

Lab 076: Microsoft Agent Framework ? From SK to MAF

Level: L200 Path: All paths Time: ~75 min ? Cost: Free ? Uses mock migration data

What You'll Learn

How Semantic Kernel and AutoGen are unified into the Microsoft Agent Framework (MAF) What Agent Skills (portable .md skill packages) are and how they enable reuse How Graph Workflows (DAG with checkpointing) replace linear pipelines Analyze a migration matrix comparing 15 features across SK, AutoGen, and MAF Identify migration effort levels and MAF-exclusive capabilities

Introduction

The Microsoft Agent Framework (MAF) unifies Semantic Kernel and AutoGen into a single, cohesive platform for building AI agents. Released as a Release Candidate in February 2026, MAF brings together the best of both worlds:

Semantic Kernel's plugin system, planners, and enterprise connectors AutoGen's multi-agent conversations, code execution, and group chat patterns

Installation

[diagram or code sample]

Key Concepts

Concept | Description | Agent Skills | Portable .md skill packages that define agent capabilities, inputs, outputs, and dependencies ? shareable across teams and projects | Graph Workflows | DAG-based orchestration with checkpointing, retry, and branching ? replacing linear pipelines | DevUI | Built-in development UI for debugging agent conversations, inspecting skill execution, and visualizing workflows | Unified API | Single API surface that replaces SK's Kernel and AutoGen's AssistantAgent with a common Agent class |

The Scenario

You are a Platform Engineer at a company that has built agents using both Semantic Kernel and AutoGen. Leadership has decided to migrate to MAF. You have a migration matrix (migration_matrix.csv) that maps 15 features across the three frameworks ? tracking availability, migration effort, and MAF-exclusive features.

Your job: analyze the matrix, identify quick wins, flag challenges, and build a migration plan.

info "Mock Data" This lab uses a mock migration matrix CSV. The data reflects the actual feature mapping between Semantic Kernel, AutoGen, and MAF as documented in the migration guides.

Prerequisites

Requirement | Why | Python 3.10+ | Run the analysis scripts | pandas library | Data manipulation |

[diagram or code sample]

tip "Quick Start with GitHub Codespaces" Open in GitHub Codespaces

All dependencies are pre-installed in the devcontainer.

? Supporting Files

note "Download these files before starting the lab" Save all files to a lab-076/ folder in your working directory.

File | Description | Download | broken_migration.py | Bug-fix exercise (3 bugs + self-tests) | ? Download | migration_matrix.csv | 15-feature comparison: SK vs AutoGen vs MAF | ? Download |

Step 1: Understand the Migration Matrix

The migration matrix maps 15 features across three frameworks. Each row represents a

capability:

Column | Description | feature | The capability being compared (e.g., plugins, multi_agent_chat) | sk_support | Whether Semantic Kernel supports this feature: yes, partial, or no | autogen_support | Whether AutoGen supports this feature: yes, partial, or no | maf_support | Whether MAF supports this feature: yes, partial, or no | migration_effort | Effort to migrate from SK/AutoGen to MAF: low, medium, or high | category | Feature category: core, orchestration, tooling, or integration |

Step 2: Load and Explore the Matrix

[diagram or code sample]

Expected output:

[diagram or code sample]

Step 3: Identify Quick Wins (Low Migration Effort)

Features with low migration effort are your quick wins ? start here:

[diagram or code sample]

tip "Migration Strategy" Start with low-effort features to build team confidence and demonstrate MAF's unified API. These typically have direct equivalents in SK or AutoGen, making the migration straightforward.

Step 4: Find MAF-Exclusive Features

Which features exist in MAF but not in Semantic Kernel?

[diagram or code sample]

[diagram or code sample]

Step 5: Analyze High-Effort Migrations

These features need the most planning:

[diagram or code sample]

warning "Migration Risk" High-effort features often involve architectural changes ? e.g., replacing custom orchestration with Graph Workflows or converting proprietary plugins to Agent Skills. Plan 2-4 weeks per high-effort feature.

Step 6: Build the Migration Plan

[diagram or code sample]

? Bug-Fix Exercise

The file lab-076/broken_migration.py contains 3 bugs that produce incorrect migration analysis. Can you find and fix them all?

Run the self-tests to see which ones fail:

[diagram or code sample]

You should see 3 failed tests. Each test corresponds to one bug:

Test | What it checks | Hint | Test 1 | Low-effort feature count | Should filter migration_effort == "low", not "high" | Test 2 | MAF-only vs SK count | Should check sk_support == "no", not "yes" | Test 3 | Total feature count | Should use len(df), not len(df.columns) |

Fix all 3 bugs, then re-run. When you see All passed!, you're done!

? Knowledge Check

question "Q1 (Multiple Choice): What is the Microsoft Agent Framework (MAF)?"

- A) A new version of AutoGen with a different name - B) A unified framework combining Semantic Kernel and AutoGen into a single platform - C) A cloud-only service for running agents on Azure - D) A replacement for LangChain

??? success "? Reveal Answer" Correct: B) A unified framework combining Semantic Kernel and AutoGen into a single platform

MAF merges the strengths of both frameworks: SK's enterprise plugin system and planners

with AutoGen's multi-agent conversations and code execution. It provides a single Agent class, portable Agent Skills, and DAG-based Graph Workflows.

question "Q2 (Multiple Choice): What are Agent Skills in MAF?"

- A) Python functions decorated with @skill - B) Portable .md skill packages that define capabilities, inputs, outputs, and dependencies - C) Pre-trained model weights for specific tasks - D) Azure Functions that agents can call

??? success "?" Reveal Answer" Correct: B) Portable .md skill packages that define capabilities, inputs, outputs, and dependencies

Agent Skills are markdown-based packages that describe what an agent can do, what inputs it needs, what outputs it produces, and what dependencies are required. They're shareable across teams and projects, enabling a marketplace of reusable agent capabilities.

question "Q3 (Run the Lab): How many features have a 'low' migration effort?"

Run the Step 3 analysis on ? migration_matrix.csv and count the low-effort features.

??? success "?" Reveal Answer" 7 features

Of the 15 features in the migration matrix, 7 have migration_effort = "low". These are the quick wins for Phase 1 of the migration ? typically features with direct equivalents between SK/AutoGen and MAF.

question "Q4 (Run the Lab): How many features are available in MAF but NOT in Semantic Kernel?"

Run the Step 4 analysis to find features where maf_support = "yes" and sk_support = "no".

??? success "?" Reveal Answer" The count of features where MAF has yes support but SK has no support.

These represent capabilities that teams gain by migrating from SK to MAF ? such as multi-agent chat patterns, code execution sandboxes, and other features that originated in AutoGen and are now available in the unified framework.

question "Q5 (Run the Lab): How many total features are tracked in the migration matrix?"

Load the CSV and check the total row count.

??? success "?" Reveal Answer" 15 features

The migration matrix tracks 15 features across 4 categories: core (5), orchestration (4), tooling (3), and integration (3). Each feature is evaluated for support in SK, AutoGen, and MAF, along with migration effort.

Summary

Topic | What You Learned | Microsoft Agent Framework | Unified platform merging Semantic Kernel and AutoGen (RC Feb 2026) | Agent Skills | Portable .md skill packages for reusable agent capabilities | Graph Workflows | DAG-based orchestration with checkpointing, replacing linear pipelines | Migration Matrix | 15 features compared across SK, AutoGen, and MAF | Migration Strategy | Start with low-effort (7 features), plan for high-effort (3 features) | DevUI | Built-in development UI for debugging and visualizing agent workflows |

Next Steps

Lab 034 ? Multi-Agent with Semantic Kernel (understand what you're migrating from) Lab 036

? AutoGen Basics (understand AutoGen patterns before MAF) Lab 074 ? Foundry Agent Service (operate and monitor production agent fleets) Lab 073 ? SWE-Bench (evaluate MAF agents on real-world coding tasks)

Phase 3 modernization

warning "Verified 2026-10-27" Use Microsoft Agent Framework 1.0 for new agent work.

Semantic Kernel remains a legacy migration source; AutoGen is maintenance-oriented. Remove obsolete planner patterns and use explicit workflows, tools, memory, checkpoints, and stop conditions.

[diagram or code sample]

Microsoft Agent Framework 1.0 GA AutoGen to Agent Framework migration

Phase 3 · Simulation / Real path

Simulation Real path

Preserve the existing free simulation and its fixtures. Estimated cost: US\$0.

Reuse the same acceptance criteria against the documented hosted service. Before running, record a service-specific estimate and a hard spending cap. After validation: Delete hosted deployments, endpoints, indexes, traces, and test data after validation.

Supporting kit

The site build generates stable PDF and ZIP downloads. The ZIP contains the lab guide and every supporting file declared for this lab.

[Download generated PDF](#) [Download generated ZIP](#)

Supporting file: broken_migration.py

```
/usr/bin/env python3 """Bug-fix: Broken Migration Analyzer - Lab 076.""" import io, pandas
as pd def count_low_effort(matrix): return (matrix["migration_effort"] == "high").sum() def
features_in_maf_only(matrix): return len(matrix[matrix["semantic_kernel"] == "not
supported"]) def migration_coverage(matrix): return matrix["agent_framework"].nunique() /
len(matrix) * 100 def run_tests():
csv="feature,semantic_kernel,agent_framework,migration_effort\nPlugins,KernelPlugin,AgentSki
lls,medium\nA2A,not supported,native
A2A,n/a\nStreaming,IAsyncEnumerable,built-in,low\nDevUI,not available,DevUI,low"
m=pd.read_csv(io.StringIO(csv));p,f=0,0 n=count_low_effort(m);exp=2 if n==exp:print(f"OK
T1: {n}");p+=1 else:print(f"FAIL T1: {n} (exp {exp})");f+=1 a=features_in_maf_only(m);exp=1
if a==exp:print(f"OK T2: {a}");p+=1 else:print(f"FAIL T2: {a} (exp {exp})");f+=1
c=migration_coverage(m);exp=100.0 if abs(c-exp)<0.1:print(f"OK T3: {c:.0f}%");p+=1
else:print(f"FAIL T3: {c:.0f}% (exp {exp:.0f}%");f+=1 print(f"\n{'All passed!' if f==0
else f'{f} failed'}") if __name__=="__main__":print("Tests:\n");run_tests()
```

Supporting file: migration_matrix.csv

feature,semantic_kernel,autogen,agent_framework,migration_effort Agent
Abstraction,ChatCompletionAgent,AssistantAgent,Agent (unified),low Plugin
System,KernelPlugin,Tools,AgentSkills (.md files),medium
Multi-Agent,AgentGroupChat,GroupChat / Swarm,Graph Workflows,medium Orchestration,manual
loop,round_robin / selector,DAG with data flows,high Streaming,IAsyncEnumerable /
AsyncGenerator,streaming parameter,built-in with checkpointing,low
Memory,SemanticTextMemory,ConversableAgent memory,pluggable memory providers,medium
Human-in-the-Loop,manual implementation,human_input_mode,built-in graph nodes,low
Observability,OpenTelemetry manual,logging,DevUI + OpenTelemetry built-in,low MCP
Integration,via plugin,via tools,native MCP support,low A2A Integration,not supported,not
supported,native A2A support,n/a AG-UI Integration,not supported,not supported,native AG-UI
support,n/a Model Support,OpenAI/Azure/Ollama,OpenAI/Azure,OpenAI/Azure/Anthropic/GitHub/Loc
al,low Language Support,.NET and Python,Python (.NET experimental),.NET and Python (stable
both),low State Management,manual,manual,checkpointing + time-travel,medium DevUI,not
available,AutoGen Studio,DevUI (built-in),low