AI agents in production, 2026: the grader decides who is shipping

Large companies say they are scaling agents. The runs that can be graded are the ones with a score, a harness, and a bill.

Last updated November 10, 2026

Executive summary

In November 2026 an agent is in production when a company can point at a repeated workflow, a harness, and a way to fail the run. A survey answer that says 'we are scaling agents' is a different object. This report keeps the two apart.

McKinsey's Global Survey, fielded May 4 to June 8, 2026, has the cleanest split. Among 1,719 respondents in 97 countries, 40 percent of people at organizations with more than $1 billion in revenue said they were scaling AI agents, up from 27 percent a year earlier. Respondents at smaller organizations stayed at 22 percent. About one in five respondents said their organization was scaling software coding agents, 31 percent at the larger enterprises. Over that same year, the share attributing any EBIT impact to AI stayed at 37 percent, and the share McKinsey calls high performers stayed at about 6 percent. Eighty percent said AI had improved their own productivity. Individual output and company profit are different measurements. [1]

Menlo Ventures' November 3 estimate says the money moved anyway. Enterprise deployment spend in their model reached $176.8 billion by September 2026, from $33 billion in 2025. On OpenRouter, they report agents generating 80 percent of tokens. Coding and chatbots, $34.0 billion and $25.0 billion, dominate a $77.3 billion application market in the same model. Those are Menlo's figures, built from a survey, filings, and third-party traces. They are not an audited industry census. [2]

The public evals agree with the spending pattern and refuse the victory lap. In the SWE-bench Verified bash-only copy stored here on November 5, the lead row is mini-SWE-agent 2.0.0 with Claude 4.5 Opus at high effort: 76.8 percent resolved, listed cost $376.95, result date 2026-02-17. On Terminal-Bench 4.0 the lead row is a different harness, Claude Code, with Opus 5.5 at max effort: 64.85 percent, 214 of 330 trials, listed cost $4,718.65. Hold the harness at Codex and the accuracy at 58.18 percent and the listed run cost is still $3,267.18 for GPT-6 Astra against $634.24 for GPT-6.1 Sol, 5.2 times the money for the same printed score. [4]

Computer use is not that chart. The OSWorld 2.0 page, as copied for this site on November 9, prints 20.6 percent binary completion for Claude Opus 4.8 on the 500-step primary metric, with a 54.8 percent partial score and 244K output tokens on that row. Andreessen Horowitz's August 10 note cites OSWorld-Verified moving from 42 percent to 85 percent, above a human figure of about 72 percent on that board. Same family of tasks, different tests, and neither number is a contract. [5][6]

The claim this report is willing to defend: fund the agent where a test, a diff, or a closed runbook can mark the run wrong. A seat on a chatbot, which McKinsey still finds to be the tool most often described as scaled, is a different purchase.

Scaling is a survey answer

McKinsey has run a global AI survey for the better part of a decade. The 2026 wave was online, in the field from May 4 to June 8, and it collected 1,719 responses across 97 countries. Responses are weighted by each country's contribution to global GDP, which keeps a small country's reply from counting like a large one's. Thirty-six percent of respondents said they work at organizations with more than $1 billion in annual revenue. The people answering are not a random sample of employees. They are the people McKinsey could reach who would fill in a survey about their company's AI use. Read every percentage below as 'share of those

respondents,' and read 'large' as that revenue cut. [1]

Inside that frame, the agent result is a size result. Forty percent of respondents from large organizations said they were scaling AI agents, up from 27 percent in the prior wave. The share at smaller organizations was 22 percent in both waves. McKinsey's own sentence is the right one: the increase in agentic AI is mostly accounted for by large enterprises. A vendor who quotes '40 percent of companies are scaling agents' has dropped the only clause that makes the number true.

The same survey says what else is being scaled, and the order matters. Chatbots are the tool most often described as scaled across the enterprise, at 47 percent of respondents. AI agents, and software coding agents specifically, each sit at about two in ten overall. Coding agents reach 31 percent among the large enterprises. So the modal 'scaled AI tool' in this survey is still a chatbot. The agent number is real, and it is not the majority experience. Treating the chatbot row and the agent row as one adoption story is how a buyer ends up purchasing a chat window and calling the rollout agentic.

McKinsey defines high performers narrowly: respondents who attribute at least 5 percent of EBIT to AI use and who describe the impact as significant. That group is about 6 percent of respondents, flat with last year. They are twice as likely as other respondents to say they are scaling software coding agents, and 2.7 times as likely to say they are scaling other agentic AI. They are also the group that feels the bill. High performers report being constrained by costs on software coding agents about three times as often as others, while the cost constraint on other tools looks similar between the groups. The firms that have pushed coding agents furthest are the firms reporting that tokens are now a design constraint. Cost is showing up in the successful deployments, not only in the stalled ones.

Two other buying facts sit next to that. Nearly a third of respondents, 32 percent, said their organization had decided against buying at least one software product or feature because it could be built in-house with agentic coding tools. Among high performers the share is nearly half, against 31 percent of other respondents. And about 20 percent of all respondents said AI operating costs, including token costs, had constrained their AI use, with about one in ten saying that for each of chatbots, agents, and coding agents separately. The build-versus-buy shift and the cost constraint are the same phenomenon seen from purchasing and from finance.

The profit line did not move with any of this. Thirty-seven percent of respondents attribute at least some EBIT impact to AI, about the same as last year. Eighty percent say AI improved their own productivity, and about half say it helped them make better decisions. Mid-level managers and individual contributors are more likely than executives to report a negative effect: 47 percent against 31 percent. A program can be popular with the person using it, uncomfortable for the person under the executive, and invisible in EBIT. Those three sentences are allowed to be true together.

"Just 14 percent of respondents from organizations using AI report that AI contributed to an overall decline in workforce size in the past year-less than half the 32 percent who, in last year's survey, expected workforce reductions over the same period." McKinsey, The state of AI in 2026. [1]

This year's expectation moved the other way. Thirty-nine percent expect an AI-related decline in total employment over the coming year, up from 32 percent, and 43 percent expect no AI-related change. The quote above is the reason this report does not treat 39 percent as a forecast of layoffs. The same question, asked of a similar instrument, overshot the reported outcome last time. Headcount is picked up again in the 2027 forecast, where it is labeled as a forecast. Here it is only the track record of the question.

- Buyers: If your company is under $1 billion in revenue, 22 percent is the base rate this survey gives you, and it did not rise. If you are over that line, ask which workflow is being scaled, what marks a run as failed, and what the retry budget is. 'Scaling' in this questionnaire is the respondent's choice of phase. It is not a success rate. The high-performer cut is the useful tell: the respondents who say coding agents are scaled are also the ones saying the token bill is now in the way.

Respondents saying their organization is scaling AI agents, by company size. Large means annual revenue above $1 billion.

	Prior-year survey	2026 survey
Large enterprises	27%	40%
Smaller organizations	22%	22%

The dollars followed work that can be checked

Menlo's fourth annual enterprise report, published November 3, 2026, is an estimate with a method, which is better than a round number and worse than a 10-K. Morning Consult fielded the underlying survey from September 8 to September 26, 2026, to nearly 400 U.S. enterprise decision-makers whose organizations represent more than $4 billion in AI spending. Dollar totals for the market are Menlo's bottoms-up model on top of those answers, public filings, and third-party reports. Revenue is an annualized run rate. They reclassified prior years so that data-labeling vendors such as Mercor sit in development rather than deployment. Anyone comparing their 2025 bar to a chart from last year's PDF should use this year's rebased figures, not the old ones. [2]

On that basis they put enterprise deployment spend at $1.7 billion in 2023, $8.5 billion in 2024 (5.0 times), $33 billion in 2025 (3.9 times), and $176.8 billion by September 2026 (5.4 times the 2025 figure). The round '$177 billion' in their headline is that $176.8 billion. Spend on building models, in the same model, is $102.5 billion, up 5.3 times. Their ratio is about $1.70 of deployment spend for each dollar going into building models. Fifty-five percent of the AI dollars, they estimate, is net-new spend rather than a transfer out of existing software or labor budgets.

The application split is the part that should change a roadmap. In Menlo's model the application market is $77.3 billion. Coding is $34.0 billion and chatbots are $25.0 billion, which they describe as 76 percent of that market. The labs' own coding agents grew from $1 billion to $24 billion, and the rest of the coding market still grew 2.9 times to $10 billion. Coding as a category is 96 percent AI-native in their taxonomy. AI assistants are $9.5 billion and healthcare is $3.7 billion. Incumbents, they say, are taking share in older software categories: IT at 91 percent, data science at 88 percent, HR at 68 percent. New agent categories and old system-of-record categories are not moving in the same direction.

One router, not the market. Menlo reports that on OpenRouter, agents generate 80 percent of tokens, 130.9 trillion a week, roughly seven times human-driven usage, and that agent traffic is up 37 times since it overtook human traffic in February. OpenRouter is a multi-model router used heavily by developers. It is not the enterprise procurement market, and it is not ChatGPT. Use the 80 percent as evidence that machine-driven loops dominate one large developer pipe. Do not use it as the share of corporate AI spend.

Model share, in their survey-weighted read of production API usage, moved toward the labs that are winning coding. Frontier labs are 69 percent of deployment dollars, up from 50 percent. Anthropic's share of enterprise API spend went from 40 percent to 55 percent, OpenAI's from 27 percent to 31 percent, and Google's from 21 percent to 9 percent. Those three named shares add to 95 percentage points. This page does not assign the remaining points, because the sentences used here do not. Open-source models are a separate cut: 19 percent of usage, up from 11 percent, and 4 percent of spend. Usage share and dollar share diverge, which is what you expect when the cheap model handles the high-volume call and the expensive model handles the call someone will grade.

Menlo's explanation for the concentration is the one this report agrees with, because the evals later in the page show the same shape. Spend concentrates where an agent can act, where the workflow repeats, and where the result can be checked. Coding has a test suite. A chatbot session

has a user who either continues or does not. A general 'AI employee' has neither, until someone builds the grader. The $3
to the word agent. It is a compliment to the unit test.

- Builders and buyers: A product whose demo is a general employee is aimed at the slice of Menlo's app market that is not coding and not chat. That remainder, on their numbers, is about $18 billion, and it includes categories where they say incumbents are gaining. Price the company against that slice, not against $177 billion. And if a board slide quotes Anthropic at 55 percent of enterprise API spend, the footnote is Menlo's survey-weighted estimate, not a stack of invoices.

Enterprise AI deployment spend in Menlo Ventures' bottoms-up model. 2026 is the figure through September, shown as an annualized run rate.

	Deployment spend
2023	$1.7B
2024	$8.5B
2025	$33B
Sep 2026	$176.8B

Coding agents are ahead because the grader already exists

SWE-bench Verified is 500 real GitHub issues. A row's score is the percent of issues whose tests passed. The bash-only slice stored on this site holds the harness still: every row is mini-SWE-agent, so the chart below is a model comparison inside one loop, not a comparison of Claude Code, Codex, and a vendor demo. The SWE-bench project says release 2.x calls tools, while release 1.x reads an action out of the model's text, and that the two generations are not necessarily comparable. The lead rows in our November 5 copy are mini-SWE-agent 2.0.0. They are tool-calling runs. [4]

The lead row is Claude 4.5 Opus (high): 76.8 percent resolved, listed cost $376.95, result date 2026-02-17, reasoning effort high. The next rows in that copy, same harness and same date, are Gemini 3 Flash (high) at 75.8 percent (listed cost $177.98) and MiniMax M2.5 (high) at 75.8 percent (listed cost $36.64). The cluster is tight. A two-point gap on 500 issues is 10 issues. It is not a generation of product advantage, and it is not a reason to rewrite a stack.

If the listed cost on the lead row is the cost of the full 500-issue run, the arithmetic is about 75 cents per issue ($376.95 / 500). This page does not have a separate per-issue invoice, so that figure stays an if. It is still the right order of magnitude to hold in your head: a minimal tool-calling harness, on a graded software task, was not a multi-dollar event per issue on that February row. Terminal-Bench, below, is a different order of magnitude because the harness and the task are different. Quoting 'coding agents cost 75 cents' without the harness name would be the mistake this section exists to prevent.

Stanford's 2026 Index is worth reading beside this table, because the Index is where a careless summary will come from. The top takeaway says performance on SWE-bench Verified rose from 60 percent to near 100 percent of meeting the human baseline in a single year. The coding section of the same report, discussing the state of the board as of February 2026, says the top models are clustered in the low-to-mid 70s and names Claude 4.5 Opus (high reasoning) at approximately 76.8 percent, with Kimi K2.5, GPT-5.2, and Gemini 3 Flash (high reasoning) between 70 and 76 percent. That 76.8 percent is the same lead this site's November 5 copy stores for mini-SWE-agent. The Index also says, in a chapter opener, that AI agents are improving but still fail roughly one in three attempts. A 76.8 percent resolve rate leaves 23.2 percent unresolved on that row, which is in the neighborhood of 'one in three' only if you round hard and forget that the row is one model plus one harness. [3][4]

Do not promote the baseline sentence. This page does not have Stanford's definition of the human baseline in a form that turns 76.8 percent resolved into 'near 100 percent.' We report the resolve rate, the harness, and the date. The baseline sentence can stay in the Index, where it is attached to whatever figure 2.1.1 is actually plotting.

The practical consequence for a coding-agent buyer is that the public number is a property of a triple: model, harness, effort. mini-SWE-agent is a

minimal loop. Claude Code and Codex, which show up on Terminal-Bench rather than in this bash-only slice, carry their o compaction, and their own retry policy. A model card that prints a SWE-bench score without the harness is borrowing the cannot reconstruct. The November harness benchmark keeps the full tables, including the team-checked rows that use ot multilingual set, which is a different task list and must not be averaged with these 500.

- Builders: Publish the harness version, whether the loop calls tools or parses text, the effort setting, and the grader next to the percent. Buyers should write the eval set into the contract and should refuse a SWE-bench number that does not name the agent. The interesting gap in this table is not between 76.8 and 75.8. It is between a minimal tool-calling harness that already clears three quarters of a graded issue set and a production repo whose tests, flaky CI, and private code the public board has never seen.

Percent of SWE-bench Verified issues resolved. Every row is mini-SWE-agent 2.0.0, the tool-calling generation. Release 1.x parses text and is left off this chart. Verified is 500 issues.

	Resolved
Claude 4.5 Opus	76.8%
Gemini 3 Flash	75.8%
MiniMax M2.5	75.8%
Claude 4.6 Opus	75.6%
GLM 5	72.8%
GPT 5.2	72.8%

The harness sets the bill

Terminal-Bench 4.0 is the current board linked from tbench.ai. Unlike the bash-only SWE-bench slice, the agent column changes by row. The copy stored here was read November 5, 2026. The board's own updated-at stamp in that copy is 2026-10-23. Accuracy and the 95 percent half-width are the leaderboard's fields. The cost cell is copied as printed, including a partial-trial note where the board includes one. Rows with a partial note are left out of the cost chart, because a partial bill is not comparable to a full one. [4]

The lead row is Opus 5.5 with Claude Code at effort max: 64.85 percent, half-width 3.11, 214 successes out of 330 trials, listed cost $4,718.65, result date 2026-09-22. Sonnet 5.5 with the same agent and effort scores 61.82 percent (204/330) at a listed cost of $7,333.64. Inside Claude Code, the higher score is the cheaper listed run. That single comparison is enough to retire 'you have to buy the most expensive setting.'

The clean price comparison holds the harness still. GPT-6 Astra with Codex at effort max scores 58.18 percent, 192 of 330, listed cost $3,267.18, date 2026-09-03. GPT-6.1 Sol with the same agent and the same effort scores 58.18 percent, 192 of 330, listed cost $634.24, date 2026-09-29. Same printed accuracy. The Astra run is listed at 5.2 times the Sol run. Dividing listed cost by recorded successes gives $17.02 per success for Astra and $3.30 for Sol. Opus 5.5 with Claude Code comes out at $22.05 per recorded success, on a higher accuracy and a different harness, so it does not enter a ranking against Codex.

Why would two Codex runs with the same accuracy and the same effort label differ by a factor of five? This page does not have the traces, so it will not invent a cause. The candidates a harness engineer would actually inspect are real: how much tool output is kept in the window, whether a failed command is retried with the full history or a compacted one, whether the effort setting spends reasoning tokens before the first shell call, and whether the run caches an environment that the other run rebuilds. That inspection is context engineering. It is the decision of what occupies the window on turn n, given the tool results from turns 1 through n-1. It is not a slogan for 'write a better prompt.'

McKinsey's cost result lands in the same place from the buyer's side. The respondents most likely to say they are scaling coding agents are about three times as likely to say costs constrained that use. A team that is not yet scaling will not see the bill, because the bill is produced by retries. A dashboard that tracks tokens per seat, or cost per employee per month, averages the human who asked one question together with the loop that took forty tool calls to go red and then green. Cost per successful task is the unit that matches the way the board already reports the work: a cost, a

success count, and a trial count, on the same row.

- Buyers: Ask the vendor for cost per successful task on your eval, with the harness version pinned. A token-price cut on the model card does not answer the question the Terminal-Bench rows raise. Inside one agent, Codex, the listed bill moved by a factor of five at a fixed score. Builders should log successes, retries, input tokens, and output tokens on the same span, and should be able to say which of those moved when the bill moved.

Cost cell copied as printed on Terminal-Bench 4.0, for four rows with a complete cost (no partial-trial note). The agent is not the same on every row.

	Listed run cost
Opus 5.5 + Claude Code	$4,718.65
Sonnet 5.5 + Claude Code	$7,333.64
GPT-6 Astra + Codex	$3,267.18
GPT-6.1 Sol + Codex	$634.24

Computer use is still a runbook

A coding agent fails in a way a test can see. A computer-use agent fails in a way a screenshot often cannot. The OSWorld 2.0 numbers this site copied on November 9 make that concrete, and they have to be quoted with the setting, because the page prints different settings on different rows. Claude Opus 4.8, maximum thinking, batched tool calls, 500-step primary metric: 20.6 percent binary completion, 54.8 percent partial score, 244K output tokens. Claude Opus 4.7, maximum thinking, batched tool calls: 18.2 percent binary. GPT-5.5 is printed as about 14 percent binary and 39K output tokens, on a row the page describes as the more token-efficient run. Partial credit is not a finished task. A 54.8 percent partial score next to a 20.6 percent binary score means the run often got somewhere and did not finish. [5]

Andreessen Horowitz, on August 10, 2026, describes a different board. They write that the best computer-using model a year earlier scored 42 percent on OSWorld-Verified and that the best score at the time of their note was 85 percent, above the roughly 72 percent humans manage on those tasks. They also write, from conversations with teams running computer-use workflows, that protocol-following tasks work and that agents break when the work leaves the runbook. The founder quote they include says the models were not good enough to use in production on their own until Opus 4.6 in February 2026. Their production examples are the long tail of software without an API: CRM updates, portal logins, ticket handling, record checks. [6]

Two boards, two numbers. 20.6 percent binary on OSWorld 2.0 and 85 percent on OSWorld-Verified cannot be averaged, and neither can be subtracted from a SWE-bench resolve rate. The task, the harness, the step limit, and the grader are all different. Quote the board's name or do not quote the percent.

The a16z piece and the OSWorld 2.0 binary rate are compatible once you stop treating them as a horse race. A narrow, cached, checkable UI path can be in production while a general desktop benchmark is still failing four times in five on a binary grader. That is the shape of the category. The browser and computer-use report lists the ten live products and the one vendor date through May 9, 2027. This page does not repeat that table. What it adds is the evaluation point: binary completion on the OSWorld 2.0 primary metric we copied is still about one in five for the best printed row, at a token cost (244K output tokens on that Opus 4.8 row) that makes an unattended retry loop expensive fast.

Tool calling is the shared mechanism, and it does not rescue the category by itself. A computer-use model emits a click, a keystroke, or a tool call. The harness applies it in a sandbox, observes the next screen, and decides whether to continue. MCP, the Model Context Protocol, is one way to expose those tools to the model through a schema. Standardizing the socket is worth doing. It does not grade the screen. A buyer who mandates MCP and stops there has standardized the plug, not the outcome. The glossary entries for tool use, harness, and MCP keep the definitions short. The

engineering point is the one in the previous section: the window fills up with observations, and somebody has to decide wh

- Builders and buyers: Ship computer use as a runbook with a check at the end: a known UI, a cached path, a human on the step that moves money or deletes data. If the task has no stable screen and no API, the 20.6 percent binary row is the relevant public number, not the 85 percent on a different board. Do not staff a general desktop employee on the evidence in this report.

What builders actually install

Surveys say what respondents claim. Package registries say what a build machine requested. They are not the same, and the registry is the less flattering one, which is why it belongs here. The download snapshot on this site was retrieved 2026-11-09. The PyPI window in it runs from 2026-10-10 through 2026-11-08, counted without mirrors. The top of that window is langchain at 155,918,257 file requests, then langgraph at 55,093,579, strands-agents at 37,592,185, claude-agent-sdk at 33,887,233, pydantic-ai-slim at 32,785,681, and openai-agents at 17,426,915. A download is a file request. CI, mirrors we failed to exclude, and dependency installs all count. It is still a better picture of the harness layer than a logo wall. [7]

Two cautions belong in the same breath as the chart. pydantic-ai-slim and pydantic-ai are separate packages, so their bars are not additive users. And the npm package `ai`, the Vercel AI SDK, is in the same snapshot at 140,196,706 file requests over the same window. It is left off the agent chart. That package is a model SDK. A download of it is not evidence that an agent loop was installed. The npm packages that are agent loops, over the same window, are led by @langchain/langgraph (15,932,146), @mastra/core (9,988,333), @openai/agents (8,171,695), @google/adk (1,085,245), and undefined (0).

Stars are the attention series, pulled separately on 2026-11-09. LangChain (langchain-ai/langchain) leads that snapshot at 149,267 stars, then AutoGen at 61,760 and CrewAI at 60,690. Stars do not convert to the PyPI bars at a stable rate, and they should not be asked to. LangChain's star count and its download count are both large. A repository can be starred and quiet, or downloaded constantly by CI and rarely starred. The stars tracker and the downloads tracker are the live pages. [8]

Release recency is a third pull, on 2026-11-05, of 19 open-source frameworks and SDKs. 14 of the 19 had a latest GitHub release within 30 days of that pull. The list, the licences, and the day counts are on the framework release report. The point for this page is narrower. The harness layer builders import is being shipped against, not abandoned. A production agent in 2026 is usually one of these loops, or a vendor harness such as Claude Code or Codex, wrapped around a model, with the company's own tools on the side. It is rarely a new runtime invented for the demo.

- Builders: Pick the harness the way you pick a database: by what you will have to operate. LangChain and LangGraph dominate file requests. Claude Agent SDK, OpenAI Agents SDK, Google ADK, CrewAI, Pydantic AI, Mastra, and Strands are the other names with a real download trail in this window. Stars are a reason to read the repo. They are not a reason to adopt it. If you publish an agent framework, report downloads and a pinned eval, not a star-count screenshot.

PyPI downloads without mirrors, 2026-10-10 through 2026-11-08. A download is a file request, not an install and not a user.

	Downloads
langchain	155.9 million
langgraph	55.1 million
strands-agents	37.6 million
claude-agent-sdk	33.9 million
pydantic-ai-slim	32.8 million
openai-agents	17.4 million

What to fund

Put the pieces in one purchase order. From McKinsey: agent scaling rose where the company was already large, coding agents are the agent type with a measurable build-versus-buy effect, profit attribution did not move, and the teams that scaled coding agents are the ones reporting the cost constraint. From Menlo: the deployment dollars in their model concentrated in coding and chat, on a router the machine-driven share of tokens is already the majority, and open-source usage is far ahead of open-source spend. From the boards: a minimal tool-calling harness clears about three quarters of SWE-bench Verified, a heavier coding harness on Terminal-Bench has a listed bill that moves by a factor of five inside a single agent at a fixed score, and the best OSWorld 2.0 binary row we copied finishes about one run in five.

That is enough to fund three lines and to leave one unfunded. Fund a coding agent where the repository has tests, and buy it on cost per green run with the harness pinned. Fund a computer-use runbook where the UI is stable and a person still approves the step that cannot be undone. Fund the harness work itself: compaction, retries, traces, and the eval that gates a change. Leave unfunded the general employee agent, the one whose pitch is that it will notice what to do. Nothing in the sources above is a measurement of that product. The chatbot seat, which is what 47 percent of McKinsey's respondents say is actually scaled, can be bought as a chatbot. Calling it an agent does not give it a grader.

The 12-month filing calendar is the page for dates a company or a government has already put on paper. This page is the cross-section of what is running. The 2027 forecast says what we think changes, and it says what would prove us wrong. List prices for the coding products sit on the 30-day pricing brief. They are a third unit, beside token price and cost per success, and they should not be mixed into the charts above.

- Buyers: Write the task class on the order: graded software work, checkable UI runbook, or chat. Ask for the harness version, the eval, the success definition, and the cost per success. A vendor who can answer those four is selling an agent. A vendor who answers with a seat price and a model name is selling a subscription.

Methodology

This page did not field a survey, recruit an expert panel, or interview companies. It copies published figures and recomputes nothing except the arithmetic it labels. There is no hidden sample of practitioners behind the sentences.

What was counted

McKinsey's figures are the shares printed in the August 25, 2026 survey report: scaling by company size, the roughly one-in-five and 31 percent coding-agent shares, the 47 percent chatbot figure, the 37 percent EBIT figure, the about-6-percent high-performer definition, the 80 percent productivity figure, the 32 percent and 'nearly half' build-versus-buy figures, the cost-constraint figures, and the 14, 32, and 39 percent workforce figures. 'Large' means annual revenue above $1 billion, which is McKinsey's cut. The survey is GDP-weighted. n is 1,719. [1]

Menlo's dollar figures are Menlo's estimates, annualized run rate, from the November 3, 2026 report. The $18.3 billion 'rest of app market' bar is this page's subtraction, $77.3 billion minus $34.0 billion minus $25.0 billion. Menlo describes coding and chatbots as 76 percent of that market. The subtraction is shown so the chart's three bars add to the total they named. It is not a segment they published under that label. The OpenRouter token shares are Menlo's account of OpenRouter, not a recount by this site. [2]

SWE-bench and Terminal-Bench figures are the rows in the November 5, 2026 copy used by the harness benchmark. The newest Verified result date in that copy is 2026-02-26. Cost-per-success is listed run cost divided by the successes field. The SWE-bench per-issue figure is listed cost divided by 500, stated as conditional on that cost being the full run. The Codex cost ratio is GPT-6 Astra's listed cost divided by GPT-6.1 Sol's listed cost, both at effort max. Rows whose cost cell mentions a partial trial are excluded from those divisions. [4]

OSWorld 2.0 figures are the three rows stored for the November 9 browser report, originally read off https://osworld-v2.xlang.ai/. They are quoted in the computer-use chapter with the setting printed on the row. They are not drawn on a chart, because 'about 14 percent' is not a point that belongs on the same axis as 20.6 and 18.2 without a label the bar cannot carry. Package counts and star counts are the snapshots retrieved 2026-11-09. Release recency uses the framework snapshot retrieved 2026-11-05T04:41:05.495715Z, with the same UTC-day distance the framework report uses, and a 30-day cutoff. [5][7][8]

What this does not measure

It does not measure how many agents are unattended in production. McKinsey measured what respondents said about scaling. Menlo estimated spend. The boards measured harnesses on public tasks. Registries measured file requests. None of those is a census of deployed agents. It does not measure quality of life, wages, or incidents. The Stanford human-baseline sentence is quoted and not converted into a resolve rate. Andreessen Horowitz's OSWorld-Verified numbers are quoted as theirs and are not merged with OSWorld 2.0. Deloitte's January 2026 enterprise survey is left out of the charts because its fieldwork was August and September 2025 and its questions are about AI programs in general. Using it as an agent-production rate would be a category error.

Forward-looking sentences are not in this report except as pointers. The workforce expectation, the Menlo growth rates, and the board scores are described as measurements with a date. What we think happens in 2027, and what would falsify it, is AI agents in 2027.

- Builders and buyers: If you cite this page, cite the source number, not 'AI Agents Info found.' We found the public record and did the division. The division is marked wherever it appears.

Sources

[1] McKinsey, The state of AI in 2026: On the road to ROI. Published August 25, 2026. Online survey in the field May 4 to June 8, 2026. PDF:
https://www.mckinsey.com/~/media/mckinsey/business%20functions/quantumblack/our%20insights/the%20state%20of%20ai/the-state-of-ai-in-2026-on-the-road-to-roi_final.pdf
https://www.mckinsey.com/capabilities/quantumblack/our-insights/the-state-of-ai

[2] Menlo Ventures, 2026: The State of Enterprise AI. Published November 3, 2026. PDF:
https://menlovc.com/wp-content/uploads/2026/11/menlo_ventures_enterprise_ai_report-2026-110326.pdf
https://menlovc.com/perspective/2026-the-state-of-enterprise-ai/

[3] Stanford HAI, Artificial Intelligence Index Report 2026. Ninth edition, April 2026. arXiv:2606.15708. PDF: https://hai.stanford.edu/assets/files/ai_index_report_2026.pdf
https://hai.stanford.edu/ai-index/2026-ai-index-report
[4] SWE-bench Verified and Terminal-Bench 4.0, as copied on this site. Copied November 5, 2026 for the harness benchmark. Terminal-Bench board: https://hub.harborframework.com/datasets/terminal-bench/terminal-bench/4?leaderboard=4-0-0&tab=leaderboard. SWE-bench Verified is 500 instances. The local copy's newest Verified result date is 2026-02-26. https://www.swebench.com/verified.html
[5] OSWorld 2.0 scores printed on the benchmark page. The three rows used here are the rows stored in the November 9, 2026 browser and computer-use report on this site. Settings are copied as printed. https://osworld-v2.xlang.ai/
[6] Andreessen Horowitz, Can agents use a computer yet?. Fabrizio Serafini, Seema Amble, and Eric Zhou, August 10, 2026. Their OSWorld-Verified figures are cited as theirs. https://a16z.com/can-agents-use-a-computer-yet-weve-got-the-data/
[7] PyPI and npm download snapshot on this site. Snapshot retrieved 2026-11-09. PyPI counts are file requests without mirrors, from pypistats. npm counts are file requests from the npm registry. A download is not a user. /trackers/downloads
[8] GitHub star snapshot and the November 5 framework-release pull. Stars retrieved 2026-11-09 from the GitHub REST API. Release recency is a separate pull at 2026-11-05T04:41:05.495715Z, published on the framework report. /trackers/stars