satisfying completions.

T5. Minimality / Non-Redundancy $\preceq$ introduces only the minimal ordering structure required to define least closure. It introduces no metric, measure, probability, dynamics, or procedural semantics.

T6. Admissibility Interface - Used as the ordering relation governing least completion in D0.12 and D0.15.

T7. Keywords ordering; preorder; least completion; non-divergence

D0.12 Clean Re-Entry Closure

T0. Term Header - ID: D0.12 - Name: Clean Re-Entry Closure - Category: Definition - Level: Meta

T1. Statement A configuration Y is a $\mathcal{K}$-closure candidate of X iff: (1) $X \preceq Y$; (2) Y is Core-admissible (D0.1); (3) for all configurations Z, if $X \preceq Z$ and Z is Core-admissible, then $Y \preceq Z$.

T2. Scope - Applies to: configurations - Range: global - Contextual limits: ordering-only; relative to this Core specification version only

T3. Dependencies (Definitional) - Depends on: {D1.0, D0.1, D0.10, D0.11}

T4. Necessity Witness - Regulatory loss: loss of determinacy; loss of enforceability - Downstream witness: Coherent closure determinacy (D0.15) and any Core term requiring "clean re-entry" become ill-formed without a Core-internal closure characterization.

T5. Minimality / Non-Redundancy This definition introduces only a least-completion closure characterization under an ordering-only relation. It introduces no generative mechanism, no algorithm, and no temporal or causal semantics.

T6. Admissibility Interface - Provides this specification-internal closure object referenced whenever kernel-satisfying least completion is required.

T7. Keywords closure; re-entry; least completion; determinacy

D0.13 Regime Indexing and Nesting

T0. Term Header - ID: D0.13 - Name: Regime Indexing and Nesting - Category: Definition - Level: Meta

T1. Statement A regime indexing structure is a partially ordered index set $(\mathsf{D}, \leq)$ together with a regime assignment $\mathsf{Reg} : \mathsf{D} \to \mathcal{P}(\mathcal{K})$ such that for all d_1, $\mathrm{d}_2 \in \mathsf{D}$: (1) $d_1 \leq d_2 \Rightarrow \mathsf{Reg}(d_1) \subseteq \mathsf{Reg}(d_2)$.

T2. Scope - Applies to: Core terms; regime-indexed admissibility characterizations - Range: global - Contextual limits: purely structural indexing; non-temporal; relative to this Core specification version only

T3. Dependencies (Definitional) - Depends on: {D0.10}

T4. Necessity Witness - Regulatory loss: loss of representability - Downstream witness: Admissible-set contraction under nesting (D0.14) becomes ill-formed without a Core-internal regime-nesting scaffold.

T5. Minimality / Non-Redundancy This definition introduces only index-ordering and constraint-set nesting as permitted meta-structural apparatus. It introduces no dynamics, no processes, and no generative commitments.

T6. Admissibility Interface - Provides the regime/nesting scaffold used to express contraction statements about admissibility under nested kernel recurrence.

T7. Keywords nesting; regimes; inheritance; indexing

D0.14 Admissible-Set Contraction Under Nesting

T0. Term Header - ID: D0.14 - Name: Admissible-Set Contraction Under Nesting - Category: Definition - Level: Meta

T1. Statement Given a regime indexing structure $(\mathsf{D}, \leq, \mathsf{Reg})$ satisfying D0.13, define for each d $\in \mathsf{D}$ the admissible set: $A_d = \{X \in \mathsf{Config} \mid X \vDash \mathsf{Reg}(d)\}$. Then admissibility contracts under nesting: $d_1 \leq d_2 \Rightarrow A_{d_2} \subseteq A_{d_1}$.

T2. Scope - Applies to: configurations; regime-indexed admissibility sets - Range: global - Contextual limits: relative to a fixed $(\mathsf{D}, \leq, \mathsf{Reg})$ satisfying D0.13; relative to this Core specification version only

T3. Dependencies (Definitional) - Depends on: {D1.0, D0.13}

T4. Necessity Witness - Regulatory loss: loss of representability - Downstream witness: Any canonical derivation that asserts admissibility contraction or descent under nested regime recurrence becomes underdetermined without a Core-internal contraction characterization.

T5. Minimality / Non-Redundancy This definition introduces only set-inclusion contraction as a structural consequence of nested inheritance. It introduces no metric, no dynamics, and no procedural evaluation semantics.

T6. Admissibility Interface - Permits Core-internal reference to contraction of admissible configurations under nested regime inheritance without importing external "descent" semantics.

T7. Keywords contraction; admissibility; nesting; descent

D0.15 Coherent Closure Determinacy

T0. Term Header - ID: D0.15 - Name: Coherent Closure Determinacy - Category: Definition - Level: Meta

T1. Statement For every configuration X, if Y and Z are each $\preceq$-least $\mathcal{K}$-closure candidates of X, then $Y \sim Z$. When such a $\preceq$-least candidate exists, denote it $\mathrm{Cl}_{\mathcal{K}}(X)$.

T2. Scope - Applies to: configurations - Range: global - Contextual limits: determinacy is asserted up to structural equivalence; relative to this Core specification version only

T3. Dependencies (Definitional) - Depends on: {D0.12, D1.6}

T4. Necessity Witness - Regulatory loss: loss of determinacy; loss of enforceability - Downstream witness: Any canonical claim that coherence yields a determinate least completion under clean re-entry becomes undecidable without an explicit determinacy binding.

T5. Minimality / Non-Redundancy Determinacy is asserted only up to structural equivalence, preventing representational variance from being misclassified as divergence while introducing no additional object-level primitives or evaluators.

T6. Admissibility Interface - Governs any Core term or canonical derivation asserting an inevitable coherent completion under clean re-entry closure.

T7. Keywords determinacy; inevitability; closure; equivalence —

D0.16 Constraint Satisfaction (Canonical)

T0. Term Header - ID: D0.16 - Name: Constraint Satisfaction (Canonical) - Category: Definition - Level: Meta

T1. Statement Let T be a Core term whose T1 statement is a well-formed binding constraint statement and whose T2 scope declares the typing and applicability conditions for T. Let E be a Core object.

```
1) Applicability. T is applicable to E iff E falls within the typing and scope declared in T

2) Satisfaction (Judgement). If T is applicable to E, then the satisfaction judgement E ⊨ T
    E ⊨ T is treated as a Core-internal primitive satisfaction judgement; this term specifie
    Interpretation discipline. Any binding use of E ⊨ T SHALL NOT rely on any satisfaction s

    Obtains (abbrev). "T obtains at E" abbreviates E ⊨ T.

    - the T1 statement of T instantiated at E under the binding conventions of this specific
    - the permitted meta-structural vocabulary.

3) Non-satisfaction. E ⊭ T ≜ (T is applicable to E) ∧ ¬(E ⊨ T).

4) Inapplicability convention. If T is not applicable to E, then satisfaction is undefined
```

T2. Scope - Applies to: all binding constraints (Primitive Definitions, Foundational Conditions, Valid Emergence Conditions, Admissibility Rules, and any binding Canonical Derivation) - Range: global - Contextual limits: declarative only; introduces no algorithmic validator; operates only over Core terms with explicit T2 scope and explicit dependency closure

T3. Dependencies (Definitional) - Depends on: {S-1, S-5, S-8, S-9}

T3b. Audit / Reflective References (Non-definitional) - Audit references: {D0.1} - Role: D0.16 may be applied to Core-admissibility claims (D0.1) without making D0.1 definitional for satisfaction itself.

T4. Necessity Witness - Regulatory loss: loss of enforceability; loss of mechanical conformance clarity - Downstream witness: Core Admissibility Characterization (D0.1) and any term using ⊨ / "satisfied" are underdetermined without a Core-internal satisfaction predicate, forcing external semantic import contrary to this specification's closure posture.

T5. Minimality / Non-Redundancy This term introduces no new object-level primitive, no new admissibility rule, and no enforcement mechanism. It binds an already-used notion ("satisfied," ⊨) to the existing schema machinery (T1/T2) and permitted meta-structural vocabulary.

T6. Admissibility Interface - Used by: D0.1 and any term or derivation whose binding statements quantify over "applicable constraints are satisfied" - Effect: makes satisfaction a Core-internal meta predicate grounded in T1/T2 scope rather than an imported semantic primitive.

T7. Keywords satisfaction; satisfies; models; $\models$; applicability; scope; conformance

D0.17 Enforceability (Canonical)

T0. Term Header - ID: D0.17 - Name: Enforceability (Canonical) - Category: Definition - Level: Meta

T1. Statement Let T be a binding constraint term with explicit T2 scope and explicit T6 interface. Let E be a Core object.

```
1) Enforceability at an object. T is enforceable at E iff all of the following hold:
    (a) Applicability is well-typed: T is applicable to E under T2(T) without importing exte
    (b) Satisfaction is Core-internal: the judgement E ⊨ T (or E ⊭ T) is well-formed and gov
    (c) No implicit method is required: applying T's interface requires no assumed operatior
    (d) No nullification-by-application: applying T does not render T (or any constraint it

2) Loss of enforceability at an object. T has loss of enforceability at E iff T is applicabl

3) Enforceability of a constraint-set at an object. A set of constraints C is enforceable at

4) Inapplicability convention. If T is not applicable to E, enforceability is undefined (nei
```

T2. Scope - Applies to: all binding constraints - Range: global - Contextual limits: declarative only; introduces no algorithmic validator; enforceability is a structural condition on applicability + satisfaction + interface legality under this specification's closure constraints

T3. Dependencies (Definitional) - Depends on: {S-3, S-4, S-5, D0.16, D0.9}

T4. Necessity Witness - Regulatory loss: modality semantics become untyped; "loss of enforceability" becomes an imported judgement - Downstream witness: MUST/MUST NOT are explicitly tied to loss of enforceability; absent a Core-internal predicate, evaluators must import external enforceability semantics.

T5. Minimality / Non-Redundancy This term adds no object-level primitive, no new admissibility rule, and no enforcement mechanism. It binds an already-invoked modal failure mode to existing this specification closure constraints.

T6. Admissibility Interface - Used by: Modality and Constitutive Force clauses citing enforceability; any necessity witness citing enforceability loss - Effect: makes "enforceability" a Core-internal meta predicate grounded in D0.16 + interface legality + nullification discipline.

T7. Keywords enforceability; enforceable; loss of enforceability; nullification; interface; conformance

D0.18 Determinacy (Canonical)

T0. Term Header - ID: D0.18 - Name: Determinacy (Canonical) - Category: Definition - Level: Meta

T1. Statement Let $\sim$ denote Structural Equivalence (D1.6) on configurations, and let $\preceq$ denote the Non-Divergence Preorder (D0.11) used to express "least completion" comparisons. Let $\mathrm{Cl}(\cdot)$ denote a closure-like assignment expressed in Core terms.

```
1) Determinacy up to equivalence. Cl is determinate up to ~ on a domain D iff for every X ∈
```

```
2) Uniqueness up to equivalence. A candidate for X is unique up to ~ iff determinacy up to ~

3) Least-candidate determinacy. If the defining characterization of Cl specifies a least can

4) No existence import. Determinacy does not assert existence. It constrains multiplicity of
```

T2. Scope - Applies to: any Core term asserting determinacy, uniqueness up to equivalence, or "least completion" uniqueness - Range: global - Contextual limits: meta-structural only; does not introduce a new closure operator; does not introduce or require an algorithmic method for finding candidates

T3. Dependencies (Definitional) - Depends on: {D1.6, D0.11}

T4. Necessity Witness - Regulatory loss: determinacy/uniqueness language becomes rhetorically theorem-like while remaining untyped - Downstream witness: any term that asserts determinacy "up to equivalence" must bind that phrase to Core-internal relations ($\sim$ via D1.6, $\preceq$ via D0.11) to avoid importing external mathematical semantics.

T5. Minimality / Non-Redundancy This term introduces no object-level primitive and asserts no existence. It binds determinacy/uniqueness language to existing this specification relations ($\sim$, $\preceq$) and to candidate characterizations already stated elsewhere.

T6. Admissibility Interface - Used by: any term asserting determinacy or uniqueness up to equivalence (including coherent-closure determinacy claims) - Effect: separates existence commitments from determinacy commitments and makes determinacy claims typed and checkable.

T7. Keywords determinacy; unique up to equivalence; least completion; closure; preorder; equivalence

D0.19 Coherence Specification Suite (CSS)

T0. Term Header - ID: D0.19 - Name: Coherence Specification Suite (CSS) - Category: Definition - Level: Meta

T1. Statement The Coherence Specification Suite (CSS) denotes the set of Coherence specifications designated as canonical modules, including at minimum this Core specification and the Coherence Axioms specification.

T2. Scope - Applies to: canonical status claims; canonical derivations; cross-document conformance and completeness conditions - Range: global - Contextual limits: suite-scoped references only; relative to this Core specification version only

T3. Dependencies (Definitional) - Depends on: {D0.7, D0.16, D0.17}

T4. Necessity Witness Without an explicit suite-level referent, plural authority claims (e.g., "Core + Axioms") are forced to overload "this specification," creating scope ambiguity and enabling inadvertent cross-document semantic leakage.

T5. Minimality / Non-Redundancy This term introduces no object-level structure; it provides only a suite-level name for cross-document governance already present in canonical status and conformance language.

T6. Admissibility Interface - Use "this specification" or "this Core" for Core-only claims. - Use "the CSS" (or "the Suite") for claims that range over canonical modules (e.g., Core + Axioms).

T7. Keywords suite; module; canonical; conformance; cross-document; Core; Axioms

D0.20 Incompatibility (Meta)

T0. Term Header - ID: D0.20 - Name: Incompatibility (Meta) - Category: Definition - Level: Meta

T1. Statement A set of binding constraints C is incompatible for an evaluation target E iff: (1) every constraint $T \in C$ is applicable to E (D0.16); and (2) either (a) there exists at least one $T \in C$ such that $E \not\models T$ (D0.16), or (b) there exists at least one $T \in C$ such that T has loss of enforceability at E (D0.17).

T2. Scope - Applies to: binding constraints (as binding T1 statements), evaluation targets - Range: relative (per target and constraint-set) - Contextual limits: meta-only; does not assert existence beyond admissibility status

T3. Dependencies (Definitional) - Depends on: {D0.16, D0.17}

T4. Necessity Witness Without a Core-internal definition of incompatibility, compatibility claims reduce to either rhetoric ("compatible by declaration") or external semantics, undermining enforceability and joint-admissibility auditing.

T5. Minimality / Non-Redundancy This definition introduces no new object-level primitives; it binds "incompatibility" solely to satisfaction and enforceability failure modes already defined in this specification.

T6. Admissibility Interface - Used to type-check joint applicability and joint enforceability claims (e.g., F12 and closure/conformance audits).

T7. Keywords incompatibility; joint enforceability; satisfaction; nullification; meta

D0.21 Reflexive Closure Certificate

T0. Term Header - ID: D0.21 - Name: Reflexive Closure Certificate - Category: Definition - Level: Meta

T1. Statement A Reflexive Closure Certificate (RCC) is a finite, schema-complete, Core-internal artifact that witnesses—without importing any external primitive, algorithm, or test—that a governed canonical module or declared kernel set is structurally well-formed and self-auditable.

An RCC MUST explicitly enumerate the following components: 1. Definitional Dependency DAG: the set of all T3 edges among included terms. 2. Acyclicity Witness: an explicit topological ordering (or equivalent well-founded ranking) for the definitional DAG (S-5). 3. Binding Surface Inventory: the set of all binding clauses (T1–T6) within the governed module. 4. Binding Reachability Map: for each binding clause, an explicit list of definitional-reachable supporting terms required to interpret it (S-8). 5. Audit Reference Inventory: the set of all T3b audit references within the governed module (S-9). 6. Canonical Material Classification Snapshot: an explicit enumeration of canonical vs non-normative material under D0.7. 7. Schema Token Conformance Report: enumeration of any non-literal schema tokens detected under S-10. 8. Typing Discipline Report: confirmation that all binding quantification over emergences uses the form $T : X \to Y$ under S-11. 9. Binding Vocabulary Audit: enumeration of any technical constructs appearing in binding clauses that are neither Core terms nor admitted meta-structural apparatus under S-12.

T2. Scope - Applies to: canonical modules; declared kernel sets governed by this Core - Range: relative (per governed module / kernel set) - Contextual limits: structural conformance, definitional well-foundedness, binding reachability, audit integrity, token conformance, typing discipline, and binding vocabulary hygiene

T3. Dependencies (Definitional) - Depends on: {S-2, S-3, S-5, S-8, S-9, S-10, S-11, S-12, D0.7}

T3b. Audit / Reflective References (Non-definitional) - Audit references: {AR11, AR8} - Role: an RCC ranges over conformance and closure characterizations without requiring them for definitional interpretation.

T4. Necessity Witness - Regulatory loss: loss of enforceability; loss of mechanical conformance clarity - Downstream witness: AR11 cannot be applied as a structurally checkable criterion without an explicit Core-internal witness artifact enumerating dependency order, binding surface, and reachability.

T5. Minimality / Non-Redundancy This term introduces no new enforcement mechanism. It supplies only the minimum structural witness components required to certify reflexive closure and self-audit without bootstrapping cycles.

T6. Admissibility Interface - The certificate itself is admissible as Canonical material only when it is schema-complete and refers exclusively to Core-internal terms.

T7. Keywords certificate; reflexive closure; audit; reachability; DAG; well-foundedness; conformance

D0.22 Differentiation Threshold

T0. Term Header - ID: D0.22 - Name: Differentiation Threshold - Category: Definition - Level: Meta

T1. Statement A differentiation threshold is reached within a canonical module or kernel region when continued use of the existing binding vocabulary, scope declarations, and dependency discipline would induce loss of containment, loss of boundedness, loss of representability, or loss of enforceability for at least one downstream Core judgement (applicability, satisfaction, enforceability, or conformance) under this specification closure constraints.

When such a threshold is reached, stabilization is admissible only by introducing an explicit new distinction expressed as one or more schema-complete Kernel Terms (T0–T6) whose necessity is witnessed (T4) and whose binding vocabulary is Core-internal under S-3 and S-8.

T2. Scope - Applies to: Core terms, canonical derivations, suite modules, governance artifacts - Range: global - Contextual limits: meta-structural only; relative to this Core specification version only

T3. Dependencies (Definitional) - Depends on: {S-3, S-4, S-5, S-8, D0.16, D0.17, AR11}

T4. Necessity Witness - Regulatory loss: loss of enforceability; loss of representability - Downstream witness: AR11 conformance and any D0.16/D0.17-governed judgement become evaluator-dependent whenever binding vocabulary or distinctions are extended without schema-complete witness structure.

T5. Minimality / Non-Redundancy This definition introduces no new primitives and no enforcement procedure. It names the constitutive condition under which new distinctions become mandatory for maintaining Core-internal enforceability and binds admissible stabilization to existing witness and closure discipline.

T6. Admissibility Interface - Used by: governance edits and RCC artifacts (D0.21) to justify when new terms/distinctions are required rather than optional. - Effect: converts “new concept needed” from editorial discretion into a witnessable, Core-internal admissibility event.

T7. Keywords differentiation; threshold; witness; stabilization; enforceability

D0.23 Propagation (Canonical)

T0. Term Header - ID: D0.23 - Name: Propagation (Canonical) - Category: Definition - Level: Meta

T1. Statement Propagation is the adjacency-compatible carrying of structural change through admissible pathways, such that any propagation relevant to admissibility is representable as occurring along a pathway (D1.9) consistent with adjacency (D1.1).

T2. Scope - Applies to: configurations, pathways, transformations - Range: relative (per source–target pair) - Contextual limits: representational-only; does not introduce dynamics or procedures

T3. Dependencies (Definitional) - Depends on: {D0.24, D1.0, D1.1, D1.7, D1.9, F3, F5}

T4. Necessity Witness - Regulatory loss: loss of representability - Downstream witness: F1 and F11 become evaluator-dependent whenever they regulate "propagation" without a Core-internal representational anchor.

T5. Minimality / Non-Redundancy This term introduces no new object-level primitives. It binds "propagation" to already-defined pathway structure and existing locality discipline.

T6. Admissibility Interface - Used by: Foundational Conditions and Admissibility Rules that regulate propagation, locality, or influence transmission.

T7. Keywords propagation; pathways; locality; adjacency

D0.24 Preservation (Canonical)

T0. Term Header - ID: D0.24 - Name: Preservation (Canonical) - Category: Definition - Level: Meta

T1. Statement For a binding constraint term C and a transformation $T : X \to Y$, "T preserves C" means: whenever C is applicable to X and Y (D0.16), if X $\models$ C then Y $\models$ C.

T2. Scope - Applies to: binding constraints, transformations, configuration pairs - Range: relative (per constraint and transformation) - Contextual limits: satisfaction-based only; introduces no temporal, causal, or procedural semantics

T3. Dependencies (Definitional) - Depends on: {D0.16, D1.0, D1.7}

T4. Necessity Witness - Regulatory loss: loss of enforceability - Downstream witness: Any binding clause asserting "preservation" becomes an imported semantic primitive without a Core-internal preservation predicate.

T5. Minimality / Non-Redundancy Preservation is defined only in terms of applicability + satisfaction; it introduces no new constraint and no new method.

T6. Admissibility Interface - Used by: binding clauses that speak about preservation of continuity, locality, or constraint satisfaction across transformations.

T7. Keywords preservation; satisfaction; applicability; $\models$

D0.25 Availability (Canonical)

T0. Term Header - ID: D0.25 - Name: Availability (Canonical) - Category: Definition - Level: Meta

T1. Statement A redistribution pathway is available from configuration X iff there exists at least one pathway P(X,Y) (D1.9) such that every transformation in P(X,Y) is a redistribution (D1.5).

T2. Scope - Applies to: configurations, pathways - Range: relative (per source configuration) - Contextual limits: existence-of-pathway only; no algorithmic search is implied

T3. Dependencies (Definitional) - Depends on: {D1.0, D1.5, D1.9}

T4. Necessity Witness - Regulatory loss: loss of enforceability - Downstream witness: F6 becomes ambiguous whenever it requires "availability" of redistribution pathways without a Core-internal meaning of availability.

T5. Minimality / Non-Redundancy This definition introduces no new primitives and asserts no dynamics; it binds "available" to existence of a redistribution-only pathway.

T6. Admissibility Interface - Used by: Foundational Conditions and Admissibility Rules that require availability of at least one redistribution pathway.

T7. Keywords availability; exists; pathway; redistribution

D0.26 Support (Canonical)

T0. Term Header - ID: D0.26 - Name: Support (Canonical) - Category: Definition - Level: Meta

T1. Statement A configuration X supports an emergence $T : X \to Y$ iff Y is derivable from X within this Core specification (D0.8).

T2. Scope - Applies to: configurations, emergences - Range: relative (per emergence) - Contextual limits: support is representational-only and grounded solely in this specification derivability notion

T3. Dependencies (Definitional) - Depends on: {D0.8, D1.0, D1.8}

T4. Necessity Witness - Regulatory loss: loss of closure; loss of enforceability - Downstream witness: VE1 becomes evaluator-dependent whenever it uses "supported by X" as constitutive provenance without a Core-internal support predicate.

T5. Minimality / Non-Redundancy Support is defined only as derivability. It introduces no new object-level notion of causation or substrate support.

T6. Admissibility Interface - Used by: Valid Emergence Conditions and Admissibility Rules that regulate emergence provenance and internal support.

T7. Keywords support; derivable; provenance; emergence

D0.27 Composition (Canonical)

T0. Term Header - ID: D0.27 - Name: Composition (Canonical) - Category: Definition - Level: Meta

T1. Statement A configuration Z is a composition of configurations {X_i} iff there exists a finite family of configurations {X_i} and a finite family of adjacency-preserving embeddings of each X_i into Z such that: (a) each embedded X_i is a sub-configuration region of Z whose induced adjacency is compatible with $\mathrm{Adj}(X_i)$; and (b) all binding constraints applicable to each X_i remain satisfiable when each X_i is treated as an induced region of Z under the Coupled-Scope posture of F14.

T2. Scope - Applies to: configurations; derived configuration families - Range: relative (per composed configuration and family) - Contextual limits: representational-only; does not assert existence; uses only adjacency preservation and binding scope posture

T3. Dependencies (Definitional) - Depends on: {D1.0, D1.1, D1.6, D0.16, D0.17, F14}

T4. Necessity Witness - Regulatory loss: loss of representability - Downstream witness: F7 becomes unenforceable without a Core-internal binding for "composition / sub-configuration" that avoids external mereology.

T5. Minimality / Non-Redundancy This definition binds "composition" to adjacency-preserving regional embedding plus binding scope posture, introducing no new object-level primitives.

T6. Admissibility Interface - Used by: F7 and any term asserting multi-scale composition or sub-configuration nesting under this specification.

T7. Keywords composition; sub-configuration; nesting; embedding; scope

D0.28 Non-Erasure (Canonical)

T0. Term Header - ID: D0.28 - Name: Non-Erasure (Canonical) - Category: Definition - Level: Meta

T1. Statement A transformation $T : X \to Y$ erases structure iff there exists a structural relation R in X such that for every configuration Y' structurally equivalent to Y (D1.6), no relation in Y' is the image of R under any adjacency-preserving isomorphism from X into Y'; T is non-erasing iff it does not erase structure.

T2. Scope - Applies to: transformations - Range: relative (per transformation) - Contextual limits: erasure is evaluated only up to structural equivalence and adjacency-preserving isomorphism; introduces no temporal or causal semantics

T3. Dependencies (Definitional) - Depends on: {D1.0, D1.1, D1.6, D1.7}

T4. Necessity Witness - Regulatory loss: loss of representability - Downstream witness: F8 is unenforceable if "erased" is treated as an external semantic primitive.

T5. Minimality / Non-Redundancy Non-erasure is bound purely to the Core's existing equivalence and adjacency apparatus, introducing no new primitives.

T6. Admissibility Interface - Used by: F8 and any term requiring conservation / persistence claims without importing external semantics.

T7. Keywords non-erasure; conservation; persistence; equivalence

D0.29. Interface Representation (Canonical)

T0. Term Header - ID: D0.29 - Name: Interface Representation (Canonical) - Category: Definition - Level: Meta

T1. Statement A binding clause provides an interface representation iff the clause explicitly names an interface carrier I such that: (a) I is either a pathway P (D1.9) or a recursion $\mathcal{R}$ (D1.10); and (b) every instance of cross-scope propagation that the clause relies upon for any binding application is constrained to be representable as occurring only along I, as governed by Propagation (D0.23) and enforceable under D0.17.

T2. Scope - Applies to: binding clauses; distinctions; pathways; recursions - Range: global - Contextual limits: representational-only; cross-boundary reliance is permitted only when mediated by an explicit Core object I

T3. Dependencies (Definitional) - Depends on: {D0.17, D0.23, D1.1, D1.9, D1.10}

T4. Necessity Witness - Regulatory loss: loss of enforceability via implicit cross-boundary reliance - Downstream witness: F14 becomes structurally uncheckable without a Core-internal criterion for what counts as an "explicit interface representation."

T5. Minimality / Non-Redundancy This definition introduces no new object-level primitives. It binds "interface representation" solely to explicit naming of an existing Core object and enforceable mediation of cross-boundary propagation under D0.17.

T6. Admissibility Interface - Used by: F14 (Coupled-Scope posture) - Effect: makes "explicit interface representation" checkable as an enforceability condition rather than an imported semantic primitive.

T7. Keywords interface; boundary; mediation; representation; scope

D0.30. Binding Lexicon Ledger (Canonical)

T0. Term Header - ID: D0.30 - Name: Binding Lexicon Ledger (Canonical) - Category: Definition - Level: Meta

T1. Statement A binding lexicon ledger for a module M, written $\mathcal{L}$_M, is a finite inventory of all symbols, named predicates, and technical phrases used with constitutive force in any binding clause (T1–T6) of M, together with a resolution class and supporting anchor for each entry, where each entry is assigned exactly one of: (a) Defined Term: the entry is bound to the T1 statement of a schema-complete Core term; (b) Admitted Apparatus: the entry is explicitly admitted as permitted meta-structural apparatus within this Core specification's meta-vocabulary clauses; (c) non-normative Gloss: the entry occurs only in non-normative material and contributes no constitutive force under D0.7; (d) Unresolved: the entry is not supported by (a), (b), or (c).

T2. Scope - Applies to: Core terms, binding clauses, canonical derivations, canonical modules, kernel sets - Range: global - Contextual limits: relative to a specified module M and to this Core specification version only; the inventory concerns constitutive vocabulary in binding clauses only

T3. Dependencies (Definitional) - Depends on: {D0.7, D0.16, D0.17, S-3, S-8, S-9, S-10, S-12}

T4. Necessity Witness - Regulatory loss: loss of enforceability; loss of determinacy - Downstream witness: Semantic reflexive closure (D0.31) becomes unenforceable and evaluator-dependent without a Core-internal notion of a finite, classed binding vocabulary inventory.

T5. Minimality / Non-Redundancy This definition introduces only an inventory-and-classification object for constitutive vocabulary. It adds no object-level primitives, no procedures, and no enforcement mechanism beyond existing closure and reachability constraints.

T6. Admissibility Interface - Used by: D0.31 and any module-level self-audit that requires binding-vocabulary closure under S-3/S-8/S-12. - Effect: makes binding-vocabulary closure auditable without importing external semantic registries.

T7. Keywords lexicon; vocabulary; binding surface; firewall; reachability; audit

D0.31. Semantic Reflexive Closure Criterion (Publication-Grade)

T0. Term Header - ID: D0.31 - Name: Semantic Reflexive Closure Criterion (Publication-Grade) - Category: Definition - Level: Meta

T1. Statement A module M is semantically reflexively closed relative to this Core specification version iff: (1) M is structurally well-formed and self-auditable under this Core specification's schema and conformance governance (D0.7, S-5, S-8, S-9, S-10, AR11); and (2) there exists a binding lexicon ledger $\mathcal{L}$_M (D0.30) such that: (a) every constitutive vocabulary entry used with binding force in any binding clause of M is present in $\mathcal{L}$_M; (b) every such entry is classified as either Defined Term or Admitted Apparatus; and (c) no constitutive vocabulary entry used with binding force in any binding clause of M is classified as Unresolved or non-normative Gloss.

T2. Scope - Applies to: canonical modules, kernel sets, canonical derivations - Range: global - Contextual limits: governance-only; does not assert existence of modules; applies only relative to this Core specification version and its binding-vocabulary firewall

T3. Dependencies (Definitional) - Depends on: {D0.30, D0.7, D0.16, D0.17, S-3, S-5, S-8, S-9, S-10, S-12}

T3b. Audit / Reflective References (Non-definitional) - Audit references: {AR11} - Role: AR11 is an audit target for structural conformance characterization and does not provide definitional content for semantic closure beyond the governance posture referenced in (1).

T4. Necessity Witness - Regulatory loss: loss of enforceability; loss of determinacy - Downstream witness: Publication-grade classification (S-13) and any claim of closure-complete self-validation becomes evaluator-dependent without a Core-internal criterion that eliminates unresolved constitutive vocabulary.

T5. Minimality / Non-Redundancy This definition adds no object-level primitives and introduces no algorithmic validator. It binds "semantic closure" solely to the existence of a finite binding lexicon ledger with empty unresolved surface, grounded in existing closure and reachability constraints.

T6. Admissibility Interface - Used by: S-13 and any suite-level publication governance. - Effect: distinguishes structurally canonical material from publication-grade, semantically closed material without altering object-level admissibility.

T7. Keywords semantic closure; reflexive closure; publication-grade; lexicon; firewall

3. FOUNDATIONAL CONDITIONS

(Meta-Structural · Normative · Global)

Foundational Conditions (F1–F14) are global regulatory constraints governing all admissible configurations, transformations, redistributions, emergences, pathways, and recursions.

They: - introduce no object-level structure, - apply universally, - regulate bounded representability and enforceability, - preserve graded coherence under all admissible operations.

Each Foundational Condition is expressed in schema-complete form.

F1. Continuity

T0. Term Header - ID: F1 - Name: Continuity - Category: Foundational Condition - Level: Meta

T1. Statement For all admissible transformations $T : X \to Y$, if T is a redistribution (D1.5), then T SHALL satisfy continuity (D1.2) so that restriction-consistent locality is preserved across adjacency-bounded scope.

T2. Scope - Applies to: transformations - Range: global - Contextual limits: none

T3. Dependencies (Definitional) - Depends on: {D0.23, D0.24, D1.0, D1.1, D1.2, D1.5, D1.7}

T4. Necessity Witness - Regulatory loss: loss of representability; loss of enforceability - Downstream witness: AR-level admissibility of redistribution becomes undecidable without a continuity regulator.

T5. Minimality / Non-Redundancy This condition uniquely regulates rupture as a failure mode distinct from bounded strain. No other condition constrains continuity of propagation under transformation.

T6. Admissibility Interface - Constrains admissible transformations and redistributions. - Referenced by AR-level enforcement of admissible pathways and emergence.

T7. Keywords continuity; propagation; boundedness; transformation

F2. Non-Negation

T0. Term Header - ID: F2 - Name: Non-Negation - Category: Foundational Condition - Level: Meta

T1. Statement For all admissible transformations and redistributions, structural relations within a configuration are regulated such that no relation simultaneously enforces and nullifies the same adjacency-compatible constraint.

T2. Scope - Applies to: transformations, redistributions - Range: global - Contextual limits: none

T3. Dependencies (Definitional) - Depends on: {D0.9, D1.0, D1.1, D1.5, D1.7}

T4. Necessity Witness - Regulatory loss: loss of containment; loss of enforceability - Downstream witness: Structural equivalence (D1.6) and expressive-capacity ordering become undecidable without non-negation.

T5. Minimality / Non-Redundancy Non-negation uniquely prevents internal structural cancellation. No other condition constrains contradictory constraint enforcement.

T6. Admissibility Interface - Constrains admissible redistributions and recursion.

T7. Keywords non-negation; constraint preservation; consistency

F3. Adjacency Preservation

T0. Term Header - ID: F3 - Name: Adjacency Preservation - Category: Foundational Condition - Level: Meta

T1. Statement For all admissible transformations, redistributions, and emergences, the adjacency structure of a configuration is regulated such that no admissible operation introduces interaction or redistribution outside the adjacency-bounded scope defined for the configuration.

T2. Scope - Applies to: transformations, redistributions, emergences - Range: global - Contextual limits: adjacency as defined per configuration

T3. Dependencies (Definitional) - Depends on: {D1.0, D1.1, D1.5, D1.7, D1.8}

T4. Necessity Witness - Regulatory loss: loss of containment - Downstream witness: Coherence and continuity enforcement collapse without adjacency-bounded propagation.

T5. Minimality / Non-Redundancy Adjacency preservation uniquely enforces locality as domain restriction. No other condition constrains the domain of admissible interaction.

T6. Admissibility Interface - Constrains all admissible operations. - Referenced by coherence evaluation and redistribution enforcement.

T7. Keywords adjacency; locality

F4. Boundedness

T0. Term Header - ID: F4 - Name: Boundedness - Category: Foundational Condition - Level: Meta

T1. Statement For all admissible configurations and transformations, structural differentials are regulated such that all relational differentials remain contained within finite, adjacency-bounded scope under admissible redistribution.

T2. Scope - Applies to: configurations, transformations - Range: global - Contextual limits: none

T3. Dependencies (Definitional) - Depends on: {D1.0, D1.1, D1.4, D1.5, D1.7}

T4. Necessity Witness - Regulatory loss: loss of boundedness; loss of representability - Downstream witness: Regulatory coherence and nonterminality become undecidable without a containment bound on strain.

T5. Minimality / Non-Redundancy Boundedness uniquely distinguishes admissible strain from uncontainable divergence. No other condition enforces finite containment of internal differentials.

T6. Admissibility Interface - Required for coherence evaluation under bounded regulatory satisfaction. - Referenced by AR-level nonterminality and continuation enforcement.

T7. Keywords boundedness; containment; strain

F5. Locality

T0. Term Header - ID: F5 - Name: Locality - Category: Foundational Condition - Level: Meta

T1. Statement For all admissible operations, redistribution is regulated such that it acts only through adjacency-mediated scope.

T2. Scope - Applies to: transformations, redistributions, emergences - Range: global - Contextual limits: adjacency-relative

T3. Dependencies (Definitional) - Depends on: {D1.0, D1.1, D1.5, D1.7, D1.8}

T4. Necessity Witness - Regulatory loss: loss of containment - Downstream witness: Adjacency preservation (F3) and continuity (F1) lose enforceable locality constraints.

T5. Minimality / Non-Redundancy Locality uniquely constrains the mode of propagation through adjacency mediation, not merely the domain of interaction. No other condition prevents non-adjacency-mediated structural influence.

T6. Admissibility Interface - Constrains redistribution and emergence pathways. - Referenced by AR-level admissibility evaluation.

T7. Keywords locality; propagation; adjacency

F6. Redistribution Support

T0. Term Header - ID: F6 - Name: Redistribution Support - Category: Foundational Condition - Level: Meta

T1. Statement For all admissible configurations admitting internal structural differentials, the configuration is regulated such that at least one adjacency-preserving redistribution pathway is available (D0.25) under bounded coherence.

T2. Scope - Applies to: configurations - Range: global - Contextual limits: configurations with admissible drivers

T3. Dependencies (Definitional) - Depends on: {D0.24, D0.25, D1.0, D1.4, D1.5, D1.9}

T4. Necessity Witness - Regulatory loss: loss of enforceability; loss of nonterminality - Downstream witness: AR9 cannot force non-equivalent emergence without guaranteed redistribution availability.

T5. Minimality / Non-Redundancy Redistribution support uniquely ensures that admissible strain can be acted upon. No other condition guarantees the existence of a resolving pathway.

T6. Admissibility Interface - Required for nonterminality enforcement under admissible drivers. - Directly referenced by AR9.

T7. Keywords redistribution; nonterminality; support

F7. Nested Scaling

T0. Term Header - ID: F7 - Name: Nested Scaling - Category: Foundational Condition - Level: Meta

T1. Statement For all admissible configurations and emergent structures, scaling relations are regulated such that coherent sub-configurations may be composed into larger configurations without loss of bounded coherence, adjacency compatibility, or enforceability under this specification constraints.

T2. Scope - Applies to: configurations, emergences - Range: global - Contextual limits: scale-relative composition only

T3. Dependencies (Definitional) - Depends on: {D0.27, D1.0, D1.1, D1.8, F1, F3, F5}

T4. Necessity Witness - Regulatory loss: loss of representability; loss of enforceability - Downstream witness: Without nested scaling, higher-order configurations cannot be composed from admissible substructures, rendering recursive emergence and hierarchical organization undecidable.

T5. Minimality / Non-Redundancy Nested scaling uniquely regulates coherence preservation across compositional levels. No prior Foundational Condition constrains multi-scale composability of admissible structure.

T6. Admissibility Interface - Required for admissible composition of emergent configurations. - Referenced by Valid Emergence Conditions governing cross-scale structure. - Supports recursive coherence under increasing expressive capacity.

T7. Keywords scaling; hierarchy; composition; recursion

F8. Conservation Through Transformation

T0. Term Header - ID: F8 - Name: Conservation Through Transformation - Category: Foundational Condition - Level: Meta

T1. Statement For all admissible transformations and redistributions, configuration structure (i.e., the adjacency-compatible relational structure of a configuration, considered up to structural equivalence) is regulated such that no admissible structure is erased; all change occurs through redistribution, reconfiguration, or re-expression of existing structure.

T2. Scope - Applies to: transformations, redistributions - Range: global - Contextual limits: none

T3. Dependencies (Definitional) - Depends on: {D0.28, D1.0, D1.1, D1.5, D1.6, D1.7}

T4. Necessity Witness - Regulatory loss: loss of representability - Downstream witness: Memory and path dependence become undecidable without conservation.

T5. Minimality / Non-Redundancy This condition uniquely prevents structural annihilation. No other condition enforces persistence of structure across transformation.

T6. Admissibility Interface - Enforced by AR-level transformation and recursion constraints.

T7. Keywords conservation; transformation; structure; persistence

F9. Degrees of Freedom for Redistribution

T0. Term Header - ID: F9 - Name: Degrees of Freedom for Redistribution - Category: Foundational Condition - Level: Meta

T1. Statement For any admissible configuration admitting redistribution, sufficient degrees of freedom are regulated such that more than one distinct adjacency-compatible redistribution pathway remains available (distinctness as defined in D0.3).

T2. Scope - Applies to: configurations - Range: global - Contextual limits: configurations with admissible drivers

T3. Dependencies (Definitional) - Depends on: {D1.4, D1.5, D1.9, D0.3}

T4. Necessity Witness - Regulatory loss: loss of nonterminality - Downstream witness: AR10 cannot enforce bounded nonterminal continuation without non-singleton redistributive freedom.

T5. Minimality / Non-Redundancy This condition uniquely ensures that admissible strain is actionable with non-singleton redistributive freedom, not merely with a single resolving path. No other condition distinguishes existence of a pathway (F6) from plural redistributive freedom.

T6. Admissibility Interface - Referenced by AR9, AR10, and recursive emergence rules.

T7. Keywords degrees of freedom; redistribution; nonterminality; plurality

F10. Non-Decreasing Expressive Capacity Under Redistribution

T0. Term Header - ID: F10 - Name: Non-Decreasing Expressive Capacity Under Redistribution - Category: Foundational Condition - Level: Meta

T1. Statement For any admissible redistribution $R : X \to Y$, expressive capacity is regulated such that $\mathcal{E}(Y) \geq \mathcal{E}(X)$.

T2. Scope - Applies to: redistributions - Range: global - Contextual limits: ordering-only

T3. Dependencies (Definitional) - Depends on: {D1.3, D1.5}

T4. Necessity Witness - Regulatory loss: loss of ordering discipline - Downstream witness: Expressive-capacity ordering checks (VE3, AR7) lose a stable redistribution baseline.

T5. Minimality / Non-Redundancy This condition constrains admissible redistribution to be order-compatible with $\mathcal{E}$.

T6. Admissibility Interface - Supports VE3 and AR7 ordering enforcement.

T7. Keywords expressive capacity; monotonicity; redistribution; ordering

F11. Locality-Compatible Propagation

T0. Term Header - ID: F11 - Name: Locality-Compatible Propagation - Category: Foundational Condition - Level: Meta

T1. Statement For all admissible transformations and redistributions, propagation is regulated such that all propagation occurs only along adjacency-compatible pathways (D1.9).

T2. Scope - Applies to: transformations, redistributions - Range: global - Contextual limits: adjacency-relative

T3. Dependencies (Definitional) - Depends on: {D0.23, D1.1, D1.5, D1.7, D1.9}

T4. Necessity Witness - Regulatory loss: loss of containment - Downstream witness: Boundary formation and interaction become incoherent.

T5. Minimality / Non-Redundancy This condition uniquely forbids non-local transmission as a propagation mode, not merely domain or mediation. No other condition constrains propagation mode at this granularity.

T6. Admissibility Interface - Referenced throughout AR-level enforcement and axiomatic derivations.

T7. Keywords locality; propagation; adjacency; pathway

F12. Constraint Compatibility

T0. Term Header - ID: F12 - Name: Constraint Compatibility - Category: Foundational Condition - Level: Meta

T1. Statement For any evaluation target E, the set of all binding constraints applicable to E under this specification MUST NOT be incompatible (D0.20). Any evaluation target for which the applicable binding constraints are incompatible is inadmissible by definition.

T2. Scope - Applies to: configurations, transformations - Range: global - Contextual limits: none

T3. Dependencies (Definitional) - Depends on: {D0.10, D0.16, D0.17, D0.20}

T3b. Audit / Reflective References (Non-definitional) - Audit references: {F1–F14, VE1–VE8, AR1–AR11} - Role: F12 ranges over the applicable binding constraint families via closure/audit targets; these audit references introduce no constitutive force beyond the binding semantics of D0.16/D0.17/D0.20.

T4. Necessity Witness - Regulatory loss: loss of enforceability - Downstream witness: specification-wide consistency collapses without compatibility enforcement.

T5. Minimality / Non-Redundancy Constraint compatibility cannot be inferred from individual constraints. It uniquely enforces joint admissibility via joint satisfaction/enforceability.

T6. Admissibility Interface - Governs joint-applicability and joint-enforceability checks across binding constraints.

F13. Minimal Sufficiency

T0. Term Header - ID: F13 - Name: Minimal Sufficiency - Category: Foundational Condition - Level: Meta

T1. Statement No binding constraint, primitive definition, or admissibility rule is admissible within this specification (or within the CSS, when invoked) unless its necessity and non-redundancy are structurally witnessed under this specification Term Schema (T4 Necessity Witness and T5 Minimality / Non-Redundancy), relative to its declared dependencies. A candidate element is superfluous iff it can be removed without inducing any downstream loss of satisfaction, enforceability, or determinacy obligations within the kernel.

T2. Scope - Applies to: kernel terms, derived specifications, canonical derivations - Range: global - Contextual limits: meta-only; concerns admissibility of specification elements, not ontological minimality

T3. Dependencies (Definitional) - Depends on: {D0.16, D0.17, D0.18, D0.19}

T4. Necessity Witness - Regulatory loss: loss of minimality discipline - Downstream witness: closure, conformance, and uniqueness governance become contestable without a Core-internal sufficiency criterion.

T5. Minimality / Non-Redundancy This condition provides a global admissibility discipline that binds "minimal sufficiency" to the already-required per-term witness structure, avoiding evaluator-dependent minimalism.

T6. Admissibility Interface - Used when admitting or auditing kernel terms, and when evaluating canonical derivations or suite modules for superfluous structure.

T7. Keywords minimality; sufficiency; kernel; non-redundancy

F14. Distinction Scope Closure

T0. Term Header - ID: F14 - Name: Distinction Scope Closure - Category: Foundational Condition - Level: Meta

T1. Statement For any evaluation target E, if binding material introduces a distinction between an interior scope of application and an exterior context and uses that distinction with constraining force (to determine applicability, satisfaction, enforceability, propagation, or admissibility), then the binding material MUST explicitly declare exactly one of the following scope postures:

1) Closed-Scope (No Cross-Scope Import): The binding material makes no use of the exterior context to justify, enforce, or validate any binding constraint application whose interior scope is the declared scope. In particular, no cross-scope propagation from exterior context into interior scope is asserted or relied upon.
2) Coupled-Scope (Explicit Interface Representation): Any reliance of the interior scope on the exterior context is represented by an explicit interface representation (D0.29) whose interface carrier I is a pathway (D1.9) or recursion (D1.10), and whose mediation constraints are enforceable under D0.17. Any cross-scope reliance asserted without such an explicit interface representation is unenforceable (D0.17) and therefore inadmissible.

This condition governs only the admissibility of how distinctions may be used in binding material and introduces no ontological claim.

T2. Scope - Applies to: binding constraints, canonical derivations, transformations, redistributions - Range: global - Contextual limits: representational-only; distinction-relative

T3. Dependencies (Definitional) - Depends on: {D0.16, D0.17, D0.20, D0.23, D1.1, F11, D0.29, D1.9, D1.10}

T4. Necessity Witness - Regulatory loss: loss of enforceability via implicit cross-boundary reliance - Downstream witness: boundary/interface reasoning becomes structurally uncheckable because cross-boundary constraint action is permitted without an explicit, enforceable interface.

T5. Minimality / Non-Redundancy This condition is the minimal closure rule required to prevent cross-boundary reliance from remaining structurally implicit. It introduces no new primitives, does not select adjacency, and does not require coupling.

T6. Admissibility Interface - Requires an explicit scope posture declaration (Closed-Scope or Coupled-Scope) for any binding distinction. - If Closed-Scope is declared, prohibits cross-boundary import from N into binding constraint application on L. - If Coupled-Scope is declared, requires an explicit interface representation (D0.29) with interface carrier a pathway (D1.9) or recursion (D1.10), enforceable under D0.17.

T7. Keywords scope; distinction; boundary; interface; coupling

VE1. Supported Emergence

T0. Term Header - ID: VE1 - Name: Supported Emergence - Category: Valid Emergence Condition - Level: Meta

T1. Statement For all admissible emergences $T : X \to Y$, emergence is regulated such that Y arises only through an admissible redistribution or recursion supported by X.

T2. Scope - Applies to: emergences - Range: global - Contextual limits: Core-internal pathways

T3. Dependencies (Definitional) - Depends on: {D0.26, D1.0, D1.5, D1.8, D1.10}

T4. Necessity Witness - Regulatory loss: loss of enforceability - Downstream witness: Emergence becomes unconstrained without support conditions.

T5. Minimality / Non-Redundancy Supported emergence uniquely prevents exogenous structure introduction. No other condition enforces emergence provenance.

T6. Admissibility Interface - Required for all VE- and AR-level emergence evaluation.

T7. Keywords emergence; support; provenance

VE2. Constraint Preservation

T0. Term Header - ID: VE2 - Name: Constraint Preservation - Category: Valid Emergence Condition - Level: Meta

T1. Statement For all admissible emergences $T : X \to Y$, all Foundational Conditions remain satisfied under the emergent configuration.

T2. Scope - Applies to: emergences - Range: global - Contextual limits: none

T3. Dependencies (Definitional) - Depends on: {D1.8, F1–F14}

T4. Necessity Witness - Regulatory loss: loss of coherence regulation - Downstream witness: Emergent configurations lose Core-admissibility without constraint preservation.

T5. Minimality / Non-Redundancy This condition uniquely enforces inheritance of foundational regulation across emergence.

T6. Admissibility Interface - Used to validate emergent configurations.

T7. Keywords constraints; preservation; coherence

VE3. Expressive-Capacity Increase

T0. Term Header - ID: VE3 - Name: Expressive-Capacity Increase - Category: Valid Emergence Condition - Level: Meta

T1. Statement For all admissible emergences $T : X \to Y$, expressive capacity is regulated such that $\mathcal{E}(Y) > \mathcal{E}(X)$.

T2. Scope - Applies to: emergences - Range: global - Contextual limits: ordering-only

T3. Dependencies (Definitional) - Depends on: {D1.3, D1.8, F10}

T4. Necessity Witness - Regulatory loss: loss of emergence distinction - Downstream witness: Emergence becomes indistinguishable from redistribution without expressive increase.

T5. Minimality / Non-Redundancy Expressive increase uniquely distinguishes emergence from reconfiguration.

T6. Admissibility Interface - Required for emergence classification and minimality evaluation.

T7. Keywords expressive capacity; increase; emergence

VE4. Minimal Emergence

T0. Term Header - ID: VE4 - Name: Minimal Emergence - Category: Valid Emergence Condition - Level: Meta

T1. Statement For all admissible emergences $T : X \to Y$, emergence is regulated such that no admissible configuration exists with expressive capacity strictly between $\mathcal{E}(X)$ and $\mathcal{E}(Y)$.

T2. Scope - Applies to: emergences - Range: relative (per emergence) - Contextual limits: expressive ordering

T3. Dependencies (Definitional) - Depends on: {D1.3, D1.8, VE3}

T4. Necessity Witness - Regulatory loss: loss of minimality - Downstream witness: Emergence selection becomes underdetermined without minimality.

T5. Minimality / Non-Redundancy This condition uniquely enforces minimal expressive transitions.

T6. Admissibility Interface - Used in emergence ordering and uniqueness evaluation.

T7. Keywords minimality; emergence; ordering

VE5. No External Introduction

T0. Term Header - ID: VE5 - Name: No External Introduction - Category: Valid Emergence Condition - Level: Meta

T1. Statement For all admissible emergences $T : X \to Y$, no structural element is introduced that is not derivable from the source configuration under admissible redistribution or recursion.

T2. Scope - Applies to: emergences - Range: global - Contextual limits: Core-internal derivability only; derivability is evaluated relative to admissible redistribution and admissible recursion pathways connecting source and target configurations

T3. Dependencies (Definitional) - Depends on: {D0.8, D1.8}

T4. Necessity Witness - Regulatory loss: loss of closure - Downstream witness: Canonical closure (AR8) and supported emergence (VE1) lose determinacy without internal introduction restriction.

T5. Minimality / Non-Redundancy This condition uniquely enforces internal derivability of novelty under admissible emergence. No other Valid Emergence Condition prohibits exogenous structural introduction at the level of emergent content.

T6. Admissibility Interface - Referenced by emergence admissibility checks. - Constrains emergence content to Core-internal provenance under admissible redistribution or recursion.

T7. Keywords internality; derivability; emergence

VE6. Admissible Pathway Existence

T0. Term Header - ID: VE6 - Name: Admissible Pathway Existence - Category: Valid Emergence Condition - Level: Meta

T1. Statement For all admissible emergences $T : X \to Y$, there exists at least one admissible redistribution or recursion pathway connecting the source and emergent configurations.

T2. Scope - Applies to: emergences - Range: global - Contextual limits: pathway-relative; pathways are evaluated relative to the source–target configuration pair

T3. Dependencies (Definitional) - Depends on: {D1.0, D1.5, D1.9, D1.10, F6}

T4. Necessity Witness - Regulatory loss: loss of enforceability - Downstream witness: Emergence becomes ungrounded without pathway existence, rendering AR5 unenforceable.

T5. Minimality / Non-Redundancy This condition uniquely enforces pathway grounding of emergence as a representable connection constraint. No other condition enforces the existence of a Core-admissible connecting pathway.

T6. Admissibility Interface - Required for emergence validation. - Directly referenced by AR5.

T7. Keywords pathways; redistribution; recursion; grounding

VE7. Structural Non-Equivalence

T0. Term Header - ID: VE7 - Name: Structural Non-Equivalence - Category: Valid Emergence Condition - Level: Meta

T1. Statement For all admissible emergences $T : X \to Y$, the emergent configuration is regulated such that $Y \nsim X$.

T2. Scope - Applies to: emergences - Range: relative (source–target pair) - Contextual limits: structural equivalence is evaluated under adjacency-preserving isomorphism only

T3. Dependencies (Definitional) - Depends on: {D1.6}

T4. Necessity Witness - Regulatory loss: loss of emergence distinction - Downstream witness: Emergence collapses into equivalence without non-equivalence enforcement, rendering AR6 unenforceable.

T5. Minimality / Non-Redundancy Non-equivalence uniquely distinguishes emergence from trivial transformation. No other Valid Emergence Condition enforces novelty as non-isomorphism.

T6. Admissibility Interface - Used to validate emergent novelty. - Directly referenced by AR6 and AR9.

T7. Keywords non-equivalence; novelty; emergence

VE8. Coherence Preservation Under Emergence

T0. Term Header - ID: VE8 - Name: Coherence Preservation Under Emergence - Category: Valid Emergence Condition - Level: Meta

T1. Statement For all admissible emergences $T : X \to Y$, the emergent configuration Y admits bounded regulatory coherence under the constraint bundle $\mathcal{C}$ (D0.0).

T2. Scope - Applies to: emergences - Range: global - Contextual limits: bounded representability only; coherence is graded and enforced under bounded satisfaction

T3. Dependencies (Definitional) - Depends on: {D0.0, F1–F14}

T4. Necessity Witness - Regulatory loss: loss of representability - Downstream witness: Emergent configurations become inadmissible or undefined without coherence preservation under emergence, rendering AR4 unenforceable.

T5. Minimality / Non-Redundancy This condition uniquely enforces bounded coherence across emergence. No other Valid Emergence Condition enforces coherence preservation as a global admissibility inheritance constraint.

T6. Admissibility Interface - Required for emergence admissibility and continuation. - Directly referenced by AR4.

T7. Keywords coherence; boundedness; emergence

Section 4 Closure Statement

(Non-Normative · Summary)

Valid Emergence Conditions VE1–VE8 are complete and apply globally to emergence governed by this specification. Admissibility of emergences is characterized by D0.1 (Admissibility Characterization) and regulated by VE1–VE8 (with enforcement via the applicable Admissibility Rules). All object-level entities referenced by VE1–VE8 are derived from D1.0–D1.10 and regulated by F1–F14. All object-level entities referenced in Valid Emergence Conditions are defined in §2 (Primitive Definitions).

5. ADMISSIBILITY RULES

(Meta-Structural · Normative · Enforcement)

Admissibility Rules specify how binding constraints are enforced over configurations, transformations, redistributions, pathways, recursions, and emergences. They regulate continuation and nonterminality, not acceptance or exclusion.

AR1. Continuity Enforcement

T0. Term Header - ID: AR1 - Name: Continuity Enforcement - Category: Admissibility Rule - Level: Meta

T1. Statement For all admissible transformations and redistributions, any redistribution SHALL satisfy continuity (D1.2) so that restriction-consistent locality is preserved under bounded redistribution.

T2. Scope - Applies to: transformations, redistributions - Range: global - Contextual limits: none

T3. Dependencies (Definitional) - Depends on: {D0.23, D0.24, D1.1, D1.5, D1.7, F1}

T4. Necessity Witness - Regulatory loss: loss of representability - Downstream witness: Redistribution pathways become inadmissible without enforceable continuity.

T5. Minimality / Non-Redundancy This rule uniquely operationalizes continuity at the enforcement level.

T6. Admissibility Interface - Enforces F1 during transformation, redistribution, and recursion.

T7. Keywords continuity; enforcement; redistribution

AR2. Non-Negation Enforcement

T0. Term Header - ID: AR2 - Name: Non-Negation Enforcement - Category: Admissibility Rule - Level: Meta

T1. Statement For all admissible redistributions and recursions, non-negation is enforced such that no transformation simultaneously enforces and nullifies the same structural constraint.

T2. Scope - Applies to: redistributions, recursions - Range: global - Contextual limits: none

T3. Dependencies (Definitional) - Depends on: {D0.9, F2, D1.5, D1.7, D1.10}

T4. Necessity Witness - Regulatory loss: loss of enforceability - Downstream witness: Structural equivalence and coherence become undecidable under contradictory enforcement.

T5. Minimality / Non-Redundancy This rule uniquely prevents contradictory redistribution and contradictory recursion.

T6. Admissibility Interface - Enforces F2 across all admissible redistributions and recursions.

T7. Keywords non-negation; enforcement; consistency

AR3. Adjacency Enforcement

T0. Term Header - ID: AR3 - Name: Adjacency Enforcement - Category: Admissibility Rule - Level: Meta

T1. Statement For all admissible operations, adjacency preservation is enforced such that no interaction, propagation, or redistribution occurs outside the adjacency-bounded scope defined for the configuration.

T2. Scope - Applies to: transformations, redistributions, emergences - Range: global - Contextual limits: adjacency-relative

T3. Dependencies (Definitional) - Depends on: {D0.23, D0.24, D1.1, D1.7, D1.8, F3}

T4. Necessity Witness - Regulatory loss: loss of containment - Downstream witness: Coherence evaluation collapses without enforced locality.

T5. Minimality / Non-Redundancy This rule uniquely enforces locality at the admissibility level.

T6. Admissibility Interface - Enforces F3 during all admissible transformations, redistributions, and emergences.

T7. Keywords adjacency; locality; enforcement

AR4. Bounded Coherence Enforcement

T0. Term Header - ID: AR4 - Name: Bounded Coherence Enforcement - Category: Admissibility Rule - Level: Meta

T1. Statement For all admissible configurations and emergences, coherence is enforced as bounded representability under this specification constraint set.

T2. Scope - Applies to: configurations, emergences - Range: global - Contextual limits: bounded satisfaction only; coherence is graded and regulated

T3. Dependencies (Definitional) - Depends on: {F4, VE8, D1.8}

T4. Necessity Witness - Regulatory loss: loss of representability - Downstream witness: Admissibility becomes binary or undefined without bounded coherence enforcement.

T5. Minimality / Non-Redundancy This rule uniquely enforces graded coherence at the admissibility level.

T6. Admissibility Interface - Governs coherence checks across canonical enforcement.

T7. Keywords coherence; boundedness; enforcement

AR5. Pathway Admissibility

T0. Term Header - ID: AR5 - Name: Pathway Admissibility - Category: Admissibility Rule - Level: Meta

T1. Statement For all admissible emergences $T : X \to Y$ and continuations, at least one admissible redistribution or recursion pathway must exist connecting source and target configurations.

T2. Scope - Applies to: emergences, continuations - Range: global - Contextual limits: pathway-relative; evaluated relative to the source–target configuration pair; continuations are as defined by D0.2

T3. Dependencies (Definitional) - Depends on: {VE6, F6, D1.9, D1.10, D0.2}

T4. Necessity Witness - Regulatory loss: loss of enforceability - Downstream witness: Emergence and continuation become ungrounded without pathway enforcement.

T5. Minimality / Non-Redundancy This rule uniquely enforces pathway grounding at the admissibility level.

T6. Admissibility Interface - Used to validate emergence and nonterminal continuation. - Enforces VE6 as an AR-level admissibility requirement.

T7. Keywords pathways; admissibility; continuation

AR6. Structural Non-Equivalence Enforcement

T0. Term Header - ID: AR6 - Name: Structural Non-Equivalence Enforcement - Category: Admissibility Rule - Level: Meta

T1. Statement For all admissible emergences $T : X \to Y$, structural non-equivalence between source and emergent configurations is enforced.

T2. Scope - Applies to: emergences - Range: relative (source–target pair) - Contextual limits: structural equivalence under adjacency-preserving isomorphism only

T3. Dependencies (Definitional) - Depends on: {VE7, D1.6, D1.8}

T4. Necessity Witness - Regulatory loss: loss of emergence distinction - Downstream witness: Emergence collapses into trivial transformation without non-equivalence enforcement.

T5. Minimality / Non-Redundancy This rule uniquely enforces emergent novelty.

T6. Admissibility Interface - Used to validate emergent structure.

T7. Keywords non-equivalence; emergence; novelty

AR7. Expressive-Capacity Ordering Enforcement

T0. Term Header - ID: AR7 - Name: Expressive-Capacity Ordering Enforcement - Category: Admissibility Rule - Level: Meta

T1. Statement For all admissible emergences $T : X \to Y$ and recursions, expressive-capacity ordering is enforced such that admissible transitions respect monotonic increase.

T2. Scope - Applies to: emergences, recursions - Range: global - Contextual limits: ordering-only

T3. Dependencies (Definitional) - Depends on: {F10, D1.3, D1.8, D1.9, D1.10}

T4. Necessity Witness - Regulatory loss: loss of enforceability - Downstream witness: Minimal emergence becomes undecidable without ordering enforcement.

T5. Minimality / Non-Redundancy This rule uniquely enforces directional emergence and recursion ordering without dynamics.

T6. Admissibility Interface - Used to evaluate admissible emergence and recursion ordering. - Enforces VE3 and supports VE4 minimality evaluation.

T7. Keywords expressive capacity; ordering; enforcement

AR8. Canonical Closure Enforcement

T0. Term Header - ID: AR8 - Name: Canonical Closure Enforcement - Category: Admissibility Rule - Level: Meta

T1. Statement For all purported canonical derivations relative to this Core specification version (as characterized by AR11) and for all admissible emergences, closure is enforced such that all referenced structures remain Core-internal (or, where explicitly suite-scoped, CSS-internal) and remain subject to the binding constraints of the applicable specification(s).

T2. Scope - Applies to: canonical derivations, emergences - Range: global - Contextual limits: Core-internal only; suite-scoped only when CSS is explicitly invoked

T3. Dependencies (Definitional) - Depends on: {D0.7, D0.19, S-8, S-9}

T3b. Audit / Reflective References (Non-definitional) - Audit references: {AR11} - Role: AR8 may reference AR11's conformance characterization as an audit target, without making AR11 a definitional prerequisite.

T4. Necessity Witness - Regulatory loss: loss of closure; loss of enforceability - Downstream witness: canonical derivations can smuggle external primitives or external enforcement mechanisms without closure enforcement.

T5. Minimality / Non-Redundancy This rule uniquely enforces closure discipline for derived canonical material; no prior admissibility rule enforces cross-document structural internality.

T6. Admissibility Interface - Governs all canonical derivations and suite-scoped extensions.

T7. Keywords closure; enforcement; derivation; suite; internality

AR9. Nonterminality Under an Admissible Driver

T0. Term Header - ID: AR9 - Name: Nonterminality Under an Admissible Driver - Category: Admissibility Rule - Level: Meta

T1. Statement For any configuration admitting bounded regulatory coherence and an unresolved admissible internal driver, nonterminality is enforced such that at least one admissible redistribution yields a structurally non-equivalent configuration.

T2. Scope - Applies to: configurations - Range: relative (per configuration) - Contextual limits: configurations with unresolved admissible internal drivers only

T3. Dependencies (Definitional) - Depends on: {D1.4, D1.5, F4, F6, VE7}

T4. Necessity Witness - Regulatory loss: loss of nonterminal enforcement - Downstream witness: Forced emergence cannot be derived without nonterminality enforcement.

T5. Minimality / Non-Redundancy This rule uniquely forces continuation under admissible strain. No other rule enforces nonterminality.

T6. Admissibility Interface - Governs forced emergence and continuation across this specification.

T7. Keywords nonterminality; drivers; enforcement

AR10. Bounded Nonterminal Emergence

T0. Term Header - ID: AR10 - Name: Bounded Nonterminal Emergence - Category: Admissibility Rule - Level: Meta

T1. Statement For any admissible configuration X admitting an unresolved admissible internal driver, nonterminality is enforced such that there exists at least one admissible redistribution $R : X \to Y$ yielding a structurally non-equivalent configuration ($Y \nsim X$), and such that this condition remains satisfiable under repeated application while all such redistributions preserve bounded coherence (AR4), continuity (AR1), adjacency (AR3), and non-negation (AR2).

T2. Scope - Applies to: configurations, emergences - Range: global - Contextual limits: configurations with unresolved admissible internal drivers only; repeated application is evaluated as recursion over admissible redistributions

T3. Dependencies (Definitional) - Depends on: {D0.24, D1.9, F4, F5, F11, AR1, AR2, AR3, AR4, AR9}

T4. Necessity Witness - Regulatory loss: loss of enforceability; loss of boundedness - Downstream witness: Recursive continuation under AR9 becomes unenforceable without a bounded, repeatable nonterminality postulate.

T5. Minimality / Non-Redundancy This rule uniquely regulates recursive continuation under nonterminality by preventing both terminal stabilization and unbounded divergence. No prior admissibility rule constrains recursive depth while preserving bounded coherence.

T6. Admissibility Interface - Governs recursive continuation following nonterminal emergence. - Constrains admissible redistribution across successive admissible continuation layers.

T7. Keywords nonterminality; boundedness; recursion; emergence

AR11. Core Conformance Characterization

T0. Term Header - ID: AR11 - Name: Core Conformance Characterization - Category: Admissibility Rule - Level: Meta

T1. Statement For any purported Core term or purported canonical derivation relative to this Core specification version, Core conformance is enforced such that it is Canonical iff: 1. all Kernel Terms it asserts are expressed as schema-complete Core terms satisfying this specification Term Schema requirements (T0–T6; T7 optional), 2. all dependencies are Core-internal and explicit, and Core-term dependencies are well-founded under an acyclic dependency ordering; schema-level constraint references (S-#) are permitted as binding governance references, 3. no object-level entity appears in any Kernel Term unless it is a Primitive Definition in §2 or is explicitly declared as meta-structural vocabulary within this Core specification, 4. no canonical derivation introduces new primitives, Kernel constraints, or enforcement mechanisms beyond those in this Core specification, and 5. non-normative sections introduce no binding constraints.

T2. Scope - Applies to: derivations, Core terms - Range: global - Contextual limits: relative to this Core specification version only

T3. Dependencies (Definitional) - Depends on: {F13, D0.7, S-2, S-3, S-5, S-6, S-7, S-8, S-9}

T3b. Audit / Reflective References (Non-definitional) - Audit references: {AR8} - Role: AR11 may characterize how canonical closure enforcement (AR8) is applied, without requiring AR8 for definitional interpretation.

T4. Necessity Witness - Regulatory loss: loss of enforceability - Downstream witness: Canonical status of downstream derivations becomes underdetermined without an enforceable conformance characterization.

T5. Minimality / Non-Redundancy This rule uniquely makes Canonical conformance this specification-internally determinate (non-procedural) by binding schema completeness, dependency closure, primitive closure, and non-extension into a single admissibility characterization. No prior admissibility rule defines Canonical conformance as an enforceable criterion for derivations.

T6. Admissibility Interface - Governs admissibility of canonical extensions, derivations, and any text asserting canonical status under this Core specification. - Provides the enforcement interface by which closure and schema authority constrain derived canonical specifications.

T7. Keywords conformance; canonicity; schema; closure; derivation

Section 5 Closure Statement

(Non-Normative · Summary)

Conformance and admissibility are characterized by AR11 (this specification Conformance Characterization) and D0.1 (Admissibility Characterization). This summary is referential only; see AR1–AR11, F1–F14, and VE1–VE8 for binding conditions and enforcement posture. All object-level entities referenced in Admissibility Rules are defined in §2 (Primitive Definitions).

Appendix: Reflexive Closure Certificate Template

(Non-Normative · Template) RCC TEMPLATE v1

```
RCC-ID: <unique identifier>
Applies-To: SC-CORE-000001 - The Coherence Core Specification v1.2.5 (Issuance ID: v39c)
Issued-By: <issuer term / authority>
Issued-On: <date>

1) Definitional Dependency DAG (T3)
- Nodes: <Kernel Term IDs>
- Edges (T → U means T depends on U): <T → U>

2) Topological Order Witness (S-5)
- Order: <term ids in topological order>
- Witness: this ordering is a topological ordering of the definitional dependency DAG; therefore the

3) Binding Inventory (Binding clauses)
- Kernel Terms (T0-T6): <term ids>
- Schema-Level Constraints: <S-#>
- Conformance Clauses: <AR#>
- Other Canonical-Binding Sections: <sections>

4) Binding Reachability Map (S-8)
For each binding clause, list required symbols/predicates/constructs and the definitional reachabili
```

```
5) Audit Reference Inventory (T3b) (S-9)
- Audit edges: <T ⇝ U>
- Non-creativity assertion: audit references are classificatory/reporting only and introduce no bind

6) Canonical Material Snapshot (D0.7)
- Canonical Binding: <sections>
- Canonical Non-Normative (if any): <sections>
- Non-Normative (firewalled under S-2): <sections>

7) Schema Token Conformance Report (S-10)
- Non-literal tokens detected: <none | list>

8) Typing Discipline Report (S-11)
- Emergence quantification conforms (T : X → Y): <yes/no; list exceptions>

9) Binding Vocabulary Audit (S-12)
- Technical constructs not grounded as Core terms or admitted apparatus: <none | list; disposition>
```

