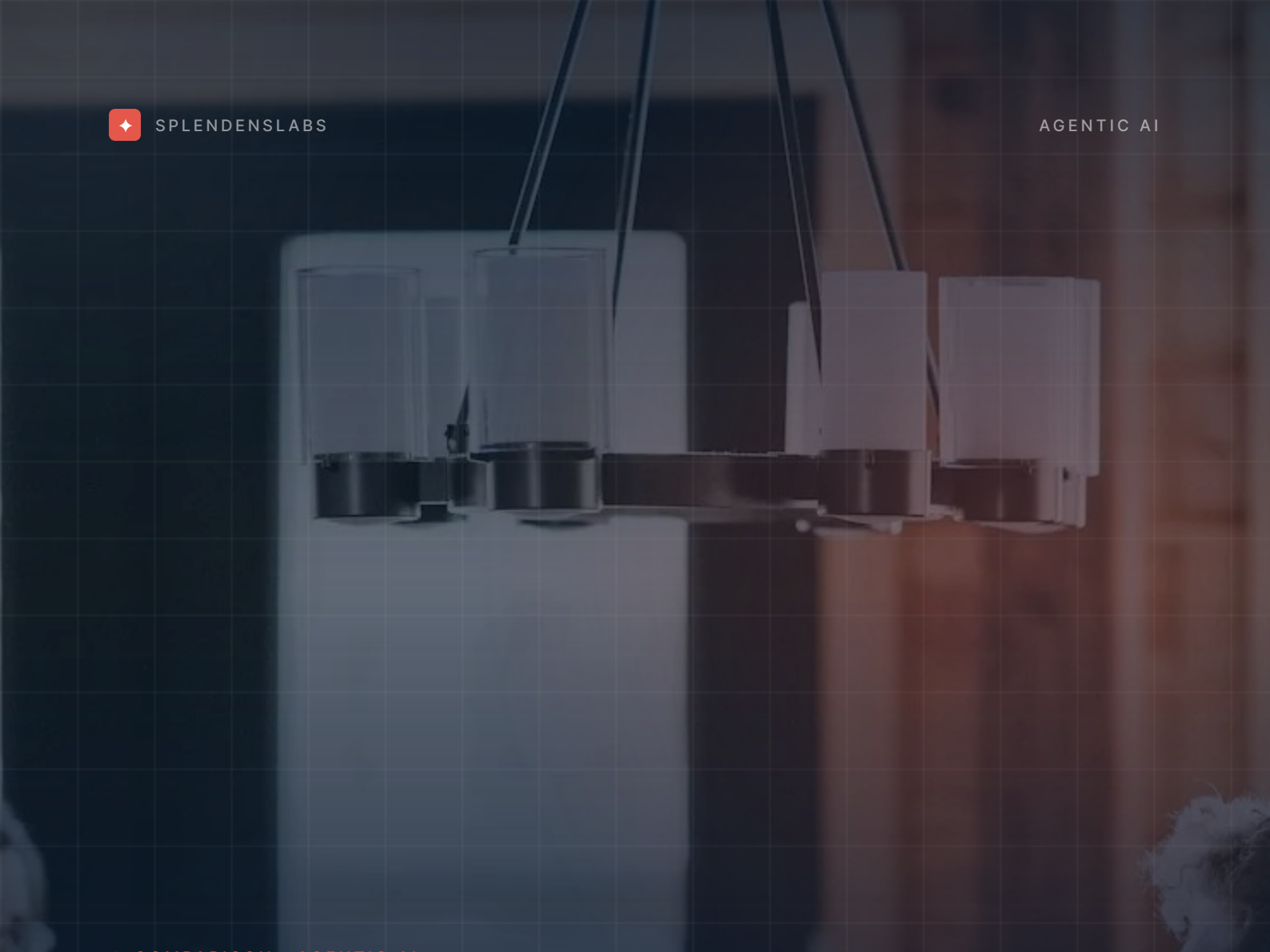


✦ COMPARISON · AGENTIC AI

CrewAI vs AutoGen

Role-based crews against conversational, event-driven agent teams. How the two best-known multi-agent frameworks model work, where each fits, and what Microsoft's framework changes mean for new projects.

FORMAT	LEVEL	READ TIME	UPDATED
Comparison	Intermediate	10 min	Oct 2026

OVERVIEW

What this comparison covers

CrewAI and AutoGen both let you build systems where several AI agents collaborate, but they model collaboration differently. CrewAI organizes agents as a crew with roles, goals and assigned tasks; AutoGen organizes them as participants exchanging messages in conversations, on top of an event-driven core. This comparison covers the mental models, control and production concerns, the state of each project, and a checklist for choosing — including when a multi-agent framework is not the right tool at all.

KEY TAKEAWAYS

- CrewAI thinks in roles, tasks and processes; AutoGen thinks in agents exchanging messages in a conversation or event stream.
- CrewAI is quicker to a working demo of a structured, task-based team; AutoGen offers more low-level flexibility.
- Microsoft has positioned Microsoft Agent Framework as the successor to AutoGen and Semantic Kernel — factor this into any new AutoGen project.
- Both need explicit guardrails: step limits, cost caps, tool permissions and human approval points.
- Many ‘multi-agent’ problems are better solved by one agent with good tools, or by a deterministic workflow.

WHO IT’S FOR

Engineering leads evaluating agent frameworks

AI engineers building multi-agent systems

Architects in Microsoft-centric organizations

Product owners scoping agentic automation

HOW TO USE IT

Read 01–02 for the mental models, use the table in 03, read the project-status note in 04 carefully, then work through the checklist in 06.

CONTENTS

01 Crews vs conversations
02 How each framework is built
03 Side by side
04 What has changed around AutoGen
05 When to choose which
06 Decision checklist

01 / MENTAL MODELS

Crews vs conversations

The fastest way to understand the two frameworks is the metaphor each one is built around.

CREWAI

A team with job descriptions

Agents get a role, goal and backstory; tasks have descriptions and expected outputs; a process decides who does what, in what order.

AUTOGEN

A group chat of agents

Agents exchange messages in teams such as round-robin or model-selected speaker chats, over an asynchronous, event-driven runtime.

CrewAI: task-driven

01 Define agents and roles → 02 Define tasks and outputs → 03 Choose process → 04 Crew executes tasks → 05 Outputs passed along → 06 Final deliverable

AutoGen: message-driven

01 Define agents and tools → 02 Form a team → 03 Task posted as message → 04 Agents take turns → 05 Termination condition met → 06 Result returned

02 / ARCHITECTURE

How each framework is built

Under the metaphors, the frameworks differ in how much structure they impose and how far down you can reach.

CREWAI

- Python framework, independent of other agent libraries
- Crews: sequential or hierarchical (manager agent) processes
- Flows: event-driven, deterministic orchestration around crews
- Built-in memory, knowledge sources and tool integrations
- Commercial platform for deployment and monitoring

AUTOGEN

- Layered: Core (event-driven actor runtime), AgentChat (high-level teams), Extensions
- Team patterns: round-robin, selector, swarm-style hand-offs
- Strong support for code-executing agents in sandboxes
- AutoGen Studio for low-code prototyping
- Python and .NET support

FIELD NOTE

Structure is a feature

CrewAI's Flows and AutoGen's Core both exist because free-form agent conversation is hard to make reliable. In production we push as much as possible into deterministic orchestration and keep open-ended agent collaboration for the steps that truly need it.

✦ 03 / COMPARISON

Side by side

Both are open source. Both are model-agnostic and work with the major hosted providers and local models.

DIMENSION	CREWAI	AUTOGEN
Core abstraction	Agents with roles, tasks, crews, flows	Agents, messages, teams, event-driven runtime
Control flow	Sequential, hierarchical, or explicit Flows	Conversation patterns plus custom runtimes
Learning curve	Gentle for the crew model	Gentle for AgentChat; steeper for Core
Determinism	Flows give explicit, testable steps	Termination conditions and custom orchestration
Code execution	Via tools	First-class code-executor agents
Languages	Python	Python and .NET
Low-code option	Commercial platform UI	AutoGen Studio
Stewardship	CrewAI, Inc. and community	Microsoft; see project status note

✦ 04 / PROJECT STATUS

What has changed around AutoGen

Framework choice is also a bet on its future. AutoGen's landscape changed in ways that matter for new projects.

01 **AutoGen was re-architected**

The 0.4 line introduced the layered Core / AgentChat / Extensions design; older 0.2 code needs migration.

02 **A community fork continues the original line**

Some original contributors maintain AG2, which keeps the earlier AutoGen programming model.

03 **Microsoft announced a successor**

Microsoft Agent Framework brings AutoGen's multi-agent ideas together with Semantic Kernel's enterprise features, and is positioned as the path forward.

VERIFY

Check status before you start

Read the current AutoGen repository and Microsoft's documentation for maintenance status, migration guidance and roadmaps. For new Microsoft-centric projects, evaluate Microsoft Agent Framework alongside AutoGen. CrewAI also releases frequently — pin versions in both cases.

05 / GUIDANCE

When to choose which

Defaults we use when scoping agentic work. Each assumes the task genuinely needs more than one agent.

CREWAI

Structured knowledge work

Research, analysis and drafting pipelines where roles and hand-offs map naturally to a team.

CREWAI

Fast proof of value

You need a convincing multi-agent prototype quickly, with readable configuration.

AUTOGEN FAMILY

Code-writing and executing agents

Agents that write, run and fix code in sandboxes as part of the loop.

AUTOGEN FAMILY

Microsoft and .NET estates

Teams on Azure and .NET — evaluate Microsoft Agent Framework as the forward-looking option.

BOTH

Mixed architectures

Wrap a crew or agent team as one tool or node inside a larger deterministic workflow.

NEITHER

Single agent or fixed workflow

If one agent with good tools, or a rules-based pipeline, does the job, skip multi-agent entirely.

Guardrails either way

- Cap turns, steps and tokens per run, and fail safely when a cap is hit.
- Give each agent only the tools and data it needs, with scoped credentials.
- Insert human approval before irreversible actions such as payments or emails.
- Trace every message and tool call, and evaluate on a fixed task set before release.

✦ 06 / DECIDE

Decision checklist

Work through this before committing to a multi-agent framework.

NEED

- [ ] We tried a single agent with tools and documented why it was not enough
- [ ] Distinct roles or skills genuinely need separate agents
- [ ] Parts of the process that must be deterministic are identified

FIT

- [ ] The team prefers a role/task model or a message/event model
- [ ] Language and platform constraints (Python, .NET, Azure) are known
- [ ] Project maintenance status and roadmap checked at time of decision

PRODUCTION

- [ ] Step, cost and time limits defined per run
- [ ] Human approval points defined for risky actions
- [ ] Tracing, evals and a regression task set in place
- [ ] Versions pinned and upgrade testing planned

> **Add agents only when you can name the job each one does that the others cannot.**
>
> SPLENDENSLABS ENGINEERING PRINCIPLE

✦ GET IN TOUCH

Ship agents that finish the job

We design and build agentic workflows on CrewAI, the AutoGen family, LangGraph or plain code, with the guardrails production needs. Bring us the process you want to automate.

- Single- vs multi-agent architecture review
- Prototype on your real workflow
- Guardrails, tracing and evaluation setup

BOOK AN AI STRATEGY SESSION →

EMAIL
hello@splendenslabs.com

WEB
splendenslabs.com

OFFICE
Sona Incubation Foundation, Salem