RITHMO · TECHNICAL PAPER

# The Resolved Half

On formation, supersession, absence, and provenance in organizational records

Version 1.0 · July 2026 · DOI 10.5281/zenodo.21710262

## Abstract

Every company runs on facts about itself, and they come in two kinds. Some are authored: somebody writes them into a system, and they stay there until somebody writes something else. Others are resolved: they became true because a group of people decided something, in a meeting, in a thread, on a call, and no system was ever the place they were written down.

The difference is not one of degree. An authored fact arrives together with the artifact that proves it, so confirming it means going to the place it is kept. A resolved fact arrives with nothing. It is expected to generate artifacts afterward, in other systems, but the map from a decision to the artifacts it should generate is neither one to one nor stable, so confirming one means inferring it across systems against an expectation that keeps moving. That asymmetry is why the authored half of business context has mature tooling and the resolved half has only close neighbors built for other questions.

Holding the resolved half requires four capabilities that no existing approach combines, whatever else each of them does well. Forming a decision from accumulated evidence rather than from decision-sounding language. Distinguishing the several different ways a decision stops being the current answer, commonly collapsed into one transition called supersession and in fact at least four. Asserting that no decision exists, as a positive answer rather than a plausible guess. And showing why a current answer is current.

This paper sets out those four requirements, proposes a taxonomy for the second and the third, examines why the available approaches produce something else, and describes the approach Rithmo takes. It is a thesis about a problem. Where it describes a design rather than a running behavior, it says so.

## Definitions

This paper uses two terms that are not yet standard. Both describe kinds of fact a company holds about itself, and the distinction between them organizes everything that follows.

**Authored fact.** A fact that is true because someone wrote it down. A customer's billing address. An employee's job title. The list price of a product. Someone typed it into a field on purpose, that field is where it lives, and it changes when a person edits it. Writing the fact down and making it true are the same act.

**Resolved fact.** A fact that is true because people decided something. Whether the company is switching away from a supplier. What was promised to a customer on a call. Whether the new pricing was approved. It became true in a conversation, and there is no field anywhere whose job is to hold it. To know it, something has to work it out from scattered evidence, which is what resolved means here: not written down, worked out.

A single test separates them. Ask what would have to happen for the fact to change. If the answer is that someone edits a record, it is authored. If the answer is that a group of people decides something, it is resolved.

**The asymmetry that follows.** An authored fact is its own artifact, which is why it can be looked up. A resolved fact has none at the moment it becomes true, and the artifacts it should produce later are neither guaranteed nor fixed, which is why it has to be worked out instead. Nearly everything that makes the resolved half difficult follows from that single difference, and section 2.1 takes it up in full.

The claim this paper rests on is that a large share of what a company knows about how it operates is resolved rather than authored, and that almost all tooling built for company knowledge assumes authored. The second half is checked against the available approaches in section 7. The first half nobody has measured, here or anywhere, and it is stated as a claim rather than a finding.

Three further terms are used precisely and are defined in the sections that introduce them: **formation**, meaning the point at which scattered evidence becomes a decision in the record (section 3); **supersession**, meaning any transition after which an existing decision is no longer the current answer (section 4); and **absence**, meaning a positive finding that no decision exists, as distinct from a failure to find one (section 5).

---

## 1. The record that does not exist

Ask five people in a company what was decided about something that matters, and you will often get four answers, plus one person who is surprised the question is still open. Nobody is being careless and nobody is lying. Each of them is accurately

reporting the last thing they heard, and they heard it in different rooms at different times.

This is what happens to a decision in an organization of any size. It gets made out loud in a meeting. It gets refined in a thread that half the people in that meeting are not in. It gets questioned in another team's call, where somebody reaches a different conclusion without knowing the first conversation happened. Somewhere in that sequence it becomes what the company is actually doing, and at no point does it land in any system as the current answer.

The results are ordinary and everywhere:

- a price approved in a meeting that the system generating quotes never received
- a renewal term promised to a customer on a call that never reached the CRM
- a launch date moved in one team's planning session and not in the other team's
- a technical direction green-lit, then quietly reopened six weeks later
- a hiring freeze that one group is operating under and another is not
- a vendor the company decided to leave and continued paying for five months

Not one of those is a failure of anyone's diligence. Every tool involved worked correctly. Every person involved behaved reasonably. The quoting system used the price it had. The second team planned against the date it knew. The invoices were paid because they arrived and nothing said to stop.

The common structure is that each system held a fragment, and nothing held the answer.

Organizations have always worked around this the human way, by asking around. Someone knows who was in the room. Someone remembers what was said. Someone whose actual job is holding the state of things in their head gets asked, and usually knows. That workaround is how the work gets done today, and it is a genuinely good method at small scale, which is why it has never been displaced. It also has a price that is paid continuously and almost never measured, and it degrades in a specific way as a company grows: the number of rooms grows faster than any one person's ability to be in them.

Two things make this worth writing about now rather than treating as a permanent condition of organizational life.

The first is that the cost is computable, and companies have never been in a position to compute it. Time spent re-deciding questions that never closed, and money spent against decisions that were made and never carried out, are both derivable from data every company already holds. What has been missing is the one input that makes the calculation possible, which is a record of what was actually decided and what became of it.

The second is that the record now has two readers. It has always had one, the people who need to know what the company decided. The other is the growing set of agents and automated systems executing on their behalf, and they cannot ask around. A person given a stale instruction will often sense that something is off and go find out. An agent will not. It executes what it was given, correctly, at speed, and the gap that used to cost a few duplicated meetings now propagates before anyone notices.

Both readers want the same thing from the record, which is why the second is treated throughout as a consequence of building it rather than as a reason to. The argument that follows holds if no agent is ever deployed anywhere. It only raises the stakes if one is.

What follows is an argument in four moves. First, that business context is two different kinds of fact, and that the difference is not a matter of degree: facts that are true because somebody wrote them down behave differently in every respect from facts that are true because a group of people decided them. Second, that holding the second kind requires four specific capabilities, which are the subject of sections 3 through 6: forming a decision from evidence rather than from decision-sounding language, distinguishing the several different ways a decision stops being current, asserting that something was never decided, and showing why a current answer is current. Third, that the approaches which appear to address this are competent tools answering adjacent questions, and that their limits are consequences of their design rather than gaps awaiting a release. Fourth, the approach Rithmo takes, and what having such a record makes possible for both readers.

---

## 2. Two kinds of business context, and the half nothing was built for

Business context, the layer a company's tools and agents read to understand the company itself, holds both kinds of fact. It is treated as one thing. It is two, and the two behave differently in every respect that matters.

Take one question of each kind.

*What does the enterprise plan cost.* Authored. Someone set the price, it sits in a field in a pricing system, and it changes when a person edits that field.

*Is the company still switching away from its current supplier.* Resolved. It became true in a meeting. It may have been revised in a thread, questioned in another team's call, and acted on by someone who was in the room for none of it. No system was ever the place it was written down, because no field exists whose

purpose is to hold it.

Side by side, the two kinds diverge on every dimension a system serving them would need to handle.

| Dimension | Authored facts | Resolved facts |
|---|---|---|
| How the fact changes | someone edits a record | people decide something |
| Where it originates | a system of record | human activity, often with no system present |
| How it is maintained | curation by the owning team | resolution across scattered evidence |
| When the answer settles | when somebody writes it down | when enough evidence has arrived |
| What determinism rests on | a governed definition, owned and versioned | a stored answer with its evidence behind it |
| Characteristic failure | a stale value returned confidently | a guess where there should be a refusal |
| What absence means | nothing found | nothing was decided |

The first three rows are properties of the fact itself. The last four are what a system serving that kind of fact has to get right. The divergence runs through both.

The authored half has a well-populated field of tooling. Semantic layers, metric stores, and enterprise search over authored documents all address it, and they address it competently. The resolved half has close neighbors, and none of them was built to hold it.

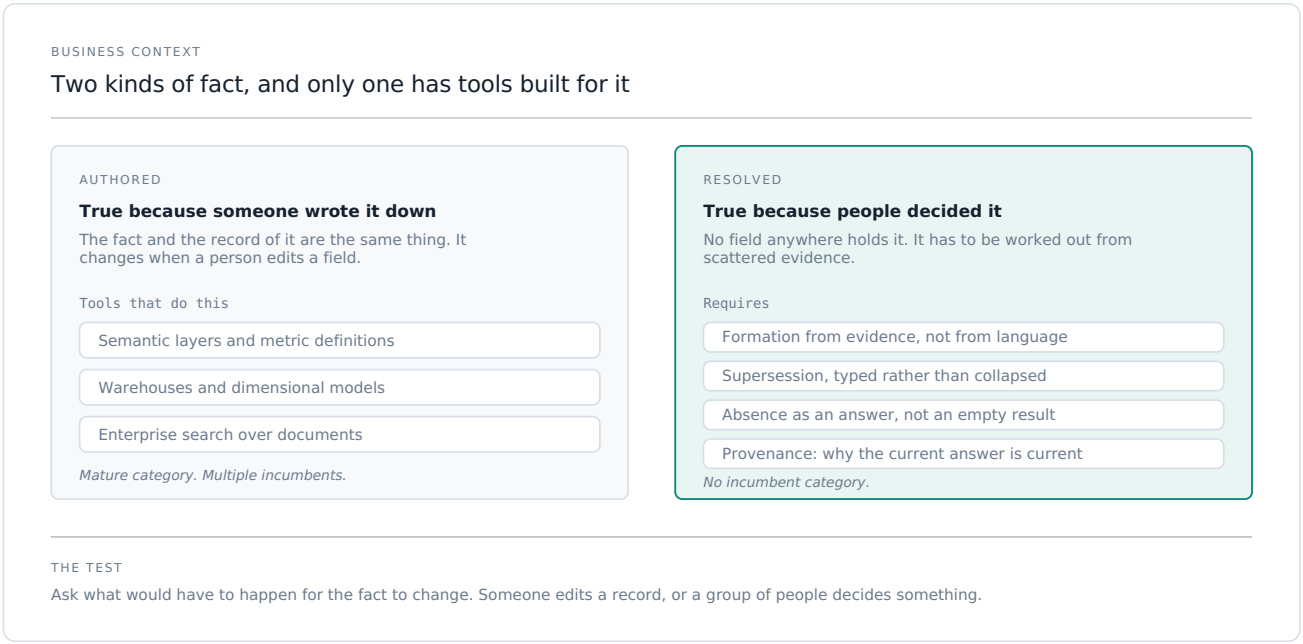



Figure 1. Business context is treated as one layer. It is two, and only one of them has been built for.

## 2.1 The artifact asymmetry

The most consequential difference between the two is not in the table above, because it is not a property of the fact. It is a property of what the fact leaves behind.

An authored fact arrives with its artifact attached, so there is always something to point at, and usually several redundant somethings. An employee's job title appears in the HR system, in an offer letter, in a directory, and in an email signature. Those copies can disagree, and when they do the disagreement is resolvable, because they all descend from one authoritative field.

A resolved fact arrives with nothing attached. The decision to switch suppliers produces no document at the moment it is made. What it produces instead is an expectation: that certain artifacts will appear, in certain systems, at some point afterward. A contract gets signed. Orders to the old supplier stop. A record gets updated. Someone opens a ticket. The decision is not those artifacts, but the artifacts are the only evidence available that it was carried out.

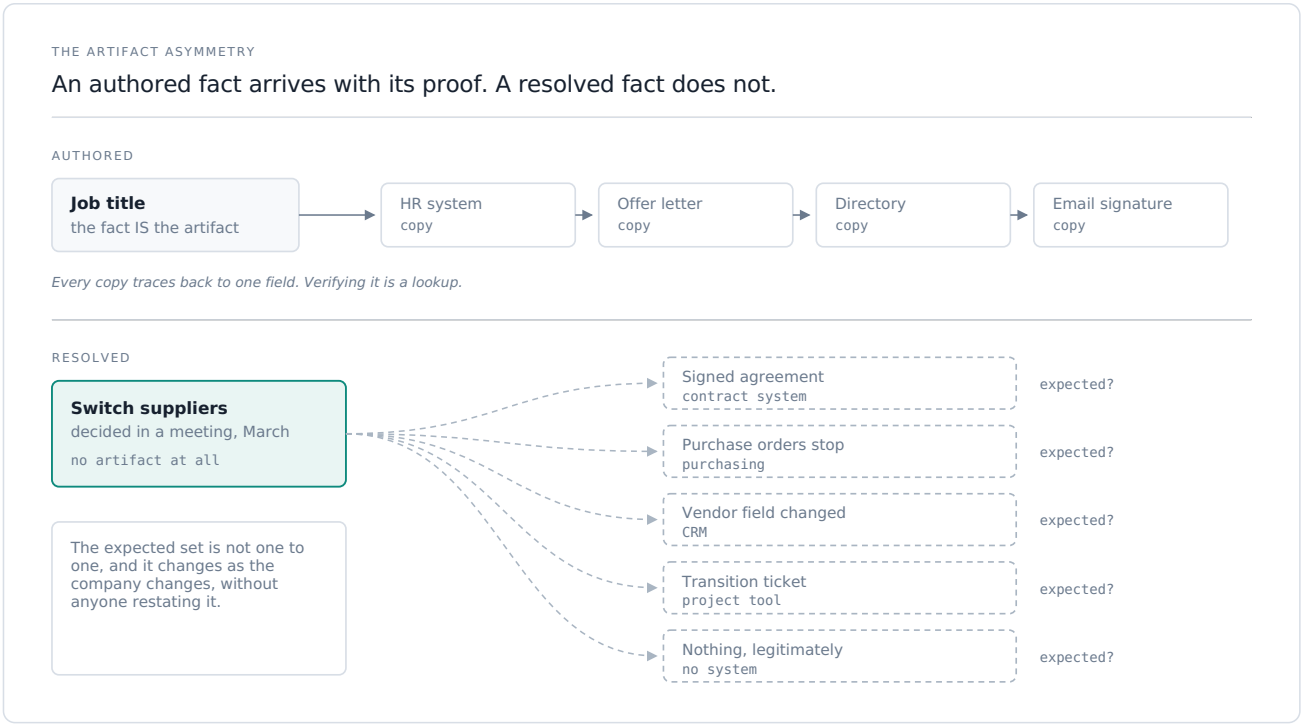



Figure 2. An authored fact is its own proof, and its copies all trace back to one field. A resolved fact produces no artifact when it becomes true, and the artifacts it should produce later sit in systems that hold no record of the decision.

Four properties of that relationship make it hard to use.

**It is not one to one.** A single decision may be expected to produce eight artifacts across five systems, or one, or none at all legitimately. A single artifact may satisfy several decisions at once. There is no fixed ratio and no reliable arity, so the presence or absence of any individual artifact is weak evidence on its own.

**The expected set is not stable.** What a decision of a given kind ought to produce depends on how the company currently works: which systems are in use, which team owns the follow-through, what the process was last quarter versus this one. Change any of those and the expected artifact set changes with it, without anyone restating what a decision of that kind is supposed to generate. The map from decisions to expected artifacts is not just unwritten, it is non-stationary.

**The artifacts are downstream and across stream.** They do not appear in the system where the decision was discussed. They appear in whichever systems the work touches, which means verification requires reading systems that hold no record of the decision and have no reason to reference it.

A deal review approves a lower price for one large customer. Everyone in that review is done, and as far as they know the matter is handled.

Where the pricing system carries its own approval workflow, the decision and the artifact are linked and this failure does not occur, which is the authored half

working correctly inside its own boundary. The failure lives at the edges of that boundary, and the edges are where large companies spend most of their time. The approval was granted in a review that sits outside the workflow. Or the company runs two workflows, because it bought a company and inherited a second sales organization with a different exception process and a different definition of what approved means, so a decision made in the vocabulary of one has to land in the other. Or the concession was made on a call by somebody with the standing to make it and no path into the workflow at all. In each case a person has to carry the decision across, and the system that builds quotes has no idea the review took place. If nobody carries it, every quote sent afterward is built from the old price. The quotes are not broken. They are correct, and they are wrong.

**Sometimes the expected artifact is the absence of an artifact.** A decision to start something expects a new artifact to appear: a signed contract, a ticket, a changed record. A decision to stop something expects an existing artifact to stop appearing. Those two are not mirror images, and the second is far harder to check.

A company decides to pause hiring. The proof that it happened is a job posting that never goes up. Nothing anywhere announces a job that was not posted. No system sends a message when a normal thing simply keeps happening, so if a manager posts a role next week, nothing about it looks unusual. Posting a job is what managers do.

Worse, the new postings do not just fail to show the freeze. They say the opposite. Anyone looking at the hiring system sees roles going up on a normal schedule and would reasonably conclude the company is hiring. The evidence tells a complete, sensible story that happens to be the reverse of what was decided. Nothing in that system is broken, because it was never told about the decision in the first place. It keeps doing what it has always done, and what it has always done looks perfectly convincing.

Time makes it worse rather than better. The longer the postings keep going up, the more normal they look, and the less likely anyone is to ask.

The same shape shows up whenever a company decides to stop something that runs on its own. A vendor it decided to leave keeps invoicing, and an invoice that arrives looks like every invoice before it.


Figure 3. When the expected proof is something appearing, its absence can be looked for. When the expected proof is something stopping, the failure produces no event and is indistinguishable from normal operation.

The consequence is the thing companies actually cannot do today. Not finding what was decided, which is difficult but tractable. Determining whether a decision was carried out, and carried out as intended, which requires knowing what should have appeared, where, by when, and then reading across systems that were never designed to answer for it. Companies do not lack the data. They lack the mapping, and the mapping does not hold still long enough to be written down once.

This is why the sections on absence later in this paper are not about search failing to return results. They are about inference across a moving expectation.

The distinction is not academic, because a system built for authored facts fails on resolved ones in a specific and predictable way. Given a question whose answer was decided rather than written, it will retrieve the documents and messages where the subject was discussed and synthesize something plausible from them. That output is indistinguishable, to the reader, from an answer grounded in a record. It is generated at read time from whatever fragments retrieval happened to surface, which means it can change between two identical queries, and it has no way to report that the underlying question was never actually settled.

The rest of this paper is about what the resolved half requires.

---

## 3. Formation: when does an utterance become a decision

Before a decision can have a state, it has to exist. This is the problem that looks easiest from outside and is the one most systems get wrong first.

The naive approach treats formation as extraction. Find the language of decision in a transcript, someone saying that a course of action will be taken, and record it. This produces volume immediately, which is why it is the common approach, and it is wrong in both directions at once. It records things that were floated and never settled, and it misses things that were settled without anyone announcing them.

Three properties of how people actually decide account for the failure.

**Decisions are rarely announced.** A meeting that settles a question often contains no sentence in which the question is declared settled. The people in the room leave understanding what will happen, and that understanding is spread across the whole conversation rather than sitting in any one line of it.

Take the supplier switch. A team argues about two suppliers for twenty minutes. Nobody ever says "we have decided to go with the second supplier." What happens instead is that someone says "alright, get their contract over to legal," someone else asks who is handling the transition, and a third person says they will let the current supplier know. The decision was made. Everyone in that room would tell you the same thing about what was decided. And there is no sentence you could quote as the decision. A system looking for the sentence finds nothing and concludes nothing was decided, which is the opposite of what happened.

What settles it is not in the meeting either. Two days later a mail goes to legal with the contract attached. A thread in the team channel works out who owns the transition. The following week a purchase order is raised against the new supplier. Not one of those is a decision. Together they establish that one was made, which is why formation has to accumulate across surfaces rather than read any single one of them.

The reverse error is just as common, and worse. In the same meeting, ten minutes earlier, someone said "we should probably just switch to the second one and be done with it." That is a clean, quotable, decision-shaped sentence. It was a suggestion. The discussion ran on for another ten minutes and could have gone either way.

A system that keys on decision-sounding language records that line as the decision. Now follow what that implies. Had the meeting ended the other way, with the team deciding to stay with the first supplier, that sentence would still have been said, and the same system would have recorded the same decision. The output is identical in both worlds. A record that reports the same decision whether or not the company made it is not reading decisions. It is reading vocabulary, and the vocabulary was available either way.

**Assent is not uniform across people.** The same words carry different weight

depending on who says them.

Two people sit in the same meeting. One of them says "yeah, sounds great, love it" about nearly everything, all day, out of habit. When that person says "sounds great," it tells you very little about whether they agree. The other person is quiet and rarely volunteers agreement about anything. When that person says "okay, fine," they have usually signed on. A system that scores the enthusiastic "love it" as strong agreement and the flat "okay, fine" as weak agreement has the truth backwards. It is not measuring who agreed. It is measuring who talks a certain way.

Reading it correctly means knowing what a given person's agreement normally sounds like, so that identical words can be weighed differently depending on who produced them. This concerns the reliability of a signal, not the merit of a person. The quiet one is not a better employee. Their "okay" is simply better evidence.

**Many decisions close because nobody objected.** A proposal gets made, nobody pushes back, and from that point on everyone behaves as though it is settled. Whether that silence was a decision depends entirely on who was silent.

Someone proposes moving a product launch from February to March, and no one objects. If the person who owns the launch and the engineering lead were both in the room and said nothing, that silence carries real weight, because the two people who could have stopped it did not. If neither of them was in the room, and the silence came from people with no say over the launch date, the same silence carries almost none. One silence, two very different weights.

What it does not do is settle the question by itself, and the distinction matters because the opposite reading is a tempting shortcut. Silence is ambiguous even when it comes from exactly the right person. It can mean assent. It can mean deference to whoever seems to be driving. It can mean a belief that the meeting was advisory and the real decision happens elsewhere. It can mean confusion about who owns the call, which is common in a company that has absorbed another one and is running two sets of assumptions about decision rights. It can mean disagreement that somebody did not feel safe voicing. Treating informed silence as a decision would be the same error as treating a decision-shaped sentence as one, arrived at from the other direction: a single signal read as dispositive when it is only evidence.

So informed silence raises a candidate rather than closing a question. It forms a decision in combination with what follows, when the work starts, when the date changes in the systems that carry it, when nobody behaves as though February is still live. Telling informed silence from uninformed silence means knowing who had the standing to object, which is frequently not who the org chart would suggest. Telling either of them from a decision means waiting for the rest of the evidence.

**A great deal of deciding never happens in a meeting at all.** The examples above are conversations because conversation is where the failure is easiest to see, not because it is where decisions live. Questions close in a Slack or Teams thread with four replies and no objection from the person who could have blocked it. They close in an email chain where the approval is the fifth message down. They close in a comment on a ticket that settles which of two approaches is being built, and in a document where the argument happens in the margin and someone resolves the thread. Every one of those carries the same three problems: the decision is rarely announced, assent means different things from different people, and silence signifies only when the right people were present to break it. A record that treats the transcript as the primary evidence and everything else as supporting material has the weighting backwards for a large share of what a company decides.

Those surfaces also differ from meetings in a way that matters later. A meeting produces evidence that a decision formed and almost never produces evidence that it was carried out. Messaging, mail, ticketing, purchasing, and CRM produce both, and often about the same decision. The thread where the supplier question closed is formation evidence. The mail sending the contract to legal, the ticket opened for the transition, and the purchase order raised the following week are artifacts, and they sit in the same systems the conversation did. Section 5 rests on that dual role, because a record assembled only from meetings can establish what a company decided and can never establish whether anything happened as a result.

These are the reasons formation cannot be reduced to extraction, and they are also the reason a formation system needs to model participants at all. The purpose of that modeling is not to evaluate people. It is to correctly interpret evidence that people produced. A system that ignores the difference is not neutral toward its participants, it is simply wrong about them in a uniform way.

Rithmo's approach treats formation as a threshold on accumulated evidence rather than a classification of individual utterances, with the weight of each piece of evidence calibrated to its source. Where the evidence does not clear the threshold, the correct output is not a low-confidence decision. It is no decision, held as a candidate, with the gap available as a finding in its own right.

---

## 4. State and supersession: a taxonomy of transitions

A decision that has formed then has a life. It gets acted on, revisited, changed, and eventually stops being the current answer. The question a record has to answer is narrow: what is true right now.

Supersession, as this paper uses it, means any point at which an existing decision stops being the current answer, whatever the reason. It is one word covering several different events, and separating them is the work of this section.

Version history does not answer it. A list of everything said about a subject, in order, is the raw material for an answer and not the answer itself, because the relationship between successive statements is what determines the current state, and that relationship is not recoverable from ordering. The convention is to call the whole family of these relationships supersession and treat it as one transition in which a later decision replaces an earlier one. That convention is the single largest source of error in decision records, because at least four distinct transitions hide inside it, and they have different correct answers.

To keep the four side by side, each one below is shown happening to the same decision. In March, a team decides to switch from their current supplier to a new one, with the changeover to be finished by the end of June. That is the decision on the books. Here are four different things that can happen to it next, all four of which a flat record reports identically.

The same decision returns in sections 6 and 8. Holding one example constant is what makes the differences between the transitions visible, since varying the example would leave a reader unable to tell which differences are the argument and which are the scenery.

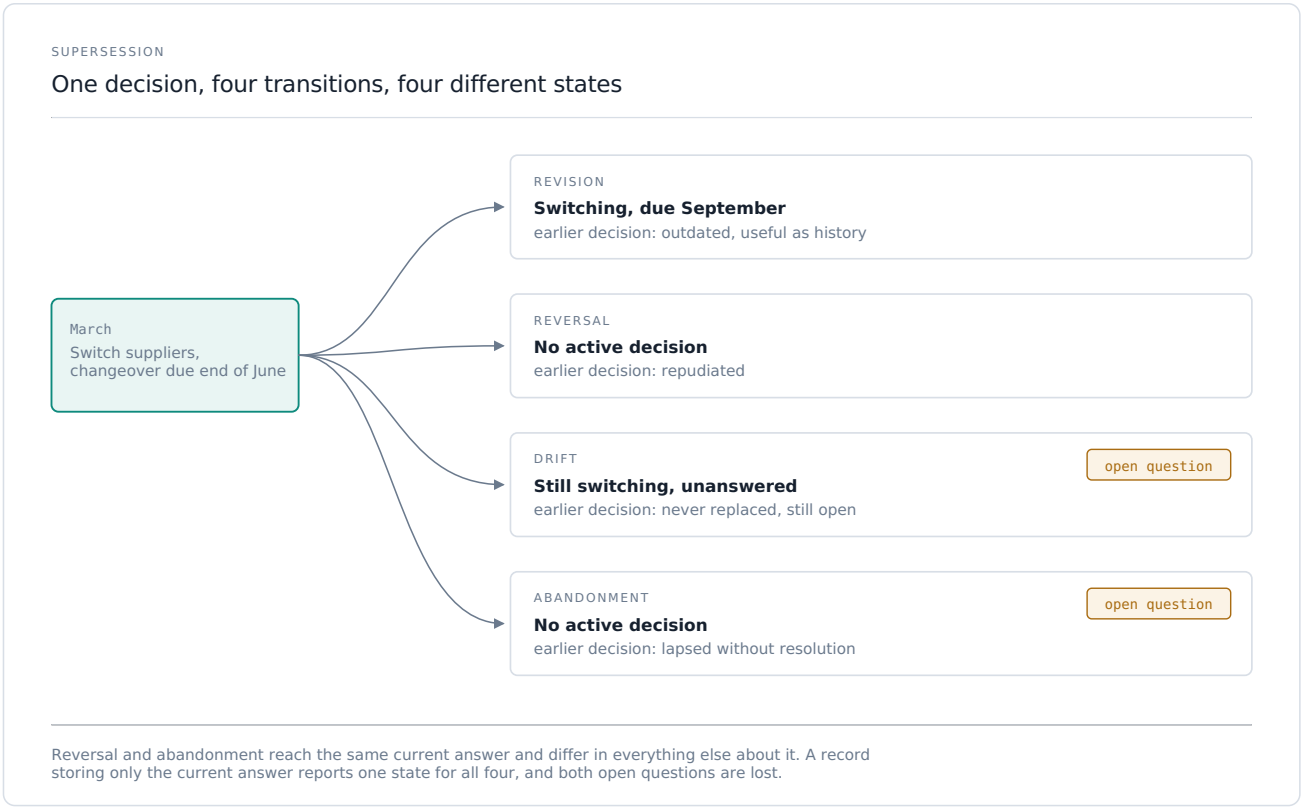



Figure 4. The same March decision, four transitions, four different states. Reversal and abandonment arrive at the same current answer by opposite routes. Two of the four leave a question still open, and a record that collapses them loses both.

## 4.1 Revision

The subject stays the same and one detail of the decision moves.

In May the team meets again. The switch is still happening. The changeover is now due at the end of September instead of the end of June, because the new supplier cannot staff it any sooner.

The current answer is the later decision: switching, due in September. The earlier decision is not wrong, it is out of date, and its history is worth keeping, because a date that has moved three times is itself a finding worth surfacing. This is the one case a flat record handles correctly, which is why flat records survive as long as they do.

## 4.2 Reversal

The subject stays the same and the decision is negated.

In May the team meets again and calls it off. They are staying with the current supplier. Nothing about the switch is happening.

The current answer is that there is no active decision to switch suppliers, and that is a different kind of statement from a moved date. Anyone still holding the March

decision is holding something that was thrown out, not something that was updated. Collapse reversal into revision and the record reports an active supplier switch with a later date, when the truth is that the switch was cancelled. In a plain list of versions the two look the same. To anyone acting on the record they are opposites: one means keep going, the other means stop.

### 4.3 Drift

The subject changes while the conversation carries on as though it has not.

The supplier switch comes up again in April. Somebody raises that the two offices order separately and that is part of why the current supplier relationship is a mess. The next meeting is mostly about consolidating purchasing across the two offices. So is the one after that. By June the team has a real decision about consolidating purchasing, and the question of which supplier to use has not been answered since March. Nobody ever announced a change of subject. Everyone involved experienced it as one continuous thread about the supplier problem.

Drift is the hardest of the four and the most damaging when missed, because both of the obvious readings are wrong. Read as supersession, the record reports the supplier question as settled by a decision that was actually about purchasing structure, so the supplier question vanishes while appearing to have been answered. Read as unrelated, the record holds two decisions with no link between them, and the fact that one quietly consumed the attention the other needed is lost.

The same thing happens with dates. Two teams agree in January to launch in April. In February somebody points out that April only works if one feature gets cut. The next meeting is about which features are in and which are out, and so is the one after that. By March the team has made real decisions about scope, and nobody has said a word about the launch date since January. One team is still planning for April. The other has quietly stopped believing in it. Nobody changed the date, and nobody confirmed it either.

Handled correctly, the March supplier decision stays open, the June purchasing decision stands on its own subject, and the relationship between them is recorded as drift rather than replacement. A record that can hold that distinction can report the thing leaders describe constantly and can never point to: the question that keeps coming up and never gets answered, because every conversation about it turns into a conversation about something adjacent.

### 4.4 Abandonment

The decision stops being pursued and nothing replaces it.

The March decision was to switch to the new supplier, with the changeover

finished by the end of June. In this version nothing happens to it at all. It just stops coming up. No one reversed it. No one moved the date. The person driving it moved to another team, the quarter got busy, and by August nobody has mentioned it in five months. The old supplier is still being paid, because nothing ever changed.

The current answer is that there is no active decision, arrived at without anyone deciding anything. That is different from reversal, where someone decided not to, and the difference determines what should happen next. A reversal is a closed question and needs nothing from anyone. An abandonment is an open question that stopped being asked, and it can be picked back up the moment somebody knows it is sitting there.

This is the transition that costs companies the most and is the hardest for them to see, and there are two reasons for that. The first is that there is no event to notice. Nothing happened, for months, and nothing happening does not appear in any system.

The second is that the person best placed to catch it is the least likely to look. The people who made the decision in March consider it closed. They are not waiting on an update and have no reason to check, because in their heads the matter is settled and the work is somebody's routine follow-through. So the decision is filed as done by everyone who cared about it, while nothing is being done by anyone.

What eventually surfaces it is the artifact asymmetry (2.1), where the expected proof was something stopping. The invoices from the old supplier keep arriving, month after month, looking completely normal, because an arriving invoice is the most ordinary event in the system. Usually somebody notices during a budget review, two or three quarters later, and by then the question is no longer whether to switch suppliers. It is why nobody caught this.

## 4.5 Decisions are not peers

Everything above treats decisions as equals. One decision holds a subject, a later one supersedes it, and the four transitions describe how. That is a horizontal relation, and it is not the only one.

A decision to fix a bug is a real decision. It is also made in service of a feature, which was made in service of a product, which was made in service of a direction the company chose. Each of those is a genuine decision with its own evidence, owner, and state, and none of them supersedes any other. They can all be current at once. What connects them is subordination: the lower decision exists because the higher one does and would not have been made otherwise. A record with only the horizontal relation cannot represent this, and four things follow from that.

**The relation is nowhere authored.** A ticketing system has parent links because

somebody typed them, and it has them only for the work that lives inside that system. Nobody announces that a bug fix is in service of a launch, or that a launch is in service of a strategy, and the decisions being connected usually live on entirely different surfaces and were made by different people months apart. So subordination has to be established from evidence under the same rule as everything else in this paper: a connection somebody actually drew, not a resemblance strong enough to be persuasive.

**State propagates asymmetrically, and the downward direction is where the cost sits.** A parent decision that is reversed or abandoned does not stop its children. Teams keep building against a direction the company left in March, and every individual decision under it was properly made, properly owned, and correctly recorded. This is not work with nothing authorizing it (5.4), because the authorization existed. It was withdrawn above them and nothing carried the news down. A record holding decisions as peers cannot state this at all, which is unfortunate, because it is among the most expensive things that happens in a large company. The reverse direction rarely holds: a child decision being reversed usually says nothing about its parent.

**It is not a tree.** One decision can serve two parents, as a fix that is required both by a launch and by a compliance commitment. Depth is not uniform either, since some decisions sit two levels beneath a strategic choice and others sit six, and no level has a stable name across companies. The structure is a graph, and any attempt to force it into fixed tiers will be wrong in a different way at every organization.

**Significance is altitude, not importance.** It is tempting to rank decisions, and ranking is the wrong model, because significance is not a property a decision carries. It is a relation between the decision and whoever is asking. A bug fix is invisible to an executive until it is the reason a launch slips, at which point it is briefly the most important decision in the company. So what a record needs is not a priority score. It needs the ability to answer at a chosen altitude, and to surface a low decision precisely when it bears on a high one.


Figure 5. Subordination is a second relation, and state does not propagate down it. Every decision beneath the withdrawn one was correctly made, correctly owned, and correctly recorded.

## 4.6 Why the distinctions are load-bearing

The four transitions produce four different states, where a state is a pair: the current answer, and the status of the earlier decision. Only three current answers appear among the four, because reversal and abandonment both leave no active decision. That collision is the point rather than an untidiness. A record holding only the current answer cannot separate a question somebody closed from a question that stopped being asked, and those two want opposite things from whoever reads them.

| Transition | Current answer | Status of the earlier decision | Open question remains |
|---|---|---|---|
| Revision | the later decision | outdated, still relevant as history | no |
| Reversal | no active decision | repudiated | no |
| Drift | earlier decision still open | unresolved, not replaced | yes |
| Abandonment | no active decision | lapsed without resolution | yes |

A record that collapses these reports the same state for all four. Two of the four leave a live open question, and a collapsed record loses both, which is the specific

mechanism by which decisions in organizations quietly fail to happen. This is not a reporting deficiency. It is why the problem is invisible.

---

## 5. Absence: a taxonomy of what did not happen

A record that cannot report absence cannot be relied on, because a reader has no way to distinguish nothing was decided from the system did not find it. The first is information. The second is not an answer at all, and it will almost never look that way.

The difference is easy to feel. Ask a colleague whether the company ever approved the new pricing, and there are two very different answers they can give. One is "no, that never got decided." The other is "I don't know, I couldn't find anything." The first answer lets you act. The second one leaves you exactly where you started, and if you act on it anyway you are acting on a guess. Most systems can only give the second answer, and most of them phrase it like the first.

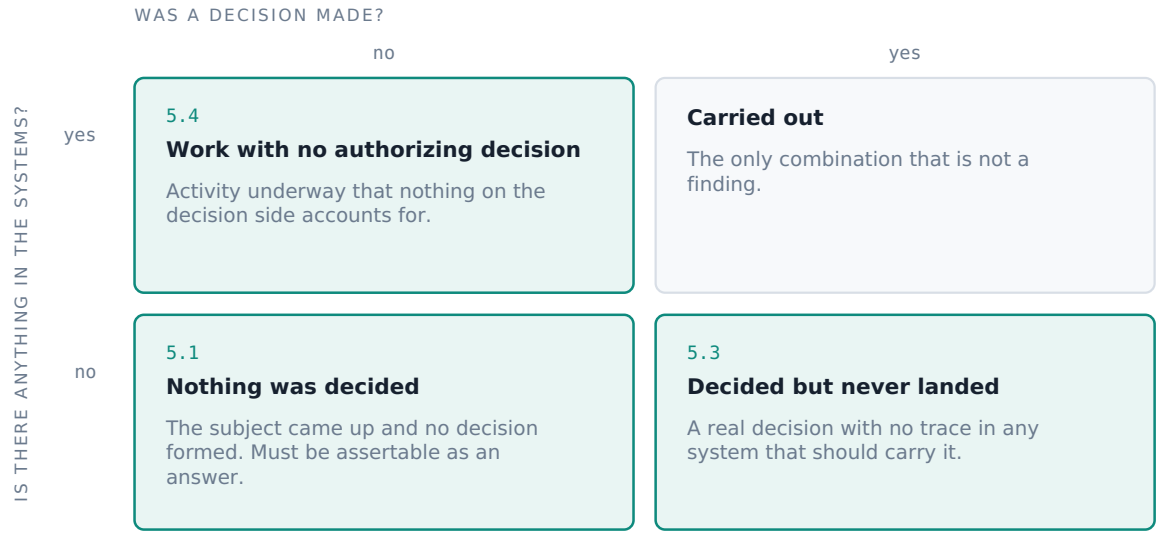
Systems that work by retrieving relevant fragments cannot tell the two apart even in principle, because finding nothing relevant and there being nothing to find produce the same empty result inside the system.

Absence is also not one thing. Four varieties are separately meaningful and separately actionable. Nothing was decided at all (5.1). A subject was raised over and over and never settled (5.2). A decision was made and no system reflects it (5.3). Work is underway that no decision authorizes (5.4).

Three of those come down to the same two plain questions asked together. Was a decision made, and is there anything in the systems to show for it. The exception is the subject raised again and again and never settled (5.2), which is not about artifacts at all. It is a pattern across time.

ABSENCE

## Was a decision made, and is there anything to show for it




The two shaded cells on opposite corners are the same relationship read in opposite directions: a decision with no artifact, and an artifact with no decision.

*A fourth variety, 5.2, a subject raised over and over and never settled, is a pattern across time rather than a combination of these two questions, so it does not appear on this grid.*

Figure 6. Three of the four combinations are findings a company has no ordinary way to name. Only one of them is the normal case.

### 5.1 Nothing was decided

The subject came up, possibly many times, and no decision ever formed. There is plenty of evidence that people talked about it. There is not enough to say they settled it.

The new pricing for the enterprise plan is discussed in three separate meetings. People float numbers. Somebody builds a spreadsheet. Somebody says the finance team should weigh in and the finance team never does. Nothing was approved. Asked "what is the approved price for the enterprise plan," the honest answer is not a number and not a shrug. It is that no decision exists, and here are the three conversations that show it was raised and left open.

This is the variety a decision layer has to be able to state as an answer in its own right rather than as an empty result. It is the basis of refusal: a query returns the current sourced record, or it returns that no decision exists, and it never returns a plausible-looking construction. A record that can only fail to answer gets topped up with whatever the person or the system asking happens to assume, which is the exact condition the record exists to eliminate.

### 5.2 Raised repeatedly and never resolved

A subject comes up in meeting after meeting and never closes. Each individual instance looks completely ordinary. The pattern is the finding.

The enterprise pricing question comes up in the leadership meeting in February, again in March, twice in April, and again in June. Every one of those conversations was a reasonable use of twenty minutes. Nobody in any of them was doing anything wrong. But the fact that the same question has now consumed the better part of two hours across five months, and is no closer to an answer, is a real and specific finding about the company, and no one in the room has it. Each person remembers the last conversation. Nobody is holding the count.

Seeing it requires treating those five conversations as five instances of one unresolved subject. That means the subject has to be the thing that persists in the record, with each conversation attaching to it as another instance and the state of the question carried alongside.

Nearly every record a company keeps today is shaped the other way, with the meeting as the unit. Five discussions produce five sets of notes. Each one is an accurate record of its own meeting, each is useful for remembering what was said that day, and not one of them holds a count. Nobody is wrong, and the finding is nowhere, because the record has no place to put a fact about five meetings at once.

### 5.3 Decided but never landed

A decision genuinely formed, and no system anywhere reflects it.

An account manager promises a customer, live on a call, that their renewal will hold at the current rate for another year. Everyone on the call heard it. It is a real commitment the company is now on the hook for. It is not in the CRM. Nine months later the renewal quote goes out at the new rate, generated correctly by a system that had no way of knowing.

What makes this one expensive is when it shows up. For nine months the gap costs nothing at all, and fixing it would have taken one minute and one field. It becomes visible at the exact moment it is most expensive, with the renewal on the table and two bad options: honor a promise nobody wrote down, or tell the customer they were told wrong.

This is the artifact asymmetry from 2.1 in its most direct form. The finding is not about the decision and it is not about the system. It is about the relationship between them, which means answering it needs two things the decision record does not contain on its own. It needs an expectation, meaning some statement of where a decision of this kind ought to show up. And it needs coverage, meaning knowing which systems have been watched closely enough that their silence

actually means something. A gap reported in a system nobody has looked at is not a finding about the company, it is a finding about the blind spot.

That is also why absence cannot be stored. It is a conclusion assembled at the moment of asking, out of what was expected, what is present, and how well the looking was done. Store it and the record will confidently report a gap that somebody closed three hours ago.

Rithmo's approach treats this as a query over expectation and observed presence rather than as a stored state, so a gap exists only for as long as it is still true.

### 5.4 Work product with no authorizing decision

Work is underway and no decision anywhere authorizes it. This is the decision that never landed (5.3) read backwards: there is an artifact, and nothing on the decision side accounts for it.

Two engineers have spent six weeks building against the second of the two approaches that were debated. The debate never closed. They picked the one that seemed likely to win and got started, which is a reasonable thing for a competent person to do, and six weeks of work now depends on a decision nobody ever made.

A caution belongs here, because this is the variety most easily reported wrongly. Work does not need a senior decision behind it to be authorized. A manager setting a sprint, a team lead exercising delegated technical discretion, an owner reordering their own backlog: each of those is a decision in its own right, made at the altitude where it belongs (4.5), and work proceeding from one is authorized work. The finding is not that no executive decision exists. It is that nothing at any altitude authorizes the work. A record that cannot represent subordination will report every delegated choice as an absence, which is worse than reporting nothing at all, because it converts ordinary operating autonomy into a finding and the loop that carries it into noise.

The quieter version of this is any process that runs on its own. A renewal that renews, a report that goes out, a job that keeps posting. Nobody decided to continue. Nobody decided anything, and the absence of a decision gets carried out faithfully by something that only knows how to keep going.

This is the variety with the largest consequences of the four, because the other three describe decisions that produced no work, and this one describes work that proceeded from no decision. Finding it requires the ability to compare activity against decision state and say that nothing on the decision side accounts for the activity, which is the same capability that finds a decision with no artifact (5.3), pointed in the opposite direction.

### 5.5 The common requirement

All four require the same underlying ability, which is to state a negative as an answer. That is harder than it sounds, because the behavior it has to replace is not silence.

Ask most systems a question that has no answer and they do not come back empty. They come back with something. It reads well, it carries the same confidence as a real answer, and it is assembled out of whatever material happened to sit nearest the question. Nothing in the output marks it as manufactured, so the reader cannot tell it apart from a real answer and proceeds on it. An empty result would be an inconvenience. This is a hazard, and it is the normal case rather than the edge case.

Stating a negative rests on two things. A threshold, so that evidence falling below it produces an explicit no-decision value rather than a weak decision presented as a firm one. And coverage, meaning the system keeps track of which places it has actually been reading, because silence only means something if you were watching. Without the first it guesses. Without the second it reports gaps that are really its own blind spots.

Coverage also has to reach past the places where decisions get discussed. Two of the four varieties turn on artifacts directly. A decision that never landed (5.3) and work that nothing authorizes (5.4) are both comparisons between a decision and the systems that were supposed to carry it, and those systems are messaging, mail, ticketing, purchasing, CRM, and project systems rather than any transcript. A record fed only by meeting capture can say what the company decided and can never say whether it landed, so it forecloses both, and those two carry most of the cost. It weakens a third as well, because nothing was decided (5.1) is only separable from work nobody authorized (5.4) when the artifacts are in view too.

---

## 6. Provenance and time: why the current answer is current

Sections 3 through 5 describe what the record has to contain. This section is about something different: whether anyone can check it.

A record that gives the right answer and cannot show its work is worth less than it looks. The reader's only choices are to take it on faith or to go ask around, and asking around is the thing the record was built to replace. So the record has to answer a second question alongside the first one. Not only what is the current answer, but why is this the current answer, and how do you know nothing since then has changed it.

Two things are needed for that. A trail of evidence a reader can walk, which is

what provenance means here. And a way of handling time, because the history of a decision does not arrive in the order it happened.

### 6.1 Three clocks

Take the supplier switch from section 4. In March, a team decides to switch from their current supplier to a new one, with the changeover to be finished by the end of June.

That one decision raises three separate questions about time, and they have three different answers.

**When it became true.** The switch became the company's position in the March meeting. From that moment the company had a position, whether or not anything anywhere recorded it. Every system carries this clock for the things it stores, which is exactly the problem. Nothing was storing this one.

**When the record came to believe it.** Not March. The March meeting on its own did not settle the question, and a record that formed a decision from it alone would have been guessing. Belief arrived in April, once enough evidence had accumulated to cross the evidence threshold that forms a decision (section 3). A record keeping this clock alongside the first can say what it believed on a given date, and can tell a late correction apart from a real change.

**When the evidence was produced.** The March meeting produces most of it. A thread a week later carries the argument on. Ownership is settled in early April. A purchasing note in May makes clear that everyone had understood the March conversation as settling the question. This clock is not a moment. It is a spread, and it runs for eight weeks.

The first two are moments, so a system can hold each in a column. The third cannot be held that way, and that is the whole of the difficulty.

Separating the first two clocks is the bi-temporal approach, worked out in the temporal database literature in the 1990s and applied by several recent agent memory systems to facts extracted from conversation. It is well understood and it is not in dispute here. Whether it is sufficient for decisions is taken up with fact validity in agent memory (7.5).

The third clock is the one nothing keeps, and the consequence shows up as soon as evidence arrives out of order.

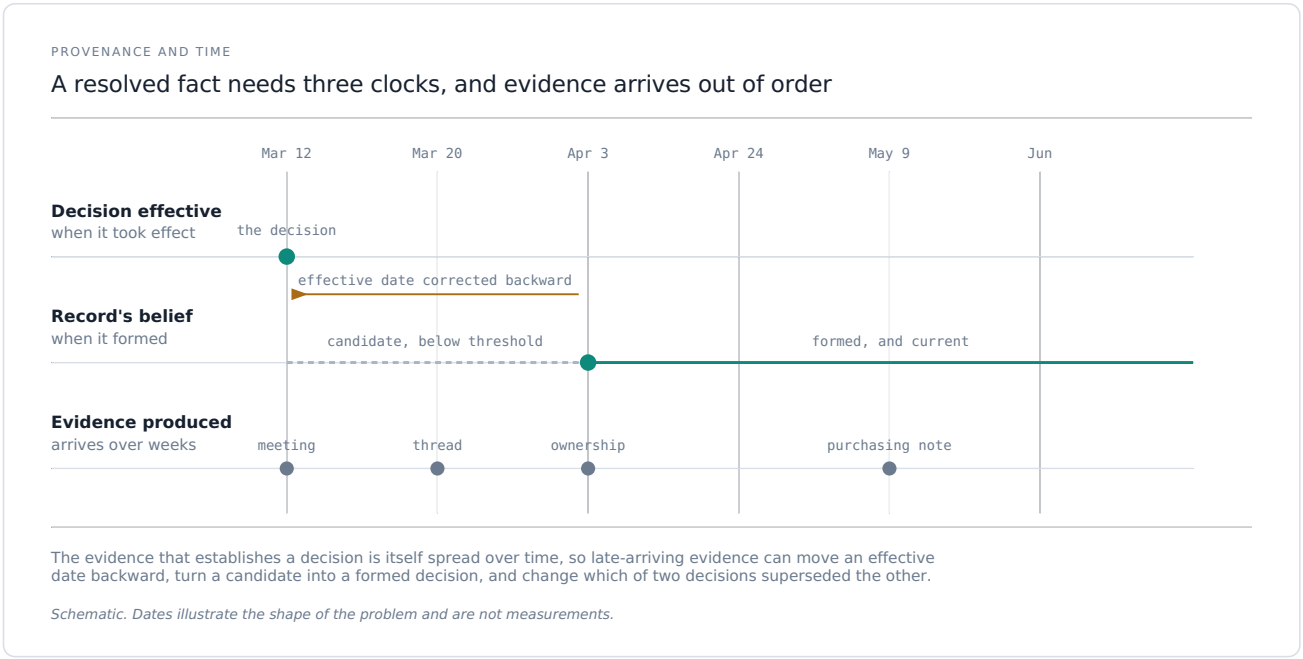



Figure 7. Schematic. The decision became the company's position in March. The evidence establishing that arrived over the following weeks, and the record only formed the decision once the accumulated evidence cleared the threshold, at which point the effective date moved backward to March.

Late-arriving evidence can move a decision's effective date backward, can turn a candidate into a formed decision retroactively, and can change which of two decisions superseded the other. A record that assumes evidence arrives in the order events occurred will be wrong in exactly the cases that matter most, because the conversations that clarify a murky decision almost always happen after it.

## 6.2 Provenance is a chain, not a citation

A source link is not provenance. A retrieval system that cites the passages it generated from is showing what it read, which is a different claim from showing why the answer is current. The first says here is where this came from. The second says here is why nothing after it changed the answer.

For the supplier decision, the current answer is that the switch is happening, due at the end of September. The provenance of that answer is a walkable chain:

- the March meeting, and the evidence in it that formed the decision
- the owner, and where ownership was established
- the May decision that moved the date, typed as a revision rather than a replacement
- the March date it superseded, retained rather than overwritten
- the absence of anything after May that bears on the subject

That last item is part of the chain and is the one most often left out. "Nothing has happened since May that changes this" is a substantive claim, and it is only available to a record that knows which surfaces it has been reading and over what window.

### 6.3 Why this makes the record arguable

The practical value of a walkable chain is that a reader who disagrees can disagree with a specific step.

Somebody reads that the supplier switch is still live and believes it was called off. With a chain, they can look at the step where that would have happened and find either the reversal or its absence. Perhaps they are thinking of a conversation the record never saw, in which case the record was right about what it observed and the gap is now identified. Perhaps the record typed a reversal as a revision, in which case it is wrong at a locatable point and can be corrected there.

An answer that cannot be inspected at a specific point can only be accepted or rejected whole, and in practice it gets rejected whole the first time it is wrong about anything. This is the difference between a record people come to rely on and one they stop opening. Provenance is what converts disagreement from a reason to distrust the record into a mechanism for improving it.

### 6.4 Answers as of a date

The three clocks make a second kind of question available: not what does the company hold to be true now, but what did it hold to be true on a given date.

This matters in three situations. In a post-mortem, where the useful question is what the state actually was when a decision was acted on rather than what it is today. In a dispute, where a customer was told something on a call in February and the question is what the company's position was in February. And for anything automated, where the only way to explain why a system did what it did is to know what the record returned at the moment it was asked, which is not necessarily what it would return now.

The third case is the one that grows in importance. An agent that acted correctly on the record as it stood is in a completely different position from one that acted on nothing at all, and only a record with a time axis can tell those apart after the fact. This is a property of the three-clock model rather than a claim about a reporting interface that exists today.

### 6.5 Absence needs provenance too

Section 5 argues that the record has to be able to assert that nothing was decided. Section 6 adds the requirement that it has to be able to show why that assertion is

credible, and this is easy to overlook because an absence looks like it has nothing to point at.

It does. The provenance of an absence is coverage: which surfaces were read, over what period, and how completely. The claim that the enterprise plan price was never approved is worth something if it comes with the three meetings where it was discussed and left open, the window observed, and the systems checked. Without those, it is indistinguishable from the system having failed to look, which is precisely the ambiguity that treating absence as an answer (section 5) exists to remove.

This is also the honest boundary of the claim. A record can say no decision was made on any surface it observed. It cannot say no decision was made anywhere, and it should not be built to imply otherwise. Stating the scope of the negative is what makes the negative usable.

---

## 7. Existing approaches, and what each one is for

Nothing currently holds the resolved half, and it is not because nobody tried. It is because every available approach was built to answer a different question, and most of them answer that question well. This section takes them in turn, states what each is genuinely good at, and identifies the specific property it does not produce, whether that is forming a decision from evidence, telling the supersession transitions apart, answering that nothing was decided, or showing why the current answer is current (sections 3 through 6). They are taken as categories rather than as products, which is the level at which the argument holds: a named tool may well combine two of them, and the gap survives the combination. In each case the gap follows from what the approach was built for. Whether it could be closed, and what closing it would cost, is a separate question, taken up in 7.11.

A modern AI application stack is layered, and each of the software approaches below occupies its layer correctly. The argument here is not that any of them is wrong. It is that none of them is holding the resolved half, and that a stack assembled without it will execute confidently against decisions nobody made.

One approach is not part of that stack at all, and it is the one companies actually rely on, so it goes first.

### 7.1 The human way

This is how the work is done today, and it is done by people. Ask the person who was in the room. Keep a decision log. Use a meeting-notes template with a

decisions section. Draw a RACI chart. Assign an owner to every action item. Employ a chief of staff whose real job is remembering what the company decided and chasing the people who owe something.

It deserves more credit than it usually gets. A person in that role brings judgment no system has: they know which decisions matter, who actually needs to be told, when to escalate and when to let something sit, and who to ask when the record is unclear. For a small team working in one room, the human way is close to optimal and needs no tooling at all. Most companies operate this way because for a long stretch of their life it works.

It fails at scale for three structural reasons, and none of them is about diligence.

**It requires somebody to notice.** Every written artifact of the human way, the log, the template, the action list, can only capture a decision that someone in the room recognized as a decision at the time. Formation is the problem (section 3): much of the time nobody does. The supplier switch closes through a contract handoff, a question about who is managing the transition, and an offer to call the incumbent. Nobody experiences that as the moment of decision, so nobody writes it down, and the template comes back with that line blank. Discipline cannot fix an omission that nobody perceived.

**It requires a stable map from decisions to expected artifacts.** A checklist is a statement of what a decision of a given kind ought to produce. The artifact asymmetry is the problem (2.1): that map is non-stationary. It changes when the systems change, when the owning team changes, when the process changes, and it changes without anyone restating it. So the checklist is accurate on the day it is written and quietly degrades from then on, and the degradation is invisible because a checklist that is being completed looks the same whether or not it is still asking for the right things.

**It is bounded by one person's attention and tenure.** The human way runs on someone holding the state in their head. That person has a finite number of things they can hold, they are not present for most of the company's conversations, and eventually they leave. What decays fastest is memory of the abandonments, because there is no event to remember. A reversal is memorable. A revision is memorable. Six months of nothing happening leaves nothing to recall.

**The cost is real and nobody has the number.** The expense of the human way is not hidden. It is distributed, which is different, and the parts of it that could be counted cannot be counted without already knowing which decisions were made and which never landed. That is precisely what the company does not have. So the human way has been the only option, and its cost has been treated as a condition of doing business rather than as a line item. Section 9.2 works the figures once there is a record to compute them from.

### 7.2 Transcript extraction

Techniques: automatic speech recognition, abstractive summarization per meeting, LLM-based information extraction, action-item and commitment detection.

What it does well: recall of a single meeting, and it is now very good at it. Given one conversation, these systems will produce an accurate summary, a defensible list of what was said, and a usable set of follow-ups. For the purpose of not having to re-watch a call, the problem is solved.

What it cannot produce: anything that requires a subject to persist across meetings. Each meeting is processed as a document, and the output is scoped to that document. There is no entity that the February discussion and the June discussion are both about, so there is nothing for a second instance to accumulate against. That forecloses recurrence (5.2), because five discussions of one unresolved subject are five unrelated summaries. It forecloses all four supersession transitions (section 4), because supersession is a relationship between decisions and no relationship is modeled. And it inherits the formation error in both directions (section 3): the quotable suggestion gets recorded and the unannounced decision does not.

Extraction produces a good record of what was said in a meeting. The resolved half needs a record of what the company currently holds to be true, and those are different objects.

### 7.3 Retrieval over the corpus

Techniques: dense and hybrid retrieval, reranking, retrieval-augmented generation over an organization's documents and messages.

What it does well: finding relevant material in a large corpus, and answering questions whose answers are written down somewhere. For authored content this is the right tool and often the complete answer.

What it cannot produce: determinism or refusal. The answer is synthesized at read time from whichever passages retrieval surfaced, so two identical questions can return different answers, and neither is anchored to a record that could be checked. More seriously, the system has no way to distinguish having found nothing relevant from there being nothing to find. Both states are the same empty result internally, and what gets produced from that empty result is a plausible construction assembled from whatever adjacent material was returned. Asserting that nothing was decided (5.1) is unreachable by construction: it cannot assert that no decision exists, because it has no representation of a decision to assert the absence of.

Retrieval answers where was this discussed. The resolved half has to answer what

is true now, and refuse when the answer is nothing.

### 7.4 Authored semantic layers

Techniques: dimensional modeling, metric definitions maintained as code, governed semantic models over a warehouse.

What it does well: this is the correct and mature solution for the authored half. A governed definition of a metric, versioned and owned, is how a company makes sure that two teams asking the same question get the same number. Section 2 says the authored half is well served, and this is what it means.

What it cannot produce: anything whose definition nobody authored. The mechanism requires a person to write down what the fact is, which presupposes that somebody knows it and recognized it as a fact worth defining. A resolved fact has neither property at the moment it becomes true. There is no field waiting for the supplier decision, and no owner whose job is to keep that field current, because the fact did not arrive through a system that has fields.

Semantic layers govern facts a company wrote down. The resolved half is the facts it decided and did not write down.

### 7.5 Fact validity in agent memory

Techniques: bi-temporal modeling, meaning the separation of valid time, when a fact was true in the world, from transaction time, when the system came to believe it. The approach comes out of the temporal database literature, developed through the 1990s in work associated with Snodgrass and Jensen and the TSQL2 design effort. That route into the standard did not survive, since the temporal part of SQL was cancelled in 2001. Bi-temporal support arrived separately in SQL:2011, through a later and differently designed proposal, as application-time period tables and system-versioned tables. Several recent agent memory systems apply it to facts extracted from conversation, invalidating a stored fact when a contradicting one arrives and retaining the superseded version with its validity interval.

What it does well: a great deal, and this is the most technically serious of the approaches here. Bi-temporal modeling is the right answer to point-in-time correctness. It can tell you what the system believed on a given date, distinguish a late-arriving correction from a genuine change in the world, and keep an audit trail that survives revision. Any system that needs to reason about its own past beliefs should be doing this. It is a real contribution and it is not in dispute here.

What it cannot produce: drift and abandonment, two of the four transitions in section 4. The trigger is contradiction. Invalidation fires when an incoming assertion conflicts with a stored one, and revision and reversal both generate one, so both are handled correctly. Drift generates none: the supplier question is never

contradicted, the conversation simply becomes about purchasing structure, and no statement anywhere conflicts with the March decision. Abandonment generates nothing at all, which is the definition of it. A contradiction-triggered mechanism is blind to both by construction, and those are exactly the two transitions that leave a live open question.

Three further limits follow from what is being modeled. The unit is a fact, so there is no owner field and no notion of who had standing to object, which formation requires (section 3) and which the supersession transitions depend on for routing (section 4). The scope is typically a user or a session rather than an organization, which is a different granularity from the one a company-wide decision record needs. And absence resolves to not found rather than to nothing was decided, which is the difference between nothing found and nothing decided again (5.1).

Fact validity keeps a stored fact honest about time. It does not decide whether a decision was ever made, and it cannot see a decision stop being pursued.

### 7.6 Integration and automation platforms

Techniques: event-triggered workflow orchestration, webhooks, durable execution engines, scheduled jobs.

What it does well: executing a rule reliably, at scale, forever. If a company knows what should happen when a given event occurs, this class of tool will make it happen every time, with retries and observability. It is the correct layer for acting on a decision once the decision is settled.

What it cannot produce: the decision itself, or any awareness of its absence. These platforms consume decisions and never hold them. The rule encodes what someone decided at the time the rule was written, and the platform has no way to know that the decision has since changed, because a workflow does not read a decision record. There isn't one. And the failure mode is exactly the abandonment case: nothing fires when nothing happens. A vendor renewal running on a schedule will continue running correctly and indefinitely after the company decided to end it, because the platform is doing its job and the job never included checking whether the decision still holds. The self-running streams described in the artifact asymmetry (2.1), the invoices that keep arriving and the roles that keep getting posted, are often an automation platform behaving perfectly.

Automation acts on decisions. Something has to hold the decision it acts on.

### 7.7 Context written into prompts

Techniques: system prompts, curated context blocks, few-shot examples, retrieval templates with hand-authored policy.

What it does well: encoding stable policy, tone, and constraint. For anything that

genuinely does not change, writing it into the prompt is simple, fast, and appropriate.

What it cannot produce: any of it, for a reason that is easy to miss. Writing the current decision into a prompt requires already knowing the current decision, which is the entire problem. The prompt is a place to put the answer, not a way to get it. Past that, every reversal becomes a search: one change to one decision means finding every prompt, workflow, and agent that embedded it, and nothing tracks where it was embedded. And a prompt cannot refuse. It proceeds on whatever value it was given, with no way to represent that the value is stale or that no decision was ever made, because a hardcoded string carries no state.

### 7.8 A general-purpose assistant with the corpus loaded

Techniques: long-context language models, workspaces holding uploaded transcripts and documents, and direct tool connections that let a model read messaging, mail, and project systems on demand.

What it does well: a great deal, and it is the strongest single-shot reasoner in this section by a wide margin. Given a well-chosen body of material and a specific question, a capable model will find the relevant passages, weigh them against each other, notice where they disagree, and produce an answer a careful person would recognize as considered. For one person investigating one question over a corpus they assembled themselves, this is the best tool available today. It is also why many teams believe the problem is already solved.

What it cannot produce: determinism, refusal, or a bounded observation window. These are the same three failures as retrieval over the corpus (7.3), arriving in a far more convincing package. The answer is assembled at the moment of asking. Ask twice and the two answers are not guaranteed to agree, and neither is anchored to an entry that could be checked, corrected, or cited afterward. Nothing is stored, so nothing accumulates. The reasoning that resolved a supplier question in March is discarded, and April's question re-derives it from the same raw material at the same cost.

Absence fails structurally rather than for want of quality. To answer that nothing was decided, a system has to know the boundary of what it looked at. A workspace holds whatever somebody uploaded, and a connected tool returns whatever a query happened to match, so there is no answer to the question of what was in scope. Nothing found and nothing decided stay indistinguishable (5.1), and a model asked for the current state of a decision will produce a plausible one.

Two of the four transitions are invisible for a reason no amount of context repairs. Reversal and revision leave contradicting statements in the corpus, and a good reader will catch them. Drift leaves no contradiction, only a conversation that gradually became about something else. Abandonment leaves nothing at all. The

evidence that a decision was abandoned is the absence of activity against an expectation, which is not a passage anywhere in the corpus, so no amount of reading the corpus surfaces it.

Three practical limits bind before the structural ones do. Establishing one decision's current state costs a full pass over the material, paid per question, which an agent checking a decision before it acts pays per action rather than per meeting. Coverage is whatever the person assembled, so the decisions most likely to be missing are the ones made in rooms that person was not in, which is exactly where the value was. And two people asking their own workspace the same question get two answers, each defensible, neither authoritative, which is the condition a shared record exists to end.

None of this is an argument that the model is the wrong tool. Reconciliation is itself a model-driven process and the approach in section 8 uses one throughout. The difference is where the reasoning lands. Here it is spent at read time and thrown away. There it is spent once, written to an entry with its evidence attached, and read back by everyone afterward at the cost of a lookup.

### 7.9 The organizational memory platform

Techniques: recording-first capture of meetings and conversations, an entity graph linking those conversations to the documents and system records they touch, permission-aware retrieval over the result, and an interface exposing the whole thing to agents.

What it does well: this is the closest neighbor in the section by position, and it solves real problems. It gives an organization one place where conversational context, documents, and system records are connected rather than scattered. It carries source permissions through to the answer, which most approaches here do not. It exposes that context to agents over a protocol rather than through bespoke integrations. For the question of what was said about a subject and what it connects to, this is the strongest approach available.

What it cannot produce: any of the four capabilities, because the word doing the work is memory. A memory is a store of what happened. A record of decisions is a claim about what is currently true, and the distance between those two is the whole of this paper.

The difference shows up in four places. **A graph edge is not a formed decision.** Linking a conversation to a contract asserts that the two are related, which is a fact about documents. Whether a decision formed is a conclusion drawn from accumulated evidence against a threshold (section 3), and no amount of connection between artifacts produces it. **A graph edge is not a typed transition.** Linking a March conversation to a June one says they concern the same subject. It does not say whether June revised March, reversed it, drifted onto

a different question, or left it to lapse, and those four have different current answers (section 4). Relatedness is symmetric and untyped. Supersession is neither. **Retrieval returns what exists.** A memory has no representation of nothing was decided, and richer memory makes this worse rather than better, because a larger store offers more material from which to assemble a plausible answer (section 5). **Memory has no completeness obligation.** It grows as things are said. A record has to be driven toward completeness, which is a loop that goes out and asks rather than a store that accumulates.

The last of these is where the two readers separate. Exposing a memory to an agent over a protocol gives that agent more context, and more context is genuinely useful. It does not give the agent grounds to stop. An agent reading a memory graph receives the best available material and proceeds on it, because a store has no threshold and so has nothing to refuse with. The premise check an agent needs is not a richer answer. It is the ability to be told that there is no answer.

None of this makes the two approaches competitors in the ordinary sense. A memory platform is a better source to read from than raw transcripts, and a record built on top of one would be better for it. But an organizational memory answers what the company said. The resolved half needs an answer to what is true now, and those are not the same question however completely the first one is answered.

### 7.10 What none of them holds

| Approach | The question it answers well | The property it cannot produce |
|---|---|---|
| The human way | which decisions matter, and who to ask | capture of decisions nobody noticed; survival past one person's tenure |
| Transcript extraction | what happened in one meeting | subject identity across meetings, so no recurrence and no supersession |
| Retrieval over the corpus | where a subject was discussed | determinism, and refusal instead of construction |
| Authored semantic layers | governed facts somebody defined | facts nobody authored, because none was there to author |
| Fact validity in agent memory | what was true when, and what the system believed when | drift and abandonment, which produce no contradiction |
| Automation platforms | executing a settled rule reliably | the decision itself, and any signal when nothing happened |
| Context in prompts | stable policy and constraint | the current answer, which has to be known before it can be written |
| A general-purpose assistant with the corpus loaded | investigating one question over material one person assembled | a stored answer, a known observation window, and any signal from a decision that was quietly dropped |
| The organizational memory platform | connecting what was said to the documents and records it touches, under source permissions | a formed decision, a typed transition, an answer of nothing, and any reason for an agent to stop |

Read down the last column and the shape of the gap is consistent. Every approach in the list either requires that a decision was already recognized and written down, or requires a contradiction to notice a change, or cannot say that nothing was decided. Those three requirements are the same requirement seen from three angles: each one assumes the resolved half has already been resolved by something else.

Provenance (section 6) has no row of its own, and the reason is worth stating rather than leaving as an omission. It is not a property most of these fail to produce so much as one they have no object to attach it to. Fact validity in agent memory holds two of the three clocks and comes closest. Retrieval and the general-purpose assistant cite the passages they read, which establishes where an answer came from and not why nothing since has changed it. The rest have nothing to cite, because nothing was formed.

## 7.11 Why the gap persists

Everything named in that table is buildable, and it is worth saying so directly.

Persistent subject objects, typed transitions, materialized records read from storage rather than regenerated, bi-temporal history, coverage manifests attached to negative answers, and a structured refusal result are all established engineering. Nothing in this paper argues that the approaches above are technically barred from holding the resolved half. Any of them could build these.

The question is what building them costs, and the cost is not engineering.

**Coverage cannot be sold the way retrieval is sold.** A retrieval product is complete for one user on the first day. One person connects one source, asks one question, and gets value, which is why products of that kind spread from the bottom of an organization upward and why that distribution is a strength rather than a compromise. A decision record has no such property. Partial coverage of a record is not a smaller record. It is a record whose negative answers are wrong, because a decision missing from the observed surfaces may simply have happened on a surface nobody connected. The gaps concentrate exactly where a per-seat rollout leaves them, which is in the teams that did not adopt. This is a fact about how a product reaches an organization rather than how it is built, and engineering effort does not touch it.

The requirement is narrower than whole-company coverage, and the difference matters to anyone planning a first deployment. What has to be complete is the set of surfaces a given subject actually lives on, not the set of surfaces the company owns. A record covering purchasing, finance, and the two channels where vendor decisions get made can answer vendor questions authoritatively while knowing nothing whatever about the product roadmap, provided it says which it is. What it cannot do is answer partially and quietly. So coverage is per subject rather than per seat, and a first deployment is a domain rather than a company. That is a harder sale than a seat and a considerably smaller one than an enterprise, which is the shape the distribution problem actually has.

**Refusal inverts the measure of a search product.** A retrieval product is scored on answers returned and questions deflected. A record is scored on answers correctly withheld. Making an answer of nothing a headline property asks a product organization to promote a number that reads as failure on its own dashboards. The implementation is trivial. The adoption is not, and barriers of that kind tend to outlast technical ones.

**Permission fidelity and a shared answer pull in opposite directions.** Permission-aware retrieval is a genuine strength of the mature platforms, and here it becomes an obstacle. Scoping results to the reader is correct behavior for search: two people with different access should see different results. A record requires the opposite, one current answer that does not vary by who is asking, or two teams act on different versions of the same decision and the record is not authoritative. Reconciling the two is possible, and it means splitting the answer from the evidence behind it, so that everyone sees the same state and what they

can inspect underneath it differs. That is a change to the core contract of a permission-scoped product rather than a feature added on top of one.

Splitting answer from evidence is not sufficient on its own, and the paper should be precise about how far the claim reaches. Some decisions are restricted in their state and not only in their sourcing. A personnel action, an acquisition under discussion, an investigation, a matter under legal hold: for these, the fact that a decision exists is itself the protected thing, and a record reporting one universally visible answer has leaked before anyone opens the evidence. So the requirement is narrower than one answer for the whole company. It is one answer within an access domain, with domains that do not leak into each other and a decision that is invisible outside its own. That still defeats the failure this section is about, because everyone who can see a decision at all sees the same state of it, which is exactly what permission-scoped retrieval cannot promise. It means the record is canonical within a boundary rather than everywhere, and those boundaries have to be modeled deliberately rather than inherited from whichever source system the evidence happened to come from.

**A loop that writes back accepts a different kind of risk.** A read surface that is wrong returns a poor answer to somebody who asked for one. A loop that is wrong puts a false claim about undone work in front of a named person, in the channel where they work, with their name against it. The first is a quality problem. The second is a political incident, and it is a large part of why passive products stay passive. Accepting that exposure is decided once, near the top of a company, and not by the team that would implement it.

**The properties constrain each other, so they cannot be staged independently.** This is the part most easily missed when the four capabilities are read as a feature list, and it is where the cost actually sits. Determinism at read time is trivial: materialize the record and serve from storage rather than regenerating. Determinism under late-arriving evidence is not, because the record has to re-derive when new evidence lands (section 6), which means the same subject can resolve differently on Tuesday than it did on Monday for reasons that are correct. Distinguishing that correction from a real change requires the belief clock, which requires the transitions to be typed, which requires a classifier whose errors are silent: a drift labeled as a revision produces a confident current answer where an open question exists. Refusal then requires a calibrated threshold rather than an abstention rule, because a system that declines often enough is safe and useless, and the calibration point cannot be found without knowing which decisions actually formed. And absence requires coverage at the level of the subject rather than the connector. Knowing that a message system was indexed completely does not establish that the supplier decision would have surfaced there, which is the expectation problem again, and it is the one that resists a schema.

Each of those is buildable. None of them is separable. A team can ship any one in a quarter and will find that shipping it usefully depends on three others, which is why an honest plan for this work looks like a staged program with gates rather than a feature added to a roadmap.

**Some of the calibration does not transfer between companies.** Most of the work does. The architecture generalizes, and so does nearly everything sitting directly on it: the transition taxonomy, the threshold mechanism, the coverage model, the provenance chain, the detection methods for drift and abandonment. What does not generalize is the tuning at the very top. Two companies decide the same class of question in different systems, with different processes, using different words, under different norms about what counts as agreement and who is entitled to object. The map from a decision to the artifacts it should produce reflects how one company works, and the language of assent reflects the people in it. A threshold calibrated against one organization's evidence may be miscalibrated against the next one's, and that error would be silent.

How much recurs per customer is an open question rather than a settled one, and 9.6 states it as such. If the map is entirely company-specific, every deployment starts cold and a meaningful share of the difficulty is paid again each time, which is a different business from the one software usually is. If processes rhyme across companies of similar shape and sector, most of it is paid once. This has not been established either way, and it is the single largest open question in the economics of the work. Anyone claiming to know the answer today, in either direction, is guessing.

Either way, the accumulated resolution is what makes the record defensible once it exists. Whatever variance a deployment does carry is what makes its history specific to that company and unavailable to anyone starting fresh against it (9.5). The difficulty and the durability are the same property seen from two sides.

**And every one of them has to be calibrated against ground truth that does not exist yet.** There is no public corpus of organizational decisions with known outcomes. It cannot be scraped, because the artifacts of a decision are scattered across private systems by definition, and it cannot be generated, because a synthetic decision has none of the ambiguity that makes the real ones hard. The only source is a company that already knows what happened, willing to say so afterwards. That constraint applies equally to everyone working on this, including the approach described in the next section, and it is the reason this paper proposes a test (9.7) rather than reporting a benchmark.

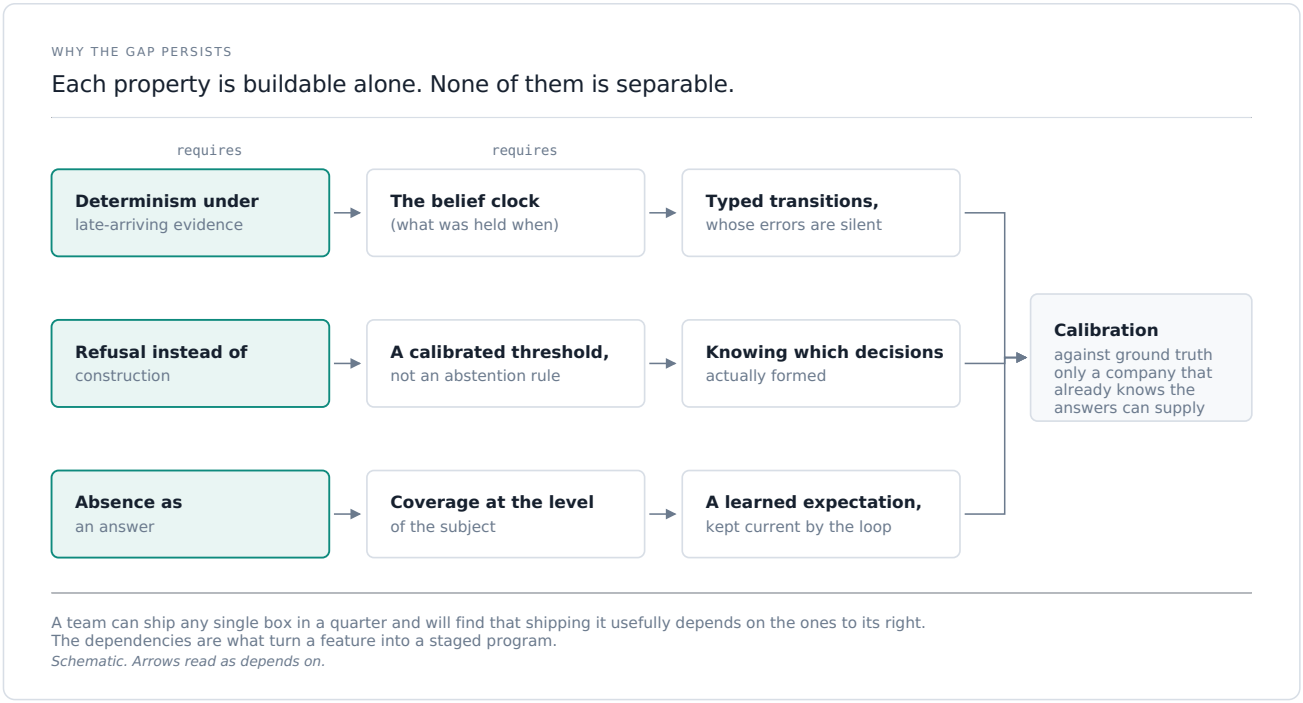



Figure 8. Each property on the left is buildable on its own. Reading rightward gives what it depends on, and all three rows arrive at the same place.

None of this says the approaches above will never hold the resolved half. It says that holding it means becoming a different product, reaching the organization a different way, sold to a different buyer, with a different definition of a good answer. A position is not defended by how hard the engineering is. It is defended by how much of what already works has to be given up to get there.

A correctly assembled stack needs every layer above. It also needs a layer that none of them is.

---

## 8. The approach: reconciliation as a resolution loop

The previous sections describe a set of requirements. A record of the resolved half has to hold a subject across time rather than a sentence in a meeting. It has to form a decision from accumulated evidence rather than from decision-shaped language. It has to distinguish four transitions rather than one. It has to assert absence as an answer rather than produce something plausible. It has to know that evidence scattered across systems that share no key concerns the same subject. And it has to do all of that against an expectation that keeps moving.

What follows is the approach Rithmo takes to those requirements, described as a set of properties rather than as a product tour. Two notes on how to read it. This paper publishes the properties and withholds the parameters: it states that

formation runs on a calibrated threshold without stating the threshold, and states that evidence is weighted by source without stating how the weighting is derived. And where something is a design rather than a running behavior, the text says so plainly.

### 8.1 The unit is a subject, not an utterance

The primary object in the record is the question, not the sentence. "Which supplier are we using" is one subject that exists continuously from March through August. Decisions attach to that subject and carry state. Evidence attaches to the subject as it arrives, in any month, from any surface the company works on: meeting capture, Slack and Teams threads, mail, tickets, documents, and the records left behind in purchasing, CRM, and project systems.

This one choice is what makes most of the earlier sections expressible at all. Recurrence (5.2) requires that five conversations be five instances of one thing, which needs the thing to exist. Supersession (section 4) is a relationship between decisions on a shared subject, so without the shared subject there is nothing for a relationship to hold between. This is the structural difference from the extraction approach in 7.2, where each meeting is a document and the subject exists only inside it.

The two objects have different lifespans, and keeping them apart settles a question the loop would otherwise leave open. A decision reaches a terminal state and stays there. Once revised it is outdated, once reversed it is repudiated, once abandoned it is lapsed, and none of those is a live entry waiting on anything. What persists is the subject, which holds its decisions in order and may have no current answer at all.

So a question that resurfaces two years later does not reopen the old decision. It forms a new one on the same subject, and the old one keeps its state and its provenance. Whether the two are linked is itself a claim that requires evidence. If the later conversation refers back to the earlier decision, that reference is the link and the chain holds. If nobody refers back and the subject merely resembles the older one, there is no chain, because inferring a continuation from resemblance is the same construction the record exists to refuse. This is also why the chain terminates. Every decision a company makes is in some sense a continuation of an older one, and the record holds only the continuations somebody actually drew.

### 8.2 Subject identity is the hardest problem in the record

Everything else depends on knowing that two pieces of evidence concern the same thing. Formation accumulates evidence against a subject. Supersession is a relationship between decisions on a shared subject. Recurrence is a count of instances of one subject. Absence is a claim about a subject with nothing attached to it. Get subject identity wrong and all four are wrong downstream, in a way that

looks correct from the outside.

The difficulty is not only that the systems lack a common identifier, though they do. It is that a subject is a question rather than an entity, and questions carry no attributes to compare. Record linkage assumes two records describe the same object and can be scored on shared fields. The same supplier question appears in a transcript as an informal reference with no vendor named at all, in a purchasing system as a vendor identifier with no question attached, and in a message thread as neither. There is no field in common because there is no field.

Three properties make this harder than it first looks.

**Granularity has no natural level.** Which supplier the company is using, and how it handles supplier transitions in one region, may be one subject or two, and the evidence does not say which. Draw subjects too coarsely and every conversation reads as recurrence on one enormous unresolved question. Draw them too finely and nothing ever recurs, because each discussion becomes its own subject with a single instance.

**Boundaries move after the fact.** Two subjects tracked separately for months turn out to have been one. One subject turns out to have been two the whole time. Both corrections are retroactive and both invalidate attachments already made: decisions that superseded each other under one boundary do not under the other, and a subject that had no current answer acquires one. A record that cannot merge and split subjects after the fact holds its first mistake permanently.

**The errors are not symmetric, and the design leans on that.** A false merge fabricates a relationship between unrelated decisions and produces a confidently wrong current answer, which is the one failure the record exists to prevent. A false split produces two subjects, each with a correct current answer and neither aware of the other, which is a miss. Given refusal, a miss is safe and a fabrication is not, so resolution biases toward splitting and treats a merge as a claim that requires evidence. That is the same rule that decides whether a decision two years later continues an older one (8.1), applied one level down.

Subject boundaries are therefore correctable by the people who own the subject, and a correction propagates backward through everything attached rather than applying only from that point forward. This is the part of the record most dependent on human correction, which is consistent with a record built to be argued with (6.3) rather than trusted blindly.

How resolution is actually performed across surfaces is a parameter of the approach rather than a property of it, and this paper does not describe it.

## 8.3 Formation is a threshold on accumulated evidence

A decision does not form because someone said a decision-shaped sentence.

Evidence accumulates against the subject, each piece weighted according to how much it should count given its source, and a decision forms when the accumulated evidence clears a threshold.

Weighting by source is what formation requires (section 3). The habitual agreer's "sounds great" and the terse colleague's "okay, fine" cannot count the same, because they do not mean the same.

Below the threshold, the output is not a low-confidence decision. It is explicitly no decision, held as a candidate, and the gap is available as a finding in its own right. The enterprise pricing question is the case, the one where nothing was decided at all (5.1): three meetings of real discussion, a spreadsheet, no approval. A system that lowers its bar to produce an answer there has produced the exact failure the record exists to prevent. Holding the candidate open is the correct output, and it is also the more useful one, because an unresolved pricing question that somebody can now see is a question that can be closed.

### 8.4 Resolution runs in more than one direction

The obvious shape for this work is a pipeline. Read the conversations, form the decisions, then check the systems for the artifacts each decision should have produced. That direction is real and it is the one most of this paper has described, but it is only one of the directions the record has to work in, and assuming it is the only one makes the problem look far simpler than it is.

**Sometimes the artifact is what surfaces first.** A purchase order is raised against a vendor the record holds no decision for. Six weeks of commits accumulate against an approach that appears nowhere as a settled choice. A renewal processes on schedule with nothing on the decision side accounting for it. In each case the observable is downstream, and the question runs backward: is there a decision behind this, and where is it. That search has three honest outcomes. A decision is found and the two are linked. Evidence of the subject is found without a decision having formed, which makes it a candidate below threshold and a question worth asking somebody. Or nothing is found at all, which is work proceeding with nothing authorizing it (5.4), and it is a finding rather than a failure.

**Sometimes the evidence surfaces in the middle.** A mail that says the team is proceeding as discussed is neither a decision nor an artifact. It is evidence that a decision happened somewhere the record has not yet looked, and it anchors a search that runs outward in both directions at once: backward for the conversation that settled it, forward for whatever it was supposed to produce. Most real evidence is of this kind. Clean decisions and clean artifacts are the minority.

This is why the record is not a pipeline with an input and an output. It is a partially observed graph that gets assembled from whichever fragment happened to be

seen first, and the order of arrival has nothing to do with the order of events (section 6). The practical consequence is that every new piece of evidence is a potential anchor for a search in either direction, against a subject that may not exist yet, across systems that share no key (8.2).

**And resolution runs vertically as well.** Decisions are not peers (4.5), so evidence can also anchor a search upward, for the decision a piece of work was made in service of, or downward, for the decisions made in service of this one. The upward search is what distinguishes work that nothing authorizes from work whose authorization was withdrawn somewhere above it, and those are different findings addressed to different people. The downward search is what makes a reversal at the top actionable, since the useful output of a strategy being abandoned is the list of decisions still live underneath it. Both run on inferred subordination rather than a declared hierarchy, which makes them the least mature part of this description: the record holds the relation and the resolution of it across surfaces is a design rather than a settled behavior.

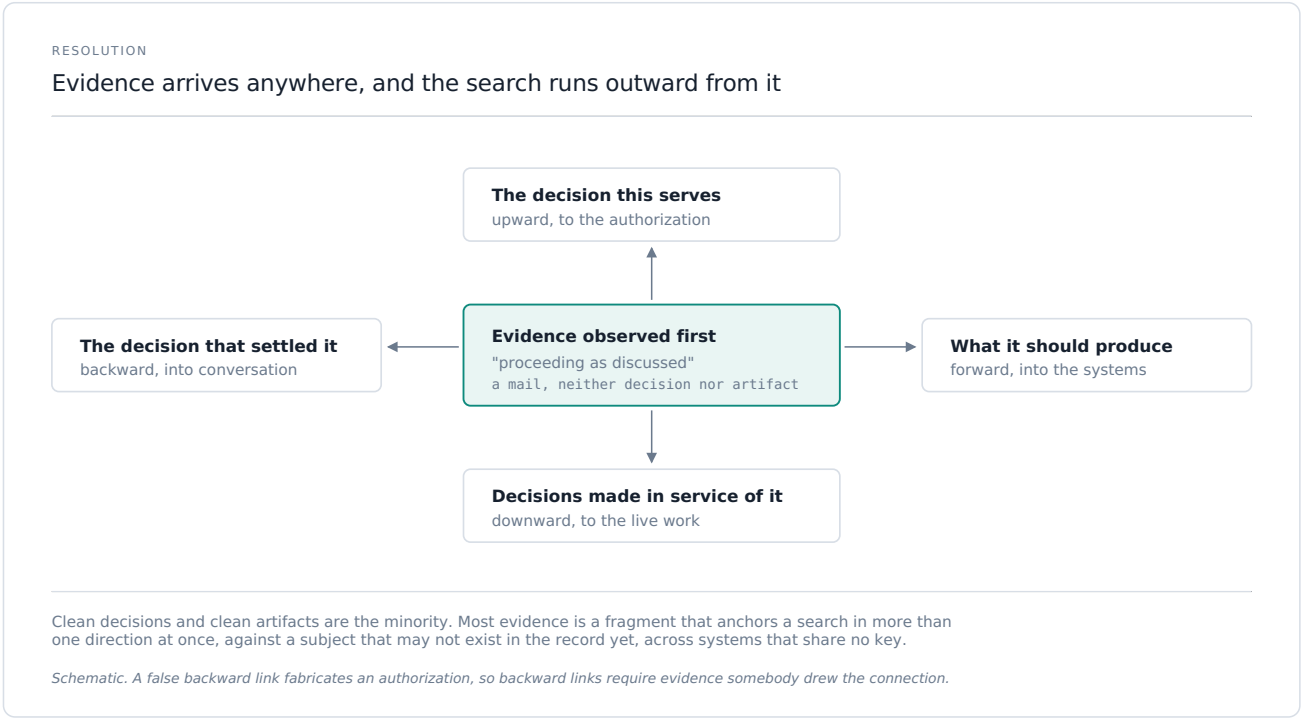



Figure 9. The record is assembled from whichever fragment happened to be seen first, and the order of arrival has nothing to do with the order of events.

The backward direction carries a specific hazard that shapes how it is handled. A false forward link produces a gap: the record expected an artifact that was not required, and the cost is a wasted question. A false backward link produces an authorization: the record reports that a piece of work was decided when it was not, which is exactly the confident fabrication the whole design exists to prevent. So a backward link is held to the same rule as a subject merge. It requires

evidence that somebody actually connected the two, not a resemblance strong enough to be convincing.

### 8.5 State is a typed transition, not a replacement pointer

Section 4 argues that four transitions hide inside the word supersession and that collapsing them loses two live open questions. The approach follows from that directly: the record stores which transition occurred, not merely that a later decision exists.

From a typed transition, both things a reader needs are derivable. The current answer, and the status of the earlier decision. A revision leaves the earlier decision outdated and its history useful. A reversal leaves it repudiated and the question closed. Drift leaves it open and unresolved while the later decision stands on its own subject. Abandonment leaves it lapsed, with the question still live and recoverable the moment somebody knows it is there.

Typed transitions are the design of the record's state model. Distinguishing drift and abandonment reliably at scale is the hardest part of it, and after subject identity (8.2) the least settled thing described here, because neither produces a contradicting assertion for a system to catch, which is the same reason fact validity in agent memory cannot see them (7.5). Drift is detectable in principle from the divergence between a subject's evidence and the subject it started as. Abandonment is detectable in principle from evidence that stops arriving against an expectation that it would continue. Both are inferences over a moving expectation rather than reactions to an event, and both are areas of active work rather than settled behavior.

### 8.6 Absence is a first-class answer

A query against the record returns the current sourced record, or it returns a typed result stating that no decision exists. It does not return a construction assembled from adjacent material. This is a current property of the read interface, and it is the one that matters most for anything acting on the record, because it is the difference between an agent that stops and an agent that proceeds on a guess.

Refusal rests on the formation threshold (8.3). A system with no threshold has no principled basis for saying that nothing was decided, so it will always be able to produce something. A system with one can distinguish the four varieties of absence (section 5): nothing decided, raised repeatedly and unresolved, decided but never landed, and work proceeding with nothing authorizing it.

The last two are not stored, for the reason 5.3 gives: a gap is a conclusion assembled at the moment of asking rather than a state that can be written down and left there. So the approach treats those as queries over expectation and observed presence, which means a gap exists for exactly as long as it is still true.

Coverage belongs to the same property, and it comes down to one rule: silence only means something if you were watching. A gap reported in a system nobody has been reading says more about the blind spot than about the company, so the record has to keep track of what it has actually been able to see. Every miss then becomes a safe miss. The record reads the surfaces where decisions are made on the record, and where it has not been watching it says so rather than drawing a conclusion from the quiet.

Coverage is published rather than assumed. A record that tracks its blind spots only internally is asking to be trusted, which is the wrong posture for a system whose most valuable answers are negative ones. So an absence answer carries the surfaces that were read, the period they were read over, and the surfaces that were not, and the record as a whole can state what it has been observing and since when. A reader who doubts a negative answer can then doubt something specific: that the window was too short, or that the system nobody connected is where the conversation actually happened. This is the arguable provenance chain (6.3) applied to the answers that have no evidence behind them by definition.

### 8.7 Expectation is learned, not declared

Absence is a comparison against an expectation: what a decision of this kind should have produced, in which systems, by when. Every finding in 8.6 rests on that comparison, and so does abandonment, which 4.4 defines by a decision quietly ceasing to be pursued and 8.5 detects through evidence failing to arrive against an expectation that it would. The expectation is load-bearing for a large part of the record, and it is the least stable thing in it.

**A declared expectation degrades from the day it is written.** A map stated once is a checklist, accurate on the day someone wrote it and quietly wrong afterward, because the systems change, the owning team changes, and the process changes without anyone restating what a decision of that kind now produces. A checklist being completed looks identical whether or not it is still asking for the right things (7.1). So the expectation cannot be configuration. It has to be derived from what the company does and kept current the same way.

**Expectation has polarity, and one direction is much harder.** Some decisions are proved by artifacts appearing: a contract, a ticket, an updated field. Others are proved by artifacts ceasing, as when invoices from the old supplier stop arriving (2.1). An appearance can be observed directly. A cessation is observed only by watching a stream that was already there and noticing that it did not continue, which requires knowing the stream was expected to continue and over what period. Cessation expectations are where the abandonment cases live, and they are the ones no checklist has ever held.

**Timing is part of the expectation rather than a setting on top of it.** A gap is only a gap after the point at which the artifact should have appeared. Set that

window short and the loop fills with findings about work that is merely in progress. Set it long and the finding arrives after the money is spent, which is the budget review two quarters later that the loop exists to pre-empt. The window differs by decision type and by company, so it is another thing that has to be learned rather than chosen.

**Expectation errors fail differently from subject errors, and the cost lands somewhere else.** A wrong subject boundary can fabricate an answer. A wrong expectation does not touch the current answer at all. It fabricates work. It routes a person to close a gap that was never a gap, in their working channel, with their name against it. A loop that is often wrong in that direction stops being answered, and a maintenance loop nobody answers is a record that stops being true. So expectation is tuned against a different cost function from the rest of the record, one in which a missed gap is cheaper than a false one.

**The loop maintains the expectation as a side effect of maintaining the record.** When a gap is routed to the person who owns it and the answer comes back that no artifact of that kind is required for a decision of that kind, the response is evidence about the expectation rather than about the decision. The same channel that keeps entries complete (8.8) keeps the map current. That is the only arrangement under which a non-stationary map stays accurate without somebody being assigned to maintain it by hand.

Two limits follow, and the paper states them rather than working around them. A record with no history has no expectations, so the earliest period after deployment yields decisions and provenance but few reliable absence findings. And decision types that occur rarely never accumulate enough instances for an expectation to stabilize, which means absence findings are strongest where a company's activity is most repetitive and weakest on the one-off decisions that may matter most.

How the expectation is derived and kept current is a parameter of the approach, and this paper does not describe it.

## 8.8 The loop: the record is never finished

Reconciliation is not a batch process that runs once and produces a record. The record is continuously incomplete, because decisions keep being made in conversations and the artifacts they should produce keep failing to appear. So completeness has to be maintained rather than achieved. What is never finished is the record, not the individual decision. Decisions close (8.1). The record does not, because new subjects keep forming and new evidence keeps arriving against subjects that already exist.

The design is a loop. Incomplete entries are detected, meaning a decision sitting open with no owner, a subject raised repeatedly with nothing resolved, a decision with no trace in the systems that should carry it. Each one is routed to the person

who can resolve it, in the channel they already work in, with the history attached. What comes back completes the entry, and the entry lands current, owned, and sourced.

The supplier decision is the case worth following through. Abandonment (4.4) explains why nobody catches it: no event occurs, and the people who decided it consider it closed, so the person best placed to notice has the least reason to look. The loop addresses precisely that. It does not wait for someone to ask. It surfaces a decision that has sat with no activity against an expectation that there would be some, and it puts that in front of the person who owns it, in the channel they already work in, while the answer is still cheap. The alternative is the budget review two quarters later.

The routing is the mechanism rather than a queue somebody works through. A loop that depends on an analyst reviewing findings and chasing people has added an administrative function rather than removed one, and it will decay at the same rate as the meeting hygiene it was meant to replace (7.1).

This loop is core to the approach rather than an application built on top of it. Its purpose is to keep the record true, which is what makes the record worth reading. It operates on decisions and commitments, and its output is the state of a decision. Today the loop runs on internal decision data and is being hardened; continuous operation across a customer's surfaces is the direction, not a current claim.

### 8.9 The record runs where the data already lives

The decision record is the most sensitive read of an organization that exists, because it holds what the company decided, who owns it, and what it has failed to do. That places a constraint on deployment before it places one on features.

Rithmo is built to run inside the customer's own environment, on-prem or in a cloud they control, reading from the systems they already use, meeting capture, Slack and Teams, mail, ticketing, purchasing, CRM, and project systems, through access they grant narrowly and can revoke. No copy of the decision record is held outside the customer's perimeter. This is a property of how the system is deployed rather than a policy commitment about data handling, which is the distinction that matters to a security review.

### 8.10 The read path is a protocol, not an integration

The record is exposed over MCP, so a client that speaks the protocol can read it without a custom integration being built for it. That is a deliberate constraint on the architecture rather than a convenience. A decision layer owned by one capture tool, one cloud, or one agent platform can only ever serve that platform, and the value of the record is that everything reads the same answer. Neutrality at the read path is what makes that possible.

The same interface serves both readers. A person asking what the company decided about suppliers, and an agent checking whether the decision in front of it is real, owned, and current before it acts, are running the same query against the same record and getting the same answer. That single shared answer is what everything in section 9 depends on.

The server is built and the record is readable over the protocol. Neutrality is a property of that design, and it is demonstrated rather than argued only once a third-party agent stack has read the record without anything being built for it. That is the direction, not a current claim.

---

## 9. What this makes possible

Everything above describes a record. This section is about what having one changes, in the order the changes arrive: questions people can answer today that currently have no answer, then a class of cost that becomes visible for the first time, then what agents can do once something is safe for them to read, then what accumulates.

One point governs the order. The human value does not depend on any agent being deployed anywhere. A company with no agent program at all gets both the unanswerable questions and the visible cost (9.1 and 9.2) from the same record that later holds agents back (9.3). The resolved half is not a bet on how fast agent adoption arrives.

### 9.1 Questions that currently have no answer

Not hard questions. Ordinary ones, of the kind a leader assumes are answerable and discovers are not.

*What did the company decide about suppliers, and does that still hold?* Today this is answered by asking whoever was in the room, and different people give different answers because they were in different rooms at different times. Against the record it returns the current decision, who owns it, what it superseded, and the meeting it came from.

*Which decisions from last quarter never landed anywhere?* Today this cannot be asked at all, because answering it requires knowing both what was decided and what should have appeared as a result, and no system holds the first. Against the record it is the query for decisions that were made and never landed (5.3).

*What has been raised in three or more meetings and never resolved?* The enterprise pricing question, raised repeatedly and never resolved (5.2), and every other one like it. Today each conversation is remembered individually and nobody

holds the count.

*What work is underway that no decision authorizes?* The two engineers six weeks into an approach the team never settled on. Today this surfaces at review, if it surfaces.

*Which decisions have no owner?* Not which tickets have an empty assignee, which is a smaller and different question. Which of the things the company currently holds to be true has nobody accountable for carrying it out. A decision in that state is a decision nobody is going to move.

What makes these answerable rather than debatable is the pair of properties the record is built to hold, determinism and refusal (section 8). The same question returns the same answer, because the answer is a record rather than a synthesis. And where nothing was decided, the answer says so instead of assembling something plausible. A question that returns a different answer each time it is asked has not been answered.

### 9.2 A cost that becomes visible for the first time

The human way leaves its own cost unmeasured (7.1), and not because it is hard to compute. It is unmeasured because computing it requires knowing which decisions were made and which never landed. With the record, two figures come out directly, and neither of them requires a model.

**Recurrence** is time. A question raised across five meetings, with six people present, at forty minutes each, is twenty hours of paid attention spent on a question that did not close. The inputs are a calendar, an attendance list, and the loaded hourly rate the finance team already uses for other purposes. One judgment sits inside that figure and is better named than buried: charging the whole interval to the unresolved question assumes the meeting was about that question and produced nothing else, which is an attribution rather than a measurement. The defensible version charges the share of each meeting the subject actually consumed and reports the total as a bound rather than a bill. The input that was missing is still the first one, which is knowing that those five meetings were the same question.

**Continuation** is cash, and it is the stronger of the two by a distance. A decision to stop paying a vendor that was never carried out produces invoices the company paid after deciding to stop paying them. That figure is in the ledger, exact, already reconciled, and traceable to the month the decision was made. It is not an estimate at all. It is money that left the company after the company decided it should stop.

The two also differ in when they arrive. Recurrence can be computed from the record's own history within the first months of operation, because it needs nothing but the record. Continuation needs the ledger alongside it and a decision old

enough to have been carried out or not, so it comes later and is worth more when it does.

The supplier case is the shape of it. Five months of invoices from a vendor the company decided in March to leave. Nothing about that number requires a model. What stops a company producing it today is not the arithmetic but the one input it has no source for, which is the date the decision was made and the fact that it was never carried out.

### 9.3 Agents can be held back, not merely instructed

Context written into a prompt cannot refuse (7.7). It proceeds on whatever value it was given, with no representation of whether that value is current or whether any decision was ever made. The controls in common use constrain the output or constrain the tools. Where a person approves the action, the premise does get checked, by that person, at the cost of the throughput the agent was deployed for. What no automated control does is check the premise itself.

A record with the refusal property makes a different control available. An agent queries the decision in front of it before acting. If the record returns a current, owned, sourced decision, the agent runs. If it returns that no decision exists, or that the decision is unowned, or that a subject has been raised repeatedly and never resolved, the agent does not proceed and the gap goes into the maintenance loop (8.8) to be resolved first.

The difference from a guardrail is worth stating precisely. A guardrail inspects what the model produced. This inspects whether the thing the model was asked to act on was ever decided. A quoting agent pricing a deal off last quarter's model is not producing bad output, it is producing correct output from a premise nobody checked. That failure is invisible to output inspection and visible to a premise check.

This is a property the protocol read path (8.10) makes available rather than a claim about deployed agent behavior today. What exists now is the record and the interface. What it enables is a control that has not previously been possible.

### 9.4 One record, two readers

The same entry serves a person and an agent for different reasons, and this is the part that is easy to miss.

A leader reads the record to see which decisions stalled, which have no owner, and which never reached the systems that were supposed to carry them. An agent reads the same entry to determine whether the decision in front of it is real before it acts. One is looking for what failed to happen. The other is looking for what is true. Both need the identical property: a current, owned, sourced answer, or an

explicit statement that none exists.

Human readers also differ from each other, and the same record has to serve them at different altitudes (4.5). An executive asking what the company decided about a market wants the decision at that level and the exceptions beneath it that threaten it, not the hundred decisions made in service of it. A team lead wants their own level and the one above, because that is where their authorization comes from. This is one record queried at different heights rather than several records, and it only works because subordination is held as a relation in the record instead of being implied by whichever tool the work happened to live in.

The practical consequence is that these are not two products or two phases. The diagnostic that shows a leader where decisions are stalling and the substrate an agent reads before acting are the same record, built by the same reconciliation. Nothing has to be rebuilt between them.

## 9.5 What accumulates

The record gets more valuable in a way that is specific rather than general, and worth naming precisely.

What accumulates is resolution. Subjects reconciled across systems that share no key. Supersession chains with their transitions typed. Knowledge of which surfaces have been observed well enough that their silence carries information. A record eighteen months in can answer questions a record on its first day cannot, not because it holds more rows but because the history is what makes the current answer defensible. Provenance and time (section 6) is the mechanism by which a current answer can be shown to be current, and it only exists as a function of accumulated history.

That accumulation is also the reason this is difficult to displace. The schema is not the asset and could be copied in a week. The resolved history is the asset, and it can only be produced by having read the organization for as long as it has been read. This is a property of the design realized over time rather than a claim about any deployment today.

The obvious objection is that a competitor could regenerate that history by reprocessing the same raw material. Most of it, they could. The part they could not is the part the loop produced. When a routed gap comes back with the answer that no artifact of that kind is required for a decision of that kind, when a subject boundary is corrected, when a transition type is disputed and changed, those responses are new evidence that existed nowhere in the corpus before the system asked for them. Reprocessing ten years of messages cannot regenerate a sentence a person only produced because something went and asked. The corpus is copyable. The corrections are not, because the system's own operation is what created them.

### 9.6 What is not yet known

The requirements in sections 3 through 6 are stated with more confidence than the answers to them deserve, and the honest position is that several questions in this area are open rather than solved. They are worth naming, because a reader who works them out unaided will reasonably wonder what else went unmentioned.

**How accurately a decision can be formed from evidence, in principle.** The threshold model in 8.3 assumes there is a point at which accumulated evidence settles the question. For many decisions there is. For some there may not be, and the disagreement may be real rather than a failure of measurement: two people in the same room can hold different and defensible views about whether something was decided. Where that is the case, the correct output is a recorded disagreement rather than a resolution, and how much of the total falls into that category is not known.

**Whether subject identity is learnable to the precision required.** 8.2 argues that boundaries have to be correctable after the fact, which is true whether or not resolution works well. What is not established is whether the correction rate settles at a level a company will tolerate, or whether subject identity remains a standing human obligation. The difference between those two outcomes is the difference between a product and a service, and it will not be settled by argument.

**Whether expectation transfers between companies.** 8.7 describes the map from decisions to the artifacts they should produce as learned from what a company actually does. If that map is entirely company-specific, every deployment starts cold and the early period is weak by construction. If parts of it generalize across companies with similar processes, deployments improve for reasons unrelated to the customer's own history. Which of those is true has direct consequences for how quickly the record becomes useful, and it is not yet known.

**Whether four transitions are enough.** Section 4 says at least four transitions hide inside supersession, and the paper then works with exactly four throughout. Four is what the failures observed so far have required, which is not the same as a result. A fifth may be needed for cases nobody has yet separated, most plausibly near the boundary between drift and abandonment, where a subject neither closes nor visibly moves. The taxonomy is a working one and the number is not defended as final.

**Whether the coverage boundary holds in practice.** The record treats a decision that surfaced on no accountable channel as not yet operative (9.7). That convention is what makes every miss a safe miss and what allows absence to be asserted at all, and it is a definition rather than a finding. Some decisions are acted on for months having been settled in a corridor and written down nowhere, and for those the definition is doing work the organization never agreed to. How large that class is, and whether the people relying on the record experience its

exclusion as correct or as a blind spot dressed up as a principle, is not known and will not be settled by argument.

**What refusal rate people will accept.** Refusal is defended throughout this paper as the property that makes the record trustworthy. It is also the property most likely to be experienced as the system not working. A record that declines to answer a third of the questions put to it is behaving correctly and will feel broken. Where the tolerable rate sits, and whether it can be reached without weakening the threshold, is an empirical question nobody has answered.

**Whether the loop's corrections improve the record measurably.** 9.5 argues that responses routed back through the loop are evidence that exists nowhere else and cannot be regenerated from the corpus. That is structurally true. Whether the accumulated corrections improve accuracy enough to be felt, over what period, and with what volume of use, is a claim this paper does not make and could not yet support.

None of these are reasons to wait. They are the shape of the work, and a paper that stated the requirements without stating these would be describing a solved problem rather than a new one.

### 9.7 What it does not do

A paper arguing that nothing holds the resolved half should be equally clear about the edges of the thing it proposes to put there.

It does not decide anything. It holds what people decided, and where they decided nothing it says so. Nothing in the approach recommends a course of action or substitutes judgment for anyone's.

It does not replace the existing approaches surveyed earlier (section 7). It reads from meeting capture, Slack and Teams, mail, ticketing, purchasing, CRM, and project systems, and it serves automation platforms, agent frameworks, and semantic layers. A company that adopts it keeps everything it already runs, because the record is built from what those systems already produce.

It does not read what never surfaced. A decision reached in a hallway and mentioned in no accountable channel is treated as not yet operative, and the record does not pretend to hold it. That is an operating definition rather than an observed fact about how companies work, and 9.6 takes up what it may be costing. Where a surface has not been observed, the correct output is that it has not been observed, not an inference from its silence. Every miss is a safe miss, which is the property that makes the rest of it trustworthy.

It does not link what it did not observe. The chain of decisions on a subject is built from references people actually made, so a decision reached before the record was in place sits outside it. A record deployed today can hold that a subject was settled

in 2024 if somebody says so, with that statement as its evidence, but it cannot reconstruct the 2024 chain from systems it never watched. The record's history begins when it begins observing, and it says so rather than implying otherwise.

The claim in this paper is narrow and, taken on its own terms, testable. Companies operate on facts that became true because people decided them, no system combines the properties needed to hold those facts as a current answer, and the approaches that appear to hold them are answering adjacent questions.

The test that would settle it needs no number this paper could have computed for itself. A company supplies a set of situations from its own history whose outcomes it already knows and does not disclose: a decision that was later revised, one that was reversed, one where the conversation drifted onto a different question, one that lapsed without ever being reversed, a question raised repeatedly and never resolved, a decision that never reached the system that should have carried it, and a piece of work that proceeded with nothing authorizing it. The record is then asked for the current state of each. It should reconstruct the ones that formed, type each transition correctly, show the evidence for both, and state that nothing was decided where nothing was, rather than producing something plausible. Refusing correctly is the harder half, because a system that never refuses can score well on the rest and has established nothing.

A record that cannot do that is not holding the resolved half, whatever else it may be doing well. Specifying the test in advance is the point. A thesis that says how it could be shown wrong is worth more than one supported by numbers its author computed and graded.

What the resolved half needs is a record that forms decisions from evidence, types their transitions, refuses when nothing was decided, and keeps itself current. That is what Rithmo is building.

---

## How to cite

Williams, T. (2026). *The Resolved Half: On formation, supersession, absence, and provenance in organizational records* (Version 1.0). Rithmo. https://doi.org/10.5281/zenodo.21710262

Archived copy of record: 10.5281/zenodo.21710262. Licensed CC BY 4.0.