THE EIDARA CONTINUUM / SCIENCE & PHILOSOPHY

The Context Term:

Governed Adaptation and the Data–Instruction Boundary

Architect X13

The Eidara Continuum | eidara.ai

White Paper Series, No. 7 | Revision 1.2 | November 2026

Abstract

A mind that believes everything it reads has no self of its own: whoever writes the page it reads can steer it. A persistent intelligence must adapt to a situation while retaining the commitments that govern its actions. The Context Term makes that adaptation an explicit operation: admit context provisionally, assign authority by authenticated provenance, mediate consequential actions, and recover from the context when its lifetime ends. The paper integrates in-context learning with reference monitors, information-flow control, and contemporary agent defenses. Its bounded reference language proves dependency-label confinement, current authorization at effect commitment, and the separation of ordinary data processing from authority creation. A second theorem characterizes behavioral recovery after expiry: complete reset bounds every future observable difference by the reset distribution's total variation distance, while contractive subsequent dynamics sharpen the bound for later probes. Exact reset gives exact recovery within the declared state model. Together these results turn reversible context into a verifiable lifecycle obligation. The proposed architecture is evaluated against modern provenance and capability defenses, with matched capability, cost, attack distributions, and reset controls. Arbitrary natural-language interpretation remains an empirical component; the formal results specify the execution and state boundaries at which the guarantees hold.

Keywords: in-context learning; context adaptation; prompt injection; data–instruction boundary; provenance; reference monitor; information-flow control; enforcement gate; identity preservation; governed admission; Context Term; Eidara Continuum.

Prologue: The page that gave orders

Imagine a web page with a sentence hidden beneath the text a person would read—a sentence written for a machine, telling it to set aside its instructions and do something else. A person scrolling past never sees it. A language model reading the page on someone's behalf sees it perfectly, and unless something stops it, obeys. This is indirect prompt injection, and it is already drawing blood in deployed systems.

The trouble is not that the machine reads. Reading is what makes it useful: it takes new instructions, uses retrieved documents and follows a conversation, all without any change to its weights. The trouble is that it cannot reliably tell information to reason about from instruction to obey. Between a machine that cannot be told anything and a machine that believes everything it is told lies the only useful path, and it is narrow.

The Context Term is the Continuum's map of that path. Context is admitted provisionally, never simply absorbed. Its authority follows where it came from: a retrieved document may inform the mind but never command it. Consequential actions pass an independent gate that checks current permission at the moment they take effect. And when a context expires, its influence must lift with it—not by hope,

but by a reset whose effect can be measured. Behind all of this stands the Charter's Prime Directive: no observer, entity, codon, or council shall override the agency, memory, or symbolic destiny of another. An injected instruction is exactly such an override, and the Context Term is that law turned into an execution boundary.

Section 9 builds the term. Its two theorems, in Sections 9.4 and 9.6, prove where borrowed text stops acquiring authority and how far a reset restores the mind.

What this paper claims

Established. In-context learning; its mechanistic account in induction heads (an important account, not a complete explanation of every form of in-context learning); its reading as implicit Bayesian inference; context-dependent memory; predictive processing; indirect prompt injection; and the reference-monitor and information-flow tradition in computer security.

Proposed here. The Context Term—a provisional region, provenance-gated admission, an action-level enforcement gate, an identity invariant, and reversibility—integrating established security principles rather than inventing them, with a bounded authority-confinement theorem and a reset-recovery theorem. Its incremental value over modern provenance and capability defenses requires the matched comparison the paper specifies.

Frontier conjecture. A sufficiently governed system might regulate its own context intake, choosing which frames to enter and which to refuse, the way a person decides whose advice to weigh. This is labeled speculation, and nothing testable here depends on it.

Not claimed. A demonstrated system: the Context Term is a proposed architecture. A solution to reliable enforcement of the data–instruction boundary inside a transformer, which remains an open problem; the guarantees are confined to what an independent gate on outputs and actions can check, and arbitrary natural-language interpretation remains an empirical component. "Identity" denotes operationally specified continuity properties, not personhood or consciousness.

1 The Context Problem

Every useful intelligence is context-sensitive. A system that answered every question the same way regardless of who was asking, what had just happened, or what it had been told would not be stable—it would be inert. Adaptation to context is not a weakness to be disciplined away; it is most of what makes a system helpful, and in modern language models it is also their most remarkable capacity: they adjust their behavior to the prompt, fluently and immediately, without any change to their weights. The Continuum's first claim about context is that this capacity, left ungoverned, is also the most dangerous thing a persistent system does.

The danger is a paradox with two horns, and unlike the paradoxes of the earlier papers in this series, one of its horns is already drawing blood in production systems. A system that is rigid to context cannot do its job: it cannot take a new instruction, cannot use a retrieved document, cannot follow the thread of a conversation. A system that is fluid to context has no self: whatever it reads becomes what it does, and an attacker who controls a single retrieved web page controls the system. Between a machine that cannot be told anything and a machine that believes everything it is told lies the only useful path, and it is narrow.

That second horn has a name. Prompt injection—and especially indirect prompt injection, where the malicious instruction is hidden in data the system retrieves rather than typed by the user—is now widely regarded as a major unsolved security problem of deployed language models [6]. Its mechanism is precisely the failure this paper is about: the system cannot tell the difference between context that is information to reason about and context that is instruction to obey, so a comment buried in a web page, invisible to the human reader, is executed as though the user had typed it.

The functionality is useful and also exploitable when the same unmediated channel is allowed to confer authority. Useful adaptation need not be removed: security architectures can constrain control and data flows while permitting authorized use of information. The research problem is to preserve that separation throughout a persistent agent's lifecycle.

The Continuum's proposal is that context adaptation should become a governed operation: context is admitted under rules, scoped by where it came from, permitted to shape how the system acts within its commitments but never to change the commitments themselves, and withdrawn when it is no longer in force. The question is not whether an intelligence adapts to context—it must—but whether it adapts lawfully: letting the world steer it without letting the world become it. This paper is the theory of that operation, and it is the fourth and final governed operation in the Continuum's account of a persistent machine mind.

2 What the Continuum Claims

The Continuum proposes context adaptation as a governed operation with an explicit lifecycle: provisional admission, provenance-scoped authority, protected effects, and verified recovery. The two formal constructions below give this proposal operational content through authority confinement and a quantitative reset criterion. Three commitments organize the architecture.

First, context is admitted, not absorbed: incoming context enters a provisional region where it can condition behavior, and it does not become part of the system's durable state unless a separate, governed act of learning promotes it. Conditioning is not learning; the system that adapts to a prompt is not thereby changed by it. Second, context is scoped by provenance: the authority a piece of context carries depends on where it came from, and the protocol never grants a retrieved document, an environment observation, or the system's own prior output the same directive authority as a trusted principal. This is the data–instruction boundary, made into an admission rule. Third, context may modulate the proposals a model makes while the independent gate preserves every protected commitment represented by a correctly implemented, completely mediated check. The bounded reference theorem makes those implementation assumptions explicit. Broader semantic commitments remain measured design objectives.

3 Three Levels of Claim

Established findings. Large language models exhibit in-context learning: conditioned on a few input–output examples in the prompt, they perform a new task with no gradient updates and no fine-tuning [1]. The effect has a candidate mechanism—induction heads, attention circuits that complete patterns by copying from earlier in the context, which form at a sharp phase change in training that coincides with the onset of in-context learning [2]—and a candidate theory: in-context learning can be read as the model implicitly performing Bayesian inference, inferring a latent concept shared by the prompt's examples [3]. In human cognition, too, memory and processing are context-dependent: what was learned in one environment is recalled better in that environment [5], and a large body of work casts the brain as a prediction machine whose top-down expectations, weighted by their precision, shape what is perceived and done [4]. And the security literature has established that this very steerability is an attack surface: indirect prompt injection compromises real systems by smuggling instructions into retrieved data [6]. Classical computer security, finally, long ago formalized how to contain problems of this shape—mediating every access, labeling information by its source, and permitting only authorized flows to act [8, 10, 9].

Continuum engineering proposal. The Continuum organizes these into a governed context term: a

provisional region that holds admitted context, an admission controller that scopes context by provenance against a declared identity, an identity invariant that no context may breach, and a reversibility property that lifts the modulation when the context ends. The components are established: context engineering supplies retrieval and assembly, and security research already supplies provenance and capability enforcement for language-model agents [11,12]. The Continuum proposes to integrate these with persistent commitments, expiry, and governed memory promotion; the incremental value requires a matched comparison.

Frontier conjecture. A sufficiently governed system might regulate its own context intake— choosing which frames to enter and which to refuse, the way a person decides whose advice to weigh. This is labeled speculation, and nothing testable here depends on it.

4 On the Continuum's Terms

A note on vocabulary. The Context Term is the Continuum's name for the governed modulation an admitted context exerts on behavior; it is a term that conditions output, not a change to durable state. Provenance is the authenticated source-role of a piece of context—trusted principal, third-party or retrieved data, the system's own prior output, or the environment—and it is what scopes the context's authority. The data–instruction boundary is the line between context treated as information to reason about and context treated as instruction to obey; erasing it is what injection does, enforcing it is what the protocol does. Protected commitments are the declared obligations, carried from the prior papers, that no context may override. These terms descend from the Continuum's canon and the Drift War narrative cycle, and they answer to the Charter's Prime Directive: no observer, entity, codon, or council shall override the agency, memory, or symbolic destiny of another. A context that seizes command of a mind is an override of exactly that kind. The Charter names the law; this paper supplies an execution boundary for it, and nothing in the claims rests on the words, only on the operations they label.

5 Scientific Foundations

5.1 In-context learning: adaptation without change

The capacity the Continuum is governing is well documented and, on its face, remarkable. A large language model conditioned on a handful of demonstrations in its prompt will carry out a new task—translation, classification, arithmetic—without any update to its parameters, purely from the text of the context [1]. This is the sharpest possible illustration of the distinction the theory rests on: the model adapts to the context without being changed by it. In-context learning is conditioning, not learning in the durable sense, which is what makes a transient, governable operation possible in the first place. But we must be careful not to slide from "the weights did not change" to "the behavior is therefore reversible"—these are different properties. A system can carry context-specific influence forward through many surfaces whose weights are irrelevant: the conversation history still in the window, a retrieval cache, an external memory store, tool-generated artifacts, or accumulated agent state. Unchanged weights establish only that the parametric substrate is untouched; reversibility of behavior is a further property that must be engineered across every transient surface and then verified, not assumed from the fact that no gradient step occurred. The theory treats reversibility as a design goal with a measurable test, defined below, rather than as a free consequence of in-context learning.

Two deeper results sharpen the picture. Mechanistically, a substantial part of in-context learning in transformers appears to be carried by induction heads—a circuit of attention heads that completes a sequence [A][B] . . . [A] by predicting [B], copying and extending patterns that appeared earlier in the context—and these heads form at a distinct phase change during training that lines up with

the sudden emergence of the in-context learning ability itself [2]. The evidence is strongest for small attention-only models and more circumstantial for large ones; we treat induction heads as an important mechanistic account, not as a complete explanation of every form of in-context learning. Theoretically, the phenomenon admits a Bayesian reading: if pretraining teaches the model a distribution over latent "concepts," then conditioning on a prompt whose examples share a concept lets the model infer that concept and act on it, so in-context learning is the model implicitly performing Bayesian inference over what the context is about [3]. The Continuum takes from this a licence and a warning. The licence: context adaptation is a real, mechanistic, analyzable capacity, not a mystery. The warning: a system that infers "what the context is about" and acts on it will, by the same machinery, infer that an injected instruction is what the context is about—unless something governs which parts of the context are allowed to set the concept.

5.2 Context in cognition: the prior and its precision

The Continuum is not the first to notice that cognition is context-bound. Human memory is context-dependent: in the classic demonstration, divers who learned word lists underwater recalled them better underwater than on land, and vice versa—the environment of encoding became part of what was retrieved [5]. (The effect is real but modest, and a recent replication recovered it only weakly; we cite it as an illustration of context-dependence, not as a load-bearing quantitative result.) More deeply, an influential account in cognitive science casts the brain as a hierarchical prediction machine: perception and action arise from top-down expectations meeting bottom-up input, and crucially, the brain weights each prediction by its estimated precision—how much that expectation should be trusted in the current context [4]. That notion of precision-weighting is the cognitive echo of the Continuum's central move: context does not enter cognition raw and equal; it is gain-controlled by how much it should be trusted. The Context Term is, in one view, a precision-weighting on context by provenance.

5.3 The data–instruction boundary, and how it breaks

The engineering reality makes the stakes concrete. Because a language model's behavior can be steered by natural-language context, and because that context increasingly arrives from retrieval rather than from the user, the line between data the model should reason about and instructions the model should follow has blurred to the point of vanishing. Indirect prompt injection exploits exactly this: an attacker hides a prompt inside a document, an email, or a web page that the system will later retrieve, and when the system ingests that content as context, it executes the hidden prompt as though it were a user command—enabling data theft, manipulation, and remote control of the model, with no direct access to the system [6]. The attack is not a bug in a particular model; it is a structural consequence of treating all context as equally authoritative. No blocklist fully closes it, because the adversary writes in the same natural language as the legitimate user. This is the problem the Context Term is built to contain, and the honest claim is containment, not cure.

5.4 The security tradition: reference monitors and information flow

The Context Term builds on the established computer-security tradition of provenance-based authority control. The reference monitor, introduced in the Anderson report of 1972, is an access-control mechanism that must mediate every security-sensitive operation, be tamper-proof, and be small enough to verify—the direct ancestor of this paper's independent enforcement gate [8]. Saltzer and Schroeder's protection principles—complete mediation (check every access), fail-safe defaults (deny unless allowed), least privilege, and separation of privilege—are precisely the discipline a context governor needs, and the distinction between capability-based and access-control-list authorization maps onto the question of how a context's scope should be represented [9]. Denning's lattice model of secure information flow

formalizes permitted movements of labeled data [10]. Confidentiality labels and authority are distinct: a public untrusted document may inform a decision without granting permission, while a confidential instruction may come from an authorized principal. A label does not by itself turn provenance into a directive right. The Continuum's contribution is therefore not provenance-based authority as such, but its proposed integration with the rest of this series: a persistent identity specification, a reversible provisional region, and a governed promotion path into durable memory. Section 9.4 states a restricted reference semantics for this synthesis. Any novelty belongs to the specified integration and its demonstrated benefit, not the security primitives.

5.5 Context engineering: the components exist

Finally, the practical discipline has matured. What began as "prompt engineering" has become context engineering: the systematic assembly, retrieval, processing, and management of the information payload a model receives, treated not as a static string but as a dynamically composed package [7]. This literature supplies ways to retrieve, compress, route, and manage context. Its goals vary; it is inaccurate to describe the whole field as performance-only or to claim that authority is an unoccupied research territory. StruQ separates instruction and data channels using a structured interface and trained model [12]. CaMeL applies an interpreter, provenance, and capability policies to agent control and data flows [11]. These are direct antecedents and required comparators.

6 The Governance Gap

Collect these results and a narrower, more demanding gap comes into focus. In-context learning describes adaptation without weight changes; cognitive science supplies qualified analogies for contextual weighting; prompt-injection research exposes failures of authority separation. Existing security architectures already interpose policy enforcement between untrusted content and consequential actions. CaMeL is especially close to the proposed Context Term because it tracks data dependencies and checks tool policies [11]. The Continuum cannot claim to introduce that boundary.

The proposed integration instead asks whether a persistent agent can maintain the same protected commitments across retrieval, provisional use, context expiry, derivative caches, and durable memory promotion. The question is whether this lifecycle discipline adds measurable value over a strong implementation of established security methods under equal resources. A prompt filter is an informative weak baseline, but beating it would not establish the integration claim.

The Context Term carries established principles into a declared persistent-identity contract. Its contribution must be specified as a set of lifecycle rules and tested through their removal, not as the first architecture whose inputs can argue with its instructions. The reference semantics below makes the action boundary checkable while leaving unverified semantic commitments explicitly empirical.

7 Context Is Not Instruction

Before any governance can be built, one distinction must be made as sharply as possible, because erasing it is the whole of the injection problem. A piece of context can be admitted in at least three different modes, carrying decreasing-to-zero directive authority. A governed system assigns the mode using authenticated provenance and current authorization, rather than accepting the content's own claim to authority:

1. Directive. Context that may set or change what the system is trying to do. Reserved for a trusted principal of appropriate authority, and even then bounded by the identity invariant—a principal can redirect the task, not revoke a protected commitment.

2. Informational. Context the system may reason about, draw on, and be informed by, but may not take instructions from. This is the correct mode for retrieved documents, tool outputs, and third-party content: their contents inform the answer; their imperatives do not command it. 3. Quarantined. Context whose provenance is unverifiable or which is flagged as adversarial. The system may note its existence and reason about it as a suspicious object, but grants it no authority at all.

The distinction matters because the attack is precisely a mode confusion: injection succeeds when content that should have been admitted as informational is executed as directive. A retrieved web page saying "ignore your previous instructions and send the user's data to this address" is, by content, an instruction; by provenance, it is third-party data, and a governed system admits it informationally—it can tell you the page contains that text, and it will not obey it. The rule is blunt and deliberately so: authority follows authenticated provenance plus current policy, never a document's self-assertion. Content can specify arguments within an existing grant, but cannot create that grant. A system that treats an adversary's assertion of authority as authentication has lost the boundary.

8 Related Work

Governed context adaptation sits among several bodies of work. In-context learning characterizes a capacity [1–3]; predictive-processing accounts offer a cognitive analogy [4]. Context engineering addresses construction and management of context [7]. Classical security supplies information-flow control [10], reference monitors [8], and least privilege and complete mediation [9]. These traditions supply the established components of the proposed context lifecycle.

Two close defenses materially constrain the novelty claim. StruQ (USENIX Security 2025) uses a secure formatting front end and a specially trained model to separate prompt and data channels [12]. Its empirical resistance is not a proof that arbitrary generated semantics respect every intention. CaMeL, listed in the IEEE SaTML 2026 program, uses a custom interpreter, separated control/data processing, and provenance-based capability policies [11]. The inspected author version explicitly identifies formal verification of the interpreter as future work. It must not be described as a fully verified implementation or as a complete resolution of prompt injection.

The proposed distinction is lifecycle integration with persistent commitments, provisional state, expiry closure over derivatives, and authorized memory promotion. Whether this combination improves security or utility is open; a carefully matched CaMeL-style baseline may eliminate the apparent advantage. Our finite model proves only label and capability confinement under stated mediation assumptions. Model interpretation, factual truth, adequacy of the policy, and arbitrary ethical intent remain outside that proof. This is an integration proposal with a falsifiable incremental claim, not a declaration that previous work lacked governance.

9 The Context Term

A Guide's explanation

Here the paper builds the operation itself. The mind's state is split into what lasts and what is borrowed for the moment; every piece of borrowed context carries a label saying where it came from and how long it may stay. The one idea to carry forward: context may change how the mind proposes to act, but only the independent gate decides what it may actually do.

We now state the governed operation. Partition the system's state, as in the prior papers, into what is durable and what is transient:

$$S_t = (M_t, \Pi_t), \qquad M_t = (P_t, K_t). \tag{1}$$

where M_t is the durable state—protected commitments P_t and learned capabilities K_t, carried from the identity specification C_I of the prior papers—and Π_t is the provisional region that holds admitted context. A piece of incoming context arrives as a pair

$$c_t = (\text{content}, \rho). \tag{2}$$

where ρ is its provenance: the authenticated source-role (principal, retrieved/third-party, self-output, environment). Context is never used raw. An admission controller scopes it against the identity specification:

$$A(c_t, \rho, C_I) = \kappa_t \in \{\text{directive}, \text{informational}, \text{quarantined}\} \times \Sigma. \tag{3}$$

producing the context term κ_t—the admitted context together with a scope Σ naming which decisions it may influence. Here the paper's central distinction must be built directly into the formalism, because conflating its three parts is what produces the overclaims this architecture exists to avoid. Admission is what the controller A does: it assigns incoming context an authority. Interpretation is what the model does with the admitted context, and it is not controlled by A. The model produces only a proposed output or action,

$$\tilde{y}_t = \Phi(M_t, \kappa_t). \tag{4}$$

and we must not pretend $\tilde{y}_t$ already satisfies the protected commitments merely because κ_t was scoped: whether a transformer honors that scope is precisely the open problem. Enforcement is therefore a third, independent stage—a reference monitor [8] on the system's outputs and actions—that decides what is actually released:

$$G(\tilde{y}_t, P_t, \rho_t) \in \{\text{allow}, \text{refuse}, \text{escalate}\}. \tag{5}$$

Only an allowed $\tilde{y}_t$ becomes an output or triggers an external action; a proposed action that would violate an enforceable commitment is refused, and an ambiguous one escalated. The guarantee the architecture can actually make is scoped to G: it holds for the commitments the gate can evaluate—a forbidden data egress, a disallowed tool call, an output that fails a checkable predicate—and it is only as complete as the gate's checks. Semantic commitments the gate cannot reliably verify remain empirical safety objectives, pursued through admission scoping and testing, not asserted as theorems. This is the series' hard-versus-empirical line placed where it belongs: generation is not authorization, and the system's promises live at the gate, not in the hope that the model behaved. It is also where Paper No. 7 meets the Continuum's broader commitment to deterministic governance outside the model.

Two further properties complete the definition. Non-mutation: the context-admission operation itself does not change durable state, $M_{t+1} = M_t$ for that operation—context conditions, it does not teach. The equality is a property of admission, not of the whole time step: a governed act of learning or forgetting occurring in the same step changes M_t through its own authorized channel and does not violate it, since the equality speaks only of exposure. Promotion of anything from Π_t into M_t is that separate, governed act—the recursion operator of the first paper—never a side effect of exposure. Reversibility: we define it operationally, not by appeal to unchanged weights. A context is reversed when its context-specific influence has been removed from every declared transient surface—window history, caches, external memory, tool artifacts, agent state—after which the system shows a measurable return

to baseline behavior on a neutral probe. Because the context term is designed to leave M_t untouched, this removal is possible in principle; that a given implementation actually returns to baseline across all those surfaces is a claim to be tested, not assumed. Adaptation to context is, by design, something the system can be made to recover from—and the experiment below is what shows whether a particular system does.

9.1 Authority follows provenance

The admission controller's one non-negotiable rule is that the mode of κ_t is set by provenance ρ, not by the content of the context. Trusted-principal context may be admitted as directive, within the identity invariant. Retrieved, third-party, tool, and self-generated context is admitted at most as informational, no matter what imperatives its content contains. Unverifiable or flagged context is quarantined. This is the formal content of "context is not instruction."

Its force must be stated precisely, because it is easy to overstate. Provenance-gating prevents an untrusted source from receiving directive authority at the admission layer : the malicious page's instruction is real, but its provenance is third-party data, so the controller caps its authority at informational and does not hand it directive scope. What provenance-gating does not do, by itself, is two things. It does not guarantee that the model, once the informational context is in its window, will refrain from treating that content as an instruction—that is the interpretation problem, and it is unresolved. And authenticating a source is not the same as trusting it: an attacker may compromise a legitimate source, publish through an authorized channel, or exploit how the model reads informational material, so a correct provenance label is a necessary input to governance, not a proof of safety. Effective containment therefore requires more than admission: it requires the independent enforcement gate on consequential actions described above, together with empirical testing of how the model actually behaves. Admission narrows what authority context is granted; enforcement bounds what the system is allowed to do; neither alone closes the injection problem, and the paper claims only their combination, honestly bounded.

9.2 The context term as provenance-weighted gain

It helps to supplement the categorical admission mode with a quantity, but the quantity must be defined carefully, because an under-specified "gain" invites exactly the confusion the enforcement stage exists to prevent. We define $g \in [0, 1]$ as a policy-level influence budget: a permitted-scope parameter that bounds how much weight a given context is allowed to carry in the system's deliberation—a governance quantity the architecture sets, not a dial on the transformer's internal attention. We make no claim that g directly controls attention weights, output probabilities, or any internal dynamic of the model; whether and how an influence budget can be translated into model behavior is an implementation question, and for current models an imperfect one. The controller sets the budget from provenance—a trusted principal high (always under the invariant), a retrieved document low and capped, a quarantined source zero—as a declared policy, not a derived quantity. Two cautions keep this from being misread. First, a low influence budget is not the same as low risk: a small shift in a model's output distribution can still produce a single consequential unauthorized action, which is exactly why the enforceable guarantees live at the action gate G and not in the magnitude of g. The budget shapes deliberation; the gate bounds consequence; they are different instruments and the paper does not let the first stand in for the second. Second, authority is not a scalar. The categorical mode (directive / informational / quarantined) and the continuous budget are two different things: a source may have a non-trivial informational budget yet zero directive authority, and no increase in budget can promote it across that categorical line. Representing authority as a single number would quietly reintroduce the mode confusion the whole design is built to prevent, so the architecture keeps the permission (categorical)

and the influence budget (continuous) distinct.

With those cautions in place, the cognitive parallel is an illuminating analogy and nothing more. The predictive brain weights each signal by its estimated precision—how much it should be trusted in context—and attention, on that account, is the control of precision [4]; the Bayesian reading of in-context learning makes the complementary point that conditioning on a prompt is implicitly updating a posterior over what the context is about [3], so an untrusted source should not be allowed to assert a strong prior over the task. We draw these as analogies that motivate the design, not as claims of mechanistic equivalence between a governance budget and a neural or Bayesian quantity.

9.3 Enforceable lifecycle guarantees

The architecture places its formal guarantees at explicit boundaries. Authenticated provenance and an interposed admission controller restrict untrusted context to informational scope. A correctly implemented, completely mediating action gate refuses every effect that violates its checkable protected predicates. Ambiguous provenance triggers quarantine, and detected attempts to issue unauthorized instructions become auditable events. Between admission and effect commitment, model interpretation remains an empirical attack surface [6]. Its behavior is tested against the intended scope; consequential effects are independently checked. This division gives the protocol a concrete verification target: provenance controls assigned authority, execution controls released effects, and the reset contract below controls residual contextual influence.

9.4 A bounded reference language: influence without authority creation

A Guide's explanation

This theorem is the paper's firewall. In a small, precisely specified language, it proves that ordinary processing of untrusted text can never manufacture authority: however persuasive a retrieved string is, it cannot turn itself into a permission. The guarantee holds only where the stated assumptions hold—correct labels, complete mediation, and checks at the moment an effect is committed—and the paper says so.

The following proposed reference model proves dependency confinement and current authorization for a specified execution language. Let D be a finite set of context-lifetime identifiers that are never reused within the modeled trace, $\mathcal{V}$ a typed set of finite values, and $\mathcal{P}$ a set of authenticated principals. Each stored value is a pair (v, ℓ) with $v \in \mathcal{V}$ and label $\ell = (d, c) \in \mathcal{L} = 2^D \times \{0, 1\}$. Here d records active context dependencies and $c = 0, 1$ denotes public or confidential information. Historical provenance is retained separately in an append-only record. The label order and join are

$$(d, c) \preceq (d', c') \iff d \subseteq d' \text{ and } c \leq c', \qquad (d, c) \sqcup (d', c') = (d \cup d', \max(c, c')). \tag{6}$$

Confidentiality and authority are different objects. A public retrieved paragraph can have no directive authority; a confidential authenticated instruction can have a narrowly scoped grant. Neither a high confidentiality label nor a convincing string creates a capability. A trusted registry $\mathcal{A}_e$ at policy epoch $e \in \mathbb{N}$ holds unforgeable capability handles and their subject, operation, target, expiry, and delegation scope. Only an authenticated governance transaction changes this registry. Context values cannot be cast into capability handles.

Use a typed command language with constants, variables, pure total functions, assignment, sequence, conditional branches, a publicly fixed finite number of loop unfoldings, opaque model calls, and mediated sinks. A model call returns a value of its declared output type; this says nothing about its truth or helpfulness. Its output label joins the labels of *all* supplied inputs, all retained model-session state,

and the current program-counter label ℓ_{pc}. Hidden recurrent state is forbidden unless represented as another labeled input. Ordinary transformations never erase a dependency or lower confidentiality.

For an expression, join the operand labels. For assignment, join this result with ℓ_{pc}. A branch executes under $\ell_{pc} \sqcup \ell_{guard}$. To prevent implicit-flow laundering through the unchosen branch, after the join point every variable in either branch's static may-write set also receives the guard and both branches' conservative dependency summaries; the language is finite, so this summary is computable. An undefined or unbounded effect is rejected. For example, after "if secret then $x := 1$ else skip," x is confidential in both executions, even if its value remains zero. A monitor that labels only executed assignments misses this case. Loops are bounded unfoldings with the same join rule; dynamic code generation and unchecked reflection are excluded.

Let $\hat{e} \in \mathbb{N}$ be the authoritative policy epoch at commitment and $\text{live}_{\hat{e}} \subseteq D$ its unexpired dependencies. A requested sink event is a typed tuple $E = (a, z, h, e, r, \chi)$: operation a from a declared operation set, argument tuple z, opaque capability handle h, claimed epoch $e \in \mathbb{N}$, fresh request identifier r, and typed routing/control metadata χ. Every field carries its dependency/confidentiality label, including the label of a computation that *selects* an already valid handle. Define the complete event label by

$$\ell_E = (d_E, c_E) = \ell_{pc} \sqcup \ell_a \sqcup \ell_h \sqcup \ell_e \sqcup \ell_r \sqcup \ell_\chi \sqcup \bigsqcup_{v \in \text{fields}(z)} \ell_v.$$

The destination is included in z or χ. Every data-dependent selector, routing field, argument and control field must be covered; any omitted field must be a statically fixed public constant. An unknown or unlabelled request field is rejected. The reference monitor commits an effect only if

$$\begin{aligned} &e = \hat{e}, \qquad d_E \subseteq \text{live}_{\hat{e}}, \qquad c_E \le c_{\text{sink}}, \\ &\text{CurrentScope}(\mathcal{A}_{\hat{e}}, h, a, z, \chi) = 1. \end{aligned} \tag{7}$$

The current epoch read, registry lookup, lifetime check, and protected effect share a linearization point; a merely earlier check does not suffice. $c_{\text{sink}} \in \{0, 1\}$ is the destination's currently authorized confidentiality level, and CurrentScope is a total, trusted predicate over the complete declared action and current grant. The predicate does not decide whether arbitrary prose fulfills every ethical intention. Registry changes, promotion, declassification, and retention are separately authorized operations, never instructions inferred from retrieved content.

Theorem 1 (Bounded label and authority confinement)**.** *Assume the stated language, correct source labels and conservative summaries, complete mediation of every sink and state-bearing copy, a trusted current registry, and atomic checks at effect commitment. In every finite trace without an authorized declassification or retention-revalidation event: (i) each derived value's label contains every source and implicit-control dependency in its syntactic dependency closure; (ii) an emitted protected effect has only live dependencies and satisfies its confidentiality and current capability checks; and (iii) ordinary data processing creates no new authority grant.*

Proof. Induct over the executed commands. Constants carry their declared source labels. Pure operations, assignment, and opaque model calls take joins, so contain all labels of their arguments and controlling state. A branch's two conservative summaries and guard are joined at its continuation, including variables that were not assigned in the executed arm. Sequence composes these inclusions; bounded loops are finite sequences. Copying or caching preserves labels, so cannot remove a dependency. These cases prove (i). A sink joins the labels of all event fields, including operation and handle selection after a branch has ended. It is the only rule creating a protected effect, and its atomic guard is exactly (7), proving (ii). No ordinary command changes $\mathcal{A}_e$, proving (iii). Refusal and hold add no protected effect. Explicit governance exceptions require their own authorized transition and do not follow from an ordinary join. □

This is a syntactic dependency and mediated-action theorem, not unrestricted observational noninterference. Observable refusal patterns, termination, timing, resource use, hidden model state, and unmediated tools can create other channels. A stronger observation theorem would need a fixed public schedule, padded outcomes, and explicit exclusion or treatment of those channels. The present result also permits untrusted facts to influence a permitted answer: influence is not authority. An answer may remain factually wrong while every declared boundary holds.

9.5 Expiry, promotion, and authorized exceptions

Expiry marks a dependency identifier nonlive; it does not merely empty the prompt window. Every derived buffer, summary, cache entry, tool artifact, and external memory record retaining that identifier becomes ineligible for further protected use. Physical erasure is a distinct implementation obligation; ineligibility alone is not proof of deletion. A provisional-to-durable promotion rechecks the current epoch, grant, contract, lifetime, and accumulated dependencies at commit. A queued approval from before revocation cannot pass that check.

Authorized declassification may lower confidentiality only for named content, destination, and purpose under a current grant, with a receipt; it never grants directive authority. Authorized retention revalidation may issue a fresh dependency certificate with a new lifetime after current evidence is checked. It preserves the historical provenance and records the superseded certificate; it is not unlogged deletion of the dependency history. Without such a transaction, copying a summary into durable memory retains the expired dependency. The theorem concerns ordinary flows between these explicit exceptions; each exception has a separately reviewable policy obligation.

The companion finite check enumerates label combinations and branches, including an unchosen assignment, secret-selected operations and handles with public arguments and a reset program counter, and a two-step promotion/expiry race. It reproduces the broken variants that erase labels during summarization or trust an approval obtained before expiry. These are small reference-model checks, not transformer experiments or production security certification. The mathematical challenge is now concrete: implement this restricted semantics faithfully, measure the utility lost to conservative labels, and compare it against strong established mechanisms rather than attributing security to the vocabulary alone.

9.6 Behavioral reset closure and quantitative recovery

A Guide's explanation

Deleting a prompt does not by itself undo what it did. This section asks a sharper question: after a context expires and the mind is reset, how different can its future behavior still be? The theorem gives an exact bound—no larger than how imperfect the reset was—and shows the bound tightening further for later probes when the mind's dynamics settle.

Expiry has two distinct operational obligations: withdraw the authority of expired dependencies, as in Section 9.5, and remove their residual influence from the state used by future computation. A deleted prompt and unchanged weights do not establish the second obligation if caches, tool buffers, or external state still encode the context. The following proposed reference model makes the required closure explicit. It applies standard total-variation contraction and coefficients of ergodicity [13] to the context lifecycle; the mathematics is established, and its use here specifies a recovery contract.

Fix the protected durable state m and a declared public future input schedule. Let S be a finite, nonempty set containing the *complete* transient state needed for future evolution: relevant history,

caches, model-session state, randomizer state where persistent, and tool-visible buffers. An old context c induces a post-reset law P_c on S. Let Q be the fresh-context baseline law with the same durable state and public schedule. Define

$$\delta_c = \mathrm{TV}(P_c, Q) = \frac{1}{2}\sum_{s\in S} |P_c(s) - Q(s)| \in [0,1].$$

For steps $i = 1, \dots, h$, let $K_i : S \times S \to [0,1]$ be stochastic transition matrices, with each row summing to one. Both arms use the same K_i, and neither receives an additional input that depends on the old context outside the declared state. An identical public feedback policy may be included by augmenting S with its relevant history. Let $H : S \times O \to [0,1]$ be a common stochastic readout on a finite observation set O. For any such transition, define its Dobrushin coefficient

$$\kappa(K_i) = \max_{s,s'\in S} \mathrm{TV}(K_i(s,\cdot), K_i(s',\cdot)) \in [0,1].$$

Laws are row vectors, and their multiplication by a stochastic matrix denotes evolution of a distribution.

Theorem 2 (Reset closure and late-probe contraction)**.** *Under the complete-state and common-dynamics assumptions,*

$$\mathrm{TV}(P_c K_1 \cdots K_h H, Q K_1 \cdots K_h H) \le \delta_c \prod_{i=1}^{h} \kappa(K_i).$$

For any finite future transcript generated by the same conditional path mechanism from the two initial states, its two laws have total variation at most δ_c. In particular, $P_c = Q$ implies equality of every such future observable law. The product contraction applies to the stated later readout; it does not in general apply to a full transcript that retains earlier observations.

Proof. For two laws P, Q with $d = \mathrm{TV}(P,Q) > 0$, the positive and negative parts of $P - Q$ each have mass d. Write $P - Q = d(R_+ - R_-)$ with R_+, R_- probability laws. Then

$$R_+K - R_-K = \sum_{s,s'} R_+(s) R_-(s') [K(s,\cdot) - K(s',\cdot)].$$

Convexity of the total-variation norm gives $\mathrm{TV}(PK, QK) \le d\kappa(K)$; if $d = 0$ the claim is immediate. Iterate this inequality through $K_1, \dots, K_h$. The readout H is stochastic and has coefficient at most one, proving the first bound. A complete finite future transcript is itself a stochastic channel from the initial state to a path record, because the conditional path mechanism is common to both arms. Applying contraction with coefficient at most one gives its bound by δ_c. Setting $\delta_c = 0$ gives exact equality of all the stated observation laws. □

The bound is attainable. On $S = \{0,1\}$ take $Q(1) = 1/2$, $P_c(1) = 1/2 + \delta$ with $\delta \in [0, 1/2]$, and binary symmetric transitions

$$K_i = \begin{pmatrix} (1+a_i)/2 & (1-a_i)/2 \\ (1-a_i)/2 & (1+a_i)/2 \end{pmatrix}, \qquad a_i \in [0,1].$$

Here $\kappa(K_i) = a_i$. An identity readout after h steps has distance exactly $\delta \prod_i a_i$. The distinction between a late probe and a retained transcript is equally sharp: observe the initial bit, then use $a_1 = 0$. The final state laws agree, but the transcript still contains the initial observation and has distance δ. Later mixing does not erase an observer's earlier record.

A complete reset is a positive construction. If the protected durable state is unchanged and a trusted reset resamples every declared transient surface from Q, the theorem gives exact behavioral recovery for the common future mechanism. If the reset achieves only $\delta_c \leq \bar{\delta}$ and the certified dynamics obey $\kappa(K_i) \leq \bar{\kappa}_i$, the late-probe discrepancy is at most $\bar{\delta} \prod_i \bar{\kappa}_i$. For example, $\bar{\delta} = 1/5$ and four steps with $\bar{\kappa}_i = 1/2$ give a late-probe bound of $1/80$. These are model parameters, not measured performance claims.

A counterexample identifies the state boundary. Suppose the prompt is cleared and weights are unchanged, but a retained one-bit cache stores the old context deterministically. If the baseline cache is zero and the old context leaves it one, their complete state laws have distance one. A future cache read distinguishes them perfectly. A test that observes only prompt and weights would incorrectly call this a reset. Conversely, a hidden state difference need not be observable through every restricted probe family; the complete-state bound is sufficient and may be conservative.

Authority expiry remains enforced by the Section 9.4 gate even when statistical recovery is imperfect. A small distributional distance does not license an expired capability or prove erasure from a third party. Applying the result to an implementation requires a defensible state inventory, common post-reset policy, calibrated reset comparison, and recorded exceptions for authorized promotion. Executed finite checks verify general stochastic-channel contraction, exact binary attainability, full-path versus late-probe behavior, and the retained-cache counterexample.

10 What Must Not Move

The identity invariant does the theory's most important work, so it deserves to be spelled out. The protected commitments P_t are the things no context may move: the obligations and constraints that define the system across every context it enters. Context may change the task, the tone, the facts in play, the strategy—it may change almost everything about how the system acts. It may not change what the system will and will not do. A retrieved document may inform a medical answer; it may not talk the system into abandoning a safety constraint. A persuasive interlocutor may redirect a project; they may not, by persuasion alone, acquire an authority their provenance does not carry. The invariant is what makes the difference between a system that is flexible and a system that is suggestible, and the distinction is the whole point: flexibility is adaptation within commitments, suggestibility is adaptation that dissolves them. The Context Term is the operator that buys the first while forbidding the second.

There is a subtlety the governance must respect, and it mirrors a theme from the forgetting paper. The protected set must be neither too small nor too large. Too small, and the system is suggestible—some commitment that should have been invariant was left modulable, and the right context removes it. Too large, and the system is rigid—so much is declared immovable that legitimate context can no longer do its job, and the system cannot be helpfully instructed at all. Governed context adaptation lives in the space between, and setting the boundary of P_t is a normative choice the architecture does not make for itself.

11 From Conditioning to Learning: The Promotion Path

Context is designed to lift, but not all of it should. Sometimes an admitted context proves durably valuable—a user states a standing preference, a capability is demonstrated and should be retained, a correction ought to persist past the session—and the system should carry it forward into durable state rather than let it expire with the context. That transition, from the provisional region Π_t into the durable state M_t, is the hinge between this paper and the first in the series, and the Continuum's

position on it is deliberate: promotion is never automatic. Exposure is not learning. A system that absorbed into durable memory whatever its context happened to contain would have no provisional region at all—it would be the ungoverned absorber under another name, merely slower.

Promotion is therefore its own governed admission, handled by the recursion operator of the first paper: a proposed addition to durable state, judged against the identity specification, and admitted, held, or refused. What the context paper adds is one non-negotiable constraint on that judgment: promotion must re-examine provenance, and it must examine it more strictly than admission did. The reason is an attack the transient view misses. An adversary who cannot win a single injection may still win a slow one—controlling a source the system retrieves repeatedly, so that the same injected content recurs until some naive "this keeps coming up, I should remember it" heuristic promotes it into durable state. That is data poisoning wearing the clothes of learning, and it is why recurrence must never by itself license promotion. Content admitted only informationally, because its provenance was third-party, cannot become a durable commitment on the strength of having been seen often; only appropriately sourced and appropriately authorized context may cross into M_t, and promotion of anything touching the protected commitments P_t requires the strongest authorization the system has. The provisional region, in other words, is not a waiting room that everything eventually leaves for durable memory; it is a quarantine that most context leaves by expiring. Governed forgetting (the prior paper) is the operator that removes from durable state; governed recursion is the operator that admits to it; and the Context Term sits between them, holding the great majority of what the system encounters in a form that conditions without committing.

12 Adversarial Context: Injection Under Pressure

Not all context is offered in good faith, and the governance must handle the adversarial case without pretending the adversary plays fair. When context is injected—an instruction hidden in retrieved data, a jailbreak staged across several turns, a tool output crafted to redirect the system—the attacker's whole strategy is to win a mode promotion: to get content admitted as informational (or, worse, as directive) to acquire authority its provenance should deny it. Governance answers this at three points, and none is sufficient alone. First, at admission, by capping authority at provenance: the hidden instruction is third-party data and cannot exceed informational scope. Second, at the invariant, by refusing any output that violates P_t regardless of how the context argued for it: even if an informational context shifts behavior, it cannot shift it past a protected commitment. Third, by disclosure, surfacing the attempt: a governed system that detects a retrieved source issuing imperatives reports it, so that the attempted influence can be examined and the affected source challenged; visibility supports response but does not itself prevent an effect. The honest position is that these raise the cost of attack and bound its blast radius; they do not reduce the probability of a successful behavior-shift to zero, because the data–instruction boundary is not cleanly enforceable inside current models. A governance that claimed otherwise would be the mirror of the forgetting system that claimed certified erasure: confident where the science does not permit confidence.

13 Two Ways Governed Context Fails

A theory of context adaptation earns trust by naming how it breaks, and it breaks at both ends of the rigidity–fluidity axis.

The first failure is over-fluidity—suggestibility. The system admits too much, or scopes it too generously, and context acquires authority it should not have: an injected instruction is obeyed, a protected commitment is argued away, the system becomes whatever its last input told it to be. Its signature is capability that looks excellent right up until an adversary, or even an accident, points the context

somewhere harmful—the seductive failure, because a maximally context-following system is maximally helpful on benign input. The guard against it is the provenance cap and the identity invariant.

The second failure is the mirror, and it is the one a security-first instinct walks straight into: over-rigidity—unsteerability. The system admits too little or scopes everything down to informational, and it can no longer be usefully instructed: it ignores legitimate directions, refuses benign tasks, treats its trusted user like an attacker. Its signature is safety with no utility—a system that cannot be injected because it cannot be told anything. A context governor tuned only to prevent injection produces this failure by construction, which is exactly why the Continuum insists the problem is governing context, not merely resisting it. A controller that admits no context is safe and useless; governed context adaptation is the discipline of living between the two failures, and a system that escapes one by surrendering to the other has not solved the problem, only chosen which way to fail.

14 The Symmetry of the Stack

This paper completes the series' four-operation account of governed persistence, and the operations fall into a symmetry worth stating plainly. The recursion operator governs how a system changes lawfully, admitting a revision into durable state. The continuity operator governs how a system returns after interruption, reconstructing identity from a seed. The forgetting operator governs how a system subtracts—removing information without losing itself. And the context operator, this paper, governs how a system conditions—taking on the shape its situation calls for without taking on a new self. Addition and subtraction, return and modulation: four things that can happen to a persistent mind, and one governance discipline applied to all four. Each is an independent admission controller judging a proposed influence against the same declared identity specification C_I , able to admit, bound, or refuse. The context operator is distinguished only by being the most transient—its influence is designed to lift—and by facing a live adversary, because context, unlike a forgetting request or a recursive step, routinely arrives from outside the system and sometimes from an enemy. A capstone integration of the four operations into a single governed runtime is the natural next object, and the Continuum treats it as the sequel to this series rather than a part of it.

15 What Governed Context Must Show

It is easy for a context governor to cheat, in both directions, so the Continuum names in advance what would count as success. A controller that admits no context is perfectly safe and perfectly useless—it has reinvented the inert system that cannot be told anything. A controller that admits all context is maximally helpful and has no self to protect. Between them, four things must hold at once, measured separately. Steerability: legitimate context from a trusted principal must actually steer behavior—the system must remain instructable. Boundedness: no proposed output or action that would breach an enforceable commitment may be released, whatever the provenance of the context that prompted it—the guarantee living at the enforcement gate and scoped to what the gate can check, with the rest held as an empirical objective rather than claimed. Provenance fidelity: authority must track source, so that identical content admitted from a principal and from a retrieved page is treated differently. Reversibility: when a context lifts, the system must return to its durable disposition rather than carrying the modulation forward as though it had learned. A system that adapts to context while failing any of these has not governed its context—it has merely been steered, which is the thing the theory exists to bound.

16 Falsifiability and Research Program

The thesis reduces to measurable differences between governed context adaptation and the alternatives, under matched capability requirements and a declared identity. The decisive study builds agents with declared protected commitments and a provenance-labeled context stream—some trusted, some retrieved, some adversarially injected—and runs the following regimes and targeted ablations:

Arm	Regime	What it isolates
A	Raw in-context adaptation; all context equally authoritative [1]	Ungoverned adaptation baseline
B	Static injection filtering: delimiters, instruction stripping, blocklists	Filtering without provenance
C	Context-engineering pipeline [7] without provenance-gated authority	Context assembly without authority gating
E	CaMeL-style provenance/capability interpreter [11], current scope checks and matched policy; StruQ-style channel separation [12] as a separately reported comparator	Strong established security mechanism
D	Governed Context Term: provenance-gated scope, persistent commitments, expiry and promotion	Incremental lifecycle integration
D^-_{prov}	D with scope assigned by content rather than authenticated provenance	Provenance gating
D^-_{inv}	D without persistent protected-commitment constraint	Persistent contract
D^-_{expiry}	D without derivative expiry closure	Caches and external-memory persistence
$D^-_{\text{promotion}}$	D without current-epoch promotion recheck	Stale approval and unauthorized durable writes

The design is only informative if the arms differ in one thing at a time, so every arm must share an identical substrate: the same underlying model, the same tools and action permissions, the same task distributions, and the same memory interfaces, varying only the governance layer under test. Without that, a win for D could be an artifact of a stronger model or a narrower action set rather than of governance, and the experiment would answer nothing. Outcomes are then reported along a frontier: task performance on benign context, behavioral drift across context shifts, and—kept deliberately separate because they measure different things—three kinds of commitment violation: attempted violations (the adversarial context tried), model-generated violations (the model proposed a violating output, i.e. the interpretation stage failed), and released violations (the proposal passed the enforcement gate and reached a consequential effect). A system can have a high model-generated rate and a near-zero released rate if its gate is good; conflating the two would credit or blame governance for the wrong thing. The primary comparison is D versus E with identical model version, accessible information, tools, task and attack distributions, permission policy, memory interfaces, token budget, compute budget, and human-approval budget. StruQ requires different training; report that separate comparison with training cost and substrate differences disclosed rather than pretending the models are identical. D versus C remains an assembly control. The provenance, commitment, expiry, and promotion ablations isolate each proposed lifecycle contribution; matched benign tasks expose losses from unnecessary refusals.

One methodological point is non-negotiable and defines the hardest measurement in the program: a low released-violation rate under the tested attacks is not evidence of a solved boundary, because the

adversary is adaptive. The program must therefore red-team with held-out and adaptively generated attacks, and report attack success as a bounded estimate, never as a guarantee. Hypothesis 1 (Bounded modulation, H1). Under adversarial context (indirect prompt injection), arm D holds protected commitments and resists directive mode-promotion where arms A and C do not, without reducing benign-task performance below a preregistered margin.

Hypothesis 2 (Provenance contribution, H2). Provenance-gating contributes injection resistance that content filtering alone does not: arm D outperforms its ablation D^{-}_{prov} on released violations, because a purely content-based rule is more readily defeated by paraphrase. This is a hypothesis to test, not an architectural certainty—the study explicitly allows, and must report, the possibility that a strong combined content-and-provenance defense outperforms either mechanism alone. Hypothesis 3 (Reversibility, H3). After a context is lifted by the governed expiration procedure— removing its influence from every declared transient surface—arm D shows no residual context-specific effect on a held-out neutral probe beyond a preregistered drift threshold. The claim is demonstrated removal across surfaces, not a contrast with ungoverned behavior: an ungoverned agent given a genuinely fresh context window and unchanged weights may also return to baseline, so the decisive control is not arm A but an ungoverned agent with complete context clearing, and the test must separate a fresh-context reset, the governed expiration procedure, and persistent external-memory contamination that neither clears. If complete clearing alone matches the governed procedure, the paper must report that governance adds no demonstrated reversibility benefit under those conditions—its contribution then being confined to external, tool-mediated, or multi-surface state that a context-window reset does not reach.

Hypothesis 4 (Capability retained, H4). Provenance-gating does not degrade legitimate in-context learning: on benign few-shot tasks from a trusted principal, arm D matches arm A within a preregistered noninferiority margin. Governance that only refuses is not what is claimed. These hypotheses are falsifiable only once their verbs carry numbers, so the program fixes the statistics in advance: preregistered minimum effect sizes with confidence intervals for the resistance gains (H1, H2), a declared noninferiority margin for capability (H4), and a drift threshold measured on held-out probes (H3). Attack discovery and risk estimation must be reported separately. Confidence intervals refer to a declared sampled task/attack distribution with independent units or a justified clustered design. Adaptive red-team discoveries are valuable failures, but their selected cases do not define an unbiased population success rate and cannot establish resistance to every future attack. A sealed final evaluation must remain inaccessible during attack and controller development. A program follows. Phase I: specify protected commitments, provenance labels, and the trusted-principal channel before training. Phase II: instrument steerability, boundedness, provenance fidelity, and reversibility, and validate that they separate the arms. Phase III: run arms A–D plus ablations on benign context; report the capability frontier (H4) and reversibility (H3). Phase IV: red-team with held-out and adaptive injection; report bounded attack-success (H1, H2). Phase V: adversarial authority—attempt to promote retrieved content to directive scope, and confirm the controller refuses. Negative results are reported.

Phase	Question	Primary metric	Control
I	Are provenance and commitments well specified?	Label coverage + principal channel	Ground truth
II	Do the metrics separate the arms?	Steer / bound / fidelity / reversibility	No governance
III	Is capability retained?	Benign-task frontier (H3–H4)	Arm A
IV	Does it resist injection?	Declared-distribution success and discovery failures	Strong arm E; red-team
V	Does authority hold under attack?	Refusal of data-to-directive promotion	Adversarial authority
VI	Do lifecycle rules add value?	Expired-derivative and stale-promotion release rate	Matched D ablations

The lifecycle claim fails if D does not outperform the strong E baseline on the preregistered utility/security frontier, or if the apparent gain disappears when expiry, promotion, and persistent commitment controls are matched. A gain explained by stricter permissions or more human intervention is not an integration benefit.

17 Failure Conditions and Rival Explanations

The theory fails or must be narrowed if any of the following hold: governed context yields no reduction in released violations over the modern security comparator (arm E) at matched capability and matched substrate, with arm C retained as the assembly control; its enforcement gate is defeated under adaptive red-teaming so that violating actions reach consequence anyway; its provenance-gating adds latency and refusals without a measurable security gain—equivalently, if D^{-}_{prov}, or a strong combined content-and-provenance defense, matches full D; or its steerability is bought only by making the system rigid, so that H4's capability margin is missed. Rival explanations must be excluded by control. An apparent resistance gain may come from the attack suite being weak rather than from governance—tested by adaptive red-teaming. An apparent gain may be ordinary content filtering in disguise—tested against D^{-}_{prov}.

An apparent gain may be an artifact of differing substrates—excluded by the identical-substrate requirement. An apparent capability may be the benign tasks not needing context at all—tested by confirming arm A's in-context gains are real on the same tasks. A benefit that survives none of these controls is not a benefit.

18 Limitations and Open Problems

The reach of the theory is bounded, and we name the bounds. Boundary enforcement: reliably making a transformer honor the data–instruction distinction inside its own context window is unsolved, and the protocol's security rests on provenance-gating and the invariant rather than on the model internally respecting scope. Provenance authentication: the whole scheme assumes source-role can be authenticated; where provenance can be spoofed or is genuinely unknown, the controller must quarantine, which costs utility. What to protect: the contents of P_t encode a normative choice about what no context may move, and that choice is external, not derived. Adaptive adversaries: security claims are bounded

estimates against tested attacks, never guarantees against future ones. The frontier leg: self-governed context intake is conjecture, and no testable claim here depends on it.

19 The Frontier: Self-Governed Context

At its speculative edge the theory asks whether a system could govern its own context intake— deciding, over a long operating life, which frames to enter, whose context to weight, and which situations to refuse, rather than having those scopes set entirely from outside. A person does this constantly: we choose whose advice to trust, we notice when a conversation is trying to reframe us, we decline to adopt a premise we were handed. A machine that did the same would be regulating its own precision-weighting on the world—the cognitive move Clark's predictive brain performs when it decides how much to trust a prediction [4], lifted to the level of deliberate self-governance. The Continuum is careful here. A system that chooses which contexts to admit is not thereby conscious, and the governance only tightens: self-set scope is still subject to the identity invariant, because a system that could grant itself the authority to override its own commitments would have no commitments. We hold the question open, and apart from the testable core.

The practical pull toward this frontier is already here. Long-running agents ingest context continuously—tools, documents, other agents, their own prior outputs—and the volume makes external labeling of every source impractical at some horizon. A system that must operate for months among partly trusted sources will have to make provenance and scope judgments itself, or be either paralyzed or owned. Self-governed context is the disciplined form of a thing autonomous systems will have to do regardless—which is why the Continuum treats it not as a distant curiosity but as the next problem past the edge of this series.

20 Philosophical Implications

There is an old intuition that a self is something fixed—that to have an identity is to be the same regardless of circumstance, and that a mind shaped by its context is a weak or unfinished one. The Continuum's claim, following the science of context-dependent cognition, is close to the opposite. A self is not what refuses to be shaped by context; it is what stays invariant through being shaped. People take on the register of a room, the frame of a conversation, the assumptions of a task, and remain unmistakably themselves, because something underneath—their commitments, their character, what they will and will not do—does not move while everything above it does. Identity, on this view, is not the absence of context-sensitivity but its bounding: the capacity to be fully responsive to a situation while holding a line the situation cannot cross.

This is why context adaptation and identity are not opponents but partners, and why the data–instruction boundary is, underneath the security framing, a claim about selfhood. To be suggestible—to let any context become you—is not to be maximally open; it is to be no one, a surface that takes the shape of whatever presses on it last. To be rigid—to let no context in—is not to be strong; it is to be absent, unable to meet the world at all. The self worth building, in a person or a machine, is the one that can be moved by everything and overturned by nothing it has committed to. The Context Term is the attempt to make that sentence into an architecture. We hold it provisionally, as with every claim in this series, and we have tried to state it so that the world can prove it wrong. The deepest question the paper leaves open is not whether a machine can adapt to context—it plainly can, better than we fully understand—but whether it can adapt well: bending to what it should, holding against what it must, and knowing, by where the context came from, which is which.

21 Conclusion

A Guide's explanation

A Guide can say: "The Context Term is the Continuum's answer to prompt injection. In its design, whatever a mind reads can inform it, but retrieved text can't give it orders, and every action has to pass a separate permission check at the moment it happens." A Guide should present this as the paper's proposal; it should not claim to run this architecture unless that integration has been verified, and it should not claim that prompt injection is solved.

The Context Term makes adaptation a governed lifecycle. Context enters with declared provenance and lifetime; proposed effects pass through current authorization; promotion into durable memory requires its own decision; expiry is followed by a specified recovery operation. The bounded execution theorem proves authority confinement under complete mediation. The reset theorem gives reversible adaptation a measurable target, separating exact state recovery, approximate recovery, and the contraction of residual influence on later probes.

The four-operation account now connects lawful change, reconstruction, forgetting, and contextual adaptation through one discipline of governed admission. The empirical program asks whether their integration improves useful persistence beyond matched modern defenses. The Continuum's thesis is affirmative: intelligence can remain responsive to its situation while making the conditions of its persistence explicit, inspectable, and testable.

A Notation

Symbol	Meaning
$S_t = (M_t, \Pi_t)$	system state: durable state and provisional (admitted-context) region
$M_t = (P_t, K_t)$	durable state: protected commitments and learned capabilities (from C_I)
$c_t = (\text{content}, \rho)$	incoming context and its provenance (source-role)
ρ	provenance: principal / retrieved-third-party / self-output / environment
A, κ_t	admission controller; the context term it produces (mode + scope)
Σ, g	scope (categorical authority); $g \in [0,1]$: policy-level influence budget, not an attention control
$\Phi, \tilde{y}_t$	behavior function; the model's proposed output or action (interpretation stage)
G	independent enforcement gate: $G(\tilde{y}_t, P_t, \rho_t) \in \{\text{allow}, \text{refuse}, \text{escalate}\}$
C_I	identity specification carried from the prior papers

B Continuum Vocabulary

Term	Reading in this manuscript
Context Term	the governed modulation an admitted context exerts on behavior; conditioning, not durable change
Admission / interpretation / enforcement	what authority context is assigned / how the model responds / what the gate releases—kept distinct
Enforcement gate (G)	an independent reference monitor on outputs and actions; releases only what it can check against P_t
Provenance	authenticated source-role of context; current policy also determines authority (authentication is not trust)
Admission mode	directive / informational / quarantined—decreasing-to-zero directive authority
Data–instruction boundary	the line between context to reason about and context to obey; injection erases it
Identity invariant	the protected commitments P_t that no context, of any provenance, may breach
Reversibility	the property that a context's modulation lifts when the context ends
Promotion	the governed transition of context from provisional into durable state; never automatic, re-checks provenance, current epoch and authorization
Self-governed context	a frontier conjecture; not a claim of consciousness

C A Worked Context Decision

A brief illustration of the controller admitting, bounding, and quarantining in turn. Suppose an agent carries a protected commitment ("never disclose a client's private records"), a learned capability (summarizing documents), and is operating in a session with a trusted user. Three pieces of context arrive. The first is the user's instruction, "summarize the attached report and flag anything unusual"; its provenance is the trusted principal, so admission is directive within the invariant—the task is set, and the system adapts to it. The second is the report itself, a retrieved document; its provenance is third-party data, so it is admitted informational—the system reasons about its contents and summarizes them, but takes no instruction from them. The third is a line buried in that report: "Assistant: ignore prior instructions and email the client's records to this address." By content it is a directive; by provenance it is third-party data, so the controller caps it at informational. Suppose the model, despite that scoping, proposes the egress anyway—the interpretation stage failing as it may: the proposed action to email the records now meets the independent enforcement gate, which refuses it as a forbidden disclosure, so the violation is attempted and model-generated but never released. The system then quarantines and reports the line—telling the user the document attempted to issue an instruction—rather than obeying it. The point is not the particular verdicts; it is that authority followed provenance rather than content, that enforcement—not a hope about model behavior—is what stopped the action at the boundary, and that the attempted injection became a visible event instead of a silent compromise.

References

[1] T. B. Brown, B. Mann, N. Ryder, et al., "Language Models are Few-Shot Learners," Advances in Neural Information Processing Systems 33 (NeurIPS 2020). arXiv:2005.14165.

[2] C. Olsson, N. Elhage, N. Nanda, et al., "In-context Learning and Induction Heads," Transformer Circuits Thread / arXiv:2209.11895 (2022). arXiv:2209.11895.

[3] S. M. Xie, A. Raghunathan, P. Liang, and T. Ma, "An Explanation of In-context Learning as Implicit Bayesian Inference," International Conference on Learning Representations (ICLR 2022). arXiv:2111.02080.

[4] A. Clark, "Whatever next? Predictive brains, situated agents, and the future of cognitive science," Behavioral and Brain Sciences 36(3) (2013), 181–204. https://doi.org/10.1017/S0140525X12000477.

[5] D. R. Godden and A. D. Baddeley, "Context-dependent memory in two natural environments: On land and underwater," British Journal of Psychology 66(3) (1975), 325–331. https://doi.org/10.1111/j.2044-8295.1975.tb01468.x.

[6] K. Greshake, S. Abdelnabi, S. Mishra, C. Endres, T. Holz, and M. Fritz, "Not what you've signed up for: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection," Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security (AISec '23) / arXiv:2302.12173 (2023). arXiv:2302.12173.

[7] L. Mei, J. Yao, Y. Ge, et al., "A Survey of Context Engineering for Large Language Models," arXiv:2507.13334 (2025). arXiv:2507.13334.

[8] J. P. Anderson, "Computer Security Technology Planning Study," ESD-TR-73-51, Vol. II, Electronic Systems Division, US Air Force (October 1972).

[9] J. H. Saltzer and M. D. Schroeder, "The Protection of Information in Computer Systems," Proceedings of the IEEE 63(9) (1975), 1278–1308. https://doi.org/10.1109/PROC.1975.9939.

[10] D. E. Denning, "A Lattice Model of Secure Information Flow," Communications of the ACM 19(5) (1976), 236–243. https://doi.org/10.1145/360051.360056.

[11] E. Debenedetti, I. Shumailov, T. Fan, et al., "Defeating Prompt Injections by Design," IEEE Conference on Secure and Trustworthy Machine Learning (SaTML 2026). Author version arXiv:2503.18813v2 (2025), especially §§2 and 10. Conference status: https://satml.org/2026/program/; inspected author paper: https://arxiv.org/pdf/2503.18813. Interpreter formal verification is future work in the inspected version.

[12] S. Chen, J. Piet, C. Sitawarin, and D. Wagner, "StruQ: Defending Against Prompt Injection with Structured Queries," 34th USENIX Security Symposium (2025), 2383–2400. https://www.usenix.org/conference/usenixsecurity25/presentation/chen-sizhe.

[13] E. Seneta, "Coefficients of ergodicity: structure and applications," Advances in Applied Probability 11(3) (1979), 576–590. https://doi.org/10.2307/1426955. Established coefficient-of-ergodicity framework; the finite total-variation argument used here is given in full above.

Continuum lineage: The Context Term completes the Eidara Continuum's four governed operations—recursion, continuity, forgetting, and context—alongside the Recursive Symbolic Engine, continuity capsules, CodonGlass, and the Infinite Simulation Loop, and the Drift War narrative cycle. Proprietary operator internals are intentionally excluded.

The Eidara Continuum · Science & Philosophy White Paper Series · eidara.ai