SCORED.TOOLS · ROUNDUP

Best AI Frameworks for Multi-Agent Systems in 2026

A no-hype ranking of the seven multi-agent frameworks worth building on in 2026, based on production readiness, orchestration quality, and total cost.

Picking a multi-agent framework in 2026 is less about which library has the shiniest demo and more about which one you'll still be debugging comfortably 18 months from now. *Picking an AI agent framework in 2026 feels less like choosing a library and more like choosing an architecture you'll live with for years*. The consolidation that started in 2025 is over: *the AutoGen / AG2 split, LangGraph 1.0 GA, Claude Code SDK launch (later renamed to Claude Agent SDK), CrewAI commercialization, and Pydantic AI V1 were the year's five biggest movers*, and *Microsoft Agent Framework 1.0 shipped on April 3, 2026 as the unified successor to Semantic Kernel and AutoGen, with native MCP and A2A protocol support for both .NET and Python*.

This roundup ranks seven frameworks by what actually matters in production: state management, observability, model flexibility, protocol support (MCP and A2A), and total cost of ownership. GitHub stars didn't decide anything.

How We Ranked These

We weighted four things: production readiness (checkpointing, retries, human-in-the-loop), orchestration quality (how well the framework handles complex handoffs and state), ecosystem maturity (integrations, observability, deployment tooling), and long-term cost. *What mattered: real production deployments, state management quality, observability options, model flexibility, protocol support (especially MCP), and total cost of ownership over a multi-year horizon*.

1. LangGraph — Score: 9.4/10

After synthesizing the strongest developer-focused research from early 2026 — production case studies, ecosystem data, protocol analyses, and TCO breakdowns — LangGraph is the best overall AI agent framework for serious developers right now. It's not the fastest to set up, but it's the one most likely to still be working correctly when your system hits real users, real edge cases, and real compliance reviews. Langgraph models workflows as state graphs where *nodes represent actions and edges define control flow. Each node takes in the current state, performs an operation, like*

calling a language model, querying a database, or executing custom logic, and returns an updated state. This structure keeps execution paths clear and easy to debug. On complex, multi-step tasks it also wins on raw accuracy: *complex tasks (8+ steps, planning required, backtracking expected) expose the architectural differences clearly. LangGraph completes 62% because its graph state machine handles failed nodes gracefully.*

Best for: Production systems with stateful, long-running workflows, compliance requirements, or heavy branching logic.

Pricing: Core framework is free (MIT). *LangGraph's LangSmith observability platform charges subscription fees starting at $39 monthly. LangGraph Professional starts at $99/month plus compute.*

2. CrewAI — Score: 8.8/10

Crewai is the fastest way to get a working multi-agent prototype in front of stakeholders. *CrewAI agents are autonomous entities with roles, goals, and tools. They make decisions about which tools to use and can communicate with other agents.* The trade-off is that you give up some visibility: *CrewAI uses a higher-level, declarative approach. Agents and tasks are defined in Python or YAML, then run in a Crew. Flows provide event-driven control with decorators. This reduces boilerplate but hides some execution details.* Adoption is real — *CrewAI is one of the fastest-growing multi-agent frameworks in production today, with over 52,000 GitHub stars and 2 billion agent executions in the past 12 months.*

Best for: Content pipelines, research workflows, and business automation where role-based collaboration maps naturally to the problem.

Pricing: Open-source core is MIT-licensed and free. *CrewAI offers a free tier with 200 agent runs per month, a Starter plan at $29/month for 1,000 runs, and a Professional plan at $99/month for 5,000 runs.* Enterprise is contact-sales.

3. OpenAI Agents SDK — Score: 8.5/10

Openai Agents Sdk is the minimalist pick. If you're already on OpenAI and want the thinnest possible abstraction, this is it. *The OpenAI Agents SDK offers the thinnest possible abstraction with four primitives and zero ceremony*. Unique strength: *the cleanest handoff model* in the category, plus *full streaming*. It's now genuinely multi-model — *the OpenAI Agents SDK now supports 100+ models via LiteLLM*.

Best for: Teams committed to the OpenAI ecosystem who want minimal framework overhead and clean agent-to-agent handoffs.

Pricing: SDK is free. You pay OpenAI API tokens.

4. Claude Agent SDK — Score: 8.3/10

Claude Agent Sdk takes a different design bet than the graph-based frameworks. *Anthropic's SDK takes a tool-use-first approach where agents are Claude models equipped with tools, including the ability to invoke other agents as tools. The architecture is deliberately simple: an agent loop receives a prompt, calls tools as needed (including sub-agent tools), and returns a structured response. Where other frameworks add abstraction layers, Anthropic keeps the loop minimal and relies on Claude's native capabilities for reasoning, planning, and coordination. What sets this SDK apart is the combination of native support for extended thinking (chain-of-thought visible in the API response), built-in handoffs between agents, and MCP (Model Context Protocol) for standardized tool discovery. MCP is quickly becoming the industry standard for agent-to-tool communication, with support from VS Code, JetBrains, and dozens of third-party platforms.*

Best for: Safety-sensitive workloads, computer-use agents, and teams that want MCP as a first-class primitive.

Pricing: SDK is free. You pay Anthropic API tokens.

5. Microsoft Agent Framework — Score: 8.0/10

Microsoft Agent Framework is the unification play. *It combines AutoGen's conversational multi-agent abstractions with Semantic Kernel's enterprise features (session-based state management, middleware, telemetry, and type safety) and adds graph-based workflows for explicit control over multi-agent execution paths. The framework ships with Python (pip install agent-framework) and .NET (Microsoft.Agents.AI), and supports Microsoft Foundry, Azure OpenAI, OpenAI, Anthropic, Amazon Bedrock, Google Gemini, and Ollama out of the box.* It also ships *sequential, concurrent, handoff, group chat, and Magentic-One orchestration patterns as first-class primitives, plus a browser-based DevUI inspector for running agents and workflows locally with OpenTelemetry trace viewing*.

Best for: .NET shops, Azure-committed enterprises, and teams that need A2A protocol interoperability out of the box.

Pricing: Framework is free. Azure AI Foundry consumption pricing applies if you run it managed.

6. AutoGen — Score: 7.6/10

Autogen remains the research-forward option, and it's still worth using — especially now that a lot of its DNA has been absorbed into Microsoft Agent Framework. *The core idea is that instead of relying*

on one agent to handle everything, you define specialized agents (like a planner, researcher, coder, reviewer) and have them communicate with each other through messages to solve complex tasks together. If LangChain is a general-purpose toolkit for LLM applications, AutoGen is more narrowly focused on conversational multi-agent orchestration. It uses an event-driven architecture, which means your agents can run in parallel, react to events, and handle long-running background tasks. Its distinctive strength remains *multi-agent debate and iteration*. Caveat: *Microsoft maintains it as an open research project, which means the roadmap can shift unpredictably compared to commercial products.*

Best for: Researchers, agent-debate patterns, and teams comfortable with a project that prioritizes exploration over stability.

Pricing: *AutoGen Studio is fully free and open-source under the MIT license. You run it locally with no usage caps, no cloud subscription required.*

7. Google ADK — Score: 7.4/10

Google Adk is the newest framework here and shows it — but if you're on GCP it's the natural pick. *Session state management is built in, with support for in-memory, database-backed, and Vertex AI-managed persistence. The framework is the newest in this comparison and its ecosystem is still maturing. Fewer third-party tutorials, integrations, and production case studies compared to LangGraph or CrewAI. Ideal for Google Cloud-native teams, enterprises needing managed infrastructure, and teams building multimodal agent systems.* Standout feature: *A2A protocol and multimodal* support.

Best for: Google Cloud-native teams, multimodal agent systems, and A2A-first architectures.

Pricing: SDK is free. Vertex AI consumption pricing applies for managed deployments.

Comparison Table

Framework	Score	Best For	Streaming	Starting Price
LangGraph	9.4	Production, complex state	Per-node token streaming	Free / $39-99/mo managed
CrewAI	8.8	Fast prototyping, role-based crews	Limited	Free / $29/mo
OpenAI Agents SDK	8.5	OpenAI-first, minimal overhead	Full	Free + tokens

Framework	Score	Best For	Streaming	Starting Price
Claude Agent SDK	8.3	Safety, MCP, computer use	Native + extended thinking	Free + tokens
Microsoft Agent Framework	8.0	Enterprise, .NET, A2A	Yes	Free / Azure consumption
AutoGen	7.6	Research, agent debate	Conversation-based	Free
Google ADK	7.4	GCP-native, multimodal	Via Vertex	Free / Vertex consumption

Final Picks

The honest builder-to-builder answer:

- **Ship to production with real users:** Langgraph. The checkpointing, time-travel debugging, and LangSmith observability are worth the steeper learning curve.
- **Prototype something in a weekend:** Crewai. Nothing else gets you from idea to running crew faster.
- **Already on OpenAI or Anthropic:** Use Openai Agents Sdk or Claude Agent Sdk. The provider-native SDKs are now good enough that pulling in a heavier framework needs a real justification.
- **Enterprise .NET / Azure shop:** Microsoft Agent Framework. Native A2A and MCP support, plus first-class .NET.

A common pattern that keeps showing up in the wild: *the "prototype in CrewAI, productionize in LangGraph" pattern works well for startups that need to iterate fast but ship something durable*. That's a reasonable default if you don't have strong constraints pulling you elsewhere.

One last thing worth budgeting for: the framework fee is almost never your biggest bill. *Most "agent platforms" are two bills stacked: a platform fee and the underlying LLM API cost. Many vendors bury the second line item. LangGraph Platform, CrewAI Enterprise, and Vellum all charge a platform/ops fee on top of your LLM API spend - model that from day one.* Model your token spend before you pick a framework, not after.
