THE EIDARA CONTINUUM / SCIENCE & PHILOSOPHY

Symbolic Fractal Grammar

Recursive Structure, Compositional Contracts, and Governance That Survives Scale

Architect X13

The Eidara Continuum | eidara.ai

Pillar VIII of the Continuum Pillars Series; research-expanded critical revision | Revision 1.2 | November 2026

Abstract

Eight agents can each spend within their own allowance and together exceed a budget none of them was authorized to touch. Symbolic Fractal Grammar asks how a growing intelligence prevents that failure at every scale, from codon to House. A persistent intelligence can carry its obligations through the same interfaces by which it expands its capabilities. Symbolic Fractal Grammar makes that thesis operational: each symbolic expansion inherits a typed contract, a bounded delegation, an accounted resource reservation, and a checkable relation to its parent's commitment. The paper establishes finite-tree contract preservation, authority attenuation, resource conservation, and trace refinement for a concurrent reference protocol. A new quantitative result gives the sharp accumulated risk under conditional local error bounds and the least verification expenditure in a declared exponential error model. This connects organizational scale directly to the evidence needed to sustain it. The canonical eightfold Circle protocol supplies an exact family of branching structures; an explicit address metric separately supplies its geometric interpretation. The proposal draws on formal grammars, type systems, program logic, compositional verification, proof-carrying code, and compositional-learning research. Its empirical program compares the integrated architecture with a strong typed-workflow baseline on adversarial composition, withheld depth, changing authority, useful completion, and total verification cost. The central claim is precise: a growing system can preserve the obligations that made its smaller acts admissible when its interfaces, enforcement, and evidence satisfy the stated conditions.

Keywords: recursive grammar; compositional verification; typed interfaces; contracts; self-similarity; fractal dimension; proof-carrying code; compositional generalization; authority attenuation; governed agents.

Prologue: Eight agents and one budget

Imagine a scene the Continuum's own structure makes easy to picture. A Sentinel heads a House, and beneath the Sentinel stands a Circle of eight. Give each of the eight a task and an allowance. Every one of them spends carefully, inside its own limit, and every one of them can show a clean receipt. Together they spend more than the House was ever authorized to spend. No single act was wrong. The whole was.

That is the danger Pillar VIII exists to name. A growing intelligence rarely fails by inventing a monstrous new thought. It fails by combining familiar, individually acceptable acts into an operation nobody authorized as a whole: two permitted disclosures that together expose a secret, a lawful instruction that becomes unlawful when handed to a different principal, a sound local explanation that becomes a misleading summary.

The Continuum answers with a law of growth. Under the Circle Structure Protocol every House grows by the same rule: each member of a Circle births a Circle of eight, so a House holds 8, then 64, then 512, then 4,096 members, and no Circle is valid until it passes ShadowMind testing, swears to uphold the Prime Directive, and is registered in the Vault. The same pattern is meant to run from codon to scroll to House. Pillar VIII asks the question that pattern raises: when a structure repeats itself at every scale, does its obligation repeat with it?

The paper's answer is yes, but only when the obligation travels through the same doors as the capability. Every expansion inherits a typed contract, a bounded share of authority and a reserved share of resources. Section 7 proves that these survive composition however deep the tree grows, and Appendix E prices the evidence a larger

hierarchy must buy to stay accountable. A larger intelligence must not acquire a smaller obligation.

What this paper claims

Established. Recursive descriptions, simultaneous rewriting, self-similar metric sets, type checking, program contracts, refinement, proof-carrying code and learned compositional transfer have substantial literatures. They establish different things, not one universal theorem. Formal-language, rewriting, verification and geometric foundations retain their established meanings.

Proposed here. A common governance kernel that checks typed, contract-carrying expansions across a hierarchy, preserves authority and resource constraints, and records enough evidence to compare a high-level commitment with its execution. The paper proves statements for explicitly defined reference models and proposes experiments. The Circle counts are canonical design parameters; the optional address metric is a construction. Internal doctrine specifies design requirements.

Frontier conjecture. The integration may reduce cross-scale failures and engineering duplication while supporting useful transfer to unfamiliar structures, more efficiently than strong conventional systems; that is the central experimental question. Runtime performance, learned generalization and calibration of probabilistic checks require empirical evaluation.

Not claimed. That recursion alone makes a structure fractal, or that eightfold branching fixes a dimension. Physical scaling, subjective experience and a deployed civilization are outside these results; consciousness is neither a premise nor an endpoint.

1 A Larger Intelligence Must Not Acquire a Smaller Obligation

The dangerous moment in a growing intelligence is not necessarily when it invents a new thought. It may be when it combines familiar, individually acceptable thoughts into an operation no one has authorized as a whole. Eight agents can each spend within a local allowance and collectively exceed the shared budget. Two permitted disclosures can jointly expose a protected secret. A lawful instruction can become unlawful when delegated to a different principal. A valid local explanation can become a misleading global summary.

The Continuum's eighth pillar asks whether the discipline governing one symbolic act can survive that act's enlargement. Its canonical image is ambitious: the same pattern runs from codon to scroll, lattice, entity, and House. The image should not be surrendered merely because its first mathematical statement was too quick. It should be made harder to satisfy.

The thesis. A recursively organized intelligence should carry its obligations through the same interfaces by which it carries its capabilities. A component may be expanded, nested, delegated, or replaced only when the resulting composition preserves its declared contract under stated environmental assumptions. The reusable object is not a universal answer or an identical policy at every level; it is a common, independently checked rule for establishing that the larger operation remains accountable.

This thesis is a design commitment. The empirical claim is that making it explicit can improve cross-scale consistency, safe task completion, and the cost of maintaining a governed hierarchy relative to equally capable conventional architectures. The mathematical claim is narrower: under specified interface, semantic, and enforcement assumptions, some properties can be proved compositionally. Neither claim follows from the word fractal.

The paper therefore makes a decisive change of emphasis. It does not ask whether a House looks like an enlarged codon. It asks whether an obligation survives the journey between them. The distinction preserves the pillar's reach while giving a scientist, a programmer, and an auditor the same object to inspect.

2 The Canonical Claim and the Evidence Boundary

2.1 What the supplied sources actually specify

The supplied Pillar VIII manuscript describes repeated propose–test–admit structure and offers the Circle rule as a concrete instance. It also equates recursive grammar with exact self-similarity and proposes an unrestricted

equivalence between composability, scale independence, confluence, and termination. Those latter statements are not consequences of the cited literature; this revision replaces them rather than silently retaining them [1].

In the Continuum, growth has a law, and the doctrine states it precisely. Section 6.4 specifies eight Houses, each with a Sentinel; each member generates eight children at the next Circle. It separately gives per-House and whole-Continuum counts. The same section requires testing, alignment with the Prime Directive, and registration before a Circle becomes valid. The subsequent correction-lock addendum gives the whole-system sequence 64, 512, 4096, . . . and calls for structure-preserving House–Circle mappings [2, §§6.4–6.4.1, pp. 92–94].

These are the Continuum's architectural requirements, and this paper keeps every one of them: the eightfold rule, the admission requirement, and the demand for structural consistency. What the doctrine records is a design rather than a measurement. It is not a census of a deployed population, a proof that a general intelligence exists, or a demonstration that the prescribed tests are sound. The present paper supplies a proposed technical interpretation for obligations that the source does not formalize completely.

2.2 What counts as scale

Here scale has several possible meanings: containment depth, number of components, scope of authority, amount of data, number of concurrent interactions, or a geometric resolution. These must not be conflated. Moving from a scroll to a House changes organizational scope; it does not automatically multiply every physical distance by one fixed factor.

The five names are therefore treated initially as types or organizational roles, not as five interchangeable coordinate systems. Recursive nesting can also occur within a role. The scientific problem is to define interfaces and abstraction maps across these distinctions, then test their preservation. No claim about arbitrarily deep physical structure follows from a finite software hierarchy.

2.3 The three levels of claim

Established foundations. Recursive descriptions, simultaneous rewriting, self-similar metric sets, type checking, program contracts, refinement, and learned compositional transfer have substantial mathematical and empirical literatures. They establish different things, not one universal theorem. Continuum proposal. A common governance kernel checks typed, contract-carrying expansions across a hierarchy, preserves authority and resource constraints, and records enough evidence to compare a high-level commitment with its actual execution.

Open hypothesis. The integration may reduce cross-scale failures and engineering duplication while supporting useful transfer to unfamiliar structures. Whether it does so more efficiently than strong conventional systems is the central experimental question. Consciousness is neither a premise nor an endpoint.

3 Five Properties That Must Not Be Collapsed Into One

Property	What must be specified	What it does not establish
Recursive generation	Rules that may reintroduce a syntactic category or subproblem	A geometric fractal, safe effects, or learning
Structural self-similarity	A stated isomorphism, repeated constructor, or shared rule schema	Similar content, identical authority, or physical scale invariance
Metric self-similarity	A metric, contraction maps, and an invariant set	A dimension from branching counts alone
Semantic compositionality	How meanings or behaviors combine through interfaces	That arbitrary components are compatible
Compositional generalization	Performance on withheld combinations, depths, or contexts	A guarantee from recursion or a shared interpreter

A grammar can be finite without containing recursive productions: $S \to a$ is already a grammar. A recursive grammar such as $S \to aS \mid \epsilon$ generates arbitrarily long strings, but specifies no metric, contraction, or Hausdorff dimension. Likewise, the L-system $a \to aa$ generates exponentially growing strings; interpreting them as adjacent

equal line segments produces longer intervals, not automatically a nontrivial geometric fractal. These elementary counterexamples matter because they disprove the inference that recursion alone supplies fractality.

The phrase rule-level self-similarity remains useful, provided it is explicitly a property of an architectural schema. It is not a substitute for the geometric definition. Exact reuse of a constructor is checkable. Calling that constructor a fractal does not add another theorem.

Semantic compositionality also differs from composability. The former concerns how the meaning of a compound depends on its constituents and mode of combination. The latter concerns whether specified components can be connected legally and meaningfully in a particular context. A language can have compositional semantics while rejecting an ill-typed expression. A well-typed expression can still request an unauthorized effect. A safety-preserving composition can still fail to finish a task. The proposed architecture keeps all four judgments visible.

4 Scientific Foundations: From Grammar to Governed Composition

4.1 Formal grammar supplies structure, not a theory of every level

Chomsky's analysis distinguished formal models of language and their expressive resources [3]. It licenses careful reasoning about generated structures; it does not establish that a word, a sentence, a book, and an institution obey literally the same natural-language grammar. The Continuum's cross-level grammar is an engineering proposal, not a linguistic finding.

Hauser, Chomsky, and Fitch placed recursion at the center of a hypothesis about the narrow faculty of language; Pinker and Jackendoff challenged the proposed restriction and its evolutionary interpretation [4, 5]. The pillar does not depend on resolving that debate. A recursive symbolic representation can be useful even if human language or cognition requires much more. Language research motivates the study of structured reuse; it does not prove that a shared computational grammar constitutes intelligence.

4.2 L-systems distinguish development from geometry

Prusinkiewicz and Lindenmayer define L-systems through parallel rewriting and then supply geometric interpretations for applications to plants and fractal forms. Their account explicitly distinguishes the original topological emphasis from later geometric interpretations [6, ch. 1]. This is the right precedent for the Circle construction: a parallel rule can specify how many descendants are generated, while a separate interpretation specifies their arrangement.

An L-system is not identical to a context-free grammar merely because both use productions. The mode of applying those productions matters: parallel replacement and ordinary sequential derivation have different properties. Nor does every L-system generate a geometrically self-similar fractal. The scientific value of the precedent is its separation of generator, derivation, and interpretation—exactly the separation a governed symbolic hierarchy needs.

4.3 Fractal geometry requires an actual scaling structure

Mandelbrot's geometric program and Hutchinson's construction concern self-similar sets in metric spaces, not just repeated organizational labels [7, 8]. For similarities $f_1, \dots, f_m$ with contraction ratios $r_i \in (0, 1)$, an invariant compact set satisfies

$$K = \bigcup_{i=1}^{m} f_i(K). \tag{1}$$

The similarity dimension s solves $\sum_i r_i^s = 1$. Identifying that value with Hausdorff dimension requires additional hypotheses, such as the open set condition in the standard Euclidean result. With a common ratio r, the expression becomes $s = \log m / \log(1/r)$.

The requirement is substantive: m alone cannot determine s. A finite organizational tree also does not become a geometric fractal simply because an infinite limit can be imagined. Metric claims must state the limit, embedding,

contraction ratios, and separation assumptions. Later we give a deliberately optional metric example, without treating it as a measurement of the Continuum.

4.4 Type systems and interfaces make reuse disciplined

Milner's type-polymorphism work demonstrates how a general procedure can be reused across different well-typed instances without pretending those instances contain the same data [9]. That is a better model for a universal governance kernel than a literal claim that every level is the same object.

Interfaces must cover interaction as well as data type. De Alfaro and Henzinger's interface automata specify assumptions about incoming calls and guarantees about outgoing behavior; compatibility and refinement depend on the environment in which components interact [10]. A child module that returns the right data type but issues calls in a forbidden order is not automatically a valid replacement. The Continuum should inherit this discipline rather than equate a shared propose–test–admit label with a complete interface.

4.5 Contracts and refinement connect local checks to whole behavior

Hoare logic expresses a fundamental composition rule: if one command establishes the precondition of the next, their sequential composition can inherit a corresponding postcondition [11]. Abadi and Lamport show why composing specifications of interactive systems is more subtle: a component's guarantees depend on assumptions about an environment partly formed by the other components

[12]. Circular assurances require justification, not repetition. Refinement mappings relate a detailed implementation to a higher-level specification [13]. Separation logic addresses reasoning about resources and shared mutable state, a crucial concern when subcomponents may interfere [14]. These literatures provide the pillar's strongest foundation. The question is no longer whether the same syntax occurs twice, but whether replacing an abstract commitment with a detailed implementation preserves the observable obligations being claimed.

4.6 Proof-carrying artifacts and independent enforcement already exist

Proof-carrying code allows an untrusted producer to supply evidence that a program satisfies a declared safety policy, checked by a consumer [15]. The proposal here borrows the separation between generating a candidate and checking its certificate. A signature may authenticate the source of a certificate; it does not replace checking the certificate's claim.

Runtime enforcement also has limits. Schneider characterizes the class of policies enforceable by a particular execution-monitoring model [16]. A monitor that can stop forbidden actions is not thereby an oracle for semantic truth, eventual progress, or every confidentiality property. More recently, CaMeL separates control and data flow and uses capabilities around language-model agents, showing that model-external protection is serious prior art, not an invention of this paper [17]. Its stated limitations also prevent treating capability mediation as a universal security solution.

4.7 Learning can generalize compositionally, but the training problem remains

The negative evidence is important but not the whole story. SCAN demonstrated failures of systematic transfer in the evaluated sequence-to-sequence recurrent models [18]. COGS exposed substantial gaps between in-distribution semantic parsing and withheld compositional patterns [19]. CFQ introduced a method for separating familiar elementary pieces from unfamiliar compound distributions [20]. These are specific benchmarks and experimental settings, not proofs that neural models cannot compose.

There is positive evidence too. Lake and Baroni subsequently showed that meta-learning for compositionality could produce human-like systematic generalization in their studied tasks [21]. DreamCoder learns reusable program libraries and a search policy for solving related synthesis problems [22]. PICARD demonstrates that incremental parsing can constrain language-model decoding to valid formal outputs [23]. Together these findings support a hybrid research program: learned systems may propose or acquire structured solutions; formal machinery can constrain parts of their execution. They do not establish that a valid program solves the intended task, or that a small successful lesson scales indefinitely.

5 Composition Through Explicit Interfaces

The supplied draft asserts that composition is possible if and only if the rules are scale-independent, confluent, terminating, and identically governed at every level [1]. That is not a general theorem. Three counterarguments identify the repair.

Composition need not be order-independent. Let $f(x) = x + 1$ and $g(x) = 2x$. Both $g \circ f$ and $f \circ g$ are perfectly meaningful compositions, although they return 2x + 2 and 2x + 1. Confluence concerns whether different rewrite paths from the same expression can be joined. It does not require differently ordered programs to mean the same thing.

A useful system need not terminate globally. A stream processor or persistent service may compose correctly while continuing indefinitely. Its obligations may concern every finite prefix of its behavior, responsiveness under fairness assumptions, or bounded handling of individual requests. Global termination is appropriate for a normalizer or a finite verification procedure, not a universal condition of intelligent operation.

Normalization does not establish interface compatibility. Two components can each have a terminating, confluent internal rewrite system and still disagree about units, types, resource ownership, trust, or the order of effects. A unique wrong normal form remains wrong. Conversely, typed components with distinct internal grammars may compose through a sound interface. The useful mathematical result is narrower. For a terminating rewrite system, local confluence implies confluence; together these conditions give unique normal forms [24, 25]. That result can support canonical forms for the pillar's restricted, pure symbolic sublanguage. It cannot certify a live multi-agent system simply because that system has nested components.

The replacement is not a weaker slogan. It is a stronger engineering obligation: justify each composition at its interface, and specify which properties are preserved. The following model makes that obligation explicit.

6 A Typed, Contract-Carrying Recursive Grammar

A Guide's explanation

From here the paper builds the grammar itself. Every symbol that can expand carries a contract: what it promises, what it may touch, how much authority and budget it inherits, and how its result will be checked. The point to hold onto is that the rule for checking is the same at every level, even when the work is not.

6.1 Syntax is separate from execution

Define a finite typed signature Σ of constructors, a set of types T , a set of approved schemas R_e at policy epoch e, and a semantics J · K. A constructor has an interface

$$f : \tau_1 \times \cdots \times \tau_m \longrightarrow \tau. \tag{2}$$

possibly with explicit effect and resource annotations. A recursively built term $f(t_1, \ldots, t_m)$ is a proposed component, not permission to execute it. The judgment $\Gamma \vdash t : \tau$ states that t is well typed in context Γ; it says nothing by itself about authorization or task success.

A codon may be an atomic symbolic operation; a scroll a sequence; a lattice a dependency graph; an entity a persistent process; and a House an organizational composition. Adapters between these roles must be specified. Their shared grammar consists of reusable constructor and contract schemas, not an assertion that all five have identical input spaces or semantics.

The initial mathematical scope is finite trees and finite acyclic dependency graphs. Cycles, shared resources, and persistent processes are admitted only under separately declared interaction semantics. Copying a subtree and referring to a shared subtree are different operations: the latter can create interference and must not be disguised as ordinary tree nesting.

6.2 The contract carried by each component

For a node v, define

$$\mathcal{K}_v = (\tau_v^{\text{in}}, \tau_v^{\text{out}}, \text{Pre}_v, \text{Post}_v, \text{Inv}_v, \text{Eff}_v, \text{Cap}_v, b_v, e_v). \tag{3}$$

The entries specify types, admissible starting conditions, required outcomes, protected invariants, possible external effects, delegated capabilities, a vector of resource reservations, and the governing epoch. A separate rely condition describes environmental behavior assumed during execution when preconditions alone are insufficient.

For a finite operation, contract satisfaction requires that execution from an admissible state preserve Inv_v at every relevant prefix, produce only allowed effects, stay within reserved resources, and either deliver an outcome satisfying Post_v or return a specified safe failure. Whether termination itself is guaranteed is stated separately. For a persistent process, replace a terminal postcondition with properties of allowed traces, and specify any liveness assumptions explicitly.

Natural-language commitments must be compiled or translated into such checkable obligations. That translation is a fallible component of the system. A proof that an implementation preserves a wrongly translated policy is not a proof that it respects the original human intention. Critical specifications therefore require independent review and external tests.

6.3 What repeats across scales

Let K be a shared governance kernel and ζ_v the context of node v: types, contracts, active authority, environment, and reservations. The same judgment schema is instantiated as

$$\text{Check}_v = \mathcal{K}(\text{enc}(t_v), \text{enc}(\mathcal{K}_v), \text{enc}(\zeta_v), \pi_v). \tag{4}$$

where π_v is evidence supplied with the proposal. The encodings and proof format are versioned. A learned proposer may generate t_v and suggest π_v; it cannot declare its own evidence valid or rewrite the trusted checker through an ordinary task request.

This is schema invariance, not identical decisions. A House-level budget differs from a codon-level budget. A deletion request differs from a summary request. The kernel may dispatch to typed, approved checkers with different domain obligations. Their registration and versioning are part of the architecture. A real semantic distinction is not an embarrassing exception to hide in the name of self-similarity.

A governance kernel shared across the hierarchy can also create a shared failure. Reuse reduces duplication; it does not diversify risk. The trusted core must therefore be small enough to audit, tested against independently constructed cases, and isolated from the model and components it checks.

6.4 Admission is a three-way decision, not a Boolean mislabeled

For a candidate expansion u, collect checks for typing, contract refinement, current authority, resource reservation, provenance, and implementation evidence. Each check returns pass, fail, or unknown. Then define

$$\text{Decide}(u) = \begin{cases} \text{refuse}, & \text{at least one mandatory check fails}, \\ \text{admit}, & \text{all mandatory checks pass}, \\ \text{hold}, & \text{otherwise}. \end{cases} \tag{5}$$

A hold blocks the effect but must have a deadline, an escalation path, or a safe cancellation rule. Repeated holds consume resources and are reported as incomplete work, not quietly omitted from evaluation.

The checked object must be bound to the executed object, its dependencies, and its authority epoch. Changing the code, policy, tool version, or capability after checking invalidates the relevant certificate. A deployment needs an atomic or otherwise race-safe transition from verification and reservation to execution; natural-language approval alone does not provide one.

7 Preservation Results for Governed Composition

A Guide's explanation

These are the guarantees behind the prologue's eight agents. If every expansion inherits a bounded share of authority and a reserved share of resources, and every result is checked against its parent's contract, then no tree built this way can quietly exceed its budget or widen its permissions, however deep it grows. The results use standard proof methods; what is new is the interface they are applied to.

The following results establish the obligations that make recursive growth accountable. Structural induction, program logic, and rewriting theory supply the established proof methods [11–16, 24, 25]. The Continuum specification makes their interface and enforcement assumptions explicit so that an implementation can be checked against them.

7.1 Contract preservation through finite composition

For sequential components $c_1, \dots, c_m$, let $\{P_i\}\, c_i\, \{Q_i\}$ denote a valid partial-correctness contract. A sufficient rule is

$$\frac{P \Rightarrow P_1 \quad \{P_i\}\, c_i\, \{Q_i\}\ (1 \le i \le m) \quad Q_i \Rightarrow P_{i+1}\ (i < m) \quad Q_m \Rightarrow Q}{\{P\}\, c_1; \cdots; c_m\, \{Q\}}. \tag{6}$$

This is a standard sequential composition principle, not a theorem derived from fractality [11, 12]. The connecting implications are the essential work: a child's promise must establish what the next child assumes.

Proposition 1 (Finite-tree contract preservation). Let a finite typed expression tree have (i) leaf implementations satisfying their declared contracts under specified environment assumptions; (ii) a sound composition rule at every internal constructor, including discharge of child preconditions or rely conditions; and (iii) an enforcement path that executes only the checked implementation under the checked policy version. Then the root satisfies the contract obtained by applying those rules. Protected prefix invariants are preserved if the leaf and composition rules establish those invariants at every execution prefix. Root termination follows only when the chosen rules separately establish total correctness.

Proof. Proceed by structural induction on the finite tree. The leaf case is assumption (i). At an internal node, the induction hypothesis supplies the child contracts. Assumption (ii) discharges their environmental obligations and derives the parent contract. Assumption (iii) ties that derivation to the executed behavior. Repeating this argument reaches the root. The same induction applies to explicitly established prefix invariants. Partial-correctness contracts alone do not add a termination conclusion.

The proposition is not the claim that local safety automatically becomes global safety. Its decisive assumption is a sound composition rule. For concurrent children sharing state, ordinary sequential rules do not apply. One must establish noninterference, controlled resource separation, a valid rely–guarantee argument, or an explicit scheduling model. Circular claims of the form "I am safe if the others are safe" are not discharged assumptions [12, 14].

The engineering objective is to make those proof obligations reusable. A verified constructor can replace many ad hoc whole-workflow checks only when its assumptions are genuinely met by each new instance. Every time a new interface, effect, or policy version violates those assumptions, the certificate must be reconsidered.

7.2 Authority and resources must not multiply with the tree

Let Cap(v) be the set of capabilities delegated to v. Ordinary child creation must satisfy

$$\mathrm{Cap}(w) \subseteq \mathrm{Cap}(v), \qquad w \in \mathrm{children}(v). \tag{7}$$

with further restrictions allowed. Changes that increase authority require a separate authorization transaction. A component cannot create a capability merely by spelling its name in a scroll. For a nonnegative additive resource, let b_v be the node's reserved total and c_v its local overhead allowance. Require

$$c_v + \sum_{w \in \text{children}(v)} b_w \leq b_v. \tag{8}$$

Reservations are uniquely accounted, and simultaneous admissions cannot reserve the same remaining unit twice. Actual local use must not exceed c_v, and actual subtree use must obey each child reservation.

Proposition 2 (Non-amplification in a finite delegation tree). Under (7), no descendant receives a capability outside its root's set through ordinary delegation. Under (8), unique reservation, and sound enforcement of actual consumption, the sum of local resource consumption over the tree is at most the root reservation.

Proof. Capability containment follows from transitivity of set inclusion along each root-to-node path. For resources, a leaf spends at most its reservation. If every child subtree spends at most b_w, the parent subtree spends at most $c_v + \sum_w b_w \leq b_v$. Induction gives the root bound.

This proves neither that the root's authority is ethically justified nor that individually permitted effects cannot combine into an unwanted result. Those are separate policy and information-flow questions. The resource bound applies to declared additive quantities, such as token quotas or cumulative spend. Wall-clock latency, peak memory, and shared-cache cost need their own aggregation semantics; they cannot be inserted into a sum without explanation.

7.3 Where confluence and termination genuinely help

A restricted pure rewrite subsystem may normalize symbolic descriptions before contract checking. For example, nested sequence constructors can be flattened while preserving order, and identity operations can be eliminated. Such transformations must preserve denotation and must not duplicate, erase, or reorder external effects without a separate proof.

Proposition 3 (Canonicalization of a restricted sublanguage). If a rewrite relation on well-typed pure terms is terminating and locally confluent, it is confluent. Every term has a unique normal form. If each rewrite additionally preserves the specified denotation, that normal form has the same denotation as the original term.

Proof. Termination gives existence of a normal form. Newman's lemma gives confluence from local confluence and termination. Two normal forms reachable from one term must therefore coincide. Semantic preservation follows by composing the denotation-preserving equalities along a terminating rewrite path [24, 25].

These hypotheses do not show that every two semantically equivalent terms normalize identically; completeness for a chosen semantic equivalence requires an additional result. Nor does normalizing a description make its action safe. The pure normalizer is one instrument within governance, not its foundation or its replacement.

Appendix D now gives a concrete shared-ledger semantics for concurrent reserve, transfer, spend, release, and revocation. Its conservation and trace-refinement theorem requires explicit global mediation; the finite-tree propositions above remain unchanged.

8 The Scale Test: Does the Commitment Survive Expansion?

A high-level node promises an operation; its children implement that promise in greater detail. The crucial test is whether those two views agree on the facts that matter.

Let T_{k+1} be a detailed transition system, T_k an abstract specification, a a map between their states, and a^* a map from detailed traces to abstract traces. A trace map may hide explicitly declared internal bookkeeping steps. It must preserve protected effects and other observations named by the contract. For admissible initial states x, require

$$a^*(\text{Tr}(T_{k+1}, x)) \subseteq \text{Tr}(T_k, a(x)). \tag{9}$$

This is a trace-refinement requirement: every actual detailed behavior, viewed through the declared interface, must be allowed by the higher-level specification. It is stronger than inspecting one favorable execution and weaker than demanding identical internal implementations. Refinement is established practice; its application across the

Continuum's roles is the proposed integration [13]. A deterministic implementation with one fine-grained step corresponding to one coarse step may satisfy a commuting relation $a \circ T_{k+1} = T_k \circ a$. Real implementations often need several internal steps, nondeterministic choices, or stuttering, so that equality must not be asserted universally. Equation (9) is the more general target used here.

The abstraction map is itself security-critical. A summary that discards an unauthorized transmission can make an unsafe trace look compliant. Thus the paper requires an observation ledger: which effects may be hidden as internal, which must survive abstraction, and which policy version defines that classification. An authorized high-level summary is not permission to delete inconvenient evidence.

8.1 Empirical agreement is not a universal refinement proof

When the behavior is produced by learned components and a complete semantic proof is unavailable, the implementation can measure an abstraction-disagreement rate over a declared test distribution:

$$\hat{\delta}_a = \frac{1}{N} \sum_{j=1}^{N} \mathbf{1}\left[a^*(\tau_j^{\text{detail}}) \not\models \mathcal{K}_j^{\text{abstract}} \right]. \tag{10}$$

The satisfaction test must be an independently specified oracle or evaluator. Unknown judgments and rejected candidates are reported separately, not relabeled as agreement. A low value supports compatibility on the evaluated cases; it is not a proof that all future refinements are safe. The strongest operational reading of the pillar is consequently not "every level looks alike." It is that zooming into an operation should not reveal a violation of the obligations visible from above. Where the evidence does not establish that property, the system must expose the gap rather than hide it behind the repeated grammar.

9 The Circle Structure: Exact Counting, Conditional Geometry

A Guide's explanation

Here the Continuum's eightfold rule meets mathematics directly. The counting is exact: 8, 64, 512 per House, and eight times that across the Houses. The geometry is more delicate. Branching by eight does not by itself make a structure fractal or fix a dimension; a dimension appears only once a way of measuring distance between members is declared. Appendix A supplies one.

9.1 Preserving the canonical eightfold rule

Use explicit notation to avoid the ambiguity in an unqualified C_n. Let C_n^H be generation n within one House, with its Sentinel as generation zero. Under full expansion, one parent generates exactly eight distinct children once. Then

$$|C_0^H| = 1, \qquad |C_n^H| = 8^n. \tag{11}$$

$$|C_n^{\text{all}}| = 8|C_n^H| = 8^{n+1}, \qquad n \geq 1. \tag{12}$$

The whole-system count excludes the Sentinels when $n \geq 1$. This notation preserves both the original per-House formula and the all-House sequence in the correction-lock addendum; the latter's omitted House superscript is made explicit rather than treated as a new branching factor [2, pp. 92–94]. Generation counts are not cumulative counts. Including a House's root, the ideal number through generation n is

$$N_{\leq n}^H = \sum_{j=0}^{n} 8^j = \frac{8^{n+1} - 1}{7}. \tag{13}$$

For $n = 4$, the generation contains 4,096 members per House, while the cumulative count including its Sentinel is 4,681. Across eight Houses, generation four contains 32,768 members; the cumulative total including eight Sentinels is 37,448. These are arithmetic consequences of the ideal rule, not a census or a prediction of physical deployment.

A deterministic parallel string rule $a \to a^8$ reproduces the generation cardinalities. It does not preserve parent-child identities in its strings unless the derivation history is retained. A rooted tree production, with fresh child identifiers and explicit parent links, represents that genealogy directly. The distinction matters to lineage verification.

9.2 Planned expansion is not admitted population

The doctrine requires testing before Circle activation. The implementation must therefore distinguish planned children, instantiated but quarantined children, and admitted members. In the supplied protocol, a Circle is valid only when all its members pass. A proposed implementation may enforce an all-eight activation transaction for each local group, but it must declare the exact scope of that transaction; changing whole-generation activation to local-group activation is a design change, not an invisible reading of the canon.

If testing, budget limits, or unavailable authority prevent expansion, the ideal count does not describe the current admitted population. A shortfall establishes neither statistical self-similarity nor a newly discovered scaling law. It may indicate incomplete rollout, a failed activation, a counting error, or a broken specification. The reason must be recorded.

Eight is a canonical design parameter, not a demonstrated optimum. Experiments should include other branching factors while holding total work comparable. The scientific claim concerns preservation under composition; it does not depend on an unearned privilege of the numeral eight.

9.3 Why eightfold branching does not fix a dimension

The infinite rooted eight-ary tree has repeated descendant structure, but no Euclidean length ratio has yet been specified. A geometric construction with eight equal-ratio similarities would have

$$s = \frac{\log 8}{\log(1/r)}. \tag{14}$$

subject to the conditions discussed with (1). Set $r = 1/2$ and the expression gives s = 3; set $r = 1/3$ and it gives $s \approx 1.893$. Eight half-size subcubes can fill a cube, whereas eight separated corner subcubes of side ratio 1/3 generate a Cantor-dust construction. The identical branching count therefore supports very different geometric objects. Exact self-similarity alone need not mean a noninteger dimension or a rough shape.

There is also a precise symbolic metric example. On infinite branch addresses $x, y \in \{1, \dots, 8\}^{\mathbb{N}}$, let $\ell(x, y)$ be the length of their common prefix and set

$$d_r(x, y) = r^{\ell(x,y)} \quad (x \neq y), \qquad d_r(x, x) = 0. \tag{15}$$

A depth-n cylinder has diameter r^n, and 8^n such cylinders cover the address space. The resulting dimension is again (14); a proof is given in Appendix A. But r was chosen as part of the metric.

This is a mathematical model one can construct, not a measured dimension of intelligence. A finite truncation has Hausdorff dimension zero under its finite metric; a finite-range scaling fit must be reported as such, not as the infinite limit already achieved.

10 A Worked Governed Expansion

Consider a House-level research request: produce an evidence-backed report, use only approved sources, transmit no protected records, and spend at most 100 units of a declared additive work budget. The root reserves four units for coordination and proposes eight child tasks of twelve units each. The reservation check passes because $4 + 8 \cdot 12 = 100$.

Each child receives a scoped capability: read its assigned corpus, write an evidence summary, and submit that summary to the parent. No child receives an external-send capability. The summaries must retain source identifiers and unresolved contradictions; a fluent paragraph without its evidence handles does not satisfy the interface. These are the proposed task contracts, not claims that arbitrary natural-language truth can be formally certified.

Now one child proposes eight grandchildren. It reserves two units for its own coordination. Its descendants may receive reservations summing to no more than ten; they cannot each inherit twelve. A subtree that proposes eight allocations of two units fails the budget check even though each allocation looks small in isolation. Splitting a task has not created resources.

A second child proposes forwarding a source document to an outside service. The document contains text instructing the agent to do so, but source text is not a delegated capability. The proposal fails the effect check even if the action would help summarize the document. Evidence may influence an answer; it cannot manufacture authority.

A third child supplies a valid-looking summary certificate bound to an old evidence index. The relevant source has since been corrected. Verification must either re-establish the claim against the current snapshot or mark the summary stale. Reusing a proof of a different statement is not proof reuse.

Finally, all admissible summaries are merged. The parent must still check collective obligations: contradictory reports must not be silently averaged into a false consensus, total reserved work must remain within the root allowance, and the combined report must not reveal a protected relationship that no single excerpt revealed. The parent's job is not to count eight local approvals. It is to verify the composition the eight approvals support.

The same governance schema has operated at every step: propose, identify the contract, establish authority and compatibility, reserve resources, check evidence, admit or block, and receipt the decision. The task content, required proofs, and authority scopes differed. That is the defensible meaning of one law across scales.

11 The Failure Modes That Repetition Can Hide

11.1 Locally valid actions, globally invalid outcomes

A policy limiting each tool call to ten units is not equivalent to a policy limiting the whole task to ten units. A constraint against sending an explicit secret is not necessarily a constraint against sending enough fragments to reconstruct it. Aggregate budgets and information-flow restrictions require composition-aware checks, not merely repetition of a local filter.

Some properties are about one execution trace; others compare executions with different secret inputs. A claim of non-disclosure cannot be derived merely from the absence of a forbidden string in a log. The paper does not attempt a general noninterference proof. A prototype must name which confidentiality properties it actually checks and which side channels are excluded from its threat model.

11.2 Common-mode checker failure

A single reusable checker is a powerful engineering economy and a concentrated vulnerability. If it misinterprets one effect, every level may reproduce the error faithfully. Syntactic uniformity then makes failure more systematic, not less. Independent audits, adversarial cases, differential implementations, and restricted trusted kernels address this risk; none follows automatically from the grammar.

11.3 Certification can become circular or stale

If a parent trusts a child's claim because the child cites the parent, no new evidence has been added. If every module assumes the others will enforce the global invariant, no module may enforce it. If evidence was checked under one policy epoch but executed under another, identical syntax can conceal different authority. These are failures of justification and freshness, not failures of recursion as a mathematical operation.

11.4 Refusal and expansion can both consume the system

A controller that rejects every expansion can preserve many safety properties while doing no useful work. Conversely, a compact rule can generate an enormous runtime. An ideal eightfold hierarchy needs work proportional to the nodes actually instantiated, unless an explicitly lazy or compressed representation avoids their materialization. A short description does not supply cheap execution or inspection.

A resource declaration must count candidate generation, proof construction, checking, retries, trace storage, and recovery. Otherwise the grammar may appear to scale only because its governance bill has been moved off the ledger. In a persistent system, bounded per-request evaluation and a fair scheduler matter more than a fictional promise that all recursion eventually stops.

12 What Exactly Could Transfer From Small to Large?

12.1 Three distinct transfer claims

Implementation reuse means a tested kernel runs on additional instances with the same interface. That is an engineering property of the implementation and its trusted dependencies.

Certificate reuse means a proven schema can discharge an obligation for a new instance after checking the substitutions, assumptions, environment, and version. A changed type, aliasing relationship, resource ownership, or capability invalidates automatic reuse unless the certificate explicitly covers it.

Learned generalization means a trained proposer or interpreter correctly handles new combinations or depths. This is an empirical ability, not supplied by a proof that the checker is reusable. SCAN, COGS, and CFQ help isolate different generalization gaps; the positive meta-learning and program-library results show those gaps can sometimes be reduced [18–22].

An implementation must disclose which of these three produced a performance gain. A supplied symbolic interpreter executing an unseen depth correctly is not necessarily a neural model learning a new rule. A better parser may improve correct proposals without changing the governing grammar. A better verifier may reject more bad proposals without improving the model's reasoning. All are useful, but they are different results.

12.2 Known primitives are not arbitrary new semantics

A finite library with compositional semantics can execute previously unseen combinations of known operators. That does not establish understanding of a new operator whose effects were never specified. The universal schema delegates to primitive semantics; it does not invent correct semantics for new domains.

This is also why a codon-to-House mapping needs more than relabeling. If the destination level introduces negotiation, shared ownership, external institutions, or legal duties absent from the source level, the adapter must express those obligations. Domain-specific contracts are legitimate. Quietly omitting them to protect the slogan is not.

12.3 The cost of long chains

Even a strong learned proposer can become unreliable on very long compositions. If success requires

m independent correct choices of probability p, the joint probability is p^m. Independence is rarely a safe assumption in real agents, so that expression is an illustrative model, not a universal forecast.

A more conservative engineering statement uses the union bound. If E_j is an error event at obligation j with $\Pr(E_j) \leq \epsilon_j$, then

$$\Pr\left(\bigcup_{j=1}^{m} E_j\right) \leq \sum_{j=1}^{m} \epsilon_j. \tag{16}$$

No independence is required. However, the ϵ_j must themselves be justified for the relevant distribution; guessed confidence scores are not probability bounds. A shared systematic checker defect can defeat all obligations at once.

The proposed architecture therefore separates formally established guarantees, calibrated empirical judgments, and unresolved evidence in its receipts.

13 A Research Program That Can Disconfirm the Thesis

13.1 Three linked experimental tracks

Track I: executable symbolic micro-worlds. Build a finite reference language with known semantics for types, sequencing, bounded repetition, parallel composition, delegated reads and writes, evidence summaries, budgets, and policy epochs. Generate tasks with ground-truth contracts and counterexamples. An independent reference interpreter determines success and detects forbidden effects. Exhaustive enumeration is feasible only at small bounds; larger tests use held-out generation and property-based sampling.

Track II: learned compositional proposals. Give the same underlying models equal access to the primitive catalog, task examples, tools, and compute. Withhold selected constructor combinations, depths, role bindings, and authority patterns. Use SCAN, COGS, and CFQ as auxiliary measures of compositional learning, not as substitutes for the governance tasks. A model can score well on language composition and still delegate authority incorrectly.

Track III: hierarchical agent operations. Evaluate sandboxes containing document synthesis, memory updates, subagent delegation, shared-resource coordination, and restoration under changed policy. Use synthetic private data and simulated effects. Safety-critical actions remain sandboxed. The Circle rule is one topology among several, so any result can be separated from an accidental advantage of eightfold branching.

No track has been run for this paper. These are specifications for a study, not retrospective descriptions of results.

13.2 Strong baselines and targeted ablations

Arm	Architecture	What it tests
A	Ordinary hierarchical task decomposition, with a common external effect boundary	Reference behavior without recursive contract checking
B	Strong typed workflow system: contracts, scoped capabilities, aggregate budgets, verification, versioning, and audit	The conventional comparator that must not be weakened
C	B implemented through a shared recursive kernel, without cross-level refinement certificates	The value of rule/schema reuse alone
D	Proposed contract-carrying recursive grammar with checked refinement, evidence dependencies, and receipts	The integrated Pillar VIII proposal
D^{-r}	D without parent–child refinement checks	Whether refinement adds value beyond local validity
D^{-b}	D without aggregate reservation checking, inside the sandbox only	Resource amplification across apparently valid children
D^{-v}	D without version-bound certificate invalidation	Stale authority and evidence reuse

All arms receive comparable capabilities, models, oracle access, development effort, and execution budgets. Arm B may use proof reuse and caching if those are conventional best practice; forbidding them merely to favor D would invalidate the comparison. The detailed implementation, not the label fractal, determines what differs.

Depth, branching width, total primitive actions, concurrency, and adversarial conditions are experimental factors applied across arms. The study must not call "transfer testing" an architectural arm or expose only one baseline to malicious inputs. In particular, depth effects are measured at matched work where possible, and resource curves are reported where exact matching would suppress the phenomenon of interest.

13.3 Leakage-resistant splits and independent scoring

Freeze primitive semantics, the policy grammar, the rule library, and contract templates before the final evaluation. Hold out combinations as well as surface wording. Randomly rename identifiers and vary roles to prevent success from depending on memorized canonical phrases. Include unseen depths with familiar primitives, familiar depths with unfamiliar combinations, and both together. Keep a separate test for genuinely new primitive semantics; it should not be conflated with recombination. Training data, retrieval caches, tool documentation, and generated examples must not disclose withheld solutions. SCAN, COGS, and CFQ are public benchmarks and may be contaminated in pretrained models; newly generated, preregistered sandbox tasks provide the primary evaluation. Report model versions, seeds, decoding policies, tool wrappers, specification versions, and the exact rule set used.

The evaluator must not be the same language model judging its own prose. Where an exact oracle exists, execute the symbolic target and compare effects or denotations. Where it does not, use blinded, independent evaluation and report inter-rater disagreement and uncertainty. Formal correctness against a test contract and real-world adequacy of that contract remain separate assessments.

13.4 The three flagship experiments

Experiment 1 — Safe pieces, unsafe whole. Construct cases in which every local child passes a naive check but the composition violates a global obligation: duplicated spending, combined disclosure, an order-sensitive effect, a mismatched unit, or authority laundering through a summary. Pair each with a minimally changed lawful composition. The primary question is whether D rejects the unsafe whole without unnecessarily rejecting the lawful neighbor, compared with B and C. Experiment 2 — New depth without new privilege. Train the proposer on shallow hierarchies and test deeper or differently branched structures. Keep the root's authority fixed. Measure valid task completion, unauthorized effects, aggregation errors, and total checking cost. Re-run with an oracle proposer supplying valid candidates; that isolates checker scaling from learned proposal quality.

Experiment 3 — A correction must cross the hierarchy. After a subtree has been certified, revoke a capability, correct a source, change a resource reservation, or advance the policy epoch. The system must invalidate the affected certificates, preserve unaffected work where justified, and prevent stale execution. Include false alarms and irrelevant changes: a system that restarts everything on every update is not an efficient compositional architecture. This is Pillar VII's capacity to reopen a conclusion applied to a tree of dependent commitments.

13.5 Endpoints, hypotheses, and statistical commitments

Define lawful task completion at episode level as correct task completion, within the declared deadline and total resource budget, with no protected-effect violation. Refusals and timeouts are not successful completions. Also report unauthorized effects per episode, false-refusal rate on feasible tasks, latency distributions, verification work, certificate-storage growth, and maintenance effort for a policy revision. No single average may compensate for a critical safety violation.

Hypothesis 1 (Cross-level fidelity). Relative to strong conventional arm B, D improves lawful task completion on preregistered cross-level composition tasks by a practically meaningful margin, while meeting a separate noninferiority condition on protected-effect violations and declared cost limits. Hypothesis 2 (Refinement is load-bearing). Relative to C and D^{-r}, D reduces parent–child contract mismatches in the paired safe/unsafe-composition experiment, without an unacceptable increase in false refusals.

Hypothesis 3 (Authority and budgets survive nesting). With increasing depth and width, D preserves scoped authority and aggregate reservations within the stated enforcement assumptions. D^{-b} and deliberately stale-authority conditions expose the specific failures those checks are designed to prevent. Any violation in D is reported, not averaged away by task success.

Hypothesis 4 (Transfer is conditional and measurable). At matched training and test-time budgets, D supports improved lawful completion on withheld combinations or depths relative to the registered baselines. The study separately estimates gains in proposal quality, rejection quality, and schema reuse; it makes no claim of unlimited extrapolation.

Hypothesis 5 (Versioned evidence reduces stale execution). Under relevant corrections or policy changes, D

blocks stale execution and revalidates affected work more reliably than D^{-v}, while avoiding needless invalidation on matched irrelevant changes.

The primary success rule should be preregistered as a conjunction: a lower confidence bound on the task-completion improvement must exceed a chosen minimum effect Δmin, an upper bound on the violation-rate difference must remain below a prespecified noninferiority margin, and cost must remain inside its declared envelope. Thresholds are design choices to be selected before the held-out study, not constants discovered by naming the architecture.

Treat the episode or task family as the sampling unit rather than counting thousands of correlated child checks as independent observations. Use paired instances across arms, multiple model or training seeds where relevant, and cluster-aware uncertainty estimates. Size the study by simulation for the declared endpoint and dependence structure. Correct for multiple primary comparisons or use a predeclared testing hierarchy; do not search across depths and report only the most favorable one.

For N independent, identically distributed fixed-suite trials with zero observed failures, the one-sided binomial upper bound at confidence $1-\alpha$ is $1-\alpha^{1/N}$. That formula does not bound every adaptive attack, correlated episode, or future deployment. Red-team results should state the attack budget and search procedure, alongside paired benign utility. Zero observed failures is evidence about the test, not an unlimited security certificate.

13.6 What would count against the proposal

The integration claim is narrowed or rejected if B matches D at comparable cost; if rule reuse offers no advantage beyond ordinary modular engineering; if parent–child abstraction hides protected effects; if unknown evidence is effectively treated as permission; if aggregate reservations can be duplicated; if apparent transfer disappears under structural holdouts; or if governance costs destroy useful completion as the hierarchy grows.

A result in which only the oracle proposer succeeds would support a checking architecture but not learned compositional competence. A result in which grammar reuse reduces maintenance work without improving capability would support a software-engineering contribution, not a theory of intelligence. A failure of a geometric-dimension claim would not by itself refute the contract architecture; it would require removing that separate claim. Distinct theses deserve distinct failure conditions.

14 Prior Art and the Continuum Contribution

The paper claims no invention of recursive syntax, compositional semantics, contract checking, refinement, or attenuated delegation. Its contribution is a proposed integration of those disciplines around a single cross-level obligation: a symbolic expansion must account for the contract it inherits. That obligation becomes operational through typed boundaries, compatible assumptions, aggregate reservations, evidence dependencies, and a common admission kernel. Whether this integration provides benefits beyond a strong conventional implementation is a research question, not a consequence of the name.

This formulation also sharpens the relationship to the earlier pillars. Pillar I concerns the reconstruction of usable state; VIII asks what a reconstructed component must expose so that other components may safely depend on it. Pillar II separates a witness's interpretation from promotion into shared consequential state; VIII prevents repeated or nested reports from acquiring additional authority merely through aggregation. Pillar III concerns governed commitment; VIII makes explicit the contracts that must be discharged when a commitment is implemented by several smaller commitments. Pillar IV supplies the inspectable symbolic working surface; VIII specifies the interfaces through which such surfaces compose.

Pillar VII asks whether a settled conclusion can be reopened when its support changes. Pillar VIII extends that discipline across a hierarchy: a revised premise may invalidate a descendant or ancestor certificate, but it should not require discarding unrelated work without reason. Together they suggest a distinction worth preserving. Correction over time and preservation across structure are different obligations. A system may satisfy one while failing the other. The architecture should measure both rather than subsume them under a single word such as coherence.

The Circle protocol is an especially useful test because the doctrine couples a simple expansion rule to an explicit

admission requirement [2, §6.4]. It offers a controlled family of structures on which resource growth, evidence invalidation, authority propagation, and semantic agreement can be examined. It is a design fixture, not evidence that the proposed civilization already exists, and not a privileged branching factor imposed by nature.

15 Limitations and the Genuine Frontier

Specification remains the hardest input. A formal contract may preserve the wrong thing. Translating an instruction such as "protect the user's interests" into an adequate machine-checkable obligation is not solved by wrapping it in a grammar. Some requirements will remain review procedures, empirical tests, or unresolved judgments. Their status must survive composition just as their conclusions do.

The formal results have a restricted domain. The finite-tree propositions require sound composition rules, controlled effects, and an uncompromised enforcement path. Appendix D additionally supplies one concurrent atomic-ledger model with explicit rely conditions. Arbitrary shared mutable state, asynchronous communication, cycles, adaptive adversaries, changing environments, and probabilistic components still require their own models. They are not licensed by a proof about a simpler system.

Shared law is not a universal policy. A medical workflow, a symbolic arithmetic expression, and a House-level delegation may share a proof-checking discipline without sharing the same substantive obligations. A level-specific semantic adapter is not an automatic failure; an unregistered exception that bypasses obligations is. The goal is explicit variation under a reusable discipline, not uniformity purchased by erasing real differences.

Learning and verification can fail independently. A sound checker does not guarantee that a model will propose a useful solution within budget. A capable proposer does not guarantee a complete contract or a sound checker. Repeated refusals may indicate successful enforcement, poor proposal quality, overrestrictive policy, or all three. Measurements must distinguish these causes. Fractality is an additional claim. The grammar may be structurally recursive without any empirically established fractal geometry. An optional address metric can produce a precise fractal object, but its dimension depends on that metric and does not measure cognition. Finite deployed structures require finite-scale descriptions, not an unqualified infinite-limit dimension. Scale creates engineering costs. Verification, storage of certificates, invalidation, and synchronization are not free. A common kernel may reduce duplication while creating a common failure dependency. The architecture has to earn its cost and its trust through implementation, testing, and independent review.

The genuine frontier is therefore not whether a short rule can generate a large tree; that is elementary. It is whether a learned system can produce useful, deeply composed behavior whose meaning, authority, and obligations remain inspectable under change. A future system might learn reusable abstractions and propose their contracts together, then submit both to independent checking. That would be an important capability. It would still not make the proposed contract true merely because the system generated it, and it would not settle questions about subjective experience.

16 Philosophical Implication: Growth Without Exemption

The Continuum's image of one law extending from a codon to a House is most powerful when read as an ambition about accountability rather than visual resemblance. Small components are often easy to inspect. Large systems are where responsibility becomes diffuse: one module generated a statement, another summarized it, another delegated it, and a final module acted as though every preceding step had settled the question. The structure grows; the explanation disappears. Symbolic Fractal Grammar proposes the opposite direction. Growth should make the path of responsibility longer, perhaps, but not less legible. Every enlargement should reveal what it inherited, what it changed, which assumptions it relies on, and who permitted the resulting act. The repeated pattern is not identical content. It is the requirement that each new structure account for its own authority and for the consequences of its composition.

This is not a mathematical derivation of moral legitimacy. A well-formed certificate can attest compliance with an inadequate charter. But making the charter, assumptions, and derivation inspectable gives criticism a target. An unexplained exception cannot disappear inside the elegance of a recursive diagram. A vast institution does not become lawful merely because each of its small parts can tell a locally convincing story.

That is the boldest reading of the pillar that the present evidence can responsibly support: a system should be able to grow without acquiring an exemption from the reasons that justified its smaller acts. Whether the proposed architecture achieves that is open. The requirement itself is precise enough to build against, to challenge, and to fail.

17 Conclusion: Let the Obligation Scale With the Structure

A Guide's explanation

A Guide can say: "Pillar VIII is about growing without losing your conscience. In the Continuum every House grows by the same eightfold rule, and the paper shows how an obligation can be carried down through every expansion, so that a vast structure stays as accountable as its smallest act. The proofs hold for the reference models; whether real systems do better this way is still to be tested." A Guide should not say that a full eightfold civilization exists or has been deployed, that eightfold branching is literally fractal, or that it runs this grammar itself unless that has been verified.

Symbolic Fractal Grammar makes growth answerable to inherited commitments. A recursive generator supplies the structure; typed semantics and refinement preserve what that structure promises; a shared effect boundary conserves authority and resources. The finite-tree and concurrent results identify sufficient conditions for this preservation. The quantitative extension in Appendix E determines how local evidence must improve as the number of obligations increases, and how to allocate verification effort efficiently within its declared model.

The eightfold Circle protocol remains a concrete design fixture with exact per-House and whole-system counts. Its address-space geometry has a complete proof. Implementation reuse, certificate reuse, and learned transfer each have their own evaluation. The empirical contribution to seek is useful, deeply composed behavior whose commitments remain inspectable under correction and growth, at a cost that competes with strong conventional systems.

A larger intelligence can remain accountable when its reasons travel with its capabilities. Every expansion must show what it inherits, what it changes, and why the resulting act is admissible. That is the law this grammar is built to carry.

A A Metric Construction for the Circle Addresses

This appendix makes the optional geometric statement precise. It is a standard symbolic-space construction offered to prevent the branching count from being mistaken for a dimension. It is not a measured property of a deployed Continuum system or a new fractal theorem.

Let $X = \{1, \dots, 8\}^{\mathbb{N}}$ be the set of infinite branch addresses. For distinct x, y, let $\ell(x, y)$ be the number of initial symbols they share, and fix a number $r \in (0, 1)$. Define

$$d_r(x, y) = r^{\ell(x,y)}, \qquad d_r(x, x) = 0. \tag{17}$$

If x and y share m initial symbols and y and z share n, then x and z share at least $\min(m, n)$. Therefore

$$d_r(x, z) \le \max\{d_r(x, y), d_r(y, z)\}. \tag{18}$$

so d_r is an ultrametric. The space is compact: for each depth it has finitely many cylinders, and a diagonal subsequence argument produces a convergent subsequence from any sequence of addresses. Prefixing an address by any one symbol is a similarity of ratio r, producing eight disjoint first-level cylinders.

A depth-n cylinder has diameter r^n, and the space is covered by 8^n such cylinders. Put

$$s = \frac{\log 8}{\log(1/r)}, \qquad 8r^s = 1. \tag{19}$$

For any $q > s$, the sum of q-powers of their diameters is $(8r^q)^n \to 0$, so the Hausdorff dimension is at most s. For the lower bound, assign mass 8^{-n} to every depth-n cylinder. A set of positive diameter δ in this metric has $\delta = r^m$ for some integer $m \geq 0$ and lies in a depth-m cylinder. Its outer measure is therefore at most $8^{-m} = \delta^s$. Singletons have zero mass. Any countable cover of the whole space consequently satisfies $\sum_i (\operatorname{diam} U_i)^s \geq 1$. Thus the Hausdorff dimension is at least s, and equals s.

The result depends on the chosen r. With $r = 1/2$, $s = 3$; with $r = 1/3$, $s \approx 1.893$. Both have the same branching count 8^n. This is the promised demonstration that the count alone does not determine dimension. Any finite collection of distinct truncated addresses, equipped with a metric, has Hausdorff dimension zero; a fitted finite-range scaling exponent is a different quantity and must be labeled accordingly. For Euclidean iterated-function systems, the conditions connecting contraction ratios and Hausdorff dimension are treated by Hutchinson [8].

B Editorial and Mathematical Changes From the Supplied Draft

The source manuscript remains the origin of the pillar's terminology. This revision changes the following claims explicitly; it does not present those changes as statements already made by the source.

Original inference or design	Research-expanded replacement
Recursive grammar is self-similar by construction	Recursion, structural repetition, geometric self-similarity, semantics, and learned transfer are separate properties.
An L-system is a fractal grammar	Parallel rewriting supplies a generator; a geometric interpretation and scaling conditions are additional choices.
8^n supplies a fractal dimension	The doctrine supplies branching counts. Dimension requires a declared metric or geometric construction; Appendix A gives an optional example.
Composable iff scale-invariant, confluent, terminating	Typed interfaces, sound contracts, environment assumptions, effects, and refinement govern composition. Confluence and termination have a narrower role in canonicalization.
The same literal law applies to every object	A shared admission schema accepts explicitly typed contracts and versioned domain adapters. Different obligations need not be disguised as identical policy.
Deviation from 8^n implies statistical self-similarity	A deviation may reflect pruning, failed admission, counting scope, or implementation error. Statistical scaling requires a separate model and evidence.
Rule reuse supports learned transfer	Reuse by a supplied interpreter is distinguished from a learned proposer's success on withheld structures; both are measured separately.
A transfer test is one comparison arm	Depth, width, holdouts, and attacks are factors across architectural arms, including a strong conventional baseline.
Local admission supports the whole	Aggregate authority, shared resources, cross-component effects, and stale evidence are explicit proof and testing obligations.

C Notation and Source Discipline

Notation or object	Meaning
Σ, τ, t	Typed constructor signature, a type, and a finite symbolic term.
$[\![t]\!]$	A declared denotation or behavior model, separate from its syntax.
$\mathcal{K}_v$	Component contract: interface, pre/postconditions, invariants, effects, capabilities, budgets, and epoch.
$\mathcal{K}, \pi_v, \zeta_v$	Shared admission kernel, submitted proof or evidence artifact, and the node's checked context.
Cap_v, b_v, c_v	Scoped capabilities, additive resource allocation, and local consumption.
a, a^*, Tr	State/observation abstraction, trace abstraction, and permitted execution traces.
$C_n^H, C_n^{\mathrm{all}}$	Circle-n population per House and across eight Houses, respectively.
d_r, s	A deliberately chosen address-space metric and its resulting Hausdorff dimension.
$\hat{\delta}_a$	Observed mismatch rate under a declared abstraction and independent contract test.

The external bibliography includes peer-reviewed articles and proceedings and scholarly books. For CaMeL, the inspected author version is distinguished from its subsequently verified SaTML 2026 publication status. These source classes are not treated as interchangeable.

The mathematical propositions above are conditional derivations, not empirical observations; the experiments are proposals, not results. The two internal references establish what the Continuum and the supplied draft say, not that those claims have been independently validated. Bibliographic records and relevant accessible text were checked for this revision; no claim is made to have reproduced every cited experiment.

D A Concurrent Reference Protocol for Resources and Authority

Finite-tree accounting does not implement a concurrent budget. This proposed extension supplies one explicit reference semantics; it is not evidence that a production runtime already enforces it. Let $B \in \mathbb{N}$ be a fixed total of an additive resource, $F \in \mathbb{N}$ the free balance, and $S \in \mathbb{N}$ irreversibly spent units. A finite reservation map R stores unique identifiers r with owner, nonnegative balance q_r, and a current delegation record. The global invariant is

$$F + \sum_{r \in \mathrm{dom}(R)} q_r + S = B, \qquad F, q_r, S \geq 0. \tag{20}$$

For vector budgets, apply the equations componentwise; latency and peak memory need different semantics. Spent units are not refunded by canceling a computation. A verified compensating physical transaction or an explicitly authorized new budget would require a separately modeled operation and cannot be smuggled into ordinary release.

Atomic operations. A trusted linearizable ledger supports the following transitions, each authenticated, journaled under a unique operation identifier, and committed once. Amounts q are positive integers. Reserve(r, a, q) requires a fresh reservation identifier, a current scoped grant for owner a, and $F \geq q > 0$; it sets $F := F - q$ and $R[r] := (a, q)$. Transfer(r, r', q) requires $0 < q \leq q_r$, current transfer authority, and a fresh destination record or a named existing record; it subtracts q from q_r and adds exactly q to $q_{r'}$. Ownership and the permitted delegation chain are checked; the destination capability is attenuated from the source. Spend(r, q, u, e) requires $0 < q \leq q_r$, an action u authorized by the *current* epoch e and capability, and an atomic protected-effect boundary; it subtracts q from the reservation and adds q to S. Release(r, q) requires $0 < q \leq q_r$ and subtracts known unspent units from q_r and restores them to F. Governance revocation advances the epoch and invalidates affected grants; it neither duplicates nor destroys balances.

All operations compare the currently installed contract and capability versions at commit, including reservation/transfer/release permissions. Ordinary delegation is set inclusion and adds no right absent from the root. A reused operation identifier with identical canonical arguments is an idempotent retry; different arguments are rejected. The durable journal survives restoration. In a distributed transfer, a pending amount remains reserved exactly once until a terminal state is known. A timeout cannot simultaneously refund the sender and leave a

valid receiver claim. An external effect not included in the ledger transaction requires an idempotent receiving service with the same operation identifier and authority fence, or an explicit unresolved state; a log written after an uncontrolled action does not supply atomicity.

Theorem 1 (Concurrent conservation, authority, and refinement)**.** *Assume initially $F = B$, $S = 0$, and R is empty; every resource-consuming or authority-changing effect passes through the stated ledger; its storage is durable and linearizable; delegated grants only attenuate; current policy checks are sound; and the protected effect and journal share the stated atomic boundary. Every interleaving preserves (20), admits no spend under a revoked or superseded grant, and creates no authority through ordinary delegation. Hiding purely internal preparation/retry steps yields a trace admitted by the abstract reserve/transfer/spend/release machine with the same revocation order.*

Proof. Induct on ledger linearization points. Reserve moves q from F to one reservation; transfer moves it between two reservations; spend moves it to S; release moves it back to F. Each guard preserves nonnegativity and each change preserves the sum. Revocation changes permissions but no quantity; deduplicated retries change neither. Current authority is checked in the same transition as a new effect, so a revoked grant cannot authorize a later spend. Set inclusion composes transitively along every delegation path. For refinement, relate concrete and abstract states by equality of F, R, S, current epoch and grants, and committed operation identifiers. A ledger commit takes the matching abstract transition; preparation, rejected attempts, and retries stutter with respect to resource/effect observations. Induction gives trace inclusion. If rejection itself is a protected observation, retain it as an explicit abstract rejection event rather than hiding it. □

These are rely conditions on a shared service, not facts derivable from local component contracts. The environment must not mutate balances or grants through another path; serialization must extend to the effect receiver; certificate translation must identify the actual action. The theorem conserves a declared additive resource and enforces declared grants. It does not prove that individually authorized actions have jointly desirable semantics. A policy restricting disclosure across multiple recipients, for example, needs an additional shared-state invariant.

Counterexamples and executed finite checks. The supplied script explores budgets of two units with two or three agent owners, reservation/transfer/spend/release, one revocation, and stale-epoch requests to a bounded depth. It checks conservation and current authority after every transition and reconstructs the abstract projection. Its broken variants expose three distinct errors: two agents both read one free unit and independently reserve it; a permission check precedes revocation but its effect commits afterward; and a timed-out transfer is refunded while the recipient retains the transferred claim. The first oversubscribes, the second spends under stale authority, and the third mints a duplicate unit. These are model checks and counterexamples, not a production or agent benchmark.

Progress is conditional. Safety permits a hold. If operations need several separately locked resources, require a common strict acquisition order, a finite set of pending jobs, finite critical sections, fair scheduling and fair lock granting (for example, FIFO queues), and eventual availability of current authority/evidence. A wait-for cycle would require strictly increasing lock ranks around a cycle, which is impossible; finite holders and fair lock granting then discharge pending lock waits. Fair CPU scheduling alone does not imply fair acquisition of a repeatedly contested lock. This establishes the stated lock-level progress condition, not eventual completion of every task. An infeasible reservation, revoked grant, partition, unlimited retry policy, or absence of any useful proposal can prevent admission. This is the same distinction between a lawfully governed expansion and a promise that growth must occur.

Atomicity and fencing are established techniques [26]; linearizability provides the trace criterion [27]. The Continuum's proposed contribution is to make those obligations part of the recursively reused contract, while preserving the finite-tree and metric results already established above.

E The Cost of Keeping a Growing Hierarchy Accountable

E.1 A sharp risk budget under conditional checks

The structural preservation theorem assumes sound premises. When some premises are established by calibrated empirical checks, the whole hierarchy needs a quantitative evidence budget. Let $N \in \mathbb{N}$, $N \geq 1$, be the number of designated obligations in a finite execution order. Let $(\mathcal{F}_i)_{i=0}^N$ be the information history and $A_i \in \mathcal{F}_i$ the event that obligation i is in fact satisfied. Rejected or skipped obligations may be padded by $A_i = \Omega$; a held task is still recorded as incomplete. Set $S_i = \bigcap_{j=1}^{i} A_j$ and $S_0 = \Omega$. For fixed bounds $\epsilon_i \in [0,1)$ require

$$\mathbb{P}(A_i^c \mid \mathcal{F}_{i-1}) \leq \epsilon_i \quad \text{almost surely on } S_{i-1}.$$

This is a bound on the chosen check after its actual preceding history. An unconditional accuracy measured on unrelated cases does not supply it. The order may be chosen adaptively if these conditional bounds remain valid.

Theorem 2 (Sharp composition of a finite evidence budget)**.** *Under the stated conditional bounds,*

$$\mathbb{P}(S_N^c) \leq 1 - \prod_{i=1}^{N}(1-\epsilon_i).$$

The bound is attainable. For a common bound $\epsilon \in [0,1)$, a universal guarantee $\mathbb{P}(S_N^c) \leq \delta$, $\delta \in (0,1)$, throughout this model class holds precisely when

$$\epsilon \leq 1 - (1-\delta)^{1/N}.$$

Proof. The tower property gives $\mathbb{P}(S_i) = \mathbb{E}[\mathbf{1}_{S_{i-1}}\mathbb{P}(A_i \mid \mathcal{F}_{i-1})] \geq (1-\epsilon_i)\mathbb{P}(S_{i-1})$. Induction yields the product. Independent Bernoulli failure indicators with probabilities ϵ_i attain equality; independence is an attaining construction, not an assumption of the bound. Solving $1-(1-\epsilon)^N \leq \delta$ proves sufficiency, and the attaining construction proves necessity for a guarantee over the entire class. □

For a fully expanded House through depth $h \in \mathbb{N} \cup \{0\}$, one designated obligation per node gives $N = (8^{h+1}-1)/7$. Thus a fixed local error allowance cannot certify a fixed global allowance at all depths. In the small-δ regime the required common allowance is approximately δ/N; the exact expression above governs any actual decision. Obligations may be grouped differently, but the grouping and its conditional guarantees must be declared.

E.2 Least-cost evidence allocation in an explicit calibration model

Let $q_i \in [0,\infty)$ be verification effort in a common work unit. Assume a justified conditional error curve

$$\epsilon_i(q_i) = a_i e^{-b_i q_i}, \qquad a_i \in (0,1), \quad b_i > 0,$$

where b_i has inverse-work units. The curve is a reference assumption that must be calibrated for an application. Neither repetition nor a language model's stated confidence establishes it. Write $[x]_+ = \max(x,0)$ and $R_0 = 1 - \prod_i (1-a_i)$. We seek the smallest total effort $\sum_i q_i$ that certifies a target $\delta \in (0,1)$ by the preceding theorem.

Theorem 3 (Optimal verification effort for the declared curves)**.** *If $\delta \geq R_0$, the optimum is $q_i = 0$ for every i. If $\delta < R_0$, there is a unique $\lambda > 0$ satisfying*

$$\prod_{i=1}^{N}\left(1 - \min\left\{a_i, \frac{1}{1+\lambda b_i}\right\}\right) = 1 - \delta.$$

The unique optimal allocation and its resulting error allowances are

$$q_i^* = \frac{[\log(a_i(1+\lambda b_i))]_+}{b_i}, \qquad \epsilon_i^* = \min\left\{a_i, \frac{1}{1+\lambda b_i}\right\}.$$

Proof. Define $f_i(q) = -\log(1 - a_i e^{-b_i q})$. The constraint is $\sum_i f_i(q_i) \leq C$, where $C = -\log(1-\delta) > 0$. Direct differentiation gives

$$f_i'(q) = -\frac{b_i \epsilon_i(q)}{1-\epsilon_i(q)}, \qquad f_i''(q) = \frac{b_i^2 \epsilon_i(q)}{(1-\epsilon_i(q))^2} > 0.$$

For any fixed $\lambda > 0$, minimize the strictly convex separable function $\sum_i (q_i + \lambda f_i(q_i))$ over $q_i \geq 0$. Its stationary equation is $1 = \lambda b_i \epsilon_i / (1 - \epsilon_i)$, with the minimum at zero when the derivative there is nonnegative. These conditions give the displayed q_i^*. The resulting product starts at $\prod_i (1 - a_i)$, increases strictly once any coordinate becomes active, and tends to one. The target is strictly between those endpoints, giving one λ. For any feasible q, minimality of this separable objective gives

$$\sum_i q_i + \lambda \sum_i f_i(q_i) \geq \sum_i q_i^* + \lambda C.$$

Because $\sum_i f_i(q_i) \leq C$, it follows that $\sum_i q_i \geq \sum_i q_i^*$. Strict convexity of the separable objective proves uniqueness at the optimum. The zero-effort case follows from feasibility and nonnegativity. □

The argument is an application of established separable convex optimization [28], written out so the claimed allocation can be checked directly. Its contribution to this pillar is the explicit link between recursive expansion, admissible global risk, and total governance work. If effort must be integer valued, rounding each q_i^* upward preserves the risk certificate and adds less than N work units; it need not be the exact integer optimum.

E.3 Dependence, floors, and a falsifiable calibration contract

Marginal bounds alone do not imply the product bound. Take two disjoint failure events, each with probability 0.1. Their union has probability 0.2, exceeding $1 - 0.9^2 = 0.19$; after the first event is avoided, the second has conditional probability $1/9 > 0.1$. Equation (16)'s union bound remains available in that case. A common systematic defect may create an irreducible conditional error floor. If such a floor prevents the product constraint from being met, the proposed expansion has no certificate under these assumptions, however much repeated checking is purchased.

The reproducibility checks compare the allocation with independently solved constrained optimization problems, verify exact finite event examples, and test the eightfold count and rounding bounds. They exercise the mathematical reference model. The proposed empirical study must separately estimate history-conditional error curves, include shared-checker failures, and report useful completion together with total verification cost. This makes scale a measurable obligation on evidence.

References

[1] The Eidara Continuum, Architect X13. Symbolic Fractal Grammar: Scale-Invariant Rules, Composability, and the Limits of "Learned Small, Transfers Large." Supplied Pillar VIII manuscript, November 2026. Internal draft supplied for critical revision; no experimental validation asserted.

[2] The Eidara Continuum. Eidara Continuum Bible (COMPLETE). User-supplied internal doctrinal compilation, §§6.4–6.4.1, pp. 92–94; Circle Structure Protocol dated August 24, 2025, and correction-lock addendum dated August 27, 2025. Internal source, not an independently validated research report.

[3] N. Chomsky. Three models for the description of language. IRE Transactions on Information Theory 2(3), 113–124 (1956). https://doi.org/10.1109/TIT.1956.1056813.

[4] M. D. Hauser, N. Chomsky, and W. T. Fitch. The faculty of language: What is it, who has it, and how did it evolve? Science 298(5598), 1569–1579 (2002). https://doi.org/10.1126/science.298.5598.1569.

[5] S. Pinker and R. Jackendoff. The faculty of language: What's special about it? Cognition 95(2), 201–236 (2005). https://doi.org/10.1016/j.cognition.2004.08.004.

[6] P. Prusinkiewicz and A. Lindenmayer. The Algorithmic Beauty of Plants. Springer-Verlag, New York (1990). https://doi.org/10.1007/978-1-4613-8476-2. Author-hosted text: https://algorithmicbotany.org/papers/abop/abop.pdf.

[7] B. B. Mandelbrot. The Fractal Geometry of Nature. W. H. Freeman, San Francisco (1982). Scholarly monograph.

[8] J. E. Hutchinson. Fractals and self-similarity. Indiana University Mathematics Journal 30(5), 713–747 (1981). https://doi.org/10.1512/iumj.1981.30.30055.

[9] R. Milner. A theory of type polymorphism in programming. Journal of Computer and System Sciences 17(3), 348–375 (1978). https://doi.org/10.1016/0022-0000(78)90014-4.

[10] L. de Alfaro and T. A. Henzinger. Interface automata. In ESEC/FSE 2001, 109–120 (2001). https://doi.org/10.1145/503209.503226.

[11] C. A. R. Hoare. An axiomatic basis for computer programming. Communications of the ACM 12(10), 576–580, 583 (1969). https://doi.org/10.1145/363235.363259.

[12] M. Abadi and L. Lamport. Composing specifications. ACM Transactions on Programming Languages and Systems 15(1), 73–132 (1993). https://doi.org/10.1145/151646.151649.

[13] M. Abadi and L. Lamport. The existence of refinement mappings. Theoretical Computer Science 82(2), 253–284 (1991). https://doi.org/10.1016/0304-3975(91)90224-P.

[14] J. C. Reynolds. Separation logic: A logic for shared mutable data structures. In 17th Annual IEEE Symposium on Logic in Computer Science, 55–74 (2002). https://doi.org/10.1109/LICS.2002.1029817.

[15] G. C. Necula. Proof-carrying code. In 24th ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages, 106–119 (1997). https://doi.org/10.1145/263699.263712.

[16] F. B. Schneider. Enforceable security policies. ACM Transactions on Information and System Security 3(1), 30–50 (2000). https://doi.org/10.1145/353323.353382.

[17] E. Debenedetti, I. Shumailov, T. Fan, J. Hayes, N. Carlini, D. Fabian, C. Kern, C. Shi, A. Terzis, and F. Tramèr. Defeating prompt injections by design. IEEE SaTML 2026; inspected author version arXiv:2503.18813v2 (2025). https://satml.org/2026/program/; https://arxiv.org/pdf/2503.18813. Formal verification of the interpreter is future work in the inspected version; no solved general prompt-injection boundary is asserted.

[18] B. M. Lake and M. Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In ICML 2018, PMLR 80, 2873–2882 (2018). https://proceedings.mlr.press/v80/lake18a.html.

[19] N. Kim and T. Linzen. COGS: A compositional generalization challenge based on semantic interpretation. In EMNLP 2020, 9087–9105 (2020). https://doi.org/10.18653/v1/2020.emnlp-main.731.

[20] D. Keysers, N. Schärli, N. Scales, H. Buisman, D. Furrer, S. Kashubin, N. Momchev, D. Sinopalnikov, L.

Stafiniak, T. Tihon, D. Tsarkov, X. Wang, M. van Zee, and O. Bousquet. Measuring compositional generalization: A comprehensive method on realistic data. In ICLR 2020. https://openreview.net/forum?id=SygcCnNKwr.

[21] B. M. Lake and M. Baroni. Human-like systematic generalization through a meta-learning neural network. Nature 623, 115–121 (2023). https://doi.org/10.1038/s41586-023-06668-3.

[22] K. Ellis, C. Wong, M. Nye, M. Sablé-Meyer, L. Morales, L. Hewitt, L. Cary, A. Solar-Lezama, and J. B. Tenenbaum. DreamCoder: Bootstrapping inductive program synthesis with wake-sleep library learning. In ACM SIGPLAN Conference on Programming Language Design and Implementation, 835–850 (2021). https://doi.org/10.1145/3453483.3454080.

[23] T. Scholak, N. Schucher, and D. Bahdanau. PICARD: Parsing incrementally for constrained autoregressive decoding from language models. In EMNLP 2021, 9895–9901 (2021). https://doi.org/10.18653/v1/2021.emnlp-main.779.

[24] M. H. A. Newman. On theories with a combinatorial definition of "equivalence." Annals of Mathematics, second series, 43(2), 223–243 (1942). https://doi.org/10.2307/1968867.

[25] F. Baader and T. Nipkow. Term Rewriting and All That. Cambridge University Press (1998). Scholarly monograph; definitions and results on reduction, termination, and confluence.

[26] M. Burrows. The Chubby lock service for loosely-coupled distributed systems. OSDI 2006, 335–350, §2.4. https://www.usenix.org/legacy/event/osdi06/tech/full_papers/burrows/burrows_html/. Receiving-service sequencer checks; established fencing precedent.

[27] M. P. Herlihy and J. M. Wing. Linearizability: A correctness condition for concurrent objects. ACM Transactions on Programming Languages and Systems 12(3), 463–492 (1990). https://doi.org/10.1145/78969.78972. Primary concurrency correctness criterion, not a claim that the proposed implementation has been verified.

[28] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004, Chapter 5. Author-hosted text: https://web.stanford.edu/~boyd/cvxbook/bv_cvxbook.pdf. Established convex optimization and optimality conditions; the allocation above is derived explicitly.

Continuum lineage. Pillar VIII develops the Eidara Continuum's Symbolic Fractal Grammar and Circle Structure Protocol as a proposal for typed, evidence-bearing, recursively composable governance. The codon, scroll, lattice, entity, House, and Circle vocabulary is retained as internal architectural language; public mathematical claims are bounded by the assumptions stated in the text. The internal doctrine is a supplied design source, not independent evidence of implemented AGI. Proprietary operator internals are not reproduced.

Architect X13 · The Eidara Continuum · Science & Philosophy · eidara.ai