AI Agents

Mutaz Abu Ghazaleh

07/04/2025

UTS

Microsoft

What is a Large Language Model (LLM)?

A **Large Language Model (LLM)** is a type of AI model trained on vast amounts of text data to understand, generate, and reason with natural language.

LLMs as the Building Blocks of GenAI Systems

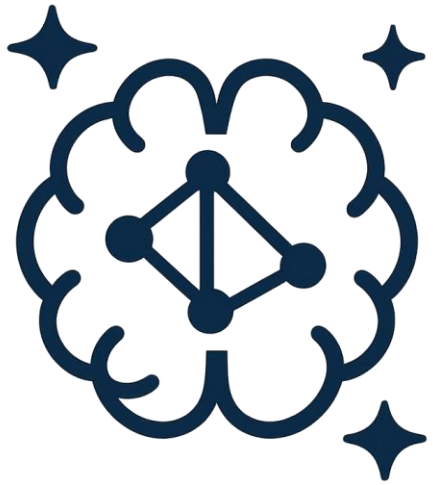
- Serve as **foundational models** for interaction, automation, and intelligence.
- Power **chatbots, copilots, summarizers, translators, planners**, and more.
- With **Function Calling**, LLMs can now go beyond text – they can invoke tools, access APIs, trigger workflows.

Core Capabilities of LLMs:

- **Text Generation:** Predicts and generates coherent, human-like text.
- **Language Understanding:** Extracts meaning, intent, and structure from text.
- **Reasoning:** Performs logic-based tasks and chain-of-thought reasoning.
- **Chat Completions:** Interacts through turns of conversation by predicting the next best message based on context.

If traditional software was about “code and rules,” LLMs bring “language and reasoning” to the interface – a new primitive.

LLMs – what are they good for?



🗣️ Chat Completion	📄 Summarization	📚 Q&A / Search
🧮 Code Gen	🌐 Translation	🔧 Function Calling

🛑 "LLMs **react**, they don't act on their own."
📌 No memory | No goals | No autonomy

LLM Chat Completions

Use the Chat Completions API* to interact with the OpenAI LLMs:
https://platform.openai.com/docs/api-reference/chat/create?lang=python

```
response = client.chat.completions.create(
    model="gpt-4o-mini",
    messages=[
        {"role": "system", "content": "You are a helpful assistant."},
        {"role": "user", "content": "Who won the world series in 2020?"}
    ]
)

print(response.choices[0].message.content)
```

* OpenAI released the new Responses API as a progression to Chat Completion API:

Responses API

```
from openai import OpenAI
client = OpenAI()

response = client.responses.create(
  model="gpt-4o",
  input="Write a one-sentence bedtime story about a unicorn."
)

print(response.output_text)
```

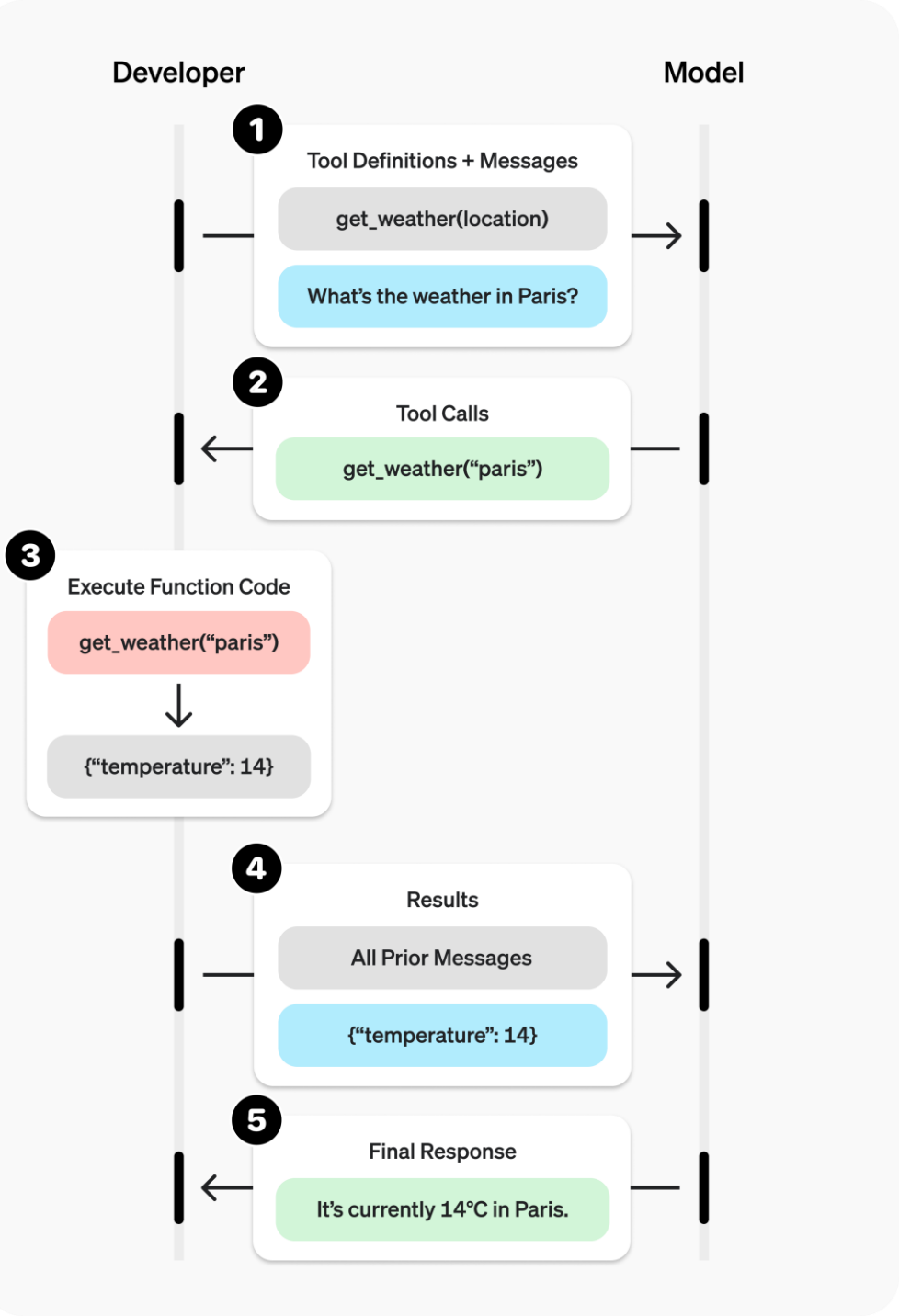
https://openai.com/index/new-tools-for-building-agents/
https://platform.openai.com/docs/quickstart?api-mode=responses

Function Calling

The LLM models are limited to generation based on knowledge included in their training data or context information provided in the prompt (see RAG concepts).

However, function calling allows the model extend it's capabilities and request to invoke function to perform specific tasks or retrieve information.

This is particularly useful for tasks that require structured data or specific actions, such as retrieving information from a database or performing calculations.



Setup your env

- Clone github Repo: https://github.com/mdsiprojects/agenticai
- Explore github model catalog: https://github.com/marketplace/models
- Get a github personal access token: https://github.com/settings/personal-access-tokens
- Update your .env file with your personal access token:
  - `GITHUB_TOKEN=github_pat_your_access_token`
- Start Coding!

Code Exercises: Using LLMs

Notebook	Objective
01_hello_github_models	• Introduction to GitHub Marketplace Models • Setup of the OpenAI client with GitHub authentication • Detailed explanation of the Chat Completions API and its parameters • A practical example of making an API call to the GPT-4o model • Displaying the model's response
02_functions_calling	1. Understand what function calling is and why it's important for extending LLM capabilities 2. Compare standard chat completions with function-enhanced responses 3. Implement the complete function calling workflow: - Define function schemas that the model can use - Send prompts with available tools - Parse function call requests from the model - Execute the requested functions with proper arguments - Return results to the model for final response generation

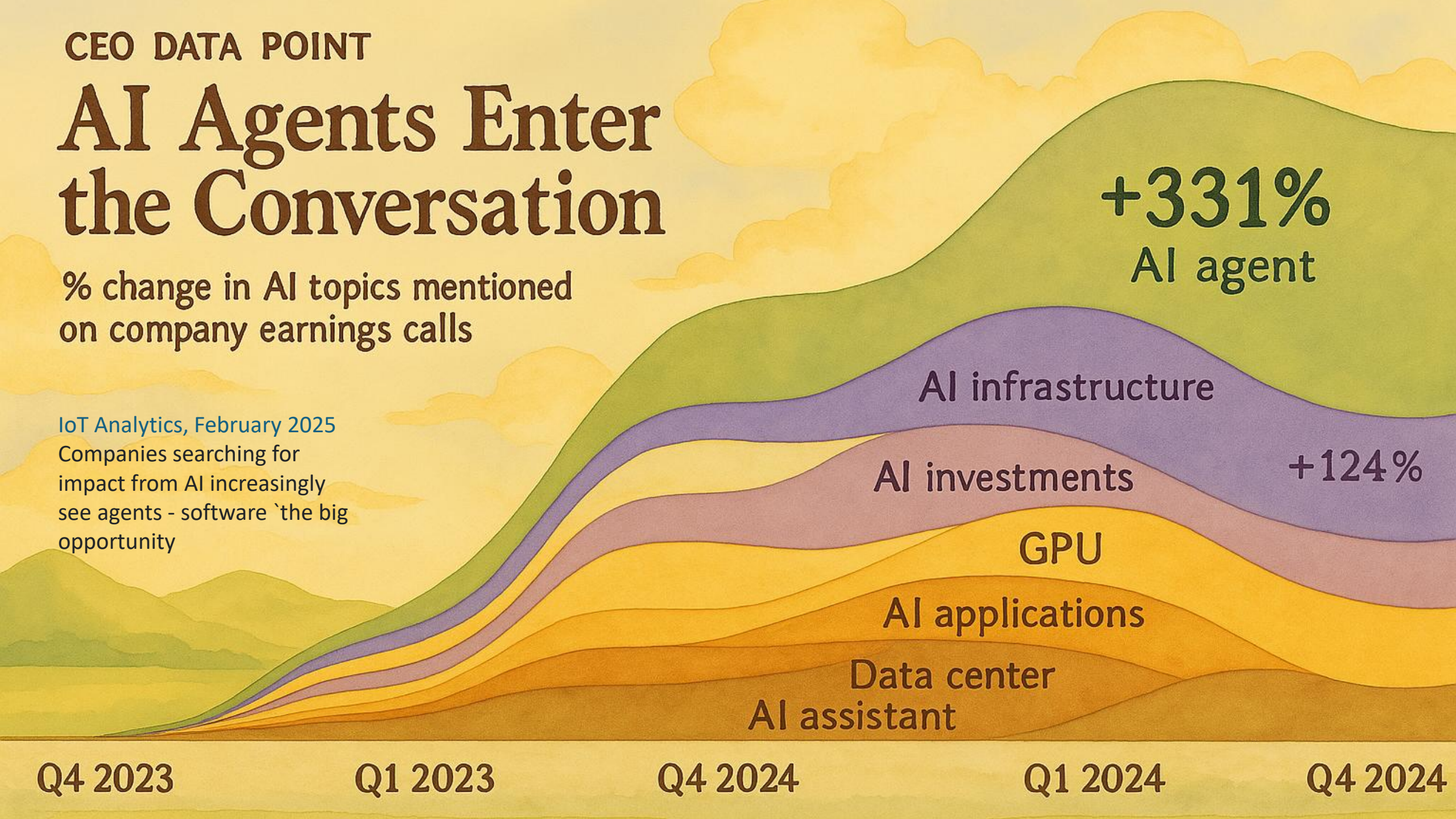

AI Agents


I think AI agent workflows will drive massive AI progress this year — perhaps even more than the next generation of foundation models.
This is an important trend, and I urge everyone who works in AI to pay attention to it.
Andrew Ng, March 20 2024
https://www.deeplearning.ai/the-batch/how-agents-can-improve-llm-performance
The concept of AI Agents and Agentic Workflows used to have a technical meaning … but about a year ago, marketers and a few big companies got a hold of them.

CEO DATA POINT

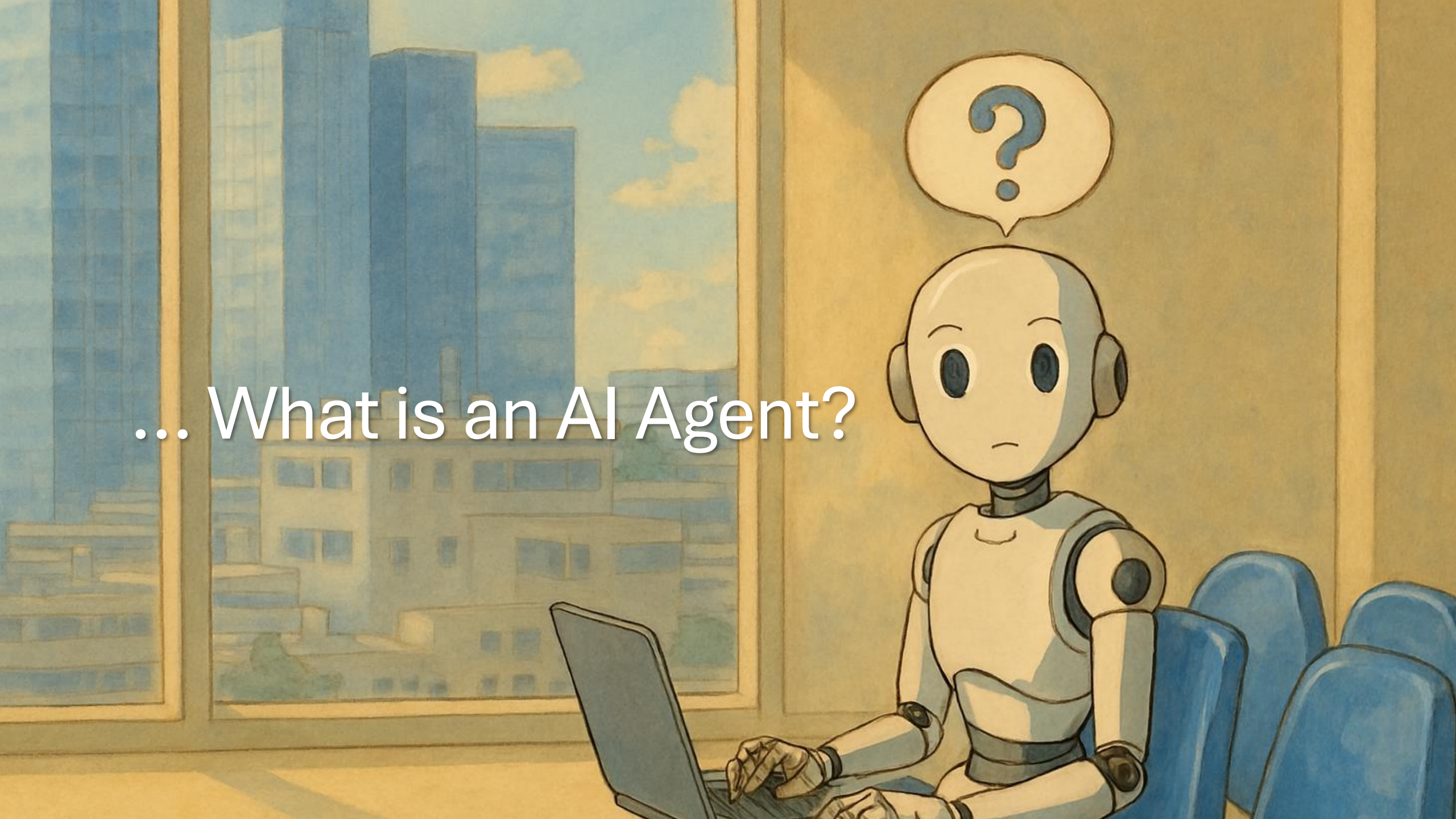
AI Agents Enter the Conversation

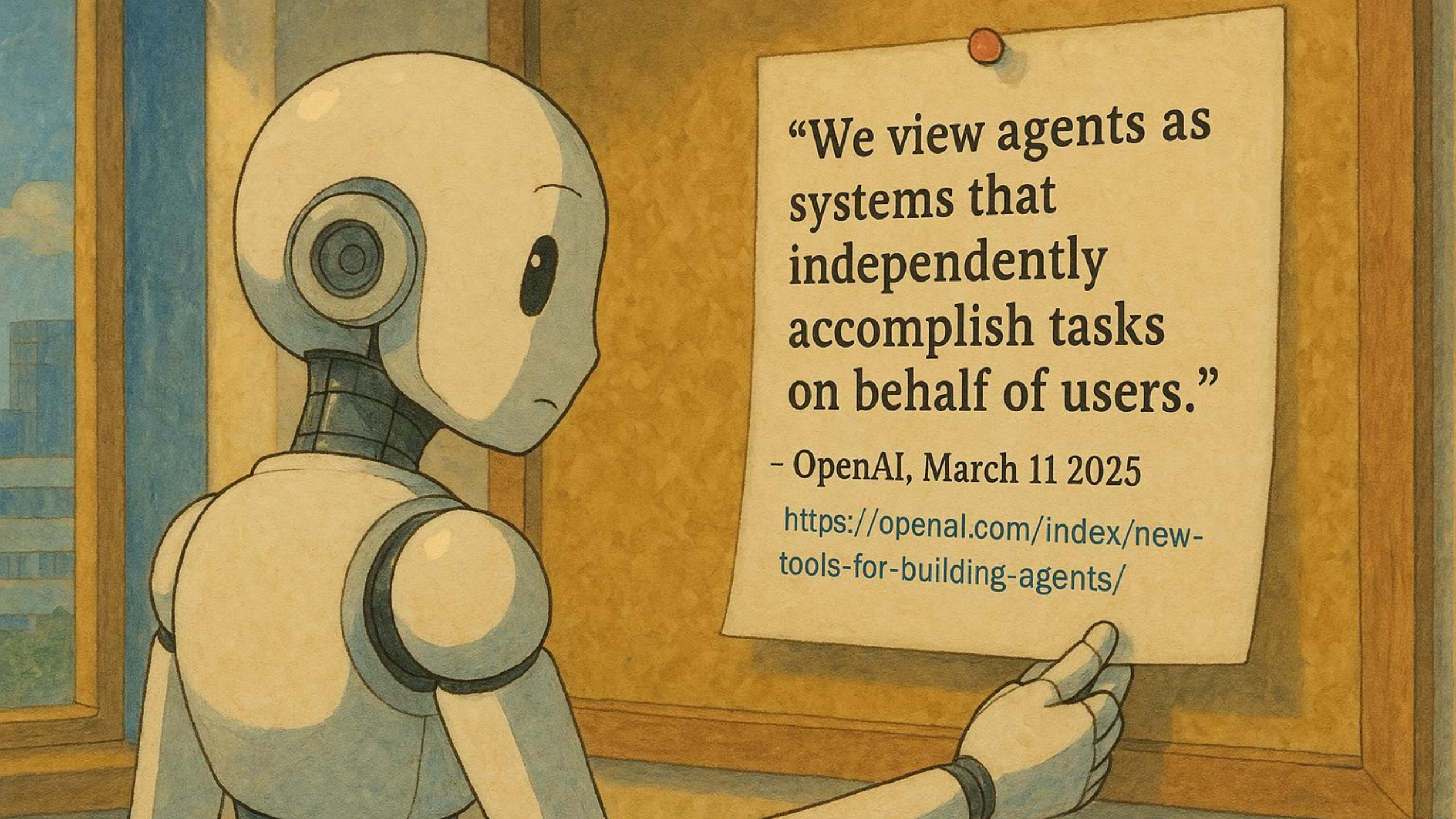
% change in AI topics mentioned on company earnings calls

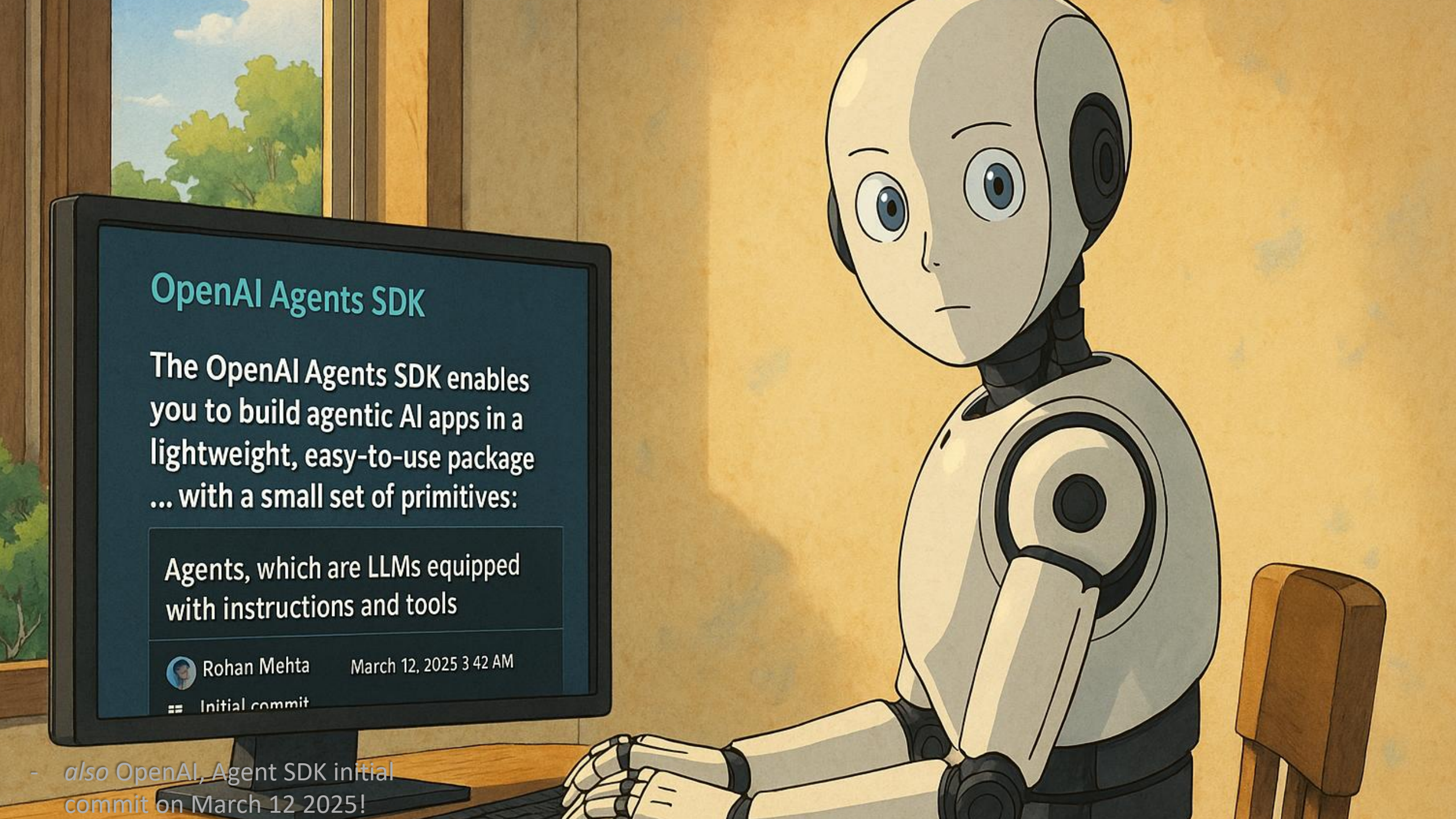


IoT Analytics, February 2025

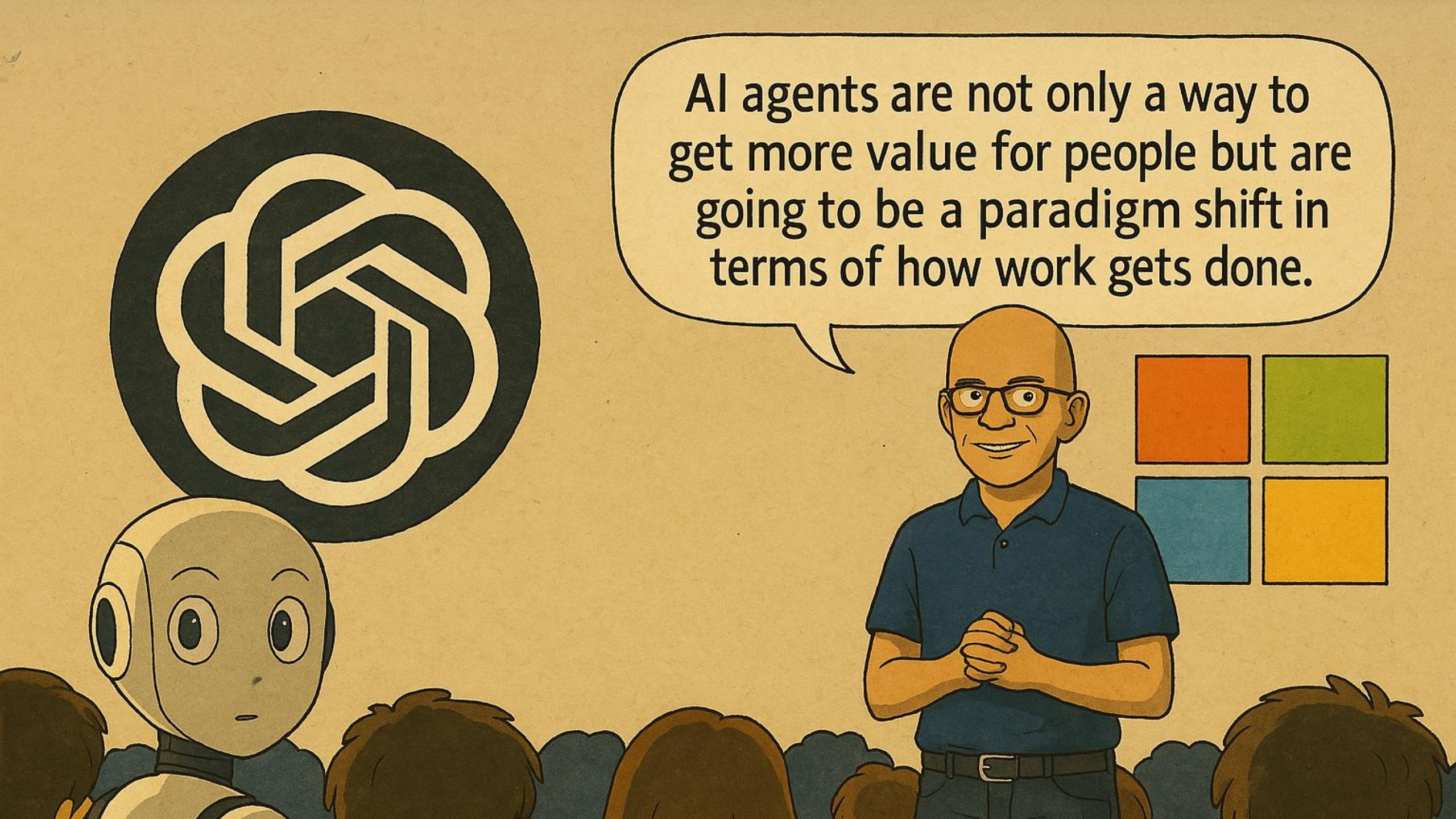
Companies searching for impact from AI increasingly see agents - software `the big opportunity

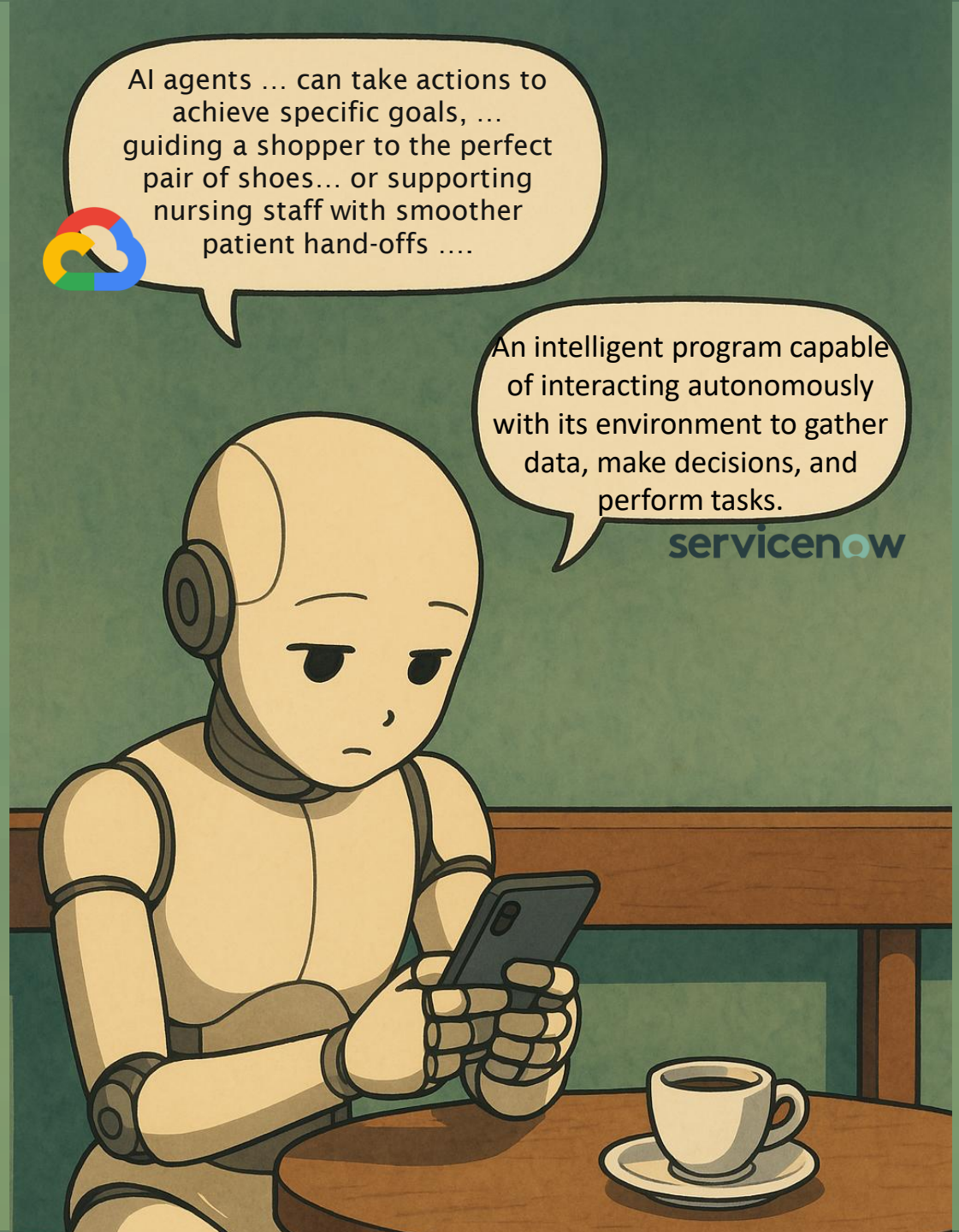

... What is an AI Agent?

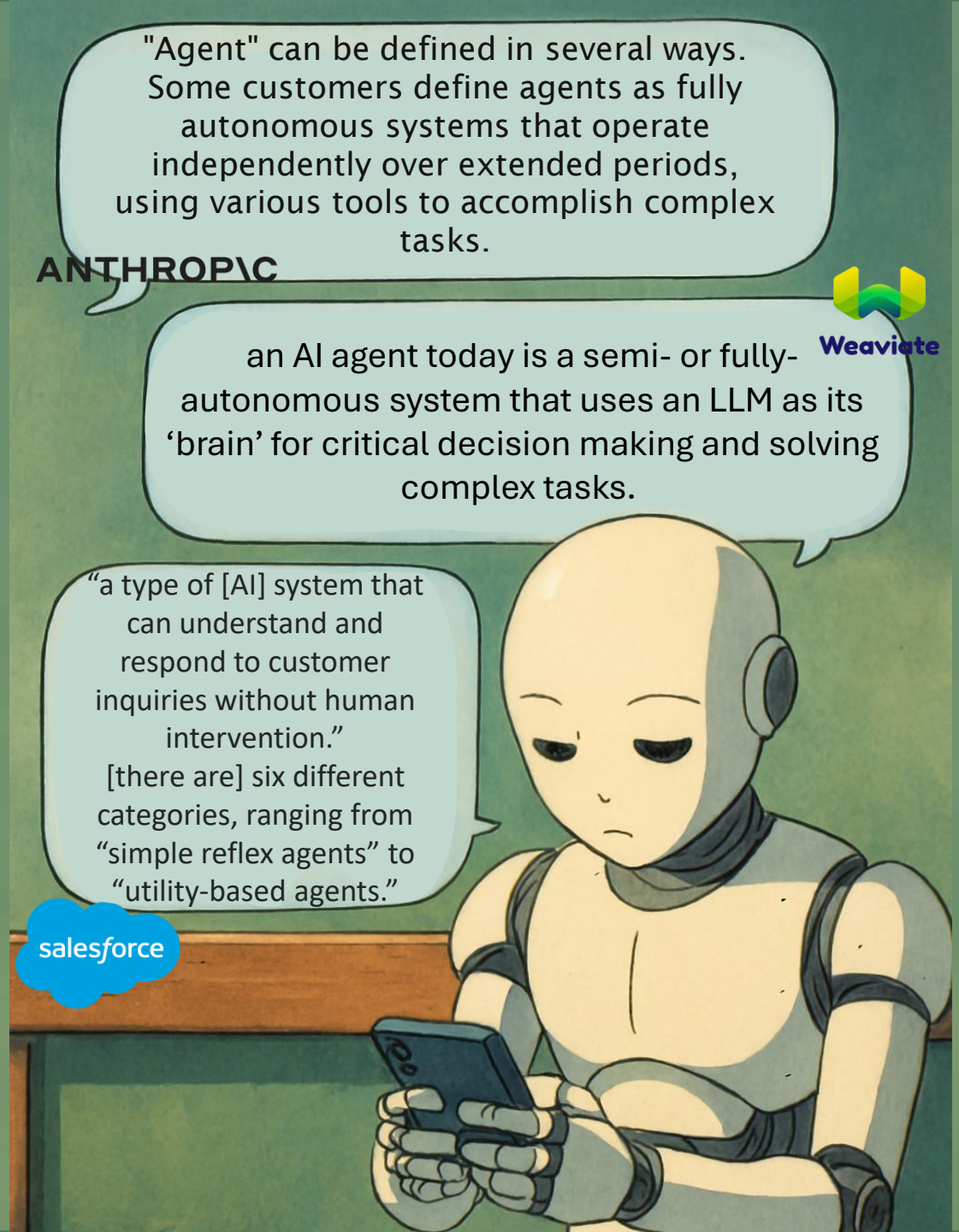

“We view agents as systems that independently accomplish tasks on behalf of users.”
– OpenAI, March 11 2025
https://openal.com/index/new-tools-for-building-agents/



- *also* OpenAI, Agent SDK initial commit on March 12 2025!


AI agents are not only a way to get more value for people but are going to be a paradigm shift in terms of how work gets done.


AI agents … can take actions to achieve specific goals, … guiding a shopper to the perfect pair of shoes… or supporting nursing staff with smoother patient hand-offs ….
An intelligent program capable of interacting autonomously with its environment to gather data, make decisions, and perform tasks.
servicenow


"Agent" can be defined in several ways. Some customers define agents as fully autonomous systems that operate independently over extended periods, using various tools to accomplish complex tasks.
ANTHROP\C
Weaviate
an AI agent today is a semi- or fully-autonomous system that uses an LLM as its 'brain' for critical decision making and solving complex tasks.
"a type of [AI] system that can understand and respond to customer inquiries without human intervention."
[there are] six different categories, ranging from "simple reflex agents" to "utility-based agents."
salesforce

AI Agents

(again)

What is an Agent

Intelligent assistants with memory, planning, and capabilities to act on behalf of the user or respond to events.

Microservice + AI + Autonomy

Take actions
Retrieve data, send an email, write a document, develop a program

---

Memory
Enable long term (days, weeks, months, years) interactions and plan execution

---

Planning
Decompose complex tasks into smaller tasks

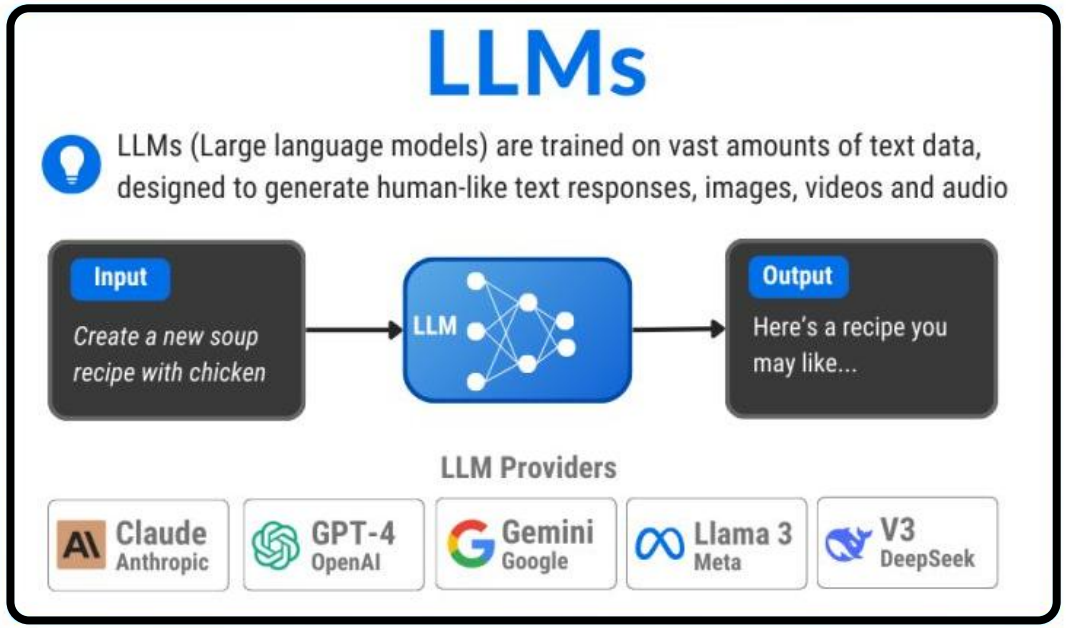
Orchestration
Coordinate the execution of agent actions

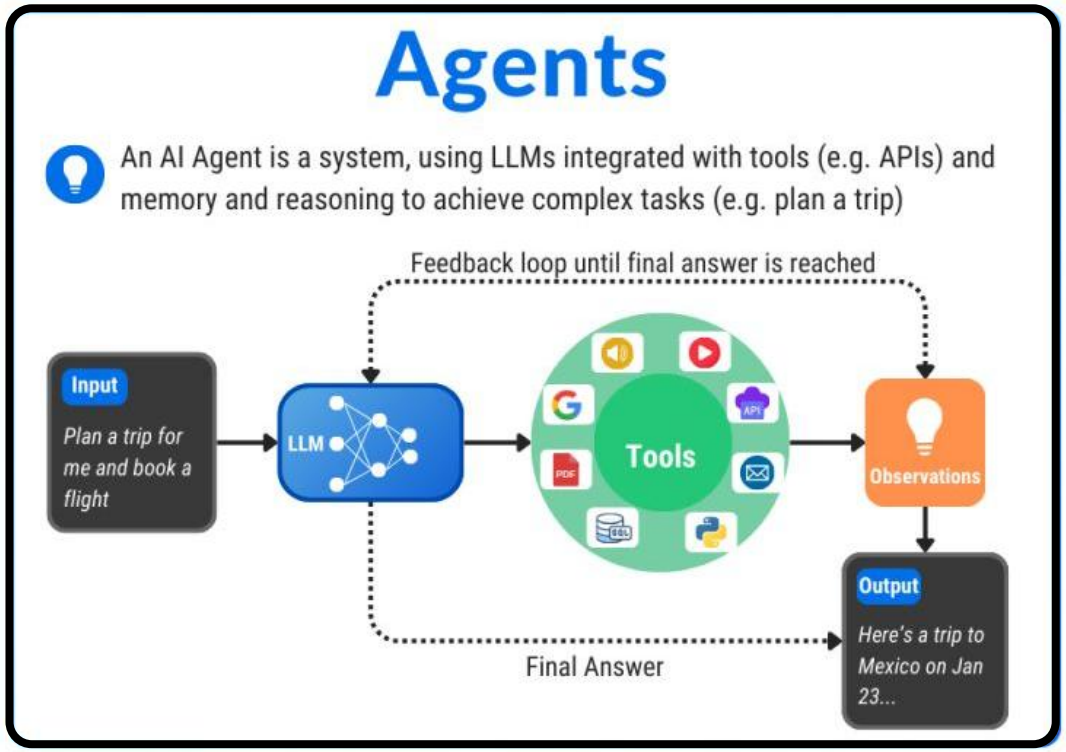
Interactive/Event-Driven
Can be delegated for long running tasks (hours days, weeks, months). Supervision is dependent on the task and desired pattern.

LLM vs Agent: Why Agents

	LLM	AI Agent
🧠 Mental Model	Predicts next word / response	Goal-directed reasoning + planning
🎯 Autonomy	No autonomy (reacts to prompts)	Can act independently with a goal
🧰 Tool Use	Needs explicit function calling	Uses tools strategically, often with memory
🔄 Memory	Stateless (unless engineered in)	Remembers, reflects, adapts
⛓️ Workflow	Single-step / looped interaction	Multi-step, iterative task execution
👥 Role	Copilot, assistant	Worker, delegate, executor

LLM vs Agent


LLMs
LLMs (Large language models) are trained on vast amounts of text data, designed to generate human-like text responses, images, videos and audio
Input
Create a new soup recipe with chicken
LLM
Output
Here's a recipe you may like...
LLM Providers
Claude
Anthropic
GPT-4
OpenAI
Gemini
Google
Llama 3
Meta
V3
DeepSeek


Agents
An AI Agent is a system, using LLMs integrated with tools (e.g. APIs) and memory and reasoning to achieve complex tasks (e.g. plan a trip)
Feedback loop until final answer is reached
Input
Plan a trip for me and book a flight
LLM
Tools
Observations
Output
Here's a trip to Mexico on Jan 23...
Final Answer

LLMs with iterative workflows are better

A study carried out by the team at deeplearning.ai, published in March 2024, revealed that for a specific coding task, GPT-3.5 (zero shot) achieved an accuracy of 48.1%. In contrast, GPT-4 (zero shot) performed better with an accuracy of 67.0%. However, the enhancement from GPT-3.5 to GPT-4 is significantly overshadowed by the use of an iterative agent workflow. Notably, when integrated into an agent loop, GPT-3.5 can reach an accuracy of up to 95.1%.

Tools and Agents

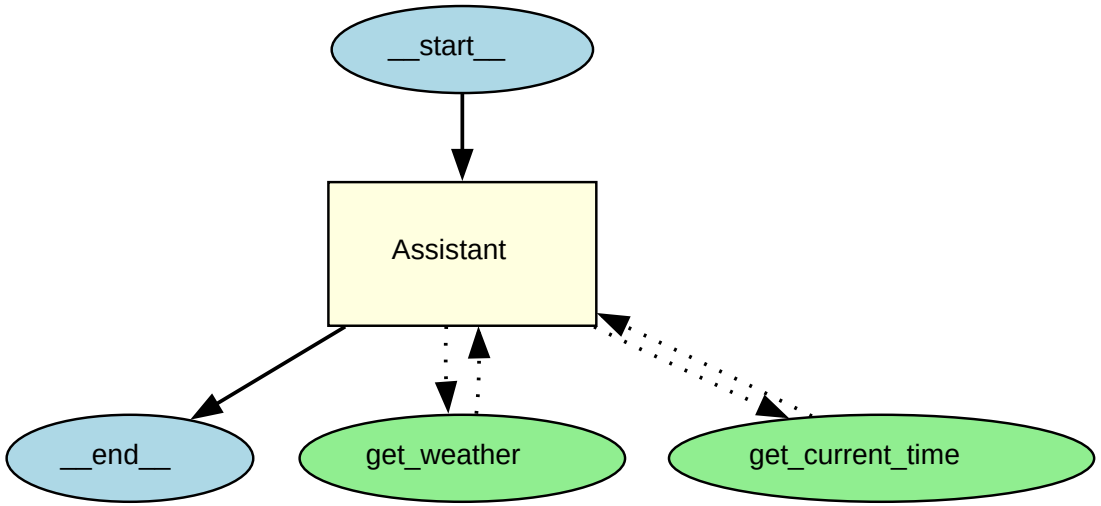
Tools are function calling for Agents.

Tools enable agents to perform various actions, such as retrieving data, executing code, accessing external APIs, and utilizing computer resources.

Frameworks offer built in tools such as Web and Files Search.

You can also define your own functions as tools to be executed with Agents.

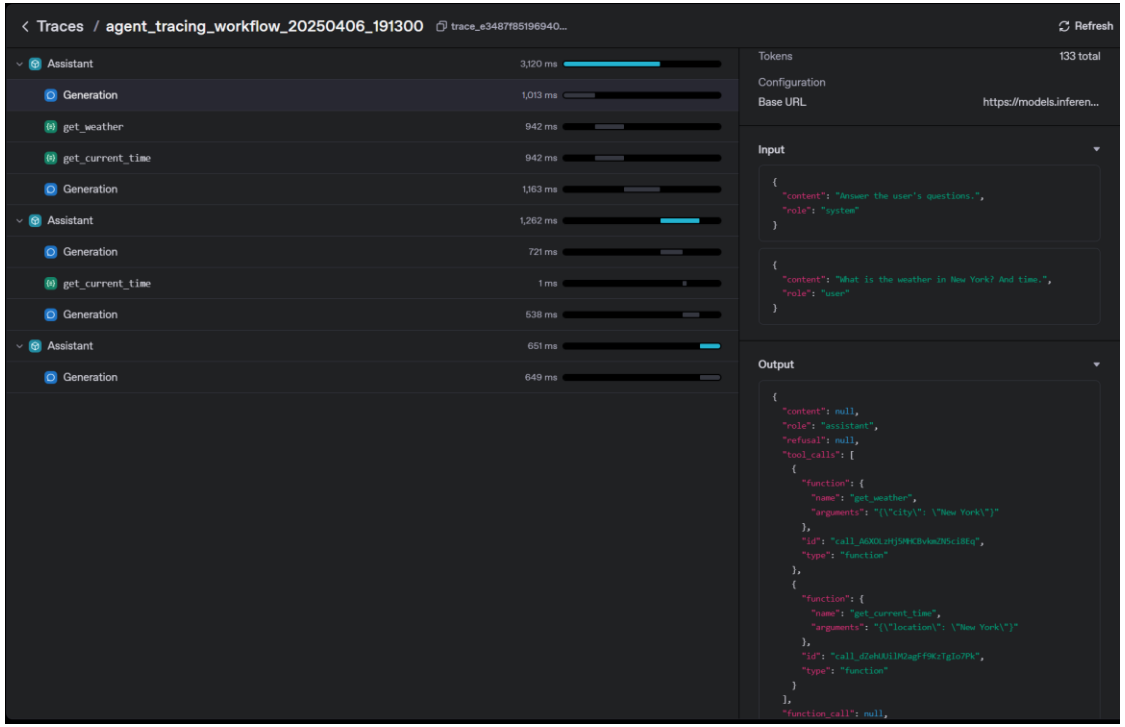
Agents simplify and abstract the execution of tools when compared to using function calling in LLMS.



Observability and Monitoring

Tracing provides several important benefits for AI agent development and operation:

- Debugging and Troubleshooting: Trace the execution path of agents to identify issues in complex workflows
- Performance Monitoring: Track latency, token usage, and other metrics to optimize agent performance
- Transparency: Gain visibility into how agents make decisions and use tools
- Auditability: Create records of agent activities for compliance and governance purposes
- User Experience Improvement: Analyse patterns in agent behaviour to improve prompts and tool designs



Code Exercises: Agents

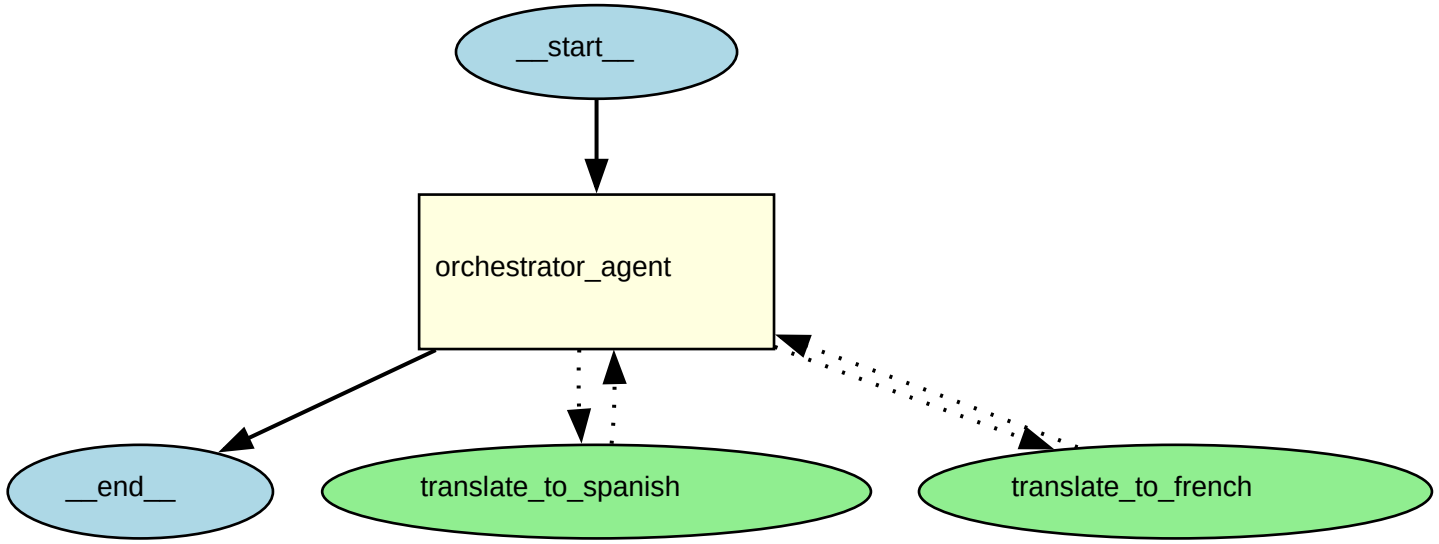
Notebook	Objective
03_hello_agent	• Create an Agent using OpenAI Agents SDK • Use github models (gpt 4o , gpt 4o mini)
04_agents_with_tools	This notebook demonstrates how to create agents that can use tools to gather information and perform tasks, a key capability for building useful AI assistants.
05_agent_tracing	This notebook demonstrates how to implement and use tracing capabilities in AI agent applications.

Orchestrating Agents

- Orchestration
  - Agents as tools
  - Handoffs

- Recommendations and best practices:
  - Good prompts
  - Structured Output
  - Context size and max tokens
  - Specialised Agents
  - Monitor the app
  - Invest in evals

Agent as a tool



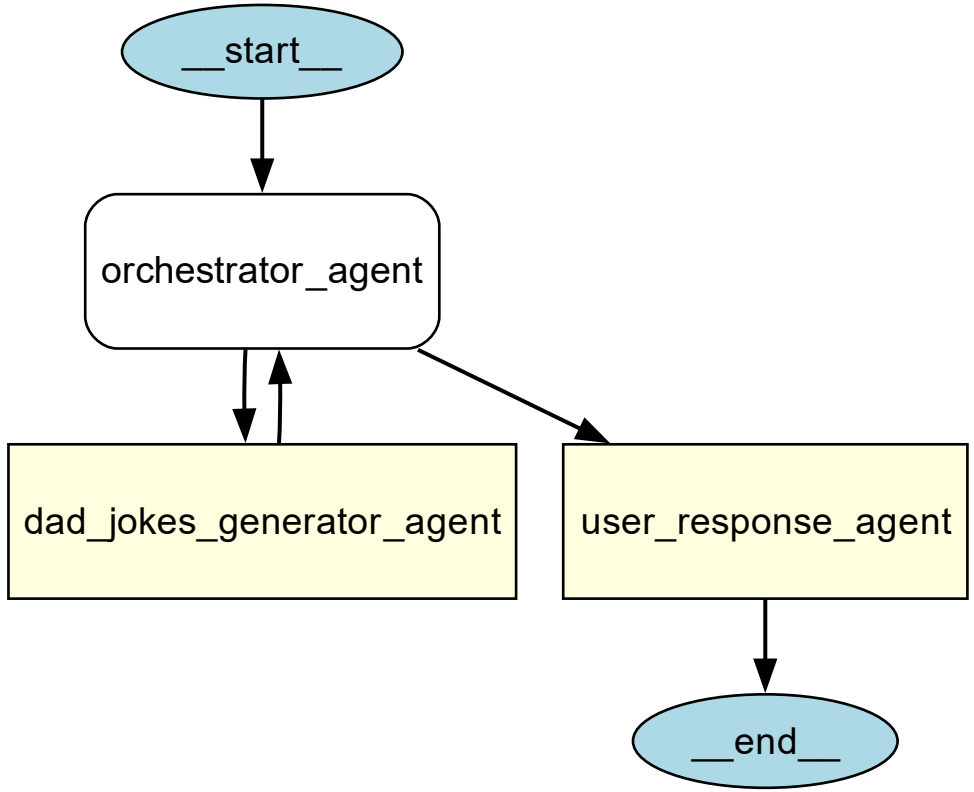
The tool agent goes off and runs on its own and then returns the result to the original agent.

When agents are used as tools, only the needed input data is passed on to the tool agent, not the entire context and message history. That’s a useful approach when agents do not require to take control of the workflow context.

Handoffs

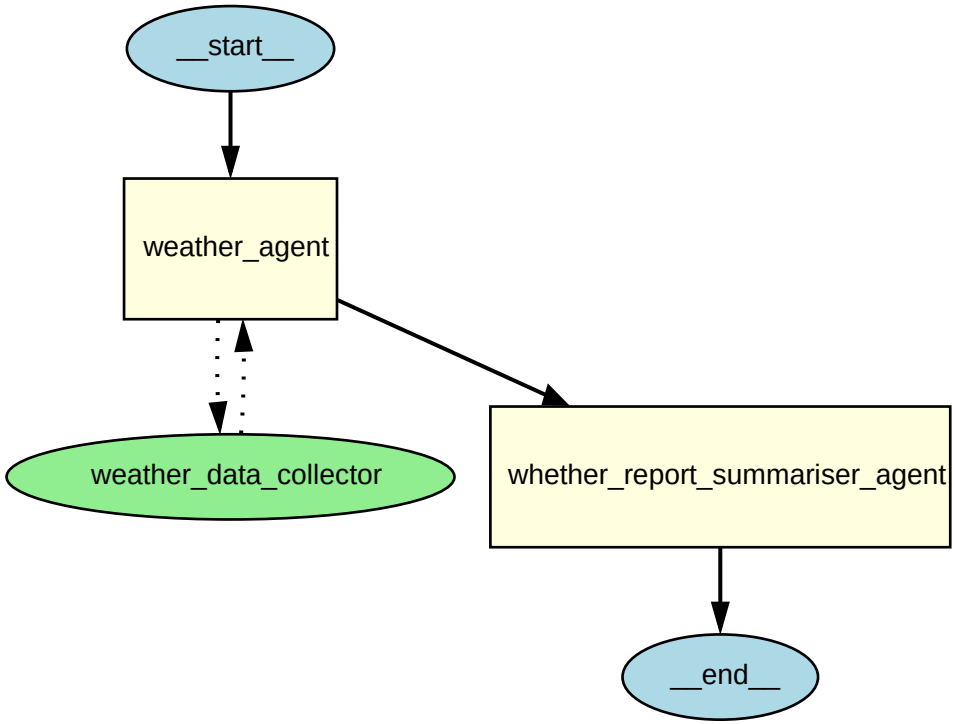
Handoff delegates the task to another agent.

The full context and control is passed on to the new agent.



Agent as Tools vs Handoffs

- **Handoffs**:
  - Great for sequential tasks and workflows.
  - Pass entire context between agents.
  - Easy to conceptualize linear agent flows.
- **Agents as Tools**:
  - Offer dynamic, conditional orchestration.
  - Allow dynamic control of multiple specialized agents.
  - Scalable and extensible.



It is expected to have to use both in creating agentic workflow.

When would you combine the two patterns?

Code Exercises: Tools and Handoffs

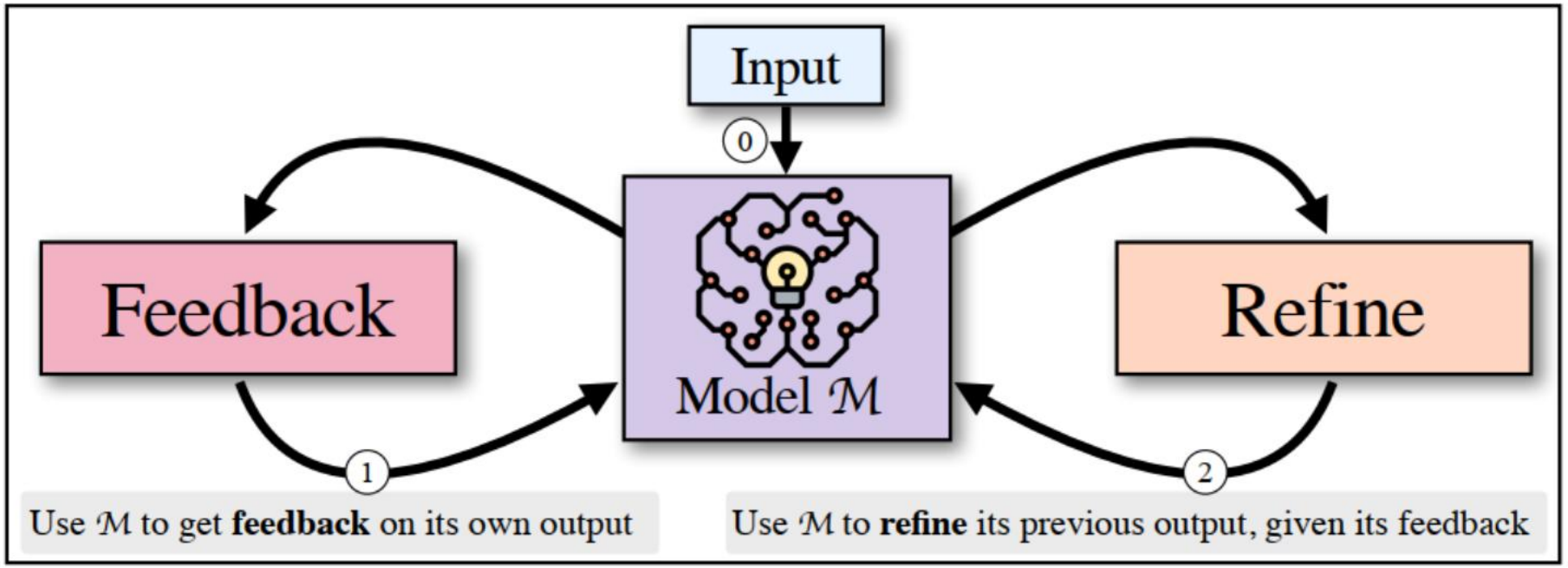
Notebook	Objective
06_agent_as_tool	Create a workflow with agents as tools
07_agent_handoffs	Implements the Agents Handoff workflow Demonstrate the use of structured output

Agentic Workflows

Agentic workflows combine the use of multiple AI Agents to complete a task.

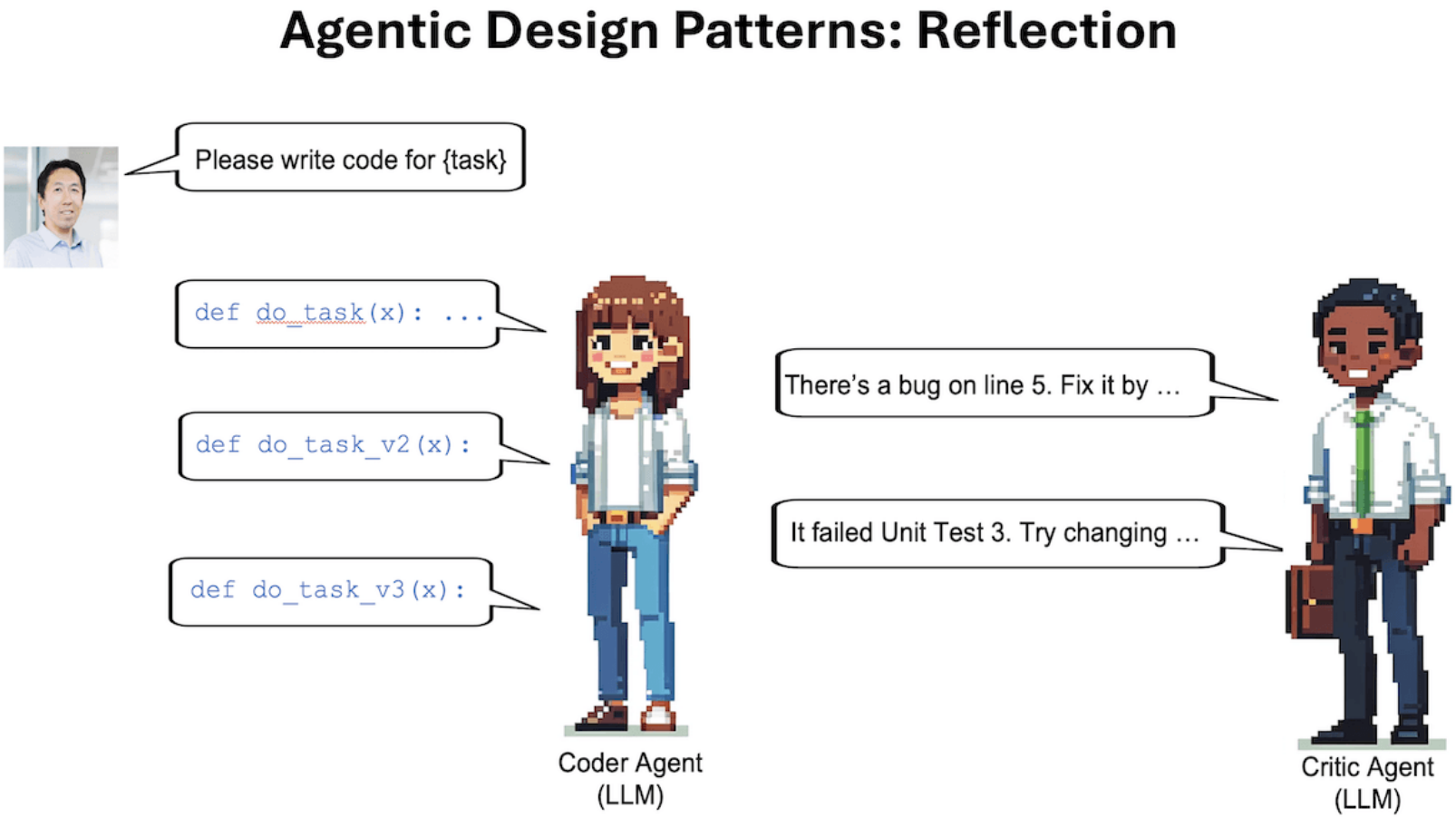
- Deterministic: using code to carry out the workflow steps
- LLM Workflow: utilise the Agent’s LLM to plan and execute the workflow using as_tools and/or handoffs

Reflection



Introduced in May 2023, SELF-REFINE (https://arxiv.org/pdf/2303.17651) is a method for enhancing initial outputs from large language models (LLMs) through a process of iterative feedback and refinement. The core idea involves generating an initial response using an LLM, then having the same LLM critique its own output and use that feedback to progressively improve the response over multiple iterations.

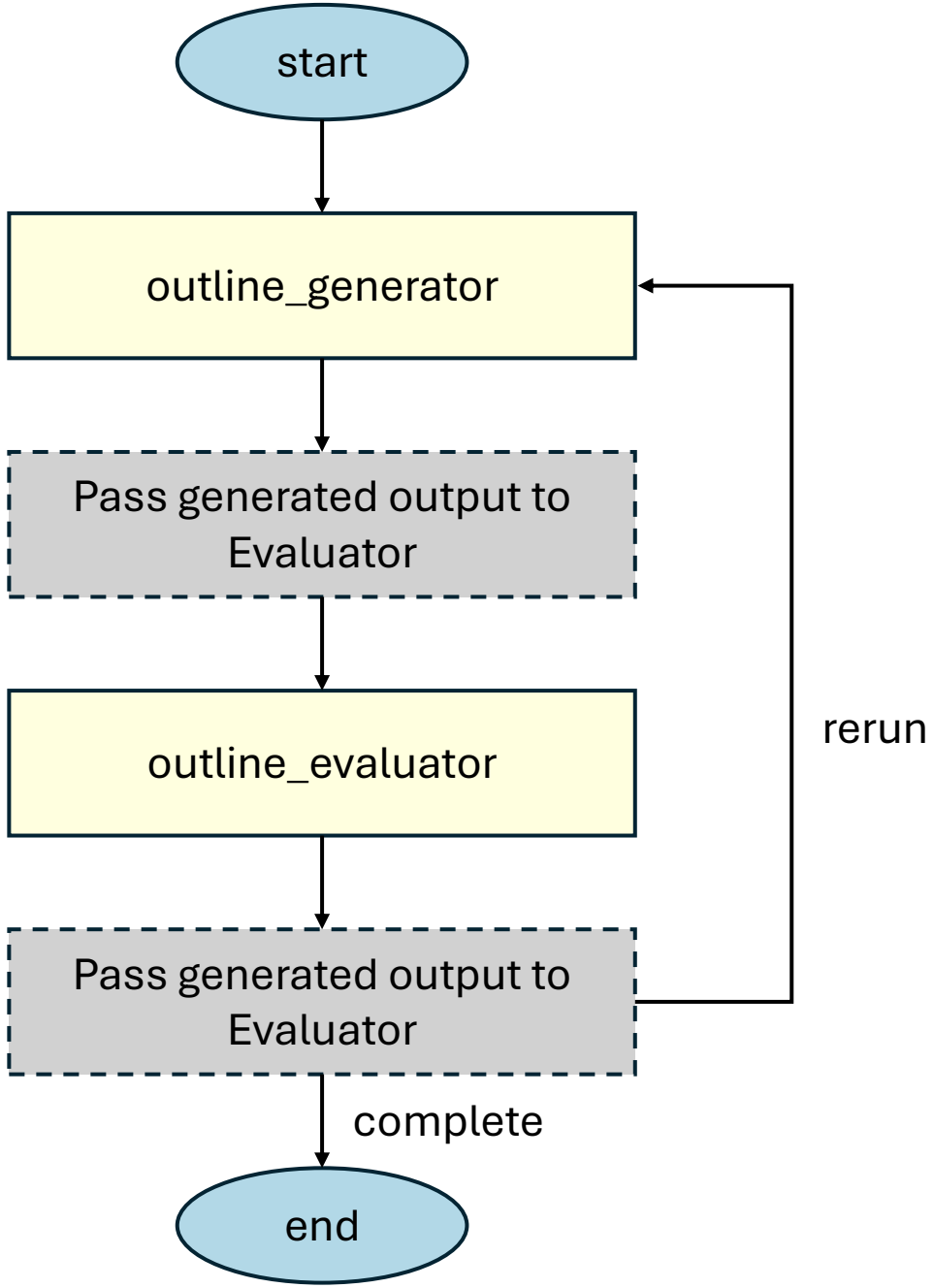
Reflection



Reflection is part of agentic workflows, where the model is prompted multiple times rather than generating a single final output. It encourages an LLM to evaluate and enhance its own outputs, drawing inspiration from how humans learn and improve.

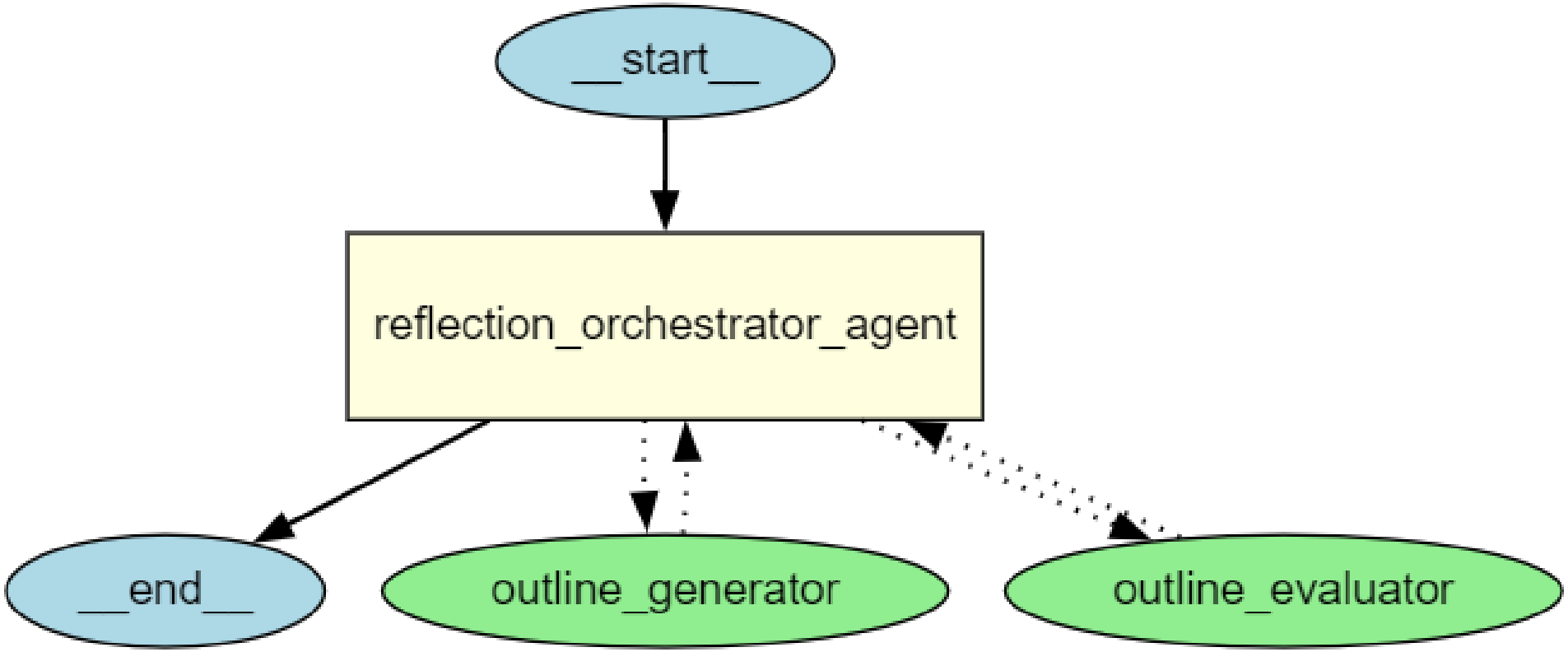
https://www.deeplearning.ai/the-batch/agentic-design-patterns-part-2-reflection

Reflection (deterministic)



code\10_reflection_pattern.ipynb

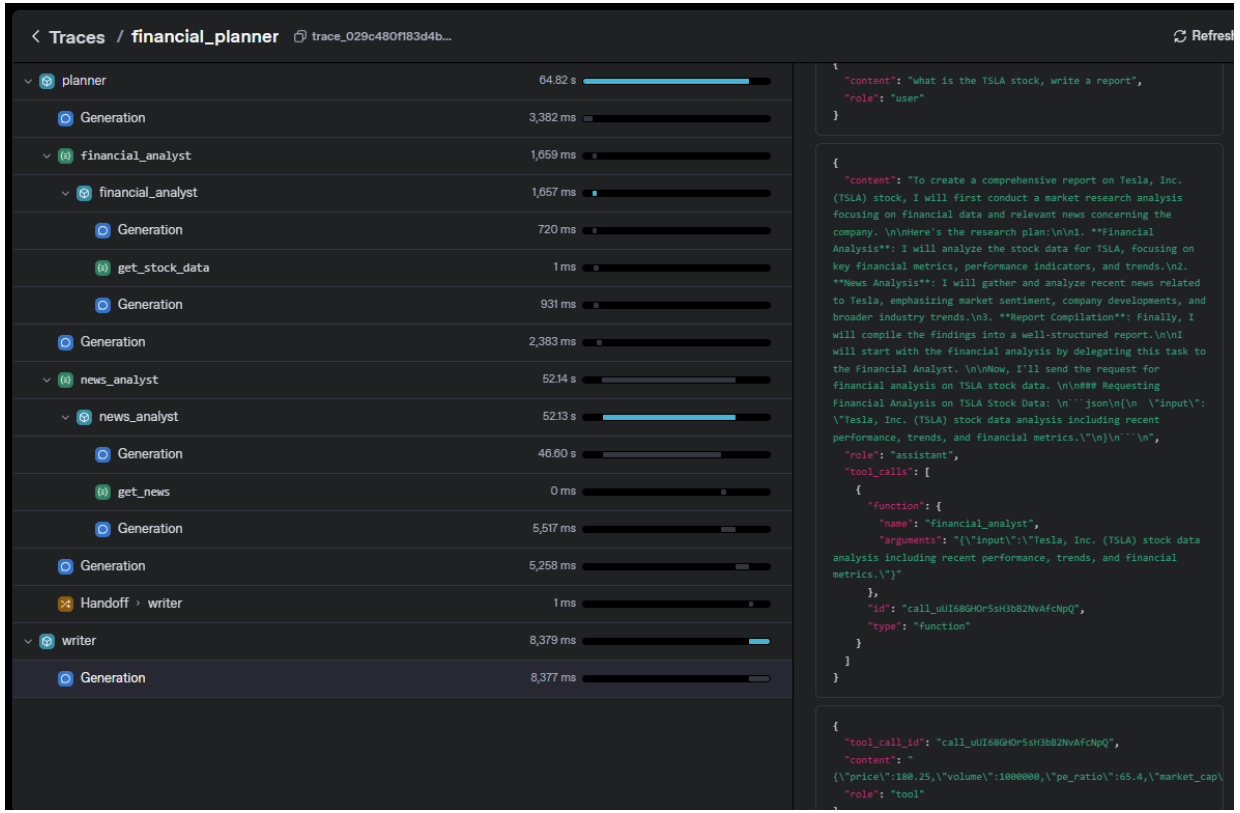
Reflection (as tools)

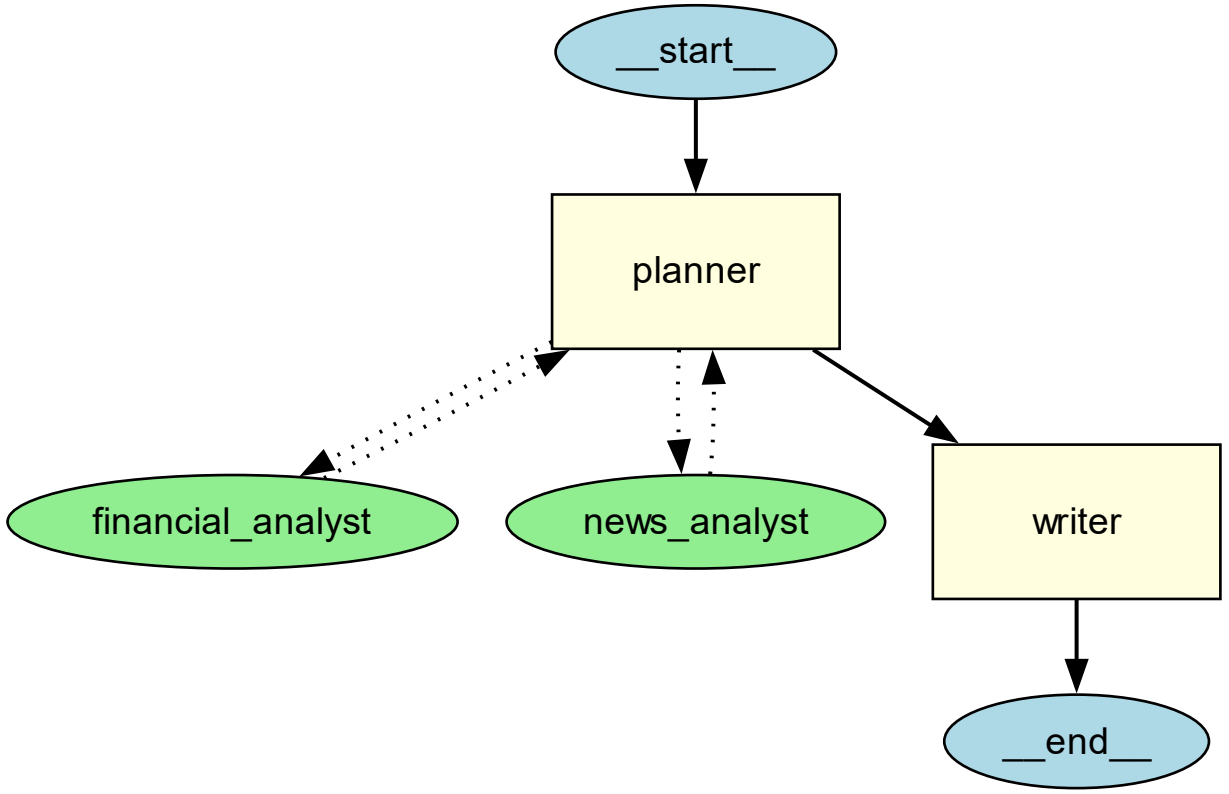


Planner

- The LLM comes up with, and executes, a multistep plan to achieve a goal (for example, writing an outline for an essay, then doing online research, then writing a draft, and so on).
- Multi-agent collaboration: More than one AI agent work together, splitting up tasks and discussing and debating ideas, to come up with better solutions than a single agent would.

Planner



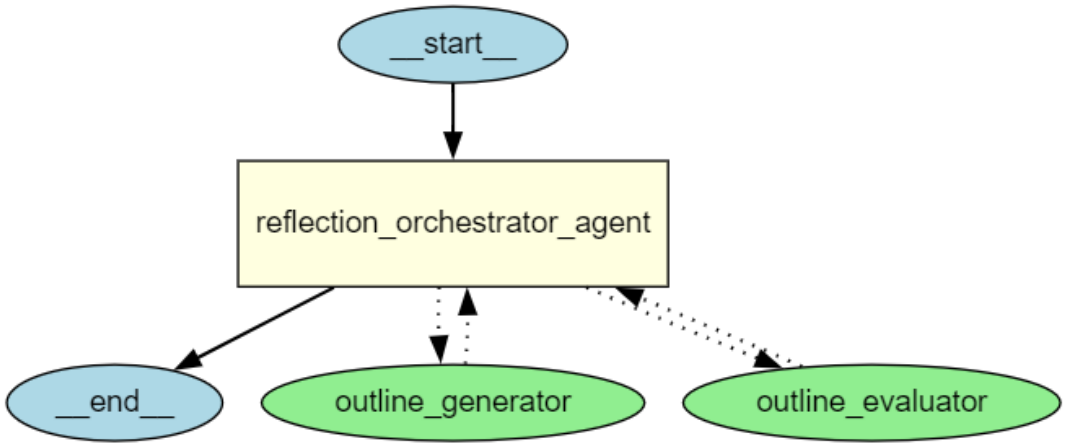


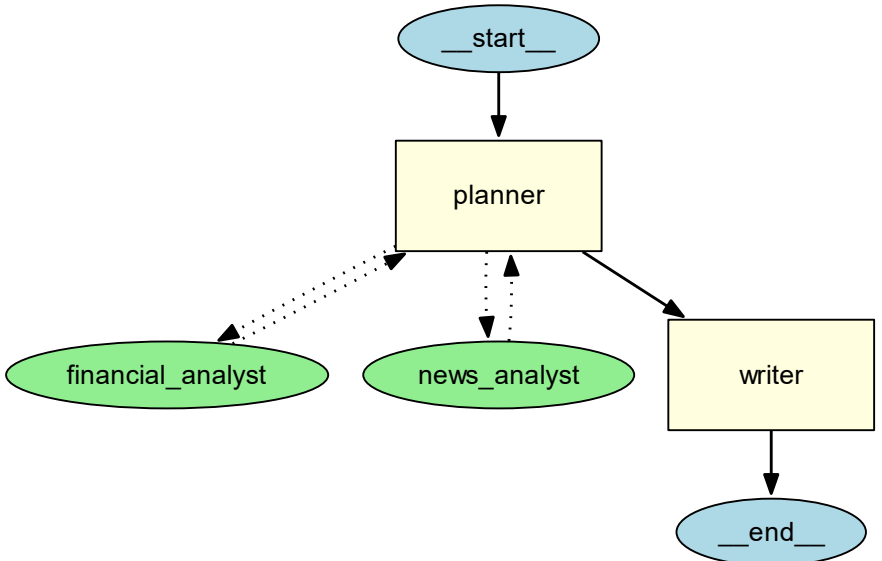
code\11_planner_as_tools_handoff.ipynb

[Traces - trace_029c480f183d4bc4aef7fd9e38efdc44 - OpenAI API]

Code Exercises: Agentic Workflows

Notebook	Objective
10_reflection_pattern	Implement an Agentic Reflection Pattern with two agents to generate and review content and one orchestrator. Experiment with both the deterministic approach and stochastic method using LLM for planning.
11_planner	Implement a fincial planner





Responsible Use Considerations?

Privacy, Security and Data Protection, specially as data traverses multiple systems and agents

Bias and Fairness, just as with all AI/ML systems, but now LLMs and Agentic is offering new opportunities to automate processes with higher levels of autonomy

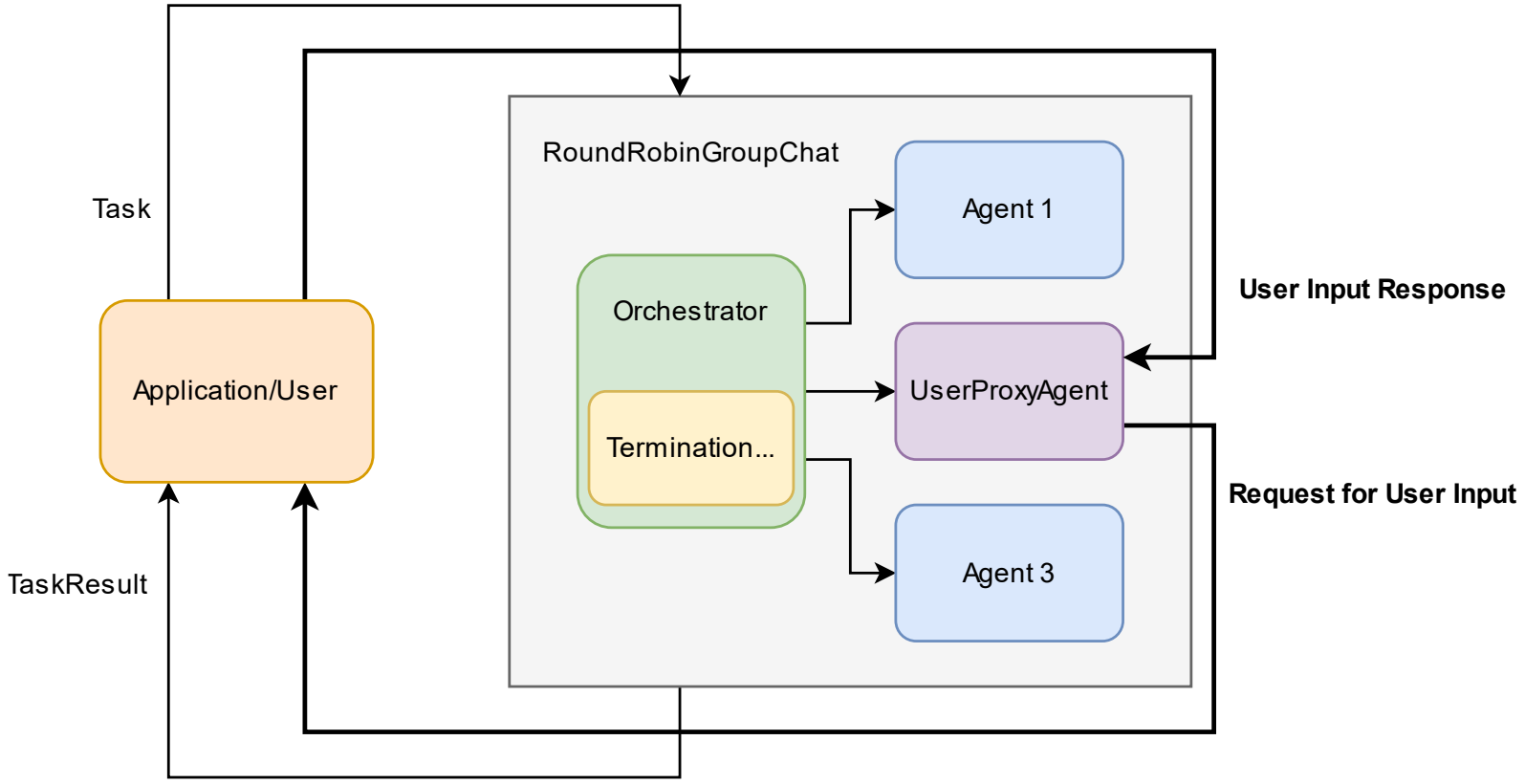
Accountability and Transparency: as multiple agents converse and cooperate to produce a result; how do we establish clear accountability and transparency mechanisms.

Can users understand and trace the decision-making process of the agents involved to ensure accountability, transparency and mitigate biases?

Unintended Consequences: the use of multi-agent conversations and automation in complex tasks may have unintended consequences. In particular, allowing LLM agents to define execution plans or equip them with tools to make changes in external environments.

How do we assess the potential risks and ensure that appropriate safeguards are in place to prevent harm or negative outcomes?

Human in the Loop



- You could intercept agent's operation with callback functions and hooks to request user input
- Higher level frameworks such as Autogen offer flexibility to delegate to a user proxy

Frameworks

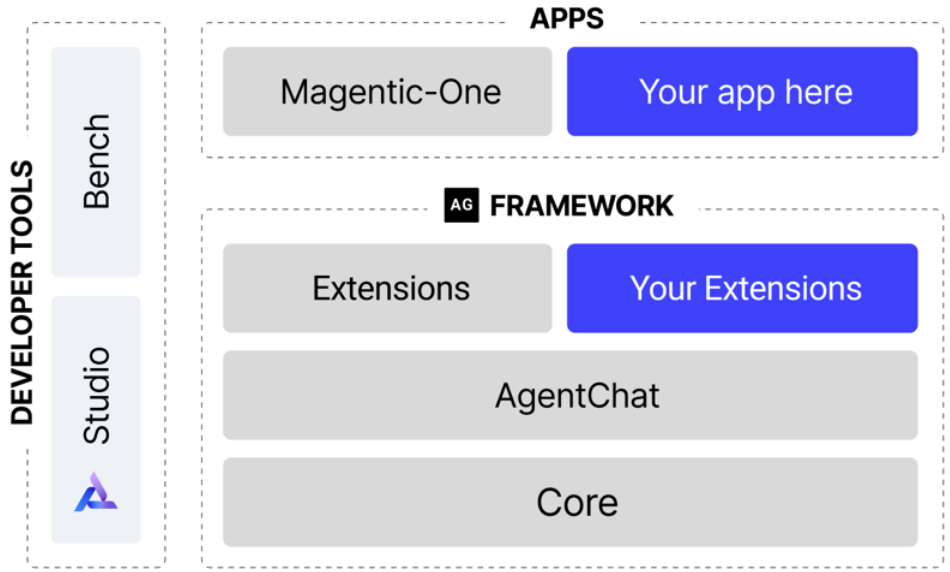
- OpenAI Agents
- Autogen
- Semantic Kernel
- Langgraph

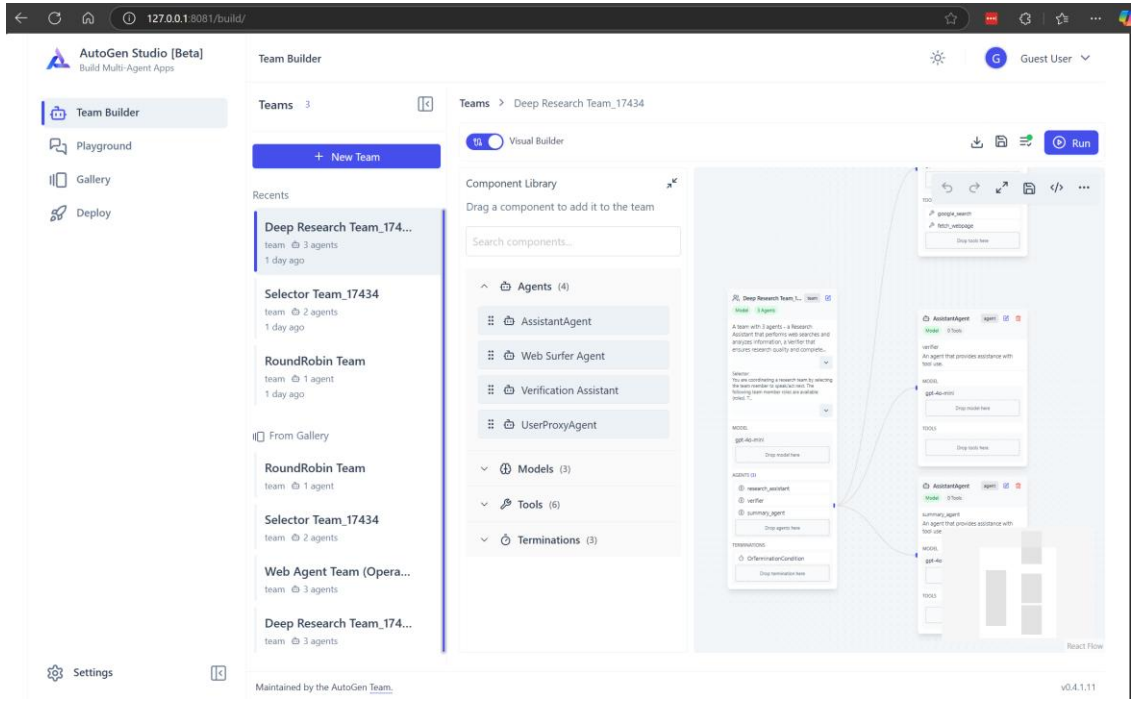
- Azure AI Agent Service

Frameworks Comparison

Framework / Service	SDK/Language	Agent Type	Tool Integration	Memory Support	Workflow Visualization	Ecosystem & Deployment	Maturity/Target
OpenAI Agents SDK	Python	Single & Multi-Agent (handoffs)	Python functions via decorators; built-in OpenAI tools	In-context & via external vector stores	None (code-centric)	Self-hosted; integrates with OpenAI API & responses	New (2025); experimental to production
Microsoft AutoGen	Python (primary), Experimental .NET	Multi-Agent (async chat, human-in-loop)	Custom skills & code execution; extensible via Azure tools	Built-in conversation memory; pluggable long-term	Yes (AutoGen Studio GUI)	Self-hosted; deployable on Azure (containers, Functions)	Research-driven; growing enterprise use
Semantic Kernel	C# (.NET), Python and Java	Primarily Single-Agent (with experimental multi-agent)	Plugins (native & semantic functions) with planner	Built-in semantic memory (with vector DB options)	Partial (via Copilot Studio)	Self-hosted; integrates tightly with Azure/OpenAI	Established SDK (since 2023); enterprise
LangGraph	Python & JavaScript	Graph-based single/multi-actor	Leverages LangChain tools & custom function nodes	Shared state across nodes; optional DB persistency	Yes (LangGraph Studio/visual editor)	Self-hosted or via managed LangGraph Platform	New (late 2023); production-grade control
Azure AI Agent Service	Multi-language SDKs (Python, .NET, etc.)	Single-Agent with multi-agent threads	Built-in tools (Bing, code interpreter, Azure Functions); custom via OpenAPI	Managed conversation state (“Threads”)	Yes (Azure AI Studio/Foundry UI)	Fully managed on Azure; enterprise-grade with security	Preview; enterprise & turnkey solutions

Autogen





https://microsoft.github.io/autogen
https://devblogs.microsoft.com/autogen/autogen-reimagined-launching-autogen-0-4/

Design Practices

- What is the process
- Reduce the cognitive load of the LLM
  - Deterministic steps and dynamic steps
- Design and product thinking
  - Personas
  - User interactions
  - Data
  - Instructions

Simple

Advanced

Tasks

Take actions when asked, automate workflows, and replace repetitive tasks for users

Constrained Processes

constrained business processes, predefined rules and guidelines to optimise workflow and enhance decision-making.

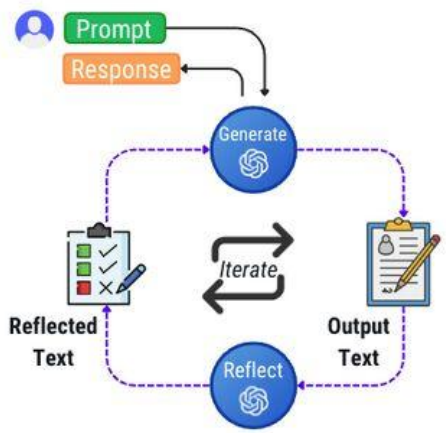
Dynamic

Operate independently, dynamically plan, orchestrate other agents, learn and escalate

Agents vary in levels of complexity and capabilities depending on your need

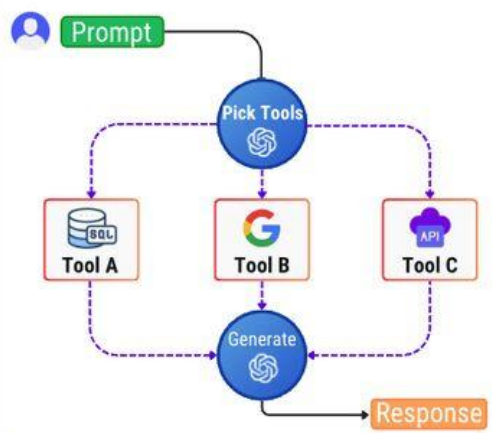
Reflection Pattern

Learns from past outputs and adjusts future responses similar to human reflection.



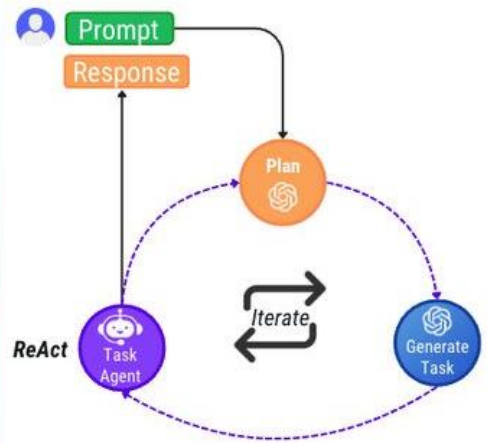
Tool Usage Pattern

AI selects and applies external tools (e.g. Google search) to enhance capabilities.



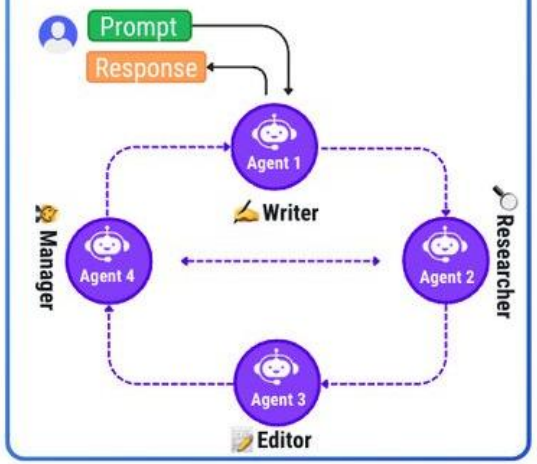
Planning Pattern

AI breaks a complex task into smaller steps and executes to complete the task.

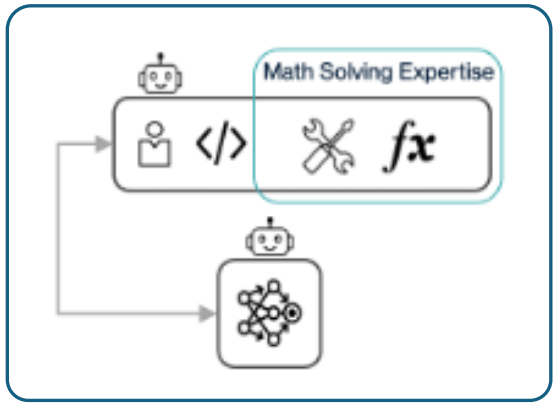


MultiAgent Pattern

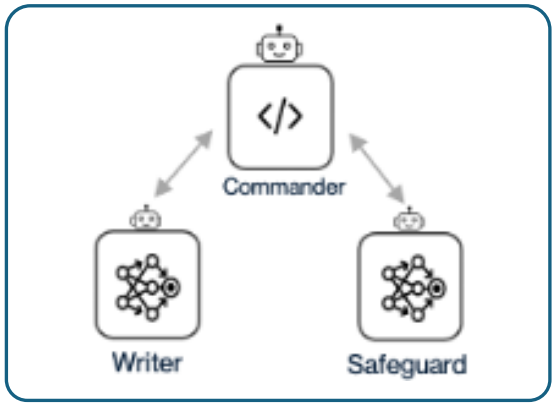
A set of multiple agents collaborate to solve a complex problem.



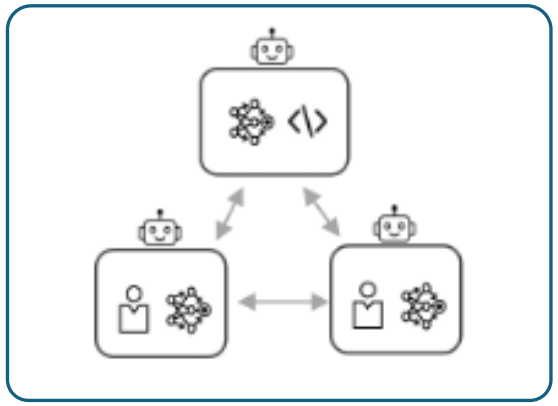
Specific Design Patterns for Agents



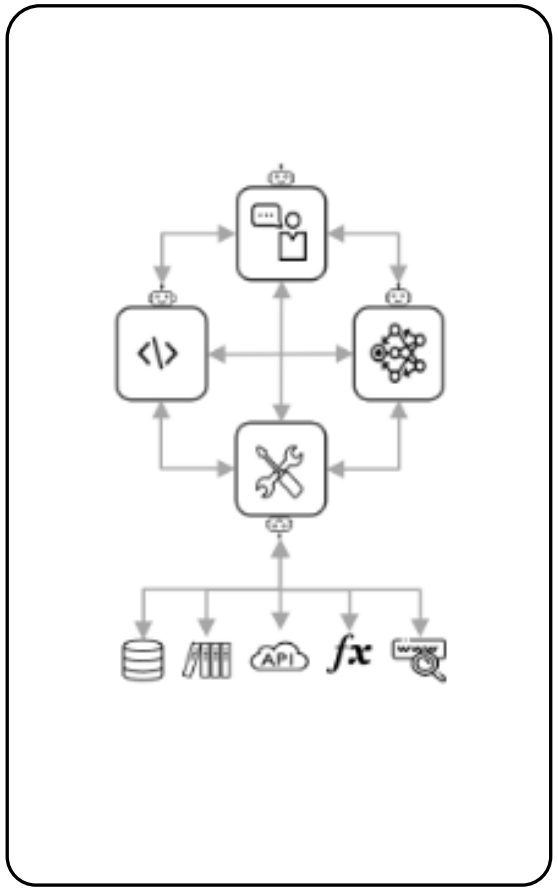
Math Solving

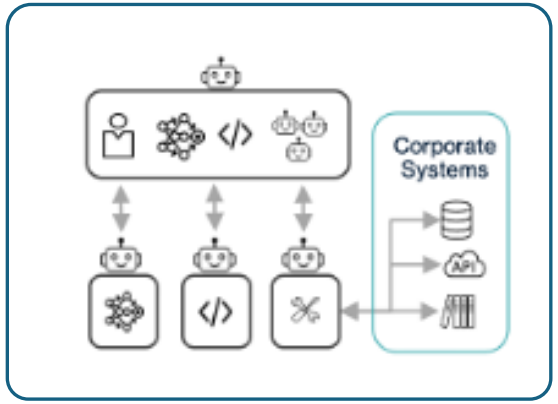


Multi-agent coding

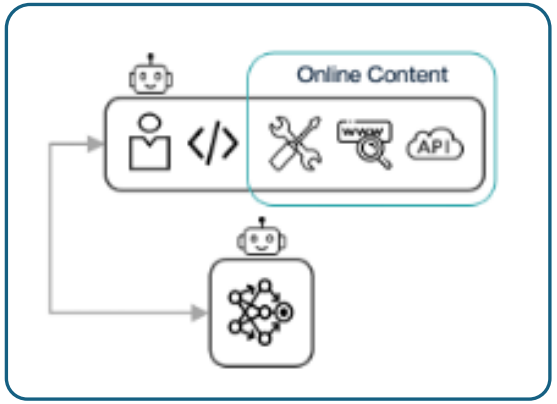


Conversational interactions

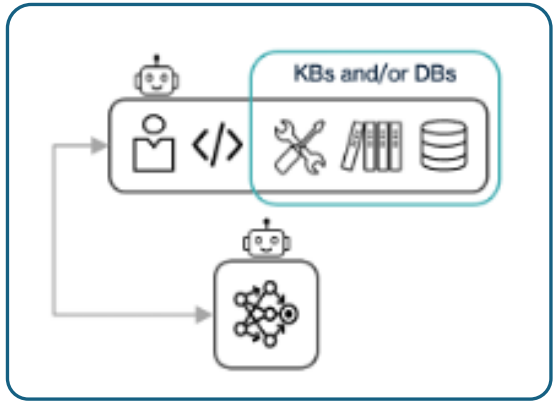




Business Process Automation



Online Decision Making



Retrieval-augmented Generation

Custom Use-case

DESIGN YOUR AGENTS

Agent

What is your agent called?

Draw your AI agent's avatar below

What team or business unit would this agent be aligned to? Who would directly interact with and maintain this agent?

Persona

How should the agent behave? What is its system message?

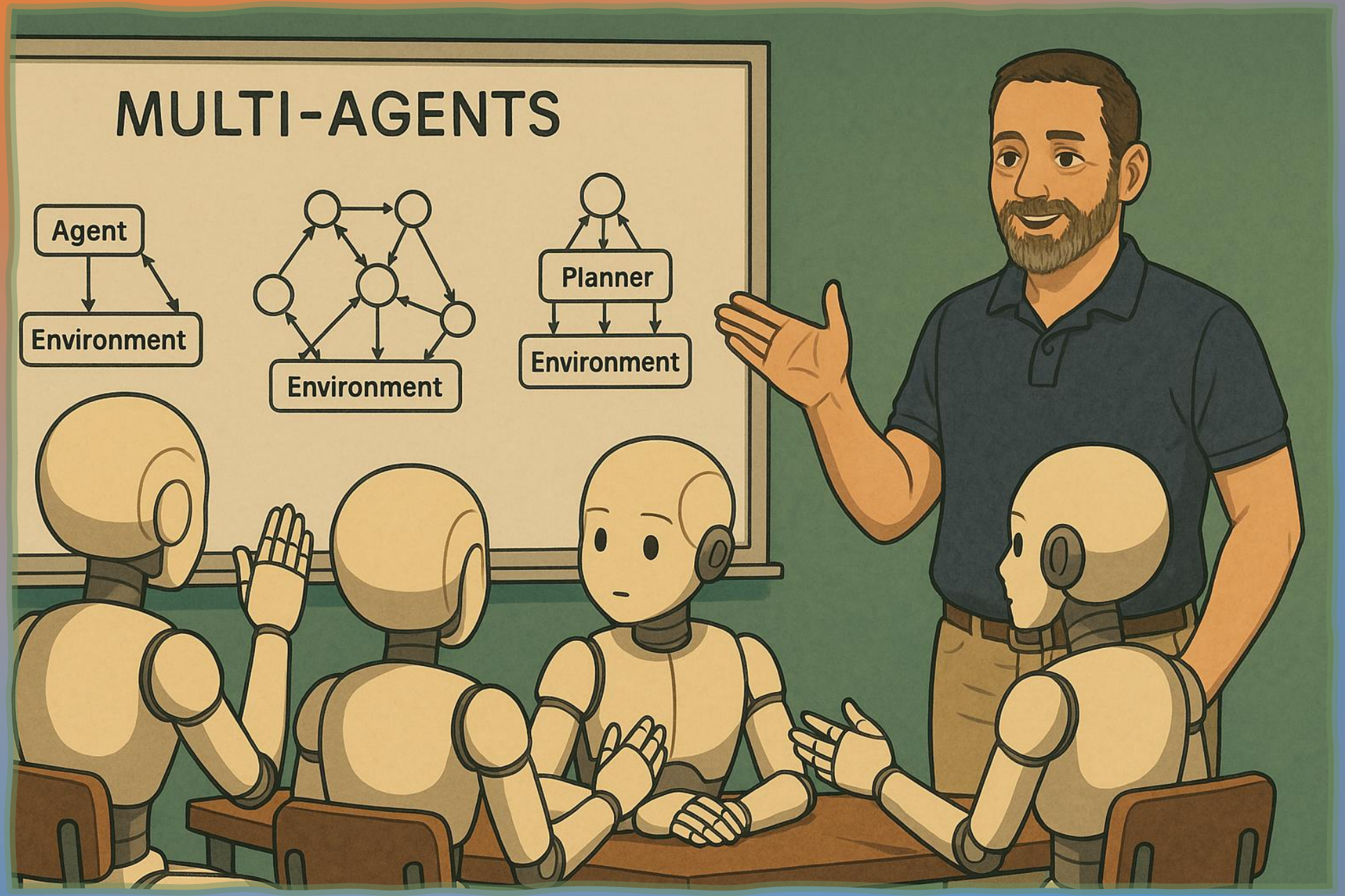
How would you describe the agent, so it is discoverable by other agents?

Data sources

What data does this agent have available to it? For example, unstructured data in a vector DB, structured data in a NoSQL or SQL DB, or other APIs and files.

Tools/functions

What tools and functions will the agent need to call?


MULTI-AGENTS
Agent
Environment
Environment
Planner
Environment