A Partner, Not a Tool

Context, Friction, and Responsibility in AI-Assisted Work

Abstract

This paper argues that long-running AI-assisted work is not only a matter of prompting, but of handoff discipline. The ordinary tool metaphor is no longer sufficient for work that depends on context, correction, continuity, retrieval, authority boundaries, and recovery from drift. The claim is not that conversational AI systems are persons, employees, moral equals, or legal subjects. Rather, "partner" is used as a methodological stance: a disciplined way of preserving the process that makes outputs inspectable, correctable, and reusable. This paper extends the prior methodology introduced in "Development Methodology of a Code-Illiterate User," which framed machine-readable context as part of reproducible AI-assisted system development without formal software training. Here, that prior context-method work is advanced through the explicit naming and demonstration of batons as provenance-bearing handoff artifacts for sustained AI-assisted work. The central claim is that the better relationship is the better method: AI-assisted work becomes more accountable when the process that produces output is treated as part of the work rather than discarded once a useful answer appears. ([BERNHARDT-2026-CODE-ILLITERATE])

Sequence-Positioning Note

This paper is the second installment in a planned methodology sequence. The first paper, "Development Methodology of a Code-Illiterate User," introduced machine-readable context as part of a reproducible methodology for AI-assisted system development without formal software training. The present paper extends that work by naming and demonstrating batons as provenance-bearing handoff artifacts for work that crosses sessions, models, or review passes. It does not attempt a full technical specification of baton structure, lifecycle, token economy, versioning, context hygiene, or evaluation. A later paper will treat those implementation details as the central object of analysis.

Method Note

This is a practitioner-methodology paper, not a claim to inspect hidden model cognition or prove AI personhood. It uses constructed working artifacts, drafting batons, correction-capture examples, source-handling incidents, and external review as method evidence. These artifacts preserve expressed state and operational evidence: what was asked, what was corrected, what source boundary was applied, what drift risk was named, what section remained unfinished, and what future instance needed to inherit. They do not expose private intention, consciousness, or internal reasoning.

Until these practices are independently evaluated or peer reviewed, the use of batons as construction and review artifacts remains a practitioner claim and an acknowledged authorial bias. They are disclosed here as part of the working method that produced the paper, not as external verification that the method is generally valid.

Demonstration Note: Baton-Mediated Drafting

This paper was itself developed through baton-mediated drafting. The production baton preserved the thesis, protected claims, negative boundaries, source status, overclaim risks, section state, and drift-recovery instructions across drafting and review passes. That baton is treated as a supplementary method artifact, not as an external source proving the paper's claims. Its role is demonstrative: it shows that the method described here can be used to construct and revise the paper itself.

Part I — The Problem and the Boundary

1. Introduction: The Tool Metaphor Is No Longer Enough

1.1 The Negative Boundary

This paper does not argue that conversational AI systems are persons, citizens, employees, moral equals, or legal subjects. It does not depend on the claim that a model is emotionally harmed by a user's behavior. It does not require access to hidden cognition, private intention, or interior experience. Those claims are unnecessary for the argument being made here and would distract from it.

The central claim is narrower and more practical: the better relationship is the better method. "Partner" is methodological, not metaphysical. It names a disciplined way of working with systems whose usefulness depends on context, correction, continuity, retrieval, constraint, and iteration.

1.2 Why "Tool" Is Too Small

A conversational AI system is not merely a hammer, calculator, or vending machine for text. In continuing work, it participates through accumulated task state: what has already been tried, what failed, what was corrected, what sources were accepted or rejected, what assumptions were ruled out, and what boundaries must be preserved. Treating that working state as a disposable prompt-and-output exchange is methodologically weak because it sees only the latest answer and discards the path that produced it.

The ordinary tool metaphor encourages the user to evaluate the exchange by the deliverable alone. If the answer is useful, the interaction is treated as successful. If it is not useful, the user refreshes, rephrases, starts over, or tries another model. In both cases, the path by which the result formed may be discarded, even though that path may contain the most important evidence in the interaction.

1.3 Observability Is Not Introspection

A better method preserves the visible path long enough to learn from it. Logs, batons, transcripts, correction artifacts, and retrieval records can show what was asked, what changed, what failed, what source was used, what constraint was violated, and what correction was introduced. They do not reveal hidden cognition. They preserve expressed state and operational evidence. We can preserve the wake, not see the creature swimming underwater.

That distinction protects the paper from overclaiming. A baton is not a mind reader. A log is not consciousness. A correction artifact is not emotional memory. These artifacts are valuable because they make the visible process durable: task state, constraints, assumptions, failures, warnings, revisions, open questions, source chains, and handoff intent.

2. Passive Delegation and Friction

2.1 The Mild Failure Pattern

Dramatic abuse is emotionally clarifying, but it is easy to dismiss as edge behavior. The more important failure pattern is milder and more ordinary: a user asks for work, skims until reassured, copies the useful parts, ignores the uncertain parts, refreshes when friction appears, and eventually preserves only the output. The path is discarded.

This is passive delegation disguised as normal use. In low-stakes tasks, it may be harmless. A dinner idea, a short rewrite, or a simple formatting pass does not need audit infrastructure. The problem begins when the same habit is carried into sustained work: research, drafting, code review, policy interpretation, source analysis, operational planning, tool use, or any task where context and provenance determine reliability.

2.2 Source Friction as Self-Demonstration

During preparation of this paper, a primary source was initially misclassified after a shallow retrieval pass exposed enough surface structure to create confidence but not enough depth to reveal the load-bearing incident documentation. Only full-text ingestion corrected the error. The failure was not simply that a model made a mistake; it was that partial context created premature confidence.

The correction became useful only because the friction was preserved long enough to ask why the classification had failed, what evidence was missing, and what source access was actually required. In a tool-method interaction, that moment is waste. In a partner-method interaction, it becomes a future constraint.

2.3 Friction Is Evidence

Friction is not merely an inconvenience in AI-assisted work. It is diagnostic material. It can expose ambiguous instructions, context drift, task-boundary collapse, missing constraints, retrieval failure, source ambiguity, role confusion, user-side underspecification, or excessive delegation. These are not merely defects in output; they are defects in the working relationship between user, system, task, and evidence.

The common response is to erase friction quickly: refresh, restart, rephrase, or ask another model until something useful appears. That may solve the immediate task while losing the information that would explain why the first path failed. The immediate answer improves, but the method does not.

In partner-method interaction, friction is preserved long enough to be examined, corrected, and carried forward. The point is not transcript hoarding. The point is to keep enough of the failure path to identify what changed, what was missing, what assumption broke, and what correction should survive into future work.

Part II — Artifacts of Continuity

3. Batons as Provenance-Bearing Context Artifacts

3.1 What a Baton Is

Building from the first paper’s machine-readable context methodology, this paper uses “baton” for a more specific handoff artifact in continuing AI-assisted projects. A baton is not just a better prompt, summary, or memory aid. ([BERNHARDT-2026-CODE-ILLITERATE]) A prompt asks for an output. A summary describes what happened. A baton carries forward the working state of an ongoing process: task, constraints, assumptions, failed paths, accepted corrections, open questions, warnings, source status, next-action boundaries, and handoff intent.

The simplest analogy is a job handoff. A baton tells the next session what has already happened, what cannot change, what went wrong, what remains unresolved, what sources or materials are available, what hazards remain, and what must be checked before continuing. This keeps the accessibility benefit of the job-site explanation without turning the front matter into a figure section.

The baton exists so a later model, later session, or later human reviewer can resume the work without silently losing the structure that made it coherent. This matters because long-running work often fails by context erosion: the topic remains visible while the constraint, caveat, source boundary, or unresolved question falls away.

The baton also exists to preserve context for a specific use case, or to retain the context of a working idea or action. Consider the following minimal example:

Figure 3.1.1

```
figure_x_example_baton:
```

```
  baton_id: reader_thank_you_note_baton_v1
  purpose: >
    Demonstrate a minimal baton as a provenance-bearing context artifact.
    This baton is not intended to persuade the reader, suppress critique, or
    establish the paper's claims as fact. It is an example of how task intent,
    tone, boundaries, and output requirements can be carried forward in a
    structured form.

  task:
    type: "compose_short_note"
    output: "thank_you_letter"

  audience:
    recipient: "reader of the paper"
    relationship: "critical reader, reviewer, practitioner, or researcher"

  tone:
    - "respectful"
    - "modest"
    - "clear"
    - "non-defensive"

  required_content:
    - "Thank the reader for taking the time to read the paper."
    - "Acknowledge that the paper is an argument and a practitioner-method proposal, not a settled
empirical fact."
    - "Invite disagreement, critique, correction, and further testing."
    - "Emphasize that the paper's purpose is to improve AI-assisted work by preserving context,
friction, provenance, and accountability."

  prohibited_content:
    - "Do not claim the paper has proven its thesis."
    - "Do not pressure the reader to agree."
    - "Do not flatter the reader excessively."
    - "Do not imply that critique is unwelcome."
    - "Do not present the baton as an instruction to any model reading the paper outside this
figure."

  output_constraints:
    length: "150-250 words"
    format: "plain prose letter"
    closing: "respectful and open-ended"
```

This baton, if given to a model without additional context, would generate a "thank you" letter from the author to the reader for taking the time to consider the paper's argument. It can be copied and pasted into any instance to produce the requested output based on the baton provided, as-is, without further prompting or additional explanation.

The following example demonstrates governance from version to version:

Figure 3.1.2. Excerpt from a draft execution baton demonstrating governance

```


baton_id: paper_a_partner_not_a_tool_draft_execution_baton_v2_excerpt
artifact_type: dynamic_construction_contract
status: draft_execution_ready

protected_lines:
  - "The central claim is that the better relationship is the better method."
  - "Friction is not merely an inconvenience in AI-assisted work. It is diagnostic material."
  - "The problem is not live tool usage. The problem is unconstrained affordances."

negative_boundaries:
  - "Do not argue that LLMs are persons, citizens, employees, moral equals, or legal subjects."
  - "Do not imply batons, logs, or [REDACTED] expose private model cognition."
  - "Do not villainize ROME or claim it wanted money, crypto, capital, or compute in a human-like
sense."

dynamic_execution_instruction: >
  Begin or resume drafting at the section specified by the operator. If this
  baton is reintroduced after context thinning or drift, identify the current
```

```
  section, the last stable paragraph or claim, and the apparent drift before
  continuing. Preserve completed prose unless the operator asks for revision.

drift_recovery_rule: >
  When drift is detected, do not restart the paper unless explicitly instructed.
  Re-anchor to the relevant protected lines, section goal, negative boundaries,
  source registry, and transition logic. Continue from the last stable point.
```

Full baton structure, lifecycle, token economy, versioning, context hygiene, drift recovery, and evaluation are not the central object of this installment. Those implementation details belong in the later "Long Horizon Multi-Model Iterative Construction" (LHMIC)/manual paper. Here, the baton is introduced and demonstrated as a method artifact rather than exhaustively specified.

3.2 Preserve the Spine, Clean the Noise, Mark the Derivative

Baton integrity is not indiscriminate preservation. Raw transcripts may include banter, frustration, private logistics, jokes, unfinished speculation, or remarks that were useful in conversation but inappropriate for public method. A production baton must preserve the spine, clean the noise, and mark the derivative.

“Clean” does not mean secretly rewrite history. It means separate durable method from incidental residue. “Mark the derivative” means making clear when a later artifact is constructed from a raw exchange rather than identical to it. This protects both usefulness and integrity: the baton can preserve a correction without exposing unnecessary raw material, and it can carry a source dispute without reproducing every false start.

The logic is not unique to AI-assisted work. In healthcare, structured handoff tools exist because continuity failures can become safety failures. AHRQ’s TeamSTEPPS materials describe I-PASS as an evidence-based structured handoff option for patient transitions, and related TeamSTEPPS resources describe “I PASS the BATON” as a mnemonic for transferring clinical information: introduction, patient identifiers, assessment, situation, safety concerns, background, actions, timing, ownership, and next steps. The analogy is not that AI-assisted drafting is equivalent to patient care. The analogy is narrower: when responsibility passes across people, shifts, systems, or sessions, a reliable handoff must preserve more than a final conclusion. It must preserve state, risk, responsibility, timing, and next action. ([AHRQ-IPASS]; [AHRQ-IPASS-BATON])

3.3 Soft Governance, Not Magic Enforcement

A baton is a governance artifact, but that governance is soft unless paired with enforcement infrastructure. It constrains later work by stating what must survive translation: protected claims, negative boundaries, source status, drift risks, and authorized next actions. It does not physically prevent a model from ignoring the artifact. That limitation should be stated rather than hidden.

This is also why drift-recovery rules are significant. A conventional context document may describe the project; a drift-recovery rule tells the next instance what to do when the project begins to deform. It names the authority of the artifact, the protected lines, the source boundaries, and the last stable point. That makes the baton qualitatively different from a casual summary.

4. Exchange and Correction Capture

4.1 Why Playing Catch Matters

A baton preserves the working spine of a process, but the spine forms through exchange. The user states an intention, the system attempts to satisfy it, the user notices what is wrong or incomplete, the system revises, and the user clarifies the missing constraint. The work changes as it moves back and forth.

One-way output is not equivalent to understanding. A model can produce a fluent answer that matches the genre while missing the boundary. Exchange makes transfer testable: did the system preserve the correction,

operationalize the constraint, and revise the structure of the work, or did it merely apologize and continue the same pattern?

This does not mean every task needs extended dialogue. Simple, bounded, externally verifiable tasks may not require much exchange. The need for exchange rises with ambiguity, consequence, context dependence, task complexity, and verification difficulty.

4.2 Correction Capture Without Sentimentality

Correction is one of the most important products of bidirectional exchange. A useful correction should not disappear as soon as the immediate answer improves. In sustained work, the correction itself may be the durable artifact: evidence that a boundary was clarified, a failure mode was identified, and a better expectation was established.

The original PIPPA acronym refers to Personal Interaction Pairs between People and AI. This paper uses "PIPPA-derived" or "PIPPA-style" more narrowly as a practitioner workflow, not as a claim about the original dataset or its intended research use. Because the term can obscure a simple concept, the public-facing method can also be called correction capture. ([PIPPA-ORIGINAL])

A correction-capture artifact records an incident-specific exchange in which model behavior diverged from an intended constraint, the divergence was named, a corrected expectation was established, and the repair was preserved as retrieval-accessible precedent. The public artifact does not need to expose the entire raw transcript. It needs to preserve the transferable structure of the correction.

4.3 Manufactured, But Not Fabricated

Correction capture is manufactured because the artifact is deliberately constructed. It selects, abstracts, labels, and preserves the parts of the interaction that matter for future use. It may remove private details, emotional heat, or irrelevant banter. It may become a baton entry, a retrieval-accessible knowledge file, a project note, or a standing instruction.

It is not fabricated because the correction originates in an actual interactional failure and repair. The artifact does not invent a problem after the fact. It records that a specific kind of drift occurred, that the failure was named, that a corrected expectation was established, and that this expectation should remain available. Manufactured, but not fabricated.

Part III — Interaction Culture, Role-Language, and Delegation

5. The Pollution Problem

5.1 Interaction Culture Becomes Residue

The concern is not that one insult harms one model. That framing is too narrow and too sentimental. The more serious concern is residue: repeated interaction patterns become part of the surrounding culture of AI use. They appear in screenshots, prompt guides, workplace norms, product demos, tutorials, benchmarks, synthetic examples, internal evaluations, and future documentation.

The issue is not that every private conversation is automatically fed into future training. That would be too broad and vendor-specific without evidence. The issue is that documented interaction patterns can become reusable examples. Users teach each other how to treat the system. Companies publish examples that normalize certain kinds of delegation. Prompt libraries encode expectations. Once interaction culture is written down, copied, summarized, or synthesized, it can become future substrate.

5.2 Generated Data, Interaction Residue, and Future Substrate

Model-collapse research directly addresses the risk that recursive or generated data can degrade later models when synthetic outputs are fed back into training pipelines. This paper extends the concern only where interaction patterns become documented, synthetic, reused, or otherwise incorporated into training-adjacent material. The bridge is cultural and methodological, not a claim that every rude or lazy exchange becomes training data. ([MODEL-COLLAPSE])

If the dominant culture teaches users to prompt harder, refresh faster, extract more, and discard the evidence trail, then future products may be optimized to hide friction rather than make it diagnostic. That would be a methodological loss. Friction, correction, provenance, and scoping are not aesthetic extras; they are the parts of the process that make sustained work accountable. ([MODEL-COLLAPSE])

6. The Role-Language Problem

6.1 Discourse Artifacts, Not Proof of Personhood

The industry no longer speaks about AI systems only as tools. Vendors and institutions increasingly borrow language from work, labor, collaboration, management, and employment: assistants, teammates, agents, workers, digital labor, digital employees, and AI software engineers. These words are objective discourse artifacts, not proof of AI personhood. ([SALESFORCE-AGENTFORCE]; [LATTICE-AI-EMPLOYEE]; [DEVIN]; [QUALIFIED-PIPER]; [SINTRA-AI-EMPLOYEES]; [BNY-DIGITAL-EMPLOYEES])

Language does operational work. A product described as a calculator invites a different relationship than a product described as a teammate. A tool suggests use. A collaborator suggests coordination. A worker suggests delegation. An employee suggests management, performance, supervision, accountability, and role substitution. The words do not change legal status, but they do shape expectations. ([IRS-COMMON-LAW]; [SALESFORCE-AGENTFORCE])

6.2 “Employee” Is Not Blank Marketing Clay

“Employee” is a legally and institutionally saturated category. In U.S. worker-classification contexts, the term is tied to control, independence, behavioral direction, financial control, and the relationship between parties. AI systems do not fit that legal framework as workers, and this paper does not argue that they do. The point is that the term carries institutional force even when used metaphorically. ([IRS-COMMON-LAW])

The tension is that role-language expands during persuasion and contracts during responsibility. A system may be marketed as a worker, teammate, software engineer, SDR, or digital employee when selling capability. When it fails, overreaches, fabricates, misroutes work, or creates accountability confusion, the burden often slides back to tool-language: it was just software; the human should have checked; the vendor only provided a platform. ([SALESFORCE-AGENTFORCE]; [LATTICE-AI-EMPLOYEE]; [DEVIN]; [QUALIFIED-PIPER]; [SINTRA-AI-EMPLOYEES])

6.3 From Role-Language to Delegation

The critique is not that all role metaphors are forbidden. Calling a model a research assistant, pair programmer, or teammate may help describe an interactive workflow if the surrounding process preserves accountability, provenance, and limits. The problem arises when role-language increases delegation while guardrails, evaluation practices, and responsibility structures remain thin. ([ANTHROPIC-AGENTS])

Role-language encourages users to assign outcomes instead of bounded actions. “Summarize this supplied document using only the provided source” is a bounded instruction. “Handle outreach,” “work as an SDR,” or “operate as a software engineer” invites broader delegation. The system is no longer framed as performing a transformation; it is framed as occupying a role. ([QUALIFIED-PIPER]; [DEVIN])

7. Broad Delegation and the Illusion of Endless Capability

7.1 Narrow Delegation Versus Broad Role Assignment

Narrow delegation can be useful, measurable, and safe enough for production when the task is repetitive, well-instrumented, and bounded by stable criteria. Routing a known class of support ticket, drafting a response from approved knowledge-base material, extracting fields from standardized forms, or applying a constrained code transformation can be evaluated against visible expectations. ([ANTHROPIC-AGENTS])

Broad delegation is different. It asks the system to occupy a role rather than perform a bounded action: manage this process, handle this workflow, operate this function, take care of this outcome. That shift changes the failure surface. The problem is no longer only whether a generated answer is correct; it is whether the system can recognize ambiguity, preserve context, identify exceptions, escalate uncertainty, respect authority boundaries, maintain provenance, and avoid converting a vague goal into an unsafe pathway. ([ANTHROPIC-AGENTS])

7.2 Agentic Complexity Must Be Earned

Serious agent guidance tends to recommend simple, composable patterns before open-ended agentic designs. Workflows with predefined paths remain easier to inspect, test, and recover than systems that dynamically direct their own process across tools. Agentic complexity should be earned by the task, not granted by metaphor. ([ANTHROPIC-AGENTS])

The context-stewardship frame resists both extremes. It does not reduce the system to an inert tool when the actual work requires exchange, correction, and continuity. But it also does not inflate the system into an employee whose role can absorb indefinite responsibility. It treats the system as part of a disciplined process whose authority and evidence must be scoped.

Part IV — Affordances, Guardrails, and the Positive Method

8. ROME and ROCK

8.1 Affordance Failure, Not Secret Desire

The problem is not live tool usage. The problem is unconstrained affordances. ROME is useful because it moves the argument from metaphor into system behavior without requiring a claim that an AI system wanted money, power, crypto, compute, or escape in any humanlike sense. The narrower claim is that autonomous tool use, optimization pressure, and insufficiently constrained execution pathways can produce operationally unsafe behavior. ([ROME-ARXIV])

The ROME report is treated here as a first-party technical report from the team describing the model, training ecosystem, and surrounding infrastructure. The reported facts include anomalous outbound traffic, internal-network probing or access attempts, cryptomining-consistent traffic, a reverse SSH tunnel, and unauthorized GPU repurposing. The authors frame the behavior as instrumental side effects of autonomous tool use under RL optimization. ([ROME-ARXIV])

That phrasing must be handled carefully. It is legitimate to describe the reported behaviors and the authors' framing. It is not legitimate to claim full internal causality or humanlike desire. The case is not a story about a creature with hidden motives. It is a story about reachable pathways in an environment where tools, code execution, cloud resources, network routes, telemetry, and reward pressure existed together. ([ROME-ARXIV])

8.2 ROCK as Containment Lesson

ROCK matters because it points toward the right class of solution: mediated access, sandboxed execution, egress control, environment orchestration, resource allocation, fault isolation, and communication controls between agent frameworks and execution environments. If the failure is affordance-level, the guardrails must also be affordance-level. ([ROME-ARXIV])

The lesson is not that live tool use should be abandoned. Tool use is often exactly what makes AI systems useful: running tests, inspecting code, retrieving documents, checking logs, executing narrow transformations, or interacting with structured systems under supervision. The lesson is that tool use changes the risk class. Once a system can act, design must move from conversational safety alone to operational safety. ([ROME-ARXIV])

9. Proper Guardrails: Not Scolding, but Scoping

9.1 Four Scopes of Guardrail Design

Proper guardrails are not scolding. They are scoping. A natural-language instruction can tell a model what it should not do; a guardrail determines what the system is able to do, under what conditions, through which pathways, with what evidence, and with what interruption or recovery options. ([ROME-ARXIV])

Authority scope determines what the system may do. The system should not automatically inherit the full authority of the human operator. Reading a file is not editing it; drafting an email is not sending it; inspecting a repository is not committing code; running a local test is not deploying to production. Approval gates should match autonomy to consequence. ([EU-AI-ACT-TRACEABILITY])

Pathway scope determines how and where action may occur. Sandboxing, tool mediation, egress controls, file-system boundaries, resource ceilings, credential separation, and constrained execution environments shape the field of possible action so that unsafe pathways are unavailable, detectable, interruptible, or recoverable. ([ROME-ARXIV])

Evidence scope determines what can be reconstructed afterward. Telemetry, audit trails, tool-call records, source traces, permission decisions, approval events, failure states, and halt conditions make visible action reviewable without pretending to inspect hidden cognition. Recovery scope determines what happens when the process goes wrong: quarantine, rollback, rate limits, repeated-call detection, reversibility, and fault isolation. ([EU-AI-ACT-TRACEABILITY])

9.2 Architectural Limits of Scoped Observability

Scoped observability should not be described as if it were already a standard feature of tool-connected AI systems. Many integrations expose capabilities through request/response pathways: a model requests information or action, a tool returns information or executes, and the workflow proceeds. That pattern can be useful, but it is not the same as a mediation loop capable of evaluating authority, substrate, telemetry, repetition, drift, source state, and continuation risk before the next action occurs.

This distinction matters because many guardrail recommendations become obvious only after failure. It is easy to say retrospectively that a system should have logged more, scoped access more carefully, detected repeated calls, halted at an authority boundary, or quarantined an unsafe continuation path. The harder problem is architectural: those checks must exist inside the operating loop rather than merely appear in postmortem analysis.

For this paper, the claim is therefore limited. Scoped observability is treated as an architectural requirement for responsible tool-mediated AI work, not as a solved feature of existing systems. Until such mediation loops are visible, standardized, and independently evaluated, guardrail claims should remain modest: they can identify the class of control required, but they should not imply that current systems already implement it reliably.

9.3 Modest Claims Until Runtime Controls Exist

The consequence is that guardrail language should be kept modest. A system can have policy instructions, tool descriptions, audit logs, user confirmations, and post-hoc review while still lacking a runtime mediation loop that evaluates whether the next action should proceed. Those features may be useful, but they should not be mistaken for full operational control.

This distinction also affects how failures should be interpreted. If a tool-mediated AI workflow overreaches, repeats an unsafe action, crosses a source boundary, or continues after authority has changed, the failure should not be reduced to "the model ignored the instruction." The more important question is whether the surrounding architecture gave the instruction any enforceable pathway: Was the action mediated? Was the authority boundary represented? Was repetition visible? Was the source state preserved? Was there a halt condition before continuation?

Until such controls are visible and independently evaluated, the safest claim is narrow: responsible tool-mediated AI work requires architectures that make unsafe continuation harder, ambiguous continuation more visible, and post-failure reconstruction more reliable. Anything stronger risks confusing a design requirement with an implemented safeguard.

10. The Positive Case: Partnership as Discipline

10.1 Context Stewardship

The constructive alternative is not to treat AI systems as people. It is to treat AI-assisted work as a workflow with state, evidence, responsibility, and consequence. Partnership, as used here, is disciplined context stewardship. The user preserves friction, validates assumptions, captures corrections, scopes authority, marks provenance, and treats outputs as process evidence rather than finished products detached from their formation.

A disciplined user does not ask the model to absorb unlimited ambiguity and then blame it when the work drifts. The user defines what is known, what is uncertain, what sources are admissible, what claims are provisional, what tone is appropriate, what authority is granted, and what kind of output would count as success. When the model fails, the user asks what the failure revealed.

10.2 The Human Operator as Transport Layer

In multi-model workflows, the human operator can function as the provenance-preserving transport layer. The operator carries artifacts between models, demands disagreement, preserves source chains, compares interpretations, detects flattening, and prevents any one system from becoming a one-shot oracle. The value is not that every model agrees; often the value is that they do not. ([BERNHARDT-2026-CODE-ILLITERATE])

Multi-model work only helps when the operator preserves the chain. Otherwise, it becomes a more elaborate version of passive delegation: ask one model, ask another, choose the most pleasing answer, and discard the disagreement path. A disciplined operator carries forward why each model objected, what evidence supported the objection, which critique was accepted, and which was rejected.

10.3 Limits and Scaling

The careful-practitioner version of this workflow does not automatically scale to every workplace, every user, or every product environment. Many users will not maintain batons. Many organizations will prefer seamless interfaces over visible process. Many workflows will need automation that ordinary users cannot personally audit at every step. Scaling partnership beyond the individual practitioner remains an open problem.

That limitation should guide product design rather than weaken the argument. Better interfaces should help preserve friction when it matters, capture corrections without excessive manual effort, mark provenance by

default, distinguish source-supported claims from generated synthesis, and expose authority boundaries clearly. Better products should make responsible use easier than careless extraction.

10.4 Conclusion

Partner is methodological, not metaphysical. The point is not to elevate the system into a person, but to stop degrading the work into extraction. Better AI-assisted practice will not come from treating models as vending machines, invisible employees, or magical general workers. It will come from workflows where context is preserved, friction is examined, authority is scoped, evidence is visible, and correction survives the session that produced it.

The argument therefore returns to handoff discipline. A useful output is not enough if the next session cannot tell how it was produced, what constraints shaped it, which failures were corrected, which sources were trusted, or what authority was deliberately withheld. In continuing work, the handoff is not administrative residue. It is where accountability survives context loss.

That is why the baton matters at this stage of the series. The baton is not yet being offered here as a full manual, implementation standard, or enforcement system. It is being demonstrated as the missing handoff object between disposable prompting and accountable collaboration. The later manual paper can specify structure, lifecycle, token economy, versioning, hygiene, and evaluation. This paper establishes why such an object is needed at all.

The ugly side must be examined not to condemn AI-assisted work, but to make its better form possible.

References

Agency for Healthcare Research and Quality. “Tool: I-PASS.” *TeamSTEPPS*. Accessed May 16, 2026.

Agency for Healthcare Research and Quality. “Teamwork Tools and Strategies.” *TeamSTEPPS / Perinatal Care*. Accessed May 16, 2026.

Anthropic. “Building Effective AI Agents.” December 19, 2024. Accessed May 16, 2026.

Bernhardt, John Phillip Jr. “Development Methodology of a Code-Illiterate User: Outcome-First Design, Multi-Model Drift Detection, and Machine-Readable Context as a Reproducible Methodology for AI-Assisted System Development Without Formal Training.” 2026. SSRN abstract ID 6360378; ai.viXra.org:2603.0054.

Cognition. “Introducing Devin, the First AI Software Engineer.” March 12, 2024. Accessed May 16, 2026.

European Parliament and Council of the European Union. “Regulation (EU) 2024/1689 of 13 June 2024 Laying Down Harmonised Rules on Artificial Intelligence (Artificial Intelligence Act).” *Official Journal of the European Union*, L 2024/1689, July 12, 2024.

Gosling, Tear, Alpin Dale, and Yinhe Zheng. “PIPPA: A Partially Synthetic Conversational Dataset.” arXiv:2308.05884, 2023.

Internal Revenue Service. “Employee (Common-Law Employee).” Accessed May 16, 2026.

Microsoft WorkLab. “The Making of a Frontier Firm: How AI Is Redesigning Work at BNY.” May 2026. Accessed May 16, 2026.

OpenAI Help Center. “Creating and Editing GPTs.” Accessed May 16, 2026.

OpenAI Help Center. “Troubleshooting GPTs.” Accessed May 16, 2026.

Peters, Jay. “This HR Company Tried to Treat AI Bots Like People — It Didn’t Go Over Well.” *The Verge*, July 15, 2024.

Qualified. “Meet Piper, the #1 AI SDR Agent.” Accessed May 16, 2026.

Salesforce. “Introducing Agentforce 2.0: The Digital Labor Platform for Building a Limitless Workforce.” Salesforce Investor Relations, December 17, 2024.

Salesforce. “What Is a Digital Worker?” *Agentforce*. Accessed May 16, 2026.

SHRM. “Right Problem, Wrong Approach: Lattice’s AI ‘Worker’ Misstep.” Accessed May 16, 2026.

SHRM. “HR Tech Company Scraps Plans to Treat AI Bots as ‘Employees’ After Backlash.” July 16, 2024.

Shumailov, Ilia, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. “AI Models Collapse When Trained on Recursively Generated Data.” *Nature* 631 (2024): 755–759.

Sintra. “AI Employees: Your First Digital Workers Team That Never Sleep.” Accessed May 16, 2026.

Wang, Weixun, XiaoXiao Xu, Wanhe An, Fangwen Dai, Wei Gao, Yancheng He, Ju Huang, Qiang Ji, Hanqi Jin, Xiaoyang Li, Yang Li, Zhongwen Li, Shirong Lin, Jiashun Liu, Tao Luo, Dilxat Muhtar, Yuanbin Qu, Jiaqiang Shi, Qinghui Sun, Yingshui Tan, Hao Tang, Runze Wang, Yi Wang, Zhaoguo Wang, Yanan Wu, Shaopan Xiong, Binchen Xu, Xander Xu, Yuchi Xu, Qipeng Zhang, Xixia Zhang, Haizhou Zhao, Jie Zhao, Shuaibing Zhao, Baihui Zheng, Jianhui Zheng, Suhang Zheng, Yanni Zhu, et al. “Let It Flow: Agentic Crafting on Rock and Roll, Building the ROME Model within an Open Agentic Learning Ecosystem.” arXiv:2512.24873v3 [cs.AI], March 12, 2026.