

Agentic AI with Azure and Copilot Studio

Build a scoped prototype and review enterprise architecture

FA-0558 · 2 days

Agentic & Generative AI · Automation, RPA & Power Platform · DevOps, Cloud & Infrastructure

A two-day technical workshop covering Copilot Studio, Microsoft Foundry, enterprise retrieval, MCP integration and governed agent deployment choices.

Introduction

This technical workshop explores agentic AI in the Microsoft ecosystem using Copilot Studio, Microsoft Foundry (formerly Azure AI Foundry), Azure-hosted models and enterprise data services. Participants author a scoped agent example and examine retrieval, tool integration and governance.

Core exercises focus on a small prototype in a prepared sandbox. Multi-agent coordination, production hosting, scaling and model adaptation are architecture clinics and demonstrations, not a promise to implement a production enterprise platform in two days. Available integrations, model features and preview capabilities depend on the selected environment.

Learning outcomes

The course teaches participants to:

- Explain agent design choices across Copilot Studio and Microsoft Foundry.
- Author, test and refine a scoped agent with fallback behaviour.
- Describe or demonstrate an authorised data/tool connection, including MCP and Dataverse where configured.
- Explain RAG components, vector retrieval, metadata filtering and grounding limitations.
- Review identity, access, runtime safeguards, monitoring and deployment trade-offs.

- Use evaluation and feedback to prioritise improvements without assuming feedback automatically retrains a model.

Prerequisites

- Strong Python programming skills
- Working understanding of cloud services (especially Azure)
- Familiarity with REST APIs, JSON, data modeling, and vector databases
- Some prior experience with LLMs, embeddings, prompt engineering, or retrieval systems is helpful (though not strictly required)
- Basic knowledge of DevOps/containerization (Docker, Kubernetes)

A prepared laptop and authorised Azure/Power Platform sandbox with the required licences, resources, role assignments and model access. Confirm service availability, preview restrictions and lab costs in advance; production data is not required.

Training outline

1. Day 1 — Authoring, enterprise data and retrieval

1. Agent architecture in Microsoft Azure

- What is a copilot vs. an agent? (Microsoft framing)
- The role of Azure OpenAI in the Copilot ecosystem
- Microsoft Foundry and Copilot Studio: distinct authoring surfaces and managed agent capabilities.
- Modular agent design patterns for selected use cases, including safety and maintainability.
- Governance considerations: content filtering, identity, access policies and audit trails; controls do not by themselves establish compliance.

2. Copilot Studio authoring

- Overview of Copilot Studio: interface, capabilities, and architecture
- Generative answers and supported knowledge sources: grounding choices, permissions and limitations.
- Low / no-code vs code-enabled agent design in Copilot Studio
- Use the selected Copilot Studio testing and debugging tools; review behaviour and trace failures.
- Best practices — versioning, iteration, conversation flows, fallback logic

3. MCP, Dataverse and authorised data access

- Recap: what is MCP (Model Context Protocol) and why it matters
- Dataverse MCP server: authorised environment/client configuration and selected tools, subject to access and billing arrangements.
- Compare supported generative-answer knowledge connections with an Azure AI Search retrieval architecture; do not assume all integrations are automatic.
- Supported connectors, MCP tools and custom API adapters; choose an appropriate integration route.
- Data access control, latency and caching choices; MCP does not bypass underlying permissions.

4. Retrieval-augmented generation

- Key concepts: embeddings, vector search, metadata filtering, context windows
- Design a vector/semantic retrieval index using Azure AI Search and inspect retrieval behaviour.
- Agent-driven retrieval vs static context injection
- Strategies for prompt templating, chunking, and relevance feedback
- Agent orchestration between retrieval and generation

2. Day 2 — Orchestration, safeguards and operation

5. Multi-agent architecture clinic

- Patterns for multi-agent systems: task decomposition, delegation, agent collaboration
- Coordination and state patterns using supported orchestration, events or message buses; MCP exposes tools/resources, not automatic shared state between agents.
- Error recovery, fallback agents, chaining agents
- Illustrative architecture review: agents supporting a multi-step business workflow.

6. Runtime safeguards

- When agents generate code or actions — sandboxing, validation, whitelists
- Approaches to limit harmful or unexpected behavior
- Logging, auditing, alerts, and human-in-the-loop oversight
- Failover plans, circuit breakers, rate limiting, and safety wrappers

7. Deployment and observability clinic

- Architecture decisions: serverless vs container vs VM

- Managing agent runtime via Azure resource governance (VMs, AKS, App Services, Functions)
- Microsoft Entra ID, managed identities, networking and access control.
- Monitoring, observability, logging, and tracing agents
- Auto-scaling strategies, cost management

8. Evaluation and improvement

- Collect evaluation evidence, logs and feedback with appropriate consent and data handling.
- Compare prompt/retrieval changes with optional model adaptation; LoRA/PEFT applicability depends on the selected model, service and deployment route.
- A/B testing, performance metrics, iteration cycles
- Governance around model drift, quality checks, rollback