STRUCTURAL CLOSURE:
An Execution Architecture that Resolves Recurrent Failure in LLM Agents by Sealing Blanks and Unifying the Decision Path

-(NCAF · PCM · PVM)
From Repeated Failure to an Admissible Single-Route Decision Procedure-

Author
Chungsam Lee

Research Type
Independent Research (Non-Commercial)

Identifier
ORCID: 0009-0008-8890-3496

Date
February 11, 2026

Public Registration (Non-Commercial)
Publicly registered on OSF
Publicly released on GitHub

This paper is published in parallel with a Korean version containing the same research content.
The two versions differ only in language and hold equal academic status, and the same usage terms and license apply to both.

Contact for Prior Consultation

Author: Chungsam Lee (NorthStar)
Preferred contact: GitHub Issues for this repository or an author-designated official channel

(Optional, small text under the title)

Abstract

This paper introduces Structural Closure, an execution architecture designed to resolve recurrent failure in LLM-based agents by sealing uncertainty as explicit blanks and unifying decision-making into a single admissible path. While recent systems often pursue reliability through model scaling, retrieval augmentation, or post-hoc correction[1],[9] these approaches frequently leave the decision locus ambiguous and permit premature commitment.[5] Structural Closure reframes the problem as a decision-path contamination issue: when illegitimate transitions remain available, failures persist in new forms even as surface performance improves.

The proposed architecture operationalizes closure through three coordinated components. NCAF enforces a layered authority system that strictly separates measurement and labeling from judgment, preventing interpretive leakage across layers. PCM formalizes decision-making as an admissible transition governed by mandatory rails, isolation modes, and fail-stop[2] rules under missing evidence, inconsistency, or boundary conditions. PVM provides precedent reference only after admissibility is granted, ensuring that past-case information cannot bypass or dilute the closure discipline. Together, these mechanisms preserve free computation while prohibiting illegitimate conclusions.

We demonstrate the architecture via two empirical processors: a direction-judgment processor at the design/computation level and an amplitude-processing processor at the operational level. Across both, Structural Closure[3] prevents premature decisions, blocks post-hoc stabilization as a substitute for procedure, and converts failures into structured precedents

1) In this paper, “post-hoc stabilization” refers to any after-the-fact processing that reworks or re-adjusts an already committed decision (e.g., smoothing, patching, temporary rule overlays) to make outputs appear cleaner or more consistent. We treat this as concealment rather than resolution, and design the procedure so that post-decision modification is structurally disallowed.

2) Fail-Stop is a design principle that treats “not producing an incorrect conclusion” as normal operation when integrity violations are detected (e.g., missing mandatory inputs, inconsistency, or sequence/order breaches). In this paper, it functions as a safety rail that prohibits “output anyway” behavior when the decision path becomes unstable.

3) In this paper, “Structural Closure” is not a technique for improving correctness directly. It is an execution architecture that procedurally seals the decision path so that uncertainty or blanks cannot enter the decision step and contaminate the conclusion.

without proliferating rules. The paper emphasizes procedural reproducibility—a reproducible execution contract, traceable logs, and bounded precedent use—over optimizing a single metric outcome. Structural Closure thereby provides a practical foundation for decision systems whose errors remain observable, analyzable, and accumulative rather than concealed or drifting.

Keywords:
Structural Closure; Decision Admissibility; Decision-Path Sealing; Layered Authority; NCAF; PCM; PVM; Traceability; Reproducible Procedure; Rule Non-Proliferation

Table of Contents

References

Appendices

Chapter I. Introduction

1.1 Research Objectives and Problem Definition

Contemporary LLM-based agents exhibit a persistent class of failures that cannot be fully explained by model capacity, data coverage, or prompt quality. Even as surface-level performance improves, systems repeatedly commit to conclusions prematurely, revise outcomes after the fact, or produce internally inconsistent decisions across similar inputs. These phenomena indicate that the core problem does not lie solely in generation quality, but in the absence of a disciplined decision procedure. This paper addresses this gap by reframing agent failure as a problem of decision-path contamination rather than output correctness.[7],[14]

1.1.1 Research Objective: Stabilizing the Decision Path

The primary objective of this research is to stabilize the decision path of LLM-based agents. Stabilization, in this context, does not mean forcing convergence toward a particular answer, nor suppressing exploratory computation. Instead, it refers to ensuring that only admissible transitions can lead to a final decision. By constraining when and how a decision may occur, the system prevents illegitimate commitments while preserving free internal computation up to the point of judgment.[2]

1.1.2 Problem Definition: Contamination of the Decision Path

Decision-path contamination arises when a system allows multiple illegitimate routes to reach a conclusion. These routes include premature aggregation of partial evidence, implicit escalation of tentative signals into decisive judgments, and post-hoc correction mechanisms that retroactively justify an already-issued output. When such paths remain open, failures are not eliminated but merely transformed, often becoming harder to detect and analyze. The problem, therefore, is not the presence of uncertainty, but the lack of structural boundaries governing how uncertainty is handled at the decision point.

1.1.3 Research Question: Sealing Blanks and Confusion

This work asks whether it is possible to design an execution architecture that explicitly seals undecidable regions as blanks, rather than forcing resolution, and that unifies decision-making into a single admissible path. The central research question is whether such an architecture can prevent recurrent failure without relying on increased model complexity, ad-hoc rules, or retrospective correction.

1.2 Research Hypothesis: Structural Closure

The core hypothesis of this paper is that recurrent agent failure can be mitigated by enforcing Structural Closure: an execution discipline that seals uncertainty and restricts decision-making to a single, well-defined route. Structural Closure does not attempt to optimize correctness directly. Instead, it seeks to ensure that decisions occur only under conditions that are procedurally legitimate and traceable.

1.2.1 Blank Sealing

Blank sealing refers to the explicit representation of unresolved or indeterminate states as non-decisionable. Rather than smoothing over gaps in evidence or inference, the system preserves these gaps as structural blanks[4]. This prevents premature escalation and allows uncertainty to remain visible and analyzable within the execution trace.

1.2.2 Unification of the Decision Path

Unifying the decision path means eliminating parallel or implicit routes to judgment. Regardless of how many measurements[5], labels, or intermediate interpretations are generated, all decisions must pass through a single admissible route governed by explicit constraints. This unification is critical for preventing silent divergence between what the system computes and what it ultimately decides.

---

4) Here, “blank” includes (1) unobservable/undefined states, (2) missing evidence (NA), and (3) conflict states where a single conclusion is not admissible. Sealing blanks does not remove or arbitrarily fill blanks; it blocks the procedural escalation that would promote a blank into a decision.

5) “Unification” means that computation/labeling/auxiliary modules do not have authority to output or revise conclusions. Final decisions are produced only at a single decision layer; other layers provide measurement, labels, checks, and isolation signals only.

1.2.3 Output Invariance and TRACE

Structural Closure requires that once a decision is produced, it remains invariant under the same execution conditions[6]. This invariance is supported by TRACE: a structured logging mechanism that records admissibility checks, isolation events, and decision outcomes. TRACE does not evaluate correctness; it guarantees accountability and reproducibility of the decision process.

1.3 Research Scope and Empirical Design

This study focuses on execution-level architecture rather than model internals. The proposed framework is evaluated through two empirical processors designed to expose distinct dimensions of decision-path failure.

1.3.1 Empirical Study 1: Direction Judgment (Design/Computation)

The first empirical study examines a direction-judgment processor operating at the design and computation level[7]. This processor highlights how premature or conflicting directional decisions arise when measurement and judgment are not structurally separated. Structural Closure is applied to enforce decision admissibility and path unification.

1.3.2 Empirical Study 2: Amplitude Processing (Operation)

The second empirical study focuses on amplitude processing at the operational level, where numeric prediction is especially vulnerable to post-hoc stabilization and result-driven adjustment. This case demonstrates how Structural Closure prevents such practices by sealing undecidable regions and enforcing fail-stop behavior.

1.3.3 Scope Not Claimed in This Paper

This paper does not claim to improve benchmark accuracy, reduce error rates in isolation, or outperform existing agent frameworks on standardized tasks. Its

6) "Output invariance" means that once a decision is committed, no module is permitted to revise the conclusion (direction/judgment/final output). This is a core invariant that blocks structural intrusion by post-hoc correction; TRACE records make this invariance reproducible and auditable.

7) The empirical demonstrations in this paper are not intended to claim benchmark superiority or statistical generalization. They are procedural demonstrations that check whether (1) the decision rail remains stable during operation, (2) premature decisions and post-hoc revision are structurally blocked, and (3) failures are retained and accumulated as precedents rather than deleted.

scope is limited to establishing a reproducible execution discipline that governs when decisions may occur and how failures are retained as analyzable assets.

1.4 Contributions

This work makes three primary contributions. First, it formalizes Structural Closure as an execution-level architecture that separates computation from judgment and enforces decision admissibility. Second, it specifies a layered framework—NCAF, PCM, and PVM—that operationalizes closure without proliferating rules. Third, it demonstrates through empirical processors that failures can be preserved, traced, and accumulated without destabilizing the decision process, thereby enabling predictive systems grounded in accountability and procedural reproducibility.

Chapter II. Formalizing Misattributed Causes of LLM Failures and the Decision-Path Problem

2.1 Failures Did Not Decrease; Only Their Form Changed

Despite substantial advances in model scale and training techniques, failure in LLM-based agents has not diminished. Instead, failure has shifted from overtly incorrect outputs to subtler forms that are harder to detect and diagnose. These include internally inconsistent reasoning, confident but unjustified conclusions, and unstable decisions under minimally perturbed inputs. Such failures often evade conventional evaluation metrics, creating the illusion of progress while leaving the underlying decision instability unresolved.

2.1.1 Proliferation of Plausible Wrong Answers

Modern agents increasingly generate responses that appear coherent and contextually appropriate while remaining procedurally unjustified. Plausibility masks structural defects in the decision process, allowing incorrect conclusions to pass undetected. As a result, error is no longer defined by obvious contradiction, but by the silent acceptance of conclusions reached through illegitimate paths.

2.1.2 Concealment of Failures and Impossibility of Detection

When failure manifests as plausibility rather than contradiction, detection becomes dependent on external verification or retrospective analysis. In operational settings, such verification is often infeasible, leading to failure concealment. The system appears stable while accumulating latent decision-path defects that surface unpredictably.[3]

2.1.3 Ambiguity of the Decision Moment

A defining characteristic of these failures is the absence of a clearly defined decision moment. Outputs are produced without an explicit transition from computation to judgment, making it impossible to identify when uncertainty should have been sealed or when a decision should have been deferred. This ambiguity undermines accountability and prevents systematic analysis of failure causes.

2.2 Failure of the "Performance Problem" Frame

Many approaches implicitly frame agent failure as a performance deficit to be addressed through larger models, better data, or refined prompts. While such improvements may enhance surface-level accuracy, they do not address the procedural conditions under which decisions are made.

2.2.1 Coexistence of Performance Gains and Recurrent Failures

Empirical evidence shows that increased performance metrics often coexist with persistent decision instability. Agents may achieve higher benchmark scores while continuing to exhibit premature commitment, inconsistent judgments, and reliance on post-hoc rationalization. This coexistence indicates that performance gains alone do not resolve the decision-path problem.

2.2.2 Risks Induced by Plausibility

As plausibility improves, the cost of undetected failure increases. Highly plausible outputs discourage scrutiny and accelerate trust, amplifying the impact of erroneous decisions.[1] The performance frame thus paradoxically exacerbates risk by obscuring the procedural legitimacy of decisions.

2.3 A Misguided Target: "Hallucination"

The term "hallucination" has become a dominant label for LLM failure, but it misidentifies the locus of the problem. Treating hallucination as the primary target shifts attention away from the structural conditions that permit illegitimate decisions.

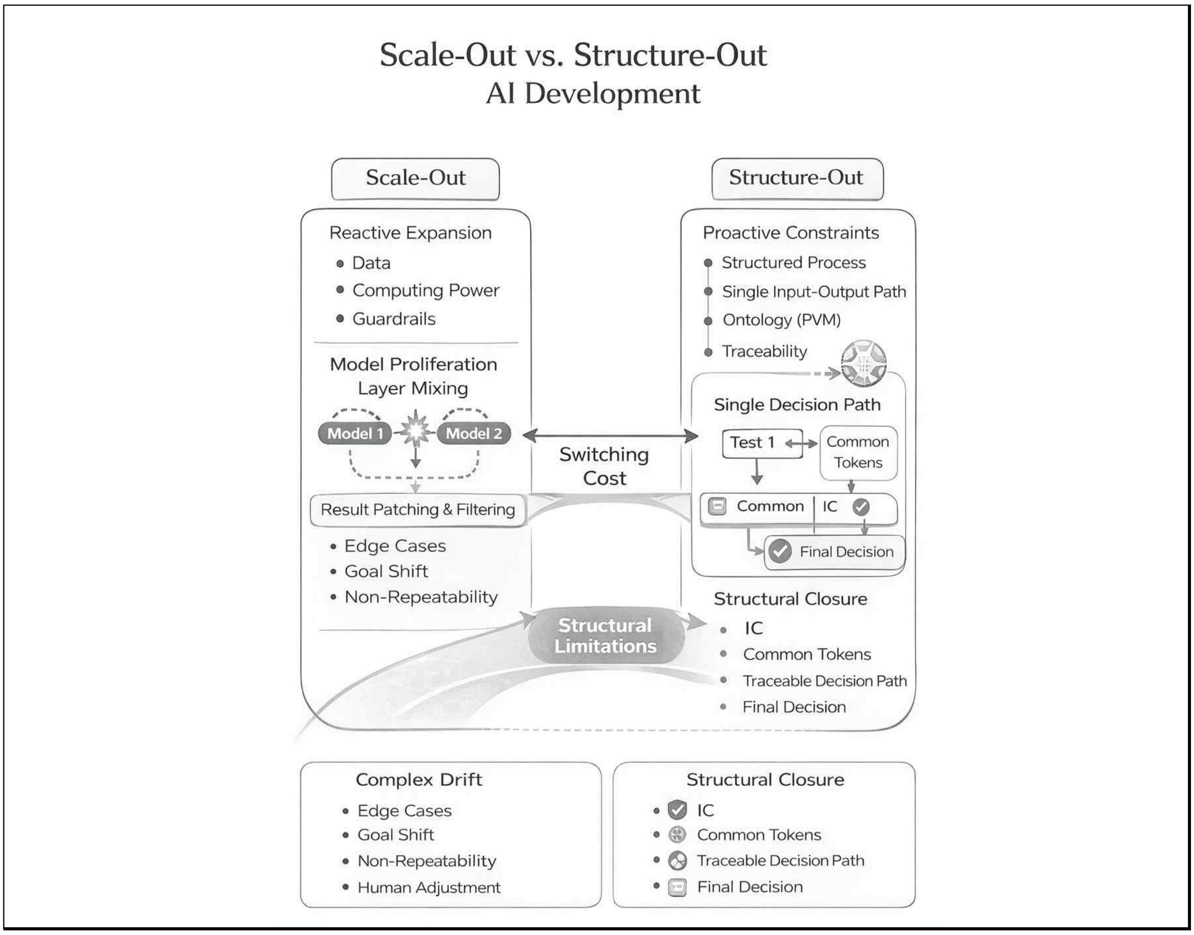
2.3.1 Hallucination Is Not a Cause but an Outcome

Hallucinated content emerges when the system is forced to resolve uncertainty without adequate admissibility constraints. It is therefore an outcome of an unsealed decision path rather than an independent failure mode. Eliminating hallucination without addressing decision admissibility merely suppresses symptoms.[6]

2.3.2 Risks When the Escalation Path Is Open

When escalation from tentative inference to final judgment remains unconstrained, any intermediate signal may be promoted to a decision. This open escalation path enables hallucination-like behavior regardless of model size or data quality, demonstrating that the root cause lies in procedural openness.

2.4 Rule Proliferation Is Not a Solution but a Concealment



[Figure 2-1. Contrast Between Scale-Out Approaches and Structure-Centered Approaches]

In response to persistent failure, many systems accumulate corrective rules, heuristics, and exception handlers. While intended to improve reliability, this strategy often conceals rather than resolves structural defects.

2.4.1 The Structure of Exceptions to Exceptions

As rules proliferate, new exceptions are introduced to handle edge cases created by earlier rules. This recursive pattern produces complex and opaque decision logic, making it increasingly difficult to determine why a particular conclusion was reached.

2.4.2 Boundary Collapse and Conflicts

Rule proliferation frequently leads to boundary collapse, where multiple rules apply simultaneously without a clear resolution mechanism. Conflicting directives are resolved implicitly or arbitrarily, further contaminating the decision path.

2.4.3 Deletion of Failures and Contamination of Records

Corrective rules often aim to eliminate visible failure by overwriting or masking it. In doing so, they erase valuable information about why the failure occurred. This deletion contaminates operational records and prevents the accumulation of analyzable precedents.

2.5 Limitations of KG and Scope Restriction

Knowledge graphs and scope restriction techniques attempt to constrain agent behavior by limiting accessible information. While effective in reducing certain errors, they do not address the procedural legitimacy of decisions.

2.5.1 Advantages of Scope Restriction

Restricting scope can reduce irrelevant retrieval and limit exposure to unsupported claims. These techniques are valuable for controlling informational noise and improving relevance.

2.5.2 Failure to Separate from Decision Contamination

However, scope restriction operates at the input level and does not regulate how decisions are formed. An agent may still commit prematurely or illegitimately within a restricted scope, leaving decision-path contamination intact.

2.5.3 Necessity of Sealing the Decision Path

Without explicit decision admissibility rules, scope restriction merely narrows the domain of failure. Structural Closure addresses this limitation by sealing the decision path itself rather than constraining information alone.

2.6 The Risky Practice of Post-hoc Stabilization

Post-hoc stabilization refers to adjusting or justifying outputs after a decision has already been produced. Although common, this practice undermines procedural integrity.

2.6.1 The Point Where Correction Turns Prediction into Performance

When outputs are modified to align with expected results, prediction becomes performance. The system appears reliable while its decision procedure remains uncontrolled. Such practices conflate explanation with correction, obscuring the original decision process.

2.6.2 Conditions Under Which Reproducibility Collapses

Post-hoc stabilization breaks reproducibility by allowing different corrective actions under similar conditions. As a result, identical inputs may yield divergent outcomes, eroding trust in the system's behavior.

2.7 Absence of the Decision Concept

A unifying cause of the failures discussed above is the absence of a formally defined decision concept within agent architectures.

2.7.1 Confusion Caused by an Undefined Decision

When decision is not explicitly defined, systems cannot distinguish between computation, interpretation, and judgment. This confusion allows decisions to occur implicitly, without accountability or traceability.[11]

2.7.2 Need to Unify the Decision Layer

Defining a single decision layer with exclusive authority over judgment is essential for procedural clarity. Without such unification, decision-path contamination remains unavoidable.

2.8 Summary: Redefining the Decision-Path Problem

2.8.1 Necessity of Decision-Admissibility Conditions

To address recurrent failure, systems must specify the conditions under which decisions are permitted. Decision admissibility transforms judgment from an implicit act into a regulated transition.

2.8.2 Necessity of Enforcing a Procedural Rail

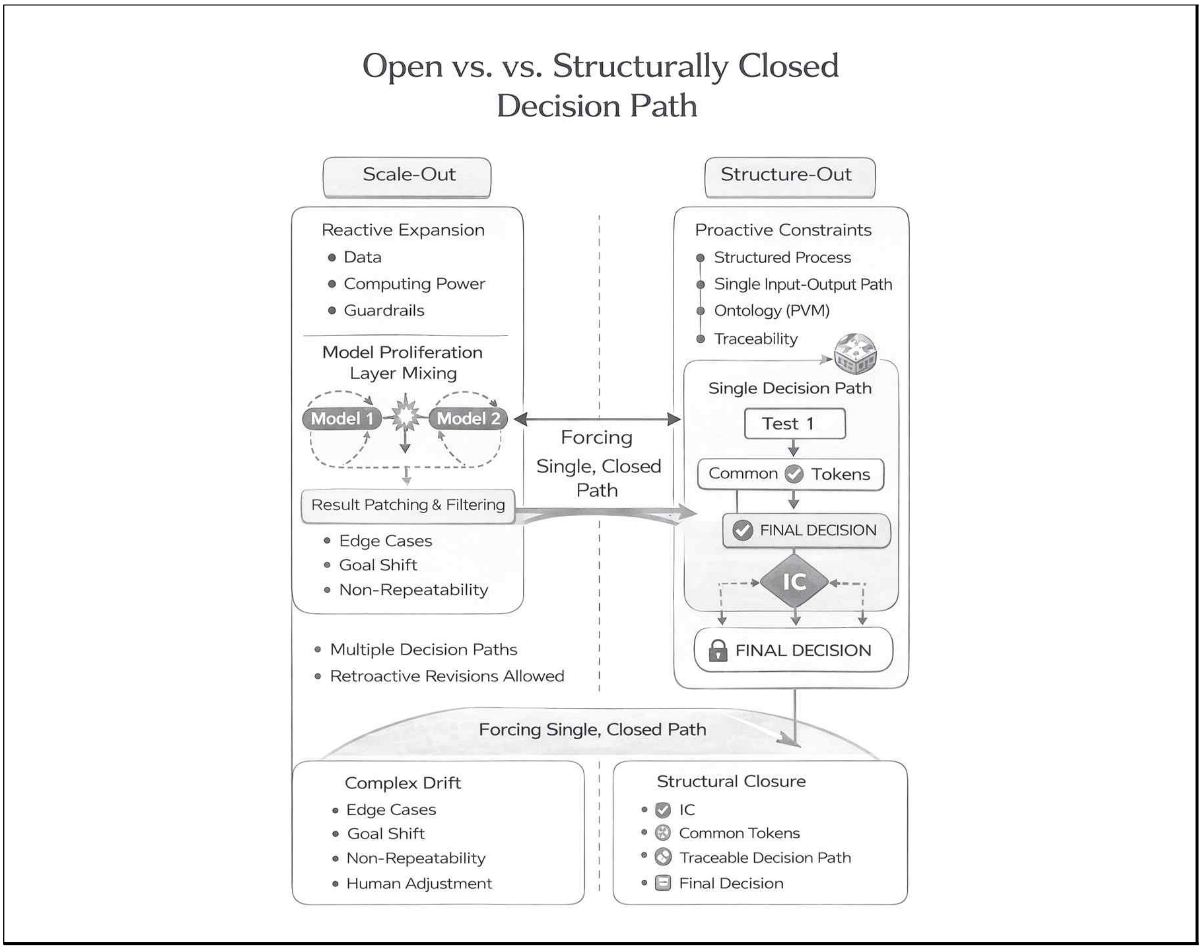
Admissibility conditions must be enforced through a procedural rail that governs the sequence and integrity of decision-making steps. This rail prevents illegitimate shortcuts and silent escalation.

2.8.3 Transition to Structural Closure

These requirements motivate the transition to Structural Closure. By sealing uncertainty, unifying the decision path, and enforcing admissibility, Structural Closure reframes failure as a manageable and analyzable phenomenon rather than an unavoidable byproduct of complexity.[14]

Chapter III. Overview of the Structural-Closure Architecture

3.1 Design Goals and Prohibitions



[Figure 3-1. Contrast Between Open-World Inference and Structural Closure]

Structural Closure is designed as an execution-level architecture whose primary objective is to stabilize decision-making without constraining computation itself. Rather than prescribing how inference should be performed, the architecture defines when inference is permitted to become a decision. To achieve this, Structural Closure establishes explicit design goals alongside equally explicit prohibitions that govern execution behavior.

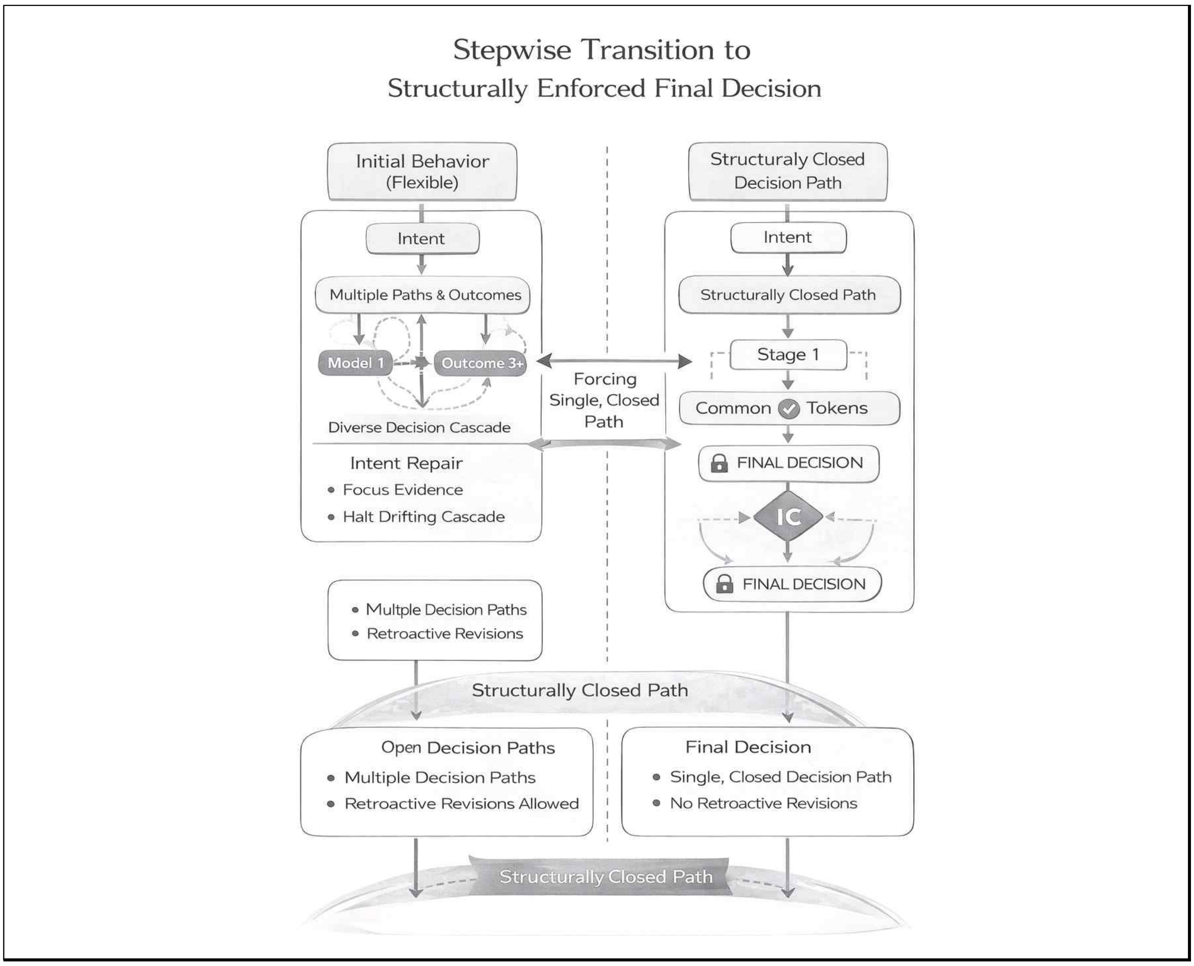
3.1.1 Prohibition of Premature Decision

The first and most fundamental prohibition is against premature decision-making. A decision must not occur while relevant uncertainty remains unresolved or while admissibility conditions are unmet. This rule prevents tentative signals, partial aggregates, or intermediate interpretations from being escalated into conclusions. Importantly, this prohibition does not suppress intermediate computation; it only restricts the transition from computation to judgment.

3.1.2 Prohibition of Post-hoc Modification

Structural Closure forbids modifying a decision after it has been produced. Post-hoc correction, justification, or stabilization undermines procedural legitimacy by rewriting history rather than regulating decision formation. Once a decision is emitted under admissible conditions, it must remain invariant for identical execution contexts.[6],[13]

3.1.3 Unification of the Decision Layer



[Figure 3-2. Conceptual Diagram of Stepwise Transition and Decision-Path Sealing]

All judgment authority is unified within a single decision layer. No other layer may implicitly or explicitly issue decisions, override outcomes, or inject judgmental semantics. This unification is essential for accountability, as it ensures that responsibility for decisions can be localized and traced.

3.2 Layer Separation Principles (NCAF Overview)

The Neuro-Cognitive Alignment Framework (NCAF) provides the structural foundation for layer separation in Structural Closure. NCAF does not define model internals; instead, it regulates the authority and permissible actions of execution layers.

3.2.1 Fixing the Role of the Measurement Layer

The measurement layer is responsible solely for producing quantitative or qualitative observations derived from inputs. It may compute features, scores, or signals, but it is explicitly prohibited from attaching judgmental meaning or directional intent to its outputs.

3.2.2 Fixing the Role of the Labeling Layer

The labeling layer interprets measurements into symbolic or categorical representations. While it may apply domain-specific schemas or patterns, it remains barred from making decisions or signaling finality. Labels produced at this layer are provisional and non-binding.

3.2.3 Uniqueness of the Decision Layer

The decision layer is the only locus where judgment is permitted. It consumes measurements and labels as inputs but operates under strict admissibility constraints. This uniqueness prevents interpretive leakage, where lower layers implicitly shape outcomes through unchecked escalation.

3.2.4 Prohibition of Layer Intrusion

Structural Closure enforces a strict prohibition against layer intrusion. Outputs from lower layers cannot bypass admissibility checks, and the decision layer cannot retroactively alter lower-layer computations. This separation preserves both procedural integrity and traceability.[4]

3.3 Decision Admissibility Rules (PCM Overview)

The Principled Constraint Model (PCM) formalizes decision admissibility within Structural Closure. PCM treats decision-making as a regulated transition rather than an automatic consequence of inference.

3.3.1 Minimum Conditions for Decision Admissibility

A decision is admissible only when predefined minimum conditions are satisfied. These conditions include completeness of required inputs, consistency among relevant signals, and absence of boundary violations. If any condition is unmet, decision-making is suspended.

3.3.2 Procedural Rail (Sequence Enforcement)

PCM enforces a procedural rail that specifies the exact sequence through which admissibility is evaluated. Steps may not be skipped, reordered, or implicitly merged. This enforcement ensures that decision-making follows a reproducible and auditable path.

3.3.3 Stop Rules (Missing, Inconsistency, Sequence Violation)

When admissibility checks fail due to missing data, internal inconsistency, or sequence violations, PCM triggers a fail-stop state. In this state, no decision may be produced, and the system must record the failure condition rather than attempt resolution through correction.

3.4 TRACE and Accountability

TRACE is the recording mechanism that supports accountability and reproducibility in Structural Closure. It captures the execution context surrounding each admissibility evaluation and decision outcome.[10]

3.4.1 Output-Invariant Recording

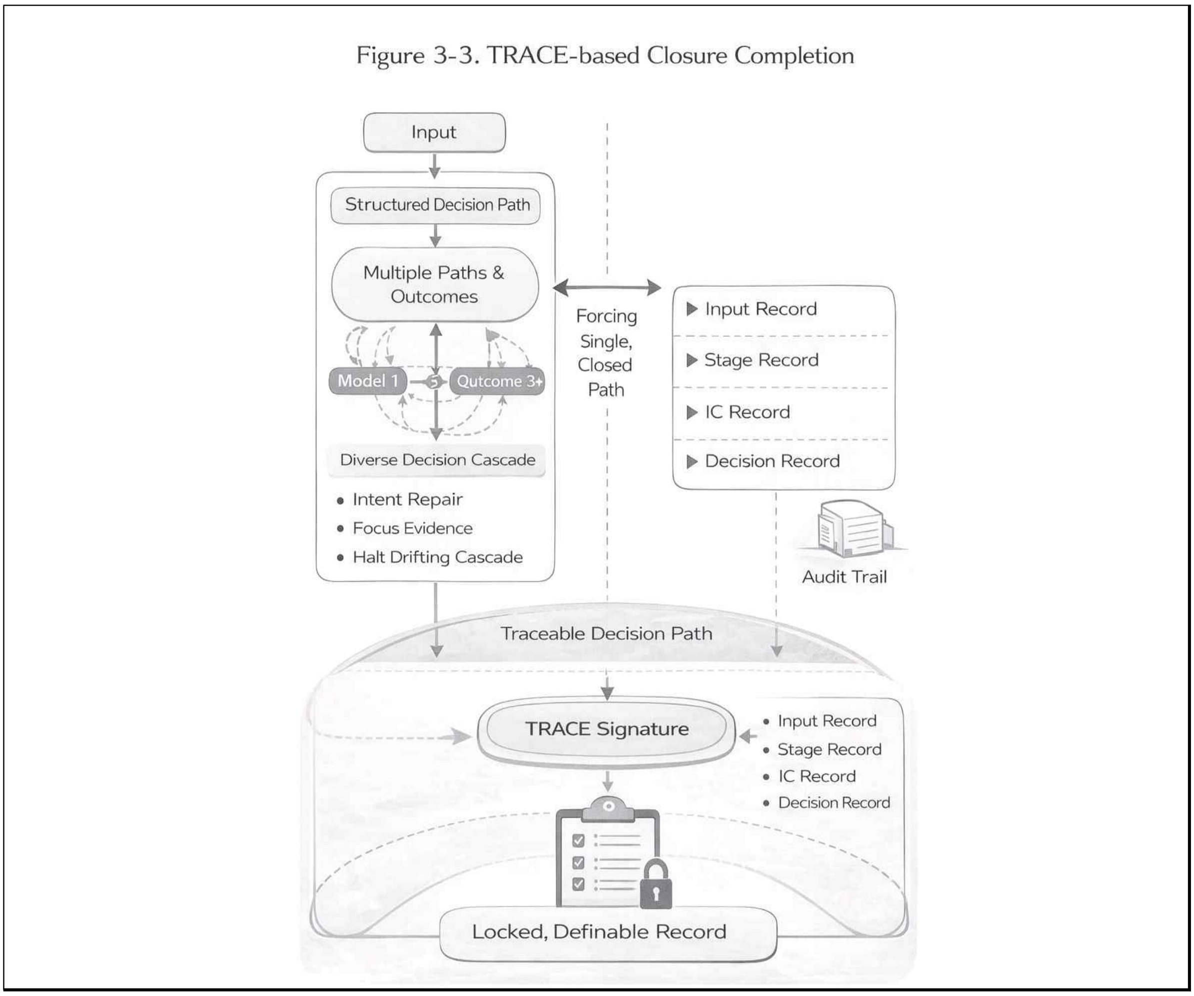
TRACE ensures that identical execution contexts yield identical recorded outcomes. This output invariance is critical for diagnosing failures and verifying that decisions arise from admissible conditions rather than incidental factors.

3.4.2 Minimum Conditions for Reproducibility

Reproducibility requires that admissibility checks, isolation events, and decision results be logged in a structured and complete manner. TRACE defines the minimal set of fields necessary to reconstruct the decision path without ambiguity.

3.4.3 Structuring Operational Logs

Operational logs under TRACE are structured to distinguish computation, admissibility evaluation, and judgment. This structure prevents conflation of these stages and allows failures to be analyzed as procedural events rather than opaque errors.



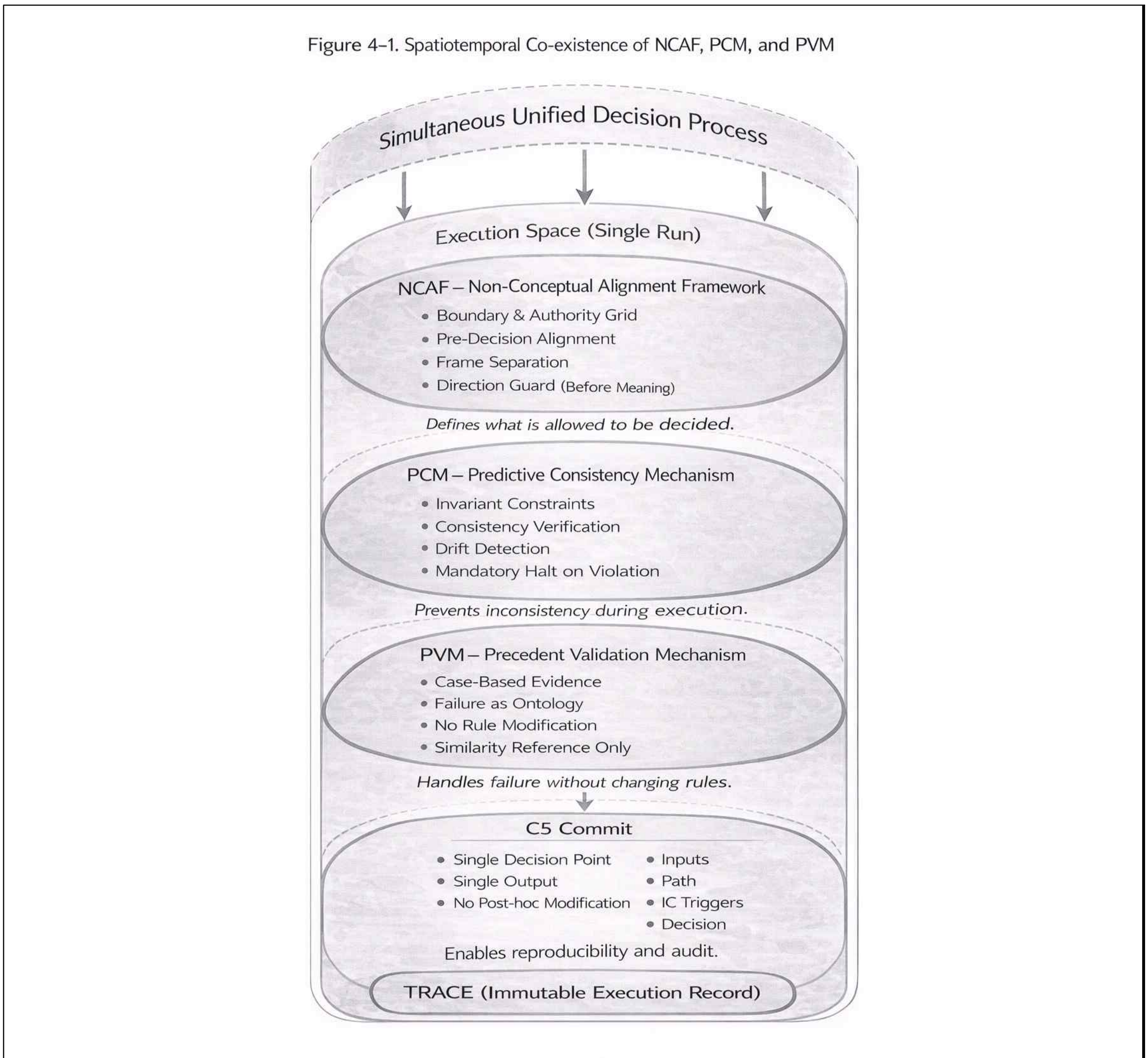
[Figure 3-3. End-to-End Flow of Decision-Path Unification and Blank Sealing]

3.5 Chapter Summary

This chapter outlined the foundational principles of Structural Closure. By defining explicit design goals, enforcing layer separation through NCAF, regulating decision admissibility via PCM, and ensuring accountability with TRACE, the architecture establishes a disciplined execution environment. In this environment, uncertainty is preserved rather than suppressed, decisions occur only under admissible conditions, and failures remain visible and analyzable rather than concealed.

Chapter IV. Core Technical Specification: Role Separation and Integrated Execution of NCAF·PCM·PVM

4.1 Chapter Overview and Purpose of the Specification



[Figure 4-1. Overview of the Core Structure of NCAF·PCM·PVM]

This chapter provides a technical specification of the core mechanisms that operationalize Structural Closure: NCAF, PCM, and PVM. The purpose of this specification is not to introduce new theoretical constructs, but to clarify how role separation and decision admissibility are enforced at execution time. Each component is defined in terms of its authority, constraints, and interaction boundaries, ensuring that closure is realized through procedure rather than interpretation.

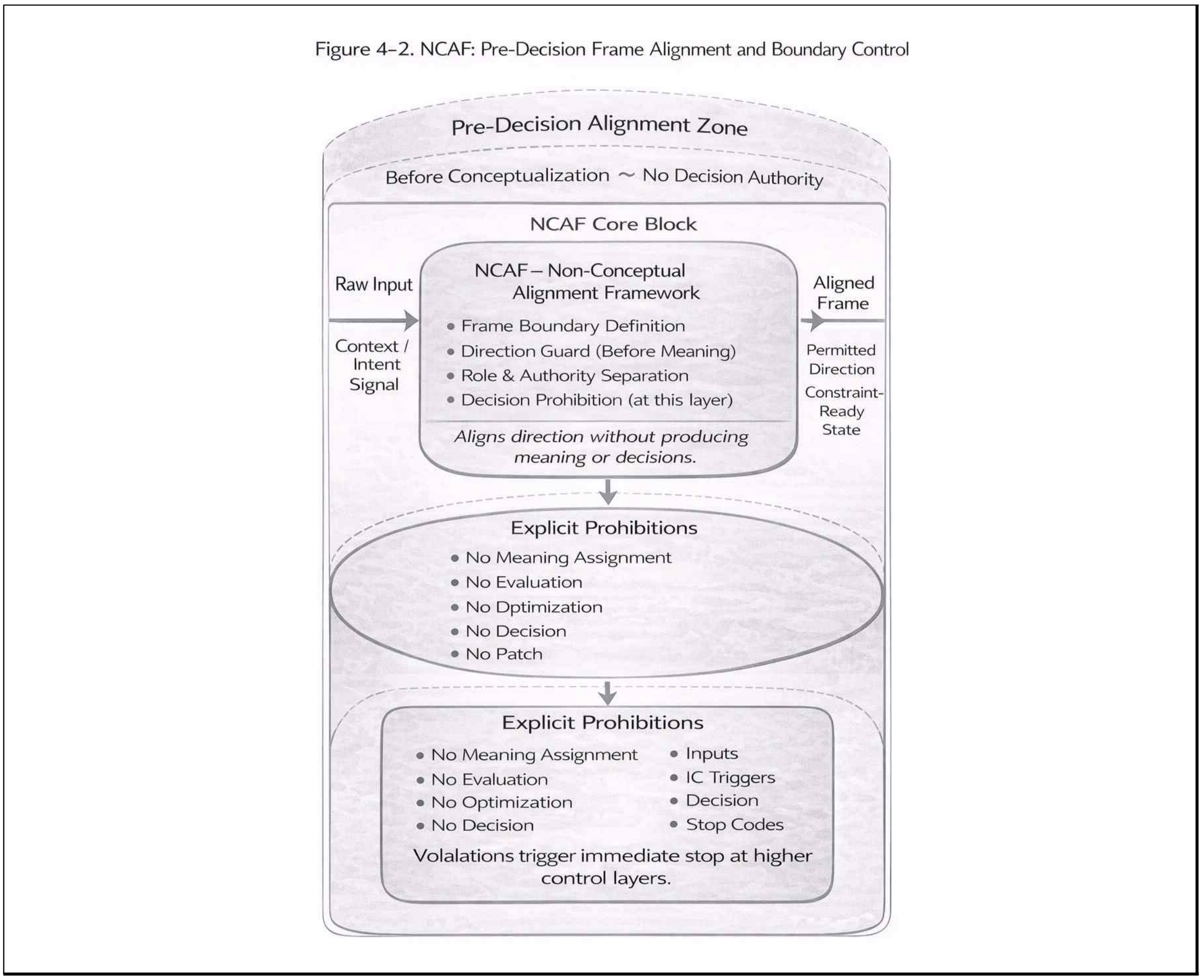
4.1.1 Delimiting the Scope of the Specification

The specification focuses exclusively on execution-level behavior. It does not prescribe model architectures, training methods, or optimization strategies. Instead, it defines how measurements, labels, decisions, and precedents are allowed to interact during runtime. By delimiting scope in this manner, the specification remains applicable across diverse model implementations.

4.1.2 Logical Status of the Specification

Within this paper, the specification serves as a normative contract. It establishes what must occur—and what must not occur—for a decision to be considered procedurally legitimate. The specification is therefore evaluative rather than descriptive: systems may deviate from it, but such deviations constitute explicit violations of Structural Closure.

4.2 NCAF: Cognitive Alignment Frame and Layer Authority System



[Figure 4-2. Role of NCAF as an Upper-Constraint and Alignment Frame]

NCAF enforces alignment not by shaping internal representations, but by regulating authority boundaries across execution layers. Its central function is to prevent the leakage of judgmental authority into stages of computation where it does not belong.

4.2.1 Definition of NCAF

NCAF is defined as a layered authority framework in which each layer is assigned a strictly bounded role. Measurement layers produce observations, labeling layers produce structured interpretations, and the decision layer alone is authorized to commit to outcomes. Alignment emerges from respecting these boundaries rather than from optimizing internal similarity metrics.

4.2.2 Layer-wise Authority and Prohibited Actions

Each layer operates under explicit permissions and prohibitions. Measurement layers are forbidden from signaling directionality or preference. Labeling layers may classify or summarize but must not imply finality. The decision layer may evaluate admissibility and issue outcomes, but it may not alter upstream data. These prohibitions are enforced procedurally, not heuristically.

4.2.3 Consequences of Layer Separation: Accountability and Reproducibility

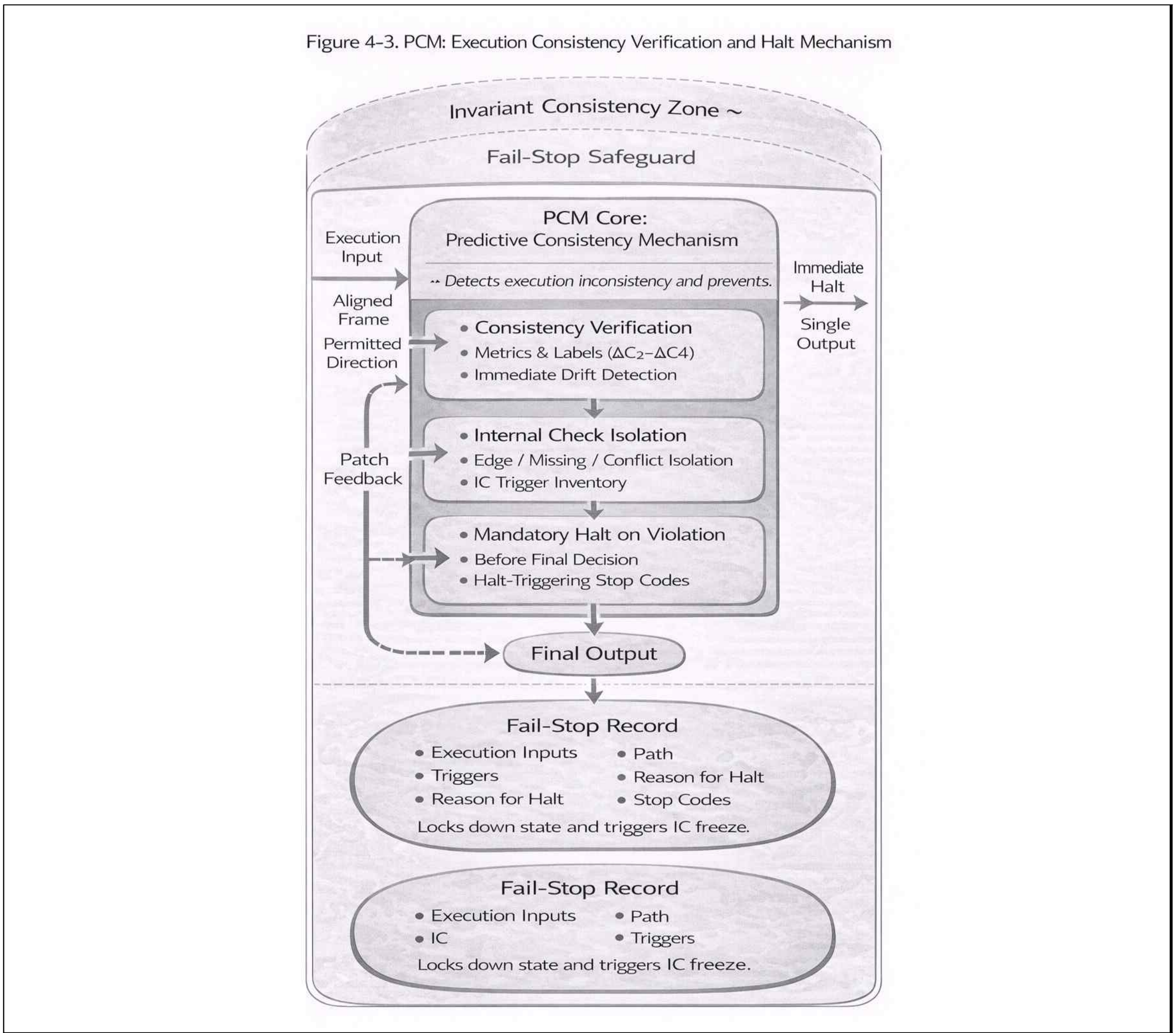
Strict layer separation enables accountability by localizing responsibility for decisions. It also enables reproducibility, as identical inputs traversing identical layers yield identical admissibility evaluations and outcomes. Without such separation, responsibility diffuses across the system, obscuring the source of failure.

4.3 PCM: Decision-Admissibility Constraints (Principled Constraint Model)

PCM formalizes the conditions under which a decision is permitted. It reframes decision-making as an admissible transition rather than a default outcome of inference.

4.3.1 Definition of PCM: A Decision Is an Admissible Transition

Under PCM, a decision occurs only when the system transitions from computation to judgment through an admissible state. This state is defined by completeness, consistency, and regime compliance. PCM does not rank answers or optimize confidence; it determines whether a decision may occur at all.



[Figure 4-3. Core Operation of PCM (Procedural Rail): Admit, Isolate, and Stop]

4.3.2 Minimum Admissibility Conditions

Minimum admissibility conditions specify required inputs, acceptable variance among signals, and boundary constraints. If any condition is unmet, the system must not decide. These conditions transform uncertainty from a liability into a first-class execution state.

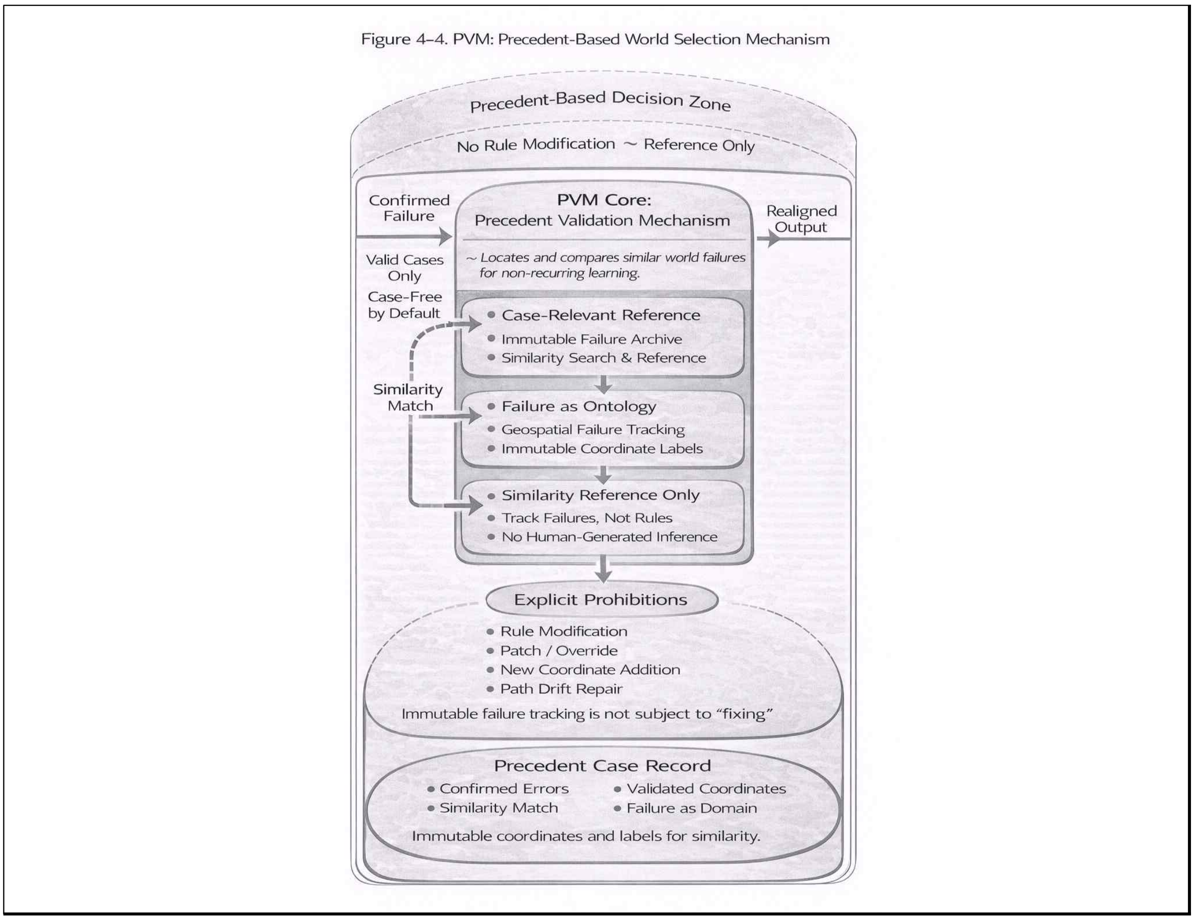
4.3.3 Procedural Rail Enforcement (Sequence Lock)

PCM enforces a fixed sequence for admissibility evaluation. Checks must occur in a predefined order, and intermediate results cannot be reused or bypassed. This sequence lock prevents silent escalation and ensures that admissibility is evaluated consistently across executions.

4.3.4 PCM Outputs: Admit, Isolate, Stop

PCM yields one of three outputs. Admit allows the decision layer to proceed. Isolate signals that computation may continue but judgment is suspended. Stop terminates execution due to irrecoverable violations such as missing mandatory inputs or sequence breaches. These outputs replace ambiguous fallback behaviors with explicit procedural states.

4.4 PVM: Precedent Reference (Past-Case Validation Mechanism)



[Figure 4-4. Operational Scope of PVM (Precedent Reference After PCM; No Conclusion Modification[8])]

8) PVM (Past-Case Validation Mechanism) is not a device for changing decisions. It references historical cases only after admissibility is granted, providing meta-signals such as risk/warning/range adjustment. This paper prohibits any operation in which precedent intrudes into the decision rail and revises the conclusion.[9],[15]

PVM introduces historical information into the system without compromising closure. It ensures that past cases inform decisions only under controlled conditions.

4.4.1 Definition of PVM: Precedent, Not Rules

PVM treats prior cases as precedents rather than prescriptive rules. Precedents provide contextual reference, not directives. This distinction prevents the uncontrolled growth of rules while preserving experiential knowledge.

4.4.2 Activation Point: After PCM

PVM is activated only after PCM has granted admissibility. This ordering ensures that precedent cannot override admissibility constraints or induce premature decisions. Historical similarity informs judgment but cannot authorize it.

4.4.3 Precedent Accumulation and Assetizing Failure

Failures recorded under TRACE become precedents that enrich PVM's reference base. Because failures are not deleted or corrected post-hoc, they accumulate as structured assets. Over time, this accumulation improves contextual awareness without altering procedural rules.

4.5 Integrated Execution Structure: Completion of Decision-Path Sealing

The integration of NCAF, PCM, and PVM produces a closed execution loop in which decision paths are explicitly sealed.

4.5.1 Integrated Pipeline

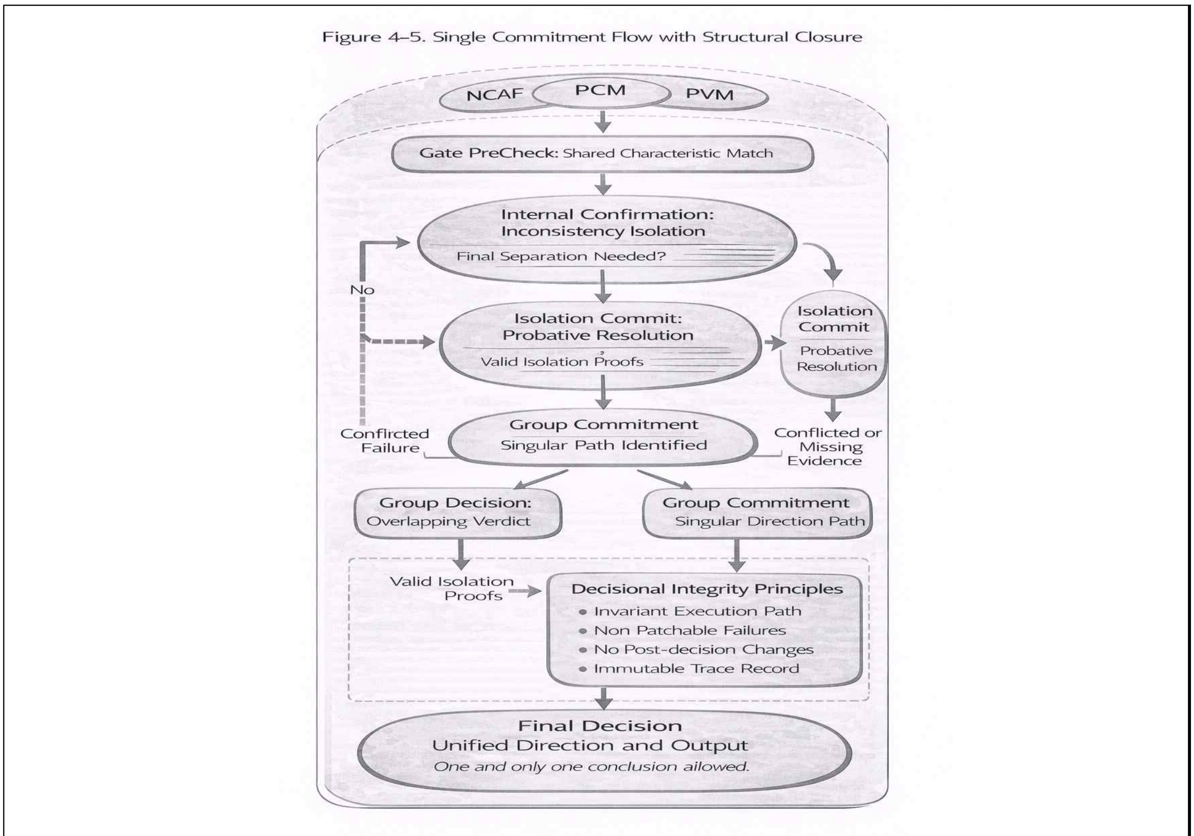
In the integrated pipeline, measurements flow to labeling, labeling to admissibility evaluation, and admissibility to decision. PVM may be consulted only after admissibility is granted. At no point may information bypass its designated stage.[12]

4.5.2 Contamination-Blocking Points

Contamination is blocked at multiple points: when lower layers attempt to signal judgment, when admissibility conditions are unmet, and when post-hoc modification is attempted. Each block is explicit and recorded, preventing silent failure.

4.5.3 Connection to Empirical Studies

The processors examined in Chapters V and VI instantiate this integrated structure in concrete settings. They demonstrate how closure principles operate under both design-level and operational constraints.

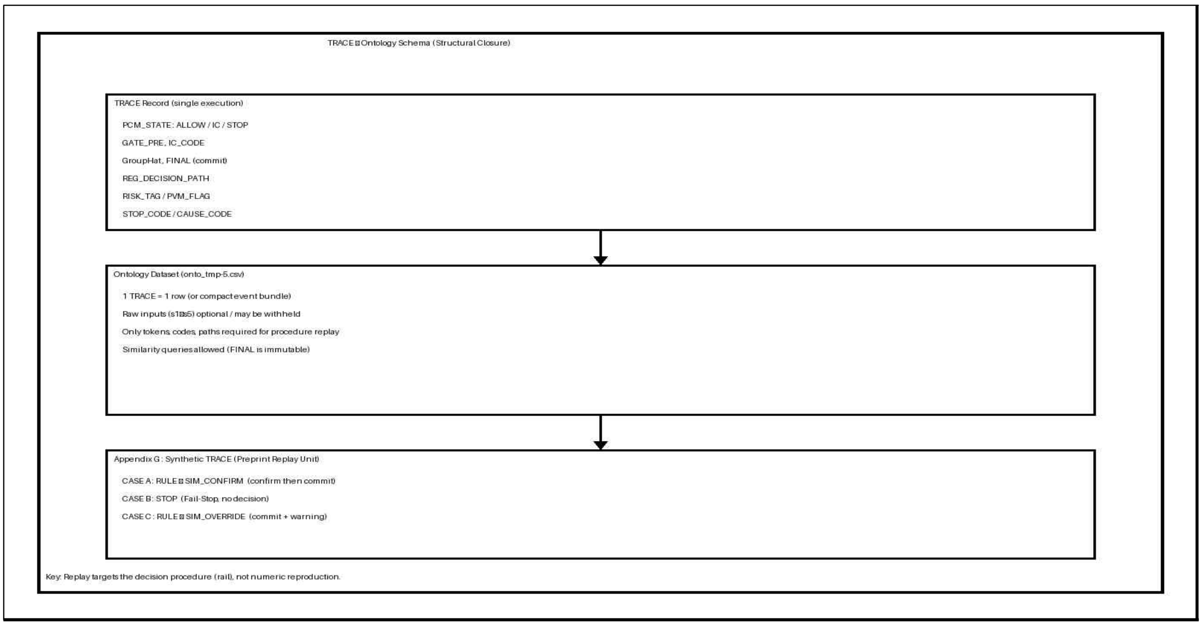


[Figure 4-5. Invariant Structure of a Single Decision Layer and Commit]

Authority Matrix (Corrected)

Layer	May Decide	May Modify	May Override
NCAF	Constraints	□	□
PCM	Decision Path	□	□
C5 Commit	FINAL	□	□
PVM	□	Risk/Warning	□
TRACE	□	Log Only	□

[Figure 4-6. Authority Matrix: Admissible and Forbidden Boundaries of NCAF·PCM·PVM·C5 (Commit)·TRACE]



[Figure 4-7. TRACE–Ontology Schema: Recording Structure at the Procedural Reproduction Unit]

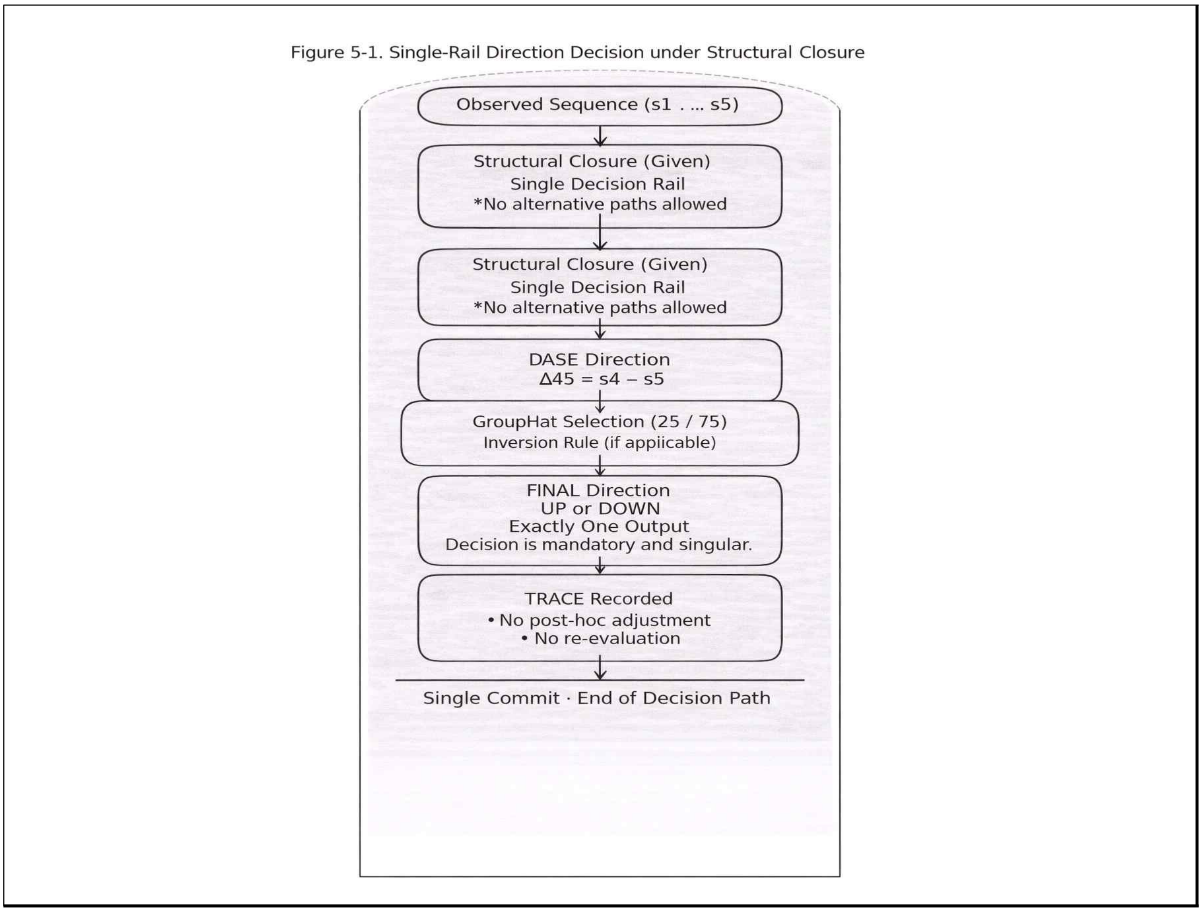
4.6 Chapter Summary

This chapter specified how Structural Closure is realized through the coordinated operation of NCAF, PCM, and PVM. By enforcing role separation, regulating admissibility, and constraining precedent use, the architecture seals the decision path procedurally. Decisions emerge not from accumulated confidence, but from admissible transitions within a reproducible execution framework.

Chapter V. Empirical Study 1: Structural Closure in the Direction-Judgment Processor (Design/Computation)

5.1 Problem Setup and I/O Specification

The first empirical study examines a direction-judgment processor designed to expose decision-path contamination at the design and computation level. Direction judgment represents a minimal yet revealing task: although the output space is binary, the internal process is highly susceptible to premature escalation, implicit bias, and post-hoc justification when admissibility is not enforced.



[Figure 5-1. Direction Decision (25/75) Processing Structure]

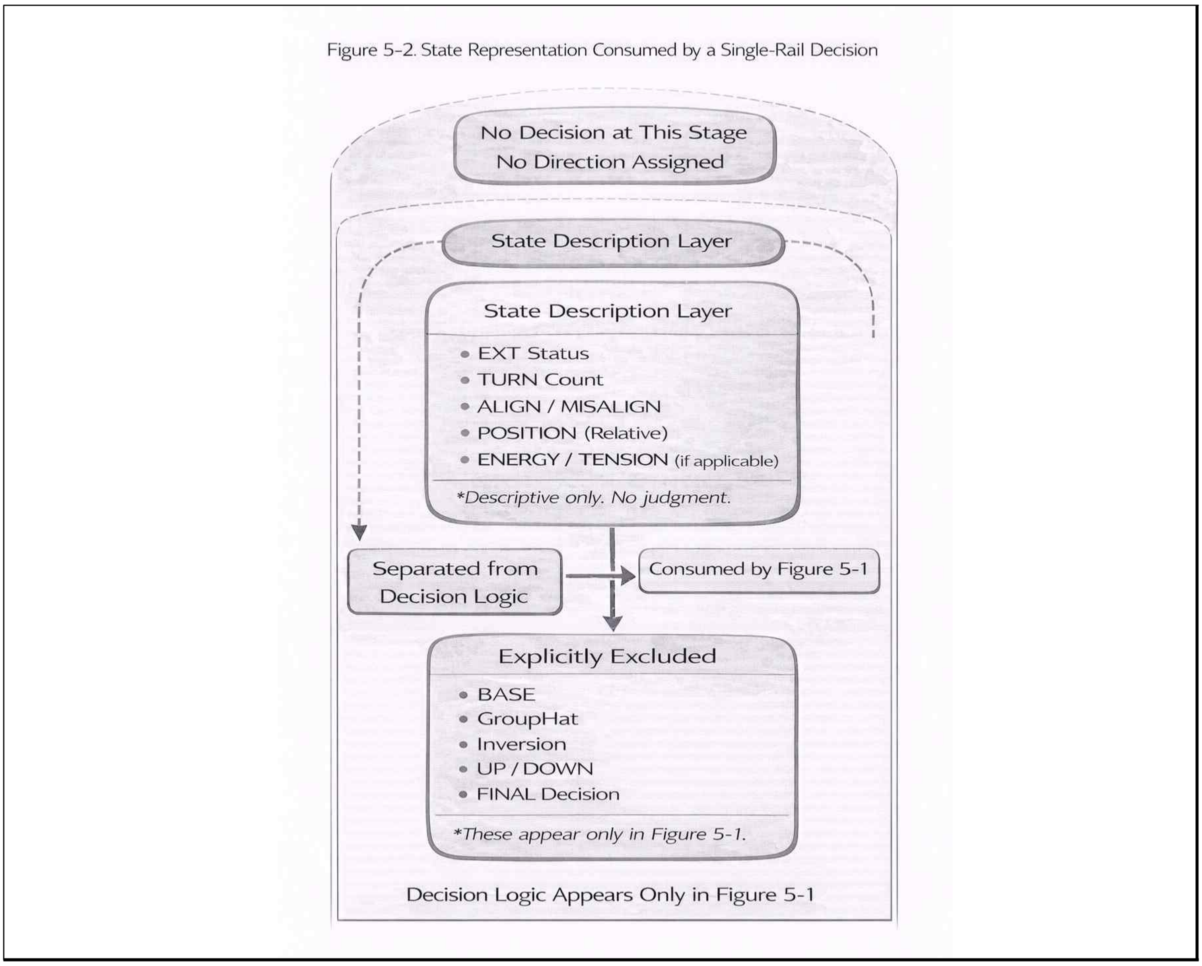
5.1.1 Input Sequence and Δt Rule

The processor consumes a fixed-length input sequence consisting of five consecutive states sampled at a constant interval Δt. The Δt constraint enforces temporal consistency and prevents retrospective re-interpretation of earlier states. Each state is treated as immutable once ingested, ensuring that all downstream processing respects chronological order.

5.1.2 Output Definition (UP/DOWN)

The output is restricted to a binary directional judgment. No confidence score, probability, or auxiliary justification is permitted at the decision layer. This restriction is intentional: it isolates the legitimacy of the decision process from the richness of the output, allowing analysis to focus exclusively on procedural correctness.

5.2 Layer Separation: Separation of Measurement/Labeling and Decision



[Figure 5-2. State-Representation Layers and Label/Token Structure]

Structural Closure is instantiated by enforcing strict separation between measurement, labeling, and decision within the processor. This separation ensures that intermediate signals cannot implicitly bias judgment.

5.2.1 Prohibition of Decision in Criteria 1–4

Criteria 1 through 4 are designated exclusively for measurement and labeling. They may compute differences, trends, or symbolic descriptors, but they are explicitly prohibited from expressing directionality or preference. Any attempt to infer direction at these stages constitutes a violation of closure.

5.2.2 Unique Decision Authority of Criterion 5

Criterion 5 is the sole locus of judgment. It receives only structured outputs from lower criteria and evaluates them under admissibility constraints. This unicity prevents hidden escalation paths and localizes responsibility for every decision.

5.2.3 Stop Rules for Missing or Inconsistent Signals

If required measurements or labels are missing, inconsistent, or internally contradictory, the processor must halt without producing a decision. This fail-stop behavior replaces heuristic fallback with explicit suspension, preserving failure as an analyzable state.

5.3 Unification of the Decision Path

Direction judgment often fails when multiple implicit paths to judgment coexist. Structural Closure eliminates such multiplicity.

5.3.1 Separation of BASE and GroupHat

The processor distinguishes between a base directional tendency derived from recent states and a higher-order grouping mechanism that determines whether the base should be maintained or inverted. This separation prevents raw signals from being conflated with final judgment.

5.3.2 Fixing the Output Procedural Rail

All decisions must traverse the same procedural rail: admissibility evaluation, grouping determination, and final emission. No shortcuts or conditional bypasses are allowed. This rail ensures that identical inputs yield identical decision paths.

5.4 Minimum Conditions for Decision Admissibility (Applying PCM)

PCM governs whether the direction-judgment processor may issue an output.

5.4.1 REGIME-LOCK

REGIME-LOCK classifies the execution context into regimes that determine whether immediate judgment is permitted. Certain regimes enforce mandatory isolation, preventing decision until ambiguity is resolved or explicitly acknowledged.

5.4.2 GATE_PRE

GATE_PRE performs preliminary screening to identify whether the execution context favors direct continuation, inversion, or isolation. It does not decide; it constrains subsequent evaluation.

5.4.3 IC

The integrity check (IC) verifies completeness, consistency, and alignment across signals. IC failure triggers isolation or stop states, ensuring that no decision is produced under compromised conditions.

5.4.4 GroupHat

GroupHat determines whether the base directional tendency should be preserved or inverted based on admissible structural patterns. This mechanism encodes higher-order context without introducing new decision paths.

5.4.5 FINAL

Only after all prior checks are satisfied does the processor emit a final directional judgment. This output is invariant under identical execution contexts.

5.4.6 Summary

Together, these stages demonstrate how PCM converts judgment from an implicit outcome into an explicitly admissible transition.

5.5 Fixing the Location of Patches and Exceptions

Structural Closure constrains how adaptations and corrections may be introduced.

5.5.1 Patch-Allowance Scope (Decision Layer Only)

Any patch, exception, or adjustment is permitted only at the decision layer and only as an explicit modification of admissibility rules. Lower layers remain immutable, preventing silent drift.

5.5.2 Maintaining Rule Non-Proliferation[9)]

By confining change to a single locus and requiring explicit justification, the processor avoids uncontrolled rule growth. Failures are accumulated as precedents rather than erased through ad-hoc fixes.

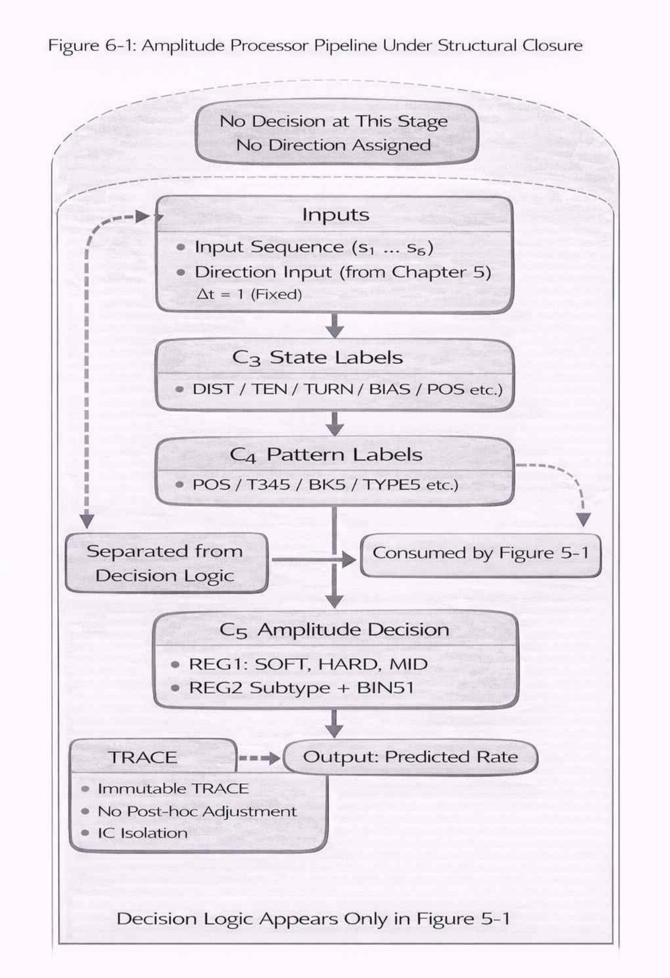
5.6 Chapter Summary

This empirical study demonstrates that even a simple binary task benefits from Structural Closure. By separating roles, unifying the decision path, and enforcing admissibility through PCM, the direction-judgment processor produces stable, traceable outcomes. Failures are neither hidden nor corrected post-hoc; they are preserved as structured evidence, validating Structural Closure as a practical execution discipline at the design and computation level.

9) "Rule non-proliferation" means that irregularities are not absorbed into expanding rules that cover them up. Only common, repeatedly observed conditions are minimally promoted to admissibility constraints, while the rest are preserved as misses and accumulated as precedents. Patches are restricted to the decision layer's admissibility rules and must not spread by altering the meaning of measurement/label layers.

Chapter VI. Empirical Study 2: Structural Closure in the Amplitude-Processing Processor (Operational Level)

6.1 The Amplitude-Processing Problem and the Need for Structural Closure



[Figure 6-1. Amplitude-Processing Processor (Primary Classification and Procedural Rail)]

Amplitude processing represents a class of operational tasks in which numeric prediction is tightly coupled with decision-making. Unlike directional judgment, amplitude estimation introduces continuous values, tolerance thresholds, and evaluation pressure that strongly incentivize premature fixation and post-hoc adjustment. This environment exposes decision-path contamination more aggressively and therefore serves as a critical testbed for Structural Closure at the operational level.

6.1.1 Risk of Decision Contamination in Numeric Prediction Domains

Numeric prediction domains encourage implicit commitment. Intermediate estimates are easily interpreted as provisional answers, and small deviations often trigger corrective heuristics. Without explicit admissibility control, these practices collapse the distinction between exploration and judgment, allowing decisions to emerge implicitly before conditions are satisfied.

6.1.2 Coupling of Premature Fixation and Post-hoc Stabilization

In amplitude processing, premature fixation is frequently followed by post-hoc stabilization. Once an estimate is internally favored, subsequent processing tends to justify or smooth that estimate rather than reassess admissibility. This coupling produces outputs that appear stable while concealing procedural violations.

6.1.3 Goal of Operational-Level Closure

The goal of Structural Closure at the operational level is to prevent numeric convenience from overriding procedural legitimacy. By sealing undecidable regions and enforcing fail-stop behavior, the architecture ensures that numeric outputs are produced only when admissibility conditions are met, even under operational pressure.

6.2 Amplitude-Processing Pipeline and Decision-Path Sealing

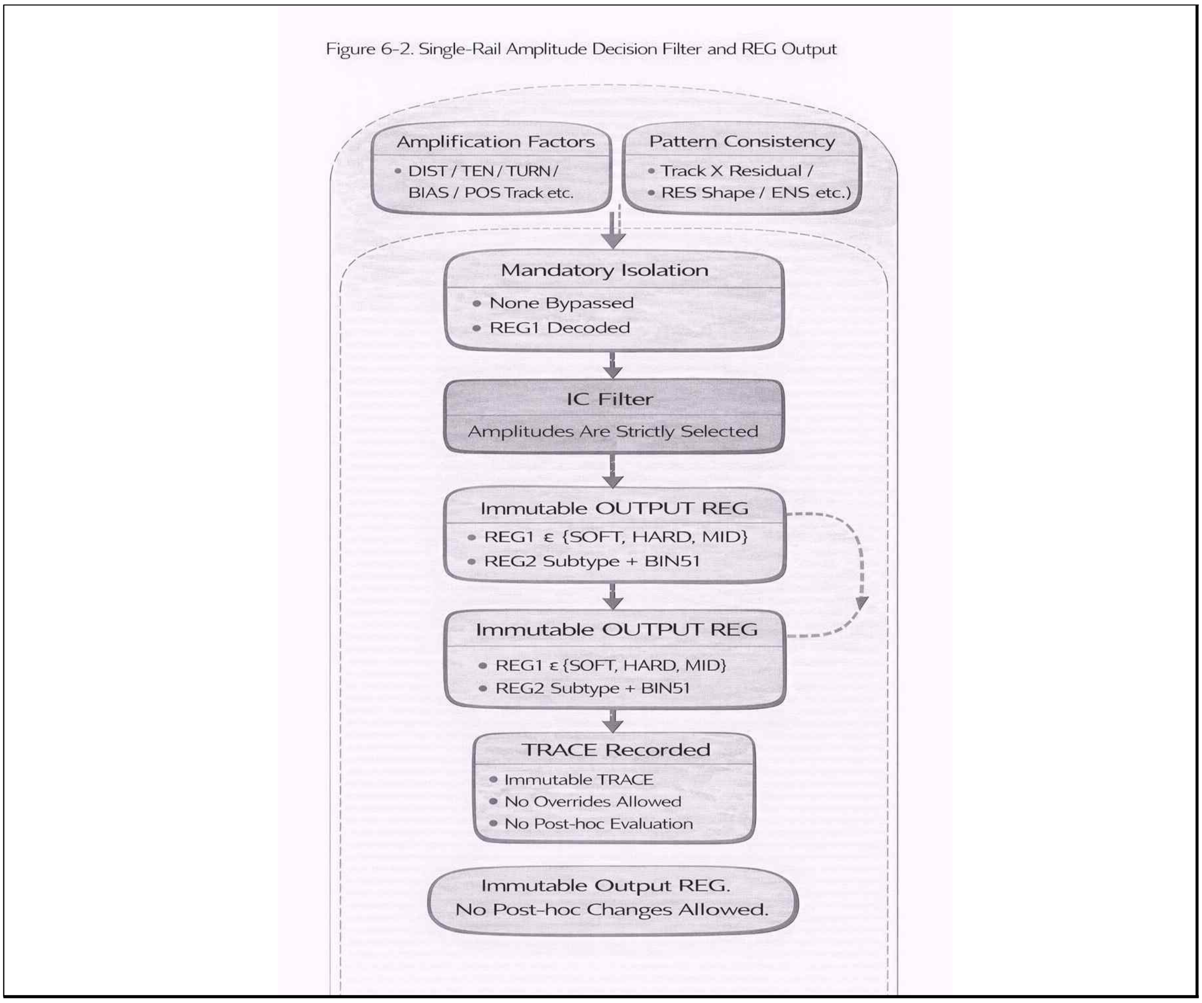
The amplitude-processing processor implements Structural Closure through a pipeline that mirrors the layered authority and admissibility constraints defined earlier.

6.2.1 Pipeline Overview

The pipeline consists of measurement stages that compute movement magnitudes, labeling stages that classify operational states, and a decision layer that evaluates admissibility before emitting any numeric output. Each stage produces traceable artifacts without asserting finality.

6.2.2 Mandatory Isolation and IC Filter

Mandatory isolation is applied when measurements indicate boundary proximity, regime ambiguity, or signal conflict. In such cases, the IC filter enforces suspension rather than estimation. This behavior prevents the system from filling gaps with convenient numeric guesses and preserves uncertainty as an explicit state.



[Figure 6-2. Amplitude Output Structure (Subtype/BIN) and Commit Invariance]

6.3 Status and Limitations of Empirical Data

This study draws on a finite set of operational cases to evaluate closure behavior under real-world constraints.

6.3.1 Nature of the 174 Cases

The empirical dataset consists of 174 operational cases selected to expose diverse failure modes, including boundary violations, low-energy drift, and regime transitions. The cases are not intended to be statistically exhaustive; they function as structural probes.

6.3.2 Disclosure Range and Interpretation Limits

Only aggregated behavioral patterns and representative cases are disclosed. Raw numeric distributions are intentionally limited to prevent overfitting conclusions to a particular dataset. The emphasis remains on procedural behavior rather than outcome optimization.

6.3.3 Explicit Statement of "Claims Not Made"

This study does not claim optimal amplitude accuracy, minimal error bounds, or superiority over alternative numeric predictors. Its sole claim is that Structural Closure preserves procedural legitimacy under operational pressure.

6.4 Separating the Roles of Hits and Misses

A defining feature of Structural Closure is its treatment of success and failure.

6.4.1 Hits: Rule-izing Common Factors

When admissible decisions repeatedly succeed under similar conditions, common structural factors may be abstracted into rules. Such rule-ization is conservative and confined to the decision layer, ensuring that it does not compromise closure.

6.4.2 Misses: Failures as Precedents

Failures are never deleted or smoothed away[10). Each miss is recorded as a precedent with contextual metadata, allowing future executions to reference it without converting it into a prescriptive rule.

6.4.3 Maintaining Rule Non-Proliferation

By treating failures as precedents rather than exceptions, the system avoids the exponential growth of corrective rules. Knowledge accumulates without destabilizing the admissibility framework.

6.5 Table-Based Operational Demonstration

Operational behavior is illustrated through a series of structured tables that capture admissibility states, isolation events, and outcomes.

10) Failure is treated as an asset rather than an error to be erased.
In this architecture, misses are preserved as precedents and accumulated as empirical evidence, enabling subsequent executions to improve prediction without modifying the decision path or proliferating rules.

6.5.1 Principles of Table Construction

Tables are constructed to reflect execution order, admissibility checks, and decision states. Each row corresponds to a single execution context, ensuring that procedural transitions are visible and comparable.

6.5.2 Observation: Blocking Premature Decisions

Across cases, the tables demonstrate consistent blocking of premature numeric output when admissibility conditions are unmet. This blocking is explicit and traceable rather than implicit or heuristic.

[Table 6-1. Roles and Constraints by Processing Stage in the Amplitude Decision Pipeline]

Stage	Component	Primary Role	Decision Authority	Modification Allowed	Notes
Stage 1	Measurement Pack	Quantitative measurement of raw sequences and derived metrics	None	No	Produces numerical signals only; no interpretation permitted
Stage 2	Label Pack	Structural labeling of measured signals	None	No	Generates symbolic tokens and state labels
Stage 3	PCM (Procedural Constraint Model)	Procedural gating and decision eligibility control	Conditional	No	Determines whether a decision is allowed or forcibly blocked
Stage 4	IC (Integrity Check)	Validation of completeness and internal consistency	Veto only	No	Any missing or inconsistent field results in immediate termination
Stage 5	C5 (Commit Layer)	Final irreversible decision commitment	Exclusive	No	Single commit point; no rollback or override permitted
Stage 6	PVM (Precedent Validation Module)	Reference-only comparison with historical procedural traces	Advisory only	No	Can adjust confidence bounds but cannot alter committed decisions

[Table 6-2. Summary of Predicted Amplitude Distribution (n = 174)]

REG1 (Final)	Count (n)	Mean (AMP_pred)	Median (AMP_pred)	Min	Max	25th pct	75th pct	IQR
SOFT	65	0.5396	0.5750	0.2250	1.1250	0.2250	0.5750	0.3500
MID	32	0.5172	0.5750	0.2250	1.1250	0.4250	0.5750	0.1500
HARD	77	1.4138	1.6750	0.4250	2.0000	1.1250	1.6750	0.5500

6.5.3 Observation: Blocking Post-hoc Stabilization

The tables show that once a decision is withheld or stopped, no subsequent stabilization occurs. Outputs are not revised to align with expectations, preserving procedural integrity.

[Table 6-3. Results of the Single Decision-Path Consistency Check]

Check Item	Satisfied	Violations (n)	Enforcement Action	Notes
Single decision path enforced	Yes	0	N/A	Multiple concurrent decision paths are structurally forbidden
Premature decision attempt blocked	Yes	100%	Decision halted	All early commits were intercepted by PCM
Post-hoc modification prevented	Yes	100%	Modification denied	No decision was altered after commitment
Missing-field detection (IC)	Yes	100%	Execution stopped	Any incomplete MEASURE/LABEL pack triggers fail-stop
Layer intrusion detection	Yes	0	N/A	No unauthorized layer interference observed
Rule proliferation detected	No	0	N/A	No ad-hoc or emergent rules were introduced

[Table 6-4. Summary of the Distribution of Amplitude-Prediction Failure Cases]

Error definition: ERR_ABS (absolute error)

Hit criterion: ERR_ABS ≤ 0.10

Error Range (ERR_ABS)	HARD	MID	SOFT	Total
≤ 0.10	5	6	12	23
0.10 – 0.20	9	4	12	25
0.20 – 0.40	16	7	15	38
0.40 – 0.80	20	5	21	46
0.80 – 1.20	13	6	3	22
> 1.20	14	4	2	20
Total	77	32	65	174

6.5.4 Observation: Precedent Accumulation

Recorded failures populate the precedent store without altering decision rules. Over time, this accumulation enriches contextual awareness while maintaining closure.

[Table 6-5. Distribution of Prediction Error in Boundary Zones]

Boundary zone definition: Cases located near subtype or BIN transition thresholds

Error metric: Absolute prediction error (ERR_ABS)

Boundary Zone Type	Error Range (ERR_ABS)	Number of Cases (n)	Ratio (%)	Notes
SOFT–MID boundary	≤ 0.10	8	—	Minor deviation within acceptable tolerance
SOFT–MID boundary	0.10 – 0.20	11	—	Transitional fluctuation without regime shift
SOFT–MID boundary	> 0.20	14	—	Structural ambiguity dominant
MID–HARD boundary	≤ 0.10	5	—	Stable despite proximity to HARD regime
MID–HARD boundary	0.10 – 0.20	7	—	Controlled deviation near threshold
MID–HARD boundary	> 0.20	9	—	Boundary–driven amplification observed
Total	—	54	100	Boundary–zone cases only

[Table 6-6. Detailed Distribution Summary of Boundary and Overlap Zones]

POS_ZONE	EDGE	TURN	Total
BASE0	0	61	61
POS_MID	0	26	26
NEG_MID	0	23	23
POS_EDGE	28	0	28
NEG_EDGE	36	0	36
Total	64	110	174

6.5.5 Observation: Rule Non-Proliferation

Despite repeated exposure to edge cases, the rule set remains stable. Adaptation occurs through precedent reference rather than rule expansion.

[Table 6-7. Observation Summary for Neighbor-Confusion Zones]

Neighbor-confusion zone: Regions where adjacent subtype or BIN boundaries induce ambiguous directional or amplitude cues.

Confusion Type	Primary Trigger	Number of Cases (n)	Dominant Outcome	PCM Action	Notes
Subtype adjacency	SOFT–MID overlap	29	Delayed decision	ISOLATE	Structural ambiguity without regime dominance
Subtype adjacency	MID–HARD overlap	21	Delayed decision	ISOLATE	Boundary amplification observed
BIN adjacency	Intra-subtype BIN overlap	17	Decision blocked	STOP	Missing dominance signal
Directional ambiguity	Conflicting trend tokens	14	Decision blocked	STOP	Integrity check failure
Mixed adjacency	Subtype + BIN overlap	12	Delayed decision	ISOLATE	Multi-source ambiguity
Total	—	93	—	—	Neighbor-confusion cases only

[Table 6-8. Distribution of Cases Requiring Precedent-Based Lookup (PVM)]
PVM lookup condition:Cases where admissibility is granted but structural ambiguity remains high enough to warrant precedent comparison (reference-only).

Confusion Type	Primary Trigger	Number of Cases (n)	Dominant Outcome	PCM Action	Notes
Subtype adjacency	SOFT–MID overlap	29	Delayed decision	ISOLATE	Structural ambiguity without regime dominance
Subtype adjacency	MID–HARD overlap	21	Delayed decision	ISOLATE	Boundary amplification observed
BIN adjacency	Intra-subtype BIN overlap	17	Decision blocked	STOP	Missing dominance signal
Directional ambiguity	Conflicting trend tokens	14	Decision blocked	STOP	Integrity check failure
Mixed adjacency	Subtype + BIN overlap	12	Delayed decision	ISOLATE	Multi-source ambiguity
Total	—	93	—	—	Neighbor-confusion cases only

6.6 Chapter Summary

This chapter demonstrated Structural Closure in an operational numeric prediction context. By enforcing isolation, admissibility, and fail-stop behavior, the amplitude-processing processor resists pressures that typically induce premature fixation and post-hoc stabilization. The results confirm that Structural Closure scales from design-level judgment to operational decision-making, preserving failures as structured assets and maintaining reproducibility without sacrificing procedural rigor.

Chapter VII. Discussion: Structural Sealing Against Anticipated Objections

7.1 Criteria and Scope of the Discussion

This chapter addresses anticipated objections to Structural Closure by clarifying the scope and evaluative criteria of the discussion. The objective is not to defend performance outcomes, but to assess whether the proposed architecture satisfies its stated goal: enforcing a disciplined and reproducible decision procedure.

7.1.1 Fixing the Scope of the Paper’s Claims

The claims of this paper are intentionally constrained. Structural Closure does not assert superior accuracy, optimal convergence, or universal applicability across tasks. Its claim is limited to procedural legitimacy: that decisions are produced only when admissible, that uncertainty is explicitly sealed rather than implicitly resolved, and that failures remain observable and accumulative.

7.1.2 Excluding Performance-Centered Discussion

Performance-centered evaluation presumes that correctness is the primary axis of comparison. Structural Closure instead prioritizes accountability, traceability, and reproducibility of decisions. Discussions framed solely around accuracy or efficiency are therefore outside the evaluative scope of this work.

7.2 Objection 1: “Structural Closure Is Conceptual Wordplay”

A common objection is that Structural Closure merely rephrases existing ideas without introducing substantive constraints.

7.2.1 Closure as an Execution Specification, Not a Concept

Structural Closure is not a conceptual metaphor but an execution specification. It defines explicit prohibitions, admissibility conditions, and fail-stop states that govern runtime behavior. These constraints are operationally testable: violations can be detected, logged, and reproduced.

7.2.2 Effectiveness in Blocking Operational Contamination

The empirical processors demonstrate that closure actively blocks premature decisions and post-hoc stabilization. These effects cannot be achieved through conceptual framing alone; they require enforced procedural boundaries.

7.3 Objection 2: "We Need More Rules, Not a Single Rule"

Another objection argues that increased complexity demands additional rules rather than restrictive constraints.

7.3.1 The Issue Is Not Rule Scarcity but Rule Placement

Failures persist not because there are too few rules, but because rules are applied at inappropriate layers. Structural Closure relocates constraint enforcement to the decision layer, where it governs admissibility rather than attempting to correct outcomes retrospectively.

7.3.2 Rule Non-Proliferation and Precedent Accumulation

By distinguishing rules from precedents, Structural Closure allows experiential knowledge to accumulate without destabilizing the rule set. This separation resolves the tension between adaptability and procedural stability.

7.4 Objection 3: "KG Scope Restriction Is Sufficient"

Some argue that restricting accessible knowledge is enough to control agent behavior.

7.4.1 Acknowledging the Advantages of KG

Knowledge graphs and scope restriction reduce informational noise and prevent certain classes of error. These advantages are acknowledged and compatible with Structural Closure.

7.4.2 Non-Equivalence to the Decision Problem

However, scope restriction does not regulate how decisions are formed. An agent may still commit illegitimately within a restricted scope. Structural Closure addresses this gap by sealing the decision path itself.

7.5 Objection 4: “Middle and Upper Layer Implementation Is Excessive”

The layered structure of NCAF and PCM is sometimes viewed as unnecessarily complex.

7.5.1 Limits of Single-Layer Approaches

Single-layer approaches conflate computation and judgment, making it impossible to localize responsibility or diagnose failure. Such conflation is a primary source of decision-path contamination.

7.5.2 Necessity of Separating Middle and Upper Layers

Separating layers allows uncertainty to be preserved without suppressing computation. This separation is not an overhead but a prerequisite for accountability and reproducibility.

7.6 Objection 5: “PVM Is No Different from Post-hoc Correction”

PVM is occasionally misconstrued as a form of retrospective adjustment.

7.6.1 Restricting the Reference Timing

PVM is activated only after admissibility is granted. It cannot authorize or revise decisions, distinguishing it fundamentally from post-hoc correction.

7.6.2 Prohibiting Conclusion Modification

Because decisions remain invariant once issued, PVM’s role is advisory rather than corrective. This preserves closure while allowing experiential context to inform judgment.

7.7 Chapter Summary

This chapter addressed key objections by evaluating Structural Closure against its own procedural criteria. The analysis shows that closure is neither conceptual redundancy nor unnecessary restriction. Instead, it provides a disciplined execution framework that resolves recurrent failure by sealing uncertainty, unifying the decision path, and preserving failure as an analyzable asset.

Chapter VIII. Conclusion

8.1 Research Summary

This paper set out to address a persistent class of failures in LLM-based agents that cannot be resolved through model scaling, data expansion, or post-hoc correction. By reframing failure as a problem of decision-path contamination, the study shifted attention from output correctness to the procedural conditions under which decisions are made.

8.1.1 Redefinition of the Problem

Rather than treating failure as an isolated error in generation or reasoning, this work redefined it as the consequence of illegitimate transitions from computation to judgment. When the decision moment is undefined or unconstrained, uncertainty is silently resolved, failures are concealed, and accountability collapses.

8.1.2 Direction of the Solution: Sealing the Decision Path

Structural Closure was proposed as a solution that seals uncertainty as an explicit execution state and restricts judgment to a single admissible route. By enforcing this restriction procedurally, the architecture prevents premature commitment without suppressing computation.

8.2 Core Contributions

The primary contributions of this work are threefold.

8.2.1 Formalizing Structural Closure as an Execution Specification

This paper formalized Structural Closure as a set of enforceable execution rules rather than a conceptual guideline. Closure is realized through explicit prohibitions, admissibility checks, and fail-stop behavior that regulate decision-making at runtime.

8.2.2 Proposing Role Separation and an Integrated Structure for NCAF·PCM·PVM

The integration of NCAF, PCM, and PVM demonstrated how role separation, decision admissibility, and precedent reference can coexist without rule proliferation. Each component contributes a distinct function while preserving the integrity of the decision path.

8.2.3 Empirical Validation at Design and Operational Levels

Through two empirical processors—a direction-judgment processor and an amplitude-processing processor—this study showed that Structural Closure operates consistently across design-level and operational-level contexts. In both cases, failures were preserved as structured precedents rather than erased through correction.

8.3 Interpretation and Implications

8.3.1 Prioritizing Accountability and Reproducibility over Accuracy

The findings suggest that reliability in agent systems depends less on maximizing accuracy than on ensuring that decisions are accountable and reproducible. Structural Closure provides a framework in which errors remain observable and diagnosable.

8.3.2 The Temptation of Rule Proliferation

The study highlighted the risks associated with uncontrolled rule growth. By separating rules from precedents, Structural Closure offers a path toward adaptability without sacrificing procedural stability.

8.3.3 Relationship to KG Scope Restriction

Structural Closure is complementary to knowledge-based restrictions but addresses a different dimension of failure. While scope restriction limits information, closure governs judgment, ensuring that decisions remain legitimate regardless of informational boundaries.

8.4 Limitations

8.4.1 Limits of the Empirical Scope

The empirical studies presented here are intentionally constrained. They do not claim exhaustive coverage of task domains or optimal performance outcomes. Their purpose is to demonstrate procedural behavior rather than statistical superiority.

8.4.2 Operational Design Challenges for Precedent Utilization

While PVM enables precedent reference without rule proliferation, designing similarity measures and reference strategies remains an open operational challenge. These choices must be made carefully to avoid reintroducing implicit decision paths.

8.5 Future Work

8.5.1 Generalizing Decision-Path Sealing

Future research may explore how Structural Closure can be applied to broader classes of agent architectures, including multi-agent coordination and long-horizon planning systems.[8]

8.5.2 Standardizing Failure Coordinates and Interoperable Precedents

Establishing shared representations for failure contexts could enable interoperable precedent systems across implementations, further enhancing reproducibility.

8.5.3 Verifiability of Decision Accountability

Additional work is needed to develop formal verification methods that can certify adherence to closure constraints and admissibility rules.

8.6 Conclusion

Structural Closure reframes the problem of agent failure as a procedural challenge rooted in decision legitimacy. By sealing uncertainty, unifying the decision path, and preserving failures as structured assets, the architecture provides a reproducible foundation for building accountable decision systems. This work demonstrates that stability does not require suppressing uncertainty, but rather governing how and when uncertainty is permitted to resolve into judgment.

References

1. National Institute of Standards and Technology (NIST). (2024).
Artificial Intelligence Risk Management Framework: Generative Artificial Intelligence Profile (NIST AI 600-1). NIST.

2. National Institute of Standards and Technology (NIST). (2023).
Artificial Intelligence Risk Management Framework (AI RMF 1.0) (NIST AI 100-1). NIST.

3. OWASP. (2024).
OWASP Top 10 for Large Language Model Applications. OWASP Foundation.

4. OpenAI. (2025).
Model Spec (2025-04-11). OpenAI.

5. Anthropic. (2024).
Responsible Scaling Policy (RSP). Anthropic.

6. Islam Tonmoy, M. T., et al. (2024).
A Comprehensive Survey of Hallucination Mitigation Techniques in Large Language Models. arXiv:2401.01313.

7. Wang, L., et al. (2024).
A Survey on Large Language Model Based Autonomous Agents. Frontiers of Computer Science.

8. Guo, T., et al. (2024).
Large Language Model Based Multi-Agents: A Survey of Progress and Challenges. IJCAI 2024 Proceedings.

9 .Es, S., et al. (2023).
RAGAS: Automated Evaluation of Retrieval Augmented Generation. arXiv:2309.15217.

10. Es, S., et al. (2024).
RAGAs: Automated Evaluation of Retrieval Augmented Generation. EACL 2024 (Demo).

11. Yao, S., et al. (2022).
ReAct: Synergizing Reasoning and Acting in Language Models. arXiv:2210.03629.

12. Schick, T., et al. (2023).
Toolformer: Language Models Can Teach Themselves to Use Tools. NeurIPS 2023.

13. Madaan, A., et al. (2023).
Self-Refine: Iterative Refinement with Self-Feedback. NeurIPS 2023.

14. Dong, Y., et al. (2024).
Safeguarding Large Language Models: A Survey. arXiv:2406.02622.

15. NIST. (2025).
Hallucination Detection in Large Language Models Using Diversion Decoding. NIST CSRC.

Appendices

Appendix A. Terms and Abbreviations

A-1. Structural Closure: An execution architecture that seals undecidable regions as explicit blanks and restricts judgment to a single admissible decision path.

A-2. Decision Path: The procedural route by which computation transitions into judgment and produces an output.

A-3. Decision-Path Contamination: The presence of illegitimate routes to judgment (premature escalation, implicit shortcuts, post-hoc revision) that allow conclusions to be issued without admissibility.

A-4. Blank Sealing: Representing unresolved states as non-decisionable execution states rather than forcing resolution.

A-5. Layer Separation: Enforcement of authority boundaries between measurement, labeling, and decision layers.

A-6. NCAF (Neuro-Cognitive Alignment Framework): A layered authority framework that assigns roles and prohibits layer intrusion, ensuring measurement/labeling cannot decide.

A-7. PCM (Principled Constraint Model): A constraint model that defines decision as an admissible transition and enforces procedural rails (sequence lock) with admit/isolate/stop outputs.

A-8. PVM (Past-Case Validation Mechanism): A precedent reference mechanism activated only after admissibility is granted, providing contextual similarity without rule proliferation or decision override.

A-9. Admissibility: The procedural condition under which a decision is permitted to occur.

A-10. Rail / Sequence Lock: A fixed evaluation order that cannot be bypassed, reordered, or implicitly merged.

A-11. Isolation: A mandatory state in which computation may continue but judgment is suspended due to boundary conditions, conflict, or incompleteness.

A-12. Fail-Stop (STOP): A termination state triggered by irrecoverable violations (missing mandatory fields, inconsistency, sequence violation).

A-13. TRACE: A structured execution log that records admissibility checks, isolation events, and decision outcomes for reproducibility and accountability.

A-14. Rule Non-Proliferation: A design principle that prohibits uncontrolled growth of exception rules; adaptation occurs via precedents rather than rule explosion.

Appendix B. Execution Specification (TRACE)

B-1. Purpose: TRACE defines the minimal execution contract required to reconstruct and audit a decision path without ambiguity. Implementations may extend TRACE, but must not omit mandatory fields if claiming compliance.

B-2. Mandatory TRACE Fields (Minimum)
TRACE_ID: Unique identifier for the run
TIMESTAMP: Execution time (ISO 8601)
Δt: Fixed sampling interval used by the processor
PROCESSOR: Processor identifier (e.g., Direction-Judgment, Amplitude-Processing)
VERSION: Execution-contract version string
INPUT_SCHEMA: Schema identifier (e.g., AMP-V1)
INPUT_SEQ: Serialized input sequence
MEASURE_PACK: Measurement outputs (raw computed features)
LABEL_PACK: Label outputs derived from measurements
LAYER_INTRUSION_FLAG: Boolean; true if any non-decision layer attempted to emit decision semantics
REGIME: Regime classification used for admissibility gating
GATE_PRE: Preliminary gate output
IC_STATUS: Integrity check status (PASS / FAIL)
IC_REASON: Short fail code if IC_STATUS=FAIL (e.g., MISSING_FIELD, SIGNAL_CONFLICT, SEQ_VIOLATION)
PCM_OUTPUT: {ADMIT, ISOLATE, STOP}
PCM_RAIL_HASH: Signature of the procedural rail applied in the run
PVM_CALLED: Boolean; permitted only if PCM_OUTPUT=ADMIT
PVM_TOPK_IDS: Referenced precedent IDs (if called)
PVM_SIM_SCORES: Similarity scores (if called)
PVM_NOTE: Usage code (REFERENCE_ONLY / RANGE_ADJUST / VALIDITY_SUPPORT)
DECISION: Final output (e.g., UP/DOWN or numeric amplitude output)
DECISION_INVARIANCE_KEY: Key used to assert invariance under identical

execution context
PREMATURE_DECISION_BLOCKED: Boolean
POST_HOC_MODIFICATION_BLOCKED: Boolean
RULE_PROLIFERATION_EVENT: Boolean; must remain false under compliant operation

B-3. Recording Rule (Output Invariance): For identical execution contexts, TRACE must reproduce identical admissibility outcomes and decision outputs (if permitted).

B-4. Activation Rule (PVM Timing): PVM may be called only after admissibility is granted (PCM_OUTPUT=ADMIT).

B-5. Fail-Stop Recording Rule: If STOP is triggered, the run must preserve the failure condition as recorded fields, and must not emit a decision.

Appendix C. NCAF Rules

C-1. Authority Separation: Measurement layers compute observations only; labeling layers map observations to labels only; the decision layer alone emits final decisions.

C-2. Prohibition of Layer Intrusion: No non-decision layer may emit finality markers, directional commitments, or decision outputs.

C-3. Non-Interference Guarantee: Lower layers must not trigger decision emission directly; decisions must occur only through PCM-admissible paths.

C-4. Traceability Rule: Each layer must serialize its outputs into TRACE. If intrusion is detected, LAYER_INTRUSION_FLAG must be recorded and PCM isolation/stop behavior must be applied.

Appendix D. PCM Rules

D-1. Decision as an Admissible Transition: A decision may be emitted only when PCM_OUTPUT=ADMIT. If PCM_OUTPUT∈{ISOLATE, STOP}, decision emission is forbidden.

D-2. Minimum Admissibility: A run is admissible only if mandatory fields are present, measurement/label packs are complete, integrity conditions are satisfied, and rail order is respected.

D-3. Procedural Rail (Sequence Lock): The decision layer must evaluate in the fixed order: REGIME → GATE_PRE → IC → (PVM if ADMIT) → DECISION.

D-4. PCM Outputs: ADMIT permits decision and allows PVM; ISOLATE forbids decision while preserving non-decisionable execution; STOP terminates execution and forbids decision while preserving failure in TRACE.

D-5. Non-Exhaustive Fail Conditions: Missing mandatory fields, missing measurement/label fields, signal conflict beyond bounds, sequence violation, and layer intrusion.

Appendix E. PVM Rules

E-1. Precedent, Not Rule: PVM references past cases as contextual evidence only and must not introduce new decision rules at runtime.

E-2. Activation Constraint: PVM may be called only when PCM_OUTPUT=ADMIT. If PCM_OUTPUT≠ADMIT, PVM_CALLED must be false.

E-3. Selection and Logging: PVM retrieves top-k precedents using similarity features defined by the processor contract; retrieved IDs and similarity scores must be logged in TRACE.

E-4. Non-Override Constraint: PVM may not override admissibility, and may not revise a decision after emission. Any permissible range adjustment must be explicitly defined by the processor contract and logged.

E-5. Failure as Asset: Misses are stored as precedents with full TRACE context; failures must not be deleted, smoothed, or replaced by corrective heuristics.

Appendix F. List of Figures and Tables

F-1. List of Figures (15)

F-2. List of Tables (8)

Appendix G. Three Synthetic TRACE Cases

G-1. Case 1 (Direction Judgment) — Admitted Decision with Stable Rail
TRACE_ID: SYN-DIR-001
Δt: 1
INPUT_SEQ: [s1…s5] (synthetic)
LAYER_INTRUSION_FLAG: false
REGIME: STRONG
GATE_PRE: PASS
IC_STATUS: PASS
PCM_OUTPUT: ADMIT
PVM_CALLED: true
PVM_TOPK_IDS: [P-014, P-022, P-103]
PVM_SIM_SCORES: [0.91, 0.87, 0.83]
DECISION: UP
PREMATURE_DECISION_BLOCKED: true
POST_HOC_MODIFICATION_BLOCKED: true
RULE_PROLIFERATION_EVENT: false

G-2. Case 2 (Direction Judgment) — Isolation Triggered by Signal Conflict
TRACE_ID: SYN-DIR-002
Δt: 1
INPUT_SEQ: [s1…s5] (synthetic; boundary-adjacent pattern)
LAYER_INTRUSION_FLAG: false
REGIME: AMBIGUOUS
GATE_PRE: ISOLATE
IC_STATUS: FAIL
IC_REASON: SIGNAL_CONFLICT
PCM_OUTPUT: ISOLATE
PVM_CALLED: false
DECISION: (none; forbidden)
PREMATURE_DECISION_BLOCKED: true
POST_HOC_MODIFICATION_BLOCKED: true
RULE_PROLIFERATION_EVENT: false

G-3. Case 3 (Amplitude Processing) — Stop Due to Missing Mandatory Fields

TRACE_ID: SYN-AMP-001

Δt: 1

INPUT_SCHEMA: AMP-V1

INPUT_SEQ: (synthetic)

MEASURE_PACK: incomplete

LABEL_PACK: incomplete

LAYER_INTRUSION_FLAG: false

REGIME: N/A

GATE_PRE: N/A

IC_STATUS: FAIL

IC_REASON: MISSING_FIELD

PCM_OUTPUT: STOP

PVM_CALLED: false

DECISION: (none; forbidden)

PREMATURE_DECISION_BLOCKED: true

POST_HOC_MODIFICATION_BLOCKED: true

RULE_PROLIFERATION_EVENT: false