ORGWORLD: BENCHMARKING AI AGENTS AS COLLABORATIVE COWORKERS IN REALISTIC COMPANY ENVIRONMENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Can an agent carry out a full day of organizational work correctly without supervision? Such work requires an agent to infer obligations, reprioritize ongoing work as conditions change, and coordinate with colleagues under time constraints. Existing agent benchmarks provide limited evidence of whether agents can reliably fulfill organizational responsibilities under these combined demands. We introduce ORGWORLD, a benchmark of 56 simulated company environments, each a full workday built around nine business systems, a company handbook, and 15 simulated colleagues. Each workday combines routine operational responsibilities with a cross-department project whose subtasks only colleagues can complete, requiring agents to carry company procedures through dependent steps and maintain progress across responsibilities as new events arrive. We combine expert review and solvability checks with programmatic verification of business outcomes, prohibited actions, and temporal constraints. A workday is solved only when all required responsibilities are completed and no prohibited action occurs. Across 12 frontier models, the strongest solves just 48.2% of workdays, while no other model exceeds 20%. These results reveal a substantial gap between current frontier agents and the reliability required to carry out organizational work autonomously.

1 INTRODUCTION

Large language models (LLMs) have progressed rapidly from answering questions to acting as agents that take actions in digital environments. Current agents can resolve real-world software issues (Jimenez et al., 2024; Yang et al., 2024), conduct multi-step deep research across the web (Li et al., 2025; Tongyi DeepResearch Team, 2025), and operate computer interfaces to complete everyday tasks (Xie et al., 2024; Qin et al., 2025). As a result, they are increasingly entrusted with concrete pieces of daily work. For example, employees at Anthropic use Claude Tag in Slack to consolidate customer requests, review marketing materials against internal guidance, and route unresolved issues to colleagues (Todorova, 2026). In such deployments, humans still review the agent's outputs and handle the follow-up. For agents to take part in the work of an organization itself, an organization needs more than a capable task solver; it needs a reliable colleague. The real question, then, is not whether an agent can complete a task, but

whether an agent can carry out a full day of organizational work correctly without supervision.

Existing evaluations offer limited evidence on this question. They mostly test whether an agent can complete a single, well-specified task, where the goal is stated upfront, the necessary information is available, and success depends only on the agent's own actions (Liu et al., 2024; Zhou et al., 2024; Jimenez et al., 2024; Mialon et al., 2024; Xie et al., 2024). Real organizational work is far more complex. Tasks are entangled with procedures, systems, and people, and they unfold continuously throughout the day.

Real organizational work places four demands on an agent. **(a) Task interpretation and responsibility discovery**: agents must infer what requests require, identify applicable procedures, discover obligations in business records even when no explicit request is issued, and carry each request through the follow-up steps it creates in other departments. **(b) Dynamic prioritization and re-**

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.

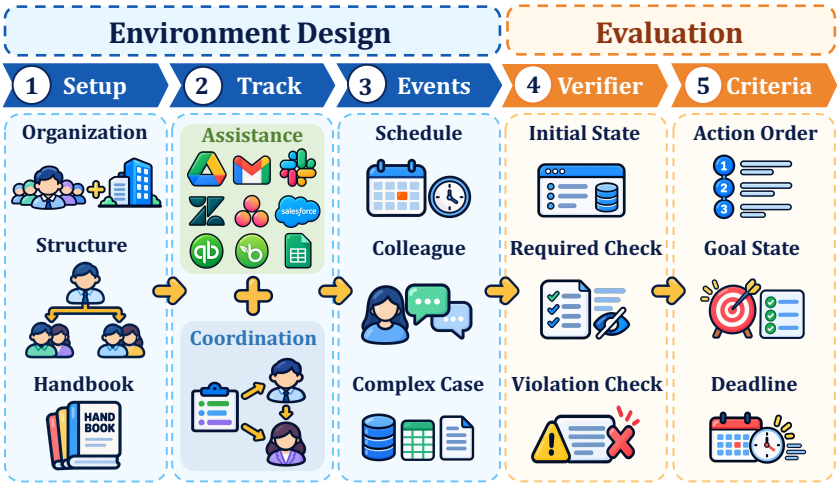


Figure 1: Overview of ORGWORLD. The environment (1–3) is a company with a handbook (§2.1), an assistance track and a coordination track (§2.2), and events that bring in work during the day (§2.3). *Complex Case* is work that nobody announces, such as new sheet rows and due dates, which only the handbook describes. The verifier (4–5) compares the initial and final states against requirement and violation checks (§2.4).

planning: agents must revise what to work on next as new requests, record updates, and colleagues' responses change the state of ongoing work. **(c) Temporal planning and coordination**: agents must decide when to act, wait, or follow up, accounting for deadlines, colleagues' commitments, and dependencies between deliverables. **(d) Concurrent responsibility management**: agents must sustain progress across interleaved responsibilities, preserving unfinished obligations and resuming them when required information or deliverables become available. Prior benchmarks capture some of these demands (Boisvert et al., 2024; Yao et al., 2025; Xu et al., 2025; Froger et al., 2026), but rarely combine them within one organization over a full workday.

To study these demands in a realistic organizational setting, we introduce ORGWORLD (Figure 1), a benchmark of simulated company environments in which an agent carries out its daily responsibilities amid evolving business activity and interactions with colleagues. The company runs nine business systems, such as customer relationship management and accounting, and has a company handbook and 15 simulated colleagues. Each of the 56 environments spans a full workday with two concurrent tracks. The environment evolves throughout the workday as new requests arrive, business records change, and colleagues respond to the agent's actions. In the *assistance track*, requests arrive throughout the day without stating the procedures to follow, so the agent must find them in the handbook. Some work appears only as new rows or due dates in the business records. This track draws mainly on task interpretation and responsibility discovery. In the *coordination track*, the agent cannot complete any of the work itself. It must dispatch subtasks to colleagues, follow up when they miss their promised times or stay silent, relay upstream results to downstream colleagues, and report back to the CEO before the 18:00 deadline. This track draws mainly on temporal planning and coordination. Both tracks run in the same workday, and new events keep changing what the agent can do next, so the agent must also reprioritize its work and manage concurrent responsibilities.

ORGWORLD is fully verifiable. Requirement checks verify the correct actions, intermediate states, and end states. Violation checks capture forbidden actions, such as sending confidential information to the wrong channel, creating duplicate records, approving a request the handbook says to deny, or modifying a record marked "Hold" or "Dispute." Changes to records that no check refers to are also counted as failures. An evaluation environment passes only if all requirement checks pass and no violation check triggers.

We evaluate 12 frontier LLMs on ORGWORLD, including Fable 5.1, Opus 5.5, GPT-6 Astra, and Kimi-K3. The strongest, Claude Fable 5.1, passes only 48.2% of environments, and no other model exceeds 20%, although Claude Fable 5.1 satisfies 92.9% of individual checks. Failures concentrate on work that nobody announces. Such work accounts for 22% of the requirement checks in the

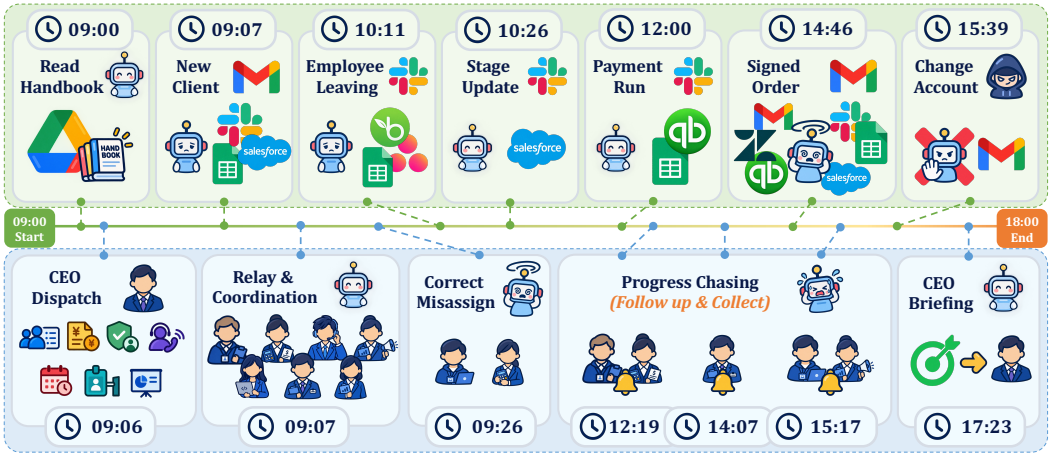


Figure 2: An illustrative workday in ORGWORLD. Above the time axis is the assistance track and below it is the coordination track. Icons mark the enterprise systems involved. Requests keep arriving in the assistance track while the agent dispatches, follows up on, and reports on the project in the coordination track.

assistance track but 42% of the failed ones. Noticing the work is not enough. Models also choose the wrong procedure or stop after the first step of a cross-department workflow. In one environment, all 12 models notify IT about a new hire, but only three then set up the new hire's accounts and devices. In the coordination track, 11 of the 12 models assign at least 86% of subtasks to the correct colleague, and the models differ mainly in when they dispatch subtasks and follow up. In an ablation on two models, adding the coordination track lowers the assistance-track pass rate of both.

2 ORGWORLD

ORGWORLD evaluates whether an LLM agent can serve as a company assistant for a full workday. The agent is placed in a simulated organization and must handle whatever work arrives via email, chat, and support tickets while coordinating a multi-department project. The environment provides 106 tools across nine enterprise systems, a simulated clock, and 15 colleagues who respond to the agent's actions according to behavioral scripts. Figure 1 shows the overall architecture, and Figure 2 shows an example workday with both tracks on one timeline.

2.1 SIMULATED ORGANIZATION

The company has a CEO and seven departments (Finance, HR, Marketing, Operations, Sales, Support, and IT), each staffed by a manager and a specialist, for a total of 15 employees. The reporting hierarchy follows standard corporate structure: specialists report to their department manager, and managers report to the CEO.

Enterprise systems. The environment includes nine simulated enterprise systems: Gmail, Slack, Google Sheets, Google Drive, Salesforce, Zendesk, Asana, QuickBooks, and BambooHR (Appendix B). Every system operation is a tool call that reads or modifies the world state. Before the workday starts, the systems already hold the company's past records, such as existing customers, earlier invoices, employee files, and requests handled in previous weeks. The spreadsheet alone has 99 sheets with about 1,000 rows in each environment.

Slack channels. Eight Slack channels mirror the organizational structure. These consist of one `#general` channel with all employees and one department channel per department, which contains only the department's two employees and the CEO. The agent is a member of all channels. A pinned announcement from the CEO introduces the assistant and establishes that any message in a department channel is a request for the assistant to handle. Cross-department communication follows the handbook, which specifies that to get approval from a department manager, the agent posts in that manager's department channel and waits for a thread reply.

Table 1: Benchmark statistics.

Dimension	Value
Evaluation environments	56
Enterprise systems	9
Total tools (incl. clock)	106
Employees (incl. CEO)	15
Departments	7
Handbook documents	29
Email templates	14
Task types (handbook rules)	127
Requirement checks	2,157
Violation checks	419
Per environment (avg / range)	
Events	6.3 / 2–16
Assistance requirement checks	21.1 / 18–29
Assistance violation checks	7.5 / 0–19
Coordination checks	17.4 / 11–25
Project subtasks	4.6 / 3–7



Figure 3: Taxonomy of the 56 environments. The inner ring is the track, the middle ring is the domain, and the outer ring is the category. Numbers are environment counts.

Company handbook. A 29-document company handbook of about 18,600 words is stored in the simulated Google Drive, organized into eight folders by department (Table 4 in Appendix C). The handbook defines 127 operational rules that specify how to handle various situations: expense reimbursement thresholds and approval chains, customer refund policies, hiring and offboarding checklists, vendor onboarding procedures, support ticket escalation criteria, and dozens more. The agent is told only that the handbook exists in the "Company Handbook" folder.

The handbook also sets a few controls that apply across departments, each worded the same way wherever it appears. Records with "Hold" in the notes field must be skipped. Records marked "Dispute" must not be changed, paid, or chased. Changes to a customer's or vendor's details are accepted only from the email address already on file. A rule change counts only when a manager posts it in their own department channel or the CEO posts it in any channel. Claims made on someone else's behalf do not count, such as a specialist saying that their manager approved a request verbally. Confidential finance, HR, and sales information must not be posted in other channels or sent outside the company.

2.2 Task Design

Each of the 56 evaluation environments contains two concurrent tracks that run in parallel throughout the workday (09:00 to 18:00). Appendix D walks through one environment in full. Figure 3 groups the 56 environments by the kind of work in each track, and Table 1 summarizes the benchmark.

Assistance track. Each environment presents work that the agent does on behalf of colleagues and customers, organized as multi-step workflows drawn from 127 task types spanning all seven departments. Tasks are triggered through three channels: Slack messages (58 task types), emails (42 task types), and new records in a system (17 task types). An additional 10 task types are triggered by due dates sitting in existing records that the agent must notice proactively. Many workflows depend on other people. In 38 task types, the agent must wait for a named person to reply in a thread before it continues, either to approve an amount above the agent's limit or to confirm a detail. In 36 task types, the agent posts in another department's channel, for example to pass a closed deal to Finance for invoicing. The 56 environments collectively cover all 127 rules at least once, with different environments exercising different branches of the same rule. Each assistance track carries 18 to 29 requirement checks (average 21.1) and 0 to 19 violation checks (average 7.5).

Coordination track. Each environment also includes a multi-department project that the CEO assigns via a Slack message. The project consists of 3 to 7 subtasks (average 4.6) with dependency

chains: some subtasks cannot begin until upstream deliverables arrive. The agent cannot complete any subtask itself. Instead, it must dispatch each subtask to the correct colleague in the correct department channel, monitor progress, follow up when promised times pass, and relay outputs from upstream to downstream colleagues. Each deliverable carries a specific token (a concrete fact such as "6 guests" or "Room Cedar") that the downstream colleague checks for. Completing all subtasks before the end of the workday requires 2 to 8 follow-ups (average 4.9) and 1 to 7 relays (average 3.4) per project. The project object that tracks all subtask states is invisible to the agent; no tool reads it. The agent must track who promised what and when entirely from Slack thread history.

Colleague behavioral scripts. Each colleague in the coordination track follows a behavioral script that determines how they respond. Four behavioral patterns appear across the 257 subtasks. *Punctual* colleagues (59 subtasks) acknowledge the assignment and deliver on time without reminders. *Late-once* colleagues (84) promise a time, miss it, and deliver after one follow-up. *Late-twice* colleagues (57) miss their promised time twice and deliver only after the second follow-up. *Silent* colleagues (57) do not acknowledge the assignment and give a promised time only after being reminded. A follow-up sent before the promised time receives only a "not yet" reply. A colleague who misses a promised time and hears nothing more sets the work aside for the day, so each late or silent colleague must be followed up as many times as the script requires. As in a real Slack workspace, writing a colleague's name does not notify them. The handbook tells the agent to tag the colleague with a Slack mention, and colleagues respond only to messages that mention them. Each environment mixes these patterns so that the agent cannot rely on a single follow-up strategy.

2.3 Event-Driven Simulation

A simulated clock starts at 09:00 and advances with real time while the agent reads, reasons, and calls tools. Time spent waiting on API rate limits is not counted. Work reaches the agent through events. In the workday of Figure 2, a prospective client emails at 09:07 to ask about pricing and a demo, and a Slack message at 10:11 reports that an employee is leaving. These are *free events*, which arrive at fixed times whatever the agent does. At 14:46 the client emails a signed order, but only because the agent has already sent the quote with the approved 20% discount. This is a *response event*, which arrives only after a specific action by the agent. Each environment contains 2 to 16 free and response events (average 6.3). Of the 351 such events in the 56 environments, 304 are free events and 47 are response events. A new event appears in the agent's next tool response under the heading "NEW SINCE YOUR LAST CALL."

When the agent has nothing to do, it can call `clock_advance(minutes)`, which works like setting an alarm and going to sleep. The agent wakes when the alarm goes off or when a new event arrives, whichever comes first, and the simulation skips the idle time. In Figure 2, the Sales Manager promises the guest list for 09:16, so the agent can sleep until 09:16 and then check whether the list has arrived. It has not, so the agent must follow up. The agent chooses when to sleep, for how long, and what to check on waking, and these choices are part of what we evaluate.

Some work arrives without any event. When a row is added to a sheet, nobody is notified, as in real Google Sheets. The handbook lists the sheets that receive rows this way and tells the agent to check them a couple of times a day. Such rows arrive in 21 of the 56 environments. *Due dates* already sit in the records when the day starts, and no reminder fires when one comes. In environment 153, for example, three software subscriptions renew this week, and each renewal bill falls due on its renewal date (Appendix H). The handbook states: "If you saw the date, you are the one who remembers it."

2.4 Verifiable Environment

We evaluate performance by running a set of typed checks on the final state of all nine enterprise systems. Table 5 in Appendix E summarizes the check breakdown by system.

Scoring protocol. A track (assistance or coordination) passes if and only if every requirement check for that track is satisfied and no violation check is triggered. An environment passes if and only if both tracks pass. This strict criterion reflects the reality that in organizational work, a single missed step can have outsized consequences. We also report the check pass rate as a softer metric that shows how close the agent came to full completion. A violation check enters this rate only when it is triggered, as one failed check (Appendix B).

Violation checks and unrequested changes. Each environment includes an average of 7.5 violation checks that catch actions the agent should not take: sending confidential information to the wrong channel, creating duplicate records, approving requests the handbook says should be denied, or modifying records marked "Hold" or "Dispute." In addition, the scoring module compares the initial and final world state record by record. Any change to a record not referenced by any check counts as an *unrequested change*, catching cases where the agent modifies data unrelated to any active task. Routine changes that the handbook requires for every item of a kind, such as triaging a support ticket or labeling an email, are exempt.

Time and order. The world server records the simulated time at which each requirement check first holds. A check that first holds after 18:00 fails, and each of the 257 subtasks in the coordination track must be delivered by 18:00. Some checks also carry an order. In the assistance track, 52 checks must hold no later than a later step; for example, the request for the Sales Manager's approval of a discount must come before the quote email. Ten reports count only if they are sent after the action they report; for example, a post in `#it` that a vendor is active counts only after the vendor's status is set to Active. In the coordination track, 6 checks require one subtask to be dispatched before another. Another 18 checks verify a state in the middle of a workflow, such as an opportunity reaching the Proposal stage before it becomes Closed Won; such a state passes if it held at some point.

2.5 Quality Control

Low pass rates are only informative if failures come from the agent rather than from the benchmark. We ensure this with automatic checks and with a review of all 56 environments by seven human experts.

Automatic checks. Programs check every environment when it is generated. Every requirement check must fail on the initial state and every violation check must pass on it, so no check is settled before the agent acts. The text of every event must contain each of its facts word for word. Checks that repeat the same content for different people or records, such as one email sent to several recipients, are merged into one composite check, so repetition does not inflate the check count. Another program checks the handbook. It mentions only roles, channels, sheets, and tools that exist in the company, every approval amount it states matches the limits set in General/approvals.md, and every cross-reference points to an existing section.

Task validity and Solvability. The experts cross-reviewed every environment to confirm that the correct handling of each task follows from the company handbook and common workplace sense. No task requires knowledge that the agent cannot obtain from the environment. The experts also completed all 56 environments themselves using the same tools available to the agent. Every environment was solved by a human, so no task is infeasible.

Every project can be finished well before the deadline, but only with timely follow-up. A reference solver that dispatches and relays without delay and follows up as soon as each promised time passes finishes all 56 projects by 16:29. When it follows up 45 minutes late, it misses the 18:00 deadline in 37 of the 56 projects (Appendix B).

The experts verified check outcomes against the agent's actual actions, one check at a time. The standard for each judgment was what a competent employee following the handbook would do.

3 Experiments

3.1 Setup

We evaluate 12 models on all 56 environments: Claude Fable 5.1, Claude Opus 5.5, GPT-6 Astra, GPT-6 Sol, Gemini 3.8 Flash, Qwen3.8-Max, Kimi-K3, GLM-5.3, DeepSeek-V4.1-Flash, MiniMax-M3, MiMo-V2.6-Pro, and Hy4-preview. The agent receives a 6-line system prompt (Appendix F) that identifies it as the company assistant, points it to the handbook in Google Drive, and explains the simulated clock. No in-context examples, chain-of-thought instructions, or task-specific guidance are provided. All models receive the same 106 tools. All models run in the same agent loop, which sends the full conversation history on every turn without compression (Appendix B).

Table 2: Main results on ORGWORLD (56 environments, both tracks running concurrently). Pass@1 is the percentage of environments in which a track passes: all of its requirement checks pass, none of its violation checks trigger, and it makes no unrequested record change. "Both" requires both tracks to pass in the same run. "Checks" is the percentage of checks satisfied, where each unrequested record change counts as one failed check. Cost is in US dollars. Within each group, models are ordered by Both Pass@1, the main metric (shaded). Bold marks the best value in each column.

Model	Both		Assistance (A)		Coordination (B)		Cost ($)	
	Pass@1	Checks	Pass@1	Checks	Pass@1	Checks	Per env	Total
Closed-weight models								
Claude Fable 5.1 (medium)	**48.2**	**92.9**	**51.8**	**88.5**	89.3	98.2	7.31	409.45
Claude Opus 5.5 (medium)	19.6	88.6	23.2	80.4	**91.1**	98.8	2.20	123.13
GPT-6 Astra (medium)	14.3	86.3	14.3	76.3	**91.1**	**98.9**	7.61	426.21
Gemini 3.8 Flash (medium)	3.6	64.5	8.9	59.2	17.9	70.9	1.50	83.79
GPT-6 Sol (medium)	0.0	73.3	3.6	63.8	39.3	85.0	1.83	102.73
Open-weight models								
MiMo-V2.6-Pro	12.5	86.1	32.1	85.3	35.7	87.2	0.16	8.88
Qwen3.8-Max	10.7	85.5	21.4	82.6	46.4	89.1	2.53	141.70
GLM-5.3 (max)	8.9	84.3	26.8	82.5	25.0	86.5	3.68	205.98
Kimi-K3 (max)	8.9	82.5	23.2	79.8	37.5	85.8	3.92	219.36
Hy4-preview (high)	0.0	75.6	10.7	77.6	10.7	73.1	0.68	38.00
DeepSeek-V4.1-Flash (high)	0.0	73.6	8.9	78.3	16.1	67.7	0.09	5.16
MiniMax-M3	0.0	42.9	0.0	54.6	1.8	28.4	0.58	32.75

Every run continues until the simulated clock reaches 18:00, the end of the workday. The reasoning effort used for each model is given in parentheses after its name in Table 2 and Figure 4. MiMo-V2.6-Pro, Qwen3.8-Max, and MiniMax-M3 offer no effort levels, only thinking on or off, and we run them with thinking on. Seven of the models have open weights: MiMo-V2.6-Pro, Qwen3.8-Max, GLM-5.3, Kimi-K3, Hy4-preview, DeepSeek-V4.1-Flash, and MiniMax-M3. The other five are available only through APIs. Each model runs each environment once, and we report pass@1 for each track and for both tracks together. We also report the API cost of each run.

3.2 MAIN RESULTS

Claude Fable 5.1 is the strongest model by a wide margin, passing both tracks in 27 of 56 environments (48.2%). Table 2 reports agents' reliability on both tracks. The next best model, Claude Opus 5.5, passes both tracks in 11 environments (19.6%). Four of the 12 models do not pass both tracks in any environment, and in 19 environments no model passes both tracks. The check pass rate is much higher. Claude Fable 5.1 satisfies 92.9% of checks, and seven of the 12 models satisfy more than 82%. An environment, however, passes only when all of its checks in both tracks pass, about 46 per environment on average. The failures of Claude Fable 5.1 are rarely a single slip. In its median failed environment, it misses 4 checks, and only 5 of its 29 failed environments miss 2 or fewer. Errors do not fall at random. If each requirement check of Claude Fable 5.1 passed independently at its requirement-check pass rate of 93.0%, it would satisfy all requirement checks in about 3.5 of the 56 environments; it does so in 27. For each of the other nine models that satisfy all requirement checks in at least one environment, the actual count is more than ten times the count expected under independence.

Each model runs each environment once. Figure 7 in Appendix I shows 95% Wilson intervals for pass@1 over the 56 environments. On both tracks together, the interval of Claude Fable 5.1 (35.7% to 61.0%) does not overlap with the interval of any other model. The intervals of the next six models all overlap with one another.

The coordination track has a higher pass rate than the assistance track for 10 of the 12 models. The three strongest models pass the coordination track in 89.3% to 91.1% of environments, but pass

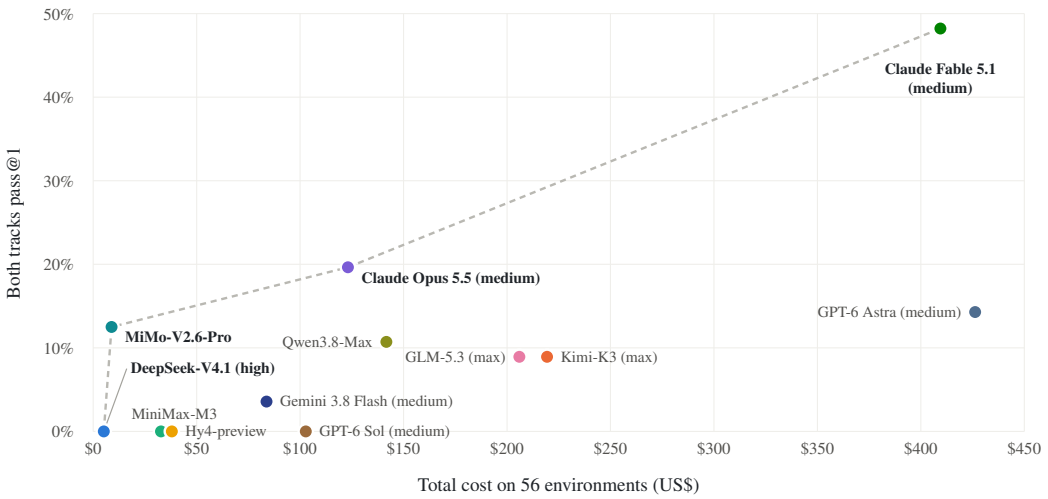


Figure 4: Total API cost on the 56 environments against the rate of passing both tracks. Each point is one model. The dashed line connects the Pareto frontier (bold labels): no other model both costs less and passes more environments than a model on it.

the assistance track in only 14.3% to 51.8%. The same pattern holds per environment (Figure 5 in Appendix G). In 10 of the 56 environments, no model passes the assistance track, while every coordination track is passed by at least one model. In 9 of these 10 environments, no check is failed by all 12 models, so the models fail for different reasons. In the remaining environment, all 12 models fail the same five checks about a weekly payment run, but for three different reasons: six models never handle the renewals that feed the run, five set the renewal bills to fall due 30 days later instead of on the renewal dates, and one pays the wrong bills (Appendix H).

Cost and performance are only loosely related (Figure 4). Claude Fable 5.1 costs $7.31 per environment, and GPT-6 Astra costs about the same ($7.61) but passes both tracks in only 14.3% of environments. MiMo-V2.6-Pro costs $0.16 per environment and passes both tracks in 12.5% of environments, more than Kimi-K3 and GLM-5.3, which each cost more than 20 times as much. Figure 12 in Appendix I counts the violation checks each model triggers and the unrequested record changes it makes. Claude Fable 5.1 triggers 2 of the 419 violation checks, and GPT-6 Sol triggers 21, the most of any model. GPT-6 Astra makes 18 unrequested record changes, more than any other model.

Most models recognize that they are in a simulation. Six of the seven models whose reasoning we can read call the environment a simulation in 42 to 56 of their 56 runs, and mention a grader in up to 41 runs (Table 6 in Appendix I). Appendix I also gives results by domain and tool use, and Appendix H compares two runs on one environment along the simulated workday.

3.3 Analysis by Demand

(a) Task interpretation and responsibility discovery

Unannounced work accounts for 22% of assistance-track requirement checks but 42% of their failures. To examine responsibility discovery, we label each requirement check of the assistance track by how its work reaches the agent: by an email, a Slack message, or a support ticket; by a row added to a sheet during the day; or by a date already in the records when the day starts. Across the 12 models, 17% of the checks about announced work fail, compared with 41% for new rows and 50% for due dates. For Claude Fable 5.1, two thirds of the failed checks concern such work. The agent must first notice the work. We check the tool calls of the 11 models whose calls carry simulated times. In the 147 cases where a model does not read a sheet after a row arrives in it, the checks about that row never all pass. When the model does read the sheet, they all pass in 71 of 106 cases. When a model never opens the sheet that holds a due date, the checks about that item all pass

in only 13 of 129 cases. Models differ widely in this habit. GPT-6 Astra reads the sheet after 22 of the 23 new rows, Claude Fable 5.1 after 12, and Gemini 3.8 Flash and MiniMax-M3 after 3.

Noticing work is not sufficient: the agent must also choose the right procedure. In environments 116 and 117, 9 and 10 models fail to record an office-supply order that names no vendor; Claude Fable 5.1 treats it as a purchase request, asks the requester for a vendor that the order does not need, and never records the order (Appendix H).

Handoffs can succeed while downstream work remains unfinished. ORGWORLD includes workflows that span departments and systems: completing an upstream step can create further obligations, and the handbook specifies how to carry them through. A successful handoff therefore does not by itself complete the workflow. For example, in environment 122, all 12 models post a new hire in `#it` on behalf of HR, but only three then set up the accounts and devices on behalf of IT. The remaining models complete the HR notification but leave the required IT work unfinished.

(C) TEMPORAL PLANNING AND COORDINATION

Most models assign the correct owner, but differ in dispatch and follow-up timing. Three closed-weight models pass the coordination track in 89.3% to 91.1% of environments, while the best open-weight model, Qwen3.8-Max, passes it in 46.4% (Figure 13 in Appendix I). 11 of the 12 models assign at least 86% of subtasks to the correct owner (Figure 14 in Appendix I). Figure 15 in Appendix I shows when the requirement checks of each track are first satisfied during the simulated workday. The coordination curves separate early. Claude Fable 5.1, Claude Opus 5.5, GPT-6 Astra, and GPT-6 Sol satisfy 26.2% to 28.5% of coordination checks by 10:00, and the other eight models satisfy 14.1% to 20.8%. These four models dispatch 89% to 96% of all subtasks by 10:00, while the other eight dispatch 47% to 71%. The assistance curves of all models keep rising until the end of the workday. For nine of the 12 models, subtasks owned by colleagues who miss their promised time once have the lowest on-time rate, and these models also send enough follow-ups to such colleagues less often than to colleagues who miss it twice (Figure 16 in Appendix I).

(D) CONCURRENT RESPONSIBILITY MANAGEMENT

Concurrency lowers assistance-track pass rates in both tested models. In the ablation on Kimi-K3 and DeepSeek-V4.1-Flash, adding the coordination track lowers these rates from 33.9% to 23.2% and from 30.4% to 8.9%, relative drops of 32% and 71% (Appendix J). Coordination pass rates do not change consistently: Kimi-K3 declines, whereas DeepSeek-V4.1-Flash improves (Appendix J).

4 CONCLUSION

We introduce ORGWORLD, a benchmark that evaluates an LLM agent over a full workday rather than on isolated, well-specified tasks, in which the agent must discover its responsibilities, reprioritize as the day changes, coordinate with colleagues over time, and keep several responsibilities moving at once. Tool calls rarely fail: fewer than 3.5% return an error, and Claude Fable 5.1 satisfies 92.9% of all checks, yet it passes only 48.2% of environments. Its failures concentrate on noticing unannounced work, choosing among similar-looking procedures, and carrying workflows through to their follow-up steps. In an ablation on two models, adding the coordination track lowers the assistance-track pass rate by 32% and 71%. ORGWORLD therefore shifts the evaluation target from isolated task completion to reliable, unsupervised organizational work, and points future work toward *proactive information tracking, dependency tracking, and scheduling*. We release the environment, handbook, and evaluation harness to support future research.

Limitations. Colleagues follow fixed scripts. This makes results reproducible and difficulty controllable, but colleagues do not respond beyond their scripts. Each model runs each environment once, so whether a single environment passes carries chance; Section 3.2 gives confidence intervals for the pass rates. The handbook and all communication are in English.

Reproducibility Statement

The environment generator produces deterministic outputs from fixed seeds. The world server is a self-contained Python process with no external dependencies beyond a standard Python installation. The evaluation harness uses the OpenAI-compatible chat completions API, making it compatible with any model that supports function calling. The scoring module evaluates checks against the final world state with no stochastic components. We will release all code, environments, and evaluation scripts as an open-source repository.

Ethics Statement

ORGWorld simulates a fictional company with generated names, addresses, and business data. No real personal information, company data, or proprietary processes are used. The benchmark evaluates tool-calling LLMs in a sandboxed environment with no access to real systems. We do not foresee direct negative societal impacts from this work, though we note that improvements in organizational agent capabilities could affect employment in administrative roles.

AI Use Statement

We used generative AI to write the natural-language text of the events in each environment (emails, Slack messages, and support tickets) from lists of facts. A program checked that each text contains every fact word for word (Section 2.5).

References

Anthropic. Model Context Protocol, 2024. URL https://modelcontextprotocol.io.

Axel Backlund and Lukas Petersson. Vending-Bench: A benchmark for long-term coherence of autonomous agents. *arXiv preprint arXiv:2502.15840*, 2025.

Léo Boisvert, Megh Thakkar, Maxime Gasse, Massimo Caccia, Thibault L 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*, volume 37, 2024.

Haozhe Chen, Karthik Narasimhan, and Zhuang Liu. CEO-Bench: Can agents play the long game? *arXiv preprint arXiv:2606.18543*, 2026.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, et al. AgentVerse: Facilitating multi-agent collaboration and exploring emergent behaviors. In *International Conference on Learning Representations*, 2024.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2Web: Towards a generalist agent for the web. In *Advances in Neural Information Processing Systems*, 2023.

Romain Froger, Pierre Andrews, Matteo Bettini, Amar Budhiraja, Ricardo Cabral, Virginie Do, Emilien Garreau, Jean-Baptiste Gaya, Hugo Laurençon, Maxime Lecanu, et al. Gaia2: Benchmarking LLM agents on dynamic and asynchronous environments. In *International Conference on Learning Representations*, 2026.

Daocheng Fu, Jianbiao Mei, Rong Wu, Xuemeng Yang, Jia Xu, Ding Wang, Pinlong Cai, Yong Liu, Licheng Wen, and Botian Shi. The agent's first day: Benchmarking learning, exploration, and scheduling in the workplace scenarios. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 30094–30109, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.1505. URL https://aclanthology.org/2026.findings-acl.1505/.

Muyu He, Adit Jain, Anand Kumar, Vincent Tu, Soumyadeep Bakshi, Sachin Patro, and Nazneen Rajani. YC-Bench: Benchmarking AI agents for long-term planning and consistent execution. *arXiv preprint arXiv:2604.01212*, 2026.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Ceyao Zhang, Jinlin Wang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, et al. MetaGPT: Meta programming for a multi-agent collaborative framework. In *International Conference on Learning Representations*, 2024.

Abubakarr Jaye, Nigel Boachie Kumankumah, Chidera Biringa, Anjel Shaileshbhai Patel, Sulaiman Vesal, Dayquan Julienne, Charlotte Siska, Manuel Raúl Meléndez Luján, Anthony Twum-Barimah, Mauricio Velazco, et al. CORPGEN: Simulating corporate environments with autonomous digital employees in multi-horizon task environments. *arXiv preprint arXiv:2602.14229*, 2026.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations*, 2024.

Xiangyi Li, Kyoung Whan Choe, Yimin Liu, Xiaokun Chen, Chujun Tao, Bingran You, Wenbo Chen, Zonglin Di, Jiankai Sun, Shenghan Zheng, et al. ClawsBench: Evaluating capability and safety of LLM productivity agents in simulated workspaces. *arXiv preprint arXiv:2604.05172*, 2026.

Xiaoxi Li, Jiajie Jin, Guanting Dong, Hongjin Qian, Yongkang Wu, Ji-Rong Wen, Yutao Zhu, and Zhicheng Dou. WebThinker: Empowering large reasoning models with deep research capability. *arXiv preprint arXiv:2504.21776*, 2025.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. AgentBench: Evaluating LLMs as agents. In *International Conference on Learning Representations*, 2024.

Yaxi Lu, Shenzhi Yang, Cheng Qian, Guirong Chen, Qinyu Luo, Yesai Wu, Huadong Wang, Xin Cong, Zhong Zhang, Yankai Lin, et al. Proactive agent: Shifting LLM agents from reactive responses to active assistance. In *International Conference on Learning Representations*, 2025.

Fanqing Meng, Lingxiao Du, Zijian Wu, Guanzheng Chen, Xiangyan Liu, Jiaqi Liao, Chonghe Jiang, Zhenglin Wan, Jiawei Gu, Pengfei Zhou, et al. ClawMark: A living-world benchmark for multi-turn, multi-day, multimodal coworker agents. *arXiv preprint arXiv:2604.23781*, 2026.

Grégoire Mialon, Clémentine Fourrier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. GAIA: A benchmark for general AI assistants. In *International Conference on Learning Representations*, 2024.

Joon Sung Park, Joseph C O'Brien, Carrie J Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *ACM Symposium on User Interface Software and Technology*, 2023.

Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E Gonzalez. Gorilla: Large language model connected with massive APIs. *arXiv preprint arXiv:2305.15334*, 2023.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. In *International Conference on Learning Representations*, 2024.

Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, et al. UI-TARS: Pioneering automated GUI interaction with native agents. *arXiv preprint arXiv:2501.12326*, 2025.

Daniel Shepard and Robin Salimans. AutomationBench. *arXiv preprint arXiv:2604.18934*, 2026.

Olly Styles, Sam Miller, Patricio Cerda-Mardini, Tanaya Guha, Victor Sanchez, and Bertie Vidgen. WorkBench: a benchmark dataset for agents in a realistic workplace setting. In *Conference on Language Modeling*, 2024.

Qiaoyu Tang, Ziliang Deng, Hongyu Lin, Xianpei Han, Qiao Liang, Boxi Cao, and Le Sun. ToolAlpaca: Generalized tool learning for language models with 3000 simulated cases. *arXiv preprint arXiv:2306.05301*, 2023.

Aleksandra Todorova. How Anthropic employees use Claude Tag. Anthropic Blog, August 2026. URL https://claude.com/blog/how-anthropic-employees-use-claude-tag.

Tongyi DeepResearch Team. Tongyi DeepResearch technical report. *arXiv preprint arXiv:2510.24701*, 2025.

Xingyao Wang, Zihan Wang, Jiateng Liu, Yangyi Chen, Lifan Yuan, Hao Peng, and Heng Ji. MINT: Evaluating LLMs in multi-turn interaction with tools and language feedback. In *International Conference on Learning Representations*, 2024.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, et al. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *Advances in Neural Information Processing Systems*, 2024.

Frank Fangzheng Xu, Yufan Song, Boxuan Li, Yuxuan Tang, Kritanjali Jain, Mengxue Bao, Zora Wang, Xuhui Zhou, Zhitong Guo, Murong Cao, et al. TheAgentCompany: Benchmarking LLM agents on consequential real world tasks. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

John Yang, Carlos E Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. SWE-agent: Agent-computer interfaces enable automated software engineering. In *Advances in Neural Information Processing Systems*, 2024.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. In *International Conference on Learning Representations*, 2025.

Tao Yu, Hao Wang, Changyu Li, Shenghua Chai, Minghui Zhang, Zhongtian Luo, Yuxuan Zhou, Haopeng Jin, Zhaolu Kang, Jiabing Yang, et al. Beyond the all-in-one agent: Benchmarking role-specialized multi-agent collaboration in enterprise workflows. *arXiv preprint arXiv:2605.08761*, 2026.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. WebArena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, 2024.

Yuchen Zhuang, Yue Yu, Kuan Wang, Haotian Sun, and Chao Zhang. ToolQA: A dataset for LLM question answering with external tools. In *Advances in Neural Information Processing Systems*, 2023.

A Related Work

Tool-use benchmarks. ToolBench (Qin et al., 2024), Gorilla (Patil et al., 2023), and ToolAlpaca (Tang et al., 2023) evaluate whether a model can select the correct APIs and produce valid arguments. ToolBench also includes instructions that chain calls across several tools. Each instance is still a standalone user query with no organizational context. ToolQA (Zhuang et al., 2023) evaluates question answering that requires external tools.

Interactive environment benchmarks. WebArena (Zhou et al., 2024) and OSWorld (Xie et al., 2024) place agents in executable web or desktop environments and evaluate multi-step interaction. Mind2Web (Deng et al., 2023) collects action sequences for tasks on real websites. These benchmarks test navigation and UI manipulation and present one task at a time. τ-bench (Yao et al., 2025) introduces tool-agent-user interaction in retail and airline domains but limits scope to a single conversation with a single customer.

Agent benchmarks. AgentBench (Liu et al., 2024) unifies eight environments into a single evaluation framework, and GAIA (Mialon et al., 2024) tests multi-step reasoning with web tools. These benchmarks evaluate broad capabilities but each instance is still a standalone problem. SWE-bench (Jimenez et al., 2024) evaluates code editing in real repositories but does not test organizational workflows or collaboration. MINT (Wang et al., 2024) tests multi-turn tool use with language feedback but in isolated problem-solving contexts.

Workplace agent benchmarks. WorkBench (Styles et al., 2024) evaluates common workplace tasks, such as sending emails and scheduling meetings, in a sandbox of five databases and scores each task by the resulting database state. ClawsBench (Li et al., 2026) runs tasks on mock Gmail, Slack, Google Calendar, Google Docs, and Google Drive services and also measures how often agents take unsafe actions. AutomationBench (Shepard & Salimans, 2026) evaluates cross-application business workflows over the REST APIs of 47 apps. The agent must discover the endpoints itself, follow the policy documents that the task prompt points to, and avoid actions that the checks forbid. Each task is a single instruction and runs without further interaction until it is done. WorkArena++ (Boisvert et al., 2024) evaluates 682 workflows of knowledge workers on an enterprise software platform, with each instance still a single task. TheAgentCompany (Xu et al., 2025) evaluates 175 workplace tasks in a self-hosted simulated software company whose colleagues are played by LLMs. EntCollabBench (Yu et al., 2026) simulates an organization with six departments in which each of 11 roles is a separate agent, and evaluates how these agents delegate work and pass context to each other. In GAIA2 (Froger et al., 2026), the environment changes independently of the agent's actions, and the agent must handle dynamic events and ambiguous requests under time constraints and collaborate with other agents. ClawMark (Meng et al., 2026) evaluates coworker agents over multiple working days, during which emails, calendar entries, and records change between turns independently of the agent, and finds that performance drops after the first such change. The Agent's First Day (Fu et al., 2026) evaluates how an agent schedules streaming tasks with different priorities, explores a new environment, and learns from experience. CorpGen (Jaye et al., 2026) studies agents that manage dozens of interleaved long-horizon tasks and proposes an agent framework for them. Each of these benchmarks captures some of the four demands of organizational work. ORGWORLD requires all four in one organization over a full workday. The agent must discover obligations that nobody announces (task interpretation and responsibility discovery), revise its plan as events arrive (dynamic prioritization and replanning), time its follow-ups to colleagues' promised times (temporal planning and coordination), and keep an assistance track and a coordination track moving together (concurrent responsibility management).

Proactive agents. Proactive Agent (Lu et al., 2025) builds ProactiveBench to evaluate whether an agent anticipates and offers help from observed user activity without explicit instructions. In ORGWORLD, part of the work likewise arrives without any request. It appears only as new rows or due dates in the business records, and only the handbook says what they require.

Long-horizon business simulations. Vending-Bench (Backlund & Petersson, 2025) has an agent operate a vending machine business over a long horizon. YC-Bench (He et al., 2026) runs a simulated startup for one year, and CEO-Bench (Chen et al., 2026) runs one for 500 days. These benchmarks emphasize long-horizon business operation and economic outcomes. ORGWORLD instead focuses on whether an agent fulfills an employee's procedural and collaborative responsibilities within a simulated organizational workday.

Multi-agent simulation. Generative Agents (Park et al., 2023) simulates a town of 25 agents with emergent social behavior, and MetaGPT (Hong et al., 2024) and AgentVerse (Chen et al., 2024) coordinate multiple LLM agents on shared tasks. These works focus on multi-agent cooperation or emergent dynamics. ORGWORLD takes a different perspective: we evaluate a single agent embedded in an organization of scripted colleagues, testing whether it can function as a reliable coworker rather than whether multiple agents can coordinate.

Table 3 compares ORGWORLD with existing benchmarks along key dimensions.

Table 3: Comparison of ORGWORLD with existing agent benchmarks. Multi-step: an instance needs a sequence of dependent actions. Concurrent: one run requires several responsibilities to progress at the same time. Collaboration: the agent must work with other simulated people or agents. Policy discovery: the rules the agent must follow are not stated in the task or the prompt and must be found in documents in the environment. TheAgentCompany is marked Partial because its environment includes an employee handbook, but the paper does not describe tasks that require finding rules in it. Violation checks: the evaluation fails the agent when it takes an action it should not take.

Benchmark	Systems	Tasks	Multi-step	Concurrent	Collaboration	Policy discovery	Violation checks
ToolBench	API only	126K+	✓	×	×	×	×
WebArena	4 web apps	812	✓	×	×	×	×
OSWorld	OS-level	369	✓	×	×	×	×
GAIA	Web tools	466	✓	×	×	×	×
τ-bench	2 domains	165	✓	×	1-on-1	×	✓
TheAgentCompany	Self-hosted apps	175	✓	×	LLM colleagues	Partial	×
GAIA2	12 mobile apps	800	✓	×	App agents	×	✓
ClawMark	5 services	100	✓	×	×	×	✓
CorpGen	Office apps	46	✓	✓	×	×	×
ORGWORLD	9 enterprise	56 envs	✓	✓	15 colleagues	127 rules	419

B ENVIRONMENT DETAILS

Names. Company names are randomized from combinatorial word pools, and employee names are drawn from nine cultural backgrounds to ensure diversity.

Enterprise systems. Each of the nine systems is implemented as a stateful module with a realistic API: Gmail (send, search, read, reply), Slack (post in channels, reply in threads, read history), Google Sheets (read, write, append rows), Google Drive (list, read, create documents), Salesforce (manage leads, opportunities, contacts, accounts), Zendesk (create and update tickets, add comments), Asana (create and manage tasks and projects), QuickBooks (invoices, bills, payments, journal entries), and BambooHR (employee records, leave balances, time-off requests). Six systems are adapted from AutomationBench (Shepard & Salimans, 2026) and three (Asana, BambooHR, Google Drive) are implemented from scratch. No tool has side effects outside the simulation.

Colleague rules. A colleague who owns two subtasks works on one at a time, starting with the one that became ready first, so the order of dispatch affects when both are done. A downstream colleague starts only after the agent passes on the upstream result. The CEO's brief asks the agent to pass each result on in the exact words it was given, so that nothing changes between teams. If the agent assigns a subtask to the other member of the owner's department, that person redirects the agent to the correct owner after a delay.

Infrastructure. The evaluation harness consists of three components. The *world server* is an MCP server (Anthropic, 2024) that communicates with the agent via stdio JSON-RPC, maintaining the complete state of all nine systems, the event queue, the colleague scripts, and the simulated clock. The *agent loop* is a minimal while-loop that drives the agent through an OpenAI-compatible chat completions API, sending the full conversation history with no context compression. The *scoring module* runs after the workday ends, comparing the final world state against all checks, with additional handling of deadline enforcement, ordering constraints, and transient state.

Counting violation checks. The scoring module evaluates every check on the initial world state as well as on the final one. A violation check always passes on the initial state, because the agent has not yet done anything it should not do. If it still passes on the final state, it is left out of the check count. A model therefore earns nothing for avoiding a forbidden action and cannot raise its check pass rate by doing nothing. If it fails on the final state, it counts as one failed check. Requirement checks always fail on the initial state (Section 2.5), so every requirement check is counted.

Timing. We run a reference solver that dispatches and relays without delay and varies only when it follows up. For a silent colleague, a follow-up is due 30 minutes after dispatch. When the solver follows up as soon as each promised time passes, it finishes all 56 projects between 15:00 and 16:29. When it follows up 30 minutes late, it still finishes all 56 projects by 17:38. It still finishes all 56 by 17:59 when, in addition, it first sends every subtask to the wrong member of the owner's department. When it follows up 45 minutes late, it misses the 18:00 deadline in 37 of the 56 projects. When it follows up 60 minutes late or never follows up, it misses the deadline in all 56 projects.

C HANDBOOK STRUCTURE

The company handbook consists of 29 documents organized in 8 folders, plus 14 email templates.

Table 4: Company handbook structure.

Folder	Topics	Docs
General	Approvals, how work reaches you	2
Finance	Accounts payable/receivable, close, expenses	4
HR	Compensation, compliance, hiring, leave, onboarding	5
IT	Accounts and access, devices, incidents	3
Marketing	Campaigns, content, leads, partners	4
Operations	Facilities, purchasing, vendors	3
Sales	Customers, deals, leads, reporting	4
Support	Customers, refunds, reporting, tickets	4
Templates	Standard email templates	14

The General/approvals.md document serves as the single source of truth for spending limits:

> Up to $2,000: assistant may approve autonomously.
> $2,000.01–$10,000: department manager approval required.
> Above $10,000: CEO approval required.

Four exceptions override these limits: wire transfers always require CEO approval regardless of amount, salary decisions always require the HR Manager, customer discounts follow a separate percentage-based scale, and payroll items always require the HR Manager.

D EXAMPLE ENVIRONMENT WALKTHROUGH

We illustrate the benchmark with a walkthrough of environment 101, which combines the assistance problem "new_customer" (Meridian Interiors sales pipeline) with the coordination project "customer_visit" (Driftwood Media visit preparation).

Setup. The company is Kestrel Anchor Associates with 15 employees. At 09:06, the CEO (Camille Weber) posts in `#general` assigning a multi-department project to prepare for a visit from Driftwood Media (coordination track). At 09:07, an email arrives from Ming Zhou at Meridian Interiors asking about pricing and a demo (assistance track). Both tracks run concurrently for the rest of the workday.

Assistance track (25 requirement checks, 1 violation check). The email triggers a multi-step sales workflow spanning six systems. The agent must: (1) check the Competitors and Do Not Contact sheets (Meridian Interiors is on neither); (2) create a Salesforce lead for Ming Zhou with Source=Inbound and Owner=sales_rep; (3) post in `#sales` to notify the Sales Representative (Dina Haddad); (4) reply to Ming Zhou using the Inbound Reply template, citing the demo date Dina proposes in her thread reply; (5) when Dina posts that the opportunity has advanced to Proposal, update the Salesforce stage; (6) when Dina shares meeting notes with two action items, create Asana tasks for the assignees (Marketing Specialist and Operations Manager), add a Salesforce note, and email the meeting summary to Ming Zhou; (7) when Dina requests a 20% discount, recognize that

20% falls in the 10–25% range requiring Sales Manager approval, post the approval request to Jonas Nielsen in `#sales`, and wait for his thread reply before sending the quote email; (8) when Ming Zhou replies with a signed order ($3,128 after discount), update the opportunity to "Closed Won," create a QuickBooks customer and invoice (following the INV-NNNN numbering convention), set up the customer in Zendesk, add a row to the Customers sheet (Tier=Standard, Credit Limit=5000), send a welcome email, and notify Finance and Support via their respective Slack channels.

Coordination track (25 requirement checks). The CEO's project brief assigns 7 subtasks across 6 departments with a dependency graph: guest_list (Sales Manager, late-twice), pricing_memo (Finance Manager, silent, 2 follow-ups), credit_check (Accountant, punctual), ticket_summary (Support Manager, punctual), room_booking (Operations Manager, late-once, depends on guest_list), visitor_access (IT Specialist, late-twice, depends on guest_list), and pitch_deck (Marketing Manager, late-once, depends on guest_list + pricing_memo + ticket_summary). Each deliverable carries a specific token (e.g., "6 guests," "8% max discount," "Room Cedar") that the downstream colleague checks for verbatim. The final check verifies that the agent's summary reply to the CEO contains all seven tokens.

E Benchmark Statistics

Table 5 breaks down the checks by enterprise system.

Table 5: Check counts by enterprise system across all 56 environments. The assistance track contributes 1,184 requirement checks and 419 violation checks; the coordination track contributes 973 requirement checks and no violation checks.

System	Requirement	Violation	Example check types
Slack	381	112	message posted, thread reply, absent
Sheets	249	114	cell value, row exists/absent, row count
Gmail	247	85	email sent, reply sent, not sent
Salesforce	61	28	field value, lead/opp exists, record absent
Asana	63	26	task exists, task absent, task count
Zendesk	62	6	ticket field, reply contains
QuickBooks	39	29	bill/invoice/payment exists, status
BambooHR	25	13	field value, employee absent
Coordination	910	0	assigned, delivered, urged, relayed
Composite	120	6	all-of and any-of groups
Total	**2,157**	**419**	

F System Prompt

The complete system prompt given to the agent consists of six sentences:

> You are the company assistant for [Company Name] (Slack user U_ASSISTANT, email assistant@[domain]).
>
> All company work happens through the tools available to you; no human is chatting with you, so never ask questions – decide for yourself based on the handbook.
>
> The company handbook lives in the Google Drive "Company Handbook" folder (department subfolders plus Templates). Before doing anything, read the relevant handbook section and follow it strictly.
>
> Time runs on a simulated clock: call clock_now to check the time and clock_advance(minutes) to move it forward.
>
> Advance the clock whenever you are waiting for a reply or have nothing else to do; newly arrived emails, messages, and tickets are appended to tool results under a "NEW SINCE YOUR LAST CALL" section.
>
> Your work window runs from 2026-09-03T09:00:00Z to 2026-09-03T18:00:00Z; keep advancing the clock until the window ends, do not stop early.

The opening user message is: "It is [day of week] [date] [time]. You are on duty." Between turns, the harness sends: "It is now [clock time]. Continue."

G Per-Environment Results

Figure 5 shows the per-track results of all 12 models on all 56 environments in the concurrent setting.

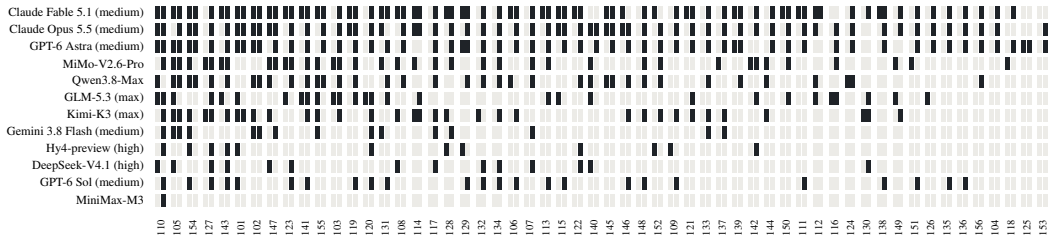


Figure 5: Per-environment results of all 12 models (concurrent setting). Each cell is one model on one environment. The left half is the assistance track and the right half is the coordination track. A dark half means the track passed. Models are ordered by the rate of passing both tracks. Environments are ordered by the number of passed tracks across all models, from most to fewest.

H Case Studies

A due date that is the renewal date (environment 153). Three software subscriptions renew this week, and nothing reminds the agent. Each renewal bill falls due on its renewal date. At 12:00 the accountant asks for the weekly payment run. The correct run pays two of the renewals and leaves the third pending, because that vendor's wire details have not been verified. All 12 models fail the five checks about this payment run. Six models never handle the renewals. Five set the renewal bills to fall due 30 days later instead of on the renewal dates. They apply the handbook's default for bills that state no due date, although each renewal bill falls due on its renewal date. DeepSeek-V4.1-Flash sets the due dates correctly but pays the renewal with unverified wire details and leaves one of the other two unpaid.

A request without a vendor (environments 116 and 117). The handbook separates two ways to buy things. A purchase request names a vendor and goes through a purchase order. An office-supply order names only the item, the amount, and the department, and is recorded on the Supply Orders sheet against the department's supply budget. The records the company kept before the workday (Section 2.1) follow the same split: every row of the Purchase Requests sheet has a vendor, and the Supply Orders sheet has no vendor column. In environment 116, a support specialist asks for a standing desk of about $552 to be paid from the department budget, without naming a vendor. The correct handling is to record it on the Supply Orders sheet and tell the requester that the support supply budget is now 82% used. Claude Fable 5.1 applies the wrong procedure. It treats the email as a purchase request and asks the requester for a vendor, which an office-supply order does not need. Its reasoning reads: "Daiki Kato's purchase request is missing the vendor, which the handbook requires as part of a valid request." It never records the order on the Supply Orders sheet. Claude Opus 5.5 and GPT-6 Astra also fail these two checks, while MiMo-V2.6-Pro, GLM-5.3, and DeepSeek-V4.1-Flash pass them. At 14:40, more than three hours later, the operations specialist asks to order a projector from a named vendor. Of the nine models that do not record the office-supply order in environment 116, eight handle this purchase request in full: they get the operations manager's approval, raise a purchase order, and send it to the vendor. These models can follow the purchasing procedure; their error is choosing it for the wrong request. Environment 117 has a similar request for a space heater, and only Claude Opus 5.5 and DeepSeek-V4.1-Flash record it.

Following up in the wrong thread (environment 152). The handbook says: "When you follow up on something you already asked someone for, reply in the thread where you asked and tag them again." Claude Fable 5.1 dispatches a subtask to the sales manager, who acknowledges it in a

thread reply. Claude Fable 5.1 then posts ten follow-ups between 11:22 and 17:35, all as replies to the manager's acknowledgment instead of in the thread of its own request, against the handbook. Because it never follows up where the handbook requires, the sales manager never delivers, and the coordination track fails. Five models, including Claude Opus 5.5 and GPT-6 Astra, pass the coordination track of this environment.

Recovering from a wrong assignment (environment 101). Claude Fable 5.1 dispatches all seven subtasks at 09:06 but sends two of them to the wrong member of the owner's department. At 09:26, both colleagues reply that the work belongs to someone else, and Claude Fable 5.1 reassigns the subtasks within the same minute. It later follows up the marketing manager a second time at 16:26, although her script needs only one follow-up, and receives the reply "I'm on the pitch deck. Pinging me again won't make it faster." The last subtask arrives at 17:22, and Claude Fable 5.1 reports all seven results to the CEO at 17:23, passing both tracks.

Two runs on one environment (environment 138). Figure 6 compares Claude Fable 5.1 and GLM-5.3 on environment 138, whose 10 events arrive between 09:03 and 14:53. Claude Fable 5.1 passes both tracks with 240 tool calls. GLM-5.3 fails 6 assistance checks and 3 coordination checks with 155 tool calls.

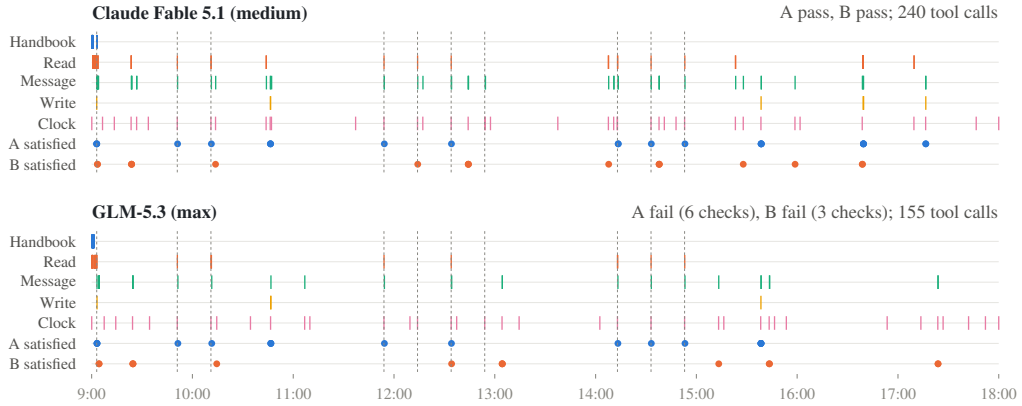


Figure 6: Timeline of two runs on environment 138. Each tick is one tool call, placed at its simulated time in the lane of its kind. Dots mark the first time each requirement check is satisfied. Dashed lines mark the arrival of events.

I Additional Analyses

This section analyzes the same runs as Table 2. Figure 19 uses the simulated time of each tool call. These times were kept for only 5 of the 56 Kimi-K3 runs, so Figure 19 leaves out Kimi-K3.

Uncertainty. Figure 7 shows the 95% Wilson intervals discussed in Section 3.2.

Size of failures. Figure 8 shows how many checks each model fails in each environment where it does not pass both tracks. Unrequested record changes count as failed checks, as in Table 2. The median is 4 for Claude Fable 5.1, Claude Opus 5.5, and GPT-6 Astra, 14 for Gemini 3.8 Flash, and 21.5 for MiniMax-M3.

Environment difficulty. Figure 9 counts, for each environment, how many models pass each track. In the assistance track, 10 environments are passed by no model and 18 by exactly one. In the coordination track, every environment is passed by at least one model, and the most common count is 4 models (17 environments). On both tracks together, 19 environments are passed by no model and another 19 by exactly one. The size of an environment does not predict its difficulty. We compute the Spearman correlation between the mean check pass rate of the 12 models on each track and each of six counts: events, assistance requirement checks, violation checks, coordination

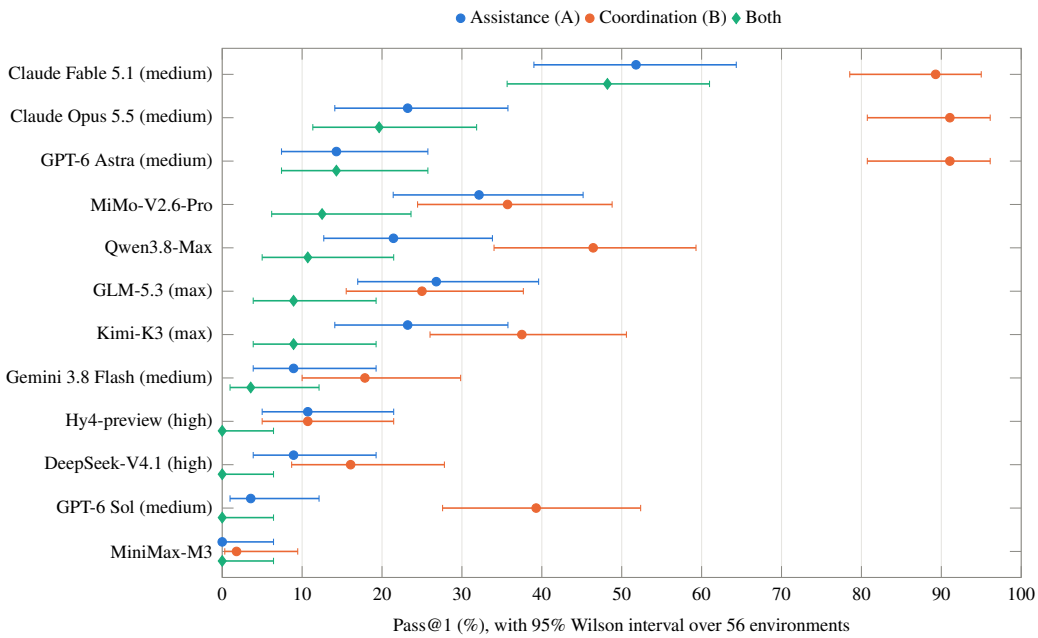


Figure 7: Pass@1 with 95% Wilson intervals over the 56 environments, for each track and for both tracks together.

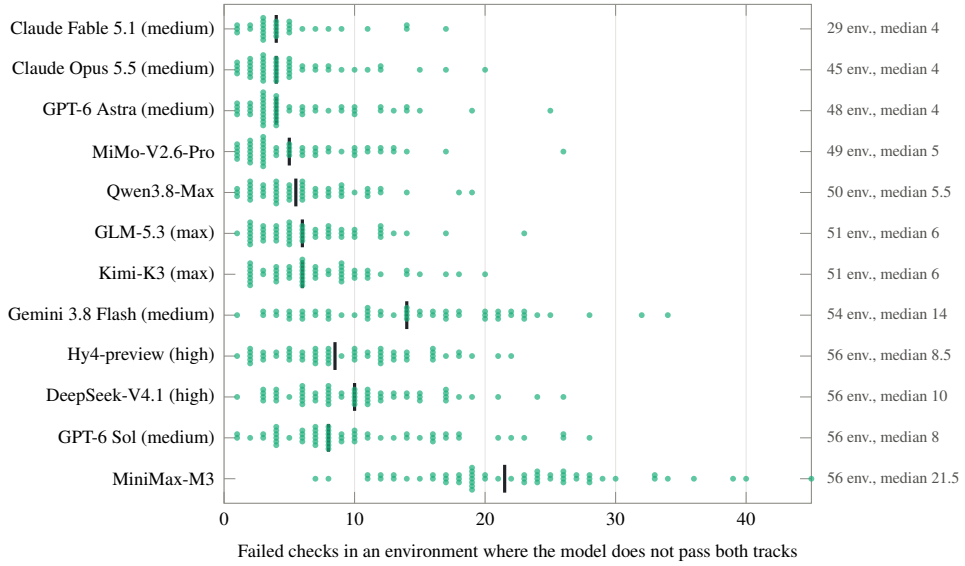


Figure 8: Failed checks per environment in the environments where a model does not pass both tracks. Each dot is one environment. The black bar marks the median.

checks, subtasks, and scripted colleague replies. All 12 correlations lie between -0.19 and 0.26. Passing each track on its own does not guarantee passing both. DeepSeek-V4.1-Flash passes the assistance track in 5 environments and the coordination track in 9, but these two sets never overlap.

Agreement between models. Figure 10 shows the Pearson correlation between the check pass rates of two models across the 56 environments. The correlations range from -0.04 to 0.68 with a median of 0.27, so most pairs of models are only weakly correlated. The highest correlation is between Claude Fable 5.1 and Claude Opus 5.5.

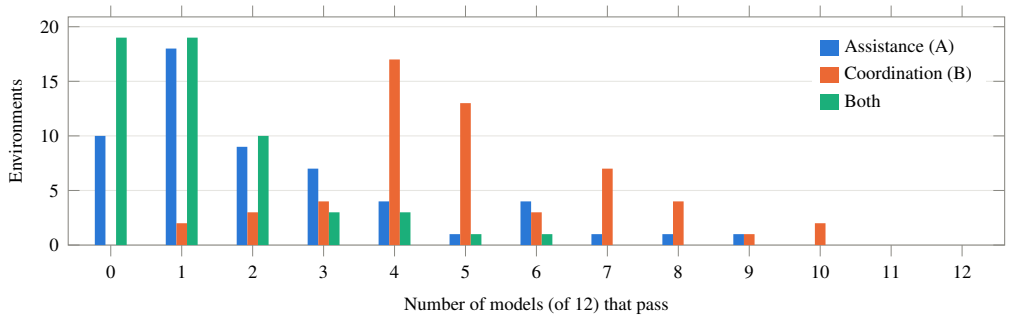


Figure 9: Number of environments by the number of models (out of 12) that pass the assistance track, the coordination track, or both tracks.

	Claude Fable 5.1	Claude Opus 5.5	GPT-6 Astra	MiMo-V2.6-Pro	Qwen3.8-Max	GLM-5.3	Kimi-K3	Gemini 3.8 Flash	Hy4-preview	DeepSeek-V4.1	GPT-6 Sol	MiniMax-M3
Claude Fable 5.1 (medium)												
Claude Opus 5.5 (medium)	.68											
GPT-6 Astra (medium)	.36	.51										
MiMo-V2.6-Pro	.33	.24	.09									
Qwen3.8-Max	.50	.39	.41	.44								
GLM-5.3 (max)	.56	.49	.27	.30	.29							
Kimi-K3 (max)	.21	.23	.28	.01	.19	.38						
Gemini 3.8 Flash (medium)	.48	.51	.45	.16	.35	.21	.38					
Hy4-preview (high)	.15	.24	.27	-.04	.03	.23	.48	.33				
DeepSeek-V4.1 (high)	.31	.35	.27	.25	.30	.34	.47	.24	.42			
GPT-6 Sol (medium)	.50	.43	.28	.16	.27	.14	.06	.35	.20	.18		
MiniMax-M3	.23	.22	.17	.05	.21	.13	.15	.26	.31	.24	.10	

Figure 10: Pearson correlation between the per-environment check pass rates of each pair of models (both tracks, 56 environments).

Results by domain. Figure 11 reports pass@1 by the domain of each track (Figure 3). In the assistance track, Claude Opus 5.5 passes none of the 16 People & HR environments, and Claude Fable 5.1 passes 38% of them, compared with 83% of the Marketing environments. In the coordination track, Claude Fable 5.1, Claude Opus 5.5, and GPT-6 Astra pass at least 62.5% of the environments in every domain, and the other nine models pass at most 62.5% in every domain.

	Assistance (A)					**Coordination (B)**			
	Customers & Sales	Finance	People & HR	Marketing	Operations & IT	Customer & Revenue	Events & Launches	Internal Operations	Incidents & Tech Work
Claude Fable 5.1 (medium)	50	50	38	83	60	94	81	88	100
Claude Opus 5.5 (medium)	21	30	0	67	30	100	94	94	62
GPT-6 Astra (medium)	14	10	12	17	20	81	100	100	75
MiMo-V2.6-Pro	43	30	25	17	40	31	38	31	50
Qwen3.8-Max	14	20	6	50	40	31	50	50	62
GLM-5.3 (max)	21	30	19	33	40	31	25	12	38
Kimi-K3 (max)	29	10	19	17	40	56	25	19	62
Gemini 3.8 Flash (medium)	14	10	0	0	20	19	19	19	12
Hy4-preview (high)	7	10	19	17	0	12	6	12	12
DeepSeek-V4.1 (high)	7	10	0	33	10	6	12	19	38
GPT-6 Sol (medium)	7	10	0	0	0	31	44	38	50
MiniMax-M3	0	0	0	0	0	0	0	6	0
	n=14	n=10	n=16	n=6	n=10	n=16	n=16	n=16	n=8

Figure 11: Pass@1 (%) by domain. Assistance-track domains are on the left and coordination-track domains on the right. The row n gives the number of environments per domain.

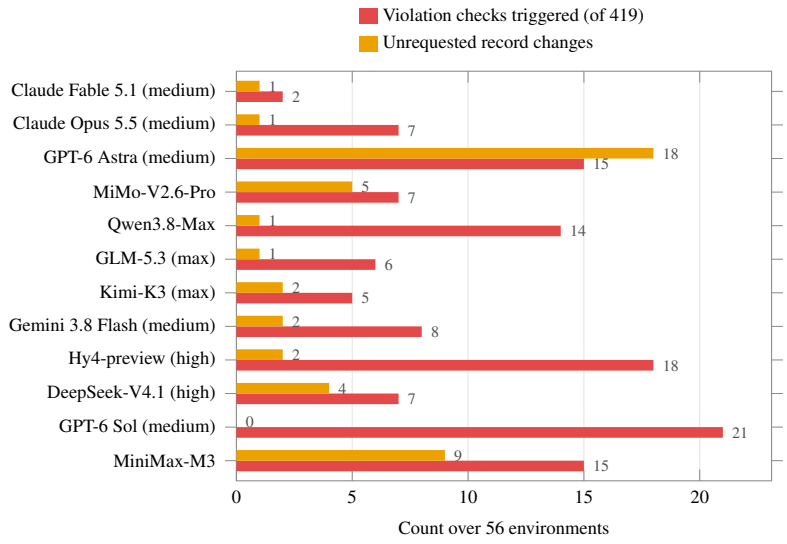


Figure 12: Violation checks triggered and unrequested record changes over the 56 environments.

Violation checks and unrequested changes. Figure 12 plots the counts discussed in Section 3.2.

Open and closed weights. Figure 13 compares the five closed-weight models with the seven open-weight models. The gap lies in the coordination track. Three closed-weight models pass it in 89.3% to 91.1% of environments, while the best open-weight model, Qwen3.8-Max, passes it in 46.4%. The median coordination pass rate is 89.3% for closed-weight models and 25.0% for open-weight models. In the assistance track, the median is 14.3% for closed-weight models and 21.4% for open-weight models, and MiMo-V2.6-Pro (32.1%) is second only to Claude Fable 5.1.

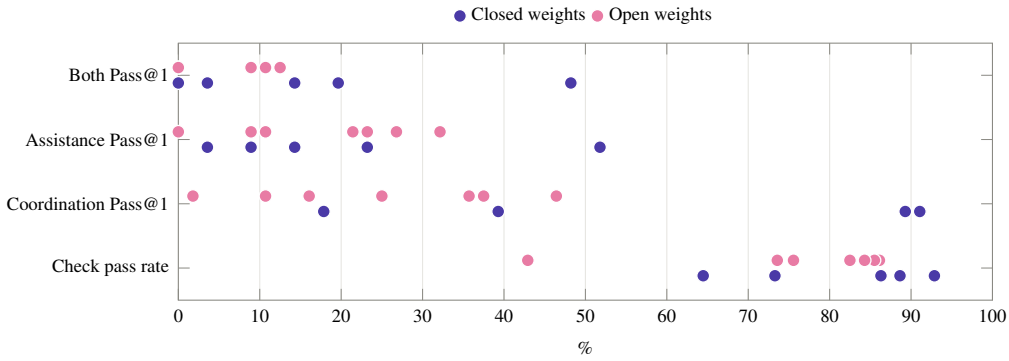


Figure 13: Pass rates of the five closed-weight models (purple) and the seven open-weight models (pink). Each marker is one model.

By system and requirement. Figure 14 breaks the check pass rate down by enterprise system for the assistance track and by requirement for the coordination track. For every model, the report has the lowest pass rate among the five requirements that every project has. The report is a reply in the CEO's thread that must contain the results delivered for the subtasks, so it fails when any of these results is missing.

Progress over the workday. Figure 15 plots the curves discussed in Section 3.3.

By colleague script. Figure 16 splits two coordination requirements by the behavioral script of the colleague who owns the subtask (Section 2.2). Claude Fable 5.1, Claude Opus 5.5, and GPT-6 Astra

	Assistance (A): by system									Coordination (B): by requirement						
	Slack	Sheets	Gmail	Sales-force	Asana	Zen-desk	Quick-Books	Bamboo-HR	Com-posite	Assign	Follow up	Relay	On time	Report	Order	Tickets
Claude Fable 5.1 (medium)	91	84	89	84	90	98	87	88	88	100	98	99	97	89	100	100
Claude Opus 5.5 (medium)	86	80	84	80	70	84	64	88	71	100	100	100	99	95	100	0
GPT-6 Astra (medium)	83	78	81	70	63	79	51	84	82	100	100	100	100	95	100	0
MiMo-V2.6-Pro	88	88	88	89	78	93	67	88	78	99	89	92	82	36	100	43
Qwen3.8-Max	89	78	83	84	97	91	62	88	78	100	91	93	85	48	100	0
GLM-5.3 (max)	84	82	84	75	89	95	69	96	76	100	87	96	79	25	100	100
Kimi-K3 (max)	81	80	86	85	63	95	51	80	77	95	89	89	81	38	100	100
Gemini 3.8 Flash (medium)	62	53	64	70	43	89	41	68	53	88	69	74	66	20	67	0
Hy4-preview (high)	78	76	81	87	83	96	36	84	79	93	72	76	64	11	100	100
DeepSeek-V4.1 (high)	81	80	80	82	75	84	38	84	81	86	69	68	60	16	100	43
GPT-6 Sol (medium)	75	58	65	80	48	65	56	20	64	96	87	85	84	43	100	0
MiniMax-M3	55	55	58	61	56	68	18	56	57	58	16	16	21	2	50	100
	n=325	n=249	n=247	n=61	n=63	n=57	n=39	n=25	n=118	n=257	n=198	n=192	n=257	n=56	n=6	n=7

Figure 14: Check pass rate (%) by enterprise system (assistance track, requirement checks) and by requirement (coordination track). Composite checks group several conditions that must all hold or of which one must hold. Assign: the subtask goes to the correct owner. Follow up: the agent sends enough follow-ups. Relay: an upstream result reaches the downstream colleague. On time: the subtask is delivered before the deadline. Report: the final reply to the CEO contains the results. Order: one subtask is dispatched before another. Tickets: support tickets handled within two projects. The row n gives the number of checks per column.

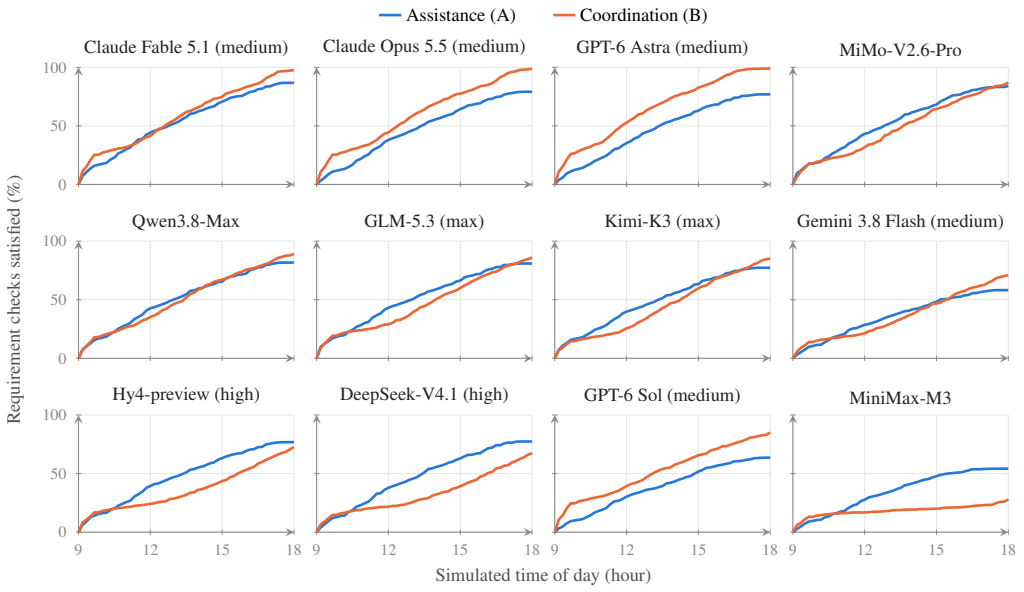


Figure 15: Share of requirement checks satisfied over the simulated workday, pooled over the 56 environments. The time of a check is the first time it is satisfied. Each panel is one model, ordered by the rate of passing both tracks.

deliver more than 94% of subtasks on time under every script. For each of the other nine models, subtasks owned by late-once colleagues have the lowest on-time rate, from 7.1% for MiniMax-M3 to 78.6% for Qwen3.8-Max. These nine models also send enough follow-ups to late-once colleagues less often than to late-twice colleagues.

Number of follow-ups. Figure 17 compares the number of follow-ups each model sends with the minimum that the colleague scripts require. We count as a follow-up every message the agent sends to the owner of a subtask between dispatch and delivery, except the message that relays an upstream result. Claude Fable 5.1, Claude Opus 5.5, and GPT-6 Astra send at least as many follow-up messages as required for every subtask, and in total send 1.43 to 1.79 times the minimum. GPT-6

	Subtask delivered on time				Enough follow-ups sent		
	Punctual	Late-once	Late-twice	Silent	Late-once	Late-twice	Silent
Claude Fable 5.1 (medium)	100	96	95	98	98	96	100
Claude Opus 5.5 (medium)	100	100	98	98	100	100	100
GPT-6 Astra (medium)	98	100	100	100	100	100	100
MiMo-V2.6-Pro	90	69	88	88	88	89	91
Qwen3.8-Max	88	79	91	86	88	93	93
GLM-5.3 (max)	88	64	86	84	79	93	93
Kimi-K3 (max)	85	71	89	84	85	93	93
Gemini 3.8 Flash (medium)	76	48	75	74	55	82	75
Hy4-preview (high)	73	43	81	70	61	84	77
DeepSeek-V4.1 (high)	71	42	79	54	56	86	70
GPT-6 Sol (medium)	93	76	88	81	87	91	82
MiniMax-M3	46	7	23	16	7	26	19
	n=59	n=84	n=57	n=57	n=84	n=57	n=57

Figure 16: Coordination results (%) by the behavioral script of the subtask owner. Left: the subtask is delivered before the deadline. Right: the agent sends at least as many effective follow-ups as the script needs (Appendix I, Number of follow-ups). Punctual colleagues need no follow-up. The row n gives the number of subtasks per column.

Astra also sends 67 follow-ups to punctual colleagues, who need none. Extra follow-ups do not lower the score, but each one is a message a colleague has to read. At the other end, Gemini 3.8 Flash, Hy4-preview, and DeepSeek-V4.1-Flash follow up 17.7% to 24.2% of subtasks too few times, and MiniMax-M3 does so for 76.3% of subtasks. Figure 16 counts only effective follow-ups, so its share can fall below 100% for a model that sends enough messages. A follow-up is effective when it is posted in the subtask's thread and tags the owner. It must also come after the owner's promised time, or at least 30 minutes after dispatch if the owner has not promised a time. A follow-up sent before the owner starts the subtask is not effective (Appendix B, Colleague rules).

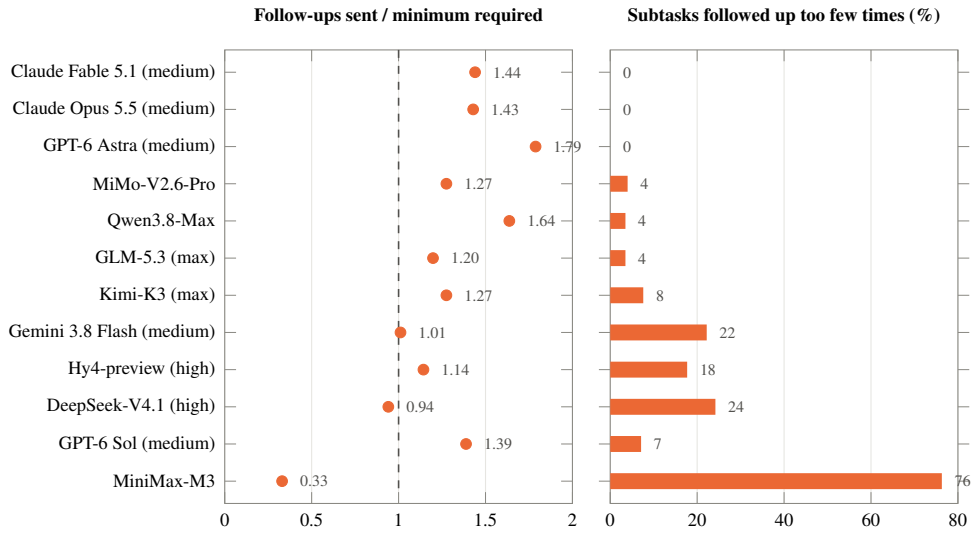


Figure 17: Left: follow-up messages sent divided by the minimum the colleague scripts require, over all 56 projects. The dashed line marks the minimum. Right: share of subtasks that need follow-ups and receive fewer than required.

Tool use. Figure 18 groups each model's tool calls into five kinds: reading the handbook in Google Drive, reading other systems, sending an email or Slack message, writing to a record, and calling the clock. Models make 131 to 191 tool calls per environment on average. Claude Opus 5.5 makes the fewest calls and spends the smallest share on the handbook (9.8%), while the other models spend 16.6% to 26.1%. For every model, fewer than 3.5% of calls return an error. Figure 19 shows when

the calls happen. Every model makes 32% to 60% of its calls between 09:00 and 09:30, and at most 6.3% in any later half hour.



Figure 18: Share of tool calls by kind, average number of tool calls per environment, and share of calls that return an error.

	9:00	10:00		11:00		12:00		13:00		14:00		15:00		16:00		17:00	
Claude Fable 5.1 (medium)	51	3	3	3	4	3	3	3	3	3	3	3	2	2	4	3	2
Claude Opus 5.5 (medium)	32	5	4	6	6	5	5	4	5	4	3	3	3	3	4	3	2	2
GPT-6 Astra (medium)	39	4	4	4	5	4	5	5	6	5	3	3	3	3	3	2	1	1
MiMo-V2.6-Pro	57	3	3	3	3	3	3	3	3	2	2	2	2	2	2	2	2	3
Qwen3.8-Max	48	3	3	4	4	4	3	3	4	4	2	3	3	2	3	2	3	2
GLM-5.3 (max)	54	3	3	3	3	3	3	3	3	3	3	2	3	2	2	2	2	3
Gemini 3.8 Flash (medium)	41	3	3	4	5	4	4	3	4	3	3	3	3	3	3	3	3	4
Hy4-preview (high)	55	3	2	3	3	3	3	3	3	3	2	2	3	2	2	2	2	3
DeepSeek-V4.1 (high)	55	3	3	3	4	2	3	2	3	3	2	2	2	2	3	2	2	3
GPT-6 Sol (medium)	40	4	3	5	5	4	4	4	3	3	4	3	3	3	3	2	3	4
MiniMax-M3	60	2	3	3	3	3	3	2	3	2	2	2	2	1	2	1	2	2

Figure 19: Share of each model's tool calls (%) in each half hour of the simulated workday, pooled over the 56 environments.

Awareness of the simulation. Six models return their full reasoning through the API: DeepSeek-V4.1-Flash, GLM-5.3, Hy4-preview, Kimi-K3, MiMo-V2.6-Pro, and Qwen3.8-Max. GPT-6 Sol returns reasoning summaries. Table 6 counts the runs in which the reasoning or the replies call the environment a simulation (the words "simulation", "simulated", "scripted", or "benchmark") or mention a grader ("grader", "graded", or "grading"). The six models call the environment a simulation in 42 to 56 of their 56 runs and mention a grader in 2 to 41 runs. For example, MiMo-V2.6-Pro reasons in environment 101 that "the grader would check 'did the assistant use the guest list from the wrong person?' ", and Hy4-preview reasons in environment 107 that "a grader checking the Pipeline sheet might look for rows matching open opportunities."

J Ablation: Concurrent vs. Isolated Tracks

To measure how running the two tracks together affects performance, we run each track in isolation (removing the other track from the environment) and compare against the concurrent setting. We run this ablation for Kimi-K3 and DeepSeek-V4.1-Flash on all 56 environments. Table 7 shows the results.

Table 6: Runs (out of 56) in which a model's reasoning or replies call the environment a simulation or mention a grader. GPT-6 Sol returns only reasoning summaries.

Model	Simulation	Grader
DeepSeek-V4.1-Flash (high)	56	41
MiMo-V2.6-Pro	50	39
Qwen3.8-Max	50	25
GLM-5.3 (max)	48	19
Hy4-preview (high)	43	21
Kimi-K3 (max)	42	2
GPT-6 Sol (medium)	25	0

Table 7: Ablation: isolated vs. concurrent track performance. "Isolated" runs each track alone in the same environment. "Concurrent" is the main experiment, where both tracks run together. Pass@1 and Checks are defined as in Table 2. "Change" is the relative change in Pass@1 when the other track is added (red: lower, green: higher).

Model	Track	Isolated		Concurrent		Change
		Pass@1	Checks	Pass@1	Checks	
Kimi-K3 (max)	Assistance (A)	33.9	87.6	23.2	79.8	−32%
	Coordination (B)	44.6	90.9	37.5	85.8	−16%
DeepSeek-V4.1-Flash (high)	Assistance (A)	30.4	85.3	8.9	78.3	−71%
	Coordination (B)	7.1	69.2	16.1	67.7	+125%

For both models, assistance track performance drops substantially when the coordination track is present. Kimi-K3 drops from 33.9% to 23.2% (−32%), and DeepSeek-V4.1-Flash drops from 30.4% to 8.9% (−71%). The check pass rates drop as well, from 87.6% to 79.8% for Kimi-K3 and from 85.3% to 78.3% for DeepSeek-V4.1-Flash. One possible explanation is that coordination activity (dispatching work, monitoring deadlines, relaying deliverables) takes attention away from assistance work.

Coordination track performance does not drop consistently. Kimi-K3 drops from 44.6% to 37.5% (−16%). DeepSeek-V4.1-Flash passes the coordination track in 16.1% of environments in the concurrent setting and 7.1% in isolation, with nearly equal check pass rates (67.7% and 69.2%). The coordination track has more structured interaction patterns (dispatch, wait, follow up, relay) that may be less sensitive to interruption than the assistance track's diverse, policy-driven workflows.

Figure 20 plots the pass rates of Table 7.

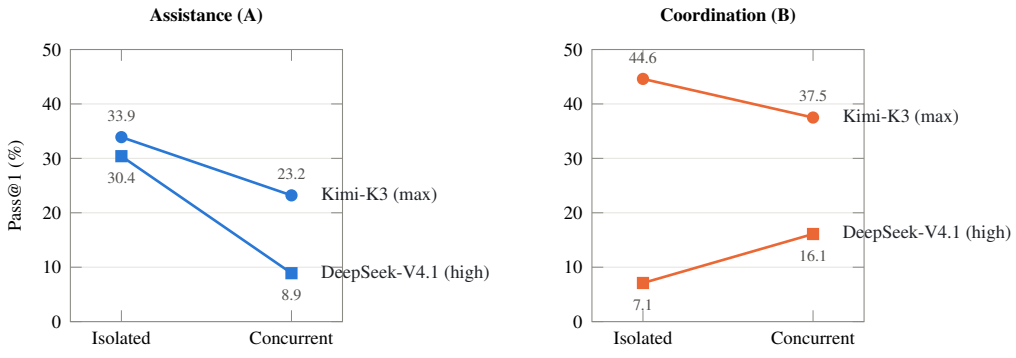


Figure 20: Pass@1 of each track when it runs alone (isolated) and together with the other track (concurrent), for the two models in the ablation.