Five Anchors: Holding Intent Invariant from Discovery to Production in Enterprise AI Delivery

Marius Golombeck
Munich, Germany
kontakt@mariusgolombeck.de
ORCID 0009-0004-5977-9917

***Abstract*—Enterprise AI initiatives fail often, and the reasons given are usually technical. The data was not ready. The model underperformed. The organisation resisted. In this paper I argue for a different and more common cause, calling it *intent drift*: the gap that opens between the outcome a sponsor asked for and the system that eventually ships, widened one reasonable decision at a time. Drift is a failure of traceability rather than of engineering. In response I set out a delivery framework of five anchors. Each anchor names the counterpart whose knowledge is required, the artefact it produces, and the condition under which the work may proceed or must stop. None of the five practices is new. The contribution is the invariant that holds across them, which is that the measurement closing an initiative must resolve back to the ambition that opened it. I formalise that chain, show where the established process models leave it unenforced, and state what each anchor costs when it is skipped. I then derive the break-even condition under which the framework repays its own overhead, which turns out to depend on a single measurable quantity: how often an organisation starts work that cannot succeed. This is a practitioner framework, and I set out its evidential limits in full.**

***Index Terms*—requirements engineering, enterprise architecture, machine learning deployment, traceability, solution architecture, project governance**

I. Introduction

A large share of enterprise AI initiatives never reach sustained production use [1]. Ask why and the answers are familiar: the data was not ready, the model underperformed, the organisation resisted the change, the technology was not mature enough. Each of these is sometimes true. None of them explains the case that motivates this paper, in which an initiative fails even though every decision taken along the way looked defensible at the time it was taken.

How large that share is remains contested, and the widely circulated percentages come from industry surveys rather than peer-reviewed work. Nothing here depends on a particular figure. The survey evidence is enough to establish that deployment is where these initiatives commonly stop, and Section VI shows that the argument holds at rates far below any of the published claims.

Call the pattern *intent drift*. A sponsor states a business outcome. Somebody who was not in that conversation turns it into requirements. Somebody optimising for integration cost turns those requirements into an architecture. The delivery team then evaluates the result against metrics chosen because they are easy to measure rather than because they matter. Every translation moves a short distance from the one before it. Nobody behaves unreasonably at any point. The system that ships meets its specification and misses its purpose.

DOI: 10.13140/RG.2.2.31649.36969

Software engineering has understood drift for a long time. It is why requirements traceability exists as a discipline [2], and why the problem was later elaborated into reference models [3]. Three properties of the enterprise AI setting make drift both likelier and harder to catch. Feasibility is unknown in advance, since whether a model can reach a required accuracy is an empirical question answered only after the money is spent. The acceptance criterion is continuous rather than binary, so some reading of a marginal result will always qualify as success. And these systems couple tightly to their data and their operating environment [4], which hides the real cost of a requirement at the moment somebody accepts it.

Nothing among the five anchors in Section IV will surprise an experienced practitioner, and each has a literature behind it. What this paper offers is the composition: a fixed order, a fixed artefact schema, and an invariant tying the artefacts together so that the ambition recorded at the start remains the thing measured at the end. The established process models supply the stages. They leave out the invariant, and the invariant is where the value sits.

Section II characterises the misalignment that drift grows out of. Section III positions the framework against requirements engineering, gate governance and the machine learning deployment literature. Section IV presents the anchors. Section V formalises the sequencing property. Section VI derives the break-even condition under which the framework pays for the overhead it adds. Section VII works through an archetypal initiative. Section VIII sets out the limitations and the evidence that would be needed to overcome them.

II. The Bilateral Assumption Problem

Enterprise AI engagements typically open with a conversation about what the technology can do rather than about what the organisation needs. The structure is symmetric, and it holds whether the buyer or the vendor makes first contact. A customer asks for something they assume they need, and a provider offers something they assume the customer wants to buy. Neither party is lying. Each reasons from incomplete

information about the other, and both have an incentive to postpone the awkward clarifying question until after the commitment is made. What results is an agreement that both can sign and neither has tested.

What makes this expensive is that it does not happen once. Figure 1 traces the shape. Two independent assumptions meet in an agreement that records neither, the ambiguity propagates downstream where each team resolves it against its own constraints, and the divergence stays invisible until deployment makes it concrete. At that point the standard institutional response is to escalate and re-specify. A new round of workshops produces a longer document, everyone signs it, and the initiative re-enters the loop at exactly the point it entered the first time, with more paper and no better-established intent. Each pass costs more than the last, because it is spent later.

That agreement is the visible problem. It is also a symptom. The conditions producing it sit in the organisation, and the following recur:

- **Communication silos**, where the party who understands the business outcome and the party who understands the data have no routine channel between them.
- **Unclear strategy**, where the initiative stands in for an unstated objective such as demonstrating innovation capability.
- **Unrealistic expectations**, calibrated on vendor demonstrations run under conditions that will never hold in production.
- **Competing priorities**, where the sponsor, the platform owner and the operational user each optimise a different quantity.
- **Organisational ambiguity**, where no single role holds the authority to declare the initiative unjustified.
- **Misunderstood incentives**, where whoever specifies the requirement is not whoever has to live with it.

Fix the symptom and you get a better-written requirements document describing the same wrong system. Fixing the causes takes a delivery process that puts the right counterpart in the room while their knowledge can still change the decision, and that writes down what they said in a form somebody can check later.

That is also where drift begins. An agreement neither party has tested contains no statement of intent precise enough to measure against later, so the first translation has nothing to be faithful to. Everything downstream inherits the ambiguity and resolves it locally, each team in the direction of its own constraints. The anchors attack the problem at this point, before there is anything to be faithful to, rather than trying to recover intent once several translations have already happened.

III. Related Work

A. Requirements engineering

Eliciting, recording and validating requirements is a mature discipline. IEEE 830 [5] and its successor ISO/IEC/IEEE 29148 [6] prescribe the structure and quality attributes of a specification, including verifiability and traceability. This framework does not compete with them. A1 and A3 can be documented under either standard. The limitation that matters here is narrower: a specification standard governs the quality of the artefact and says nothing about the survival of the intent that produced it. A specification can be complete, unambiguous and verifiable while describing a system that has stopped serving the outcome it was written for.

B. Traceability

Gotel and Finkelstein [2] identified requirements traceability as a problem in its own right and separated two halves of it. Pre-requirements traceability links a requirement back to the need that produced it; post-requirements traceability links it forward into design and implementation. They found the first half systematically under-supported. Ramesh and Jarke [3] later derived reference models from practice. The invariant in Section V is a narrow and enforceable special case of pre-requirements traceability, restricted to a single chain and ending in a measurement obligation instead of a document.

C. Phase and gate models

Stage-gate governance [7] and the spiral model [8] both structure delivery as a sequence of checkpoints at which continuation is decided. CRISP-DM [9] supplies the data-mining equivalent and remains the most widely taught process model for analytics work. These models establish that gates should exist. What they underdetermine is the content of the gate criterion. In practice a gate is passed once the phase deliverable is complete, which measures activity and not continued justification.

Iterative and agile methods sit differently rather than in opposition. A definition of done governs whether an increment is finished; it says nothing about whether the increment serves the outcome that funded the work, and the shorter the iteration the more often that question goes unasked. The anchors are compatible with iterative delivery and constrain something it leaves open, which is the subject of Section V.

D. Machine learning in production

A body of work now documents why machine learning systems are hard to operate. Sculley et al. [4] characterise the hidden technical debt that arises from entanglement and undeclared consumers. Breck et al. [10] propose a production readiness rubric. Amershi et al. [11] report on the engineering practices of teams building such systems, and Paleyes et al. [1] survey deployment case studies across sectors. Almost all of this concerns the interval after a system has been specified. Davenport and Ronanki [12] take up the selection question from the management side without prescribing an enforceable mechanism. The gap addressed here lies between the two: the interval in which an ambition becomes a specification, and the obligation to carry that ambition through to the measurement that closes the initiative.

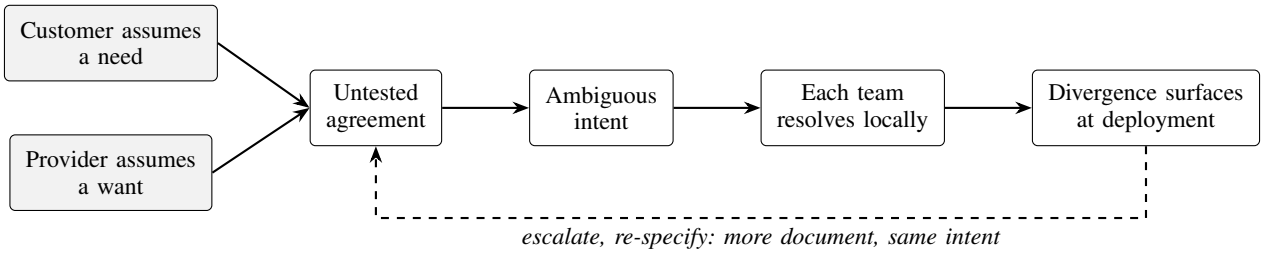

Fig. 1. The bilateral assumption loop. Both parties arrive holding an assumption about the other, and the agreement they sign records neither. Downstream, each team resolves the resulting ambiguity in the direction of its own constraints, so divergence stays invisible until deployment. The customary remedy is to escalate and re-specify, which returns the initiative to the same untested agreement with more documentation attached and the intent no better established. The anchors of Section IV intervene at the entry to the loop rather than at the escalation.

E. Value measurement

The balanced scorecard [13] established the practice of tying initiatives to measures a finance function recognises. A5 takes that position and adds a sequencing constraint. The measure has to be derived from the ambition recorded in A1, not chosen independently once somebody sits down to write the business case.

IV. The Five Anchors

Five elements define each anchor: the question it answers, the counterpart whose knowledge is required, the artefact it produces, the exit criterion that permits progression, and the characteristic failure when it is skipped. Table I gives the schema and the subsections elaborate.

A. A1*: Separate ambition from requirements*

The sponsor describes an outcome. Not a system. That distinction is the whole point of the anchor, and it collapses constantly. A sponsor who says "we need a forecasting model" has named a system and buried the outcome, which might have been fewer write-offs, better service levels, or the ability to defend an inventory position in front of a board. Each of those admits different solutions, and several of them involve no model at all.

The artefact is a short ambition statement in the sponsor's own vocabulary, recorded word for word. That matters more than it sounds: paraphrase is the first increment of drift.

Skipping it: the initiative optimises a system property that nobody outside the delivery team values.

B. A2*: Establish what the data will bear*

Data owners and IT operations answer which of the inputs the ambition assumes actually exist, at what quality, at what latency and under what access constraints. One rule governs this anchor: a conditional is not evidence. "We could extract that" is a different claim from "that is extracted," and the distance between them is frequently an entire project.

A good share of initiatives ought to die here, and here is where dying is cheap. Discovering a missing input at A2 costs a meeting. Discovering it after A4 costs a programme.

Skipping it: an architecture gets designed around inputs that were promised rather than present, and the schedule quietly absorbs an unplanned data engineering programme.

C. A3*: Agree what "good enough" means*

Two parties sign the acceptance threshold. The end users who will be measured against the system's output establish what performance is useful. The risk, compliance or safety functions empowered to veto deployment establish what performance is permissible. A threshold carrying only one signature is not a threshold.

All of this happens *before* anything is built. Agree a threshold after the results are in and what you have is a rationalisation, because model performance is continuous and a defensible rationalisation is therefore always available. Earlier work on requirements specification under IEEE 830 for a safety-relevant warning system [14] shows the discipline of fixing acceptance conditions ahead of design.

Skipping it: metric substitution, where the number reported at the end of the initiative is whichever one the system managed to hit.

D. A4*: Design the smallest sufficient architecture*

Enterprise architecture and security settle four questions: the integration surface, the data residency position, the failure modes, and the operational cost at steady state. The word *smallest* is load-bearing. The target is the least architecture that satisfies the threshold fixed in A3, and anything past that minimum is a permanent operational liability bought in exchange for a capability nobody agreed to measure.

Regulated and safety-relevant settings inherit their constraints instead of negotiating them. Data residency and lawfulness of processing follow from the applicable data protection regime [15]. In embedded and automotive contexts, sector standards prescribe the security architecture [16].

Skipping it: a system that hits its accuracy threshold and cannot be operated, integrated, or lawfully run where it is needed.

E. A5*: Validate the economics*

The sponsor and the finance function test whether measurable economic value justifies the initiative. The exit criterion is deliberately hard: the business case has to hold under assumptions somebody in finance will defend with the delivery team out of the room. A case that needs its author present to survive scrutiny is not a case.

TABLE I
THE FIVE ANCHORS, THEIR COUNTERPARTS, ARTEFACTS AND EXIT CRITERIA

	Anchor	Counterpart	Artefact	Exit criterion
A1	Separate ambition from requirements	Business sponsor	Ambition statement	An outcome is stated in business terms, with no system implied
A2	Establish what the data will bear	Data owners, IT operations	Data feasibility record	Every required input is observed to exist, not asserted to
A3	Agree what "good enough" means	End users, champions, risk and compliance	Acceptance threshold	A threshold is signed by both the measured and the empowered-to-veto
A4	Design the smallest sufficient architecture	Enterprise architecture, security	Reference architecture	Integration surface, data residency and failure modes are settled
A5	Validate the economics	Sponsor, finance	Business case	The case holds under assumptions finance will defend unaided

This anchor exists to head off the most common terminal state of enterprise AI work, where a technically successful system never scales because nobody can say what it returns.

Skipping it: the initiative ends as a demonstration and gets dropped at the next budget cycle.

F. What the five do not yet amount to

Run in isolation, these are five good meetings. An organisation can hold all five, produce all five artefacts, and still deliver a system that answers a question nobody asked, because nothing so far requires the artefacts to have anything to do with each other. A3 can fix a threshold that has drifted from the ambition in A1, and every individual anchor will still have passed. What turns five meetings into a framework is the relation between their outputs, and that is what the next section states.

V. THE SEQUENCING PROPERTY

Section IV describes five practices. None is new. This section states the property that makes them a framework instead of a checklist. What follows is a bookkeeping formalism rather than a theory, and it earns its place by doing two things prose cannot: separating two failure modes that ordinary language runs together, and showing that the property can be checked one link at a time instead of audited at the end.

A. Artefacts and justification

Treat each artefact as a finite set of claims. The ambition statement α is a set of desired outcomes, the data feasibility record δ a set of findings about available inputs, the acceptance threshold τ a set of conditions, the architecture σ a set of design commitments, and the business case β a set of value assertions. Write $X_1, \ldots, X_5$ for $\alpha, \delta, \tau, \sigma, \beta$ and μ for the measurement taken after deployment.

μ is not a sixth anchor. It is A1 executed a second time, with the same counterpart: the sponsor who stated the ambition returns to say whether it happened, and the finance function that signed the business case at A5 confirms the number. Naming this explicitly matters, because an obligation with no owner is an obligation nobody discharges, and a measurement that no named person is required to produce is the easiest artefact in the chain to skip. Its exit criterion is the invariant itself, stated below.

Definition 1 (Justification map)**.** A *justification map* for the link $X_i \to X_{i+1}$ is a function

$$j_i : X_{i+1} \longrightarrow \mathcal{P}(X_i) \tag{1}$$

assigning to each element of the successor the set of predecessor elements that support it.

Definition 2 (Local conformance)**.** Link i is *conformant* when $j_i(x) \neq \emptyset$ for every $x \in X_{i+1}$. Elements with $j_i(x) = \emptyset$ are *orphans*, written

$$O_i = \{\, x \in X_{i+1} : j_i(x) = \emptyset \,\}. \tag{2}$$

An orphan is a commitment nobody can trace to a reason. A requirement that entered because somebody senior asked for it, an architectural component added for generality, a metric chosen because it was easy to compute: each is an orphan at its own link.

B. Soundness and coverage

Let j^* denote the transitive composition $j_1 \circ \cdots \circ j_5$, mapping an element of μ to the elements of α that ultimately support it.

Proposition 1 (Local checks suffice)**.** *If every link is conformant, then $j^*(m) \neq \emptyset$ for all $m \in \mu$.*

Proof. Composition of total relations is total. If $j_i(x) \neq \emptyset$ for all $x \in X_{i+1}$ and all i, then by induction on the length of the chain every $m \in \mu$ has a non-empty preimage in $X_1 = \alpha$. □

Proposition 1 is trivial as mathematics and load-bearing as practice. Global traceability never has to be audited as a whole. It follows from five local checks, each performed at the anchor where the relevant counterpart is already in the room. That is what makes the invariant enforceable rather than aspirational.

Conformance alone is not sufficient, and the reason is worth stating precisely.

Definition 3 (Coverage)**.** *μ covers α* when

$$\forall\, a \in \alpha \;\; \exists\, m \in \mu : a \in j^*(m). \tag{3}$$

Proposition 2 (Independence)**.** *Conformance and coverage are independent. Neither implies the other.*

Proof. For conformance without coverage, take $\alpha = \{a_1, a_2\}$ and a chain in which every artefact derives solely from a_1. Every link is conformant, yet $a_2 \notin j^*(m)$ for all m, so coverage fails. For coverage without conformance, extend any covering chain with one additional element $x \in \sigma$ having $j_4(x) = \emptyset$. Coverage is unaffected and conformance fails. □

The two halves name two different project failures. Failed conformance is scope creep: work in the system that nothing asked for. Failed coverage is quiet de-scoping: part of the ambition that fell out along the way and will be absent from the closing report, usually without anyone deciding to drop it. Table II shows that no established process model requires either.

Definition 4 (Drift)**.** The *drift* of a chain is the total orphan count

$$D = \sum_{i=1}^{5} |O_i| \;+\; \big|\{\, a \in \alpha : \nexists\, m \in \mu,\; a \in j^*(m) \,\}\big|, \tag{4}$$

so that $D = 0$ exactly when the chain is both conformant and covering.

D is an integer available during delivery, not a retrospective judgement. Each term is countable at the anchor that produces it.

None of this is exotic, which makes its absence from the established process models the more striking. Table II sets out the comparison on three questions. Does the model define stages? Nearly all do. Does it name the counterpart who has to sign each stage off? None does, which leaves the gate to be passed by whoever is already in the room. Does it require that the measurement closing the work resolve to the ambition that opened it? None does, which is the gap this paper is about.

C. Rejection as a first-class outcome

The invariant gets misread as a promise to build whatever the sponsor first said. It does the opposite, and Definition 2 says why.

Corollary 1 (Local decidability of rejection)**.** *Whether a proposed element x may enter X_{i+1} is decided by evaluating $j_i(x)$ at link i alone, without reference to any later anchor.*

Proof. Immediate from Definition 2: admissibility of x is the condition $j_i(x) \neq \emptyset$, which mentions only X_i and x. □

The practical content of Corollary 1 is that refusal needs no escalation and no forecast. The question is not whether the requested item is a good idea, which is arguable, but whether anyone can name the element of the predecessor artefact that it serves, which is checkable in the meeting. Without a fixed α the check is unavailable, and any addition can be justified against some locally plausible objective.

Two behaviours look similar from outside and are worth separating. *Fold-and-accept* admits scope because the requester has standing. *Anchored steering* admits scope that demonstrably serves α and refuses the rest. The framework exists to make the second possible. Watching work actually get refused is the clearest sign that the invariant is being enforced rather than merely written down.

D. Iteration and re-entry

The chain is drawn as a line, and delivery is not one. Thresholds get revised when a model turns out better or worse than expected, architectures get revisited when an integration proves harder than scoped, and data that was absent at A2 sometimes arrives later. Nothing in the formalism forbids this, and it is worth saying why.

Revision is re-entry at the anchor that owns the artefact, not an exception to the framework. Changing τ means returning to A3 with the counterparts who signed it, which in turn invalidates the justification maps of every artefact downstream: j_3, j_4 and j_5 must be re-established, because elements of σ justified by the old threshold may have no support under the new one. That cascade is the point rather than an inconvenience. An architecture component that survives a threshold revision unexamined is precisely an orphan in the sense of Definition 2, and the cost of re-establishing the maps is what makes late revision visibly expensive instead of invisibly so.

The one move the framework does forbid is revising α to match what was built. That is not iteration but retrofit, and it is indistinguishable in the final report from having hit the original target.

E. Why the ordering is not arbitrary

Each anchor consumes what its predecessor produced and is underdetermined without it. A threshold (A3) cannot be set without knowing what the data will support (A2). An architecture (A4) sized before a threshold exists is sized against a guess. A business case (A5) assembled before the architecture is known cannot include the cost of running it. Reordering the anchors does more than shuffle the sequence: it removes the input that makes the later anchor decidable. Building the business case first to secure funding, which is common practice, inverts the dependency completely and produces a number the later anchors can only contradict.

VI. The Economics of Stopping Early

A framework that adds process steps has to justify their cost. No empirical comparison is available here, so what follows is an analytical model with parameters the reader supplies. It yields a break-even condition rather than a savings figure, which is the strongest honest claim the available evidence supports.

TABLE II
COVERAGE OF THE ESTABLISHED PROCESS MODELS

Model	Defines stages	Names counterparts	Terminal invariant
IEEE 830 / 29148 [5], [6]	partial	no	no
Stage-gate [7]	yes	no	no
Spiral [8]	yes	no	no
CRISP-DM [9]	yes	no	no
ML Test Score [10]	no	no	no
Five Anchors	yes	yes	yes

A. Cost escalation

The premise is not new. Boehm [17] established that the cost of correcting a defect rises sharply with the phase in which it surfaces, and Boehm and Basili [18] later put the ratio between a requirements defect caught during requirements and the same defect caught in the field at roughly two orders of magnitude. An initiative built on data that does not exist is a requirements defect of exactly this kind.

Write C for the cost of carrying an initiative through to deployment, and c_i for the cumulative cost at the point anchor i completes, so that $c_1 < \cdots < c_5 < C$. Write k for the overhead of running the anchors themselves: the workshops, the counterpart time, the artefacts.

B. The value of an abandonment option

Each anchor is a cheap test that buys the option to stop before an expensive commitment. Let q be the proportion of initiatives that are non-viable for a reason some anchor can detect, and let c_d be the cumulative cost at the anchor that detects it.

Without the framework a non-viable initiative runs to C and is written off. With it, the same initiative stops at cost c_d. Expected cost per initiative is then

$$\mathbb{E}[C_{\text{anchored}}] = k + q\,c_d + (1-q)\,C, \tag{5}$$

$$\mathbb{E}[C_{\text{baseline}}] = C, \tag{6}$$

and the expected saving is their difference,

$$S = q\,(C - c_d) - k. \tag{7}$$

Proposition 3 (Break-even termination rate)**.** *The framework pays for itself when*

$$q > \theta, \qquad \theta = \frac{k}{C - c_d}. \tag{8}$$

Proof. Set $S > 0$ in (7) and solve for q. Since $c_d < C$ the denominator is positive and the inequality direction is preserved. □

C. What the threshold looks like

Read θ as the minimum share of initiatives that must be non-viable before the framework pays for itself. The parameters are organisation-specific, so Table III tabulates it across plausible ranges, with the overhead k down the rows and the detection cost c_d across the columns, both as fractions of C. Taking the middle cell: if running the anchors costs five per cent of a project's budget and the problem surfaces once two per cent has been spent, then $\theta = 5.1\%$, so roughly one initiative in twenty has to be a dud before the process breaks even. The values bracket a range rather than assert a figure.

TABLE III
BREAK-EVEN TERMINATION RATE θ, WITH ANCHOR OVERHEAD k AND DETECTION COST c_d AS FRACTIONS OF TOTAL DELIVERY COST C

	Detection cost c_d		
Overhead k	$0.02\,C$	$0.10\,C$	$0.25\,C$
$0.02\,C$	2.0 %	2.2 %	2.7 %
$0.05\,C$	5.1 %	5.6 %	6.7 %
$0.10\,C$	10.2 %	11.1 %	13.3 %

Across the whole range θ sits between roughly two and thirteen per cent. The framework pays whenever more than about one initiative in eight is non-viable in a way an anchor would catch, and pays comfortably at one in twenty.

This is deliberately not an appeal to an industry failure statistic. Such figures exist, they vary widely, and they are mostly vendor surveys. The argument does not need them. An organisation knows its own history: it can count the initiatives it started in the last three years, count those that were quietly shelved, deployed and abandoned, or completed without anyone able to say what changed, and compare that fraction against θ. The threshold is low enough that the comparison is rarely close, and the reader who finds their own rate below two per cent has no misalignment problem and no need for this framework.

D. A portfolio in currency

Ratios are hard to feel, so Table IV runs the same model over a portfolio with money attached. Take twenty initiatives in a year, each costing EUR 500,000 to carry through to deployment, so EUR 10 million of delivery in total. Anchor overhead at five per cent of project cost is EUR 25,000 per initiative. Detection at A2, after two per cent has been spent, is EUR 10,000. Suppose five of the twenty turn out non-viable, a quarter of the portfolio. The figure is chosen to be illustrative, not typical; substitute your own.

TABLE IV
THE MODEL WITH FIGURES ATTACHED. TWENTY INITIATIVES AT EUR 500,000 EACH, ANCHOR OVERHEAD EUR 25,000 PER INITIATIVE, DETECTION AT A2 COSTING EUR 10,000, FIVE INITIATIVES NON-VIABLE. ALL FIGURES IN EUR.

	Baseline	**Anchored**
Anchor overhead, all 20 initiatives	0	500,000
Spend on the 5 non-viable	2,500,000	50,000
Spend on the 15 viable	7,500,000	7,500,000
Total outlay	**10,000,000**	**8,050,000**
Net saving	0	**1,950,000**

Read it down the columns. Without the anchors, all twenty initiatives run to completion and the portfolio costs EUR 10 million, of which EUR 2.5 million bought five systems nobody could use. With the anchors, those five stop at A2 for EUR 10,000 apiece, so the same five cost EUR 50,000 instead of EUR 2.5 million. Set against that saving is EUR 500,000 of overhead, charged on all twenty including the fifteen that proceed and benefit from nothing but the checks. The portfolio costs EUR 8.05 million and the organisation keeps EUR 1.95 million, just under a fifth of what it was going to spend.

Break-even for this portfolio is $\theta = 25{,}000/490{,}000 = 5.1$ per cent, which is 1.02 initiatives out of twenty. One dud a year pays for running the anchors across everything. Every dud after the first is return.

Three features of (7) are worth drawing out. Detection cost enters only through $C - c_d$, so moving detection earlier matters most when it moves it a long way, which is why A2 carries disproportionate weight. The overhead k is incurred on every initiative while the saving accrues only on the non-viable ones, so the case strengthens as portfolio size grows and individual outcomes average out. And S is a lower bound: it counts only avoided write-offs and gives no credit for the $(1-q)$ initiatives that proceed, which are better targeted for having passed the same checks.

E. Where the value comes from

Equation (7) counts one thing only: money not spent finishing work that should have stopped. That is the most quantifiable channel and the smallest. Four others follow from results already established, and each is a recurring benefit rather than a one-off.

Released capacity. The binding constraint in most organisations is qualified people, not budget. Terminating at A2 returns a team to the portfolio with the calendar mostly intact, where terminating after deployment returns it having consumed a year. The saving in (7) is denominated in cost; the saving that matters is denominated in the next initiative those people can staff.

Lower cost to run, permanently. A4 sizes to the threshold rather than to the budget. Every component not admitted is a component never operated, never patched, never migrated and never handed to a successor. Capital cost is paid once. Operational cost is paid every year the system lives, which is why the smallest sufficient architecture compounds in a way that a one-time saving does not.

Refusal without escalation. Corollary 1 makes admissibility decidable at the anchor where the request arrives. The practical effect is that "no" stops being a political act requiring seniority and becomes a factual observation that nobody can name the element this serves. Organisations do not lack the will to refuse scope. They lack a defensible basis for it, and the cost of that gap is paid in scope nobody wanted and nobody could argue against.

Closure that survives contact with finance. This is the channel most often overlooked. Coverage, Definition 3, requires that every element of α be measured by something in μ. An initiative that succeeds without coverage has succeeded unprovably: the sponsor believes it worked, the finance function sees a number that answers a question nobody asked, and the follow-on investment goes elsewhere. Failed coverage does not destroy value. It destroys the evidence of value, which for the purposes of the next budget cycle is the same thing.

F. What this model does not establish

The model assumes detection is certain once an anchor is applied, which overstates it. Anchors reduce the probability of missing a fatal problem; they do not drive it to zero. It treats cost as a scalar and ignores opportunity cost, which understates the case, since the scarce resource in most organisations is qualified people rather than budget. It says nothing about q, which is the one parameter that would need measuring and the one this paper cannot supply.

What the model does establish is the shape of the argument. The framework is worth running when non-viable initiatives are common enough, the threshold is computable from parameters an organisation already knows, and the threshold is low.

VII. ILLUSTRATIVE APPLICATION

The following traces an archetypal initiative to make the mechanics concrete. The example is constructed rather than observed, chosen because the pattern recurs across sectors, and no claim of empirical support rests on it.

An operations director sponsors work described initially as "we want predictive maintenance." A1 replaces that with an ambition statement: unplanned downtime on a named asset class is to fall, because downtime is the quantity on which the sponsor is measured. The system implied by the original phrasing has gone. The outcome remains.

A2 finds condition-monitoring telemetry for roughly two thirds of the asset class, maintenance work orders that record the intervention rather than the fault, and therefore no failure labels of the kind the ambition assumes. That is the decisive finding of the whole initiative, and it arrives before a single euro is spent on modelling. Scope contracts to the instrumented subset, and the labelling question gets raised rather than assumed away.

A3 yields a threshold negotiated with the maintenance planners who will act on alerts and the safety function that can veto automated intervention. The planners establish that an alert is useful only with enough lead time to schedule against. The safety function establishes that a missed critical failure carries a cost no precision improvement offsets. The resulting threshold is asymmetric, and a delivery team optimising a symmetric objective would never have arrived at it.

A4 sizes the architecture to that threshold. The threshold tolerates batch scoring, so a streaming platform is excluded despite being technically attractive. Residency constraints on the telemetry decide where inference runs. What comes out is smaller than anyone originally pictured, and smaller is the objective.

A5 tests the case at the scope that survived A2, not the scope first imagined. If the instrumented subset turns out too small to justify the investment, the right outcome is termination, with the labelling gap written down as the precondition for a future attempt. Terminating here counts as the process working.

Finally, μ is the change in unplanned downtime on the named asset class, which is the quantity recorded in α. It is neither model precision, a property of σ, nor alert lead time, a property of τ.

VIII. Limitations and Threats to Validity

This is a practitioner framework and its evidential status should be read accordingly.

a) No empirical validation.: The paper offers no controlled comparison against an alternative process. The framework comes out of practice and is argued analytically. Nothing here demonstrates that it outperforms CRISP-DM or stage-gate governance on any measured dimension.

b) Single-observer derivation.: One practitioner abstracted the anchors from one professional trajectory. Practices that look necessary from that vantage point may turn out to be artefacts of the sectors, organisation sizes and engagement models encountered.

c) Survivorship and attribution.: Initiatives that succeeded under this process may have succeeded for unrelated reasons, and the process cannot claim credit for the terminations it recommended without knowing the counterfactual.

d) Construct validity of the invariant.: The claim is that μ resolving to α is the property that matters. The benefit may instead come from the forced presence of specific counterpart roles, which the anchors also mandate. Nothing here separates the two effects.

e) Applicability boundary.: The framework assumes an identifiable sponsor, an identifiable finance function and an architecture authority. Organisations without them, including early-stage companies and research settings, fall outside its scope.

Validating it properly would take a multi-site study classifying initiatives by anchor conformance and measuring outcomes against the ambition statement rather than against delivery milestones. A weaker but far more tractable design would code existing project documentation for the derivation links defined in Section V and test whether their presence predicts sustained production use.

IX. Conclusion

Enterprise AI initiatives fail for reasons usually described in technical terms that turn out to be structural. The structural cause examined here is intent drift: a sequence of individually reasonable translations that together separate the delivered system from the outcome that justified building it. The five anchors bind each translation to a counterpart and a named artefact, and the sequencing property requires that the measurement closing an initiative resolve back to the ambition that opened it.

None of the five practices is new, and this paper has not claimed otherwise. The claim is narrower. Established process models supply stages without an invariant; an invariant is cheap to state and enforceable in practice; and holding one is what turns a set of sensible practices into a process capable of refusing work.

The case for adopting it reduces to four statements, each derived above rather than asserted. The framework repays its own overhead once more than two to thirteen per cent of initiatives are non-viable, a threshold computable from (8) using numbers an organisation already has, and one that published attrition rates clear by a wide margin. Enforcement costs five local checks rather than a programme-wide audit, because Proposition 1 makes global traceability follow from local ones. Refusal becomes a factual question rather than a political one, because Corollary 1 decides admissibility at the anchor where the request arrives. And an initiative that succeeds can prove it, because coverage requires the closing measurement to answer the question the sponsor actually asked.

The last of these is the one practitioners underrate. Most delivery processes are built to prevent failure. This one is also built to make success legible, and in an organisation that allocates next year's budget on the evidence of this year's results, an unprovable success and a failure are worth the same.

Drift should be instrumented directly. Record the derivation links as first-class artefacts instead of as documentation, and D from Definition 4 becomes a number available during delivery rather than a judgement available afterwards. A process that can count its own drift can be corrected while correction is still cheap, which is the whole argument in miniature.

About the Author

Marius Golombeck holds an M.Sc. and a B.Sc. in Business Informatics from the University of Applied Sciences and Arts Dortmund, graduating with honours. His master's thesis applied convolutional neural networks to travel destination prediction from multivariate parameters [19]. During his studies he worked on fault detection in vehicle air pressure system sensor data [20], with a BMBF research grant (01S17075) and published on requirements specification [14], automotive functional cybersecurity [16] and data protection under the GDPR [15]. He began his career in software engineering at

Porsche Motorsport, then founded and led the company's Esports Racing programme. Since 2022 he has worked for C3 AI in enterprise AI solution architecture, acting as principal technical advisor to large enterprise clients across industrial manufacturing, building technologies and supply chain. The framework presented here was abstracted from that practice. The views expressed are the author's own and do not represent those of any employer or client.

REFERENCES

[1] A. Paleyes, R.-G. Urma, and N. D. Lawrence, "Challenges in deploying machine learning: A survey of case studies," *ACM Computing Surveys*, vol. 55, no. 6, pp. 1–29, 2022.

[2] O. C. Z. Gotel and A. C. W. Finkelstein, "An analysis of the requirements traceability problem," in *Proc. IEEE Int. Conf. on Requirements Engineering*, 1994, pp. 94–101.

[3] B. Ramesh and M. Jarke, "Toward reference models for requirements traceability," *IEEE Transactions on Software Engineering*, vol. 27, no. 1, pp. 58–93, 2001.

[4] D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, J.-F. Crespo, and D. Dennison, "Hidden technical debt in machine learning systems," in *Advances in Neural Information Processing Systems 28*, 2015, pp. 2503–2511.

[5] *IEEE Recommended Practice for Software Requirements Specifications*, IEEE Std. IEEE Std 830-1998, 1998.

[6] *Systems and Software Engineering – Life Cycle Processes – Requirements Engineering*, ISO/IEC/IEEE Std. ISO/IEC/IEEE 29 148:2018, 2018.

[7] R. G. Cooper, "Stage-gate systems: A new tool for managing new products," *Business Horizons*, vol. 33, no. 3, pp. 44–54, 1990.

[8] B. W. Boehm, "A spiral model of software development and enhancement," *Computer*, vol. 21, no. 5, pp. 61–72, 1988.

[9] P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, and R. Wirth, "CRISP-DM 1.0: Step-by-step data mining guide," The CRISP-DM Consortium, Tech. Rep., 2000.

[10] E. Breck, S. Cai, E. Nielsen, M. Salib, and D. Sculley, "The ML test score: A rubric for ML production readiness and technical debt reduction," in *Proc. IEEE Int. Conf. on Big Data*, 2017, pp. 1123–1132.

[11] S. Amershi, A. Begel, C. Bird, R. DeLine, H. Gall, E. Kamar, N. Nagappan, B. Nushi, and T. Zimmermann, "Software engineering for machine learning: A case study," in *Proc. 41st Int. Conf. on Software Engineering: Software Engineering in Practice*, 2019, pp. 291–300.

[12] T. H. Davenport and R. Ronanki, "Artificial intelligence for the real world," *Harvard Business Review*, vol. 96, no. 1, pp. 108–116, 2018.

[13] R. S. Kaplan and D. P. Norton, "The balanced scorecard: Measures that drive performance," *Harvard Business Review*, vol. 70, no. 1, pp. 71–79, 1992.

[14] H. Orlowski, M. Golombeck, M. Filusch, and L. Müller, "Accident warnings during edge case traffic events: Requirements specification," University of Applied Sciences and Arts Dortmund, Tech. Rep., 2020.

[15] M. Golombeck and H. Orlowski, "Options for anonymising personal data under the general data protection regulation," University of Applied Sciences and Arts Dortmund, Tech. Rep., 2018.

[16] M. Golombeck, "Functional cybersecurity: Standards and concepts," University of Applied Sciences and Arts Dortmund, Tech. Rep., 2020.

[17] B. W. Boehm, *Software Engineering Economics*. Englewood Cliffs, NJ: Prentice-Hall, 1981.

[18] B. Boehm and V. R. Basili, "Software defect reduction top 10 list," *Computer*, vol. 34, no. 1, pp. 135–137, 2001.

[19] M. Golombeck, "Machine learning-based prediction of travel destinations based on multivariate parameters," Master's thesis, University of Applied Sciences and Arts Dortmund, 2020.

[20] L. Dupont, M. Golombeck, and S. Tübben, "Detecting faulty air pressure system sensor data in scania trucks," University of Applied Sciences and Arts Dortmund, Tech. Rep., 2019.