Microsoft

Introduction to AI Agents and Multi-Agent Systems with Semantic Kernel

Henri Schulte
AI Partner Solution Architect
Microsoft Denmark

March 2025

Microsoft

Henri Schulte

Azure AI Partner Solution Architect
Microsoft Denmark

/in/henrischulte

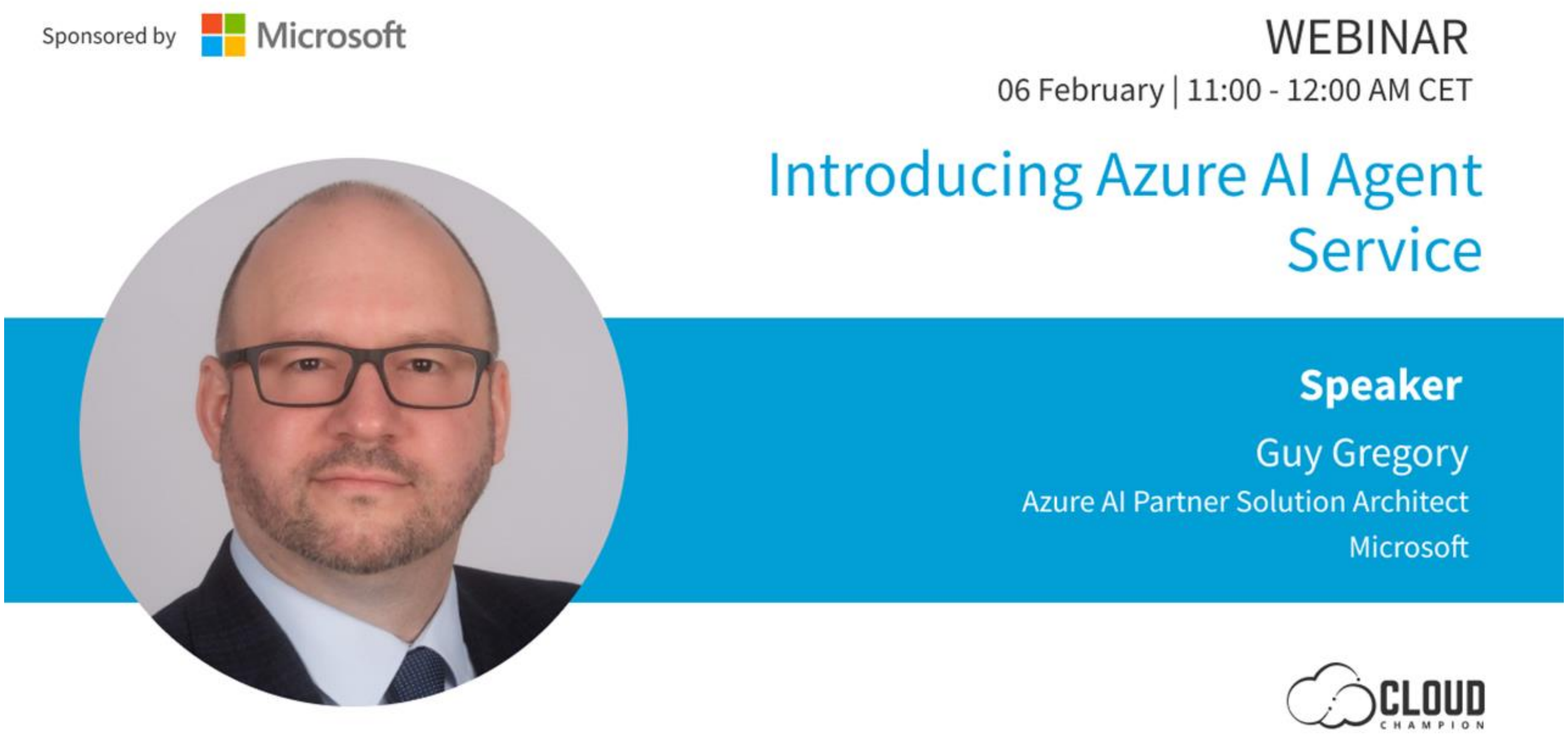
Housekeeping

Slides are available at
aka.ms/IntroToMultiAgents

Feel free to submit questions for the Q&A at the end!

Previous session

Highly recommended to watch this session



Sponsored by Microsoft

WEBINAR

06 February | 11:00 - 12:00 AM CET

Introducing Azure AI Agent Service

Speaker

Guy Gregory

Azure AI Partner Solution Architect

Microsoft

CLOUD CHAMPION

https://aka.ms/cloudchampion/agents

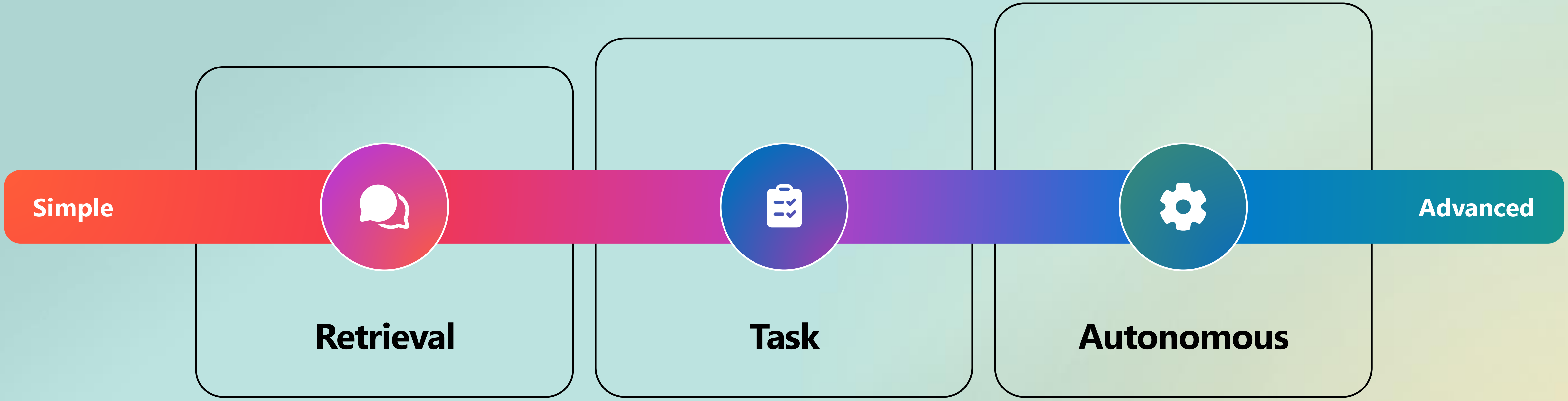
What to expect from this session

- Quick recap of AI Agents
- Why multi-agents?
- Semantic Kernel Agent Framework
- Let's build a multi-agent system

Quick recap of AI Agents

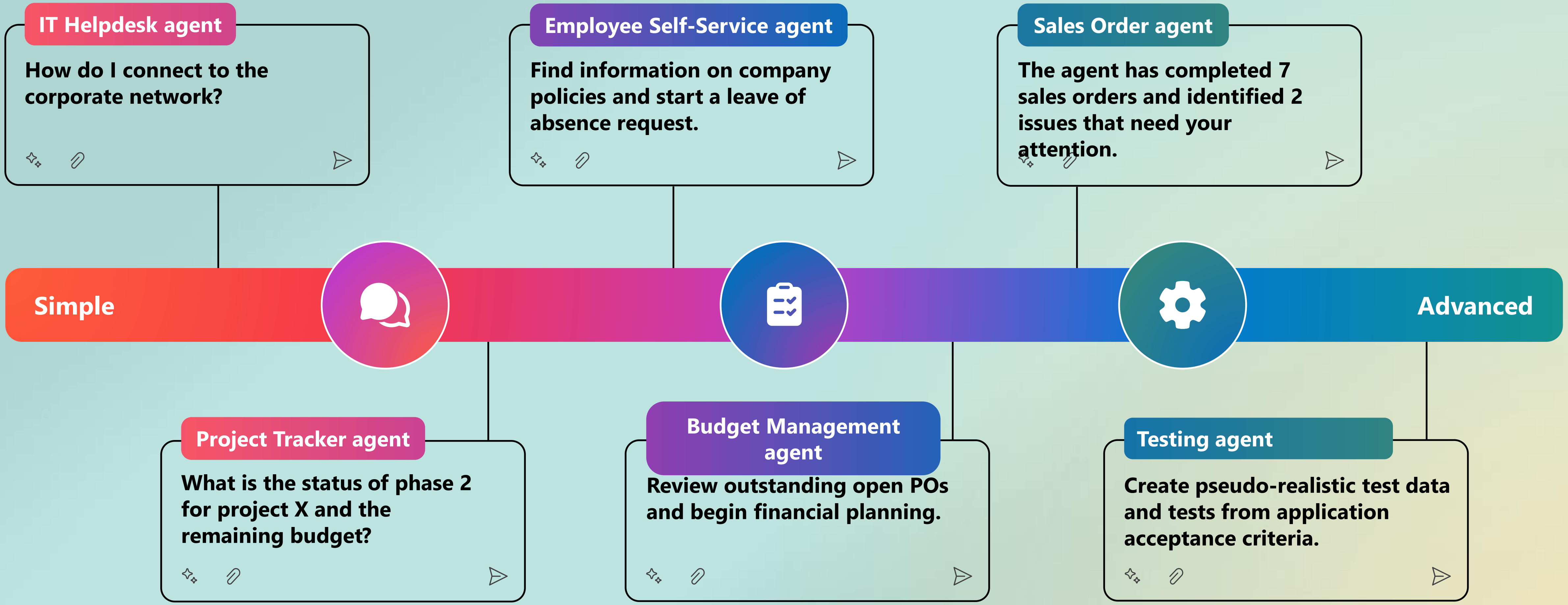
What are agents?

Agents are programs that use AI to automate and execute business processes, working alongside or on behalf of a person, team or organization

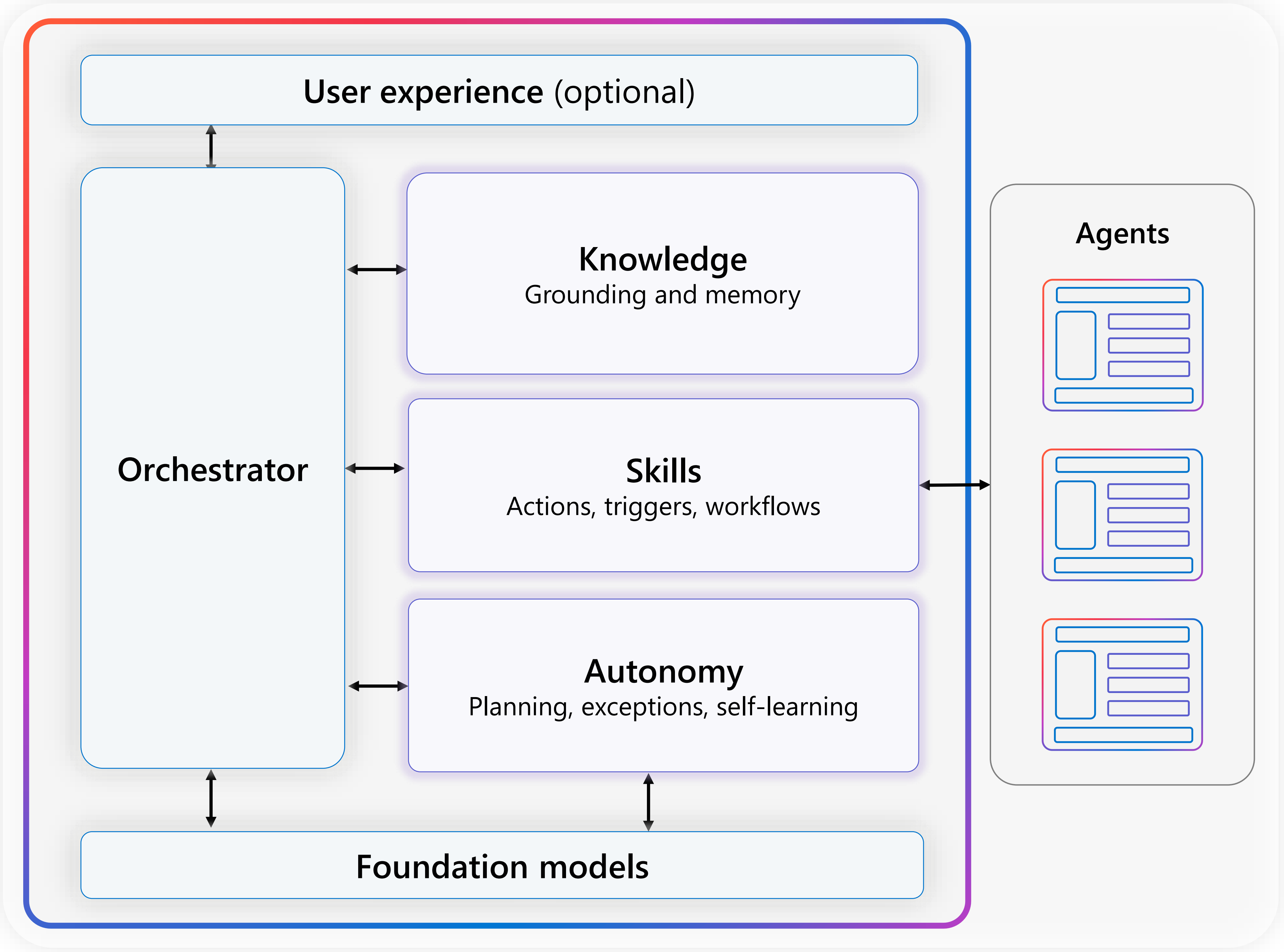


Agents vary complexity and capabilities depending on your need

What agents can do for your customers


IT Helpdesk agent
How do I connect to the corporate network?
Employee Self-Service agent
Find information on company policies and start a leave of absence request.
Sales Order agent
The agent has completed 7 sales orders and identified 2 issues that need your attention.
Simple
Advanced
Project Tracker agent
What is the status of phase 2 for project X and the remaining budget?
Budget Management agent
Review outstanding open POs and begin financial planning.
Testing agent
Create pseudo-realistic test data and tests from application acceptance criteria.

Agent architecture


User experience (optional)
Orchestrator
Knowledge
Grounding and memory
Skills
Actions, triggers, workflows
Autonomy
Planning, exceptions, self-learning
Foundation models
Agents

Public Preview

Azure AI Agent Service

Empower developers to securely build, deploy, and scale AI agents with ease

Rapid Development and Automation	Extensive Data Connections	Flexible Model Selection	Enterprise-grade Security

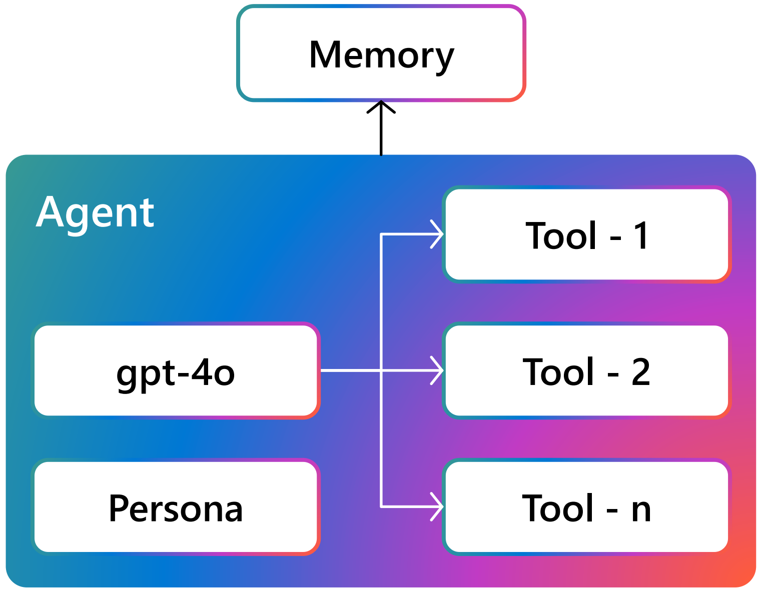
AI.Azure.com

Why multi-agents?

Challenges of building general purpose agents

Components of a general-purpose AI agent

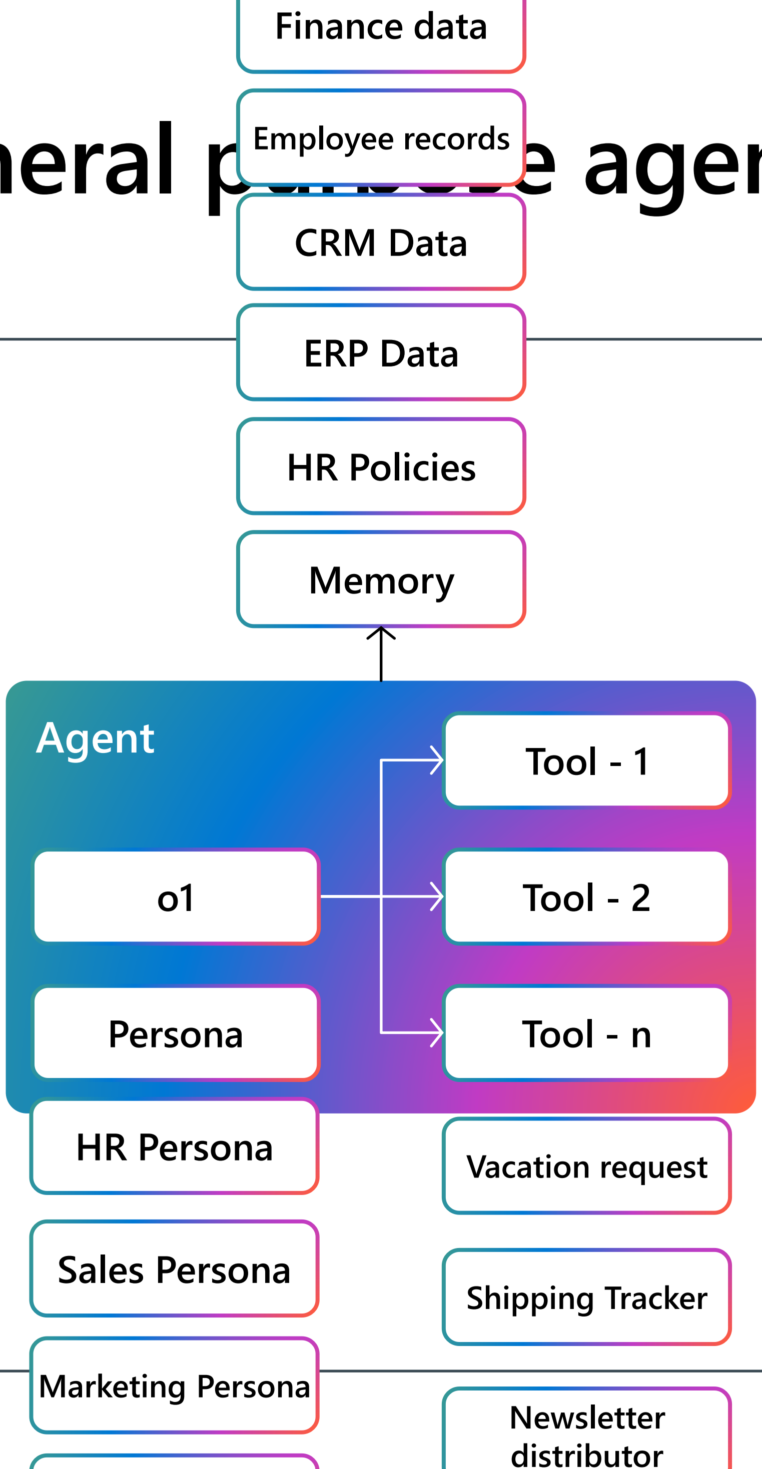
- **Model** (gpt4-o, o1 etc.)
- **Persona** (system prompt)
    - General instructions
    - Safeguards
    - Task-specific instructions
- **Tools** (function code calls)
    - General tools
    - Task-specific tools
- **Knowledge & Memory** (Grounding)
    - Conversation history
    - General knowledge sources (e.g., web)
    - Task-specific knowledge



Challenges of building general purpose agents

Components of a general-purpose AI agent

- **Model** (gpt4-o, o1 etc.)
- **Persona** (system prompt)
  - General instructions
  - Safeguards
  - Task-specific instructions
- **Tools** (function code calls)
  - General tools
  - Task-specific tools
- **Knowledge & Memory** (Grounding)
  - Conversation history
  - General knowledge sources (e.g., web)
  - Task-specific knowledge



Single Agent Architecture - Scaling

As the system grows you might run into scaling challenges

Too many tools. Tool hallucinations

Agent context (a.k.a. prompt) grows too much and it fails to follow instructions

Handling complex and dynamics tasks spanning different business domains

Multi agent architecture opportunities

Manageability – Modular agents reduce development and testing complexity

Predictability – More control over application flow using structured agents communication

Flexibility – Ease to incorporate new agents as solution domains increase

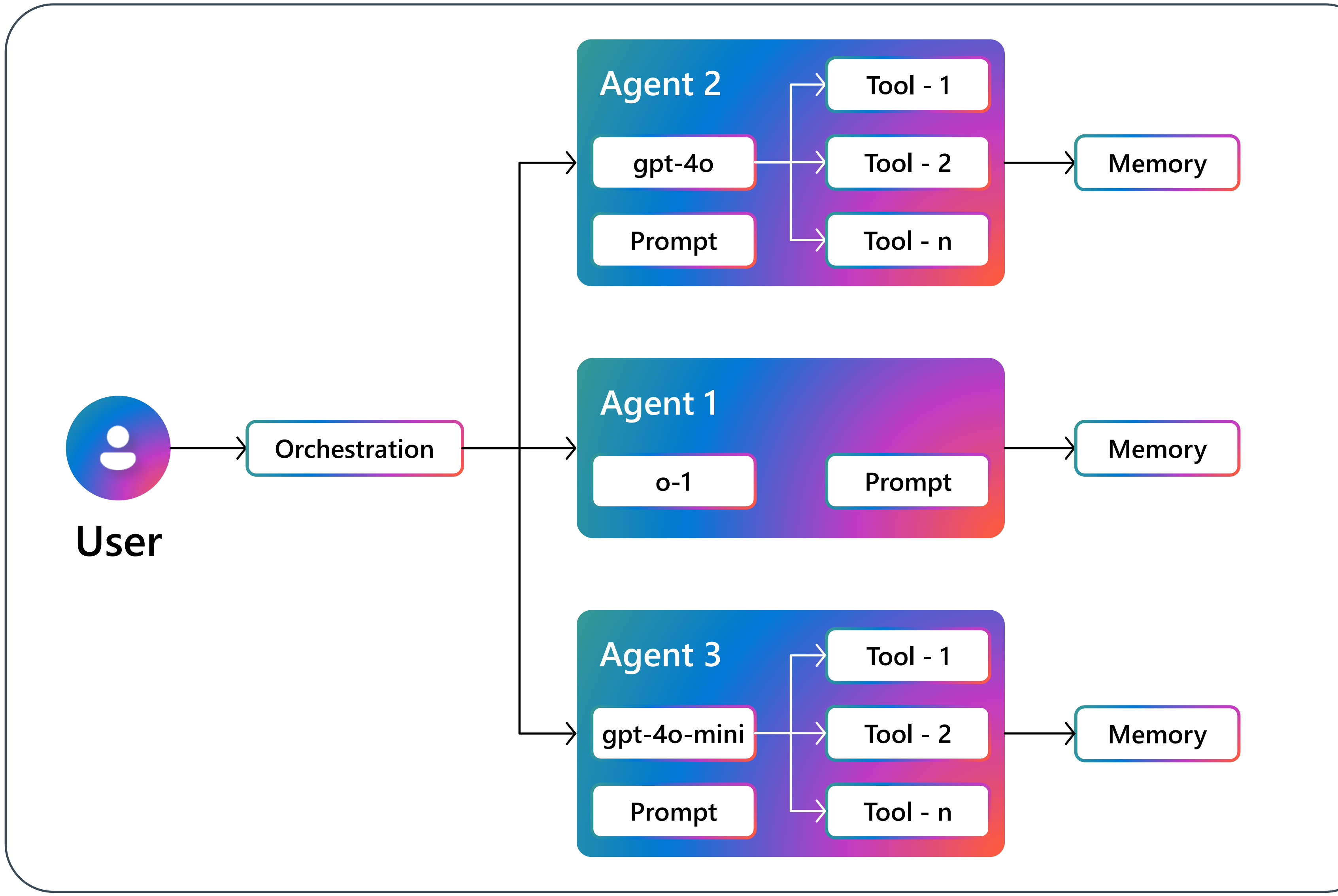
Multi Agent Logical Architecture

Each agent is specialized in different tasks or aspects of a problem

Agents can communicate and coordinate with each other. Structured orchestration is crucial

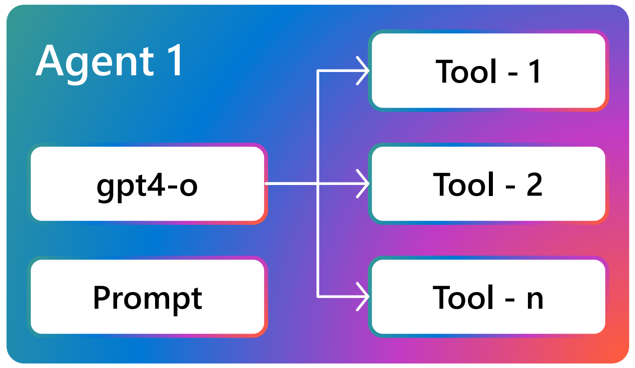
2 primary categories based on orchestration types

- Vertical Architecture
- Horizontal Architecture

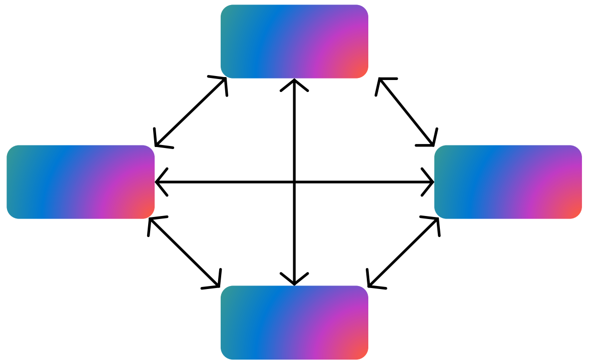


Agents orchestration and communication styles

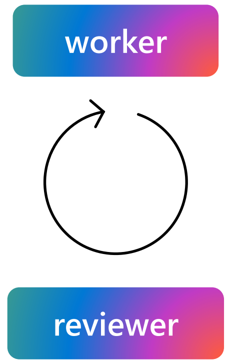
Single Agent



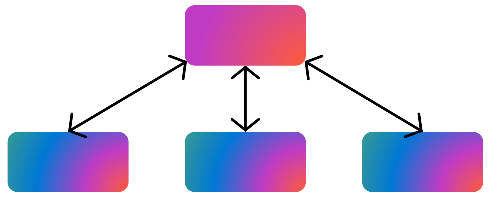
Network



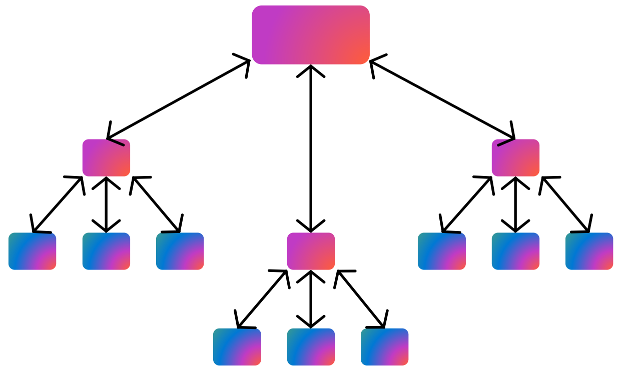
Reflection



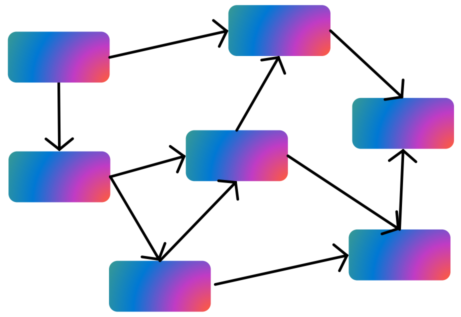
Supervisor



Hierarchical



Custom

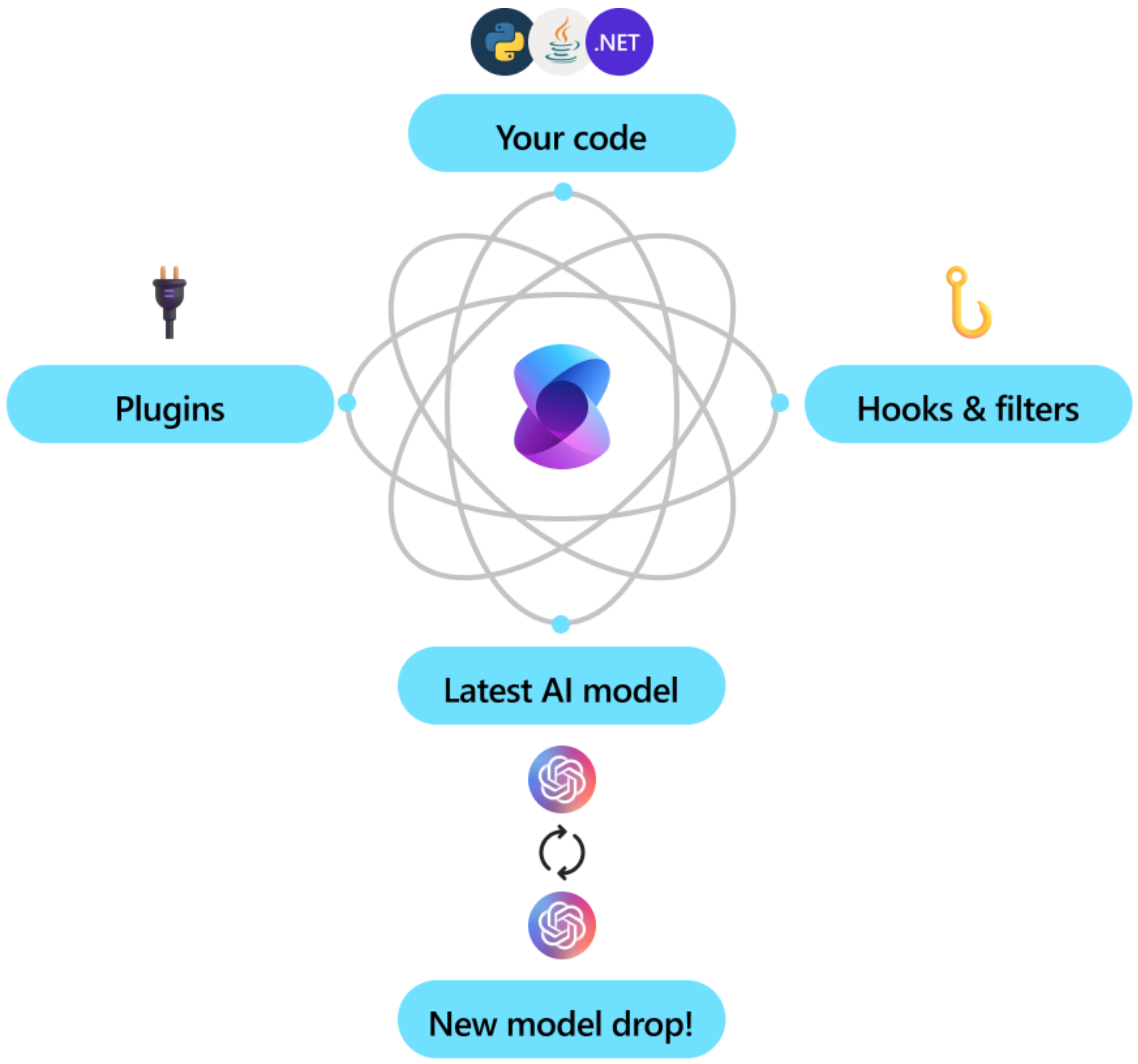


Semantic Kernel Agent Framework

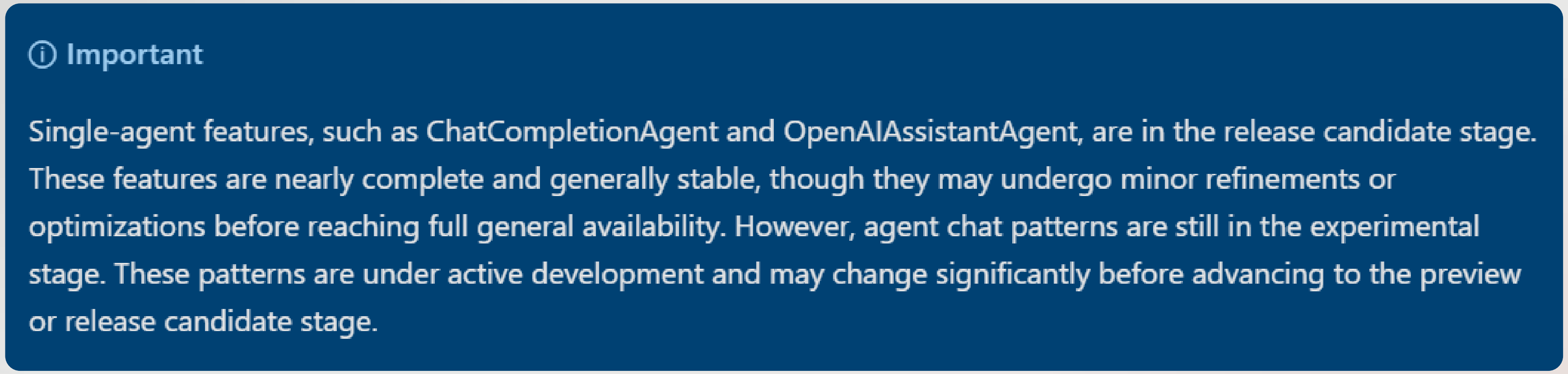
Semantic Kernel

Semantic Kernel is a lightweight, open-source development kit that lets you easily build AI agents and integrate the latest AI models into your C#, Python, or Java codebase. It serves as an efficient middleware that enables rapid delivery of enterprise-grade solutions.

Introduction to Semantic Kernel | Microsoft Learn



Note that we're using some experimental features!



ⓘ Important

Single-agent features, such as ChatCompletionAgent and OpenAIAssistantAgent, are in the release candidate stage. These features are nearly complete and generally stable, though they may undergo minor refinements or optimizations before reaching full general availability. However, agent chat patterns are still in the experimental stage. These patterns are under active development and may change significantly before advancing to the preview or release candidate stage.

Agent Framework concepts

Kernel

The Kernel serves as the **core object** that drives AI operations and interactions.
To create any agent within this framework, a Kernel instance is required as it provides the **foundational context and capabilities** for the agent's functionality.
The Kernel acts as the engine for **processing instructions**, **managing state**, and **invoking** the necessary **AI services** that power the agent's behavior.

Agent

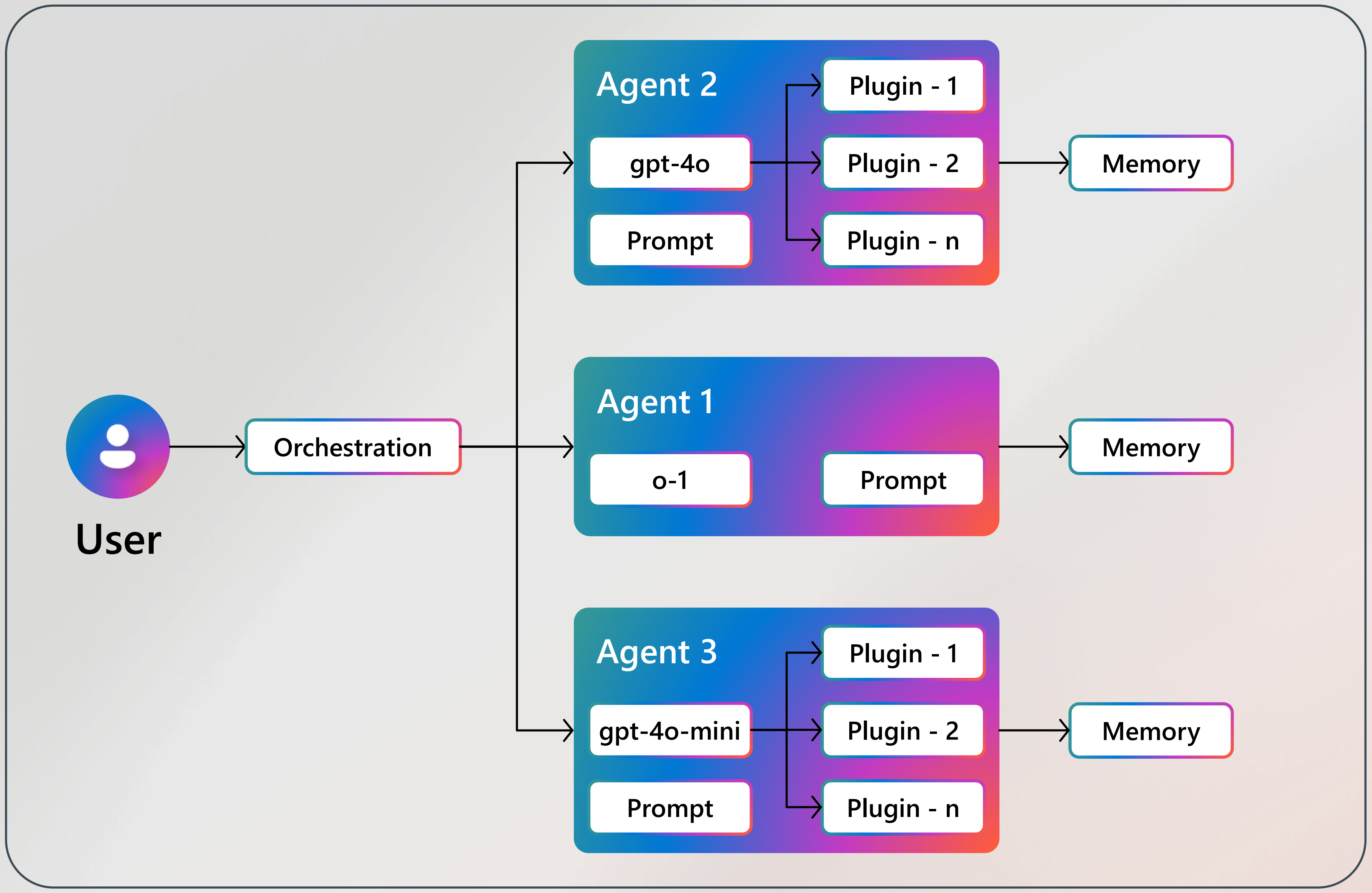
The abstract Agent class serves as the core abstraction for all types of agents, providing a **foundational structure** that can be extended to create more specialized agents.
Agents can either be **invoked directly** to perform tasks or orchestrated within an **AgentChat**, where multiple agents may collaborate or interact dynamically with user inputs.

Agent Chat

This class provides the essential capabilities for managing agent interactions within a **chat environment**.
Building on this, the **AgentGroupChat** class extends these capabilities by offering a **strategy-based** container, which allows multiple agents to collaborate across numerous interactions within the same conversation.

Semantic Kernel Agent Architecture | Microsoft Learn

AgentGroupChat



Types of Agents

Chat Completion Agent

Chat Completion is fundamentally a protocol for a chat-based interaction with an AI model where the chat-history maintained and presented to the model with each request.
A chat completion agent can leverage any of these AI services to generate responses, whether directed to a user or another agent.

- AzureChatCompletion
- OpenAIChatCompletion

OpenAI Assistant Agent

The OpenAI Assistant API is a specialized interface designed for more advanced and interactive AI capabilities, enabling developers to create personalized and multi-step task-oriented agents. Unlike the Chat Completion API, which focuses on simple conversational exchanges, the Assistant API allows for dynamic, goal-driven interactions with additional features like code-interpreter and file-search.

Azure AI Agent

An AzureAIAgent is designed to provide advanced conversational capabilities with seamless tool integration. It automates tool calling, eliminating the need for manual parsing and invocation. The agent also securely manages conversation history using threads, reducing the overhead of maintaining state. Additionally, it supports a variety of built-in tools, including file retrieval, code execution, and data interaction via Bing, Azure AI Search, Azure Functions, and OpenAPI.

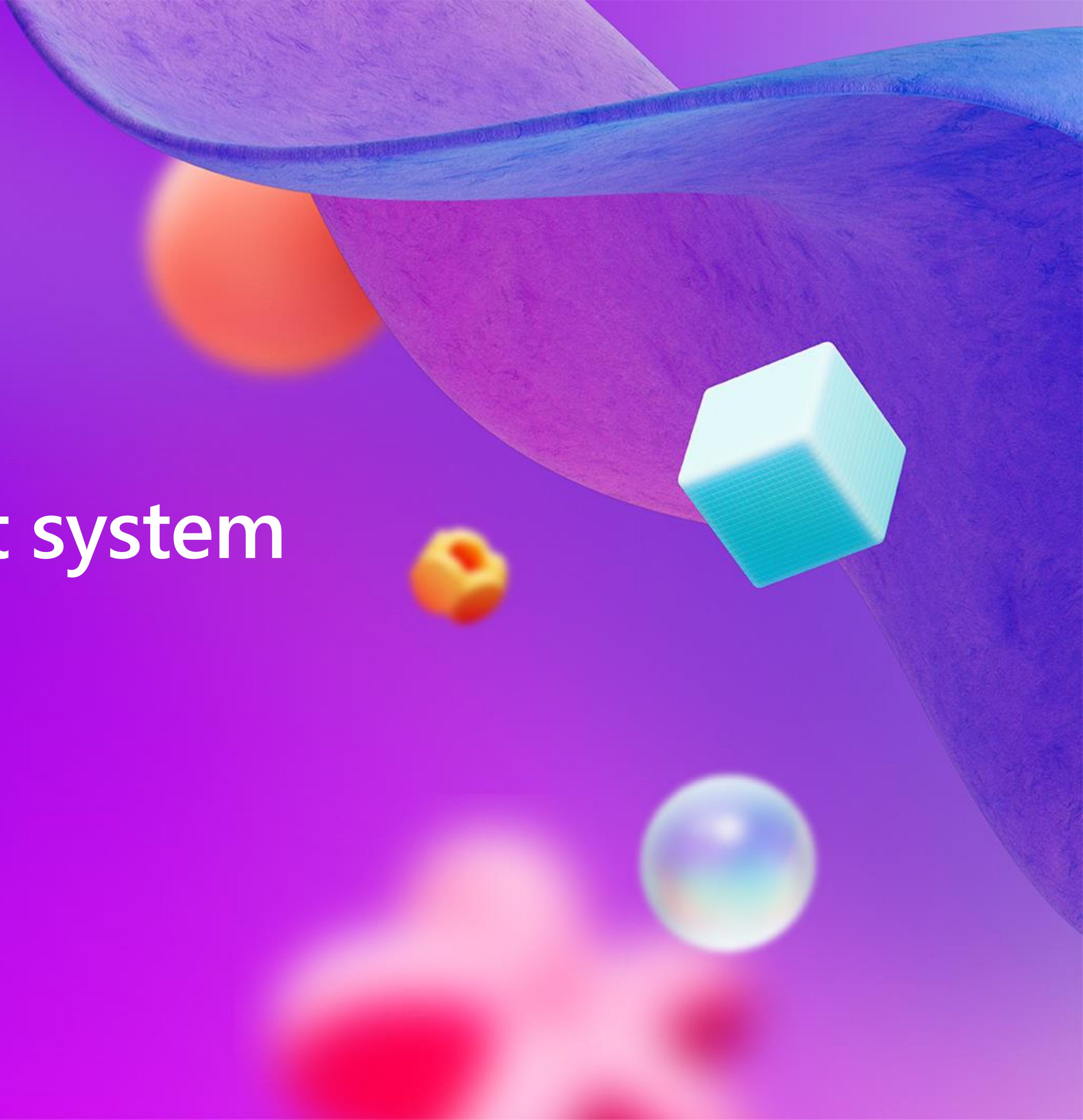
Exploring the Semantic Kernel ChatCompletionAgent | Microsoft Learn

Exploring the Semantic Kernel OpenAI Assistant Agent | Microsoft Learn

Exploring the Semantic Kernel Azure AI Agent Agent | Microsoft Learn

Building a multi-agent system

With Azure AI Agent Service and Semantic Kernel

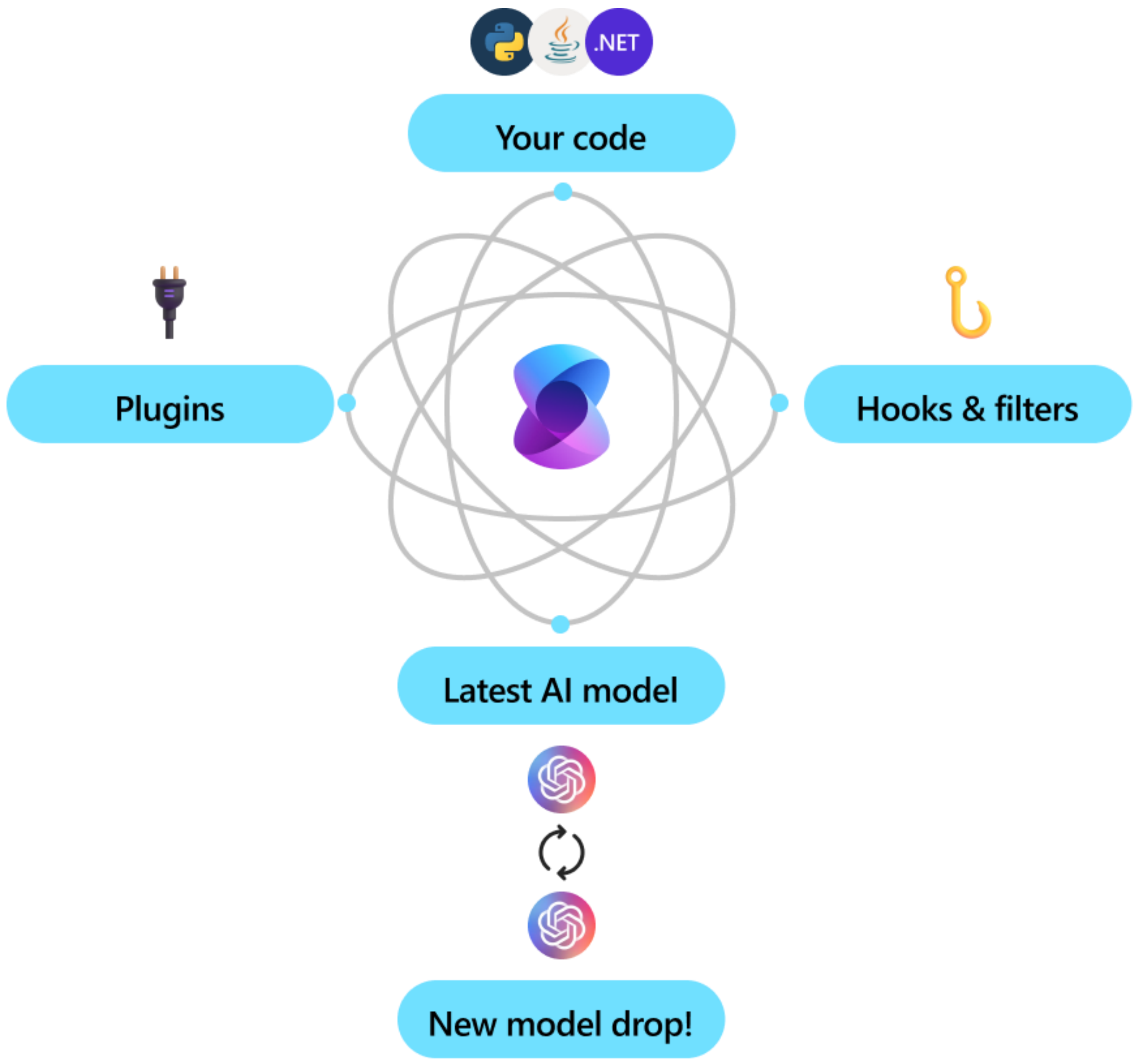


Semantic Kernel

microsoft/semantic-kernel: Integrate cutting-edge LLM technology quickly and easily into your apps

Introduction to Semantic Kernel | Microsoft Learn

How to quickly start with Semantic Kernel | Microsoft Learn



Sample code for Semantic Kernel agents

semantic-kernel/python/semantic_kernel/agents at main · microsoft/semantic-kernel

1

Single-agent

Deploy agents with **Azure AI Foundry**

Managed agent
micro-services

2

Multi-agent

Orchestrate them together with **AutoGen** and **Semantic Kernel**

State-of-the-art
research SDK

Production-ready
and stable SDK

Ideation → **Production**

Multi-Agent Implementation Key Learnings

When to use agents

Flexible user tasks requiring multiple skills and domains on the backend

Customer support scenarios, intelligent code editors, personalized digital content creation

Tradeoff:
Latency + Cost **vs** Flexibility + Quality

Do I need agent frameworks to start?

Most are experimental or have not stable apis yet. Different abstractions

Option 1: Use an agent framework. SK for production, Autogen for AI state of the art

Option 2: Use simple and composable apis like memory and tools in AI frameworks. Implement multi-agent conversation boilerplate code

Multi-Agent Implementation Key Learnings

Multi-agent autonomy and predictability friction

2 levels: Agent autonomy and multi-agent collaboration autonomy

Avoid tools hallucination: wrong tools, pre-fabricated, wrong calls sequence, wrong params

Well documented tools. Consistent api methods naming

Human-in-the-loop as your safety net

Predictable multi-agent conversations: share tools among agents, max loop iterations

Resources

- **Guy’s agent session:** aka.ms/CloudChampion/agents
- **Documentation:** Semantic Kernel Agent Framework | Microsoft Learn
- **Getting started samples:** semantic-kernel/python/samples/getting_started_with_agents/README.md at main · microsoft/semantic-kernel

Thank you!

Feel free to submit questions through the Q&A feature on this session or reach out on LinkedIn!

The recording will be made available on CloudChampion. Slides can be found at aka.ms/IntroToMultiAgents