



Microsoft Agent Framework in Practice: What the Docs Promise and What –prerelease Tells You

Two .NET packages, both still documented with - -prerelease on Microsoft Learn pages updated October 4 and October 7, 2026. That flag is the most useful piece of information in the whole documentation set.

WHERE THIS LANDS

Microsoft Agent Framework is the successor to Semantic Kernel and AutoGen, but as of the October 7, 2026 update to the “Your First Agent” page, the .NET install line is still `dotnet add package Microsoft.Agents.AI.Foundry - -prerelease`. Two documented packages carry that flag, and no GA announcement appeared in the research for this piece. Build prototypes on it now, but do not sign a 2027 platform commitment to it yet.



Microsoft Agent Framework is a library for building and orchestrating LLM agents, with a workflow layer on top for multi-agent coordination. It exists because Microsoft had two overlapping things (Semantic Kernel, aimed at enterprise .NET, and AutoGen, which came out of research and grew a Python following) and needed one story. The framework consolidates them: agent abstractions, tool calling, workflow orchestration, and bindings into Microsoft Foundry.

The consolidation is real work, and the documented pieces look sensible. What the documentation does not do is pretend the shipping state is further along than it is.

The install line Microsoft has not changed since August

The [Agents in Workflows page](#), dated August 25, 2026 and updated October 4, 2026, instructs `dotnet add package Microsoft.Agents.AI.Workflows --prerelease`. The same page shows `Microsoft.Agents.AI.Foundry --prerelease`. The [Your First Agent page](#), updated three days later on October 7, 2026, carries the same flag on the Foundry package.

Documentation pages get updated for all sorts of reasons, and an update does not mean every line was rewritten. But two packages, two pages, two different October dates, and the flag survives both passes. The straightforward reading is that the .NET packages are not GA.

! The version pins disagree

Microsoft Foundry docs pin `azure-ai-projects>=2.1.0` for the raw Azure AI Projects SDK, while a separate [A2A page](#) dated October 3, 2026 cites `>=2.5.0`. Four minor versions apart on official pages in the same month. If your dependency resolution is strict, resolve this yourself before you trust either number.

Package naming is inconsistent in a second way. “Your First Agent” and the overview page omit the Workflows package from install instructions entirely, so a developer who follows the getting-started path and then reaches for workflows has to discover the extra package on a different page. On the Python side, Learn confirms `pip install agent-framework-foundry` plus `pip install azure-identity`, and documents `agent-framework` as a metapackage that pulls in `agent-framework-core` plus optional



integrations. The dependency surface is wider than the one-line install suggests, which matters if you audit transitive dependencies before shipping.

Model choice, MCP and A2A are the parts that are built out

The capability list from the docs is not thin, which is what makes the prerelease flag interesting rather than disqualifying.

Model providers documented on the [overview page](#) include Microsoft Foundry, Azure OpenAI, OpenAI, Anthropic and Ollama. At the model layer this is not a single-vendor framework, and Ollama in that list means local development without an Azure round trip.

Agents can call MCP servers, and per the [Hosted MCP Tools page](#) dated October 5, 2026, Foundry supports both managed and custom MCP integrations. The framework also ships A2A-compatible agent implementations and can connect to remote agents over the A2A protocol. Both protocols being first-class matters more than it sounds: an agent built here can call tools and peers that were not built here.

Foundry workflows can be authored three ways: visually in a designer, declaratively in YAML, or in code. I read the YAML path as the one worth betting on. Visual designers are hard to review and hard to diff, and code is the most expressive but the least portable across a team. Declarative YAML is the format you can put in Git and argue about in a pull request.

WHAT WORKS

- Five or more documented model providers, including Anthropic and local Ollama, so the model layer is not locked to Azure.
- MCP as a first-class tool mechanism, managed and custom, documented October 5, 2026.
- A2A-compatible agents and remote agent connections, so interop with non-Microsoft agents is a designed path.
- Three workflow authoring modes, including declarative YAML that fits normal code review.

ROUGH EDGES

- Both documented .NET packages still install with - - prerelease on pages updated October 4 and 7, 2026.
- Conflicting `azure-ai-projects` pins across official pages (`>=2.1.0` vs `>=2.5.0`).
- Getting-started pages omit the Workflows package, so the install path is incomplete depending on which page you land on.
- Azure dependence is the recurring trade-off in [practitioner comparisons](#), with durability typically tied to Cosmos DB or Redis.
- No GA announcement, no GA blog post, and no confirmed Ignite 2026 date surfaced in this research run.



That last point needs care. Absence of evidence in one search pass is not proof that nothing was announced. What I can say is narrower: nothing in the sources retrieved for this brief confirms GA. The MIT license claim for the GitHub repo could not be verified against repository metadata in this run, so I am not going to state it as fact. If licensing drives your decision, check the repo yourself rather than trusting a secondary summary. I made the same argument about [what an MIT license actually buys you](#) on a different agent harness, and the logic holds here.

Azure-first .NET shops can prototype, cloud-neutral teams get less

It depends on what “build on” means in your organisation.

If you are an Azure-first .NET shop already running Semantic Kernel, the migration path is the thing to investigate. Prototype on Agent Framework, keep Semantic Kernel in production, and move when the packages drop the flag.

If you are AWS-first or deliberately cloud-neutral, the model-layer flexibility (Anthropic, OpenAI, Ollama) does not buy you much, because the durability story points at Cosmos DB or Redis and the workflow designer lives in Foundry. The framework is portable where models are concerned and Azure-shaped everywhere else.

MY TAKE

The prerelease flag is a cost you can price. Prototyping against prerelease packages is normal engineering. Signing a platform decision for an eighteen-month roadmap against them is reckless, because you are committing to API surfaces that Microsoft has explicitly reserved the right to change. My rule: anything with - -prerelease in the official install line can carry an experiment, a demo, or an internal tool. It cannot carry a revenue path without a written answer from your Microsoft account team on GA timing. The MCP and A2A support is the part I would actually build around, since those are protocols rather than a vendor's object model, and they survive if you later switch frameworks.

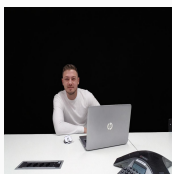
Microsoft is publishing enterprise positioning and prerelease install instructions on the same documentation set in the same week, and it is not hiding the second one. I would



Microsoft Agent Framework in Practice: What the Docs Promise and What –prerelease Tells You

rather have a vendor that leaves the flag visible than one that quietly ships 1.0 and patches breaking changes into minor versions.

If you are weighing this against LangGraph, CrewAI, or an automation stack instead of a framework, the shape of the question changes, and I went through [that comparison for CTOs](#) separately. If you are sitting on a Semantic Kernel codebase and trying to work out whether this quarter is the right time to move, you can [reach me here](#).



WRITTEN BY

Artur Markus

Software engineer in Basel building AI infrastructure and agentic systems, with a background in pharmaceutical automation, GxP compliance, and validation. I help teams turn AI from a chatbot into a reliable operational layer.

Book a meeting →