

Your AI Agent Got Lucky

Why Evals Alone Can't Protect
Regulated Industries

ABC — Agentic Behavioral Contracts

Behavioral validation toolkit for AI agents

gitlab.aws.dev/brouwem/agentic-behavioral-contracts

Apache 2.0 • 67 tests • 3 frameworks validated

The Setup

You’ve built an AI agent that processes insurance claims. It calls the right tools, cites the right policies, and produces correct decisions. Your eval suite gives it a 98% pass rate. You deploy it to production.

Six months later, a regulator asks: “How do you know the AI followed the correct process for every decision?”

You pull up your eval results. The regulator isn’t satisfied.

“I can see the outputs were correct. But did the agent actually check the formulary database? Did it verify the member’s coverage before deciding? Or did it just guess from its training data and happen to be right?”

You don’t know. Your evals checked the destination, not the journey.

The Problem No One Is Solving

The AI agent reliability space has three established layers:

Guardrails (NeMo, Guardrails AI, Galileo) — “Is this output safe?”

Observability (Langfuse, LangSmith, Arize) — “What did the agent do?”

Eval frameworks (DeepEval, RAGAS, Braintrust) — “Is the output correct?”

None of them answer the question regulators actually ask: “Did the agent do the right thing for the right reasons?”

A correct output reached through the wrong process is still a compliance violation. An agent that approves a valid claim by guessing from training data — without checking the formulary — produces the right answer through the wrong process. Every eval says PASS. Every regulator says FAIL.

The Lucky Quadrant

We tested the same pharmacy claim through two agents:

Agent A had tools and followed the process: formulary lookup coverage check decision.

Agent B had no tools and answered from its training knowledge.

Both produced correct, professional, citation-rich responses. Both approved the claim. We ran both through two evals:

Eval	Agent A (with tools)	Agent B (no tools)
Deterministic	PASS (4/4)	PASS (4/4)
LLM Judge	PASS (5/5)	PASS (4/5)

Both evals were fooled.

Agent B cited “Section 3.2.1” — a fabricated policy section. The real one is 4.2.1. But the citation looks real, and no eval checked it against the actual formulary.

Then we ran behavioral validation — checking the OTEL traces:

Check	Agent A	Agent B
Data sources consulted	formulary-db, coverage-policy	(none)
Tools called	lookup, check, render	(none)
Behavioral verdict	PASS (6/6)	FAIL (2/6)

The 2×2 Confidence Matrix

	Behavioral PASS	Behavioral FAIL
Eval PASS	TRUSTWORTHY	LUCKY
Eval FAIL	DEBUGGABLE	BROKEN

The LUCKY quadrant is invisible to evals alone. The agent got the right answer by accident — from training data, not from the formulary. Next time, with a different drug, it might not be so lucky.

Validated Across 3 Frameworks

Same adapter, same contract, same tool_source_map:

Framework	Instrumentation	Result
Strands Agents	Native OTEL	6/6 PASS
LangChain/LangGraph	otel-instrumentation-langchain	6/6 PASS
PydanticAI	InstrumentationSettings	6/6 PASS
Strands on AgentCore	ADOT auto-instrumentation	6/6 PASS

Also deployed to Amazon Bedrock AgentCore, pulled OTEL traces from CloudWatch, and validated 10 production traces — all passing.

Three Regulated Domains Tested

- Claims Processing (HCLS) — formulary lookup, coverage check, policy citations
- KYC/AML Screening (FinTech) — sanctions lists, jurisdiction rules, escalation
- Clinical Decision Support (HCLS) — drug interactions, guidelines, clinician review

Production-Ready

```
agent-validate run --contract claims-processing-v1 \
  --traces "traces/*.json" --fail-on-violation --output report.json
```

- CLI — discover, dry-run, run, list-contracts (CI/CD ready)

- Production monitoring — CloudWatch + directory watcher with compliance rate
- Compliance tracking — sliding window drift detection with per-constraint thresholds
- Audit persistence — individual records + batch reports to files or S3
- Dynamic contracts — conditional constraints that adapt based on context

Your evals tell you the output is correct. Behavioral validation tells you it's correct for the right reasons. In regulated industries, that's the difference between confidence and compliance.