The Canonical Completion Object Theorem for Shadow Theory

Canonical Completion as a Certified Initiality Criterion in a Public Admissible Completion Category

Jeremy Rodgers*
Independent Researcher

July 2026

Abstract

Paper 1 of this stack established that a public readout or quotient presentation need not determine realization-equivalence. Paper 2 then showed that readout loss becomes publicly obstructive only when it is certified as an active, uncleared failure of an essential public closure slot. The present paper supplies the next step: it defines when a response to such a completion need is *canonical*.

The answer is categorical but deliberately conditional. A public completion is canonical exactly when it is an initial object in the public admissible completion category determined by a public target Q, its public closure target P_Q, and certified public category data. Thus the Canonical Completion Object, when it exists, is

$$\mathsf{CCO}^{\mathrm{pub}}_{\mathfrak{K}}(Q, P_Q) = \mathrm{Init}\big(\mathsf{Comp}^{\mathrm{pub}}_{\mathfrak{K}}(Q, P_Q)\big).$$

This is a canonicality criterion, not an existence theorem. Completion need does not imply canonical completion. A category-forming handoff does not by itself supply objects, morphisms, identities, composition, or category laws. Candidate completions, branch completions, open-comparability states, failed proposals, and empty categories are all noncanonical unless sound certified initiality is supplied. We formalize the public completion context, the certified category-data discipline, typed public refinement morphisms with explicit hom-typing, certificate-gated composition safety, the initiality-based canonicality criterion, and a seven-valued ordered status classifier. We then define the public output card that Paper 3 passes to Paper 4, while explicitly denying that Paper 3 emits a Tier-1 artifact or an empirical validation. The framework is presented as a public formal theory of readout, completion, admissibility, initiality, status, and handoff.

Paper status in the stack. This is Paper 3 of the Shadow Theory stack. It depends on Paper 1 (Readout Non-Equivalence) and Paper 2 (Completion Necessity), whose results it imports as stated assumptions, and it hands off to Paper 4 (Tier-1 Shadow Compiler), which owns all down-compilation. Paper 3 defines the canonicality criterion and its public output card only; it does not compile Tier-1 artifacts, and it makes no empirical claim.

Contents

1 Introduction **2**

2 Upstream Imports and Nonclaims **3**
2.1 Import from Paper 1 3
2.2 Import from Paper 2 3
2.3 Default category-forming handoffs 4

*Website: everythingequation.com

3 Public Completion Context **4**

4 Public-Admissible Completions **5**

5 Certified Public Category Data and Hom-Typing **6**

6 Public Refinement Morphisms **7**

7 Composition Safety and the Category Laws **7**

8 Certified Public Completion Category Formation **9**

9 Initiality and the Canonicality Criterion **10**

10 Public CCO Status Classification **12**

11 Output Card and Handoff to Paper 4 **18**

12 Scope Guards and Forbidden Promotions **19**

13 Schematic Examples **19**

14 Conclusion **22**

A Expanded Definitions and Adversarial Cases **22**

1 Introduction

Shadow Theory separates public readout from realization structure. Paper 1 [1] showed that a public readout can present a quotient-like object without determining the realization structure from which that readout arose. Paper 2 [2] then introduced a public discipline for recognizing when readout loss matters: the loss must be certified as active, uncleared, and closure-relevant relative to an independent public closure target.

This paper addresses the next question in the stack. Suppose a public target Q has received a completion need or completion handoff from Paper 2. What makes a proposed completion *canonical*?

The answer cannot be "whatever completion is useful," nor "whatever completion first repairs the visible defect." The Shadow Theory stack requires a canonicality criterion that separates the *existence* of a completion response from the *canonicality* of that response. The criterion used here is initiality. A public completion is canonical when it is the initial object of the public admissible completion category for (Q, P_Q), relative to certified public category data. In this sense the canonical completion is the public-admissible completion from which every other admissible completion receives a unique public refinement morphism.

The main result is therefore a criterion rather than a construction:

canonical completion = certified initiality in the public admissible completion category.

The word "certified" is essential and is not decorative. This paper does not claim that completion categories automatically exist, nor that completion need automatically produces initial objects. Category formation requires supplied and certified public category data. Initiality requires a sound initiality certificate, backed either by a proof of the universal-arrow property or by a public initiality contract that is itself certified sound. Everything downstream — the status classifier, the output card, the handoff to Paper 4 — is gated on these certificates.

Contribution and boundaries. The contribution is a public categorical framework with five hard built-in boundaries that are preserved throughout and restated formally in Section 12:

$$\mathsf{CompletionNeed} \not\Rightarrow \mathsf{CCO}, \qquad \mathsf{Candidate} \not\Rightarrow \mathsf{Certified}, \qquad \mathsf{Branch} \not\Rightarrow \mathsf{CCO},$$

$$\mathsf{CCO}^{\mathrm{pub}}_{\mathfrak{K}}(Q, P_Q) \not\Rightarrow \mathsf{T1Artifact}^{\mathrm{pub}}_{Q}, \qquad \mathsf{CCO}^{\mathrm{pub}}_{\mathfrak{K}}(Q, P_Q) \not\Rightarrow \mathsf{EmpiricalValidation}.$$

Paper 3 stops at the CCO output card and the Paper 4 handoff. It does not perform Tier-1 artifact compilation; that is owned by Paper 4 [3]. It does not adjudicate empirical validity. The paper is written entirely in the terminology of Shadow Theory: admissibility, certificates, output policy, and handoff objects are treated as explicit formal data of the present framework.

The result is intended for later mathematical-physics use through the Shadow Theory stack, but the present paper proves only the categorical completion criterion and its handoff discipline; it introduces no physical model, and any physical reading is deferred to the downstream stack.

Organization. Section 2 records the upstream imports from Papers 1 and 2 and the default category-forming and non-category handoff classes. Section 3 fixes the public completion context and the universe convention. Section 4 defines public-admissible completions, completion objects, and completion types. Section 5 defines the certified public category data and the explicit hom-typing discipline, and Section 6 defines typed public refinement morphisms. Section 7 proves the composition-safety and category-law lemmas. Section 8 states the conditional category-formation theorem (Theorem 8.2). Section 9 defines initiality, the initiality certificate and its soundness, the Canonical Completion Object, and proves the canonicality criterion (Theorem 9.8). Section 10 develops the refinement preorder, certified non-refinement, and the seven-valued ordered status classifier (Theorem 10.13). Section 11 defines the output card and the Paper 4 handoff (Theorem 11.4). Section 12 states the scope guards and forbidden promotions. Section 13 gives schematic examples. Section 14 concludes. An appendix collects expanded definitions and adversarial cases.

2 Upstream Imports and Nonclaims

2.1 Import from Paper 1

Paper 1 supplies the readout non-equivalence principle: for a realization/readout map $\rho : \mathcal{C} \to S$ with $S = \mathrm{Im}(\rho)$,

$$S \cong \mathcal{C}/\!\sim_\rho \quad \not\Rightarrow \quad S \cong_{\mathrm{real}} \mathcal{C}.$$

An exact public quotient presentation need not be realization-structure equivalent to the domain it presents. This result motivates completion machinery, but by itself it implies neither a completion need, nor a completion category, nor a canonical completion.

2.2 Import from Paper 2

Paper 2 supplies the public obstruction interface. A readout loss becomes publicly obstructive only when it is certified as an active, uncleared failure of an essential public closure slot. On that basis Paper 2 may emit a completion need, an obstruction, a downgrade, a no-go, a residue route, or a non-category route. Paper 3 imports only the resulting public handoff; it does not re-prove Paper 1 or Paper 2, and it treats the imported handoff as data to be checked, not as a licence for canonicality.

2.3 Default category-forming handoffs

The following need classes are the default Paper-3 category-forming handoff classes, *provided that* certified public category data are also supplied:

$$\mathsf{RecoveryNeeded},\quad \mathsf{EnrichmentNeeded},\quad \mathsf{TypedCompletionNeeded},\quad \mathsf{SelectorNeeded}.$$

A category-forming handoff indicates that a completion category *may* be formed once the required public object, morphism, typing, identity, composition, closure, identity-law, associativity, and universe data are supplied and certified. It does not mean that the route name alone forms a category.

By contrast, the following are non-category routes by default for Paper-3 completion-category purposes:

$$\mathsf{CalibrationNeeded},\ \mathsf{ContractNeeded},\ \mathsf{CarrierNeeded},\ \mathsf{EffectiveRouteNeeded},$$

$$\mathsf{ResidueOnlyNeeded},\ \mathsf{NoCarrierStopNeeded},\ \mathsf{NoGoCandidateNeeded},\ \mathsf{DowngradeNeeded}.$$

Such routes may carry internal structure elsewhere in the Shadow Theory stack, but they are not treated as Paper-3 public completion categories for P_Q unless separate category data are supplied and certified.

Proposition 2.1 (Completion need does not imply canonical completion)**.**

$$\mathsf{CompletionNeedRecord}_Q^{\mathrm{pub}} \not\Rightarrow \mathsf{CCO}_{\mathfrak{K}}^{\mathrm{pub}}(Q, P_Q).$$

Proof. A completion need is an interface record, not a categorical result. Downstream of it, the completion may fail to form a category (no certified category data), may form an empty category (no admissible objects), may produce one or more candidates without certified initiality, may branch under certified non-dominance, may remain in open comparability, or may fail public admissibility outright. Each of these is a distinct terminal state of the classifier of Section 10 in which no soundly certified initial object is emitted. Hence the need record does not entail $\mathsf{CCO}_{\mathfrak{K}}^{\mathrm{pub}}(Q, P_Q)$. □

3 Public Completion Context

Definition 3.1 (Public target and closure target)**.** Let Q be a public target. Let P_Q be the independent public closure target associated with Q. The closure target records what must be preserved, recovered, enriched, selected, typed, or otherwise completed before a public output concerning Q may claim the relevant closure status. The predicate $\mathsf{Cl}_{P_Q}^{\mathrm{pub}}$ denotes public closure with respect to P_Q.

Definition 3.2 (Public completion context)**.** A public completion context for (Q, P_Q) is a tuple

$$\mathfrak{K}_Q^{\mathrm{pub}} = (Q,\ P_Q,\ \mathcal{L}_Q^{(0)},\ \mathcal{A}_Q^{(0.5)},\ \mathcal{P}_Q^{\mathrm{pub}},\ \mathsf{NeedClass}_Q,\ \mathsf{ObstructionCard}_{P_Q}^{\mathrm{pub}},\ R_Q^{\mathrm{total}},\ \sigma_Q^{\mathrm{total}},$$

$$\mathsf{CarrierRoute}_Q,\ \mathsf{AllowedClaims}_Q,\ \mathsf{ForbiddenPromotion}_Q,\ \mathsf{Adm}_Q^{\mathrm{pub}},\ \mathsf{Read}_Q^{\mathrm{pub}},\ \mathsf{Cl}_{P_Q}^{\mathrm{pub}}).$$

The context records, in order, the public target and closure target; the public law-level and admissibility layers $\mathcal{L}_Q^{(0)}$ and $\mathcal{A}_Q^{(0.5)}$ imported from the stack; the public readout packet $\mathcal{P}_Q^{\mathrm{pub}}$; the active need class; the public obstruction card; the total residue state R_Q^{total} and total status state $\sigma_Q^{\mathrm{total}}$; the carrier route; the allowed public claims; the forbidden promotions; and the public admissibility predicate, public readout rule, and public closure predicate. The context is the fixed public data relative to which all subsequent categorical notions are defined.

Definition 3.3 (Universe convention)**.** All public completion categories are formed relative to a declared universe $\mathcal{U}_{\mathrm{pub}}$. Object collections are $\mathcal{U}_{\mathrm{pub}}$-classes unless otherwise stated. Hom-collections are $\mathcal{U}_{\mathrm{pub}}$-sets when local smallness is required; local smallness is then included as part of the supplied public category data (Definition 5.1). Initiality and uniqueness claims in this paper are interpreted inside the declared public universe. The universe and local-smallness assumptions ensure that the hom-classes and the quantification "for every public-admissible completion C" in the definition of initiality (Definition 9.1) are formed inside the declared category universe, so that the initial-arrow quantification ranges over a legitimate $\mathcal{U}_{\mathrm{pub}}$-collection rather than an unbounded totality.

Remark 3.4 (Notation convention for $\mathfrak{K}$). Throughout the paper, when a public context $\mathfrak{K}_Q^{\mathrm{pub}}$ is fixed, the shorter symbol $\mathfrak{K}$ is used in subscripts, as in

$$\mathsf{Comp}_{\mathfrak{K}}^{\mathrm{pub}}(Q, P_Q),$$

to denote the category and category-data formed relative to that fixed public context. No independent bare context $\mathfrak{K}$ is being introduced; every $\mathfrak{K}$-subscripted object is understood relative to the fixed $\mathfrak{K}_Q^{\mathrm{pub}}$.

4 Public-Admissible Completions

Definition 4.1 (Public-admissible completion)**.** A proposed completion C of (Q, P_Q) is *public-admissible* relative to $\mathfrak{K}_Q^{\mathrm{pub}}$ when $\mathsf{Adm}_Q^{\mathrm{pub}}(C) = 1$. The admissibility predicate requires at least the following clauses:

(A1) **Target preservation.** Q and P_Q are not replaced or mutated.

(A2) **Readout compatibility.** The public readout is preserved or refined without changing its declared scope.

(A3) **Status compatibility.** Public status updates do not overpromote the output.

(A4) **Residue accounting.** Every residue is projected, bounded, blocking, eliminated, carried, discharged, or reclassified.

(A5) **No hidden unclassified residue.** $R_{\mathrm{hidden}}^{\mathrm{unclassified}} = \varnothing$.

(A6) **Carrier-route discipline.** The carrier route is not upgraded without public authorization.

(A7) **Forbidden-promotion discipline.** Active forbidden promotions remain active unless validly discharged by a public theorem or public contract.

(A8) **Need-class repair.** The active public need class is addressed.

(A9) **Obligation visibility.** Proof, calibration, consistency, residue, and carrier obligations are explicit.

(A10) **Public language only.** The completion is expressible in Shadow Theory language.

Remark 4.2 (Frozen admissibility predicate). For each application of the Paper-3 classifier, the predicate $\mathsf{Adm}_Q^{\mathrm{pub}}$ is frozen as part of the public completion context $\mathfrak{K}_Q^{\mathrm{pub}}$. The clauses (A1)–(A10) are the standing required clauses; the "at least" phrasing permits a context to declare additional standing clauses, but only as part of its fixed data. Further admissibility clauses may be added only by forming a new public context $\mathfrak{K}_{Q,+}^{\mathrm{pub}}$ with a new predicate $\mathsf{Adm}_{Q,+}^{\mathrm{pub}}$. Such an extension may change the object collection or the emitted status, but it is then a *new* classification problem, not a silent modification of the old one. In particular every classification result in Section 10 is stated relative to a fixed frozen $\mathsf{Adm}_Q^{\mathrm{pub}}$.

Definition 4.3 (Public completion object)**.** A public completion object has the schematic form

$$C_Q^{\mathrm{pub}} = (\mathsf{CompletionType}_Q,\ \mathsf{CompletionData}_Q,\ \mathsf{RepairRoute}_Q,\ \mathsf{ResidueAction}_Q,\ \mathsf{StatusUpdate}_Q,\ \mathsf{ReadoutUpdate}_Q,\ \mathsf{ObligationUpdate}_Q,\ \mathsf{CarrierRouteUpdate}_Q,\ \mathsf{AllowedClaimsUpdate}_Q,\ \mathsf{ForbiddenPromotionUpdate}_Q).$$

It enters the public completion category only if it is public-admissible in the sense of Definition 4.1.

Definition 4.4 (Completion types and data signatures)**.** The default completion types are

$$\mathsf{RecoveryCompletion},\quad \mathsf{EnrichmentCompletion},\quad \mathsf{SelectorCompletion},\quad \mathsf{TypedStructuralCompletion},\quad \mathsf{HybridCompletion},$$

each carrying a public data signature. A recovery completion carries recovery data, a role-preservation condition, a residue update, and a status update. An enrichment completion carries an enriched public packet, an inclusion or refinement map, a residue update, and a status update. A selector completion carries a selector with its domain and codomain, a selector-admissibility condition, a residue update, and a status update. A typed structural completion carries a completed public packet, a completion inclusion or repair map, a public closure predicate, a residue update, a status update, and an obligation set. A hybrid completion carries a finite compatible family of completion components, a compatibility certificate, a merged residue update, a merged status update, and a merged obligation set.

A hybrid completion is public-admissible only when all components preserve the same (Q, P_Q), their residue and status updates are jointly compatible, their carrier-route updates do not conflict, their forbidden promotions are jointly preserved, and their obligations are merged without suppression.

5 Certified Public Category Data and Hom-Typing

Definition 5.1 (Public category data)**.** The public completion category is *not* inferred from a need-class label. It is formed only when certified public category data are supplied. For $\mathfrak{K}_Q^{\mathrm{pub}}$, define

$$\mathsf{PublicCategoryData}_{\mathfrak{K}} = (\mathrm{Obj}_{\mathfrak{K}},\ \mathrm{Hom}_{\mathfrak{K}},\ \mathsf{TypeSig}_{\mathfrak{K}},\ \mathsf{HomTyping}_{\mathfrak{K}},\ \mathrm{id}_{\mathfrak{K}},\ \mathrm{comp}_{\mathfrak{K}},\ \mathsf{ClosureCert}_{\mathfrak{K}},\ \mathsf{IdentityLawCert}_{\mathfrak{K}},\ \mathsf{AssociativityCert}_{\mathfrak{K}},\ \mathsf{Universe}_{\mathfrak{K}}).$$

Here $\mathrm{Obj}_{\mathfrak{K}}$ is the collection of public-admissible completion objects; $\mathrm{Hom}_{\mathfrak{K}}(C_1, C_2)$ is the typed collection of public refinement morphisms from C_1 to C_2; $\mathsf{TypeSig}_{\mathfrak{K}}$ assigns completion-data signatures to objects; $\mathsf{HomTyping}_{\mathfrak{K}}$ determines whether a hom-class is well typed, empty, or requires a bridge; $\mathrm{id}_{\mathfrak{K}}$ assigns identities; $\mathrm{comp}_{\mathfrak{K}}$ supplies composition; and $\mathsf{ClosureCert}_{\mathfrak{K}}$, $\mathsf{IdentityLawCert}_{\mathfrak{K}}$, $\mathsf{AssociativityCert}_{\mathfrak{K}}$ certify closure under composition, the identity laws, and associativity respectively. $\mathsf{Universe}_{\mathfrak{K}}$ records the universe and local-smallness data of Definition 3.3. All ten components are *supplied and certified* public data, not quantities inferred from the presence of a need class.

Remark 5.2 (Certification is data, not inference). The distinction fixed by Definition 5.1 is the load-bearing one for the paper. A category-forming need class (Section 2) signals only that a category *may* be formed. Whether it *is* formed depends entirely on whether $\mathsf{PublicCategoryData}_{\mathfrak{K}}$ is supplied and its certificates pass. Nothing in this paper constructs $\mathsf{PublicCategoryData}_{\mathfrak{K}}$ from a route label.

Definition 5.3 (Hom-typing)**.** For public completions C_1, C_2, the hom-class $\mathrm{Hom}_{\mathfrak{K}}(C_1, C_2)$ is defined only when $\mathsf{HomTyping}_{\mathfrak{K}}$ permits it.

- If C_1 and C_2 share compatible completion-data signatures, *same-type* homs may be supplied.
- If the completion types differ, a typed public translation or refinement bridge must be supplied.
- If no such bridge is supplied, then

$$\mathrm{Hom}_{\mathfrak{K}}(C_1, C_2) = \varnothing.$$

For hybrid completions, homs into or out of the hybrid require componentwise compatible hom-data together with a merge-compatibility certificate. Thus heterogeneous homs never arise by default: they exist only when an explicit typed bridge and the relevant merge certificates are present.

Definition 5.4 (Bridge policy)**.** A *bridge policy* for $\mathsf{HomTyping}_{\mathfrak{K}}$ is the public rule specifying which typed translation/refinement bridges are admissible, which bridge omissions are recorded, and which heterogeneous hom-classes are consequently empty. The bridge policy is part of $\mathsf{PublicCategoryData}_{\mathfrak{K}}$ and is frozen before CCO status classification.

Changing the bridge policy changes the public category data, and hence changes the category $\mathsf{Comp}^{\mathrm{pub}}_{\mathfrak{K}}(Q, P_Q)$ itself; a bridge policy cannot be used silently after the fact to promote a preferred completion. Moreover, if bridge incompleteness is material to comparability or to the existence or uniqueness of a universal arrow, the classifier must return a noncertified status — such as CCO_OPEN_COMPARABILITY or CCO_CANDIDATE — rather than CCO_CERTIFIED. Thus a CCO verdict can never be manufactured by selectively withholding bridges: any such incompleteness that would affect initiality forces a noncertified status.

6 Public Refinement Morphisms

Definition 6.1 (Typed public refinement morphism)**.** A *public refinement morphism* $f : C_1 \to C_2$ is an element of $\mathrm{Hom}_{\mathfrak{K}}(C_1, C_2)$ satisfying two layers of conditions.

Typing layer. The morphism is well typed: the source and target completion types are declared, the hom-class is permitted by $\mathsf{HomTyping}_{\mathfrak{K}}$ (Definition 5.3), and the domain and codomain data are specified.

Public-constraint layer. The morphism preserves the public constraints: Q, P_Q, and $\mathfrak{K}^{\mathrm{pub}}_Q$ are preserved; the readout is preserved or refined; residue is not hidden; status is not overpromoted; the carrier route is not illegally promoted; forbidden promotions are preserved unless validly discharged by a public theorem or contract; obligations are preserved or discharged; and public language is preserved.

7 Composition Safety and the Category Laws

Definition 7.1 (Composition safety certificate)**.** A *composition safety certificate* records that the public constraints of a refinement morphism are stable under composition. It comprises:

(1) a residue transition certificate;

(2) a status transition certificate;

(3) a carrier transition certificate;

(4) a forbidden-promotion certificate;

(5) an obligation transition certificate;

(6) a certificate-preservation clause requiring downstream morphisms to preserve prior transition certificates.

Residue transitions may carry, discharge, reclassify, bound, or mark residue as blocking, but every such transition must be recorded. Status transitions must obey public status rules. Carrier-route transitions cannot upgrade without authorization. Forbidden promotions remain active unless discharged by a public theorem or contract. Obligations are preserved, discharged, or replaced by stronger explicit obligations.

Remark 7.2 (Certificate-gated composition). No transition of residue, status, carrier route, forbidden promotion, or obligation "composes" merely because it occurs. It composes only *with preserved public transition certificates*, as required by clause (6) of Definition 7.1. This certificate-gating is what prevents composition from silently laundering hidden residue or an overpromotion across a chain of morphisms.

Lemma 7.3 (Identity refinements are public-admissible)**.** *For every public-admissible completion C, the identity map $\mathrm{id}_C : C \to C$ supplied by $\mathrm{id}_{\mathfrak{K}}$ is a public refinement morphism.*

Proof. The identity changes no field of C. It therefore preserves Q, P_Q, $\mathfrak{K}_Q^{\mathrm{pub}}$, the readout, the residue state, the status state, the carrier route, the forbidden promotions, the obligations, and the public language, so the public-constraint layer of Definition 6.1 is met trivially. The typing layer is met because $\mathrm{id}_{\mathfrak{K}}$ supplies the identity together with its typing and identity-law data as part of $\mathsf{PublicCategoryData}_{\mathfrak{K}}$. Hence id_C is a public refinement morphism. □

Lemma 7.4 (Certificate-relative composition safety)**.** *Let $f : C_1 \to C_2$ and $g : C_2 \to C_3$ be typed public refinement morphisms carrying valid composition safety certificates, and suppose the downstream morphism g preserves the transition certificates of the upstream morphism f. Then $g \circ f$ preserves residue visibility, status discipline, carrier discipline, forbidden-promotion discipline, and obligation discipline.*

Proof. This lemma is not an unconditional derivation of transition preservation; it records the consequence of the certificate-preservation clause in the supplied composition-safety data, under the stated hypothesis that g preserves the transition certificates of f. Granting that hypothesis: each public transition made by f is recorded by its composition safety certificate, and by clause (6) of Definition 7.1 the morphism g either preserves that certificate or supplies a certified public transition from the output of f. Consequently no residue, status change, carrier route, forbidden promotion, or obligation recorded by f is erased when g is applied, and every transition introduced by g is itself certificate-visible. The composite transition is therefore public and certificate-visible in each of the five disciplines. The conclusion is thus explicitly relative to the supplied and preserved certificates, not a claim that arbitrary composites are automatically safe. □

Lemma 7.5 (Public refinement morphisms compose)**.** *If $f : C_1 \to C_2$ and $g : C_2 \to C_3$ are typed public refinement morphisms whose composite is supplied by $\mathrm{comp}_{\mathfrak{K}}$ and whose composition safety certificates are compatible, then $g \circ f : C_1 \to C_3$ is a public refinement morphism.*

Proof. For the typing layer, $\mathsf{HomTyping}_{\mathfrak{K}}$ together with $\mathrm{comp}_{\mathfrak{K}}$ supplies the typed composite in $\mathrm{Hom}_{\mathfrak{K}}(C_1, C_3)$. For the public-constraint layer, Lemma 7.4 supplies preservation of residue, status, carrier route, forbidden promotion, and obligation constraints; target and readout preservation compose because both f and g preserve Q, P_Q, and readout, and closure of these preservations is certified by $\mathsf{ClosureCert}_{\mathfrak{K}}$. Hence $g \circ f$ satisfies both layers of Definition 6.1. □

Lemma 7.6 (Associativity of public refinement composition)**.** *If f, g, h are composable public refinement morphisms and associativity is certified by* $\mathsf{AssociativityCert}_{\mathfrak{K}}$*, then*

$$h \circ (g \circ f) = (h \circ g) \circ f.$$

Proof. Associativity of the underlying composite is one of the supplied and certified public category-data certificates. Each side is a public refinement morphism by two applications of Lemma 7.5, and $\mathsf{AssociativityCert}_{\mathfrak{K}}$ certifies that the two bracketings denote the same morphism of $\mathrm{Hom}_{\mathfrak{K}}(C_1, C_4)$. □

8 Certified Public Completion Category Formation

Definition 8.1 (Public completion category)**.** Given $\mathfrak{K}_Q^{\mathrm{pub}}$ and certified $\mathsf{PublicCategoryData}_{\mathfrak{K}}$, the *public completion category* $\mathsf{Comp}_{\mathfrak{K}}^{\mathrm{pub}}(Q, P_Q)$ has object collection $\mathrm{Obj}_{\mathfrak{K}}$, hom-collections $\mathrm{Hom}_{\mathfrak{K}}(C_1, C_2)$, identities $\mathrm{id}_{\mathfrak{K}}$, and composition $\mathrm{comp}_{\mathfrak{K}}$.

Theorem 8.2 (Certified Public Completion Category Formation)**.** *Let Q be a public target and P_Q its public closure target. Suppose:*

(1) *Paper 2 supplies a category-forming handoff for (Q, P_Q);*

(2) *the public completion context $\mathfrak{K}_Q^{\mathrm{pub}}$ is formed; and*

(3) *certified public category data $\mathsf{PublicCategoryData}_{\mathfrak{K}}$ are supplied.*

Then $\mathsf{Comp}_{\mathfrak{K}}^{\mathrm{pub}}(Q, P_Q)$ is a category.

Proof. Objects and hom-collections are supplied by $\mathsf{PublicCategoryData}_{\mathfrak{K}}$. Identities are supplied by $\mathrm{id}_{\mathfrak{K}}$, and Lemma 7.3 shows they are public-admissible refinement morphisms. Composition is supplied by $\mathrm{comp}_{\mathfrak{K}}$, and Lemma 7.5 shows the hom-collections are closed under it. The identity laws hold by $\mathsf{IdentityLawCert}_{\mathfrak{K}}$, and associativity holds by $\mathsf{AssociativityCert}_{\mathfrak{K}}$ as in Lemma 7.6. Local smallness, where required, is recorded by $\mathsf{Universe}_{\mathfrak{K}}$ (Definition 3.3). Hence the data satisfy the category axioms. □

Remark 8.3 (What the formation theorem does and does not do). Theorem 8.2 is conditional on hypothesis (3). It does not construct $\mathsf{PublicCategoryData}_{\mathfrak{K}}$ from a route label, and in particular it does not derive objects, morphisms, identities, composition, or the three category-law certificates from the presence of a category-forming need class. It verifies only that *supplied and certified* public category data assemble into a public completion category. If any component of $\mathsf{PublicCategoryData}_{\mathfrak{K}}$ is missing or its certificate fails, the category is not formed, and the classifier of Section 10 returns CCO_UNFORMED.

Remark 8.4 (Role of the formation theorem). Theorem 8.2 is not advertised as a construction of category data from a need class. Its role is to separate three public levels that are routinely conflated: a completion need, certified category data, and canonical initiality. The mathematical content of Paper 3 is the certificate discipline that keeps these levels distinct together with the canonicality criterion (Theorem 9.8) that links them, not a claim that a need class automatically constructs a category or that a category automatically contains an initial object. Read this way, the formation and criterion theorems are the two hinges of a public type discipline for completion claims: the first says when a category exists to speak of, the second says which object in it, if any, earns the canonical label.

Remark 8.5 (Certificate-soundness unpacking). The lemmas of Section 7 and Theorem 8.2 are certificate-soundness unpacking results, not unconditional derivations. They say that once the declared identity, composition, associativity, and closure certificates are supplied and check,

the completion data assemble into the corresponding certified category; the category axioms are discharged *by* those certificates rather than proved from below. This is deliberate: the certificates are the obligations, and the lemmas record what follows once they are met. They are not intended to replace the certificate obligations, and the paper nowhere claims the axioms hold for data lacking the certificates. The mathematical content is the discipline that keeps these obligations explicit and separates them from the canonicality criterion, not a re-derivation of the category axioms.

9 Initiality and the Canonicality Criterion

Definition 9.1 (Public initial completion)**.** An object $C_0 \in \mathsf{Comp}^{\mathrm{pub}}_{\mathfrak{K}}(Q, P_Q)$ is *public-initial* if for every public-admissible completion C there exists a unique public refinement morphism $f_{0C} : C_0 \to C$. Uniqueness is uniqueness in the hom-collection $\mathrm{Hom}_{\mathfrak{K}}(C_0, C)$.

Remark 9.2 (Why initial rather than terminal). The refinement arrows of Definition 6.1 are oriented from the minimally committed, universally compatible completion toward more specified admissible completions: a public refinement morphism $C \to C'$ refines C into the more committed C' while preserving all public constraints. Under this orientation an *initial* object is the canonical one — it is the unique minimal certified completion from which every other admissible completion receives a unique refinement morphism, i.e. the least-committed completion that every admissible completion refines away from. A terminal object, by contrast, would be a maximally committed completion into which everything maps; that is not the canonical completion but a fully specified endpoint, and it confers no canonicality (Example 13.11). Reversing the refinement-arrow convention yields the dual terminal formulation in the opposite category; the mathematics is unchanged, and the choice of initial is the choice to treat the minimally committed universally compatible completion as canonical.

Definition 9.3 (Initiality certificate)**.** An *initiality certificate* $\mathsf{InitCert}_Q$ is a public certification record asserting that a proposed object C_0 is initial in $\mathsf{Comp}^{\mathrm{pub}}_{\mathfrak{K}}(Q, P_Q)$. It records the formed category, the proposed initial object, the universal-arrow property, the public admissibility checks, the residue discipline, the status discipline, the carrier-route preservation, the forbidden-promotion preservation, and the scope, assumptions, and domain of the claim.

Definition 9.4 (Initiality-contract soundness)**.** A public initiality contract $\mathcal{K}_{\mathrm{init}}$ is *sound* only if it declares:

(K1) its scope, including Q, P_Q, $\mathfrak{K}^{\mathrm{pub}}_Q$, and $\mathsf{PublicCategoryData}_{\mathfrak{K}}$;

(K2) the proposed initial object C_0;

(K3) the universal-arrow obligation it supplies or delegates;

(K4) the public authority, prior public theorem, or previously accepted public contract under which that obligation is discharged;

(K5) the failure conditions under which the contract does *not* certify initiality;

(K6) compatibility with residue, status, carrier-route, and forbidden-promotion discipline;

(K7) an explicit guarantee that no hidden choice function or undeclared selector is used to crown C_0.

Write $\mathsf{InitContractSound}_Q(\mathcal{K}_{\mathrm{init}}) = 1$ when (K1)–(K7) hold. A contract failing any clause does not certify initiality.

Definition 9.5 (Initiality-certificate soundness)**.** $\mathsf{InitCertSoundness}_Q$ passes only if $\mathsf{InitCert}_Q$ is backed by either

(1) a proof of the universal-arrow property (existence and uniqueness of f_{0C} for every admissible C); or

(2) a public initiality contract $\mathcal{K}_{\text{init}}$ with $\mathsf{InitContractSound}_Q(\mathcal{K}_{\text{init}}) = 1$ (Definition 9.4).

It fails if the certificate is merely a label, if arrow uniqueness is unsupported, if the admissibility of the universal arrows is not checked, or — in the contract case — if any clause of $\mathsf{InitContractSound}_Q$ fails. A bare declared contract, unaccompanied by the clauses of Definition 9.4, is never sufficient.

Definition 9.6 (Canonical Completion Object)**.** If an initial object exists in $\mathsf{Comp}^{\mathrm{pub}}_{\mathfrak{K}}(Q, P_Q)$ and $\mathsf{InitCertSoundness}_Q$ passes for it, define

$$\boxed{\mathsf{CCO}^{\mathrm{pub}}_{\mathfrak{K}}(Q, P_Q) = \mathrm{Init}(\mathsf{Comp}^{\mathrm{pub}}_{\mathfrak{K}}(Q, P_Q)).}$$

If no initial object exists, or if initiality is not soundly certified, then $\mathsf{CCO}^{\mathrm{pub}}_{\mathfrak{K}}(Q, P_Q)$ is undefined.

Definition 9.7 (Public-admissible isomorphism)**.** A *public-admissible isomorphism* between objects C and D of $\mathsf{Comp}^{\mathrm{pub}}_{\mathfrak{K}}(Q, P_Q)$ is a pair of public refinement morphisms

$$u : C \to D, \qquad v : D \to C, \qquad \text{with} \qquad v \circ u = \mathrm{id}_C, \quad u \circ v = \mathrm{id}_D\,.$$

Both u and v are public refinement morphisms by definition, and their composites are public refinement morphisms by Lemma 7.5; thus a public-admissible isomorphism is an isomorphism internal to $\mathsf{Comp}^{\mathrm{pub}}_{\mathfrak{K}}(Q, P_Q)$.

Theorem 9.8 (Certified Initiality Criterion for the CCO)**.** *Let $\mathsf{Comp}^{\mathrm{pub}}_{\mathfrak{K}}(Q, P_Q)$ be a certified public completion category. If C_0 is public-initial and $\mathsf{InitCertSoundness}_Q$ passes for C_0, then*

$$\mathsf{CCO}^{\mathrm{pub}}_{\mathfrak{K}}(Q, P_Q) = C_0,$$

and C_0 is unique up to unique public-admissible isomorphism.

Proof. The identity $\mathsf{CCO}^{\mathrm{pub}}_{\mathfrak{K}}(Q, P_Q) = C_0$ is Definition 9.6 applied to the soundly certified initial object C_0.

For uniqueness, suppose C_0 and C_0' are both public-initial. By initiality of C_0 there is a unique arrow $u : C_0 \to C_0'$; by initiality of C_0' there is a unique arrow $v : C_0' \to C_0$. The composite $v \circ u : C_0 \to C_0$ is an endomorphism of the initial object C_0, and by uniqueness of arrows from C_0 to itself it equals id_{C_0}; symmetrically $u \circ v = \mathrm{id}_{C_0'}$. Both composites are public refinement morphisms by Lemma 7.5, so u and v constitute a public-admissible isomorphism $C_0 \cong C_0'$ in the sense of Definition 9.7. Its uniqueness follows from the same universal-arrow uniqueness that produced u and v. □

Remark 9.9 (Role of the standard uniqueness fact). The uniqueness of initial objects up to unique isomorphism is the standard categorical fact (Mac Lane [4], Awodey [5]), and the uniqueness argument above is its ordinary form. The contribution of Theorem 9.8 is not to reprove ordinary category theory. It is to *identify* the Canonical Completion Object with certified initiality inside the declared completion category, and to bind that initiality to the paper's admissibility, certificate, route, residue, and status discipline: the initial object here is one whose universal arrows are public refinement morphisms in the sense of Definition 6.1, whose initiality is soundly certified in the sense of Definition 9.5, and whose canonical status is emitted only through the priority classifier of Definition 10.11. The theorem's role is to anchor the stack's notion of canonical completion in a certified category, using the standard categorical fact where it applies and adding the certificate discipline that the bare fact does not supply.

Corollary 9.10 (Uniqueness up to unique public-admissible isomorphism)**.** *If a soundly certified Canonical Completion Object exists for* (Q, P_Q) *relative to* $\mathfrak{K}_Q^{\mathrm{pub}}$*, it is determined uniquely up to a unique public-admissible isomorphism.*

Proof. Immediate from the uniqueness clause of Theorem 9.8. □

Remark 9.11 (Criterion, not existence). Theorem 9.8 is a canonicality criterion. It presupposes that a public-initial object exists and that its initiality is soundly certified; it asserts nothing about whether such an object exists in a given category. Existence is a separate matter, adjudicated case by case and reported by the status classifier of Section 10. In particular a category may be nonempty, may branch, or may remain in open comparability without possessing any initial object.

10 Public CCO Status Classification

Definition 10.1 (Public refinement preorder)**.** For objects C_1, C_2 in $\mathsf{Comp}_{\mathfrak{K}}^{\mathrm{pub}}(Q, P_Q)$, write $C_1 \preceq_{\mathrm{pub}} C_2$ when there exists a public refinement morphism $C_1 \to C_2$. This relation is a preorder: reflexivity holds by the identity morphisms (Lemma 7.3) and transitivity holds by composition (Lemma 7.5).

Definition 10.2 (Certified non-refinement)**.** $\mathsf{NoPubRefinementCert}(C_1, C_2)$ is a public certificate that, under the declared public category data, no public refinement morphism from C_1 to C_2 exists. Absence of a *known* refinement morphism is not the same as certified non-refinement; only the certificate discharges the negative.

Remark 10.3 (The phrase "under certificate"). Throughout the status discipline, a predicate is said to hold *under certificate* when a checked public certificate establishes it, as opposed to its holding by absence of evidence or failure of search. In particular "$\mathsf{Live}_{\mathfrak{K}} = \varnothing$ under certificate" means a checked certificate establishes that the live set is empty (every object certified-disqualified), not merely that no live candidate has been found; and "certified route exhaustion" means a checked certificate establishes the absence of every route, not the failure to locate one. This is the same certified-negative discipline that distinguishes $\mathsf{NoPubRefinementCert}$ from absence of a known morphism: certified negatives are discharged, never inferred from silence. Where a certified negative is not available, the conservative residual $\mathsf{CCO\_OPEN\_COMPARABILITY}$ is emitted (Remark 10.10).

Lemma 10.4 (Schematic discharge of certified non-refinement)**.** $\mathsf{NoPubRefinementCert}(C_1, C_2)$ *is a certified negative existential and is not established by failure to find a morphism. It may be discharged by any of the following declared mechanisms, each of which certifies that no public refinement morphism* $C_1 \to C_2$ *exists.*

(N1) ***Finite declared hom-table.*** *If* $\mathsf{HomTyping}_{\mathfrak{K}}$ *presents the hom-classes as a finite declared table and* $\mathrm{Hom}_{\mathfrak{K}}(C_1, C_2) = \varnothing$ *in that table, then no morphism exists by inspection.*

(N2) ***Thin category / preorder negative.*** *If the category is thin, a public refinement morphism* $C_1 \to C_2$ *exists iff* $C_1 \preceq_{\mathrm{pub}} C_2$*; a certified negative* $C_1 \not\preceq_{\mathrm{pub}} C_2$ *in the generating preorder therefore discharges* $\mathsf{NoPubRefinementCert}(C_1, C_2)$*.*

(N3) ***Refinement-preserved invariant obstruction.*** *Suppose the public context supplies, as declared public invariant data, a public invariant category* $\mathcal{V}$ *together with a functor*

$$I : \mathsf{Comp}_{\mathfrak{K}}^{\mathrm{pub}}(Q, P_Q) \to \mathcal{V}$$

— *an optional component of the supplied public data of the same status as* $\mathsf{PublicCategoryData}_{\mathfrak{K}}$*, present exactly when an invariant obstruction is to be used — so that any public refinement morphism* $C_1 \to C_2$ *induces a morphism* $I(C_1) \to I(C_2)$ *in* $\mathcal{V}$ *by functoriality. If the declared invariant data certify*

$$\mathrm{Hom}_{\mathcal{V}}(I(C_1), I(C_2)) = \varnothing,$$

then no public refinement morphism $C_1 \to C_2$ *exists, since such a morphism would induce a* $\mathcal{V}$*-morphism* $I(C_1) \to I(C_2)$ *that is certified not to exist. This discharges* $\mathsf{NoPubRefinementCert}(C_1, C_2)$*. When no invariant data* $(\mathcal{V}, I)$ *are supplied, mechanism (N3) is simply unavailable, and* NoPubRefinementCert *must be discharged by one of the other mechanisms or an explicit non-existence proof.*

(N4) ***Constraint-preservation mismatch.*** *If every candidate morphism* $C_1 \to C_2$ *would violate a required slot-preservation, route-preservation, status-preservation, or certificate-preservation condition of Definition 6.1, then no public refinement morphism exists: the public-constraint layer cannot be satisfied.*

(N5) ***Checker-boundary exhaustion.*** *If the declared morphism vocabulary is finite and an admissible finite search over all morphism forms permitted by* $\mathsf{HomTyping}_{\mathfrak{K}}$ *exhausts them with none admissible, the exhaustion certifies non-existence.*

Absent one of (N1)–(N5) or an explicitly supplied certified non-existence proof, $\mathsf{NoPubRefinementCert}(C_1, C_2)$ *is not discharged, and the state routes to* CCO_OPEN_COMPARABILITY *rather than to* CCO_BRANCH.

Proof. Each mechanism certifies the negative directly. (N1) and (N5) are finite inspections of the declared hom-data, sound because $\mathsf{HomTyping}_{\mathfrak{K}}$ and the frozen bridge policy (Definition 5.4) fix exactly which hom-classes and morphism forms are admissible. (N2) is the defining equivalence of morphisms with the preorder in a thin category, so a certified preorder negative is a certified morphism negative. (N3) is contrapositive: by functoriality of I, a refinement morphism $C_1 \to C_2$ would induce a $\mathcal{V}$-morphism $I(C_1) \to I(C_2)$; since $\mathrm{Hom}_{\mathcal{V}}(I(C_1), I(C_2)) = \varnothing$ is certified, no such $\mathcal{V}$-morphism exists, so no refinement morphism can exist. (N4) observes that the public-constraint layer of Definition 6.1 is a necessary condition on any refinement morphism; if it is certified unsatisfiable for every candidate, no morphism qualifies. In each case the certificate discharges a genuine non-existence claim, not a mere search failure. □

Lemma 10.5 (Schematic discharge of certified route exhaustion)**.** *The route-exhaustion certificate underlying* CCO_FAILED *(Definition 10.11, clause (S5)) is a certified negative — that no live admissible candidate and no admissible branch or open-comparability route remains — and, like* NoPubRefinementCert*, is not established by mere failure to find a candidate. It may be discharged by the following declared mechanism.*

(X1) ***Certified per-object disqualification.*** *If* $\mathrm{Obj}_{\mathfrak{K}}$ *is finite and every* $C \in \mathrm{Obj}_{\mathfrak{K}}$ *carries a certified disqualification — a failed-admissibility, failed-closure, failed-bridge, or failed-initiality certificate eliminating* C *from* $\mathsf{Live}_{\mathfrak{K}}$ *(Definition 10.8) — then* $\mathsf{Live}_{\mathfrak{K}} = \varnothing$ *under certificate.*

(X2) ***Certified route closure.*** *If the declared route set is finite and every declared route carries a certified no-route verdict — no branch route (certified failure of non-dominance discharge) and no open-comparability route (certified that no further comparison, bridge, or dominance datum is available) — then no admissible candidate, branch, or open-comparability route remains.*

Route exhaustion is discharged only when both (X1) and (X2) hold: the live set is certified-empty and every route is certified-closed. Absent such certificates, exhaustion is not discharged, and the state routes to CCO_OPEN_COMPARABILITY *(the conservative residual of Remark 10.10), not to* CCO_FAILED.

Proof. Each mechanism certifies its negative by finite check over declared data. (X1) is a finite conjunction of per-object disqualification certificates; when all hold, no object is live, so $\mathsf{Live}_{\mathfrak{K}} = \varnothing$ under certificate. (X2) is a finite conjunction of per-route no-route certificates; when all hold, no branch or open-comparability route remains. Conjoined with the absence of a certified initial object, these are exactly the (S5) firing conditions of Definition 10.11. As with NoPubRefinementCert, the certificate discharges a genuine non-existence claim; failure merely to find a candidate or route does not discharge it, and the conservative default is CCO_OPEN_COMPARABILITY, never CCO_FAILED. □

Definition 10.6 (Certified public non-dominance)**.** Objects C_1, C_2 are *certified publicly non-dominated* when

$$\mathsf{NoPubRefinementCert}(C_1, C_2) \quad \text{and} \quad \mathsf{NoPubRefinementCert}(C_2, C_1).$$

Definition 10.7 (CCO status values)**.** Paper 3 uses the seven public output statuses, displayed here in the classifier priority order (S1)–(S7) of Definition 10.11:

$$\mathsf{CCO\_UNFORMED},\quad \mathsf{CCO\_EMPTY},\quad \mathsf{CCO\_CERTIFIED},\quad \mathsf{CCO\_BRANCH},$$
$$\mathsf{CCO\_FAILED},\quad \mathsf{CCO\_CANDIDATE},\quad \mathsf{CCO\_OPEN\_COMPARABILITY}.$$

A *failed proposal* is not itself global failure: it records that one specific proposed canonical object violates public admissibility. Global CCO_FAILED is emitted only when no certified initial object exists *and* all active canonicality attempts fail under certificate with no remaining live admissible candidate, branch, or open-comparability route.

> **Failed-proposal warning.** A failed proposal is *local*. Global CCO_FAILED is not emitted merely because some proposed canonical object fails public admissibility; it is emitted only when no soundly certified initial object exists *and* no live admissible candidate, branch, or open-comparability route remains. In particular a locally failed proposal never outranks a coexisting soundly certified initial object (Example 13.7).

Definition 10.8 (Live admissible candidate)**.** A completion object $C \in \mathrm{Obj}_{\mathfrak{K}}$ is a *live admissible candidate* when all of the following hold:

(L1) C is public-admissible as a completion object (Definition 4.1);

(L2) C satisfies the declared completion-object well-formedness and local closure requirements relevant to candidate status;

(L3) no declared failure, route-exhaustion, failed-admissibility, failed-closure, failed-bridge, failed-comparison, or disqualification certificate has eliminated C;

(L4) the certificates required for treating C as live are present and check.

Membership in $\mathrm{Obj}_{\mathfrak{K}}$ is not the same as being a live admissible candidate: an object may remain in $\mathrm{Obj}_{\mathfrak{K}}$ while a disqualification or failed-closure certificate has removed it from the live set. Write $\mathsf{Live}_{\mathfrak{K}} \subseteq \mathrm{Obj}_{\mathfrak{K}}$ for the set of live admissible candidates.

Definition 10.9 (Positive candidate-eligibility certificate)**.** A *positive candidate-eligibility certificate* $\mathsf{CandidateEligibilityCert}(C)$ certifies, for a completion object C, all of the following:

(E1) C is public-admissible (Definition 4.1);

(E2) C is live, i.e. $C \in \mathsf{Live}_{\mathfrak{K}}$ and no disqualification certificate has eliminated it;

(E3) C carries the required local closure, bridge, and route certificates needed for candidate status;

(E4) the comparison and classifier data are sufficient for candidate eligibility — no missing comparison, bridge, dominance, or route datum is material to whether C may be treated as a candidate.

A positive admissibility record alone is *not* a positive candidate-eligibility certificate: admissibility is clause (E1) only, whereas candidate eligibility additionally requires liveness (E2), the local closure/bridge/route certificates (E3), and comparison-sufficiency (E4). In particular, objects that are individually admissible may fail candidate eligibility because comparison or bridge data material to their standing are missing.

Remark 10.10 (CCO_OPEN_COMPARABILITY is the conservative residual). CCO_OPEN_COMPARABILITY is the conservative non-final residual for both of the following:

- missing or under-certified comparison, bridge, or dominance data (comparison-incomplete states); and
- suspected but uncertified exhaustion or disqualification (states that may be dead ends but carry no route-exhaustion certificate).

If failure is suspected but not certified, the classifier emits CCO_OPEN_COMPARABILITY, not CCO_FAILED: CCO_FAILED requires a certificate of route exhaustion (Definition 10.11, clause (S5)), and the paper does not claim failure without one. This is not a weakness but the certification discipline: an uncertified dead end is reported as open, not as certified failure. Paper 4 treats CCO_OPEN_COMPARABILITY as non-final under either reading.

Definition 10.11 (Ordered CCO status classifier)**.** The status classifier is a certificate-presented public *output policy*, not a claim that the raw status predicates are semantically disjoint. Each status fires only when its certificate predicate is presented and checks; the classifier evaluates the statuses in the declared priority order and emits the highest-priority status whose certificate checks. It stops at the first satisfied clause:

(S1) Emit CCO_UNFORMED if no category-forming handoff, public completion context, or certified public category data exist.

(S2) Emit CCO_EMPTY if certified category data exist but no completion objects have been formed, i.e. $\mathrm{Obj}_{\mathfrak{K}} = \varnothing$.

(S3) Emit CCO_CERTIFIED if a certified initial object exists and $\mathsf{InitCertSoundness}_Q$ passes.

(S4) Emit CCO_BRANCH if multiple live admissible candidates (Definition 10.8) exist, certified non-refinement data (Definition 10.2) discharge certified non-dominance (Definition 10.6) across the live branch set, and no certified initial object exists.

(S5) Emit CCO_FAILED if $\mathrm{Obj}_{\mathfrak{K}}$ is formed and may be nonempty, but certified route exhaustion holds: no certified initial object exists, every object or route has been certified-disqualified or route-exhausted, and no live admissible candidate remains ($\mathsf{Live}_{\mathfrak{K}} = \varnothing$ under certificate).

(S6) Emit CCO_CANDIDATE only if at least one live admissible candidate C carries a checked positive candidate-eligibility certificate $\mathsf{CandidateEligibilityCert}(C)$ (Definition 10.9) and the classifier certifies that no higher-priority status (S1)–(S5) applies. Positive candidate eligibility is a certificate, not a default.

(S7) Emit CCO_OPEN_COMPARABILITY otherwise: whenever the state is formed and nonempty but partial, ambiguous, under-certified, or comparison-incomplete — required comparison, bridge, closure, route, or classifier information is missing or under-certified, so no higher status and no positive candidate-eligibility certificate checks. This includes suspected but uncertified exhaustion (Remark 10.10).

Only CCO_CERTIFIED permits the public CCO claim. The residual status is CCO_OPEN_COMPARABILITY, a non-final, non-canonical status; CCO_CANDIDATE is never the default catch-all. Any multi-object state whose comparability is not sufficiently certified routes to CCO_OPEN_COMPARABILITY, so Paper 4 is never handed a candidate as though it were positively certified.

Lemma 10.12 (Classifier state-space coverage)**.** *Relative to a fixed public completion context and frozen public category data (Remark 4.2 and Definition 5.4), the certificate-presented priority classifier of Definition 10.11 assigns at least one status to every public state. The argument is not that a fixed list of semantic cases is exhaustive; it is that the residual status* CCO_OPEN_COMPARABILITY *is a genuine default that fires whenever no higher-priority certificate and no positive candidate certificate check.*

Proof. Fix a public state. Evaluate the certificate predicates in priority order.

If the category is not formed — no category-forming handoff, context, or certified category data — the CCO_UNFORMED certificate of (S1) checks. Otherwise the category is formed; if its object collection is empty, the CCO_EMPTY certificate of (S2) checks. Otherwise the category is formed and nonempty.

On a formed, nonempty state, evaluate the remaining certificates in order. If a certified initial object with $\mathsf{InitCertSoundness}_Q$ is presented, (S3) checks. If not, and if certified non-refinement data discharge certified non-dominance across a live branch set of at least two objects, (S4) checks. If not, and if certified route exhaustion is presented, (S5) checks. If not, and if a positive candidate-eligibility certificate is presented for at least one live admissible completion, (S6) checks.

If none of (S1)–(S6) checks, the state is formed and nonempty but no higher-priority certificate and no positive candidate certificate is present: the state is partial, ambiguous, under-certified, or comparison-incomplete. This is exactly the firing condition of the residual clause (S7), so CCO_OPEN_COMPARABILITY is emitted. Thus (S7) is not a semantic catch-all that we hope is exhaustive; it is the declared default whose firing condition is the *negation* of "some higher or candidate certificate checks," which is decidable given the presented certificates. Every formed, nonempty state therefore receives CCO_OPEN_COMPARABILITY if it receives nothing higher, and every state receives at least one status. □

Theorem 10.13 (Public CCO Status Classification)**.** *For every public completion context, the ordered CCO classifier of Definition 10.11 emits exactly one of the seven statuses. Only* CCO_CERTIFIED *permits the public claim* $\mathsf{CCO}^{\mathrm{pub}}_{\mathfrak{K}}(Q, P_Q)$.

Proof. At most one. The classifier evaluates the certificate predicates in the fixed priority order (S1)–(S7) and halts at the first that checks, so it emits at most one status.

At least one. By Lemma 10.12, every state receives a status: if some certificate among (S1)–(S6) checks, the highest-priority such status is emitted; otherwise the residual clause (S7) fires and CCO_OPEN_COMPARABILITY is emitted. The key point is that (S7) is not a hoped-for exhaustive list but a declared default whose firing condition is precisely "no (S1)–(S6) certificate checks," so on any formed, nonempty state that is not otherwise

classified, CCO_OPEN_COMPARABILITY is guaranteed. In particular CCO_CANDIDATE (S6) fires only on a positive candidate-eligibility certificate, never as a fall-through: an under-certified or comparison-incomplete multi-object state, lacking that positive certificate, routes to CCO_OPEN_COMPARABILITY (S7), not to CCO_CANDIDATE.

The CCO_EMPTY/CCO_FAILED/CCO_CANDIDATE *boundary.* These three statuses are cleanly separated by liveness (Definition 10.8) and by certification of exhaustion, and the ordering walks them without overlap.

- If $\mathrm{Obj}_{\mathfrak{K}} = \varnothing$, no completion object was formed, so (S2) fires and the status is CCO_EMPTY.
- If $\mathrm{Obj}_{\mathfrak{K}} \neq \varnothing$ but no live admissible candidate remains — every object or route is certified-disqualified or route-exhausted, so $\mathsf{Live}_{\mathfrak{K}} = \varnothing$ under certificate — and no certified initial object exists, then (S5) fires and the status is CCO_FAILED. This is distinct from CCO_EMPTY: the object collection is nonempty but the live set is certified-empty.
- If at least one live admissible candidate C carries a checked $\mathsf{CandidateEligibilityCert}(C)$ (Definition 10.9) and no higher status applies, then (S6) fires and the status is CCO_CANDIDATE. This is distinct from CCO_FAILED: a live eligible candidate exists, so the live set is nonempty and exhaustion does not hold.
- If neither the absence of live candidates nor route exhaustion is *certified* — the state may be a dead end, but no route-exhaustion certificate is present — then (S5) does not fire, and lacking a positive candidate-eligibility certificate (S6) does not fire either, so (S7) fires and the status is CCO_OPEN_COMPARABILITY, not CCO_FAILED (Remark 10.10).

Thus CCO_EMPTY requires $\mathrm{Obj}_{\mathfrak{K}} = \varnothing$; CCO_FAILED requires $\mathrm{Obj}_{\mathfrak{K}} \neq \varnothing$ with certified-empty live set; CCO_CANDIDATE requires a checked candidate-eligibility certificate on a live object; and any residual uncertified state is CCO_OPEN_COMPARABILITY.

Exactly one. Combining the two bounds, the classifier emits exactly one status. The permission of the CCO claim only under (S3) is Definition 9.6. Because the residual is the non-final, non-canonical CCO_OPEN_COMPARABILITY rather than the positively-loaded CCO_CANDIDATE, no state is silently handed to Paper 4 as a certified candidate on the strength of a fall-through alone. □

Remark 10.14 (Ordering resolves overlap). The exactly-one conclusion is a property of the ordered policy, not of the underlying predicates. Several raw predicates may hold at once — for instance a certified initial object may coexist with a locally failed proposal — and the ordering is what selects a single emitted status. Placing CCO_CERTIFIED (S3) above the CCO_BRANCH, CCO_FAILED, CCO_CANDIDATE, and CCO_OPEN_COMPARABILITY statuses encodes the design decision that a sound certified initial object outranks any coexisting candidate defect (see Example 13.7).

Remark 10.15 (Why Branch fires on liveness while Candidate requires eligibility). The bar for CCO_BRANCH (S4) is liveness plus certified non-dominance, whereas the bar for CCO_CANDIDATE (S6) is the stronger candidate-eligibility certificate (Definition 10.9, including comparison-sufficiency (E4)). The asymmetry is deliberate and reflects that CCO_BRANCH is a *stronger, certified* verdict than CCO_CANDIDATE. A branch is emitted only when certified non-refinement in both directions (Definition 10.6) has been discharged across the live set: the comparison between the branch objects is not missing but positively certified to be a non-dominance. A candidate defect that would block eligibility — a missing comparison, bridge, or dominance datum (E4) — is exactly an absence of certified comparison, which cannot coexist with a discharged non-dominance certificate on the same pair. Thus CCO_BRANCH does not fire on "weaker" data than CCO_CANDIDATE; it fires on a certified negative that CCO_CANDIDATE-eligibility does not require, and it correctly outranks CCO_CANDIDATE because a certified branch is a more determinate verdict than an uncertified candidacy. A missing

datum unrelated to the branch determination does not block CCO_BRANCH precisely because it is immaterial to the certified non-dominance; were it material, the non-dominance certificate could not have been discharged, and the state would route to CCO_OPEN_COMPARABILITY instead.

11 Output Card and Handoff to Paper 4

Definition 11.1 (CCO output card)**.** Paper 3 emits the public output card

$$\mathsf{CCOOutputCard}_Q^{\mathrm{pub}} = (Q,\ P_Q,\ \mathfrak{K}_Q^{\mathrm{pub}},\ \mathsf{CCOStatus}_Q,\ \mathsf{CompletionObject}_Q,\ \mathsf{InitCert}_Q,$$
$$R_Q^{\mathrm{after}},\ \sigma_Q^{\mathrm{after}},\ \mathsf{CarrierRoute}_Q,\ \mathsf{AllowedPublicClaim}_Q,\ \mathsf{ForbiddenPromotion}_Q,$$
$$\mathcal{O}_Q^{\mathrm{CCO}},\ \mathsf{Paper4Handoff}_Q),$$

recording the target and closure target, the context, the emitted CCO status, the completion object (when one is carried), the initiality certificate, the post-completion residue and status states, the carrier route, the allowed public claim, the forbidden promotions, the outstanding CCO-level obligations $\mathcal{O}_Q^{\mathrm{CCO}}$, and the Paper 4 handoff class.

Definition 11.2 (Paper 4 handoff classes)**.** The Paper 4 handoff class is one of

$$\mathsf{FullTier1CompilerEligible},\quad \mathsf{CandidateTier1CompilerEligible},$$
$$\mathsf{OpenComparabilityRoute},\quad \mathsf{BranchCompilerRequired},\quad \mathsf{EffectiveCompilerRoute},$$
$$\mathsf{ResidueOnlyRoute},\quad \mathsf{NoCarrierStopRoute},\quad \mathsf{NoHandoff}.$$

Full compiler eligibility requires certified CCO status, a suitable carrier route, no relevant blocking completion residue, and the required public contracts. Blocking residue may still permit a lower-status handoff, but not full CCO-based compiler eligibility.

Lemma 11.3 (The CCO output card is not a Tier-1 artifact)**.**

$$\mathsf{CCOOutputCard}_Q^{\mathrm{pub}} \not\Rightarrow \mathsf{T1Artifact}_Q^{\mathrm{pub}}.$$

Proof. The output card records completion status, residue, status, carrier route, allowed claims, forbidden promotions, obligations, and a handoff class. A Tier-1 artifact additionally requires the downstream artifact grammar and the down-compilation procedure, both of which are owned by Paper 4 [3] and are absent from the card. Therefore the card is a public handoff object, not a Tier-1 artifact. □

Theorem 11.4 (Public CCO Output and Paper 4 Handoff)**.** *For every public completion context, Paper 3 emits a well-formed CCO output card and a safe Paper 4 handoff class determined by the CCO status, the residue state, the status certificate, the carrier route, the allowed claims, and the forbidden promotions. Paper 4 may attempt full Tier-1 public compilation only when the handoff class certifies full compiler eligibility and all required public contracts are present.*

Proof. By Theorem 10.13 the classifier emits exactly one CCO status, and the output card of Definition 11.1 packages precisely that status together with the boundary data it references. The handoff class of Definition 11.2 is a function of these card fields, so it is well defined and unique for each card. Lemma 11.3 prevents the card from being promoted into a Tier-1 artifact within Paper 3. Full compiler eligibility is, by Definition 11.2, gated on certified CCO status and the required contracts, which is exactly the stated condition on Paper 4. □

12 Scope Guards and Forbidden Promotions

The following propositions restate, as formal nonclaims, the boundaries preserved throughout the paper. Each is a guard against a characteristic overreading of the categorical criterion.

Proposition 12.1 (Completion need does not imply CCO)**.**

$$\mathsf{CompletionNeedRecord}_Q^{\mathrm{pub}} \not\Rightarrow \mathsf{CCO}_{\mathfrak{K}}^{\mathrm{pub}}(Q, P_Q).$$

(Restatement of Proposition 2.1.)

Proposition 12.2 (Candidate is not canonical)**.** CCO_CANDIDATE $\not\Rightarrow$ CCO_CERTIFIED. *A public-admissible completion without a soundly certified universal-arrow property is a candidate, not a Canonical Completion Object.*

Proposition 12.3 (Open comparability is not branch certification)**.**

$$\mathsf{CCO\_OPEN\_COMPARABILITY} \not\Rightarrow \mathsf{CCO\_BRANCH}.$$

Unknown comparability is the absence of certified non-refinement data; it is not certified non-dominance (Definition 10.6).

Proposition 12.4 (Branch is not canonicality)**.** CCO_BRANCH $\not\Rightarrow \mathsf{CCO}_{\mathfrak{K}}^{\mathrm{pub}}(Q, P_Q)$. *A branch records certified non-dominance among live candidates, not the existence of an initial object.*

Proposition 12.5 (Empty category must be emitted)**.** *If the public completion category is formed but has no public-admissible objects, the classifier emits* CCO_EMPTY. *It must not silently proceed as though a candidate or CCO exists.*

Proposition 12.6 (CCO is context-relative)**.** $\mathsf{CCO}_{\mathfrak{K}}^{\mathrm{pub}}(Q, P_Q)$ *is meaningful only relative to* Q, P_Q, $\mathfrak{K}_Q^{\mathrm{pub}}$, *the supplied public category data, and the public admissibility predicate. A change in any of these may change the category, its objects, or its initial object.*

Proposition 12.7 (CCO is not a Tier-1 artifact)**.** $\mathsf{CCO}_{\mathfrak{K}}^{\mathrm{pub}}(Q, P_Q) \not\Rightarrow \mathsf{T1Artifact}_Q^{\mathrm{pub}}$. *Down-compilation into a public Tier-1 artifact is owned by Paper 4.*

Proposition 12.8 (CCO is not empirical validation)**.** $\mathsf{CCO}_{\mathfrak{K}}^{\mathrm{pub}}(Q, P_Q) \not\Rightarrow \mathsf{EmpiricalValidation}$. *Certified initiality in a public completion category is a categorical property, not an empirical claim about the world.*

13 Schematic Examples

The examples are deliberately schematic. None is an empirical physics claim; each illustrates a single classifier or typing behaviour.

Example 13.1 (Recovery completion candidate)**.** Paper 2 emits RecoveryNeeded. A recovery object supplies lost-role data and a role-preservation condition and is public-admissible with a positive candidate-eligibility certificate, but no universal-arrow property is certified. By clause (S6) the status is CCO_CANDIDATE, not CCO_CERTIFIED.

Example 13.2 (Enrichment completion candidate)**.** Paper 2 emits EnrichmentNeeded. An enrichment object supplies an enriched public packet with an inclusion map, residue update, and status update, and is admissible with a positive candidate-eligibility certificate. Absent certified initiality, the status is again CCO_CANDIDATE by clause (S6).

Example 13.3 (Selector completion candidate)**.** Paper 2 emits SelectorNeeded. A selector object supplies an admissible selector with its domain and codomain, together with residue and status updates, and carries a positive candidate-eligibility certificate. It is admissible but uncertified for initiality, hence CCO_CANDIDATE by clause (S6).

Example 13.4 (Typed structural completion candidate)**.** Here Paper 2 emits the need class TypedCompletionNeeded. A typed structural object supplies a completed public packet, a repair map, a public closure predicate, and an obligation set, together with a positive candidate-eligibility certificate. Admissible but uncertified for initiality: CCO_CANDIDATE by clause (S6).

Example 13.5 (Category-forming need with failed category data)**.** RecoveryNeeded is emitted, but the proposed morphisms are not closed under composition, so $\mathsf{ClosureCert}_{\mathfrak{K}}$ cannot be issued and $\mathsf{PublicCategoryData}_{\mathfrak{K}}$ is incomplete. The route name does not form a category. By clause (S1) the classifier emits CCO_UNFORMED (or records a category-data failure), not a CCO. This is the concrete content of Remark 5.2.

Example 13.6 (Empty public completion category)**.** A category-forming handoff and certified public category data are supplied, but no proposed completion satisfies public admissibility, so $\mathrm{Obj}_{\mathfrak{K}} = \varnothing$. By clause (S2) the classifier emits CCO_EMPTY, not CCO_CANDIDATE and not CCO_CERTIFIED. The result is a visible obstruction, not a hidden completion (cf. Proposition 12.5).

Example 13.7 (Failed proposal plus certified initial object)**.** One proposed canonical object hides residue and is recorded as a FailedProposal; a distinct object C_0 is public-initial and $\mathsf{InitCertSoundness}_Q$ passes for it. The local failed proposal does not force global failure. By the ordering of Definition 10.11 clause (S3) fires first, so the global status is CCO_CERTIFIED and $\mathsf{CCO}^{\mathrm{pub}}_{\mathfrak{K}}(Q, P_Q) = C_0$ (cf. Remark 10.14).

Example 13.8 (Unknown comparability)**.** Two admissible candidates exist, but between them neither a refinement morphism nor a certified non-refinement certificate is supplied. Comparability is uncertified, so the status is CCO_OPEN_COMPARABILITY by clause (S7), not CCO_BRANCH.

Example 13.9 (Certified branch via discharged non-refinement)**.** This example shows the certificate discipline doing discriminating work: it routes to CCO_BRANCH only because NoPubRefinementCert is genuinely discharged, not merely unknown. Let C_1 and C_2 be public-admissible completions carrying a refinement-preserved public invariant, represented by a functor $I : \mathsf{Comp}^{\mathrm{pub}}_{\mathfrak{K}}(Q, P_Q) \to \mathcal{V}$ to a declared invariant category $\mathcal{V}$ — say a carrier-route tag preserved by every refinement morphism (Definition 6.1, public-constraint layer) — with $\mathrm{Hom}_{\mathcal{V}}(I(C_1), I(C_2)) = \varnothing$ and $\mathrm{Hom}_{\mathcal{V}}(I(C_2), I(C_1)) = \varnothing$ certified. By mechanism (N3) of Lemma 10.4, any refinement morphism $C_1 \to C_2$ would induce a $\mathcal{V}$-morphism $I(C_1) \to I(C_2)$ that is certified not to exist, so

$$\mathsf{NoPubRefinementCert}(C_1, C_2) \quad \text{and} \quad \mathsf{NoPubRefinementCert}(C_2, C_1)$$

are both discharged, giving certified non-dominance (Definition 10.6). With no certified initial object, the classifier returns CCO_BRANCH by clause (S4). No CCO claim is permitted. Note that had the invariant obstruction *not* been certified — if the values were merely not-yet-compared — the state would route to CCO_OPEN_COMPARABILITY (S7) instead: the difference between CCO_BRANCH and CCO_OPEN_COMPARABILITY is exactly whether NoPubRefinementCert is discharged.

Example 13.10 (Under-certified comparison routes to OpenComp, not Candidate)**.** This example shows the residual discipline declining to over-certify. Two admissible completions D_1, D_2 are live and each carries a positive admissibility record, but the bridge policy (Definition 5.4)

has *not* supplied the typed bridge that would let their heterogeneous hom-classes be compared, and no NoPubRefinementCert certificate is discharged in either direction by any mechanism of Lemma 10.4. A naive reading might promote one of D_1, D_2 to a candidate CCO. The classifier does not. Although D_1 and D_2 are individually admissible, neither carries a positive candidate-eligibility certificate CandidateEligibilityCert (Definition 10.9): the missing comparison and bridge data are material to candidate eligibility, so clause (E4) fails even though the admissibility clause (E1) holds. A positive admissibility record is not a candidate-eligibility certificate. Consequently: no certified initial object (so not S3), no discharged non-dominance (so not S4, per Lemma 10.4), no certified route exhaustion (so not S5), and no positive candidate-eligibility certificate (so not S6). The state is comparison-incomplete, so clause (S7) fires and the status is CCO_OPEN_COMPARABILITY. The completion is not handed to Paper 4 as a certified candidate; the missing bridge is reported as an open comparability obstruction. This is the certificate discipline changing the verdict: without the priority classifier, the tempting move is to crown a candidate; with it, the under-certified state is correctly non-final.

Example 13.11 (Terminal but not initial)**.** An object T may receive a *unique* public refinement morphism $C \to T$ from every completion C, making it terminal in the public completion category. But unless T also has a unique public refinement morphism $T \to C$ to every object C, it is not initial and hence not a CCO. The CCO criterion uses the initial direction (Definition 9.1); terminality in the completion category confers no canonicality.

Example 13.12 (Thin preorder with a least object)**.** If the category is thin, so that there is at most one morphism between any two objects, initiality reduces to being a least public-admissible completion under $\preceq_{\mathrm{pub}}$. If such a least object exists and its initiality is soundly certified, it is the CCO by Theorem 9.8.

Example 13.13 (Finite non-vacuity witness)**.** This example is fully explicit and shows the antecedents of Theorem 9.8 can be met *by proof*, not by contract. Let

$$\mathrm{Obj}_{\mathfrak{K}} = \{C_0, C_1, C_2\}$$

and define a thin category by the preorder generated by

$$C_0 \preceq_{\mathrm{pub}} C_1, \qquad C_0 \preceq_{\mathrm{pub}} C_2,$$

together with reflexivity and no further comparabilities. The only non-identity hom-classes are therefore

$$\mathrm{Hom}_{\mathfrak{K}}(C_0, C_1) = \{\alpha_1\}, \qquad \mathrm{Hom}_{\mathfrak{K}}(C_0, C_2) = \{\alpha_2\},$$

and every other non-identity hom-class is empty. Composition is forced by the identities, and associativity holds because the category is thin (at most one morphism between any two objects).

Assume the three objects are public-admissible under the frozen predicate $\mathsf{Adm}_Q^{\mathrm{pub}}$ (Remark 4.2), and assume the displayed hom-table is exactly what $\mathsf{HomTyping}_{\mathfrak{K}}$ and the frozen bridge policy (Definition 5.4) permit. Then C_0 is public-initial by direct inspection: the unique arrow $C_0 \to C_0$ is id_{C_0}, the unique arrow $C_0 \to C_1$ is α_1, and the unique arrow $C_0 \to C_2$ is α_2. Existence and uniqueness of the universal arrow thus hold as a finite check, so $\mathsf{InitCertSoundness}_Q$ passes by clause (1) of Definition 9.5 — a proof of the universal-arrow property — with *no* initiality contract invoked. The classifier returns CCO_CERTIFIED by clause (S3), and

$$\mathsf{CCO}_{\mathfrak{K}}^{\mathrm{pub}}(Q, P_Q) = C_0.$$

The construction is schematic: the objects, morphisms, and admissibility are stipulated to exhibit the criterion in force, and no empirical or physical content is claimed.

Example 13.14 (Groupoid-like isomorphic branch)**.** Candidates are mutually isomorphic but carry nontrivial automorphisms, so the arrow from a proposed initial object to a given target is not unique. Isomorphism supplies existence of arrows but not their uniqueness, and initiality fails. Existence of arrows is insufficient; the universal arrow must be unique. Such a configuration is reported as CCO_BRANCH or CCO_OPEN_COMPARABILITY according to what non-refinement data are certified, never as CCO_CERTIFIED.

Example 13.15 (Schematic heterogeneous bridge)**.** Let C_{sel} be a selector completion and C_{typ} a typed structural completion. The hom $C_{\mathrm{sel}} \to C_{\mathrm{typ}}$ is empty (Definition 5.3) unless a typed bridge is supplied showing how the selector data induce a typed structural inclusion, together with residue- and status-merge certificates. With the bridge and merge certificates present, a single heterogeneous refinement morphism exists; without them, both objects may be individually admissible while the hom-class remains empty.

Example 13.16 (Non-category route with category-like internal structure)**.** A calibration protocol may have locally composable calibration steps. Nevertheless, unless it supplies Paper-3 public completion-category data for (Q, P_Q), it remains a non-category route by default (Section 2). Category-like structure elsewhere in the stack is not the same as forming $\mathsf{Comp}^{\mathrm{pub}}_{\mathfrak{K}}(Q, P_Q)$.

14 Conclusion

Paper 3 supplies the canonicality criterion for the Shadow Theory stack. It does not claim that every completion need yields a category, that every completion category contains an initial object, or that every useful completion is canonical. It states a stricter rule:

> A public completion is canonical exactly when it is soundly certified as initial in the public admissible completion category.

The Canonical Completion Object is therefore a public categorical object, not a Tier-1 artifact and not an empirical validation. It is the correct handoff object for Paper 4 only after its category, initiality, residue, status, carrier, and forbidden-promotion conditions have been made explicit and certified. This completes the Paper-3 bridge from completion necessity to compiler handoff: Paper 2 identifies when completion is needed; Paper 3 identifies when completion is canonical; Paper 4 determines whether the resulting output can be down-compiled into a public Tier-1 artifact.

> $\mathsf{CCO}^{\mathrm{pub}}_{\mathfrak{K}}(Q, P_Q)$ is the soundly certified initial object of $\mathsf{Comp}^{\mathrm{pub}}_{\mathfrak{K}}(Q, P_Q)$, when such an object exists.

A Expanded Definitions and Adversarial Cases

This appendix records the standard ways a naive canonicality argument fails, together with the mechanism in the main text that handles each. None of these mechanisms is new; each is a pointer to where the discipline is enforced.

Route-name category inference. Inferring a category from a category-forming need class is blocked by Definition 5.1 and Remark 5.2: category formation is gated on supplied, certified $\mathsf{PublicCategoryData}_{\mathfrak{K}}$, and Theorem 8.2 is conditional on it. Example 13.5 exhibits the failure explicitly.

Heterogeneous hom by default. Assuming a morphism between differently typed completions is blocked by the hom-typing rule of Definition 5.3: heterogeneous homs are empty unless a typed bridge and the relevant merge certificates are supplied (Example 13.15).

Silent composition of residue or status. Claiming that "residue, status, carrier, or obligation discharge composes" without preserved certificates is blocked by the certificate-preservation clause of Definition 7.1 and by Remark 7.2: composition preserves the disciplines only with preserved public transition certificates (Lemma 7.4).

Label-only initiality. Treating $\mathsf{InitCert}_Q$ as a label is blocked by Definitions 9.4 and 9.5: soundness requires either a proof of the universal-arrow property or a public initiality contract satisfying every clause of $\mathsf{InitContractSound}_Q$, and it fails on an unsupported uniqueness claim or a bare declared contract.

Isomorphism mistaken for initiality. Reading mutual isomorphism as canonicality is blocked by the uniqueness requirement in Definition 9.1 and illustrated in Example 13.14: existence of arrows does not supply the unique universal arrow when automorphisms are nontrivial.

Terminal mistaken for initial. Reading a terminal-like object as canonical is blocked by the direction of the universal property in Definition 9.1 (Example 13.11).

Local failure promoted to global failure. Promoting a single failed proposal to global CCO_FAILED is blocked by Definition 10.7 and the classifier ordering: global failure requires that no certified initial object exists and no live admissible candidate, branch, or open-comparability route remains under certificate (Example 13.7).

Unknown comparability promoted to branch. Reporting a branch without certified non-dominance is blocked by Definition 10.2 and clause (S7): absence of a known morphism is not NoPubRefinementCert, so uncertified comparability routes to CCO_OPEN_COMPARABILITY, not CCO_BRANCH (Example 13.8).

CCO promoted to artifact or empirical claim. Promoting the output card or the CCO into a Tier-1 artifact or an empirical validation is blocked by Lemma 11.3 and Propositions 12.7–12.8: down-compilation belongs to Paper 4, and empirical adjudication is outside the categorical criterion entirely.

Note on stack references. Papers 1 and 2 are treated as prior papers in the Shadow Theory stack; their formal statements are imported here as stack assumptions rather than re-proved, and Paper 4 is referenced only to fix the downstream compilation boundary. The results of the present paper are stated relative to those imports.

References

[1] J. Rodgers, *The Readout Non-Equivalence Theorem for Bounded Realized Domains*, Shadow Theory / Everything Equation programme, 2026. https://everythingequation.com.

[2] J. Rodgers, *Completion Necessity for Readout-Non-Equivalent Domains*, Shadow Theory / Everything Equation programme, 2026. https://everythingequation.com.

[3] J. Rodgers, *The Tier-1 Shadow Compiler*, Paper 4, Shadow Theory / Everything Equation programme, in preparation. https://everythingequation.com.

[4] S. Mac Lane, *Categories for the Working Mathematician*, 2nd ed., Graduate Texts in Mathematics **5**, Springer, New York, 1998.

[5] S. Awodey, *Category Theory*, 2nd ed., Oxford Logic Guides **52**, Oxford University Press, Oxford, 2010.

[6] J. Rodgers, *Everything Equation: research notes and supplementary materials*, https://everythingequation.com.