What Are AI Agent Frameworks?

Before you compare LangGraph against CrewAI against AutoGen, it helps to know what a framework actually is — the scaffolding that turns a raw language model into something that can plan, use tools, and act. Here is the plain-English foundation: what problem frameworks solve, what parts they give you, and how to choose one.

Alain AI Lab Research · Published July 14, 2026 · 10 min read

An AI agent framework is a piece of software that gives you the reusable scaffolding for building an agent, so you are not assembling the whole machine from scratch every time. If a language model is an engine, a framework is the chassis, wiring, and controls that turn that engine into a car you can actually drive. It handles the repetitive, fiddly plumbing — the loop that calls the model, the code that lets the model use tools, the memory that carries context, the logic that coordinates several agents — and lets you focus on the job you are trying to get done. This report is the foundation. Once you understand what a framework *is*, the head-to-head comparison of LangGraph, CrewAI, AutoGen and multi-agent systems will make far more sense.

AT A GLANCE

WHAT IT IS	SOLVES	PIONEER
Reusable scaffolding for building agents	**The loop, tools, memory, state, orchestration**	**LangChain, October 2022**
NOT A FRAMEWORK	ALTERNATIVE	TOP RISK
MCP — an open standard (Nov 2024)	**Build direct on the model API**	**Frameworks don’t make agents safe**

01

What a Framework Actually Is

Strip away the branding and an AI agent framework is a toolkit — a library of pre-built components that solve the problems every agent developer faces. It sits between you and the raw model API, offering clean, named building blocks: a way to define tools, a way to store memory, a way to run the agent's decision loop, a way to wire multiple agents together. You still write code, but you write far less of it, and the parts that are hard to get right are handled for you.

The word "framework" matters. A framework is not just a collection of helper functions you call when convenient; it imposes a structure, a way of thinking about agents, that you build inside. That is its strength and its cost. The structure gives you speed and proven patterns. It also means you are adopting someone else's mental model, and you inherit its assumptions. Choosing a framework is therefore less like buying a tool and more like choosing a workshop to build in.

02

The Problem Frameworks Solve

To see why frameworks exist, picture building an agent without one. You start with a language model behind an API. You send it a prompt. It replies with text saying it wants to search the web. Now *you* must parse that reply, recognise it as a tool request, actually run the search, format the result, send it back to the model, and wait for the next step — then repeat the whole cycle until the task is done. You also have to remember everything said so far, because the model itself is stateless and forgets between calls. You have to handle errors, retries, and the moment when the agent should stop.

That loop — prompt, parse, act, feed back, repeat — is the beating heart of every agent, and writing it by hand is tedious and error-prone. If you want to see exactly how that cycle runs, the report on how AI agents work walks through it step by step. A framework's core purpose is to give you that loop, correct and battle-tested, out of the box — so you spend your time on what your agent should do, not on the machinery that lets it do anything at all.

03

The Building Blocks Every Framework Provides

Different frameworks look different on the surface, but underneath they offer the same small set of parts. The first is the **agent loop** — the control flow that drives perceive, reason, act, observe. The second is **tool calling**, sometimes called function calling: a clean way to hand the model a menu of actions it can take. This capability was popularised by

OpenAI, which shipped function calling in June 2023, and later by Anthropic's tool use; frameworks wrap the raw mechanism into something you can declare in a few lines.

The third block is **memory**. Because the model forgets between calls, frameworks provide short-term memory for the current conversation and longer-term memory that often uses a vector database — a store of numerical embeddings that lets an agent retrieve relevant past information, the technique known as retrieval-augmented generation. The fourth is **state management**, tracking where the agent is in a multi-step task. The fifth is **orchestration** — the coordination layer for running several agents together. Learn these five parts and you can read any framework's documentation, because they are all describing the same anatomy in their own dialect.

> A framework does not add intelligence. The intelligence is the model. The framework adds structure — the reliable, repeatable plumbing that lets that intelligence be pointed at a real task without you rebuilding the pipes each time.

04

Why the Major Frameworks Feel Different

Given a common anatomy, why do the popular frameworks feel so unlike each other? Because each makes a different bet about how agents should be organised. LangChain, first released in October 2022, popularised the idea of a "chain" — composing model calls and tools into a pipeline — and its companion LangGraph models an agent's workflow as a graph of nodes and edges, which allows loops and branching rather than a single straight line. CrewAI takes a more human metaphor: it organises work around role-based "crews," where each agent has a role, a goal, and tasks, collaborating like a small team.

Microsoft's AutoGen grew up around multi-agent conversation, and in October 2025 Microsoft folded it and its Semantic Kernel project into a single Agent Framework, released in public preview on October 1, 2025. The differences are real but they sit on top of the same building blocks. That is the whole point of learning the foundation first: once you know what the parts are, the named comparison becomes a question of philosophy and fit, not a wall of unfamiliar jargon.

05

Frameworks, Protocols, and Raw APIs

Not everything in this space is a framework, and the distinction is worth keeping straight. A **protocol** is a shared standard for how systems talk, not a toolkit for building them. Anthropic's Model Context Protocol, introduced on November 25, 2024, is exactly this: an open standard for connecting agents to tools and data through a common interface, so that an integration written once can be reused everywhere. MCP complements frameworks; it does not replace them. A framework might *use* MCP to reach a data source.

At the other extreme, some teams skip frameworks entirely and build directly on the model API. This is a legitimate choice, not a mistake. Anthropic's own engineering guidance notes that heavy abstraction can obscure what the model is actually receiving and doing, and recommends starting simple. Building direct gives you maximum control and transparency at the cost of writing more plumbing yourself. The honest rule is that a framework is a convenience, not a requirement — and the right amount of framework is the least that solves your problem.

06

How To Choose One

With the landscape clear, choosing comes down to a handful of honest questions. Do you need one agent or many? Single-agent tasks rarely justify a heavy multi-agent framework. How much autonomy versus control do you want — a predictable, human-authored workflow, or a model that directs its own path? Anthropic's December 2024 essay "Building Effective Agents" draws exactly this line between workflows, where models follow predefined code paths, and agents, where they steer themselves.

Beyond that, weigh the ecosystem and integrations you will lean on, the production essentials of observability and debugging, and lock-in risk — how hard it would be to leave later. There is no universally best framework; there is only the best fit for a specific job, team, and tolerance for abstraction. A team shipping a simple support agent and a team orchestrating a dozen specialised agents should make different choices, and both can be right.

07

Frameworks in Crypto

In crypto, frameworks gain an extra layer: blockchain toolkits and plugins that let an agent read on-chain data and submit transactions — wallet integrations, RPC connections to nodes, adapters for DeFi protocols. This is what turns a general agent framework into

something that can act inside an agentic system on a live network. But the framework supplies the plumbing, not the safety.

Because a confirmed on-chain transaction is final — there is no support desk to reverse it — crypto agents built on any framework still need constrained execution rails around them: account-abstraction wallets under the ERC-4337 standard, session keys that grant narrow, time-boxed permissions, and multi-party computation that splits signing authority so no single agent holds an unrestricted key. The framework makes the agent easy to build. These rails are what make it safe to let loose with money. Skipping them does not make an agent autonomous; it makes it unsecured.

08

What a Framework Will Not Do For You

The most important thing to understand about frameworks is their limit. A framework makes an agent easier to build; it does not make it correct, and it does not make it safe. Security, permissioning, and human oversight remain your responsibility. The OWASP security project ranks **prompt injection** as the number-one risk to language-model applications in its 2025 list — hostile text hidden in a webpage or document that hijacks the agent's instructions — and frameworks that auto-execute tools widen that attack surface, not narrow it. OWASP names a companion failure, **excessive agency**: giving a system more permission and autonomy than its reliability earns.

There is a quieter risk too. Adopting a framework means adopting a dependency: a breaking update or a vulnerable component in the framework flows down into everything you built on top of it, which OWASP tracks as a supply-chain concern. The early autonomous projects AutoGPT and BabyAGI, both from 2023, showed how fragile framework-style loops can be — getting stuck, losing the thread, burning cost without finishing. The tooling is far better now, but the lesson stands: a framework is a powerful accelerant and nothing more. It is worth asking, of any agent you build, the same blunt question — what is the worst thing it is allowed to do without asking a human first?

"If the iron be blunt, and he do not whet the edge, then must he put to more strength: but wisdom is profitable to direct."

ECCLESIASTES 10:10

METHODOLOGY & SOURCES

This report was produced by a parallel multi-agent research process, with each claim independently checked and labelled verified, partial, or uncertain before inclusion. Figures are stated as approximate where precise public data does not exist; no specific decimals, dates, or dollar figures were invented, and unverified “nine-figure AI-agent hack” claims were excluded as unsupported.

Primary references include LangChain’s October 2022 release and the LangGraph graph model; CrewAI’s role-based crews; Microsoft’s Agent Framework public preview (October 1, 2025) unifying AutoGen and Semantic Kernel; OpenAI’s function calling (June 2023); Anthropic’s Model Context Protocol (November 25, 2024) and “Building Effective Agents” (December 19, 2024); Gartner’s October 21, 2024 naming of agentic AI as a top trend; the OWASP Top 10 for LLM Applications (2025) on prompt injection, excessive agency, and supply-chain risk; and the ERC-4337 account-abstraction standard. Educational content only; not financial advice.

Related reading: How Do AI Agents Work? · AI Agent Frameworks: LangGraph, CrewAI & AutoGen · What Is an Agentic System?

Alain AI Lab — independent crypto & AI research.
intelligencecrypto.org · This document is for educational purposes only and is not financial advice.