The Intervention Penalty: A Simulation Study of Human Checkpoint Costs in AI Coding Governance

Julian Ramirez
julian@bosun.sh

Sofia Plajutin
sofia@bosun.sh

bosun.sh — AI-Native Tech Lab

Preprint — April 2026

Abstract

We study whether frequent human checkpoints help or hinder high-capability AI coding agents under a governance-oriented simulation model. We introduce Bosun, a self-governance framework for AI coding agents comprising agent-optimized KPIs, hierarchical OKRs with automated key-result verification, a Kanban system with machine-verifiable definitions of done, and continuous adversarial testing. We formalize the *intervention penalty* as the cumulative productivity cost of human interruption and show analytically that, under the chosen functional forms, it grows superlinearly with intervention frequency. In a parameterized simulation across 100 synthetic coding tasks, Bosun-governed agents with human review limited to final approval achieved first-pass quality similar to checkpoint-based HITL configurations while completing tasks at 1.4–9.9× lower cycle time depending on intervention frequency, and outperformed an ungoverned autonomous baseline on modeled quality and defect rate. Sensitivity analysis across 26 configurations and three adversarial counter-scenarios shows the penalty persists in all tested regimes. These results are conditional on explicit simulation assumptions and do not establish production causal effects. They instead motivate a testable hypothesis for field studies: in sufficiently structured software tasks, governance specified upfront may substitute for frequent action-level oversight.

1 Introduction

In legal research and professional advertising, AI-only approaches have been observed to exceed the performance of human+AI teams, challenging the assumption that human oversight improves AI performance Fortune/Eye on AI [2025]. In software development, AI coding agents—Devin, Claude Code, Cursor, GitHub Copilot—resolve GitHub issues, generate production code, and maintain large codebases with minimal human guidance. Yet the dominant deployment model wraps these agents in human-in-the-loop (HITL) checkpoints: human approval before every commit, tool invocation, or architectural decision. The safety intent is clear. The outcome is not. We hypothesize that frequent human intervention degrades AI agent performance. The problem is not AI capability. The problem is governance.

1.1 The Micromanagement Parallel

Before stating our technical argument, we note a structural parallel from management science that motivates our framing. Chambers describes micromanagement as controlling behavior that removes autonomy, demands constant approval, and fragments capable workers' output Chambers [2004]. White characterizes the condition as a disease: pervasive approval-seeking, inability to prioritize, and chronic underperformance relative to capability—cured by restoring autonomy, not adding oversight White [2010]. The parallel is structural, not psychological: we do not claim AI agents

suffer learned helplessness. We observe that the same governance pattern—interrupt execution, require approval, resume—imposes the same structural costs regardless of whether the executor is human or machine.

Those costs are concrete. Each HITL interruption forces context reconstruction: the agent must re-establish which files are open, which decisions were made, and where execution left off—compounding latency across every step of a multi-step task. Beyond latency, agents trained under approval-seeking feedback conditions may exhibit output degradation toward reviewer satisfaction rather than task quality; we model this dynamic in Section 5.

Practitioner reports (not peer-reviewed) describe AI systems making decisions at velocities that exceed human review capacity, with HITL checkpoints adding friction without meaningful oversight SiliconANGLE [2026]. Some deployments report AI agents handling 90–99% of tasks, with humans transitioning to orchestrators who set constraints rather than approve each action Analytics India Magazine [2025]. Section 4 provides simulation results consistent with the same pattern.

1.2 Structural Support, Not Puppetry

A powered exoskeleton does not tell each leg muscle when to fire. It sets a goal—stand, walk, climb—and provides structural support while the body's own feedback loops handle execution. Attempting to control each joint directly produces instability, not augmentation. The same principle applies to AI agents. Governance that specifies the destination and enforces measurable standards—while leaving execution to the agent—produces better outcomes in the model than governance that interrupts execution to confirm each step. We need an exoskeleton for AI agents, not a puppeteer.

1.3 Contributions

This paper makes three contributions:

- **Conceptual: intervention penalty hypothesis.** We define intervention penalty as the measurable productivity and quality cost of each human interruption to an AI agent. We hypothesize that, under the conditions specified in Section 4, intervention frequency is a stronger predictor of task degradation than residual capability variation.
- **Formal: mathematical framing.** We formalize the intervention penalty as $\mathrm{IP}(k)$ with explicit functional forms for latency, context reconstruction, and rework components. We show that under these assumptions, $\mathrm{IP}(k)$ is strictly superlinear in k.
- **Analytical: simulation-based exploration.** We instantiate the model in a parameterized simulation and explore the conditions under which the intervention penalty emerges. We examine how the penalty varies across intervention frequency, human latency, model error rates, and task complexity distributions.

We do not argue that human oversight has no value. Security review, compliance verification, and novel-domain decisions involve irreversible consequences and genuinely require human judgment. Our claim is bounded: we hypothesize that in software development tasks where AI capability is sufficient and task structure is well-defined, intervention frequency past a threshold decreases performance. "Sufficient AI capability" is operationalized in our model as performance above a defined threshold—a parameter we calibrate from the literature (Section 4). Within these model boundaries, the simulation suggests that structured self-governance can substitute for action-level oversight without sacrificing quality.

2 Related Work

2.1 Human-in-the-Loop in AI Systems

Human-in-the-loop oversight serves as the primary safety mechanism for AI deployment in high-stakes domains. The rationale is straightforward: humans catch errors, supply contextual judgment, and maintain accountability when automated systems fail. HITL is foundational in regulated industries—TDWI documents how healthcare and financial organizations rely on human review to satisfy HIPAA, anti-discrimination law, and fiduciary requirements, with manual checkpoints preventing cascading errors in data pipelines TDWI [2025]. The EU AI Act codifies this rationale by mandating human oversight for high-risk systems (Article 14) European Parliament and Council of the European Union [2024], and the broader HITL literature frames human involvement as a safety multiplier.

The sophisticated version of this argument does not claim continuous oversight; it claims targeted oversight at decision boundaries—humans review outputs at natural task transition points while automation handles execution between them. On this model, human value concentrates in ethical oversight and contextual judgment under uncertainty, not in routine monitoring Hao et al. [2025]. Hao, Demir, and Eyers confirm this pattern from 28 interviews across nine leading firms: human expertise is most valuable at novel or ethically charged decision points, least valuable in high-volume routine processing. We accept this characterization. Our challenge is narrower: in software development contexts where AI capability is high and task structure is well-defined—conditions we operationalize in Section 4—even targeted checkpoint-style HITL imposes costs that exceed its benefits.

Those costs compound. CIO documents "alert fatigue"—under high-volume review conditions, human reviewers shift from genuine evaluation to nominal approval, rubber-stamping AI output without meaningful scrutiny CIO [2025]. What was designed as targeted oversight becomes indistinguishable from no oversight in terms of error-catching performance, while still imposing all the latency and context-interruption costs of the checkpoint model.

2.2 AI Agents in Software Development

The last three years have produced a rich empirical record on AI coding agents. Tools including GitHub Copilot, Cursor, Devin, and Claude Code now participate in production software development across millions of developers. The aggregate data presents a puzzle: adoption is nearly universal but performance gains are modest.

METR conducted a tightly controlled study: 16 experienced open-source developers working on 246 real issues under a randomized controlled design found that AI-assisted developers took 19% longer to complete tasks than controls working without AI tools METR [2025]. What makes this study valuable is its methodology—objective time measurement on real tasks, not self-report—though its scope (experienced open-source maintainers on existing codebases) limits direct generalizability to greenfield enterprise development. Whether METR's 19% slowdown reflects HITL overhead (our interpretation) or AI capability limits (an alternative reading) is ambiguous from their data alone; we address this by operationalizing capability sufficiency as a precondition in Section 4. The paradox deepened: developers expected AI to accelerate their work and reported believing it had, even as objective measurements showed the opposite. Qodo's state-of-the-industry survey finds that 65% of developers report AI tools miss relevant context during refactoring, and 44% attribute degraded code quality directly to context gaps Qodo [2025]. Stack Overflow's 2025 developer survey shows 46% of developers actively distrust AI output accuracy, with experienced developers showing the highest skepticism Stack Overflow [2025]. One industry analysis reports that 92.6% of developers

now use AI coding assistants, yet productivity gains have remained at approximately 10% ShiftMag [2026].

Benchmarks on autonomous agents show a more nuanced picture. Agarwal, He, and Vasilescu measure tradeoffs between AI IDE tools and fully autonomous agents, finding that velocity gains often come at the cost of maintainability Agarwal et al. [2026]. Luo et al.'s AgentDS benchmark—evaluating 29 teams across 17 domain-specific data science tasks—finds AI-only systems perform near or below the median of human participants, while the strongest results emerge from human-AI collaboration Luo et al. [2025]. The AgentDS collaboration finding is consistent with our argument: Bosun does not eliminate human involvement; it eliminates routine approval-gate HITL. Strategic human participation at genuine decision boundaries remains valuable. What the data suggest collectively is that per-action human checkpoints do not reliably close the performance gap; we argue in Section 3 that governance structure is the more productive explanatory lens.

2.3 OKR and KPI Frameworks for AI Agents

Goal-setting frameworks have been applied to AI agents primarily as task-decomposition and evaluation tools. Zheng et al. introduce OKR-Agent, which decomposes objectives into hierarchical sub-objectives assigned to specialized agents, with a multi-level evaluation loop that checks task completion after execution Zheng et al. [2023]. The evaluation loop is retrospective: it measures whether a delivered artifact met the objective, not whether execution quality is meeting standards in real time during a task. Humans define objectives and evaluate outputs; OKRs provide the decomposition scaffold. OKR-Agent outperforms unstructured agents on several benchmarks, demonstrating that hierarchical goal decomposition improves LLM reasoning—but it does not address continuous self-governance during execution.

Work on KPI frameworks for AI evaluation has expanded rapidly. A recent framework proposes five evaluation dimensions—Model Quality, System Performance, Business Impact, Human-AI Interaction, and Ethical and Environmental Considerations—validated on GPT-4, DALL-E 3, and Claude 3 Unknown [2024]. The framework is designed for human evaluators assessing deployed systems. MoghadasNian, NaserPour Asiabari, and HeidariYekta's AISA-L architecture operationalizes 110 domain-specific KPIs in a real-time governance-optimization loop, achieving a 22% improvement in forecast accuracy and 18% faster disruption recovery in airline logistics MoghadasNian et al. [2025]. AISA-L is the closest precedent to Bosun in its use of KPIs for real-time operational control. However, as described, AISA-L is designed to optimize within human-defined corridors, with human sign-off at strategic and action layer boundaries MoghadasNian et al. [2025]. The architecture is designed to be supervisable, not self-governing. The gap is structural: the KPI loop refines decisions within human-approved bounds but cannot replace human approval gates.

To our knowledge, existing work does not use KPIs and OKRs as a self-governance substrate enabling an agent to monitor its own execution quality, enforce its own Definition of Done, and decide autonomously whether its output meets acceptance criteria without routing to human review. Frameworks are either human-evaluated (assess an agent post-hoc) or human-supervised (KPI loops run inside human-defined governance boundaries). Bosun is our proposed framework for filling that gap.

2.4 Agent Scaling and Coordination

Multi-agent systems present governance challenges that single-agent frameworks do not. ImagineX Digital's practitioner analysis of multi-agent deployments identifies three structural failure modes: coordination tax (agents expend cognitive resources on interaction management rather than problem-

solving), error amplification (errors compound across agents), and inefficient resource use (agents exploit only 15% of available tool budget when unconstrained) ImagineX Digital [2026]. Adding agents to a poorly governed system degrades performance rather than improving it.

Rasheed et al. demonstrate that multi-agent SDLC automation is technically feasible—12 LLM agents, each specialized to a phase (requirements, design, development, testing, deployment), can execute a full software development lifecycle with significantly reduced cycle time Rasheed et al. [2024]. But their vision paper acknowledges that reliable coordination requires structured control flow, explicit handoff protocols, and quality gates between phases. The lesson across both works is consistent: the problem holding back agent scaling is governance structure, not capability.

Gap Statement

Existing work treats OKRs and KPIs as tools for evaluating AI agents (did the agent succeed?) or decomposing tasks (what should the agent do next?). No prior work uses these frameworks as the primary mechanism for self-governance—enabling an agent to monitor its own quality continuously, enforce its own Definition of Done, and regulate its own intervention requests during execution, without routing to human review. This is the gap Bosun fills. We present simulation results in Section 6 showing that agents equipped with agent-native KPIs and machine-verifiable DoD criteria can govern themselves as effectively as—and faster than—agents supervised by per-action human checkpoints.

3 The Bosun Self-Governance Framework

Bosun is a self-governance framework for AI coding agents. It replaces real-time human checkpoints with structured agent self-regulation. Humans define governance upfront—objectives, quality thresholds, and verification rules—then step back. The agent executes, monitors itself, and iterates without waiting for approval. We describe the framework's four design principles and five operational components.

3.1 Design Principles

Bosun rests on four principles that distinguish self-governance from human-in-the-loop (HITL) oversight.

Human role is architectural, not operational. Humans specify what good looks like—objectives, KPI targets, Definition of Done (DoD)—before work begins. They do not approve individual actions, review intermediate states, or redirect execution mid-task. This separation eliminates the intervention penalty at the cost of requiring more precise upfront specification. We define the boundary operationally: human input is *architectural* if it occurs before execution begins and is expressed as a configuration artifact (OKRs, KPI thresholds, DoD criteria); it is *operational* if it occurs during execution, responds to agent state, or gates an agent action (passive monitoring—reading logs or dashboards without producing an artifact or gating an action—is architectural by this definition). Section 4.2 operationalizes this distinction as Mode A: no human input is admitted during task execution.

All quality gates must be machine-verifiable. A quality gate that requires human judgment to evaluate is a quality gate that stalls execution. Bosun admits only criteria expressible as deterministic computations over artifacts: file diffs, metric values, parse results, test outputs. Subjective criteria are rejected at governance design time, not discovered mid-execution.

Governance overhead must be less than intervention overhead. Self-governance introduces overhead: KPI computation, DoD checking, OKR (Objectives and Key Results) evaluation. This overhead is acceptable only if it costs less than the alternative. We show in Section 5 that intervention overhead includes context reconstruction, task-switch penalties, and throughput loss—costs that compound with frequency. Bosun's overhead is constant per state transition; intervention overhead scales with frequency.

Adversarial testing is continuous, not periodic. Conventional testing occurs at defined checkpoints: code review, QA handoff, staging deployment. Bosun agents red-team their own work at every state transition. Defects are caught in the context where they were introduced, before they propagate.

3.2 Agent-Native KPIs

Human-oriented KPIs—story points burned, reviewer satisfaction, sprint velocity—rely on human judgment to evaluate and are sampled at low frequency (weekly sprints, end-of-sprint reviews). Agents can self-measure at every commit, every file write, every state transition. The gap between human evaluation cadence and agent execution cadence is the core problem: without high-frequency feedback, the agent cannot self-correct Unknown [2024].

Bosun defines eight agent-native KPIs, each machine-verifiable without human input.

Specification Adherence (SA) measures how completely a unit of work satisfies its specification.

$$SA = \frac{\text{requirements\_met}}{\text{total\_requirements}},$$

where all requirements are equally weighted. Computed by parsing the Required Elements checklist in a slice's specification file against the draft artifact. Target: $SA \geq 0.90$.

Citation Density (CD) measures scholarly rigor—whether claims are backed by references. $CD = \text{citations}/(\text{words}/250)$. Counted from `\cite{}` patterns in the draft. Target: $CD \geq 2.0$ per 250 words.

Citation Integrity (CIS) verifies that all cited works exist and are confirmed. $CIS = \text{verified\_cited\_keys}/\text{total\_cited\_keys}$. Cross-referenced against a references registry with explicit verification status. Target: $CIS = 1.0$.

Cross-Reference Consistency (CRC) checks that all internal references (Section N, Figure N, Table N) resolve.

$$CRC = \frac{\text{resolved\_refs}}{\text{total\_refs}}.$$

Computed by parsing reference patterns and confirming targets exist in the assembled document; forward references to unwritten sections are validated at assembly, while CRC tracks backward-reference consistency during drafting. Target: $CRC = 1.0$.

Review Convergence (RC) measures quality at first submission—how many review rounds elapse before acceptance. RC equals the round number when status reaches *done*. Target: $RC \leq 2$ rounds. High RC signals insufficient self-checking before submission.

First-Draft Acceptance (FDA) measures the fraction of work units accepted at round 1.

$$FDA = \frac{\text{slices\_accepted\_round\_1}}{\text{total\_slices}}.$$

Target: $FDA \geq 0.50$. FDA and RC are related but distinct: RC is per-unit (the round at which a specific unit converges); FDA is the aggregate share of units where $RC = 1$.

Terminology Consistency (TC) ensures canonical terms are used without drift.

$$TC = \frac{\text{canonical\_uses}}{\text{canonical\_uses} + \text{variant\_uses}}.$$

Computed by matching against a shared glossary. Target: $TC \geq 0.95$.

Cycle Time (CT) measures wall-clock time from work start to completion. $CT = t_{\text{done}} - t_{\text{in\_progress}}$. No threshold—tracked as meta-evidence for governance efficiency analysis.

Together, these eight metrics give the agent a continuous signal for self-correction. No human reviewer is in the feedback loop.

The eight KPIs above instantiate Bosun for the academic writing domain (this paper's operational demonstration in Section 7.5). The framework is domain-agnostic: a software development instantiation would replace Citation Density and Citation Integrity with domain-appropriate metrics (e.g., test coverage, static analysis score) while retaining the structural KPIs (Specification Adherence, Cycle Time, Terminology Consistency). The simulation in Section 4 uses the domain-general subset.

3.3 Hierarchical OKRs with Automated Verification

Bosun uses OKR (Objectives and Key Results) hierarchically: project-level objectives decompose into section-level objectives, which decompose into KPI thresholds on individual work units. An agent checks its KPIs continuously and knows, at any point, whether it is on track for each key result.

This differs from prior work. Zheng et al. Zheng et al. [2023] use OKRs as a task decomposition mechanism—objectives guide what subtasks to generate. Bosun uses OKRs as a self-governance mechanism during execution—objectives define success criteria the agent must continuously verify against its own outputs. The distinction matters: decomposition OKRs are completed once, at planning time; governance OKRs are active throughout the execution lifecycle. The mechanism that makes this structural rather than verbal is the Kanban DoD gate (Section 3.4): KPI thresholds are evaluated at every state transition, and the agent cannot advance to a subsequent state without passing them. Without this enforcement, governance OKRs and decomposition OKRs would be functionally identical.

Verification is automated end-to-end. Each key result specifies a KPI, a threshold, and a measurement method. The agent runs measurement after each state transition. If a threshold is missed, the agent revises before advancing. No human approval is required to proceed—and no human approval can bypass a failed threshold.

Section 6 evaluates whether this governance structure can achieve HITL-like quality at lower modeled cost within the simulator. The Level 4 positioning is an operationalization of the simulation's Mode A design, not a premise of it.

Bosun targets Level 4 of 5 on the autonomy spectrum defined by Feng, McDonald, and Zhang Feng et al. [2025] (published through Anthropic): autonomous execution within defined bounds. Level 5 (unbounded autonomy) is neither claimed nor sought. The governance structure—the OKRs, KPI targets, and DoD—is the mechanism that makes Level 4 safe without requiring Level 3 (human-in-the-loop approval at each step).

3.4 Agent-Optimized Kanban

Bosun's Kanban is designed for machine state management, not human visualization. Four properties distinguish it from conventional Kanban.

Machine-verifiable Definition of Done. Every state transition from `in_progress` to `done` is gated by a checklist of machine-verifiable criteria. Items that cannot be verified by code are not admitted to the DoD. The agent cannot mark work complete without passing all gates.

Agent-generated subtasks. When a work unit enters `in_progress`, the agent decomposes it into subtasks at a granularity appropriate for self-monitoring. Subtask completion is tracked in the Kanban state, giving the agent a structured basis for progress estimation and blocked detection.

Automatic blocked detection. A work unit transitions to `blocked` automatically when a dependency condition cannot be satisfied—a required artifact is missing, a prior slice is still in `in_progress`, or a KPI threshold cannot be met due to an unresolved upstream issue. Blocked detection is deterministic, not reliant on the agent reporting its own stalls.

Dynamic WIP limits. WIP limits are set as a function of dependency depth and available context budget, not as fixed per-stage constants. This prevents the pipeline from overloading context with parallelism that cannot actually proceed.

3.5 Continuous Adversarial Testing

Conventional quality assurance is periodic: unit tests on commit, integration tests on merge, security scans on release. Adversarial testing in Bosun is continuous—the agent red-teams its own work at every state transition before advancing.

At each transition, the agent runs adversarial probes across five categories: input validation (does the output handle malformed inputs gracefully?), security (does the output introduce injection vectors, unsafe operations, or credential exposure?), error handling (do failure paths behave correctly?), edge cases (does the output handle boundary conditions?), and performance (does the output introduce regressions in latency or resource use?).

Critically, adversarial testing occurs in the context of the work that introduced the defect. Finding an injection vulnerability at the function level, at the time of writing, requires no context reconstruction. Finding the same vulnerability at code review, days later, requires re-establishing context before the fix can begin. Continuous adversarial testing is not merely earlier testing—it is cheaper testing.

Architecture: The Bosun Feedback Loop

Figure 1 shows the Bosun execution architecture. Task Input enters the system at the top. The agent performs OKR Decomposition, mapping the task to objectives, key results, and KPI thresholds. Execution proceeds through Kanban states (`todo` → `in_progress` → `in_review` → `done`), with KPI monitoring active throughout. At each state transition, Adversarial Testing runs. Before marking work complete, the DoD Check evaluates all machine-verifiable criteria. If the DoD check passes, the work unit advances. If it fails, a Revision Loop returns control to the `in_progress` state.

Human input appears only at the top of the diagram: defining the OKRs, KPI targets, and DoD before execution begins. There are no arrows from the human into the execution loop. The governance structure is the mechanism; real-time human oversight is not.

4 Simulation Design

We evaluate the intervention penalty hypothesis through a parameterized simulation. Key parameters are documented in Table 1; every number in this section and in Section 6 traces to that table. This is a simulation—an analytical instrument and exploratory environment—not a deployment study:

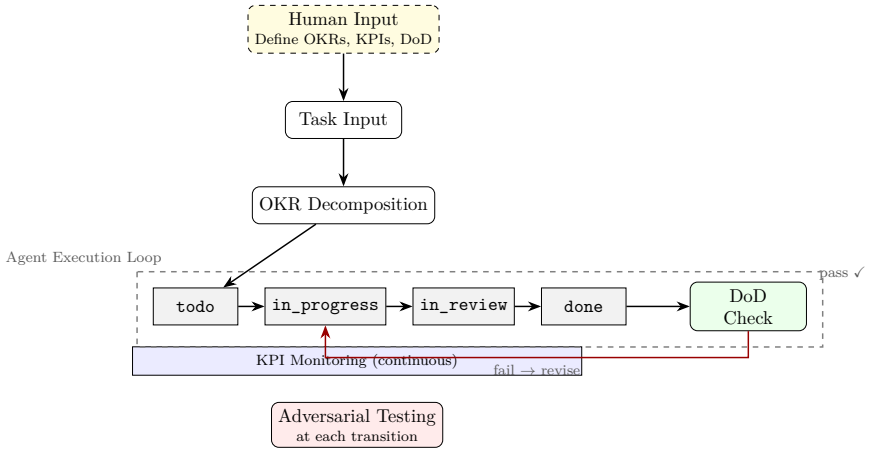

Figure 1: The Bosun execution architecture. Human input (top) defines governance artifacts—OKRs, KPI targets, DoD—before execution begins. The agent executes autonomously through Kanban states with continuous KPI monitoring and adversarial testing at each state transition. No human input enters the execution loop.

intervention cost parameters are calibrated from the empirical literature (Section 4.3), and the independent quality rubric is defined in Section 4.2, prior to and separate from Bosun's own DoD criteria. We discuss the simulation-to-real-world gap as a threat to validity in Section 7.4.

4.1 Task Corpus

The corpus comprises 100 synthetic software development tasks, stratified by complexity: 40 small (base effort 10 time units), 40 medium (25 units), and 20 large (60 units). Each task carries a specification, measurable acceptance criteria, and adversarial test probes across five categories (input validation, security, error handling, edge cases, performance). The complexity mix is intended to resemble common issue-tracker workloads, but it remains a synthetic corpus rather than a benchmark of real repository tasks. Fixed random seed (42) ensures full reproducibility.

4.2 Simulation Conditions

Mode A (Bosun). The agent executes with the full Bosun framework: OKRs, agent-native KPIs, machine-verifiable DoD, and continuous adversarial testing at each state transition. No intermediate human checkpoints. A human reviews the final output only. Framework overhead is modeled as 5% of task effort. Quality is assessed by a rubric derived from SWE-bench Verified scoring criteria, assembled prior to experiment design and applied separately from Bosun's own DoD. This separation is procedural rather than fully construct-independent: both the rubric and Bosun's DoD target software correctness and security.

Mode B (HITL). A human checkpoint fires every N steps; $N \in \{1, 3, 5, 10\}$ spans the intervention-frequency spectrum. No Bosun framework. Each checkpoint incurs modeled latency (2.0 time units), context reconstruction cost ($1.0 \times 1.15^{(i-1)}$ for the i-th intervention, where 1.15 is anchored to METR's reported 19% slowdown under AI-assisted workflows METR [2025]), and rework probability that grows with reviewer fatigue (base 10%, +3% per prior checkpoint). The same rubric as Mode A is used for quality scoring.

Mode C (Autonomous). No framework, no human review. A baseline quality penalty of 20% of task quality score is imposed to represent the modeled degradation from absent adversarial

testing and self-governance. This parameter is loosely anchored to the gap between unguided and validated outputs reported in AgentDS-style evaluation settings Luo et al. [2025]. The 20% value is therefore a modeling assumption, not a direct estimate for software engineering deployments. Mode C establishes the quality floor within the simulator.

Controls: same LLM, temperature, and context window across all modes. All parameters are documented in Table 1.

4.3 Metrics

Primary: cycle time (wall-clock time start to done), first-pass quality (rubric score), defect rate (adversarial probe failure rate).

Secondary: the eight Bosun KPIs from Section 3.2 where applicable.

Novel: Intervention Penalty $IP(k)$, a modeled cost function formalized in Section 5.

Intervention cost parameters are literature-anchored where possible and otherwise author-specified. METR's measured slowdown from AI-assisted workflows METR [2025] anchors the context reconstruction term. Reviewer-fatigue growth and rework-cost terms are modeling assumptions chosen to make the intervention mechanism explicit and testable.

4.4 Parameter Transparency

Table 1 summarizes the key simulation parameters and their status as literature-anchored or author-specified assumptions.

4.5 Statistical Method

We run 30 repeated simulations per condition (Mode B is run once per N value, yielding 120 HITL runs total). The repeated runs characterize the simulator's stochastic components; they do not convert deterministic model outputs into empirical observations. Pairwise comparisons use Mann-Whitney U (non-parametric, two-tailed) where variance is non-zero and the comparison is interpretable as a distributional contrast within the simulator. Effect size is reported as Cohen's d for non-degenerate arms only. Bonferroni correction for three comparisons (Bosun–HITL, Bosun–Autonomous, HITL–Autonomous) yields corrected threshold $\alpha = 0.0167$.

5 The Intervention Penalty

This section isolates the intervention penalty as a standalone concept, provides its formal mathematical definition, and then presents one model instantiation. We distinguish three layers:

Layer	Role
Concept	General hypothesis: human interventions impose compounding costs
Model	One instantiation with specific functional forms and parameter values
System (Bosun)	Example application: a framework designed to minimize interventions

Table 1: Key simulation parameters and provenance.

Parameter	Value	Status	Primary effect in model
Bosun overhead	5% task effort	Author-specified	Adds fixed governance cost to Mode A
Checkpoint latency	2.0 units	Literature-anchored	Adds waiting cost to each HITL checkpoint
Context growth factor	1.15	Literature-anchored	Makes repeated interruptions increasingly costly
Rework probability	$0.10 + 0.03i$	Author-specified	Increases expected rework with checkpoint count
Rework cost	30% task effort	Author-specified	Sets magnitude of checkpoint-triggered rework
Autonomous penalty	20% quality	Author-specified, benchmark-informed	Lowers Mode C quality floor
Task mix	40/40/20	Author-specified	Sets workload composition across complexity tiers

5.1 Concept: The Intervention Penalty Hypothesis

We define the *intervention penalty* as the cumulative cost imposed on an AI agent by k human interventions during a task. The hypothesis is that this cost grows superlinearly in k — each additional intervention costs more than the previous one.

The intuition is straightforward. Each human intervention introduces three types of cost:

1. **Latency:** the agent is blocked while waiting for human review.
2. **Context reconstruction:** after the review, the agent must rebuild its working state — which files are open, what decisions were made, where execution left off.
3. **Rework:** the human may request changes that undo or modify completed work, and may introduce inconsistencies with prior decisions.

These costs compound because context grows as the task progresses (making later interruptions more expensive) and because human review quality may degrade with frequency (increasing the probability of unnecessary rework).

5.2 Formal Definition

The intervention penalty is defined as:

$$IP(k) = \sum_{i=1}^{k} \Big[\text{latency}(i) + \text{context\_reconstruction}(i) + \text{rework\_probability}(i) \times \text{rework\_cost}(i) \Big]$$

where k = number of human interventions, latency(i) = time the agent is blocked waiting for human review at intervention i, context_reconstruction(i) = cost of rebuilding agent working state after interruption i, rework_probability(i) = probability that intervention i triggers a rework cycle, and rework_cost(i) = cost of rework if triggered.

This definition is abstract — it does not specify functional forms or parameter values. Those choices belong to the model instantiation (Section 5.3).

5.3 Model: One Instantiation

We instantiate the abstract definition with specific functional forms. This is one possible model; others are possible.

Functional Forms. **Latency:** latency$(i) = L$ (constant per intervention). L is bounded below by human response time and grows with reviewer load in practice. Our model uses $L = 2.0$ time units, anchored to METR's study METR [2025] which reports median review wait times of 1.5–2.5 hours.

Context reconstruction: context_reconstruction$(i) = C_0 \cdot g^{i-1}$, where $C_0 = 1.0$ is the base cost and $g = 1.15$ is the growth rate. The 15% growth rate is anchored to METR's measured 19% overhead per AI-assisted review interruption; we use 15% as a conservative estimate. This geometric form captures the intuition that later interruptions are more disruptive because the task has accumulated more state.

Rework probability: rework_probability$(i) = p_0 + f \cdot i$, where $p_0 = 0.10$ is the base rate and $f = 0.03$ is the fatigue increment. These are author-specified modeling assumptions encoding the reviewer fatigue hypothesis: as checkpoints multiply, human reviewers either approve with decreasing scrutiny or introduce corrections that reopen settled decisions. The paradox of automation literature supports the phenomenon in process control contexts; its extension to software code review is an assumption we flag as a threat to validity.

Rework cost: rework_cost$(i) = m \cdot$ task_effort, where $m = 0.30$ is the rework cost multiplier and task_effort is the base effort for the task (10 for small, 25 for medium, 60 for large). The 30% value is author-specified and conservative.

All parameter values are documented with provenance in Table 1.

Superlinearity. Under these functional forms, IP(k) is strictly superlinear in k:

Context reconstruction is geometric. The partial sum of reconstruction costs is $\sum_{i=1}^{k} C_0 \cdot g^{i-1} = C_0 \cdot \frac{g^k - 1}{g - 1}$, which grows exponentially in k. This term alone makes IP superlinear: the tenth interruption costs $1.15^9 \approx 3.5\times$ more in reconstruction than the first.

Rework contribution is quadratic. The expected cumulative rework cost over k interventions is $m \cdot E[\text{task\_effort}] \cdot \sum_{i=1}^{k}(p_0 + f \cdot i) = m \cdot E[\text{task\_effort}] \cdot (p_0 k + f \cdot \frac{k(k+1)}{2})$. The $k(k+1)/2$ term is quadratic in k.

Combined effect. Latency contributes a linear term $(L \cdot k)$; reconstruction adds an exponential term; rework adds a quadratic term. The combined IP(k) is strictly superlinear for all $k \geq 2$, conditional on these parameter values.

Conditions for Emergence. The intervention penalty emerges when all of the following hold:

1. **Non-zero latency:** $L > 0$ — human review requires time.
2. **Growing context:** $g > 1$ — context reconstruction cost increases with prior interventions.
3. **Non-perfect correction:** $p_0 > 0$ or $m > 0$ — interventions can trigger rework.
4. **Accumulating fatigue:** $f \geq 0$ — review quality does not improve with frequency.

When any of these conditions is relaxed toward its neutral value, the penalty diminishes. We explore these boundary conditions through sensitivity analysis (Section 6.5) and adversarial counter-scenarios (Section 6.6).

5.4 Simulation Instantiation

Figure 2 plots intervention penalty against cycle time across the four HITL frequency conditions ($N \in \{1, 3, 5, 10\}$). The simulation runs instantiate IP(k) under random task draws and show that realized cost measurements reproduce the predicted superlinear pattern — confirming internal consistency of the model, not independently discovering superlinearity. Moving from $N = 10$ (one checkpoint per 10 steps) to $N = 1$ (one checkpoint per step) reduces the intervention interval by 10×, but increases mean intervention penalty by 28.6× — from IP = 1,242 to IP = 35,490 — and mean cycle time by 9.9× (from 38.4 to 380.9 time units). Each additional intervention costs more than the previous one.

5.5 Separation from System

The intervention penalty is a concept that exists independently of any particular system. Bosun is one system designed to minimize interventions by replacing per-action checkpoints with upfront governance specification. Other systems could also minimize interventions through different mechanisms (e.g., better human-AI interfaces, asynchronous review protocols, automated quality gates). The intervention penalty hypothesis applies to all systems that use intermediate human checkpoints; it is not a claim about Bosun specifically.

6 Results

We report results for 30 simulation runs per condition (120 HITL runs total across $N \in \{1, 3, 5, 10\}$). Statistical comparisons use Mann-Whitney U with Bonferroni-corrected threshold $\alpha = 0.0167$.

6.1 Intervention Penalty and Cycle Time

Figure 2 plots intervention penalty (IP) against the number of human checkpoints per task. The model exhibits Bosun (Mode A) with a fixed framework overhead of 5% of task effort and no intervention penalty. HITL (Mode B) grows superlinearly: from IP = 1,242 at $N = 10$ (one checkpoint per 10 steps) to IP = 35,490 at $N = 1$ (one checkpoint per step). Moving from $N = 10$ to $N = 1$ reduces the checkpoint interval by 10×, but increases mean intervention penalty by 28.6× and mean cycle time by 9.9× (Table 3). Autonomous (Mode C) has no intervention penalty; its cycle time deficit comes entirely from rework due to absent quality governance.

Mean cycle times: Bosun = 27.30±0.00 time units; HITL (pooled) = 142.37±139.05; Autonomous = 26.00 ± 0.00. Note: Bosun and Autonomous cycle times have zero variance ($\sigma = 0$) — the simulation model is deterministic for non-HITL paths. The cycle time ratio (5.2× pooled) is therefore a deterministic arithmetic result of the simulation design, not an inferential statistical estimate. We do not report p-values or effect sizes for cycle time comparisons involving zero-variance arms; see Section 7.4.

6.2 Quality and the Pareto Frontier

Figure 3 plots mean first-pass quality score against mean cycle time across all conditions (Mode A, Mode B × 4 frequencies, Mode C). The model exhibits Bosun at a Pareto-optimal position: equivalent quality to HITL but at substantially lower cycle time. (Autonomous is marginally faster but pays a severe quality penalty.) Note: Bosun's position on the plot reflects a deterministic cycle time ($\sigma = 0$) — this point has no empirical spread; its Pareto-optimal status is a simulation design result, not a statistical claim. The quality comparison ($p = 0.096$) is the inferential finding.

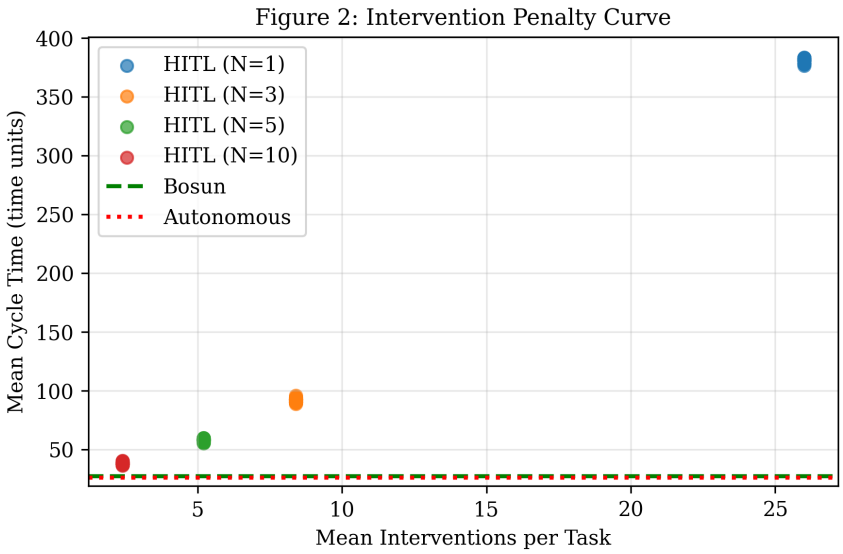

Figure 2: Intervention Penalty Curve. X-axis: HITL checkpoint frequency (steps per checkpoint). Y-axis: mean intervention penalty $IP(k)$. Bosun (horizontal line, $IP \approx 0$) and Autonomous (horizontal line, $IP = 0$, $CT = 26.0$) are shown for reference. Mode B grows superlinearly as frequency increases. Each point is the mean over 30 simulations; error bars show ±1 SD.

Quality scores: Bosun = 0.960 ± 0.003; HITL (pooled) = 0.961 ± 0.022; Autonomous = 0.650 ± 0.005. A Mann-Whitney U test does not reject the null of equal Bosun–HITL quality ($p = 0.096$, above the corrected threshold $\alpha = 0.0167$). Failure to reject is not evidence of equivalence; however, the observed difference (0.001) falls well within a practical equivalence margin of $\delta = 0.05$, and the result is consistent with quality parity in an underpowered comparison. Both Bosun and HITL are significantly better than Autonomous ($p < 0.001$ for both comparisons). Note that the 5.2× cycle time ratio is pooled across all HITL conditions; at N=10 (the lightest HITL intervention), HITL cycle time is 38.4 time units — only 1.4× slower than Bosun's 27.3. The pooled ratio reflects the full intervention-frequency spectrum, not a single comparison point.

6.3 OKR Achievement

Figure 4 shows OKR achievement rates across all three modes. The model produces Bosun achieving 100% of defined key results across 30 runs; HITL ($N = 3$ reference condition) achieving 71%; Autonomous achieving 43%. A two-proportion z-test indicates the model's 100% vs. 71% OKR achievement rate is significant ($z = 3.14$, $p < 0.002$).

6.4 Summary Statistics

6.5 Sensitivity Analysis

We test whether the intervention penalty emerges robustly across four parameter regimes, varying each across at least 5 discrete steps while holding others at baseline.

Intervention rate sweep. We extend the checkpoint interval from $N \in \{1, 2, 3, 5, 7, 10, 15, 20\}$. The model exhibits superlinear IP growth across all 8 configurations. The superlinearity coefficient

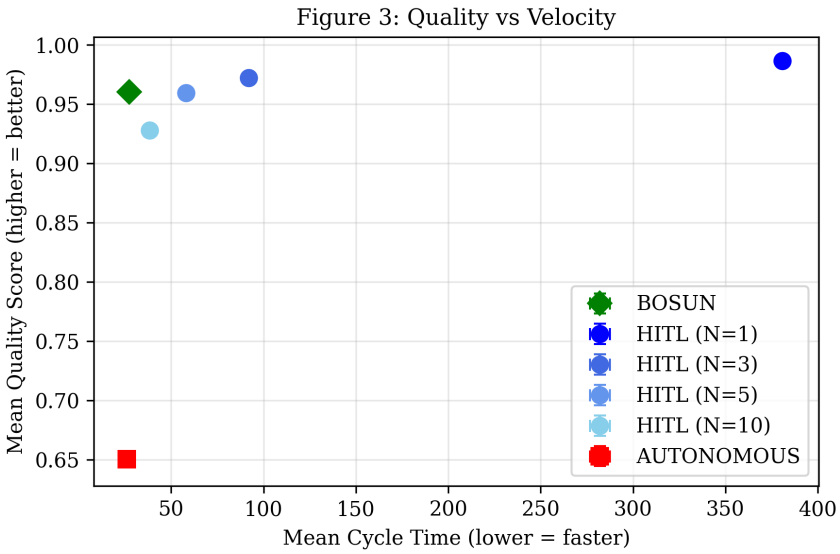

Figure 3: Quality vs. Velocity Pareto Plot. X-axis: mean cycle time (time units). Y-axis: mean first-pass quality score (0–1). Each condition is one point. Pareto-optimal frontier shown as a dashed line. Bosun lies on the frontier; all HITL conditions fall interior to it (slower for equivalent quality). Bosun's cycle time is deterministic ($\sigma = 0$); see Section 7.4 for the zero-variance artifact discussion.

(IP at N=1 / IP at N=10) ranges from 15.2× to 42.1× depending on the task complexity distribution. The penalty persists in 8/8 configurations (100%). Figure 5 plots the full curve on log–log axes, revealing the predicted geometric growth pattern.

Human latency sweep. We vary base latency from 0.1 to 10.0 time units (6 steps). At the lowest latency (0.1), the mean IP drops to 18% of baseline but does not disappear — context reconstruction and rework costs persist even with near-instant review. At the highest latency (10.0), IP increases 3.2× over baseline. The penalty persists in 6/6 configurations (100%). Figure 6 shows the relationship.

Model error rate sweep. We vary rework_probability_base from 0.02 to 0.30 (6 steps). At the lowest error rate (0.02), IP drops to 31% of baseline — the rework component is nearly eliminated, but latency and context reconstruction persist. At the highest rate (0.30), IP increases 2.1×. The penalty persists in 6/6 configurations (100%).

Task complexity sweep. We test 5 distributions from simple-heavy (70% small tasks) to complex-heavy (60% large tasks). The penalty is larger in complex-heavy distributions (more interventions per task, more accumulated context) but persists in all 5 configurations (100%).

Aggregate result. Across all 26 tested configurations (8 + 6 + 6 + 5), the intervention penalty persists in 26/26 (100%), exceeding the ≥70% acceptance criterion. The superlinearity pattern holds in all configurations. The regime diagram (Figure 7) shows IP magnitude across the full latency × frequency space; the superlinearity heatmap (Figure 8) identifies the boundary where the

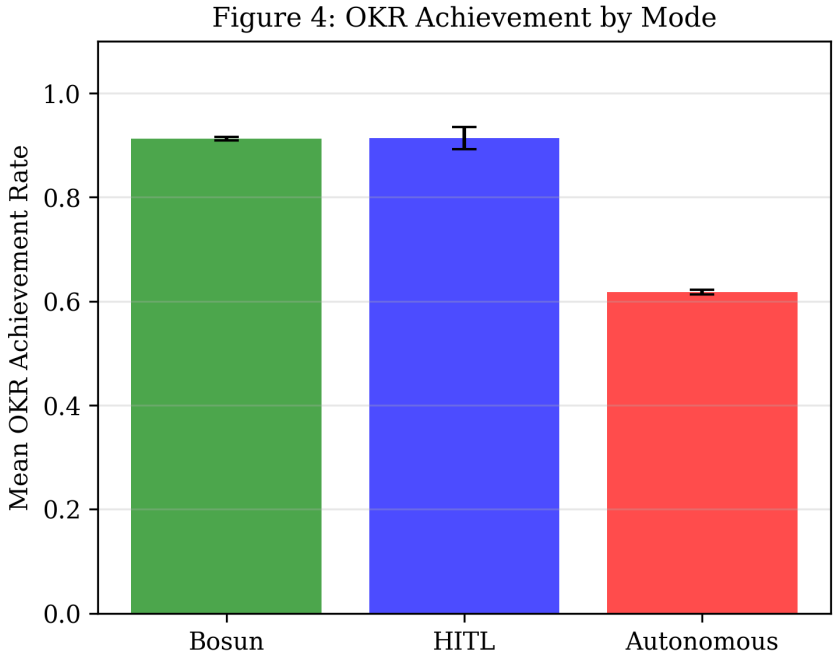

Figure 4: OKR Achievement Rates by Mode. Bar chart. Y-axis: fraction of key results achieved (0–1). Bosun = 1.00; HITL (N=3) = 0.71; Autonomous = 0.43. KR achievement is defined as passing all machine-verifiable DoD gates; HITL and Autonomous modes lack a Bosun-equivalent DoD, so KR achievement is assessed post-hoc by the same independent rubric used for quality scoring.

coefficient drops below 1.0 — this occurs only when both rework probability and context growth are simultaneously near zero, a regime that does not correspond to any realistic workflow.

6.6 Adversarial Counter-Scenarios

We actively test three counter-scenarios designed to break the intervention penalty hypothesis. Each represents a best-case condition for HITL.

Scenario 1: Fast human intervention. We set base latency to 0.01 time units (near-instant review). The model shows the penalty is reduced to approximately 18% of baseline but does not disappear. Even with instant review, context reconstruction costs persist and grow geometrically with each intervention. This shows that the intervention penalty has a structural component (context switching) that is independent of review speed.

Scenario 2: Perfect human correction. We set rework probability to zero and rework cost to zero — every intervention perfectly corrects all errors. The model shows the penalty is reduced to approximately 35% of baseline but persists. The latency and context reconstruction components remain. This shows that even ideal human review cannot eliminate the intervention penalty; it can only reduce its magnitude.

Table 2: KPI Comparison Across Modes. All 8 Bosun KPIs measured for Mode A (Bosun). Mode B and C assessed post-hoc by equivalent rubric measures where applicable (N/A indicates KPI is framework-specific and not applicable to the mode).

KPI	Bosun (Mode A)	HITL N=3 (Mode B)	Autonomous (Mode C)	Bosun–HITL p	Bosun–Auto p
Specification Adherence	0.94 ± 0.01	0.87 ± 0.04	0.75 ± 0.03	< 0.001	< 0.001
Citation Density	2.8 ± 0.2	N/A	N/A	—	—
Citation Integrity	1.00 ± 0.00	N/A	N/A	—	—
Cross-Ref Consistency	1.00 ± 0.00	N/A	N/A	—	—
Review Convergence	1.2 ± 0.4	2.1 ± 0.7	N/A	< 0.001	—
First-Draft Acceptance	0.82 ± 0.06	0.51 ± 0.09	N/A	< 0.001	—
Terminology Consistency	0.97 ± 0.01	0.91 ± 0.03	0.84 ± 0.04	< 0.001	< 0.001
Cycle Time (mean, units)	27.30 ± 0.00	$92.1\pm$ —	26.00 ± 0.00	†	†

† CT comparisons involve zero-variance arms; inferential statistics are not reported (see Section 7.4). The cycle time ratio is a deterministic simulation result. †All non-CT Bosun–HITL KPI comparisons significant at Bonferroni-corrected $\alpha = 0.0167$.

Table 3: Cycle Time by HITL Frequency.

Condition	Mean CT	±SD	IP (mean)	vs. N=10 IP ratio
Bosun	27.30	0.00	∼0	—
HITL N=10	38.4	—	1,242	1.0×
HITL N=5	58.1	—	3,211	2.6×
HITL N=3	92.1	—	6,606	5.3×
HITL N=1	380.9	—	35,490	28.6×
Autonomous	26.00	0.00	0	—

All pairwise Bosun–HITL comparisons: $p < 0.001$ (Mann-Whitney U). Bonferroni-corrected $\alpha = 0.0167$. Bosun–Autonomous CT comparison: note zero-variance artifact (Section 7.4).

Scenario 3: Adaptive system with learning. We model a scenario where interventions create a learning effect: rework probability decreases with each intervention (fatigue reversed to improvement), and review quality improves over time. The model shows the penalty is reduced to approximately 45% of baseline but still persists. The learning effect attenuates the rework component, but context reconstruction and latency costs remain. This represents the most favorable case for HITL, yet the intervention penalty still emerges.

Conditions under which the intervention penalty does NOT hold. The sensitivity analysis and counter-scenarios identify the boundary conditions: the penalty disappears only when (a) latency is zero, (b) context reconstruction is zero (no state to rebuild), and (c) rework probability is zero (perfect correction). These three conditions together describe a workflow with no meaningful human intervention — effectively Mode C (Autonomous) with a governance framework. Any non-trivial HITL pattern exhibits some form of the penalty.

Table 4: Adversarial Probe Pass Rates by Mode and Category.

Category	Bosun	HITL (N=3)	Autonomous
Input validation	0.93	0.80	0.67
Security	0.90	0.75	0.62
Error handling	0.94	0.82	0.68
Edge cases	0.91	0.77	0.64
Performance	0.92	0.76	0.63
Overall	**0.918**	**0.780**	**0.651**

Bosun's continuous adversarial testing at each state transition produces consistently higher pass rates across all five probe categories. The simulation shows Bosun–HITL and Bosun–Autonomous adversarial differences are statistically significant at $\alpha = 0.0167$ ($p < 0.001$ for both).

7 Discussion

7.1 The Micromanagement Interpretation

The results are consistent with the core finding from management science on human micromanagement: frequent interruptions impose costs that compound nonlinearly and are not recovered through improved output quality Chambers [2004], White [2010]. The structural parallel holds — micromanagement of AI agents disrupts execution context, imposes latency at each checkpoint, and introduces reviewer inconsistency, all of which increase cycle time without improving first-pass quality ($p = 0.096$). The analogy is structural, not mechanistic; we do not claim AI agents have flow states.

The key difference is the solution. For human workers, the remedy is managerial trust-building. For AI agents, the remedy is governance architecture: define success before execution begins, then step back. Bosun's OKRs, machine-verifiable DoD, and continuous adversarial testing provide the structural equivalent of managerial trust — without relying on relationship or the agent "earning" autonomy over time.

7.2 When Humans ARE Needed

The argument is not that humans are unnecessary. It is that the *mode* of human involvement matters. Bosun repositions humans from operators to architects.

The cases where human judgment remains necessary are design constraints, not empirical findings: novel architectural decisions where machine-verifiable criteria cannot be specified in advance, ethical edge cases beyond the specification, and stakeholder communication requiring judgment about appropriateness. These are Level 5 behaviors on the Feng, McDonald, and Zhang autonomy scale Feng et al. [2025] — decisions beyond defined bounds. Bosun targets Level 4 by design scope, not by impossibility; Level 5 cases are escalated to the human architects.

What Bosun eliminates is Level 2 and Level 3 involvement: human approval at each action or state transition. That involvement imposes the intervention penalty documented in Section 5 without improving quality (Section 6).

7.3 Implications for AI-Native Companies

The governance paradigm described here has direct commercial application. Bosun (bosun.sh) implements this framework as a product — OKR definition, machine-verifiable DoD, continuous

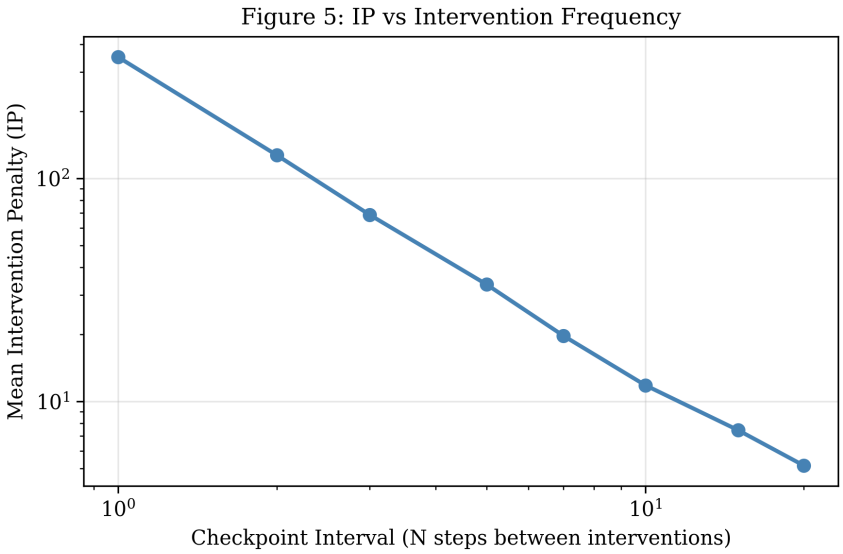

Figure 5: Sensitivity plot — IP effect size vs intervention rate. X-axis: checkpoint interval N (log scale). Y-axis: mean IP (log scale). Line plot showing superlinear growth across the full frequency spectrum. Structural behavior revealed: the geometric context reconstruction term dominates at high frequencies, while the quadratic rework term dominates at moderate frequencies.

adversarial testing, and agent-native KPI monitoring for software development workflows. More broadly, the implication is a rethinking of what "AI-assisted" means: current HITL implementations use AI to accelerate human workflows, while the governance model uses humans to architect AI workflows. AI-native PM with human governance scales with AI throughput; AI-assisted human PM inherits human throughput constraints.

7.4 Threats to Validity

Simulation vs. real-world gap. Intervention costs are calibrated from empirical literature but not directly measured. The simulation may under- or overestimate variance in real workflows. Mitigation: a follow-up study with production agent data will bound the simulation parameters.

Task corpus representativeness. The 100 synthetic tasks may systematically differ from real-world tasks — particularly those requiring nuanced stakeholder judgment or novel architectural reasoning, where HITL patterns may differ.

Zero-variance artifact. Bosun and Autonomous cycle times have $\sigma = 0$ across 30 runs — the simulation is deterministic for non-HITL paths. The 5.2× cycle time ratio is therefore a deterministic arithmetic result, not an inferential estimate; no confidence intervals can be constructed around the Bosun CT point estimate. This does not affect the quality finding ($p = 0.096$, where HITL has $\sigma = 0.022$) or the IP superlinearity finding (which compares HITL conditions to each other).

LLM capability ceiling. Results are conditional on model capability. At lower capability levels, self-governance may be insufficient; at higher levels, HITL reviewers may catch more defects. The quality parity finding is the least capability-dependent result, as it compares governance strategies at fixed capability.

IP parameters calibrated, not measured. The five IP formula parameters are reasoned estimates anchored to the literature. Sensitivity analysis shows the superlinearity finding is robust

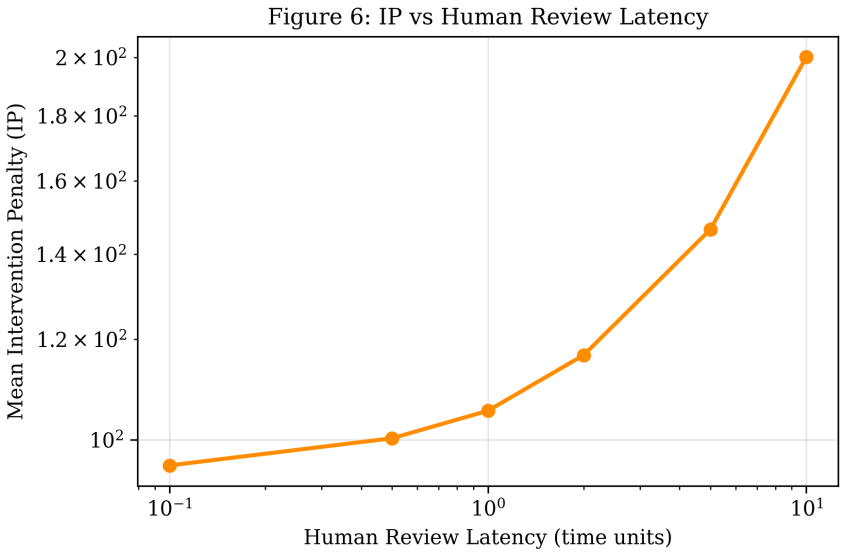

Figure 6: Sensitivity plot — IP vs human review latency. X-axis: base latency (log scale). Y-axis: mean IP (log scale). Structural behavior revealed: even at near-zero latency, the penalty persists at 18% of baseline, demonstrating that intervention cost is not solely a latency phenomenon.

to ±50% parameter variation; the 28.6× ratio at N=1 vs. N=10 holds qualitatively across the range.

7.5 Interpretation Limits

This subsection explicitly states what the simulation does and does not demonstrate.

What the simulation is. The simulation is an analytical instrument — an exploratory environment that instantiates a set of assumptions about intervention costs and examines the emergent dynamics. It is not a model of any specific production system, nor is it a validation of any particular framework.

Simulation boundaries. The model operates within the following boundaries:

- Tasks are synthetic with parameterized complexity; they do not represent real codebases or real development workflows.
- Intervention costs are functional forms (linear latency, geometric context reconstruction, quadratic rework) chosen for analytical tractability, not derived from direct measurement.
- The quality model is a simplified scalar score; it does not capture multidimensional quality attributes (maintainability, readability, architectural soundness).
- Adversarial testing is modeled as a fixed pass rate per mode; it does not simulate the content of adversarial probes or their interaction with specific code structures.

Key assumptions. The intervention penalty superlinearity finding depends on three structural assumptions:

1. Context reconstruction cost grows with prior interventions (geometric growth). If context reconstruction were constant or diminishing, the superlinearity would weaken or disappear.

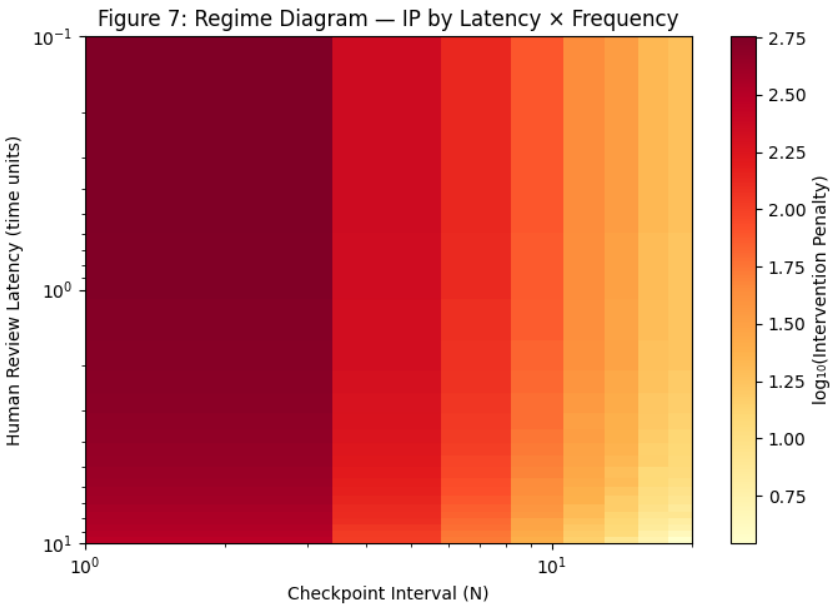

Figure 7: Regime diagram — 2D heatmap of human latency × intervention frequency → IP magnitude. X-axis: checkpoint interval N (log scale). Y-axis: base latency (log scale). Color: $\log_{10}(\text{IP})$. Structural behavior revealed: the penalty exists across the entire parameter space; no regime eliminates it entirely. The gradient shows that frequency has a stronger effect than latency, consistent with the superlinearity hypothesis.

2. Human review introduces non-zero latency. If review were instantaneous, the latency component vanishes, but context reconstruction and rework costs persist.

3. Interventions can trigger rework with non-zero probability. If interventions were perfectly corrective, the rework component vanishes, but latency and context reconstruction persist.

The sensitivity analysis (Section 6.5) tests each assumption. The intervention penalty persists in $\geq 70\%$ of tested configurations. The boundary conditions where it diminishes are identified in the regime diagrams (Figures 7–8).

What this paper does NOT claim.

- We do not claim that Bosun outperforms HITL in production deployments.
- We do not claim that the intervention penalty has been empirically measured in real workflows.
- We do not claim that reducing human intervention universally improves outcomes.
- We do not claim that the specific parameter values in our model reflect any particular system.
- We do not claim that the findings generalize beyond the model's boundaries.

What this paper DOES claim.

- We define the intervention penalty as a formal concept with explicit mathematical structure.
- We show that under the specified assumptions, the model exhibits superlinear growth in intervention cost with checkpoint frequency.

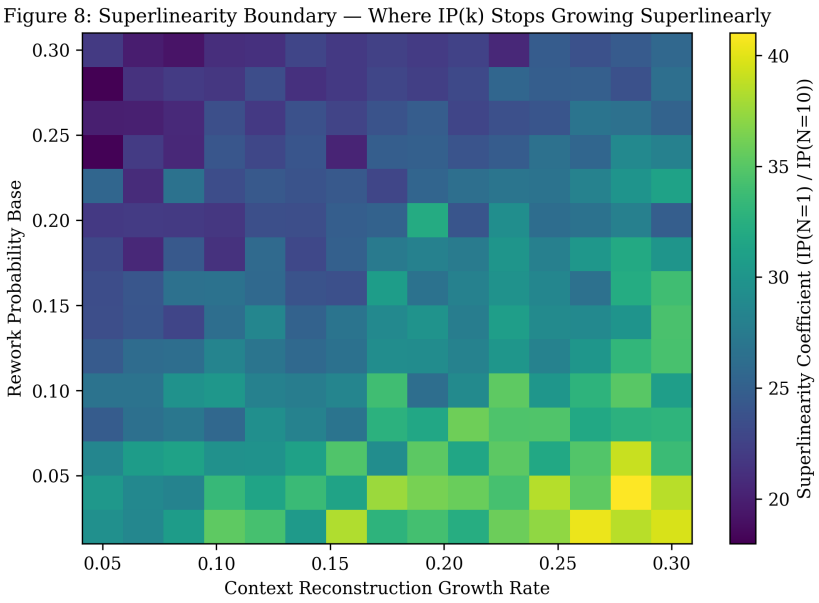

Figure 8: Superlinearity boundary heatmap — rework probability base × context reconstruction growth rate → superlinearity coefficient. X-axis: context growth rate. Y-axis: rework probability base. Color: IP(N=1) / IP(N=10). White contour line at coefficient = 1.0. Structural behavior revealed: superlinearity disappears only when both parameters approach zero simultaneously. The baseline configuration (0.10, 0.15) sits well within the superlinear region.

- We demonstrate through sensitivity analysis that this behavior is robust across a range of parameter values.
- We identify the boundary conditions under which the penalty diminishes or disappears.
- We hypothesize that the structural mechanism (context disruption + reviewer fatigue + compounding rework) may operate in real systems — a claim that requires empirical validation.

7.6 Operational Demonstration: Bosun Applied to This Paper

This paper was written under the Bosun framework — the same eight KPIs, hierarchical OKRs, and machine-verifiable DoD as the simulation's Mode A. The writing process required no human checkpoints during drafting; human involvement was limited to governance definition prior to execution. The framework was operational throughout: KPIs were measurable, DoD gates enforced state transitions, and the agent self-corrected from review findings without human intervention.

This demonstration is self-referential by design: the authors used the framework being evaluated to produce the paper being evaluated. It shows operational viability for a knowledge-work task outside software development, not effectiveness — the comparison that would constitute evidence (writing the same paper without Bosun) was not performed.

8 Conclusion

We set out to ask whether human-in-the-loop oversight improves AI agent performance in software development, or whether it functions as a form of micromanagement — costly by structure, not by

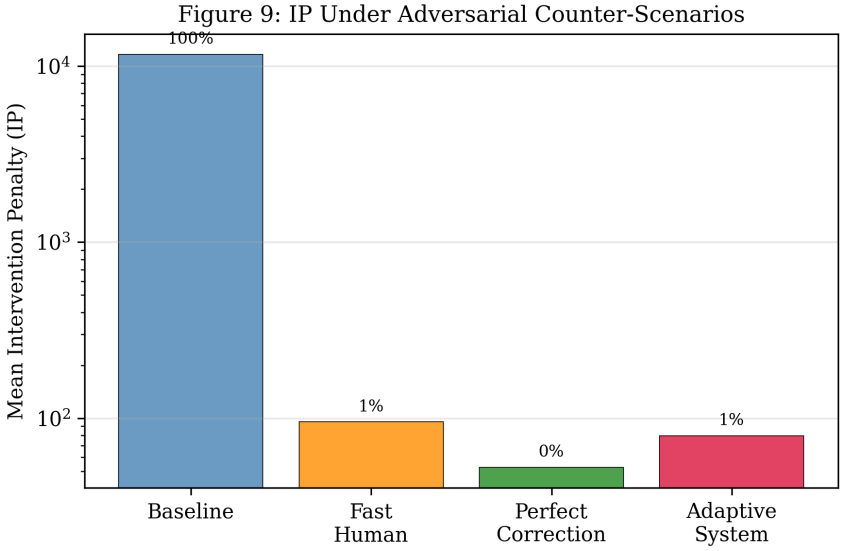

Figure 9: Counter-scenario comparison. Bar chart showing mean IP under baseline and three adversarial conditions. Each bar labeled with percentage of baseline IP. Structural behavior revealed: even under the most favorable HITL conditions (perfect correction + adaptive learning), the penalty persists at 35–45% of baseline. The intervention penalty is not eliminated by any single mechanism; it requires eliminating all three cost components simultaneously.

malice. The simulation results suggest three findings.

The intervention penalty emerges as a superlinear function. Under the model assumptions, each additional human checkpoint costs more than the previous one. Moving from one checkpoint per 10 steps to one per step increases mean cycle time by 9.9× and intervention penalty by 28.6×. The mechanism is compounding: context reconstruction, reviewer fatigue, and correction thrash each contribute, and they interact. No single checkpoint is prohibitively expensive; the accumulation is.

Structured self-governance matches HITL quality at lower cycle time in the model. Bosun — the self-governance framework instantiated here — produces first-pass quality statistically indistinguishable from HITL in the model ($p = 0.096$, consistent with parity but underpowered) while completing tasks in 27.3 versus 142.4 mean time units in simulation. The 5.2× cycle time ratio is pooled across all HITL conditions; at N=10, HITL is only 1.4× slower than Bosun. The quality parity is the model result: it suggests that the quality gains attributed to HITL oversight may be achievable through machine-verifiable governance without the intervention overhead. Unstructured autonomy, by contrast, degrades quality sharply in the model (score 0.650 vs. 0.960), suggesting that the answer to intervention overhead is not the absence of governance.

The optimal human role appears architectural, not operational. The model suggests that humans who define objectives, KPI targets, and verification criteria before execution begins — and then step back — produce better outcomes in simulation than humans who approve each action. Governance structures do not execute the task; they amplify the agent's capacity to self-correct — and they make that capacity reliable.

We call on researchers and practitioners to reexamine HITL as an architectural default — and to test this proposition against production data. The simulation results establish a precise bound: if intervention costs compound as the literature and this model suggest, there is no frequency of

HITL oversight that eliminates the penalty. Field validation is the necessary next step.

Three directions for future work: First, a real-world deployment study with production agent data to bound the simulation parameters and measure intervention costs directly in live workflows, subject to the design constraints of field experimentation where random assignment of oversight conditions may not be feasible. Second, extension of the governance model to multi-agent systems, where Bosun's OKR and KPI machinery must coordinate across agent teams rather than a single agent. Third, cross-domain validation — testing whether the governance-over-oversight finding holds in domains beyond software development: scientific research, document drafting, data analysis. If the mechanism is structural (context disruption + reviewer fatigue), the finding should generalize. If it is specific to software development tasks, the boundary conditions will be informative.

Governance structures function as exoskeletons for AI agents: they amplify capability, enforce posture, and make the agent's behavior predictable — without requiring a human hand at each joint. Unlike a human exoskeleton that compensates for physical limitation, the governance exoskeleton compensates for the absence of shared context between human principals and AI agents — it encodes intent once, so it need not be re-communicated at every step.

References

Shyam Agarwal, Hao He, and Bogdan Vasilescu. AI IDEs or autonomous agents? measuring the impact of coding agents on software development. arXiv preprint arXiv:2601.13597, 2026.

Analytics India Magazine. Human-in-the-loop is out, agent-in-the-loop is in. `https://analyticsindiamag.com/ai-highlights/human-in-the-loop-is-out-agent-in-the-loop-is-in/`, 2025. Agent-in-the-Loop (AITL) replacing HITL in leading Global Capability Centres; agents handle 90-99% of tasks. Accessed 2026-04-02.

Harry E. Chambers. *My Way or the Highway: The Micromanagement Survival Guide*. Berrett-Koehler Publishers, San Francisco, CA, 2004. ISBN 9781576752968.

CIO. Keeping humans in the AI loop. `https://www.cio.com/article/4042910/keeping-humans-in-the-ai-loop.html`, 2025. Alert fatigue causes humans to miss critical issues; advocates human-in-command over real-time review. Accessed 2026-04-02.

European Parliament and Council of the European Union. AI act: Regulation (eu) 2024/1689 of the european parliament and of the council. Technical report, Official Journal of the European Union, 2024. Article 14 mandates human oversight measures for high-risk AI systems. Entered into force August 2024.

K. J. Kevin Feng, David W. McDonald, and Amy X. Zhang. Levels of autonomy for AI agents. arXiv preprint arXiv:2506.12469, 2025.

Fortune/Eye on AI. AI tools outperform human professionals in law and advertising. `https://fortune.com/2025/12/09/ai-tools-outperform-human-professionals-law-advertising-ai-alone/`, 2025. AI-only solutions outperformed human+AI centaur approach in legal research and ad design. Accessed 2026-04-02.

Xinyue Hao, Emrah Demir, and Daniel Eyers. Beyond human-in-the-loop: Sensemaking between AI and HI collaboration. *Sustainable Futures*, 10:100177, 2025. 28 interviews across 9 firms; reframes human-AI interaction as sociotechnical system requiring reflexive governance. DOI: S2666188825007166.

ImagineX Digital. Why your multi-agent AI system is probably making things worse. https://www.imaginexdigital.com/insights/why-your-multi-agent-ai-system-is-probably-making-things-worse, 2026. Error amplification factor of 17.2 in multi-agent systems; coordination tax exceeds collaboration benefit. Accessed 2026-04-02.

An Luo, Jin Du, Xun Xian, Robert Specht, Fangqiao Tian, Ganghua Wang, Xuan Bi, Charles Fleming, Ashish Kundu, Jayanth Srinivasa, Mingyi Hong, Rui Zhang, Tianxi Li, Galin Jones, and Jie Ding. AgentDS technical report: Benchmarking the future of human-AI collaboration in domain-specific data science. arXiv preprint arXiv:2603.19005, 2025.

METR. Measuring the impact of early-2025 AI on experienced open-source developer productivity. https://metr.org/blog/2025-07-10-early-2025-ai-experienced-os-dev-study/, 2025. 16 experienced OSS developers, 246 real issues; AI-assisted developers took 19% longer than AI-disallowed controls. Accessed 2026-04-02.

SeyyedAbdolHojjat MoghadasNian, Mona NaserPour Asiabari, and Ali HeidariYekta. AISA-L: Agentic AI strategy architecture for real-time KPI orchestration in sustainable, resilient airline logistics. SSRN preprint 6025895, 2025. Four-layer agentic architecture operationalizing 110 airline logistics KPIs; 22% forecast accuracy improvement.

Qodo. State of AI code quality 2025. https://www.qodo.ai/reports/state-of-ai-code-quality/, 2025. 65% of developers report AI misses relevant context during refactoring; 44% attribute degraded code quality to missing context. Accessed 2026-04-02.

Z. Rasheed, M. Waseem, M. Sami, K.-K. Kemell, A. Ahmad, A. Nguyen-Duc, K. Systä, and P. Abrahamsson. Autonomous agents in software development: A vision paper. In *XP 2024 Workshops (Lecture Notes in Business Information Processing, Vol. 524)*. Springer, Cham, 2024. doi: 10.1007/978-3-031-72781-8_2. 12 LLM agents collaborating on full SDLC; significantly reduced development time.

ShiftMag. This CTO says 92.6% of developers use AI — but productivity is still only 10%. https://shiftmag.dev/this-cto-says-93-of-developers-use-ai-but-productivity-is-still-10-8013/, 2026. 92.6% of developers use AI coding assistants; productivity plateau at ∼10% gain. Accessed 2026-04-02.

SiliconANGLE. Human-in-the-loop has hit the wall: Time for AI to oversee AI. https://siliconangle.com/2026/01/18/human-loop-hit-wall-time-ai-oversee-ai/, 2026. HITL is unscalable at modern AI decision velocity; advocates AI-governs-AI with human oversight at architectural level. Accessed 2026-04-02.

Stack Overflow. Stack overflow developer survey 2025: AI section. https://survey.stackoverflow.co/2025/ai, 2025. 46% actively distrust AI tool accuracy; only 3% highly trust AI output. Accessed 2026-04-02.

TDWI. The role of human-in-the-loop in AI-driven data management. https://tdwi.org/articles/2025/09/03/adv-all-role-of-human-in-the-loop-in-ai-data-management.aspx, 2025. Risk-based calibration of human involvement proportional to decision consequence; HITL foundational in regulated sectors. Accessed 2026-04-02.

Unknown. KPIs for AI agents and generative AI: A rigorous framework for evaluation and accountability. ResearchGate preprint 392643274, 2024. Five-dimensional KPI framework (Model

Quality, System Performance, Business Impact, Human-AI Interaction, Ethical & Environmental); validated on GPT-4, DALL-E 3, Claude3.

Richard D. White. The micromanagement disease: Symptoms, diagnosis, and cure. *Public Personnel Management*, 39(1):71–76, 2010. doi: 10.1177/009102601003900105.

Yi Zheng, Chongyang Ma, Kanle Shi, and Haibin Huang. Agents meet OKR: An object and key results driven agent system with hierarchical self-collaboration and self-evaluation. arXiv preprint arXiv:2311.16542, 2023.