Market landscape

ar001 · 20 June 2026

The finding

This is our prior-art map. Two populations test agents, and we care about both: the **agent builders** from the field catalogued below, where our *customers* are, and the **eval/testing frameworks**, the dev-tool layer where a real *competitor* would emerge. We asked every one the same question: when a score moves between versions, can it tell a **real regression from LLM run-to-run noise**?

The builders can't, they score once. The frameworks are a generation ahead (several now re-run tests N times) but stop at the same line: they hand you the runs and a mean, and leave you to eyeball it. A noise-aware red/green that gates CI is exactly testpath's wedge. Teardowns below, first ten builders (nine plus **Ada** as a managed contrast), then eight frameworks. Features verified from each vendor's docs, Jun 2026.

The platform universe

The full field of AI support-agent platforms, split **builder** (the customer owns the agent and its quality, *their* customers are our targets) vs. **managed** (the vendor owns quality, mostly noise). The *why* is in the lead-gen approach (ar002); each builder's own testing is torn down in the sections that follow.

Platform	Type	Customers (est.)	Stage	Website	What it is
Botpress	Builder	≈ thousands	Series B	botpress.com	Developer-friendly agent builder (open core)
Voiceflow	Builder	≈ thousands	Series A	voiceflow.com	Visual, no-code → pro-code agent builder
Rasa	Builder	≈ hundreds	Series C	rasa.com	Open-source, self-hosted conversational AI
Cognigy	Builder	≈ hundreds	Acquired (NICE)	cognigy.com	Enterprise conversational AI build platform
Kore.ai	Builder	≈ 400	Series D + PE	kore.ai	Enterprise AI agent build platform
Parlant	Builder	≈ dozens	Seed	parlant.io	Open-source agent framework

Platform	Type	Customers (est.)	Stage	Website	What it is
Inkeep	Builder	≈ dozens	Seed	inkeep.com	AI support/ copilot built from your docs
Salesforce Agentforce	Builder	≈ thousands	Public (Salesforce)	salesforce.com	Build agents inside the Salesforce platform
Yellow.ai	Builder	≈ 1,000	Series C	yellow.ai	Multichannel conversational AI build platform
Ada	Managed	≈ hundreds	Series C	ada.cx	Autonomous no-code AI customer service
Sierra	Managed	≈ dozens	Series E	sierra.ai	Managed conversational AI agents
Decagon	Managed	≈ dozens	Series D	decagon.ai	Managed AI support agents
Intercom Fin	Managed	≈ thousands	Private (Intercom)	intercom.com	Turnkey AI support agent inside Intercom
Forethought	Managed	≈ hundreds	Acquired (Zendesk)	forethought.ai	Managed AI customer support
Gradient Labs	Managed	≈ dozens	Series A	gradient-labs.ai	Managed AI support agent
Lorikeet	Managed	≈ dozens	Series A	lorikeet.ai	Managed AI support agent for complex CX
Tidio Lyro	Managed	≈ 10k+	Series B (Tidio)	tidio.com	Turnkey SMB AI support agent
Zendesk AI	Managed	≈ thousands+	Private (PE)	zendesk.com	Helpdesk-native turnkey AI agents
ASAPP	Managed	≈ hundreds	Series C	asapp.com	Managed enterprise CX AI

Platform	Type	Customers (est.)	Stage	Website	What it is
eesel AI	Managed	≈ hundreds	Seed	eesel.ai	Turnkey AI support over your existing tools

***Stage** is verified via web research (Jun 2026). Type, customer counts, and websites remain first-pass.*

Botpress

- **Offers:** Visual Studio has only a manual emulator + (paid) analytics, no saved tests. The code-first ADK shipped **Evals** (Mar 2026): declarative tests with response/tool/state assertions, an LLM-judge, and CLI/CI.
- **Philosophy:** Split, visual users iterate by hand + watch dashboards; the dev ADK adopts a real test-driven loop.
- **Gap:** ADK evals run **once**, single pass/fail, no repeats, variance, or noise handling. The popular visual product has no regression testing at all.

Voiceflow

- **Offers:** The most built-out, in-canvas Test Tool, native **Tests** (turns + AI simulation; response/routing/tool-call checks, parallel), **Evaluations** (LLM-judge on every transcript), agent-to-agent goal tests, a CLI, and a dev→staging→prod pipeline.
- **Philosophy:** Transcript-driven observability, capture every conversation, auto-score it, jump back to the canvas.
- **Gap:** Docs literally say "AI responses may vary… focus on goal achievement," with **no** re-run/stability mechanism. Silent regressions rely on brittle exact-match or coarse trend-watching.

Rasa

- **Offers:** Gold-standard **deterministic** testing, `rasa test nlu/core` (F1, confusion matrices, cross-validation), CALM E2E tests with ≈11 assertions + two LLM-judge ("relevant"/"grounded") checks, CI-first.
- **Philosophy:** Conversation-Driven Development, turn real conversations into tests; testing as a first-class engineering discipline.
- **Gap:** Rock-solid for classic NLU; **thin for the stochastic LLM layer**, no non-determinism, multi-run, or variance handling.

Cognigy

- **Offers: Playbooks** (exact 1:1-match regression scripts) + a newer LLM-judge **Simulator** (synthetic conversations, success-rate scoring, scheduled runs) + Insights analytics.
- **Philosophy:** Enterprise analytics-driven optimization, deterministic scripts plus a synthetic "performance lab."
- **Gap:** Playbooks break on stochastic output (official advice: mock with static data); the Simulator only watches aggregate rates, no per-case version diffing to catch silent regressions.

Kore.ai

- **Offers:** Mature **Batch Testing** (NLU accuracy) + **Conversation Testing** (assertions, flow regression), plus a newer "Evaluation Studio" (agent/tool evaluators, LLM-judge, adversarial/persona sims, observability).
- **Philosophy:** Continuous train → test → tune lifecycle across technical/quality/safety/business metrics.
- **Gap:** Battle-tested tooling is deterministic NLU; the newer agent-eval *names* "LLM change management" but ships no documented noise-tolerant regression mechanism.

Parlant

- **Offers:** No agent regression harness at all, only a **design-time guideline linter** (coherence checker for contradictory rules) plus runtime reasoning (ARQ) and traces. It once shipped a `parlant-test` entry point and *removed* it.
- **Philosophy:** Misalignment is a structural design problem, fix behavior by editing guidelines, not by running test suites.
- **Gap:** Nothing for users on stochastic output: no multi-run, variance, golden conversations, or version comparison. Tellingly, Emcie's **own CI** uses a `pytest-stochastics` majority-vote plugin, they understand the problem, they just don't ship it to customers.

Inkeep

- **Offers:** A genuine eval layer, custom LLM-judge **evaluators**, **datasets/test suites** (CSV or SDK), batch + **online (sampled) evals**, retroactive scoring of past conversations, full CI API.
- **Philosophy:** Programmatic, eval-driven + feedback-driven, codify judges, run offline and on live traffic.
- **Gap:** Every item scored **once**; "reruns" compare point scores, so a 1–2-point judge swing from noise is indistinguishable from a real regression, left entirely to the user.

Salesforce Agentforce

- **Offers: Testing Center** (sandbox), AI/synthetic test-case generation, batch/parallel testing (Topic/Action/Response pass %), a metadata **Testing API**, and `sf agent test` CI/CD (JUnit/TAP/JSON).
- **Philosophy:** "Agentic Lifecycle Management", test in sandbox to an "acceptable accuracy," deploy, monitor.
- **Gap:** Single-pass semantic pass/fail, no native multi-run, variance, or pass-rate thresholds; hallucinations can pass, and practitioners report **hand-rolling "3+ run averaging."** No test coverage is required before production.

Yellow.ai

- **Offers:** "Agentic testing" (Evaluation + multi-turn Simulation), LLM-judge accuracy/empathy thresholds, auto test-gen from KB/sessions, regression checks across sandbox→prod, plus legacy NLU tests + Insights analytics.
- **Philosophy:** Automated QA "built for AI's probabilistic nature", parallel LLM-judged scenarios + a self-learning production loop.
- **Gap:** Still single-run threshold pass/fail, no repeats, variance, or version-diff/drift detection. "Real regression or noise?" goes unaddressed.

Ada — managed, the contrast

- **Offers:** Interactive testing, **Simulations at scale** (LLM-judge pass/fail vs expected outcomes), plus in-production **Automated Resolutions** scoring, a Reviewer Model, and a coaching loop.
- **Philosophy:** Managed, but Ada *preaches* customer ownership, while supplying all the eval machinery itself.
- **Gap (why weaker targets):** Eval is **baked into the turnkey stack**, so customers feel covered and the felt pain is low, even though Ada's own docs concede "variability is expected between runs." This is the managed pattern: their customers ≈ noise for us.

Eval & testing frameworks. The builders are where our customers sit; these dev-tool frameworks are where a real competitor would come from, and they're a generation ahead, several shipping multi-run *repeats*. Same question, tougher test. Eight below, ordered roughly worst → closest on the wedge.

OpenAI Evals

- **Offers:** Two things, an open-source benchmark registry (string-match + model-graded "LLM-as-judge" templates, a completion-function hook for agents) and a hosted platform **Evals** product (graders, `pass_threshold`, run comparison, CI). The platform is **deprecated**: read-only 2026-10-31, shutdown 2026-11-30, users steered to promptfoo.
- **Philosophy:** Highest-number-is-best benchmarking. The OSS repo is now janitorial; the platform was task-oriented prompt iteration.
- **Gap:** No per-sample repeats (`--seed` *suppresses* randomness). Its one statistic, `bootstrap_std`, resamples *across the dataset*, modelling sampling error, not run-to-run LLM variance, and the platform reports no variance/CI at all. No significance test; the official regression cookbook literally tells you to observe the bad run "has a score much lower than the baseline." A dead foil, differentiate against the live tools.

Ragas

- **Offers:** Reference-free LLM-judged RAG/agent metrics (faithfulness, context precision/recall, tool-call & goal accuracy), knowledge-graph synthetic test-set generation, an `evaluate()` API, and CSV "Experiments." CI is DIY (wrap in pytest, assert on thresholds).
- **Philosophy:** A metrics *library*, not a platform, decompose quality into LLM-judge metrics and trust prompt engineering, not statistics, to make them reliable.
- **Gap:** Runs each sample **once**, `evaluate()` has no repeats/seed, returns point estimates, no variance or CI. Docs concede metrics are "somewhat non-deterministic" but ship no remedy; practitioners bolt on bootstrapped CIs by hand. "Real or noise?" is entirely the user's problem.

Langfuse

- **Offers:** Datasets + dataset-run experiments (UI/SDK/CI), LLM-judge & code evaluators, and a side-by-side run comparison with score/cost/latency deltas and threshold filters, all atop its core product, production tracing.
- **Philosophy:** Observability-first; offline eval is bolted onto the tracing spine, human-in-the-loop interpretation over automated gating.
- **Gap:** No repeats parameter, no variance/CI, no significance test, comparison is mean-vs-mean with eyeballed deltas. The tell: when users hand-roll repeats, the compare view doesn't even average across traces correctly (bug #4541, open since Dec 2024 through the Apr 2026 experiments rebuild). Stochasticity is left to the user.

DeepEval

- **Offers:** "Pytest for LLMs", `deepeval test run` with pytest-style assertions, 50+ metrics (G-Eval and other LLM-judges), datasets/goldens, CI/CD, baseline ("official") runs, and the Confident AI cloud for run comparison. A `-r` flag repeats each case.
- **Philosophy:** Fold evals into CI/CD; every metric has a fixed threshold, a case passes if its score clears it, regression = a drop below threshold vs a baseline.
- **Gap:** Noise-blind. It admits G-Eval is "**NOT** deterministic" (its probability-weighted score stabilizes *one* call, not across runs); `-r` reruns as independent pass/fails with no averaging, variance, or CI; version comparison is pure threshold pass/fail, no significance test. The fix for judge variance ("run several and average") is left to you.

promptfoo

- **Offers:** Declarative YAML evals (deterministic + LLM-judge assertions), datasets, model/prompt comparison, red-teaming, strong CI/CD. Regression is a static gate: `--repeat N`, a global `PROMPTFOO_PASS_RATE_THRESHOLD` (default 100%), exit-code failure.
- **Philosophy:** Prompts as unit tests; its prescribed answer to non-determinism is to *suppress* it (`temperature: 0`, semantic graders), with `--repeat` offered mainly to surface "flickering" tests for a human to eyeball.
- **Gap:** `--repeat N` stores each run but computes no variance, stddev, or CI, and comparison stays point-vs-point with no significance test. No flaky quarantine, no per-test pass-rate, a $0.92 \rightarrow 0.88$ move is reported as a raw delta. The community asked for exactly this (issue #1932); it's still **open**. It has the data primitive, not the inference layer.

Arize Phoenix

- **Offers:** OTel tracing + an evals library (LLM-judge/code), datasets/experiments with a `repetitions` param (client v1.20.0, Sep 2025), a "Compare Experiments" diff UI, online evals on live traffic, and DIY CI gating on a mean-score threshold.
- **Philosophy:** Production-observability-first; experiments/CI are a secondary, assemble-it-yourself developer feature.
- **Gap:** `repetitions` genuinely re-runs each example (docs cite LLM stochasticity as the reason), then aggregates by **averaging only**: no variance, SEM, or CI anywhere, and comparison is a visual "improved/regressed" diff with no test. Its marketing line "is it real, significant, or just noise?" maps to no shipped feature. Hands you N samples, keeps the mean, leaves the statistics to you.

LangSmith

- **Offers:** Datasets, off-the-shelf + custom (LLM-judge) evaluators, experiments, a multi-experiment **Comparison View** with green/red regressed highlighting, pairwise evals, first-class **pytest/Vitest integrations** (Jan 2025) and CI gating on metric thresholds. `num_repetitions` runs each example N times.
- **Philosophy:** Observability-and-experimentation first, trace everything, build datasets from production traffic, treat regression as a visual diff against a baseline.
- **Gap:** The closest to surfacing the ingredients, `num_repetitions` reports per-score **average *and* stddev**, but it stops there. Comparison and CI gate on point scores with no significance test, CI, or flaky mechanism; a $0.84 \rightarrow 0.81$ drop shades red whether real or jitter. Its own CI guidance even tells you to hand-tune thresholds *below* the mean to dodge variance false-fails. Ingredients present; inference left to the human.

Braintrust

- **Offers:** A serious eval platform, immutable comparable experiments, datasets, code/LLM-judge scorers, a comparison UI with diff mode and "order by regressions," CI/CD, online scoring, and **trials** (`trialCount`, 3–5 recommended) that re-run each input and bucket the results.
- **Philosophy:** Evals-as-engineering-discipline, every run an immutable artifact, gate CI on score changes, diff what moved.
- **Gap:** The **closest competitor, worth watching.** Trials + variance capture are first-class ("measure variance, get more robust scores"). But the *significance test* is the soft spot: product docs describe only mean aggregation and a "regression" sort that flags any negative delta, the phrase "statistical significance… real or noise" appears only in **marketing copy**, with no method, p-value, or CI in the technical docs. Until that's reproducible in the UI, noise-aware gating on top of their trials data stays differentiated.

The takeaway

Two populations, one blind spot, at different depths.

The **builders** ship real testing (emulators, judges, analytics) and zero stochasticity handling: they score a test once. The tell is teams hand-rolling multi-run averaging (Agentforce) and a vendor running a stochastic-vote suite internally while shipping none of it to customers (Parlant).

The **frameworks** are a generation ahead. Five of the eight now re-run tests N times, promptfoo `--repeat`, DeepEval `-r`, Phoenix `repetitions`, LangSmith `num_repetitions`, Braintrust `trials`. So "run it many times" is becoming table stakes, not a moat. But every one stops at the same line: they collect the runs, hand you a **mean**, and leave you to eyeball whether a drop is real. None ship the inference layer, variance → confidence interval → significance test → a noise-aware CI verdict. The evidence this is the live gap: promptfoo's flickering-test request (#1932) sits open, Langfuse's multi-run averaging is a year-old bug, LangSmith's own CI guidance tells you to hand-tune thresholds *under* the mean to dodge false fails, and Braintrust's "real or noise?" claim lives in marketing copy, not the docs.

So the wedge sharpens. testpath isn't "we repeat and they don't", repeats are commoditizing. It's the **statistics on top of the repeats**: turn N runs into a sound real-or-noise red/green that gates CI. **Braintrust is the one to watch** (trials and variance already; a real significance test would narrow the lane); the rest leave it wide open.

And the method is proven, not speculative, frontier eval work already does this: Anthropic's *Adding Error Bars to Evals* (paired differences, clustered standard errors) and the UK AI Safety Institute's **Inspect** (`epochs` + standard errors). Nobody has packaged it for the **regression-CI loop** the builders' customers actually run. That makes testpath an **execution/productization** play on a solved statistical problem, lower science risk, higher "ship it well" bar.

Two strategic reads survive intact. **Chase builders' customers, not managed services':** Ada bakes eval into the turnkey stack, so its customers stop feeling the pain (their docs even concede "variability is expected between runs"). And because testpath *complements* existing eval rather than replacing it, every builder, and every framework, is also a potential **integration** partner.