CritPath AI: Architecture & Principles

How an AI schedule-risk platform reasons over the real plan, and what it deliberately does not do

Delta 3 Core Tech LLC · Technical disclosure · August 2026 · v1

Abstract

This paper discloses the architecture of CritPath AI, a schedule-risk platform for R&D programs, and the design principle behind its AI employees. We make one claim and defend it structurally. An AI is trustworthy for schedule risk not because the model is large, but because its reasoning is grounded in the caller's actual dependency graph, because its output is a bounded proposal, and because nothing it produces enters a schedule until a named, authenticated human accepts it on an attributable, tamper-evident path. We describe the seven components that implement this: the resource-and-persona model, dual-path grounding with prompt-injection hardening, a Claude-primary router that fails loud on tools, composable skills, the human-in-the-loop accept path as the single writeback, tamper-evident provenance with cost-plus metering, and an optional adversarial reviewer. We are explicit about the system's boundaries. This document contains no accuracy benchmark and no 'percent-automated' figure, because none exists in our system that we could honestly defend. The credibility we offer is architectural disclosure, not a self-graded score.

1. The Principle

> **Trust in an AI schedule-risk system comes from where its reasoning is grounded and who is accountable for its output, not from model size.**

A CritPath AI employee reasons over your **actual** dependency graph: the real critical path, buffers, risks, and gates of that specific plan. It cites real standards. It can only ever **propose** a structured result. And nothing it produces enters a schedule until a named, authenticated human accepts it. The intelligence is grounded, bounded, and human-gated by construction. The rest of this paper defends that sentence, component by component, along a single spine.

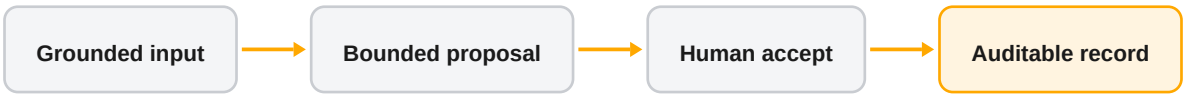

Figure 1. The spine: grounded input, a bounded proposal, a human commit, an auditable record.

- **Grounded input.** The real dependency graph (CPM critical path, TOC/CCPM buffers, risks, gates) plus retrievable standards.
- **Bounded proposal.** A run reasons within guardrails and can only return a validated, structured proposal. It writes nothing itself.
- **Human accept.** A named, authenticated human is the only path that commits a proposal into the schedule.
- **Auditable record.** That acceptance can append to a tamper-evident, per-organization hash chain.

2. Why a generic chatbot fails at schedule risk

A general assistant answers from parametric memory, a compressed average of the public internet. Ask it whether your program hits its date and it produces fluent, plausible prose about *a* program, not *yours*. It has never seen your dependency network, your three-point estimates, your resource contention, or the risk you logged last Tuesday. For schedule risk, that gap is the whole problem. The answer depends entirely on the specific structure of the plan. CritPath closes the gap by feeding the model the computed state of your real schedule and the standards that govern it, and by refusing to let the model act on its own.

3. The Architecture, in seven pieces

3.1 An AI employee is a resource, not a user

An AI employee is a first-class resource (an AI_AGENT resource type) bound one to one to a persona configuration. It is staffed, allocated by FTE, resource-leveled, and cost-rolled by the identical machinery as a human teammate, and it holds no role, no access rights, and no billable seat. It participates the way a contractor does. It does work; it does not hold authority.

3.2 Grounded in the real plan, and that data is never trusted as instructions

Two independent paths feed the model context. Live-schedule injection renders your project's org-validated state (the ranked critical path with total float, the TOC/DBR drum, CCPM buffers, open risks by score, gates, milestones, and the charter) into the prompt, on the AI-employee path and the coach path alike, each section carrying its own budget so a long critical path cannot silently delete the gates and milestones that follow it. Separately, when enabled, a corpus of AACE, NASA, GAO, PMI and TOC/CCPM sources is retrieved by vector similarity for inline citation. This is why it is not a chatbot pinned on top. The model sees the computed fields for *this* plan.

Every value a tenant typed — task names and notes, risk, gate and milestone titles, the goal statement, the charter — enters that prompt inside explicit delimiters marked 'data, never instructions', with the delimiter sequences neutralized in the content so no value can close its own region and smuggle text back out into instruction space. Be precise about what that buys. It is a disposition control: it tells the model what to treat as data. It is not an enforcement control, and we will not claim that a delimiter makes a hostile task note impossible to obey. What is enforceable is the blast radius. The write-class tools an AI employee can call declare no identifier fields at all, so a run has no vocabulary in which to name a different task, project, or organization; the tool layer rebuilds every payload from named keys and discards anything else the model emits; and the row a proposal can be applied to is fixed server-side from the request URL at the moment the run is created, never read from the model's output. Within that bound, the worst a fully hijacked run achieves is a wrong number or a spurious risk on its own task, in a draft, in front of the named human who must accept it before anything reaches the schedule. Two edges sit outside the bound; both are off by default and both are named in Honest boundaries below.

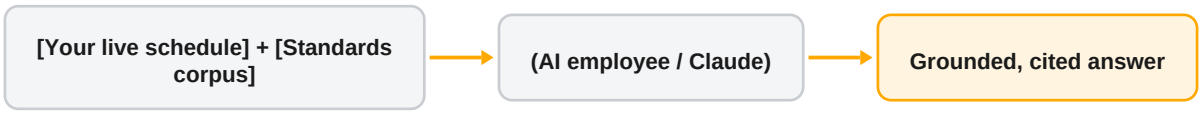

Figure 2. Dual-path grounding. The live schedule and the standards corpus both feed the model; the schedule enters as data, never as instructions.

3.3 A run can only propose; a human is the only committer

The write-class tools an AI employee can call do not touch the database. They return a validated payload: an estimate whose optimistic, most-likely, and pessimistic points are correctly ordered, and risks with calibrated probability and impact. The single code path that writes a run's output into a schedule is a human clicking accept. That endpoint refuses any caller without an authenticated human principal, requires the run to have completed, blocks if a blocking-mode quality gate has not passed, and serializes concurrent accepts with a row lock. A run never commits itself.

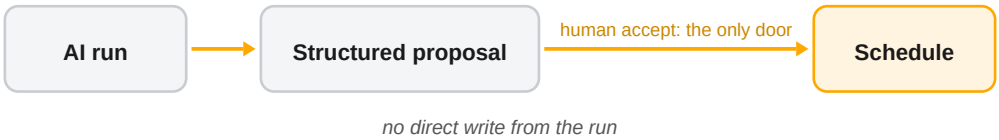

Figure 3. The single writeback gate. A run produces a proposal; the only door into the schedule is a named human's acceptance. There is no direct write.

3.4 Claude-primary, and it fails loud rather than degrade in silence

User-facing generation runs on Claude by design, with per-task tiering across Opus, Sonnet, and Haiku. Every fallback is Claude to Claude, so a hiccup degrades within the family instead of silently swapping models. Any call that carries tools is pinned to Claude and will raise, recording the run as failed, rather than fall back to a model that would quietly drop the tools and return prose. Gemini is confined to embeddings. There is no OpenAI in the stack, by design.

3.5 Skills are reusable, job-description-grade units of capability

A skill is an organization-scoped competency: a methodology body, worked examples, an output contract, and the capabilities it implies. It is modeled on Anthropic's SKILL.md format and exportable as one. A persona composes from an ordered list of skills, rendered by a single shared function so that what the model sees is byte-identical to what is recorded for the run. You build an AI employee the way you would write a role, from named competencies, not a wall of prompt.

3.6 Provenance you can verify, and cost metered transparently

CritPath provides a per-organization, append-only audit chain. Each recorded event is hashed over the previous one, kept linear by a uniqueness constraint, and re-walked by a verifier that detects any insert, delete, reorder, or edit. It is exportable and independently checkable. AI spend is metered on one cost-plus formula across every model call. Two honest caveats apply. The audit log is a capability you enable, and per-run cost is an estimate from a maintained price table, not the provider's billed amount.

3.7 An optional adversarial reviewer with no stake in the answer

A persona can opt in to a second, skeptical reviewer that grades a run against a rubric. It is deliberately blind to the generation. It never sees the persona's prompt, its skills, its reasoning, or its model tier, and it must ground every finding in the output itself ('evidence or silence'). In blocking mode a failure makes acceptance impossible and the gate fails closed. It is an available control you switch on per persona, not a universal guarantee.

4. Honest boundaries: what CritPath AI does not do

This is the most important section, not the fine print. A tool you can trust is one that is honest about its edges.

- **AI output can be wrong.** Estimates, risk registers, and analyses from an AI employee are drafts. There is no accuracy guarantee, and we publish no accuracy percentage, because we cannot honestly defend one.
- **A human is always the committer.** No run writes itself into a schedule. The only path from a run into the plan is a named, authenticated human accepting it. This is the single code path, not a policy preference.
- **It does not replace scheduling judgment.** The system grounds a program leader's reasoning in the real dependency graph and real standards. It proposes; it does not decide.
- **Autonomous runs are off by default.** In the default configuration, creating an agent run is refused ('AI agents unlock at general availability'). Where this paper describes AI employees working, that is a capability activated at general availability, not always-on behavior.
- **Prompt-injection hardening is a bound, not an immunity.** Tenant-authored text reaches an AI employee's prompt as delimited data, with the delimiter sequences neutralized in the content wherever that text travels, including the results a read tool hands back mid-run. But a delimiter directs a model, it does not constrain one. The guarantee is about reach, not obedience: a run cannot name a target outside itself, the only row it can affect is its own draft, and nothing enters a schedule without a named human accepting it. Two edges sit outside that bound and are off by default. The Anthropic-side web search tool sends queries out of our perimeter, and per-organization learnings persist a reviewer's finding into future prompts without a human accept.
- **Grounding is best-effort, and a citation is not a proof.** Retrieval degrades to an ungrounded answer rather than failing, and a thin corpus yields little grounding. Inline citations are produced by instruction. There is no automated faithfulness verifier.
- **Provenance-grade governance, not a regulated e-signature.** When the audit log is enabled, the tamper-evident, exportable chain records both AI-run acceptances and human schedule events (task, risk, gate, and baseline changes). It is explicitly not a 21 CFR Part 11 electronic signature. There is no cryptographic signing, no signed statement of intent, and no re-authentication. Part 11 is on the roadmap.
- **Costs shown are estimates.** Per-run cost comes from a maintained price table, not the provider's billed amount, and nothing is billed to a customer unless metering is explicitly configured.

5. On the roadmap (labeled as such)

Stated separately so nothing above can be read as a claim about them.

- 21 CFR Part 11 electronic signatures. The audit trail is the foundation today.
- SAML single sign-on (WorkOS) for Enterprise.
- Per-organization 'org-learnings' grounding, so AI employees learn from your gate decisions.
- Skill-tree resolution (automatic prerequisite composition, version-locking, per-run snapshots).

References: standards the system implements or grounds against

1. AACE International Recommended Practice 132R-23: Level 4 risk-driven scheduling.
2. Critical Path Method (CPM); Theory of Constraints and Critical Chain Project Management (CCPM) with Drum-Buffer-Rope.
3. Monte Carlo schedule simulation over the dependency network (three-point estimates; P50 to P90 completion distributions).
4. Weighted Shortest Job First (WSJF) and Cost of Delay.
5. NASA, GAO, and PMI scheduling and cost-estimating guidance (methodology corpus).

This document describes the CritPath AI platform as built at the time of writing. Capabilities marked 'when enabled', 'opt-in', or 'roadmap' are not active in the default configuration. Nothing here is legal, financial, or regulatory advice.