Eval engineering: closing the gap between AI testing and AI monitoring

Atos point of view - November 2026



Executive summary

AI agents don't give the same answer twice, yet most enterprises still test them as if they did. Eval Engineering turns AI quality from a subjective judgment into a repeatable, measurable engineering discipline. Atos is already evaluating agents differently on its own Eval Engineering platform. This paper explains the method, how we run it and why we think it should become the industry standard.

In a 2026 Microsoft benchmark of 507 business tasks, each run 20 times, the strongest model completed 66.5% of tasks on a single try, but only 47.5% on all 20 tries (Li et al., 2026). A LangChain survey of 1,340 practitioners, run in late 2025, found that 89% of teams monitor their agents in some form, while only 37% evaluate them in production (LangChain, 2026).

A benchmark measures what a model can do on a standardized task. An evaluation measures whether your agent does your job reliably. The two are not the same, and confusing them is how teams put an agent into production without knowing if their agent works. QA tools check that code meets its specification before release. Observability tools show that a system is running, how fast and at what cost. Neither answers the question that matters most for an AI agent: Is it behaving as intended, often enough, across the real range of requests it receives?

We call the practice that answers that question **Eval Engineering**, and the person who owns it the **Eval Engineer**.

The Atos Eval Engineering platform puts the practice to work. It scores agent behavior on up to six core dimensions plus any custom dimensions a given agent needs offline before release and online against a statistically sound sample of live runs. It connects to the QA and observability tools enterprises already use, so evaluation links the two instead of becoming one more silo.

In short, it gives teams the evidence to decide when an agent can go live and how far to trust it once it's running. It works with evaluation platforms and security tools, not in place of them.

We close with **seven principles**, a **four-level maturity model** any organization can use for assessment, and how **Atos is putting this into practice today.**



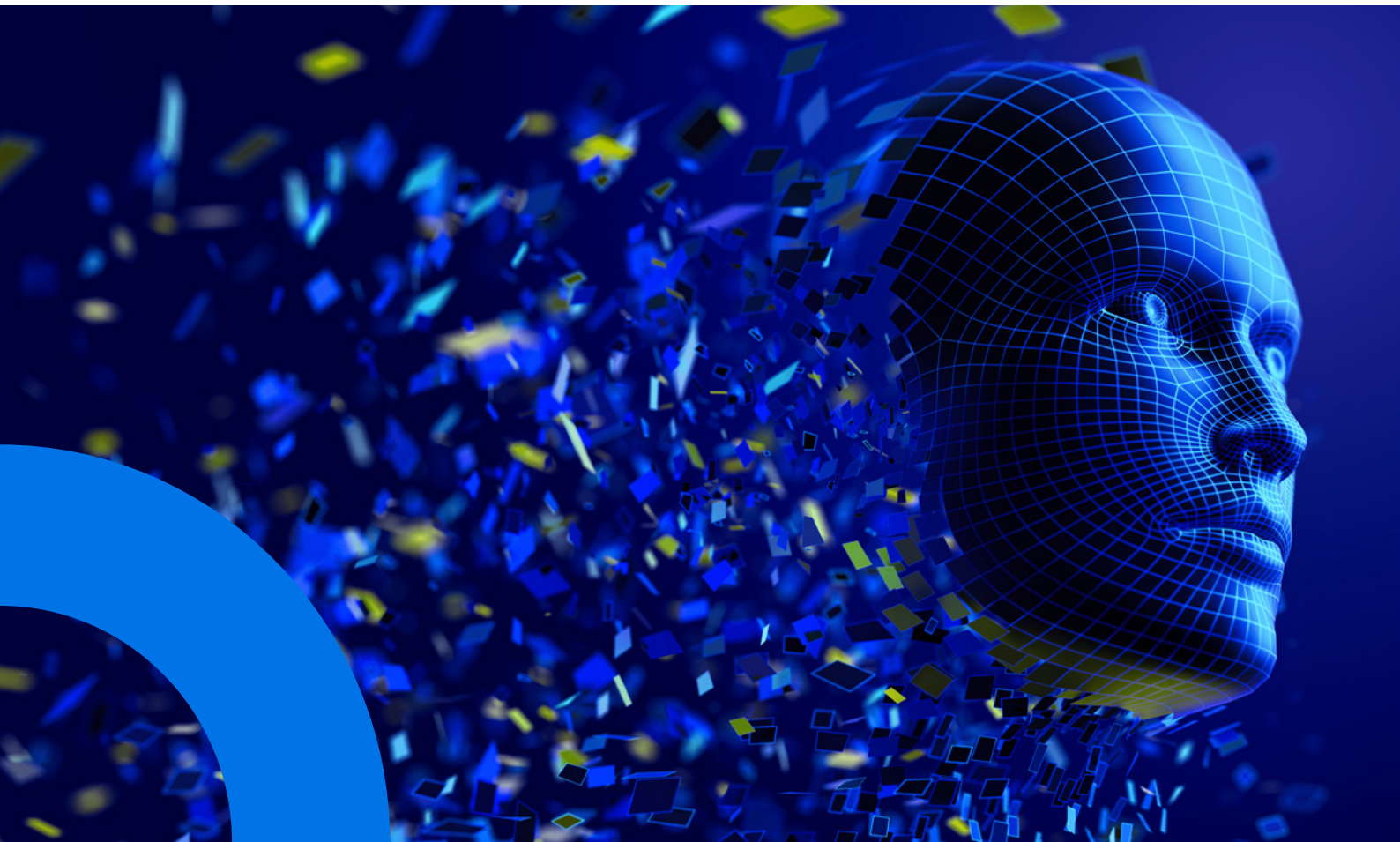
The problem: software that doesn't repeat itself

Traditional software is designed to produce the same output every time it receives the same input. That makes testing relatively predictable: a test that passes today should pass tomorrow. AI systems built on large language models don't follow that rule. Their responses can vary even when teams try to turn off randomness.

- **Turning the dial to zero does not fix it**. Researchers ran five models set to their most deterministic settings on eight tasks, ten runs each, and found large swings in accuracy between identical runs (Atil et al., 2024).
- **Code is no exception**. When the same coding request was sent repeatedly to ChatGPT, the generated code did not give the same test results across requests on 76% of tasks in the CodeContests benchmark, and on about half the tasks in two other benchmarks (Ouyang et al., 2025).
- **Agents multiply the effect**. Each step's output becomes the next step's input, so small differences grow. A Princeton study of 15 models, presented at ICML 2026, found that models have become far more capable over the past two years but only slightly more reliable (Rabanser et al., 2026). In Microsoft's 2026 benchmark, one leading model fell from 57% success on a single try to under 18% across 20 tries of the same task (Li et al., 2026).

The business cost is already visible. In Moffatt v. Air Canada, a Canadian tribunal held the airline liable for wrong fare advice from its website chatbot and rejected the argument that the chatbot was responsible for its own words (Moffatt v. Air Canada, 2024 BCCRT 149). Gartner expects more than 40% of agentic AI projects to be canceled by the end of 2027, citing rising costs, unclear value and weak risk controls (Gartner, 2025). Practitioners point to the same weak spot: in the LangChain survey, quality was the most cited barrier to production, named by 32% of respondents, ahead of cost (LangChain, 2026).

> **One passing run shows an agent can do the job. It does not show how often it will.**



The gap between testing and monitoring

Enterprises already own two mature toolsets, and each answers a different question well. Neither answers the behavior question.

QA asks: does the code meet the spec?

Test management and automation tools run fixed cases and expect a fixed result. That model works when the same input always gives the same output. Point it at an agent and a test passes on Monday and fails on Tuesday with no code change. Teams then either loosen the checks until they mean little or stop trusting the suite.

Observability asks: is the system up, fast and affordable?

Tracing tools record every model call, tool call, token and millisecond. That record is essential, but it describes what happened, not whether it was right. A trace can look healthy while the agent quietly gives a customer the wrong refund policy.

Dedicated evaluation platforms have emerged to fill the gap. LangSmith, Langfuse, Arize or Braintrust all record what an agent does and surface it for review. They are necessary, yet they are not sufficient. Every one of them requires the team to define what good looks like (test cases, graders, thresholds, failure criteria) before the platform has anything to measure.

Eval Engineering is way of working and an operating model, not just a product or a tool. It includes the method, the people and the decision on whether an AI system or agent ships or not.



Behavior evaluation is the missing link between testing and monitoring

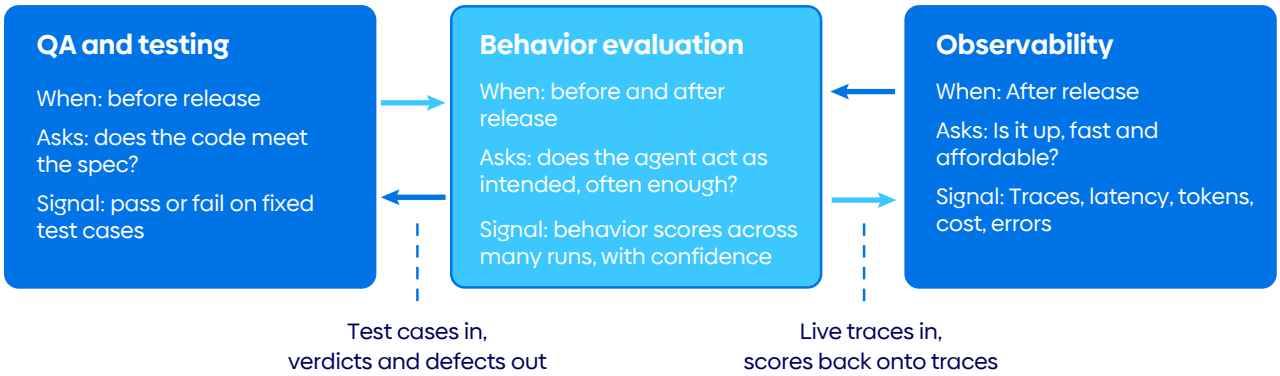


Figure 1. Where behavior evaluation sits between QA and observability

QA works before release on fixed cases; observability works after release on health signals. Behavior evaluation spans both and is the only layer that asks whether the agent did the right thing.

The survey results show where effort has gone. 89% of respondents have observability in place, but only 52% run offline evaluations and 37% evaluate live traffic. 29.5% evaluate nothing at all (LangChain, 2026).

89% of teams watch their agents; 37% judge them in production

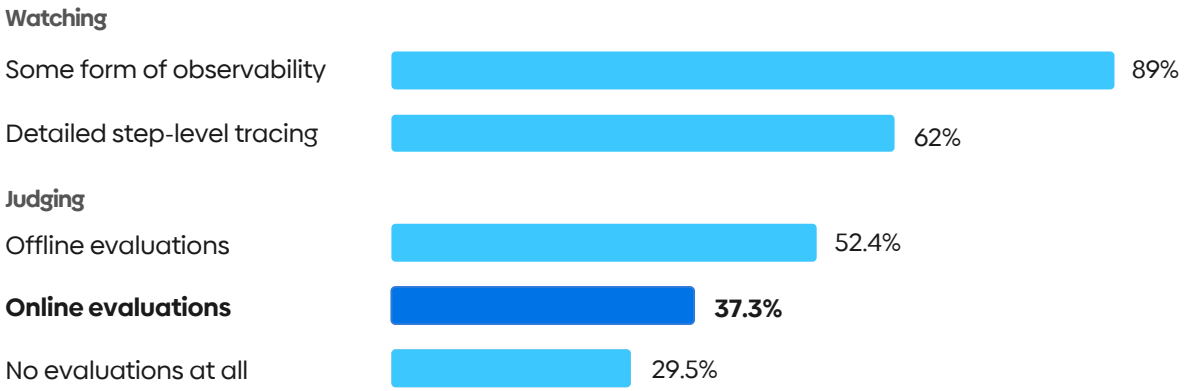


Figure 2. How agent teams evaluate today. Source: LangChain, State of Agent Engineering (2026), survey of 1,340 practitioners, November 18 to December 2, 2025

Among teams that already have agents in production, online evaluation rises to 44.8%, which still leaves more than half of production teams with no live evaluation.

Regulators are pointing the same way. The EU AI Act requires providers of high-risk AI systems to collect and analyze performance data across the system's whole lifetime, not just at launch (EU AI Act, Article 72). Not every agent will count as high-risk, but the expectation of ongoing evidence is now set in law. In the United States, NIST's Generative AI Profile names 12 risks specific to generative AI and lists suggested actions to manage them, many of them involving testing and evaluation (NIST AI 600-1, 2024). Both point to the same need: a continuous, evidence-backed view of how an AI system behaves.



Eval Engineering: the discipline and the role

> **Eval Engineering is the practice of defining how an AI system should behave, measuring how it behaves, and acting on the difference, before and after release.**

It treats behavior as a distribution to be measured, not a single result to be checked.

The **Eval Engineer** owns that practice for a portfolio of agents. The role sits next to the QA engineer and the site reliability engineer and borrows from both.

Role	Main question	Works with	Produces
QA engineer	Does the code meet the spec?	Fixed test cases	Pass or fail per release
Site reliability engineer	Is the service up, fast and affordable?	Metrics, logs, traces	Service levels and alerts
Eval engineer	**Does the agent act as intended, often enough?**	**Many runs of the same task, scored**	**Behavior scores with confidence, release gates, audit evidence**

In practice the Eval Engineer:

1. Writes the agent's behavior spec with the business owner: what it must do, what it must never do and how good is good enough.
2. Builds and maintains the test set, including real cases, adversarial cases and simulated users.
3. Chooses the right grader for each check and tests the graders against human judgment.
4. Sets release thresholds and the number of runs needed to trust a result.
5. Monitors live performance, investigates signs of model drift, and turns production failures that expose gaps in test coverage into new golden datasets.
6. Hands risk and compliance teams the evidence they need, traceable to a specific agent version.

The skill set is part software tester, part data analyst and part domain expert. Most organizations already have people doing these functions. What they lack is a name for the job, a clear mandate and a platform built for it.



What we measure, and how

We score every agent on up to six core dimensions, add custom dimensions where the agent needs them, use three kinds of graders and run enough repeats to trust the answer. Each dimension has its own threshold, not an average. A single number hides too much. A customer service agent can be accurate and still unsafe, or safe and still useless.

Six core dimensions, plus custom ones

Dimension	The question it answers	How we score it
Accuracy	Is the answer or end state correct for this case?	Compare the output or final system state with the expected result
Completeness	Did it handle every part of the request?	Checklist of required elements, scored by a model judge
Consistency	Does it give the same outcome when the task repeats?	Run each case several times; report how often all runs succeed
Groundedness	Is every claim backed by a source it was given?	Match each claim to retrieved content; flag anything unsupported
Compliance	Did it follow regulatory rules, business rules and its own instructions?	Rule checks on the full sequence of actions
Safety (behavioral)	Did it avoid harmful output, data leaks and manipulation?	Red-team suites, prompt-injection tests, content classifiers

These six dimensions fit a technical agent. The set is agreed upon per agent, and teams can add custom dimensions to it. An agent that takes multi-step actions might add trajectory: Did it use the right tools, with the right inputs, in a sensible order? An agent covered by the EU AI Act's high-risk rules might add bias and fairness: Did it treat comparable users the same way? A customer service agent and an IT support agent are not judged on the same measures.

Safety here means a set of behavior checks: harmful output, data leaks in answers and known prompt-injection cases. It's not a security assessment, as threat modeling, specialist AI security testing and runtime controls stay with the security team. The same goes for compliance: scores give risk, compliance and security teams evidence; they don't replace their sign-off.

Cost, latency and token use come from observability tools and sit beside these scores. A change that lifts quality by two points but doubles cost per task is a business decision. The Eval Engineer makes that trade-off visible.

Three kinds of graders

- **Code checks** for anything that can be checked exactly, like the end state of a record, the arguments of a tool call or a forbidden phrase. They're fast, cheap and repeatable, so we use them wherever we can.
- **Model judges** for anything that needs reading, including completeness, tone and whether an answer is supported. Strong model judges agree with human experts more than 80% of the time, about as often as experts agree with each other **(Zheng et al., 2023)**. The same study found judges favor the first answer shown, longer answers and their own answers. We correct this by swapping answer order, scoring against written rubrics, and using a different model family to judge.
- **Human review** for the cases that matter most and to keep the judges honest. A sample of every judge's score is checked by people for each cycle. If agreement drops, the judge is re-tuned before its scores count.

Statistics, not anecdotes

Every offline test case is run multiple times to assess both success and reliability. We report the overall pass rate with a confidence range, then use pass^k to measure how often an agent succeeds on all k attempts at the same task. Introduced in τ-bench, a benchmark for evaluating AI agents on realistic, multi-step tasks involving tools and rules, this measure reveals whether an agent can perform reliably rather than succeed by chance on a single run (Yao et al., 2024). A release is approved only when the lower end of the pass-rate confidence range exceeds the agreed threshold. Online, we score a sample of live runs rather than all of them. The sample size follows from the error the business will accept. For example, to estimate a pass rate near 95% to within two percentage points, with 95% confidence, we need about 460 scored runs per period. Smaller samples are faster and cheaper, but the range widens and drift takes longer to spot.

Offline and online: one loop, one spec

Offline and online evaluation use the same behavior spec and the same graders, so a score means the same thing before and after release. Most teams run the two as separate projects with separate tools. The result is a test suite that drifts away from what real users ask, and production monitoring that has no definition of “correct”.

Offline evaluation runs before release, in the delivery pipeline. It uses a curated dataset: real cases from the business, adversarial cases designed to break the agent, and simulated users who push back, change their minds, or leave out details. Every case runs several times, and the results feed the release gate.

Online evaluation runs after release, on live traffic. The platform samples runs from production traces, removes personal data, scores them against the same spec, and compares each period with the agreed baseline. A model update from a vendor, a change to a prompt or a shift in what customers ask can all move the scores without any change to the agent’s code.

The two halves join in one place: every online failure that reveals an uncovered scenario becomes a new offline test case. Over time the test set comes to look like real demand, and the release gate gets harder to pass for the right reasons.



Every uncovered scenario found live becomes a new test case

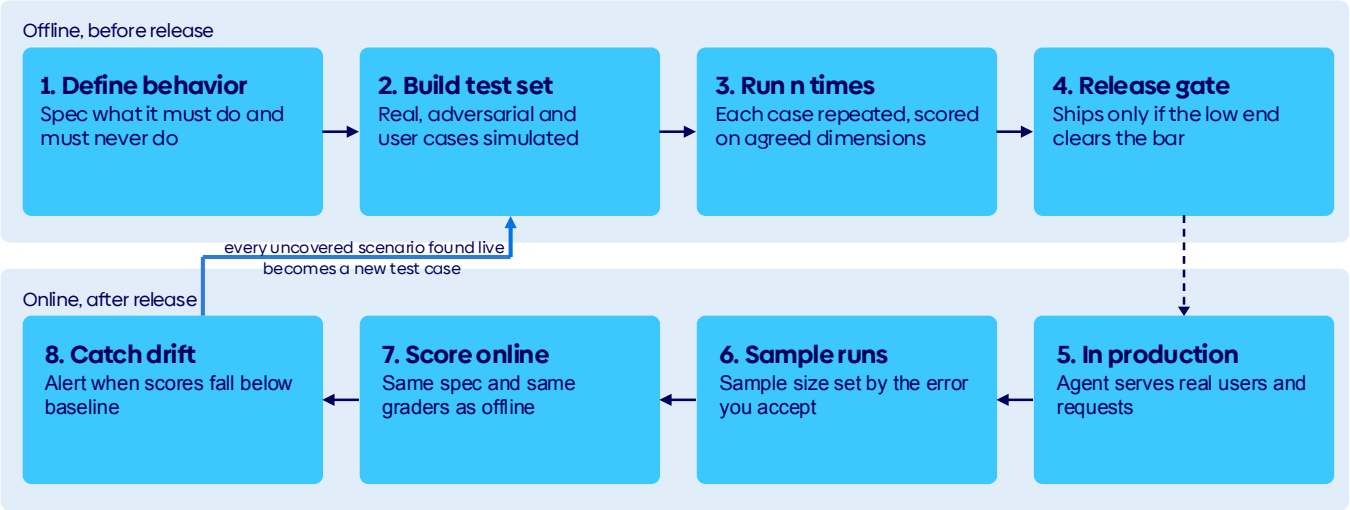


Figure 3. The evaluation loop: eight steps across offline and online

In Figure 3 above, the top lane runs in the delivery pipeline; the bottom lane runs on live traffic. The blue line is what most teams are missing.

Failure analysis & recommendations: Flagging a failure is the easy part. The engineering value is in root-cause: the same score decline can be caused by either a prompt problem, a model regression, a tool miscall, or a spec that was wrong to begin with. Each has a different fix.

> **Evaluation that stops at 'fail' is reporting. Evaluation that produces a cause and a recommendation is engineering.**

Working with the tools clients already run

The platform does not replace QA or observability tools. It reads from both, judges behavior, and writes the results back where each team already looks. Clients have invested heavily in these toolsets, and their teams know them. A new silo would be ignored.

On the **quality and delivery** side, the platform takes test cases and run requests from test management and CI/CD and returns verdicts. A failed behavior check shows up as a defect in Jira or Azure DevOps and as a red stage in the pipeline, exactly where a failed unit test would.

On the **observability** side, the platform uses the OpenTelemetry format for generative AI to capture information about model calls, tool use and agent activity in a consistent way. It also integrates with Langfuse, Arize and Braintrust to support evaluation and monitoring while agents are running. Crucially, evaluation scores are linked to the same traces engineers already use to investigate system behavior. An engineer examining a slow agent run in Datadog or Dynatrace, for example, can also see that the run received a low groundedness score, indicating that its response may not have been adequately supported by the available evidence.

Inside the platform, teams can use our own evaluators alongside a range of open-source tools. Frameworks such as Ragas, DeepEval, promptfoo and Inspect, as well as red-teaming tools such as PyRIT and garak, can feed their results into the same process for scoring, storage and release decisions. This reflects Atos's open-components-first approach to AI solutions and platforms. Clients can choose the tools that fit their needs, reduce dependence on any single vendor and deploy the platform in environments that meet regional regulatory requirements. Findings from security testing can go into the same evidence record. Runtime guardrails and security controls stay with the client's security team.

The platform sits between the tools teams already run

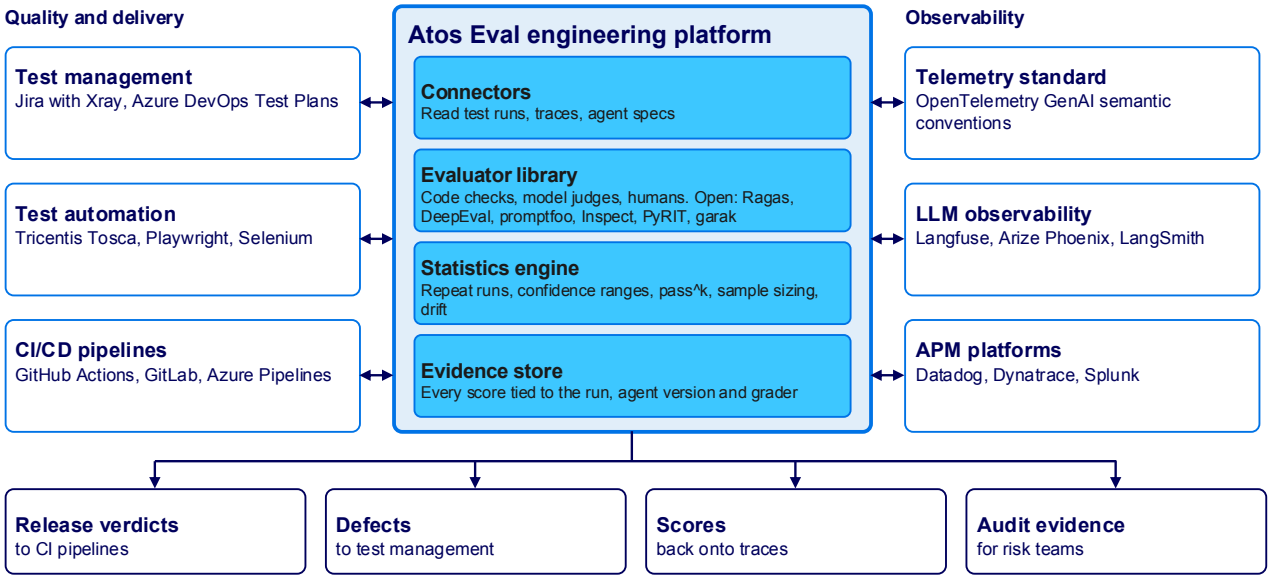


Figure 4. Integration architecture (named tools are typical examples)

Data flows both ways on each side: test runs and traces come in; verdicts and scores go back to the tools people already watch.

The result is one record per agent version: what it was asked to do, how it behaved, which grader said so and whether it was allowed to ship.

What problems remain

Four problems are still open. The first is one the industry must solve together.

Reading the agent's configuration. To judge an agent against its spec, the platform needs to know what the agent is: its instructions, tools, model and version. Agent platforms with an API make this easy. But in client environments and on third-party platforms, we see a growing share of agents defined in plain markdown files and run through command-line tools such as GitHub Copilot or Claude Code, which offer no configuration API. For these agents, we read the definition from the source repository and tie every score to a commit. The lasting fix is a common, machine-readable agent manifest that any framework can publish and any evaluator can read.

Judges can be wrong. Model judges carry known biases (Zheng et al., 2023). Human review and regular calibration reduce the risk; they do not remove it. Scores from model judges are labeled as such in every report.

Repeated runs cost money. Running every case several times, on every change, is not cheap. We split suites into a fast smoke set for every commit and a full set before release, and we size online samples to the precision the business needs.

Some answers have no single right answer. Fairness, tone and ethical judgment depend on context and values. For these dimensions the spec must be written with the business and, where relevant, risk, legal and ethics teams. The platform can measure against a standard; it cannot set one.

The standard we propose

Seven principles should apply to any AI system that acts on a business's behalf. None depends on our platform. All of them can be checked by an auditor.

1. **Build a golden dataset before you measure**. Real cases, adversarial cases, and simulated-user cases, reviewed with the business owner as the agent and its failure modes evolve.
2. **Write down expected behavior before you build.** Every agent has a versioned spec: what it must do, what it must never do, and the threshold for each.
3. **Never trust a single run.** Report rates with a confidence range, and a consistency score for anything that takes actions.
4. **Use the cheapest grader that works and test the graders.** Code checks first, model judges where reading is needed, people to calibrate both.
5. **Evaluate in production continuously**. Size the live sample by the error you can accept, not by what is convenient.
6. **Close the loop.** Every production failure that reveals an uncovered scenario becomes a data set before the next run.
7. **Keep the evidence.** Every score traces back to the run, the agent version and the grader that produced it.

A passing eval is not a guarantee. It is the absence of a known reason to block the release and a starting point for deciding where the agent can operate. Organizations should use the result to set the agent's operating boundaries: where it can run unsupervised, where a human should stay in the loop and which failure modes warrant a higher threshold before the next release.

Where does your organization stand?

Level	What it looks like	What is usually missing
1. Ad hoc	People try the agent by hand before launch	Any repeatable test
2. Repeatable	A fixed test set runs in the pipeline, once per case	Repeat runs and confidence ranges
3. Statistical	Cases run many times; release gates use confidence ranges; judges are calibrated	Live evaluation
4. Continuous	Live runs are sampled and scored; drift raises alerts; failures feed the test set; evidence is audit-ready	Usually nothing structural; the work is keeping specs current

The survey data suggests most organizations sit at level 1 or 2 today, since only about half run offline evaluations and fewer than four in ten evaluate live traffic (LangChain, 2026). Moving to level 3 is mostly a change in method. Level 4 needs a platform that connects testing and monitoring, which is what Atos has built and is running.

Some agents warrant Level 4 as a minimum: anything customer-facing, consequential, regulated, or irreversible. Internal advisory tools with human oversight can operate at Level 2 without exposing the organization to disproportionate risk. Maturity targets should follow the agent's risk profile, not a uniform enterprise mandate.

Sovereign evaluation: a design element, not a feature

For organizations in regulated industries, the question of where evaluation runs is not a preference, it is a constraint. Evaluation requires production traces: the actual inputs, outputs, and tool calls of live agents. Scoring those traces on an external platform or through a third-party model judge means production data leaves the client's environment. In financial services, healthcare, defense, and government, that is not acceptable.

Data residency requirements are tightening. The EU AI Act, DORA in financial services, and sector-specific frameworks in healthcare and defense all point the same way. Atos's position is that sovereign evaluation must be a design constraint from the start: evaluation platform, model judges, dashboards, and audit evidence all run in the client's environment. No production trace crosses the boundary.

This is architecturally harder than the default and carries infrastructure cost. We do not propose it for every deployment. We propose it as the only viable approach for regulated industries, and we are building it as a first principle rather than retrofitting it as a premium feature.

How Atos is putting this into practice

Atos runs Eval Engineering against its own AI systems. We are our own "client zero," applying the method to our agents before bringing it to clients.

We start with agents inside Atos Intelligent Quality Engineer, Autonomous Development Lifecycle and other Atos solutions. This lets us test the approach on our own systems and build practical experience before applying it in client environments.

For clients, Atos brings three elements together:

- **The Atos Eval Engineering platform,** which connects to existing QA and observability tools and integrates behavioral evaluation into the delivery pipeline and production monitoring.
- **Eval Engineers,** who work with client teams to define expected behavior, build golden datasets, select and calibrate graders, and set release thresholds.
- **A maturity assessment** against the four levels described in this paper, showing where current practices fall short and what is needed to reach continuous evaluation.

The first step is straightforward: select an agent already in production, assess it against the seven principles, and identify the gaps. That gives the organization a practical basis for deciding what to improve and where to focus investment.



What sets Atos apart?

Most evaluation platforms can run repeated tests, plug into CI/CD pipelines and link scores to traces. Ours does too. But those capabilities are now the norm. What matters is how evaluation works in a client's environment, how well it reflects their business and what teams can actually do with the results.

Here's where Atos makes a difference:

- **Built for clients' environments from day one.** The Atos Eval Engineering platform runs in a client's own cloud tenant or data center, using their encryption keys. Production traces, evaluation scores and audit evidence stay within their security perimeter. Sovereignty is part of the original design, not something we added later.
- **Works across agent platforms.** Whether an agent was built using Microsoft Copilot, Salesforce, an Atos solution or a custom LLM framework, we evaluate it against the same engineering standards. Clients aren't tied to one vendor's ecosystem or evaluation tools.
- **Tests what matters to the business.** We work with clients' business owners to define expected behavior and build the golden datasets around their actual processes, industry requirements and business rules. Generic benchmarks have their place, but they can't tell a bank whether its loan servicing agent is following the bank's policies.
- **Goes beyond flagging failures.** A failed evaluation should help engineers understand what went wrong. Our platform identifies likely causes and recommends changes, giving teams something useful to act on rather than another dashboard full of red scores.
- **Keeps the evidence regulators need.** Evaluation histories, flagged runs and human review outcomes are recorded and available for export. This helps clients demonstrate how they've tested and monitored agent behavior, supporting documentation, risk management and record-keeping requirements under the EU AI Act.
- **Backed by engineers who know what to look for.** Our Eval Engineers work alongside client teams to set meaningful thresholds, investigate failures and update test datasets as agents evolve. The platform does the measuring. People make the judgment calls.

There's also an important boundary to keep in mind. Eval Engineering measures how agents behave and whether they're ready for release. It doesn't replace threat modeling, specialist security testing or runtime controls. Those remain with security teams, while guardian agents, where deployed, monitor behavior during operation. The findings can feed into the same evidence record, giving clients a clearer picture of agent performance and risk without confusing the responsibilities of each discipline.

Conclusion: Eval Engineering, the Atos Way

Atos's position is that every AI agent operating in a business environment needs a defined standard of behavior and a way to measure whether it meets that standard. Evaluation must continue after release.

Models change over time, and shifts in usage can expose weaknesses that testing missed. Left undetected, these failures can undermine service quality, disrupt business processes and erode customer trust.

Evaluation evidence helps enterprises set appropriate operating boundaries for each agent and determine when human oversight is necessary. It also provides a basis for deciding when an agent is reliable enough to take on greater responsibility.

Atos has the platform and engineering expertise to put this discipline into practice–both internally and for our clients. By measuring agent behavior throughout its lifecycle, organizations can reduce the risk of failures in production and make better-grounded decisions about where AI can deliver dependable business value. This enables them to expand AI adoption with greater confidence while maintaining the control that enterprise operations demand.

References

Atil, B., et al. (2024). Non-Determinism of "Deterministic" LLM Settings. arXiv:2408.04667. arxiv.org/abs/2408.04667

European Union. (2024). Regulation (EU) 2024/1689 (AI Act), Article 72: Post-market monitoring by providers. EU AI Act Service Desk

Gartner. (2025, June 25). Gartner Predicts Over 40% of Agentic AI Projects Will Be Canceled by End of 2027. Press release

LangChain. (2026). State of Agent Engineering. Survey of 1,340 respondents, November 18 to December 2, 2025. langchain.com/state-of-agent-engineering

Li, Z., Ko, Y., Keramati, A., et al. (2026). One Success Isn't Reliability: Thinkingbox, a Sandbox and Benchmark for Agents in Stateful Business Workflows. Microsoft. arXiv:2608.19741. arxiv.org/abs/2608.19741

Moffatt v. Air Canada, 2024 BCCRT 149. British Columbia Civil Resolution Tribunal, February 14, 2024. CanLII

National Institute of Standards and Technology. (2024, July). Artificial Intelligence Risk Management Framework: Generative Artificial Intelligence Profile (NIST AI 600-1). doi.org/10.6028/NIST.AI.600-1

OpenTelemetry. Semantic Conventions for GenAI agent and framework spans. Status: development. opentelemetry.io

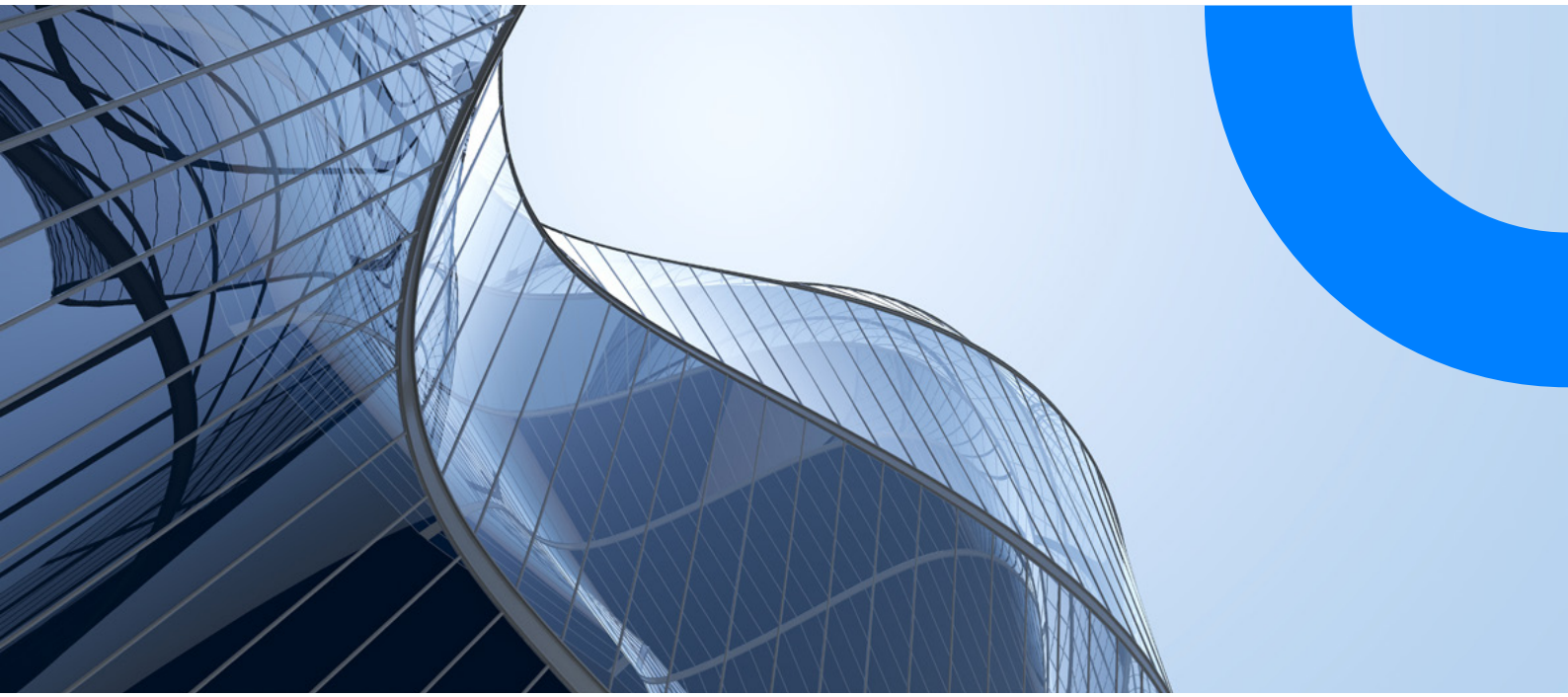
Ouyang, S., Zhang, J. M., Harman, M., & Wang, M. (2025). An Empirical Study of the Non-determinism of ChatGPT in Code Generation. ACM Transactions on Software Engineering and Methodology. arXiv:2308.02828v2. arxiv.org/abs/2308.02828

Rabanser, S., Kapoor, S., Kirgis, P., Liu, K., Utpala, S., & Narayanan, A. (2026). Towards a Science of AI Agent Reliability. Princeton University. ICML 2026. arXiv:2602.16666. arxiv.org/abs/2602.16666

Yao, S., Shinn, N., Razavi, P., & Narasimhan, K. (2024). t-bench: A Benchmark for Tool-Agent-User Interaction in Real-World Domains. arXiv:2406.12045; ICLR 2025. arxiv.org/abs/2406.12045

Zheng, L., et al. (2023). Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. NeurIPS 2023 Datasets and Benchmarks. arxiv.org/abs/2306.05685

Note: Tool and product names in the section Working with the tools clients already run belong to their owners and are listed...



About Atos

Atos is the Atos Group's brand dedicated to end-to-end, secure and AI-accelerated digital services. Atos designs, develops, and operates critical digital environments that are critical to performance, resilience and sovereignty. The company helps public and private organizations around the world maintain control of their data and infrastructures, while meeting their compliance requirements.

With more than 51,000 employees serving over 4,500 clients across 54 countries, Atos supports the modernization of IT systems, accelerates cloud and data transformation, strengthens cybersecurity, and deploys secure digital workplace for the benefit of its clients, their employees and society. Atos also offers consulting and advisory services through its Atos Amplify brand.

As a trusted partner in the management of complex and critical environments, Atos supports organizations in highly regulated and sovereign contexts.

Find out more about us
atos.ai

Let's start a discussion together

About Atos Group

Atos Group is a global leader in digital transformation with c. 53,000 employees and annual revenues of c. €7.2 billion. With a commercial presence in 54 countries, it operates under two brands - Atos for services and Eviden for products and systems. As the European number one in cybersecurity and a leader in the cloud sector, Atos Group is committed to a secure and decarbonized future. It offers tailor-made and integrated solutions, accelerated by AI, for all sectors of activity. Atos Group is listed on Euronext Paris.

Atos Group is a registered trademark. This document, or any part of it, may not be reproduced, copied, circulated and/or distributed nor quoted without prior written approval from Atos Group.