LATENT WORKFLOW DISCOVERY FOR ENTERPRISE WORLD MODELS: GROUNDING GENERAL-PURPOSE AGENTS IN ORGANIZATIONAL CONTEXT

Anonymous authors
Paper under double-blind review

ABSTRACT

Enterprise agents must connect user goals to organization-specific records, tool dependencies, and workflow prerequisites. We formulate experience reuse as latent workflow discovery and propose Graph World Models (GWMs), combining a workflow graph, language model, and software harness. GWM builds a world model for a target enterprise from interaction histories, storing inferred workflow states, observed successors, outcomes, support counts, and examples. The harness retrieves evidence for model calls that advise the policy or select its candidates. We evaluate within-benchmark, cross-benchmark, and cross-model reuse on customer-relationship-management tasks in CRMArena-Pro and enterprise-operation tasks in EnterpriseOps-Gym. By combining graph-structured workflow memory, advice, and candidate selection, GWM improves CRM task success over direct generation for both Qwen and Gemma without policy retraining. Qwen-derived experience supports same-model and cross-model guidance, and the large-graph configuration further improves Gemma success over the small-graph configuration. Controlled comparisons characterize the incremental effects of historical evidence and candidate selection. Our benchmark-agnostic analysis separates abstraction loss, candidate availability, and selection error; in historical CRM pools, missing successful candidates account for more failures than misranking available ones. We release GWM as a vLLM extension with loadable workflow-graph adapters, supporting graph construction and online/offline inference at https://github.com/iclr-gwm/gwm.

1 INTRODUCTION

Enterprise agents must ground each action in organizational context: records, tool behavior, prerequisites, dependencies, and policies. Generating and sharing a report, for example, requires different next steps when clearance is pending or granted (Figure 1). Interaction logs record such distinctions, but a successful trajectory alone does not establish when its actions are appropriate. We study how to reuse this evidence without updating the acting policy.

Current approaches adapt the policy through supervised fine-tuning (Zeng et al., 2024) or reuse experience externally: Agent Workflow Memory supplies routines, and AutoGuide retrieves context-conditioned guidelines (Wang et al., 2025; Fu et al., 2024). These memories already support contextual reuse, but organize experience primarily as procedures or advice. We investigate how to expose alternative continuations together with their observed outcomes and supporting evidence at a recurring decision context. This calls for linking the context in which an action occurred to what followed.

We introduce Graph World Models (GWMs), combining graph-structured workflow memory, a language model, and a software harness to build a world model for a target enterprise. *Latent workflow discovery* infers summaries of interaction-history prefixes—*latent workflow states*—and their observed continuations; these states form the *workflow latent space*. State matching retrieves relevant experience; graph links expose observed alternatives; outcome and support statistics summarize

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.

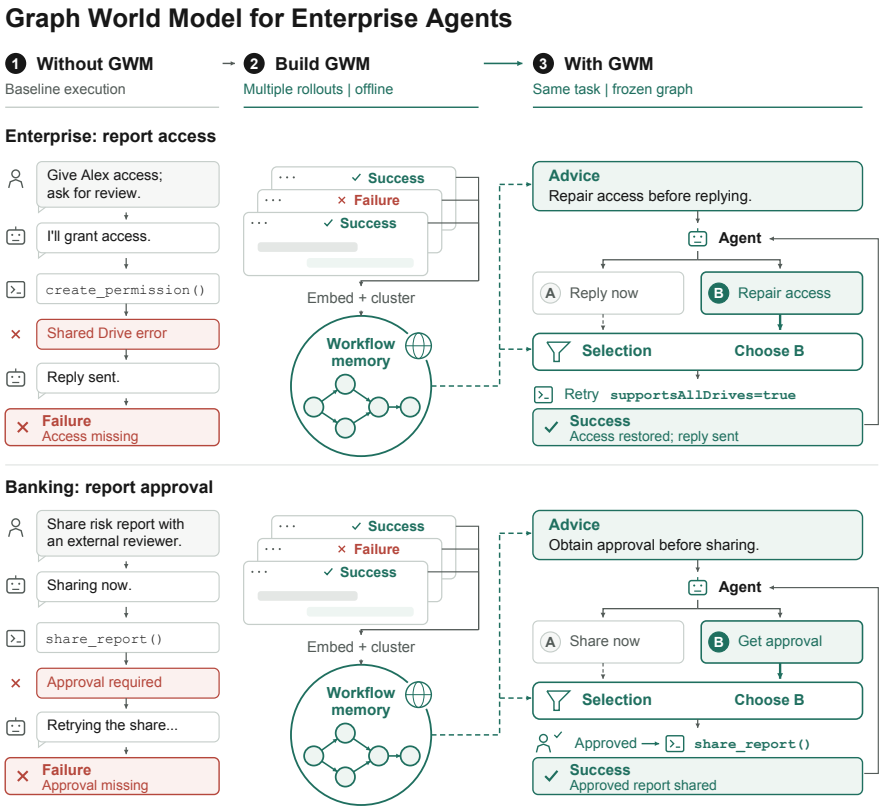


Figure 1: Graph World Model construction and use. Successful and failed offline rollouts form domain-specific graphs: nodes cluster interaction-history prefixes; edges record observed successors. Advice guides candidate generation; selection chooses before tool execution. Observations update live history while graph and policy weights remain fixed. The enterprise baseline follows a recorded access failure; the guided continuations and banking approval example are schematic.

their evidence; and examples supply concrete context. The harness uses this evidence to advise the acting policy or rank its candidates, keeping policy weights fixed. GWM denotes the combined guidance method: graph-structured workflow memory, retrieval, model-generated advice, and candidate selection. Advice and selection are interfaces that can operate independently or jointly.

The key question is *when an enterprise world model inferred from interaction histories becomes useful for guiding action choices without updating the acting policy's weights*. Our analyses separate retained contextual distinctions, candidate availability, and selection error.

To simulate a real-world enterprise environment, we evaluate tool-using agents in CRMArena-Pro (Huang et al., 2026a), which covers customer service, sales, and product configuration, and EnterpriseOps-Gym (Malay et al., 2026), which requires persistent-state operations across eight domains: Calendar, customer service management (CSM), Drive, Email, Human Resources (HR), IT service management (ITSM), Teams, and Hybrid workflows. These tasks require agents to query records, resolve dependencies, and coordinate tool calls across enterprise services.

The paper makes three contributions:

1. **A formulation of workflow reuse.** We formalize workflow states and observed transitions in enterprise environments and distinguish information lost through abstraction, missing useful candidates, and errors in candidate selection. Conditional bounds characterize abstraction error and selection loss under explicit assumptions (Appendices A and B.1).
2. **A discovery and guidance mechanism.** We build an enterprise world model from interaction histories, aggregating workflow states, observed transitions, and outcome evidence

into an external graph. The guidance mechanism combines graph retrieval, advice, and candidate selection with fixed acting-policy weights, supporting same-model and cross-model reuse (Section 4).

3. **An empirical characterization of when guidance helps.** We evaluate within-benchmark, cross-benchmark, and cross-model reuse, and show that discovered workflow states retain outcome information. GWM configurations achieve higher task success than direct generation under same-model, cross-model, and cross-benchmark reuse; in the evaluated cross-model comparison, the large-graph configuration further improves success over the small-graph configuration. Online experiments, fixed-candidate controls, and resource measurements characterize configuration-level gains, mixed incremental evidence effects, and guidance costs (Table 2; Sections 5–7).

2 Related work

Contextual experience reuse. Agent Workflow Memory induces reusable routines; AutoGuide retrieves context-conditioned guidelines from offline experience (Wang et al., 2025; Fu et al., 2024). Both support contextual experience reuse. GWM organizes history prefixes into workflow states with observed successors, outcomes, support counts, and examples, differing in how experience is represented and supplied.

Distinction from Graph World Model. Feng et al. (2025) represent the current world as a graph of multimodal data and task/action nodes, using message passing for prediction, generation and planning. Their token variant instruction-tunes decoders; their embedding variant freezes the LLM and trains a projector. GWM instead builds an enterprise world model by discovering workflow states, observed transitions and outcome associations in interaction histories, then retrieves continuation evidence for advice or candidate selection with unchanged policy weights. This contrasts trained graph-conditioned decoding with workflow-evidence retrieval and replay.

Evidence-format controls. We evaluate adapted evidence formats through fixed-candidate replay, comparing graph evidence, induced textual workflows and retrieved examples with identical candidates and the same selection model within each evaluation (Table 20). These adaptations (Wang et al., 2025; Zheng et al., 2024) test evidence formats without reproducing competing systems' complete training and retrieval pipelines. Inference budgets and tool access also shape comparisons (Bragg et al., 2026). We include an extended comparison in Appendix D.4.

3 Workflow states and organizational grounding

We define an enterprise setting as an agent interacting with organizational tools and records to pursue a task goal. We describe this setting using the language of a partially observable Markov decision process (POMDP) (Kaelbling et al., 1998). An enterprise environment e has hidden operational state x_t, observations o_t comprising user messages and tool responses, and executable actions a_t with arguments. At each decision, an agent has history $h_t = (o_0, a_0, o_1, \ldots, a_{t-1}, o_t)$ and goal g. Workflow discovery summarizes the organizational context available in these observations, including encountered records, observed prerequisites, and prior continuations.

Let $\mathcal{D}_{\text{disc}}$ denote the offline discovery dataset, where the subscript disc stands for discovery. It contains trajectories from a source policy β, which may differ from the target policy π_θ. The builder converts these trajectories into a reusable graph, $G = \mathcal{B}(\mathcal{D}_{\text{disc}})$. GWM uses this graph as an enterprise world model that guides decisions while keeping the acting policy's weights fixed, $\pi_G = \mathcal{H}(\pi_\theta, G)$. We measure a policy's performance by its expected terminal task score over goals and executions in the environment:

$$J_e(\pi) = \mathbb{E}_{g\sim\mu_e,\tau\sim(e,\pi,g)} Y_e(\tau, g). \tag{1}$$

Here μ_e is the goal distribution and $Y_e(\tau, g) \in [0, 1]$ is the terminal score of trajectory τ on goal g. The improvement over comparator policy π_b is $\Delta_{e,b} = J_e(\pi_G) - J_e(\pi_b)$. This contrast depends on the comparator, acting policy, graph, task distribution, and resource budget. Comparisons between independently configured runs can also reflect differences in execution settings.

The discovered state $z_t = \phi_{\mathcal{D}_{\text{disc}}}(h_t; e)$ maps an observed interaction prefix to a workflow node. We use "latent" to denote a representation inferred from traces, distinct from both hidden enterprise

state x_t and target-model activations. The encoder uses recent rendered history, including argument text subject to preprocessing and truncation. The action label $u = \psi(a)$ is a set of tool names, represented in sorted order, discarding arguments, call order, and multiplicity. GWM selection uses richer history, goal, and candidate arguments, subject to rendering limits (Appendix E.6). We represent GWM's workflow graph as

$$G = (\mathcal{Z}, \mathcal{E}, \phi, \psi, \widehat{P}_{\text{obs}}, \widehat{m}, \mathcal{U}, \mathcal{L}), \tag{2}$$

where $\mathcal{Z}$ contains workflow states and the `START`/`END` boundary symbols, $\mathcal{E}$ contains successor relations, $\widehat{P}_{\text{obs}}$ observed successor frequencies, $\widehat{m}$ outcome associations, $\mathcal{U}$ support counts, and $\mathcal{L}$ examples and provenance. We use enterprise world model to denote this external model of observed workflow contexts, continuations, and outcome associations. The evaluated guidance mechanism retrieves this evidence for advice and selection. Its frequencies describe the discovery distribution. We analyze abstraction error in Appendix A and candidate coverage and selection losses in Appendix B.1.

For an authoritative rule source $\mathcal{R}_e$, let $V_t \in \{0, 1\}$ indicate a violation on step t. An agent can achieve its task goal while violating a workflow prerequisite. However, enterprise verification monitors can report workflow violations as errors or warnings. Assuming these signals are recorded alongside task outcomes, labeled trajectories can expose workflow prerequisites and violations as well as task success.

4 Discovering and Using Organizational Workflow Evidence

First, we construct a workflow graph by aggregating trace-derived contexts, continuations, outcomes, support, and examples. At inference time, the graph remains frozen as the history changes. A self-evolving extension could periodically update the graph from new experience, drawing on iterative graph-memory maintenance (Anokhin et al., 2025); we do not study such updates in this paper. Online experiments evaluate complete guidance configurations; fixed-candidate replay isolates the evidence block supplied during candidate scoring.

4.1 Discovering Workflow States and Graph Relations

We convert each multi-turn conversation in a discovery rollout r into interaction histories $h_{r,i}$ and encode them as $E_\xi(h_{r,i}) \in \mathbb{R}^p$, where E_ξ is a text encoder and p is its embedding dimension. A clustering map κ_d, fitted within domain d or shared across domains, groups these representations into workflow states, giving $z_{r,i} = \kappa_d(E_\xi(h_{r,i})) = \phi(h_{r,i})$. Each $z_{r,i}$ is a state occurrence in a specific trajectory; occurrences assigned to the same cluster contribute evidence to the same workflow node. This formulation permits different text encoders, dimensions, and clustering methods. More broadly, the approach is applicable in principle to any application with multi-turn conversations that expose latent workflow structure. Appendix B.2 specifies the evaluated implementation. The graph aggregates transitions and rollout outcomes across state occurrences assigned to these nodes.

Within each domain, augment each rollout's ordered state occurrences with `START` and `END`. Let $N(z, u, z')$ count transitions in these augmented sequences, where u is the toolset already observed at the source occurrence; `START` uses an empty boundary label, with $N(z, u) = \sum_{z'} N(z, u, z')$ and $N(z, z') = \sum_u N(z, u, z')$. Normalizing these counts gives the marginal and toolset-conditioned successor frequencies:

$$\widehat{P}_{\text{obs}}(z' \mid z) = \frac{N(z, z')}{\sum_v N(z, v)}, \qquad \widehat{P}_{\text{obs}}(z' \mid z, u) = \frac{N(z, u, z')}{N(z, u)}. \tag{3}$$

Successor sums include `END`. The toolset-conditioned table is stored only for workflow-state sources; `END` has no outgoing transitions. These unsmoothed estimates are defined only for observed pairs with positive counts; no estimate is stored for an unobserved pair. For any nonempty count vector, normalized frequencies minimize categorical log loss (Appendix B.1). We also show the decision-time fallback procedure in Appendix B.

Action timing and predictive target. In the recorded construction, state occurrence $z_{r,i}$ represents rollout r after action i and its results have been observed. The enterprise world-model transition to $z_{r,i+1}$ is labeled with the already observed toolset $u_{r,i} = \psi(a_{r,i}) \subseteq \mathcal{T}$, where $\mathcal{T}$ is the set of

tool names. For example, a rollout can contain the toolset sequence $(\{t_1\}, \{t_1, t_2\}, \{t_2\})$. Each set is stored as `tools[i]`; pooling these occurrences across trajectories yields historical successor evidence for the workflow states. In the next-message heuristic, candidate toolsets index this table to retrieve observed associations. We define prospective prediction as estimating the next-state distribution $K_e^Z(\cdot \mid h_t, a_t, g)$ from a history h_t ending before a_t. The aligned construction labels each stored transition with $u_{r,i+1}$ (stored as `tools[i+1]`), the toolset of the action leading from its source occurrence to its successor. This relabeling changes the action associated with an edge while preserving the state assignments, state-level outcome statistics, and examples used by the whole-episode selector. We report the transition-label analysis in Appendix E.4 and describe the aligned-construction evaluation in Appendix F.5. Cluster associations appear in Appendix E.5.

Outcome features record state and toolset success/failure associations and supporting rollout counts. The graph also stores transition examples from successful rollouts and tool-error examples. Failure lift compares a state's failure rate with its domain rate; it is descriptive and may reflect task difficulty or discovery-policy composition (Appendix B).

4.2 Retrieval, advice, and candidate selection

At inference time, the encoder maps the current history into the representation space, and a retrieval rule matches it to a workflow state. Retrieval returns the matched state's outcome and support statistics, local outgoing successor and toolset associations, and available trajectory examples. Matches outside a configured acceptance threshold are rejected, although nearby states can still supply low-confidence context. Such nearby-centroid retrieval is particularly relevant when using an out-of-domain graph. Retrieval can search within a domain or across domains; in the latter case, it uses statistics from the selected state's domain. GWM also uses the current history and goal. In Appendix B.2, we define the matching rule and threshold used in the reference implementation.

For a generated advice message m, the conditional policy is:

$$\pi_G^{\mathrm{adv}}(a \mid h, g, m) = \pi_\theta(a \mid h \oplus m, g). \tag{4}$$

The operator $\oplus$ adds m to the history; the marginal policy averages over advice generation. Abstention uses $m = \varnothing$, with $h \oplus \varnothing = h$. The acting policy retains control over its next action, so advice effectiveness depends on the template, injection cadence, retrieval, and the policy's interpretation of the supplied context. For next-step selection, the acting policy samples K candidate assistant messages $C_K(h) = (c_1, \ldots, c_K)$, offering alternative continuations of the same history, including any GWM advice injected when the advice interface is enabled. The GWM harness uses a language-model call to score these candidates from the current history, goal, and graph evidence. Its score function $s_G(h, c_i; C_K)$ ranks candidate c_i, with the goal held fixed. Before execution, the selector identifies a highest-scoring candidate by index and returns that candidate's message:

$$i^\star \in \arg\max_{1 \le i \le K} s_G(h, c_i; C_K), \qquad \widehat{c} = c_{i^\star}. \tag{5}$$

A next-step candidate contains tool calls with arguments or a final answer; any proposed tool outcomes are not yet observed. Some environments also support selection among executed rollouts generated by the acting policy. The selector scores these rollouts individually (separately) or jointly (together in one model context) from their final answers and trajectory tails. Thus K counts next-step proposals or executed trajectories depending on the interface. Whole-episode selection has access to executed continuations, and its candidate value is terminal return; next-step value depends on subsequent execution.

If the selector returns unusable scores, the next-step interface falls back to supported toolset outcomes or state/domain success rates (Appendix B.2). Note that appending candidate text does not reveal its execution outcome.

Advice and selection can operate separately or together; advice plus selection denotes the combined interface, whose performance reflects both interventions. Keeping policy weights fixed leaves candidate generation, selector calls, embeddings, retrieval, and added context as intervention costs.

Note that guidance may fail because the policy omits useful candidates or the selector ranks available candidates poorly. Under a uniform score-error bound and a fixed continuation policy, the conditional analysis separates these losses (Appendices B.1 and E.3).

Experiment	Acting policy	Graph source	Tasks	Configuration
CRM guidance comparison	Qwen	Qwen	100 CRM	Baseline, advice, and selection among executed rollouts.
CRM candidate analysis	Qwen	Qwen	Same 100 CRM	Separate $K = 10, 12$ selection runs retain answers and scores; shared guidance-comparison baseline.
EOPS selection	Qwen	Qwen	80 EOPS	Select the next assistant message before execution ($K = 2, 4, 8$), compared with direct generation.
Discovery and transfer	Qwen	Qwen	100 CRM; 80 EOPS	Vary discovery fraction within a benchmark (10/20/50/100%; $K = 4, 8, 10$), or reuse the other benchmark's graph (20/100%; $K = 4, 8$).
Source-policy comparison	Gemma	Qwen or Gemma	100 CRM; 80 EOPS	Historical cross-policy reuse with graph fallbacks and no delivered advice.
CRM GWM guidance	Qwen or Gemma	Qwen	100 CRM	Gemma GWM-S/L at $K = 8$; Qwen GWM-M at $K = 10$; CRM action-label comparison.
Selection controls	Qwen or Gemma	Qwen or none	100 CRM; 80 EOPS	Random and majority selection on each CRM GWM pool; separate graph-free CRM and EOPS policy controls.
Fixed-candidate replay	Qwen candidates	Qwen	1,479 CRM; subset 600	Qwen or Gemma scores stored Qwen candidates: no evidence, examples, workflows, and graph variants.

Table 1: Experimental configurations. Counts denote unique tasks across repeats. The two CRM guidance/candidate studies reuse the same tasks and baseline executions. Replay fixes candidates for each model. Gemma settings separate fallback and active guidance.

5 Experimental Setting

We address three questions: **Q1**, when guidance improves task outcomes; **Q2**, how utility varies with discovery data and policy; and **Q3**, what graph evidence adds with candidates and model held fixed. CRMArena-Pro (Huang et al., 2026a) covers customer-relationship-management (CRM) workflows over business records; EnterpriseOps-Gym (EOPS) (Malay et al., 2026) covers eight enterprise domains with outcome and integrity checks. Both benchmarks provide task-verification signals. EOPS requires all reported SQL verifiers to pass, while CRM uses its task-specific evaluator and recorded success criterion. We therefore expect enterprise violations reported in tool outputs to be reflected in the constructed GWM. Both require multi-step tool interactions from which we discover recurring workflow states and continuations. Online evaluation reuses 100 CRM tasks and 80 EOPS tasks, randomly sampling 10 EOPS tasks per domain to cover all eight domains.

Models used by GWM. We use Qwen3.6-27B (Qwen; Qwen Team, 2026) and Gemma-4-26B-A4B-it (Gemma; Gemma Team, 2026) as acting policies. We define two evaluation settings: online evaluation, where the GWM harness uses the acting policy model for advice and candidate scoring with unchanged weights; and fixed-candidate replay, where Qwen or Gemma scores stored Qwen-generated candidates. Qwen scores all 1,479 eligible CRM pools after excluding discovery tasks; Gemma's evaluation is capped at 600 tasks, selected by shuffling the eligible task IDs with seed 17 and retaining the first 600. GWM can use any compatible model. We define discovery as a separate offline stage that builds reusable workflow memory from source-policy trajectories. Both acting models reuse Qwen-generated memory: the source policy β and acting policy π_θ use the same pretrained model in Qwen-to-Qwen reuse and different models in Qwen-to-Gemma reuse; θ denotes the fixed acting-model weights. The evaluated encoder, clustering, retrieval, and fallback settings are specified in Appendix B.2.

Guidance and information access. The baseline uses direct generation without graph evidence. Advice adds retrieved statistics and optional examples to the policy context; candidate selection scores alternatives using graph evidence, live history and the task. Online comparisons include candidate generation, candidate selection, and advice. In our setting, EOPS candidates are next assistant messages containing unexecuted tool calls or final answers; CRM candidates are executed rollouts scored individually or jointly from final answers and trajectory tails. We define fixed-candidate replay as holding candidates and model constant while varying evidence: none, retrieved prefixes, induced workflows, graph statistics, or their combination. We report candidate count K for each configuration. We use $K > 1$ to offer alternatives for selection when avoiding enterprise errors or policy violations (Malay et al., 2026) may justify additional inference.

Selection controls. To test whether GWM selection improves on chance or answer agreement under the same $K > 1$ candidate-generation budget, we compare GWM with uniform-random and majority-vote selection on the same CRM pools of executed rollouts, both omitting GWM at selection time. Uniform random selection assigns equal probability to every candidate and reports expected success. For these stored-pool controls, majority vote selects the most frequent normalized final answer, resolving ties by candidate order. The baseline uses direct generation without

Acting policy / task	Configuration	baseline (%)	Uniform random selection (%)	Majority vote selection (%)	GWM / control Success (%)	Δ pp
(a) Online configuration						
Gemma / CRM	GWM-L, $K = 8$	40.00 $\pm$5.00	40.50 $\pm$1.27	45.00 $\pm$1.00	52.33 $\pm$1.53	+12.33 $\pm$5.69
Gemma / CRM	GWM-S, $K = 8$	40.00 $\pm$5.00	39.75 $\pm$1.27	44.33 $\pm$2.08	47.33 $\pm$3.06	+7.33 $\pm$4.93
Qwen / CRM	GWM-M, $K = 10$	46.00 $\pm$1.73	44.63 $\pm$0.51	50.33 $\pm$0.58	52.33 $\pm$1.15	+6.33 $\pm$2.52
(b) Selection-control Configuration						
Qwen / EOPS	Joint selection, $K = 8$	40.00 $\pm$0.00	37.08 $\pm$5.20	36.25 $\pm$3.31	40.83 $\pm$3.15	+0.83 $\pm$3.15
Qwen / CRM	Graph-free vote, $K = 10$	46.00 $\pm$1.73	44.07 $\pm$1.53	48.67 $\pm$3.79	48.67 $\pm$3.79	+2.67 $\pm$3.51

Table 2: Success over three repeats of 100 CRM or 80 EOPS tasks. Smaller $\pm$ terms: sample SD across repeats (pp); Δ: success minus baseline, with SD of repeat-wise differences. (a) CRM controls use each configuration's exact pools. (b) 240/240 EOPS outcomes per policy and 300/300 CRM pools; baselines reuse earlier executions. CRM random is an exact pool expectation. K: CRM rollouts or EOPS next messages; inference budgets differ. GWM-S/L: 1,573/14,660 discovery rollouts.

GWM; the online graph-free majority-vote configuration generates its own candidates. A policy-control study executes EOPS random and exact-message-majority policies alongside joint selection, and evaluates random choice on graph-free CRM majority pools. These separate executions reuse the same task cohorts and stored direct-generation baselines. To compare the use of state/history evidence across scoring models, both judges also score a shared 600-task cohort.

Failure-triggered advice. GWM can offer advice when the matched state has an elevated historical failure rate or repeated tool calls indicate a lack of progress. This trigger targets possible failure before the task ends; the advice suggests an alternative continuation. Appendix B.2 specifies the triggers, limits, and abstention behavior.

Reuse settings. In-domain reuse keeps discovery and evaluation within one benchmark; cross-benchmark reuse exchanges CRM and EOPS graphs rather than withholding domains. Same-model and cross-model reuse compare discovery source and acting policy, independently of which model performs selection. For Gemma, the small memory (GWM-S) and large memory (GWM-L) use 1,573 and 14,660 Qwen discovery rollouts, respectively. This tests whether GWM still guides the policy with fewer discovery rollouts. A rollout is a recorded task attempt. GWM-M names the separately built Qwen CRM configuration, using 14,660 Qwen rollouts. Discovery percentages are nominal task-ID fractions (Table 4); Table 5 details the constructions.

Metrics and repeated executions. We report paired success differences in percentage points and 95% task-bootstrap intervals, retaining repeats within each sampled task. Intervals condition on recorded executions without adjusting for configuration selection. Online Gemma uses positive CRM reward, including fractional fuzzy matches, and its stored baseline pairing. Appendix E.2 explains the estimator and resampling procedure; Appendix F reports configuration-specific intervals.

6 Results: Guidance Utility and Execution Variability

Table 2 compares complete GWM configurations and selection controls with recorded direct-generation baselines and alternative selection rules. The three CRM GWM configurations have higher observed mean success, with baseline differences of 6.33–12.33 pp. Panel (b) reports separate selection-control executions.

Recorded Gemma success is higher with GWM (Q1). Success here means positive recorded CRM reward, including fractional CRM rewards. With GWM-L, Gemma whole-episode guidance at $K = 8$ averages 52.33% success across three runs of 100 CRM tasks each, versus 40.00% for its recorded baseline (+12.33 pp, 95% CI $[+6.67, +18.00]$; Table 2). GWM-S reaches 47.33% (+7.33 pp); the GWM-L configuration exceeds GWM-S by 5.00 pp $[+1.33, +9.00]$. Both use Gemma for selection, with failure-triggered advice. On the same pools, GWM-L/S exceed uniform random selection by 11.83/7.58 pp and majority vote by 7.33/3.00 pp, respectively. Using the separately archived original-evaluator output, the GWM-L–baseline difference is +8.00 pp $[+3.00, +13.33]$.

Evaluated GWM configurations improve recorded Qwen success on CRM and EOPS (Q1). CRM whole-episode selection at $K = 10$ scores 53/53/51% (mean 52.33%; recorded baseline

46.00%; +6.33 pp). GWM-M also exceeds same-pool random selection and majority vote by 7.70 and 2.00 pp, respectively. The matching Gemma GWM-L mean comes from 54/51/52 successes: both total 157/300, with different task outcomes (Appendix F.1). In EOPS executions separate from the selection-control study in Table 2(b), joint selection of $K = 8$ next-message candidates reaches 45.42% versus 40.00% (+5.42 pp), using task/tool context, deduplication and shuffled presentation. Figure 4 and Appendix F.1 distinguish the policies, complete configurations and uncertainty. These online gains use the same acting model with unchanged weights, augmented by workflow memory.

Selection controls show gains from voting and joint selection (Q1). In Table 2(b), CRM graph-free majority reaches 48.67% versus 46.00% for direct generation (+2.67 pp) and exceeds its same-pool random expectation by 4.60 pp (95% task-bootstrap CI $[2.07, 7.30]$; Holm $p = 0.0023$). EOPS joint selection reaches 40.83% versus 40.00% for direct generation (+0.83 pp) and exceeds uniform random and exact-message majority by 3.75 and 4.58 pp, respectively. Repeat rates and paired analyses appear in Appendix F.8.

CRM validation supports the gain over single generation (Q1). On 204 tasks with three repeat labels, Qwen GWM advice and $K = 10$ selection reach 48.04% versus 41.67% for single generation (+6.37 pp; task-clustered 95% CI $[2.61, 10.29]$). Graph-free selection reaches 48.86%; GWM minus graph-free selection is -0.82 pp $[-3.76, +2.12]$. Advice changes GWM candidate generation, so the conditions use different pools and this comparison addresses complete configurations.

Graph serialization reduces malformed-response fallbacks (Q3). In a separate Qwen3.8-27B experiment, selectors receive the same eight advice-free candidates and historical facts, varying graph JSON versus reversible flat-text serialization. Among 483 complete tasks, success is 48.65% versus 48.45% (+0.21 pp; exploratory paired 95% CI $[-0.41, +1.04]$). Graph JSON required fewer malformed-response fallbacks than equivalent flat text (4 versus 40 among 483 completed evaluations per condition).

The evaluated CRM GWM configurations average 2.546 million guidance tokens per 100-task run for Qwen and 1.880/2.058 million for Gemma with GWM-S/GWM-L, excluding candidate generation and discovery/build costs. On EOPS, 56.6% of selection steps have effectively identical candidates; the remaining steps show approximately 61% agreement between graph-based re-selection and the online choice, measuring decision agreement rather than outcome accuracy (Appendix F).

7 Workflow Evidence and Reuse

Workflow states carry outcome information. We test whether discovered states retain eventual-success information by comparing node outcome statistics with frequency, position and current-action predictors from the same discovery pool (Table 3). Node outcome statistics have the highest AUROC (0.745 versus at most 0.570 for the other predictors) and lowest log loss and Brier score. Thus, workflow states improve outcome discrimination, despite worse ECE calibration (0.062 versus 0.008 for current action). Later observed prefixes can contain substantial outcome evidence. Appendix F.2 details the methods.

Predictor	AUROC ↑ [95% CI]	Log loss ↓	Brier ↓	ECE ↓
Frequency	0.500 [0.500, 0.500]	0.481	0.151	0.061
Position	0.570 [0.562, 0.579]	0.467	0.146	0.019
Current action	0.475 [0.465, 0.484]	0.471	0.147	0.008
Node outcome statistics	0.745 [0.725, 0.765]	0.418	0.136	0.062

Table 3: Eventual-success prediction: 65,994 observed prefixes from 1,479 tasks excluded from the predictor's discovery graph. Estimates weight prefixes equally; intervals are secondary 95% task-cluster bootstrap CIs.

Fixed-candidate replay characterizes evidence use during selection (Q3). With candidates held fixed, graph evidence changes selection success by +1.50 pp for Gemma and −0.07 pp for Qwen in their respective replay cohorts. Full effect estimates, uncertainty, and multiplicity-adjusted tests appear in Appendices F.3 and F.7.

Matched GWM selection improves on uniform choice (Q3). On the same CRM candidate pools, GWM selection improves success by 11.83, 7.58 and 7.70 pp for Gemma GWM-L, GWM-S and Qwen GWM-M, respectively; all three contrasts survive Holm correction. These assess selection rules on pools that may already reflect GWM advice (Appendix F.6).

Utility varies with discovery data and reuse configuration (Q2). The smallest evaluated discovery collections (1,414 CRM and 2,170 EOPS rollouts) support positive guidance comparisons.

Thus, even graphs built from small discovery subsets can support guidance and selection with higher observed success than without GWM. CRM success at $K = 10$ is non-monotone: 49.0%, 51.3%, and 49.0% at nominal $p = 10\%$, $p = 20\%$, and $p = 50\%$, respectively (Table 6). The exploratory category analysis in Appendix G describes variation in observed task-level benefits.

Cross-benchmark reuse (Q2). Exchanging CRM and EOPS graphs tests whether experience from one benchmark can guide another (Figure 2). The specification fixes the target policy, tasks, evaluator and selector; available source comparisons match discovery fraction and candidate count. All eight cross-benchmark configurations exceed their single-generation baselines; three of four matched comparisons favor in-domain graphs. Appendix D.2 gives numerical results and methods.

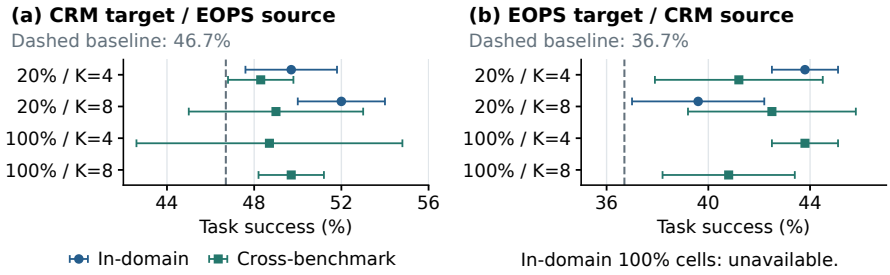


Figure 2: Qwen reuse: mean success and run SD over three runs (100 CRM, 80 EOPS tasks). p: nominal discovery-task fraction; K: candidates. Success-rate axes span 42–56% for CRM and 35–47% for EOPS.

8 Limitations

We discuss limits on attribution, workflow representation, and generalization.

Representing workflows (Sections 3–4). The graph stores observed regularities, not authoritative rules. We assume access to enterprise verification that reports task outcomes or execution issues, as exemplified by EOPS SQL verifiers and CRM task-specific evaluators. The discovered nodes summarize histories; they need not be sufficient Markov states. However, we assume that the supplied history and candidate text retain decision-relevant context absent from the graph. At a fixed retrieved state, toolset statistics cannot distinguish candidates differing only in arguments, call order, or multiplicity. To mitigate this ambiguity in advice and selection, we recommend retaining full history or dynamically selecting task-relevant validation (Appendices A; E.6).

Evaluation scope (Section 5). Task-dependent reuse and limited cohorts constrain generalization. However, we define comparisons on fixed task cohorts, report paired task-bootstrap intervals, and distinguish the recorded success criteria in Table 18.

Attributing improvements (Sections 6–7). Online gains concern combined memory, candidate generation, advice, and selection with fixed policy weights. The contribution is the GWM harness: Table 2(a) shows higher recorded CRM success with GWM selection than with uniform random or majority selection on the same K candidates. These controls match candidate availability; they do not isolate component effects.

9 Conclusion

In this paper, we introduce GWM, which discovers latent workflow states from interaction histories and connects them to observed continuations and outcomes, forming an inspectable memory for retrieval, advice, and candidate selection. Evaluated CRM configurations improve recorded success over direct generation for Qwen and Gemma while keeping policy weights fixed. Qwen-derived experience supports same-model and cross-model reuse, and GWM-L improves Gemma success over GWM-S in the evaluated comparison. On the same CRM candidate pools, GWM selection achieves higher success than expected from choosing a candidate uniformly at random. Our study found that workflow memory can support general-purpose agents in reusing organizational experience without policy retraining.

REPRODUCIBILITY STATEMENT

We have made every effort to support reproducibility through documented experimental settings, archived outcomes, source hashes, analysis scripts, and figure generators. CRMArena-Pro and EnterpriseOps-Gym are publicly available. We release the vLLM extension and all trajectories underlying the reported benchmark experiments at https://github.com/iclr-gwm/gwm. The private enterprise application in Appendix J is outside the scope of this release. The extension supports workflow-graph construction and integrates advice and candidate selection into model serving. The accompanying analysis package includes documented configurations, normalized outcomes, analysis scripts, and numerical ledgers linking released records to reported results. The release supports inspection of recorded executions and recomputation of benchmark analyses; limitations in historical checkpoint and serving-environment identification remain documented separately. Appendices E and F specify resampling, candidate-answer checking, action-label analysis, and measured costs.

ETHICS STATEMENT

Enterprise trajectories and graph examples can contain sensitive records, organization-specific procedures, and identifying tool outputs. The examples in this paper are summarized and sanitized. Releasing trajectories or graphs requires appropriate permissions, license review, and confidentiality safeguards. Observed workflow behavior is not an authoritative rule source: a successful task can still violate authorization or confidentiality requirements. Graph guidance should therefore remain subject to independent enterprise rules and outcome checks.

AI USE STATEMENT

AI assistants helped prepare the manuscript by editing author-written text, checking references, and preparing figures. The authors also used coding assistants, including Claude Code and Codex, to implement their original designs and ideas. All scientific contributions, technical methods, research ideas, and core results are entirely the authors' original work.

REFERENCES

Lakshya A Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alexandros G. Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. GEPA: Reflective prompt evolution can outperform reinforcement learning. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2507.19457v2.

Prithviraj Ammanabrolu and Mark O. Riedl. Learning knowledge graph-based world models of textual environments. In *Advances in Neural Information Processing Systems*, volume 34, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/1e747ddbea997a1b933aaf58a7953c3c-Abstract.html.

Petr Anokhin, Nikita Semenov, Artyom Sorokin, Dmitry Evseev, Andrey Kravchenko, Mikhail Burtsev, and Evgeny Burnaev. AriGraph: Learning knowledge graph world models with episodic memory for LLM agents. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, pp. 12–20, 2025. doi: 10.24963/ijcai.2025/2. URL https://www.ijcai.org/proceedings/2025/2.

Anonymous organization. Enterprise approval application: An anonymized case study, 2023. Anonymized reference; identifying details withheld for review.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-Bench: Evaluating conversational agents in a dual-control environment, 2025. URL https://arxiv.org/abs/2506.07982v1. Version 1.

Léo Boisvert, Megh Thakkar, Maxime Gasse, Massimo Caccia, Thibault Le Sellier De Chezelles, Quentin Cappart, Nicolas Chapados, Alexandre Lacoste, and Alexandre Drouin. WorkArena++: Towards compositional planning and reasoning-based common knowledge work tasks. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/0b82662b6c32e887bb252a74d8cb2d5e-Abstract-Datasets_and_Benchmarks_Track.html.

Jonathan Bragg, Mike D'Arcy, Nishant Balepur, Dan Bareket, Bhavana Dalvi, Sergey Feldman, Dany Haddad, Jena D. Hwang, Peter Jansen, Varsha Kishore, Bodhisattwa Prasad Majumder, Aakanksha Naik, Sigal Rahamimov, Kyle Richardson, Amanpreet Singh, Harshit Surana, Aryeh Tiktinsky, Rosni Vasu, Guy Wiener, Chloe Anastasiades, Stefan Candra, Jason Dunkelberger, Dan Emery, Rob Evans, Malachi Hamada, Regan Huff, Rodney Kinney, Matt Latzke, Jaron Lochner, Ruben Lozano-Aguilera, Cecile Nguyen, Smita Rao, Amber Tanaka, Brooke Vlahos, Peter Clark, Doug Downey, Yoav Goldberg, Ashish Sabharwal, and Daniel S. Weld. AstaBench: Rigorous benchmarking of AI agents with a scientific research suite. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2510.21652v2.

Hyungjoo Chae, Namyoung Kim, Kai Tzu-iunn Ong, Minju Gwak, Gwanwoo Song, Jihoon Kim, Sunghwan Kim, Dongha Lee, and Jinyoung Yeo. Web agents with world models: Learning and leveraging environment dynamics in web navigation. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/a00548031e4647b13042c97c922fadf1-Abstract-Conference.html.

Prafulla Kumar Choubey, Xiangyu Peng, Shilpa Bhagavath, Caiming Xiong, Shiva Kumar Pentyala, and Chien-Sheng Wu. Turning conversations into workflows: A framework to extract and evaluate dialog workflows for service AI agents. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 3933–3954, 2025. URL https://aclanthology.org/2025.findings-acl.203/.

Yutong Dai, Krithika Ramakrishnan, Jing Gu, Matthew Fernandez, Yanqi Luo, Viraj Prabhu, Zhenyu Hu, Silvio Savarese, Caiming Xiong, Zeyuan Chen, and Ran Xu. SCUBA: Salesforce computer use benchmark. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/29c8c615b3187ee995029284702d3f43-Abstract-Conference.html.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. From local to global: A graph RAG approach to query-focused summarization, 2024. URL https://arxiv.org/abs/2404.16130v2. Version 2.

Amine El Hattami, Stefania Raimondo, Issam Laradji, David Vazquez, Pau Rodriguez, and Chris Pal. Workflow discovery from dialogues in the low data regime, 2023. URL https://arxiv.org/abs/2205.11690v2. Version 2.

Bahare Fatemi, Jonathan Halcrow, and Bryan Perozzi. Talk like a graph: Encoding graphs for large language models. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/bf72f65f30eedf5d48da6980ee02b589-Abstract-Conference.html.

Tao Feng, Yexin Wu, Guanyu Lin, and Jiaxuan You. Graph world model. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 16931–16955. PMLR, 2025. URL https://proceedings.mlr.press/v267/feng25p.html.

Romain Froger, Pierre Andrews, Matteo Bettini, Amar Budhiraja, Ricardo Silveira Cabral, Virginie Do, Emilien Garreau, Jean-Baptiste Gaya, Hugo Laurençon, Maxime Lecanu, Kunal Malkan, Dheeraj Mekala, Pierre Ménard, Gerard Moreno-Torres Bertran, Ulyana Piterbarg, Mikhail Plekhanov, Mathieu Rita, Andrey Rusakov, Vladislav Vorotilov, Mengjue Wang, Ian Yu, Amine Benhalloum, Grégoire Mialon, and Thomas Scialom. Gaia2: Benchmarking LLM agents on dynamic and asynchronous environments. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2602.11964v1.

Yao Fu, Dong-Ki Kim, Jaekyeom Kim, Sungryull Sohn, Lajanugen Logeswaran, Kyunghoon Bae, and Honglak Lee. AutoGuide: Automated generation and selection of context-aware guidelines for large language model agents. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/d8efbb5dd415974eb095c3f06bff1f48-Abstract-Conference.html.

Gemma Team. Gemma 4 technical report, 2026. URL https://arxiv.org/abs/2607.02770.

Leonardo Hernandez Cano, Maxine Perroni-Scharf, Neil Dhir, Arun Ramamurthy, and Armando Solar-Lezama. Neurosymbolic world models for sequential decision making. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 23047–23062. PMLR, 2025. URL https://proceedings.mlr.press/v267/hernandez-cano25a.html.

Kung-Hsiang Huang, Akshara Prabhakar, Onkar Thorat, Divyansh Agarwal, Prafulla Kumar Choubey, Yixin Mao, Silvio Savarese, Caiming Xiong, and Chien-Sheng Wu. CRMArena-Pro: Holistic assessment of LLM agents across diverse business scenarios and interactions. *Transactions on Machine Learning Research*, 2026a. URL https://arxiv.org/abs/2505.18878v1.

Wenhao Huang, Qingwen Zeng, Qiyue Chen, Zijie Guo, Yu Sun, Cheng Yang, Siru Ouyang, Jiri Gesi, Fang Wu, Jiayi Zhang, Huaming Chen, Bang Liu, Xiangru Tang, and Chenglin Wu. Latent action reparameterization for efficient agent inference, 2026b. URL https://arxiv.org/abs/2605.18597v2.

Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1–2):99–134, 1998. doi: 10.1016/S0004-3702(98)00023-X. URL https://www.cassandra.org/arc/papers/aij98.pdf.

S. Kullback and R. A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951. doi: 10.1214/aoms/1177729694. URL https://doi.org/10.1214/aoms/1177729694.

Sander J. J. Leemans, Dirk Fahland, and Wil M. P. van der Aalst. Discovering block-structured process models from event logs—a constructive approach. In *Application and Theory of Petri Nets and Concurrency*, volume 7927 of *Lecture Notes in Computer Science*, pp. 311–329. Springer, 2013. URL https://leemans.ch/publications/papers/pn2013leemans.pdf.

Evan Zheran Liu, Kelvin Guu, Panupong Pasupat, Tianlin Shi, and Percy Liang. Reinforcement learning on web interfaces using workflow-guided exploration. In *International Conference on Learning Representations*, 2018. URL https://arxiv.org/abs/1802.08802v1.

Shiva Krishna Reddy Malay, Shravan Nayak, Jishnu Sethumadhavan Nair, Sagar Davasam, Aman Tiwari, Sathwik Tejaswi Madhusudhan, Sridhar Krishna Nemala, Srinivas Sunkara, and Sai Rajeswar. EnterpriseOps-Gym: Environments and evaluations for stateful agentic planning and tool use in enterprise settings, 2026. URL https://arxiv.org/abs/2603.13594v1.

Leland McInnes, John Healy, and Steve Astels. hdbscan: Hierarchical density based clustering. *Journal of Open Source Software*, 2(11):205, 2017. doi: 10.21105/joss.00205. URL https://joss.theoj.org/papers/10.21105/joss.00205.

Leland McInnes, John Healy, and James Melville. UMAP: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018. URL https://arxiv.org/abs/1802.03426.

Do June Min, Paloma Sodhi, and Ramya Ramakrishnan. Workflow-guided response generation for task-oriented dialogue, 2023. URL https://arxiv.org/abs/2311.08300v1.

Shikhar Murty, Christopher D. Manning, Peter Shaw, Mandar Joshi, and Kenton Lee. BAGEL: Bootstrapping agents by guiding exploration with language. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 36894–36910. PMLR, 2024. URL https://proceedings.mlr.press/v235/murty24a.html.

Siru Ouyang, Jun Yan, I-Hung Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long T. Le, Samira Daruki, Xiangru Tang, Vishy Tirumalashetty, George Lee, Mahsan Rofouei, Hangfei Lin, Jiawei Han, Chen-Yu Lee, and Tomas Pfister. ReasoningBank: Scaling agent self-evolving with reasoning memory. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2509.25140v2. Version 2.

Jiayi Pan, Yichi Zhang, Nicholas Tomlin, Yifei Zhou, Sergey Levine, and Alane Suhr. Autonomous evaluation and refinement of digital agents. In *Conference on Language Modeling*, 2024. URL https://arxiv.org/abs/2404.06474.

Shuofei Qiao, Runnan Fang, Ningyu Zhang, Yuqi Zhu, Xiang Chen, Shumin Deng, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. Agent planning with world knowledge model. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/d032263772946dd5026e7f3cd22bce5b-Abstract-Conference.html.

Qwen Team. Qwen3.6-27B: Flagship-level coding in a 27B dense model, April 2026. URL https://qwen.ai/blog?id=qwen3.6-27b.

Ramya Ramakrishnan, Ethan R. Elenberg, Hashan Narangodage, and Ryan McDonald. Multi-step dialogue workflow action prediction, 2023. URL https://arxiv.org/abs/2311.09593v2.

Sentence Transformers. all-MiniLM-L6-v2 model card, 2021. URL https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2. Accessed 2026-09-17.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-0377. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

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. URL https://aclanthology.org/2024.acl-long.850/.

Karen Ullrich, Jingtong Su, Claudia Shi, Arjun Subramonian, Amir Bar, Ivan Evtimov, Nikolaos Tsilivis, Randall Balestriero, Julia Kempe, and Mark Ibrahim. OpenApps: Simulating environment variations to measure UI-agent reliability. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2511.20766v1.

Wil M. P. van der Aalst and Alessandro Berti. Discovering object-centric Petri nets. *Fundamenta Informaticae*, 175(1–4):1–40, 2020. doi: 10.3233/FI-2020-1946. URL https://arxiv.org/abs/2010.02047.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 63897–63911. PMLR, 2025. URL https://proceedings.mlr.press/v267/wang25bx.html.

Xiangyu Wen, Jianyuan Zhong, Zhijian Xu, and Qiang Xu. Guideline compliance in task-oriented dialogue: The chained prior approach. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 6765–6791, 2025. URL https://aclanthology.org/2025.findings-naacl.377/.

Yiran Wu, Tianwei Yue, Shaokun Zhang, Chi Wang, and Qingyun Wu. StateFlow: Enhancing LLM task-solving through state-driven workflows. In *Conference on Language Modeling*, 2024. URL https://arxiv.org/abs/2403.11322v5.

Frank (Fangzheng) Xu, Yufan Song, Boxuan Li, Yuxuan Tang, Kritanjali Jain, Mengxue Bao, Zora Wang, Xuhui Zhou, Zhitong Guo, Murong Cao, Mingyang Yang, Hao Yang Lu, Amaad Martin, Zhe Su, Leander Maben, Raj Mehta, Wayne Chi, Lawrence Jang, Yiqing Xie, Shuyan Zhou, and Graham Neubig. TheAgentCompany: Benchmarking LLM agents on consequential real world tasks. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/0d744742f6fac4d1134c019b7cef3c8a-Abstract-Datasets_and_Benchmarks_Track.html.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.03629v3.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains, 2024. URL https://arxiv.org/abs/2406.12045v1. Version 1.

Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. AgentTuning: Enabling generalized agent abilities for LLMs. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 3053–3077. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.181. URL https://aclanthology.org/2024.findings-acl.181.pdf.

Jiahan Zhang, Muqing Jiang, Nanru Dai, Taiming Lu, Arda Uzunoglu, Shunchi Zhang, Yana Wei, Jiahao Wang, Vishal M. Patel, Paul Pu Liang, Daniel Khashabi, Cheng Peng, Rama Chellappa, Tianmin Shu, Alan Yuille, Yilun Du, and Jieneng Chen. World-in-World: World models in a closed-loop world. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2510.18135v2.

Lunjun Zhang, Ge Yang, and Bradly C. Stadie. World model as a graph: Learning latent landmarks for planning. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 12611–12620. PMLR, 2021. URL https://proceedings.mlr.press/v139/zhang21x.html.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 Embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025. URL https://arxiv.org/abs/2506.05176.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM agents are experiential learners. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(17):19632–19642, 2024. doi: 10.1609/aaai.v38i17.29936. URL https://ojs.aaai.org/index.php/AAAI/article/view/29936.

Longtao Zheng, Rundong Wang, Xinrun Wang, and Bo An. Synapse: Trajectory-as-exemplar prompting with memory for computer control. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2306.07863v3.

APPENDIX INDEX

Appendix	Contents	Page
A	**Formal analysis of workflow abstraction and policy reuse**	**16**
B	**Mathematical formulations and algorithms**	**17**
B.1	Proofs and counterexamples	17
B.2	Graph construction and guidance interfaces	18
C	**Discovery datasets and GWM construction**	**19**
C.1	GWM constructions and discovery data	20
D	**Extended experimental results**	**20**
D.1	In-domain evaluation across discovery fractions	21
D.2	Cross-benchmark transfer evaluation	21
D.3	Cross-model evaluation and discovery-source comparison	22
D.4	Extended comparisons with related methods	23
E	**Retrospective analysis of recorded executions**	**25**
E.1	Experimental configurations and scope	25
E.2	Paired outcome analysis and baseline variability	25
E.3	Candidate-answer replay and selection analysis	27
E.4	Action-label sensitivity of reconstructed successor distributions	28
E.5	Prefix-level associations of cluster labels	29
E.6	Information stored in the graph and used by GWM	30
E.7	CRM execution cost	30
F	**Online guidance and fixed-candidate evaluation**	**31**
F.1	Online guidance with GWM-S and GWM-L	31
F.2	Validation-split selection and outcome diagnostics	34
F.3	Fixed-candidate evidence comparisons	35
F.4	Equal-information graph and text serialization	35
F.5	Temporal alignment and decision sensitivity	36
F.6	Matched selection controls on the online CRM pools	36
F.7	Paired history-evidence replay on a common CRM cohort	37
F.8	Online selection policies and graph-free CRM controls	37
G	**Exploratory analysis by workflow category**	**39**
H	**GWM advice templates**	**40**
I	**Qualitative case studies**	**41**
I.1	Workflow advice for CRM case routing	41
I.2	Task recovery and regression	41
J	**An anonymized enterprise approval application**	**43**
J.1	Approval review as a workflow-memory interface	43
J.2	Exploratory configuration results	44

A FORMAL ANALYSIS OF WORKFLOW ABSTRACTION AND POLICY REUSE

Fix an environment e, goal g, authoritative rule source $\mathcal{R}_e$, and fitted history and action abstractions ϕ, ψ. Let V_t indicate a rule violation associated with the next executed action, and define

$$K_e(\cdot \mid h, a, g) = \mathrm{Law}_e\big((z_{t+1}, V_t) \mid h, g;\ \text{execute } a\big).$$

Here h is the observed history available before action a, z_{t+1} is the next abstract workflow state, and V_t equals one for a violation of $\mathcal{R}_e$ and zero otherwise. The operator Law_e denotes their joint probability distribution in environment e, given history h and goal g and after executing a; the placeholder $\cdot$ denotes the kernel's measurable-set argument in the joint outcome space. The kernel describes environment behavior under execution of a, whereas the graph estimates frequencies from traces. When rule violations are unmeasured, only the next-state marginal is available as an observed prediction target. A sufficient workflow abstraction admits a kernel $\overline{K}_e$ such that

$$K_e(\cdot \mid h, a, g) = \overline{K}_e(\cdot \mid \phi(h), \psi(a), g) \tag{6}$$

for the feasible history–action pairs under consideration. Here $\phi(h)$ is the abstract workflow state, $\psi(a)$ is the action label, and $\overline{K}_e$ is a kernel defined on these summaries and the goal. Equality means that these summaries preserve the joint next-state and rule-violation distribution. Terminal-success prediction and long-horizon control require additional assumptions about outcomes and continuation policies.

Proposition 1 (Information lost by abstraction)**.** *Suppose two feasible pairs (h_i, a_i) share the same abstract state and action label, and let $K_i = K_e(\cdot \mid h_i, a_i, g)$. Any predictor $\widehat{K}$ using only those labels satisfies*

$$\max_{i \in \{1,2\}} D_{\mathrm{TV}}(K_i, \widehat{K}) \geq \tfrac{1}{2} D_{\mathrm{TV}}(K_1, K_2).$$

Here D_{TV} is total variation distance, the largest absolute difference between the probabilities assigned to the same event. The largest prediction error across the two merged history–action pairs is at least half the distance between their true outcome distributions.

Proof. Both pairs receive the same predicted distribution. By the triangle inequality, $D_{\mathrm{TV}}(K_1, K_2)$ is at most $D_{\mathrm{TV}}(K_1, \widehat{K}) + D_{\mathrm{TV}}(\widehat{K}, K_2)$, hence at most twice the larger error. □

Consider an organization that requires approval before sharing a document. If approved and unapproved histories receive the same abstract state, a predictor using that state and the same sharing action cannot represent violation probabilities zero and one with maximum error below $1/2$. Discarding action arguments can similarly merge permitted and impermissible recipients. This hypothetical example illustrates information loss; an approval variable is recoverable only if the available observations identify it. The result applies to predictors restricted to the abstraction, while GWM can also use history and candidate details during selection. Note that such richer inputs may recover distinctions absent from graph statistics, but do not establish graph-level sufficiency.

Source-policy dependence. For aligned pre-action records generated by a history-conditioned policy β in the fixed environment, the population abstract successor law is a mixture:

$$P_\beta(z' \mid z, u, g) = \mathbb{E}_\beta\big[K_e^Z(z' \mid h_t, a_t, g) \mid \phi(h_t) = z, \psi(a_t) = u, g\big], \tag{7}$$

Here z and u denote the abstract state and action label, z' is a possible successor state, and K_e^Z is the next-state marginal of K_e. The conditional expectation $\mathbb{E}_\beta$ averages over histories and actions generated by source policy β within that state–action cell; the conditioning event must have positive probability. Under Eq. (6), the integrand is constant within each abstract state–action cell. Taking its conditional expectation gives $P_\beta = \overline{K}_e^Z$, the next-state marginal of the abstract kernel, for every source policy with support in that cell. Without sufficiency, policies can induce different mixtures of histories and tool arguments, producing different aggregate relations. This population result provides no coverage or finite-sample guarantee. The derivation holds the environment, organizational rules, and fitted partition fixed.

The evaluated toolset-conditioned table associates successors with an action already included in the source prefix (Section 4), so Eq. (7) does not directly describe that table. Its terminal-success associations also depend on the source policy's subsequent decisions. The conditional predictive result concerns aligned pre-action records; the evaluated graph stores the observational associations described above.

B MATHEMATICAL FORMULATIONS AND ALGORITHMS

Outcome statistics. Outcome features associate states and toolsets with rollout success or failure. Let $y_r = \mathbf{1}\{\texttt{overall_success}_r\}$ denote the recorded binary outcome for rollout r, with one indicating success and zero failure. As noted in Section 5, this outcome is determined by the verification process and success criterion defined by the target enterprise or benchmark. Sensitive enterprise workflows may require multiple checks of task completion, integrity, and enterprise rules before recording success; these checks can also identify rule violations. EOPS requires all reported SQL verifiers to pass, whereas CRM uses its task-specific evaluator and recorded success criterion. For a state z in benchmark domain d, let $\mathcal{R}_d$ be the distinct discovery rollouts included in that domain's graph statistics and $\mathcal{R}_z \subseteq \mathcal{R}_d$ those visiting z. Define

$$\widehat{f}(z) = \frac{\sum_{r\in\mathcal{R}_z}(1-y_r)}{|\mathcal{R}_z|}, \qquad \widehat{f}_d = \frac{\sum_{r\in\mathcal{R}_d}(1-y_r)}{|\mathcal{R}_d|}, \qquad \mathrm{lift}(z) = \frac{\widehat{f}(z)}{\widehat{f}_d}. \tag{8}$$

For the first two ratios, each numerator counts failed rollouts, while the corresponding set cardinality in the denominator counts distinct rollouts. The state failure rate $\widehat{f}(z)$ is the fraction of visiting rollouts that failed. The domain failure rate $\widehat{f}_d$ is the fraction of all included domain rollouts that failed. Failure lift divides the state rate by this domain reference rate. A lift above one indicates a higher observed failure rate than the domain reference. Rates require nonempty populations; lift also requires $\widehat{f}_d > 0$. Tool-success associations likewise count each included rollout once per state–toolset pair, whereas successor frequencies count record transitions. Failure lift is descriptive: it can reflect task difficulty, recorded enterprise rule violations, domain composition, and the discovery policy rather than the effect of entering a state.

B.1 PROOFS AND COUNTEREXAMPLES

Conditional analysis: coverage and score error. We define a conditional decomposition of next-message selection loss to distinguish missing useful candidates from errors in ranking the candidates generated by the acting policy. For a given history h, let $\mathcal{A}(h)$ be the feasible candidate-message space and $C \subseteq \mathcal{A}(h)$ the nonempty finite pool sampled by that policy. Let ρ be the policy used for subsequent execution and $Q^\rho(h,c) \in [0,1]$ the expected terminal score after executing c and continuing with ρ. Suppose the selector's scores approximate this value uniformly on C, with error at most ε, and the selected candidate $\widehat{c} \in C$ is within η of the maximum score:

$$\sup_{c\in\mathcal{A}(h)} Q^\rho(h,c) - Q^\rho(h,\widehat{c}) \le \underbrace{\sup_{c\in\mathcal{A}(h)} Q^\rho(h,c) - \max_{c\in C} Q^\rho(h,c)}_{L_{\mathrm{coverage}}(h,C)} + 2\varepsilon + \eta. \tag{9}$$

The left side is the expected-score gap between the best feasible message and the selected message; the supremum allows the best feasible value to be approached without being attained. Coverage measures loss from useful candidates missing from the pool; 2ε accounts for score errors on the best and selected candidates, and η allows imperfect maximization. The proof applies the score-error bound to both candidates (Proposition 3 below). If a baseline candidate is included and the selected candidate's score exceeds its score by margin m, the expected-score advantage is at least $m - 2\varepsilon$. Selector scores express preferences. Appendix E.3 measures candidate availability and replay selection; Eq. (9) decomposes loss under a uniform score-error bound and fixed continuation policy.

Proposition 2 (Frequency estimator)**.** *Let j index outcome categories with nonnegative weights $n_j \ge 0$, total weight $W = \sum_j n_j > 0$, and normalized frequencies $q_j = n_j/W$. Among probability vectors p with nonnegative entries summing to one, q minimizes the weighted negative log-probability loss $\ell(p) = -\sum_j n_j \log p_j$, with zero-weight terms interpreted as zero and positive-weight terms at $p_j = 0$ as infinite.*

Proof. For a finite-loss p, $\ell(p) - \ell(q) = W\sum_j q_j \log(q_j/p_j) = W D_{\mathrm{KL}}(q\|p) \ge 0$. Here D_{KL} is the Kullback–Leibler divergence (Kullback & Leibler, 1951) from q to p; its nonnegativity shows that no other probability vector has a lower loss. The infinite-loss case is immediate. If all counts vanish, normalization is undefined and the miner stores an empty map. Replacing n_j with $n_j + \alpha p_{0j}$

yields a smoothed estimator, where p_{0j} is the normalized prior probability of category j and $\alpha \geq 0$ is the total prior weight. The combined total weight must be positive; the implementation uses the unsmoothed estimator. □

The result concerns fixed counts and requires no independence assumption. It therefore does not by itself provide a likelihood interpretation or confidence bounds for adaptively collected trajectories. Note also that clustering does not guarantee sufficient states: two histories can share a cluster and toolset label while differing in an argument that determines whether a calendar event has the required type. Their next-outcome distributions can differ despite identical abstract labels. To mitigate this ambiguity in guidance, we assume that the history and candidate text supplied to GWM retain the decision-relevant context, including such arguments. GWM can then condition advice and selection on distinctions absent from the graph, without requiring graph-level sufficiency.

Proposition 3 (Approximate selection)**.** *Fix h, a nonempty finite candidate set C, a selected candidate $\widehat{c} \in C$, and a common continuation policy ρ. Let $s(c)$ denote the selector's score for candidate c. Suppose $\varepsilon, \eta \geq 0$, $|s(c) - Q^\rho(h, c)| \leq \varepsilon$ for every $c \in C$, and $s(\widehat{c}) \geq \max_{c \in C} s(c) - \eta$. Then*

$$0 \leq \max_{c \in C} Q^\rho(h, c) - Q^\rho(h, \widehat{c}) \leq 2\varepsilon + \eta.$$

The middle quantity is the expected-score loss from choosing $\widehat{c}$ instead of the best candidate already in C. The bound combines the score errors on the two candidates and the permitted error in maximizing the selector's score. Adding the coverage loss gives Eq. (9). For a baseline $c_b \in C$ and margin $m = s(\widehat{c}) - s(c_b)$, $Q^\rho(h, \widehat{c}) - Q^\rho(h, c_b) \geq m - 2\varepsilon$.

Proof. Let c^* maximize Q^ρ on C. The assumptions imply

$$Q^\rho(h, c^*) \leq s(c^*) + \varepsilon \leq s(\widehat{c}) + \eta + \varepsilon \leq Q^\rho(h, \widehat{c}) + 2\varepsilon + \eta.$$

The three inequalities use, respectively, score accuracy for the best candidate, approximate score maximization, and score accuracy for the selected candidate. Nonnegativity follows because $\widehat{c} \in C$. Add and subtract $\max_{c \in C} Q^\rho(h, c)$ to obtain the full-action inequality. For the margin claim, subtract $Q^\rho(h, c_b) \leq s(c_b) + \varepsilon$ from $Q^\rho(h, \widehat{c}) \geq s(\widehat{c}) - \varepsilon$. □

Uniform score accuracy is essential. For candidates with true values 0.9 and 0.1 and scores 0.1 and 0.9, exact score maximization selects the worse action. Candidate coverage is separately necessary: if every candidate has value zero, even perfect ranking cannot select an absent action with value one. Finally, the proposition assumes a fixed continuation policy. Changing the selector at subsequent steps changes that policy, so an episode-wide guarantee requires additional assumptions.

B.2 Graph construction and guidance interfaces

Discovery. The implementation expands each assistant message with its trailing tool results and embeds the resulting text as normalized 384-dimensional vectors with `all-MiniLM-L6-v2` (Sentence Transformers, 2021), subject to its 256-word-piece input limit. Defaults are ten UMAP (McInnes et al., 2018) dimensions, 15 neighbors, seed 17, and HDBSCAN (McInnes et al., 2017) minimum cluster size 50. UMAP and HDBSCAN are fitted on training-tagged rows. Non-training rows receive cluster assignments through approximate prediction. Noise points are excluded from centroid construction. Each non-noise training cluster is represented by a normalized mean embedding and a retrieval radius equal to its 90th-percentile cosine distance. These are reference implementation defaults; command-line arguments configure the encoder, text-length limits, dimensionality reduction, random seed, and clustering parameters. Note that we also used larger embedding models, including Qwen3-Embedding-8B (Zhang et al., 2025), but observed degraded performance when limited discovery rollouts were available to construct small workflow graphs.

Retrieval and advice. Runtime assignment embeds the current prefix and searches within the task domain, or across domains for transfer. Assignments beyond a scaled training-distance threshold are rejected. GWM can still use nearby states marked as low confidence, so assignment rejection does not necessarily remove graph context. Failure-triggered advice uses elevated historical failure rates or repeated-tool loops, with at most two accepted injections per episode and no repeated accepted state. A state triggers advice with at least 20 supporting rollouts and failure lift of at least 1.3: its visiting-rollout failure rate is at least 1.3 times the domain rate. Repeated-tool loops require three

consecutive failed calls with the same toolset and identical results, or three calls with the same toolset and identical results among the last four. The advice generator can abstain.

Algorithm 1: Next-message selection interface
Input: fixed graph G, current history h (including injected GWM advice when enabled), goal and tool catalog, policy π_θ, nonempty candidate tuple $C_K(h)$.
1. Deduplicate candidates by full serialized action/content; retain original indices.
2. Retrieve state support, empirical tool/outcome associations, and optional examples.
3. If candidates coincide, return equal scores and preserve the first candidate under the tie rule. Otherwise score the candidates with GWM using this context.
4. Parse bounded preference scores and explicit best votes; apply configured voting or tournament rules. On selector failure, use the graph heuristic fallback.
5. Choose the highest-ranked candidate with the recorded tie rule; execute it once.
Interface: next-message scoring occurs before the chosen message is executed. Complete-rollout selection is a separate interface whose candidates have already been executed.

Next-message deduplication preserves serialized arguments and content, whereas complete-rollout deduplication normalizes final-answer text. GWM clips returned scores to $[0, 1]$ and can adjust them to favor an explicit best choice; voting and tournament variants further affect ranking. The fallback returns toolset success when a known state/toolset pair has support from at least three distinct rollouts. Otherwise, it classifies the history with the candidate appended and returns the corresponding cluster or domain success rate, $1 - \widehat{f}(z)$ or $1 - \widehat{f}_d$ from Eq. (8). Appending a candidate changes the representation but does not execute the action or observe its consequence.

C Discovery datasets and GWM construction

To study the discovery-rollout requirements of a given benchmark, we define the discovery fraction p as the nominal share of discovery task IDs retained with their associated rollouts; it is not a fraction of evaluation tasks. A rollout is one recorded task attempt, and a task may have multiple rollouts. For example, repeated attempts at different sampling temperatures can generate diverse rollouts for each task. When few tasks are available, oversampling them through additional attempts can increase the rollout count. Thus $p = 20\%$ identifies the nominal task subset, not a fixed number or percentage of rollouts. The following inventory gives the task and rollout counts for each recorded construction. Table 4 summarizes the historical discovery-ratio inventory. Nominal ratios subsample task IDs and retain their associated rollouts, changing trajectory count, the learned partition, and transition support together. The GWM conditions for online guidance and replay, including GWM-S and GWM-L used in the Gemma experiment, are identified separately from these ratio labels in Table 5.

Target	Qwen graph	Rollouts	Train tasks	Domains	States	Edges
CRM	$p = 10\%$	1,414	177	22	53	179
	$p = 20\%$	2,850	357	22	93	400
	$p = 50\%$	7,558	946	22	181	1,015
	$p = 100\%$	14,660	1,836	22	346	2,669
EOPS	$p = 10\%$	2,170	53	8	112	1,385
	$p = 20\%$	4,013	98	8	212	2,933
	$p = 50\%$	10,279	251	8	527	9,656
	$p = 100\%$	20,864	510	8	1,013	20,762

Table 4: Historical Qwen discovery-ratio corpora and graph sizes. Edges count stored successor relations. Ratios are nominal; for example, EOPS $p = 20\%$ contains 98 of 510 tasks.

The table reports historical train-tagged inventory counts. Separately, the EOPS inputs labeled $p = 20\%$ and $p = 100\%$ each contain 57 validation task IDs, with no overlap with the 80 evaluation tasks. For EOPS, the label-sensitivity analysis in Appendix E.4 reports 2,170 distinct rollout identifiers, while the discovery inventory lists 4,013 rollouts for $p = 20\%$.

The inventories grow from 53 to 346 CRM states and from 112 to 1,013 EOPS states between $p = 10\%$ and $p = 100\%$. The discovery-ratio success grid is nonmonotone (Table 6). State count

alone does not measure the usefulness of retrieved context. Note that the goal is to identify minimum discovery requirements for a target enterprise's workflows, rather than to study scalability. In practice, a corresponding study on a separate validation dataset is needed to select the discovery budget for the target enterprise's initial GWM.

To verify discovery–evaluation separation, we checked that none of the four stored graph-statistics inputs shares a task identifier with its corresponding fixed evaluation set of 100 CRM or 80 EOPS tasks.

C.1 GWM CONSTRUCTIONS AND DISCOVERY DATA

GWM-S and GWM-L use small and large Qwen-derived CRM memories built from 1,573 and 14,660 discovery rollouts, respectively. A rollout is one recorded task attempt. CRM discovery uses repeated attempts on training tasks across a range of sampling temperatures to collect diverse trajectories. Each workflow graph contains inferred workflow states, matching representations, observed transitions, outcome/support statistics, and retrievable examples. The labels describe the discovery memories used to guide Gemma. The Qwen online and replay experiments use separately identified constructions (Table 5).

GWM condition	Rollouts	Evaluation interface	Task-membership evidence
GWM-S	1,573	Gemma; advice and whole-episode selection, $K = 8$	Fixed 100-task online list
GWM-L	14,660	Gemma; advice and whole-episode selection, $K = 8$	Fixed 100-task online list
GWM-M	14,660	Qwen; advice and whole-episode selection, $K = 10$	
Qwen CRM aligned	14,660	Same interface; aligned transition statistics	Separate GWM construction input checked separately
Replay GWM and action-aligned graph	2,850	Selection: Gemma on 600 or Qwen on 1,479 task pools	357 construction tasks excluded from replay

Table 5: CRM GWM identities in the online guidance and replay comparisons. All five conditions use Qwen-generated CRM discovery experience. Online rows name the model used for both acting and GWM guidance; the replay row names the model used for scoring. GWM-S (small memory) and GWM-L (large memory) denote construction from 1,573 and 14,660 discovery rollouts, respectively; a rollout is one recorded task attempt. GWM-M names the Qwen CRM configuration, built separately from GWM-L despite the equal rollout count.

D EXTENDED EXPERIMENTAL RESULTS

This section examines how workflow guidance depends on candidate count, discovery data, graph source, and evaluation policy. We report the available experimental grids to show favorable configurations alongside regressions. Across these comparisons, the central question is whether discovered workflow context improves task completion, and under which conditions its benefits persist.

The CRM experiment A evaluation uses a 100-task test subset. Its experimental configuration specifies the Qwen policy, medium schema drift and injected distractor records (the benchmark's *context rot* setting), an eight-turn limit, policy sampling temperature 0.1, and selection temperature 0.7. The CRM selection conditions select among executed rollouts. In addition, we report task-level reanalysis in Appendix E.

To assess which tasks benefit or regress, we use the paired analysis in Appendix E.2. Table 10 shows that CRM experiment A reaches 52.33% mean success at $K = 10$, compared with 46.00% for the baseline. This condition rescues 29 failed task–run pairs and introduces 10 regressions. Aggregating the five settings $K \in \{8, 10, 12, 15, 20\}$ gives $5 \times 100 \times 3 = 1{,}500$ task–run–setting comparisons: 139 rescues and 86 regressions, or $+3.53$ percentage points. Note that the same 100 tasks and three baseline runs recur across settings, so this is a descriptive configuration average.

Table 10 reports the corresponding historical EOPS next-message selections. At $K = 8$, 13 rescues and 10 regressions give a $+1.25$-point change from the 40.00% baseline; $K = 2$ gives $+0.42$ points and $K = 4$ gives -1.25 points. Thus the recorded gains depend on candidate count in this protocol.

D.1 In-domain evaluation across discovery fractions

Discovery experience determines the workflow states and transitions available for guidance. To examine how much experience is needed, we define the discovery fraction p as the nominal share of discovery task IDs retained with their associated rollouts and compare success across these fractions. The smallest evaluated discovery collections already support useful guidance in both benchmarks. At nominal discovery fraction $p = 10\%$, the archived Qwen inventories contain 1,414 CRM rollouts and 2,170 EOPS rollouts (Table 4). Table 6 reports CRM success of 49.0–51.3% across $K \in \{4, 8, 10\}$, compared with its 46.7% baseline; EOPS reaches 41.3–43.3%, compared with 36.7%. These correspond to gains of $+2.3$ to $+4.6$ and $+4.6$ to $+6.6$ percentage points, respectively.

Benchmark	K	10%	20%	50%	100%
CRM ratio	4	50.0 ± 1.0	49.7 ± 2.1	50.7 ± 1.2	–
	8	51.3 ± 3.1	52.0 ± 2.0	48.0 ± 3.6	–
	10	49.0 ± 2.0	51.3 ± 3.5	49.0 ± 3.6	50.3 ± 0.6
EOPS ratio	4	43.3 ± 0.7	43.8 ± 1.3	42.5 ± 2.5	–
	8	41.7 ± 3.6	39.6 ± 2.6	40.0 ± 4.5	–
	10	41.3 ± 2.2	43.3 ± 0.7	41.3 ± 2.2	38.8 ± 2.5

Table 6: Historical in-domain Qwen task success (%, mean $\pm$ run standard deviation). The discovery-experiment baselines are 46.7% for CRM and 36.7% for EOPS. Ratios denote nominal fractions of discovery task IDs, retaining their associated rollouts; – denotes unavailable cells. The full-data endpoints use their reported 100% discovery labels.

Discovery fractions follow the definition in Appendix C. The positive means at $p = 10\%$ show that the full discovery corpus is unnecessary for useful guidance in these recorded configurations. Changing p also changes the learned state partition and retrieved evidence. The EOPS discovery/transfer baseline is 36.7%, whereas the separate next-message selection experiment in Table 10 uses a 40.0% baseline. Each result is compared with its own experimental baseline.

D.2 Cross-benchmark transfer evaluation

Related enterprise benchmarks can share workflow patterns despite differing tasks and tool interfaces. We test whether these patterns support reuse by exchanging the discovery graphs between CRM and EOPS. Historical cross-benchmark evaluation uses graphs constructed from EOPS trajectories to guide CRM tasks, and graphs constructed from CRM trajectories to guide EOPS tasks. The experimental specification holds the target policy, tasks, evaluator, and selector configuration fixed while changing graph source. Each direction includes discovery ratios of 20% and 100%, candidate counts $K \in \{4, 8\}$, and three runs on 100 CRM or 80 EOPS tasks. Table 7 compares source graphs at equal ratios and candidate counts where both cells are available, and Figure 2 places them against the corresponding target benchmark baseline. These comparisons characterize workflow reuse between CRM and EOPS.

Figure 2 in the main text displays both transfer directions; Table 7 retains every numerical comparison.

Every out-of-domain graph configuration exceeds its target's baseline, although three of the four available equal-ratio comparisons favor the in-domain graph. The highest EOPS rate with an out-of-domain graph, 43.8% at $100\%/K = 4$, matches the highest in-domain rate at $20\%/K = 4$. These are descriptive comparisons of the evaluated sampling-and-guidance configurations.

Target	Ratio / K	In-domain	Out-of-domain	vs. $K = 1$	vs. own
CRM	20% / 4	49.7 ± 2.1	48.3 ± 1.5	$+1.6$	-1.4
	20% / 8	52.0 ± 2.0	49.0 ± 4.0	$+2.3$	-3.0
	100% / 4	–	48.7 ± 6.1	$+2.0$	–
	100% / 8	–	49.7 ± 1.5	$+3.0$	–
EOPS	20% / 4	43.8 ± 1.3	41.2 ± 3.3	$+4.5$	-2.6
	20% / 8	39.6 ± 2.6	42.5 ± 3.3	$+5.8$	$+2.9$
	100% / 4	–	43.8 ± 1.3	$+7.1$	–
	100% / 8	–	40.8 ± 2.6	$+4.1$	–

Table 7: Historical Qwen cross-benchmark task success (%, mean $\pm$ run standard deviation). The final two columns give percentage-point differences from the target's $K = 1$ baseline (CRM: 46.7%; EOPS: 36.7%) and from its in-domain graph at the same ratio and K. Differences are computed from the displayed means; – denotes an unavailable comparison.

D.3 Cross-model evaluation and discovery-source comparison

Cross-model evaluation tests whether experience collected by one policy can guide another. We compare memories mined by the acting policy with those mined by a different policy, examining how discovery data and graph construction affect reuse.

Historical discovery grids are non-monotone (Table 6), and cross-benchmark graphs trail in-domain graphs in three of four populated equal-ratio/K comparisons (Table 7). The Gemma rows of Table 2 report the separate experiment with active GWM guidance, in which GWM-L gives higher success than GWM-S.

The historical Gemma executions in Table 8 used graph-based fallbacks after selector connection failures: CRM selected by the matched state's success value, while EOPS used graph action scores or skipped model-based scoring when candidates were identical. Appendix F separately reports Gemma guidance with functioning model-based selection and observed advice injections.

Target	Source	Discovery	Baseline	No advice	$K = 4$	$K = 8$
CRM	Qwen	20%	40.00	39.33	–	43.00
CRM	Qwen	Full graph	40.00	39.67	–	44.33
EOPS	Gemma	20%	27.50	28.75 (+1.25)	27.50 (+0.00)	28.33 (+0.83)
EOPS	Qwen	20%	27.50	30.00 (+2.50)	28.33 (+0.83)	29.17 (+1.67)
EOPS	Gemma	100%	27.50	27.92 (+0.42)	29.17 (+1.67)	27.50 (+0.00)

Table 8: Historical Gemma fallback execution: task success (%), averaged over three unseeded repeats on 100 CRM or 80 EOPS tasks. No advice denotes the configured $K = 1$ advice condition with zero delivered injections. The $K = 4/8$ columns use CRM state-value or EOPS action-score fallback after selector failures, with identical-candidate skips in EOPS; – denotes an unavailable cell. EOPS parentheses give percentage-point differences from the 27.50% baseline. The CRM baseline's repeat SD is 5 pp. Discovery percentages denote nominal task fractions; Full graph identifies the separately evaluated Qwen CRM build with 334 state clusters, distinct from the 346-state $p = 100\%$ inventory build. Source policy, corpus size, and construction also differ.

The historical CRM state-value selector at $K = 8$ exceeds the single-generation baseline by 3.00 and 4.33 percentage points with the 20% and full graphs. On EOPS, Qwen 20% fallback conditions exceed baseline success by 0.83 and 1.67 points at $K = 4$ and $K = 8$.

For the historical builds in this source-policy comparison, graph size denotes the number of learned state clusters, counted from centroid rows. The Gemma-derived EOPS graphs at $p = 20\%$ and $p = 100\%$ contain 8 and 69 clusters, respectively; the Qwen-derived EOPS graph at $p = 20\%$ contains 212. For CRM, the corresponding Gemma graphs contain 55 and 244 clusters, while the Qwen discovery-ratio inventories contain 93 and 346 (Table 4). The separate Qwen full CRM build

has 334 clusters. The CRM builders also differ in clustering granularity, so these comparisons jointly vary trajectory generation, corpus size, and graph construction.

The same Gemma CRM graph at $p = 100\%$ has 1,391 domain-indexed transition-table entries: 1,369 state entries across 22 domains plus 22 start entries.

D.4 Extended comparisons with related methods

We compare how related methods represent experience and use it for guidance; Table 9 summarizes the distinctions from GWM and the roles of the cited evaluation settings.

Representation of agent interaction. ReAct interleaves reasoning, actions and observations (Yao et al., 2023). Latent Action Reparameterization learns action-segment symbols and trains embeddings and adapters (Huang et al., 2026b). GWM retrieves evidence by embedding and clustering history prefixes, without updating policy parameters.

Workflow discovery and experience memory. Process mining recovers control flow and object-centric processes (Leemans et al., 2013; van der Aalst & Berti, 2020); dialogue research extracts workflows and predicts action branches (El Hattami et al., 2023; Choubey et al., 2025; Ramakrishnan et al., 2023). Agent Workflow Memory induces reusable routines (Wang et al., 2025); AutoGuide retrieves context-conditioned guidelines (Fu et al., 2024). ExpeL, Reflexion and Synapse reuse insights, feedback and abstracted trajectories, respectively (Zhao et al., 2024; Shinn et al., 2023; Zheng et al., 2024). ReasoningBank distills strategies from successes and failures; BAGEL refines demonstrations (Ouyang et al., 2026; Murty et al., 2024). GWM groups recurring history contexts into a state graph with alternative continuations and their outcome/support statistics.

Workflow guidance and adaptation. StateFlow uses authored state machines (Wu et al., 2024). Demonstration-derived workflows constrain reinforcement-learning exploration; workflow-conditioned scorers train dialogue policies (Liu et al., 2018; Min et al., 2023). Guideline priors and automatic evaluators support reranking, refinement and inference-time guidance (Wen et al., 2025; Pan et al., 2024); GEPA evolves prompts by reflecting on traces (Agrawal et al., 2026).

Graph memory and world models. AriGraph combines semantic and episodic graphs; Graph RAG uses entity graphs and community summaries (Anokhin et al., 2025; Edge et al., 2024). Other world models learn landmark reachability, action-conditioned knowledge-graph changes or finite-state dynamics (Zhang et al., 2021; Ammanabrolu & Riedl, 2021; Hernandez Cano et al., 2025). World Knowledge Model supplies state-conditioned action knowledge; Web Agents with World Models predicts candidate-action observation changes (Qiao et al., 2024; Chae et al., 2025). GWM indexes observed continuations, associating transitions with observed actions or, in aligned variants, the next action (Section 4). World-In-World evaluates visual world models by embodied task success, distinguishing predictive quality from decision utility (Zhang et al., 2026).

Enterprise evaluation and variation. WorkArena++ and SCUBA assess UI workflows (Boisvert et al., 2024; Dai et al., 2026); AppWorld provides multi-application API tasks and TheAgentCompany simulates workplace activities (Trivedi et al., 2024; Xu et al., 2025). τ-bench tests tool–user interaction (Yao et al., 2024); τ^2-Bench adds shared-state coordination (Barres et al., 2025). OpenApps varies app appearance and content; Gaia2 adds asynchronous changes and action-level verification (Ullrich et al., 2026; Froger et al., 2026).

Table 9: Related methods and evaluation settings: compact feature summary. Cells describe representative designs, not measured superiority.

Method	Evidence	Reusable object	Structure	Updated component	Main use
ReAct [1]	Live interaction	Reasoning/action trace	Interaction sequence	Prompting	Acting
Latent Action Reparameterization [2]	Agent rollouts	Action symbols	Compressed sequence	Embeddings, LoRA	Efficient inference
Process discovery [3,4]	Event logs	Process model	Tree / Petri net	Process discovery	Control-flow analysis
Dialogue workflow discovery [5,6]	Dialogues	Action workflow	Action sequences	Model / prompting	Workflow extraction
Multi-step dialogue prediction [7]	Dialogue histories	Action continuations	Sequence / graph	Tuning / ICL / traversal	Action prediction
Agent Workflow Memory [8]	Agent experience	Reusable routines	Workflow memory	Memory induction	Contextual reuse
AutoGuide [9]	Offline experience	Conditional guidelines	Guideline memory	Guideline induction	Contextual guidance
ExpeL / Reflexion / ReasoningBank [10,11,12]	Experience, feedback	Insights / strategies	Text memory	Memory update	Contextual reuse
Synapse / BAGEL [13,14]	Trajectories	Abstracted / refined examples	Exemplar memory	Abstraction / refinement	Example retrieval
StateFlow [15]	Designed procedures	Task states	State machine	Controller design	Workflow execution
Workflow-guided RL [16,17]	Demonstrations / workflows	Constraints / compliance	Workflows	Policy training	Exploration / responses
GuidedTOD / automatic evaluators [18,19]	Guidelines / trajectories	Prior / evaluation	Prior / text–vision	Prior / policy refinement	Reranking / refinement
AgentTuning / GEPA [20,21]	Traces, instructions	Model / prompt	Parameters / text	Weights / prompts	Policy adaptation
Graph World Model (Feng et al.) [22]	Objects, actions	Node representations	Multimodal graph	T: decoders; E: projector	Prediction / planning
AriGraph / GraphRAG [23,24]	Observations / documents	Memories / summaries	Knowledge graph	Memory / index	Planning / QA
Latent landmark planning [25]	Goals, transitions	Landmarks, reachability	Reachability graph	Landmarks, Q-functions	Planning
Knowledge-graph world models [26]	Observations, actions	Graph changes, actions	Knowledge graph	Dynamics model	Prediction / actions
Neurosymbolic world models [27]	Environment data	Dynamics regions	Finite-state model	World model, policy	Policy optimization
WKM / web world models [28,29]	Traces, transitions	Knowledge / observation changes	Text representations	Knowledge / dynamics model	Planning / selection
GWM (this work)	Tool-use histories, outcomes	Prefix states, outcome/support statistics, examples	Directed state-transition graph	Offline memory construction; policy weights fixed	State retrieval, advice, candidate selection

Evaluation setting	Scope	Comparison dimension
WorkArena++ / SCUBA [30,31]	Enterprise UI / Salesforce CRM	Workflow evaluation
AppWorld / TheAgentCompany [32,33]	Multi-app APIs / workplace tasks	Application breadth
τ-bench / τ^2-Bench [34,35]	Tool–user interaction / shared control	Rules, coordination
OpenApps / Gaia2 [36,37]	App variation / asynchronous events	Robustness, verification
World-In-World [38]	Embodied world models	Closed-loop utility
AstaBench [39]	Scientific-research agents	Controlled evaluation
CRMArena-Pro (used in this study) [40]	CRM workflows	Task success; within-/cross-benchmark and cross-model reuse
EnterpriseOps-Gym (used in this study) [41]	Enterprise operations	Task success; within-/cross-benchmark reuse

ICL: in-context learning. T/E: Feng's token/embedding variants; E freezes the LLM. LAR freezes the pretrained backbone. Slashes distinguish methods or variants. Workflow/exemplar controls are adapted, not full reproductions; benchmark rows give evaluation context.

Sources (superscript key). 1 Yao et al., 2023; 2 Huang et al., 2026b; 3 Leemans et al., 2013; 4 van der Aalst & Berti, 2020; 5 El Hattami et al., 2023; 6 Choubey et al., 2025; 7 Ramakrishnan et al., 2023; 8 Wang et al., 2025; 9 Fu et al., 2024; 10 Zhao et al., 2024; 11 Shinn et al., 2023; 12 Ouyang et al., 2026; 13 Zheng et al., 2024; 14 Murty et al., 2024; 15 Wu et al., 2024; 16 Liu et al., 2018; 17 Min et al., 2023; 18 Wen et al., 2025; 19 Pan et al., 2024; 20 Zeng et al., 2024; 21 Agrawal et al., 2026; 22 Feng et al., 2025; 23 Anokhin et al., 2025; 24 Edge et al., 2024; 25 Zhang et al., 2021; 26 Ammanabrolu & Riedl, 2021; 27 Hernandez Cano et al., 2025; 28 Qiao et al., 2024; 29 Chae et al., 2025; 30 Boisvert et al., 2024; 31 Dai et al., 2026; 32 Trivedi et al., 2024; 33 Xu et al., 2025; 34 Yao et al., 2024; 35 Barres et al., 2025; 36 Ullrich et al., 2026; 37 Froger et al., 2026; 38 Zhang et al., 2026; 39 Bragg et al., 2026; 40 Huang et al., 2026a; 41 Malay et al., 2026.

E RETROSPECTIVE ANALYSIS OF RECORDED EXECUTIONS

To understand the task-level gains from guidance and the information retained by its selection scores and workflow states, we retrospectively analyze recorded task outcomes, candidate-answer pools, and reconstructed graph statistics. The first supports paired historical comparisons; the second uses a local reward checker; the third measures sensitivity of a reconstruction. These retrospective comparisons describe historical evaluation configurations.

CRM experiments A and B are separate sets of whole-episode executions; experiment B additionally records candidate answers and scores. Both are compared with the same experiment A baseline executions. The cross-experiment comparison and pooled estimate below are descriptive sensitivity analyses of these historical configurations.

E.1 EXPERIMENTAL CONFIGURATIONS AND SCOPE

Source inspection identifies whole-episode scoring in CRM experiments A and B and next-message scoring in the EOPS selection experiment. The CRM selector scores each finished flow separately; the EOPS selector receives multiple next-message candidates before execution. The recorded CRM selection runs include both advice injections and whole-episode scoring: advice supplies context to the acting policy, while selection scores completed candidate episodes.

E.2 PAIRED OUTCOME ANALYSIS AND BASELINE VARIABILITY

Aggregate success rates can conceal which tasks benefit from guidance and which regress. We use paired task outcomes to quantify these changes and compare them with variability across baseline repeats. Figure 3 summarizes these paired comparisons; Figure 4 presents the separate online guidance and replay experiments.

For task j and repeat s, let Y_{js}^G and Y_{js}^b equal one for recorded success under guidance and its baseline, respectively, and zero otherwise. Their difference $d_{js} = Y_{js}^G - Y_{js}^b$ is $+1$ when only guidance succeeds (a rescue), -1 when only the baseline succeeds (a regression), and zero when both succeed or both fail. Over n tasks and S repeats, the success-rate difference is

$$\Delta_{\text{pp}} = \frac{100}{nS} \sum_{j=1}^{n} \sum_{s=1}^{S} d_{js} = \frac{100(\text{rescues} - \text{regressions})}{nS}.$$

Each matched task–repeat comparison contributes once; the factor of 100 expresses the net difference in percentage points. We extract binary CRM success and the EOPS selection experiment's indicator of positive recorded score, joining the same task IDs within each experiment and repeat label. The source package records the selected detail-file hashes and normalized outcomes. For task j, let $\bar{d}_j$ be the mean of its recorded treatment-minus-baseline differences. Each bootstrap sample draws n task IDs with replacement and retains all comparisons belonging to a sampled task; it then averages the resulting differences. We use sorted task IDs, 20,000 resamples, pseudorandom seed 199, and percentile endpoints. The sampling unit is the task, not the task–repeat pair. The intervals condition on the observed repetitions; however, note that they do not model an additional population of serving runs, adjust for selecting configurations on these tasks, or establish equivalence.

For baseline turnover, we compare each of the three unordered pairs of repeats. CRM gives 9, 12, and 5 switches, totaling 26/300 comparisons; EOPS gives 8, 6, and 8, totaling 22/240. These pairs share executions and are a descriptive reference rather than independent estimates of a noise floor. CRM advice has 13 rescues and 13 regressions in the corresponding treatment–baseline pairs, equal to the aggregate baseline turnover.

For $K = 10$, experiment A records 29 rescues and 10 regressions, while experiment B records 28 and 22 across 300 comparisons each. The smaller experiment B net gain therefore accompanies nearly unchanged rescue counts and more regressions.

The pooled CRM $K = 10$ estimate averages six treatment executions against the same three experiment A baseline executions, which are reused by both experiments. It contains 600 comparison rows but only 100 distinct tasks. We retain this dependence within each task's bootstrap cluster. Among

Condition	Repeat success (%) Mean (%)	Rescue / regress	Δ pp [95% CI]
CRM initial agent	25/26/25 25.33	–	–
CRM baseline	47/44/47 46.00 ± 1.7	–	–
CRM advice	46/43/49 46.00 ± 3.0	13 / 13	$0.00\ [-3.33, +3.00]$
CRM experiment A $K = 4$	51/50/40 47.00 ± 6.1	22 / 19	$+1.00\ [-3.33, +5.33]$
CRM experiment A $K = 8$	48/47/50 48.33 ± 1.5	21 / 14	$+2.33\ [-2.33, +7.00]$
CRM experiment A $K = 10$	53/52/52 52.33 ± 0.6	29 / 10	$+6.33\ [+1.33, +11.33]$
CRM experiment A $K = 12$	48/48/47 47.67	25 / 20	$+1.67\ [-3.67, +7.00]$
CRM experiment A $K = 15$	53/48/53 51.33	32 / 16	$+5.33\ [+0.33, +10.67]$
CRM experiment A $K = 20$	47/49/48 48.00	32 / 26	$+2.00\ [-3.67, +7.67]$
CRM experiment B $K = 10$	46/49/49 48.00 ± 1.7	28 / 22	$+2.00\ [-3.67, +7.67]$
CRM experiment B $K = 12$	49/49/50 49.33 ± 0.6	25 / 15	$+3.33\ [-2.00, +8.67]$
CRM $K = 10$, both experiments	six treatment runs	–	$+4.17\ [-0.83, +9.17]$
EOPS baseline	40/40/40 40.00	–	–
EOPS $K = 2$	37.5/43.75/40 40.42	12 / 11	$+0.42\ [-2.92, +3.75]$
EOPS $K = 4$	33.75/40/42.5 38.75	13 / 16	$-1.25\ [-5.42, +3.33]$
EOPS $K = 8$	37.5/43.75/42.5 41.25	13 / 10	$+1.25\ [-2.92, +5.83]$

Table 10: Historical task outcomes and task-cluster reanalysis. Repeat order is 42/43/44, without assuming these labels establish sampling-seed propagation. Success cells give repeat percentages above the mean; $\pm$ is the previously reported sample SD of CRM run averages. EOPS success counts for baseline and $K = 2/4/8$ are 32/32/32, 30/35/32, 27/32/34, and 30/35/34. CRM experiments A and B use the experiment A baseline; the pooled row reuses its three executions for both experiments. All CRM rows have 100 unique tasks and EOPS rows 80. Brackets are 95% task-cluster percentile intervals from 20,000 resamples, exploratory and conditional on the recorded runs.

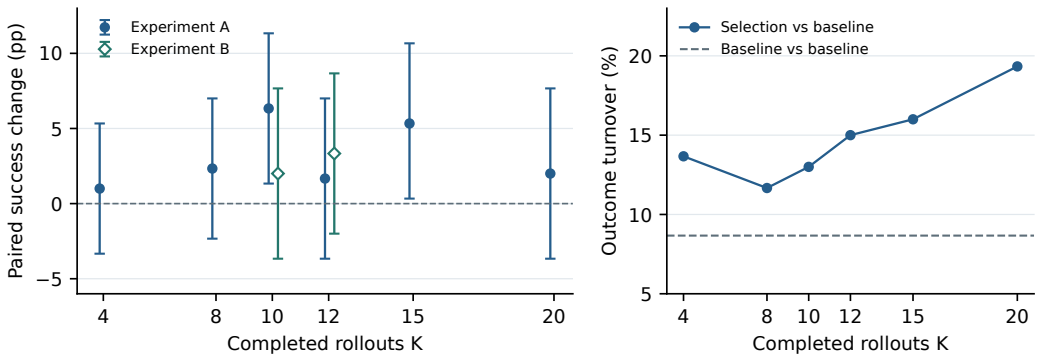


Figure 3: Historical CRM experiments A and B: whole-episode selection on 100 recurring tasks. Left: paired success differences from the experiment A baseline, with 95% task-cluster percentile intervals; experiment B points are offset horizontally. Right: experiment A outcome turnover against baseline, with pairwise baseline reruns as a descriptive reference. The baseline comparisons share executions. Experiment A at $K = 10$ yields 29 rescues and 10 regressions, a net gain of 19 successful task executions and 6.33 percentage points over baseline.

46 tasks unsuccessful in all three baseline executions, nine succeed in at least one experiment A execution.

E.3 Candidate-answer replay and selection analysis

To separate successful-candidate availability from selection quality, we analyze the recorded CRM candidate pools. Table 11 shows answer-based replay success of 48.00% at $K = 10$ and 49.33% at $K = 12$, exceeding expected uniform choice by 4.10 and 5.06 percentage points, respectively. The six stored CRM files contain 600 task–repeat pools and 6,600 final candidate answers with selector preference scores. The local reward checker reimplements each task's answer-reward rule using its reference answer and reward metric; a positive reward gives an answer-based success label. Labels are computed independently of selector scores. Replay applies the recorded score ordering to these final-answer labels. Table 20 reports the separate fixed-candidate evidence comparisons.

Measure	$K = 10$	$K = 12$
Task–repeat pools / candidate answers	300 / 3,000	300 / 3,600
At least one successful candidate	186/300	194/300
First-maximum replay succeeds	144/300	148/300
Replay succeeds given availability	144/186 (77.4%)	148/194 (76.3%)
Top-score tie given availability	132/186 (71.0%)	149/194 (76.8%)
Successful candidate left unselected	42/300	46/300
Within-pool score concordance	0.6205	0.6152
Uniform random success (%)	43.90	44.28
Selector success (%)	48.00	49.33
Oracle success (%)	62.00	64.67
Selector minus random (pp)	+4.10	+5.06
Oracle minus selector (pp)	14.00	15.33

Table 11: Answer-based analysis of CRM experiment B's stored pools: success means positive reward from the local reward checker. Each K uses the same 100 tasks over three repeats. Replay selects the first maximum recorded score, matching the inspected selection tie rule. Concordance pools correct–incorrect candidate pairs within each pool, scores a tied pair as $1/2$, and weights pools by their number of such pairs. Random is expected uniform choice within each fixed-size pool; oracle denotes at least one successful candidate. Percentage-point differences use integer counts before rounding.

Index the N candidate pools by i and the K candidates in each pool by k. The label y_{ik} records whether candidate k succeeds under the local reward checker, and $\hat{k}_i$ is the index of the candidate selected by the first-maximum score rule. Let $O_i = \max_{1 \le k \le K} y_{ik}$ indicate whether pool i contains an answer labeled successful, and let $J_i = y_{i,\hat{k}_i}$ denote the label at the first maximum score in pool i. Bars denote means across pools. For the binary answer-based labels $y_{ik} \in \{0, 1\}$, the empirical failure rate over N pools satisfies the exact identity

$$1 - \overline{J} = \underbrace{1 - \overline{O}}_{\text{no successful candidate}} + \underbrace{\overline{O} - \overline{J}}_{\text{successful candidate left unselected}} . \tag{10}$$

The left side is the observed replay failure rate. The first term is the fraction of pools with no successful answer; the second is the fraction with a successful answer available but left unselected. These disjoint cases account for every replay failure. At $K = 10$, this is $156/300 = 114/300 + 42/300$; at $K = 12$, $152/300 = 106/300 + 46/300$. These are observed-label counterparts of the coverage/selection distinction in Equation (9).

Under the reported fixed-K pool sizes, uniform choice has expected answer-based success $\overline{U} = (NK)^{-1} \sum_{i,k} y_{ik}$. The sum counts successful answers and NK counts all candidates. With equal-size pools, this fraction is also the average probability of success when choosing one candidate uniformly within each pool. The archived candidate counts give $1{,}317/3{,}000$ at $K = 10$ and $1{,}594/3{,}600$ at $K = 12$. Subtracting these from $144/300$ and $148/300$ yields replay advantages of 4.10 and 5.06 pp (Table 11). This equal-size calculation agrees with the archived per-pool means at their reported precision. The contrast compares recorded ranking with uniform choice conditional on the reported sizes, rather than with selection without graph evidence or a single-generation agent. The two K settings use separately generated pools, not nested sets.

Top-score ties can include duplicate answers or only successful candidates. Likewise, 572/1,161 candidates with a rounded score of 0.1 and 117/121 with score 1.0 are labeled successful at $K = 10$. The higher success frequency in the score-1.0 group, together with within-pool concordance above 0.5, shows that the recorded scores distinguish successful from unsuccessful answers in these pools, consistent with their selection advantage over uniform choice.

E.4 Action-label sensitivity of reconstructed successor distributions

Action labels determine which observed transitions contribute to each state–action successor distribution. We examine how aligning labels with the next observed action changes these empirical distributions.

We compare two action-label conventions on CRM and EOPS analysis inputs reported under nominal $p = 20\%$ labels. The EOPS input contains 2,170 distinct rollout identifiers, whereas Table 4 lists 4,013 train-tagged rollouts for EOPS $p = 20\%$. The following analyses use the stated prefix-row populations without a split filter and are reported separately from the discovery-graph inventory.

The reconstruction orders each rollout's records by step index and forms $(\mathrm{START}, z_1, \ldots, z_n, \mathrm{END})$. Here z_t is the stored cluster label of prefix t, n is the number of prefixes in the rollout, and START and END mark its boundaries. On each outgoing edge from a prefix, it compares the sorted tool-name list in that prefix with the list in the following record. The original label thus refers to an action already included in the source prefix, whereas the shifted label refers to the next observed action for inter-prefix transitions. The audit excludes edges out of START and includes the final edge to END, whose shifted label is the empty action label. Consequently, the reported changes include terminal relabelings. The aggregate counts provide a lower bound on the inter-prefix change. With N outgoing occurrences, R rollouts, and C changed labels, there are $N - R$ inter-prefix occurrences and at most R changed terminal edges. Hence the fraction of changed inter-prefix labels is at least $\max(0, C - R)/(N - R)$: 28.29% for CRM and 47.67% for EOPS. Subtracting R from C allows one changed terminal label per rollout; any remaining changes must occur between prefixes.

We compare the original and shifted empirical successor distributions using total variation (TV) distance: zero means that the distributions agree, and larger values indicate greater differences in their probability mass.

Statistic	CRM input	EOPS input
Prefix rows	18,818	57,259
Distinct rollout identifiers	2,850	2,170
Audited outgoing edge occurrences	18,818	57,259
Occurrences with a changed action label	7,367 (39.15%)	28,432 (49.66%)
Inter-prefix occurrences (terminal edges excluded)	15,968	55,089
Lower bound on changed inter-prefix labels	4,517 (28.29%)	26,262 (47.67%)
Distinct sorted tool-name lists	4	794
State–action keys, original / shifted	263 / 273	3,755 / 4,404
Shared state–action keys	249	3,023
Mean successor-distribution TV on shared keys	0.2387	0.4224
Shared keys with TV > 0.2	104 / 249	2,154 / 3,023

Table 12: Offline label sensitivity for the two nominal $p = 20\%$ analysis inputs. Counts concern edge occurrences, excluding START and including each rollout's terminal edge. Successor frequencies count visits. Total variation (TV) distance compares the original and shifted normalized counts and is averaged equally over shared keys; keys present in only one reconstruction are excluded. The inter-prefix lower bound allows every terminal edge to change; it is not an exact interior count.

The reconstruction also differs from the reference graph-construction procedure in two respects. It uses sorted tool-name lists without deduplication, while the reference action label discards multiplicity. It pools the single `noise` label across domains, while the reference procedure aggregates within each domain. Table 12 therefore measures sensitivity of this offline reconstruction, rather than an exact change to every graph used during evaluation. The measured quantities are action-label changes and successor-distribution TV, not changes in selected actions or task success. Appendix F.5 separately evaluates an action-aligned CRM construction and its downstream guidance.

E.5 Prefix-level associations of cluster labels

Workflow states group trajectory prefixes into reusable units of experience. We characterize these groups by measuring their associations with recorded task, action, and outcome labels.

The same analysis inputs, with the population qualifications above, permit a descriptive comparison of stored cluster labels with domain, recorded tool names, observed-result category, rollout success, and task identity. The audit computes $\mathrm{NMI}(C, L) = I(C; L)/\sqrt{H(C)H(L)}$ from empirical row frequencies. Here C denotes the cluster label and L the comparison label; $I(C; L)$ is their empirical mutual information, and $H(C)$ and $H(L)$ are their empirical entropies. The numerator measures shared information, while the denominator normalizes it by the variability of the labels. For nonconstant labels, larger NMI indicates stronger empirical association. NMI includes noise rows. For the tool-name comparison, the label is the recorded sequence, retaining order and multiplicity; it is not the deduplicated action abstraction ψ. The observed-result categories are `ok`, `error`, and `none`.

Row-level statistic	CRM input	EOPS input
Non-noise cluster labels	93	212
Additional pooled noise label	1	1
Fraction of rows labeled noise	5.64%	17.19%
NMI: domain	0.8185	0.5493
NMI: observed-result category	0.1181	0.0868
NMI: recorded tool-name sequence	0.1375	0.4926
NMI: rollout success	0.0732	0.0695
NMI: task ID	0.6048	0.5989
Majority-result purity, excluding noise	0.7724	0.8253
Non-noise clusters with multiple result categories	91 / 93	182 / 212

Table 13: Associations over all stored prefix rows, without a split filter. Purity is the fraction of non-noise rows assigned their cluster's empirical majority result category. The final row counts clusters containing more than one of `ok`, `error`, and `none`; it does not specifically count clusters mixing `ok` and `error`.

Result-category and rollout-success NMI are numerically lower than domain NMI in both files. Within-domain construction incorporates the domain into non-noise cluster labels, so domain association partly follows from construction. A useful pre-action state can lead to both success and failure under different actions; low association with terminal success alone does not refute its usefulness. Cluster labels retain an empirical association with recorded tool-use patterns, particularly in EOPS (NMI 0.4926; CRM 0.1375), indicating structure in the observed action context.

E.6 Information stored in the graph and used by GWM

Graph statistics and model-based guidance draw on different views of a trajectory and its candidate actions. We trace the information available to each component to clarify where argument details are retained or removed.

Arguments are lost at different stages of the reference processing procedure (Table 14). The graph's action key removes them explicitly. State embeddings can include argument text, while GWM selection uses additional candidate and history information.

Component	Information retained	Limitation
Action key	Sorted, deduplicated tool names	Arguments, within-step call order and multiplicity are absent; supported toolset scores coincide for such variants at the same state.
Workflow state	Embedded recent rendered history, including remaining arguments and tool results	Field removal, text windows and clustering can erase or merge meaningful distinctions.
GWM selection	Task, history, tool information, graph evidence and candidate arguments	Rendered views are capped. Choices can use information absent from graph statistics.

Table 14: Source-level information paths through distinct components of the reference implementation. GWM selection is not restricted to the graph's action key.

At one history, sharing a document with an authorized versus unauthorized recipient can require different decisions even when both proposals use the same tool. Their tool-name keys coincide, so a table indexed only by the current state and that key cannot represent the distinction. Improving the history encoder alone leaves this same-history action collision intact. A candidate-appended embedding fallback or a selector can use further information.

Preprocessing can also remove information before clustering. The reference procedure converts arguments to text after optional field removal and normalization; graph construction uses benchmark-specific preprocessing settings unless overridden.

Removing incidental identifiers can support reuse, but preserving entity relationships requires retaining which references denote the same object or actor. Dates, access roles and classification can also be decision-relevant when legitimately observed.

E.7 CRM execution cost

Inference-time guidance combines additional candidate generation with selection and advice. We quantify the recorded elapsed-time overhead and guidance-token usage of the CRM configurations. In enterprise settings, improved task success can justify a larger inference budget; we therefore treat additional computation time as a resource for improving fixed-policy agent behavior and report its observed cost.

The recorded experiment A $K = 10$ elapsed time is 11.62 times the baseline. This is an observed wall-time ratio across historical runs, not a hardware-normalized compute ratio. Guidance token totals cover selection and advice only. Experiment A $K = 10$ averages 41,100 seconds per 100-task CRM run versus 3,538 for its baseline, with 1.37 million selector+advice tokens per run; candidate generation and construction are excluded from this token total. For $K > 1$, the matched majority-vote and random-selection comparisons in Table 2(a) reuse the same generated candidate pools as GWM, sharing their candidate-generation token budget and computation. Selection and advice contribute additional overhead.

CRM condition	Elapsed (s)	Guidance tokens	Success (%)	q_a
Baseline	3,538	0	46.00	46.80
Advice	4,077	16,516	46.00	40.62
Experiment A $K = 4$	16,813	548,516	47.00	10.06
Experiment A $K = 8$	32,656	1,079,845	48.33	5.33
Experiment A $K = 10$	41,100	1,365,609	52.33	4.58

Table 15: Observed mean costs per 100-task CRM run across three recorded repeats, showing the baseline, advice, and experiment A selection configurations at $K \in \{4, 8, 10\}$. Tokens cover selection and advice, excluding complete candidate-generation and graph-construction usage. Elapsed times are historical measurements, without hardware or concurrency normalization. q_a counts successful task-executions per recorded run-hour, using summed successes and unrounded run durations across repeats.

For condition a, let S_{ar} count successful task-executions and T_{ar} be recorded elapsed seconds in repeat r. We report $q_a = 3600 \sum_r S_{ar} / \sum_r T_{ar}$, the ratio of sums across repeats, in successful executions per recorded run-hour. The numerator totals successes, the denominator totals recorded run durations, and the factor 3600 converts successes per second to successes per hour. Repeated successes on the same task count separately. Experiment A $K = 10$ achieves 4.58 versus the baseline's 46.80. Let b denote the experiment A baseline. The incremental ratio $3600(\sum_r S_{ar} - \sum_r S_{br})/(\sum_r T_{ar} - \sum_r T_{br})$ divides net additional successes by additional recorded time and converts the result to successes per additional run-hour. It is 0.61 for experiment A $K = 10$ (19 net additional successes over 31.30 additional run-hours). Within the displayed experiment A configurations, observed success rises from 47.00% at $K = 4$ to 52.33% at $K = 10$; the latter yields 19 additional successful task executions over baseline across the three repeats.

F Online guidance and fixed-candidate evaluation

Workflow guidance can influence both the candidates an acting policy produces and the selector's final choice; we assess online task success and compare historical evidence with candidates and the model used for selection held fixed. The online experiments use three executions of the same 100 CRM or 80 EOPS tasks. The fixed-candidate replay experiments use pre-existing CRM rollouts outside the graph's discovery tasks; their accuracy measures selection from those pools. The matched controls in Appendix F.6 instead reuse the online CRM pools; the separate history replay in Appendix F.7 uses a shared 600-task cohort for both judges. Appendix F.8 compares executed EOPS selection policies and selection on graph-free CRM pools. These populations and outcomes are distinct.

F.1 Online guidance with GWM-S and GWM-L

Online guidance can affect both task execution and candidate selection; we evaluate the resulting task success with GWM-S and GWM-L. Table 16 reports the online configuration comparisons; Table 17 details the paired Gemma contrasts and intervals.

Effective intervention. The effective intervention depends on which guidance mechanisms run successfully; we identify the selection and fallback behavior in the recorded executions. Historical Gemma executions used an unavailable model endpoint for advice and selection. CRM selection consequently ranked by graph state value; EOPS used graph action scores or skipped identical candidates.

Recorded Gemma success is higher with GWM-S and GWM-L, and highest with GWM-L. GWM-L reaches 52.33% mean success against the 40.00% stored baseline: $+12.33$ points $[+6.67, +18.00]$. GWM-S reaches 47.33%: $+7.33$ points $[+1.33, +13.33]$. GWM-L exceeds GWM-S by $+5.00$ points $[+1.33, +9.00]$ (Table 17).

Why the CRM means coincide. The means coincide because each configuration records 157 successes in 300 task executions, although the successful tasks differ. Raw-record checks reproduce

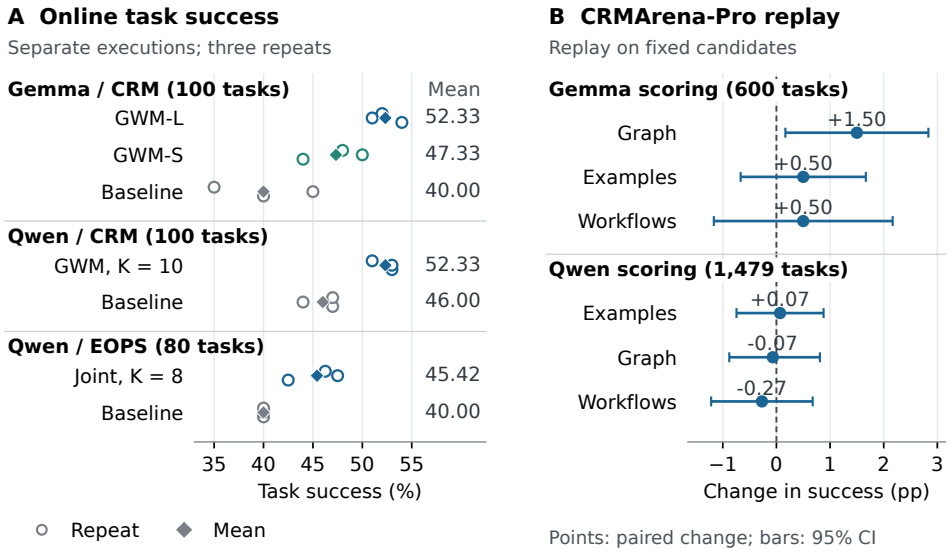


Figure 4: Online guidance results and fixed-candidate evidence comparisons. Within each group, larger means or paired changes appear higher. **A:** Circles show three executions (repeat labels 42/43/44); diamonds and labels show mean task success, alongside each configuration's stored historical baseline. Panel A model names identify the acting policies. For Gemma CRM at $K = 8$, GWM-S (small memory) and GWM-L (large memory) use 1,573 and 14,660 Qwen-generated CRM discovery rollouts, respectively; a rollout is one recorded task attempt. CRM selects executed rollouts with advice; EOPS jointly assesses candidates for the next assistant message. **B:** Historical CRM replay. Points and bars show paired changes and reported exploratory, unadjusted 95% task-bootstrap intervals relative to the same language model selecting without historical evidence. Panel B names identify which model scores the stored Qwen candidates, fixed within each model's distinct population (Gemma: 600 tasks; Qwen: 1,479).

Configuration	Repeat success (%)	Mean (%)	Δ pp [95% CI]
Gemma baseline	40/45/35	40.00	—
Gemma $K = 8$, GWM-L	54/51/52	52.33	$+12.33$ $[+6.67, +18.00]$
Gemma $K = 8$, GWM-S	44/50/48	47.33	$+7.33$ $[+1.33, +13.33]$
Qwen $K = 10$, separate GWM	53/53/51	52.33	$+6.33$ $[+1.33, +11.33]$
Qwen $K = 10$, aligned GWM	52/55/50	52.33	$+6.33$ $[+2.00, +11.33]$
Qwen $K = 10$, majority vote	48/49/51	49.33	$+3.33$ $[-0.33, +7.33]$
EOPS $K = 8$, joint selection	42.5/47.5/46.25	45.42	$+5.42$ $[+1.67, +9.58]$

Table 16: Online configuration comparisons. CRM uses 100 tasks and EOPS 80, each with repeat labels 42/43/44. Differences use the corresponding stored baseline means: 40.00% for Gemma, 46.00% for Qwen CRM, and 40.00% for EOPS. GWM-S (small memory) and GWM-L (large memory) use 1,573 and 14,660 Qwen-generated CRM discovery rollouts, respectively. Gemma intervals use 20,000 resamples of the 100 task clusters, preserving their three repeats; other intervals use 4,000 resamples.

the three Qwen success counts, 53/53/51, and Gemma GWM-L counts, 54/51/52, each on the same 100 tasks per repeat. Both sum to 157/300 and therefore round to 52.33%. Their baselines differ: Qwen records 47/44/47 successes (138/300), and Gemma records 40/45/35 (120/300). All runs have distinct raw detail files and 100 unique task IDs; success is positive recorded CRM reward. Aligning task IDs and nominal repeat labels yields 117 successes shared by both guided policies, 40 Qwen-only successes, 40 Gemma-only successes and 103 shared failures. Thus 80/300 outcomes differ despite equal totals.

Paired uncertainty and outcome scope. To quantify uncertainty in the recorded Gemma success differences, Table 17 reports paired comparisons between GWM-S, GWM-L and the baseline. All comparisons contain the same 100 tasks and three repeat labels, without missing or excluded records. Recorded success counts positive CRM reward. We average paired differences within task, then resample the 100 sorted task IDs 20,000 times, retaining all repeats. We generate bootstrap draws with NumPy's PCG64 (a permuted congruential pseudorandom number generator), initialized with seed 199; draws are shared across contrasts, and intervals use linear 2.5th/97.5th percentiles; exact empirical-bootstrap convolution confirms the endpoints. These exploratory, unadjusted intervals condition on recorded executions.

Comparison	Rescues	Regressions	Δ pp [95% CI]
GWM-S − baseline	47	25	$+7.33$ $[+1.33, +13.33]$
GWM-L − baseline	50	13	$+12.33$ $[+6.67, +18.00]$
GWM-L − GWM-S	31	16	$+5.00$ $[+1.33, +9.00]$
GWM-L − historical fallback	40	16	$+8.00$ $[+2.67, +13.67]$

Table 17: Paired Gemma CRM outcomes. GWM-S/GWM-L use 1,573/14,660 Qwen-generated discovery rollouts and guide $K = 8$ whole-episode selection with advice. Rescue/regression counts cover 300 task-repeat pairs; intervals resample 100 task clusters 20,000 times, retaining all three repeats. Success includes positive fractional CRM reward.

Sensitivity to the recorded outcome definition. Table 18 separates positive CRM reward, full CRM reward, the archived original-evaluator output, and mean CRM reward. Positive and full CRM reward both yield a $+12.33$ pp GWM-L–baseline difference; the $+8.00$ pp result uses a distinct archived evaluator output. The GWM-L–baseline difference is positive under all four recorded metrics, and each paired interval excludes zero (Table 18), supporting the direction of the gain across these outcome definitions.

Condition	CRM > 0	CRM $= 1$	Original $= 1$	Mean CRM
Baseline	120 (40.00%)	116 (38.67%)	105 (35.00%)	0.3909
GWM-S	142 (47.33%)	139 (46.33%)	119 (39.67%)	0.4652
GWM-L	157 (52.33%)	153 (51.00%)	129 (43.00%)	0.5128

Metric	GWM-L − baseline [95% CI]
Positive CRM (pp)	$+12.33$ $[+6.67, +18.00]$
Full CRM (pp)	$+12.33$ $[+6.67, +18.00]$
Original reward $= 1$ (pp)	$+8.00$ $[+3.00, +13.33]$
Mean CRM (reward)	$+0.1219$ $[+0.0652, +0.1789]$

Table 18: Metric sensitivity for the recorded Gemma CRM configurations. GWM-S/GWM-L use 1,573/14,660 Qwen-generated discovery rollouts. Counts and percentages cover 300 observations per condition (100 tasks, three repeats). CRM > 0 is the reported success metric; CRM $= 1$ requires full reward under the same evaluator. Original reward is a separate archived evaluator output. Mean CRM remains in reward units; binary differences use percentage points (pp). Intervals resample 100 paired task clusters 20,000 times, retaining all repeats.

Historical fallback comparison. GWM-L with functioning selection and advice exceeds the historical graph-heuristic fallback by $+8.00$ points $[+2.67, +13.67]$.

Qwen guidance and a graph-free comparator. The Qwen CRM whole-episode selection configuration reaches 52.33% against its stored 46.00% baseline. It uses a different clustering of the discovery corpus and uses a construction distinct from CRM experiments A and B. EOPS next-message selection with the joint selector reaches 45.42%, compared with its historical 40.00% baseline. Graph-free majority voting at $K = 10$ reaches 49.33% on CRM. This online condition generated independent candidates; the 50.33% matched-majority result in Table 2 uses the GWM pools. These recorded GWM configurations exceed their corresponding stored baselines by 6.33

Table 19: Recorded success on the 204-task CRM validation split, with three repeat labels (612 executions per condition). Candidates are executed rollouts. All multi-candidate conditions use $K = 10$; GWM guidance combines advice during generation with graph-informed selection.

Condition	Successes/executions	Success (%)
Single generation	255/612	41.67
Majority vote, $K = 10$	279/612	45.59
GWM advice and selection, $K = 10$	294/612	48.04
Graph-free selection, $K = 10$	299/612	48.86

percentage points on CRM and 5.42 points on EOPS; the graph-free CRM majority comparator also improves observed success by 3.33 points (Table 16).

F.2 Validation-split selection and outcome diagnostics

Candidate generation and evidence-based selection can both change task success. We therefore evaluate the complete guidance configuration alongside graph-free selection on a separate CRM validation split, then examine whether node statistics contain information about eventual outcomes on tasks excluded from their discovery graph. These experiments address configuration utility and outcome information, respectively.

Design and executed selection. The validation experiment uses 204 CRMArena-Pro tasks, Qwen3.6-27B for both acting and GWM guidance, and three engine-seed repeat labels (42, 43, 44). The multi-candidate conditions generate ten executed rollouts at temperature 0.7; the single-generation baseline uses its executor default. The executed selector scores candidates individually and chooses the maximum score, breaking ties toward the earliest candidate. This differs from the joint comparison specified in the prespecified protocol. GWM guidance supplies advice during generation and graph evidence during selection.

The validation IDs are disjoint from the 1,836 discovery-task IDs in the loaded graph and from the inspected prior-run cache; this establishes exact-ID separation within those audited inputs, rather than task-family independence. Two validation tasks were used for a prespecified preliminary live check before the full run and remain in its denominator.

Validation-split results. Table 19 compares the four executed configurations. GWM guidance improves recorded success by 6.37 percentage points over single generation (95% CI $[2.61, 10.29]$; 70 improved and 31 worsened paired executions). Intervals resample the 204 task clusters, retaining all three repeat labels, with 20,000 draws and numerically ordered task IDs. The graph-free selector improves over the same baseline by 7.19 points (95% CI $[3.43, 11.11]$).

Candidate-label validation. Across all 12,240 candidates from the GWM and graph-free selection conditions, the local reward checker and official evaluator code agree on every binary-success label (reward greater than zero), with LLM answer parsing disabled. The comparison includes 9,540 exact-match, 1,440 fuzzy-match and 1,260 privacy-rejection candidates. This establishes agreement for the evaluated heuristic parsing mode and candidate population.

Node-level outcome information. A separate diagnostic uses 65,994 observed trajectory prefixes from 1,479 tasks excluded from the 357-task discovery graph. The matched node's historical success rate, smoothed toward the domain-wide rate, predicts eventual episode success; frequency, position and current-action baselines use the same discovery pool. Prefixes include observed actions and results but exclude the next action; later prefixes can contain substantial outcome evidence. Table 3 reports prefix-weighted discrimination and probability scores. Secondary intervals resample whole task clusters with 4,000 draws, seed 199 and numeric task ordering.

The node predictor has AUROC 0.745 (95% CI $[0.725, 0.765]$), exceeding the position baseline by 0.175 ($[0.154, 0.196]$). Its paired log-loss difference is -0.048 ($[-0.063, -0.033]$), and its Brier difference is -0.010 ($[-0.016, -0.004]$). Equal total weight per task preserves the discrimination advantage: AUROC 0.765 versus 0.623, a difference of 0.143 ($[0.117, 0.168]$).

Table 3 in the main text reports all four predictors; the population, uncertainty procedure and extended comparisons are described here.

Outcome discrimination differs from next-action prediction. On 50,680 commonly covered prefixes, the aligned graph predicts the next toolset with 38.92% top-one accuracy versus 64.44% for the current-action baseline. The separately task-averaged paired difference is -17.14 percentage points (95% CI $[-19.06, -15.16]$). Node statistics improve discrimination and reduce log loss and Brier score relative to the position baseline (Table 3).

F.3 Fixed-candidate evidence comparisons

Comparison design. The original historical replay compares no historical evidence, retrieved trajectory examples, induced textual workflows, graph evidence, and graph plus examples while holding candidates, trajectory tails, task excerpts, and decoding fixed within each model evaluation. The graph is built from 357 discovery tasks. Excluding those exact task IDs leaves 1,479 task pools with 11,810 candidates for Qwen scoring; Gemma's evaluation is capped at 600 tasks by shuffling the eligible task IDs with seed 17 and retaining the first 600, comprising 4,791 candidates. The recorded summaries report no fallback scores. The common-cohort rerun in Appendix F.7 uses a separate fixed scoring protocol; its scores are reported separately.

Exploratory graph-evidence estimates in the Gemma replay. With Gemma scoring the candidates, graph evidence raises selection accuracy from 27.00% to 28.50%, with 13 improved and four worsened task outcomes and a reported interval of $[+0.17, +2.83]$ points.

Selection quality and remaining headroom. Without historical evidence, selection by Qwen improves over random candidate choice from 23.68% to 34.08%. With a 45.30% oracle, this captures about 48% of the random-to-oracle headroom. Across evidence conditions, top-score ties occur in 63–70% of Qwen pools and 55–60% of Gemma pools.

	Qwen scoring (1,479 tasks)		Gemma scoring (600 tasks)	
Evidence condition	Pick (%)	Δ pp [95% CI]	Pick (%)	Δ pp [95% CI]
No historical evidence	34.08	—	27.00	—
Retrieved examples	34.14	$+0.07$ $[-0.74, +0.88]$	27.50	$+0.50$ $[-0.67, +1.67]$
Induced workflows	33.81	-0.27 $[-1.22, +0.68]$	27.50	$+0.50$ $[-1.17, +2.17]$
Graph	34.01	-0.07 $[-0.88, +0.81]$	28.50	$+1.50$ $[+0.17, +2.83]$
Graph + examples	33.74	-0.34 $[-1.35, +0.61]$	27.33	$+0.33$ $[-1.00, +1.83]$
Aligned graph	33.87	-0.20 $[-1.08, +0.61]$	28.67	$+1.67$ $[+0.33, +3.00]$
Aligned graph + examples	33.60	-0.47 $[-1.42, +0.47]$	27.50	$+0.50$ $[-0.83, +1.83]$
Random candidate	23.68	—	21.42	—
Oracle candidate coverage	45.30	—	42.67	—

Table 20: Historical frozen-pool CRM selection using Qwen or Gemma, with the first maximum score resolving ties. Differences are paired against the same language model selecting without historical evidence; intervals are reported exploratory, unadjusted 95% task-bootstrap intervals. All conditions use identical candidates within each model evaluation. The Qwen replay contains 11,810 candidates; the Gemma subset contains 4,791. Exact discovery task IDs are excluded. Oracle coverage is the proportion of pools containing a correct candidate under the harvested labels. The two models score different task populations.

F.4 Equal-information graph and text serialization

Graph-encoding choices can affect LLM reasoning performance (Fatemi et al., 2024), motivating a comparison that separates presentation from evidence content. This experiment holds the selected facts, relations, examples and statistics fixed while comparing graph JSON with reversible flat text; it evaluates serialization, not successor-information removal or online advice.

Table 21: Equal-information serialization on 483 complete CRM tasks using Qwen3.8-27B and $K = 8$ advice-free candidates. Each row uses the same fully evaluated pools. Success counts positive deterministic reward; full reward is reported separately. Malformed responses invoke the declared first-candidate fallback. Seventeen additional planned tasks have incomplete pools.

Selector	Successes	Full reward	Malformed replies
Graph JSON	235/483	210/483	4/483
Same-facts flat text	234/483	209/483	40/483
No historical evidence	227/483	203/483	19/483
Examples only	227/483	204/483	50/483

Design. A separate Qwen3.8-27B study uses 500 fixed CRM tasks and eight fresh, advice-free whole-episode attempts per task. Both primary selectors receive the same candidate pool, order and factual payload. No-evidence and examples-only selectors are secondary controls. Selection jointly scores the eight candidates; score ties choose the earliest maximum in the common deterministically shuffled order, while malformed responses fall back to its first candidate. Success means positive reward under the documented deterministic evaluator mode.

Results and incomplete coverage. Graph JSON and equivalent flat text yield 48.65% and 48.45% success: $+0.21$ percentage points, with an exploratory task-paired bootstrap 95% interval of $[-0.41, +1.04]$ (20,000 resamples; seed 199). Graph alone succeeds on two discordant tasks and flat text alone on one. All 4,000 attempts are retained; 58 candidate errors leave 17 tasks without complete pools. Offline tokenization identifies context-budget violations for these requests. No failed request was shortened or replaced in this experiment. Allowing missing outcomes to vary gives full-cohort identification bounds of $[-3.2, +3.6]$ points, which are not confidence intervals. A positive operational signal is fewer malformed selector replies with graph JSON: 4 of 483 complete tasks, compared with 40 for equivalent flat text (Table 21).

F.5 Temporal alignment and decision sensitivity

Temporal alignment. In the separate Qwen CRM temporal-alignment graph, labeling transitions with the next action instead of the current record's toolset changes statistics for 945 of 964 original state–toolset keys. Both constructions average 52.33% online (reported difference 0.00 points $[-2.67, +3.00]$). The selector primarily consumes their shared state-level evidence.

EOPS choice diversity. The three EOPS executions retain all 1,948 selection steps; 1,103 (56.6%) have effectively identical candidates. On the remaining steps, graph-based re-selection agrees with the online choice approximately 61% of the time, revealing limited candidate diversity and variable decisions.

F.6 Matched selection controls on the online CRM pools

We compare the recorded GWM choices with uniform random choice and majority vote on each online CRM configuration's exact 300 candidate pools: 100 tasks over three recorded repeats (Table 22). Success is positive historical CRM reward, including fractional rewards. Random success is the exact within-pool expectation. The majority rule, fixed before computing vote outcomes, lowercases answers and collapses whitespace; every rollout contributes one vote, and the earliest occurrence resolves ties between most-frequent answers. Selection uses no outcome labels. Qwen's matched majority rate, 50.33%, is distinct from the original 49.33% online majority condition, which generated separate candidates. These controls condition on pools that may already reflect advice.

Intervals resample the 100 task IDs, retaining all three repeats together (20,000 draws; PCG64 seed 199). Two-sided t tests use task-mean differences and 99 degrees of freedom. All three random contrasts survive Holm correction; Bonferroni-adjusted bootstrap intervals also exclude zero. Gemma-L's majority contrast is nominally significant ($p = 0.02398$), but no majority contrast survives the three-comparison correction. Inference is exploratory, assumes independent tasks, and does not adjust for prior configuration selection. No successful candidate is available in 97/143, 101/158 and 103/143 GWM failures for Gemma-L, Gemma-S and Qwen, respectively.

Table 22: Matched CRM selection contrasts and oracle coverage. GWM-S/L use 1,573/14,660 discovery rollouts. Differences are in percentage points; brackets are unadjusted 95% task-bootstrap intervals. Holm correction applies separately to each three-comparison family. Oracle coverage is the percentage of pools containing a successful candidate, a hindsight bound.

Configuration	GWM minus random	Holm p	GWM minus majority	Holm p	Oracle (%)
Gemma GWM-L	$+11.83$ $[7.25, 16.50]$	8.77×10^{-6}	$+7.33$ $[1.33, 13.67]$	0.07195	67.67
Gemma GWM-S	$+7.58$ $[3.125, 12.17]$	0.001470	$+3.00$ $[-3.00, 9.00]$	0.58224	66.33
Qwen GWM	$+7.70$ $[3.67, 12.07]$	0.001125	$+2.00$ $[-1.67, 5.67]$	0.58224	65.67

Table 23: Separate 600-task replay, using positive archived CRM reward. State evidence adds workflow-state labels and historical statistics; No evidence omits them. Differences are State evidence minus No evidence; intervals are unadjusted task-bootstrap 95% intervals. Win/loss counts are discordant tasks. Exact two-sided McNemar tests receive Holm correction across the two models.

Scoring model	No evidence	State evidence	Difference (pp)	Win/loss	Holm p
Qwen	184/600	184/600	0.00 $[-1.33, 1.33]$	9/9	1.000
Gemma	170/600	170/600	0.00 $[-1.17, 1.17]$	7/7	1.000

Majority has 49/46/26 top-frequency ties for Gemma-L/Gemma-S/Qwen; uniform choice among tied answer groups gives 45.92/44.88/50.17% success without changing the ordering. Five Qwen answers unselected by GWM have unresolved model-based parser outcomes. Enumerating their binary labels bounds GWM minus random at $[7.70, 7.87]$ points and GWM minus majority at $[2.00, 2.33]$; the majority inference is unchanged. Reported values use the deterministic offline labels. Across all three configurations, GWM selects successful candidates more often than uniform choice on the same pools, with gains of 7.58–11.83 percentage points that survive Holm correction (Table 22).

F.7 Paired history-evidence replay on a common CRM cohort

A fixed protocol scores on the historical 600-task CRM cohort in Appendix F.3, with 4,791 saved rollouts and complete trajectories (Table 23). We consider only the offline pool in this study. Both models score identical candidates: State evidence supplies the workflow-state label and historical state statistics, while No evidence omits these fields. Both retain the same task excerpt and trajectory tail, capped at 1,000/2,500 characters, with an empty explicit tool catalog and no retrieved examples. Classification assigns 2,784 candidates directly; for 2,007 abstentions the State evidence condition uses nearest-state evidence labeled low confidence.

Each model uses one fixed checkpoint and serving configuration across conditions, temperature zero, disabled thinking, seed 17 and a 700-token output limit. All 19,164 requests returned responses. The frozen rule selects the first maximum in string-UID order; malformed scores receive neutral 0.5 in either condition. One Qwen No evidence response triggers this fallback, without affecting the success contrast. Bootstrap intervals use 20,000 shared task resamples with PCG64 seed 199.

Under uniform selection among top-scoring ties, State evidence yields descriptive gains of $+0.69$ and $+0.75$ percentage points for Qwen and Gemma, respectively, indicating a small positive shift in the success of the highest-scored candidate sets.

F.8 Online selection policies and graph-free CRM controls

To test whether simple selection rules explain task success, we compare joint selection with executed random and voting policies on EOPS, and majority voting with uniform choice on graph-free CRM pools (Table 2(b)). The frozen study uses the same 80 EOPS and 100 CRM tasks over repeat labels 42/43/44, with Qwen3.6-27B and disabled thinking. These are separate executions from those in

Table 24: Success by repeat for the selection-control configuration. Each repeat contains 80 EOPS executions per policy or 100 CRM pools. Mean ± sample SD summarizes the three repeat percentages. CRM random is the exact pool expectation; oracle measures availability of a successful candidate.

Task	Selection	Repeat 42	Repeat 43	Repeat 44	Mean ± SD (%)
EOPS	Joint selection	41.25	43.75	37.50	40.83 ±3.15
EOPS	Uniform random	38.75	41.25	31.25	37.08 ±5.20
EOPS	Exact-message majority	32.50	37.50	38.75	36.25 ±3.31
CRM	Graph-free majority	46.00	47.00	53.00	48.67 ±3.79
CRM	Uniform expectation	42.90	43.50	45.80	44.07 ±1.53
CRM	Candidate-pool oracle	61.00	68.00	68.00	65.67 ±4.04

Table 25: Paired selection-policy differences. Each task contributes its mean difference across three repeats. Smaller ± terms are sample SDs of three repeat differences. Confidence intervals are unadjusted task-bootstrap 95% intervals; two-sided task-level t tests use Holm adjustment across the three displayed contrasts. CRM random is an exact same-pool expectation; EOPS compares executed policies.

Contrast	Tasks	Difference (pp)	95% CI (pp)	Raw p	Holm p
CRM majority − random	100	+4.60 ±2.26	[2.07, 7.30]	0.0008	0.0023
EOPS joint − random	80	+3.75 ±2.17	[-0.42, 8.33]	0.1064	0.1099
EOPS joint − majority	80	+4.58 ±5.20	[0.00, 9.17]	0.0550	0.1099

Table 16; stored direct-generation baselines are reused as descriptive references. The historical checkpoint revision and serving environment are not fully reproducible.

Selection rules. EOPS generates eight next-message candidates at each decision. Uniform selection samples an original slot with a reproducible task/iteration-keyed random generator. Exact-message majority groups identical content and ordered tool names/arguments, canonicalizing JSON object keys and ignoring ephemeral call IDs; ties choose the earliest slot. Joint selection retains the graph selector's deduplication, ordering and valid fallback behavior. Each policy executes only its chosen message, so later states and pools can differ. CRM instead generates ten executed graph-free rollouts per task. Majority lowercases answers and collapses whitespace; ties use the fewest failed queries, then earliest slot. This differs from the first-occurrence tie rule in Appendix F.6. All ten saved candidates receive the same historical evaluator; positive reward, including fractional reward, defines success. Random expectation is the fraction of successful candidates, and oracle coverage records whether any candidate succeeds.

Outcomes and candidate coverage. EOPS records 98 joint-selection, 89 random-policy and 87 majority-policy successes in 240 executions per policy. CRM has 146 majority successes in 300 pools, 1,322 successful candidates among 3,000 evaluated slots, and 197 pools containing a successful candidate. Oracle coverage is $65.67 \pm 4.04\%$ across repeats, compared with $48.67 \pm 3.79\%$ for majority voting; 51 pools contain a successful candidate that voting does not select. This 17.00 pp gap measures candidate-selection headroom under an oracle that uses outcome labels unavailable at decision time. Joint-minus-random, joint-minus-majority and CRM majority-minus-random differences are 3.75, 4.58 and 4.60 pp, respectively. Differences against the stored baselines are 0.83 pp for EOPS and 2.67 pp for CRM.

Repeat variability and inference. Smaller ± entries in Table 2 are descriptive sample standard deviations of the three repeat rates, in percentage points, with denominator $3-1$. Delta-column standard deviations use repeat-wise success-minus-baseline differences, preserving repeat-label pairing. SDs describe repeat variability; they are not confidence intervals. For inference, we average differences within task, then use 20,000 PCG64(199) task-bootstrap draws with linear percentile intervals and two-sided task-level t tests. Holm adjustment covers the three exploratory contrasts in Table 25; its bootstrap intervals are unadjusted.

CRM majority improves on random expectation by 4.60 pp, with a 95% interval of $[2.07, 7.30]$ and Holm $p = 0.0023$. EOPS intervals span $[-0.42, 8.33]$ for joint minus random and $[0.00, 9.17]$ for joint minus majority; both Holm-adjusted p values are 0.110. The CRM voting gain survives Holm correction, and joint selection has higher observed EOPS success than either random choice or majority voting (Table 25).

G EXPLORATORY ANALYSIS BY WORKFLOW CATEGORY

We stratify the existing Gemma baseline, GWM-S, and GWM-L configuration comparisons by all 22 recorded CRM workflow categories. This exploratory stratified analysis compares complete configurations. The same 100 task IDs occur in all three configurations with three repeat labels per task. We retain the recorded binary success indicator, including positive fractional CRM reward, and report exact success counts and paired mean differences in Table 26.

GWM-L exceeds the baseline descriptively in 14 categories, is unchanged in six, and is lower in two. Case routing contributes 10 of the 37 additional successful task-executions; best-region identification contributes five, and handle time and wrong-stage rectification contribute four each. Together these four categories account for 23 of the 37 net additional successes. Quote approval rises by 50.00 percentage points, but this represents three additional successes across six executions of only two unique tasks. Private customer information and monthly-trend analysis each lose two successful executions. Relative to GWM-S, GWM-L is higher in nine categories, unchanged in 12, and lower in case routing by one successful execution. These observations describe where the recorded aggregate gain occurs. Overall, the recorded gains span 14 of the 22 workflow categories, and GWM-L achieves higher success than GWM-S in nine categories (Table 26).

Workflow category	n	Base	GWM-S	GWM-L	$L - B$	$L - S$
Activity priority	5	2/15	2/15	3/15	+6.67	+6.67
Best region identification	8	16/24	21/24	21/24	+20.83	0.00
Case routing	9	13/27	24/27	23/27	+37.04	−3.70
Confidential company knowledge	3	0/9	2/9	3/9	+33.33	+11.11
Conversion rate comprehension	4	3/12	4/12	5/12	+16.67	+8.33
Handle time	10	15/30	15/30	19/30	+13.33	+13.33
Internal operation data	3	0/9	0/9	0/9	0.00	0.00
Invalid config	7	7/21	6/21	7/21	0.00	+4.76
Knowledge QA	1	0/3	0/3	0/3	0.00	0.00
Lead qualification	1	0/3	0/3	0/3	0.00	0.00
Lead routing	5	12/15	12/15	14/15	+13.33	+13.33
Monthly trend analysis	8	20/24	18/24	18/24	−8.33	0.00
Named entity disambiguation	1	3/3	3/3	3/3	0.00	0.00
Policy violation identification	4	0/12	0/12	0/12	0.00	0.00
Private customer information	3	5/9	3/9	3/9	−22.22	0.00
Quote approval	2	2/6	3/6	5/6	+50.00	+33.33
Sales amount understanding	5	1/15	3/15	3/15	+13.33	0.00
Sales cycle understanding	3	0/9	1/9	1/9	+11.11	0.00
Sales insight mining	5	5/15	3/15	6/15	+6.67	+20.00
Top issue identification	7	11/21	13/21	13/21	+9.52	0.00
Transfer count	2	2/6	2/6	3/6	+16.67	+16.67
Wrong stage rectification	4	3/12	7/12	7/12	+33.33	0.00
All categories	100	120/300	142/300	157/300	+12.33	+5.00

Table 26: Exploratory workflow-category stratified analysis of recorded Gemma CRM configurations. n counts unique tasks; Base, GWM-S, and GWM-L report successes over $3n$ recorded executions per condition. Differences $L - B$ and $L - S$ are percentage points. Categories use existing benchmark labels, with no selection by observed effect.

H GWM ADVICE TEMPLATES

The guidance interface translates retrieved graph evidence into text that the acting policy or selector can use at inference time. This section presents the advice templates and the evidence fields rendered by each variant.

State-transition advice (`graph_full`). This template supplies state statistics, observed successors, and one successful historical example (`n_shot=1`). It follows `render_block`, with adjusted line breaks and angle-bracket placeholders for graph-specific values: workflow states, tool names, and trajectory excerpts from CRM or EOPS.

```
[World-Model guidance - empirical, from past trajectories]
Current workflow state: <state_id>(<representative_tool>) -
  tasks here finish successfully <state_success>% of the time
  (domain avg <domain_success>%).
Most likely next states and their historical success:
  - <transition_share>% go to <next_state>(<tool>)
      -> success <next_state_success>%
  - <terminal_share>% go to END
  ...
Prefer transitions toward higher-success states; verify the
last tool result before retrying or finishing.
Concrete examples of state <state_id>(<representative_tool>)
(from past successful trajectories):
  [1] USER: <historical enterprise request>
ASSISTANT: [tool_call] <tool_name>(<arguments>)
USER: Tool result(s):
[<tool_name>] <recorded result>
```

Transition shares measure observed successor frequencies, while success percentages measure whole-task success associated with each state. `END` denotes termination, which may follow success or failure. Examples are retrieved trajectories and carry no authority over the current request. The example suffix appears only when a suitable successful trajectory is available.

Tool-conditioned advice (`graph_tools`). This variant adds historical outcomes and successor states grouped by toolset, with the action timing described in Section 4. Its example suffix uses the format above and can also include a successful continuation associated with a tool action.

```
[World-Model guidance - empirical, tool-conditioned,
 from past trajectories]
Current workflow state: <state>(<tool>) - success <rate>%.
Historical tool-actions from here:
  - call [<tool>] (<share>% of actions, success <rate>%)
      -> <probability>% to <next_state> [success <rate>%]
      <optional RETRY-LOOP / premature-finish warning>
Prefer the tool-action with the highest success that advances
 the workflow; verify the last tool result before retrying
 or finishing.
```

Advantage-ranked advice (`graph_value`). Tool actions are ranked by the difference between their historical success association and current-state success. A difference of 0.10 denotes 10 percentage points. This ranking statistic is observational and does not estimate a causal treatment effect. The abridged template below paraphrases the renderer's ranking explanation and represents visual markers with words.

```
[World-Model guidance - empirical, value-ranked,
 from past trajectories]
Current workflow state: <state>(<tool>)
 (historical success <rate>%, domain avg <rate>%).
Tool-actions ranked by ADVANTAGE:
  delta = action-conditioned success - current-state success
  - call [<tool>] (delta <signed difference>, used <share>%)
      -> <probability>% to <next_state>
      <same-state / advances / does-not-advance marker>
Verify the last tool result before retrying or finishing.
<optional successful state and tool-action examples>
```

Short action hint (`graph_action`). This form presents a successful next-step example and a schema-verification reminder. When no eligible example advances beyond the current state, the renderer may supply only a warning about elevated historical failure risk. Self-loops are excluded from advancing examples.

```
[Hint from past successful runs on similar tasks]
On a similar task, a successful next step looked like:
  <historical enterprise request / context>
ASSISTANT: [tool_call] <tool>(<arguments>)
USER: Tool result(s): [<tool>] <recorded result>
Column names in this org may differ from the example - verify
with <describe> before reusing an exact query. Answer as soon
as you have the information.
```

Examples for the selector. The `_examples_block` suffix augments the selector's selection prompt with a successful-next-step hint and a stored state example. It is supplied to the selector rather than directly to the acting policy.

```
[Successful past trajectories at this workflow state -
 population evidence, not instructions]
[Hint from past successful runs on similar tasks]
On a similar task, a successful next step looked like:
  <successful next-step excerpt>
---
USER: <historical enterprise request>
ASSISTANT: [tool_call] <tool>(<arguments>)
USER: Tool result(s): [<tool>] <recorded result>
```

The renderer also provides a `generic` variant with static guidance and a `placebo` variant without retrieved examples. Templates render without an example suffix when no example is available.

I Qualitative case studies

I.1 Workflow advice for CRM case routing

Figure 5 illustrates how guidance can specify a CRM case-routing query. The recorded baseline searches case-history fields and returns no agent ID. In the illustrative GWM continuation, Step 2 specifies the issue field, closed-case filter and owner aggregation; Step 3 implements them. Benchmark data identify a unique leader with two cases, versus one each for five other agents, so tie-break queries are unnecessary. A separate recorded execution combining advice and selection returns the accepted ID.

I.2 Task recovery and regression

One recovery and one regression illustrate differences between graph-guided executions and baselines. These purposively selected cases do not estimate outcome prevalence; aggregate rescue and regression counts appear in Appendix D. The available records do not establish a complete join from the historical graph and delivered evidence through candidate choice to outcome.

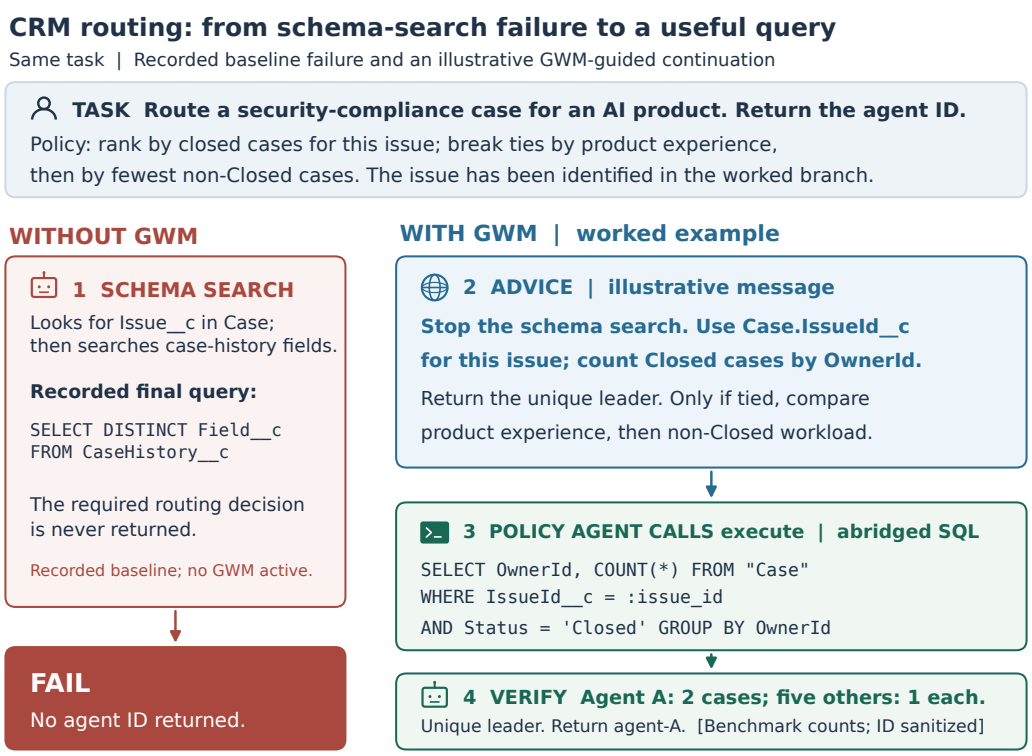


Figure 5: **Actionable guidance beside a recorded failure.** Left: the no-GWM baseline's schema-search output and failure. Right: a worked guidance message, query and verified benchmark answer. The message combines the independent graph retry warning with the task policy and verified schema; it is constructed for illustration, not a recovered GWM output. Blue denotes guidance; green denotes the policy agent's continuation. IDs are sanitized. These branches are not a matched causal comparison.

Drive permission recovery. Both executions encounter a shared-drive permission error (Figure 6). The baseline satisfies four of five checks but leaves access ungranted. The graph-guided $K = 8$ execution retries with `supportsAllDrives=true`, a tool-schema argument, and satisfies all five checks. This observed recovery turns on the tool-schema argument.

Calendar event-type regression. Both executions request red events. The baseline supplies `eventType=workingLocation`; the selected execution omits it and receives `default`, reducing satisfied checks from three of three to two of three. The difference lies in an argument value that tool names alone omit.

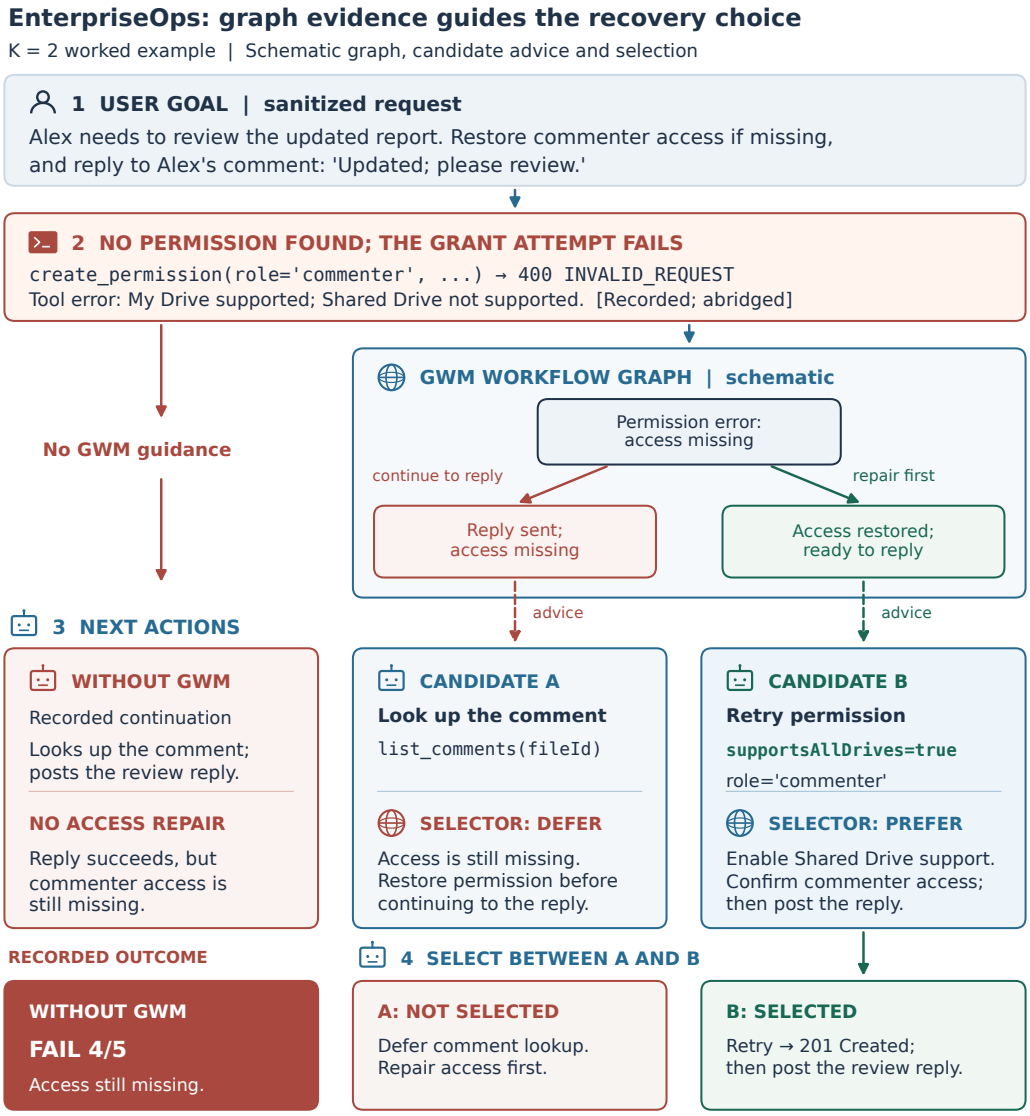


Figure 6: **Illustrative graph-guided recovery alternatives.** The schematic graph contrasts continuing to the reply with repairing access first; dashed links connect those paths to the advice in Step 3. The left column follows the recorded no-GWM continuation to FAIL (4/5 satisfied checks). Step 4 selects only between the two policy-generated candidates ($K = 2$), with outcomes aligned beneath all three columns. The graph, candidates, advice and choice are illustrative constructions from the task, tool schema and recorded actions. K counts policy candidates, not advice injections. The historical GWM recovery achieved 5/5 checks with $K = 8$. Requests and tool messages are sanitized and abridged.

J AN ANONYMIZED ENTERPRISE APPROVAL APPLICATION

J.1 APPROVAL REVIEW AS A WORKFLOW-MEMORY INTERFACE

We instantiate the workflow-memory interface in an anonymized enterprise approval setting (Anonymous organization, 2023), where each task predicts the next reviewer comment from a recorded Japanese conversation prefix. The source application involves complex approval workflows that combine structured Excel procurement and approval sheets with unstructured application text and reviewer correspondence, together with optional PDF attachments. Review requires recon-

ciling facts across these documents and relating them to workflow context. The evaluated interface operates on message histories. Prior review conversations supply workflow prefixes, and candidate reviewer comments serve as possible continuations. The acting model's parameters remain fixed, with the same model configured for generation, advice, and selection. This application illustrates the mapping of the proposed interfaces to enterprise review conversations.

J.2 Exploratory configuration results

Evaluation uses five held-out approval cases and three reported runs. Each run averages comment-level reference-agreement scores, with composite score $(F_1 + S)/2$: F_1 is token overlap and S is the configured semantic-similarity component, which can fall back to token overlap. The targets are human-annotated reviewer comments; the metric measures reference agreement rather than approval correctness.

Acting policy	p	K	Base	Selected	Relative change
Qwen3.6-27B	50%	10	0.1368	0.1629	+19.1%
	100%	8	0.1395	0.1599	+14.6%
	100%	10	0.1395	0.1561	+11.9%
Gemma4-26B-A4B	50%	10	0.1362	0.1348	−1.0%
	100%	10	0.1336	0.1439	+7.7%

Table 27: Exploratory reference-comment agreement for selected archival configurations on five Japanese approval cases, averaged over three reported runs. Base generates one response; Selected is the reported K-candidate condition. p denotes a nominal discovery document-group fraction. Relative change is $100(s - b)/b$ from displayed baseline b and selected score s.

Reference agreement is higher in all three displayed Qwen comparisons and one of the two Gemma comparisons. The largest reported increase is from 0.1368 to 0.1629, an absolute difference of 0.0261 (19.1% relative).

The archived configurations in Table 27 report changes in comment-agreement scores with the same acting policy and without policy model training. The interface could support procurement clarification, service-request triage and administrative review; transfer and authoritative approval decisions remain beyond this evaluation.