Continuity Spaces

History, Interface, and Authority in a Shared Semantic Environment

Flyxion
Independent Researcher

November 2026

Abstract

This essay develops the design of a collaboration environment in which people participate through ordinary applications, such as editors, instruments, media players, and games, while their actions contribute to persistent artifacts and histories held in common. The organizing object is a semantic space whose points are histories considered up to the questions the system has promised to answer about them, and whose geometry is generated by admissible transformations and their costs. Interfaces appear as partial presentations of that space, repositories as bounded regions of it, and a navigable tunnel environment as one interpretation of it among several. The essay supplies a proof spine: a sequence of definitions and propositions establishing when heterogeneous interfaces can share one history-sensitive structure while preserving the distinctions needed for continuation, verification, and authority. Identity is treated separately and afterward, as a statistical hypothesis about whether a new trajectory continues an earlier one, and the essay proves both a limit on what such recognition can achieve and a containment result showing how a mistaken recognition can be kept from executing any request in a declared class of protected operations. The architecture is separable from the recognition research: the shared history and interface system can be built and tested before behavioral identity is reliable, and the essay states which of its claims are proved, which are design commitments, and which remain experimental.

Contents

1 Purpose and Scope **3**

2 Situated Action and the Semantic Space **4**

3 Histories and Admissible Continuation **5**

4 Semantic States as Classes of Histories **6**

5 Interfaces as Partial Presentations **7**

6 Compatible Overlaps and Global Assembly **8**

7 Geometry from Admissible Transformations **10**

8 Provisional Sessions and Hash Identities **12**

9 Evidence of Continuity **13**

10 Authority Under Uncertain Identity **16**

11 Admissibility, Verification, and Refusal **19**

12 Public Rules and Confidential Checks **20**

13 Repair and Collaboration **20**

14 Storage, Execution, and Interoperability **21**

15 A Staged Prototype and Research Programme **22**

16 The Integration Theorem and the Status of the Claims **23**

1 Purpose and Scope

Repository hosting has become the default institutional form for collaborative technical work. A hosting service of the familiar kind supplies versioning, provenance, review, discovery, and access control, and it does so by treating the repository as the unit of work and the account as the unit of agency. Both choices are reasonable for source code maintained by people who already think of their work as files changed through commits. They fit less well when the work is a performance, an annotation made while reading, a route found through a difficult body of material, or a correction offered from inside an application that has no notion of a commit. In those cases the contributor must leave the setting in which the work was done, translate it into the repository's vocabulary, and attach it to an account whose only relation to the work is that someone typed a password. The translation discards most of what made the action intelligible, and the account says nothing about whether the person now acting is the person who acted before.

The proposal examined here inverts the arrangement. Participation happens wherever people already work. An editor, a MIDI instrument, a voice interface, a swipe keyboard, a game, and a document viewer each record the actions taken through them as events in a shared environment, together with the conditions under which those actions occurred. Artifacts, versions, and collaborative relationships are derived from the accumulated histories. A repository in the conventional sense becomes a bounded region of this environment with its own acceptance conditions, and a spatial explorer that renders the environment as a system of tunnels becomes one way of viewing it. Nobody is required to use the explorer in order to participate, and nothing recorded through another interface is inaccessible to it.

Three layers have to be kept apart for the proposal to be assessed fairly. The first is the underlying architecture: events, histories, interface adapters, storage, and mediated authority. This layer uses established techniques and can be specified with ordinary rigor. The second is a set of proposed identity mechanisms, in which a session begins with a provisional hash identifier and its association with earlier histories is estimated from behavioral evidence. The third is a set of hypotheses that require experiment, including whether behavioral continuity recognizes returning participants reliably, whether publicly disclosed knowledge loses its identifying value in the way the model predicts, and whether the spatial interpretation actually helps people recover and continue their work. The essay is organized so that the first layer does not depend on the third. Its mathematical sections prove what can be proved about the first layer and state exactly what assumptions the second would need.

The functions of a hosting service that the environment would replace are versioning, provenance, collaboration, discovery, and access control. The functions it adds are interface independence, the preservation of interaction history alongside artifact state, identity by continuity of activity, and authority graded by the reversibility of what is being attempted. The essay takes the formal apparatus of SpherePOP [4] interface-first. From the introductory text it uses the notion of admissibility, under which an operand tuple is admissible for an operator when it lies in the domain of that operator's partial function, and the treatment of an inadmissible application as an inert term that stays in place and is not an error. The verification outcomes, the witness plane, and the reopening operation used in the later sections belong to the wider operational vocabulary of the framework and are not defined in that text, so they are stated here in full as stipulations of the present design. It is also continuous with the earlier specification *Semantic Geometry and Corrective Navigation* [5], whose central requirement

is retained throughout: every authoritative relationship must be typed, attributable, and reconstructible from an event history, and spatial proximity may express such a relationship but never establishes one. The correspondence between the two documents is direct. The specification's append-only event ledger is what the present essay formalizes as an execution history. Its semantic graph is a derived representation whose exactness is assessed relative to the queries it preserves, using the criterion of Section 4. Its spatial projection is a derived view in the sense of Section 6. The specification's division of labor is kept as well: retrieval proposes candidates, adjudication determines evidential status, and admissibility governs which continuations are possible. What the present essay adds is the treatment of interfaces as partial presentations of one structure, the proofs of what such a structure preserves, and the separation of identity and authority.

2 Situated Action and the Semantic Space

An action receives its significance from the conditions in which it occurs. Pressing a key can enter a letter, sound a note, advance through a corridor, or execute a command. The physical input does not determine which of these has happened. What determines it is the interface that was active, the artifact being changed, the history that preceded the action, and the continuations that become available afterward. This is a familiar observation in studies of situated practice [19, 8], and it has a direct architectural consequence: a system that records only the physical input, or only the resulting artifact, has discarded the information that fixes what was done.

The term *semantic space* will be used for a structure whose local organization reflects what actions mean in context, what they depend on, and what they make possible. The phrase "semantic manifold" is attractive and is used in this project's working vocabulary, but it carries a technical commitment that should not be assumed. A manifold is locally Euclidean, and nothing guarantees that the space of situated actions has that property everywhere. Some regions may be continuous, as when a parameter such as tempo or pressure varies smoothly. Others are discrete or branching, as when one of several commands is chosen. Still others may require descriptions that overlap only partially. The construction given below produces a shared space without presupposing manifold structure, and Section 6 states what additional hypotheses the stronger name would require.

The environment preserves the relationships among an action's several descriptions. A musical performance is at once an acoustic event, a sequence of motor actions, a transformation of a composition, and a traversal of the shared environment. These are different projections of one recorded history. The architecture does not choose among them. It keeps the history and lets each interface, and each later inquiry, take the projection it needs.

The basic objects are few. An *event* is a recorded occurrence with its provenance. A *history* is a finite sequence of events that the system accepts as valid. An *artifact* is a value computed from a history, such as the current text of a document or the current state of a composition. A *relationship* is a typed, attributable link between objects that can be reconstructed from events. A *session* is a maximal stretch of activity carried out under one provisional identifier. An *accepted transformation* is a proposed change that has passed the acceptance conditions of the region in which it was proposed. Everything else in the essay is built from these.

3 Histories and Admissible Continuation

Let E be a set of events and let E^* be the set of finite sequences over E, with the empty sequence ε and the prefix relation $\preceq$. The system's rules determine a set $\mathcal{H} \subseteq E^*$ of valid histories. The rules are assumed to be such that validity is prefix-closed: if $h \in \mathcal{H}$ and $h' \preceq h$, then $h' \in \mathcal{H}$. This says only that a valid history was valid at every earlier moment.

The sequential form of a history needs a word of justification, since collaborative activity is not sequential. Each event names zero or more *parent* events, and the parent relation over the events of a history is a directed acyclic graph, the *causal graph*, in which a merge appears as an event with several parents and concurrent work appears as events with no path between them. A history in $\mathcal{H}$ is an *execution history*: the order in which the record of a region admitted events. Validity includes the requirement that every parent of an event occur earlier in the sequence, so each history is a linear extension of the causal graph it encodes. The two structures should not be confused. One causal graph generally has many linear extensions, these are different elements of $\mathcal{H}$, and nothing proved below makes them equivalent. The order of admission is treated in this essay as a recorded fact on which queries are entitled to depend. Identifying histories that differ only by exchanging adjacent causally independent events, in the manner of trace theory [12], would yield a quotient indifferent to the order of receipt, and that identification is legitimate exactly when every declared query is invariant under such exchanges. The invariance is a strong condition on the design of queries and is not assumed here. The causal quotient is left as a subsequent construction.

Definition 3.1. For $h \in \mathcal{H}$, the *admissible continuation set* of h is

$$\mathrm{Cont}(h) = \{z \in E^* : hz \in \mathcal{H}\}.$$

An interface never shows a history as such. It shows a state computed from it. Let $O : \mathcal{H} \to X$ be an *observation map*, where X is the set of states an interface exposes. The first proposition records why the architecture cannot be built on observable states alone.

Proposition 3.2 (Insufficiency of observation)**.** *Suppose there exist $h_1, h_2 \in \mathcal{H}$ with $O(h_1) = O(h_2)$ and $\mathrm{Cont}(h_1) \neq \mathrm{Cont}(h_2)$. Then no procedure whose only input from the history is $O(h)$ can correctly decide, for every $h \in \mathcal{H}$ and every $z \in E^*$, whether $z \in \mathrm{Cont}(h)$.*

Proof. Choose z in the symmetric difference of $\mathrm{Cont}(h_1)$ and $\mathrm{Cont}(h_2)$; without loss of generality $z \in \mathrm{Cont}(h_1) \setminus \mathrm{Cont}(h_2)$. A procedure with access only to the observation receives the same input $(O(h_1), z) = (O(h_2), z)$ in both cases and therefore returns the same answer. The correct answers differ, so the procedure errs on at least one of the two histories. □

The proposition is elementary, and its hypothesis is satisfied constantly in practice. Two documents with identical text may differ in whether a pending review obligation has been waived, under a policy that makes the waiver bear on what may be accepted next. Two sessions displaying the same position may differ in whether a secret route has already been disclosed to a third party. Two branches with identical contents may differ in which merges remain admissible. In each case the visible arrangement recurs while the available future differs, and Proposition 3.2 says that a system which has kept only the visible arrangement cannot tell these futures apart. The architecture must preserve history, or a summary of history sufficient for the questions it intends to answer. The next section makes "sufficient" precise.

4 Semantic States as Classes of Histories

Let $\mathcal{Q}$ be a family of *queries*, each a function $q : \mathcal{H} \to V_q$ into some set of values. The family represents everything the system promises to answer about a history: the state of each artifact, the admissible continuations, the verification obligations outstanding, and the authority held by each participant. It is assumed throughout that $\mathrm{Cont} \in \mathcal{Q}$.

Definition 4.1. Two histories are $\mathcal{Q}$-*equivalent*, written $h_1 \sim_{\mathcal{Q}} h_2$, when $q(h_1) = q(h_2)$ for every $q \in \mathcal{Q}$. The *semantic state space* is the quotient $\mathcal{S} = \mathcal{H}/\!\sim_{\mathcal{Q}}$, and the class of h is written $[h]$.

Proposition 4.2 (Exact representation)**.** *Let $r : \mathcal{H} \to R$ be any map into a set of representations. Every query in $\mathcal{Q}$ factors through r, in the sense that for each $q \in \mathcal{Q}$ there is a function $\tilde{q} : R \to V_q$ with $q = \tilde{q} \circ r$, if and only if $r(h_1) = r(h_2)$ implies $h_1 \sim_{\mathcal{Q}} h_2$.*

Proof. Suppose each q factors as $\tilde{q} \circ r$. If $r(h_1) = r(h_2)$ then $q(h_1) = \tilde{q}(r(h_1)) = \tilde{q}(r(h_2)) = q(h_2)$ for every $q \in \mathcal{Q}$, so $h_1 \sim_{\mathcal{Q}} h_2$. Conversely, suppose r identifies only equivalent histories. For $q \in \mathcal{Q}$ and ρ in the image of r, set $\tilde{q}(\rho) = q(h)$ for any h with $r(h) = \rho$. The value does not depend on the choice of h, because any two such histories are $\mathcal{Q}$-equivalent. Extend $\tilde{q}$ arbitrarily outside the image of r; the choice plays no role, since $\tilde{q} \circ r$ evaluates $\tilde{q}$ only on the image. Then $q = \tilde{q} \circ r$. □

Proposition 4.2 is the criterion for compression. A summary may discard a distinction exactly when that distinction cannot change any answer the system has promised to preserve. The quotient $\mathcal{S}$ is the coarsest representation that is still exact, and every exact representation refines it. The result also explains why the state space is not fixed once and for all. If the family of queries grows, for instance because the system begins to answer questions about interaction timing that it previously ignored, the equivalence becomes finer and histories that were formerly interchangeable must be distinguished. This is a consideration for retention policy: an event log discarded after summarization can support only the queries the summary was designed for.

For the quotient to carry dynamics, appending an event must respect the equivalence. This does not hold for an arbitrary family of queries, and the condition that secures it is worth stating because it constrains how queries may be declared.

Definition 4.3. The family $\mathcal{Q}$ is *closed under extension* if for each $q \in \mathcal{Q}$ and each $e \in E$ the query q_e belongs to $\mathcal{Q}$, where $q_e(h) = q(he)$ when $he \in \mathcal{H}$ and $q_e(h) = \bot$ otherwise, with $\bot$ a value outside V_q.

Lemma 4.4 (Congruence)**.** *Suppose $\mathcal{Q}$ contains* Cont *and is closed under extension, and let $h_1 \sim_{\mathcal{Q}} h_2$. Then for every $e \in E$ either both $h_1 e$ and $h_2 e$ lie outside $\mathcal{H}$, or both lie in $\mathcal{H}$ and $h_1 e \sim_{\mathcal{Q}} h_2 e$. Consequently the relation $[h] \xrightarrow{e} [he]$, for $he \in \mathcal{H}$, is a well-defined, deterministic labelled transition relation on $\mathcal{S}$.*

Proof. Since $\mathrm{Cont} \in \mathcal{Q}$ we have $\mathrm{Cont}(h_1) = \mathrm{Cont}(h_2)$, so $h_1 e \in \mathcal{H}$ exactly when $h_2 e \in \mathcal{H}$. In the valid case, for each $q \in \mathcal{Q}$ we have $q(h_1 e) = q_e(h_1) = q_e(h_2) = q(h_2 e)$ because $q_e \in \mathcal{Q}$. Hence $h_1 e \sim_{\mathcal{Q}} h_2 e$, and the target class $[he]$ depends only on $[h]$ and e. □

When $\mathcal{Q}$ consists of the continuation queries alone, $\sim_{\mathcal{Q}}$ is the right congruence familiar from the theory of automata [15], and $\mathcal{S}$ is the state set of the minimal machine recognizing $\mathcal{H}$. The present setting generalizes that construction by letting the family of distinctions be chosen according to what the environment has undertaken to preserve.

5 Interfaces as Partial Presentations

Keyboards, instruments, voice, swipe input, editors, and spatial exploration are local coordinate systems through which participants reach portions of the semantic space. Each makes certain distinctions easy to express and others difficult or unavailable. Changing from one keyboard layout to another alters the relation between physical movement and symbolic output. Changing from typing to speech alters it far more. The formal treatment has to permit this variety without requiring any interface to express everything.

Definition 5.1. An *interface presentation* i consists of a set U_i of local states, a set A_i of local actions, a partial *interpretation map* $\phi_i : U_i \rightharpoonup \mathcal{S}$, and an *action interpretation* assigning to each local action $a \in A_i$ performed in a local state u in the domain of ϕ_i an event $e \in E$, when one exists, such that the local successor u' satisfies $\phi_i(u) \xrightarrow{e} \phi_i(u')$. The event carries *provenance*: the identity of the interface, the version of the mapping in force, and the operating conditions relevant to interpretation.

Partiality appears twice. The map ϕ_i is partial because an interface may have local states with no shared meaning, such as a cursor blink or a transient menu. The action interpretation is partial because an interface need not be able to express every semantic transition. A MIDI controller cannot approve a review, and a document viewer cannot play a chord. Neither limitation is a defect of the architecture. Each presentation covers a region, and the question for the next section is what happens where regions overlap.

The requirement that events carry provenance is what allows multiple later interpretations of the same recorded activity. If the originating input, the mapping version, and the operating conditions are kept, then a keystroke sequence entered under a randomized layout can be replayed deterministically under the mapping that was in force, and the result can be compared with the same physical sequence interpreted under a different mapping. Replay is exact when the adapter is a pure function of the recorded input and recorded conditions. It is approximate when the adapter depended on something that was never captured, such as the internal state of a speech recognizer since replaced. Backward compatibility has a hard limit here: activity that was never recorded cannot be reinterpreted, and no later improvement to an adapter can recover distinctions the original recording did not preserve.

Proposition 5.2 (Output equality is not interaction equivalence)**.** *Let $\alpha : \mathcal{H} \to V_\alpha$ be an artifact query in $\mathcal{Q}$. If $h_1, h_2 \in \mathcal{H}$ satisfy $\alpha(h_1) = \alpha(h_2)$ but $q(h_1) \neq q(h_2)$ for some $q \in \mathcal{Q}$, then every representation that is exact for $\mathcal{Q}$ assigns h_1 and h_2 distinct representatives.*

Proof. By hypothesis $h_1 \not\sim_{\mathcal{Q}} h_2$. By Proposition 4.2, an exact representation identifies only $\mathcal{Q}$-equivalent histories. □

Identical text dictated by voice and typed on a randomized keyboard therefore shares an artifact state while remaining two points of the semantic space, provided the system has declared any query that distinguishes them, such as one about timing, effort, or the evidence the interaction supplies about who performed it. Compatibility between interfaces is then a matter of specifying which relationships survive translation. Two interfaces may preserve the resulting text while differing in timing, gesture, and expressive possibility. Their overlap is partial, and the architecture records both the shared result and the distinctions belonging to each mode of participation.

6 Compatible Overlaps and Global Assembly

On regions that more than one interface can express, the presentations must agree. The agreement is formulated here for presentations whose overlaps admit invertible changes of coordinates, which is the case in which a clean gluing theorem is available. Lossy interfaces are discussed afterward.

For the purposes of assembly, regard each presentation i as a triple $(U_i, T_i, (q_i)_{q\in\mathcal{Q}})$, where $T_i \subseteq U_i \times E \times U_i$ is the set of interpreted local transitions and $q_i : U_i \to V_q$ gives the value of each declared query on local states. Each T_i is assumed deterministic: for given u and e there is at most one u' with $(u, e, u') \in T_i$.

Definition 6.1. An *atlas of presentations* is a family of presentations indexed by a set J, together with subsets $U_{ij} \subseteq U_i$ and bijections $t_{ij} : U_{ij} \to U_{ji}$ for $i, j \in J$, satisfying the following conditions.

(A1) $U_{ii} = U_i$ and t_{ii} is the identity; $t_{ji} = t_{ij}^{-1}$.

(A2) $t_{ij}(U_{ij} \cap U_{ik}) = U_{ji} \cap U_{jk}$, and $t_{jk} \circ t_{ij} = t_{ik}$ on $U_{ij} \cap U_{ik}$.

(A3) $q_j \circ t_{ij} = q_i$ on U_{ij} for every $q \in \mathcal{Q}$.

(A4) If $u \in U_{ij}$, $(u, e, u') \in T_i$, and $(t_{ij}(u), e, v) \in T_j$, then $u' \in U_{ij}$ and $t_{ij}(u') = v$.

(A5) t_{ij} maps $U_{ij} \cap \operatorname{dom}\phi_i$ onto $U_{ji} \cap \operatorname{dom}\phi_j$, with $\phi_j \circ t_{ij} = \phi_i$ there; and $q_i(u) = \bar{q}_{\mathcal{S}}(\phi_i(u))$ for every $q \in \mathcal{Q}$ and every $u \in \operatorname{dom}\phi_i$, where $\bar{q}_{\mathcal{S}} : \mathcal{S} \to V_q$ is the function induced by q on the quotient.

Condition (A2) is the usual overlap consistency requirement. Condition (A3) says that a change of interface does not change the answer to any declared query. Condition (A4) says that where two interfaces can both express the same event from the same shared state, they arrive at the same shared state. Condition (A5) ties the local apparatus to the semantic quotient: interpretation maps agree on overlaps, with matching domains, and the local value of a query is the value of the declared semantic query at the interpreted state. Without it the local answers could be mutually consistent and still unrelated to the semantics they are supposed to report. All five conditions are imposed for every ordered pair of indices, so each holds in both directions across an overlap.

Theorem 6.2 (Conditional assembly)**.** *Let an atlas of presentations be given. On the disjoint union $\bigsqcup_{i\in J} U_i$ define $(i, u) \approx (j, v)$ when $u \in U_{ij}$ and $t_{ij}(u) = v$. Then:*

(i) $\approx$ is an equivalence relation, and each U_i embeds injectively in the quotient $M = \bigsqcup_i U_i/\!\approx$;

(ii) each query $q \in \mathcal{Q}$ induces a well-defined function $\bar{q} : M \to V_q$ with $\bar{q}[(i, u)] = q_i(u)$;

(iii) the relation $T \subseteq M \times E \times M$ containing $([(i, u)], e, [(i, u')])$ whenever $(u, e, u') \in T_i$ is deterministic.

(iv) the interpretation maps assemble into a single partial map $\phi : M \rightharpoonup \mathcal{S}$ with $\phi[(i, u)] = \phi_i(u)$; on its domain $\bar{q} = \bar{q}_{\mathcal{S}} \circ \phi$ for every $q \in \mathcal{Q}$; and whenever $(x, e, y) \in T$ with x in the domain of ϕ, there is a transition $\phi(x) \xrightarrow{e} \phi(y)$ in $\mathcal{S}$.

Proof. (i) Reflexivity and symmetry follow from (A1). For transitivity, let $(i,u) \approx (j,v)$ and $(j,v) \approx (k,w)$. Then $v = t_{ij}(u) \in U_{ji} \cap U_{jk}$, which by (A2) equals $t_{ij}(U_{ij} \cap U_{ik})$. Since t_{ij} is injective, $u \in U_{ij} \cap U_{ik}$, and $t_{ik}(u) = t_{jk}(t_{ij}(u)) = w$. Hence $(i,u) \approx (k,w)$. Injectivity of $U_i \to M$ holds because $(i,u) \approx (i,u')$ means $u' = t_{ii}(u) = u$.

(ii) If $(i,u) \approx (j,v)$ then $q_j(v) = q_j(t_{ij}(u)) = q_i(u)$ by (A3).

(iii) Suppose (x,e,y) and (x,e,y') both lie in T, arising from $(u,e,u') \in T_i$ and $(v,e,v') \in T_j$ with $[(i,u)] = x = [(j,v)]$. Then $u \in U_{ij}$ and $t_{ij}(u) = v$. Applying (A4) to the ordered pair (i,j), with the transitions $(u,e,u') \in T_i$ and $(v,e,v') \in T_j$, gives $u' \in U_{ij}$ and $t_{ij}(u') = v'$, so $y = [(i,u')] = [(j,v')] = y'$.

(iv) By (A5), if $(i,u) \approx (j,v)$ then u lies in the domain of ϕ_i exactly when v lies in the domain of ϕ_j, and in that case $\phi_j(v) = \phi_i(u)$. Hence ϕ is well defined. The identity $\bar{q} = \bar{q}_{\mathcal{S}} \circ \phi$ is the second clause of (A5) read on representatives. The statement about transitions is the defining property of an action interpretation, applied in any presentation in which the transition arises. □

The theorem establishes a shared structure assembled from local ones, and it deserves a careful reading of what it does not establish. It does not say the assembled object is a manifold. That would require a topology on M under which each point has a neighborhood homeomorphic to an open subset of a Euclidean space, with the t_{ij} continuous, and the events of interest give no reason to expect uniform dimension. A region in which a performer varies three continuous parameters adjoins a region in which a single discrete choice is made. The appropriate general setting is more likely a stratified space [7], a complex of cells of varying dimension, or simply a typed graph whose nodes carry local continuous parameters. The last of these is what a first implementation should use, since it permits the stronger geometric claims to be tested instead of assumed.

Remark 6.3 (What a manifold would require). To call M a manifold one would need, beyond (A1) to (A5), a topology on M in which each U_i is open and each t_{ij} is a homeomorphism between open sets; a neighborhood of every point homeomorphic to an open subset of some $\mathbb{R}^n$, with n locally constant and the resulting charts compatible with the t_{ij}; and the usual separation and countability conditions. None of these follows from the hypotheses of Theorem 6.2, which are purely set-theoretic and are satisfied by discrete structures. The word "manifold" is therefore used in this project as a name for a hypothesis about particular regions, to be established region by region where continuous parameters are present.

The theorem also does not cover lossy interfaces, and these are the ordinary case. A viewer that shows text while discarding timing has no invertible coordinate change with an editor that records both. The correct relation is a projection, and its properties can be stated exactly.

Definition 6.4. A *derived view* of M is a set W together with a surjection $p : M \to W$. The view is *faithful* for a query $q \in \mathcal{Q}$ if $p(x) = p(y)$ implies $\bar{q}(x) = \bar{q}(y)$. It is *transition-compatible* if, whenever $p(x) = p(y)$ and $(x,e,x'), (y,e,y') \in T$, we have $p(x') = p(y')$. It is *enabledness-preserving* if, whenever $p(x) = p(y)$, each event e has a transition from x in T exactly when it has one from y. The *induced relation* is $T_W = \{(p(x), e, p(x')) : (x,e,x') \in T\}$.

Proposition 6.5 (Derived views)**.** *Let $p : M \to W$ be a derived view.*

(i) A query $q \in \mathcal{Q}$ can be answered from the view state, in the sense that $\bar{q} = q_W \circ p$ for some function q_W on W, if and only if the view is faithful for q.

(ii) The induced relation T_W is deterministic if and only if the view is transition-compatible.

(iii) If the view is not faithful for q, no procedure whose only input is the view state answers q correctly at every state of M.

(iv) If the view is transition-compatible and enabledness-preserving, then for every $x \in M$, every event e, and every $w \in W$, we have $(p(x), e, w) \in T_W$ if and only if there is x' with $(x, e, x') \in T$ and $p(x') = w$.

Proof. (i) is the argument of Proposition 4.2 with p in place of r and $\bar{q}$ in place of q; since p is surjective no extension is needed. (ii) Two elements (w, e, w_1) and (w, e, w_2) of T_W arise from transitions (x, e, x') and (y, e, y') of T with $p(x) = p(y) = w$, $p(x') = w_1$, and $p(y') = w_2$, so the requirement $w_1 = w_2$ for all such pairs is transition-compatibility. (iii) Choose x, y with $p(x) = p(y)$ and $\bar{q}(x) \neq \bar{q}(y)$ and argue as in Proposition 3.2. (iv) If $(x, e, x') \in T$ and $p(x') = w$ then $(p(x), e, w) \in T_W$ by definition. Conversely, an element $(p(x), e, w)$ of T_W arises from some $(y, e, y') \in T$ with $p(y) = p(x)$ and $p(y') = w$. Enabledness-preservation supplies x' with $(x, e, x') \in T$, and transition-compatibility gives $p(x') = p(y') = w$. □

A derived view thus has a definite set of queries it can answer, namely those constant on its fibres. Whether it can run on its own is a further matter with two parts, and the proposition separates them. Transition-compatibility makes the outcome of an event predictable from the view state whenever the event occurs. It does not make the view a reliable guide to which events can occur: two underlying states with the same view state may differ in whether an event is available, and the induced relation will then offer an action that the actual state does not admit. Part (iv) says that enabledness-preservation is what removes this defect. A view with both properties determines from its own state whether an action is available and what view state results, and can serve as an executable abstraction of M. A view faithful for the continuation query is enabledness-preserving, since the events available at a state are the one-step continuations. A view with neither property can still be displayed consistently. It is a display and nothing more: availability must be checked against the finer structure before an action is offered, and the view must be recomputed after each event. In the proposed environment the event history is authoritative, the assembled structure M is the finest presentation, and most interfaces, the tunnel explorer among them, are derived views of it.

7 Geometry from Admissible Transformations

What makes two states close? Visual resemblance and textual similarity contribute something, but the framework developed here gives priority to the transformations required to move between states. Two artifacts are close when a small authorized revision connects them, when they support similar next actions, or when they depend on the same unresolved condition. Distance is to be derived from admissible paths.

Definition 7.1. A *cost* is a function c assigning to each transition $x \xrightarrow{e} y$ of $\mathcal{S}$ a value in $[0, \infty]$. For a finite path γ of transitions, $\mathrm{cost}(\gamma)$ is the sum of the costs of its transitions, with the empty path having cost 0. The associated *path distance* is

$$d_c(x, y) = \inf_{\gamma : x \rightsquigarrow y} \mathrm{cost}(\gamma),$$

with $d_c(x, y) = \infty$ when no path exists.

Two ways of excluding a step are available, and they are given different jobs. A step forbidden by the rules to everyone is excluded by absence: the extended sequence is not in $\mathcal{H}$, there is no transition, and no cost can make it traversable. A step that exists in $\mathcal{S}$ but is unavailable from a particular standpoint, for instance to a participant lacking a grant, is assigned infinite cost by the cost function for that standpoint. The two agree in their effect on distance, since an absent transition and an infinitely costly one contribute nothing finite to any infimum. They differ in what a view may show. An infinitely costly transition is a door that exists and is closed to this participant, and an absent one is a wall.

Proposition 7.2. *For every cost c, the function d_c satisfies $d_c(x, x) = 0$ and $d_c(x, z) \leq d_c(x, y) + d_c(y, z)$. It need not be symmetric, and $d_c(x, y) = 0$ does not imply $x = y$.*

Proof. The empty path gives $d_c(x, x) = 0$. If either term on the right of the inequality is infinite there is nothing to prove. Otherwise, for any $\epsilon > 0$ choose paths $\gamma_1 : x \rightsquigarrow y$ and $\gamma_2 : y \rightsquigarrow z$ with costs within ϵ of the respective infima. Their concatenation is a path from x to z of cost at most $d_c(x, y) + d_c(y, z) + 2\epsilon$. For asymmetry, take a single transition $x \to y$ with no path back. For the failure of separation, take a transition of cost zero between distinct states. □

The structure obtained is an extended directed pseudometric, which is exactly a generalized metric space in Lawvere's sense [11]. Its asymmetry is the formal trace of irreversibility. The cost of publishing and the cost of unpublishing are different quantities, and in many cases the second is infinite. There is also no single distinguished cost. Physical effort, elapsed time, conceptual prerequisite, authorization requirement, and irreversible consequence each define a distance, and these distances disagree. Two states one keystroke apart may be separated by an authorization that takes a week to obtain. A useful geometry retains the whole family $(d_c)_c$ and lets each view declare which member it is displaying.

The dependence of this geometry on history can be stated sharply. A coarse family of queries can produce a state space with cycles, in which a participant appears to return to an earlier state, though coarseness alone does not force them. A sufficiently fine family excludes them.

Proposition 7.3 (Recurrence is a property of projection)**.** *Suppose $\mathcal{Q}$ contains the length query $n(h) = |h|$. Then the transition graph of $\mathcal{S}$ has no cycles. In particular, for distinct states x, y there cannot be paths both from x to y and from y to x, so for any cost c at most one of $d_c(x, y)$ and $d_c(y, x)$ is finite.*

Proof. The induced function $\bar{n}$ on $\mathcal{S}$ increases by exactly one along every transition, so no path of positive length returns to its starting state. Paths from x to y and from y to x with $x \neq y$ would concatenate to such a path. Finally, $d_c(x, y) < \infty$ requires that some path from x to y exist, since the infimum over the empty set is ∞. □

The proposition gives precise content to the claim that reaching an apparently familiar state does not restore the earlier situation. Whether return is possible depends on which distinctions are being tracked. With respect to the text of a document, an edit followed by its reversal is a return. With respect to a family of queries that includes who has seen the intermediate version, it is a displacement to a new state from which the earlier one cannot be reached. A participant who has acquired knowledge, disclosed a secret, or committed another person to a

dependency occupies a state whose artifact projection may coincide with an earlier one and whose continuation set does not. The environment's geometry therefore evolves with activity: a path can open possibilities for one participant, close them for another, and change the evidential meaning of a later action.

Definition 7.4. Fix a cost c and a budget $b \in [0, \infty]$. The *tunnel* from x to y at budget b is the set of admissible paths $\gamma : x \rightsquigarrow y$ with $\mathrm{cost}(\gamma) \leq b$.

Tunnels are passages through admissible states, and their shape is determined by what is excluded. A narrow passage is a stretch along which few alternatives remain admissible. A junction is a state from which several continuations are available. A closed passage is a transition present in the space whose cost from the viewer's standpoint is infinite, because an obligation is unmet, a grant is missing, or a dependency is unresolved. Reading, programming, performance, and exploration generate different kinds of paths through the same structure. The spatial rendering of this structure in a tunnel explorer must preserve branching, return, dependency, and obstruction, since these are the relationships the rendering exists to convey. It must also avoid suggesting that every semantic relationship has a unique physical location. A layout is a computed projection, reproducible from a declared key consisting of the ledger version, the layout algorithm and its version, its parameters, and its random seed [5], and what it places side by side is side by side by a choice of embedding. The authoritative relationships remain those reconstructible from events.

8 Provisional Sessions and Hash Identities

A new session receives an identifier without supplying a username. Its founding event contains a freshly generated random nonce and a public key whose private half stays with the session, and the *founding identifier* is the cryptographic hash of that event under a fixed canonical encoding. The nonce makes it overwhelmingly unlikely, though not impossible, that two sessions beginning in otherwise identical circumstances receive the same identifier, and the canonical encoding ensures that the same event hashes to the same value whichever implementation computes it. Every later event lists the hashes of its parents, so the hash of any event commits to its entire causal past in the manner of a hash chain [13]. Three identifiers then have to be kept apart. The founding identifier is stable and names the session for as long as it exists. The hash of any later event identifies a causal subgraph, namely that event together with all its ancestors, and the hash of a session's most recent event is its *head identifier*. Neither identifies an execution history. Events admitted earlier in a region's record that are concurrent with a given event are not among its ancestors, and its hash says nothing about them. The order of admission distinguished in Section 3 therefore receives a commitment of its own. For a region whose record has admitted $e_1, \ldots, e_k$, with H the hash function, put

$$L_0 = H(\text{region initialization}), \qquad L_k = H\big(\text{encode}(L_{k-1}, H(e_k))\big).$$

The *log commitment* L_k names the execution history $e_1 \cdots e_k$, and it is this value, not an event hash, that identifies an element of $\mathcal{H}$. The pair is serialized under the same canonical encoding as events, for the same reason.

What a hash supplies is a computational commitment to encoded content. Finding different content with the same hash is assumed infeasible, so no party can alter an earlier event without

changing every identifier downstream of it. A hash establishes nothing further. Knowing a head identifier does not show possession of the history it names, since identifiers circulate freely, and equality of hashes does not authenticate who wrote the content. Authorship in the narrow sense is supplied by signatures: an event signed under the founding key was produced by a holder of that key. Even this concerns keys and not persons. Whether the person now acting is the person who produced some earlier history is a separate question, and it has a probabilistic answer at best. The environment therefore distinguishes two kinds of statement. The first kind concerns histories and is exact up to the cryptographic assumptions: this event belongs to that causal past, or this sequence of admissions is the one committed to by that log value. The second concerns the association between histories and is evidential: the trajectory recorded under one provisional identifier is, on present evidence, likely to continue the trajectory recorded under another.

Associations are themselves events. An association between two histories is recorded with the evidence on which it was proposed and the policy under which it was accepted. Histories may remain separate, may become provisionally associated, or may have an association revoked. Revocation appends a further event and leaves the association and its grounds in the record. This matters for two reasons. An erroneous association that is simply deleted leaves no trace of the actions taken on its strength, and those actions may need to be reviewed. And the pattern of associations proposed and withdrawn is the principal data from which the recognition procedure's errors can be measured. Provenance survives correction because correction is recorded as a continuation and never as an erasure.

9 Evidence of Continuity

Within the geometry of Section 7, a provisional identifier collects a trajectory, and recognition is the estimation of whether a new trajectory continues an earlier one across changes of interface and circumstance. The evidence available for such an estimate consists of situated capabilities: practiced movements, knowledge of routes, musical phrasing, habitual workflows, characteristic recoveries from error, and the manner in which an interrupted activity is resumed. Behavioral signals of this general kind have been studied for decades in the narrower setting of keystroke timing [14]. The proposal extends the idea from a single channel to patterns of transition across several, and that extension is the part most in need of experimental support.

The model is built around competing explanations. A trajectory that resembles an earlier one may have been produced by the same person, by someone coached by that person, by someone following a published walkthrough, by a replay of recorded input, or by an automated agent trained on the earlier history. A trajectory that fails to resemble an earlier one may come from a different person, or from the same person using unfamiliar hardware, a shared device, or an assistive technology, or from the same person who is tired, injured, or out of practice. A mismatch should accordingly produce a revised set of hypotheses with revised weights. It should never produce an automatic accusation, and it should never by itself reassign an identity. Disability and fatigue in particular must be modeled as ordinary sources of variation, since a recognition system that treats departure from a motor baseline as evidence of imposture will lock out precisely the people least able to absorb the cost.

Let I denote the hypothesis that the present actor is the one associated with a candidate history, let B be observed behavior, and let D be the information environment, including all instruction publicly available at the time of observation. The evidential force of B is expressed

by the likelihood ratio

$$\Lambda(B; D) = \frac{P(B \mid I, D)}{P(B \mid \neg I, D)},$$

in the tradition of weighing evidence by the factor it contributes to the odds [6]. No numerical values for such quantities are offered in this essay. They would have to be estimated from data the project does not yet have, and the propositions below are stated so as to hold whatever the values turn out to be.

Proposition 9.1 (Conditional disclosure)**.** *Let D and D' be information environments with $P(B \mid I, D') = P(B \mid I, D) > 0$ and $P(B \mid \neg I, D') > P(B \mid \neg I, D) > 0$. Then $\Lambda(B; D') < \Lambda(B; D)$.*

Proof. The numerators agree and are positive, and the denominator has strictly increased. □

The proof is immediate, and the weight lies in the hypotheses. Positivity of the common numerator is needed, since behavior the candidate would never produce has likelihood ratio zero before and after disclosure. That case is degenerate, the observation refuting I outright, and it is excluded for that reason. Disclosure does not reduce the force of every signal. A walkthrough reveals which route to take through a difficult passage. It conveys much less about the timing with which a practiced participant takes it. Publication may also change the candidate's own behavior, in which case the first hypothesis fails. What the proposition does specify is the condition under which a formerly distinctive piece of knowledge loses identifying force: others have become more able to reproduce it while the candidate's ability is unchanged.

This connects recognition to discovery. The environment contains latent specifications, regularities that participants can encounter and learn through activity. For each such regularity the system tracks four separate events: its implementation, the first observed recognition of it by a participant, its public disclosure, and its demonstrated reproducibility by others. The discovery remains attributed to its discoverer permanently, as a matter of provenance. The identifying value of knowing the regularity declines as it becomes reachable through instruction. The underlying location is unchanged while the social geometry around it changes substantially: a formerly obscure passage becomes widely reachable, and traversing it says less about who is traversing. The endpoint remains observable, and the history explaining how access to it was acquired becomes the informative thing. One caution governs the use of this tracking. The absence of an observed disclosure is uncertainty about whether disclosure occurred. It is never proof of secrecy, since instruction passes through channels the system cannot observe.

Three variables are in play here and they have to be kept separate: when a piece of information became available, whether a given actor acquired it, and whether that actor demonstrates the corresponding capability. Detecting a public walkthrough establishes an opportunity for acquisition and does not establish who made use of it. Failing to detect one does not establish that a successful actor found the route unaided. Nor can the gap be closed by asking. An actor's own account of what it knows, or a session's presentation of itself as unfamiliar with a route, is one more observation to be explained and carries no special authority about how a capability was acquired. Recent work on language models makes the point concretely in a neighboring domain. Qian and colleagues [16] tested whether such models can tell dated financial facts they already know from outcomes they are predicting. On their controlled facts, a subset chosen so that the simple strategy of repeating the previous month's value fails, the strongest model answered 76 percent correctly while claiming to know 38 percent,

and answered correctly 61 percent of the facts it said it did not know. Instructing it to answer as of a historical date reduced its claims of prior knowledge from 39 percent to 1 percent while leaving its accuracy essentially unchanged. Two of the other three frontier models tested showed the same pattern. The fourth moved its claims in the opposite direction, again with no material change in accuracy. The authors conclude that a stated cutoff, a self-report, and a declared date each fail to separate known from unknown outcomes, and they recommend measuring knowledge boundaries from behavior. The study concerns dated facts in one economy, and its authors note that a model can be right without recall when it extrapolates a trend, so their measured boundaries do not certify the absence of prior exposure. It demonstrates nothing about behavioral recognition or about the propositions of this section. Its bearing is methodological. Where automated agents can take part, imitate participants, or absorb published technique, a declaration about the conditions of a performance does not establish the conditions under which the capability was acquired. Provenance records therefore need to be accompanied by behavioral audits that probe what an actor can in fact do, and even a successful traversal observed under audit is not taken to reveal its own acquisition history.

Evidence accumulates across observations, and the manner of accumulation is where naive schemes go wrong. For observations $B_1, \ldots, B_n$ the chain rule gives

$$\log \Lambda(B_{1:n}; D) = \sum_{k=1}^{n} \log \frac{P(B_k \mid I, B_{<k}, D)}{P(B_k \mid \neg I, B_{<k}, D)},$$

wherever the conditional probabilities are defined and positive. Each term measures what the kth observation adds given the earlier ones.

Proposition 9.2 (Determined observations add nothing)**.** *Suppose there is a single function f such that $B_k = f(B_{<k}, D)$ almost surely under both I and $\neg I$. Then for every realized value of $B_{<k}$ that has positive probability under both hypotheses, the kth term of the sum is zero.*

Proof. Given such a value of $B_{<k}$, the realized B_k equals $f(B_{<k}, D)$ and has conditional probability one under each hypothesis. □

The requirement of a common function is essential. If the next observation were determined under each hypothesis by a different function, observing it could be decisive. The proposition applies to a replayed recording, a macro, or an adaptive control that reproduces an earlier gesture once the generating process is known: its further outputs are then determined by what has already been observed and add nothing. The discovery that a macro or a replay is in use is another matter and can itself be informative, since it bears directly on which of the competing explanations holds. What has to be avoided is counting each repetition as a fresh confirmation. The same caution applies in weaker form to observations that are merely correlated through shared instruction or common environmental conditions: two participants taught by the same teacher share mannerisms that are evidence of the teacher. The system must therefore distinguish the number of observations from the amount of independent evidence, and must model the principal sources of dependence explicitly.

A separate device, which should not be confused with the preceding analysis, is a scoring rule with diminishing returns, such as a score growing with the logarithm of the count of observations of a given kind. The concavity of the logarithm is a mathematical fact, and it guarantees that the thousandth repetition of a behavior moves the score much less than the first. That is a sensible policy. It is not a derivation of calibrated confidence. Whether such a

score corresponds to any probability of identity is an empirical matter that requires validation against known outcomes. Three quantities are thus kept apart in the design: an evidential score computed by a declared rule, a calibrated probability if and when one has been validated, and a permission decision, which Section 10 makes on different grounds altogether.

Proposition 9.3 (Identifiability limit)**.** *Let I and J be hypotheses about who is acting, and suppose they induce the same probability distribution over every observation available to the system. Then for every prior assigning positive probability to both hypotheses, and every observation B of positive probability, the posterior odds of I against J equal the prior odds.*

Proof. By Bayes' rule the posterior odds are the prior odds multiplied by $P(B \mid I)/P(B \mid J)$, and this ratio is one. □

No recognition procedure can do better than its information permits. If a coached imitator, a replay, or an automated agent produces observations distributed exactly as the original participant's are, no quantity of further observation of the same kind distinguishes them. More data helps only when it discriminates. This is the reason coaching, replay, automation, and shared environments enter the model as explicit alternatives. Assuming them absent does not make a recognizer stronger. It makes its stated confidence wrong. Recognition by continuity is and remains evidence subject to revision, and the geometry of a trajectory never amounts to a proof of personhood.

10 Authority Under Uncertain Identity

Since recognition is fallible in principle, permissions cannot be a function of recognition scores alone. The design separates the two and grades authority by reversibility. A provisional session may immediately explore, read what is public, create new artifacts, and propose changes, since all of these are recoverable: an unwanted proposal can be declined and an unwanted artifact ignored. Operations whose effects cannot be undone, including deletion, publication, transfer of ownership, and changes to recovery arrangements, lie behind what this project calls the root shield, and they require authorization of a stronger and independent kind. The reversibility gate is the rule that assigns each operation to one side or the other according to whether its effects can be withdrawn.

The security content of this arrangement can be proved under the classical assumptions of complete mediation and capability enforcement [1, 17, 3]. The model has to be explicit about what is checked, since a monitor that tests only the name of an operation protects very little. A *request* is a tuple $\rho = (a, x, o, \theta, \delta)$ consisting of an acting principal a, a target resource x, an operation o, its arguments θ, and a delegation scope δ recording on whose authority and within what limits a claims to act. Principals are sessions and services. In each state every principal a holds a set $\mathrm{cap}(a)$ of *grants*, and a fixed relation says when a grant *covers* a request, taking all five components into account. Each session also carries a behavioral score. A request ρ is *made for* a principal b if b is its acting principal or its delegation scope names b, and ρ_b denotes the request with the same resource, operation, and arguments, with b as acting principal and an empty scope. A *reference monitor* is a component that executes a request ρ with acting principal a only if some grant in $\mathrm{cap}(a)$ covers ρ and, for every principal b named in its delegation scope, some grant in $\mathrm{cap}(b)$ covers ρ_b. A service acting for a session can thus do only what the service and the session could each have done. The *protected class* P is a declared

set of requests, intended to contain those whose effects cannot be withdrawn. Membership in P depends only on the effect requested, that is, on the resource, the operation, and the arguments, so that $\rho \in P$ implies $\rho_b \in P$ for every principal b. A grant is *protected* if it covers at least one request in P. A *credential* for a principal is an object whose validity is decided by a predicate that takes no behavioral score as input, such as a signature under a held key.

Theorem 10.1 (Conditional containment)**.** *Let s be a provisional session and P a protected class. Assume:*

(C0) every step of the system is either a score update, which changes behavioral scores and no capability set, or the execution of a request by the reference monitor;

(C1) every effect of a request in P, and every change to a capability set, including issuance, delegation, revocation, and the acceptance of a credential, occurs only as the execution of a request by the monitor;

(C2) in the initial state of s, the set $\mathrm{cap}(s)$ *contains no protected grant;*

(C3) a step adds a protected grant to $\mathrm{cap}(a)$ *only by issuance, whose precondition, checked by the monitor, is a valid credential for a, or by delegation, a request that names a as recipient and whose acting principal already holds a protected grant covering everything the new grant covers;*

(C4) the delegation scope of a request is attached by the monitor and cannot be removed by the issuer, so that every request issued by any principal in the course of serving a request made for s is itself made for s;

(C5) no step that creates, re-issues, or recovers a credential, and no delegation of a protected grant, has a precondition that depends on a behavioral score.

Then in any execution in which no valid credential for s is accepted and no delegation of a grant names s as recipient, no request in P made for s is executed, whatever values the behavioral scores of s or of any other session take. In particular s executes no request in P, and no service executes one in the course of serving a request made for s. Moreover, whether the hypothesis of this conclusion holds is unaffected by the value of any behavioral score.

Proof. By induction on the length of the execution, $\mathrm{cap}(s)$ contains no protected grant in any state. This holds initially by (C2). A score update leaves every capability set unchanged by (C0). Any other step is the execution of a request by the monitor, and by (C3) it can add a protected grant to $\mathrm{cap}(s)$ only through a credential for s or a delegation naming s, both excluded by hypothesis. Now let $\rho \in P$ be made for s. By (C1) its effect occurs only if the monitor executes it. If s is the acting principal, execution requires a grant in $\mathrm{cap}(s)$ covering ρ. If the delegation scope names s, execution requires a grant in $\mathrm{cap}(s)$ covering ρ_s, and $\rho_s \in P$ because membership in P depends only on the effect. In either case the grant would be protected, and there is none. The final assertion follows from (C4), under which every request issued in the course of serving a request made for s is made for s. The last sentence of the theorem restates (C5): the steps by which a credential for s could come to exist, and those by which a protected grant could be delegated to s, have preconditions in which no score appears. □

The theorem says nothing in favor of the recognition procedure. It proves that one kind of consequence of that procedure's errors can be bounded by architecture: a mistaken behavioral match, taken alone, cannot bring about any effect in the protected class. The scope of that statement is exactly the class P and the components the covering relation examines. Deletion, publication, transfer, and changes to recovery are contained if they are placed in P. Reading what a session was wrongly allowed to read, disclosing it elsewhere, and consuming resources can all do lasting harm, and they are contained only to the extent that the corresponding requests are also declared protected. The theorem does not show that mistaken recognition is harmless. It shows that the harm can be confined to whatever lies outside a class the designers choose and must defend.

The hypotheses are substantive engineering obligations. Complete mediation fails whenever some path to a protected effect bypasses the monitor, and the acceptance of a credential must itself be a monitored operation, since a credential check performed elsewhere could be satisfied without the monitor's knowledge. Two things called delegation appear in the theorem and should not be confused. Delegation of a grant, governed by (C3), moves authority from one principal to another. Delegation scope, governed by (C4) and the monitor's rule, records that a principal with authority of its own is acting for another. The second is what prevents a session from laundering a protected request through a service that holds protected grants: the service's request carries the session in its scope, and the monitor refuses it because the session could not have made the request itself. Condition (C4) is what makes this dependable, since a scope that the issuer could strip would protect nothing. The covering relation and the attachment of scopes are accordingly part of the trusted computing base. They must be decidable, and simple enough to audit, since every guarantee in this section is relative to their correctness. Independence of credentials has two layers, and both are needed. A validity predicate that takes no score as input is insufficient if the credential itself can be obtained by being recognized, since recognition would then reach the protected class by one further step. Condition (C5) closes that route by extending the requirement to issuance, re-issuance, and recovery, and this determines how recovery must be designed. When recognition fails for a legitimate participant, as it will after injury or a long absence, recovery has to proceed through an independent channel, such as a held key, a recovery code, or attestation by designated others, and changes to those arrangements are themselves in the protected class.

Between the provisional floor and the root shield lies a middle range in which behavioral evidence may properly matter. Condition (C0) forbids a score update from changing capabilities directly, and it leaves room for a monitored elevation operation that consults the score and adds grants that are not protected, such as permission for a session to edit its own earlier work without review. An error here costs a reversal and nothing more. Recognition thereby has a useful role while its failures are confined to the region where failures are cheap. The same arrangement is the principal protection for participants whose behavior legitimately varies. A person whose motor patterns change through disability, injury, or fatigue may fail to be recognized, and the cost of that failure is bounded. Provisional participation remains available to them. Permissions in the middle range that had been extended on behavioral evidence may have to be regained, through renewed recognition or through an independent credential, and that is a real cost though a recoverable one. Everything above the root shield was never reachable by behavior in the first place.

11 Admissibility, Verification, and Refusal

A proposed change is a candidate continuation of a history, and a region of the environment accepts or declines it according to its acceptance conditions. The narrowest layer of admissibility is the one defined in SpherePOP [4]: an application is admissible when its operands lie within the domain on which the operator is defined, and an application outside that domain remains in the term as an inert circle, neither executed nor treated as a fault. Admissibility in the sense of Section 3 is broader. A continuation belongs to $\mathrm{Cont}(h)$ only if it is defined in this narrow sense and also satisfies the authorization, evidence, and policy conditions of the region. The rest of this section stipulates the apparatus that governs those further conditions. Two structural commitments come first. The first is the separation of the event history from the witness plane: the record of what was done and proposed is kept apart from the record of what was checked, by whom, with what result. A verification is evidence about a history and does not become part of the history it concerns, so re-verification under a revised procedure adds to the witness plane and leaves the event history untouched. For acceptance to remain a function of the history, as Section 4 requires of every query, an acceptance event cites by hash the exact entries of the witness plane on which it relied. The witness plane may then grow or be revised without changing what any existing history says was relied upon, and the inputs to every past acceptance are reproducible from the history together with the cited entries. The same discipline has to govern the future. Whether a continuation is admissible may depend only on witness entries that are immutable and cited in the history, and on policy versions that are themselves recorded as events. It may never depend on an unrecorded "latest" state of the witness plane, since two holders of the same history could then disagree about what it permits. The second is that obligations are recorded whether they were discharged or waived. A waived obligation is a fact about the accepted continuation and remains visible. Whether it also matters for the future depends on policy. Where a region's rules make a waiver bear on what may later be accepted, two histories with identical artifacts that differ in the waiver have different continuation sets, and by Proposition 3.2 a system that forgot the waiver could not respect the difference.

Verification has four distinct outcomes: PASS, FAIL, INDET, and ERROR. The first two report that a check ran and reached a verdict. INDET reports that the check ran and could not decide. ERROR reports that the check did not run properly. Collapsing the last two into failure would misdescribe the situation in both cases, treating an undecided property as refuted and a broken tool as a judgment on the work. Keeping them apart lets the acceptance policy of a region say explicitly what it does with each.

Refusal is structured. It resembles the inert application in one respect and differs from it in another, and the difference should stay explicit. An application outside the domain of a partial function is undefined under that operator: there is no result it could have had. A refused proposal may be well formed and fully executable, and is declined because an authorization, a piece of evidence, or a policy condition is missing. The first is a fact about the operator and the second a decision by a region, which different conditions or a later reconsideration could reverse. The comparison carries one point only: both are retained in the record and can be revisited. A declined proposal is retained together with the conditions that were unmet, so that the proposer knows what would have to change and later readers know that the alternative was considered. Reconsideration proceeds through REOPEN, which resumes a declined or closed matter as a new continuation carrying its predecessor's record. Competing proposals are

preserved side by side. The environment does not require that a region's history be a single line with the alternatives discarded, and the alternatives are often the most informative part of the record, since they show which continuations were available and on what grounds one was preferred.

12 Public Rules and Confidential Checks

The environment contains regularities of three kinds, and conflating them would undermine its legitimacy. Published governance commitments state what the system may and may not do to a participant: which operations are protected, what evidence is required for elevation, how an association may be contested, and what happens to recorded activity. These are public and fixed in advance. Undiscovered environmental regularities are the latent specifications of Section 9. They are discoverable through activity, and their being unannounced is part of what makes finding them informative. Confidential verification procedures are checks whose details are withheld so that they cannot simply be rehearsed.

The third kind raises an obvious concern. A consequential decision made by a hidden procedure appears arbitrary to the person it affects. The resolution is to confine confidentiality to the instance while publishing the class. The rules state what sort of evidence a check may consider, what a check is permitted to decide, and what recourse follows an adverse result. The particular challenge varies and is not announced. Since Theorem 10.1 keeps protected operations out of reach of behavioral evidence altogether, the decisions that confidential checks influence are confined to the recoverable range, where an adverse result delays a participant and deprives them of nothing permanently.

Contest must be a real mechanism. A participant who believes an association mistaken, whether a false link to another's history or a failure to recognize their own, can open a challenge that is itself an event, is examined on independent evidence, and ends in a recorded outcome with the association upheld or revoked. Hidden metrics must also be audited for what they measure. A metric intended to track continuity of a person may in fact track continuity of a device, a keyboard, a network location, or a time of day, and it will then perform well until the incidental proxy changes and fail without warning afterward. Evaluating whether a confidential check measures continuity itself requires testing it against cases in which the person persists and the proxy changes, and cases in which the proxy persists and the person changes.

13 Repair and Collaboration

Repair is a constitutive operation of the environment. A mistaken association, a broken dependency, a failed transformation, or a misleading representation creates an obstruction, and the response to an obstruction is a recorded correction. The system preserves what was attempted, what prevented continuation, and what revision made continuation possible. Following the specification [5], a repair event also records the scope traversed in looking for dependents, the graph version examined, and whether the search for dependents was complete, partial, or of unknown completeness, and each dependent found receives a recorded status of its own. A repair that claims no more than it checked can be extended later without being contradicted. Successful repairs alter the organization of the space. A correction that is commonly needed becomes an accessible route, with low cost in the geometry of Section 7. A

failure that recurs across participants points to a defective mapping in some interface adapter or an inadequate description of the task. Over time the environment accumulates knowledge about how work can continue, and that knowledge is stored in the same form as everything else, as typed and attributable events.

Collaboration is the encounter of histories that partially overlap. Participants share artifacts, dependencies, or goals while approaching them through different interfaces and holding different knowledge. Their trajectories need not be merged into a single account. Systems for concurrent editing have shown that independently produced histories can be reconciled without a central arbiter under suitable algebraic conditions [18], and the local-first tradition has argued for keeping each participant's record primary [9]. The present design takes over that orientation and adds a requirement of declared reconciliation. Lines of work that assign conflicting values to the same artifact cannot all have their queries preserved unchanged by a merge. Each region therefore declares a reconciliation rule stating which queries a merge must preserve from every line it joins, such as provenance and outstanding obligations, and which receive a newly resolved value, such as the contents of a contested artifact. A merge is admissible when it satisfies that rule, and the resolution it chose is recorded in the merge event.

A repository is then a bounded region of the wider structure, comprising a set of artifacts, their recorded transformations, and acceptance conditions. The familiar operations keep their precise operational meanings and gain a description in the present terms. A fork is a pair of events sharing a parent, each beginning its own line of continuation. A merge is an event with more than one parent that satisfies the acceptance conditions and the reconciliation rule. A review is an entry in the witness plane. A refusal is a recorded decision that a proposed continuation is not accepted, with its reasons. The causal ordering of events across participants is partial, as it is in any distributed system [10], and it is carried by the parent relation. The execution history of a region linearizes that order in the manner described in Section 3, and the environment does not present the order of admission as a causal claim.

14 Storage, Execution, and Interoperability

The implementation follows from the formal structure. Content is stored immutably and addressed by its hash, so that an artifact version, once written, cannot change beneath those who refer to it. Event histories are append-only chains over that store. Authority is the one part of the system that requires coordination, since two parties must not both be told that they hold an exclusive capability, and updates to capability sets therefore pass through a coordinated service that implements the reference monitor of Theorem 10.1. Computation, including interface adapters, verification procedures, and replay, runs in disposable workers that hold no authoritative state: a worker reads events, computes, and emits proposed events or entries for the witness plane, and its loss costs only the repetition of the computation. Views, including document renderings, repository browsers, timelines, musical displays, and the tunnel explorer, are derived from histories and can be discarded and rebuilt.

Interoperation with Git is necessary if existing projects are to participate [2]. Import treats each commit as an event whose provenance records that it arrived by import, with the commit graph mapped to the parent relation. Interaction history is absent for imported material, and the import records that absence explicitly. Export projects a region onto its artifact states and their ancestry and writes them as commits, discarding what Git cannot express. Export is a derived view in the sense of Proposition 6.5: artifact queries are constant on its fibres and

survive a round trip through Git, while interaction queries do not. Accepting that loss is a design choice made for the sake of interoperation. The propositions describe its extent and do not license it.

An environment that records activity across applications holds sensitive material, and privacy has to be designed in from the start. Payloads are encrypted under keys held by participants, so the storage layer cannot read the content it stores. The coordinated service that implements the reference monitor works from a different and smaller body of information: the envelope of each event, comprising its type, parent hashes, founding identifier, and signature, together with grants and requests. It decides who may do what to which resource without reading what the resource contains. Complete mediation in the sense of (C1) is accordingly a claim about effects on the shared record. A participant who holds a key can read the corresponding content locally and repeat it elsewhere, and no monitor mediates that. Checks that must inspect content, such as verification, run in workers to which a participant has delegated a key for the purpose, and their results enter the witness plane. The envelope is itself revealing, since patterns of activity can be read from it. What it exposes is a cost of coordination, to be minimized and declared. Retention is governed by declared policy, and Proposition 4.2 states the price of discarding events: only the queries for which the retained summary is exact remain answerable. Redaction is handled by removing content from the store while retaining its hash in the chain, so that the history's integrity survives and the fact of redaction is itself visible. The price is real. Replay through a redacted event is no longer possible, queries that needed the payload become unanswerable, and copies already held elsewhere are beyond the system's reach. A further distinction needs emphasis because it is easy to blur. Observing activity in an application and holding permission to operate that application are separate grants. An adapter that records what a participant does in an editor receives no capability to issue commands to the editor on the participant's behalf, and any such capability is a consequential grant subject to the monitor.

15 A Staged Prototype and Research Programme

The first stage is a small shared event system connected to two interfaces of different character: a text editor and a MIDI instrument. Each records events with provenance to a common store. A tunnel view is generated from the same histories, with nothing recorded specially for it. This stage tests the architectural core, namely the propositions of Sections 4 through 7, in the form of a typed graph with local continuous parameters. Its measures are replay fidelity, meaning whether recorded activity reproduces under the recorded mapping; query preservation across interfaces, meaning whether the conditions of Theorem 6.2 hold where the two interfaces overlap; and the practical cost of keeping full histories.

The second stage introduces behavioral matching in observation only. The recognition procedure proposes associations among provisional sessions, the proposals are recorded, and they control nothing. Participants who consent supply ground truth through independent credentials. This yields measurements of the two error rates that matter, false association of distinct people and failure to recognize a returning participant, under deliberately varied conditions including changes of hardware, coached imitation, replay, and assistive input. It also permits a direct test of the disclosure model: a route or technique is published, and the change in its discriminating value is measured against the prediction of Proposition 9.1.

Only after those measurements exist would the third stage allow recognition to affect

permissions, and then only within the recoverable range. Accessibility is evaluated at every stage, with particular attention to whether participants whose motor behavior is variable are recognized less reliably and what the cost of that is to them. The spatial interpretation is evaluated against alternatives, including directory browsing, keyword search, and typed graph navigation without spatial layout, on the task it is supposed to serve: whether someone returning to interrupted work finds their way back to it and continues. Success for the architecture is defined independently of all this. The architectural core succeeds if replay fidelity is high in the first stage and the atlas conditions hold where the editor and the instrument overlap. The later stages test the recognition and spatial hypotheses, and their failure would leave that result standing.

Each experimental claim has a stated condition of failure. If false associations cannot be brought low enough under coached imitation, recognition remains advisory. If disclosure does not reduce discriminating value as modeled, the disclosure tracking is retained for provenance and removed from evidence weighting. If the tunnel view does not outperform plainer views for recovery of work, it becomes an optional rendering and the architecture loses nothing, since no part of the shared structure depends on it.

16 The Integration Theorem and the Status of the Claims

The constructions of the preceding sections combine into a single statement of what the architecture can be shown to deliver.

Theorem 16.1 (Integration)**.** *Let $\mathcal{H}$ be a prefix-closed set of histories, and let $\mathcal{Q}$ be a family of queries closed under extension and containing the continuation query and an authority query. Suppose the participating interfaces form an atlas of presentations satisfying* (A1) *to* (A5)*, whose interpreted actions are events carrying provenance, and suppose the system satisfies conditions* (C0) *to* (C5) *of Theorem 10.1 for a provisional session s and a protected class P. Then:*

(i) the semantic state space $\mathcal{S}$ carries a well-defined deterministic transition structure, and every exact representation of histories refines it;

(ii) the presentations assemble into a common structure M with a partial map $\phi : M \rightharpoonup \mathcal{S}$, on which every declared query has a value independent of the interface through which a state is reached and equal to the value of the semantic query at the interpreted state;

(iii) every assignment of costs to transitions yields an extended directed pseudometric on $\mathcal{S}$, and hence a geometry of admissible paths;

(iv) in any execution in which no valid credential for s is accepted and no delegation of a grant names s as recipient, no request in P made for s is executed, whether by s or by a service acting for it, whatever values the behavioral scores take.

Proof. Part (i) is Lemma 4.4 together with Proposition 4.2. Part (ii) is Theorem 6.2. Part (iii) is Proposition 7.2. Part (iv) is Theorem 10.1. □

In plain terms, given sufficient representations of history, compatible interface interpretations, provenance-preserving transitions, and completely mediated authority, heterogeneous

applications can participate in one semantic environment while preserving the continuation and authorization distinctions the environment has declared. The histories in question are execution histories, and the result is silent on whether two orders of admission of the same causal graph are equivalent. The theorem is conditional, and its conditions are the engineering specification. Verifying that a particular adapter satisfies the overlap conditions, or that a particular deployment achieves complete mediation, is work the theorem does not do.

The claims of the essay accordingly fall into three groups. Those proved here, under stated assumptions, are the insufficiency of observation, the criterion for exact representation, the congruence that gives the state space its dynamics, the conditional assembly of presentations and its agreement with the semantic queries, the characterization of derived views, the pseudometric structure, the acyclicity of sufficiently fine state spaces, the conditional effect of disclosure on a likelihood ratio, the vacuity of determined observations, the identifiability limit, and the containment of mistaken recognition with respect to a declared protected class. Those that are design commitments, justified by argument and not by proof, are the choice of events with provenance as the authoritative record, the treatment of associations and revocations as events, the grading of authority by reversibility, the separation of event history from witness plane with four verification outcomes, and the confinement of confidentiality to instances within published classes. Those that are experimental are whether behavioral continuity across interfaces recognizes returning people at acceptable error rates, whether any evidential score can be calibrated, whether disclosure-aware weighting resists imitation, whether the assembled space has enough regular structure in practice to merit geometric language beyond the typed graph, and whether a spatial rendering helps anyone.

The dependency among these groups runs in one direction. The shared history and interface system rests only on the first two groups and can be built, used, and evaluated while every claim in the third remains open. The decisive question for the semantic space as a thesis is whether the structure preserves useful continuations across interfaces: whether someone can begin through music, documents, or code, encounter the spatial representation later, and find there the relationships their activity had already established. The proof spine shows that such preservation is coherent and says exactly what it requires. Whether it is valuable is for the prototype to determine.

References

[1] J. P. Anderson. *Computer Security Technology Planning Study.* Technical Report ESD-TR-73-51, Air Force Electronic Systems Division, 1972.

[2] S. Chacon and B. Straub. *Pro Git*, 2nd edition. Apress, 2014.

[3] J. B. Dennis and E. C. Van Horn. Programming Semantics for Multiprogrammed Computations. *Communications of the ACM*, 9(3):143–155, 1966.

[4] Flyxion. *An Introduction to SpherePOP: Nested Circles, Typed Operators, and Reduction.* October 2026. `https://standardgalactic.github.io/spherepop/an_introduction_to_spherepop.pdf`.

[5] Flyxion. *Semantic Geometry and Corrective Navigation: Event Semantics, Spatial Projection, and Admissible Continuation.* Specification, October 2026. `https://standardgalactic.github.io/memnet/corrective-navigation/semantic-geometry-spec.pdf`.

[6] I. J. Good. *Probability and the Weighing of Evidence.* Charles Griffin, London, 1950.

[7] M. Goresky and R. MacPherson. *Stratified Morse Theory.* Springer, 1988.

[8] E. Hutchins. *Cognition in the Wild.* MIT Press, 1995.

[9] M. Kleppmann, A. Wiggins, P. van Hardenberg, and M. McGranaghan. Local-First Software: You Own Your Data, in Spite of the Cloud. In *Proceedings of the 2019 ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software (Onward!)*, pages 154–178, 2019.

[10] L. Lamport. Time, Clocks, and the Ordering of Events in a Distributed System. *Communications of the ACM*, 21(7):558–565, 1978.

[11] F. W. Lawvere. Metric Spaces, Generalized Logic, and Closed Categories. *Rendiconti del Seminario Matematico e Fisico di Milano*, 43:135–166, 1973.

[12] A. Mazurkiewicz. Trace Theory. In *Petri Nets: Applications and Relationships to Other Models of Concurrency*, Lecture Notes in Computer Science 255, pages 279–324. Springer, 1987.

[13] R. C. Merkle. A Digital Signature Based on a Conventional Encryption Function. In *Advances in Cryptology: CRYPTO '87*, pages 369–378. Springer, 1988.

[14] F. Monrose and A. D. Rubin. Keystroke Dynamics as a Biometric for Authentication. *Future Generation Computer Systems*, 16(4):351–359, 2000.

[15] A. Nerode. Linear Automaton Transformations. *Proceedings of the American Mathematical Society*, 9(4):541–544, 1958.

[16] L. Qian, Y. Wang, X. Peng, Y. Cao, Y. Jiang, Z. Xie, A. Xu, W. Cao, J. Chen, Z. Li, J. Huang, A. Lopez-Lira, Y. Tang, S. Ananiadou, and J.-Y. Nie. LLMs Don't Know What They Know: Benchmarking Temporal Self-Knowledge for Financial Backtesting. arXiv:2611.06808, 2026.

[17] J. H. Saltzer and M. D. Schroeder. The Protection of Information in Computer Systems. *Proceedings of the IEEE*, 63(9):1278–1308, 1975.

[18] M. Shapiro, N. Preguiça, C. Baquero, and M. Zawirski. Conflict-Free Replicated Data Types. In *Stabilization, Safety, and Security of Distributed Systems*, pages 386–400. Springer, 2011.

[19] L. A. Suchman. *Plans and Situated Actions: The Problem of Human-Machine Communication*. Cambridge University Press, 1987.