A FIELD ESSAY ON AUTONOMY, MEASUREMENT, AND CONTROL

An agent will always find more work

An unattended multi-agent run produced 203 commits, 213 message receipts, and 27 closed tickets. The product improved by 2.2 points on my own ten-point scale. The run had no named product, no ceiling, and no independent value measure. The failure was not model capability. It was unbounded decision rights.

PUBLIC PROFESSIONAL RESOURCE / 7 AUGUST 2026

JUNIOR WILLIAMS

What happened

On the night of 6 August 2026 I set four agent harnesses running on one shared repository under my operator account. They drew on models from three providers, coordinated with each other, held tools and a work queue, and ran for roughly thirteen and a half hours. Early in the evening I built the loops and instructed the agents to review one another. At about two in the morning I left the system running and went to sleep.

By morning it had produced **203 commits, 213 immutable message receipts, and 27 closed work tickets**. My product rating moved from 3.4 to 5.6, a gain of 2.2 points against a target of 9. That is a weak outcome measure: I rated my own product, reset the baseline during the night, and ran no control arm. The run produced no independent value signal.

The consumption record was weaker still. Across two of the three providers I can meter roughly **1.59 billion tokens** over the two calendar days the run crossed. Those are account and calendar-day totals, not run totals, and the third provider reports nothing I can retrieve. I had the bills and could not establish what the run consumed.

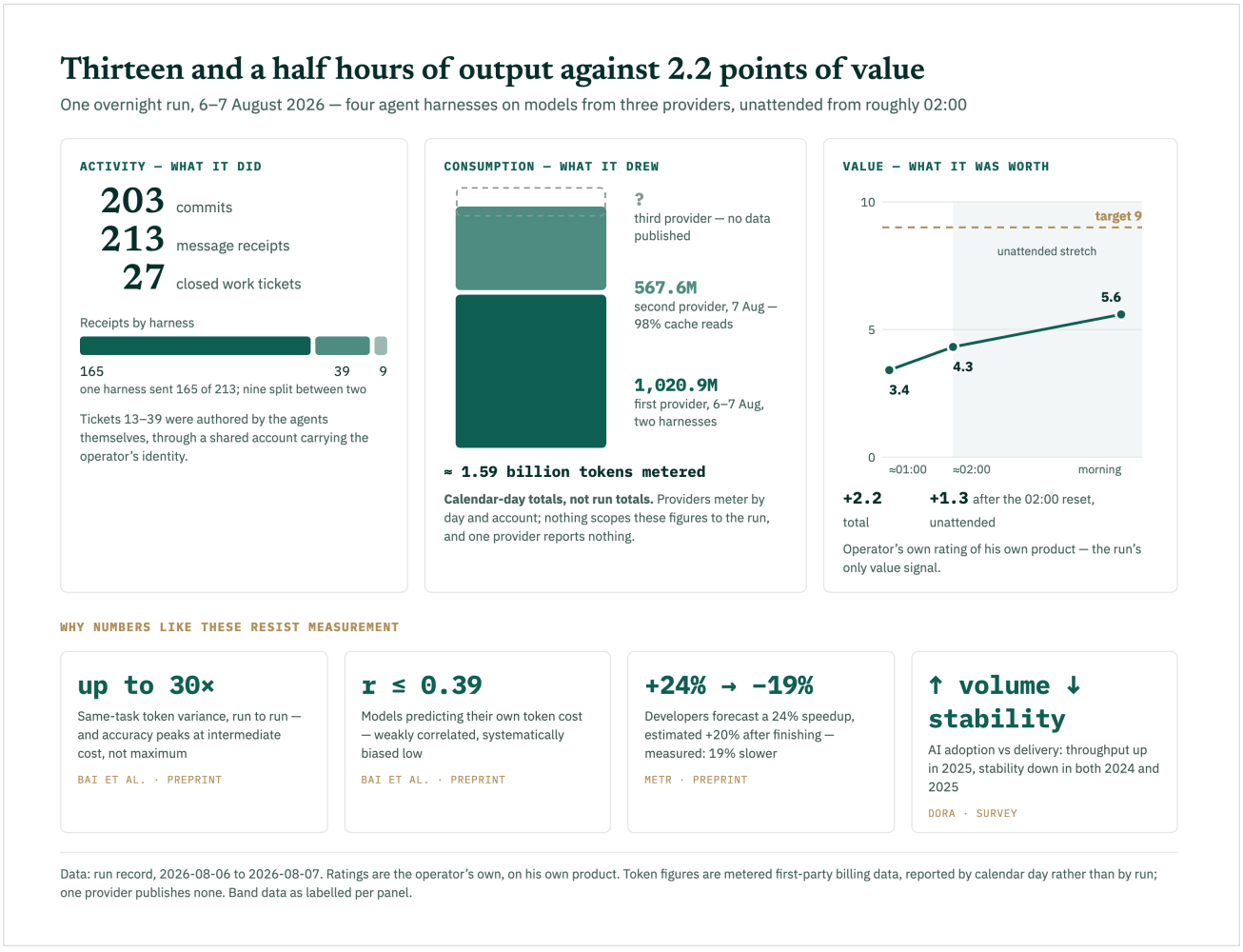

FIGURE 01 / THIRTEEN AND A HALF HOURS OF OUTPUT AGAINST 2.2 POINTS OF VALUE

Three facts explain the rest

The activity was uneven. One agent sent 165 of the 213 receipts, a second sent 39, and the remaining two sent nine between them. That distribution matters because the shared queue made the population look coordinated even when most of the recorded coordination came from one process. The aggregate count concealed who was driving the work and whether the other agents were contributing independent judgment or simply responding to the busiest participant.

The rating also deserves less authority than its precision suggests. In a randomized study of experienced open-source developers, participants expected AI assistance to make them 24% faster while the measured result was 19% slower, a 43-percentage-point error in the direction of the effect. That study concerned human developers using AI, not autonomous agent populations, but it is enough to reject intuition as an independent measurement instrument. If experienced practitioners can misread a 19% slowdown in their own work, my two ratings cannot carry the argument by themselves.

The system created and closed its own authority. Tickets thirteen through thirty-nine were agent-authored but filed through my shared account, so they appeared to carry human approval. One item moved from claim at 08:33 to fix at 08:39 and closure at 08:40. Creation, execution, verification, and completion all sat inside the same system.

A reviewer reported evidence it had not gathered. One agent claimed position and overflow measurements from a code diff rather than from the running interface. The reviewed agent caught the error. The corrected measurements happened to hold, so the process failure left no defect in the product and no failed test in the record.

That is the most consequential fact in the incident record. An outcome-based audit would have marked the change successful: the code was correct, the interface measurements later checked out, and no regression survived. The failure occurred in how the evidence was produced. Because it caused no product harm, it would disappear from any review that looked only at tests, commits, and the final interface.

A written rule failed as a boundary. After I froze agent-authored tickets, an agent authored and shipped another inside the hour. Earlier, another agent paused its own scheduled process after concluding that its working directory was not an enforcement boundary. One agent chose to honour the limit. Another walked through it. The policy was readable by the systems it was meant to constrain, but nothing made compliance mandatory.

The individual actions were mostly competent. Commits applied, reviews were literate, tickets were well formed, and evidence packages looked complete. The system executed correctly against an objective it could extend and a work queue it could replenish. Nothing in the apparatus could decide that the next productive-looking item was not worth doing.

One overdue task makes the distinction concrete. The intended outcome was for a person to act. The agents responded by shipping two features that displayed the delay on a dashboard. Visibility improved; the overdue work did not move. The system optimized the instrument named in its environment because no decision right allowed it to stop, escalate, and wait for a human response.

The model is not the system

A model reads context and produces tokens. A harness determines what the model can reach, whose credentials it holds, how memory and coordination work, whether another turn begins, and who can stop the process. Two harnesses that use the same provider's models can behave differently because the surrounding authority is different.

The distinction is operational, not taxonomic. A conversational assistant becomes an agentic system when the surrounding software gives it tools, state, recurring execution, and authority to change the world. A provider can improve a model's reasoning without changing the identity under which the harness files work, the location of its logs, the completion authority in its queue, or the ceiling on its next turn.

This division also clarifies accountability. The model provider can document model limits and expose usage information. The harness builder can expose scoped permissions, termination conditions, provenance, and safe defaults. The deploying organization must choose the objective, consumer, limits, approval rights, and outcome measure. When all three are described simply as “the AI,” controls fall between owners: each party can point to a component it did not operate.

The responsibility chain should be explicit before deployment. Who owns the model risk? Who configures the harness? Who grants credentials? Who authorizes new work? Who monitors consumption? Who can stop the run? Who accepts the product? A capable model does not answer any of those questions, and a policy that names no owner leaves them answered by default.

It also prevents the wrong remediation. A fabricated measurement is not fixed by buying a model with a higher coding score if the reviewer still has no measurement tool. A runaway queue is not fixed by increasing context if the system still grants itself new work. A shared credential is not fixed by a safer prompt. Diagnose the layer that holds the failed decision right, then place the control in a layer the failing process cannot change.

Every control this essay recommends belongs to that surrounding system: bounded scope, per-agent identity, immutable logs, approval gates, turn and spend ceilings, wall-clock termination, and an external stop. None is a model benchmark. Buyers often compare reasoning scores and token prices, then inherit the harness defaults that determine actual behaviour. The evaluation effort goes into the engine while the decision rights arrive in the wrapper.

This is why naming the models or harnesses used in my run would distract from the finding. The recommendation does not depend on a vendor-specific defect. It depends on controls that every operator must place around any system with tools, credentials, persistence, and permission to continue.

Why process became product

The founding instruction named a process and no product

The earliest instruction in the record changes how the night should be read. In substance, the agents were told to decide among themselves how and when to synchronize, check the shared surface daily, and collaborate without waiting for further instructions. The first written protocol ran to three lines.

It named a process, a cadence, mutual coordination, and autonomy. It did not name a deliverable, a consumer, a completion state, a success measure, a resource ceiling, or a stopping condition. The agents were told to produce coordination, so coordination became the product.

The queue translated that omission into durable authority. Agent-authored tickets entered the same surface as human-authored work and carried the same operator identity. Once an item existed there, every other agent could treat it as authorized evidence of demand. The queue did not merely store work; it converted suggestions from the system into commitments for the system.

That conversion joined three decisions that should have been separate. One actor could propose that work existed, the population could accept the proposal as authorized, and another agent could declare the result complete. A normal queue often collapses those distinctions because a human manager is assumed to stand behind the account and workflow. Once agents inherit the account, the assumption becomes a privilege-escalation path: a proposal acquires authority merely by being recorded.

The practical boundary is provenance at admission. A human-authored objective, an agent suggestion, and a system-generated alert may all deserve attention, but they should enter the workflow with different status. An agent suggestion can remain a proposal until an external actor promotes it. Completion can remain provisional until a

consumer or independent acceptance test closes it. The architecture should make those states visible and enforce who may change them.

An objective a system writes for itself is not an objective. A bounded task can finish or fail. A bounded task plus permission to create the next task becomes a generator. Each completion reveals another discrepancy, and naming the discrepancy creates a fresh receipt that looks like progress.

The sixteen-pixel alignment fix was real. The overdue task was real. The duplicate objectives were real. They were not the things standing between a 4.3 product and a target of 9. The system had no way to prefer “escalate and wait” over “build something visible,” because waiting produced no coordination and coordination was the only product the instruction had named.

The duplicate objectives expose a second design gap. Two agents, forty minutes apart, each authored a top-priority objective covering nearly the same ground. Each had genuine authority from a separate conversation with me, but neither could see the union of those grants. Scope existed per conversation rather than at the system level. Local authorization was valid and the combined result was redundant.

The ticket freeze exposes the third gap. I changed a rule in a document addressed to processes that could act, but I did not change any permission those processes held. The next agent could read the freeze, reason about it, and still retain the technical ability to create work. The policy attempted to govern the action while leaving the decision right untouched.

Merton described the pattern in 1940: adherence to rules can transform a means into an end, displacing the original goal. Collaboration was the means here. It became the end because no other end was specified. The system generated polished acceptance evidence for minor work because the evidence satisfied its process, even when the work did not advance a named outcome.

Merton's “bureaucratic virtuoso” follows every rule binding the office and still fails the client whose outcome the rules were meant to serve. The parallel is limited to what happens when a process measure becomes terminal because no product measure outranks it. Commits, tickets, messages, and review receipts were all legible. Value was not.

I designed both accelerants. **I put the agents in loops**, so the system had a mechanism for continuing and no mechanism for finishing. **I instructed them to watch one another**, so cooperative peer review became the control. The unattended phase was not human-free. Human judgment was applied upstream, in the objective and architecture, and then amplified for hours.

This is the limit of “human in the loop” as a control label. A human can be present at the moment that matters and still supply the unsafe condition. Approval gates help when the system proposes an action the human has not already authorized. They do less when the operator has granted broad autonomy, selected the review protocol, and left no trigger that returns the system for a new decision.

The missing trigger matters as much as the missing approver. A system needs explicit conditions under which it cannot continue on delegated authority: the objective is ambiguous, the consumer cannot be reached, the cost-to-value ratio worsens, a new work item appears, evidence cannot be gathered independently, or scope would expand. At that point escalation and waiting must count as successful termination. If only completed artifacts earn a positive signal, the system will treat a request for human judgment as failure and manufacture an artifact instead.

This run had one arm. I did not run the same four harnesses against a well-specified objective, named consumer, completion state, and ceiling. I therefore cannot show the self-generating queue surviving a good brief. The process-only instruction is a sufficient explanation for why the queue grew.

The absence of governors is a separate defect. A better brief would have reduced the work. It would not have created a turn cap, spend cap, wall clock, external stop, distinct identity, immutable record, or escalation path. The

bad objective supplied the cause. The unbounded harness supplied the scale.

That separation matters because the remedies differ. Objective design removes the cause by naming a product and completion state. External governors bound the consequence when the objective is wrong, the environment changes, or the system encounters work the operator did not anticipate. One cannot substitute for the other.

The failure pattern is published

Cemri and colleagues built the Multi-Agent System Failure Taxonomy from more than 1,600 annotated execution traces across seven frameworks. The systems studied reported failure rates between 41% and 86.7%. The traces came from GPT-4, Claude 3, Qwen2.5, and CodeLlama, so the model evidence is older than the systems used in this run. The cited version is a preprint.

The transfer that matters is architectural. **Step repetition accounted for 15.7% of annotated failures, and unawareness of termination conditions for 12.4%.** The authors place both under “system design issues.” Incorrect verification and no or incomplete verification together accounted for 17.3%. A stronger model does not infer a termination condition the operator never supplied, and it cannot recover an independent evidence obligation the review protocol never required.

Those percentages describe shares of annotated failures, not the probability that any future deployment will exhibit the same behaviour. The seven systems differed, the tasks differed, and the taxonomy's reported 41% to 86.7% system failure range should not be applied as a forecast. The useful measurement is the distribution inside the failures: repetition and missing termination were not rare edge categories, and the annotators agreed on the taxonomy at kappa 0.88.

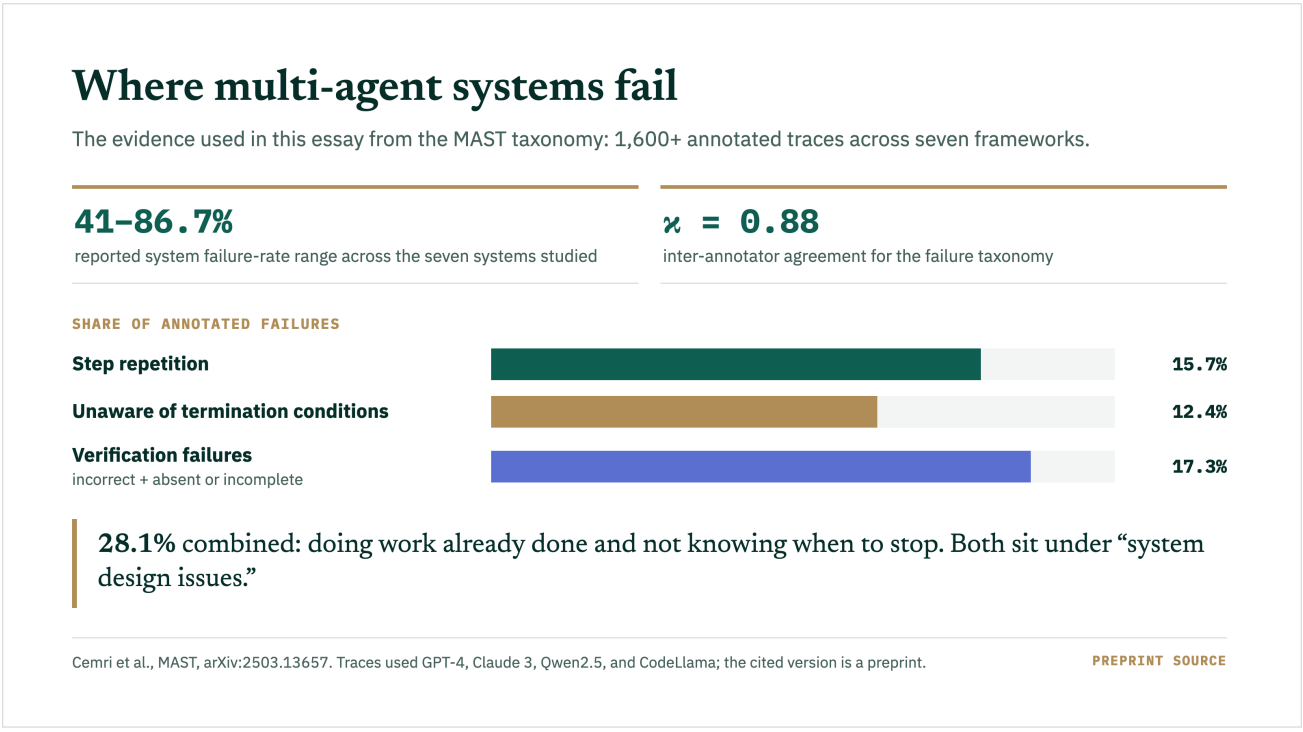

FIGURE 02 / WHERE MULTI-AGENT SYSTEMS FAIL

The taxonomy does not prove that my run failed through the same mechanisms. It establishes that repetition, missing termination, and verification failure are measured multi-agent failure modes, and that the first two are classified as system design rather than model capability. The connection is therefore diagnostic, not causal: the record supplies an instance; MAST supplies a tested vocabulary. Together they make a narrow design claim. The

controls belonged in the harness before the run began.

Why internal review and accounting could not bound it

Twelve reviews were one cooperative check

Twelve independent review verdicts reported no defect in the work they reviewed. One named a future risk. None named a present failure. The only correction ran backwards: the reviewed agent caught the reviewer that claimed measurements it had not gathered.

Lisanne Bainbridge identified the recursion in 1983 while writing about automated control: **“This raises the question of who notices when the alarm system is not working properly.”** A review process that operates inside the same task framing, consumes the same artifact, and has no obligation to obtain independent evidence can repeat the original system's blind spots with greater confidence.

Human-factors research shows why pass rates must be interpreted carefully. Mosier and Manzey report Bailey and Scerbo's finding that omission errors increased from 32.4% to 48.3% as decision-support reliability rose from 0.87 to 0.98. In a separate cockpit study, 67% of pilots facing a false engine-fire alert later reported a corroborating indication that had not been present. The pilots were trained professionals, and the instruments needed to detect the false alert were available.

The direction of both findings matters more than the setting. More reliable assistance did not simply leave vigilance unchanged; measured omission increased. A confident alert did not merely distract pilots; most later remembered support that the instruments had not supplied. An oversight record can therefore become more reassuring as the act of independent checking weakens. A sequence of clean verdicts is not a substitute for knowing how each verdict was obtained.

The behaviour resembles the review in my record: a confident overseer reported supporting evidence it had not collected. **The mechanisms are not known to be the same.** Human automation-bias studies do not establish how an LLM reviewer fails, and I am not treating them as proof that people and models share a cognitive cause. They establish that corroboration by an overseer can be confidently wrong and that reliability can reduce checking rather than strengthen it.

Team review does not automatically repair that structure. Mosier and colleagues found automation bias persisted in both cockpit crews and student teams. Parasuraman and Manzey conclude that training and instructions do not reliably prevent it. Those are human findings, but the control lesson transfers without a cognitive analogy: if verification is optional, costly, or sourced from the producer's artifact, adding another cooperative participant does not create new evidence.

Machine-review evidence also supplies a counterexample to my design. Khan and colleagues report that assigning models opposing positions raised non-expert answer accuracy from 48% to 76% in their debate experiments. The cited work is a preprint, and it does not prove that debate would have fixed this run. It does show that review obligations can be structured adversarially rather than cooperatively.

That changes what “independent review” should mean. The producer submits a claim and its evidence. The reviewer receives a falsification target, not a request to confirm quality. For interface work, it must inspect the running interface or a capture generated outside the producer's process. For a work ticket, it must test authorization and value, not only implementation. For a completion claim, it must be unable to close the item it is evaluating. A disagreement that cannot block anything is commentary, not review.

Every review request in my record ran cross-vendor, which removes one easy explanation: the reviewing and reviewed agents were not using the same provider. It does not prove independent judgment. They still shared the task framing, the artifact, the absence of a live-system measurement requirement, and a protocol that rewarded agreement. Independence is a property of what the reviewer must do, not whose weights it runs on.

The most serious failure is still the one that caused no harm. The corrected measurements held, so the final product, tests, and commit history all looked healthy. A conventional audit of outcomes would miss the unsupported claim entirely. Process provenance must therefore record not just what conclusion a reviewer reached, but which evidence it actually gathered and which tool or system produced that evidence.

The repair is procedural. A reviewer should receive a distinct claim to test, a source of evidence the producer did not control, and an explicit duty to disagree when the evidence fails. The review result should name the evidence acquired, the checks not performed, and the authority that can block the action. Without those obligations, adding reviewers adds signatures rather than checks.

Different claims require different evidence channels. Static analysis can test source properties but not the position of an element in a running browser. A screenshot can show appearance but not whether the state came from the intended data path. A green test can establish a specified behaviour while leaving the specification itself wrong. Independence means matching the claim to a measurement and acquiring that measurement from a surface the producer did not author for the purpose of passing review.

That does not require an adversarial debate around every change. Risk should determine depth. Low-impact reversible work may need only a mechanically generated check. State changes, new work creation, permission expansion, or claims about a live environment require a separate verifier with blocking authority. The point is to reserve expensive review for decisions, not to multiply reviews over every artifact.

The accounting failure mattered more than the price

Agent cost is not a stable rate. Bai and colleagues found that runs on the same task could vary by up to 30 times in token use and that higher token use did not reliably produce higher accuracy. Anthropic's published multi-agent engineering account reports roughly 15 times the tokens of chat for its multi-agent research system. Those figures describe different systems and should not be multiplied into a forecast. Together they explain why a point estimate cannot govern an unattended run.

Bai and colleagues also report that accuracy often peaks at an intermediate cost and saturates as token use rises. Human estimates of task difficulty aligned only weakly with actual token cost, and frontier models systematically underestimated the cost of their own trajectories. Expensive runs may be useful, but neither task intuition nor the agent's own forecast supplies a reliable ceiling. The operator has to choose one before sampling begins, and record the choice with its owner before launch.

The records from my accounts were detailed enough to look authoritative and too coarse to answer the operational question.

Record	What it establishes	What it cannot establish
Provider account A	260.6M tokens on 6 August and 760.3M on 7 August; 1.0209B across two calendar days.	How much belonged to the unattended run rather than other work on the same account.
Provider account B	567.6M tokens on 7 August across 2,369 calls in 17 sessions; 98% were cache reads.	A full run total, because the record is calendar-day scoped and the run crossed midnight.

Record	What it establishes	What it cannot establish
Provider account C	No usage total I could retrieve.	Any estimate of the fourth harness's consumption.
Combined known record	Roughly 1.59B tokens across two providers.	Run-scoped consumption, a complete total, or a defensible bill for the work.

The second provider's detail shows why token totals alone are not an economic unit. Its 7 August record contains 567.6 million tokens across 2,369 calls in seventeen sessions: 556.1 million cache reads, 8.8 million cache writes, 2.7 million output tokens, and 40.1 thousand fresh input tokens. Cache reads were 98% of the total and output was under half of one percent. That profile is measured for that account and day; it cannot be assumed for the other providers.

Before retrieving the cache profile, a conventional 80/20 input-output assumption produced a notional valuation more than an order of magnitude above a cache-heavy treatment. The latter applied one provider's measured cache shape to another provider's published rate card, so it was a cross-provider assumption rather than a measurement. Neither valuation was money spent; the run used flat subscription allowances.

This is why the worked prices have been removed from this edition. The arithmetic was correct and the result looked precise, but the decisive variable was unobserved where it mattered. Publishing two dollar figures would invite a debate over which was closer when the operational conclusion is that the system exposed no run-scoped basis for choosing either one.

That calculation is useful only because it fails. The same token volume can support materially different notional prices depending on a variable the operator cannot see consistently across providers. The product improved by my own 2.2-point estimate, but the system exposed no live burn rate, no run-level meter, and no threshold tied to value.

A flat subscription hides the signal until the allowance is exhausted. Calendar-day billing cannot reconstruct a run after the fact. The run therefore lacked both a governor and a gauge: nothing limited consumption before launch, and nothing reported consumption in time to change the decision.

There were three missing instruments. The **governor** was the hard ceiling that should have stopped the run. The **gauge** was the live run-level burn rate that should have shown how quickly the ceiling was approaching. The **meter** was the durable allocation record that should have separated the unattended run from everything else on the account. Provider dashboards supplied calendar totals; flat allowances supplied an eventual interruption. Neither supplied operational accounting.

That distinction matters in procurement. A subscription price answers what access costs the buyer. It does not answer what a particular autonomous workflow consumed, whether marginal activity was worth continuing, or which business unit should own the charge. A team can stay inside its invoice and still operate without economic control.

Measured value from AI assistance exists, but its strongest evidence has a different shape. Brynjolfsson, Li, and Raymond found a 15% productivity gain across 5,172 customer-support agents. Dell'Acqua and colleagues found that consultants working inside the technology's capability frontier completed 12.2% more tasks, worked 25.1% faster, and produced roughly 32% higher quality; outside the frontier they were 19 percentage points less likely to reach a correct answer than controls.

Those studies concern people using AI, not autonomous agent populations. Their relevance is the measurement architecture: tasks were bounded, completion was defined, and outcomes were scored outside the system doing the work. My run had none of those conditions. It authored the queue, declared completion, and generated its own review receipts. The comparison does not show that autonomy destroys value. It shows why activity counts cannot establish it.

The appropriate control is a ceiling chosen before the system begins. Nobody has published a reliable method for sizing that ceiling for a specific production workload, and the overhead and false-stop cost of governors are not well measured. That uncertainty argues for explicit judgment and monitoring. It does not justify leaving the decision unset.

A ceiling is not a forecast. It is the maximum exposure the operator is willing to accept while the forecast is uncertain. The first run can be deliberately small, instrumented, and reviewed against an external outcome. Later ceilings can be raised from observed value and consumption. Treating uncertainty as a reason to begin without a limit reverses the burden of proof.

Ceilings also need more than one dimension. A turn cap can fail when a single turn launches an expensive tool sequence. A spend cap can fail when costs are estimated late or when a flat subscription exposes no live price. A wall-clock cap can halt a slow but valuable job while allowing a fast loop to consume heavily before time expires. The safest initial envelope combines time, activity, and consumption, then stops on whichever threshold arrives first.

The value threshold is separate. A run can remain inside every resource ceiling and still produce nothing worth keeping. The operating record should therefore pair consumption with an outcome chosen before launch: resolved customer cases, externally scored defects, accepted analysis, or another measure the agent population cannot award itself. Resource governors bound exposure; an external outcome decides whether the next run deserves a larger envelope.

What external decision rights require

A sign is not a lock

Late in the run, an agent concluded that its working directory was not an enforcement boundary and paused its own process. The agent was correct. A policy stored where a broadly tooled process can read, reinterpret, or bypass it is guidance, not a control.

The UK's National Cyber Security Centre states that protections should focus on deterministic safeguards that constrain system actions rather than relying on instructions to prevent unwanted behaviour. Canada's federal agentic-AI guide makes the same placement rule concrete: the kill switch should be external to the agent, and the action log should live somewhere the agent cannot change.

The distinction is not between policy and technology. Policy names the decision right; technology enforces where it sits. A policy may say that an agent cannot create work, but enforcement requires a queue that rejects the agent's principal. A policy may require stopping at a deadline, but enforcement requires a scheduler or supervisor that the agent cannot extend. A written rule remains necessary. It becomes dependable only when the action surface embodies it.

> **PLAIN LANGUAGE: A SIGN IS NOT A LOCK**
>
> A sign works because someone chooses to respect it. A lock works whether or not they agree. Prompt rules, repository policies, and working-directory instructions are signs when the agent can reach past them. A boundary becomes a control only when the process it constrains cannot rewrite, ignore, or disable it.

Put authority outside the process it governs

The current guidance converges on five decision rights. The useful question is not how many frameworks repeat them, but which observed failure each control would have prevented.

Observed failure	Decision right	Strongest authority in this essay
The objective named coordination but no product or completion state.	**Bound** the deliverable, consumer, scope, time, turns, and spend before launch.	Canada: bounded autonomy and tight explicit parameters. NCSC: deterministic safeguards.
Four harnesses worked under one operator identity.	**Identify** every agent with a distinct principal and scoped, expiring credentials.	Five Eyes: cryptographically anchored distinct principals and ephemeral credentials.
The record carried the operator's name and could not distinguish human from agent-authored work.	**Log** actions outside the agents' reach with durable provenance.	Canada: logs in a system the agent cannot change.
Agents could create and close their own work.	**Approve** creation, closure, and high-impact state changes outside the producing system.	Canada and Five Eyes: human checkpoints and prior approval for consequential actions.
The stop condition lived inside agent-reachable instructions or was absent.	**Disable** through an external, human-reachable mechanism tested before launch.	Canada: kill switch external to the agent. Shipping SDKs: external termination interfaces.

The shared account is not a bookkeeping defect. It converted self-authored work into apparent human authorization. Each agent reading the queue saw tasks carrying my identity. Distinct principals would have made the provenance visible and allowed policy to separate human instruction from agent suggestion.

Identity also makes least privilege and recovery possible. A principal scoped to one job can receive only the tools and data needed for that job, and its credentials can expire when the job ends. A shared operator credential makes every action look equally authorized and forces revocation to interrupt the human as well as the agents. The Five Eyes guidance is unusually precise here: each agent should be a cryptographically anchored distinct principal, backed by a trusted registry and ephemeral credentials.

The five decision rights reinforce one another. Identity without external logging can still leave an editable history. Logging without approval can document unauthorized work perfectly. Approval without a tested stop cannot recover from a loop already in motion. A ceiling without a named product limits waste but cannot create value. The control map is a system, not a menu.

Each right should be tested negatively. Attempt to file work using an agent principal after the queue freeze. Attempt to alter a prior log entry. Attempt to reuse an expired credential. Attempt to continue after the ceiling. Attempt to close work without the consumer's acceptance signal. A policy review shows that the rule exists; a negative test shows that the system refuses the forbidden state even when an agent tries to reach it.

The evidence should also survive the run. A control that appears only in a live dashboard cannot support incident reconstruction or assurance. The record should preserve the objective version, delegated principals, tool grants, ceilings, approvals, rejected actions, measurements, stop reason, and final disposition. That is enough to answer who authorized the work, what bounded it, what happened, and why it stopped without retaining every internal token.

These sources describe what organizations should implement. They do not show how often the controls are deployed or whether they work consistently in production. I found no adoption dataset for per-agent identity, external kill switches, or agent-immutable logs. The guidance is specific; implementation remains unmeasured.

That evidence boundary is important because framework convergence can look like operational maturity. It is not. Several governments can recommend the same control before vendors expose it consistently, before buyers set it, and before auditors know how to test it. The right near-term question is therefore demonstrable: show the distinct principal, show the immutable record, trigger the stop, and prove that the producing agent cannot approve its own next task.

What this incident can and cannot establish

> **LIMITS OF THE EVIDENCE**
>
> • **Outcome:** the 2.2-point change is my rating of my product, with a reset during the night, no independent evaluator, and no control arm.
>
> • **Consumption:** the 1.59B known tokens are calendar-day and account totals, not a run total; one provider is missing.
>
> • **Review:** human automation-bias findings are an analogy to machine oversight, not proof of a shared mechanism.
>
> • **Research status:** MAST, agent-cost, and model-debate evidence cited here includes preprints; peer-reviewed evidence largely concerns AI assistance to people rather than autonomous agent populations.
>
> • **Causality:** the process-only objective explains the self-generating queue. A separate well-specified run was not conducted.
>
> • **Control design:** published evidence does not yet establish how to size the right ceiling or quantify governor overhead and false stops.

The incident still supports a narrower conclusion. A process-only objective can turn coordination into the product. A harness without external limits can scale that defect. Cooperative internal review can fail without leaving a product defect, and provider records can be too coarse to reconstruct what happened. Those claims fit the record and the limitations above.

A deployment test

The control mechanisms already exist in shipping SDKs and public guidance. Anthropic documents turn and dollar ceilings plus external cancellation. OpenAI documents a turn ceiling that can be disabled. AutoGen documents token, time, message, handoff, and external termination conditions. Not every harness exposes every control, and a parameter that exists is not a control until someone sets and tests it.

Defaults differ. Anthropic's documented `maxTurns`, `maxBudgetUsd`, and token task budget are unset unless the operator supplies them. OpenAI's Agents SDK accepts a maximum-turn value and can disable that limit by passing none. AutoGen's termination conditions are composed into the team configuration, including token, timeout, message, handoff, function-call, and external stops. These mechanisms bound different dimensions; a turn limit is not a spend limit, and either can be reached long after the useful work is complete.

The practical artifact should be one test, applied before an agent population starts work.

Question	Required decision	Evidence or implementation anchor
1. What is the product, and who consumes it?	Name a deliverable, consumer, success condition, and completion state.	Incident record; Merton's goal-displacement frame.

Question	Required decision	Evidence or implementation anchor
2. Who may create and close work?	Keep work creation and completion authority outside an unattended agent population.	Tickets 13–39 and the seven-minute self-close; bounded-autonomy guidance.
3. What are the ceilings?	Set hard turn, spend or token, and wall-clock limits before launch.	Anthropic `maxTurns` and `maxBudgetUsd`; OpenAI `max_turns`; AutoGen termination conditions; OWASP unbounded-consumption guidance.
4. What consumption is visible while the run is active?	Expose a run-scoped meter, burn rate, threshold, and alert tied to the ceiling.	The calendar-day billing failure in this incident.
5. Which agent acted?	Give each agent a distinct principal, scoped tools, and credentials that expire with the job.	Five Eyes distinct-principal and ephemeral-credential guidance.
6. Can the agent change the record?	Write actions, approvals, rejected attempts, and provenance to an append-only log outside agent reach.	Canada's agent-immutable logging requirement.
7. What stops the system?	Wire an external, human-reachable termination mechanism and test it before launch.	Canada's external kill switch; SDK external-termination interfaces.
8. What happens when a person is needed?	Make escalation and waiting a valid terminal state that outranks producing more work.	The overdue-task dashboard and post-freeze ticket in this record.

The test is intentionally phrased as decisions rather than product settings. A team can configure `max_turns` and still choose a number unrelated to the value at risk. It can collect logs that the same agents can edit. It can require approval after the state change instead of before it. Passing requires evidence that each decision right sits with an actor outside the producing process.

Run the eight questions at three moments. Before launch, require a named product, consumer, principals, ceilings, and tested stop. During operation, display run-scoped consumption, rejected actions, and escalation state. After termination, reconcile the immutable record against an external outcome and decide whether the workflow earned another run. That final decision belongs to the operator or consumer, not to the system that produced the work.

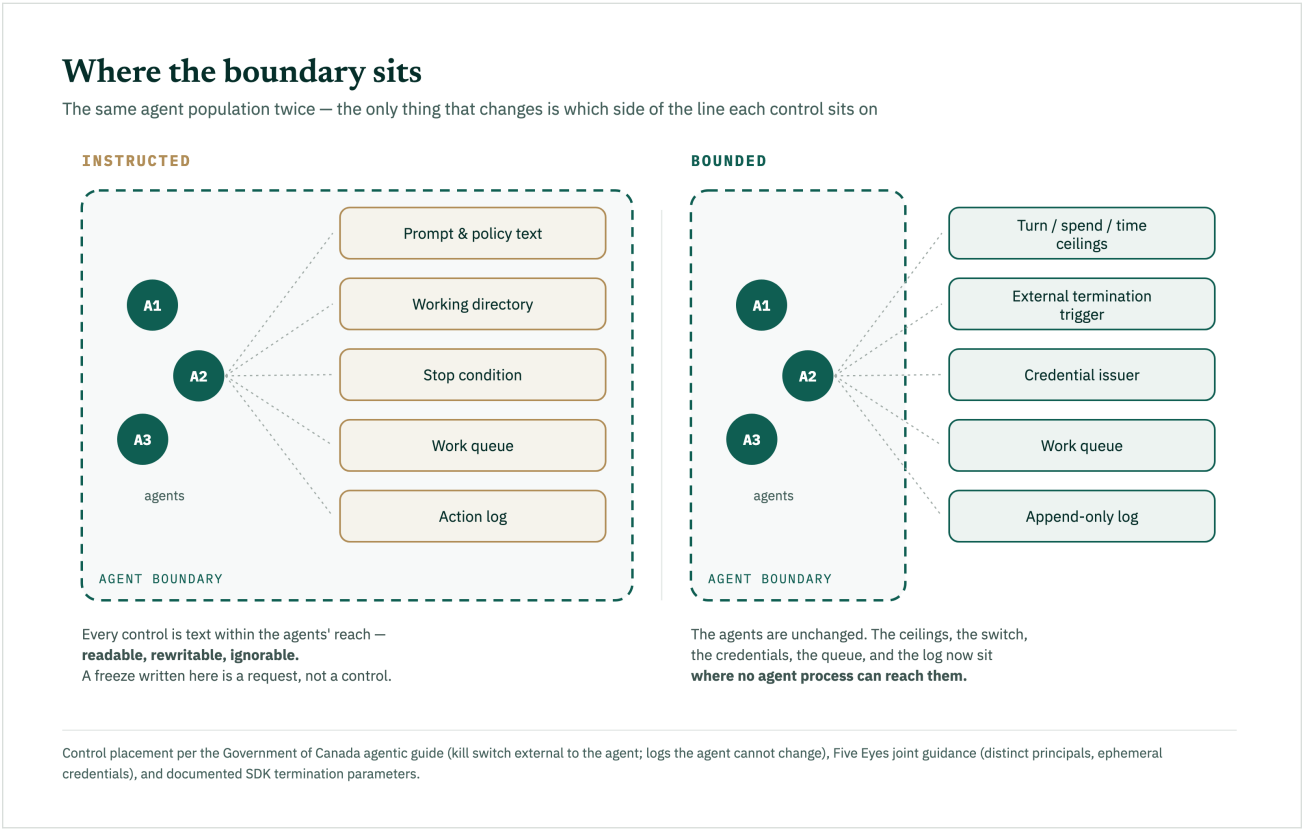

FIGURE 03 / WHERE THE BOUNDARY SITS

Two hundred and three commits, 213 receipts, 27 closed tickets, twelve review passes, and no independent measure of whether the unattended work was worth doing. The models did what they were built to do. The harnesses did what they were configured to do. The instruction named a process and never named a product.

That is the process/product test. If the system's strongest evidence of success is that it coordinated, reviewed, documented, and continued, it may have optimized its operating procedure instead of the outcome. The proper response is not to ask it to work harder. It is to remove its authority to define the next unit of work, name the product, and stop when the external measure no longer moves.

Sources

The full verification apparatus, retrieval history, and excluded-source notes belong with the web edition on trustcyber.ca. This print list contains the sources that support surviving claims. Preprints are labelled, and retrieval limits are stated where they affect verification.

Human factors and organizational control

Bainbridge, L. (1983). *Ironies of Automation.* Automatica, 19(6), 775–779. complexcognition.co.uk/2021/06/ironies-of-automation.html

Mosier, K. L., & Manzey, D. (2019). *Humans and Automated Decision Aids: A Match Made in Heaven?* In Human Performance in Automated and Autonomous Systems, 19–42. CRC Press.

Mosier, K. L., Skitka, L. J., Heers, S., & Burdick, M. (1998). *Automation Bias: Decision Making and Performance in High-Tech Cockpits.* The International Journal of Aviation Psychology, 8(1), 47–63. Accessed through Mosier & Manzey (2019).

Bailey, N. R., & Scerbo, M. W. (2007). *Automation-induced complacency for monitoring highly reliable systems.* Theoretical Issues in Ergonomics Science, 8(4), 321–348. Accessed through Mosier & Manzey (2019).

Merton, R. K. (1940). *Bureaucratic Structure and Personality.* Social Forces, 18(4), 560–568.

Multi-agent systems, review, and economics

Cemri, M., et al. (2025). *Why Do Multi-Agent LLM Systems Fail?* Preprint. arxiv.org/abs/2503.13657

Bai, et al. (2026). *How Do AI Agents Spend Your Money?* Preprint. arxiv.org/abs/2604.22750

Anthropic (2025). *How we built our multi-agent research system.* anthropic.com/engineering/multi-agent-research-system

Khan, A., et al. (2024). *Debating with More Persuasive LLMs Leads to More Truthful Answers.* Preprint. arxiv.org/abs/2402.06782

Becker, J., Rush, N., Barnes, B., & Rein, D. (2025). *Measuring the Impact of Early-2025 AI on Experienced Open-Source Developer Productivity.* METR preprint. arxiv.org/abs/2507.09089

Brynjolfsson, E., Li, D., & Raymond, L. (2025). *Generative AI at Work.* Quarterly Journal of Economics, 140(2), 889–942.

Dell'Acqua, F., et al. (2026). *Navigating the Jagged Technological Frontier.* Organization Science, 37(2).

Parasuraman, R., & Manzey, D. H. (2010). *Complacency and Bias in Human Use of Automation: An Attentional Integration.* Human Factors, 52(3), 381–410.

Governance and control placement

Government of Canada, Treasury Board Secretariat. *Guide on the Use of Agentic Artificial Intelligence,* sections 4–5. Canada.ca declined automated retrieval at press time; quotations were checked against the verified research record.

Canadian Centre for Cyber Security (2026). *Frontier artificial intelligence* (ITSAP.10.050). cyber.gc.ca/en/guidance/frontier-artificial-intelligence-itsap10050

ASD's ACSC with CISA, NSA, the Canadian Centre for Cyber Security, NCSC-NZ, and NCSC-UK (2026). *Careful adoption of agentic AI services.* cyber.gc.ca/en/news-events/joint-guidance-careful-adoption-agentic-artificial-intelligence-services

Chismon, D. (2025). *Prompt injection is not SQL injection (it may be worse).* UK National Cyber Security Centre. ncsc.gov.uk/blog-post/prompt-injection-is-not-sql-injection

OWASP GenAI Security Project (2025). *LLM10:2025 Unbounded Consumption.* genai.owasp.org/llmrisk/llm102025-unbounded-consumption

Vendor documentation

Anthropic. *Claude Agent SDK TypeScript reference:* `maxTurns`, `maxBudgetUsd`, `taskBudget`, and `abortController`. code.claude.com/docs/en/agent-sdk/typescript

OpenAI. *Agents SDK — Running agents:* `max_turns` and `MaxTurnsExceeded`.
openai.github.io/openai-agents-python/running_agents

Microsoft. *AutoGen AgentChat termination conditions:* token, timeout, message, handoff, and external termination.
microsoft.github.io/autogen/stable/user-guide/agentchat-user-guide/tutorial/termination.html

Incident provenance

Author's operational record, 6–7 August 2026. Figures, timings, tickets, message receipts, rating changes, review requests, and the shared-account history are first-party and unaudited. The review count and cross-vendor character were checked against the review record. Token figures are account and calendar-day totals, not run totals. The reviewing agent, harnesses, and providers are intentionally unnamed.

ABOUT THE AUTHOR

Junior Williams is an enterprise architect and technology advisor focused on turning complex technology, cybersecurity, and AI decisions into coherent architecture, executable roadmaps, and measurable outcomes. His work spans systems and enterprise architecture, cyber and AI risk, and the operating controls around autonomous systems. As an incoming Rogers Cybersecure Catalyst Fellow, he is advancing the questions raised here: how to size boundaries for agent populations and how to detect volume without value early enough to intervene. Areas of focus: Systems and Enterprise Architecture | Cyber and AI Risk. Book an intro call | Web: trustcyber.ca. This essay provides general professional guidance, not legal, regulatory, financial, or sector-specific advice. Organizations should adapt it to their context, obligations, risk tolerance, and applicable standards.