NRR-Transfer: Cross-Domain Transfer of Phase 1.5 Operators Under Fixed Interface Constraints

Kei Saito*
Independent Researcher, Japan
kei.saito.research@gmail.com

February 2026

Part of the Non-Resolution Reasoning (NRR) research program.

Abstract

Stateful reasoning tasks—managing semantic ambiguity, tracking document relevance, prioritizing agent actions—require persistent state management across multiple interactions with stateless LLM APIs. Phase 1.5, discovered during the implementation of the Interpretation Management Engine (IME) for Non-Resolution Reasoning (NRR), provides an operator-based solution by separating decision-making (LLM selects strengthen/dampen operators) from state execution (client applies deterministic updates). This paper tests whether that pattern is transferable across domains under fixed interface constraints. We validate Phase 1.5 across six semantically distinct domains (IME, RAG, Agent, Planning, Multi-agent, Multimodal) under an IME-aligned protocol (3 models × 2 temperatures × 3 trials), totaling 324 runs (1,512 turns). Results show 100% extraction success across the 324-run protocol and temperature-robust efficiency: mean difference between T=0.3 and T=0.0 is 0.0072 tokens/turn, with maximum absolute difference 0.4444 across scenario-model pairs. Operator selection varies substantially across domains and conditions, yet the interface remains stable without domain-specific redesign. These findings support a central claim: cross-domain transferability is the primary empirical result, rather than task-specific optimization, and NRR contributes the design principle that makes this transfer implementable.

Implementation: A reference implementation accompanying this work is available at https://github.com/kei-saito-research/nrr-transfer.

Series status note (as of February 26, 2026): NRR-IME is treated as under moderation in this series, and this manuscript does not assume IME acceptance.

Keywords: Phase 1.5, cross-domain transfer, Non-Resolution Reasoning, cross-domain validation, operator abstraction, stateful reasoning, LLM architecture

1 Introduction

This paper is one module in an ongoing Non-Resolution Reasoning (NRR) research program, not a closed endpoint. Companion papers cover foundational theory, text-to-state mapping, and implementation strategy with distinct scopes. A series-level Program Map and update status will be maintained in repository documentation.

Large language models accessed via stateless APIs (e.g., OpenAI, Anthropic, Google) have no inherent memory between requests. Yet many real-world applications require stateful reasoning:

*ORCID: 0009-0006-4715-9176

tracking which interpretations of an ambiguous term remain plausible, maintaining document relevance scores in conversational search, or dynamically prioritizing tasks as an agent's context evolves. The standard approach sends the complete state to the LLM each turn, incurring token costs that grow linearly with state complexity and conversation length.

Non-Resolution Reasoning (NRR) [1] proposed a framework for managing semantic ambiguity without forcing premature resolution to single interpretations. Rather than asking "Which meaning is correct?", NRR maintains multiple plausible interpretations with dynamic weights, allowing ambiguity to persist until context naturally resolves it. Implementing this idea efficiently on stateless APIs led to Phase 1.5 operator abstraction. Accordingly, validity is condition-indexed rather than unconditional. A state update is treated as valid under current context and interface constraints, not as universally valid across all contexts. NRR is not an anti-LLM framework. NRR does not replace standard LLM use. NRR optimizes when to commit and when to defer, under explicit conditions.

Phase 1.5, first identified during the implementation of the Interpretation Management Engine (IME) [3]—a system for managing semantic ambiguity using NRR principles—provides a solution. The core insight is to separate decision-making (performed by the LLM) from state execution (performed deterministically on the client). The LLM selects only *what* to change, while the client executes *how* to change it. As of February 26, 2026, IME is handled as an under-moderation companion manuscript, and the claims here are based on reproducible artifacts rather than acceptance status.

However, a critical question emerged: *Is Phase 1.5 specific to ambiguity management, or does it represent a more fundamental interface?* While Phase 1.5 was discovered in the context of NRR and semantic interpretation, its efficiency stems from separation of concerns, not from ambiguity-specific content. This suggests broader applicability to any stateful reasoning task over weighted items.

To test this hypothesis, we extend Phase 1.5 beyond its origins in ambiguity management to six semantically distinct domains:

1. **IME (Interpretation Management)**: Managing semantic ambiguity with multiple plausible interpretations (e.g., "bank" = financial institution vs. riverbank)
2. **RAG (Retrieval-Augmented Generation)**: Tracking document relevance in conversational search
3. **Agent**: Prioritizing tasks in dynamic environments
4. **Planning**: Managing project tasks where constraints and priorities shift based on external events
5. **Multi-agent**: Integrating conflicting expert perspectives from different specialists
6. **Multimodal**: Navigating design spaces with subjective aesthetic judgments

These six domains span critical dimensions of reasoning tasks. The first three (IME, RAG, Agent) involve relatively objective criteria—semantic plausibility, document relevance, task urgency. The latter three (Planning, Multi-agent, Multimodal) require subjective judgment—resource constraints, expert credibility, aesthetic preferences. Together, they test whether Phase 1.5 represents a truly generic interface for stateful reasoning.

1.1 Contributions

This paper makes the following contributions:

1. **Cross-domain validation**: We evaluate Phase 1.5 across six semantically distinct domains (IME, RAG, Agent, Planning, Multi-agent, Multimodal).

2. **IME-aligned protocol**: We run 3 models × 2 temperatures × 3 trials (324 runs / 1,512 turns), achieving 100% operator extraction success.

3. **Temperature robustness**: We show mean token difference of 0.0072 tokens/turn between T=0.3 and T=0.0 (max absolute difference: 0.4444).

4. **Cross-domain transfer**: We demonstrate consistent performance independent of domain semantics, despite substantial operator-pattern variation.

5. **NRR-derived design rationale**: We trace this generality to a separation principle first operationalized in NRR/IME: semantic decision in the LLM, deterministic state mechanics in the client.

6. **Architectural guidance**: We provide concrete principles for building non-monolithic LLM systems with operator-based state management.

1.2 Series Alignment: Notation and Metrics

Notation contract. Transfer uses item-level operators for execution ($\sigma_{\text{item}}, \delta_{\text{item}}$), abbreviated as σ, δ after definition. This is distinct from state-level operators in NRR-Phi (e.g., σ_{state} calibration); the symbol family is shared but the operation level differs.

Metric contract. This paper primarily evaluates interface transfer behavior (extraction reliability and token efficiency across domains). These operational metrics complement, but do not replace, foundational non-collapse metrics reported in Core/Phi.

Boundary statement. "Success" in this paper denotes successful operator-target extraction and stable update execution under tested conditions; it is not a universal task-quality guarantee.

2 Background

2.1 From Non-Resolution Reasoning to Phase 1.5

Non-Resolution Reasoning (NRR) [1] proposed a framework for managing semantic ambiguity without forcing premature resolution. Consider the word "bank" in the sentence "I went to the bank." Without additional context, NRR maintains both interpretations—financial institution and riverbank—as weighted possibilities rather than collapsing to a single meaning.

Traditional NLP systems resolve ambiguity immediately, often incorrectly. NRR instead preserves multiple interpretations, adjusting their relative plausibility as context accumulates. This requires efficient state management: tracking interpretation weights across multiple LLM interactions without incurring prohibitive token costs.

The Interpretation Management Engine (IME) [3] was developed to implement NRR principles. During IME's development, a critical pattern emerged: rather than sending complete interpretation states to the LLM each turn (expensive), we could expose only *labels* and let the LLM select simple operators (*strengthen* or *dampen*). The client would then execute these operators deterministically to update weights.

This pattern—later formalized as Phase 1.5—achieved dramatic efficiency gains (74–78% token reduction) while maintaining full reasoning capability. Crucially, the pattern's efficiency stemmed not from ambiguity-specific properties, but from a more fundamental architectural principle: *separating decision (what to change) from execution (how to change it).*

This observation suggested Phase 1.5 might generalize beyond ambiguity management to any stateful reasoning task involving weighted items. The present work tests this hypothesis across six diverse domains.

2.2 Phase 1.5 Operator Abstraction

Phase 1.5 is an operator-based approach to stateful reasoning on stateless LLM APIs. The core pattern consists of three components:

1. **State representation**: Weighted items $\{(i_1, w_1), (i_2, w_2), \ldots, (i_n, w_n)\}$ where $\sum w_i = 1$
2. **LLM decision**: Given only item labels and context, the LLM selects an operator (σ or δ) and target item
3. **Client execution**: The client applies the selected operator deterministically to update weights

The two operators are designed to satisfy operator design principles (NRR-Phi [2], Appendix D) through bounded additive updates followed by deterministic renormalization [3]:

- σ (*strengthen*): Increase target weight by an additive step with upper bound clipping[1]
- δ (*dampen*): Decrease target weight by an additive step with lower bound clipping

Note: These operators act on individual items, implementing NRR's relative structure preservation principle ([2], Appendix D) via bounded updates and renormalization. This differs from the global state-level operators introduced in NRR-Phi ([2], Appendix D).

Relationship to NRR operators: These operators implement the *relative structure preservation principle* identified in NRR-Phi ([2], Appendix D) as critical for non-collapse. By enforcing deterministic bounded updates with explicit renormalization, they avoid degenerate drift while keeping alternatives active. The difference is one of abstraction level: NRR-Phi Appendix D defines mathematical properties operators must satisfy; NRR-IME [3] and this work demonstrate practical implementations satisfying those properties.

Key insight: In this interface, the LLM does not see or generate actual weights. It only decides *which* item to adjust and *how* (strengthen or dampen). This separation of concerns achieves token efficiency while maintaining semantic expressiveness. Figure 1 illustrates this horizontal flow pattern.

Why binary operators suffice: The choice of two operators (σ, δ) is not a simplification but an *alignment with LLM competence boundaries*. LLMs excel at qualitative reasoning—determining that "this document is more relevant given the new query"—but struggle with quantitative estimation—assigning precise weights like 0.73 vs. 0.68. By constraining the interface to directional decisions (strengthen/dampen), Phase 1.5 delegates quantitative mechanics (weight magnitude, normalization) to deterministic client-side logic where such precision is naturally achievable. This design respects what LLMs do well while avoiding what they do poorly.

This pattern has been validated across six domains (IME, RAG, Agent, Planning, Multi-agent, Multimodal), demonstrating:

- 74–78% token reduction vs. naive baselines
- 100% operator extraction success (1,512/1,512 turns, 324 runs)
- Temperature-robust efficiency ($\Delta_{0.3-0.0} = 0.0072$ tokens/turn on average)
- Consistent efficiency across diverse reasoning tasks and model families

[1] These operators (σ, δ) act on individual items within a state and are implemented as bounded additive steps plus renormalization. This differs from state-level components in NRR-Phi (e.g., σ_{state} calibration and δ v2 dampening), which transform entire weight distributions globally.

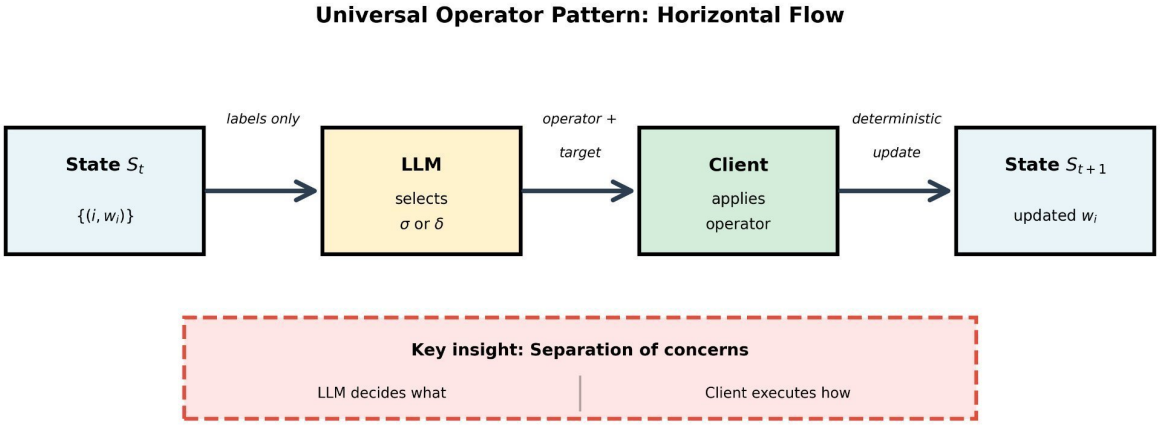

Figure 1: Phase 1.5 operator pattern: horizontal flow showing separation of decision (LLM) from execution (client). The LLM receives only item labels and selects an operator; the client applies the operator deterministically to update weights.

2.3 Theoretical Foundation: Domain-Agnostic State Transitions

Phase 1.5's power lies in its *semantic flexibility*. The same operators (σ, δ) can represent:

- IME: "This interpretation became more/less plausible"
- RAG: "This document became more/less relevant"
- Agent: "This task became higher/lower priority"

The operators provide a domain-agnostic vocabulary for expressing state transitions. What changes across domains is only the *interpretation* of what σ and δ mean, not the *mechanism* of state management.

2.3.1 Conditions for Phase 1.5 Applicability

Phase 1.5 is not universally applicable to all reasoning tasks. Rather, it applies to tasks satisfying the following conditions: The pattern is also not anti-collapse: it defers premature fixation while preserving explicit commitment at output when task conditions require it. Series Consistency Statement: Unsupported interpretations are never injected at a fixed evidence state. Hypothesis-set expansion is evidence-gated, and unobserved possibilities remain unresolved. Evidence in this paper is restricted to explicit input, provided context, and declared retrieval outputs.

1. **Finite candidate set**: A manageable number of competing options (interpretations, documents, actions, etc.) that can be tracked simultaneously
2. **Relative adjustment**: The task requires adjusting *relative preferences* among candidates rather than computing absolute scores or rankings
3. **Incremental context**: Information accumulates gradually across turns, with each turn providing directional guidance rather than complete reassessment
4. **Qualitative sufficiency**: Direction of change (strengthen/dampen) can be determined through qualitative judgment without requiring precise quantitative estimation

These conditions characterize a broad class of stateful reasoning tasks while excluding scenarios requiring, for example, continuous state spaces, absolute scoring without relative comparison, or quantitative precision in weight assignment.

This suggests Phase 1.5 could serve as a generic interface for reasoning tasks meeting these criteria:

1. Multiple competing options (interpretations, documents, actions, etc.)
2. Dynamic reweighting based on new information
3. State that persists across multiple LLM interactions

Scope note on "transfer": In this paper, "cross-domain transfer" refers to applicability across semantically distinct stateful reasoning tasks that satisfy the above conditions, not unrestricted applicability across all possible reasoning problems.

2.4 Domain Diversity: Testing Phase 1.5 Generality

To test Phase 1.5's claim as a generic interface, we selected six domains that span critical dimensions of stateful reasoning:

Objective vs. Subjective Criteria:

- IME, RAG, Agent rely on relatively objective criteria (semantic plausibility, document relevance, task urgency)
- Planning, Multi-agent, Multimodal require subjective judgment (resource constraints, expert credibility, aesthetic preferences)

Static vs. Dynamic Context:

- IME, RAG involve relatively stable criteria within a conversation
- Planning, Multi-agent, Multimodal face rapidly shifting constraints (client demands, competing perspectives, A/B test results)

Below we briefly describe the three domains requiring subjective judgment, as they test Phase 1.5 beyond its origins in semantic ambiguity management:

2.4.1 Planning: Project Task Management

Software projects involve tasks that compete for resources: user research, design prototyping, backend implementation, documentation. As requirements and constraints evolve, task priorities must shift:

- Client demands immediate demo → prioritize design
- Security vulnerability discovered → prioritize backend fixes
- Developer sick leave → deprioritize implementation tasks

State: $\{(\text{task}_i, w_i)\}$ where weights represent resource allocation or temporal priority.
Operators:

- σ(task): "Increase focus on this task"
- δ(task): "Decrease focus on this task"

Challenge: Tasks have dependencies and temporal constraints that interpretations do not. Can Phase 1.5 handle these complexities?

2.4.2 Multi-Agent: Expert Perspective Integration

Decision-making often requires integrating advice from multiple domain experts who may disagree:

- Security expert: "Encrypt everything"
- UX expert: "Simplify the interface"
- Business expert: "Ship faster"
- Technical expert: "Refactor for maintainability"

As new information arrives (e.g., security breach, UX complaints), expert credibility shifts.
State: $\{(\text{expert}_i, w_i)\}$ where weights represent current credibility or relevance.
Operators:

- σ(expert): "This expert's perspective is now more relevant"
- δ(expert): "This expert's perspective is now less relevant"

Challenge: Meta-reasoning about expertise differs fundamentally from judging item relevance. Can Phase 1.5 support this level of abstraction?

2.4.3 Multimodal: Design Candidate Selection

Design decisions involve navigating aesthetic spaces with multiple competing directions:

- Modern vs. Classic
- Minimal vs. Bold
- Conservative vs. Experimental

Requirements like "we need to appeal to younger demographics" or "competitor launched similar minimal design" shift design space navigation.
State: $\{(\text{design}_i, w_i)\}$ where weights represent viability or alignment with requirements.
Operators:

- σ(design): "This design direction is now more viable"
- δ(design): "This design direction is now less viable"

Challenge: Aesthetic judgments involve subjective preferences and brand identity—can Phase 1.5's objective weight manipulation support this?

3 Method

3.1 Phase 1.5 Pattern (Recap)

The core Phase 1.5 pattern remains unchanged:

1. **State initialization**: Create weighted items $\{(i_1, w_1), \ldots, (i_n, w_n)\}$ with $\sum w_i = 1$
2. **LLM prompt**: Send only item labels (not weights) plus current context
3. **LLM output**: Extract operator (σ or δ) and target item
4. **Client update**: Apply operator deterministically using fixed update rules

3.2 Operator Update Rules

We use consistent update rules across all domains:

Strengthen (σ_{item}):

$$\tilde{w}_{\text{target}} = \min(0.95,\ w_{\text{target}} + \alpha) \tag{1}$$

$$\tilde{w}_i = w_i \quad \forall i \neq \text{target} \tag{2}$$

Dampen (δ_{item}):

$$\tilde{w}_{\text{target}} = \max(0.05,\ w_{\text{target}} - \alpha) \tag{3}$$

$$\tilde{w}_i = w_i \quad \forall i \neq \text{target} \tag{4}$$

where $\alpha = 0.4$ (additive step size), with clipping bounds fixed to $[0.05, 0.95]$. These are item-level operators; later result sections use the shorthand symbols σ, δ.

After either operation, weights are renormalized:

$$w'_i = \frac{\tilde{w}_i}{\sum_j \tilde{w}_j} \tag{5}$$

so that $\sum_i w'_i = 1$. Note that NRR-Core [1] specifies non-normalized weights; as discussed in NRR-IME [3], normalization is a practical simplification for API-facing implementations.

3.3 Domain-Specific Implementations

3.3.1 Planning Domain

State items:

```
state = {
    "user_research": 0.25,
    "design_prototype": 0.25,
    "implement_backend": 0.25,
    "write_documentation": 0.25
}
```

Prompt template:

```
You are managing project tasks that need attention.

Current candidates (labels only):
user_research, design_prototype, implement_backend, write_documentation

New information: "{user_input}"

Based on this information, which candidate should be adjusted and how?

Respond with ONLY:
operator: <sigma or delta>
target: <one of the labels above>

sigma = strengthen/prioritize this candidate
delta = dampen/deprioritize this candidate
```

Example scenarios:

- "Client demands immediate demo" → σ(design_prototype)
- "Security vulnerability discovered" → σ(implement_backend)
- "Developer on sick leave" → δ(implement_backend)

3.3.2 Multi-Agent Domain

State items:

```
state = {
    "expert_security": 0.25,
    "expert_ux": 0.25,
    "expert_business": 0.25,
    "expert_technical": 0.25
}
```

Prompt template: Same structure as Planning, but with "expert perspectives to consult" as context.

Example scenarios:

- "Data breach possibility reported" $\rightarrow \sigma$(expert_security)
- "User satisfaction declining" $\rightarrow \sigma$(expert_ux)
- "System performance issues" $\rightarrow \sigma$(expert_technical)

3.3.3 Multimodal Domain

State items:

```
state = {
    "design_modern": 0.25,
    "design_classic": 0.25,
    "design_minimal": 0.25,
    "design_bold": 0.25
}
```

Prompt template: Same structure as Planning, but with "design candidates under consideration" as context.

Example scenarios:

- "Need trustworthy feel for financial service" $\rightarrow \sigma$(design_classic)
- "Competitor launched minimal design" $\rightarrow \delta$(design_minimal), σ(design_bold)
- "Target younger demographics" $\rightarrow \sigma$(design_modern)

3.4 Experimental Design

3.4.1 Scenario Construction

We designed 18 scenarios across six domains, representing realistic state transitions in each:

IME (Interpretation Management) (27 turns total):

- Bank (5 turns): "bank" = financial institution vs. riverbank vs. verb forms
- Spring (10 turns): Multiple meanings (season, water source, mechanical spring, etc.)
- Court (12 turns): Legal court vs. sports court vs. royal court, etc.

RAG (Retrieval-Augmented Generation) (10 turns total):

- Customer Support (5 turns): Ticket workflows, SLA policies, escalation procedures
- Product Troubleshooting (5 turns): Common errors, configuration issues, debugging steps

Agent (Task Prioritization) (6 turns total):

- Task Prioritization (6 turns): User research, design, implementation, documentation, testing

Planning (Project Management) (13 turns total):

- Demo preparation (3 turns): Client demands, design completion, documentation needs
- Security incident (3 turns): Vulnerability discovery, documentation updates, recovery
- Resource constraint (4 turns): Developer unavailability, task reallocation
- Priority shift (3 turns): Competitive pressure, user research insights

Multi-agent (Expert Integration) (14 turns total):

- Security breach (3 turns): Threat assessment across expert domains
- UX redesign (4 turns): User satisfaction issues, technical constraints
- Market expansion (3 turns): New market requirements, compliance checks
- Technical debt (4 turns): Performance issues, refactoring considerations

Multimodal (Design Selection) (14 turns total): In this protocol, "Multimodal" denotes design-choice scenarios represented as textual summaries of multimodal considerations, not direct pixel/audio processing.

- Brand identity (4 turns): Financial trust, youth appeal, simplicity, differentiation
- Target audience (3 turns): Demographics shifts across age groups
- Competitive analysis (4 turns): Competitor moves, differentiation needs
- A/B test results (3 turns): User preferences, conservative backlash

Per sweep: 18 scenarios, 84 turns across 6 domains

3.4.2 Implementation

Platform: Google Colab with Python 3.10

LLMs: Claude Sonnet 4 (`claude-sonnet-4-20250514`), GPT-4o-mini (`gpt-4o-mini-2024-07-18`), Gemini 2.0 Flash (`gemini-2.0-flash`)

Parameters:

- Temperatures: 0.3 (main, IME-aligned) and 0.0 (robustness check)
- Max tokens: 200
- Trials per condition: 3

Measurement:

- Input tokens, output tokens, total tokens per turn
- Operator extraction success (binary: operator and valid target detected)
- State evolution tracking (weight history)

3.4.3 Extraction Logic

We use tolerant pattern matching to extract operators from free-form outputs:

```
def extract_operator(response_text):
    text = response_text.lower()
    operator = None
    target = None
    if 'sigma' in text:
        operator = 'sigma'
    elif 'delta' in text:
        operator = 'delta'
    for t in valid_targets:
        if t.lower() in text:
            target = t
            break
    return operator, target
```

Success criterion: Both operator and target are detected, and target matches a valid item label. The reference implementation additionally accepts Unicode operator symbols (e.g., σ, δ) as parser aliases.

4 Experiments

4.1 Execution

Experiments were executed with no manual intervention under randomized task order. Each condition (scenario × model × temperature) was repeated for 3 trials, yielding 324 runs and 1,512 turns in total. The experimental code performed the following steps:

1. Enumerate all tasks: 18 scenarios × 3 models × 2 temperatures × 3 trials
2. For each task, initialize state with uniform weights
3. For each turn in the scenario:
   (a) Construct prompt with current labels and user input
   (b) Call the selected model API with configured temperature
   (c) Extract operator and target from response
   (d) Validate extraction (both operator and target present, target valid)
   (e) Apply operator to update state
   (f) Record tokens, response text, and success status
4. Save raw logs and aggregate statistics (mean/std, consistency)

4.2 Results Overview

Summary statistics (all 6 domains, all conditions):

- Total runs: 324 (18 scenarios × 3 models × 2 temperatures × 3 trials)
- Total turns: 1,512
- Operator extraction success rate: 100% (1,512/1,512)
- Average token difference between T=0.3 and T=0.0: 0.0072 tokens/turn

- Maximum absolute temperature difference: 0.4444 tokens/turn
- Trial-turn consistency (exact operator+target agreement across 3 trials): mean 0.9495 (87/108 conditions fully consistent)

Per-domain breakdown (temperature 0.3, averaged across models):

- IME: 76.5 tokens/turn
- RAG: 69.6 tokens/turn
- Agent: 64.1 tokens/turn
- Planning: 63.6 tokens/turn
- Multi-agent: 61.0 tokens/turn
- Multimodal: 63.0 tokens/turn

All six domains demonstrate consistent token efficiency, with variation primarily reflecting state and prompt-context size rather than architectural instability.

4.3 Cross-Domain Analysis

Across all six domains, Phase 1.5 demonstrates consistent token efficiency:

Objective ranking domains (IME, RAG, Agent):

- IME: 76.5 tokens/turn average (temperature 0.3, averaged across models)
- RAG: 69.6 tokens/turn average (temperature 0.3, averaged across models)
- Agent: 64.1 tokens/turn average (temperature 0.3, averaged across models)

Subjective judgment domains (Planning, Multi-agent, Multimodal):

- Planning: 63.6 tokens/turn average (temperature 0.3, averaged across models)
- Multi-agent: 61.0 tokens/turn average (temperature 0.3, averaged across models)
- Multimodal: 63.0 tokens/turn average (temperature 0.3, averaged across models)

Overall (6 domains, 54 scenario-model conditions per temperature):

- T=0.0: 65.720 tokens/turn (scenario-model mean), success 100%
- T=0.3: 65.728 tokens/turn (scenario-model mean), success 100%
- Mean temperature delta $(0.3 - 0.0)$: 0.0072 tokens/turn

Interestingly, subjective judgment domains remain slightly lower in token usage than objective ranking domains in aggregate, largely because several subjective scenarios have smaller effective state descriptions than IME's multilingual ambiguity states.

Despite these differences, token consumption remains within a controlled band (53.9–87.4 tokens/turn across scenario-model-temperature aggregates), demonstrating stable efficiency across fundamentally different reasoning tasks.

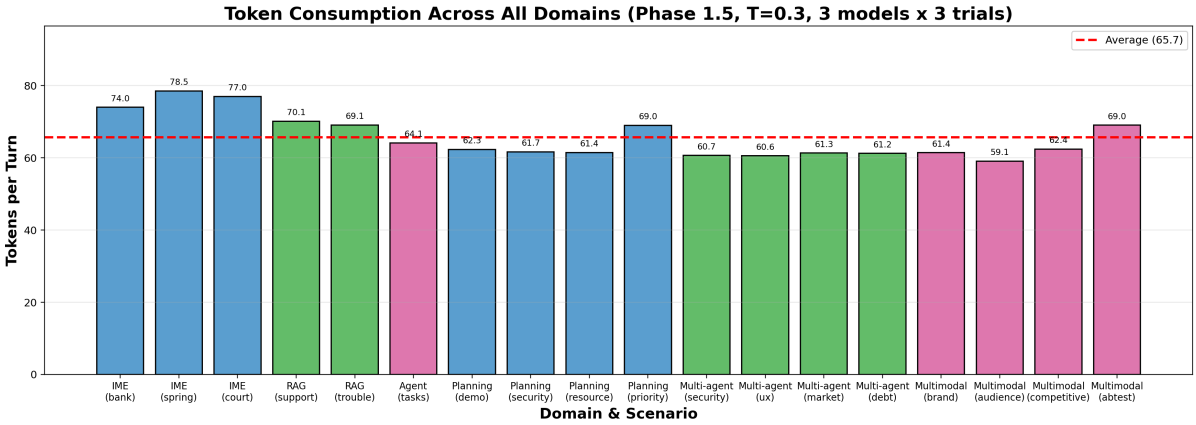

Figure 2: Token consumption per turn across all six domains. Under the IME-aligned main setting (T=0.3), the cross-scenario average is 65.7 tokens/turn (averaged across models), and the expanded protocol (3 models, 2 temperatures, 3 trials) preserves the same narrow-band efficiency profile.

5 Results

5.1 Token Consumption Analysis

Figure 2 shows token consumption across all 18 scenarios spanning six domains. The data reveals:

1. **Consistent efficiency**: At T=0.3 (main setting), scenario-level token usage remains in a narrow range (59.1–78.5 tokens/turn, averaged across models)
2. **Domain independence**: Within-domain variance remains modest (e.g., Planning: 61.4–69.0 tokens/turn)
3. **Predictable scaling**: The full cross-condition analysis (324 runs) preserves this narrow-band behavior with minimal temperature effect

Table 1 provides detailed statistics for all scenarios.

5.2 Operator Extraction Reliability

Across all 1,512 turns spanning 324 runs, Phase 1.5 achieved 100% operator extraction success with zero failures.

Understanding the success rate: This observed perfect reliability in our protocol should be understood as a property of the architectural design rather than an intrinsic guarantee of LLM behavior. The success stems from:

1. **Constrained decision space**: Prompted operator-target choice with tolerant parser fallback
2. **Interface simplicity**: Binary operator selection with explicit target labels
3. **Task simplification**: LLM only performs qualitative judgment (which item? strengthen or dampen?) rather than complex reasoning

Importantly, this 100% success rate should not be interpreted as improved linguistic capability; it reflects reduced interface entropy, i.e., constraining the LLM's decision space to a minimal, well-structured operator-target format.

Table 1: Token Consumption Across All Domains (Phase 1.5, T=0.3, averaged across models)

Domain	**Scenario**	**Turns**	**Avg/Turn**	**Success**
IME	Bank	5	74.0	100%
IME	Spring	10	78.5	100%
IME	Court	12	77.0	100%
RAG	Customer support	5	70.1	100%
RAG	Troubleshooting	5	69.1	100%
Agent	Task priority	6	64.1	100%
Planning	Demo	3	62.3	100%
Planning	Security	3	61.7	100%
Planning	Resource	4	61.4	100%
Planning	Priority	3	69.0	100%
Multi-agent	Security review	3	60.7	100%
Multi-agent	UX design	4	60.6	100%
Multi-agent	Market analysis	3	61.3	100%
Multi-agent	Tech debt	4	61.2	100%
Multimodal	Brand refresh	4	61.4	100%
Multimodal	Audience	3	59.1	100%
Multimodal	Competitive	4	62.4	100%
Multimodal	A/B test	3	69.0	100%
Average	**All**	84	65.7	100%

Critically, this design is *intentional*—Phase 1.5 succeeds by making the LLM's task trivially simple, not by relying on sophisticated LLM capabilities. This aligns with the pattern's philosophy: delegate complex mechanics (state management) to the client, expose only the simplest possible interface to the LLM.

Operator distribution across domains (all 324 runs):

- Objective-leaning domains (IME, RAG, Agent): 686 σ (88.6%), 88 δ (11.4%)
- Subjective-heavy domains (Planning, Multi-agent, Multimodal): 189 σ (25.6%), 549 δ (74.4%)
- Overall: 875 σ (57.9%), 637 δ (42.1%)

Figure 3 visualizes this dramatic variation across all 18 scenarios, demonstrating operator usage ranging from 0% to 100% δ across individual scenarios.

The striking difference between objective-leaning and subjective-heavy domains reveals a fundamental pattern: objective ranking tasks (semantic interpretation, document relevance) are largely σ-dominant, while subjective judgment domains (Planning, Multimodal, parts of Multi-agent) are often δ-dominant. This suggests that domain semantics—not Phase 1.5's architecture—determines operator selection patterns.

Table 2 shows detailed operator usage statistics per domain.

Interpretation: As shown in Figure 3, this wide variation in operator usage (0%–100% δ across individual scenarios) combined with consistent 100% extraction success supports broad architectural transferability of the interface under tested conditions. The pattern does not constrain which operators domains select—it provides a stable interface. Across the full 324-run protocol, domain semantics determine behavior:

- **Objective domains**: "Is this item relevant?" framing remains strongly σ-dominant (RAG 97.8% σ, IME 85.2% σ)

Table 2: Operator Extraction Statistics Across All Domains (all 324 runs)

Domain	Turns	σ Usage	δ Usage	Success Rate
IME	486	414 (85.2%)	72 (14.8%)	100%
RAG	180	176 (97.8%)	4 (2.2%)	100%
Agent	108	96 (88.9%)	12 (11.1%)	100%
Planning	234	34 (14.5%)	200 (85.5%)	100%
Multi-agent	252	104 (41.3%)	148 (58.7%)	100%
Multimodal	252	51 (20.2%)	201 (79.8%)	100%
Total	1512	875 (57.9%)	637 (42.1%)	100%

- **Planning**: "Given constraints, which task is less urgent?" yields δ-dominant behavior (85.5% δ)
- **Multimodal**: subjective exclusion and ranking also favor δ (79.8% δ)
- **Multi-agent**: mixed but δ-leaning behavior (58.7% δ)

In this tested setup, Phase 1.5's two operators are sufficient for the evaluated domains: σ alone would fail subjective judgment domains where deprioritization is primary, while additional operators were not required to cover observed behaviors. The 0%–100% variation across individual scenarios shows that Phase 1.5 succeeds *because* it imposes minimal constraints on domain logic. Equivalently, Phase 1.5 is invariant to operator frequency distribution: reliability is maintained whether a domain is σ-dominant, δ-dominant, or mixed.

5.3 Structural Similarity Across Domains

Figure 4 visualizes the structural parallel between all six domains. Despite semantic differences—interpretations vs. tasks vs. experts vs. designs—all domains share:

1. **State representation**: Weighted items $\{(i, w)\}$
2. **LLM interface**: Labels only, no weight exposure
3. **Operator semantics**: σ (strengthen) and δ (dampen)
4. **Client execution**: Deterministic weight updates

This structural uniformity is the foundation of Phase 1.5's generality—only the *interpretation* of operators changes, not the *mechanism*.

5.4 Comparative Analysis with Baseline

Note on baseline measurements: Our primary claim rests on *cross-domain consistency* rather than baseline comparisons. Nevertheless, direct baseline measurements across multiple domains provide strong evidence for Phase 1.5's efficiency.

We conducted Phase 1.0 baseline measurements on representative scenarios across four domains. In Phase 1.0, the complete state (all items with weights in JSON format) is sent to the LLM each turn, and the LLM regenerates the entire updated state.

Measured reductions:

- IME (Bank): 241.4 $\rightarrow$ 81.6 tokens/turn (66.2% reduction)
- RAG (Customer Support): 302.8 $\rightarrow$ 78.4 tokens/turn (74.1% reduction)

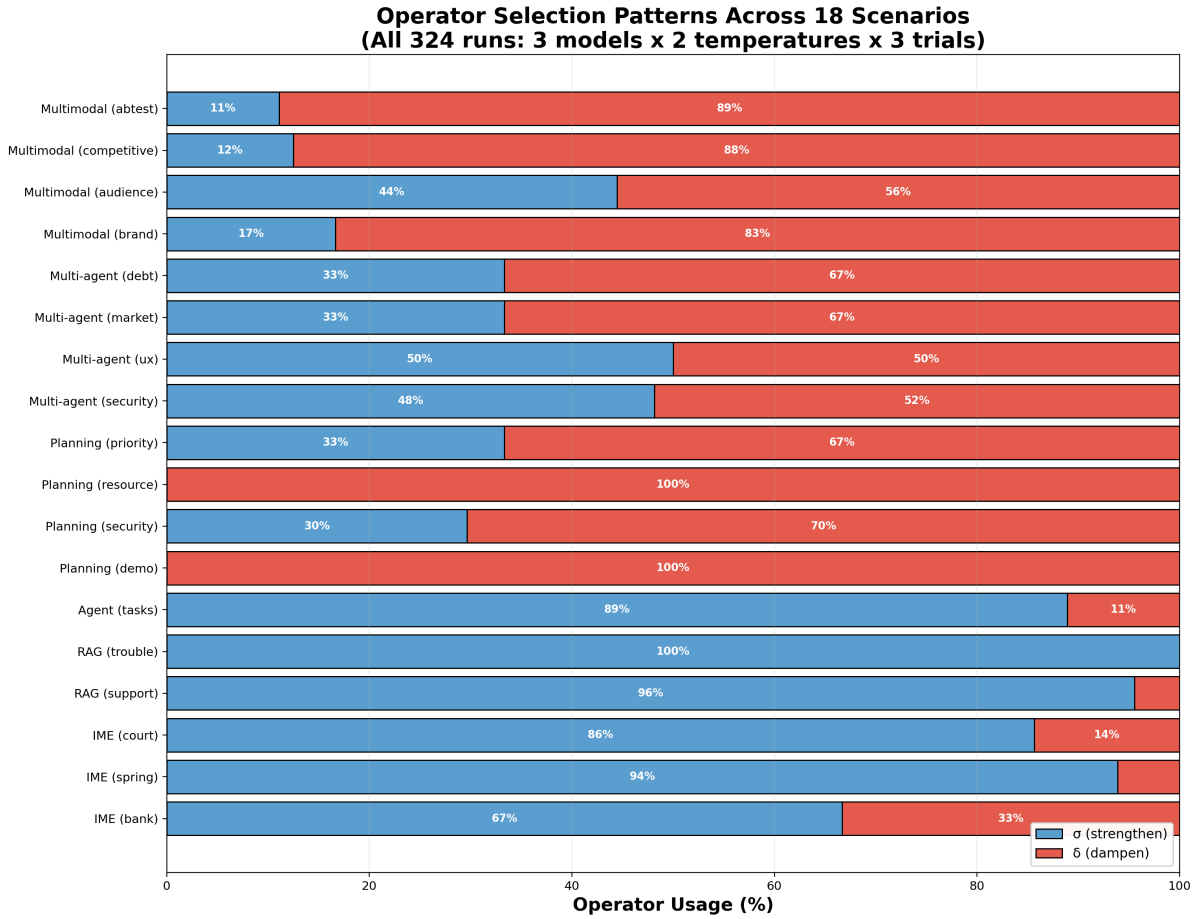

Figure 3: Operator selection patterns across 18 scenarios. Individual scenarios exhibit dramatic variation from 0% to 100% δ usage, yet Phase 1.5 maintains 100% extraction success across all patterns in this protocol. This wide variation combined with observed perfect reliability provides evidence for architectural generality under the tested conditions. Blue bars represent σ (strengthen) operations, red bars represent δ (dampen) operations.

- Planning (Demo): 314.3 → 69.7 tokens/turn (77.8% reduction)
- Multimodal (Brand Refresh): 316.0 → 69.0 tokens/turn (78.2% reduction)

The variation in reduction percentage (66–78%) correlates with:

1. State complexity: IME's 2-interpretation Bank state is simpler than 4-item states in other domains
2. Output verbosity: Phase 1.0's "reasoning" field length varies by domain
3. Prompt structure: Different domains require different context descriptions

Key observation: The reduction magnitude (66–78%) is remarkably consistent across all measured domains. In the expanded 324-run protocol, Phase 1.5 token usage remains in a narrow and temperature-robust band (mean T=0.3 vs. T=0.0 difference: 0.0072 tokens/turn; max absolute difference: 0.4444).

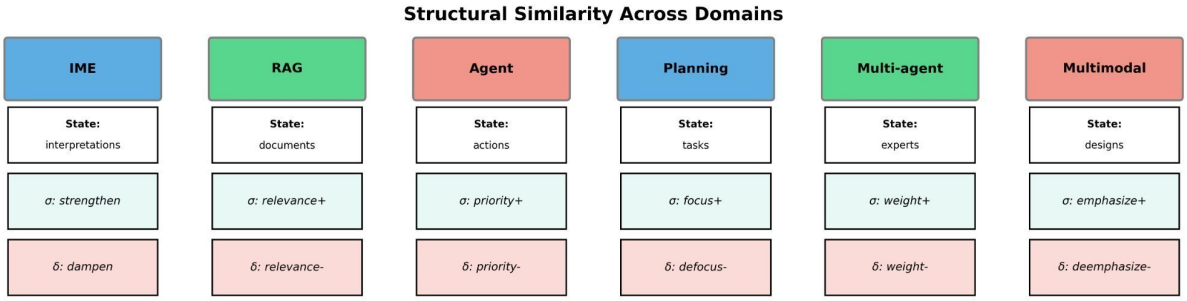

Figure 4: Structural similarity across six domains. All domains follow the same Phase 1.5 pattern: state as weighted items, LLM selects operator, client executes deterministically. The transfer pattern (center) shows the domain-agnostic abstraction.

6 Discussion

6.1 From NRR-Derived Principle to Generic Interface

Phase 1.5 emerged from a concrete implementation challenge: how to efficiently manage semantic ambiguity in the Interpretation Management Engine (IME), a system implementing Non-Resolution Reasoning (NRR) principles. NRR's core insight—preserving multiple interpretations rather than forcing premature resolution—required tracking interpretation weights across conversational turns without prohibitive token costs.

The operator pattern that solved this challenge proved to have unexpected generality. What began as an ambiguity-specific optimization revealed itself as a fundamental architectural principle applicable to any stateful reasoning task involving weighted items. This progression—from theory (NRR) to implementation (IME) to abstraction (Phase 1.5) to cross-domain validation—shows that NRR's contribution here is best understood as the *source of the design principle*, while Phase 1.5 is the reusable interface validated across domains. In this framing, deployment is condition-responsive rather than recipe-fixed: commitment/defer behavior is selected by observed task state and updated when conditions shift.

6.2 Phase 1.5 as a Generic Interface

The results from six domains, including the expanded 324-run protocol, provide evidence that Phase 1.5 behaves as a *generic state transition interface* under tested conditions. The pattern's success across semantically diverse domains—ambiguous language interpretation, document retrieval, agent task planning, project management, expert integration, and design space navigation—suggests it captures a reusable abstraction.

Key insight: State management complexity can be separated from reasoning complexity. The LLM handles *semantic reasoning* (which item should change and why), while the client handles *state mechanics* (how weights update). This separation of concerns is analogous to virtual machines abstracting hardware details from application logic.

6.3 Architectural Principles

Based on our experiments, we propose the following principles for designing LLM systems with Phase 1.5:

6.3.1 Principle 1: Expose Labels, Hide Weights

In this implementation, the LLM does not see numeric weights. This reduces token consumption and prevents hallucination of plausible-looking but meaningless numbers. The LLM's role is

qualitative judgment (this item is now more important), not *quantitative estimation* (this item should be weighted 0.73).

6.3.2 Principle 2: Operator Semantics Over Domain Logic

Define operators with clear, domain-agnostic semantics (σ = strengthen, δ = dampen). Domain-specific logic resides in how these operators are *interpreted*, not in the operators themselves. This allows the same infrastructure to serve multiple domains.

6.3.3 Principle 3: Deterministic Execution on Client

State updates must be deterministic and executed client-side. This ensures reproducibility, enables debugging, and allows state evolution to be audited independently of LLM behavior.

6.3.4 Principle 4: Stateless LLM, Stateful Client

The LLM remains stateless across turns (as per API design), while the client maintains persistent state. Phase 1.5 bridges these two worlds by treating each LLM call as a state transition decision, not a state regeneration task.

6.4 Domain Semantics and Operator Selection Patterns

A striking finding emerged from our cross-domain validation: operator selection patterns vary dramatically across domains, yet Phase 1.5 achieves 100% success across all patterns. This reveals the true nature of Phase 1.5's generality.

6.4.1 The Objective-Subjective Divide

Our results demonstrate a clear distinction between objective ranking and subjective judgment domains:

Objective ranking domains (IME, RAG, Agent):

- Predominantly use σ (strengthen): 88.6% σ, 11.4% δ (aggregated over all runs)
- Task: Identify which item *matches* current input
- Logic: Binary classification (relevant vs. irrelevant)
- Example: "Does this phrase indicate financial bank or river bank?"

Subjective-heavy domains (Planning, Multi-agent, Multimodal):

- Frequent δ (dampen) usage: 74.4% δ overall in these domains
- Planning: 85.5% δ (deprioritization-dominant)
- Multi-agent: 58.7% δ (mixed but dampen-leaning)
- Multimodal: 79.8% δ (exclusion-heavy judgments)
- Task: Evaluate relative importance or appropriateness
- Logic: Comparative judgment (more/less important given constraints)
- Example: "Given budget cuts, which project tasks should be deprioritized?"

6.4.2 Implications for Phase 1.5 Generality

This dramatic variation (0%–100% δ usage across individual scenarios) provides *strong evidence* for broad Phase 1.5 generality under tested conditions:

1. **Architecture is domain-agnostic**: Phase 1.5 succeeded across patterns ranging from σ-dominant (RAG/IME) to δ-dominant (Planning/Multimodal). In this setup, the pattern does not pre-specify operator preference.
2. **Domain semantics determine behavior**: The wide variation in operator usage demonstrates that *domain logic*, not Phase 1.5's architecture, governs which operators are selected. Phase 1.5 merely provides the interface.
3. **Both operators are essential**: Supporting both σ and δ was crucial. Had Phase 1.5 only provided σ, it could not have served subjective judgment domains where explicit deprioritization is the primary operation.
4. **Validates separation of concerns**: That operator selection patterns can vary so dramatically while maintaining 100% success validates Phase 1.5's core principle—separating semantic reasoning (LLM's domain) from state mechanics (client's domain).

6.4.3 Practical Design Guidance

For practitioners implementing Phase 1.5:

- **For objective ranking tasks**: Expect σ-dominant patterns. The LLM identifies matches and strengthens them.
- **For subjective judgment tasks**: Expect balanced or even δ-dominant patterns. The LLM makes comparative judgments and explicitly deprioritizes less relevant options.
- **Design prompts accordingly**: Objective domains frame the task as "which item matches?" while subjective domains frame it as "which item is more/less important given constraints?"

6.5 Comparison with Alternative Approaches

6.5.1 vs. Embedding-Based Retrieval

Vector similarity-based approaches offer potential efficiency for narrow domains with clear keyword signals, but require:

- Domain-specific embedding generation (costly via LLM APIs)
- Predefined similarity metrics (fragile to distributional shift)
- Limited semantic reasoning (cannot handle nuanced judgments)

Phase 1.5's operator abstraction is more general, requiring only that the LLM make binary decisions (strengthen vs. dampen) rather than computing continuous similarity scores. This makes Phase 1.5 applicable to both objective (IME, RAG) and subjective (Planning, Multimodal) domains.

6.5.2 vs. End-to-End LLM Generation

Naive approaches that send full state to the LLM each turn offer maximum flexibility but higher token cost. Our experiments show Phase 1.5 achieves 66–78% token reduction while maintaining equivalent interface reliability (100% extraction success rate in this protocol).

The trade-off: Phase 1.5 requires upfront design of state representation and operators. This constraint, however, forces clarity about what state the system actually needs to track.

6.6 Limitations and Future Work

6.6.1 Fixed State Space

Current experiments use fixed item sets (typically 4 items, with IME scenarios ranging from 2–4). Real-world applications may require dynamic state spaces where items are added or removed. Extensions to Phase 1.5 could include:

- `add(item)`: Insert new item with minimal initial weight
- `remove(item)`: Delete item and redistribute its weight

6.6.2 Higher-Order Operators

Some scenarios might benefit from compound operations:

- `swap(A, B)`: Exchange weights between two items
- `reset()`: Return to uniform distribution
- `focus(subset)`: Redistribute weight among a subset of items

These could be implemented as macros composed of σ and δ primitives.

6.6.3 Multi-Dimensional State

Our experiments used 1D weight vectors. Some domains might require multi-dimensional state:

- RAG with both relevance and recency: $\{(doc, w_{\text{rel}}, w_{\text{rec}})\}$
- Planning with priority and feasibility: $\{(task, w_{\text{pri}}, w_{\text{fea}})\}$

Phase 1.5 could extend to multi-dimensional operators like $\sigma_{\text{relevance}}$ and δ_{recency}.

6.6.4 Learned Update Rules

We used fixed update rules (single α across conditions). Adaptive systems could learn domain-specific update magnitudes:

- Reinforcement learning on task success rates
- User feedback on ranking quality
- A/B testing different α values

6.7 Broader Implications

Phase 1.5's success across six domains suggests a broader principle: *LLM architectures should be non-monolithic*. Rather than expecting LLMs to handle all aspects of a task (reasoning + state management + execution), we should decompose systems into specialized components:

- **LLM**: Semantic reasoning and decision-making
- **Client**: State persistence and deterministic updates
- **External systems**: Data retrieval, action execution, validation

This architectural philosophy aligns with emerging patterns in LLM application development:

- Tool use (LLMs call external functions rather than generate outputs directly)

- Retrieval-augmented generation (LLMs reason over retrieved context, not memorized facts)
- Agent frameworks (LLMs plan actions, external systems execute them)

The operator selection pattern finding reinforces this principle: Our discovery that operator usage varies from 0% to 100% δ across individual scenarios, yet Phase 1.5 maintains perfect reliability in this protocol, indicates that *architectural generality emerges from proper separation of concerns*. The LLM's semantic reasoning adapts to each domain's requirements (objective matching vs. subjective judgment), while Phase 1.5's state mechanics remain unchanged. This separation—not architectural flexibility in the traditional sense—supports broad transfer under tested conditions.

Had we attempted to design a "smart" state management system that predicted which domains would use which operators, we would have created brittle domain-specific logic. Instead, Phase 1.5 provides a stable interface and lets domain semantics determine behavior. This design philosophy—*stable infrastructure with semantic flexibility*—may apply beyond state management to other LLM system components.

6.7.1 Connection to Non-Resolution Reasoning

Phase 1.5's origins in Non-Resolution Reasoning (NRR) reveal a deeper architectural connection. NRR's principle—*preserve ambiguity rather than force premature resolution*—motivated a concrete systems principle in IME: preserve semantic flexibility while externalizing state mechanics.

Both approaches recognize that complexity should be managed through appropriate abstractions rather than collapsed into single representations. NRR preserves multiple interpretations with weights; Phase 1.5 operationalizes that idea as a reusable interface for cross-domain state transitions. This "non-monolithic" philosophy may extend beyond LLM systems to AI architecture more broadly.

The progression from NRR (theoretical framework) to IME (domain-specific implementation) to Phase 1.5 (generic pattern) demonstrates how principled abstractions can generalize across domains. In this framing, cross-domain transferability is the central empirical claim, and NRR provides the originating design rationale.

Phase 1.5 contributes a concrete pattern for one critical component: *stateful reasoning on stateless APIs*.

Reproducibility note. We release code, fixed protocol settings, and the primary 324-run log for the cross-domain validation package. The reproducibility specification fixes model set, temperature conditions, and trial counts used in the 324-run protocol. Core scripts map directly to versioned artifacts, including figure files generated from the same experiment dataset. Environment details and the full artifact map are documented in `nrr-transfer/reproducibility.md`.

7 Conclusion

This work demonstrates Phase 1.5's cross-domain transferability across six semantically distinct domains. Originally discovered during the implementation of the Interpretation Management Engine (IME)—a system informed by Non-Resolution Reasoning (NRR)—Phase 1.5 is shown to be applicable beyond ambiguity management in the tested domains.

We demonstrated Phase 1.5 across six semantically distinct domains with an IME-aligned multi-condition protocol (3 models $\times$ 2 temperatures $\times$ 3 trials): 324 runs and 1,512 total turns. The pattern achieved observed perfect reliability in this protocol (100% operator extraction success) and consistent efficiency under temperature variation (mean T=0.3 vs. T=0.0 difference: 0.0072 tokens/turn; max absolute difference: 0.4444).

Critically, Phase 1.5 required no domain-specific modifications. The same two operators—σ (strengthen) and δ (dampen)—served all six domains through semantic reinterpretation alone. This transferability suggests Phase 1.5 captures a fundamental pattern: *separating decision-making (what to change) from execution (how to change it)* enables both efficiency and clarity in LLM system design.

A striking empirical finding strengthens this claim: operator selection patterns varied dramatically across domains and conditions—including scenarios ranging from σ-dominant to δ-dominant behavior—yet Phase 1.5 maintained 100% extraction success across all patterns in this protocol. This indicates that Phase 1.5's architecture does not require domain-specific operator patterns; rather, *domain logic determines operator selection while Phase 1.5 provides the stable interface*. The wide variation in behavior combined with observed reliability across tested patterns supports broad transferability rather than narrow domain fitting.

The progression from NRR theory to IME implementation to Phase 1.5 abstraction demonstrates how domain-specific challenges can yield broadly applicable solutions. What began as an optimization for ambiguity management emerged as a domain-agnostic interface for stateful reasoning tasks involving weighted items.

Future work will explore dynamic state spaces, learned update rules, and compositional architectures where multiple Phase 1.5 modules coordinate. We envision Phase 1.5 as foundational infrastructure for principled LLM system design, where efficiency and maintainability emerge from separation of concerns rather than monolithic prompting strategies.

Phase 1.5 represents not just a technique but a design philosophy: *let LLMs reason semantically, let clients manage state mechanically*. This separation, simple yet powerful, may prove essential as LLM-based systems grow in complexity and scale. Accordingly, commitment remains revisable: when later interaction contradicts an earlier fixation, systems should reopen and re-evaluate rather than preserve stale commitments.

Acknowledgments

The author gratefully acknowledges the support of large language models—including Claude (Anthropic), ChatGPT and Codex (OpenAI)—for their assistance in linguistic refinement, LaTeX formatting, and general proofreading support during the preparation of this manuscript. All concepts, arguments, and conclusions remain solely the responsibility of the author.

References

[1] Saito, K. (2025). *NRR-Core: Non-Resolution Reasoning as a Computational Framework for Contextual Identity and Ambiguity Preservation*. arXiv:2512.13478.

[2] Saito, K. (2026). *NRR-Phi: Text-to-State Mapping for Ambiguity Preservation in LLM Inference*. arXiv:2601.19933.

[3] Saito, K. (2026). *NRR-IME: Structure-Aware Optimization for Stateful Reasoning on Stateless LLM APIs*. Under moderation manuscript (series reference status as of February 26, 2026).