

Agentic Design Patterns and Orchestration

CS 224G



OpenAI Operator

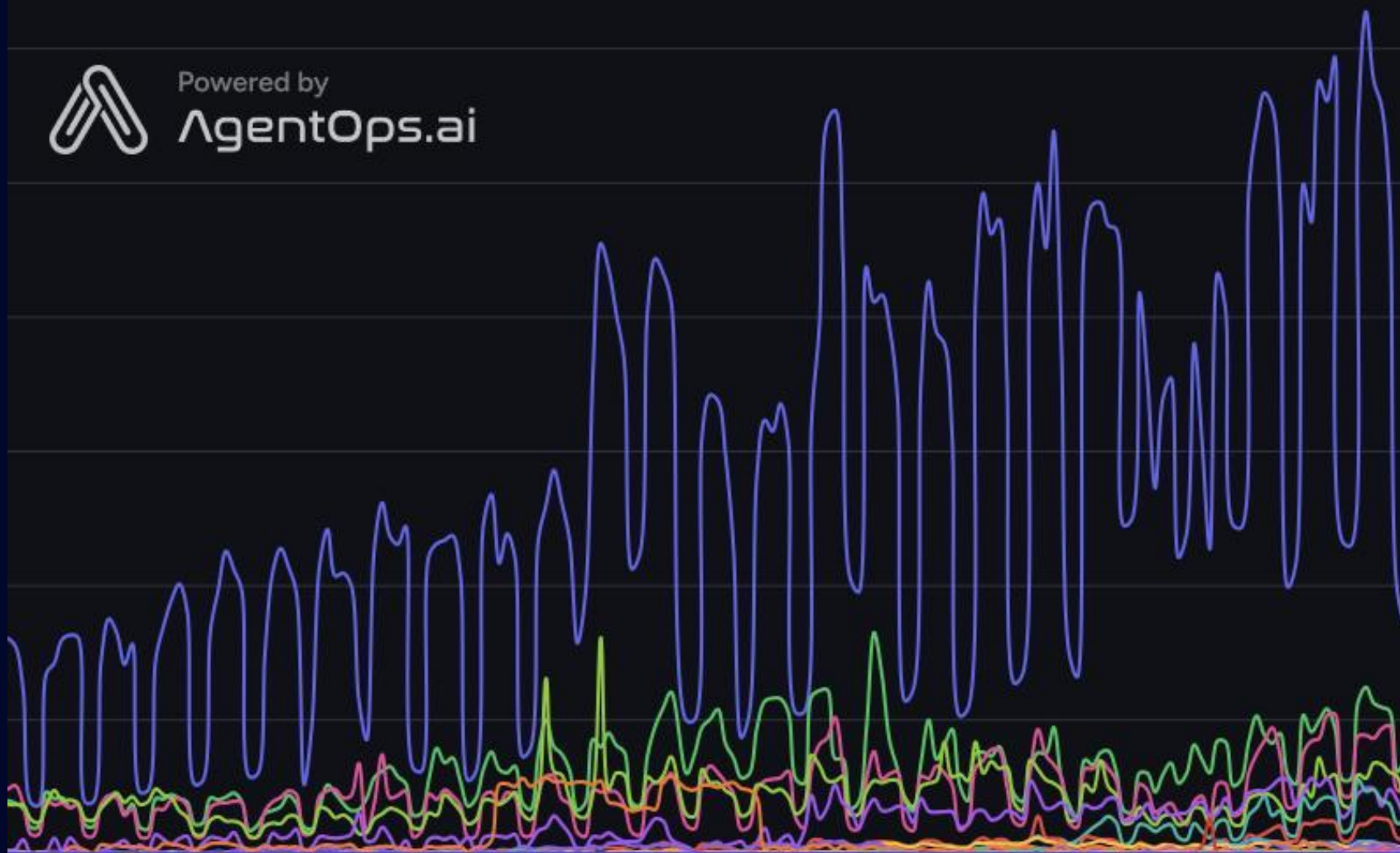
What is the most widely used agent framework?



LangGraph (2.5M/mo) CrewAI (504.4K/mo) PyAutoGen (390.3K/mo) Haystack (268.7K/mo) Phidata (253.7K/mo)
Pydantic AI (177.6K/mo) AutoGen AgentChat (91.3K/mo) SmolAgents (43.9K/mo) Llama Stack (39.6K/mo)
AutoGen Studio (28.0K/mo) Burr (25.4K/mo) PraisonAI Agents (13.0K/mo) ControlFlow (12.7K/mo) Julep (8.6K/mo)
Camel AI (6.0K/mo) Lavague (2.8K/mo) Llama Agents (1.3K/mo)



Powered by
AgentOps.ai



Sep 2024

Oct 2024

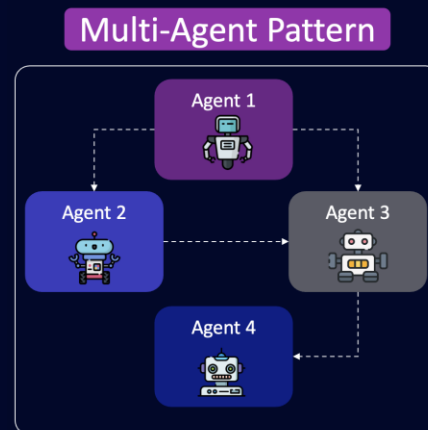
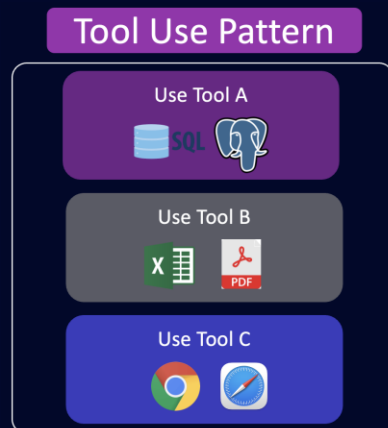
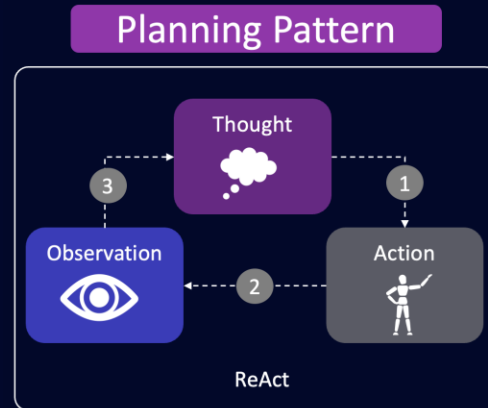
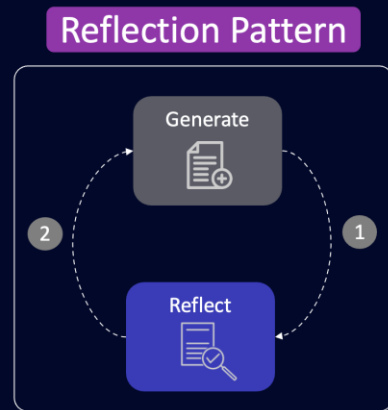
Nov 2024

Dec 2024

Jan 2025

posted by
@AlexReibman
on Jan 27th

Agentic Design Patterns



Agent Frameworks



For the Challenge

① Each team come up with own agent use case (e.g. a sub use case of what you are working on)

② Enter the use case in this spreadsheet



Agentic Design Patterns

Follow along on this github repo

<https://rb.gy/ecmgxe>

Agent Frameworks



Framework Comparison

Framework	Focus	Memory	Language	Github Stars	Edge
Langgraph	LLM applications, graph-based agent workflows	State management	Python	8.8k	Extensive community, control over flow and state of the application
CrewAI	Role-based agents	Shared context, task history	Python	26.3k	Human-like agent interaction
Autogen	Task automation	Conversational memory, context management	Python/ C	39.2k	Suited for Code Gen
Swarm	Multi-agent exploration	Task history	Python	18.6k	Lightweight

The Challenge

- ① Find the strengths and weaknesses of the different frameworks
- ② Implement your use case with one or multiple frameworks.
- ③ Submit strengths and weaknesses in this form



→ **Team that submits most answers gets to present their use case**

Do we really need agent frameworks?

Pros:

- Faster head start
- Slightly less code
- Good for exploration

Cons:

- Building agents from scratch is not that hard (1-3 .py files)
- Need for re-work as frameworks evolve

leonie@godryft.com

@lcfreisi