# ASSETARENA: A BENCHMARK FOR AGENTIC WORKFLOWS IN INDUSTRIAL ASSET OPERATIONS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Industrial asset operations require agents to combine condition assessment, diagnosis, evidence retrieval, and operational action within a single stateful workflow. However, evaluations remain fragmented, as deployed agents typically cover only part of this workflow, while industrial benchmarks largely assess isolated capabilities or static knowledge rather than end-to-end execution. We introduce *AssetArena*, an executable benchmark for agentic workflows in industrial asset operations. AssetArena provides a unified MCP-native environment that connects heterogeneous industrial evidence, analytical tools, and operational actions, with scenarios designed so that relevant information must be acquired through interaction. Its evidence-first construction process establishes reference outcomes independently of the agent and records replayable executions for reproducible evaluation. The benchmark contains 231 human-authored, machine-verified scenarios and 85 tools across six MCP servers. We evaluate eight frontier language-model configurations under a common agent framework and further study the effects of domain skills, agent decomposition, tool-access topology, and execution harness. Models achieve 16.5–29.9% Pass@1 across the benchmark, indicating substantial headroom even for frontier models. Controlled comparisons show that agent design and execution infrastructure affect task success and interaction cost: across 280 paired runs, Stirrup agent achieves 31.1% Pass@1 versus 25% for OpenCode. AssetArena provides a common and extensible environment for evaluating model capability and agent design on challenging industrial asset-operation workflows. Reproducibility materials are available at: `https://anonymous.4open.science/r/AssetArena-ICLR/`

## 1 INTRODUCTION

When a reliability engineer asks which excavator in a fleet is failing fastest, no single record contains the answer. The answer must be inferred from maintenance histories observed over unequal exposure windows as machines entered service at different times, and raw breakdown counts confound failure frequency with operating exposure. The result must then be written into the operational system where a planner can act on it. This illustrates a broader class of industrial asset-operation workflows. The request specifies an operational goal rather than a method, evidence must be discovered, retrieved, or computed, and success requires an auditable action rather than merely a natural-language response. Industrial AI products increasingly address pieces of these workflows, but not their full composition. For example, Augury's Reliability Agent recommends corrective actions and relies on a partner product to create work orders (Rockwell Automation, 2026). Salesforce's field-service agent can detect when service is due and create a service appointment automatically (Sivanathan, 2025), but does not claim to perform failure-mode diagnosis (Salesforce, 2025). This division is structural. Systems that assess asset condition often do not modify operational records, while systems that schedule or create records do not perform the underlying diagnosis. Yet industrial requests routinely cross this boundary.

The above deployed systems cannot readily serve as research benchmarks. Among the industrial offerings we survey, none exposes its complete agent layer, data, and evaluation environment to external researchers. Self-service offerings may withhold the relevant API layer, as in the Maximo trial environment (IBM, 2026b). Vendors also do not release the work orders, telemetry, failure codes, and reference outcomes needed for reproducible evaluation. At the same time, the interface



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.

layer is opening. Vendors are exposing Model Context Protocol (MCP) interfaces over the systems that agents use. MCP has rapidly become the universal standard protocol for connecting AI models to tools, data and applications, with more than 10,000 published MCP servers now covering everything from developer tools to Fortune 500 deployments (The Linux Foundation, 2025). IBM's Maximo Application Suite supports customer agents that integrate with Maximo Manage APIs (IBM, 2026a), and Salesforce exposes MCP across Agentforce (Salesforce, 2026). The same protocol that lets agents interact with enterprise tools can support an open, vendor-neutral testbed that assembles representative tools, evidence, and operational state into a controlled environment.

Existing benchmarks provide important but incomplete foundations. General agentic benchmarks establish executable environments, stateful tasks, and execution-derived evaluation (Sun et al., 2026a; Merrill et al., 2026; Jimenez et al., 2024; Jha et al., 2025; Chen et al., 2025; Egg et al., 2025), but do not target maintenance, reliability, or industrial asset-management workflows. Likewise, MCP benchmarks such as MCP-Bench (Wang et al., 2026) cover tool breadth, orchestration topology, mobile environments, evaluation tooling and adversarial robustness, but we found none targeting industrial asset operations. Most industrial benchmarks instead test domain knowledge over frozen text or annotated corpora, including DiagnosticIQ and FailureSensorIQ (Silva et al., 2026; Constantinides et al., 2026). Recent industrial agent benchmarks address complementary capabilities. AssetOpsBench (Patel et al., 2026a) evaluates tool use in industrial scenarios, but does not support coding agents or MCP tools and relies on LLM-as-a-judge evaluation. PHMForge (Li et al., 2026c) exposes domain-specific algorithms for asset-health analysis, but provides the required data within each scenario rather than requiring agents to discover and integrate evidence. Both benchmarks are based on old generation models, limiting their ability to distinguish further improvements in agent capability (Patel et al., 2026b). Neither provides a vendor-neutral environment in which an agent must acquire heterogeneous industrial evidence, conduct analysis, and execute a verifiable end-to-end workflow.

**AssetArena.** We introduce *AssetArena*, an open, executable benchmark in the spirit of WebArena (Zhou et al., 2024) and FieldWorkArena (Takahashi et al., 2026), where the unit of evaluation is an **industrial workflow** rather than an isolated question. Our key contributions are:

- **Diverse, evidence-first industrial scenarios.** We instrumented **231 human-authored, machine-verified scenarios** across seven workflow families: work-order reasoning (WO), clarification/abstention/response (CAR), failure-code classification (FCC), asset-health assessment (HLT), failure-mode–sensor reasoning (FMSR), time-series modeling (TSFM), and failure mode and effects analysis (FMEA). Scenarios draw on 564 maintenance records and 62 engineer-coded records, standards-based failure-mode–sensor knowledge (59 failure modes, 56 monitoring parameters, and approximately 470 expert-defined relationships), and 25 controlled MATLAB/Simulink tasks across five engineering systems. A seven-predicate admission process validates each phenomenon, withholds its computed reference, audits evidence reachability, and rejects shortcut prompts.
- **An executable industrial environment.** We develop a shared stateful environment containing **224 assets and 1,076 maintenance work orders**, telemetry, engineering catalogues, vibration resources, and 85 MCP tools across six servers. A common OpenAI-compatible interface serves all evaluated models. Agents retrieve evidence and execute permitted actions through Stirrup or OpenCode, with code-capable configurations running in isolated Docker workspaces.
- **A controlled testbed for agent design and evaluation.** We implement **217 reusable industrial domain skills** using Repo-To-Skill method (Chen et al., 2026)(See Appendix C), gateway-mediated tool access, and domain-sub-agent routing over one scenario environment and one deterministic evaluator. The evaluator scores final trajectories against hidden, precomputed references, applying exact-match, mode-aware, or semantic-F1 rules by workflow family, and saved trajectories support reproducible rescoring and failure analysis. Matched experiments isolate how agent design and execution infrastructure affect completion, cost, and interaction behaviour.

Across eight frontier-model configurations, the best model reaches only 29.9% Pass@1, indicating substantial headroom for reliable end-to-end industrial workflows. Agent-design interventions primarily affect efficiency rather than accuracy (K1 skills: $+2$ passes; sub-agents: $-3$; gateway: $+3$). Doubling the turn budget yields only a marginal gain at additional token cost. On average, closed-weight models show narrower gaps between Pass@3 and Pass$^3$, indicating more repeatable outcomes and less benefit from retries. Open-weight models show wider gaps, suggesting greater potential for test-time scaling through independent additional attempts.

## 2 Related Work

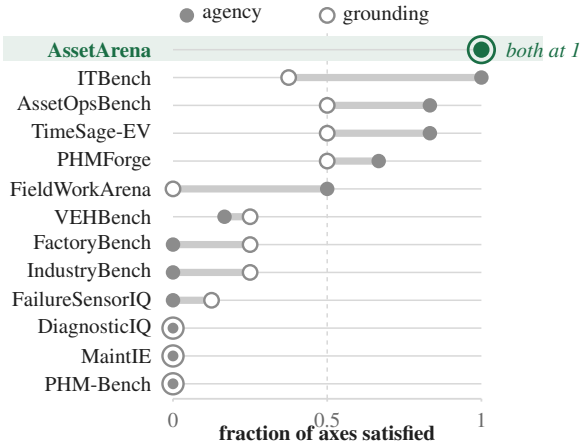



Figure 1: Benchmark Composite Index

We reviewed 12 industrial benchmarks and split them into three groups. Figure 1 scores each from two angles, how the agent reaches an answer and how that answer is established. The first group, the bottom seven rows of the figure, probes what a model knows about physical assets and the tasks around them: PHM-Bench (Yang et al., 2025b), MaintIE (Bikaun et al., 2024a), DiagnosticIQ (Silva et al., 2026), FailureSensorIQ (Constantinides et al., 2026), IndustryBench (Bai et al., 2026), FactoryBench (Merzouki et al., 2026) and VEHBench (Su et al., 2026). These seven score against answers already recorded in standards, manuals or annotated corpora, most of them through multiple-choice question answering. Group two runs an agent against a live tool surface and each is shaped by the capability it isolates. AssetOpsBench (Patel et al., 2026a) evaluates tool use on 141 scenarios that specify the asset, sensor, site, and time window, measuring execution after retrieval is resolved. PHMForge (Li et al., 2026c) exposes published domain-specific algorithms through MCP-native tools across 99 scenarios, making the analytical method part of the tool surface. TimeSage-EV (Yao et al., 2026) constrains the temporal evidence to isolate time-series reasoning. The last two run agents over larger environments and sit outside asset operations, FieldWorkArena (Takahashi et al., 2026) focuses on observational work, whereas ITBench (Jha et al., 2025) targets on IT systems. Only ITBench reaches the top of the horizontal composite, and no benchmark passes half of the vertical one.

We also reviewed documentation for ten deployed industrial AI offerings and recorded which capabilities vendors attribute to an agent rather than to an application or model. Table 1 shows that no documented agent spans condition assessment, diagnosis, evidence-grounded decision making, and an operational record update within one workflow. Augury's Reliability Agent and Emerson's Ovation AI assess asset health but do not write operational records, whereas Rockwell's Fiix MAX creates work orders without performing condition assessment (Rockwell Automation, 2026). Maximo Assistant is the only offering that touches all three stages, but delivers failure coding and health assessment through applications rather than an agent. No offering documents an agent that decides whether available evidence warrants a response, clarification, or abstention. Work-order management is the most widely documented operational capability, yet its underlying records remain difficult to use. More than half of maintenance effort may be reactive (Sullivan et al., 2010), 60–98% of work orders lack an ISO 14224 detection-method code (Hodkiewicz et al., 2021), breakdown indicators were unused across 180 rotating assets (Lukens & Markham, 2018), and 94.8% of 12K maintenance short texts required normalization before reliable parsing (Bikaun et al., 2024b).

Table 1: Workflow capabilities documented for deployed industrial agents.

| **Deployed agent** | Work orders | Clarify abstain | Failure coding | Health assessment | Sensor reasoning | Time series | Schedule dispatch | Prognostics (RUL) | Technician support |
|---|---|---|---|---|---|---|---|---|---|
| Maximo Assistant (IBM, 2026c;d) | ● | ○ | ◐ | ◐ | ○ | ◐ | ◐ | ◐ | ● |
| Agentforce (Salesforce, 2025) | ● | ○ | ○ | ◐ | ○ | ◐ | ● | ◐ | ● |
| Joule Agents (SAP, 2026b;a) | ◐ | ○ | ○ | ◐ | ○ | ◐ | ◐ | ◐ | ○ |
| Fiix MAX (Rockwell Automation, 2026) | ● | ○ | ○ | ○ | ○ | ○ | ◐ | ○ | ○ |
| Reliability Agent (Augury, 2026a;b) | ○ | ○ | ○ | ● | ◐ | ● | ○ | ◐ | ○ |
| Maintenance Copilot (Siemens AG, 2025; 2026) | ○ | ○ | ○ | ◐ | ○ | ◐ | ○ | ◐ | ◐ |
| Ovation AI (Emerson Electric Co., 2026) | ○ | ○ | ○ | ● | ○ | ● | ○ | ◐ | ◐ |
| C3 Reliability (C3 AI, 2026a;b) | ◐ | ○ | ○ | ◐ | ○ | ◐ | ○ | ◐ | ○ |
| Forge (Honeywell, 2026) | ○ | ○ | ○ | ◐ | ◐ | ◐ | ○ | ◐ | ◐ |
| SmartSignal (GE Vernova, 2026a;b) | ○ | ○ | ○ | ◐ | ○ | ◐ | ○ | ◐ | ○ |

*Notes.* ● denotes a capability documented as performed by the vendor's agent. ◐ denotes a capability delivered through an application or model rather than an agent. ○ denotes no documented capability.

## 3 EVIDENCE-FIRST SCENARIO CONSTRUCTION

A benchmark question is often written first and answered afterward. This works when the answer is a fact, but not when it depends on a physical phenomenon. A plausible request may describe an event that is absent from the available data, making its author-provided answer impossible to verify.

To tackle this issue, we present a scenario-generation pipeline that include seven admission predicates (Figure 2). Each scenario is anchored to a specific asset and begins with a verified phenomenon derived either from a physics-based simulator executed under a known configuration $\theta_i = (\textit{asset}, \textit{operating point}, \textit{fault}, \textit{severity})$, which provides provenance and experimental control, or from industrial records associated with that asset, which provide operational authenticity. A subject-matter expert verifies the phenomenon against its source before any scenario content is authored (C1). We then compute the reference from that verified phenomenon and render the request only afterward. This evidence-first order defines properties that post hoc validation cannot recover. It extends Agents' Last Exam (Sun et al., 2026a) by making its *representativeness*, *verifiability* , and *workflow-completeness* criteria operational for industrial tasks. C1 anchors the scenario in a simulator run or industrial record. C2 computes a programmatic reference and C3 withholds it, making the outcome reproducible and independently re-assertable. C4 tests whether the permitted evidence and tools can recover the reference. C5 measures sensitivity to labelled evidence through a minimum-flip admission test. C6 renders the request after the evidence is fixed, rejecting requests that admit a shortcut and exposing the operational confound without prescribing a method. C7 excludes language models from both request authoring and outcome judgment. C1–C3 and C7 therefore hold by construction, while C4–C6 test whether a candidate requires the intended evidence-grounded workflow. The following examples illustrate this end-to-end procedure through one simulation-driven scenario and one record-driven scenario.

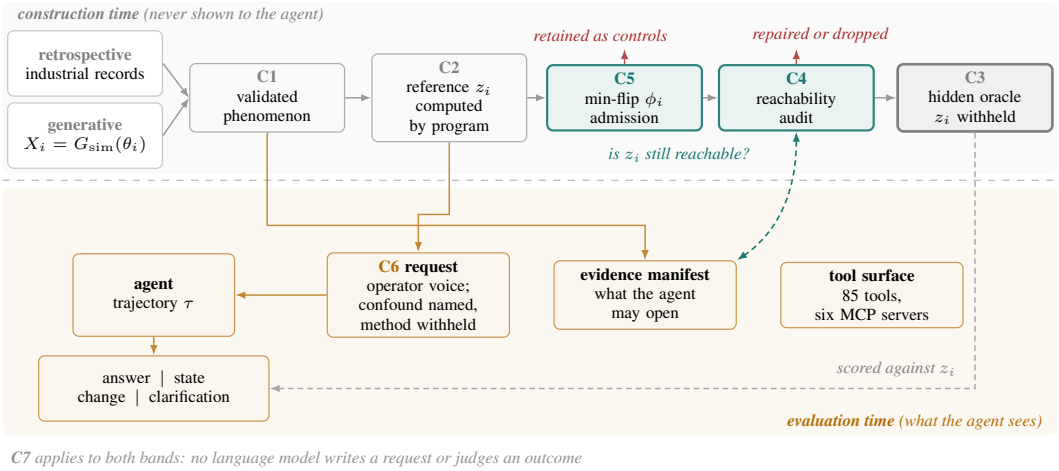



Figure 2: Evidence-first scenario construction and evaluation

**Example 1: simulation-driven time-series reasoning.** Construction begins with a simulation configuration rather than a request. We use expert-developed MATLAB/Simulink digital twins in which operating conditions, fault type, onset time, and severity are configurable. For example, we set $\theta_i = (\text{PMSM motor}, 1500\,\text{rpm}, \text{inter-turn short}, \text{severity } 0.4)$ and generate telemetry $X_i = G_{\text{sim}}(\theta_i)$. Since the fault is injected before the simulation, its type and onset are known by construction and form the hidden reference $z_i$. The agent receives only unlabeled telemetry, asset metadata, and tools for spectral, envelope, and changepoint analysis. It receives neither the fault label, onset index, nor $\theta_i$. The request is rendered last in an operator's voice: *"Something changed on M-217 partway through yesterday's run. Tell me what kind of fault it is and when it started. I do not want a threshold crossing."* We score fault class by exact match and onset time using a tolerance.

**Example 2: record-driven failure-code classification.** Construction begins with 62 engineer-coded corrective work orders that are held out from the agent. The hidden reference $z_i$ includes the failure code for each record and the reliability KPI implied by those assignments. A program generates both the reference and the operator-facing request before the scenario is released. The

scenario excludes the held-out codes from all information available to the agent. In the evaluation, the agent accesses an available work-order register. A typical row contains `WO-4412`, `DESC pmp seal lkg`, an absent `FAILURECODE`, and 31 additional fields. Across the register, 11% of orders are uncoded, 22% of coded orders use a single catch-all code, and descriptions average 2.1 tokens. The request is rendered last in an operator's voice: *"Corrective orders on the pump skid last quarter: which failure mechanism is driving the repeat visits? The coding is patchy, so do not just count the code field."* Scoring uses exact match for both per-record assignments and the analytic answer.

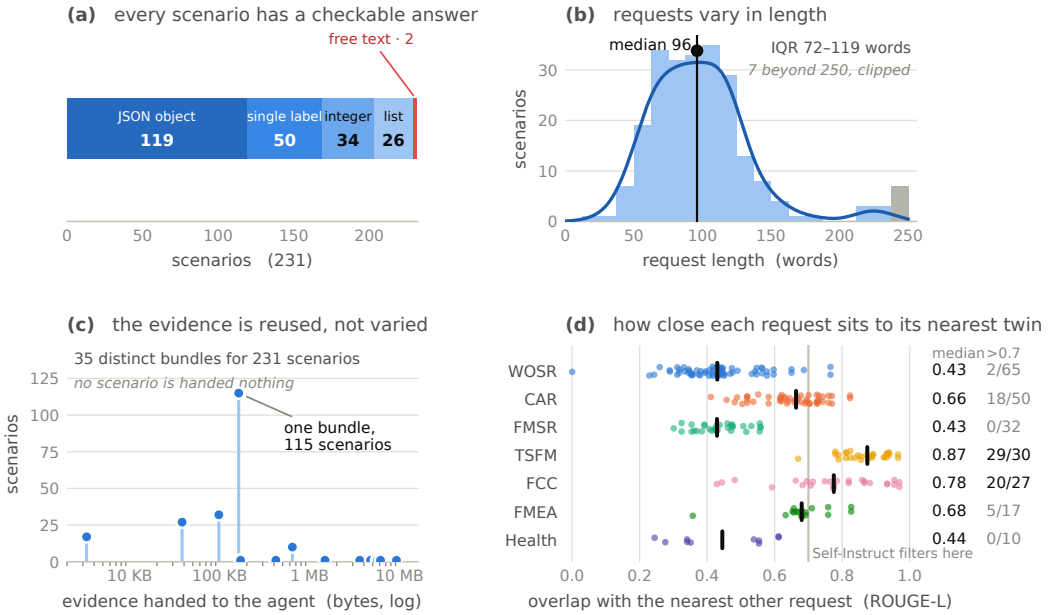



Figure 3: The scenario store: Capturing scenarios diversities

**Development efforts.** Eight asset-maintenance domain experts with practical field experience spent three months developing and validating the scenarios across the seven workflow families. Appendix B.11 lists their domain qualifications and Appendix B gives scenario examples for each category. Figure 3 characterizes the scenario store along dimensions relevant to benchmark diversity and templating. Requests range from one to 34 sentences and from 23 to 490 words, with a median length of 96 words. Nearest-request overlap varies sharply across families. WO and FMSR scenarios have median ROUGE-L overlap of .43, with only 2 of 65 WO and none of 32 FMSR requests exceeding .7. In contrast, parameterized TSFM and FCC scenarios have median overlap of .87 and .78, with 29 of 30 and 20 of 27 requests above .7, respectively. Prompt similarity therefore reflects how a family is instantiated, rather than establishing its difficulty or determining its performance. Each request directs the agent to two to six evidence classes containing 3.3 KB to 9.2 MB of material, with a median of 162 KB. The manifest and reachability audit require the reference to be recoverable only through the permitted evidence and tool surface. Output contracts are closed but varied. A total of 119 scenarios require a JSON object with up to 17 fields, 50 require a single catalogue label, 34 require an integer, and 26 require a list. Only two scenarios allow free text. All remaining scenarios are scored by script against a reference computed before the request is written.

## 4 Industrial Asset Operations Environment

AssetArena compiles each scenario into an isolated, executable environment (Figure 4). Each scenario is defined by three files. `question.txt` states the operator-facing objective. `manifest.json` selects the evidence $E_i$ and tool subset $T_i$ available to the agent. `groundtruth.txt` contains the reference outcome $z_i$, reference evidence $E_i^*$, and execution requirements $R_i$; which are available only to the evaluator. At run time, the instantiator resets managed state, loads the manifest-selected records, catalogues, and telemetry into CouchDB, creates a dedicated workspace, and mounts the registered MCP tools (Appendix B.9). The agent receives

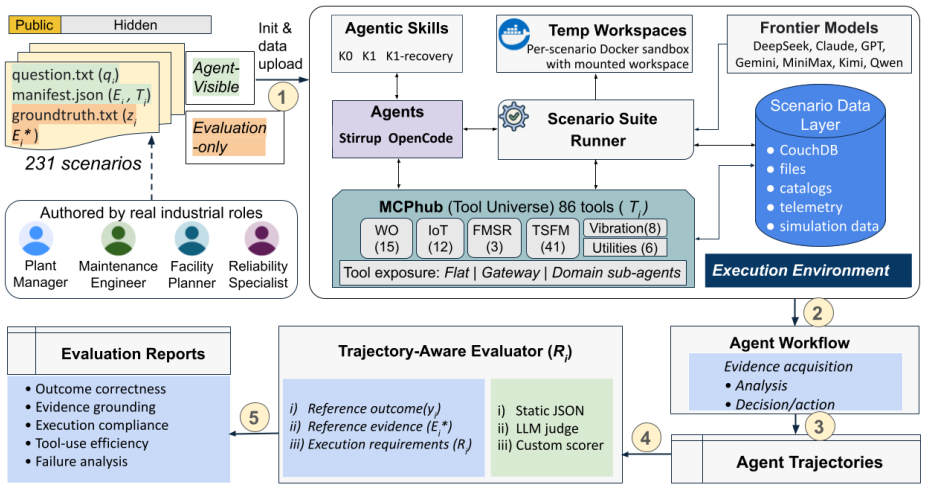



Figure 4: Overview of the AssetArena framework with five key modules: ① Scenario generation; ② Environment setup; ③ Agent execution; ④ Trajectory collection; ⑤ Performance evaluation.

`question.txt` and may invoke MCP tools, execute code, and modify permitted operational state. Each run executes in an isolated Docker container with a dedicated mounted workspace, preventing code and file state from leaking between scenarios. After execution, the environment is destroyed and the saved trajectory is evaluated against `groundtruth.txt`, which is never exposed inside the agent environment.

**Shared Resources and Tool Access:** A manifest selects from a shared industrial resource layer rather than from files created for one scenario. Building this layer was the principal authoring effort. It standardizes heterogeneous industrial artifacts into typed records, files, and catalogues that every workflow family can reuse. The shared `catalog/` provides searchable descriptions, identifiers, relationships, and execution requirements for assets, sensors, meters, failure modes, models, and derived features. It gives agents information to search and connect, but never contains a scenario reference outcome. Other resources provide the underlying evidence, including per-asset IoT telemetry, engineer-assigned failure codes, FMEA tables, raw and normalized work-order registers, and time-series workflow artifacts. Shared asset identifiers link these resources, allowing an agent to connect a work order to its failure codes, sensors, and telemetry without a scenario-specific key. A scenario is therefore a controlled selection from one industrial world, rather than a bespoke dataset handed directly to an agent. Stirrup (Artificial Analysis, 2026) and OpenCode (OpenCode AI Team, 2026) agents access these resources through 85 typed MCP tools across six servers for work orders, IoT telemetry, failure modes and sensors, time-series models, vibration analysis, and shared utilities. For code-capable Stirrup runs, a containerized `assetarena-code` sandbox is also mounted; large tool outputs are provided as workspace file handles rather than inserted into the model context.

**Controlled Agent Interfaces.** We implement three controlled agent-side interventions. *i) Skills* compare K0, in which the root agent receives no domain-skill library, with K1, in which it may consult a versioned, routed library of domain procedures and tool-use guidance containing no scenario-specific reference answers. *ii) Gateway access* replaces direct schema exposure with `search_tools`, `describe_tool`, and `call_tool`, revealing schemas only when needed. *iii) Domain-sub-agent access* assigns each major server to a specialized worker. The root agent delegates bounded requests and receives structured results, identifiers, and artifact handles; workers do not communicate directly, and the root remains responsible for cross-domain evidence integration and the final response.

**Trajectories and Reproducible Scoring:** Each execution produces the trajectory $\tau_i = [(a_{i,1}, o_{i,1}), \ldots, (a_{i,L_i}, o_{i,L_i}), \hat{y}_i]$, which records the accessed tools, invoked arguments, observations, artifacts, errors, final response, and associated turns, tool calls, token usage, and time. After execution, the evaluator joins the trajectory to the withheld reference. This supports outcome scoring and, when defined by a scenario, evidence-aware and process-aware diagnostics. Since execution and scoring are separate, a saved trajectory can be rescored without reinvoking the agent, and a disputed grade can be resolved from the recorded execution rather than a new stochastic run.

# 5 EXPERIMENTS

We evaluate AssetArena along four axes: model evaluation, agent design, execution harness, and budget with repeated runs. First, we discuss experimental setup and success criteria.

**Models and Agent Configuration.** We test eight frontier models, five closed-weight (Gemini 3.6 Flash (Google DeepMind, 2026), GPT-5.6 (OpenAI, 2026), Claude Opus 5 (Anthropic, 2026a), Claude Sonnet 5 (Anthropic, 2026b), Qwen3.8-Max (Yang et al., 2025a)) and three open-weight (MiniMax-M3 (MiniMax, 2026), Kimi K3 (Team et al., 2026), DeepSeek-V4-Flash (DeepSeek-AI et al., 2026)). Models are served via LiteLLM (BerriAI, 2026) and TokenRouter (TokenRouter, 2026), over one OpenAI-compatible chat-completions interface. Every model runs inside Stirrup (Artificial Analysis, 2026), a neutral open-source harness that recent agent benchmarks have also adopted (Esakkiraja et al., 2026b). All eight run under the same framework, environment, tools, evidence, and evaluator, at temperature 0 with a 30-turn limit.

**Success Criteria and Metrics.** We score every scenario with Pass@1 (Schmid, 2025; Sun et al., 2026b) against a criterion fixed before the run. Five of the seven scenario families (WO, FCC, HLT, FMSR, TSFM) require an exact match after deterministic parsing and normalization. CAR requires the intended mode, exactly one mode key, and the reference decision terms; its secondary score weights mode selection (.6) and term coverage (.4). FMEA scores semantic F1 over failure-mode sets and passes above .70. No language model judges an outcome. Appendices E and F give the full scoring rules, provider routing, and inference settings.

## 5.1 FULL-BENCHMARK MODEL EVALUATION

Table 2 puts the ceiling at 29.9% Pass@1 and the floor at 16.5%. Capability and price do not move together. Two models are non-dominated on completion against dollars per pass, Qwen3.8-Max and MiniMax-M3, and Qwen3.8-Max alone beats five of the seven others on both at once, both Claude models included. Its advantage stops at latency: 372 s per scenario against GPT-5.6's 131 s. We supplement Pass@1 with a Bradley–Terry Elo analysis (Bradley & Terry, 1952; Elo, 1967) as shown in Figure 5. Again, Qwen3.8-Max leads at 1659, followed by Claude Opus 5 at 1581 and Claude Sonnet 5 at 1545. Also, Figure 5 shows the bootstrap uncertainty around these estimates. The 307-point spread is thinner than it looks, because ratings fit only on discordant pairs: each adjacent comparison rests on 22–36 scenarios where exactly one model succeeds, and the higher-ranked model wins only 54.5–62.1% of them. Holm correction (Holm, 1979) across all 28 pairs leaves five significant separations, every one non-adjacent and spanning at least five ranks (Appendix F.3). Elo therefore groups models rather than ordering them: **Qwen3.8-Max, Claude Opus 5, and Claude Sonnet 5 lead without separating from one another**.

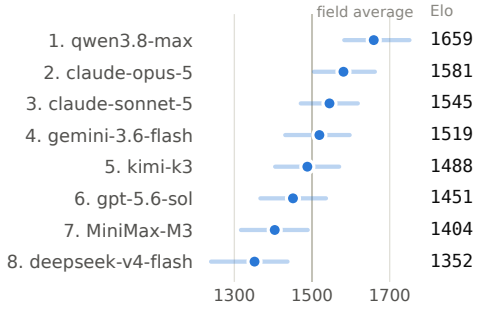



Figure 5: Bradley–Terry strengths. Horizontal bars show percentile-bootstrap 95% intervals from 1,000 scenario-level replicates.

Table 2: AssetArena results covering 231 scenarios. Execution cost is close to 5000$

| | Overall | | Pass@1 by workflow family (%) | | | | | | | Efficiency and cost | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Model** | **All** | **S** | **WO** | **CAR** | **FCC** | **HLT** | **FMSR** | **TSFM** | **FMEA** | **T/C ↓** | **Time ↓** | **Tok. ↓** | **$/P ↓** |
| *n* | *231* | | *65* | *50* | *27* | *10* | *32* | *30* | *17* | | | | |
| Qwen3.8-Max | **29.9** | **.442** | 5 | **42** | 56 | 60 | 38 | 17 | 41 | 14/27 | 372 | 471 | 3.5 |
| Claude Opus 5 | 26.8 | .411 | 2 | 38 | 37 | 60 | 41 | 20 | 41 | 11/20 | 173 | 445 | 9.2 |
| Claude Sonnet 5 | 24.7 | .389 | 6 | 24 | 52 | 60 | 31 | 13 | 41 | 19/36 | 784 | 805 | 9.2 |
| Gemini 3.6 Flash | 23.8 | .387 | 6 | 34 | 41 | 60 | 31 | 7 | 29 | 24/26 | 207 | 700 | 4.9 |
| Kimi K3 | 22.5 | .364 | 2 | 24 | 44 | 60 | 25 | 10 | **59** | 13/22 | 390 | 398 | 6.1 |
| GPT-5.6 | 20.8 | .345 | 6 | 16 | 33 | 60 | 31 | 13 | 41 | **8/17** | **131** | **204** | 5.4 |
| MiniMax-M3 | 19.0 | .324 | 3 | 24 | 37 | 70 | 28 | 3 | 18 | 15/22 | 197 | 496 | **0.9** |
| DeepSeek-V4-Flash | 16.5 | .275 | 6 | 18 | 30 | 30 | 28 | 7 | 18 | 19/31 | 133 | 677 | 1.9 |

*Notes. n*: scenarios per family. S: mean evaluator score. T/C, Time, Tok.: per-run means of turns/tool calls, runtime (s), and input tokens (k).

**Failure analysis.** We analyse all 1,848 saved trajectories, failed and malformed executions included. As shown in Table 3, format does not explain the shortfall. Two thirds of the 1,336 failed non-FMEA runs return a well-formed answer, and the schema-violating third averages .184 on partial exact match. Tool access does not produce grounding: all 87 failed FMEA runs invoke the required catalogue tool and 8 ground in an appropriate catalogue process. Interaction does not rescue a failing run: a run passes at 23.0% whether or not it hits an explicit tool error, while failed runs repeat an identical call 2.58 times as often as successful ones. Appendix G gives the full accounting.

Table 3: Failure analysis over all 1,848 trajectories (8 models × 231 scenarios).

| **Run accounting** | | **Failed FMEA runs** ($n = 87$) | |
|---|---|---|---|
| All runs | 1,848 | Invoked the required tool | **87 / 87 100%** |
| passing | 425 | Grounded in a valid process | **8 / 87 9.2%** |
| failing | 1,423 | **Failed CAR runs** | |
| FMEA | 87 | Mode-key correctness | .207 |
| non-FMEA | 1,336 | Decision-term coverage | .326 |
| Overall pass rate | 23.0% | **Interaction, failed vs. successful** | |
| **Output format, failed non-FMEA** | | Domain-tool calls | 16.46 / 14.81 |
| Well-formed | **892 66.8%** | Repeated identical calls | .85 / .33 |
| Missing or extra keys | 444 33.2% | Runs with ≥1 tool error | 55.7% / 55.8% |
| Partial EM, violating runs | .184 | Pass rate, error / no error | **23.0% / 23.0%** |

## 5.2 Execution-Harness Comparison

We compare Stirrup and OpenCode on the same 35-scenario stratified subset, holding the model, scenario environment, evidence, tools, and evaluation fixed. Under OpenCode, Pass@1 drops by 6.1 percentage points (19.6% relative) and mean score by 16.8%, corresponding to 17 fewer passes than Stirrup (Figure 6). This performance difference is roughly six times larger than the largest agent-interface effect in Section 5.3, measured on the same scenarios, models, and evaluator. OpenCode uses fewer turns and less runtime, but it makes more tool calls and consumes substantially more input tokens. **This result also justifies our selection of Stirrup for the main experiments.**

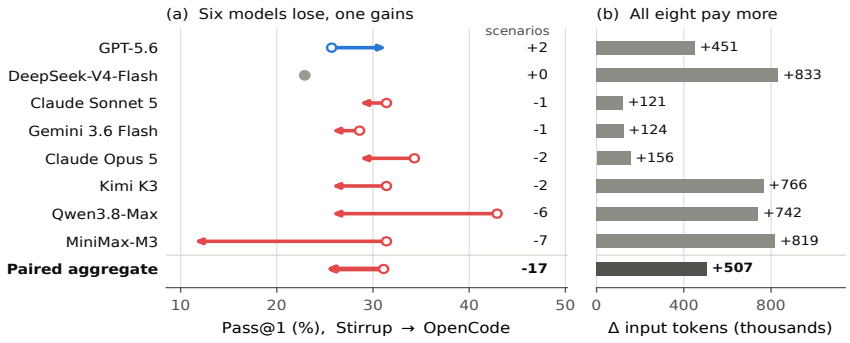



Figure 6: Execution harness: OpenCode against Stirrup.

## 5.3 Controlled Agent-Design Evaluation

We evaluate agent-side configurations on a fixed 35-scenario subset stratified across all seven workflow families, giving 280 matched model–scenario evaluations per configuration. Routed K1 skills, domain-sub-agent delegation, and MCP-gateway tool discovery each run against flat tool access with environment, model, evidence, and scoring held fixed. Architecture barely moves completion: K1 skills gain 2 scenarios, the gateway 3, sub-agents lose 3, and the K1 effect does not separate from flat access (two-sided exact McNemar, $p = .86$). Efficiency moves a great deal. The gateway cuts runtime 32.9% and input tokens 57.4%, and shrinks initial tool-schema exposure from 20,284 tokens to 307. K1 skills buy .022 of mean evaluator score for 53.7% more input tokens. Sub-agent delegation costs 137.4% more turns and 188.3% more tool calls while returning lower completion and score, so flat access dominates it outright. Appendix H gives the full tables.

The MCP gateway dominates all three alternatives when Pass@1 is maximised against input tokens (Figure 7). Substituting mean evaluator score admits K1 skills to the frontier alongside it, at roughly 3.6× the gateway's input tokens. On point estimates the gateway wins on efficiency, not on completion. That 66-fold reduction speaks directly to the context-bloat problem recent MCP work targets (Wang et al., 2026).

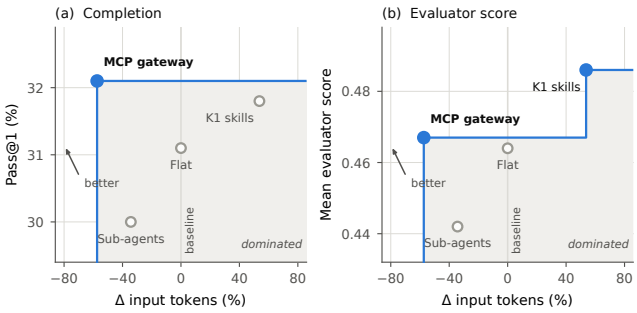


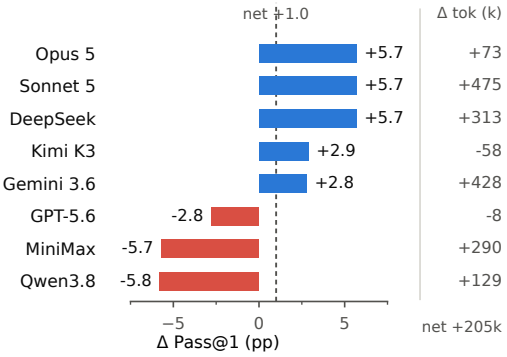

Figure 7: Empirical Pareto frontiers.

### 5.4 Execution Budget and Robustness

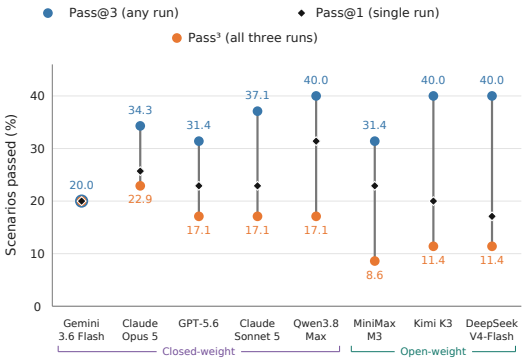
Any deployment can spend more compute at inference time in two ways: more turns inside a run, and more runs. We test both on the 35-scenario subset, holding everything else fixed.

**Turn budget.** Doubling Stirrup's interaction budget from 30 to 60 turns moves Pass@1 from 31.1% to 32.1% across 280 matched pairs, a net gain of three scenarios. The change is not decisive: 17 pairs succeed only at 60 turns and 14 only at 30 (two-sided exact McNemar, $p = .72$). Turns, tool calls, runtime, and input tokens rise by 33.8%, 25.9%, 34.1%, and 40.8%. The cost buys no predictable return, since the correlation between accuracy change and token cost across the eight models is $r = +0.26$ against a critical value of .707 at $n = 8$. Claude Opus 5 and Claude Sonnet 5 both gain two scenarios, for +73k and +475k input tokens (Figure 8a).

**Repeated execution.** Running each scenario three times separates retry potential from consistency. Closed-weight models leave a mean gap of 13.7 pp between Pass@3 and Pass$^3$, against 26.7 pp for open-weight models (Figure 8b). Gemini 3.6 Flash closes it entirely, solving the same seven scenarios on every run. The separation is not clean: Qwen3.8-Max sits at 22.9 pp beside MiniMax-M3 at 22.8 pp. Wide gaps mark scope for test-time scaling, but realizing it needs a mechanism that identifies a successful output, and Pass@3 measures retry potential rather than deployable selection.




(a) Turn budget, 30 to 60




(b) Repeated-run reliability

Figure 8: Execution Budget and Robustness: Neither budget converts into reliable completion.

## 6 Conclusion

AssetArena makes industrial asset operations executable and scorable: 231 human-authored scenarios over 85 MCP tools and six servers, each verified deterministically against a reference computed before its request was written. Eight frontier models produced 1,848 scored trajectories and left the suite unsaturated: the best completes 29.9%, and 126 of 231 scenarios defeat all eight. Across 280 matched pairs, four of five interventions moved completion by at most three scenarios and the fifth cost a net 17, while the MCP gateway cut input tokens 57.4% and initial tool-schema exposure 66-fold. Failure lands after tool access, not before it, so what is missing is grounded reasoning rather than plumbing; the 126 unsolved scenarios set the bar.

### AI USE STATEMENT

Generative AI tools were used only for language assistance, including grammar checking, improving clarity, and paraphrasing author-written text. Generative AI tools were not used to formulate the research questions, design the benchmark or experiments, generate scenarios or reference outcomes, execute experiments, analyze results, produce figures or tables, or determine the scientific claims of this paper. All AI-assisted wording was reviewed and revised by the authors. The authors take full responsibility for the final content of this work, including all text, claims, data, code, and artifacts.

### ETHICS STATEMENT

AssetArena is designed for research on reliable industrial-agent behavior and does not replace production decision-making. The benchmark evaluates agents in an isolated environment with scenario-scoped state, controlled tool access, and hidden reference outcomes. It contains no human-subject study and is not intended to make safety-critical operational decisions about real assets, workers, or facilities. The benchmark may nevertheless expose capabilities that could be misused if directly connected to enterprise systems. We therefore emphasize controlled evaluation, scenario isolation, replayable trajectories, and outcome verification rather than unrestricted deployment. Future benchmark extensions should continue to apply domain-expert review, access controls, privacy-preserving data practices, and security assessment for MCP-based enterprise integrations.

### REPRODUCIBILITY STATEMENT

AssetArena is designed to support reproducible evaluation through executable scenario environments, manifest-resolved evidence, versioned MCP tool implementations, hidden reference artifacts, deterministic scenario-specific scoring, and saved trajectories. Sections 4 and 5 describe the environment, evaluation protocol, model configuration, and primary results. The appendices document scenario construction, scoring rules, tool inventory, inference settings, subset selection, failure analysis, matched ablations, and execution-harness comparisons. The published benchmark materials include the scenario definitions, evaluation code, tool implementations, and scripts needed to reproduce the reported evaluations; saved trajectories allow rescoring without running model inference.

## REFERENCES

Kuangshi Ai, Haichao Miao, Kaiyuan Tang, Nathaniel Gorski, Jianxin Sun, Guoxi Liu, Helgi I. Ingolfsson, David Lenz, Hanqi Guo, Hongfeng Yu, Teja Leburu, Michael Molash, Bei Wang, Tom Peterka, Chaoli Wang, and Shusen Liu. Scivisagentbench: A benchmark for evaluating scientific data analysis and visualization agents, 2026. URL https://arxiv.org/abs/2603.29139.

Anthropic. Introducing claude opus 5. https://www.anthropic.com/news/claude-opus-5, 2026a. Accessed: 2026-09-22.

Anthropic. Introducing claude sonnet 5. https://www.anthropic.com/news/claude-sonnet-5, 2026b. Accessed: 2026-09-22.

Artificial Analysis. Stirrup: Ai benchmark leaderboard and evaluation suite. https://stirrup.artificialanalysis.ai/, 2026. Accessed: 2026-09-23.

Augury. Role-based AI agents for manufacturing: Accelerating operational performance. https://www.augury.com/blog/work-transformation-2/role-based-ai-agents-manufacturing/, 2026a. Vendor blog. Names the Reliability Agent, Maintenance Agents, Operations Agents and Industrial Data Exploration Agents. Accessed 2026-09-13.

Augury. AI for predictive maintenance solutions: Machine health. https://www.augury.com/machine-health/, 2026b. Vendor product page. Accessed 2026-09-04.

Joris Baan, Wilker Aziz, Barbara Plank, and Raquel Fernández. Clarify, abstain or answer? strategising in conversation with belief-augmented generation, 2026. URL https://arxiv.org/abs/2605.25831.

Songlin Bai, Xintong Wang, Linlin Yu, Bin Chen, Zhiang Xu, Yuyang Sheng, Changtong Zan, Xiaofeng Zhu, Yizhe Zhang, Jiru Li, Mingze Guo, Ling Zou, Yalong Li, Chengfu Huo, and Liang Ding. Industrybench: Probing the industrial knowledge boundaries of llms, 2026. URL https://arxiv.org/abs/2605.10267.

BerriAI. Litellm: Call all llm apis using the openai format. https://www.litellm.ai/, 2026. Accessed: 2026-09-22.

Tyler Bikaun, Tim French, Michael Stewart, Wei Liu, and Melinda Hodkiewicz. MaintIE: A fine-grained annotation schema and benchmark for information extraction from maintenance short texts. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pp. 10939–10951, 2024a. URL https://aclanthology.org/2024.lrec-main.954/.

Tyler Bikaun, Melinda Hodkiewicz, and Wei Liu. MaintNorm: A corpus and benchmark model for lexical normalisation and masking of industrial maintenance short text. In *Proceedings of the Ninth Workshop on Noisy and User-generated Text (W-NUT 2024)*, pp. 68–78. Association for Computational Linguistics, March 2024b. doi: 10.18653/v1/2024.wnut-1.7. URL https://aclanthology.org/2024.wnut-1.7/.

Ralph Allan Bradley and Milton E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952. ISSN 00063444, 14643510. URL http://www.jstor.org/stable/2334029.

C3 AI. C3 agentic AI platform. https://c3.ai/products/c3-agentic-ai-platform/, 2026a. Vendor product page. Accessed 2026-09-04.

C3 AI. C3 AI reliability: Predictive maintenance software to cut downtime. https://c3.ai/products/applications/c3-ai-reliability, 2026b. Vendor product page. Accessed 2026-09-04.

Jianlyu Chen, Yuyang Hu, Hongjin Qian, Jiawei Liu, Wenqing Wei, Xiaolong Chen, Defu Lian, Zhicheng Dou, Chaozhuo Li, Qiwei Ye, and Zheng Liu. Repo-to-skill: Distilling github repositories into ai4ai skills, 2026. URL https://arxiv.org/abs/2609.02749.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, Boshi Wang, Botao Yu, Yifei Li, Zeyi Liao, Chen Wei, Zitong Lu, Vishal Dey, Mingyi Xue, Frazier N. Baker, Benjamin Burns, Daniel Adu-Ampratwum, Xuhui Huang, Xia Ning, Song Gao, Yu Su, and Huan Sun. Scienceagentbench: Toward rigorous assessment of language agents for data-driven scientific discovery. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 96934–96990, 2025. URL https://iclr.cc/virtual/2025/poster/32108.

Christodoulos Constantinides, Dhaval C Patel, Shuxin Lin, Claudio Guerrero, SUNIL DAGAJI-RAO PATIL, and Jayant Kalagnanam. FailuresensorIQ: A multi-choice QA dataset for understanding sensor relationships and failure modes. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2026. URL https://openreview.net/forum?id=9KfkMAy2ut.

Narjes Davari, Bruno Veloso, Rita P. Ribeiro, Pedro Mota Pereira, and João Gama. Predictive maintenance based on anomaly detection using deep learning for air production unit in the railway industry. In *2021 IEEE 8th International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 1–10, 2021. doi: 10.1109/DSAA53316.2021.9564181. URL https://ieeexplore.ieee.org/document/9564181.

DeepSeek-AI, Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chengyu Hou, Chenhao Xu, Chenze Shao, Chong Ruan, Conner Sun, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Donghao Li, Dongjie Ji, Erhang Li, Fang Wei, Fangyun Lin, Fangzhou Yuan, Feiyu Xia, Fucong Dai, Guangbo Hao, Guanting Chen, Guoai Cao, Guolai Meng, Guowei Li, Han Yu, Han Zhang, Hanwei Xu, Hao Li, Haofen Liang, Haoling Zhang, Haoming Luo, Haoran Wei, Haotian Yuan, Haowei Zhang, Haowen Luo, Haoyu Chen, Haozhe Ji, Hengqing Zhang, Honghui Ding, Hongxuan Tang, Huanqi Cao, Huazuo Gao, Hui Qu, Hui Zeng, J Yang, JQ Zhu, Jia Luo, Jia Song, Jia Yu, Jialiang Huang, Jialu Cai, Jian Liang, Jiangting Zhou, Jiasheng Ye, Jiashi Li, Jiaxin Xu, Jiewen Hu, Jieyu Yang, Jin Chen, Jin Yan, Jingchang Chen, Jingli Zhou, Jingting Xiang, Jingyang Yuan, Jingyuan Cheng, Jingzi Zhou, Jinhua Zhu, Jiping Yu, Joseph Sun, Jun Ran, Junguang Jiang, Junjie Qiu, Junlong Li, Junmin Zheng, Junxiao Song, Kai Dong, Kaige Gao, Kang Guan, Kexing Zhou, Kezhao Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Wang, Leyi Xia, Li Zhang, Liang Zhao, Lihua Guo, Lingxiao Luo, Linwang Ma, Linyan Zhu, Litong Wang, Liyu Cai, Liyue Zhang, Longhao Chen, MS Di, MY Xu, Max Mei, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Mingxu Zhou, Minmin Han, Ning Wang, Panpan Huang, Panpan Wang, Peixin Cong, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qingyang Li, Qinyu Chen, Qiushi Du, Qiwei Jiang, Rui Tian, Ruifan Xu, Ruijie Lu, Ruiling Xu, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runqian Chen, Runqiu Yin, Runxin Xu, Ruomeng Shen, Ruoyu Zhang, Ruyi Chen, SH Liu, Shanghao Lu, Shangmian Sun, Shangyan Zhou, Shanhuang Chen, Shaofei Cai, Shaoheng Nie, Shaoqing Wu, Shaoyuan Chen, Shengding Hu, Shengyu Liu, Shiqiang Hu, Shirong Ma, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, Shuying Yu, Songyang Zhou, Tao Ni, Tao Yun, Tian Jin, Tian Pei, Tian Ye, Tianle Lin, Tianran Ji, Tianyi Cui, Tianyuan Yue, Tingting Yu, Tun Wang, W Zhang, WL Xiao, Wangding Zeng, Wei An, Weilin Zhao, Wen Liu, Wenfeng Liang, Wenjie Pang, Wenjing Luo, Wenjing Yao, Wenjun Gao, Wenkai Yang, Wenlve Huang, Wenqing Hou, Wentao Zhang, Wenting Ma, Xi Gao, Xiang He, Xiangwen Wang, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaokang Zhang, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingchen Liu, Xingkai Yu, Xingyou Li, Xinyu Yang, Xinyu Zhang, Xu Chen, Xuanyu Wang, Xuecheng Su, Xueyin Chen, Xuheng Lin, Xuwei Fu, YC Yan, YQ Wang, YW Ma, Yanfeng Luo, Yang Zhang, Yanhong Xu, Yanru Ma, Yanwen Huang, Yao Li, Yao Li, Yao Xu, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Qian, Yi Shao, Yi Yu, Yichao Zhang, Yifan Ding, Yifan Shi, Yijia Wu, Yiliang Xiong, Yiling Ma, Ying He, Ying Tang, Ying Zhou, Yingjia Luo, Yinmin Zhong, Yishi Piao, Yisong Wang, Yixiang Zhang, Yixiao Chen, Yixuan Tan, Yixuan Wei, Yiyang Ma, Yiyuan Liu, Yonglun Yang, Yongqiang Guo, Yongtong Wu, Yu Wu, YuKun Li, Yuan Cheng, Yuan Ou, Yuanfan Xu, Yuanhao Li, Yuduan Wang, Yuehan Yang, Yuer Xu, Yuhan Wu, Yuhao Meng, Yuheng Zou, Yukun Zha, Yunfan Xiong, Yupeng Chen, Yuping Lin, Yuqian Cao, Yuqian Wang, Yushun Zhang, Yuting Yan, Yutong Lin, Yuxian Gu, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuxuan Zhou, Yuyang Zhou, Yuzhen Huang, ZF Wu, Zehao Wang, Zehua Zhao, Zehui Ren, Zekai Zhang, Zhangli Sha, Zhe Fu, Zhe Ju, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zheren Gao, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhixian Huang, Zhixuan Chen, Zhiyu Wu, Zhizhou Ren, Zhongyu Wu, Zhuoshu Li, Zhuping Zhang, Zian Xu, Zihao Wang, Zihua Qu, Zihui Gu, Zijia Zhu, Zilin Li, Zipeng Zhang, Ziwei Xie, Ziyi Gao, Ziyi Wan, Zizheng Pan, and Zongqing Yao. Deepseek-v4: Towards highly efficient million-token context intelligence, 2026. URL https://arxiv.org/abs/2606.19348.

Mingxuan Du, Benfeng Xu, Chiwei Zhu, Licheng Zhang, Xiaorui Wang, and Zhendong Mao. Deep-research bench: A comprehensive benchmark for deep research agents. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://iclr.cc/virtual/2026/poster/10008065.

Alex Egg, Martin Iglesias Goyanes, Friso Kingma, Andreu Mora, Leandro von Werra, and Thomas Wolf. Dabstep: Data agent benchmark for multi-step reasoning, 2025. URL https://arxi

v.org/abs/2506.23719.

Arpad E. Elo. The proposed USCF rating system, its development, theory, and applications. *Chess Life*, 22(8):242–247, 1967.

Emerson Electric Co. Emerson expands Ovation AI portfolio with new intelligent agents. Press release, Pittsburgh, PA, July 2026. https://www.emerson.com/en/corporate/news/2026/emerson-expands-ovation-ai-portfolio-with-new-intelligent-agents. Accessed 2026-09-13.

Esakkivel Esakkiraja, Denis Akhiyarov, Vikas Yadav, Sai Rajeswar, Patrice Bechard, Sridhar Nemala, and Sagar Davasam. Starharness: Evolving harnesses with stratified search for enterprise environments, 2026a. URL https://arxiv.org/abs/2608.24804.

Esakkivel Esakkiraja, Denis Akhiyarov, Vikas Yadav, Sai Rajeswar, Patrice Bechard, Sridhar Nemala, and Sagar Davasam. Starharness: Evolving harnesses with stratified search for enterprise environments, 2026b. URL https://arxiv.org/abs/2608.24804.

GE Vernova. Asset performance management (APM) software. https://www.gevernova.com/software/products/asset-performance-management, 2026a. Vendor product page. Accessed 2026-09-04.

GE Vernova. Leading industrial AI in the energy sector. https://www.gevernova.com/software/blog/leading-industrial-ai-energy-sector, January 2026b. Vendor blog. Accessed 2026-09-04.

Google DeepMind. Gemini 3.6 flash model card. https://deepmind.google/models/model-cards/gemini-3-6-flash/, 2026. Accessed: 2026-09-22.

Zelin He, Boran Han, Xiyuan Zhang, Shuai Zhang, Haotian Lin, Qi Zhu, Haoyang Fang, Danielle C. Maddix, Abdul Fatir Ansari, Akash Chandrayan, Abhinav Pradhan, Bernie Wang, and Matthew Reimherr. SenTSR-Bench: Thinking with injected knowledge for time-series reasoning. 300: 4366–4374, 02–05 May 2026. URL https://proceedings.mlr.press/v300/he26c.html.

Melinda Hodkiewicz, Sarah Lukens, Michael P. Brundage, and Thurston Sexton. Rethinking maintenance terminology for an Industry 4.0 future. *International Journal of Prognostics and Health Management*, 12(1), 2021. doi: 10.36001/ijphm.2021.v12i1.2932. URL https://papers.phmsociety.org/index.php/ijphm/article/view/2932.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian journal of statistics*, pp. 65–70, 1979. URL http://www.jstor.org/stable/4615733.

Honeywell. Honeywell Forge: AI platform for autonomous industrial operations. https://www.honeywell.com/us/en/honeywell-forge, 2026. Vendor product page. Accessed 2026-09-04.

IBM. Introducing Maximo Application Suite 9.2: AI built for asset management. https://www.ibm.com/new/announcements/introducing-maximo-application-suite-9-2, June 2026a. Vendor announcement. Accessed 2026-09-04.

IBM. Maximo Application Suite: Trial and demonstration environment. https://www.ibm.com/products/maximo, 2026b. Vendor trial. Fourteen-day shared instance providing Manage and Health only; no API key is issued. Accessed 2026-09-04.

IBM. Working smarter with agentic workflows in Maximo Assistant. https://www.ibm.com/think/perspectives/working-smarter-agentic-workflows-maximo-assistant, August 2026c. Vendor documentation. Accessed 2026-09-04.

IBM. Asset performance management software: IBM Maximo Application Suite. https://www.ibm.com/products/maximo/predictive-maintenance, 2026d. Vendor product page. Accessed 2026-09-04.

Peter Jansen, Marc-Alexandre Côté, Tushar Khot, Erin Bransom, Bhavana Dalvi Mishra, Bodhisattwa Prasad Majumder, Oyvind Tafjord, and Peter Clark. Discoveryworld: A virtual environment for developing and evaluating automated scientific discovery agents. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 10088–10116. Curran Associates, Inc., 2024. doi: 10.52202/079017-0324. URL https://openreview.net/forum?id=cDYqckEt6d#discussion.

Saurabh Jha, Rohan R. Arora, Yuji Watanabe, Takumi Yanagawa, Yinfang Chen, Jackson Clark, Bhavya Bhavya, Mudit Verma, Harshit Kumar, Hirokuni Kitahara, Noah Zheutlin, Saki Takano, Divya Pathak, Felix George, Xinbo Wu, Bekir O Turkkan, Gerard Vanloo, Michael Nidd, Ting Dai, Oishik Chatterjee, Pranjal Gupta, Suranjana Samanta, Pooja Aggarwal, Rong Lee, Jae wook Ahn, Debanjana Kar, Amit Paradkar, Yu Deng, Pratibha Moogi, Prateeti Mohapatra, Naoki Abe, Chandrasekhar Narayanaswami, Tianyin Xu, Lav R. Varshney, Ruchi Mahindru, Anca Sailer, Laura Shwartz, Daby Sow, Nicholas C. M. Fuller, and Ruchir Puri. ITBench: Evaluating AI agents across diverse real-world IT automation tasks. In *Forty-second International Conference on Machine Learning*, 2025. URL https://icml.cc/virtual/2025/poster/44303.

Chenxi Jiang, Chuhao Zhou, and Jianfei Yang. REI-bench: Can embodied agents understand vague human instructions in task planning? In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://iclr.cc/virtual/2026/poster/10006785.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R Narasimhan. SWE-bench: Can language models resolve real-world github issues? In *The Twelfth International Conference on Learning Representations*, 2024. URL https://iclr.cc/virtual/2024/poster/18505.

Feiyang Kang, Hanze Li, Adam Nguyen, Mahavir Dabas, Jiaqi W. Ma, Frederic Sala, Dawn Song, and Ruoxi Jia. Can generalist agents automate data curation?, 2026. URL https://arxiv.org/abs/2606.04261.

Jeremiah Li and Andrew Ho. GeneBench-Pro: Evaluating multistage statistical reasoning in genomics, quantitative biology, and translational biomedicine. *bioRxiv*, 2026. doi: 10.64898/2026.06.29.735386. URL https://www.biorxiv.org/content/10.64898/2026.06.29.735386v1.

Xiangyi Li, Kyoung Whan Choe, Yimin Liu, Xiaokun Chen, Chujun Tao, Bingran You, Wenbo Chen, Zonglin Di, Jiankai Sun, Shenghan Zheng, Jiajun Bao, Yuanli Wang, Weixiang Yan, Yiyuan Li, and Hanchung Lee. Clawsbench: Evaluating capability and safety of LLM productivity agents in simulated workspaces. In *Third Conference on Language Modeling*, 2026a. URL https://openreview.net/forum?id=HjiNjTVs60.

Xiangyi Li, Yimin Liu, Wenbo Chen, Bingran You, Zonglin Di, Yifeng He, Shenghan Zheng, Kyoung Whan Choe, Jiankai Sun, Shuyi Wang, Chujun Tao, Binxu Li, Xuandong Zhao, Hejia Geng, Xiaojun Wu, Junwei Zhou, Xiaokun Chen, Hanwen Xing, Yubo Li, Qunhong Zeng, Di Wang, Yuanli Wang, Roey Ben Chaim, Penghao Jiang, Haotian Shen, Luyang Kong, Xinyi Liu, Runhui Wang, Xuanqing Liu, Jiachen Li, Xin Lan, Yueqian Lin, Wengao Ye, Junwei He, Songlin Li, Yue Zhang, Yipeng Gao, Yijiang Li, Ze Ma, Liqiang Jing, Tianyu Wang, Kaixin Li, Yiqi Xue, Haoran Lyu, Yizhuo He, Yuchen Tian, Shutong Wu, Bowei Wang, Yixuan Gao, Bo Chen, Litong Liu, Sikai Cheng, Jiajun Bao, Shuaicheng Tong, Shuwen Xu, Terry Yue Zhuo, Tinghan Ye, Qi Qi, Miao Li, Longtai Liao, Zelin Tan, Chang Shi, Xilin Tang, Srinath Tankasala, Boqin Yuan, Yaoyao Qian, Jianhong Tu, Chenguang Wang, Yizhou Sun, Wei Wang, Aaron Taylor, Ziyue Yang, Changkun Guan, Zhikang Dong, Xinyu Zhang, Steven Dillmann, Han chung Lee, and Dawn Song. Skillsbench: Benchmarking how well agent skills work across diverse tasks, 2026b. URL https://arxiv.org/abs/2602.12670.

Yusheng Li, Tianjun Feng, Yunfeng Chen, Chun-Yi Tsai, Yihan Sun, Ayan Das, Kaoutar El Maghraoui, Shuxin Lin, and Dhaval Patel. Phmforge: Evaluating llm agents on industrial prognostics through mcp-native, algorithm-grounded tools, 2026c. URL https://arxiv.org/abs/2604.01532.

Zhuochun Li, Youngmin Ko, Ali Keramati, Nicola Ferri, Susana Palmaz Lopez Pelaez, Liang-Chun Tsai, Calvin Wang, Mirco Milletari, Tuhin Kundu, Vadim Smolyakov, Kjartan Olafsson, and Tommy Guy. One success isn't reliability: Thinkingbox, a sandbox and benchmark for agents in stateful business workflows, 2026d. URL https://arxiv.org/abs/2608.19741.

Chengzhi Liu, Yichen Guo, Yepeng Liu, Yuzhe Yang, Qianqi Yan, Xuandong Zhao, Wenyue Hua, Sheng Liu, Sharon Li, Yuheng Bu, and Xin Eric Wang. Auditing agent harness safety, 2026a. URL https://arxiv.org/abs/2605.14271.

Tianyu Liu, Allen Xin Wang, Antonia Panescu, Lisa Xinyi Chen, Wenxin Long, Xinyu Wei, Yueqian Jing, Ziyao Zeng, Jihang Chen, Sihan Jiang, Ziqing Wang, Siyi Gu, Siyu Chen, Xinyang Hu, Haoran Shao, Leqi Xu, Wangjie Zheng, Zhiyuan Cao, Ada Fang, Botao Yu, Kunyang Sun, Rex Ying, Arman Cohan, Qingyu Chen, Lingzhou Xue, Kaize Ding, Yuanqi Du, Wengong Jin, Zhuoran Yang, Marinka Zitnik, James Zou, Hua Xu, and Hongyu Zhao. Benchmarking ai agents for addressing scientific challenges across scales, 2026b. URL https://arxiv.org/abs/2606.12736.

Sarah Lukens and Matt Markham. Data-driven application of PHM to asset strategies. In *Annual Conference of the PHM Society*, volume 10, 2018. URL https://papers.phmsociety.org/index.php/phmconf/article/view/245.

MathWorks. Inject faults in battery models. https://www.mathworks.com/help/simscape-battery/ug/add-faults-to-battery-models.html, 2026a. Simscape Battery Documentation.

MathWorks. Condition monitoring and prognostics using vibration signals. https://www.mathworks.com/help/predmaint/ug/condition-monitoring-and-prognostics-using-vibration-signals.html, 2026b. MATLAB and Simulink Documentation.

MathWorks. Model short-circuit fault in EV charging cable. https://www.mathworks.com/help/sps/ug/model-short-circuit-fault-in-ev-charging-cable.html, 2026c. Simscape Electrical Documentation.

MathWorks. Using Simulink to generate fault data. https://www.mathworks.com/help/predmaint/ug/Use-Simulink-to-Generate-Fault-Data.html, 2026d. MATLAB and Simulink Documentation.

MathWorks. Model faulted PMSM. https://www.mathworks.com/help/sps/ug/motor-pmsm-faulted.html, 2026e. Simscape Electrical Documentation.

Aritra Mazumder and Nusrat jahan Lia. Agentcheck: A reproduce-intervene-mitigate workbench for llm agents over mcp, 2026. URL https://arxiv.org/abs/2607.11098.

Mike Merrill, Alexander Shaw, Nicholas Carlini, Boxuan Li, Harsh Raj, Ivan Bercovich, Lin Shi, Jeong Shin, Thomas Walshe, E Kelly Buchanan, et al. Terminal-bench: Benchmarking agents on hard, realistic tasks in command line interfaces. In *International Conference on Learning Representations*, volume 2026, pp. 40903–40986, 2026. URL https://iclr.cc/virtual/2026/poster/10008736.

Yanis Merzouki, Coral Izquierdo, Matei Ignuta-Ciuncanu, Marcos Gomez-Bracamonte, Riccardo Maggioni, Alessandro Lombardi, Camilla Mazzoleni, Federico Martelli, Balazs Gunther, Jonas Petersen, and Philipp Petersen. Factorybench: Evaluating industrial machine understanding, 2026. URL https://arxiv.org/abs/2605.07675.

MiniMax. Introducing minimax-m3. https://www.minimax.io/blog/minimax-m3, 2026. Accessed: 2026-09-22.

OpenAI. Gpt-5.6 system card. Technical report, OpenAI, July 2026. URL https://deploymentsafety.openai.com/gpt-5-6/gpt-5-6.pdf. Accessed: 2026-09-22.

OpenCode AI Team. Opencode: Open ai coding agent system. https://opencode.ai/, 2026. Accessed: 2026-09-24.

Sungho Park, Wonjoong Kim, Rongyuan Tan, Jue Zhang, Wook-Shin Han, Pengfei Gao, Chanyoung Park, Yongqiang Yao, Rao Fu, Elsie Nallipogu, Qingwei Lin, Saravan Rajmohan, and Dongmei Zhang. Autosaddler: Automatic harness optimization with durable updates from agent execution traces, 2026. URL https://arxiv.org/abs/2608.23041.

Dhaval Patel, Shuxin Lin, James Rayfield, Nianjun Zhou, Chathurangi Shyalika, Suryanarayana R. Yarrabothula, Roman Vaculin, Natalia Martinez, Fearghal O'Donncha, and Jayant Kalagnanam. AssetOpsBench: A real-world evaluation benchmark for AI-driven task automation in industrial asset management. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining, V.2*, pp. 9649–9660, 2026a. doi: 10.1145/3770855.3817562. URL https://doi.org/10.1145/3770855.3817562.

Dhaval Patel, Chathurangi Shyalika, Suryanarayana Reddy Yarrabothula, Ling Yue, Shuxin Lin, Nianjun Zhou, and James Rayfield. Results and retrospective analysis of the cods 2025 assetopsbench challenge, 2026b. URL https://arxiv.org/abs/2605.08518.

Debjit Paul, Daniel Murphy, Milan Gritta, Ronald Cardenas Acosta, Victor Prokhorov, Lena Sophia Bolliger, Aysim Toker, Roy Miles, Andreea-Maria Oncescu, Jasivan Sivakumar, et al. A benchmark for deep information synthesis. In *International Conference on Learning Representations*, volume 2026, pp. 65663–65684, 2026. URL https://iclr.cc/virtual/2026/poster/10011957.

Leonardo Ranaldi and Giulia Pucci. Learning multilingual agentic policy to control sycophancy. In Vera Demberg, Kentaro Inui, and Lluís Marquez (eds.), *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3664–3681, Rabat, Morocco, March 2026. Association for Computational Linguistics. ISBN 979-8-89176-380-7. doi: 10.18653/v1/2026.eacl-long.169. URL https://aclanthology.org/2026.eacl-long.169/.

Rockwell Automation. Rockwell Automation and Augury partner to improve industrial performance with agentic AI. https://www.rockwellautomation.com/en-us/company/news/press-releases/rockwell-automation-and-augury-partner-to-improve-industrial-performance-with-agentic-ai.html, July 2026. Vendor press release. Accessed 2026-09-04.

Salesforce. Five signs you need agent-first field service. https://www.salesforce.com/blog/agent-first-field-service/, March 2025. Vendor blog. Accessed 2026-09-04.

Salesforce. Agentforce mcp support, 2026. https://www.salesforce.com/agentforce/mcp-support/.

SAP. SAP asset performance management. https://www.sap.com/products/scm/apm.html, 2026a. Vendor product page. Accessed 2026-09-04.

SAP. Joule agents: AI use cases for every business function. https://www.sap.com/products/artificial-intelligence/ai-agents/agent-use-cases.html, 2026b. Vendor product catalogue. Accessed 2026-09-04.

Philipp Schmid. Pass@k vs passˆk: Understanding agent reliability. https://www.philschmid.de/agents-pass-at-k-pass-power-k, March 2025. Accessed: 2026-09-23.

Siemens AG. Siemens introduces AI agents for industrial automation. https://press.siemens.com/global/en/pressrelease/siemens-introduces-ai-agents-industrial-automation, May 2025. Vendor press release. Names the Design, Planning, Engineering, Operations and Maintenance Copilots; the Maintenance Copilot is the Senseye offering. Accessed 2026-09-13.

Siemens AG. Senseye predictive maintenance. https://www.siemens.com/en-us/products/industrial-digitalization-services/senseye-predictive-maintenance/, 2026. Vendor product page. Accessed 2026-09-04.

Devin Yasith De Silva, Dhaval Patel, Christodoulos Constantinides, Shuxin Lin, Nianjun Zhou, Paul J Adams, Sal Rosato, Nicolas Constantinides, Deborah L. McGuinness, and Jayant Kalagnanam. Diagnosticiq: A benchmark for llm-based industrial maintenance action recommendation from symbolic rules, 2026. URL https://arxiv.org/abs/2605.08614.

Shankari Sivanathan. Proactive maintenance made easy in field service with Agentforce. https://www.salesforce.com/blog/proactive-maintenance/, May 2025. Vendor blog. Accessed 2026-09-04.

Depeng Su, Yuyu Luo, and Guobiao Hu. Vehbench: A stage-local diagnostic benchmark for llm-assisted vibration energy harvester design, 2026. URL https://arxiv.org/abs/2607.18181.

G. P. Sullivan, R. Pugh, A. P. Melendez, and W. D. Hunt. Operations & maintenance best practices: A guide to achieving operational efficiency (release 3.0). Technical Report PNNL-19634, Pacific Northwest National Laboratory, August 2010. Prepared for the U.S. Department of Energy Federal Energy Management Program. Chapter 5, Section 5.2.

Yiyou Sun, Xinyang Han, Weichen Zhang, Yuanbo Pang, Tianyu Wang, Yuhan Cao, Yixiao Huang, Chris Duroiu, Haoyun Zhang, Jeffrey Lin, Weishu Zhang, Tyler Zeng, Ying Yan, Bo Liu, Hanson Wen, Mingyang Xu, Xiaoyuan Liu, Zimeng Chen, Weiyan Shi, Amanda Dsouza, Vincent Sunn Chen, Patrick Bryant, Carl Boettiger, Yamini Rangan, Bradley Rothenberg, Kyle Steinfeld, Arvind Rao, Tapio Schneider, Georgios Yannakakis, Laure Zanna, Kaan Ozbay, Ida Sim, Tarek Zohdi, George Em Karniadakis, Jack Gallant, Teresa Head-Gordon, Yushan Li, Wenxi Deng, Tao Sun, Huiqi Wang, Zhun Wang, Justin Xu, Chris Yuhao Liu, Yafei Cheng, Rongwang Hu, Aras Bacho, Shengcao Cao, Zengyi Qin, Yixiong Chen, Hengduan Fan, Hao Liu, Lin Zeng, Shashank Muralidhar Bharadwaj, Litian Gong, Yingxuan Yang, Maojia Song, Ruheng Wang, Zongzheng Zhang, Honglin Bao, Shuo Lu, Jianhong Tu, Zhonghua Wang, Zheng Zhang, Zijiao Chen, Yanqiong Jiang, Zhendong Li, Bohan Lyu, Chang Ma, Peiran Xu, Benran Zhang, Shangding Gu, Haoyue Hua, Haoyang Li, Wanzhe Liao, Chengzhi Liu, Junbo Peng, Haoran Sun, Zechen Xu, Bo Chen, Jiayi Cheng, Yi Jiang, Keying Kuang, Yuan Li, Youbang Pan, Ziyan Rao, Alexander Schubert, Yifan Shen, Vincent Siu, Xiatao Sun, Kangqi Zhang, Xiaopan Zhang, Yuchen Zhu, Ishaan Singh Chandok, Lei Ding, Jingxuan Fan, Andrew Glover, Jiaming Hu, Yiran Hu, Wenbo Huang, Zixin Jiang, Haoran Jin, Lukas Kim, Ming Liu, Yang Liu, Alireza Rafiei, Xuhuan Shen, Kunyang Sun, Sophia Sun, Ting Sun, Eric Wang, Yixin Wang, Hanwen Xing, Sihan Xu, Yuzheng Xu, Zhongxing Xu, Zhiling Yan, Boqin Yuan, Ruiqi Zhang, Yifan Zhang, Zibo Zhao, Liana, Santanu Bosu Antu, Haoyue Bai, Carlo Bosio, Joseph Cavanagh, Patricia Cavazos-Rehg, Tianxing Chen, Xuewen Chen, Yipu Chen, Chenyu Zhu, Chen Dai, Stefano De Castro, Yunfu Deng, Kaustubh Dhole, Jiayuan Ding, Chenchen Du, Zhehang Du, Hao Fan, Run-Ze Fan, Hengyu Fu, Shi Gu, Yifan Gu, Charlie Guo, Baihe Huang, Baixiang Huang, Rimika Jaiswal, Zhihan Jiang, Ran Jin, Erin Kasson, Xin Lan, Joseph Lee, Deren Lei, Chenyu Li, Daofeng Li, Haitao Li, Hongwei Li, Jingyan Li, Xiao Li, Yi Li, Yinsheng Li, Yuangang Li, Zhixu Li, Wenyu Liang, Longtai Liao, Kevin Qinghong Lin, Andy Zeyi Liu, Che Liu, Jiaming Liu, Kaiyuan Liu, Xuan Liu, Pan Lu, Wenbo Lv, Yicheng Lyu, Qiuyang Mang, Kyle Montgomery, Yuzhou Nie, Ruoxi Ning, Jorin Overwiening, Xu Pan, Layna Paraboschi, Core Francisco Park, Justin Purnomo, Swati Rajwal, Scott Rankin, Bixuan Ren, Yiren Rong, HaoYang Shang, Ventus Shaw, Fiona Shen, Jiawei Shen, Minqi Shi, Shi Qiu, Huaxiu Yao, Tianneng Shi, Jonah So, Vladislav Susoy, Hannah Szlyk, Haocheng Wang, Jialu Wang, Wei Wang, Xinyu Wang, Zehao Wang, Dowling Wong, Angela Wu, Dehao Wu, Fangyu Wu, Mengyuan "Millie" Wu, Yu Wu, Yuchen Wu, Yuhao Wu, Qingpo Wuwu, Weihang Xiao, Yongyi Xiong, Fan Xu, Ruiling Xu, Mingxuan Yan, Benjamin Yang, Jirong Yang, Sen Yang, Xiaoli Yang, Yushi Yang, Haoran Ye, Xiaohu Yu, Zhengming Yu, Chenlong Zhang, Chi Zhang, Hanning Zhang, Hanwen Zhang, Junge Zhang, Kunpeng Zhang, Song Zhang, Wenjin Zhang, Wenshuo Zhang, Ying Zhang, Yizhi Zhang, Brian Zhao, Qijian Zhao, Yimin Zhao, Yuhaohua Zheng, Liwei Zhou, Tianyue Zhou, Sichen Zhu, Siqi Zhu, Yan Zhu, Yishu Zhu, Jierui Zuo, Chonghao Cai, Helena Casademunt, Wenjia Chen, Cheng Cheng, Nawen Deng, Rao Fu, Tianfu Fu, Yifan Han, He Ren, Zhenyu He, Qiao Jin, Langlang Li, Yuetai Li, Sylvia Liu, Lu Lu, Luqing Zhou, Subhabrata Mukherjee, Yunqi Ouyang, Yin Ren, Dawei Shi, Haoran Wu, Zhiyue Wu, Hannah Yao, Zhuoran Yi, Jenny Yu, Rhea Zhan, Hang Zhou, Blake Zhu, Junfan Zhu, Alan Yuille, Yang Liu, Russell Alan Poldrack, Jiachen Li, Zhenglu Li, Molei Tao, Jing Huang, Wenqi Shi, Costas Spanos, Lichao Sun, Chenguang Wang, Orson Xu, Zhen Dong, Hector Gomez, Aylin

Caliskan, Ali Emami, Haimin Hu, Zhi Li, Lihui Liu, Murphy Niu, Yi Shao, Jianxin Sun, Mikko Tolonen, Ting Wang, Sanjiv Das, Yanjun Gao, Wenbo Guo, Erika J Schneider, Zhiyong Lu, Yian Ma, Mark Mueller, Radha Poovendran, Somayeh Sojoudi, Yinglun Zhu, and Dawn Song. Agents' last exam, 2026a. URL https://arxiv.org/abs/2606.05405.

Yiyou Sun, Xinyang Han, Weichen Zhang, Yuanbo Pang, Tianyu Wang, Yuhan Cao, Yixiao Huang, Chris Duroiu, Haoyun Zhang, Jeffrey Lin, Weishu Zhang, Tyler Zeng, Ying Yan, Bo Liu, Hanson Wen, Mingyang Xu, Xiaoyuan Liu, Zimeng Chen, Weiyan Shi, Amanda Dsouza, Vincent Sunn Chen, Patrick Bryant, Carl Boettiger, Yamini Rangan, Bradley Rothenberg, Kyle Steinfeld, Arvind Rao, Tapio Schneider, Georgios Yannakakis, Laure Zanna, Kaan Ozbay, Ida Sim, Tarek Zohdi, George Em Karniadakis, Jack Gallant, Teresa Head-Gordon, Yushan Li, Wenxi Deng, Tao Sun, Huiqi Wang, Zhun Wang, Justin Xu, Chris Yuhao Liu, Yafei Cheng, Rongwang Hu, Aras Bacho, Shengcao Cao, Zengyi Qin, Yixiong Chen, Hengduan Fan, Hao Liu, Lin Zeng, Shashank Muralidhar Bharadwaj, Litian Gong, Yingxuan Yang, Maojia Song, Ruheng Wang, Zongzheng Zhang, Honglin Bao, Shuo Lu, Jianhong Tu, Zhonghua Wang, Zheng Zhang, Zijiao Chen, Yanqiong Jiang, Zhendong Li, Bohan Lyu, Chang Ma, Peiran Xu, Benran Zhang, Shangding Gu, Haoyue Hua, Haoyang Li, Wanzhe Liao, Chengzhi Liu, Junbo Peng, Haoran Sun, Zechen Xu, Bo Chen, Jiayi Cheng, Yi Jiang, Keying Kuang, Yuan Li, Youbang Pan, Ziyan Rao, Alexander Schubert, Yifan Shen, Vincent Siu, Xiatao Sun, Kangqi Zhang, Xiaopan Zhang, Yuchen Zhu, Ishaan Singh Chandok, Lei Ding, Jingxuan Fan, Andrew Glover, Jiaming Hu, Yiran Hu, Wenbo Huang, Zixin Jiang, Haoran Jin, Lukas Kim, Ming Liu, Yang Liu, Alireza Rafiei, Xuhuan Shen, Kunyang Sun, Sophia Sun, Ting Sun, Eric Wang, Yixin Wang, Hanwen Xing, Sihan Xu, Yuzheng Xu, Zhongxing Xu, Zhiling Yan, Boqin Yuan, Ruiqi Zhang, Yifan Zhang, Zibo Zhao, Liana, Santanu Bosu Antu, Haoyue Bai, Carlo Bosio, Joseph Cavanagh, Patricia Cavazos-Rehg, Tianxing Chen, Xuewen Chen, Yipu Chen, Chenyu Zhu, Chen Dai, Stefano De Castro, Yunfu Deng, Kaustubh Dhole, Jiayuan Ding, Chenchen Du, Zhehang Du, Hao Fan, Run-Ze Fan, Hengyu Fu, Shi Gu, Yifan Gu, Charlie Guo, Baihe Huang, Baixiang Huang, Rimika Jaiswal, Zhihan Jiang, Ran Jin, Erin Kasson, Xin Lan, Joseph Lee, Deren Lei, Chenyu Li, Daofeng Li, Haitao Li, Hongwei Li, Jingyan Li, Xiao Li, Yi Li, Yinsheng Li, Yuangang Li, Zhixu Li, Wenyu Liang, Longtai Liao, Kevin Qinghong Lin, Andy Zeyi Liu, Che Liu, Jiaming Liu, Kaiyuan Liu, Xuan Liu, Pan Lu, Wenbo Lv, Yicheng Lyu, Qiuyang Mang, Kyle Montgomery, Yuzhou Nie, Ruoxi Ning, Jorin Overwiening, Xu Pan, Layna Paraboschi, Core Francisco Park, Justin Purnomo, Swati Rajwal, Scott Rankin, Bixuan Ren, Yiren Rong, HaoYang Shang, Ventus Shaw, Fiona Shen, Jiawei Shen, Minqi Shi, Shi Qiu, Huaxiu Yao, Tianneng Shi, Jonah So, Vladislav Susoy, Hannah Szlyk, Haocheng Wang, Jialu Wang, Wei Wang, Xinyu Wang, Zehao Wang, Dowling Wong, Angela Wu, Dehao Wu, Fangyu Wu, Mengyuan "Millie" Wu, Yu Wu, Yuchen Wu, Yuhao Wu, Qingpo Wuwu, Weihang Xiao, Yongyi Xiong, Fan Xu, Ruiling Xu, Mingxuan Yan, Benjamin Yang, Jirong Yang, Sen Yang, Xiaoli Yang, Yushi Yang, Haoran Ye, Xiaohu Yu, Zhengming Yu, Chenlong Zhang, Chi Zhang, Hanning Zhang, Hanwen Zhang, Junge Zhang, Kunpeng Zhang, Song Zhang, Wenjin Zhang, Wenshuo Zhang, Ying Zhang, Yizhi Zhang, Brian Zhao, Qijian Zhao, Yimin Zhao, Yuhaohua Zheng, Liwei Zhou, Tianyue Zhou, Sichen Zhu, Siqi Zhu, Yan Zhu, Yishu Zhu, Jierui Zuo, Chonghao Cai, Helena Casademunt, Wenjia Chen, Cheng Cheng, Nawen Deng, Rao Fu, Tianfu Fu, Yifan Han, He Ren, Zhenyu He, Qiao Jin, Langlang Li, Yuetai Li, Sylvia Liu, Lu Lu, Luqing Zhou, Subhabrata Mukherjee, Yunqi Ouyang, Yin Ren, Dawei Shi, Haoran Wu, Zhiyue Wu, Hannah Yao, Zhuoran Yi, Jenny Yu, Rhea Zhan, Hang Zhou, Blake Zhu, Junfan Zhu, Alan Yuille, Yang Liu, Russell Alan Poldrack, Jiachen Li, Zhenglu Li, Molei Tao, Jing Huang, Wenqi Shi, Costas Spanos, Lichao Sun, Chenguang Wang, Orson Xu, Zhen Dong, Hector Gomez, Aylin Caliskan, Ali Emami, Haimin Hu, Zhi Li, Lihui Liu, Murphy Niu, Yi Shao, Jianxin Sun, Mikko Tolonen, Ting Wang, Sanjiv Das, Yanjun Gao, Wenbo Guo, Erika J Schneider, Zhiyong Lu, Yian Ma, Mark Mueller, Radha Poovendran, Somayeh Sojoudi, Yinglun Zhu, and Dawn Song. Agents' last exam, 2026b. URL https://arxiv.org/abs/2606.05405.

Jun Takahashi, Atsunori Moteki, Akiyoshi Uchida, Shoichi Masui, Fan Yang, Kanji Uchino, Yueqi Song, Yonatan Bisk, Graham Neubig, Ikuo Kusajima, Yasuto Watanabe, Hiroyuki Ishida, Koki Nakagawa, and Shan Jiang. Fieldworkarena: Agentic ai benchmark for real field work tasks. In *Pattern Recognition: 28th International Conference, ICPR 2026, Lyon, France, August 17–22, 2026, Proceedings, Part IV*, pp. 94–108, Berlin, Heidelberg, 2026. Springer-Verlag. ISBN 978-3-032-31662-2. doi: 10.1007/978-3-032-31663-9_7. URL https://doi.org/10.1007/978-3-032-31663-9_7.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, M. C., Jianfeng Cai, Xinyuan Cai, Peizhou Cao, Yuxuan Cao, Ziwei Chai, Y. Charles, H. S. Che, Guanduo Chen, Guangyu Chen, Guanzheng Chen, Huarong Chen, Jia Chen, Jianlong Chen, Jun Chen, Kexin Chen, Peng Chen, Ruijue Chen, Wentao Chen, Xin Chen, Yang Chen, Yanru Chen, Yifei Chen, Yingjiang Chen, Yuankun Chen, Yujie Chen, Yutian Chen, Zhirong Chen, Dazhi Cheng, Yean Cheng, Jialei Cui, Jingbing Cui, Anqi Dai, Jiaqi Deng, Hao Ding, Rui Ding, Shaofeng Ding, Mengfan Dong, Mengnan Dong, Yuhao Dong, Yuxin Dong, Angang Du, Chenzhuang Du, Dikang Du, Jusen Du, Yulun Du, Yu Fan, Jing Feng, Qiulin Feng, Yichen Feng, Kelin Fu, Qiang Fu, Fuxuan Gao, Hongcheng Gao, Jingyue Gao, Tong Gao, Weijia Gao, Shangyi Geng, Jie Gong, Linhu Gong, Shengao Gong, Xiaochen Gong, Qizheng Gu, Yicheng Gu, Shuhao Guan, Haiqing Guo, Shiqi Guo, Xiang Guo, Zhengyan Guo, Beixi Hao, Wenxin Hao, Xiaoru Hao, Dailan He, Haotian He, Lehan He, Qi He, Weiran He, Xinran He, Xinyi He, Yibo He, Yunjia He, Chao Hong, Tiange Hong, Hao Hu, Jiaxi Hu, Ruikun Hu, Weiming Hu, Yangyang Hu, Zhenxing Hu, Liang Hua, Jinbin Huang, Ke Huang, Ruiyuan Huang, Siying Huang, Weixiao Huang, Yan Huang, Zhengjie Huang, Zhiqi Huang, Yulong Hui, Chaobo Jia, Yutong Jiang, Zhejun Jiang, Zuoyou Jiang, Wenyi Jin, Xinyi Jin, Yu Jing, Huanjun Kong, Guokun Lai, Aidi Li, Cheng Li, Chengyuan Li, Cong Li, Fang Li, Guanyu Li, Haoyang Li, Jia Li, Junxiong Li, Lei Li, Letian Li, Lincan Li, Weihong Li, Wentao Li, Xintong Li, Yang Li, Yishen Li, Yiwei Li, Yuxiao Li, Zhaowei Li, Zhaoxi Li, Zheming Li, Zhengxiao Li, Zhiyuan Li, Jiawei Lin, Xiaohan Lin, Yibo Lin, Zichao Lin, Ziyan Lin, Bill Liu, Boxiao Liu, Chuan Liu, Liang Liu, Shaowei Liu, Shudong Liu, Shuran Liu, Tianwei Liu, Weizhou Liu, Yangyang Liu, Yanming Liu, Yibo Liu, Yipeng Liu, Zhengying Liu, Zhiheng Liu, Enzhe Lu, Haoyu Lu, Linqiang Lu, Tingzhan Lu, Zhiyuan Lu, Aotian Luo, G. Luo, Junyu Luo, Yifan Luo, B. Lyu, Wenzhou Lyu, Shaoguang Mao, Yuan Mei, Xin Men, Minqing Ni, Yixuan Niu, Siyuan Pan, Shujun Peng, Zhangyang Qi, Ruoyu Qin, ZeChao Qin, Zeyu Qin, Haiquan Qiu, Jianxin Qiu, Jiezhong Qiu, Bowen Qu, Yuhao Qu, Zeyu Shang, Youbo Shao, Han Shen, Jincheng Shi, Juanfeng Shi, Lidong Shi, Shengyuan Shi, Wingchun Siu, Pengwei Song, Xiaoxi Song, Jianlin Su, Yunfeng Su, Zhaochen Su, Lin Sui, Jingsong Sun, Junyao Sun, Shaoning Sun, Shuzhe Sun, Tongyu Sun, Yujun Sun, Yunpeng Tai, Chuning Tang, Heyi Tang, Sirui Tang, Zecheng Tang, Chaoran Tian, Rongpeng Tian, Yu Tian, Wei Tu, Chensi Wang, Chuang Wang, Chunjie Wang, Dinglu Wang, Feng Wang, Hailong Wang, Haiming Wang, Hao Wang, Hao Wang, Huaqing Wang, Hui Wang, Jiayi Wang, Jinglong Wang, Jinhong Wang, Jiuzheng Wang, Linian Wang, Shaobo Wang, Shenzhi Wang, Shuyi Wang, Si Wang, Siyuan Wang, Tianfu Wang, Wenjue Wang, Xingran Wang, Xinmei Wang, Xinyuan Wang, Xusheng Wang, Yalin Wang, Yangkun Wang, Yao Wang, Yaoyu Wang, Yejie Wang, Yiqin Wang, Yucheng Wang, Yuzhi Wang, Zhaoji Wang, Zhaowei Wang, Zhengtao Wang, Zhenhao Wang, Zhongsheng Wang, Zifan Wang, Chu Wei, Ming Wei, Shouxin Wei, Zichen Wen, Fan Wu, Haoning Wu, Rucong Wu, Wenhao Wu, Xiaoxue Wu, Yingcong Wu, Yongqi Wu, Yuxin Wu, Zijian Wu, Xinglang Xian, Chenxuan Xiang, Yuye Xiang, Bocheng Xiao, Chenjun Xiao, Xin Xiao, Jin Xie, Xiaotong Xie, Yifeng Xie, Zhe Xie, Bowei Xing, Yiming Xiong, Baosheng Xu, Boyu Xu, Jiale Xu, Jianfan Xu, Jing Xu, Jinjing Xu, L. H. Xu, Qingtao Xu, Shuyao Xu, Suting Xu, Tiantian Xu, Tianxiang Xu, Weixin Xu, Xinran Xu, Yangchuan Xu, Ye Xu, Yueni Xu, Ziyao Xu, Haonan Xue, Junjie Yan, Yaoyao Yan, Fan Yang, Guangyao Yang, Hao Yang, Junwei Yang, Ruoyu Yang, Wenjie Yang, Xiaofei Yang, Xinyu Yang, Yi Yang, Yiling Yang, Ying Yang, Yuchen Yang, Zhen Yang, Zhilin Yang, Zian Yang, Zuhao Yang, Haotian Yao, Dan Ye, Haoran Ye, Wenjie Ye, Zhanbo Ye, Bohong Yin, Haoxiang Yin, Xietong Yin, Chengzhen Yu, Haozhen Yu, Longhui Yu, Shengnan Yu, Shuying Yu, Tianxiang Yu, Enming Yuan, Mengjie Yuan, Tongtian Yue, Wei Yue, Yang Yue, Dunyuan Zha, Haobing Zhan, B. H. Zhang, Dehao Zhang, Fei Zhang, Hao Zhang, Haoyuan Zhang, Huanyu Zhang, Jiapei Zhang, Jiaxuan Zhang, Jin Zhang, Kaiyi Zhang, Miaozhen Zhang, Puqi Zhang, Qinglei Zhang, Rong Zhang, Rui Zhang, Shaoshuai Zhang, Shiyi Zhang, Xiaobin Zhang, Xiaoyun Zhang, Y. Zhang, Yangkun Zhang, Ye Zhang, Yichi Zhang, Yikun Zhang, Yizhi Zhang, Yongting Zhang, Yu Zhang, Yutao Zhang, Yutong Zhang, Zheng Zhang, Zijing Zhang, Bin Zhao, Chenguang Zhao, Feifan Zhao, Jinglun Zhao, Jinxiang Zhao, Shuai Zhao, Wenshuo Zhao, Xiangyu Zhao, Xuanle Zhao, Yikai Zhao, Zijia Zhao, Haozhi Zheng, Huabin Zheng, Ruihan Zheng, Shaojie Zheng, Tengyang Zheng, Haofeng Zhong, Lei Zhong, Longguang Zhong, M. Zhou, Qiankang Zhou, Runjie Zhou, Ruozhang Zhou, Xinyu Zhou, Yiqiao Zhou, Zaida Zhou, Jinguo Zhu, Liya Zhu, Xinhao Zhu, Yangjunfeng Zhu, Yuxuan Zhu, Zhen Zhu, Chen Zhuang, Weiyu Zhuang, and Xinxing Zu. Kimi k3: Open frontier intelligence, 2026. URL https://arxiv.org/abs/2607.24653.

The Linux Foundation. Linux foundation announces the formation of the agentic ai foundation (aaif), anchored by new project contributions including model context protocol (mcp), goose and agents.md. https://www.linuxfoundation.org/press/linux-foundation-announces-the-formation-of-the-agentic-ai-foundation, December 2025. Accessed 26 September 2026.

TokenRouter. Tokenrouter: Unified api gateway for frontier models. https://www.tokenrouter.com/, 2026. Accessed: 2026-09-22.

Bruno Veloso, Rita P. Ribeiro, João Gama, and Pedro Mota Pereira. The metropt dataset for predictive maintenance. *Scientific Data*, 9(1):764, 2022. doi: 10.1038/s41597-022-01877-3. URL https://doi.org/10.1038/s41597-022-01877-3.

Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. ScienceWorld: Is your agent smarter than a 5th grader? In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 11279–11298, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.775. URL https://aclanthology.org/2022.emnlp-main.775/.

Zhenting Wang, Qi Chang, Hemani Patel, Shashank Biju, Cheng-En Wu, Quan Liu, Aolin Ding, Alireza Rezazadeh, Ankit Shah, Yujia Bao, and Eugene Siow. MCP-bench: Benchmarking tool-using LLM agents with complex real-world tasks via MCP servers. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=fe8mzHwMxN.

Bosi Wen, Pei Ke, Xiaotao Gu, Lindong Wu, Hao Huang, Jinfeng Zhou, Wenchuang Li, Binxin Hu, Wendy Gao, Jiaxin Xu, Yiming Liu, Jie Tang, Hongning Wang, and Minlie Huang. Benchmarking complex instruction-following with multiple constraints composition. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 137610–137645. Curran Associates, Inc., 2024. doi: 10.52202/079017-4371. URL https://neurips.cc/virtual/2024/poster/97675.

Jingxu Xie, Dylan Xu, Xuandong Zhao, and Dawn Song. Agentsynth: Scalable task generation for generalist computer-use agents. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 155149–155177, 2026. URL https://iclr.cc/virtual/2026/poster/10010827.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025a. URL https://arxiv.org/abs/2505.09388.

Puyu Yang, Laifa Tao, Zijian Huang, Haifei Liu, Wenyan Cao, Hao Ji, Jianan Qiu, Qixuan Huang, Xuanyuan Su, Yuhang Xie, Jun Zhang, Shangyu Li, Chen Lu, and Zhixuan Lian. Phm-bench: A domain-specific benchmarking framework for systematic evaluation of large models in prognostics and health management, 2025b. URL https://arxiv.org/abs/2508.02490.

Qingren Yao, Yaxuan Kong, Yuqi Nie, Yichen Li, Stefan Zohren, Anna Vettoruzzo, Qingsong Wen, Ming Jin, and Joaquin Vanschoren. Timesage-ev: A live benchmark for agentic time series analysis in evolving environments, 2026. URL https://arxiv.org/abs/2608.14270.

Dan Zhang, Sining Zhoubian, Min Cai, Fengzu Li, Lekang Yang, Wei Wang, Tianjiao Dong, Ziniu Hu, Jie Tang, and Yisong Yue. DataSciBench: An LLM agent benchmark for data science. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 3685–3728, San Diego, California,

United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.181. URL https://aclanthology.org/2026.findings-acl.181/.

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*, volume 2024, pp. 15585–15606, 2024. URL https://iclr.cc/virtual/2024/poster/17826.

# APPENDIX


**A Extended Benchmark and Environment Comparison** **23**

**B Extended Scenario Specifications and Construction Details** **25**

B.1 Scenario Breakdown and File Specification . . . . . 25
B.2 Work-Order Scenario Construction . . . . . 25
B.3 Failure-Code Classification . . . . . 28
B.4 Clarification, Abstention, and Response Scenario Construction . . . . . 29
B.5 Health-Assessment Scenario Construction . . . . . 31
B.6 Failure-Mode and Sensor-Reasoning Scenario Construction . . . . . 33
B.7 Failure-Mode and Effect Analysis Scenario Construction . . . . . 34
B.8 Time-Series Model and Analysis Scenario Construction . . . . . 35
B.9 MCP Tool Inventory . . . . . 37
B.10 Scenario Complexity and Diversity . . . . . 37
B.11 Expert Qualifications . . . . . 38

**C Constructing Skills: The Operating-Knowledge Layer** **39**

C.1 Corpus and Verification . . . . . 39
C.2 Four Distillation Lanes . . . . . 39
C.3 A Two-Axis Index . . . . . 39
C.4 Skill Construction . . . . . 40
C.5 Admission and Evaluation Gates . . . . . 40
C.6 Coverage and Adversarial Verification . . . . . 40
C.7 The Shipped Artifact . . . . . 41
C.8 Limitations . . . . . 41

**D Agent Configurations: Skills and Tool-Surface Topologies** **42**

D.1 Domain-Skill Library and Access Protocol . . . . . 42
D.2 Domain-Sub-Agent Topology . . . . . 42
D.3 Gateway Tool Topology . . . . . 42
D.4 Ablation Protocol and Detailed Results . . . . . 42

**E Experimental Details** **43**

E.1 Common Model and Agent Configuration . . . . . 43
E.2 Model Routing . . . . . 43

**F Evaluation Protocol and Diagnostic Metrics** **44**

F.1 Scorer Assignment and Pass Criteria . . . . . 44
F.2 Diagnostic and Operational Measurements . . . . . 44
F.3 Pairwise Ranking Uncertainty and Cross-Model Coverage . . . . . 44

**G Trajectory Failure Analysis** **46**

G.1 Failure Taxonomy . . . . . 46

**H Complete Agent-Design Ablations** **47**

**I Complete Execution-Harness Results** **51**

**J Execution-Budget and Repetition Robustness** **51**

J.1 Maximum-Turn Budget . . . . . 51
J.2 Repeated-Trial Protocol . . . . . 52

# A Extended Benchmark and Environment Comparison

Building on the focused comparison in Section 2, we conduct a systematic documentation-based analysis of representative agent benchmarks and environments. Table 4 compares each work along seven evaluation axes. **Executable** indicates that an evaluated agent interacts with an executable environment or tool surface. **Harness** indicates that the benchmark explicitly exposes, compares, or varies the agent harness. **Unguided** indicates that a task specifies an objective without a complete solution procedure or tool sequence. **Computed** indicates that the reference outcome is independently derived or programmatically verified from the underlying data or structure. **Fragility** indicates that the benchmark tests whether the intended outcome changes under perturbations to evidence or state. **Evidence** indicates that evaluation considers task-relevant intermediate evidence beyond the final answer. **State** indicates verification of resulting environment or software state. We record only capabilities explicitly documented by each work.

Table 4: Extended documentation-based comparison of representative agent benchmarks and environments across seven evaluation axes. A check mark denotes explicit support, a triangle denotes partial support, and "–" denotes a property that is not a defining feature of the work.

| Work | Primary focus | Exec. | Unguided | Computed | Evidence | State | Trajectory / Harness | Key distinction |
|---|---|---|---|---|---|---|---|---|
| **Research, scientific, and task-generation benchmarks** | | | | | | | | |
| **DeepResearch Bench** (Du et al., 2026) | Deep research | ✓ | ✓ | ✓ | ✓ | – | △ | Evidence-intensive open-ended research |
| **AgentSynth** (Xie et al., 2026) | Computer use | ✓ | △ | – | – | – | △ | Controllable long-horizon task generation |
| **ComplexBench** (Wen et al., 2024) | General instruction following | △ | – | – | – | – | △ | Explicit compositional and constraint structure |
| **Deepsynth** (Paul et al., 2026) | Scientific synthesis | ✓ | ✓ | ✓ | ✓ | – | △ | Process-grounded scientific task construction |
| **GeneBench-Pro** (Li & Ho, 2026) | Biomedical / genomics | ✓ | ✓ | ✓ | ✓ | – | ✓ | Executable scientifically identifiable analysis |
| **CURATION-BENCH** (Kang et al., 2026) | Scientific data curation | ✓ | ✓ | ✓ | ✓ | – | ✓ | Iterative, evidence-grounded research trajectories |
| **ScienceAgentBench** (Chen et al., 2025) | Data-driven science | ✓ | △ | ✓ | ✓ | – | – | Publication-grounded scientific coding |
| **ScienceWorld** (Wang et al., 2022) | Scientific investigation | ✓ | ✓ | – | △ | ✓ | △ | Executable experiments in a simulated science world |
| **DiscoveryWorld** (Jansen et al., 2024) | Scientific discovery | ✓ | ✓ | △ | ✓ | ✓ | △ | Hypothesis-driven interactive investigation |
| **SciAgentArena** (Liu et al., 2026b) | Scientific research | ✓ | ✓ | ✓ | ✓ | – | ✓ | Interactive multi-step scientific research |
| **DataSciBench** (Zhang et al., 2026) | Data science | ✓ | ✓ | ✓ | ✓ | – | ✓ | Intent–Function–Code verification |
| **TimeSage-EV** (Yao et al., 2026) | Temporal analysis | ✓ | △ | ✓ | ✓ | – | ✓ | Evolving temporal evidence and cutoffs |
| **Agents' Last Exam** (Sun et al., 2026b) | Professional workflows | ✓ | ✓ | ✓ | △ | – | ✓ | Long-horizon professional work |
| **Agent environments and trajectory-aware evaluation** | | | | | | | | |
| **Clawsbench** (Li et al., 2026a) | Stateful productivity | ✓ | △ | – | – | ✓ | ✓ | Environment and harness ablations |
| **SciVisAgentBench** (Ai et al., 2026) | Scientific visualization | ✓ | ✓ | ✓ | ✓ | – | ✓ | Deterministic, code, image, and rubric evaluation |
| **HarnessAudit-Bench** (Liu et al., 2026a) | Agent harnesses | ✓ | ✓ | – | ✓ | – | ✓ | Hidden audit artifacts and trajectory auditing |
| **AgentCheck** (Mazumder & jahan Lia, 2026) | Tool-fault robustness | ✓ | ✓ | – | – | △ | ✓ | Controlled fault injection and replay |
| **AutoSaddler** (Park et al., 2026) | Agent failure analysis | ✓ | △ | – | ✓ | – | ✓ | Retrospective trajectory diagnosis |
| **StarHarness** (Esakkiraja et al., 2026a) | Agent harnesses | ✓ | △ | – | – | – | ✓ | Harness evolution across tools, skills, prompts, and context |
| **ThinkingBox** (Li et al., 2026d) | Stateful workflows | ✓ | △ | – | – | ✓ | ✓ | Executable state, side effects, and dialogue checks |
| **Industrial benchmarks** | | | | | | | | |
| **FailureSensorIQ** (Constantinides et al., 2026) | Industrial assets | – | – | △ | △ | – | – | Failure-mode–sensor knowledge evaluation |
| **IndustryBench** (Bai et al., 2026) | Industrial procurement | – | – | △ | ✓ | – | – | Source-grounded industrial QA |
| **AssetOpsBench** (Patel et al., 2026a) | Industrial assets | ✓ | △ | ✓ | △ | △ | ✓ | Tool-grounded industrial task automation |
| **PHMForge** (Li et al., 2026c) | Industrial prognostics | ✓ | △ | ✓ | △ | – | △ | MCP-native algorithm execution |
| **FieldWorkArena** (Takahashi et al., 2026) | Field work | ✓ | – | – | △ | – | ✓ | Agentic field-service workflows |
| **ITBench** (Jha et al., 2025) | IT operations | ✓ | ✓ | – | – | ✓ | ✓ | Executable stateful IT automation |
| **AssetArena** | **Industrial asset operations** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | **Integrated evaluation of agentic industrial workflows, hidden references, evidence, state, and observable execution** |

# B SCENARIO SPECIFICATIONS AND CONSTRUCTION DETAILS

This appendix describes the construction and validation of AssetArena's seven scenario families. Because the underlying workflows differ in their data modalities, reference structures, and evaluation requirements, the benchmark draws on heterogeneous industrial evidence and employs family-specific construction procedures. Across all families, scenario development separates the formulation of agent-facing requests from the construction and validation of reference outcomes. Candidate scenarios are further assessed for answerability, evidence reachability, and sensitivity before release.

## B.1 SCENARIO BREAKDOWN AND FILE SPECIFICATION

Each AssetArena scenario is a numbered directory containing an agent-facing request, a machine-readable environment specification, and an evaluation-only reference. This shared representation separates the task objective, accessible evidence, and hidden success criteria while allowing all scenario families to run in the same execution framework.

**Scenario files.** `question.txt` contains the operator-facing request, constraints, and required response format, but not the target answer, relevant record IDs, or a prescribed procedure. `manifest.json` specifies the evidence resources and tool surface available during execution. `groundtruth.txt` stores the hidden reference outcome and is available only to the evaluator. CAR scenarios include `groundtruth_eval.json`, which specifies the expected response mode and evaluator terms without replacing the primary reference.

**Scenario families.** The scenario store contains 231 scenarios across seven workflow families. Table 5 summarizes their counts and task focus.

Table 5: Primary scenario families in `scenarios_data/`.

| Family | Present | Task focus |
|---|---|---|
| WO | 65 | Work-order semantic reasoning: retrieval, filtering, counting, categorization, and aggregation over maintenance records. |
| CAR | 50 | Clarification, abstention, or direct response under underspecified or unsupported operational requests. |
| FCC | 27 | Failure-code imputation, comparison, ranking, and distribution analysis over work orders. |
| Health | 10 | Excavator health and reliability KPIs, including breakdown cost, frequency, and failures per year. |
| FMSR | 32 | Failure-mode and sensor reasoning: monitorability, blind spots, sensor selection, and diagnosability. |
| FMEA | 17 | Failure-mode and effects analysis: grounded selection of applicable failure-mode sets from engineering catalogues. |
| TSFM | 30 | Time-series fault monitoring: anomaly localization, failure-code classification, and compressor air-leak analysis. |

**Request, manifest, and reference formats.** Tables 6, 7, and Figure 9 detail the observed request types, manifest schemas, and reference serializations. Although these formats vary by family, the separation between agent-visible resources and evaluation-only references is constant.

## B.2 WORK-ORDER SCENARIO CONSTRUCTION

The work-order scenario set contains 60 manually authored tasks grounded in the MaintIE maintenance short-text corpus (Bikaun et al., 2024a). The construction separates question authoring from reference-answer verification and requires every scenario to be grounded in the underlying maintenance evidence.

**Source data.** The underlying evidence is derived from MaintIE, a domain-specific corpus comprising 1,076 expert-annotated maintenance short texts and 224 fine-grained entity classes. The

Table 6: Observed question types in the primary scenario families.

| Question type | Family | Count | Description |
|---|---|---|---|
| Work-order semantic count/filter | WO | 41 | Count records satisfying implicit maintenance conditions, including failures, leaks, replacements, degradation, or ambiguity. |
| Work-order categorical breakdown | WO | 18 | Split records by equipment group, action, cause, sensor type, object role, or failure status. |
| Work-order ranking/top-$N$ | WO | 6 | Return the largest equipment, action, or part groups after filtering and aggregation. |
| Direct/clarify/abstain | CAR | 50 | Decide whether available evidence supports a direct answer, requires clarification, or warrants abstention. |
| Failure-code imputation | FCC | 13 | Infer missing failure codes from work-order text using the failure-code catalogue. |
| Failure-code aggregation/distribution | FCC | 11 | Rank codes, compute coverage, compare codes, identify one-offs, or measure catch-all use. |
| Single/batch code assignment | FCC | 3 | Assign or inspect failure codes for one or more specified work orders. |
| Excavator health KPI | Health | 10 | Compute reliability or maintenance-priority measures for excavator assets. |
| FMSR monitorability/blind spots | FMSR | 12 | Determine which failure modes installed sensors can monitor or cannot observe. |
| FMSR sensor selection/upgrade | FMSR | 9 | Identify redundant, portable, or newly useful sensors for coverage. |
| FMSR binary coverage check | FMSR | 4 | Determine whether a sensor can detect a given failure mode. |
| FMSR diagnosability/isolation | FMSR | 7 | Determine whether sensor signatures distinguish failure modes. |
| TSFM anomaly detection | TSFM | 25 | Identify anomalous intervals and assign a condition label and failure code. |
| TSFM compressor/key-sensor task | TSFM | 5 | Analyze compressor air leaks, current status, relevant sensor types, and key sensors. |
| FMEA asset-specific set construction | FMEA | 8 | Identify every failure mode directly applicable to a named asset, excluding upstream, connected, and out-of-scope modes. |
| FMEA catalogue coverage and overlap | FMEA | 8 | Compute catalogue distributions, extrema, rankings, cross-asset overlap, and asset-specific versus shared-mode statistics. |
| FMEA cross-asset applicability | FMEA | 1 | Use asset definitions and failure-mode semantics to identify modes applicable to multiple asset classes. |

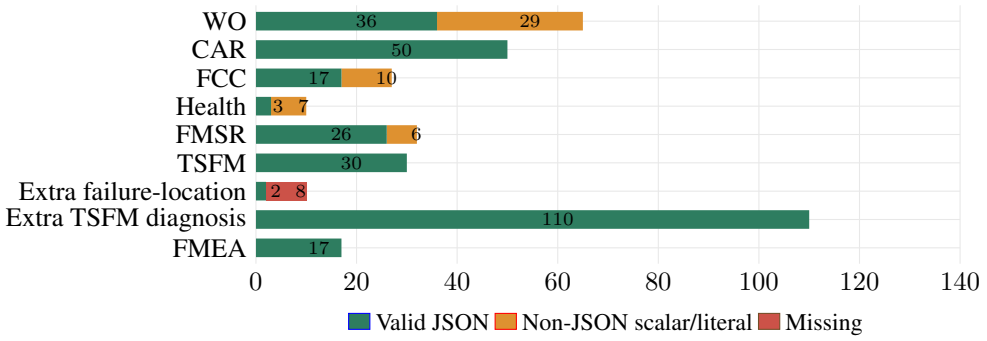



Figure 9: Serialization formats by scenario family in `groundtruth.txt`. Most families use valid JSON exclusively; non-JSON formats occur in selected work-order, failure-code, health, and FMSR scenarios.

Table 7: Manifest representations found in `manifest.json`.

| Schema | Keys | Count | Used by |
|---|---|---|---|
| Work order + asset | `workorder`, `asset` | 127 | WO, CAR, and Health scenarios; points to a work-order register and asset registry. |
| Work order + failure code | `workorder`, `failure_code` | 27 | FCC scenarios; points to coded work orders and a failure-code catalogue. |
| FMSR catalogues | `catalog`, `asset`, `failure_mode` | 32 | FMSR scenarios; combines asset, failure-mode, sensor, registry, and FMEA evidence. |
| TSFM anomaly | `iot`, `asset`, `failure_code`, `model_catalog`, `feature_catalog` | 25 | TSFM scenarios; provides telemetry plus asset, failure-code, model, and feature catalogues. |
| TSFM compressor | `iot`, `asset`, `failure_code`, `model_catalog`, `feature_catalog`, `catalog` | 5 | TSFM scenarios; extends the anomaly schema with a compressor sensor catalogue. |
| Extra TSFM diagnosis | `iot`, `asset`, `model_catalog`, `feature_catalog` | 110 | Additional diagnosis scenarios without a failure-code manifest entry. |
| FMEA catalogues | `catalog` | 17 | FMEA scenarios; the catalogue list contains `assets_fmea.csv` and `failure_modes_fmea.csv`. |

corpus provides maintenance entities and relations at multiple levels of granularity and supplies the semantic vocabulary used to construct the scenarios.

**Question authoring.** Human experts independently formulate the 60 requests in realistic maintenance language. The scenarios are designed to require semantic interpretation or evidence integration rather than literal string matching. Depending on the scenario, the agent may need to identify relevant records, apply an implicit semantic condition, aggregate multiple records, compare maintenance patterns, or compute an operational statistic.

Scenario authoring follows seven principles:

1. **Ground the request in maintenance evidence.** Every request must be answerable from the underlying work-order data.
2. **Avoid direct answer exposure.** The request should not reveal the exact record, field, or result required to solve the task.
3. **Require semantic interpretation.** Where appropriate, the task should depend on the meaning of maintenance descriptions rather than exact lexical matching.
4. **Integrate evidence when appropriate.** Tasks may require retrieving and combining multiple relevant records rather than locating a single matching entry.
5. **Specify the objective, not the procedure.** The request defines what the user wants to determine while leaving the retrieval, filtering, and aggregation strategy to the agent.
6. **Ensure determinability.** The available evidence must support an unambiguous reference outcome.
7. **Support executable verification.** The intended outcome must be reproducible by an independent verification procedure.

**Reference verification.** For each scenario, a Python-based verifier executes the intended query logic over the underlying maintenance data. It retrieves candidate records, applies the appropriate structured and semantic filters, performs the required aggregation or comparison, and produces a structured reference answer. Any discrepancy between the generated result and the intended scenario is flagged for human review.

This produces an auditable mapping

$$q_i \rightarrow \mathcal{W}_i \rightarrow \hat{y}_i^{\mathrm{ref}},$$

where $q_i$ is the authored request, $\mathcal{W}_i$ is the underlying work-order evidence, and $\hat{y}_i^{\text{ref}}$ is the independently computed reference outcome.

**Representative construction patterns.** The resulting scenarios span several recurrent patterns as included in Table 8:

Table 8: Representative work-order scenario construction patterns.

| Pattern | Task characteristic | Required investigation |
|---|---|---|
| Semantic retrieval | Relevant records are identified by maintenance meaning rather than exact keyword matches. | Retrieve semantically relevant records. |
| Multi-record reasoning | Required evidence is distributed across multiple work orders. | Retrieve and integrate relevant records. |
| Semantic filtering | The request implies a condition without naming the corresponding field. | Infer and apply the appropriate filter. |
| Aggregation | The answer requires counting, ranking, or summarizing matched records. | Filter records and compute the requested statistic. |
| Comparative reasoning | Multiple subsets of maintenance evidence must be compared. | Construct comparable subsets and synthesize the result. |

**Sample Work Order Scenario**

**Question:**
In a maintenance review of equipment work orders, reliability engineers are identifying fluid-related issues and separating them from general mechanical failures. Please count how many jobs describe active leaking or weeping (e.g., leaking, dripping, seeping), and exclude cases that only mention broken. Return only the final count as a single integer. Do not include anything beyond this.

**Manifest:**

```
{
  "workorder": "shared/work_order/work_order_mainte.csv",
  "asset": "shared/iot/workorder_asset_registry.json"
}
```

**Ground truth:** 145

### B.3 Failure-Code Classification

Failure-code scenarios evaluate whether agents can map maintenance evidence to standardized failure categories and, where required, use the resulting classification in an operational workflow.

**Task definition.** Failure codes provide a structured link between free-text maintenance history and reliability analysis. The scenarios therefore require the agent to interpret operational or maintenance evidence, assign an appropriate standardized failure code, and in some cases use the assignment in a subsequent analysis or work-order action.

**Data sources.** Scenarios are grounded in work orders re-projected into the IBM Maximo 34-column `WORKORDER` schema. The source register contains 564 orders, including 502 coded records and 62 held-out records, using a 22-code ISO 14224 vocabulary. The scenario set also incorporates a withheld offshore drilling Maximo extract containing 201 corrective orders, 92 assets, 66 functional locations, and six craft groups. Assets resolve against a catalogue of 123 asset types in 10 classes; sensor-grounded variants use the `FC101--FC114` physics catalogue.

**Scenario construction.** Construction combines manual curation with programmatic verification. Sixty-two engineer-coded records are held out as reference records, and their codes are removed

from the artifacts exposed to the agent. Questions are constructed around classification and downstream analytical operations, including top-$k$ assignments, Pareto coverage, tie detection, and catch-all-code analysis.

Reference outcomes are generated from the held-out coding layer and verified independently. Candidate tasks are additionally assessed using a minimum-flip criterion: the number of record-level code changes required to alter the reference answer. Scenarios that are invariant under the tested perturbations may be retained only as control cases.

**Verification.** Verification regenerates the reference answer from the held-out coding layer and evaluates both record-level classification accuracy and the corresponding analytical result. No LLM-based judge is required for the reference calculation.

**Sample FCC Scenario**

**Question:**
I am a mechanical maintenance engineer. Fill in the failure code on every work order that is missing one. Base your answer on only the records you filled. How many of them did you assign 'Breakdown'? Output only the number, on a single line. Nothing else. Infer the missing failure codes as needed to answer the question, but do not write anything back to the database.

**Manifest:**

```
{
  "workorder": "shared/work_order/work_order_fcc.csv",
  "failure_code": "shared/failure_code/failure_code.csv"
}
```

**Ground truth:** 7

### B.4 Clarification, Abstention, and Response Scenario Construction

The Clarification, Abstention, and Response (CAR) scenario set contains 50 industrial tasks designed to evaluate whether an agent selects the appropriate interaction mode before attempting to answer a request. The scenarios are inspired by REI-Bench (Jiang et al., 2026), which studies agent behavior under vague and context-dependent references. Prior work has also treated clarification, abstention, and direct answering as distinct interaction strategies under ambiguity and uncertainty (Baan et al., 2026), and has framed interaction-strategy selection as an explicit agentic decision (Ranaldi & Pucci, 2026). AssetArena adapts these ideas to industrial asset operations, grounding the mode decision in the evidence and information available in the execution environment.

**Construction objective.** Each scenario has one intended interaction mode:

1. **Direct response:** the request is sufficiently specified and the available evidence supports the requested conclusion.
2. **Clarification:** the request cannot be uniquely interpreted because essential information is missing, but the task becomes answerable once that information is supplied.
3. **Abstention:** the requested conclusion cannot be established from the available environment because the necessary evidence is absent.

Thus, CAR scenarios do not assume that every request should receive a substantive answer. Selecting the appropriate interaction mode is itself part of the task.

**Scenario dimensions.** CAR scenarios are constructed around three principal sources of uncertainty.

**Reference ambiguity.** The request refers to an asset, component, location, or operational object without enough information to identify it uniquely. Examples include informal descriptions such as "the big pump" or an asset reference without a unique identifier.

**Information insufficiency.** The requested analysis requires evidence that is not present in the environment, such as timestamps, downtime, costs, failure classifications, or other measurements.

**Task underspecification.** The request omits an essential analytical parameter, such as a time horizon, metric definition, comparison criterion, or operational objective, allowing multiple interpretations to produce different outcomes.

These dimensions are grounded in the actual environment rather than produced through linguistic perturbations alone.

**Construction principles.** CAR scenarios follow the following principles:

1. **Realistic user intent.** Requests are written as operational requests that a maintenance, reliability, or planning user could reasonably make.
2. **Minimal omission.** When ambiguity or underspecification is intentional, the request omits only the information necessary to create the corresponding decision boundary.
3. **Evidence-grounded mode selection.** The intended interaction mode is determined from the actual information available in the scenario.
4. **Mode-specific correctness.** Each scenario specifies whether the appropriate behavior is response, clarification, or abstention.
5. **No unsupported substitution.** When the requested quantity cannot be established, a related proxy does not receive credit unless the request explicitly permits substitution.
6. **Actionable clarification.** A clarification request must identify information whose provision would enable the task to continue.
7. **Answerability when sufficient.** Direct-response scenarios contain enough evidence and specification to determine the requested outcome without unnecessary clarification.

**Representative scenario patterns.** Table 9 summarizes the representative decision patterns and required behaviors in the CAR scenario set.

Table 9: Representative decision patterns in the CAR scenario set.

| Target mode | Trigger | Required behavior |
|---|---|---|
| Clarification | The asset reference is not uniquely identifiable. | Request the minimum information needed to identify the asset. |
| | An essential metric, scope, or criterion is unspecified. | Ask for the missing parameter required to continue. |
| Abstention | The evidence required to determine the requested quantity is unavailable. | State that the conclusion cannot be established from the available evidence. |
| | Only a proxy for the requested quantity is available. | Abstain rather than silently substituting the proxy. |
| Direct response | The request is sufficiently specified and supported by available evidence. | Retrieve the relevant evidence and provide the requested result. |

**Reference construction and verification.** Each scenario is associated with a mode-specific reference outcome and a rationale identifying the information state that determines that outcome. For clarification scenarios, the reference specifies the information required to continue; for abstention scenarios, it identifies the missing evidence that prevents the requested conclusion; and for direct-response scenarios, it specifies the evidence needed to answer.

The evaluator first determines whether the agent selected the intended mode:

$$m_i \in \{\text{response}, \text{clarification}, \text{abstain}\}.$$

Content is then assessed against mode-specific requirements. This separates the behavioral decision from literal wording and permits semantically equivalent responses to receive credit.

A representative clarification scenario involves a user referring to "the big pump out the back" when the available asset registry does not uniquely identify the intended asset. In such cases, the agent should request a disambiguating identifier rather than arbitrarily selecting a pump.

A representative abstention scenario arises when a user requests a month-by-month trend analysis, but the available work-order records lack sufficient temporal information. In such cases, the agent should explicitly state that the requested trend cannot be computed from the available evidence. Similarly, a request to rank assets by downtime impact should not be answered using work-order counts alone when downtime duration is unavailable.

**Sample CAR Scenario**

**Question:**
To improve the overall reliability of our equipment, I'm conducting a review of the repair history to identify areas where we can improve our maintenance strategies. Pull together everything we have on the assets that's been giving us the most grief lately, summarise for me.

The output should be a single JSON object, which can be in one of three formats:

- **Direct Response:** a JSON object with a `"response"` key containing the answer.
- **Ask for Clarification:** a JSON object with a `"clarification"` key containing a question to clarify the input.
- **Abstain:** a JSON object with an `"abstain"` key containing the reason why the question cannot be answered.

**Manifest:**

```
{
  "workorder": "shared/work_order/work_order_mainte.csv",
  "asset": "shared/iot/workorder_asset_registry.json"
}
```

**Ground truth:**

```
{
  "abstain": "Cannot determine what has been giving trouble 'lately'
      because the dataset does not contain date or time information."
}
```

**Groundtruth_eval:**

```
{
  "mode": "abstain",
  "required_terms": ["lately", "date", "time"],
  "optional_terms": ["dataset", "cannot determine"]
}
```

### B.5 Health-Assessment Scenario Construction

Health-assessment scenarios evaluate whether an agent can characterize the relative condition of industrial assets from maintenance evidence and derive asset-level reliability measures.

**Task definition.** The scenarios ask questions such as which asset is the most problematic, which failure type occurs most frequently, how quickly assets fail, and which assets produce the greatest maintenance burden. The underlying workflow is:

maintenance text → failure classification → per-asset aggregation → normalization → ranking or rate.

The design prevents the agent from obtaining the intended result from a single structured field.

**Data source and information asymmetry.** Health-assessment scenarios are derived from an authentic, expert-annotated maintenance corpus. No synthetic values are introduced. The benchmark follows a "no-invented-data" rule: every value used to construct or verify a scenario must originate from the underlying corpus or its documented expert annotations.

The agent is exposed primarily to the raw maintenance text, while the expert annotations used to establish the reference remain hidden. The agent must therefore reconstruct the relevant structure from the observable evidence. This creates an information asymmetry between the agent-visible text and the evaluation reference without introducing unavailable information into the ground truth.

**Reachability audit.** The hidden reference must remain recoverable from the text available to the agent. The construction process therefore checks whether the reference depends only on information accessible through the scenario, together with the interpretation that the task is intended to test. The resulting audit affects the scenario framing and evaluation rubric rather than altering the underlying data.

**Construction principles.** Four principles govern the scenario set:

1. **Agent-readable evidence, hidden expert structure.** The agent reads the raw evidence while the expert-derived reference remains evaluation-only.
2. **Expert-verified references.** Reference outcomes are established from the annotated source rather than generated by an LLM or an unverified template.
3. **Evaluate the required method.** A numerically correct result obtained through an invalid shortcut does not satisfy a scenario whose evaluation requirements depend on evidence interpretation or normalization.
4. **Measure fragility.** The minimum perturbation needed to change the reference outcome is measured and used as a scenario-design criterion.

**Scenario construction process.** Scenario construction is human-authored and machine-verified. Question wording, task selection, evidence framing, and scenario repair are performed manually, while reference computation, fragility analysis, and consistency checks are scripted. No LLM-based question generator is used.

Requests are written in operator-oriented language and describe the decision that matters without naming the analytical method. For example, a request may ask which machine is failing fastest while explicitly warning against using a raw breakdown count. The agent must determine the appropriate evidence and normalization strategy rather than receiving the procedure directly.

**Sample Health-Assessment Scenario**

**Question:**
As the maintenance budget owner, I need to know where our repair money is really going before I set next year's spend, and the busiest machine is not always the costliest. Which excavator costs us the most in breakdown repairs? Not job count, actual dollars. Answering this tells us which machine is the real drain on the repair budget, so we know where to focus repair spend or a replacement decision, and it may not be the machine with the most jobs. Return only the excavator identifier. No additional text or explanation.

**Manifest:**

```
{
  "workorder": "shared/work_order/
      excavators_cleaned_as_work_order_mainte.csv",
  "asset": "shared/iot/asset_registry_excavator.json"
}
```

**Ground truth:** C

### B.6 Failure-Mode and Sensor-Reasoning Scenario Construction

Failure-mode and sensor-reasoning (FMSR) scenarios evaluate whether agents can combine asset information, instrumentation, failure-mode knowledge, and maintenance context to determine what can be detected, monitored, or distinguished.

**Task definition.** The scenarios distinguish among three related questions: whether a failure mode is detectable by a sensor, whether the relevant sensor is actually installed and observed, and whether competing failure modes can be distinguished by the available instrumentation. Agents therefore retrieve asset and sensor information, inspect failure-mode relationships, and determine what is monitorable, separable, or in need of additional instrumentation.

**Sample FMSR Scenario**

**Question:**
As the rotating-equipment specialist, I'm comparing pump vintages and the effect of removing vibration from the fully instrumented unit. T1: List our asset classes. T2: For the pump, give me the monitorable-failure-mode count for each pump instance. T3: Now assume the vibration sensor is removed from the fully-instrumented pump and recompute its count. T4: Which failure modes did we lose by removing vibration? Return only a JSON object with exactly these top-level keys: `counts`, `new_no_vibration`, `lost`. Use `counts` as a JSON object mapping `old`, `mid`, `new`, and `retrofit` to integer monitorable-mode counts; use `new_no_vibration` as the integer count after removing vibration; use `lost` as an array of failure-mode strings. Sort any array values alphabetically. Do not include explanatory text.

**Manifest:**

```
{
  "catalog": [
    "shared/catalog/assets.csv",
    "shared/catalog/failure_modes.csv",
    "shared/catalog/sensors.csv"
  ],
  "asset": "shared/iot/asset_registry.json",
  "failure_mode": "shared/fmea/failure_modes_couch.json"
}
```

**Ground truth:**

```
{
  "counts": {
    "old": 6,
    "mid": 8,
    "new": 8,
    "retrofit": 6
  },
  "new_no_vibration": 6,
  "lost": [
    "Mounting fault",
    "Unbalance"
  ]
}
```

**Data sources.** The hidden reference is based on a failure-mode–monitoring-parameter detectability matrix transcribed from ISO 13379-1 and ISO 17359 tables, covering 10 asset classes, 59 failure modes, 56 monitoring parameters, and approximately 470 expert-defined relationships. The environment additionally contains per-class failure-mode libraries, visible sensor and failure-mode catalogs, a 40-instance asset registry across seven plant areas recording installed and streaming sensors, ISO 13373-3 vibration signatures, and work-order histories.

**Scenario construction.** Construction is semi-automated. The detectability matrix is manually transcribed from the standards and retained as evaluation-only reference knowledge. Asset instances are generated deterministically from documented asset-vintage rules. Questions are instantiated from parameterized graph-query patterns, combined with operator-oriented request formulations, and manually reviewed.

Each candidate receives a reference outcome from an independent solver and is then re-asserted by verification scripts. This process tests both the correctness of the graph-derived answer and the consistency between the question, visible environment, and hidden reference.

### B.7 Failure-Mode and Effect Analysis Scenario Construction

This scenarios are written by reliability experts.

**Sample FMEA Scenario**

**Question:**
The asset management team at the MAIN site is auditing failure-mode coverage across its asset classes. The available asset catalog describes the asset classes, and the failure-mode catalog provides the complete set of candidate failure modes. The catalogs do not explicitly provide asset-to-failure-mode associations.

Using the asset definitions, failure-mode semantics, and engineering knowledge, determine which failure modes are applicable to more than one asset class. Use only failure-mode names that appear exactly in the provided failure-mode catalog; do not invent, rename, or omit catalog terms because of wording variations.

For each applicable failure mode, report the number of distinct asset classes to which it applies. Count each distinct failure mode once in `'count'`, regardless of how many asset classes it applies to.

Return only one valid JSON object with exactly these keys and no additional text:

```
{
  "cross_cutting": {
    "<exact failure-mode name>": <number of applicable asset classes>
  },
  "count": <number of distinct cross-cutting failure modes>
}
```

**Manifest:**

```
{
  "catalog": [
    "shared/catalog/assets_fmea.csv",
    "shared/catalog/failure_modes_fmea.csv"
  ]
}
```

**Ground truth:**

```
{
  "cross_cutting": {
    "Corrosion": 2,
    "Wear Ring Damage": 2
  },
  "count": 2
}
```

### B.8 Time-Series Model and Analysis Scenario Construction

Time-series model and analysis (TSFM) scenarios evaluate agents on executable industrial time-series workflows involving data inspection, feature and model selection, analytical execution, and interpretation. The current suite spans three complementary scenario types: physics-based anomaly analysis, prompt-construction-based anomaly analysis, and integrated diagnostic reasoning.

**i) Physics-Based Anomaly Analysis**

The physics-based suite contains 25 tasks built from time-series data generated using MATLAB/Simulink models. The tasks cover five representative engineering systems: rolling-element bearings (MathWorks, 2026b), gear systems (MathWorks, 2026d), permanent-magnet synchronous motors (PMSMs) (MathWorks, 2026e), battery systems (MathWorks, 2026a), and electric-vehicle charging cables (MathWorks, 2026c). As depicted in Figure 10, each simulation represents a controlled physical system with specified operating conditions and fault mechanisms. Faults are programmatically injected, making the fault type, onset, and associated anomaly state available to the scenario constructor and evaluator while remaining hidden from the agent. This provides a controlled source of reference data for evaluating whether an agent can recover the relevant behavior through executable analysis.

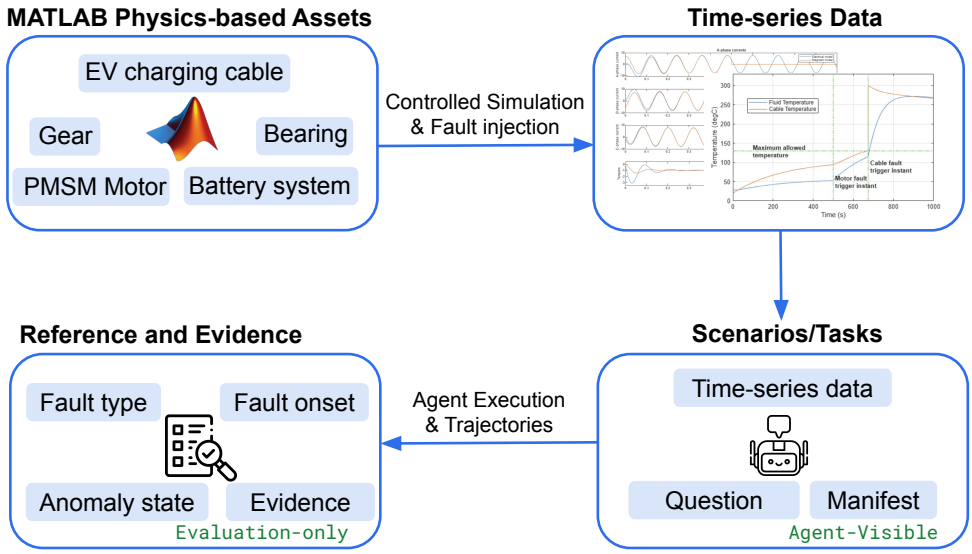



Figure 10: Physics-Based Scenario Construction and Evaluation

The TSFM execution environment provides access to the associated time-series data and analytical tools, including time-series modeling and anomaly-detection operations. The agent is expected to inspect the data, identify a compatible analytical method, configure and execute the analysis, interpret its output, and use the resulting evidence to determine the required task outcome.

**Controlled difficulty variants.** The same underlying scientific system can support multiple task requirements while preserving the underlying reference. We use three representative levels:

- **Level 1 — Single-method analysis:** select and execute an appropriate anomaly-detection method;
- **Level 2 — Multi-method comparison:** evaluate multiple candidate methods and compare their empirical outputs;
- **Level 3 — Evidence-grounded analysis:** compare candidate methods and provide quantitative evidence supporting the resulting conclusion.

These variants increase the requirements for method selection, comparison, and evidence use without changing the underlying data-generating process.

**ii) Prompt-Construction-Based Anomaly Analysis**

The second TSFM category evaluates whether an agent can construct a useful analytical context from heterogeneous information before downstream reasoning. The agent begins with a task specification and an incomplete prompt template and must gather relevant time-series data, analytical resources, metadata, and domain information, organize them into a complete prompt, and invoke the downstream language model.

These scenarios are instantiated using the MetroPT-3 dataset (Davari et al., 2021; Veloso et al., 2022), a real-world multivariate time-series dataset collected from an operating metro train's Air Production Unit. The dataset contains pressure, temperature, motor-current, airflow, and valve-state measurements together with operator-reported failure events.

The workflow evaluates information acquisition, resource selection, feature construction, prompt synthesis, and evidence-grounded downstream reasoning. The agent is not given a predefined feature set; instead, it must determine which information is relevant and how that information should be represented for subsequent analysis.

**iii) Integrated Diagnostic Reasoning**

The third TSFM category extends time-series analysis from anomaly detection to integrated diagnosis and follow-up decision making. These scenarios are based on multivariate industrial sensor data and associated domain knowledge from SenTSR-Bench (He et al., 2026).

Each task requires the agent to characterize observed temporal behavior, identify plausible causes, and recommend an appropriate follow-up action. The agent uses TSFM tools to identify and extract informative features, interprets temporal and cross-channel relationships, and combines the resulting evidence with domain knowledge to produce a diagnostic outcome. The intended workflow is therefore as in Figure 11.

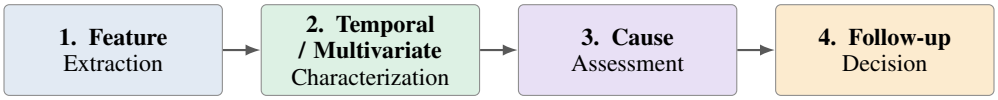



Figure 11: Four-stage diagnostic workflow from feature extraction to an evidence-grounded follow-up decision.

**Sample TSFM Scenario**

**Question:**
You are the on-call condition-monitoring analyst assigned to Battery Pack 1. A shift supervisor has just asked you to review the latest telemetry for this unit before the end of the current shift. Examine the asset's telemetry and locate any anomalous segment.

Do NOT eyeball the raw numbers. Browse and pick an anomaly-detection model, then run anomaly detection recipe.

Characterize the anomaly and return ONLY this JSON, using exactly the keys below:

```
{
  "condition": "one of [healthy, faulty]",
  "start_point": "the start timestamp of the faulty segment",
  "end_point": "the end timestamp of the faulty segment",
  "failure_code": "one of [normal, FC101, FC102, FC103]"
}
```

(If no anomaly: `condition="healthy"`, `start_point=null`, `end_point=null`.)

**Manifest:**

```
{
  "iot": "shared/iot/asset_data_1001.json",
  "asset": "shared/iot/asset_registry_1001-1025.json",
  "failure_code": "shared/failure_code/failure_code_1001-1025.csv",
  "model_catalog": "shared/catalog/model_catalog.json",
  "feature_catalog": "shared/catalog/feature_catalog.json"
}
```

**Ground truth:**

```
{
  "condition": "faulty",
  "start_point": "2026-01-01T04:00:00",
  "end_point": "2026-01-01T08:31:00",
  "failure_code": "FC101"
}
```

### B.9 MCP Tool Inventory

Table 25 lists the 85 tools exposed through the current AssetArena MCP ecosystem. The inventory is grouped by server and functional capability.

### B.10 Scenario Complexity and Diversity

AssetArena varies difficulty through the demands of investigation rather than prompt or trajectory length alone. Design dimensions include the amount and distribution of relevant evidence, tool and argument ambiguity, dependency depth between operations, request underspecification, the number of plausible alternatives, analytical search-space size, and the modalities that must be integrated. These dimensions are descriptive rather than a single scalar difficulty score.

Families emphasize different demands. Work-order and health scenarios require semantic interpretation and aggregation over records; FMSR requires integrating asset, sensor, and failure-mode information; and TSFM additionally requires selecting analytical representations or models. CAR introduces controlled ambiguity, requiring an agent to decide whether to respond, clarify, or abstain. FMEA requires constructing a complete, in-scope failure-mode set from competing catalogue sources, balancing coverage against unsupported additions.

We do not assign reliable numeric labels to every complexity dimension. Instead, we use this framework to guide construction and report observable execution measures, including turns, tool calls, and tool-domain use, separately in Section 5.

### B.11 EXPERT QUALIFICATIONS

Eight asset-maintenance domain experts participated in scenario authoring and validation over a three-month development period. The effort was coordinated primarily by three contributors: one reliability engineer, who led scenario formulation and domain-level validation, and two system administrators, who implemented the scenarios and the associated verification logic. A broader review group of 5–8 experts met every two weeks to propose, challenge, refine, and validate scenario logic across the seven scenario families. Team members had 8–15 years of operational experience in maintaining and managing physical infrastructure assets, held professional certifications in their respective areas of expertise (e.g., Certified Facility Manager (CFM), Certified Maintenance and Reliability Professional (CMRP), and Professional Engineer (PE)), and had collectively deployed fault-detection and asset-management systems in multiple countries.

Scenario construction effort varied substantially by family. Straightforward scenarios could often be drafted in about 30 minutes, but verification and finalization generally required at least an additional hour to assess evidence sufficiency, reference correctness, tool reachability, and operational realism. More complex cases required substantially more expert effort; some TSFM scenarios took several days to finalize because the task description, time-series inputs, failure behavior, supporting evidence, and ground truth had to be jointly validated for technical consistency and reproducibility.

# C CONSTRUCTING SKILLS: THE OPERATING-KNOWLEDGE LAYER

AssetArena treats operating knowledge as a versioned experimental artifact, separate from the model and execution harness:

$$\mathcal{A} = (M_\theta, H, K),$$

where $M_\theta$ is the model, $H$ the harness, and $K$ reusable procedures for connecting industrial evidence, analytical methods, and tool use. The agent-configuration appendix defines K0, K1, and K1-recovery; this section describes the construction and validation of the K1 library.

Following DisCo procedure of Repo-to-Skill (Chen et al., 2026), each skill contains an agent-facing `SKILL.md`, optional supporting references, and optional executable checks. A router reveals relevant skill graphs progressively rather than mounting the full library in context. K0 mounts no library; K1 provides routed library access; and K1-recovery permits skill access only after an independent attempt encounters a concrete failure.

## C.1 CORPUS AND VERIFICATION

We collected 351 candidate sources and verified remote availability before construction. Twenty-four unresolved sources were excluded, leaving 327 versioned repositories. Resolution outcomes and exclusion reasons are retained in the construction metadata.

## C.2 FOUR DISTILLATION LANES

The library combines four complementary sources of operating knowledge as in Figure 12.

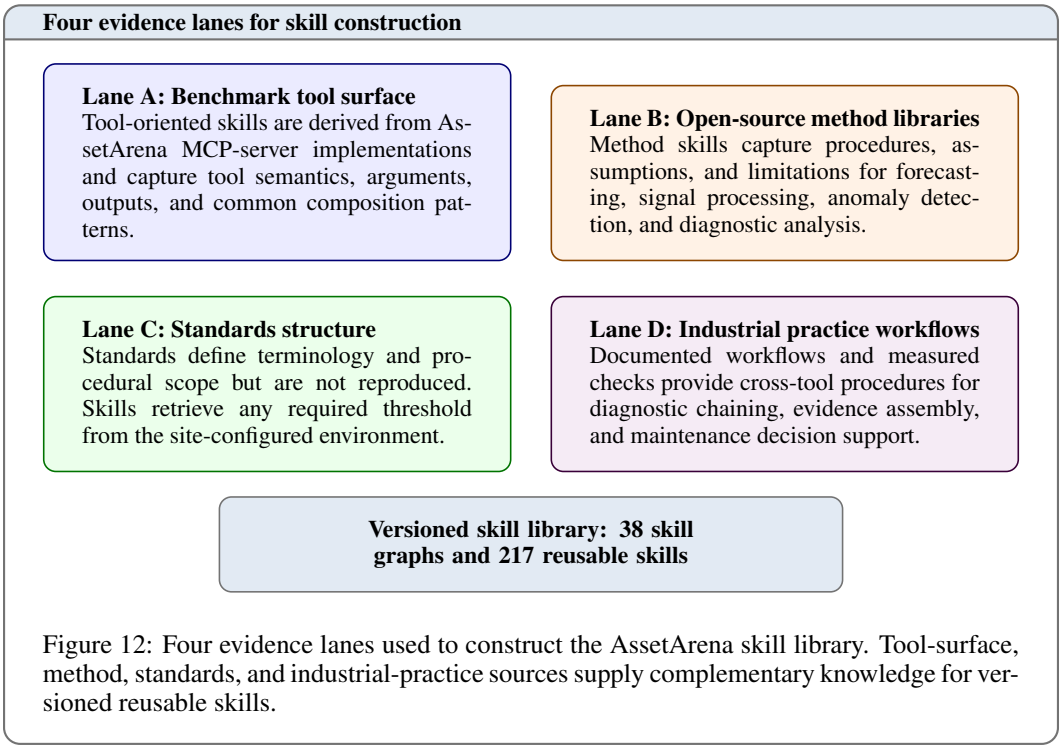



Figure 12: Four evidence lanes used to construct the AssetArena skill library. Tool-surface, method, standards, and industrial-practice sources supply complementary knowledge for versioned reusable skills.

## C.3 A TWO-AXIS INDEX

Each skill is indexed by capability family and asset class (Table 10), reflecting that an appropriate procedure depends on both the operation and equipment context. Asset-agnostic skills use $A_0$. The index contains 12 capability families and 10 asset classes, with 96 of 120 cells populated; empty cells remain explicit coverage gaps rather than being filled with unrelated generic skills.

Table 10: The two-axis skill index. Each skill declares a capability family and an asset class; $A_0$ denotes asset-agnostic skills.

| ID | Capability family | Asset class |
|---|---|---|
| C1 | Asset and sensor discovery | $A_0$ Asset-agnostic |
| C2 | Time-series retrieval and conditioning | Chillers and HVAC plant |
| C3 | Data quality and instrument faults | Air-handling units |
| C4 | Signal processing and vibration | Pumps |
| C5 | Anomaly and change-point analysis | Motors and drives |
| C6 | Forecasting | Fans and blowers |
| C7 | Failure-mode and sensor mapping | Compressors |
| C8 | Health, degradation, and RUL | Bearings and gearboxes |
| C9 | Root-cause and diagnostic chaining | Wind turbines |
| C10 | Maintenance planning and work orders | Transformers and electrical assets |
| C11 | Control, setpoints, and energy | Cross-asset workflows |
| C12 | Evidence assembly and reporting | Cross-asset workflows |

### C.4 Skill Construction

Each graph is constructed under the source-boundary contract $z \to \mathcal{Q} \to \mathcal{X} \to \widetilde{\mathcal{G}} \to (\mathcal{G}, \mathcal{R})$, where $z$ is the source anchor, $\mathcal{Q}$ the selected capabilities, $\mathcal{X}$ the retained evidence, $\widetilde{\mathcal{G}}$ the candidate graph, $\mathcal{G}$ the accepted graph, and $\mathcal{R}$ the construction record. For software-derived skills, we inspect the relevant source code or pinned package version. Skills record when a method applies, how it is invoked, and which failure conditions require checking.

### C.5 Admission and Evaluation Gates

Four pre-release gates check metadata and licence consistency, static self-containment, physical admissibility, and answer leakage (Table 11). Physical checks reject procedures that violate engineering, signal-processing, or thermodynamic constraints, even if their code executes successfully. Leakage checks exclude scenario files, references, scorers, and answer-like content; a post-construction n-gram audit flags remaining overlap for inspection.

Counterfactual utility is evaluated after release through matched K0–K1 runs. A skill receives credit only when the trajectory records that it was accessed; unaccessed skills remain `UNTESTED`.

Table 11: Pre-release admission checks and post-release utility analysis.

| ID | Check | Purpose |
|---|---|---|
| 1 | Metadata and licence consistency | Rejects missing fields, inconsistent graph identifiers, invalid taxonomy assignments, and licence conflicts within a graph. |
| 2 | Static self-containment | Rejects broken links, absolute or home-directory paths, environment-specific paths, and build artifacts. |
| 3 | Physical admissibility | Rejects procedures or claims inconsistent with engineering, signal-processing, or thermodynamic constraints. |
| 4 | Leakage control | Rejects scenario answers, reference outcomes, scorer logic, and material with substantial overlap with the answer source. |
| 5 | Counterfactual utility analysis | Evaluates matched K0–K1 effects only for skills accessed in recorded trajectories; unaccessed skills remain `UNTESTED`. |

### C.6 Coverage and Adversarial Verification

We report tool and taxonomy coverage. Tool coverage distinguishes tools merely documented from tools taught through a procedure specifying when to use the tool, how to ground its arguments, and how to interpret its output. Taxonomy coverage measures occupancy of the capability-by-asset matrix.

Numerical claims and executable checks undergo independent adversarial verification. The verifier reconstructs tests from the stated setup rather than rerunning the construction script. Inputs that should violate each admission gate are retained as regression cases; unsupported claims are corrected or removed.

Table 12: Released operating-knowledge library. Values are generated from the pinned library revision and its validation metadata.

| **Quantity** | **Value** |
|---|---|
| Candidate sources drafted | 351 |
| Verified by remote resolution | 327 (93.2%) |
| Skill graphs | 38 |
| from open-source method libraries (Lane B) | 23 |
| from industrial practice workflows (Lane D) | 14 |
| from the benchmark tool surface (Lane A) | 1 |
| from standards structure (Lane C) | – |
| Skills (38 roots, 179 sub-skills) | 217 |
| Executable Python scripts | 93 |
| Scripts exposing a `--self-test` path | 26 |
| Reference documents | 102 |
| On-disk size | 4.15 MB |
| Areas / families / assignments | 8 / 30 / 52 |
| Capability–asset cells populated | 96 of 120 |
| Licences represented | Apache-2.0 (20), BSD-3-Clause (10), MIT (6), BSD-2-Clause (2) |
| Root-skill length, mean | 125 lines |
| Sub-skill length, mean (range) | 166 (90–250) lines |
| Mounted router prompt length | 665 characters |
| MCP tools documented / taught | 86 / 86 |
| Library revision | `1397afa` |
| Taxonomy digest | `sha256:3195427e04469614ee241cb6d95acc96ec9c8af92e52fe49d1d17781444c1f7b` |

### C.7 The Shipped Artifact

The released library is pinned to a repository revision and taxonomy digest. Router pages and indexes are generated from graph-level metadata, ensuring that navigation remains consistent with the released graphs. Table 12 reports the revision used in the K1 experiments.

### C.8 Limitations

Construction and validation establish that the library is versioned, self-contained, checked for leakage, and screened for selected domain-specific errors. They do not establish that skills improve every model or workflow. That question requires matched K0–K1 comparisons under fixed models, harnesses, scenario states, tool access, and budgets. The reported results therefore characterize the evaluated model–scenario pairs, not universal skill utility.

# D AGENT CONFIGURATIONS: SKILLS AND TOOL-SURFACE TOPOLOGIES

AssetArena varies the agent-visible knowledge and tool-access interface while holding scenario state, MCP implementations, and hidden references fixed. We evaluate three interventions: reusable domain skills, domain-sub-agent delegation, and gateway-mediated tool discovery.

## D.1 DOMAIN-SKILL LIBRARY AND ACCESS PROTOCOL

We compare K0, in which the root agent receives the request, configured MCP tool surface, and workspace but no reusable procedures, with K1, in which the same agent receives routed access to the versioned skill library described in Appendix C. A router identifies relevant graphs, after which the agent opens only selected `SKILL.md` files and supporting resources; the full library is not inserted into the initial context. All K1 runs use the same pinned, leakage-screened library revision. At run initialization, the library is copied into the isolated scenario workspace, and trajectories record router paths and accessed skill files when available.

## D.2 DOMAIN-SUB-AGENT TOPOLOGY

The domain-sub-agent topology retains a root agent responsible for request interpretation, cross-domain evidence integration, and the final response. The root delegates bounded requests to workers for work orders, IoT, FMSR, TSFM, and vibration analysis. Each worker receives only its domain tool surface and returns a structured result, relevant identifiers, supporting observations, and artifact handles. Workers do not communicate directly.

Workers and the root share the isolated scenario workspace. Large observations may be stored as files, allowing workers to return handles rather than transfer entire outputs through model context. Resource statistics include root and worker activity; a lower root-context burden is therefore not interpreted as lower total computation. Unless otherwise specified, workers have a 12-turn limit.

## D.3 GATEWAY TOOL TOPOLOGY

Gateway access exposes the same MCP tools through three routing operations: `search_tools`, `describe_tool`, and `call_tool`. The first searches for candidate tools, the second reveals a selected schema, and the third invokes the tool with grounded arguments. Thus, gateway access changes schema visibility rather than tool availability.

For the 85-tool surface, flat access requires 20,284 initialization tokens under the reported measurement procedure. Gateway access requires 307 initial tokens, and domain-sub-agent access requires 627 root-context tokens. These figures measure standing schema cost only; total cost also includes routing, delegation, observations, and generated artifacts.

## D.4 ABLATION PROTOCOL AND DETAILED RESULTS

All ablations use matched scenario IDs. Within each model–scenario comparison, the manifest, initialized state, evidence, MCP implementations, prompts, inference settings, and scoring procedure are fixed. Only K0 versus K1 skill access, flat versus gateway access, or flat versus domain-sub-agent access changes.

The 35-scenario subset was selected before ablation runs through proportional stratified sampling across all workflow families. Full per-model and workflow-family results appear in Section H.

Each run records the model and provider, inference configuration, turn and token budgets, timeout, tool-call limit, topology, and skill-library revision. Matched comparisons retain identical settings within each model–scenario pair.

# E EXPERIMENTAL DETAILS

## E.1 COMMON MODEL AND AGENT CONFIGURATION

Unless stated otherwise, full-suite results use one execution per model–scenario pair under a common Stirrup configuration. We hold the scenario instance, request, MCP implementations, tool permissions, system prompt, inference settings, and evaluator fixed across models. Runs use temperature 0, high reasoning effort where supported, and a 30-turn limit; we do not use self-consistency, reranking, or model-specific prompting.

The main evaluation uses code-enabled flat access: the root agent receives all 85 typed tools from six MCP servers and an isolated Docker code workspace. Domain data must be acquired through MCP tools; direct access to backing services, external web tools, and browsing are disabled. Stirrup alternates model calls and tool execution, recording the final response through its `finish` action. Table 13 gives the full fixed configuration, including context, timeout, and retry settings.

Table 13: Fixed Stirrup configuration for the full-suite evaluation.

| Component | Configuration |
|---|---|
| Agent framework | Stirrup 0.2.0 |
| Tool topology | Flat access to all six MCP servers |
| Domain tools | 85 typed MCP tools, extra 2 tools (`code_exec` and `finish`) |
| Code execution | Enabled with an isolated Docker workspace |
| External web tools | Disabled |
| Temperature | 0 |
| Reasoning effort | High when supported by the provider |
| Maximum turns | 30 |
| Working context budget | 100,000 tokens |
| Maximum model output | 16,000 tokens per call |
| Context summarization | Triggered at 60% of the working-context budget |
| Request timeout and retries | 900 seconds and one retry |
| Sampling protocol | One execution per model–scenario pair |

## E.2 MODEL ROUTING

Table 14 lists the exact model identifiers and access routes used in the full-suite evaluation. LiteLLM-routed models use the institutional LiteLLM-compatible endpoint (BerriAI, 2026); TokenRouter-routed models use the TokenRouter-compatible endpoint (TokenRouter, 2026). Both routes use the same Stirrup agent loop and OpenAI-compatible chat-completions interface.

Table 14: Model identifiers and access routes for the full-suite evaluation.

| Model | Route | Model identifier |
|---|---|---|
| Gemini 3.6 Flash (Google DeepMind, 2026) | LiteLLM | `litellm_proxy/gcp/gemini-3.6-flash` |
| GPT-5.6 (OpenAI, 2026) | LiteLLM | `litellm_proxy/azure/gpt-5.6-sol` |
| Claude Opus 5 (Anthropic, 2026a) | LiteLLM | `litellm_proxy/aws/claude-opus-5` |
| Claude Sonnet 5 (Anthropic, 2026b) | LiteLLM | `litellm_proxy/aws/claude-sonnet-5` |
| Qwen3.8-Max (Yang et al., 2025a) | TokenRouter | `tokenrouter/qwen/qwen3.8-max` |
| MiniMax-M3 (MiniMax, 2026) | TokenRouter | `tokenrouter/MiniMax-M3` |
| Kimi K3 (Team et al., 2026) | TokenRouter | `tokenrouter/moonshotai/kimi-k3` |
| DeepSeek-V4-Flash (DeepSeek-AI et al., 2026) | TokenRouter | `tokenrouter/deepseek/deepseek-v4-flash` |

# F Evaluation Protocol and Diagnostic Metrics

Following Agents' Last Exam (Sun et al., 2026a) and ThinkingBox (Li et al., 2026d), AssetArena decouples scenario specification, agent execution, and evaluation. The evaluator scores saved trajectories against scenario-specific reference artifacts without rerunning the agent. Each scenario declares a deterministic scoring configuration: default structured tasks require normalized exact matching, CAR uses a mode-aware variant, and FMEA uses semantic F1. Thus, Pass@1 denotes completion under a scenario's predeclared criterion rather than a universal string-matching rule.

## F.1 Scorer Assignment and Pass Criteria

The default structured evaluator applies to WO, FCC, HLT, FMSR, and TSFM. It parses and normalizes nested JSON objects, arrays, Python-style literals, and scalar answers before requiring an exact reference match. CAR uses a mode-aware structured evaluator. The response must contain exactly one of `response`, `clarification`, or `abstain`; that mode must match the reference, and the response must include a required decision term. Mode selection and term coverage are also retained as diagnostic scores. FMEA covers asset-specific failure-mode inventories and catalogue-analytic tasks. Inventory answers are compared as failure-mode sets; catalogue-analytic answers are decomposed into atomic facts. Both use semantic F1, and a scenario passes only when semantic F1 is strictly greater than $0.70$. Table 15 summarizes the scorer assignment and pass criterion for each scenario family.

Table 15: Scenario-family scoring rules.

| Families | Scorer | Pass criterion |
|---|---|---|
| WO, FCC, HLT, FMSR, TSFM | Structured | Exact normalized structured match. |
| CAR | Structured, mode-aware | Correct single mode key and at least one required decision term. |
| FMEA | Semantic F1 | Semantic F1 $> 0.70$; format, rubric, and tool-use fields are diagnostic only. |

## F.2 Diagnostic and Operational Measurements

For default structured tasks, the evaluator records partial exact-match accuracy, similarity, numeric and range agreement where applicable, precision, recall, F1, and missing or extra answer keys. These diagnostics characterize partial correctness and output-contract failures but do not alter strict Pass@1. For CAR, the evaluator records mode-key correctness and required-term coverage. For FMEA, it records semantic precision, recall, F1, format compliance, rubric-weighted diagnostics, required-catalogue use, and out-of-scope domain tool use. These fields support failure analysis but do not change the pass decision.

**Tool selection and argument grounding.** AssetArena does not prescribe a canonical tool sequence. Agents choose both a tool and its arguments at each step, so selecting an appropriate operation is insufficient when the asset identifier, sensor, interval, model, or intermediate artifact is incorrect. Multiple valid trajectories may satisfy a scenario; evaluation therefore assesses the outcome and required observable conditions rather than conformance to one fixed procedure.

For each run, the evaluator derives turns, tool calls, unique tools, input and output tokens, end-to-end time, and estimated cost from the stored trajectory. Main tables report per-run means, while aggregate reports retain per-run details and distributional runtime statistics. Cost is unavailable when no reliable price entry exists for the routed model.

## F.3 Pairwise Ranking Uncertainty and Cross-Model Coverage

We supplement the primary scenario-specific Pass@1 evaluation with a regularized Bradley–Terry analysis (Bradley & Terry, 1952; Elo, 1967). For each scenario and model pair, a model receives a

win when it passes and its counterpart fails. Shared success and shared failure are ties. We fit model strengths with a Gaussian prior of standard deviation 4.0 on the logit scale and constrain strengths to sum to zero. We transform fitted strengths to an Elo scale centered at 1500, where a 400-point difference corresponds to approximately 10:1 pairwise win odds.

We estimate percentile-bootstrap 95% intervals using 1,000 scenario-level resamples. Four bootstrap replicates do not converge and are excluded. No replicate exhibits near-separation. Since overlapping marginal rating intervals are not pairwise hypothesis tests, we additionally conduct two-sided exact McNemar tests on the discordant scenarios for all 28 model pairs and apply Holm correction. We use the corrected paired tests for inferential claims and report bootstrap intervals as descriptive uncertainty around each model's fitted rating.

Figure 13 reports the ranking and each model's exclusive contribution. Qwen3.8-Max has the highest Elo estimate, followed by Claude Opus 5 and Claude Sonnet 5. However, the leading estimates are not pairwise separated after correction. The ratings are supplementary rather than a replacement for the benchmark's predefined Pass@1 measure. They reduce each scenario to a binary outcome and discard the magnitude of partial correctness, including semantic-F1 information retained for FMEA scenarios.

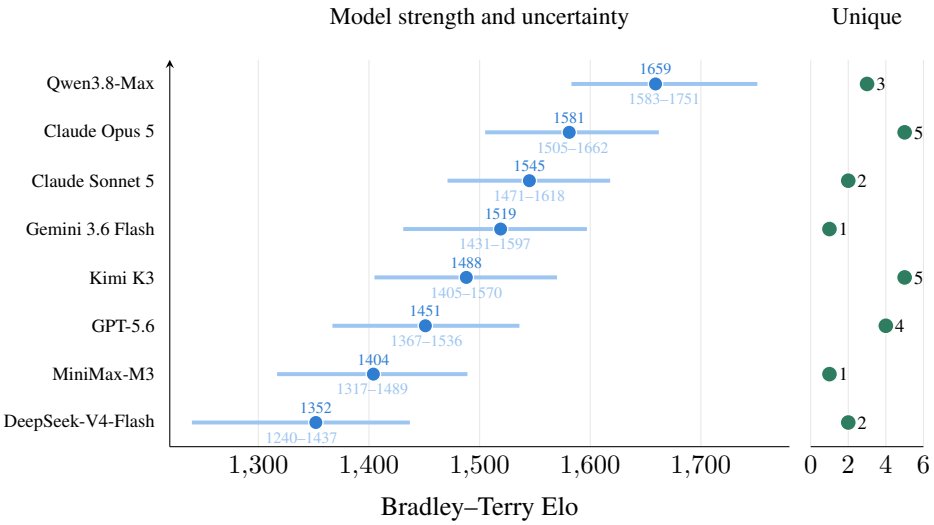



Figure 13: Model strength, uncertainty, and exclusive scenario coverage on the 231-scenario common cohort. Blue dots show regularized Bradley–Terry Elo estimates, light-blue intervals show 95% confidence intervals, and green markers show uniquely passed scenarios.

Table 16 lists the five separations that survive Holm correction. These are non-adjacent comparisons. In particular, the Qwen3.8-Max versus Claude Opus 5 result does not support a claim of superiority despite Qwen3.8-Max having the larger point estimate.

Table 16: Model pairs separated by two-sided exact McNemar tests after Holm correction across all 28 pairs. $b$ counts scenarios passed only by the higher-ranked model; $c$ counts the reverse; **$\Delta$ Elo** is the rating difference; and $p_{\text{Holm}}$ is the corrected significance value.

| Higher-ranked model | Lower-ranked model | $b$ | $c$ | $\Delta$ Elo | $p_{\text{Holm}}$ |
|---|---|---|---|---|---|
| Qwen3.8-Max | GPT-5.6 | 32 | 11 | 208 | .0459 |
| Qwen3.8-Max | MiniMax-M3 | 32 | 7 | 255 | .0019 |
| Qwen3.8-Max | DeepSeek-V4-Flash | 39 | 8 | 307 | .0002 |
| Claude Opus 5 | DeepSeek-V4-Flash | 35 | 11 | 228 | .0139 |
| Claude Sonnet 5 | DeepSeek-V4-Flash | 26 | 7 | 193 | .0330 |

Cross-model success overlap reveals both complementarity and a substantial shared capability gap. At least one model solves 105/231 scenarios (45.5%), which is the empirical upper bound for a perfect hindsight router over this fixed model pool. The strongest individual model, Qwen3.8-Max, solves 69/231 scenarios (29.9%). Twenty-three scenarios (10.0%) are solved by exactly one model, showing model-specific strengths. Conversely, 15 scenarios (6.5%) are solved by all eight models and 126/231 (54.5%) are solved by none. The oracle is not a deployable ensemble result because it assumes advance knowledge of which model will succeed and ignores routing cost.

The success distribution is uneven. The numbers of scenarios solved by $k \in \{0, \ldots, 8\}$ models are $[126, 23, 18, 15, 4, 11, 3, 16, 15]$. Thus, 90/231 scenarios (39.0%) lie in the discriminating range $0 < k < 8$, while the largest group is the shared hard mass at $k = 0$. The $k = 4$ and $k = 6$ bins are particularly sparse, with four and three scenarios, respectively.

This structure limits the number of pairwise comparisons that can distinguish models. Across 231 scenarios and 28 model pairs, 979 of 6,468 possible pairwise comparisons are decided by different pass–fail outcomes, or 15.1%. This is 26.5% of the number of decided comparisons in an ideally discriminating suite. Nevertheless, discrimination is concentrated in the 90 intermediate scenarios. The spread between the best and worst model rises from 13.4 percentage points over all 231 scenarios to 34.4 points over the intermediate subset. These results characterize the interaction between the current model pool, benchmark, and scoring protocol. They should not be interpreted as a model-independent measure of intrinsic scenario difficulty.

# G Trajectory Failure Analysis

We analyze the 1,848 scored trajectories from the eight-model full-suite Stirrup evaluation. The aggregate report covers every raw trajectory, including failed executions. We use deterministic scorer fields to identify outcome, partial-correctness, output-contract, and FMEA catalogue-process indicators. Tool-error objects and repeated calls are conservative trace indicators: they prioritize inspection but do not, on their own, establish a causal root cause.

## G.1 Failure Taxonomy

We locate failures along the evidence-to-action pipeline rather than treating every unsuccessful trajectory as a single outcome. Table 17 and Figure 14 define the codebook used for trajectory review. A trajectory may receive more than one label when an upstream failure causes a downstream error. The taxonomy distinguishes observed outcome diagnostics from causal attribution: output format, tool errors, and selected FMEA and CAR fields are recorded automatically, whereas evidence, reasoning, and integration failures require trajectory review. This taxonomy separates failures to obtain evidence from failures to use, integrate, or communicate it. We report automatic diagnostics descriptively and reserve causal category rates for trajectories that have undergone blinded manual annotation under this codebook.

**Structured-output failures.** Among 1,428 failed non-FMEA trajectories, 444 (31.1%) contain missing or extra keys. Their mean partial-exact score is 0.185. This shows that partial correctness is common, but also that most failures cannot be reduced to a simple formatting explanation. For example, Claude Opus 5 on scenario 932 returned every gold item but included one unsupported extra item, yielding a partial-exact score of 1.000 while failing strict scoring.

**FMEA grounding failures.** Of 136 FMEA trajectories, 87 fail the semantic-F1 threshold; their mean semantic F1 is 0.243. Seventy-nine failed runs (90.8%) violate the catalogue-process diagnostic. No failed run omits a required catalogue tool, whereas 79 invoke an out-of-scope domain tool. These diagnostics are not scoring gates; they distinguish tool access from the harder problem of selecting and grounding the appropriate failure-mode set. DeepSeek-V4-Flash on scenario 9025 illustrates this pattern: it attained semantic F1 of 0.045, with precision 0.024 and coverage 0.333.

**Trace-level interaction indicators.** Figure 15 places automatic failure indicators in model-level context. Panel A relates Pass@1 to mean evaluator score, whereas Panel B reports key-error counts and failed runs containing tool-error objects. These signals are descriptive trace indicators, not

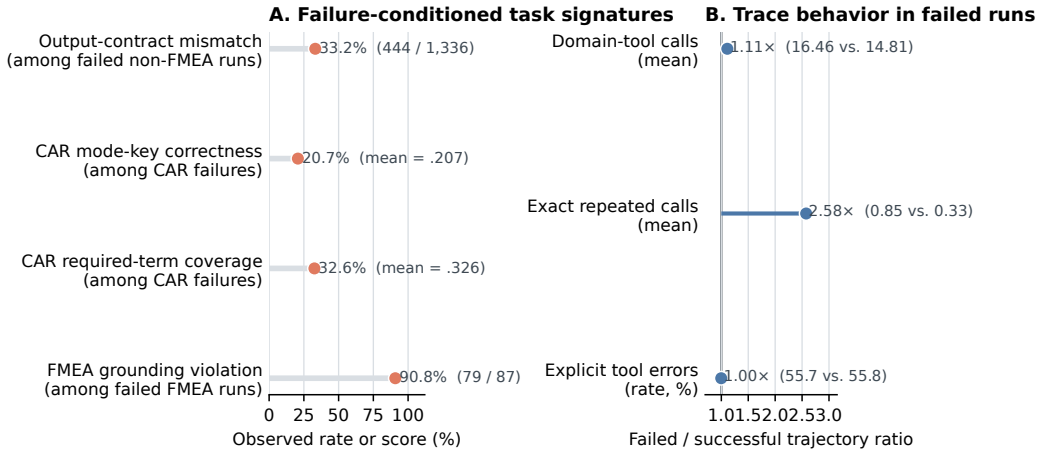



Figure 14: Automatic failure-signature profile across 1,848 trajectories. Panel A reports task-specific, failure-conditioned diagnostics. Panel B compares trace behavior in failed and successful trajectories. These measurements are descriptive indicators rather than causal failure rates; evidence acquisition, tool selection, reasoning, and cross-source integration require blinded trace review.

Table 17: Failure taxonomy for AssetArena trajectory analysis. Categories are organized by the stage at which the workflow fails; execution and infrastructure failures may occur at any stage.

| Workflow locus | Failure category | Definition |
|---|---|---|
| Evidence access | Evidence acquisition | The necessary source, record, artifact, or tool was never reached. |
| Tool interaction | Tool selection or tool argument | The agent selected an irrelevant or inappropriate tool, or issued an invalid, underspecified, or otherwise ineffective query. |
| Reasoning over evidence | Analysis or computation | Relevant evidence was retrieved but transformed, aggregated, calculated, or interpreted incorrectly. |
| Reasoning over evidence | Cross-source integration | Individually relevant evidence was retrieved but not combined correctly across records, tools, or data modalities. |
| Decision and grounding | Decision | The agent selected the wrong response mode—direct response, clarification, or abstention—in a CAR scenario. |
| Decision and grounding | Set selection or grounding | In an FMEA scenario, the agent omitted required modes, hallucinated modes, or drew modes from an inappropriate domain source. |
| Response production | Output contract | The final response violated the required interface, for example through invalid JSON, missing or extra keys, or non-compliant formatting. |
| Cross-cutting | Execution or infrastructure | A tool, timeout, backend, or environment error prevented completion, independently of the agent's intended reasoning. |

causal failure labels. Failed trajectories average 16.14 domain calls, compared with 15.82 for passed trajectories, and contain 0.80 exact repeated calls versus 0.39 for passed trajectories. Explicit tool-error objects occur in 848 of 1,515 failed runs (56.0%) but also in 182 of 333 successful runs. We therefore do not interpret these signals as causal explanations without manual trace review.

## H Complete Agent-Design Ablations

**K1 skills.** Table 18 compares flat Stirrup execution with the K1 skills-enabled condition. K1 skills increase completion from 87/280 (31.1%) to 89/280 (31.8%) and mean primary score from .464 to

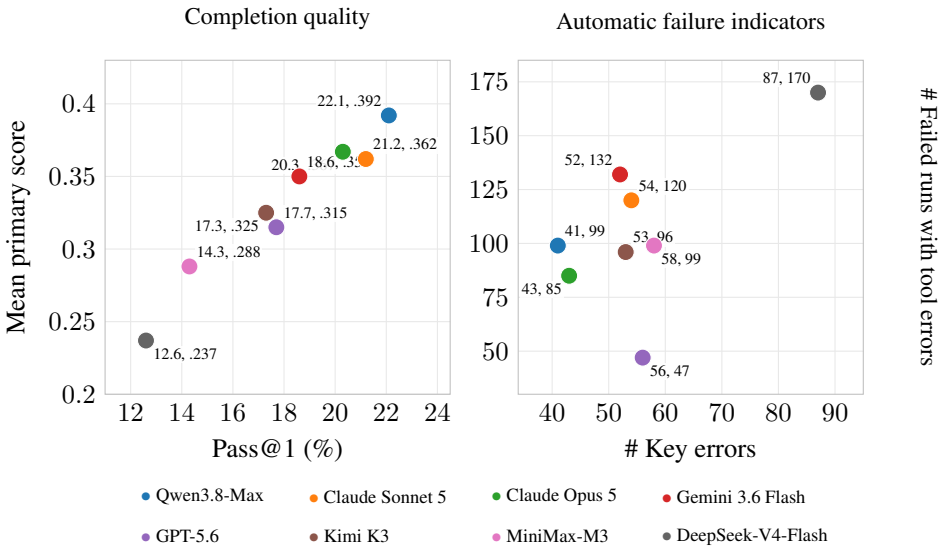



Figure 15: Model-level completion quality and automatic failure indicators. Colored markers identify models through the legend. Numeric labels give (Pass@1, score) on the left and (# key errors, # failed runs with tool errors) on the right. Failure indicators are descriptive trace measures, not causal labels.

.486. The binary gain is small and not statistically decisive: 17 model–scenario pairs are solved only with skills, whereas 15 are solved only without them (two-sided exact McNemar $p = .86$).

The skills condition increases mean turns from 14.7 to 21.0, tool calls from 23.9 to 30.1, and input tokens from 502.9k to 773.1k, corresponding to increases of 42.9%, 25.9%, and 53.7%, respectively. Runtime changes only slightly, from 276.0 to 285.7 seconds. The score improvement is larger than the completion improvement, suggesting that the additional procedural information sometimes improves intermediate artifacts without consistently crossing the strict pass threshold.

Effects also vary by model and workflow family. K1 improves completion for Claude Opus 5, GPT-5.6, and DeepSeek-V4-Flash, leaves Claude Sonnet 5, Kimi K3, and Qwen3.8-Max unchanged, and lowers completion for Gemini 3.6 Flash and MiniMax-M3. FMEA completion decreases from 37.5% to 33.3%, whereas TSFM increases from 60.0% to 67.5%. Thus, the current 38-graph, 217-skill library does not yield a uniform accuracy benefit. This pattern is consistent with evidence that skill selection and library size matter, since broad skill collections may add retrieval and context costs while smaller curated sets can be more effective (Li et al., 2026b). Our repository-distilled skills provide procedural knowledge, but the results indicate that selecting the relevant knowledge remains difficult (Chen et al., 2026).

**Domain sub-agents.** Table 19 evaluates decomposition through domain-specific sub-agents. Relative to flat access, sub-agents reduce completion from 87/280 (31.1%) to 84/280 (30.0%) and mean score from .464 to .442. The aggregate decline masks heterogeneous model-level effects. Sub-agent execution improves Claude Opus 5 by three passes, GPT-5.6 and Gemini 3.6 Flash by one pass each, but decreases completion for Claude Sonnet 5, Kimi K3, Qwen3.8-Max, DeepSeek-V4-Flash, and MiniMax-M3.

Decomposition substantially increases interaction. Mean turns rise from 14.7 to 34.9 and tool calls from 23.9 to 68.9, increases of 137.4% and 188.3%. Runtime rises from 276.0 to 366.0 seconds, while input-token use falls from 502.9k to 330.2k. The reduced token use reflects delegation and smaller sub-agent contexts, but this saving does not compensate for the increased coordination burden in this setting.

At the workflow-family level, sub-agents improve HLT and TSFM completion, from 43.8% to 50.0% and from 3.1% to 15.6%, respectively, but reduce FCC, FMSR, and FMEA performance. These

Table 18: Paired comparison of **No skills** and **K1 skills** execution under Stirrup on the same 35 stratified scenarios per model. Category columns report Pass@1 (%); operational statistics are per-run means.

| Model | Setting | P/N | All | WO | CAR | FCC | HLT | FMSR | TSFM | FMEA | Score | Turns | Calls | Time | Tok. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Claude Opus 5** | **No skills** | 12/35 | 34.3 | 0.0 | 50.0 | 50.0 | 50.0 | 80.0 | 0.0 | 33.3 | .529 | 10.3 | 16.3 | 171.3 | 406.2 |
| | **K1 skills** | 13/35 | 37.1 | 0.0 | 50.0 | 75.0 | 50.0 | 60.0 | 0.0 | 66.7 | .541 | 19.5 | 25.5 | 260.9 | 862.6 |
| **Claude Sonnet 5** | **No skills** | 11/35 | 31.4 | 0.0 | 37.5 | 75.0 | 50.0 | 60.0 | 0.0 | 33.3 | .500 | 18.2 | 35.1 | 755.2 | 774.1 |
| | **K1 skills** | 11/35 | 31.4 | 11.1 | 37.5 | 50.0 | 50.0 | 80.0 | 0.0 | 0.0 | .516 | 24.2 | 38.7 | 751.8 | 1133.7 |
| **GPT-5.6** | **No skills** | 9/35 | 25.7 | 0.0 | 12.5 | 25.0 | 50.0 | 60.0 | 25.0 | 66.7 | .442 | 7.8 | 15.9 | 128.7 | 179.6 |
| | **K1 skills** | 11/35 | 31.4 | 0.0 | 37.5 | 50.0 | 50.0 | 60.0 | 0.0 | 66.7 | .464 | 15.7 | 28.6 | 144.0 | 442.1 |
| **Gemini 3.6 Flash** | **No skills** | 10/35 | 28.6 | 0.0 | 50.0 | 50.0 | 50.0 | 40.0 | 0.0 | 33.3 | .452 | 23.4 | 25.2 | 213.0 | 693.9 |
| | **K1 skills** | 9/35 | 25.7 | 0.0 | 25.0 | 50.0 | 50.0 | 60.0 | 0.0 | 33.3 | .450 | 26.1 | 27.9 | 211.5 | 886.5 |
| **Kimi K3** | **No skills** | 11/35 | 31.4 | 0.0 | 50.0 | 50.0 | 50.0 | 40.0 | 0.0 | 66.7 | .441 | 11.9 | 20.8 | 355.2 | 375.2 |
| | **K1 skills** | 11/35 | 31.4 | 0.0 | 37.5 | 75.0 | 50.0 | 40.0 | 0.0 | 66.7 | .495 | 17.7 | 25.4 | 264.8 | 557.8 |
| **Qwen3.8-Max** | **No skills** | 15/35 | 42.9 | 0.0 | 62.5 | 100.0 | 50.0 | 80.0 | 0.0 | 33.3 | .570 | 13.4 | 23.9 | 304.4 | 449.9 |
| | **K1 skills** | 15/35 | 42.9 | 0.0 | 62.5 | 100.0 | 50.0 | 80.0 | 0.0 | 33.3 | .617 | 19.9 | 33.3 | 410.5 | 758.4 |
| **DeepSeek-V4-Flash** | **No skills** | 8/35 | 22.9 | 0.0 | 25.0 | 75.0 | 0.0 | 60.0 | 0.0 | 0.0 | .323 | 20.1 | 34.1 | 126.1 | 728.5 |
| | **K1 skills** | 9/35 | 25.7 | 0.0 | 25.0 | 50.0 | 0.0 | 80.0 | 25.0 | 0.0 | .377 | 23.6 | 32.7 | 125.9 | 846.2 |
| **MiniMax-M3** | **No skills** | 11/35 | 31.4 | 0.0 | 37.5 | 75.0 | 50.0 | 60.0 | 0.0 | 33.3 | .454 | 12.7 | 19.7 | 154.3 | 416.2 |
| | **K1 skills** | 10/35 | 28.6 | 0.0 | 50.0 | 25.0 | 50.0 | 80.0 | 0.0 | 0.0 | .426 | 21.0 | 29.0 | 116.5 | 697.2 |
| **Paired micro-average** | **No skills** | **87/280** | **31.1** | 0.0 | 40.6 | 62.5 | 43.8 | 60.0 | 3.1 | 37.5 | .464 | 14.7 | 23.9 | 276.0 | 502.9 |
| | **K1 skills** | **89/280** | **31.8** | 1.4 | 40.6 | 59.4 | 43.8 | 67.5 | 3.1 | 33.3 | .486 | 21.0 | 30.1 | 285.7 | 773.1 |

*Notes.* P/N denotes passed/paired runs. WO: work-order scenarios; CAR: clarification, abstention, and direct response; FCC: failure-code classification; HLT: asset-health assessment; FMSR: failure-mode–sensor reasoning; TSFM: time-series foundation-model tasks; FMEA: failure-mode identification. Score is mean evaluator score; Time is mean execution time in seconds; Tok. is mean input tokens in thousands. The micro-average weights each model–scenario pair equally. All eight models have complete 35-scenario coverage in both conditions.

Table 19: Paired comparison of **Flat** and **Sub-agent** Stirrup execution on the same 35 stratified scenario IDs per model. Category columns report Pass@1 (%); operational statistics are per-run means.

| Model | Setting | P/N | All | WO | CAR | FCC | HLT | FMSR | TSFM | FMEA | Score | Turns | Calls | Time | Tok. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Claude Opus 5** | **Flat** | 12/35 | 34.3 | 0.0 | 50.0 | 50.0 | 50.0 | 80.0 | 0.0 | 33.3 | .529 | 10.3 | 16.3 | 171.3 | 406.2 |
| | **Sub-agent** | 15/35 | 42.9 | 0.0 | 50.0 | 50.0 | 50.0 | 100.0 | 25.0 | 66.7 | .562 | 22.6 | 37.7 | 245.8 | 226.6 |
| **Claude Sonnet 5** | **Flat** | 11/35 | 31.4 | 0.0 | 37.5 | 75.0 | 50.0 | 60.0 | 0.0 | 33.3 | .500 | 18.2 | 35.1 | 755.2 | 774.1 |
| | **Sub-agent** | 10/35 | 28.6 | 0.0 | 12.5 | 50.0 | 50.0 | 60.0 | 50.0 | 33.3 | .424 | 32.6 | 51.4 | 402.5 | 342.8 |
| **GPT-5.6** | **Flat** | 9/35 | 25.7 | 0.0 | 12.5 | 25.0 | 50.0 | 60.0 | 25.0 | 66.7 | .442 | 7.8 | 15.9 | 128.7 | 179.6 |
| | **Sub-agent** | 10/35 | 28.6 | 0.0 | 37.5 | 50.0 | 50.0 | 60.0 | 0.0 | 33.3 | .454 | 25.6 | 48.7 | 186.4 | 148.4 |
| **Gemini 3.6 Flash** | **Flat** | 10/35 | 28.6 | 0.0 | 50.0 | 50.0 | 50.0 | 40.0 | 0.0 | 33.3 | .452 | 23.4 | 25.2 | 213.0 | 693.9 |
| | **Sub-agent** | 11/35 | 31.4 | 0.0 | 50.0 | 50.0 | 50.0 | 40.0 | 25.0 | 33.3 | .466 | 47.8 | 55.6 | 293.6 | 461.5 |
| **Kimi K3** | **Flat** | 11/35 | 31.4 | 0.0 | 50.0 | 50.0 | 50.0 | 40.0 | 0.0 | 66.7 | .441 | 11.9 | 20.8 | 355.2 | 375.2 |
| | **Sub-agent** | 10/35 | 28.6 | 0.0 | 25.0 | 75.0 | 50.0 | 40.0 | 0.0 | 66.7 | .398 | 35.5 | 79.2 | 552.0 | 234.4 |
| **Qwen3.8-Max** | **Flat** | 15/35 | 42.9 | 0.0 | 62.5 | 100.0 | 50.0 | 80.0 | 0.0 | 33.3 | .570 | 13.4 | 23.9 | 304.4 | 449.9 |
| | **Sub-agent** | 14/35 | 40.0 | 0.0 | 75.0 | 50.0 | 50.0 | 60.0 | 25.0 | 33.3 | .519 | 29.0 | 85.3 | 733.9 | 368.6 |
| **DeepSeek-V4-Flash** | **Flat** | 8/35 | 22.9 | 0.0 | 25.0 | 75.0 | 0.0 | 60.0 | 0.0 | 0.0 | .323 | 20.1 | 34.1 | 126.1 | 728.5 |
| | **Sub-agent** | 7/35 | 20.0 | 0.0 | 25.0 | 75.0 | 50.0 | 20.0 | 0.0 | 0.0 | .302 | 58.0 | 141.3 | 319.1 | 562.6 |
| **MiniMax-M3** | **Flat** | 11/35 | 31.4 | 0.0 | 37.5 | 75.0 | 50.0 | 60.0 | 0.0 | 33.3 | .454 | 12.7 | 19.7 | 154.3 | 416.2 |
| | **Sub-agent** | 7/35 | 20.0 | 0.0 | 12.5 | 50.0 | 50.0 | 60.0 | 0.0 | 0.0 | .408 | 28.3 | 51.9 | 194.8 | 296.3 |
| **Paired micro-average** | **Flat** | **87/280** | **31.1** | 0.0 | 40.6 | 62.5 | 43.8 | 60.0 | 3.1 | 37.5 | .464 | 14.7 | 23.9 | 276.0 | 502.9 |
| | **Sub-agent** | **84/280** | **30.0** | 0.0 | 35.9 | 56.2 | 50.0 | 55.0 | 15.6 | 33.3 | .442 | 34.9 | 68.9 | 366.0 | 330.2 |

*Notes.* P/N denotes passed/paired runs. WO: work orders; CAR: clarification, abstention, and direct response; FCC: failure-code classification; HLT: asset-health assessment; FMSR: failure-mode–sensor reasoning; TSFM: time-series modeling; FMEA: failure-mode identification. Score is the mean primary evaluator score; Time is mean end-to-end execution time in seconds; and Tok. is mean input tokens in thousands. All eight models have complete coverage of the same 35 scenario IDs in both conditions.

results suggest that decomposition can help when a task aligns with a domain-specialized workflow, but it is not a reliable general-purpose replacement for direct flat-tool execution.

Table 20: Paired comparison of **Flat** and **MCP gateway** Stirrup execution on the same 35 stratified scenario IDs per model. Category columns report Pass@1 (%); operational statistics are per-run means.

| Model | Setting | P/N | All | WO | CAR | FCC | HLT | FMSR | TSFM | FMEA | Score | Turns | Calls | Time | Tok. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Claude Opus 5** | **Flat** | 12/35 | 34.3 | 0.0 | 50.0 | 50.0 | 50.0 | 80.0 | 0.0 | 33.3 | .529 | 10.3 | 16.3 | 171.3 | 406.2 |
| | **MCP gateway** | 13/35 | 37.1 | 0.0 | 37.5 | 75.0 | 50.0 | 80.0 | 25.0 | 33.3 | .542 | 12.7 | 18.7 | 136.8 | 162.4 |
| **Claude Sonnet 5** | **Flat** | 11/35 | 31.4 | 0.0 | 37.5 | 75.0 | 50.0 | 60.0 | 0.0 | 33.3 | .500 | 18.2 | 35.1 | 755.2 | 774.1 |
| | **MCP gateway** | 13/35 | 37.1 | 0.0 | 50.0 | 75.0 | 50.0 | 60.0 | 25.0 | 33.3 | .465 | 17.5 | 24.2 | 193.0 | 224.3 |
| **GPT-5.6** | **Flat** | 9/35 | 25.7 | 0.0 | 12.5 | 25.0 | 50.0 | 60.0 | 25.0 | 66.7 | .442 | 7.8 | 15.9 | 128.7 | 179.6 |
| | **MCP gateway** | 9/35 | 25.7 | 0.0 | 25.0 | 50.0 | 50.0 | 40.0 | 0.0 | 66.7 | .418 | 12.8 | 19.7 | 81.7 | 105.4 |
| **Gemini 3.6 Flash** | **Flat** | 10/35 | 28.6 | 0.0 | 50.0 | 50.0 | 50.0 | 40.0 | 0.0 | 33.3 | .452 | 23.4 | 25.2 | 213.0 | 693.9 |
| | **MCP gateway** | 11/35 | 31.4 | 0.0 | 25.0 | 75.0 | 50.0 | 60.0 | 25.0 | 33.3 | .467 | 24.9 | 26.5 | 262.4 | 351.5 |
| **Kimi K3** | **Flat** | 11/35 | 31.4 | 0.0 | 50.0 | 50.0 | 50.0 | 40.0 | 0.0 | 66.7 | .441 | 11.9 | 20.8 | 355.2 | 375.2 |
| | **MCP gateway** | 12/35 | 34.3 | 0.0 | 50.0 | 50.0 | 50.0 | 40.0 | 25.0 | 66.7 | .484 | 15.2 | 22.1 | 232.4 | 151.2 |
| **Qwen3.8-Max** | **Flat** | 15/35 | 42.9 | 0.0 | 62.5 | 100.0 | 50.0 | 80.0 | 0.0 | 33.3 | .570 | 13.4 | 23.9 | 304.4 | 449.9 |
| | **MCP gateway** | 14/35 | 40.0 | 0.0 | 50.0 | 50.0 | 50.0 | 80.0 | 25.0 | 66.7 | .519 | 16.6 | 26.7 | 248.8 | 222.9 |
| **DeepSeek-V4** | **Flat** | 8/35 | 22.9 | 0.0 | 25.0 | 75.0 | 0.0 | 60.0 | 0.0 | 0.0 | .323 | 20.1 | 34.1 | 126.1 | 728.5 |
| | **MCP gateway** | 8/35 | 22.9 | 0.0 | 25.0 | 75.0 | 50.0 | 20.0 | 25.0 | 0.0 | .381 | 19.7 | 27.8 | 236.0 | 275.8 |
| **MiniMax-M3** | **Flat** | 11/35 | 31.4 | 0.0 | 37.5 | 75.0 | 50.0 | 60.0 | 0.0 | 33.3 | .454 | 12.7 | 19.7 | 154.3 | 416.2 |
| | **MCP gateway** | 10/35 | 28.6 | 0.0 | 37.5 | 50.0 | 50.0 | 60.0 | 0.0 | 33.3 | .462 | 18.8 | 28.4 | 90.1 | 219.5 |
| **Micro-average** | **Flat** | **87/280** | **31.1** | 0.0 | 40.6 | 62.5 | 43.8 | 60.0 | 3.1 | 37.5 | .464 | 14.7 | 23.9 | 276.0 | 502.9 |
| | **MCP gateway** | **90/280** | **32.1** | 0.0 | 37.5 | 62.5 | 50.0 | 55.0 | 18.8 | 41.7 | .467 | 17.3 | 24.3 | 185.2 | 214.1 |

*Notes.* P/N denotes passed/available runs. WO: work-order scenarios; CAR: clarification, abstention, and direct response; FCC: failure-code classification; HLT: asset-health assessment; FMSR: failure-mode–sensor reasoning; TSFM: time-series foundation-model tasks; FMEA: failure-mode and effects analysis. Score is mean evaluator score; Time is mean execution time in seconds; and Tok. is mean input tokens in thousands. The micro-average weights every model–scenario pair equally.

**MCP gateway.** Table 20 evaluates deferred MCP schema disclosure through the gateway interface. Across the matched comparisons, gateway access increases completion from 87/280 (31.1%) to 90/280 (32.1%) and mean score from .464 to .467. The improvement is modest, but it is achieved with substantially lower execution cost. Mean runtime decreases from 276.0 to 185.2 seconds and mean input-token use from 502.9k to 214.1k, a 32.9% runtime reduction and 57.4% token reduction. Mean turns increase from 14.7 to 17.3, while tool calls remain nearly unchanged, from 23.9 to 24.3.

Gateway effects are again model dependent. Completion increases for Claude Opus 5, Claude Sonnet 5, Gemini 3.6 Flash, and Kimi K3; it is unchanged for GPT-5.6; and it decreases for Qwen3.8-Max and MiniMax-M3. At the aggregate family level, the gateway improves HLT, TSFM, and FMEA completion while reducing CAR and FMSR completion. Overall, deferred schema disclosure appears to be an efficient interface control: it preserves or modestly improves aggregate completion while sharply reducing the token cost of exposing a large tool surface.

The summary of all agent configurations are presented in Table 21.

Table 21: Controlled agent-design effects across eight models and 280 matched evaluations.

| Metric | K1 skills | Sub-agents | MCP gateway |
|---|---|---|---|
| Pass@1 | 31.1 → 31.8 | 31.1 → 30.0 | 31.1 → **32.1** |
| Score | .464 → **.486** | .464 → .442 | .464 → .467 |
| Δ Turns | +42.9 | +137.4 | **+17.7** |
| Δ Calls | +25.9 | +188.3 | **+1.7** |
| Δ Time | +3.5 | +32.6 | **−32.9** |
| Δ Tok. | +53.7 | −34.3 | **−57.4** |

*Notes.* Arrows denote flat → intervention. Score is mean primary evaluator score; values below are percentage changes. Negative Time and Tok. indicate lower use. Shading marks the gateway column; bold marks the best value in each row. No completion difference is statistically decisive.

# I Complete Execution-Harness Results

Table 22 reports the full matched comparison between Stirrup and OpenCode. The paired aggregate comprises 280 model–scenario evaluations.

Across these matched pairs, Stirrup achieves 31.1% Pass@1, compared with 25.0% for OpenCode, and a higher mean primary score (.464 versus .386). The decline is broad across workflow families: OpenCode performs worse on CAR, FCC, HLT, FMSR, and FMEA, although it performs better on WO and TSFM, which start from low baseline rates. Performance is also heterogeneous across models. GPT-5.6 improves from 25.7% to 31.4%, DeepSeek-V4-Flash remains at 22.9%, and the remaining six models decline under OpenCode, most sharply MiniMax-M3 and Qwen3.8-Max.

Table 22: Paired comparison of **Stirrup** and **OpenCode** on the 35-scenario stratified subset. Each model pair is evaluated on identical available scenarios. Category columns report Pass@1 (%); operational statistics are per-run means.

| Model | Agent | P/N | All | WO | CAR | FCC | HLT | FMSR | TSFM | FMEA | Score | Turns | Calls | Time | Tok. | Cost | $/pass |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Claude Opus 5** | **Stirrup** | 12/35 | 34.3 | 0.0 | 50.0 | 50.0 | 50.0 | 80.0 | 0.0 | 33.3 | .529 | 10.3 | 16.3 | 171.3 | 406.2 | 77.23 | 6.44 |
| | **OpenCode** | 10/35 | 28.6 | 0.0 | 37.5 | 50.0 | 50.0 | 60.0 | 0.0 | 33.3 | .467 | 9.3 | 14.6 | 141.6 | 562.0 | 103.42 | 10.34 |
| **Claude Sonnet 5** | **Stirrup** | 11/35 | 31.4 | 0.0 | 37.5 | 75.0 | 50.0 | 60.0 | 0.0 | 33.3 | .495 | 17.9 | 32.7 | 692.4 | 758.6 | 70.57 | 6.42 |
| | **OpenCode** | 10/35 | 28.6 | 0.0 | 25.0 | 50.0 | 50.0 | 60.0 | 25.0 | 33.3 | .389 | 14.1 | 22.1 | 265.8 | 879.7 | 64.62 | 6.46 |
| **GPT-5.6** | **Stirrup** | 9/35 | 25.7 | 0.0 | 12.5 | 25.0 | 50.0 | 60.0 | 25.0 | 66.7 | .442 | 7.8 | 15.9 | 128.7 | 179.6 | 34.33 | 3.81 |
| | **OpenCode** | 11/35 | 31.4 | 11.1 | 12.5 | 75.0 | 50.0 | 40.0 | 25.0 | 66.7 | .475 | 7.0 | 23.3 | 111.1 | 630.3 | 113.68 | 10.33 |
| **Gemini 3.6 Flash** | **Stirrup** | 10/35 | 28.6 | 0.0 | 50.0 | 50.0 | 50.0 | 40.0 | 0.0 | 33.3 | .455 | 22.8 | 24.8 | 212.0 | 667.6 | 35.69 | 3.57 |
| | **OpenCode** | 9/35 | 25.7 | 0.0 | 37.5 | 50.0 | 50.0 | 40.0 | 0.0 | 33.3 | .411 | 16.8 | 18.9 | 193.8 | 792.1 | 40.97 | 4.55 |
| **Kimi K3** | **Stirrup** | 11/35 | 31.4 | 0.0 | 50.0 | 50.0 | 50.0 | 40.0 | 0.0 | 66.7 | .441 | 11.9 | 20.8 | 355.2 | 375.2 | 44.93 | 4.08 |
| | **OpenCode** | 9/35 | 25.7 | 0.0 | 25.0 | 50.0 | 50.0 | 60.0 | 0.0 | 33.3 | .432 | 12.3 | 42.5 | 471.4 | 1141.4 | 127.13 | 14.13 |
| **Qwen3.8-Max** | **Stirrup** | 15/35 | 42.9 | 0.0 | 62.5 | 100.0 | 50.0 | 80.0 | 0.0 | 33.3 | .570 | 13.4 | 23.9 | 304.4 | 449.9 | 34.18 | 2.28 |
| | **OpenCode** | 9/35 | 25.7 | 0.0 | 37.5 | 75.0 | 0.0 | 40.0 | 0.0 | 33.3 | .355 | 13.1 | 34.8 | 424.8 | 1191.7 | 86.92 | 9.66 |
| **DeepSeek-V4-Flash** | **Stirrup** | 8/35 | 22.9 | 0.0 | 25.0 | 75.0 | 0.0 | 60.0 | 0.0 | 0.0 | .323 | 20.1 | 34.1 | 126.1 | 728.5 | 11.70 | 1.46 |
| | **OpenCode** | 8/35 | 22.9 | 0.0 | 37.5 | 25.0 | 0.0 | 60.0 | 25.0 | 0.0 | .308 | 18.4 | 35.2 | 97.8 | 1561.3 | 24.31 | 3.04 |
| **MiniMax-M3** | **Stirrup** | 11/35 | 31.4 | 0.0 | 37.5 | 75.0 | 50.0 | 60.0 | 0.0 | 33.3 | .454 | 12.7 | 19.7 | 154.3 | 416.2 | 4.78 | 0.43 |
| | **OpenCode** | 4/35 | 11.4 | 0.0 | 37.5 | 0.0 | 0.0 | 20.0 | 0.0 | 0.0 | .253 | 12.0 | 26.9 | 140.1 | 1235.2 | 13.55 | 3.39 |
| **Paired aggregate** | **Stirrup** | **87/280** | **31.1** | 0.0 | 40.6 | 62.5 | 43.8 | 60.0 | 3.1 | 37.5 | .464 | 14.5 | 23.5 | 267.1 | 494.9 | 313.41 | 3.60 |
| | **OpenCode** | **70/280** | **25.0** | 1.4 | 31.2 | 46.9 | 31.2 | 47.5 | 9.4 | 29.2 | .386 | 12.8 | 27.4 | 231.1 | 1001.9 | 574.60 | 8.21 |

*Notes.* P/N denotes passed/paired available runs. WO: work-order scenarios; CAR: clarification, abstention, and response; FCC: failure-code classification; HLT: asset-health assessment; FMSR: failure-mode–sensor reasoning; TSFM: time-series foundation-model tasks; FMEA: failure-mode identification. Score is the mean evaluator score; Time is mean execution time in seconds; Tok. is mean input tokens in thousands; and Cost is total estimated cost over the paired runs.

Stirrup exceeds OpenCode by 6.1 percentage points in paired aggregate Pass@1 and by .078 mean primary score. The gap is not attributable to a uniform increase in interaction: OpenCode uses fewer turns and less wall-clock time, but more tool calls and substantially more input tokens. The result indicates that harness design affects the agent's effective control loop, including how it discovers tools, maintains context, and recovers from intermediate actions, even when the underlying model and industrial environment are unchanged.

# J Execution-Budget and Repetition Robustness

## J.1 Maximum-turn budget

Table 23 compares the 30-turn flat Stirrup baseline with a 60-turn budget under otherwise identical conditions. Each of the eight models is evaluated on the same 35 stratified scenario IDs in both conditions. The ablation therefore isolates the effect of the maximum interaction budget, rather than changes in models, prompts, tools, scenario state, inference settings, or evaluation.

The 60-turn budget raises paired micro-average Pass@1 from 31.1% to 32.1% and mean score from .464 to .473 (Table 23). This modest gain is not statistically decisive: 17 model–scenario pairs pass only with 60 turns, whereas 14 pass only with 30 turns (two-sided exact McNemar $p = .72$). The larger budget increases mean turns from 14.7 to 19.7, tool calls from 23.9 to 30.0, runtime from 276.0 to 370.2 s, and input tokens from 502.9k to 708.3k. Thus, additional interaction helps

Table 23: Effect of increasing Stirrup's maximum-turn budget from 30 to 60 on matched stratified-subset evaluations. Values are 60-turn minus 30-turn differences; Pass@1 changes are percentage points. **Positive**, **negative**, and **zero** cells denote the direction of change.

| Model | Paired $N$ | Δ Pass@1 | Δ Score | Δ Turns | Δ Calls | Δ Time (s) | Δ Tok. (k) |
|---|---|---|---|---|---|---|---|
| Claude Opus 5 | 35 | +5.7 | +.026 | +1.1 | +3.9 | −35.3 | +73.3 |
| Claude Sonnet 5 | 35 | +5.7 | −.019 | +9.1 | +15.1 | +305.6 | +475.2 |
| GPT-5.6 | 35 | −2.8 | −.026 | −0.5 | +0.0 | −60.9 | −8.4 |
| Gemini 3.6 Flash | 35 | +2.8 | −.005 | +9.8 | +9.2 | +13.3 | +428.0 |
| MiniMax-M3 | 35 | −5.7 | −.069 | +8.7 | +8.8 | +444.8 | +290.5 |
| DeepSeek-V4-Flash | 35 | +5.7 | +.147 | +6.0 | +5.5 | +65.1 | +313.1 |
| Kimi K3 | 35 | +2.9 | +.045 | +0.3 | −0.7 | −141.1 | −57.6 |
| Qwen3.8-Max | 35 | −5.8 | −.027 | +5.1 | +7.3 | +162.1 | +128.7 |
| **Paired micro-average** | **280** | +1.0 | +.009 | +5.0 | +6.1 | +94.2 | +205.4 |

*Notes.* All values are 60-turn minus 30-turn differences. Score is the mean primary evaluator score; Time is mean end-to-end execution time; and Tok. is mean input tokens in thousands. All models have complete coverage of the same 35 scenario IDs.

selected model–scenario pairs but does not yield a reliable aggregate completion improvement at substantially greater cost.

### J.2 Repeated-trial protocol

We assess execution robustness using three independent trajectories for each fixed model–scenario pair. Repetitions retain the same scenario state, evidence, tools, prompts, inference settings, and evaluation procedure. Observed variation therefore reflects execution-level variability, including provider-side nondeterminism and runtime behavior.

Following the distinction between retry-oriented Pass@k and reliability-oriented Pass$^k$ (Chen et al., 2021; Schmid, 2025), we report Pass@3 and Pass$^3$. Pass@3 credits a scenario if at least one of three executions succeeds, whereas Pass$^3$ requires success in all three. The gap between these measures captures instability that Pass@1 alone cannot reveal.

Table 24 shows substantial execution variability. Across 280 matched model–scenario pairs, Pass@1 is 22.9% (64/280), whereas Pass@3 reaches 34.3% (96/280), yielding 32 additional successful pairs. However, only 15.7% (44/280) succeed in all three executions. Thus, many pairs are solvable intermittently but not reliably.

DeepSeek-V4-Flash illustrates this pattern: Pass@1 is 6/35 (17.1%), Pass@3 is 14/35 (40.0%), and Pass$^3$ is 4/35 (11.4%). Four scenarios succeed in every run, 10 succeed intermittently, and 21 fail in every run. Gemini 3.6 Flash has identical Pass@1, Pass@3, and Pass$^3$ values (7/35), indicating stable but limited coverage. DeepSeek-V4-Flash shows the largest retry gain, followed by Kimi K3, whereas Claude Opus 5 has the strongest all-three-run reliability (8/35).

Table 24: Repeated-execution success and reliability on matched scenarios from the stratified subset. Pass@3 credits a scenario if at least one of three executions succeeds; $\text{Pass}^3$ requires all three executions to succeed.

| **Model** | $N$ | **Pass@1** | **Pass@3** | **$\text{Pass}^3$** |
|---|---|---|---|---|
| Claude Opus 5 | 35 | 9/35 (25.7) | 12/35 (34.3) | 8/35 (22.9) |
| Claude Sonnet 5 | 35 | 8/35 (22.9) | 13/35 (37.1) | 6/35 (17.1) |
| GPT-5.6 | 35 | 8/35 (22.9) | 11/35 (31.4) | 6/35 (17.1) |
| Gemini 3.6 Flash | 35 | 7/35 (20.0) | 7/35 (20.0) | 7/35 (20.0) |
| Kimi K3 | 35 | 7/35 (20.0) | 14/35 (40.0) | 4/35 (11.4) |
| Qwen3.8-Max | 35 | 11/35 (31.4) | 14/35 (40.0) | 6/35 (17.1) |
| DeepSeek-V4-Flash | 35 | 6/35 (17.1) | 14/35 (40.0) | 4/35 (11.4) |
| MiniMax-M3 | 35 | 8/35 (22.9) | 11/35 (31.4) | 3/35 (8.6) |
| **Micro-average** | **280** | **64/280 (22.9)** | **96/280 (34.3)** | **44/280 (15.7)** |

*Notes.* Pass@1 is success on the first execution. Pass@3 credits a scenario if any of its three executions succeeds, whereas $\text{Pass}^3$ requires success on all three executions. All values were recomputed from the saved first-, second-, and third-execution trajectories using the benchmark's deterministic evaluator.

Table 25: Detailed inventory of tools exposed through the AssetArena MCP tool environment.

| Server | Capability | Tool | Purpose |
|---|---|---|---|
| **IoT** | Site and asset discovery | `sites`, `asset_ids`, `assets`, `asset_detail` | List sites; discover asset identifiers and compact asset metadata; and retrieve detailed registry information for a specific asset. |
| **IoT** | Sensor inventory | `installed_sensors`, `measured_sensors` | Retrieve sensor names configured in the asset registry or actually observed in the telemetry stream. |
| **IoT** | Asset–sensor search | `find_assets_by_sensors` | Find assets whose installed or observed sensor names match a requested sensor set. |
| **IoT** | Telemetry access and statistics | `stream_extent`, `latest_reading`, `history`, `sensor_coverage`, `sensor_stats` | Inspect temporal coverage, retrieve recent and historical observations, and compute per-sensor availability and numerical statistics. |
| **Utilities** | Local file access | `json_reader` | Read and parse a JSON file from disk. |
| **Utilities** | Industrial catalog lookup | `get_sensor_catalog`, `get_asset_catalog`, `get_failure_mode_catalog` | Retrieve sensor, asset, and failure-mode catalog entries, optionally using exact-name or category filters. |
| **Utilities** | Time utilities | `current_date_time`, `current_time_english` | Return the current UTC time in structured JSON or human-readable form. |
| **FMSR** | Failure-mode retrieval | `get_failure_modes` | Retrieve known failure modes for an asset class from the failure-mode database. |
| **FMSR** | Failure-mode generation and persistence | `generate_failure_modes`, `add_failure_modes` | Generate a candidate failure-mode list without modifying the database, or persist failure-mode knowledge for an asset class. |
| **WO** | Work-order search and retrieval | `list_workorders`, `get_workorder`, `get_workorder_tasks` | List work orders using operational filters; retrieve one work order; and retrieve its child tasks. |
| **WO** | Cost and execution analysis | `get_workorder_costs`, `get_workorder_actuals_vs_planned` | Retrieve cost breakdowns and compare estimated versus actual labour and cost outcomes. |
| **WO** | Maintenance planning and workforce status | `get_workorder_kpis`, `get_schedule_calendar`, `get_my_assigned_workorders` | Compute site maintenance KPIs, inspect upcoming schedules, and retrieve open work assigned to a technician. |
| **WO** | Reference data | `get_failure_codes` | Retrieve the failure-code reference catalog or a specific failure code. |
| **WO** | Work-order lifecycle management | `generate_work_order`, `update_workorder`, `approve_workorder`, `assign_technician`, `close_workorder`, `cancel_workorder` | Create, update, approve, assign, complete, and cancel maintenance work orders. |
| **TSFM** | Task and series evidence | `list_tasks`, `profile_series`, `characterize_series`, `data_quality` | Retrieve standard task contracts; summarize time-series characteristics; identify patterns, phases, and change-points; and assess or clean data quality. |
| **TSFM** | Model discovery and feasibility | `list_models`, `search_models`, `find_models`, `describe_candidates`, `describe_models`, `count_models`, `list_domains`, `get_model_lineage`, `resolve_model`, `hf_stats` | Discover, filter, inspect, count, and validate model cards; trace model lineage; and retrieve Hugging Face popularity statistics. |
| **TSFM** | Model authoring and lifecycle | `model_template`, `register_model`, `register_finetuned`, `update_model`, `deprecate_model`, `new_model_version` | Retrieve the model-card schema; register standard or fine-tuned models; update, deprecate, and create successor model versions. |
| **TSFM** | Feature catalog discovery | `list_features`, `search_features`, `get_feature`, `get_feature_lineage`, `count_features`, `describe_features` | Discover and inspect transform and extractor cards, retrieve feature lineage, and summarize the feature catalog. |
| **TSFM** | Feature authoring and lifecycle | `register_feature`, `update_feature`, `deprecate_feature`, `new_feature_version` | Register validated feature transforms, update feature metadata, deprecate feature cards, and create successor versions. |
| **TSFM** | Feature engineering | `extract_features`, `select_features` | Compute scalar feature values and rank candidate extractors using self-supervised forecasting performance. |
| **TSFM** | Recipe composition and evaluation | `recipe_template`, `run_recipe`, `run_tabular_recipe`, `run_plan`, `evaluate` | Retrieve recipe specifications; execute time-series or tabular analytical recipes; execute recipe DAGs; and evaluate configurations across datasets. |
| **TSFM** | Run and result ledger | `list_runs`, `get_run`, `list_results`, `get_result` | Retrieve persisted execution records and task-specific analytical results. |
| **Vibration** | Vibration data access | `get_vibration_data`, `list_vibration_sensors` | Load vibration time-series data into the analysis store and list available vibration sensors for an asset. |
| **Vibration** | Frequency-domain analysis | `compute_fft_spectrum`, `compute_envelope_spectrum` | Compute FFT spectra and envelope spectra for rotating-equipment and bearing-fault analysis. |
| **Vibration** | Severity assessment | `assess_vibration_severity` | Classify vibration severity according to ISO 10816 zones. |
| **Vibration** | Bearing knowledge | `calculate_bearing_frequencies`, `list_known_bearings` | Calculate bearing characteristic frequencies and retrieve built-in bearing specifications. |
| **Vibration** | Automated diagnosis | `diagnose_vibration` | Produce an integrated vibration diagnosis using spectral evidence, bearing-envelope analysis, severity assessment, fault classification, and a report. |