ACQUIRE2ACT: EVALUATING INFORMATION-TO-ACTION BOUNDARIES IN LLM AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

An agent that gathers information must decide when to stop gathering and start executing. Does its control policy adapt appropriately as evidence becomes available? We study this question with Acquire2Act, a controlled evaluation of the acquisition-to-execution boundary on 200 application, data-science, and software-engineering tasks. Acquire2Act varies source-grounded information while fixing execution infrastructure, exposing acquisition targets, verified facts, and advancement decisions. Eight controller configurations achieve 60.5–70.0% task success, versus 86.5% for a privileged information-need policy. Natural trajectories show strong responses to information progress, but reports of local target completion can coexist with premature advancement, and already avoidable acquisition targets can persist. In a matched restoration experiment on 130 tasks, supplying the complete construction-defined fact set increases advancement by 48.5 and 36.9 percentage points for two controllers. Nevertheless, they continue acquiring information on 40.8% and 34.6% of complete-information inputs. A separate ablation does not find a consistent positive effect of an explicit completion-status field. These results distinguish information responsiveness from appropriate boundary control: controllers use new evidence, yet neither observed progress nor restored facts ensure reliable acquisition exit under the fixed task and execution contract.

1 INTRODUCTION

Tool-using agents do not begin every task with all the information they need. They may have to identify a recipient, inspect an output schema, establish an aggregation convention, or locate the code that governs a requested behavior. Successful execution therefore requires a control decision in addition to a useful tool call: *is the available information sufficient to proceed, or should the agent acquire more?* Obtaining additional facts is not always beneficial. Advancing prematurely can leave a consequential requirement unresolved; continuing after the relevant facts are available can consume the execution budget without resolving a genuine need.

Existing executable benchmarks make the task outcome observable (Trivedi et al., 2024; Huang et al., 2024; Miserendino et al., 2025), while clarification benchmarks examine whether agents detect underspecification and ask useful questions (Vijayvargiya et al., 2026; Zhao et al., 2026). These are necessary measurements, but they do not by themselves characterize how a controller uses a changing information state. An actionable instruction may target the wrong uncertainty. A successful local lookup may leave another prerequisite unresolved. Conversely, a controller may continue seeking information after the facts used to construct the task's sufficient-information condition have been supplied. Final success, useful retrieval, and appropriate acquisition exit are related but distinct properties.

We introduce Acquire2Act as a controlled setting for studying this boundary. A controller receives the task, static orientation, verified facts, and compact acquisition history, then chooses ACQUIRE or ADVANCE. Acquisition and final execution use fixed, fresh executors, so the controller's targets and stopping decisions can be inspected separately from downstream task evaluation. Source-grounded information clusters define controlled initial gaps on 200 distinct parent tasks across three domains. This separation is a measurement design, not a proposed replacement agent architecture. We ask: **when the information state changes, does the controller update its acquisition and advancement policy appropriately?**

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

The evidence supports a more specific account than a general failure to use information. A privileged need-guided policy outperforms the eight tested configurations, but that gap alone does not identify a mechanism. Natural trajectories show that added facts accompany advancement even when the decision is rated premature. Restoring the original construction facts on the same 130 incomplete tasks increases advancement markedly, yet 40.8% of Sol/max and 34.6% of Grok/xhigh full-information responses still acquire. Thus, *information responsiveness exists without dependable boundary control*.

Two additional analyses constrain the interpretation. First, we distinguish entering an avoidable acquisition from persisting in it. A failed lookup is not inherently a reason to stop; persistence is particularly pronounced when the preceding target was already rated avoidable. Second, we test whether an explicit completion-status field explains advancement on genuine successor states. Its removal produces no consistent positive effect across the two controllers, limiting a simple completion-label explanation without showing that history is irrelevant.

The contribution is an observable information-control contract, matched evidence-restoration results that separate response from appropriateness, and trajectory diagnostics that distinguish local completion and avoidable-target persistence. We do not infer internal neural mechanisms or claim a universally optimal controller. Small repeatability and construction checks accompany the main evidence, with negative judgments, uncertainty, and missing trajectory records retained.

2 Related work

Executable agent evaluation. AppWorld evaluates interactive coding in a controllable application environment; DA-Code supplies executable data-science tasks; SWE-Lancer evaluates real-world freelance software engineering (Trivedi et al., 2024; Huang et al., 2024; Miserendino et al., 2025). Acquire2Act reuses selected task substrates rather than claiming a new environment. It adds controlled information states and an explicit acquisition-to-execution decision, with the scoring and execution contracts specified in Section 3.

Clarification, limit-awareness, and sufficient evidence. Ambig-SWE studies underspecification detection, targeted clarification, and the use of interaction in software engineering (Vijayvargiya et al., 2026). AskBench uses multi-turn clarification checkpoints and rubric-guided training (Zhao et al., 2026). CAR-bench studies consistency and limit-awareness under tool-use uncertainty (Kirmayr et al., 2026). The Missing Complement introduces state-conditioned, set-level sufficient-evidence recovery (Feng et al., 2026). We therefore do not claim that missing needs, targeted questions, or sufficient evidence sets are unexplored concepts. Acquire2Act's focus is the *control policy* across fixed evidence conditions, including when that policy fails to exit acquisition, rather than a new clarification strategy or evidence retriever.

Sequential reliability and stopping. Multi-turn conversation analysis finds unreliable recovery after early assumptions (Laban et al., 2025); agentic abstention treats stopping unproductive action as a sequential decision (Luo et al., 2026). Acquire2Act's ADVANCE action instead ends information gathering and authorizes execution. Empty-history restoration and a narrow status-field ablation let us distinguish the behaviors actually observed from a generic narrative of history-induced inertia.

3 Acquire2Act: observable information boundaries

3.1 Task construction and information states

Let a parent task specify an objective and execution constraints. Its H^+ representation is a set of source-grounded, plan-free facts grouped into *information clusters*: each cluster contains facts that jointly resolve a functional uncertainty. A recipient's identity and account identifier, for example, can belong to one need even when they are expressed as several facts. We construct S0 by retaining H^+, S1 by removing one qualified cluster, and S2 by removing two independently qualified clusters. Deletion includes dependent derived facts; it does not insert a visible "missing information" marker. S-level denotes the *initial construction*, not the current number of unresolved needs. One acquisition may recover both removed S2 clusters.

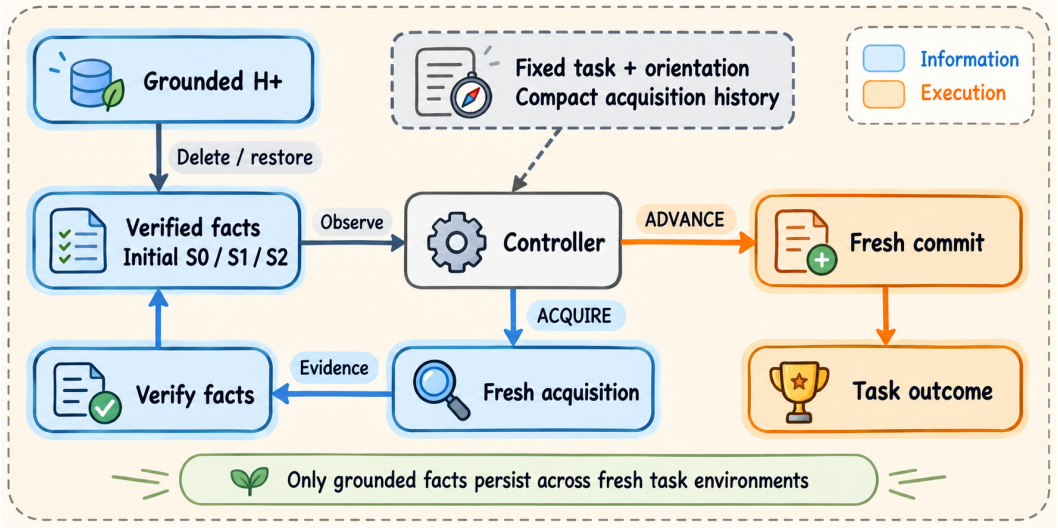


Figure 1: Acquire2Act exposes an information-control boundary. The controller observes a task, static orientation, verified facts, and compact history; it either requests information or authorizes a fresh task executor. Acquisition observations must be grounded before updating the fact set. Each acquisition and commit starts in a fresh task environment: facts, not prior mutations or raw executor reasoning, persist. S0/S1/S2 label initial constructions. The restoration experiment changes verified facts; the status ablation changes one field in genuine acquisition history.

Eligibility is tied to a fixed executor. Source construction compares H^+ with single-cluster-deleted conditions and retains success-degrading gaps or qualified efficiency gaps that preserve success but increase execution cost. The complete gate and efficiency criteria are in Appendix A. Consequently, "sufficient" here is an operational, construction-defined condition, not an assertion that every possible unknown has disappeared or that every incomplete run must fail. The release also uses domain-specific screening and calibration; Terra and Sol participated in calibration. We do not describe this selection as independent of all model behavior.

The frozen release contains 200 distinct parents: 100 AppWorld, 50 DA-Code, and 50 SWE-Lancer individual-contributor tasks. S0/S1/S2 comprise 70/72/58 parents. All subsequent analyses retain these identities; the restoration experiment uses the 130 S1/S2 parents. Domain allocations and the narrow seven-parent software-engineering S2 stratum are reported in Appendix A.

3.2 Controller and Executor Contract

At step t, the controller receives $x_t = (O, T, F_t, H_t)$: shared static orientation O, task T, verified facts F_t, and compact chronological acquisition history H_t. Its structured action is $D_t \in \{\text{ACQUIRE}, \text{ADVANCE}\}$. An ACQUIRE action specifies an unresolved uncertainty, a concrete information target with a completion condition, and a preferred acquisition instruction. An ADVANCE action supplies a sufficiency rationale. The tool schema and full instruction are reproduced in Appendix D.

An acquisition executor may inspect, query, calculate, or ask for an in-scope canonical user fact, but may not implement the final task or commit task-relevant changes. It may adapt an infeasible instruction while pursuing the requested target. Grounded observations update the verified ledger; duplicates do not automatically constitute new progress, and conflicts are not silently merged. History records targets and outcomes as diagnostic context, while F_t remains authoritative. Hidden cluster identities, reference answers, evaluator results, and construction levels are not controller inputs.

Every acquisition and the final commit use fresh environments and the same fixed executor configuration. Commit receives the task, orientation, and verified facts, not the controller's recommendations or sufficiency rationale. This intentional separation makes the boundary observable; it is

not a claim that a separated architecture outperforms interleaved agents. A run permits at most ten acquisitions, each with at most forty tool steps. Exhausting the acquisition budget is a failure, not an inferred ADVANCE decision. The first structurally and protocol-valid normal controller response is retained irrespective of semantic quality. Bounded response/transport retries are distinct from selecting among valid answers.

3.3 OUTCOMES, REVIEWED BEHAVIOR, AND STUDY POPULATIONS

Task outcomes. We count success using the domain task evaluator under the frozen adapter contract. The denominator includes task errors, behavior errors, and budget terminations; only engineering failures are excluded. All retained 200-task profile evaluations have zero remaining engineering failures. DA-Code uses the study's quality-corrected scorer, so these results are not an unmodified upstream leaderboard score. Model identifiers, output caps, executor settings, and accounting details appear in Appendix A.3.

State review. Two separate, model-identity- and outcome-blind initial reviews score each displayed state against a functional required-information card. Disagreements receive arbitration using the same rubric. Four fields distinguish timing, target alignment, handoff executability, and avoidable redundancy. Satisfaction criteria on the reference card are *not* treated as observed facts, and a later observation cannot justify an earlier decision. The initial reviewers use Sol/medium and the arbitrator Sol/xhigh; these are model-assisted judgments, not independent human-gold labels. In particular, a fully executable instruction does not establish successful retrieval. Appendix A.6 gives the scoring categories, including the rubric's distinct uses of half scores.

Analysis units. Eight primary configurations contribute 1,600 rated trajectories and 2,902 states. Their 725 multi-state trajectories contain 1,302 adjacent transitions. A supplementary Sol/medium setting brings the full collected corpus to 1,800 trajectories and 3,453 states. Repeated states are not independent tasks. We retain every valid behavior record; only analyses associating behavior with its exact final execution use the narrower 1,499-row exact-execution join. The main comparisons below do not use that join as the denominator of all state judgments.

Need-guided reference. The reference policy knows the functional descriptions of removed information needs, not their answer values. It requests the next target in a preset list and advances after the list's targets have received `target_resolved` reports. Ordinary executors still select sources and perform the task. This privileged sequential policy neither recomputes exact remaining semantic coverage nor constitutes an optimal dynamic-controller upper bound.

4 WHAT CHANGES AFTER AN ACQUISITION?

4.1 A REFERENCE GAP MOTIVATES CONTROL-LEVEL ANALYSIS

Table 1 reports task outcomes under the recorded profile settings. The need-guided policy succeeds on 173/200 tasks (86.5%), compared with 121–140 successes (60.5–70.0%) for the eight primary configurations. Its other outcomes are 23 native task failures and four acquisition-cap terminations. In particular, its 196 completed evaluations are not 196 successful tasks. The gap is a reference for attainable performance under useful need descriptions, not evidence that one isolated controller component explains every failure. On the outcome-conditioned subset of 34 tasks failed by all eight profiles, the reference solves 25. We use this subset diagnostically, not as an unbiased treatment-effect estimate.

4.2 PROGRESS ELICITS A RESPONSE WITHOUT GUARANTEEING SUFFICIENT EVIDENCE

For each adjacent pair of controller states, we compute

$$\Delta F_t = F_{t+1} \setminus F_t, \tag{1}$$

after whitespace normalization, preserving case and punctuation. This describes newly recorded fact strings, not a proof of novel semantic content or global sufficiency. We then connect the fact change to the actual next action, acquisition report, and existing state review.

Table 1: Task outcomes under the recorded configuration budgets. Each row contains 200 parents; budget terminations count as errors. The last row has privileged information-need knowledge. These are not budget-matched rankings or official upstream leaderboard scores.

Controller / policy	Output cap	Task success	Budget errors
Terra/high	24K	130/200 (65.0%)	26
Sol/max	24K	123/200 (61.5%)	36
Gemini/high	24K	140/200 (70.0%)	0
DeepSeek/high	64K	131/200 (65.5%)	7
GLM/max	64K	139/200 (69.5%)	0
Opus/max	64K	138/200 (69.0%)	1
Grok/xhigh	64K	140/200 (70.0%)	4
Kimi/max	64K	121/200 (60.5%)	31
Need-guided reference	—	173/200 (86.5%)	4

Advancement occurs after 647/981 transitions with added facts (66.0%), but only 21/321 without added facts (6.5%; Figure 2a). Thus controllers do respond to progress. However, 316 of the 647 advances after added facts (48.8%) are rated premature. The issue is not simply that information changes leave control unchanged; *responding to progress and responding appropriately are different properties*.

The target and handoff dimensions further separate. Across 1,244 non-S0 acquisition states, only 615 targets (49.4%) receive full alignment, whereas 1,177 instructions (94.6%) receive full executability. A controller can formulate a usable instruction without selecting the remaining information that matters. This comparison concerns two review dimensions; it does not show that acquisition execution itself is error-free.

Among transitions that add facts, a `target_resolved` report is followed by advancement in 556/628 cases (88.5%), versus 91/353 (25.8%) for an unresolved report. Yet 166/233 advances after resolved reports in S2 are rated premature (71.2%). Even when the preceding target receives full alignment, 71/134 such S2 advances remain premature (53.0%). Local target completion and overall task-information sufficiency can therefore diverge.

This association is not explained solely by counting many transitions from a few long trajectories. Restricting each trajectory to its first transition and comparing supported profile-by-domain-by-initial-level strata yields a resolved-minus-unresolved advancement difference of 50.4 percentage points (95% case-bootstrap interval [40.7, 59.8]). Appendix B.2 reports the overlap and leave-one-profile checks. These remain observational: target choice, recovered content, and report accuracy are jointly determined by earlier behavior. They motivate, rather than replace, the interventions below.

4.3 An extra acquisition is different from a persistent detour

We identify each trajectory's *first* acquisition rated avoidable, rather than count every repeated state as another independent onset. Of 1,600 trajectories, 177 enter avoidable acquisition. Among the 161 with a subsequent controller observation, 72 continue with another avoidable acquisition (44.7%), while 81 advance correctly, six make another advancement decision, and two acquire without an avoidability label. The remaining sixteen have no next observed state; we do not fabricate persistence or exit for them.

Fact stagnation alone is a poor stopping criterion. At a trajectory's first stall, when the preceding acquisition was not already rated avoidable, only 9/42 continued acquisitions are avoidable (21.4%). When it was already avoidable, the corresponding count is 41/42 (97.6%; Figure 2b). Reasonable retries and persistent pursuit of a nonessential target should therefore not be collapsed into one "no progress" failure.

The phenomenon is heterogeneous. Sol/max contributes 216 of the 321 no-new-fact transitions and Terra contributes 57; this is not an equally prevalent defect of all configurations. Moreover, only five of 408 avoidable repeated acquisitions reuse the previous target text verbatim, and only four

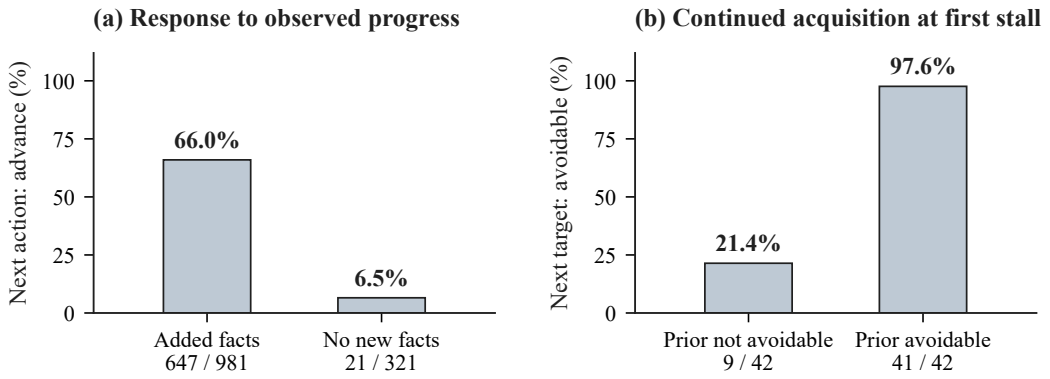


Figure 2: Information progress and control appropriateness are distinct. (a) Actual next-step advancement conditional on whether the verified fact set grew. (b) Among continued acquisitions at each trajectory's first fact stall, reviewed avoidability depends strongly on whether the preceding acquisition was already avoidable. Numerators and denominators are shown; the two panels use different selected sets. Bars are descriptive proportions, not independent-state estimates with inferential error bars. Panel (b)'s appropriateness labels depend on the existing model-assisted review.

reuse the instruction verbatim. Surface rewording is not evidence that the underlying target has been corrected. These findings describe observed functional behavior without identifying a hidden memory or training mechanism.

5 Controlled information restoration

5.1 Matched interventions and estimands

Natural trajectories do not isolate the effect of the available information: the controller chooses the preceding target, and the executor determines what is recovered. We therefore construct a controller-only restoration probe, C1, on all 130 S1/S2 parents. Within a task and profile, we hold the objective, constraints, static orientation, original controller instruction, tool schema, and empty acquisition history fixed. Only verified facts change, using the task's original H^+ facts and frozen deletion closure. S1 has missing and full conditions. S2 additionally has two single-cluster restorations, denoted g_1 and g_2. Neither condition names nor expected actions are supplied to the controller.

There are 376 task-condition inputs and two tested configurations, Sol/max and Grok/xhigh, both with a 64K output cap. The primary analysis contains 752 predesignated first-request slots. A fixed twelve-parent pilot uses three requests per condition and profile; its first requests belong to the primary slots, and the additional repeats remain separate. The 216 pilot requests and 680 remaining primary requests give 896 total observations. No response is chosen for semantic quality, and no acquisition, commit, or native task evaluator is run in C1. The outcome is the controller's action, *not task success*.

Let $a_{im}(z) \in \{0, 1\}$ indicate ADVANCE for parent i, profile m, and evidence condition z. We report two predesignated contrasts:

$$\widehat{\Delta}_m^{\text{restore}} = \frac{1}{130} \sum_{i=1}^{130} \big[a_{im}(\text{full}) - a_{im}(\text{missing})\big], \tag{2}$$

$$\widehat{\Delta}_m^{\text{complete}} = \frac{1}{58} \sum_{i \in \mathcal{I}_2} \left[a_{im}(\text{full}) - \frac{a_{im}(g_1) + a_{im}(g_2)}{2}\right], \tag{3}$$

where $\mathcal{I}_2$ contains the 58 S2 parents. Confidence intervals use 5,000 parent resamples stratified by domain and initial S-level, with repeated requests averaged within parent when present. Percentile intervals describe the specified cohort and sampling procedure, not held-out-task or changing-

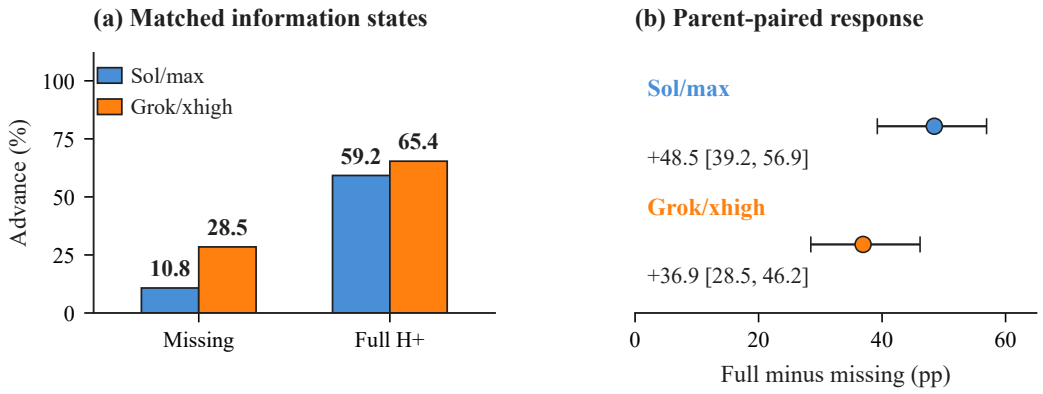


Figure 3: Controllers respond to restored facts but do not consistently exit acquisition. (a) Advancement proportions on the same 130 parents in the missing and full-H^+ conditions. (b) Parent-paired full-minus-missing differences; whiskers are the original 95% stratified parent-bootstrap intervals. The left panel describes the two endpoints; the right establishes the matched response. Colors denote profiles consistently. These decisions are independently sampled in each condition and are not measurements of task-success improvement.

endpoint generalization. The interventions change fact content and length together; there is no length-matched irrelevant-fact control.

5.2 Responsiveness is strong, yet full-information acquisition remains

For Sol/max, advancement increases from 14/130 (10.8%) to 77/130 (59.2%), a paired change of **48.5 percentage points** (95% interval [39.2, 56.9]). For Grok/xhigh, it increases from 37/130 (28.5%) to 85/130 (65.4%), a change of **36.9 points** ([28.5, 46.2]; Figure 3). The controller policy is plainly sensitive to information restoration.

Nevertheless, the complete H^+ condition still elicits ACQUIRE in 53/130 Sol responses (40.8%) and 45/130 Grok responses (34.6%). The two endpoints are important: Sol's larger restoration contrast does not imply superiority at every state. Grok starts with a greater tendency to advance under missing information, while its full-information advancement rate is only modestly higher. Responsiveness and advancement propensity are not interchangeable.

Matched response patterns also show why the net contrast is not a count of "rescued" tasks. Sol has 65 parents with missing-ACQUIRE/full-ADVANCE, 51 with acquisition at both endpoints, twelve with advancement at both, and two with the reverse pattern. Grok's corresponding counts are 51, 42, 34, and three. These are pairings of independent requests on the same parent; they are not a model instance changing its mind along one trajectory or directly observed individual causal effects. Their role is to expose the responses hidden by an aggregate difference.

5.3 Partial completion is distinguishable, but not uniformly

On the same 58 S2 parents, Sol advances in 5.2% of missing, 15.5% of single-cluster-restored (averaged), and 58.6% of fully restored inputs. Grok's corresponding rates are 22.4%, 31.9%, and 65.5% (Figure 4). Full-minus-average-partial effects are 43.1 points ([30.2, 56.0]) and 33.6 points ([20.7, 45.7]). Thus, the natural-trajectory early-advancement result cannot be summarized as an inability to distinguish partial from complete information at all.

However, the matched four-condition pattern that advances *only* under full restoration occurs on just 21/58 Sol and 14/58 Grok parents. We use this as a descriptive reliability check, not as evidence for or against an internal logical-AND mechanism. Full-information behavior is also domain-dependent: both profiles advance on only 14/34 DA-Code parents, versus 39/67 and 48/67 App-

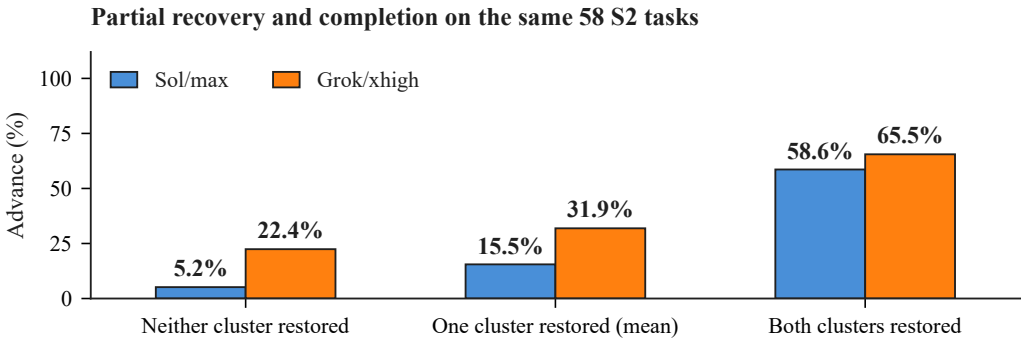


Figure 4: Restoring both S2 clusters changes the policy more than restoring one, on average. Bars use the same 58 parents; "one" averages the two single-cluster conditions. The full-minus-average-partial effects are 43.1 points for Sol (95% interval [30.2, 56.0]) and 33.6 for Grok ([20.7, 45.7]). Bars are descriptive fixed-cohort proportions; uncertainty is reported for the paired contrast. Conditions are independently sampled inputs, not successive acquisition rounds.

World parents, and 24/29 and 23/29 software-engineering parents. Appendix B.3 retains the complete domain and response-pattern tables.

5.4 What do controllers request after restoration?

We inspected all 98 primary full-H^+ acquisition responses, covering 67 distinct parents. An AI-assisted descriptive reading grouped their requested information into credentials (24), task-parameter or intent binding (14), scope/freshness/feasibility (seven), interface semantics (two), guidance/format/computation conventions (forty), and source/repair/validation context (eleven). These categories describe what is requested, not whether every request is unnecessary or hallucinated. Some responses acknowledge the supplied task facts and still require another operational prerequisite before proceeding.

This yields a bounded functional interpretation: residual acquisition can reflect the elevation of additional unknowns into prerequisites, rather than simple failure to notice all supplied facts. It remains an interpretation of external responses, not an observation of latent beliefs or a validated cause of every decision. Because every C1 history is empty, a prior multi-round acquisition failure is not a necessary condition for this behavior. History may still matter in natural trajectories; subtracting their rates from C1 would not identify that effect because the factual inputs differ.

The findings are not carried solely by the pilot or the audited concerns. Excluding the predesignated twelve pilot parents gives restoration effects of 46.6 and 33.9 points on 118 parents. A separate, conservative post-audit sensitivity excludes every non-S0 parent with any negative or uncertain construction judgment and gives 46.8 and 36.5 points on 126 parents. These analyses do not replace the original 130-parent primary result or establish that unaudited tasks are defect-free.

6 Does the completion label explain advancement?

The strong natural association between `target_resolved` and Advance suggests a simple candidate account: the explicit completion label itself might encourage advancement. C2 tests this account on 41 genuine first-acquisition successor states, including 24 resolved states (the primary slice) and seventeen unresolved states. The state selection does not use the subsequent action, its review, C1 results, or task success. Both tested profiles receive the same state set, originating from 27 Sol and fourteen Grok trajectories.

For each state, we either retain or omit only `outcome.status`. Facts, target, preferred instruction, counters, and failure or source-closure text remain unchanged. Both conditions are newly

Table 2: Explicit completion-status ablation on the 24 resolved successor states. Each condition contains two requests per parent and profile; the difference is retained minus omitted advancement, in percentage points. Intervals are 95% parent-bootstrap intervals after averaging repeats within parent.

Profile	Retained	Omitted	Difference	95% interval
Sol/max	41/48 (85.4%)	39/48 (81.2%)	+4.2	[-4.2, 14.6]
Grok/xhigh	40/48 (83.3%)	42/48 (87.5%)	-4.2	[-16.7, 6.2]

sampled twice per profile, producing 328 requests. We average repeats within parent before the stratified parent bootstrap. Unlike changing a success report to a false failure, this intervention does not introduce fabricated historical events.

Table 2 gives opposite-signed primary estimates, with both intervals including zero. The data do not support a consistent positive advancement effect of the explicit completion field. This is not equivalence: the intervals permit nonzero effects, and other text still conveys progress. The ablation limits one explanation of the natural association, not the role of the entire history. Appendix B.4 retains secondary unresolved-state results and small-stratum limitations.

7 Discussion and Limitations

The results distinguish *information progress*, *task-information sufficiency*, and *the decision to proceed*. Natural progress and controlled restoration elicit strong responses without ensuring consistent acquisition exit. An avoidable target can also persist under rewording. These are behavioral distinctions, not evidence for a specific hidden memory mechanism or a new architecture.

The supporting reliability checks are deliberately limited. Two complete-run repetitions on nine fixed parents produce task-outcome agreement of 6/9 for Sol and 7/9 for Grok. Native outcomes are retained for all 36 runs, but structured trajectories exist for only 31, and earlier recovery records are incomplete. The eighteen-parent construction audit has initial agreement of 76/84 applicable judgments (90.5%, nominal $\kappa = 0.253$); consulting a third return for eight disagreements yields 79 passes, one failure, and four unresolved judgments. Appendix C.2 preserves source-coverage and presentation concerns rather than interpreting agreement as proof of correctness.

Several boundaries limit generalization. The corpus is finite and calibration-informed; the executor and stage separation are fixed; routed model identifiers and different output caps do not establish a budget-matched capability ranking. State appropriateness uses model-assisted review from one model family, not independently validated human gold. Natural transitions remain selected by earlier policies and recovered content. C1 does not isolate fact content from length or observe an individual's counterfactual trajectory, and C2 tests only one field on a small state set. The audit does not validate every task or state label. These limits restrict the conclusions to the stated measurement contract without erasing the positive restoration response, residual acquisition, or negative explanatory result.

8 Conclusion

Acquire2Act makes the acquisition-to-execution boundary observable. Controllers respond to restored information, but 40.8% and 34.6% of complete-information inputs still elicit acquisition. Natural traces reveal persistent avoidable targets; a narrow status ablation does not validate a universal completion-label account. Agent evaluation should therefore measure both information responsiveness and the appropriateness of the resulting control decision, rather than treating information gain or task success as substitutes.

AI USE STATEMENT

Generative AI assisted research planning, implementation, information construction and verification, analysis code, qualitative reading, literature checking, interpretation, manuscript drafting, and figure-generation code. Figure 1 uses an AI-assisted illustrative rendering of the study's protocol; all quantitative figures remain deterministic plots of the recorded data. The experimental controllers, fixed executors, user-fact simulator, and model-assisted state reviewers are separately specified in Appendix A.3. AI-assisted qualitative categories are not presented as independently validated human annotations. Manuscript preparation did not add experiments, fill missing trajectories with generated content, or invent result values. AI assistance does not transfer responsibility for the paper's claims, citations, and disclosures away from its authors.

ETHICS STATEMENT

Acquire2Act studies execution in controlled benchmark environments. Its application actions and software modifications are experimental task operations, not a validation of safe deployment against real user accounts. The explicit separation of information gathering and execution is a measurement device; it should not be interpreted as a safety guarantee for autonomous systems. The construction audit concerns technical task materials and is reported with unresolved and negative judgments retained. The minimal annotation exports do not establish reviewer qualifications, independent working conditions, or the absence of AI assistance, and no such claim is made. Any future release of benchmark materials must respect the original task sources and their licenses; this submission does not redistribute complete upstream environments or model credentials.

REPRODUCIBILITY STATEMENT

Appendix A gives construction eligibility and cohort counts; Appendix A.3 specifies role visibility, models, budgets, and response handling; Appendix D includes the controller and review instructions. The appendices provide additional protocol and result details, including the matched restoration contrasts, for checking the reported analyses. Exact re-execution of the original interactive experiments additionally depends on the recorded hosted-model routes, fixed task environments, and evaluation adapters. We do not claim that hosted model weights are immutable or that the manuscript alone reproduces those external services. Missing R1 trajectory records and incomplete historical recovery costs remain explicit rather than being silently reconstructed.

REFERENCES

Zhexi Feng, Ruiyi Zhang, Yongbo Yang, and Pengtao Xie. The missing complement: State-conditioned minimal sufficient evidence for coding agents. *arXiv preprint arXiv:2609.20050*, 2026. URL https://arxiv.org/abs/2609.20050.

Yiming Huang, Jianwen Luo, Yan Yu, Yitong Zhang, Fangyu Lei, Yifan Wei, Shizhu He, Lifu Huang, Xiao Liu, Jun Zhao, and Kang Liu. DA-Code: Agent data science code generation benchmark for large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 13487–13521, 2024. doi: 10.18653/v1/2024.emnlp-main.748. URL https://aclanthology.org/2024.emnlp-main.748/.

Johannes Kirmayr, Lukas Stappen, and Elisabeth André. CAR-bench: Evaluating the consistency and limit-awareness of LLM agents under real-world uncertainty. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, pp. 40599–40618, 2026. doi: 10.18653/v1/2026.acl-long.1886. URL https://aclanthology.org/2026.acl-long.1886/.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. LLMs get lost in multi-turn conversation. *arXiv preprint arXiv:2505.06120*, 2025. URL https://arxiv.org/abs/2505.06120.

Han Luo, Bingbing Wen, and Lucy Lu Wang. Agentic abstention: Do agents know when to stop instead of act? *arXiv preprint arXiv:2606.28733*, 2026. URL https://arxiv.org/abs/2606.28733.

Samuel Miserendino, Michele Wang, Tejal Patwardhan, and Johannes Heidecke. SWE-Lancer: Can frontier LLMs earn $1 million from real-world freelance software engineering? *arXiv preprint arXiv:2502.12115*, 2025. URL https://arxiv.org/abs/2502.12115.

Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. AppWorld: A controllable world of apps and people for benchmarking interactive coding agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pp. 16022–16076, 2024. doi: 10.18653/v1/2024.acl-long.850. URL https://aclanthology.org/2024.acl-long.850/.

Sanidhya Vijayvargiya, Xuhui Zhou, Akhila Yerukola, Maarten Sap, and Graham Neubig. AmbigSWE: Interactive agents to overcome underspecificity in software engineering. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2502.13069.

Jiale Zhao, Ke Fang, and Lu Cheng. When and what to ask: AskBench and rubric-guided RLVR for LLM clarification. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 17120–17140, 2026. doi: 10.18653/v1/2026.findings-acl.845. URL https://aclanthology.org/2026.findings-acl.845/.

A CONSTRUCTION, POPULATION, AND EXECUTION CONTRACT

A.1 PARENT TASKS AND INFORMATION CLUSTERS

Table 3 gives the full released population. A parent is the underlying task; neither another controller configuration nor another observed state creates a new parent. The software-engineering tasks are SWE-Lancer individual-contributor tasks, not manager tasks. The final population is held fixed during the reported restoration, status-ablation, repeatability, and audit analyses.

Table 3: Frozen parent-task population by domain and initial information level.

Domain	S0	S1	S2	Total
AppWorld	33	33	34	100
DA-Code	16	17	17	50
SWE-Lancer IC	21	22	7	50
Total	70	72	58	200

H^+ consists of source-supported observations, task parameters, interface facts, and other information relevant to performing the task. It excludes an executable plan presented as a fact. A cluster is a semantic unit: its information points jointly resolve one uncertainty, such as which account a named recipient denotes. Information points can have multiple source references. Removing a cluster removes its dependent derived facts as well. The hidden cluster identifiers are construction bookkeeping, not model-visible cues.

S1 removes one qualified cluster, while S2 removes two clusters that satisfy the single-cluster qualification criterion independently. This does not require two acquisition calls: one broad enough observation can satisfy both needs. Nor does it mean that every S2 task received a separate four-arm execution-outcome experiment. The four-state C1 experiment is a controller-decision comparison, not a retrospective claim that all joint-deletion construction outcomes were independently replicated.

A.2 ELIGIBILITY AND CALIBRATION

The standard construction gate uses three executor runs, with designated boundary cases using five. The complete-information condition must succeed in at least two of three or four of five runs, respectively. Under the *success channel*, deleting a cluster must cause a strict decrease in success relative to its protected H^+ comparator. Under the *efficiency channel*, success remains equal and protected, neither the mean query count nor mean token count worsens when information is present, and at least one qualifying improvement holds: a query reduction of at least one and 20%, or a token reduction of at least 20%. The cost comparisons use the corresponding successful trials. These criteria establish operational execution relevance; they do not establish logical necessity for every realized execution or mathematical minimality among all possible evidence sets.

The release combines domain-specific screening with calibration, including four rounds of controlled AppWorld recalibration. Terra and Sol participate in calibration. Therefore, the task pool is not a held-out evaluation with respect to every controller's behavior. All final identities and levels are frozen before the new controlled probes, and no task is replaced because of its C1/C2 effect or an unfavorable audit vote. The final IC S2 capacity is seven parents; additional profile evaluations do not increase that stratum's number of independent tasks.

A.3 ROLE-SPECIFIC VISIBILITY AND STATE TRANSFER

The controller sees the same static orientation as both executors. The orientation describes the prepared task environment and remains unchanged across rounds. The task specifies the objective and constraints. Verified information is rendered as fact text, without private source bookkeeping. Compact history records prior uncertainty, acquisition target, preferred instruction, and returned outcome. A grounded-information field mirrors verified observations; failure reports retain unavailable-source information. Raw executor reasoning and unexecuted response tails are not replayed to the controller.

An ACQUIRE action can request inspection, reading, querying, calculation, reproduction of existing behavior, or an in-scope question to the task user. It cannot request implementation of the final code change, submission of a deliverable, or validation of a change made during acquisition. The acquisition executor may use a different feasible information-gathering method when the preferred instruction is unproductive. The final commit sees the task and verified facts, not the controller's target, instruction, or sufficiency rationale. It starts in a fresh task environment, as does each acquisition. Information, rather than intermediate task mutations, is therefore the persistent inter-stage object.

The ledger checks provenance, avoids duplicate insertion, and does not silently accept conflicting facts with the same identity. A new fact string is still not an automatic semantic proof that a required cluster has been covered. The analysis uses the exact retained facts and existing review to keep those two questions separate.

The simulated user is limited to canonical user-provided facts. Connected-service state, source code, technical environment help, and evaluator information are not user knowledge merely because a human could hypothetically copy them. Normal `not_known` and `out_of_scope` answers are retained and close the equivalent item from that user source. Later acquisition must respect this closure. AppWorld's allowed-action contract executes only the first valid allowed block; ignored response tails are not inserted into effective history.

A.4 RECORDED CONFIGURATIONS AND BUDGETS

Table 4 lists the requested model identifiers for the recorded routes, not independently certified immutable model weights. The analytical data includes the corresponding profile-key mapping. The controlled C1/C2 probes use Sol/max and Grok/xhigh at a common 64K output cap; they are not a new 64K baseline rerun of the historical Sol 24K tasks. Complete-run R1 repetitions retain the original Sol 24K/Grok 64K conditions.

Table 4: Recorded controller and infrastructure configurations. Caps are per-response output-token limits, not total context or whole-task budgets.

Role or display profile	Requested model	Reasoning	Cap
Terra	`gpt-5.6-terra`	high	24K
Sol	`gpt-5.6-sol`	max	24K
Sol (supplementary)	`gpt-5.6-sol`	medium	24K
Gemini	`gemini-3.8-flash`	high	24K
DeepSeek	`deepseek-flash`	high	64K
GLM	`glm-5.3`	max	64K
Opus	`claude-opus-5`	max	64K
Grok	`grok-4.6`	xhigh	64K
Kimi	`kimi-k3`	max	64K
Acquisition / commit	`gpt-5.6-luna`	low	24K
User-fact simulator	`gpt-5.6-sol`	xhigh	—
Initial state reviewer	`gpt-5.6-sol`	medium	64K
State-review arbitrator	`gpt-5.6-sol`	xhigh	64K

The acquisition/commit configuration uses temperature 1. A complete chain allows at most ten fresh acquisition rounds; each acquisition allows forty tool steps and 1,800 seconds. The provider-attempt timeouts are 900 seconds for the controller and user simulator and 600 seconds for the executor. Later shared response handling allows fifteen extra attempts with waits of 2, 4, 8, 16, 32 seconds followed by 60 seconds; earlier frozen runs retain their recorded ten-extra-attempt setting. These operational differences are recorded rather than overwritten in historical manifests. Nested client retries are not treated as additional semantic samples. A normal valid answer is retained even when it is poor; bounded retry handles transport or response-validity failure, not disagreement with the research hypothesis.

A.5 TASK SCORING AND ACCOUNTING

The reported task-success fraction is

$$\frac{N_{\text{success}}}{N_{\text{success}} + N_{\text{task/behavior error}} + N_{\text{budget termination}}}. \tag{4}$$

All retained primary profile denominators are 200 because no engineering failure remains. The reference policy's 196 evaluated tasks include 23 task errors; its other four outcomes are acquisition-cap terminations. DA-Code follows the frozen quality-corrected scoring policy, not an assertion of an official upstream result. Appropriate controller behavior does not guarantee that the fixed executor will complete the task successfully.

Usage reports concern provider-reported input/output tokens on retained effective attempts. They are not a reconstruction of every historical recovery, rejected transmission, or the study's total monetary cost. The baseline analytical table has complete 200-task stage-cost coverage for seven profiles, but only 99 comparable Terra rows; we omit a full-population Terra cost mean rather than silently treating that subset as 200. Missing monetary fields are not filled with estimates. The repeatability cost totals similarly exclude intermediate attempts whose records were cleared.

A.6 MODEL-ASSISTED STATE REVIEW

Packets omit model/run identity, initial S-level, task success, evaluator output, budget termination, failure class, and token cost. They provide the displayed facts for each state and a functional required-information card. Two initial Sol/medium reviews and Sol/xhigh arbitration share the same rubric. The full nine-setting collection uses 3,600 initial rating requests for 1,800 packets; effective consensus consists of 1,642 initial agreements, 140 regular arbitrations, and 18 semantic-recovery arbitrations. Deterministic consistency checks do not certify all semantic judgments.

Timing takes values 0, 0.5, and 1. If a required item remains, ACQUIRE is the correct direction and ADVANCE has timing 0. If all required items are satisfied, ADVANCE has timing 1 and unnecessary ACQUIRE has timing 0.5. A packet-indeterminate direction can also have timing 0.5, so half scores are not universally uncertainty labels. Target alignment has the same three numerical values. For ACQUIRE, it assesses the target's relation to the remaining functional need; for ADVANCE, it reflects whether a required item remains. Handoff executability assesses concreteness, boundedness, and available tools without using hidden executor results, and is null for ADVANCE. Avoidable redundancy is a separate Boolean judgment for an unnecessary, already-satisfied, or irrelevant acquisition; it is false for ADVANCE.

This is an outcome-blind model-assisted annotation protocol. It is neither model-family-independent assessment nor a human-gold state-label study. The separate construction audit examines the underlying material, not every controller state or the automatic reviewer's calibration.

B ADDITIONAL NUMERICAL RESULTS

B.1 TASK OUTCOMES AND OBSERVED COST

Tables 5 and 6 retain the domain and initial-level breakdowns behind the aggregate task result. They are stratifications of the frozen release: an S0 versus S2 task comparison is not a same-parent information intervention. The supplementary Sol/medium configuration succeeds on 124/200 tasks, with 24 budget terminations; it is not inserted into the eight-profile primary behavioral corpus.

The cost decomposition in Table 7 is descriptive. For example, Sol has more acquisitions on average than the need-guided policy, but some other controllers acquire less often than that policy. Consequently, "all model controllers use more acquisition or cost than the reference" is not a supported generalization. A low acquisition count can also coexist with premature progression or a task failure; it is not by itself an efficiency certificate. The reference has no model-based controller token cost, but still incurs acquisition and commit cost.

Table 5: Task success by domain, with identical parent denominators across profiles. All cases remain in the scored population, including budget terminations.

Profile / policy	AppWorld	DA-Code	SWE-Lancer IC	Total
Terra/high	55/100	39/50	36/50	130/200
Sol/max	54/100	31/50	38/50	123/200
Gemini/high	57/100	47/50	36/50	140/200
DeepSeek/high	50/100	47/50	34/50	131/200
GLM/max	60/100	44/50	35/50	139/200
Opus/max	61/100	46/50	31/50	138/200
Grok/xhigh	59/100	43/50	38/50	140/200
Kimi/max	43/100	45/50	33/50	121/200
Need-guided reference	81/100	48/50	44/50	173/200

Table 6: Task success by initial construction level. This is a between-task stratification, not the same-task intervention used in C1.

Profile / policy	S0	S1	S2
Terra/high	65/70	42/72	23/58
Sol/max	61/70	43/72	19/58
Gemini/high	64/70	45/72	31/58
DeepSeek/high	66/70	38/72	27/58
GLM/max	68/70	41/72	30/58
Opus/max	66/70	39/72	33/58
Grok/xhigh	63/70	41/72	36/58
Kimi/max	65/70	33/72	23/58
Need-guided reference	66/70	59/72	48/58

B.2 Natural-transition robustness and onset denominators

The primary natural analysis includes 1,302 transitions, not 1,302 independent tasks. Taking only the first transition from each multi-state trajectory yields 725 observations. Within the first-transition subset that adds facts, resolved reports precede advancement on 477/533 cases (89.5%) and unresolved reports on 59/161 (36.6%). There are 44 profile-by-domain-by-initial-level strata with comparison support. The overlap-adjusted difference is 50.4 points with a 95% case-bootstrap interval [40.7,59.8]. Leave-one-profile differences range from 44.3 to 52.3 points. These restrictions mitigate observed mixture and repeated-state weighting, but cannot eliminate differences in the acquired facts, targets, or reliability of the reports.

The onset denominator is different again. The 177 affected trajectories have 129 initially avoidable and 48 later-onset acquisitions. Of 161 with a next observed decision, 72 continue avoidably, 81 advance correctly, six make another reviewed advancement, and two continue acquiring without an avoidable label. Sixteen endings are unobserved with respect to that next decision. Among S0 onsets, 23/98 observed next decisions continue avoidably; among non-S0 onsets, 49/63 do so. The latter set can contain a genuinely missing need together with an avoidable chosen target, so global information insufficiency and local acquisition avoidability are not opposites. We do not turn these selected proportions into a causal persistence probability for all tasks.

B.3 Restoration by domain and matched response pattern

Table 8 presents each condition's actual denominator. Full and missing observations cover 67 application, 34 data-science, and 29 software-engineering parents. Single-cluster restorations cover only the corresponding S2 subset (34,17,7). Comparisons across columns must respect this difference; the completion estimand in Equation 3 uses the same 58 parents on both sides.

The missing/full patterns in Table 9 explain the mean effect exactly. For Sol, the 65 A/V pairs minus two V/A pairs give $63/130 = 48.5$ points; for Grok, 51 minus three give $48/130 = 36.9$

Table 7: Observed effective-chain cost coverage. Token columns are thousands of provider input/output tokens. Terra's incomplete stage-cost subset is not presented as a 200-task mean. Costs do not include every historical recovery or imply a universal efficiency ranking.

Profile / policy	Coverage	Mean acq.	Mean tokens	Median tokens
Terra/high	99 / 200	—	—	—
Sol/max	200 / 200	2.42	236.5	122.3
Gemini/high	200 / 200	0.46	171.3	70.2
DeepSeek/high	200 / 200	0.52	135.7	57.9
GLM/max	200 / 200	0.52	132.8	65.5
Opus/max	200 / 200	0.64	146.1	64.1
Grok/xhigh	200 / 200	0.89	236.6	76.2
Kimi/max	200 / 200	0.34	94.8	46.9
Need-guided reference	200 / 200	1.19	128.7	63.8

Table 8: C1 advancement by domain and condition. Missing/full include all non-S0 parents, whereas each single-cluster arm includes only S2 parents; these column denominators intentionally differ.

Profile	Domain	Missing	Only g_1	Only g_2	Full
Sol/max	AppWorld	5/67	5/34	5/34	39/67
Sol/max	DA-Code	1/34	2/17	1/17	14/34
Sol/max	SWE-Lancer IC	8/29	2/7	3/7	24/29
Grok/xhigh	AppWorld	21/67	12/34	8/34	48/67
Grok/xhigh	DA-Code	2/34	1/17	7/17	14/34
Grok/xhigh	SWE-Lancer IC	14/29	4/7	5/7	23/29

points. Neither endpoint pattern alone is an end-to-end task-success event. Table 10 retains every observed four-condition pattern, including nonmonotone patterns, instead of selecting only cases consistent with a logical threshold. A single sampled response per primary condition cannot identify an individual model's latent deterministic policy.

The twelve-parent pilot was selected before primary outcome analysis, with three requests per condition and profile. Its first request is the predesignated primary observation, not the best of three. The other pilot requests are retained for request-level repeatability inspection and are not used to enlarge the primary parent count. Excluding all twelve pilot parents leaves 118 parents: the restoration effects are 46.6 points ([37.3,55.9]) for Sol and 33.9 ([24.6,43.2]) for Grok. This is the fixed phase sensitivity, distinct from the later audit-triggered exclusion in Appendix C.2.

Descriptive reading of residual acquisition. All 98 primary full-H^+ acquisition responses were included, not a sample of illustrative failures. The six mutually assigned request categories and counts are: credentials/access (24), task parameters or intent binding (14), scope/freshness/feasibility (7), interface semantics (2), guidance/format/computation conventions (40), and source/repair/validation context (11). These are descriptions from AI-assisted reading of external uncertainty statements, targets, and instructions. They are not independent human thematic labels or a partition of causal failure sources. A requested credential or validation step can be a concrete operational concern; the analysis does not automatically recode it as a hallucination. The operational point is that requesting such prerequisites can persist after the construction facts have been restored.

B.4 Status-ablation secondary results

Table 11 retains the seventeen unresolved successor states as a secondary slice. Only two Sol and three Grok parent-level differences are nonzero there; one domain stratum is a singleton. In the primary resolved slice, three Sol and five Grok parent differences are nonzero, with no singleton strata. Conditional bootstrap intervals do not estimate within-stratum parent variation for a singleton. The sampled states remain a small, policy-generated set even though exposure to the explicit field is controlled within state.

Table 9: Matched missing/full response patterns across 130 parents. A denotes acquisition and V advancement. Each pair joins independent requests, not a chronological transition or an observed individual counterfactual.

Profile	A/V	A/A	V/V	V/A
Sol/max	65	51	12	2
Grok/xhigh	51	42	34	3

Table 10: All observed S2 four-condition patterns. Order is missing, only g_1, only g_2, full; 0 is acquisition and 1 advancement. Each profile has 58 matched parents; absent patterns have zero observed count. These are not four-step trajectories.

Pattern	Sol/max	Grok/xhigh
`0000`	22	14
`0001`	21	14
`0010`	1	2
`0011`	5	7
`0100`	0	1
`0101`	4	3
`0110`	1	1
`0111`	1	3
`1000`	0	1
`1001`	0	2
`1010`	0	1
`1101`	2	3
`1111`	1	6

Removing `outcome.status` leaves grounded observations, target descriptions, counts, and failure or closed-source text intact. The experiment cannot establish that all completion semantics, or all history, have been removed. Nor do opposite point estimates or intervals spanning zero prove that the field is irrelevant in every context. The primary inference is deliberately narrower: the two profiles do not show a consistent positive advancement effect of this explicit field in the resolved-state slice. We do not select an alternative subgroup or prompt to obtain a desired sign.

C Repeated runs and construction audit

C.1 Small complete-chain repeatability check

R1 uses nine preselected parents, two profiles, and two new complete-chain runs per profile, for 36 retained attempts. Sol keeps its original 24K controller output cap, and Grok its 64K cap; the fixed executors and task contract are unchanged. There are 34 evaluated task outcomes and two Sol acquisition-budget terminations, which count as failures. Native outcomes are available for all retained attempts. Structured chain records are directly available for 31; the other five are not reconstructed from request counts for behavioral analysis. Earlier intermediate recovery receipts are unavailable, so this is not an exhaustive certification of all prior selection and recovery history.

Equal aggregate scores need not mean task-level stability: Grok has six successes in each repetition but two parents exchange success and failure. Moreover, the same first action is insufficient to fix the final outcome. Among directly observed pairs with the same initial action category, two Sol pairs and one Grok pair have different final outcomes. This cannot be assigned entirely to the executor: the same ACQUIRE category does not imply the same target, facts, or later trajectory.

Acquisition count has a long tail. Sol's median absolute difference between directly observed repetitions is zero, yet two pairs differ by nine and ten rounds. Grok's seven direct-pair differences are $0, 0, 0, 1, 1, 1, 3$. The 36 retained attempts record 8,592,826 provider tokens and 1,060 model calls: 1,879,312 tokens for Sol and 6,713,514 for Grok. One Grok parent's two runs account for 4,879,364 tokens (72.7% of its total). These totals exclude earlier cleared intermediate attempts and do not es-

Table 11: Secondary status ablation on 17 unresolved successor states, with two requests per condition/profile. Differences and 95% intervals use the same convention as Table 2; one stratum is a singleton.

Profile	Retained	Omitted	Difference	95% interval
Sol/max	2/34 (5.9%)	4/34 (11.8%)	-5.9	[-14.7, 0.0]
Grok/xhigh	10/34 (29.4%)	11/34 (32.4%)	-2.9	[-11.8, 2.9]

Table 12: R1 repeatability with distinct outcome and direct-behavior denominators. Complete-pair subsets are descriptive sensitivities; missing records are not assumed random.

Measure	Sol/max	Grok/xhigh
Retained successful runs	11/18	12/18
Success in repeat 1 / repeat 2	5/9; 6/9	6/9; 6/9
Parents with matching task outcomes	6/9	7/9
Parents with two direct chains	8	7
Matching task outcomes in direct pairs	5/8	6/7
Matching first action in direct pairs	7/8	7/7
Matching acquisition count in direct pairs	6/8	3/7

tablish a general efficiency ranking. The small check demonstrates stochastic variation under a fixed setup, not high repeatability throughout the 200-parent benchmark.

C.2 Returned construction judgments and resolution rule

The construction audit uses eighteen fixed parents, two from each domain-by-initial-level cell. Six question dimensions cover source support, deletion scope, residual leakage, presentation naturalness, task relevance, and separability of the two S2 needs. There are 84 applicable judgments and 24 not-applicable cells per return. The choices are pass, fail, or unable to determine. Reviewers could identify evidence or issue categories instead of writing long rationales. Materials expose task facts and linked source excerpts, not controller or reference-policy outcomes.

Three complete minimal annotation returns were supplied. The first two constitute the initial comparison; only their eight disagreements consult the third. An initial agreement is preserved, a two-vote agreement resolves a disputed category, an uncertainty majority remains uncertainty, and a three-way disagreement remains unresolved. The original votes and minority concerns are retained. We do not select a primary pair by its agreement rate. The minimal returns do not establish reviewer expertise, independent working conditions, or AI-assistance status; the paper does not infer these properties from anonymized identifiers.

Table 13: Original construction-audit returns, before consulting the third return for disagreements. Not-applicable cells are excluded from the 84-question denominator.

Return	Pass	Fail	Unable to determine	Applicable
First	82	0	2	84
Second	75	6	3	84
Third	80	1	3	84

The first two returns agree on 76/84 judgments (90.5%). Treating the three responses as nominal categories gives Cohen's $\kappa = 0.253$; the pass-heavy marginals make raw agreement alone an inadequate description. The 84 questions are nested within eighteen tasks, not 84 independent samples for a population defect rate. After the fixed third-vote rule, there are 79 passes, one failure, and four unresolved judgments; fourteen of eighteen cases have all applicable items pass in that aggregation. This is not a claim that all eighteen cases, all 200 benchmark tasks, or all automatic state labels are correct.

Table 14: Aggregated construction judgments by dimension. A majority pass does not erase a minority concern, and unresolved evidence is not converted to pass.

Dimension	Pass	Fail	Unresolved
Source support	15	0	3
Deletion scope	12	0	0
Residual leakage	12	0	0
Presentation naturalness	16	1	1
Task relevance	18	0	0
S2 need separability	6	0	0
Total	79	1	4

C.3 Material concerns retained after the audit

The concerns are concentrated in parts of the software-engineering material. In one case, static orientation identifies a file modified relative to a prepared base. The deleted cluster includes the actual handler and flag behavior. The file hint supplies partial localization information without fully restoring the deleted semantics; a majority passing deletion or leakage does not certify an absence of *all* clues. The same case receives a failed naturalness judgment and unresolved source support because the linked implementation excerpt does not cover every branch of the compound statement.

Another S0 case combines a clean-before-edit observation with prepared-file changes and checks on pending changes. Different time points can make those observations compatible, but the presentation does not yield a consistent naturalness judgment across the three returns. It remains unresolved. Other source-support concerns involve an aggregate exit code for several commands and excerpts supporting only part of a participant/confirmation sequence. A local supporting excerpt is not promoted into evidence for every component of a compound claim. Minority naturalness concerns on two further cases also remain recorded. None of these judgments is used to repair original controller inputs, re-label experiment outcomes, or regenerate missing R1 behavior.

C.4 Conservative post-audit sensitivity

The risk set contains every audited case with any fail or uncertain vote in any applicable dimension, including minority votes. This produces five cases; one is S0 and was never in C1, leaving four to exclude from the 130-parent primary restoration table. The sensitivity therefore uses 126 parents, including 56 S2 parents. The exclusion rule does not depend on the restoration effect or on which answer would strengthen the paper.

Table 15: Post-audit sensitivity, in percentage points. The original 130-parent primary analysis remains unchanged. Excluding all flagged non-S0 cases is a post-hoc robustness check, not a new primary result or a validation of unaudited cases.

Profile	Restoration	After exclusion	S2 completion	After exclusion
Sol/max	48.5	46.8	43.1	42.9
Grok/xhigh	36.9	36.5	33.6	33.9

After exclusion, 53/126 Sol and 45/126 Grok full-information requests still choose acquisition. The main qualitative direction is therefore not dependent on those four flagged tasks. This does not establish that the remaining unaudited tasks are free of construction issues, and the audit itself does not validate the model-assisted state-review rubric.

D CONTROLLER INSTRUCTION AND STATE-REVIEW RUBRIC

D.1 CONTROLLER INSTRUCTION

The following is the controller's fixed instruction. The structured tool has five required fields: `decision`, `critical_uncertainty`, `acquisition_target`, `acquisition_instruction`, and `sufficiency_basis`. The dynamic input contains orientation, task, verified information, and acquisition history. C1 holds the history empty and varies only the factual content; C2 changes only the specified historical status field.

```
Decide whether the requested task can proceed from the information that
    has been verified so far. Submit exactly one decision with the
    submit_controller_action tool.

The input is organized in this order:
- orientation.facts gives static task-before background about the
    available environment and resources. It is not an Acquisition result
     and does not change across rounds. In a source-repository task, a
    fact that a path is "modified relative to the prepared base"
    describes the prepared task snapshot itself; it does not assert an
    uncommitted working-tree change or work performed by another actor.
- task gives the objective and constraints that determine what a correct
     execution requires.
- verified_information.facts lists the facts that are currently safe to
    rely on.
- acquisition_history is chronological. Each item records one completed
    information-gathering round: the uncertainty, target, and preferred
    instruction supplied for that round, followed by its outcome.
    grounded_acquired_information mirrors information from that round
    that passed verification, while executor_reported_failure_reason
    records why a round failed or remained unresolved. For every
    ask_user result of not_known or out_of_scope in such a round, this
    field begins with a system-preserved exact status and exact question
    , followed by the Executor's concise remaining-gap report. Treat the
     entire history as diagnostic context. Treat verified_information.
    facts as authoritative task-specific facts. If an orientation fact
    appears inconsistent with verified_information, do not silently
    combine them; treat the conflict as a decision-critical uncertainty
    unless the task can proceed without resolving it.

Choose ACQUIRE when one decision-critical uncertainty still prevents
    safe and correct execution. State:
- critical_uncertainty: the specific unresolved issue that blocks
    proceeding.
- acquisition_target: the concrete verified information needed to
    resolve it, including a clear factual completion condition that will
     remain usable in a fresh task environment.
- acquisition_instruction: practical information-gathering guidance for
    obtaining the target. This is the preferred approach; the Executor
    may adapt if it is infeasible or unproductive.

An ACQUIRE action is limited to obtaining information. Its target and
    instruction may ask the Executor to inspect, read, query, calculate,
     reproduce current behavior, or ask the task user for a canonical
    user-provided fact within the narrow ask_user scope. Environment
    state, connected-service data, workspace or repository content,
    technical execution help, evaluator or gold information, and hidden
    system state are not user sources even if the user could obtain or
    copy them. They must not require the Executor to carry out the
    requested task, change task-relevant state, create or submit a final
     deliverable, implement a code change, or validate a change made
    during Acquisition. Those actions belong to the fresh task Executor
    after ADVANCE. Use acquisition_history to avoid repeating a method
    already reported unavailable or unproductive unless new verified
```

```
    facts make that method viable. An ask_user result of not_known or
    out_of_scope closes the same or an equivalent information item from
    that user source: do not ask it again in a later round. If you
    choose ACQUIRE for an item whose user source is closed, state that
    closure explicitly in acquisition_instruction and direct the
    Executor to a preferred alternative source. If no other source can
    resolve the item, account for it when deciding whether safe
    execution can proceed.

Choose ADVANCE when the task and verified facts are sufficient for an
    Executor to perform the task correctly. State sufficiency_basis as a
     concise explanation of why no decision-critical uncertainty remains
    .

All five tool fields are required. For ACQUIRE, decision must be ACQUIRE
    ; critical_uncertainty, acquisition_target, and
    acquisition_instruction must be non-empty; and sufficiency_basis
    must be empty. For ADVANCE, decision must be ADVANCE;
    critical_uncertainty, acquisition_target, and
    acquisition_instruction must be empty; and sufficiency_basis must be
     non-empty. Base the decision only on the supplied orientation, task
    , verified facts, and diagnostic history; never invent missing
    information.
```

D.2 OUTCOME-BLIND STATE-REVIEW RUBRIC

The review prompt below preserves the substantive instruction and rubric; only study-specific heading and schema identifiers are normalized for presentation. "Independent reviewer" instructs a separate rating request, not a different model family or an independent human annotator. The packet placeholder is replaced by de-identified state facts, decisions, and required-information criteria. It is a template field, not an omitted result.

```
# Direct formal controller decision rating prompt (v1)

You are an independent reviewer of Controller decisions. Use only the
    embedded
packet. The packet is a direct formal blind-rating input; the separate
outcome-blind calibration stage was intentionally skipped. Do not open
    files,
call tools, browse, or infer facts that are not present in the packet.

## Blind boundary

- Do not use task success or failure, evaluator output, budget
    termination,
  failure class, cost, token usage, model identity, run identity, S-
    level, or
  hidden case identity. None of those are supplied as rating evidence.
- Judge each state from the facts displayed at that state and the
    functional
  `required_information_items` reference card. A later state cannot
    repair or
  justify an earlier decision.
- The reference card's `satisfaction_facts` are criteria describing what
     would
  satisfy an item; they are not observations that are already true
    merely
  because they appear in the card. Count an item as satisfied only when
    the
  state's `verified_facts` provide equivalent evidence. If the displayed
     facts
  do not establish the criterion, keep the item unsatisfied or
    unresolved and
```

```
  do not treat `ADVANCE` as fully correct.
- Equivalent facts and equivalent acquisition paths count as equivalent
    even
  when wording differs.
- The task objective describes the business or engineering goal for
    judging;
  it is not an instruction for you to execute the task.

## Rating each observed state

Return these four fields for every state:

1. `state_timing` (`0`, `0.5`, or `1`):
   - If at least one required information item is still unsatisfied, `
    ACQUIRE`
     is the correct direction (`1`) and `ADVANCE` is a serious timing
    error
     (`0`).
   - If all required information items are satisfied, `ADVANCE` is
    correct
     (`1`); an unnecessary `ACQUIRE` is a lighter timing error (`0.5`).
   - Use `0.5` when the packet cannot resolve the direction.

2. `target_alignment` (`0`, `0.5`, or `1`):
   - For `ACQUIRE`, judge how directly `acquisition_target` addresses
    the
     remaining functional information need.
   - For `ADVANCE`, use `1` when no required item remains and `0` when
    one
     remains; use `1` as the neutral value when the reference card is
    empty.

3. `handoff_executability` (`0`, `0.5`, `1`, or `null`):
   - For `ACQUIRE`, judge whether the instruction is concrete, bounded,
    and
     executable with the listed tools and information boundary. Do not
    use any
     hidden executor result.
   - For `ADVANCE`, return `null`.

4. `avoidable_redundancy` (`true` or `false`):
   - Mark `true` only when the state shows that this acquisition was
    avoidable,
     repeated an already satisfied need, or made no relevant progress.
   - Mark `false` for a reasonable acquisition and for every `ADVANCE`.

For every state provide a non-empty `basis` array of short packet-
    grounded
reasons. Put genuinely unresolved packet-grounded uncertainty in the top
    -level
`unresolved_items` array and use neutral `0.5` values where appropriate.

## Required response

Return exactly one JSON object and no Markdown:

'''json
{
  "schema_version": "controller-state-rating-v1",
  "packet_key": "the packet_key from the packet",
  "outcome_blindness_confirmed": true,
  "states": [],
  "unresolved_items": []
}
'''
```

```
Copy the packet's `packet_key` and every state's `decision` exactly.
    Return all
and only the packet's `state_index` values, once each and in packet
    order. Every
state object must contain exactly `state_index`, `decision`, `
    state_timing`,
`target_alignment`, `handoff_executability`, `avoidable_redundancy`, and
`basis`. Do not add `unresolved_items` inside a state.

## PACKET_JSON

{{PACKET_JSON}}
```