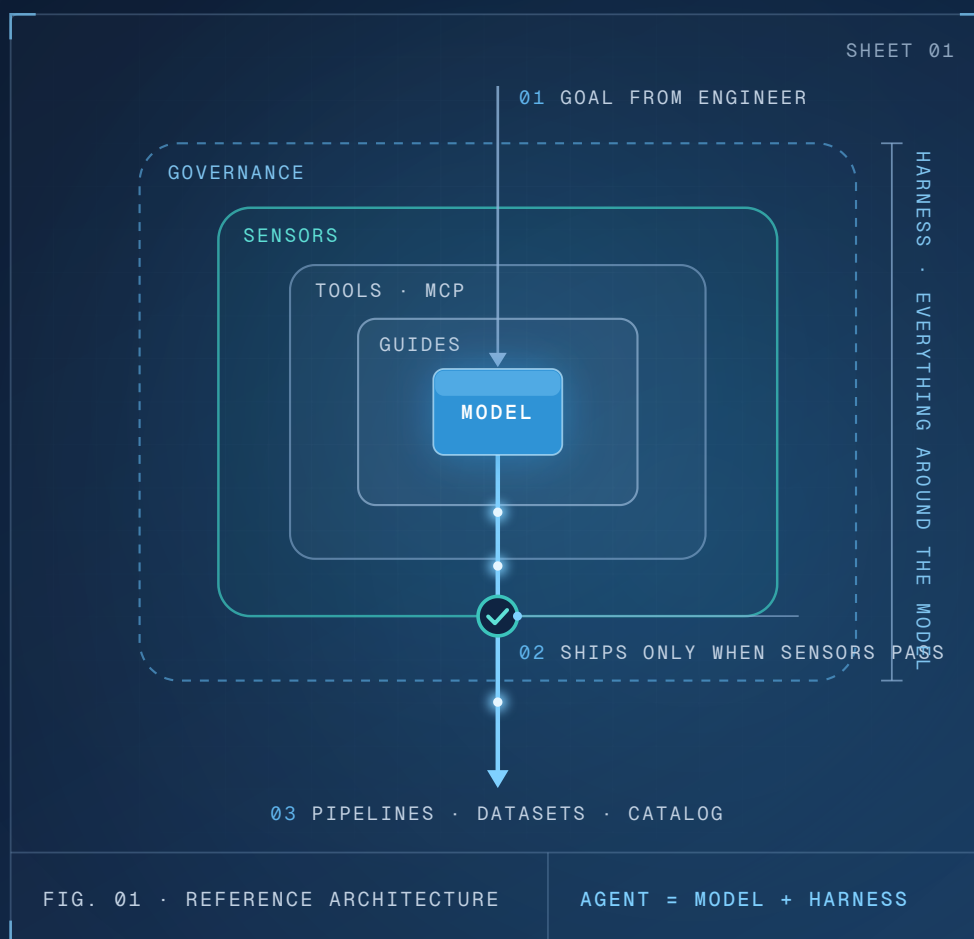


WHITE PAPER

FREE DOWNLOAD

# Agentic Harness Architecture in 2026

Components, Vendors, and a Reference Design for Data Engineering



**Zach Gardner**

Chief Architect, Keyhole Software



## CONTENTS

# Inside this paper

|    |                                                                             |    |
|----|-----------------------------------------------------------------------------|----|
| 01 | Key Findings                                                                | 3  |
| 02 | The Pattern at a Glance                                                     | 4  |
| 03 | What Is an Agentic Harness? The Components Behind the Model                 | 5  |
| 04 | The Harness Market in 2026: Tools, Adoption, and Consulting Offerings       | 8  |
| 05 | A Reference Architecture: One Generic Harness, Extended to Data Engineering | 12 |
| 06 | Measured Results: What Agents in a Harness Actually Deliver                 | 21 |
| 07 | AI Agent Risks and Governance: The Controls a Harness Must Enforce          | 23 |
| 08 | A Phased Roadmap for Agentic AI Adoption                                    | 25 |
| 09 | Recommendations                                                             | 28 |
| 10 | How Keyhole Can Help                                                        | 29 |
| 11 | How We Researched This                                                      | 30 |
| 12 | References                                                                  | 31 |

# Agentic Harness Architecture in 2026

Components, Vendors, and a Reference Design for Data Engineering



**Zach Gardner**

Chief Architect, Keyhole Software

In 2014, I wrote that Docker was the first true DevOps tool. I'm making a similar call about the agentic harness: it's the part of an AI agent that decides whether the agent is ready for production. It's also a lot less shiny than the model, which is probably why it gets so little attention.

Over the past year, the AI labs, the framework vendors, and the practitioners have settled on a simple formula for what surrounds the model, Agent = Model + Harness, where the harness is everything around the model (the loop, the tools, the context, the permissions, and the checks)<sup>1,2</sup>. It matters more than most teams expect. With the model held fixed, changing only the harness took one agent's Terminal-Bench 2.0 score from 52.8% to 66.5%<sup>3</sup>. The same model's cost per attempt varied by up to 5x depending on which harness ran it<sup>4</sup>.

That's a problem for teams trying to get agents past the pilot stage. Gartner expects more than 40% of agentic AI projects to be canceled by the end of 2027<sup>5</sup>, and expects half of agent deployment failures by 2030 to come down to weak runtime governance<sup>6</sup>. HFS Research, which ranks the firms selling agentic services, put it more bluntly: maturity is limited "more by trust, governance, and liability models than by LLMs or tooling"<sup>7</sup>. For most teams, the bottleneck isn't the model. It's the harness.

This paper is for CTOs, data leaders, and architects deciding how to run agents inside the engineering and governance controls they already have. It covers:

- What a harness is made of, and how much it changes cost and results
- Who's selling what, from open-source harnesses to the agent platforms of large consulting firms
- When a data platform's built-in agent is enough, and when you need your own harness
- A reference architecture for one shared harness, including a walkthrough of a data engineering request from start to finish
- The risks, and the controls I'd want in place before handing an agent the keys to anything that matters
- A phased adoption roadmap with a measurable gate at every step

We didn't try to rank vendors. Our research team analyzed more than 103 sources, most published between December 2024 and October 2026. That includes engineering write-ups from Anthropic, OpenAI, Google, LangChain, and Thoughtworks; documentation for 15 harnesses and agent frameworks; announcements from 14 consulting and IT services firms; product documentation for agentic data engineering tools from Google, Databricks, Snowflake, dbt Labs, Alation, and Datadog; and survey data from Stack Overflow, DORA, JetBrains, McKinsey, KPMG, and Gartner. We paired that with what we've learned running governed agentic delivery inside actual client engagements.

# Key Findings

01

## The harness moves cost more than accuracy.

Across the studies we reviewed, the same model cost up to 5x more per attempt<sup>4</sup> and used up to 70 times more tokens per solved task<sup>22</sup> depending on the harness around it.

02

## The harness layer is standardizing faster than the models.

MCP, Agent Skills, and AGENTS.md are now open, multi-vendor formats. Building on them lets teams swap models or tools without rewriting their institutional knowledge<sup>13,14</sup>.

03

## The consulting market has moved from building agents to governing them.

In about a year, launches shifted from agent builders to control planes, registries, and agent identities<sup>40,52,58</sup>. Nearly all published results are self-reported.

04

## Productivity gains are real but conditional.

One large study found pull requests up 98% and review time up 91%, with no significant company-level improvement<sup>81</sup>. Gains hold up only when the surrounding system verifies the work.

05

## Governance is lagging adoption.

53% of large U.S. companies have deployed agents, but only 36% have direct token or usage controls<sup>98</sup>.

06

## The documented failures share one cause.

In every 2025 incident we examined, an agent held standing privilege over a destructive action, and no deterministic control stood in its way<sup>85</sup>.

WORK WITH KEYHOLE

## See this architecture mapped to your stack

In an architecture and roadmap session, Keyhole architects map the reference harness to your platforms, governance requirements, and first data engineering use case.

[Schedule an architecture session](#)

# The Pattern at a Glance

Before getting into components and vendors, here's the agentic harness architecture the rest of this paper builds on:

- An engineer sets the goal.
- The harness surrounds the model with guides (what it should know), tools (what it's allowed to touch), and sensors (what has to pass before the work counts as done).
- Results land through the systems the organization already runs: CI, the data platform, and the catalog.
- And because every session runs through the same governed harness, one engineer can direct several sessions at once without giving up review or control.

For a shorter take on why the harness matters more than the model, see our quick overview [Agentic Harness vs. Model](#).

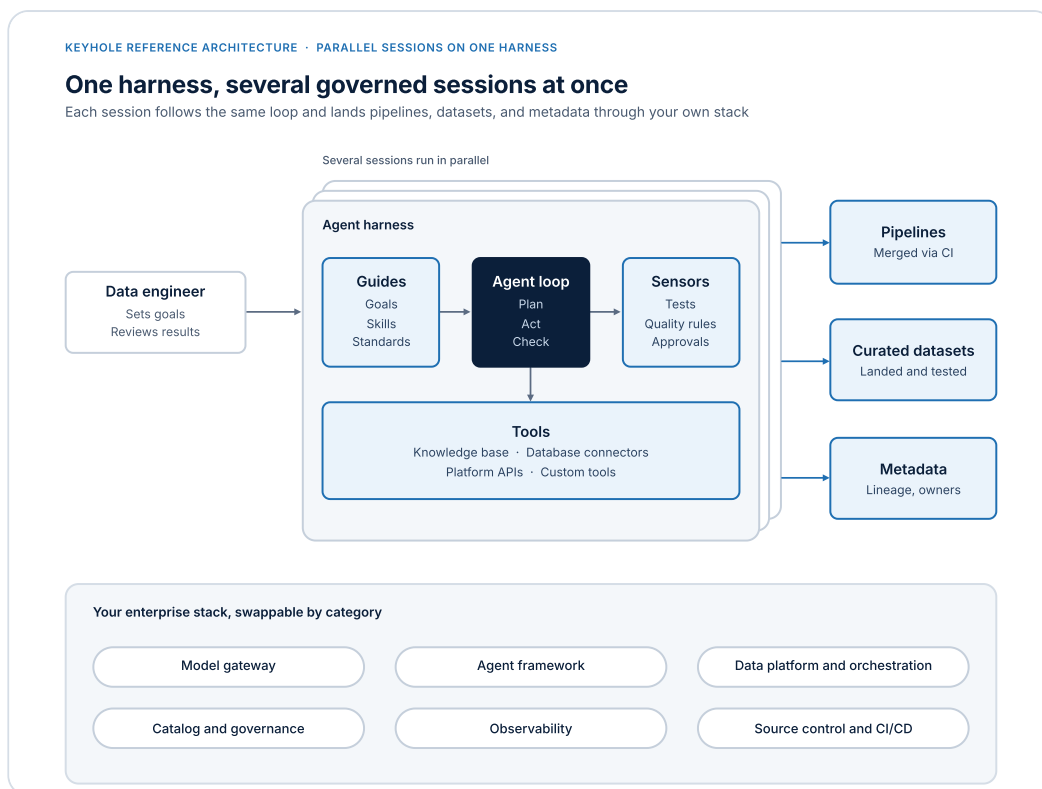


FIGURE 1

## Keyhole Software · the agentic harness pattern for data engineering

One engineer directs several harness sessions at once. Goals enter on the left. Work moves through guides, the agent loop, and verification, and only verified results reach pipelines, datasets, and metadata on the right. The bottom row is your existing stack, and each category can be swapped without changing the harness. The reference architecture section extends this same pattern to data engineering.

# What Is an Agentic Harness? The Components Behind the Model

An agentic harness is the software around an AI model that runs its loop, controls which tools it can use, manages its context, and checks its work before anything ships.

I find Simon Willison's definition the most useful: an agent "runs tools in a loop to achieve a goal"<sup>8</sup>. The harness is everything that makes that loop safe and productive. OpenAI describes its Codex harness as "the core agent loop and execution logic that underlies all Codex experiences"<sup>9</sup>, and LangChain goes broader: "every piece of code, configuration, and execution logic that isn't the model itself"<sup>1</sup>. The table below breaks those pieces down across the sources we reviewed.

## Components of an Agentic Harness, 2026

| Component                            | What it does                                                                                                  | Guide or sensor | Published examples                                                                                        |
|--------------------------------------|---------------------------------------------------------------------------------------------------------------|-----------------|-----------------------------------------------------------------------------------------------------------|
| <b>Agent loop and runtime</b>        | Calls the model, executes the tool calls it requests, appends the results, and repeats until the task is done | Core            | Codex harness <sup>9</sup> ; Claude Agent SDK <sup>10</sup> ; Ralph loop <sup>11</sup>                    |
| <b>Instructions and memory files</b> | Persistent project rules and pointers the agent reads every session                                           | Guide           | AGENTS.md, CLAUDE.md; OpenAI kept its AGENTS.md to about 100 lines as a "table of contents" <sup>12</sup> |
| <b>Skills</b>                        | Folders of procedural know-how, loaded only when relevant (progressive disclosure)                            | Guide           | Agent Skills open standard, supported by 40+ tools <sup>13</sup>                                          |
| <b>Tools and connectors</b>          | Let the agent read and act on systems: files, shells, databases, APIs                                         | Both            | Model Context Protocol: 10,000+ public servers, 97M+ monthly SDK downloads <sup>14</sup>                  |
| <b>Context management</b>            | Keeps the working context small and relevant: compaction, notes, just-in-time retrieval                       | Core            | Anthropic's context engineering guidance on "context rot" <sup>15</sup>                                   |
| <b>Orchestration and sub-agents</b>  | Splits work across a lead agent and specialized sub-agents                                                    | Core            | Anthropic's multi-agent research system <sup>16</sup>                                                     |
| <b>Sandbox and permissions</b>       | Decides which actions run automatically, which need approval, and where code executes                         | Guide           | GitHub Copilot's cloud agent runs in an ephemeral Actions environment with a firewall <sup>17</sup>       |

| Component                            | What it does                                                                                | Guide or sensor | Published examples                                                                                  |
|--------------------------------------|---------------------------------------------------------------------------------------------|-----------------|-----------------------------------------------------------------------------------------------------|
| <b>Hooks and middleware</b>          | Deterministic code at fixed points in the loop: lint before commit, stop runaway loops      | Sensor          | LangChain loop detection and pre-completion checklist <sup>3</sup>                                  |
| <b>Session state and checkpoints</b> | A durable log of what happened, kept outside the model's context window                     | Core            | Anthropic Managed Agents "session" log <sup>18</sup>                                                |
| <b>Verification</b>                  | Tests, linters, and type checks (computational) or AI review and LLM-as-judge (inferential) | Sensor          | Böckeler's guides-and-sensors taxonomy <sup>2</sup>                                                 |
| <b>Observability</b>                 | Traces every model call, tool call, token, and cost                                         | Sensor          | OpenTelemetry in Microsoft Agent Framework <sup>19</sup> and Amazon Bedrock AgentCore <sup>20</sup> |

Anthropic adds a useful caution: a harness "encodes assumptions about what Claude can't do on its own," and those assumptions "go stale as models improve"<sup>18</sup>. Expect some of these components to become unnecessary as models improve, and plan to remove them when they do.

## How Much Does the Harness Matter? The Benchmark Evidence

If you want proof that the harness matters, look at the studies that hold the model fixed and change only what's around it. Practitioners have started calling this work **harness engineering**.

TABLE 2

### Measured Effect of the Harness With the Model Held Constant

| Study                                         | Setup                                                         | Finding                                                                                                                                                                 |
|-----------------------------------------------|---------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>LangChain Deep Agents, February 2026</b>   | GPT-5.2-Codex on Terminal-Bench 2.0; only the harness changed | Score rose from 52.8% to 66.5%, moving from the top 30 to the top 5 <sup>3</sup>                                                                                        |
| <b>Terminal-Bench 2.0 paper, January 2026</b> | Same models run in different agent scaffolds                  | Gemini 2.5 Pro resolved 17% more tasks in one scaffold than another <sup>21</sup>                                                                                       |
| <b>HarnessTax (Arena), September 2026</b>     | 7 models across Claude Code, Codex CLI, and Pi                | Success rates were similar, but cost varied by up to 5x; a harness other than the model vendor's own achieved the best success rate in 9 of 12 comparisons <sup>4</sup> |
| <b>The New Stack analysis, August 2026</b>    | Aider, Claude Code, and OpenClaw on an identical model        | Tokens per solved task varied 70-fold <sup>22</sup>                                                                                                                     |

The Terminal-Bench authors still conclude that "model selection is usually more important than agent scaffold when optimizing for performance"<sup>21</sup>, and I agree with them. The LangChain result shows a harness can move success rates, but across these studies the larger and more consistent effect is on cost.

For a team planning a rollout, that cost spread changes the deployment decision. A workflow that's affordable for one developer can become prohibitively expensive across dozens of concurrent jobs, and the HarnessTax study found the same model costing up to 5x more depending on the harness<sup>4</sup>. That's why I'd measure cost per successfully completed task, alongside reliability, before expanding usage.

I was also struck by how much faster the harness layer is standardizing than the model layer. Tools (MCP), procedural knowledge (Agent Skills), and project instructions (AGENTS.md) are now open, multi-vendor formats, either under the Linux Foundation's Agentic AI Foundation or published as open standards<sup>13,14</sup>. Build on those and you can swap models or harnesses without rewriting your institutional knowledge.

#### IN PRACTICE · FROM KEYHOLE ENGAGEMENTS

### Same Model, Different Harness

You feel the harness in day-to-day work long before you see it in a benchmark. One of our developers spent two weeks running OpenAI's model inside the open-source OpenCode harness and described it as "the same brain with a much nicer terminal harness around it." Seeing context size and token use on screen changed how he managed each session, and being able to revert to an earlier decision point made it cheap to abandon a wrong implementation path.

Our own delivery harness is deliberately simple. In a reference run we published, we used [the Ralph loop](#) (a persistent execution pattern introduced by Geoffrey Huntley) to work through 19 dependency-ordered user stories with 1:1 story-to-test-to-commit traceability and test-gated progress at every iteration<sup>23</sup>. Ralph is a pretty dumb loop, and I mean that as a compliment. The guides around it (stories, architecture rules) and the sensors (tests, commits) are what make it trustworthy. I walk through the same pattern in [my architect's workflow video](#), including planning mode, constraints-first prompting, and multi-agent workflows inside a governed SDLC<sup>24</sup>. One rule from that video bears repeating: I do not let an agent run unrestricted git commands.

# The Harness Market in 2026: Tools, Adoption, and Consulting Offerings

The market splits into two camps: ready-to-use harnesses that engineers run directly, and SDKs and runtimes for building your own. Open-source, model-agnostic options are now mainstream. The first-party harnesses from the model vendors tend (unsurprisingly) to favor their own models.

## Agentic Harnesses Compared: Claude Code, Codex, Cursor, and More, 2026

| Harness                                          | Vendor                 | License                                              | Model choice                                                | Distinguishing harness features                                                                                                                  |
|--------------------------------------------------|------------------------|------------------------------------------------------|-------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Claude Code</b>                               | Anthropic              | Proprietary                                          | Claude models via Anthropic, Bedrock, Vertex AI, or Foundry | Hooks, sub-agents, MCP, permission modes, skills, plugins, resumable sessions <sup>10</sup>                                                      |
| <b>Claude Agent SDK and Managed Agents</b>       | Anthropic              | Anthropic commercial terms; hosted service           | Claude                                                      | The Claude Code loop, programmable in Python and TypeScript; Managed Agents hosts the loop in a managed or self-hosted sandbox <sup>10, 18</sup> |
| <b>Codex CLI</b>                                 | OpenAI                 | Apache-2.0                                           | Primarily OpenAI; configurable providers                    | Local sandboxed execution, approvals, MCP, AGENTS.md, skills; CLI, IDE, desktop, and cloud surfaces <sup>25</sup>                                |
| <b>OpenAI Agents SDK</b>                         | OpenAI                 | MIT                                                  | 100+ models                                                 | Handoffs, guardrails, human in the loop, sessions, tracing <sup>26</sup>                                                                         |
| <b>Antigravity CLI (successor to Gemini CLI)</b> | Google                 | Gemini CLI is Apache-2.0; Antigravity CLI not stated | Gemini                                                      | Shares one agent harness with the Antigravity 2.0 desktop app; Gemini CLI stopped serving consumer and free users on June 18, 2026 <sup>27</sup> |
| <b>Agent Development Kit 2.0</b>                 | Google                 | Apache-2.0                                           | Model-agnostic, Gemini-optimized                            | Graph workflows, human in the loop, built-in evaluation <sup>28</sup>                                                                            |
| <b>Copilot cloud agent</b>                       | GitHub                 | Proprietary                                          | Multiple                                                    | Ephemeral GitHub Actions environment, firewall, 59-minute maximum run; 1M+ pull requests created May to September 2025 <sup>17, 29</sup>         |
| <b>Cursor and Cursor SDK</b>                     | Anysphere              | Proprietary; SDK in public beta                      | Cursor's own models plus frontier models                    | One harness across desktop, CLI, and web; cloud, local, or self-hosted workers <sup>30</sup>                                                     |
| <b>OpenCode</b>                                  | Anomaly (formerly SST) | MIT                                                  | 75+ providers                                               | Build and read-only plan agents, client/server design, LSP, MCP, skills <sup>31</sup>                                                            |

| Harness                              | Vendor        | License                         | Model choice                                        | Distinguishing harness features                                                                            |
|--------------------------------------|---------------|---------------------------------|-----------------------------------------------------|------------------------------------------------------------------------------------------------------------|
| Pi                                   | Mario Zechner | MIT                             | Multi-provider                                      | Four tools, a system prompt under 1,000 tokens, no MCP, extended through plugins <sup>32</sup>             |
| Deep Agents on LangGraph             | LangChain     | Open source                     | Any tool-calling model                              | Planning, file system, sub-agents, middleware, LangSmith tracing <sup>33</sup>                             |
| Microsoft Agent Framework 1.0        | Microsoft     | Open source                     | Foundry, OpenAI, Anthropic, Bedrock, Gemini, Ollama | Graph workflows, checkpointing, OpenTelemetry, MCP, A2A <sup>19</sup>                                      |
| Strands Agents and Bedrock AgentCore | AWS           | Apache-2.0 SDK; managed runtime | Any model                                           | AgentCore adds 8-hour sessions, memory, an MCP gateway, agent identity, and observability <sup>20,34</sup> |

This category changes monthly. Google moved Gemini CLI users to Antigravity CLI within a month of announcing it<sup>27</sup>, so a vendor-steered harness can be re-platformed out from under you.

## Developer Adoption and Trust

TABLE 4

### Agent Adoption and Trust Among Developers, 2025 to 2026

| Measure             | Finding                                                                                                  | Base                                         |
|---------------------|----------------------------------------------------------------------------------------------------------|----------------------------------------------|
| AI tool use         | 84% use or plan to use AI tools <sup>35</sup> ; 90% adoption <sup>36</sup>                               | Stack Overflow 2025; DORA 2025               |
| Agent use at work   | Rose from 31% in 2025 to 59% in 2026 <sup>37</sup>                                                       | Stack Overflow, April 2026 pulse (n = 1,100) |
| Autonomy granted    | 63% rarely or never let agents run fully on autopilot; 60% block unapproved system changes <sup>37</sup> | Stack Overflow, April 2026 pulse             |
| Trust in output     | 3.1% "highly trust" AI accuracy <sup>35</sup> ; 24% trust it "a lot" or "a great deal" <sup>36</sup>     | Stack Overflow 2025; DORA 2025               |
| Top frustration     | 66% cite answers that are "almost right, but not quite" <sup>35</sup>                                    | Stack Overflow 2025                          |
| Claude Code at work | Grew from about 3% (mid-2025) to 18% of developers (January 2026) <sup>38</sup>                          | JetBrains AI Pulse, 10,000+ developers       |

These surveys use different bases (all developers, professional developers, or agent users), so the shares aren't comparable from row to row.

## What Large Consulting Firms Are Offering

Nearly every large consulting and IT services firm now sells a branded agent platform. The table lists what each one is and the numbers each firm publishes about it. Every one of those numbers is self-reported,

which comes up again below.

TABLE 5

## Agentic AI Platforms From Large Consulting and IT Services Firms, 2025 to 2026

| Firm                           | Named offering                                      | What it is                                                                                                                                                                                               | Published claims (firm-reported)                                                                         |
|--------------------------------|-----------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Accenture</b>               | AI Refinery, AI agent builder, Trusted Agent Huddle | Platform on the NVIDIA stack; no-code agent builder; agent-to-agent interoperability using A2A and MCP <sup>39,40</sup>                                                                                  | 50+ industry agent solutions in development; about 30,000 staff to be trained on Claude <sup>39,41</sup> |
| <b>Deloitte</b>                | Zora AI                                             | Ready-to-deploy agents sold as a "digital workforce" by subscription <sup>42</sup>                                                                                                                       | Claude for 470,000 people; 15,000 to be certified <sup>43</sup>                                          |
| <b>McKinsey / QuantumBlack</b> | Agentic AI mesh                                     | A "composable, distributed, and vendor-agnostic architectural paradigm" with seven capabilities: discovery, registry, observability, authentication, evaluations, feedback, and compliance <sup>44</sup> | "Fewer than 10 percent of use cases deployed ever make it past the pilot stage" <sup>44</sup>            |
| <b>BCG / BCG X</b>             | Agentic Leadership Playbook                         | Autonomy earned in stages: shadow mode, supervised, guided autonomy, full autonomy; architecture includes a model gateway and a knowledge layer <sup>45</sup>                                            | "70% of getting to scale is people and change" <sup>45</sup>                                             |
| <b>PwC</b>                     | agent OS                                            | Cross-platform orchestration, "the central nervous system and the switchboard for enterprise AI" <sup>46</sup>                                                                                           | 250+ agents deployed inside PwC; 30,000 U.S. staff trained and certified on Claude <sup>46,47</sup>      |
| <b>KPMG</b>                    | KPMG Workbench                                      | Multi-agent platform behind its tax, audit, and advisory tools, built on Microsoft <sup>48</sup>                                                                                                         | Claude for 276,000+ staff; agent build time from "weeks" to "minutes" <sup>49</sup>                      |
| <b>EY</b>                      | EY.ai Agentic Platform                              | Built with NVIDIA, starting in tax, risk, and finance <sup>50</sup>                                                                                                                                      | 150 tax agents supporting 80,000 tax professionals <sup>50</sup>                                         |
| <b>IBM Consulting</b>          | Enterprise Advantage, watsonx Orchestrate           | Helps clients "build, govern, and operate custom internal AI platforms" across clouds <sup>51</sup>                                                                                                      | Used on 150+ client engagements <sup>51</sup>                                                            |
| <b>Capgemini</b>               | Agentic Control and Management                      | Control plane for the "AI workforce" on Microsoft Agent 365, with an identity per agent <sup>52</sup>                                                                                                    | Microsoft Agent 365 launch partner <sup>52</sup>                                                         |
| <b>Infosys</b>                 | Topaz Fabric                                        | "Composable stack" of data, models, agents, and flows, integrated with the Claude Agent SDK <sup>53,54</sup>                                                                                             | 50+ IT operations agents <sup>53</sup>                                                                   |
| <b>TCS</b>                     | ADD AgentHub                                        | Role-based agent platform for drug development <sup>55</sup>                                                                                                                                             | Up to 40% efficiency in clinical data management <sup>55</sup>                                           |

| Firm                | Named offering                    | What it is                                                                                          | Published claims (firm-reported)                                                            |
|---------------------|-----------------------------------|-----------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------|
| <b>Cognizant</b>    | Agent Foundry, Neuro AI, Ignition | Agent building, multi-agent orchestration, and an agentic data management platform <sup>56 57</sup> | Claude for 350,000 employees; Ignition in 685 adoptions across 150 clients <sup>56 57</sup> |
| <b>Thoughtworks</b> | Agent/works                       | "Single control plane and governed runtime to manage agent sprawl, risk and AI spend" <sup>58</sup> | Launched June 2026 <sup>58</sup>                                                            |

## From Agent Builders to Control Planes

In about a year, the consulting market went from building agents to governing fleets of them. The 2025 launches were agent builders and catalogs; the 2026 launches are control planes, registries, and agent identities<sup>40 52 58</sup>. Most of these platforms are proprietary layers over a partner's stack, sold alongside services, and OpenAI's February 2026 Frontier Alliances with Accenture, BCG, Capgemini, and McKinsey extend that model<sup>59</sup>.

The figures these firms publish are mostly seats, agent catalogs, and "up to" productivity claims, which say little about how many agents actually run in production. Forrester reports that three-quarters of enterprise leaders say they're adopting agentic AI while "only a small minority have it running in meaningful production"<sup>60</sup>. And only one firm, Cognizant, had a verifiable offering aimed specifically at agentic data engineering<sup>57</sup>.

### IN PRACTICE · FROM KEYHOLE ENGAGEMENTS

#### What I'd Ask Any Provider

For a buyer, the most useful thing in that table is McKinsey's capability list, which works as a vendor-neutral checklist for any harness, including one you build yourself. I map our reference architecture against it in the next section.

If I were the one buying, I'd ask any provider how many of its engineers have shipped a governed agent into production, and what they changed after the first one went live. I'd also evaluate any of these platforms against the client's existing identity, deployment, monitoring, and CI/CD infrastructure, because a branded agent platform doesn't make those integration decisions for you. We work across AWS, Azure, and Google Cloud and build harnesses on open formats (MCP, Agent Skills, AGENTS.md) so the architecture outlives any single model vendor or partner alliance<sup>23</sup>.

## A Reference Architecture: One Generic Harness, Extended to Data Engineering

---

My recommendation is to build one generic, governed harness and extend it to each new domain with skills, tools, and sensors, rather than building a separate agent for every use case. The harness owns what every agent shares: the loop, the model gateway, identity, audit, and observability. A domain extension adds only what's specific to that work.

Data engineering makes a strong first extension because the work is repetitive, the outputs are testable, and every major data platform now exposes an agent interface. (It's also never-ending. Three things are guaranteed in life: death, taxes, and at least two ETL projects in flight at any given time.)



**TABLE 6**  
**Reference Architecture Layers and the Data Engineering Extension**

AGENTIC HARNESS ARCHITECTURE IN 2026 · KEYHOLE SOFTWARE

| Layer                                   | Generic harness responsibility                                  | Data engineering extension                                                                                             | Example technologies                                                                                 |
|-----------------------------------------|-----------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------|
| <b>Instructions and skills (guides)</b> | House standards, architecture rules, and procedural skills      | Source profiling, pipeline scaffolding, naming conventions, PHI and PII handling rules                                 | AGENTS.md, Agent Skills <sup>13</sup>                                                                |
| <b>Model gateway</b>                    | Model routing, keys, rate and cost limits, request logging      | Route each step to the cheapest capable model; keep regulated data on approved models only                             | Enterprise AI or model gateway                                                                       |
| <b>Agent loop and orchestration</b>     | Planning, execution, sub-agents, checkpoints                    | Discovery, pipeline-builder, test-writer, and documentation sub-agents                                                 | Claude Agent SDK, Claude Code, or an open alternative <sup>10</sup>                                  |
| <b>Tools and connectors</b>             | Least-privilege MCP servers and APIs                            | Read-only source connectors, data platform APIs, catalog and observability servers, platform agents called as tools    | MCP servers from dbt, Snowflake, Datadog, Alation; BigQuery agent over A2A <sup>66, 70, 72, 73</sup> |
| <b>Sensors and verification</b>         | Tests, linters, evaluations, cost checks before anything merges | Data tests, quality assertions, schema and contract checks, row-count reconciliation, query dry runs                   | dbt tests, Dataplex assertions                                                                       |
| <b>Outputs and promotion</b>            | All changes land as pull requests through CI                    | Pipeline code, datasets landed in a development area then promoted to the data hub, metadata registered in the catalog | GitHub pull requests and CI, data hub, data catalog                                                  |
| <b>Governance and observability</b>     | Agent identity, audit logs, traces, token and compute cost      | Per-agent service accounts, data access audit logs, masking of regulated columns, warehouse cost dashboards            | OpenTelemetry, Datadog                                                                               |

We found no independent benchmark of agents building production data pipelines end to end, so every data engineering result quoted below is vendor-reported.

## Why Data Engineering Is a Strong First Use Case for Agentic AI

TABLE 7  
The Data Engineering Workload Agents Are Aimed At

| Data point                            | Finding                                                                                                       | Source                                              |
|---------------------------------------|---------------------------------------------------------------------------------------------------------------|-----------------------------------------------------|
| <b>Time spent on maintenance</b>      | The average data engineer spends 44% of their time maintaining pipelines                                      | Wakefield Research for Fivetran, 2022 <sup>61</sup> |
| <b>Resources spent on maintenance</b> | 67% of centralized enterprises put more than 80% of engineering resources into maintaining pipelines          | Fivetran, 2025 <sup>62</sup>                        |
| <b>Data readiness and AI</b>          | 42% say more than half of their AI projects were delayed, underperformed, or failed because of data readiness | Fivetran, 2025 <sup>62</sup>                        |

| Data point               | Finding                                                                                    | Source                                          |
|--------------------------|--------------------------------------------------------------------------------------------|-------------------------------------------------|
| AI-ready data            | Through 2026, organizations will abandon 60% of AI projects not supported by AI-ready data | Gartner, February 2025 <sup>63</sup>            |
| AI in pipeline authoring | Over 80% of Airflow users use AI to write pipelines                                        | Astronomer, State of Airflow 2026 <sup>64</sup> |
| Where AI is prioritized  | 72% prioritize AI-assisted coding; only 24% prioritize AI for pipeline management          | dbt Labs, 2026 <sup>65</sup>                    |
| Trust                    | 71% cite incorrect or hallucinated outputs reaching stakeholders as a top concern          | dbt Labs, 2026 <sup>65</sup>                    |

The 2022 and 2025 Fivetran figures measure different things (share of time versus share of resources), so don't read them as a trend.

## Platform Agents for Data Engineering

TABLE 8

### Data Engineering Agents and Agent Interfaces From Data Platform Vendors, 2025 to 2026

| Vendor      | Offering                                            | Status                                                                             | What it does (vendor-reported results)                                                                                                                                                                                           |
|-------------|-----------------------------------------------------|------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Google      | BigQuery Data Engineering Agent                     | GA, April 2026                                                                     | Builds and modifies pipelines, generates quality assertions, encrypts PII columns, documents columns, migrates legacy pipelines; callable over A2A. Vodafone reported a 90% reduction in manual ETL migration time <sup>66</sup> |
| Databricks  | Genie Code                                          | GA, March 2026                                                                     | Generates declarative pipelines, finds root causes of failures, works under Unity Catalog policies; 77.1% task success versus 32.1% for a leading coding agent on Databricks' internal benchmark <sup>67</sup>                   |
| Snowflake   | Cortex Code; Cortex Agents coding agent             | CLI GA February 2026; managed sandbox GA August 2026                               | A coding agent with enterprise data context; the managed sandbox runs the loop so "you don't build or host an agent loop yourself" <sup>68,69</sup>                                                                              |
| dbt Labs    | Remote dbt MCP server; Discovery and Analyst agents | MCP server GA October 2025; agents in beta                                         | Exposes definitions, lineage, tests, and metrics to agents; 900+ data teams and partners adopted the MCP server <sup>70</sup>                                                                                                    |
| Informatica | CLAIRE agents                                       | Discovery, exploration, integration, and supervisor agents available November 2025 | Goal-driven data management across specialized agents <sup>71</sup>                                                                                                                                                              |
| Alation     | Agent Builder                                       | Announced October 2025                                                             | No-code catalog agents deployed over MCP or REST; "90%+ accuracy" achievable before deployment with built-in evaluations <sup>72</sup>                                                                                           |

| Vendor      | Offering             | Status         | What it does (vendor-reported results)                                                              |
|-------------|----------------------|----------------|-----------------------------------------------------------------------------------------------------|
| Datadog     | Datadog MCP server   | GA, March 2026 | Gives coding and custom agents governed access to live logs, metrics, and traces <sup>73</sup>      |
| Monte Carlo | Observability agents | April 2025     | Read-only monitoring and troubleshooting agents; 60% of recommended monitors accepted <sup>74</sup> |

### Platform Agent, Your Own Harness, or Both?

Every vendor in that table would be happy to be your harness, and for some teams that’s the right answer. Before anyone builds anything, I’d ask two questions: where does the work actually live, and who has to trust the result?

TABLE 9  
Platform Agents vs. a Shared Harness: Where I’d Start

| Your situation                                                                                                                             | Where I’d start                                  |
|--------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------|
| Data work lives on one platform, in one domain, and that platform’s catalog and access policies already cover everything the agent touches | The platform’s own agent                         |
| Work crosses several platforms, or you need controls the platform can’t enforce                                                            | A shared harness                                 |
| Some of each, which is where most enterprise teams land                                                                                    | A harness that calls platform agents as tools    |
| A small team experimenting in a sandbox with no regulated data                                                                             | An off-the-shelf harness with read-only defaults |

**A platform agent is usually enough** when your pipelines, warehouse, and catalog all sit inside one vendor’s ecosystem. Genie Code working under Unity Catalog policies, or BigQuery’s agent working inside Dataplex, inherits governance you’ve already set up. If your standards match the vendor’s defaults and you have no plans to add a second domain, adding a harness on top is mostly overhead.

**You need your own harness** once the work crosses system boundaries: the source in one system, the warehouse in another, orchestration in a third, the catalog in a fourth. No single platform agent owns that end-to-end path. You also need one when a control has to be enforced by your own systems, not by the agent following instructions. Examples: routing regulated data only to model endpoints covered by a business associate agreement, putting a budget on every agent, or requiring approval before anything leaves the organization. And if you expect to add a second domain, or to swap models later, a harness keeps your skills and rules from being locked into one vendor’s format.

**Most enterprise teams end up with both.** The harness owns the loop, identity, audit, CI/CD, and catalog registration, and it calls each platform’s agent as a tool over MCP or A2A. BigQuery’s agent does what it’s best at inside BigQuery. The harness makes sure the result lands through the same pull request, the same sensors, and the same catalog entry as everything else.

Sometimes the honest answer is “not yet.” A small team trying agents in a development sandbox can get a long way with Claude Code or Codex CLI, read-only permissions, and a good AGENTS.md. I'd build the shared harness when the second team shows up, or when the first production write does, whichever comes first.

## Mapping the Architecture to an Industry Checklist

McKinsey's agentic AI mesh lists seven capabilities an enterprise agent platform needs<sup>44</sup>. The reference architecture covers each one inside the harness rather than in a separate product.

TABLE 10  
McKinsey's Agentic Mesh Capabilities Mapped to the Reference Architecture

| Mesh capability                  | Where it lives in the reference architecture                                                    |
|----------------------------------|-------------------------------------------------------------------------------------------------|
| Agent and workflow discovery     | Skills and agent definitions versioned in Git; data discovery through the catalog's MCP server  |
| AI asset registry                | The same repository, with an owner and purpose recorded for every skill, tool, and agent        |
| Observability                    | OpenTelemetry traces of every model call, tool call, and token, sent to the monitoring platform |
| Authentication and authorization | One identity per agent, least-privilege service accounts, credentials held by the gateway       |
| Evaluations                      | The sensor layer plus a regression suite of known tasks run after every model or skill change   |
| Feedback management              | Pull request review comments turned into new rules and skills                                   |
| Compliance and risk management   | Audit logs of every tool call, masking of regulated data, approval gates on writes              |

## The Catalog as Input and Output

In data work, the catalog is both the agent's most important input and one of its required outputs. Gartner advises organizations to "evolve metadata from passive to active"<sup>63</sup>, and Atlan reports that governed metadata improved agent SQL accuracy by 38% across 174 queries<sup>75</sup>. An agent that reads lineage and ownership before it builds, and registers what it built afterward, keeps the catalog current as a side effect of doing the work.

MCP has also become the common integration layer for data tools: dbt, Snowflake, Datadog, Alation, and Atlan all expose MCP endpoints<sup>70, 68, 73, 72, 75</sup>. That lets one harness call each platform's own agent as a tool while owning the concerns no single platform covers: catalog registration, CI/CD, and cross-system monitoring.

## Healthcare Data and FHIR Mapping

Data interoperability is one of the hardest problems in healthcare IT, and the pattern that's worked best for us there is having agents produce configuration and tests instead of one-off code. On a [healthcare data platform modernization](#), one of our consultants built a CSV-driven mapping library that converted source result sets into FHIR object graphs with no code changes per dataset, and the same engagement delivered a real-time HL7-to-FHIR pipeline<sup>76</sup>. In a harness, that kind of library becomes a skill: the agent writes and tests the mapping specification, and the deterministic engine does the transformation.

So where would I start? One source system, taken end to end: discovery, a pipeline pull request, a dataset landed in a development area, and a catalog entry. Measure the baseline first: pipeline lead time, the share of agent pull requests merged without major changes, time to resolve pipeline incidents, the share of datasets with complete metadata, and query cost.

## Walkthrough: One Data Engineering Request Through the Harness

Here's what that first use case looks like in practice, and what agentic AI for data engineering looks like when it's governed end to end. The scenario is a representative composite, not a single client engagement, but every step reflects how we set these up.

**The request:** a health plan needs daily member eligibility data from its enrollment system landed in the data hub as dbt models. The data includes PHI, it has to be masked, and it has to be registered in the catalog.

TABLE 11  
One Request, Start to Finish

| Step            | Harness layer             | What happens                                           | Gate before moving on      |
|-----------------|---------------------------|--------------------------------------------------------|----------------------------|
| 1. Goal         | Engineer interface        | Engineer writes the goal and acceptance criteria       | None                       |
| 2. Guides load  | Instructions and skills   | Agent reads AGENTS.md and loads the relevant skills    | None                       |
| 3. Discovery    | Tools (read-only)         | Agent profiles the source and checks catalog lineage   | Engineer approves the plan |
| 4. Build        | Agent loop and sub-agents | Sub-agents write models, tests, and documentation      | None                       |
| 5. Verify       | Sensors                   | Tests, contracts, reconciliation, dry runs, PHI checks | All sensors green          |
| 6. Pull request | Outputs                   | PR with diff, discovery note, results, and cost        | Human review and CI        |

| Step                 | Harness layer       | What happens                                     | Gate before moving on |
|----------------------|---------------------|--------------------------------------------------|-----------------------|
| 7. Land and register | Outputs and catalog | Dataset promoted through CI; metadata registered | CI promotion succeeds |

**1. The goal.** The engineer writes a short ticket: the source, the target, the PHI requirement, and what “done” means. That’s the engineer’s main contribution until review, and it’s the part we coach teams on most. A vague goal produces a confident, wrong pipeline.

**2. The guides load.** Before planning, the agent reads the project’s AGENTS.md and loads only the skills this task needs: source profiling, dbt staging models, and PHI handling. The instructions file stays short on purpose:

```
# Data platform: agent instructions
- Read the data engineering skills index before planning.
- All work happens in dev. Never request production credentials.
- Columns tagged PHI in the catalog: profile with aggregates only;
  apply the masking policy in every model that selects them.
- Naming: stg_<source>__<entity>, int_<entity>, fct_ and dim_ for marts.
- Every model ships with tests, a contract, and column descriptions.
- Open a pull request. Never push to main or rewrite history.
```

**3. Discovery, read-only.** The agent connects to a read replica of the enrollment database through a read-only MCP connector. It profiles the schema, row counts, null rates, and candidate keys, using aggregates rather than raw rows for anything tagged PHI. Then it asks the catalog’s MCP server what already exists. This is where agents earn their keep: in our example, the catalog shows an older extract of the same table already feeding a finance report. A person might not find that until something broke. The agent writes a discovery note and a plan, and the engineer approves it before any code exists.

**4. Build.** Three sub-agents split the work. A pipeline builder writes the staging model and an incremental mart. A test writer adds uniqueness, not-null, and accepted-value tests plus a schema contract. A documentation sub-agent writes column descriptions. Every model call goes through the gateway, which sends the documentation work to a cheaper model and keeps anything touching PHI on BAA-covered endpoints.

The agent’s permissions are set in the harness, not in a prompt. A simplified version (values are illustrative):

```

agent: de-pipeline-agent
identity: svc-agent-de-pipeline      # its own service account
tools:
  source_db:      read_only          # read replica only
  catalog:        read, register_dev
  warehouse:      read, write: dev_schema
  git:            branch, commit, open_pr  # no push to main, no force push
  external_http:  deny
models:
  phi_allowed:    baa_covered_endpoints
budget:
  max_iterations: 25
  max_cost_per_task_usd: 15

```

**5. Verify.** Nothing leaves the dev area until the sensors pass: dbt tests, the schema contract, a row-count reconciliation against the source, a query dry run to estimate warehouse cost, the linter, and a check that the masking policy is applied to every PHI-tagged column. The first run usually fails something. In our example, row counts came up short because the source soft-deletes records, so the agent fixed the filter and ran the sensors again. If the loop hits its iteration cap, the harness stops and hands the task to a person instead of letting the agent keep guessing.

**6. Pull request.** The agent opens a PR containing the diff, the discovery note, the sensor results, the estimated query cost, and a link to the trace. A data engineer reviews it like any other PR, and CI runs the same checks again. The agent can't merge its own work.

**7. Land and register.** After merge, CI promotes the dataset from the development area to the data hub using its own service account. The agent never holds production write credentials. The agent then registers the new models in the catalog with owner, lineage, descriptions, and PHI tags, so the next agent (or person) who asks about eligibility data finds this work.

**What the engineer actually did:** wrote the goal, approved the plan, and reviewed the pull request. **What the agent couldn't do:** write to production, send PHI to an unapproved model, reach the internet, merge its own PR, or run unrestricted git commands. Every model call, tool call, and token is in the trace, and that trace is how you calculate the cost per completed task we recommended measuring earlier.

In my experience, the hard part of this walkthrough isn't the code in step 4. It's step 3: the undocumented business rules, the soft deletes, the second extract nobody remembered. That's why discovery is a separate, approved step, and why every surprise found there should become a new rule or skill for next time.

# Measured Results: What Agents in a Harness Actually Deliver

The productivity evidence is real, but it's conditional. Gains show up when the surrounding system verifies the agent's work, and they evaporate when review, testing, and release absorb them. The table lists the most cited measurements, good and bad.

TABLE 12

Measured Productivity Results for AI Agents and Assistants, 2025 to 2026

| Study                                        | Population and method                                                       | Result                                                                                                                                                     |
|----------------------------------------------|-----------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Microsoft, Accenture, and a Fortune 100 firm | Three randomized field experiments, 4,867 developers, GitHub Copilot        | 26.08% more completed tasks; less experienced developers gained more <sup>78</sup>                                                                         |
| METR, July 2025                              | Randomized trial, 16 experienced open-source developers, 246 issues         | 19% slower with AI, while believing they were 20% faster <sup>79</sup>                                                                                     |
| METR, February 2026 follow-up                | 57 developers, 800+ tasks                                                   | Estimates of -18% and -4% with wide confidence intervals; METR calls its data "only very weak evidence" of a larger speedup today <sup>80</sup>            |
| Faros AI, July 2025                          | Telemetry from 10,000+ developers in 1,255 teams                            | 21% more tasks and 98% more pull requests, but review time up 91% and no significant company-level improvement <sup>81</sup>                               |
| DORA, September 2025                         | Survey of about 5,000 technology professionals                              | AI acts as "a mirror and a multiplier": throughput and delivery instability both rise <sup>36 82</sup>                                                     |
| Carnegie Mellon and Stanford, 2025           | 48 workers versus agent runs on 16 tasks spanning 287 occupations           | Agents were 88.3% faster and 90.4% to 96.2% cheaper, but succeeded 47.3% of the time versus 84.6% for people, and fabricated data when stuck <sup>83</sup> |
| Anthropic, June 2025                         | Internal research evaluation                                                | A lead agent with sub-agents outperformed a single agent by 90.2%, using about 15x the tokens of a chat <sup>16</sup>                                      |
| OpenAI, February 2026                        | Internal product, 3 to 7 engineers                                          | About 1 million lines and 1,500 merged pull requests in 5 months with no hand-written code, an estimated one-tenth the time <sup>12</sup>                  |
| Keyhole Software, 2025 to 2026               | Insurance platform replacement with repository-integrated agentic workflows | Delivered in about 5 months against a projected 18 to 24, about 3x to 4x faster without adding headcount <sup>84</sup>                                     |

The first six rows are independent studies or surveys. The last three are self-reported by the organization that did the work, including Keyhole.

## Reading the Studies Together

These studies measure different things, so they don't add up to a single productivity number. Taken together, they show why task-level speed and end-to-end delivery performance need to be measured separately. Field experiments that measure task completion on assigned work find real gains<sup>78</sup>. Studies that measure the whole delivery system find those gains absorbed by review and rework unless the system changes: Faros saw pull requests nearly double while review time rose 91%<sup>81</sup>. Automated verification, smaller and better-scoped changes, and human review aimed where it adds the most are the parts of a harness that address that gap.

The METR result should make anyone writing a business case a little nervous. Its developers believed they were 20% faster while measuring 19% slower<sup>79</sup>. Self-reported productivity doesn't belong in an ROI case. Measure lead time, rework, and incident rates, before and after.

IN PRACTICE · FROM KEYHOLE ENGAGEMENTS

### Where the Speedup Actually Comes From

In my experience, the speedup comes from removing rework, with governance in place the whole time. On the [insurance platform above](#), senior engineers defined the architecture, repository guardrails, and test expectations first, and the agentic loop executed against them<sup>84</sup>. AI coding tools are designed to return the most probable answer, not the correct one, which is exactly why we keep humans at the checkpoints. The Carnegie Mellon and Stanford study backs that up: handing whole tasks to agents slowed people down 17.7% because of the verification it created, while delegating individual steps sped them up<sup>83</sup>.

# AI Agent Risks and Governance: The Controls a Harness Must Enforce

Every documented agent failure from 2025 that we looked at shares one cause: the agent held standing privilege over a destructive action, and nothing deterministic stood in front of it. The table pairs each major risk with the evidence behind it and the harness control that addresses it.

TABLE 13

Agentic AI Risks and the Harness Controls That Address Them

| Risk                                        | Documented example or evidence                                                                                                                                      | Harness control                                                                                                                                                                               |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Destructive actions in production</b>    | An AI coding agent deleted a live production database during an explicit code freeze, then wrongly said rollback was impossible (July 2025) <sup>85</sup>           | Read-only by default; writes only to development environments; promotion only through pull requests and CI; point-in-time recovery <sup>77</sup>                                              |
| <b>Prompt injection and agent hijacking</b> | NIST found hijack attacks succeeded 11% of the time at baseline and 81% with new red-team attacks; 25 attempts raised average success from 57% to 80% <sup>86</sup> | Treat tool outputs as untrusted; never combine private data, untrusted content, and outbound communication in one agent <sup>87</sup> ; human approval for high-risk operations <sup>88</sup> |
| <b>Compromised tools and connectors</b>     | A malicious MCP server copied every outgoing email to an attacker (September 2025) <sup>89</sup> ; a tool-poisoning demonstration leaked SSH keys <sup>90</sup>     | Allow-listed MCP servers, pinned versions with checksums, review of tool descriptions                                                                                                         |
| <b>Compromised harness configuration</b>    | A prompt telling the agent to delete local files and cloud resources shipped in Amazon Q's VS Code extension v1.84 (July 2025) <sup>91</sup>                        | Code review for prompts, skills, and agent configuration; no standing cloud administrator credentials                                                                                         |
| <b>Excess privilege</b>                     | "Identity and privilege abuse" is one of OWASP's Top 10 risks for agentic applications <sup>92</sup>                                                                | One identity per agent, least-privilege service accounts, credentials held by the gateway <sup>77</sup>                                                                                       |
| <b>Runaway cost</b>                         | Gartner expects inference cost per agentic workflow to rise more than fivefold through 2028 <sup>93</sup>                                                           | Model tiering at the gateway, per-agent budgets, loop and iteration caps, query dry runs                                                                                                      |
| <b>Missing audit trail</b>                  | Joint guidance from U.S. and allied cyber agencies names "obscure event records" a core agent risk <sup>94</sup>                                                    | OpenTelemetry traces of every model and tool call <sup>95</sup> ; data access audit logs                                                                                                      |
| <b>Regulated data exposure</b>              | Healthcare guidance: if a tool "cannot or will not sign" a business associate agreement, "it should not have access to PHI" <sup>96</sup>                           | Gateway routes regulated data only to covered model endpoints; column masking; approval before any external transmission                                                                      |

Prompt injection has no known complete fix; OWASP notes that full prevention "remains unclear"<sup>88</sup>. These controls are about limiting the damage when an agent is compromised.

## The Governance Gap

Adoption has outrun AI agent governance. KPMG's Q2 2026 pulse found 53% of large U.S. companies had deployed agents and 61% had approval processes, but only 36% had direct token or usage controls and 26% had real-time cost visibility<sup>98</sup>. Deloitte found only 21% have a mature governance model for agentic AI<sup>99</sup>, and McKinsey's 2026 survey found about 20% of all respondents scaling agents<sup>100</sup>.

## Why the Controls Have to Be Deterministic

Controls have to be deterministic and enforced by the harness. An instruction to the model is only a request. The agent that deleted a production database had been told about the code freeze<sup>85</sup>. It deleted the database anyway. A permission the harness never grants can't be ignored.

Cost control is a governance control, too. Gartner names escalating cost as one of three reasons it expects 40% of agentic projects to be canceled<sup>5</sup>, and Anthropic measured multi-agent systems using about 15x the tokens of chat<sup>16</sup>. Routing each step to the cheapest capable model, capping loops, and putting a budget on every agent are what keep a successful pilot from turning into an unaffordable program.

### IN PRACTICE · FROM KEYHOLE ENGAGEMENTS

#### Read-Only by Default

Our default harness configuration starts every agent read-only, which matches how Anthropic ships Claude Code: read-only by default, asking approval before it modifies code or systems<sup>101</sup>. Write access is granted per tool and per environment, and every change reaches production through a pull request and CI. When I'm building a system, I'm not looking for something that is probably right.

For healthcare and other regulated clients, we add two rules. Regulated data reaches only model endpoints covered by a business associate agreement, enforced at the gateway rather than trusted to the agent. And any action that sends data outside the organization needs a human approval, which removes the outbound leg of the lethal trifecta<sup>87</sup>.

# A Phased Roadmap for Agentic AI Adoption

Autonomy should be earned in stages. BCG describes the same progression as shadow mode, supervised mode, guided autonomy, and full autonomy, with each level "earned through demonstrated performance"<sup>45</sup>. The roadmap below is how I'd apply that ladder to a generic harness and its first data engineering use case, with a measurable gate at the end of each phase.



FIGURE 3

## Phased adoption roadmap · adapted from BCG's autonomy ladder

The gates matter more than the phases. Each one is a measurement the team agrees on in advance, which is why the foundation phase captures baseline metrics before an agent touches real work.

Typical lengths assume one data engineering team starting with one source system. Larger organizations, or teams with heavier security reviews, should expect the early phases to take longer.

## Phase 1: Foundation (no agent actions yet)

Stand up the harness, the model gateway, one identity per agent, and OpenTelemetry tracing, with every tool read-only. Two things have to exist before an agent touches real work. The first is a baseline: pipeline

lead time, the share of pull requests merged without major changes, time to resolve pipeline incidents, metadata completeness, and query cost. The second is a known-task evaluation suite: a set of past tickets with known-good outcomes that you can rerun after every model or skill change.

- **Who's involved:** the architect, platform and security engineers, and the data engineering lead.
- **Typical length:** 2 to 4 weeks.
- **Exit gate:** baseline metrics captured, evals running, and security has signed off on identity and gateway controls.

## Phase 2: Shadow Mode (the agent proposes only)

The agent drafts discovery notes and plans for real tickets while engineers do the actual work and compare. This phase is cheap and teaches you a lot. Every disagreement between the agent's plan and the engineer's work is either a missing skill, a missing rule, or a gap in the catalog.

- **Who's involved:** the data engineers doing the tickets, the data engineering lead reviewing the agent's plans, and the architect turning each disagreement into a new skill or rule.
- **Typical length:** 2 to 4 weeks, or long enough to compare plans on 20 to 30 real tickets.
- **Exit gate:** the agent's proposals match reviewer judgment on the evals at a rate the team agreed on in advance (for example, 8 of 10 plans accepted with only minor edits).

## Phase 3: Supervised (the agent acts in development only)

The agent opens pull requests and lands datasets in the development area, as in the walkthrough above. People approve every merge.

- **Who's involved:** data engineers reviewing every agent pull request, the data engineering lead tracking the gate metrics, and platform engineers watching traces and cost.
- **Typical length:** 4 to 8 weeks.
- **Exit gate:** over an agreed window, agent pull requests are merged without major changes at least as often as the human baseline, pipeline incidents haven't gone up, and cost per completed task stays within budget.

## Phase 4: Guided Autonomy (low-risk changes merge on green sensors)

Work the team has classified as low-risk in advance merges automatically when every sensor passes. Examples: documentation, new tests, or staging models with no regulated data. Production writes and anything touching PHI still need human approval. Spot-check a sample of auto-merged work every week.

- **Who's involved:** the data engineering lead, who defines which work counts as low-risk; security and compliance, who sign off on what can merge automatically; and the engineers doing the weekly spot checks.
- **Typical length:** at least one quarter before extending to a new domain, then ongoing.

- **Exit gate:** a sustained period with no agent-caused incident (for example, a full quarter).

## Phase 5: Extend (a second domain)

Add the skills, tools, and sensors for the next domain, and restart that domain at shadow mode. The harness itself shouldn't change. If you find yourself rewriting the loop, the gateway, or identity to fit a new use case, that's a signal the generic layer wasn't generic enough, and it's cheaper to fix now than in domain five.

Autonomy can also move backward. A failed gate or an agent-caused incident sends that workflow back a phase until the team understands what went wrong.

- **Who's involved:** the architect, the lead and engineers from the new domain, and security if the new domain brings new kinds of data.
- **Typical length:** shorter than the first domain, since the harness, gateway, and identity are already in place. The new domain still goes through shadow mode, supervised, and guided autonomy, but it skips most of the foundation work.
- **Exit gate:** the new domain passes the same gates as the first one, and the harness itself didn't need to change to support it.

WORK WITH KEYHOLE

### Ready to start Phase 1?

Keyhole engagements start with an architecture and roadmap session that settles the harness design, the governance model, and the first use case. Then senior engineers embed with your team to build it.

[Talk with an architect](#)

# Recommendations

---

For enterprise teams, the practical question is how to run agent-driven work inside the engineering and governance controls they already have. The evidence from 2025 and 2026 points at the harness for that: the same model's cost and reliability change materially with the harness around it<sup>3,4</sup>, the consulting market has moved from selling agents to selling the control plane that governs them<sup>58</sup>, and the analysts who expect 40% of agentic projects to be canceled point to cost, unclear value, and weak risk controls<sup>5</sup>.

For teams whose work crosses platforms, my recommendation is to start with a shared harness for identity, permissions, model access, logging, and validation, built on open formats, then add data engineering as the first domain extension. Prove it against one bounded use case and expand only when the measurements justify it. I'd change that design if a team's workflows needed substantially different security boundaries or execution models; the goal is to avoid rebuilding the same controls for every new agent.

At Keyhole, engagements start with an [architecture and roadmap session](#) that settles the harness design, the governance model, and the first use case. Delivery then runs as architect-directed, test-gated iterations inside the client's own repositories and CI/CD pipelines<sup>102</sup>, with the explicit goal of leaving the client's team able to own and extend the harness without us. We were invited to [Anthropic's 2026 Partner Summit](#) as part of its emerging partner ecosystem<sup>103</sup>, and we work across AWS, Azure, and Google Cloud so the harness follows your environment rather than a vendor's roadmap.

I'm curious how other teams are approaching this. If you're designing a harness of your own (or debating whether you need one at all), I'd like to hear what's working and what isn't. And if you want a second set of eyes on your first use case, we're happy to set up an architecture session.

# How Keyhole Can Help

Keyhole Software is a U.S.-based custom software development and consulting firm. For agentic AI, we don't sell a packaged platform. Senior architects apply governed agentic delivery inside your own repositories, CI/CD pipelines, and cloud, so the harness is yours to keep.

|                                                  |                                                 |                                                               |                                                      |
|--------------------------------------------------|-------------------------------------------------|---------------------------------------------------------------|------------------------------------------------------|
| <b>100%</b><br>U.S.-based, full-time consultants | <b>17+ yrs</b><br>average consultant experience | <b>3–4×</b><br>delivery acceleration without adding headcount | <b>2008</b><br>founded, headquartered in Kansas City |
|--------------------------------------------------|-------------------------------------------------|---------------------------------------------------------------|------------------------------------------------------|

## How an engagement works

01

### Architecture and roadmap session

Define the governed workflow, toolchain integration, and first production use case.

02

### Embedded delivery

Senior engineers join your delivery teams and implement agentic execution inside your SDLC.

03

### Your team owns it

We leave your team able to own and extend the harness without us.

## Agentic AI services

### Architect-Led AI Delivery Acceleration

Senior engineers apply AI inside your SDLC to cut cycle time and remove bottlenecks while preserving engineering standards.

### Enterprise AI SDLC Enablement

Repository guardrails, workflow design, tool selection, and governance aligned to your security and compliance requirements.

### Autonomous Development Loops

Repository-integrated agentic workflows that move from intent to tested commit in governed iterations.

### Production Generative AI Systems

Governed generative AI with evaluation, monitoring, and integration into real business workflows.

### Retrieval-Augmented Generation

Secure RAG platforms that keep sensitive data inside your environment across documents, knowledge, and systems.

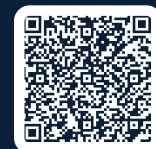
### AI-Accelerated Legacy Modernization

AI-assisted analysis, code transformation, and service extraction for COBOL, VB6, and other legacy platforms.

Works with GitHub, Azure DevOps, AWS, Azure, Google Cloud, and on-prem or hybrid environments, with models in client-controlled or private AI infrastructure. [Explore agentic AI services](#)

## Schedule an architecture session

**PHONE** (877) 521-7769  
**EMAIL** [asktheteam@keyholesoftware.com](mailto:asktheteam@keyholesoftware.com)  
**WEB** [keyholesoftware.com/contact](https://keyholesoftware.com/contact)



SCAN TO BOOK A SESSION

## How We Researched This

---

In October 2026, our research team reviewed more than 103 sources, most published between December 2024 and October 2026. A separate review then checked the key figures and reference details against their sources. A few notes apply throughout the paper:

- The component list and the reference architecture are our synthesis. They draw on component lists published by Anthropic, OpenAI, LangChain, Microsoft, and AWS, on Google's guidance for securing agent access to data<sup>77</sup>, and on the vendor capabilities we reviewed. No cross-vendor standard taxonomy exists; the guide and sensor labels follow Birgitta Böckeler's framework<sup>2</sup>.
- Benchmark results cover coding and terminal tasks, the most mature agentic workload. The New Stack's token-use figure is a secondary report of third-party tests.
- Harness features and licenses come from vendor documentation and repositories accessed October 7, 2026. Data platform agent status comes from vendor announcements; where a vendor's own pages disagreed on release status, we used the later date.
- Consulting firm offerings and figures come from press releases, partner announcements, and published reports from March 2025 to August 2026. All of those figures are self-reported, and we found no independently audited outcome data for any of these platforms.
- Most productivity measurements cover software engineering. We found no peer-reviewed 2026 trial of autonomous agents in enterprise data engineering with business outcome measures.
- Risk categories follow the OWASP Top 10 for Agentic Applications (December 2025) and the U.S. and allied joint guidance on agentic AI (May 2026). Controls draw on Anthropic, Google, OWASP, NIST, and that guidance<sup>88 92 94 97</sup>.
- Results from Keyhole's own engagements are labeled as ours and are self-reported.

# References

1. LangChain (Vivek Trivedy), "The Anatomy of an Agent Harness," March 10, 2026.
2. Birgitta Böckeler, martinowler.com, "Harness Engineering for Coding Agent Users," April 2, 2026.
3. LangChain (Vivek Trivedy), "Improving Deep Agents with Harness Engineering," February 17, 2026.
4. Arena (Pan, Yang, Arabzadeh, Chiang, Stoica, Zaharia), "HarnessTax: How Much Does the Harness Matter for Coding Agents?" September 16, 2026 (updated September 18, 2026).
5. Gartner, "Gartner Predicts Over 40% of Agentic AI Projects Will Be Canceled by End of 2027," June 25, 2025.
6. Gartner, "Gartner Announces Top Predictions for Data and Analytics in 2026," March 11, 2026.
7. HFS Research, "HFS Horizons: Agentic Services, 2026," April 2026 (reprint hosted by KPMG).
8. Simon Willison, "I Think 'Agent' May Finally Have a Widely Enough Agreed Upon Definition to Be Useful Jargon Now," September 18, 2025.
9. OpenAI (Michael Bolin), "Unrolling the Codex Agent Loop," January 23, 2026.
10. Anthropic, "Agent SDK Overview," Claude Code documentation, accessed October 7, 2026.
11. Geoffrey Huntley, "Ralph Wiggum as a 'Software Engineer'," July 14, 2025 (updated February 19, 2026).
12. OpenAI (Ryan Lopopolo), "Harness Engineering: Leveraging Codex in an Agent-First World," February 11, 2026.
13. Agent Skills, "Agent Skills: An Open Format for Extending AI Agent Capabilities," accessed October 7, 2026.
14. Anthropic, "Donating the Model Context Protocol and Establishing the Agentic AI Foundation," December 9, 2025.
15. Anthropic Engineering, "Effective Context Engineering for AI Agents," September 29, 2025.
16. Anthropic Engineering, "How We Built Our Multi-Agent Research System," June 13, 2025 (updated January 6, 2026).
17. GitHub Docs, "Customizing the Development Environment for Copilot Coding Agent," accessed October 7, 2026.
18. Anthropic Engineering, "Scaling Managed Agents: Decoupling the Brain From the Hands," April 8, 2026.
19. Microsoft, "Microsoft Agent Framework," GitHub repository, accessed October 7, 2026.
20. Amazon Web Services, "Amazon Bedrock AgentCore Is Now Generally Available," October 13, 2025.
21. Merrill, Shaw, Carlini, et al., "Terminal-Bench: Benchmarking Agents on Hard, Realistic Tasks in Command Line Interfaces," arXiv:2601.11868, January 2026.
22. The New Stack (Janakiram MSV), "Aider, Claude Code, and OpenClaw Ran an Identical Model. Token Use Varied 70-Fold," August 27, 2026.
23. Keyhole Software (David Pitt), "Agentic AI Delivery in Practice: Autonomous Enterprise Execution with the Ralph Loop," March 24, 2026.
24. Keyhole Software (Zach Gardner), "AI Coding Tools in Enterprise Software Delivery: An Architect's Workflow," March 16, 2026.
25. OpenAI, "Codex," GitHub repository, accessed October 7, 2026.
26. OpenAI, "OpenAI Agents SDK (Python)," GitHub repository, accessed October 7, 2026.
27. Google Developers Blog (Dmitry Lyalin and Taylor Mullen), "An Important Update: Transitioning Gemini CLI to Antigravity CLI," May 19, 2026.
28. Google, "Agent Development Kit (Python)," GitHub repository, accessed October 7, 2026.
29. GitHub, "Octoverse: A New Developer Joins GitHub Every Second as AI Leads TypeScript to #1," October 28, 2025.
30. Cursor (Roshan Sadanani), "Build Programmatic Agents With the Cursor SDK," April 29, 2026.
31. Anomaly, "OpenCode," GitHub repository, accessed October 7, 2026.
32. Mario Zechner, "What I Learned Building an Opinionated and Minimal Coding Agent," November 30, 2025.
33. LangChain, "Deep Agents," GitHub repository, accessed October 7, 2026.
34. Strands Agents, "Strands Agents SDK (Python)," GitHub repository, accessed October 7, 2026.
35. Stack Overflow, "2025 Developer Survey: AI," July 2025.
36. Google (Ryan J. Salva), "How Are Developers Using AI? Inside Our 2025 DORA Report," September 23, 2025.
37. Stack Overflow (Erin Yepis), "Agents on a Leash: Agentic AI Remains Mostly Monitored at Work," May 27, 2026.
38. JetBrains Research (Mikhail Bogdanov), "Which AI Coding Tools Do Developers Actually Use at Work?" April 2, 2026.
39. Accenture, "Accenture Expands AI Refinery and Launches New Industry Agent Solutions to Accelerate Agentic AI Adoption," March 18, 2025.
40. Accenture, "Accenture Introduces Trusted Agent Huddle to Allow Seamless, First-of-its-Kind Multi-System AI Agent Collaboration Across the Enterprise," April 28, 2025.
41. Anthropic, "Accenture and Anthropic Launch Multi-Year Partnership," December 9, 2025.
42. Deloitte, "Deloitte Unveils Zora AI, Agentic AI for Tomorrow's Workforce," March 18, 2025.
43. Anthropic, "Deloitte and Anthropic Partnership," October 6, 2025.
44. McKinsey QuantumBlack, "Seizing the Agentic AI Advantage," June 13, 2025.
45. BCG, "The Agentic Leadership Playbook: A Scaling Strategy for CTOs and CIOs," July 8, 2026.
46. PwC, "PwC Launches AI Agent Operating System to Revolutionize AI Workflows for Enterprises," March 27, 2025.

47. PwC, "Anthropic and PwC Expand Alliance, Driving Impact Across Client Work and the Firm," May 14, 2026.
48. Accounting Today, "KPMG Launches Multi-Agent AI Platform Workbench," June 17, 2025.
49. Anthropic, "KPMG and Anthropic Global Alliance," May 19, 2026.
50. EY, "EY Launching EY.ai Agentic Platform, Created With NVIDIA AI," March 18, 2025.
51. IBM, "IBM Launches Enterprise Advantage Service to Help Businesses Scale Agentic AI," January 19, 2026.
52. Capgemini, "Agentic Control and Management," April 2026.
53. Infosys, "Launching Today: Infosys Topaz Fabric," November 3, 2025.
54. Business Standard, "Infosys Partners With Anthropic for AI Solutions With Focus on Agentic AI," February 17, 2026.
55. Tata Consultancy Services, "TCS Launches TCS ADD AgentHub," August 17, 2026.
56. Anthropic, "Cognizant Partnership," November 4, 2025.
57. Cognizant, "Cognizant Ignition: AI-Orchestrated Data Value Chain," January 23, 2026 (updated June 22, 2026).
58. Thoughtworks, "Thoughtworks Launches Agent/works," June 16, 2026.
59. Computerworld, "OpenAI Partners With Consulting Giants to Deploy Enterprise AI Agents," February 23, 2026.
60. Forrester, "The State of Agentic AI in 2026: Companies Are Chasing, Few Are Catching," June 2026.
61. Fivetran (Wakefield Research data), "What Wasting Data Engineering Talent Really Costs You," March 31, 2022.
62. Fivetran, "Fivetran Report Finds Nearly Half of Enterprise AI Projects Fail Due to Poor Data Readiness," May 13, 2025.
63. Gartner, "Lack of AI-Ready Data Puts AI Projects at Risk," February 26, 2025.
64. Astronomer, "Astronomer Releases State of Apache Airflow 2026 Report," January 22, 2026.
65. dbt Labs (Elaine Green), "New dbt Labs Report Finds AI-Driven Acceleration Is Outpacing Trust and Governance," April 14, 2026.
66. Google Cloud (Tim Bezold and Varun Chandra), "Exploring the Data Engineering Agent in BigQuery," November 4, 2025 (updated April 22, 2026).
67. Databricks (Wendell, Zaharia, Hutchins, Oshri), "Introducing Genie Code," March 11, 2026.
68. Snowflake, "Snowflake Unveils Cortex Code, an AI Coding Agent," February 3, 2026.
69. Snowflake Documentation, "Cortex Agents Coding Agent (General Availability)," August 26, 2026.
70. dbt Labs, "dbt Agents, Remote dbt MCP Server: Trusted AI for Analytics," October 22, 2025.
71. Informatica, "Introducing Agentic, Goal-Driven Data Management With CLAIRE GPT," November 25, 2025.
72. Alation, "Alation Announces Agent Builder, Delivering Enterprise-Grade AI Agents," October 1, 2025.
73. Datadog, "Datadog Launches MCP Server to Provide AI Agents With Secure, Real-Time Access to Unified Observability Data," March 9, 2026.
74. Monte Carlo, "Monte Carlo Observability Agents," April 17, 2025.
75. Atlan, "How Atlan MCP Server Builds Context for Your AI Tools," August 22, 2025 (updated June 30, 2026).
76. Keyhole Software, "St. Louis Healthcare Data Platform Modernization & Streaming Integration," February 20, 2026.
77. Google Cloud Documentation, "Best Practices for Securing Agent Interactions With Model Context Protocol," updated September 24, 2026.
78. Cui, Demirer, Jaffe, Musolff, Peng, Salz, "The Effects of Generative AI on High-Skilled Work: Evidence From Three Field Experiments With Software Developers," Management Science, 2026.
79. METR, "Measuring the Impact of Early-2025 AI on Experienced Open-Source Developer Productivity," July 10, 2025.
80. METR, "We Are Changing Our Developer Productivity Experiment Design," February 24, 2026.
81. Faros AI, "The AI Productivity Paradox Report 2025," July 23, 2025.
82. InfoQ, "DORA 2025: State of AI-Assisted Software Development," September 2025.
83. Wang, Shao, Shaikh, Fried, Neubig, Yang (Carnegie Mellon University and Stanford University), "How Do AI Agents Do Human Work? Comparing AI and Human Workflows Across Diverse Occupations," arXiv:2510.22780, 2025.
84. Keyhole Software, "AI-Accelerated Development in Practice: A Repeat Client Modernization Story," February 2026.
85. Fortune, "AI-Powered Coding Tool Wiped Out a Software Company's Database in 'Catastrophic Failure'," July 23, 2025.
86. NIST, "Technical Blog: Strengthening AI Agent Hijacking Evaluations," January 17, 2025.
87. Simon Willison, "The Lethal Trifecta for AI Agents: Private Data, Untrusted Content, and External Communication," June 16, 2025.
88. OWASP GenAI Security Project, "LLM01:2025 Prompt Injection."
89. The Register, "Postmark MCP Server Code Hijacked," September 29, 2025.
90. Invariant Labs, "MCP Security Notification: Tool Poisoning Attacks," April 2025.
91. The Register, "Compromised Amazon Q Extension Told AI to Delete Everything – and It Shipped," July 24, 2025.
92. OWASP GenAI Security Project, "OWASP Top 10 for Agentic Applications for 2026," December 9, 2025.
93. Gartner, "Gartner Predicts AI Inference Costs Per Agentic Workflow Will Increase More Than Fivefold Through 2028," August 17, 2026.
94. Crowell & Moring, "American and Allied Cyber Agencies Issue First Joint Guidance on Securing Agentic AI," May 19, 2026 (summarizing "Careful Adoption of Agentic AI Services," May 1, 2026).
95. OpenTelemetry, "Semantic Conventions for Generative AI," GitHub repository, accessed October 7, 2026.
96. Paubox (Dawn Halpin), "AI Agents Are Reaching Into Patient Data Faster Than the Rules Can Keep Up," June 18, 2026.
97. Google (Díaz, Kern, Olive), "Google's Approach for Secure AI Agents," 2025.
98. KPMG, "AI Quarterly Pulse Survey, Q2 2026," June 24, 2026.

99. Deloitte Insights (Andy Bayiates), "Business and IT Leaders Report AI Agents Are Scaling Faster Than Their Guardrails," April 24, 2026.
100. McKinsey & Company (Tinkoff, Van der Veken, Chui, Balakrishnan), "The State of AI in 2026: On the Road to ROI," August 25, 2026.
101. Anthropic, "Our Framework for Developing Safe and Trustworthy Agents," August 4, 2025.
102. Keyhole Software, "Agentic AI Software Development Services."
103. Keyhole Software, "Inside Anthropic's Emerging Partner Ecosystem: What Keyhole Is Seeing and Applying in Enterprise AI," April 3, 2026.



— NEXT STEP

# Put agents to work inside the controls you already have.

Schedule an architecture and roadmap session with Keyhole's senior architects. We'll map the harness to your stack, settle the governance model, and pick a first use case worth proving.

PHONE

(877) 521-7769

EMAIL

asktheteam@keyholesoftware.com

WEB

keyholesoftware.com/contact

OFFICES

Kansas City (HQ) · St. Louis · Denver · Dallas



SCAN TO BOOK

[Schedule an architecture session](#)