Inscription and Authority

Flyxion
Independent Researcher

30 September 2026

Abstract

Every shared knowledge surface faces a question that is usually asked in one breath: who may contribute. The question hides two. The first is who may put a record into the store. The second is who may change what readers are shown as standing. This essay argues that the first should be answered "anyone," that the second is where authority properly lives, and that the two can be separated exactly. The monograph *Adversaria: A Specification for a Public Claim Graph* does this with two planes. An authorization plane holds capabilities, revocations, invalidations and restrictions. An argument plane holds evidence, claims and arguments. The authorization plane is computed without reading any label from the argument plane, and a capability never enters the defeat graph, so authority supplies no argumentative force of its own. It can still affect what stands by controlling admission, which is why that control is made visible and contestable.. Capabilities are prospective unless explicitly invalidated. Restrictions are finite, are justified by ordinary arguments, and are evaluated on a meta surface that contains no restrictions, so a restriction cannot gate the objections that would defeat it. Every refusal carries a reason that any replica computes identically. The essay states these results, works an invented timeline and a meta-surface example whose labels were checked by computation, and lists the ten invariants against gate capture with the status of each. It does not claim that good governance follows from the design. The design makes authority visible and contestable. It does not make authority disappear, and it does not make it wise.

1 Two questions in one

A contributor arrives with a record: a piece of evidence, a claim, an objection to someone else's argument. Whether the system should accept it is usually treated as a single decision, and the decision is usually made by a person or a rule that holds authority over the whole surface. Acceptance means storage, visibility, weight and the ability to change what other readers see, all at once.

Bundling these produces a familiar dilemma. Make acceptance easy, and the surface is open to spam and volume posing as consensus. Make it hard, and the surface becomes the property of whoever controls the gate, which then becomes the most valuable thing to capture.

The proposal examined here is to pull the bundle apart. Storing a record is one act. Letting the record alter a shared display is another. The first can be open to everyone and the second can be governed, provided that governing it never reaches inside the evaluation of arguments. The organizing sentence of the specification is that the freedom to inscribe is not the same as the authority to modify a shared projection, and the rest of this essay is an exercise in taking that sentence literally.

2 Inscription is open

A ledger in the monograph is a set of records, each with a type, a payload, references to earlier records, an author, a claimed time and a declared origin. A set of records is *valid* if it is closed, so that every reference resolves to a member; if identifiers agree with contents; and if the references can be put in a linear order in which every record follows what it references (Chapter 9). Validity is a property of the set. It mentions no policy.

That is the whole basis of the following result (Chapter 16): for every finite set of records that is closed, has consistent identifiers and an acyclic reference order, the set is a valid ledger under every policy, whatever gates the policy contains. The proof is one line. Validity does not mention policy, and gates enter only in the disposition function, which decides what a record is allowed to do on a given surface. A gate cannot make a ledger invalid, because gates are not part of what validity asks.

The claim is narrower than it may sound and deserves a careful reading.

- It is not a claim that a record is shown, read or counted. The pipeline of Chapter 14 has five stages: inscribe, validate, admit, project and activate. Inscription appends a record and returns a receipt. Validation checks structure: references resolve, identifiers agree with contents, the payload conforms to its type's schema, and no executable payload sits in a field that must be inert. Admission is where the policy first acts. Only admitted records take part in evaluation on a surface.
- It is not a claim that every byte string is accepted. A record that is malformed fails validation, and a record that lacks what its type requires, such as a claim without a scope, receives a non-admitted disposition with a stated reason. The claim is that well-formed records are not turned away at the door by the gate machinery.
- It is a claim about dependence. Inscription depends on the record and on nothing else. It does not depend on policy, on gates, or on labels. Admission depends on the record, on the dispositions of what it references, on the policy, on the authorization projection and on ledger time, and explicitly not on routing, the reader, view counts or the claimed time.

Stored, admitted and endorsed are three different things (Chapter 14). A record can be stored and not admitted. An admitted argument can be labelled OUT or UND. An argument can stand and have no recorded endorsement from anyone. A contributor whose record is stored but not admitted has lost nothing the design promised: the submission contract of Chapter 14 owes every submitter a receipt, retention of the record or a declared term for it, exportability of the ledger, and an explicit disposition with a reason. It does not owe

immediate indexing, review, inference or prominent placement. Attention, computation and review time are the scarce things, and the design charges for promotion into shared attention and never for speech (Chapter 17). The ledger has no quota on claims per actor.

An ungated actor can still inscribe a claim, cite evidence, prepare an objection, keep a personal or group projection, fork a dossier, and request review. A gated record is kept and not discarded, keeps its identifier, and can become admissible later without resubmission. What such an actor cannot do is change what a particular shared surface shows without meeting that surface's conditions.

3 What authority is over

If inscription is open, what is left for authority to govern? The specification's answer is that authority is over a *projection* of the ledger onto a surface. A surface has a policy. Given a valid ledger and a policy, each record receives one of seven dispositions: admitted, duplicate, malformed, unscoped, quarantined, gated or refused (Chapter 9). The set of admitted records is the active set, and the labeling of arguments, the regions of standing and the status vector are all computed on the active set and on nothing else.

Two things follow. First, dispositions are relative to a surface. A record gated on the principal dossier may be admitted in a personal or group projection whose policy has no such gate. It is stored in both places and has one identifier. A dispute about authority is then always a dispute about a particular surface, and communities that disagree about a gate can each keep their own projection over the same records without copying anything.

Second, what a gate can change is bounded. The monograph lists the things a gate governs as contribution to a shared surface: the principal dossier, default status displays, high-ranking search results, notifications, and established mappings between ontologies. A gate decides whether a contribution takes part in the shared evaluation. It does not decide how a participating contribution is evaluated. The statement in Chapter 9 is exact: the labeling depends on the ledger only through the active set and the policy, and does not depend on routing weights, view counts, notification state, search rank, contributor reputation, controversy flags or agreement tallies. Flagging a claim as controversial may legitimately gate admission, and it has no bearing on status. Flagging a claim does not make it false or unsupported, and unflagging it does not support it.

4 Two planes

The separation is enforced by a layering of the records themselves (Chapter 16). The *authorization plane* consists of the anchor, capability, revocation, invalidation, restriction and clearance records. The *argument plane* consists of all other records. The disposition of an authorization record depends only on its own fields, on authorization records it references, on the root authorities the policy names, and on ledger time. The disposition of an argument-plane record reads the authorization projection for its gate outcome and reads

the dispositions of the argument-plane records it references. It reads nothing else.

The layering can be written as a short diagram.

$$\begin{array}{ll} \text{ledger} & \longrightarrow \text{authorization projection } C \\ \text{ledger and } C & \longrightarrow \text{active set of arguments} \\ \text{active set} & \longrightarrow \text{defeat labeling} \end{array}$$

Three results follow (Chapter 16, the proposition on planes). The authorization projection is a function of the authorization records and the policy, and does not depend on any argument-plane record or on any label. The active set is a function of the ledger and the policy, and is the single set of arguments the defeat semantics evaluates. And no authorization record is a node of the defeat graph. Put informally, authority lives in a layer that cannot see conclusions, and conclusions are computed in a layer that cannot see authority except through the single question of whether a record was admitted.

The consequence most worth stating is the following (Chapter 16, capability confers no standing). Passing a gate makes a record eligible for admission. It does not change the label of any argument, does not change any argument's defeat relations to others, and does not defeat anything. A capable actor's argument is IN, OUT or UND by the same rules as anyone else's.

The wording calls for care, because an admitted argument does enter the active set and so does take part in the labeling. What the result excludes is an effect beyond that. Before admission the argument is not in the set. After admission the argument is evaluated as any other argument in its position would be, and the fact that its author holds a capability is nowhere an input. The capability changes *whether* the argument participates and never *how* it is judged.

Whether it participates can matter a great deal. An authority that kept a decisive objection out of the active set would leave the targeted conclusion standing on the remaining arguments. The separation of planes keeps a capability from functioning as evidence or as a defeater, but it does not keep selective admission from influencing the result. The reasons, appeals and meta surface of Sections 7 and 8 are therefore central safeguards, not supplementary governance features.

One loose end is worth closing. In practice capability is granted on the basis of what has survived, which means, indirectly, on labels. The worry is that this closes a loop: labels would depend on the active set, the active set on capabilities, and capabilities on labels. The specification breaks the loop by requiring that a capability be a record. Labels may influence who decides to inscribe a capability record, but once inscribed the record is data, and the disposition of an authorization record never reads the argument plane. Credit and reputation stay outside both planes and enter only when someone inscribes a record on their basis. The dependence runs one way: routing may use labels to order what a reader sees, and neither storage, admission nor evaluation depends on routing (Chapter 13).

5 Capability, revocation and invalidation

A *capability record* states that an actor is authorized to perform acts of a given family over a given scope during an effective interval of ledger time, on stated grounds. An act family is a class of records that call for a common competence: adding archival evidence, supplying a warrant for a causal inference, raising an objection, proposing a mapping, restricting a claim, withdrawing an argument. Competence for one family does not transfer to another. A *revocation record* refers to a capability record. An *invalidation record* also refers to one, is admitted only if its author holds a capability of the audit family, and cites the finding on which it rests.

An act *holds* a capability, in the monograph's definition, when it references the capability record, the record is in the authorization projection, the act's author is the grantee, the act's scope lies inside the capability's scope, the act's ledger time lies in the half-open effective interval, no admitted revocation of the capability has ledger time at or before the act's, and no admitted invalidation of it exists.

5.1 Ledger time is not something a contributor chooses

Every clause above is stated in terms of ledger time, and the design takes care that ledger time cannot be chosen by the actor it governs. Ledger time is the position of a record in the accepted order of a surface, derived from anchor records. It is not a field of the record. The claimed time that a contributor asserts about an event is provenance: it is evidence like any other and is never consulted by admission. The monograph states the consequence as a design rule. No contributor-supplied field determines ledger time, and admission, capability validity, revocation and restriction applicability are evaluated at ledger time and never at claimed time (Chapter 16, design rule on ledger time). Backdating an act into an expired interval, or postdating it past the end of a restriction, is not available to a contributor, who can only delay or hasten submission.

This shifts the trust question to the anchoring authority that orders records into ledger time. A surface that distrusts its authority audits the anchors, and a forked anchor chain makes acts whose first anchoring is ambiguous gated with the mechanism equivocation. The design does not remove the anchoring authority. It names it in the policy and makes its misbehavior detectable.

5.2 Revocation is prospective

The monograph proves (Chapter 16) that if a revocation is admitted, every act that held the capability with ledger time before the revocation keeps its disposition, and acts at or after it do not hold the capability. Only an invalidation changes the disposition of earlier acts, and it does so explicitly, by making them refused with the mechanism invalidated capability.

The two instruments answer different needs. Revocation says the actor should not act from now on. Invalidation says the capability should not have been issued, and records the finding

that supports the claim. Because the second rewrites nothing silently, the consequence shows as a change in disposition with a named mechanism. The ledger still contains the earlier acts and everything others did with them. A historical projection indexed by an earlier prefix of the ledger is a fixed object and never changes (Chapter 16, historical projections are immutable). A later projection is a different object, carrying its own index, that differs from the earlier one only through records of later ledger time.

5.3 An invented timeline

The following timeline is invented. One capability, κ_1, lets an actor Ana perform objection acts over the units CW and CD during ledger times $[2, 8)$. A revocation of κ_1 is admitted at ledger time 5. An invalidation of κ_1, by an auditor holding the audit capability, is admitted at ledger time 9. Five acts are tried. The dispositions were computed from the holding test above, once on the prefix of ledger time 8 and once on the prefix of ledger time 10.

act	author, family, scope	time	what differs	prefix 8	prefix 10
r_0	Ana, objection, CW	1	before the interval opens	gated	gated
r_1	Ana, objection, CW	3	inside the interval, before the revocation	admitted	refused
r_2	Ana, objection, CD	6	after the revocation	gated	gated
r_3	Ben, objection, CW	4	author is not the grantee	gated	gated
r_4	Ana, evidence, CW	4	family is not objection	gated	gated

Only r_1 changes between the two prefixes, and it changes because of the invalidation, not the revocation. The revocation at time 5 left r_1 admitted, as the proposition says, and gated r_2. Each gated act carries a reason naming the failing requirement, such as *outside effective interval* or *revoked*, and the refused act carries the mechanism *invalidated capability*. The table is a worked instance and not a proof. The prospective property was also checked by brute force on two thousand random timelines of one capability with random acts, revocation times and effective intervals. In every case, adding a revocation left earlier acts unchanged and kept every act at or after it from holding the capability, and adding an invalidation refused every earlier act that had held the capability. Both checks test the statements against the holding test as defined, and neither is part of any proof.

5.4 Clearance

A gated record can also become admissible by a *clearance record*, which refers to the gated record, belongs to the authorization plane, and is admitted if its author holds a capability for clearing acts at the ledger time of the clearance. The cleared argument is then an ordinary member of the active set, and has at each unit the label it would have had if it had passed the gate. Clearance is an authorization record and so is no node of the defeat graph. Revoking the clearing capability later leaves the cleared records as they are, and invalidating it does not (Chapter 16, clearance keeps one active set).

6 Gates that test competence and not belief

A gate is legitimate, in the monograph's terms, only if what it tests is a competence that the act requires. A gate that asks whether the actor accepts the community's conclusion tests belief, and the specification excludes it by a structural requirement. A gate is *position-neutral* if, for every actor, act family and scope, it is passed by an act on a claim exactly when it is passed by the corresponding act on the claim with the opposite kernel and the same scope.

For contested subjects the specification recommends a reconstruction gate. To modify the default projection of a claim over some scope, an actor inscribes a reconstruction record that references, for each unit at which claims of the kernel exist, an admitted argument for some claim on the frontier of the status preorder (or for a claim the dossier policy names in its place), and the same for the opposite kernel; a claim stating the principal empirical uncertainty; a distinction or scope claim stating the principal scope distinction; and, for each of the two positions, an observation that would count against it. The first two items are checked by computation. The other three are graded by graders who are not the author.

The monograph proves that the reconstruction gate is position-neutral (Chapter 16). An actor who passes may act on the claim and on its opposite, and one who fails may act on neither. The reason is simple: the conditions depend on the pair consisting of a kernel and its negation, and they are unchanged when the two are exchanged.

The result guarantees one structural property and no more. It does not guarantee that graders are unbiased, and it does not guarantee that the tests are well chosen. Those are matters for the invariants of Section 9. What the result excludes is a gate whose conditions depend on which side the actor takes. An actor may pass while opposing the prevailing view, and may fail while agreeing with it. Design rules support the gate: tests are generated from the current dossier and not from a list held by administrators; equivalent competencies have alternative demonstrations, such as a reviewed record of contributions or the reproduction of a result; and, for high-conflict topics, grading is blind to identity and stated position, with periodic reversed-polarity items so that bias in either direction shows in the audit.

7 Restrictions lock surfaces and never topics

Sometimes a surface needs protection: a burst of coordinated activity, a risk of personal harm, a legal constraint. The specification provides restrictions, and builds in four limits that keep a restriction from becoming a permanent topic ban.

A *restriction record* names a rule from the policy, references to records showing that the rule's observable trigger holds, a surface, a scope, a set of act families that are gated, an effective interval with a review point, and a reference to a *justification argument* concluding that the restriction is warranted. Triggers the policy may name include unusually high attack density, repeated reversion, a rapid influx of new actors, credible risk of personal harm, legal or medical actionability, conflict among qualified projections, and evidence of coordinated manipulation. The policy names the rules, the record must cite records showing

the trigger, and a restriction on the ground that a moderator dislikes a topic is not of this form.

The four limits are these.

1. *Finite.* Every restriction has a finite duration, has no effect on acts outside its interval, and needs no record to end it. A renewal is itself a restriction record with its own justification (Chapter 16, restrictions are finite and expire).
2. *Scoped.* It gates named act families, over a named scope, on one surface. A restriction does not stop inscription, which is open in every case, and it does not touch other surfaces.
3. *Justified by an ordinary argument.* The justification is an argument like any other and can be objected to by ordinary objections.
4. *Stratified.* A restriction is evaluated on a *meta surface* whose policy contains no restrictions, and the label of the justification and of the objections to it are computed there. No restriction can gate an objection to its own justification, and no act of withdrawal, appeal or request for review is ever gated by the restriction it concerns. A renewal requires that the justification argument not be OUT on the meta surface at the renewal's own ledger time, computed on the prefix preceding it. Emergency restrictions may take effect at once, with a short interval.

The fourth limit is the one that prevents circularity. Were the label of the justification computed on the restricted surface, a restriction could gate the objections that would defeat it, and the restriction's own standing would depend on its own effect.

7.1 An invented meta-surface example

The following example is schematic and invented. A restriction on a busy surface is justified by an argument j that a rapid influx of new actors has occurred, resting on one evidence item e, a count of new accounts. An objection o is filed that undermines e by concluding that the report of e is not accurate at the unit: the accounts came from a single batch import, and a batch is one independence family (Chapter 17).

evaluation	label of the justification j
meta surface, objection o admitted	j is OUT, o is IN; renewal is refused
restricted surface, o gated by the restriction it concerns	j is IN; the restriction would renew itself
meta surface, o a bare rebuttal of j	j and o both UND; this condition does not refuse renewal

The first two rows show why stratification matters. On the meta surface the objection is admitted and the justification falls. On a restricted surface where the objection is gated, the justification stands, and the restriction sustains itself by removing the very act that would end it. The third row is a caution about the rule. An objection that only rebuts the justification leaves both arguments unresolved, and the renewal rule refuses renewal only when the justification is OUT, so on this condition alone a symmetric stalemate on the

meta surface does not end a restriction. A community that wants a stricter test can specify one in its policy. The labels of the three rows were computed from the defeat semantics of Chapter 7 and agree with the table.

8 Dispositions come with reasons

Every non-admission has a reason, and the reason is computed. For a record whose disposition is not admitted, the reason is a tuple: the governing rule, the failing field or requirement, references to the records that evidence the failure, the deciding mechanism, and the repair and appeal path. The mechanism is one of a fixed list: schema check, disposition rule, gate, restriction, clearance, equivocation, invalidated capability, or an explicit decision record.

The monograph proves (Chapter 16) that the reason is a function of the set of records and of the policy. Two replicas holding the same records and the same policy compute the same reason for every record. The proof follows from the determinism of dispositions (Chapter 9), since the reason records which clause of the definition produced the value, with the references that clause used. Two consequences matter for governance. There is no private reason for a refusal, because a reason that cannot be recomputed is not a reason in this sense. And gate statistics, meaning who is refused under which rule and how often, are computed by replay from the ledger, so an audit for disparate exclusion needs no privileged data.

An *appeal* is an argument, filed on the meta surface, that a stated rule was misapplied or is itself unwarranted. It uses the ordinary machinery of objection. The specification adopts a design rule that a successful appeal repairs the rule: an appeal whose argument stands yields a new version of the gate or rule, with a supersession reference, and the clearance of similarly situated records follows from the revised rule and not from a one-off exception. The earlier version stays in the lineage. This is a design rule and not a theorem. Its purpose is to keep appeals from being settled by exception, which would leave the rule itself unchanged and the next contributor in the same position.

9 The capture table

The capability system is the most valuable point of attack in the design, and the specification lists ten invariants against capture with the way each is enforced. The table below follows the monograph. The last column sorts the ten by the kind of guarantee, using the chapter's own distinctions: a proposition of the model, a validation check on records, or a design requirement that depends on how a community specifies its gates.

	invariant	enforced by	kind
1	No actor authors, grades and finally approves the same test	the three role fields of a grant record are distinct actors	validation
2	No viewpoint is required unless the capability is about representing it	position neutrality of reconstruction gates; design elsewhere	proposition, in part
3	Every failure maps to a declared competency	a gated reason names a declared competency identifier	validation
4	Equivalent competencies have alternative demonstrations	a gate specification lists at least two routes	design
5	Gate creation and renewal require public reasons	a restriction without a justification reference is malformed	validation
6	Gate statistics are auditable for disparate exclusion	reasons are determined by the ledger and policy	proposition
7	A successful appeal repairs precedent	the design rule on appeals	design
8	Administrators are subject to the gate when performing gated acts	gate predicates refer to actors only through capability records	design, by construction
9	Capability cannot substitute for evidence	capability confers no standing	proposition
10	Capability cannot become hereditary through endorsement alone	grounds of a grant reference the grantee's own acts or a verification record, never stances of others	validation

The monograph's own tally is three propositions, four validation checks and three design requirements, and the table reproduces it, with row 2 counted as a proposition where the reconstruction gate applies and a design requirement elsewhere, and row 8 counted as a consequence of how gate predicates are specified. The purpose of sorting them is the one the chapter gives: no design requirement should be read as a guarantee. The three design rows, and row 2 outside the reconstruction gate, depend on choices a community makes about its gates. The four validation rows depend on the validation step being implemented and run as specified. Those are the rows where an attack would be aimed. Rows 6 and 9 hold by the definitions of dispositions and labels, given that replicas compute as specified.

10 What the design buys

The results amount to an argument of definite shape. Inscription is open, so nobody is silenced by exclusion from the record. Authority is exercised over surfaces, by records, at ledger time. It never enters the defeat graph, so it supplies no argumentative force of its own, although its control over admission can still change which conclusions stand. Every limit on authority is itself a record with a stated reason, is finite when it is a restriction, and can be objected to on a surface where it has no power to suppress the objection. Every refusal can be recomputed.

Authority nonetheless remains. Some party names the root authorities in the policy, op-

erates the anchoring service, and grades the reconstruction tasks. The design makes each a named role whose actions are recorded and whose statistics can be audited, and it does not make any of them unnecessary. Which policy a surface runs is decided by whoever operates the surface, and the monograph does not turn that decision into a formal act. A community whose operators adopt a bad policy has a recorded, contestable, replayable bad policy, which is better than an invisible one and is still bad.

11 What this essay does not show

It does not show that governance quality is a consequence of the design. That is an institutional matter, and the essay has argued only that the design makes authority explicit, bounded and contestable.

It does not show that authority can be removed. Nor does it show that selective admission is harmless: it shows only that the policy, its reasons and the route of appeal are recorded and can be contested. Root authorities, anchoring and grading are named roles, and each can be held badly.

It does not show that the ten invariants are the right ones, that they are complete, or that any deployed system would satisfy them. Three of the ten are propositions of the model, and the rest depend on validation implementations or on community choices. It does not show that position-neutral gates are sufficient for fairness, only that they exclude gates that test belief.

It does not show that the meta-surface rule is optimal. The example shows that a symmetric stalemate on the meta surface does not by itself end a restriction under the stated renewal rule, and that a community wanting otherwise must say so in its policy. It does not address who may issue revocations, since the monograph leaves the admission of revocation records to policy. It does not address federation, in which surfaces with different policies exchange records, nor the incentives of contributors, nor legal duties of operators. The timeline and the meta-surface example are invented and are labelled as such. Their dispositions and labels were computed from the stated definitions and are checks of the examples, not evidence about any real community. Related work is deliberately not surveyed in this draft.

Notes

The monograph is *Adversaria: A Specification for a Public Claim Graph*. Chapters used: Chapter 16 (ledger time, the two planes, capabilities, revocation, invalidation, gates, clearance, position-neutral and reconstruction gates, restrictions, dispositions with reasons, appeals, the capture table); Chapter 14 (the five stages of the pipeline, stored versus admitted versus endorsed, totality of dispositions, the submission contract); Chapter 17 (attention controls, batches as one independence family); Chapter 9 (validity of ledgers, dispositions and the active set, evaluation depends on arguments only); Chapter 13 (one-way dependence of routing on evaluation); Chapter 7 (labeling used in the meta-surface example).