Microsoft

Building Intelligent Apps with Sematic Kernel

Mark Harrison – Microsoft, Developer Specialist

Microsoft Azure

mark.harrison@microsoft.com

This deck is “work in progress” / not final.

Hello

MARK HARRISON

Developer Specialist

mark.harrison@microsoft.com

App Innovation

Improve Outcomes - Differentiated value

App Modernization & Migration

Cloud Native

Developer Velocity

Integration

Citizen Dev Low Code

Intelligent Apps

Accelerate software delivery | Do more with less | Increase agility and resilience | Reduce technical debt

Industrial Revolutions

The major industrial revolutions are transformative periods of technological, economic, and societal change that have shaped modern civilization.

We are now entering the next great industrial revolution — where technology goes beyond automation to systems that think, learn, and adapt, driving smarter decisions and new levels of innovation.

The only thing constant is change

The key to surviving any technology revolution is leading it.

A resilient organisation will:

- have an awareness of disruptive technology
- develop talent that can make the most of it.

It is not the strongest of the species that survives,
nor the most intelligent that survives.
It is the one that is most adaptable to change.
— Charles Darwin

Artificial Intelligence

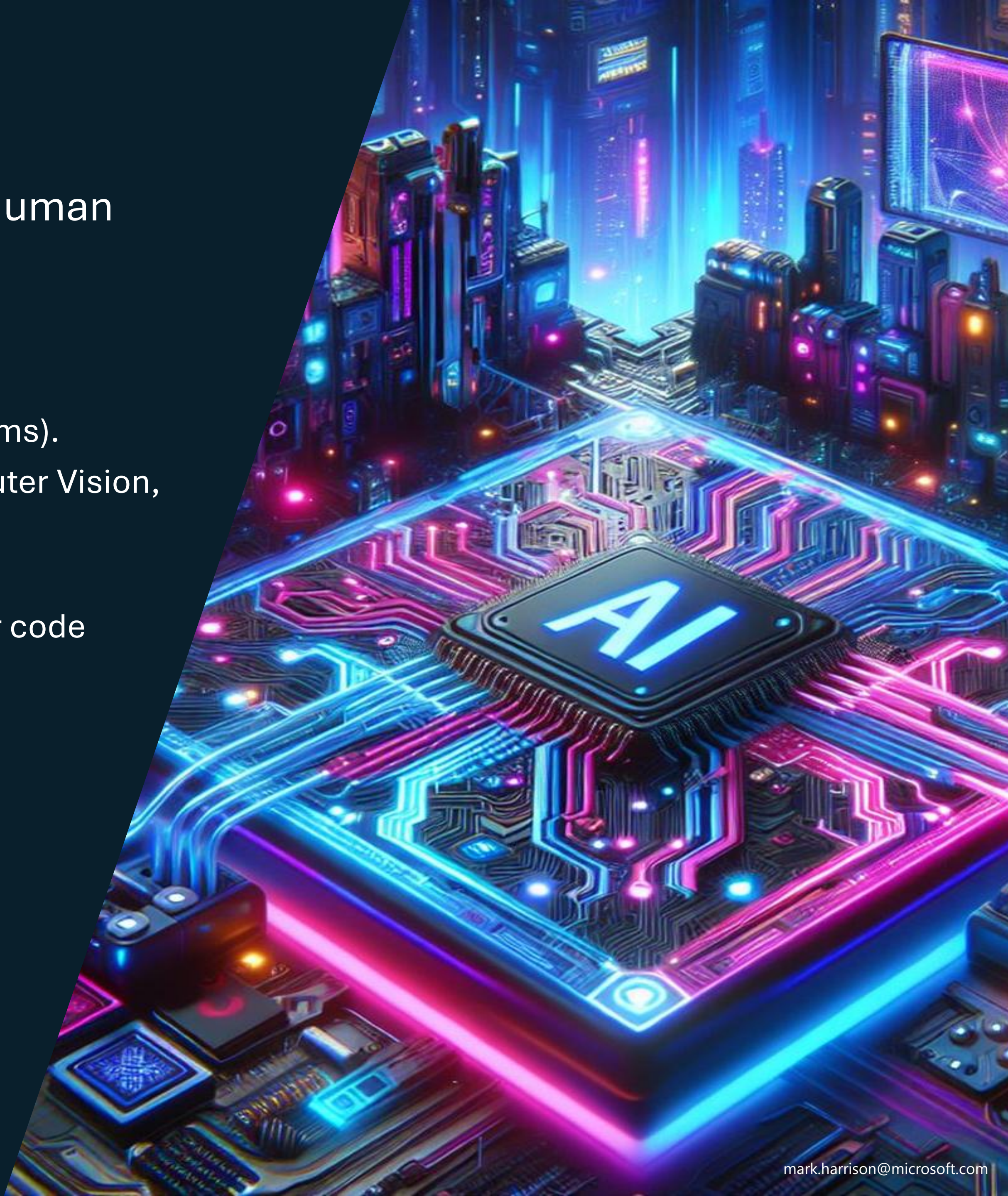
Artificial Intelligence refers to systems that simulate human intelligence to perform tasks.

Core Capabilities of AI

- Learning: Adapts based on data (Machine Learning).
- Reasoning: Makes decisions or solves problems (Decision Systems).
- Perception: Understands visual, auditory, or textual data (Computer Vision, Natural Language Processing).
- Action: Performs autonomous tasks (Robotics, Automation).
- Generation: Creates new content such as text, images, music, or code (Generative AI).

Multimodal models—capable of processing text, images, and audio simultaneously

AI Agents and Agentic AI are terms often used in discussions about artificial intelligence, but they refer to different concepts.



AI Agents

“Digital Workers” that eliminate manual tasks, enhance operational efficiency, and enable intelligent decision-making.

Limited Autonomy: Operate without human intervention – but do not adapt or learn.

Perception: Input data-driven.

Decision-making: Apply rules or machine learning models.

Action: Executes actions to achieve its goals / produce output data

Example:

A customer service chatbot to handle routine queries and reduce human workload.

Agentic AI

Agentic AI refers to systems that act independently and exhibit autonomy in their decision-making to achieve their goals.

Autonomy: Can operate and make decisions without direct human control.

Adaptive: Can learn from experiences and adapt their behaviour to improve over time.

Proactive: Takes initiative rather than just reacting to inputs / pre-defined rules.

Examples:

Advanced AI in scientific research that learns from vast datasets, formulates hypotheses on its own, and adapts by running experiments to refine its understanding.

Multi-Agent Systems

Multi-agent systems consist of multiple AI agents working together—often collaboratively—to solve complex problems.

This modular approach enables each agent to specialize in a specific task, enhancing efficiency and scalability, much like microservice architectures.

However, key challenges include coordination, communication, and distributed problem-solving.

Example: Smart Building

AI Agent:

Responsible for a specific task, e.g. smart temperature control.

Multi-Agent System:

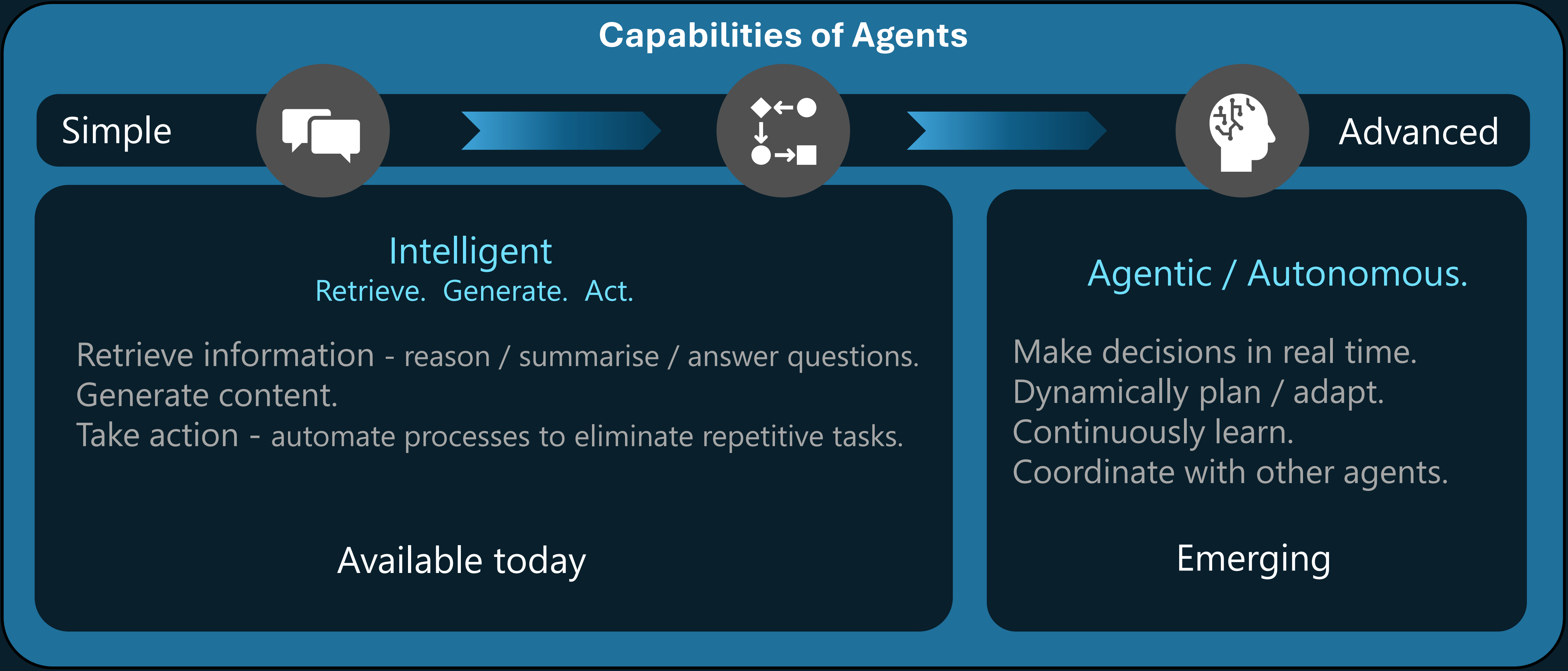
Multiple agents - such as heating, lighting, security, air quality, lift maintenance, water management, occupancy detection, weather - working together to optimize building performance.

Agentic AI:

Doesn’t just follow preset rules; it continuously learns from the building’s patterns and adapts, making changes to optimize things like energy usage, air quality, carbon footprint, comfort in real-time.

Over time, it gets better at predicting when and where adjustments need to be made, even before a problem happens.

Are we there yet ...



Capabilities of Agents

Simple → Advanced

Intelligent

Retrieve. Generate. Act.

Retrieve information - reason / summarise / answer questions.
Generate content.
Take action - automate processes to eliminate repetitive tasks.

Available today

Agentic / Autonomous.

Make decisions in real time.
Dynamically plan / adapt.
Continuously learn.
Coordinate with other agents.

Emerging

AI & Developers

Building software using AI

To reduce think time / reduce keystrokes / help as a learning assistant.

GitHub Copilot

https://github.com/features/copilot

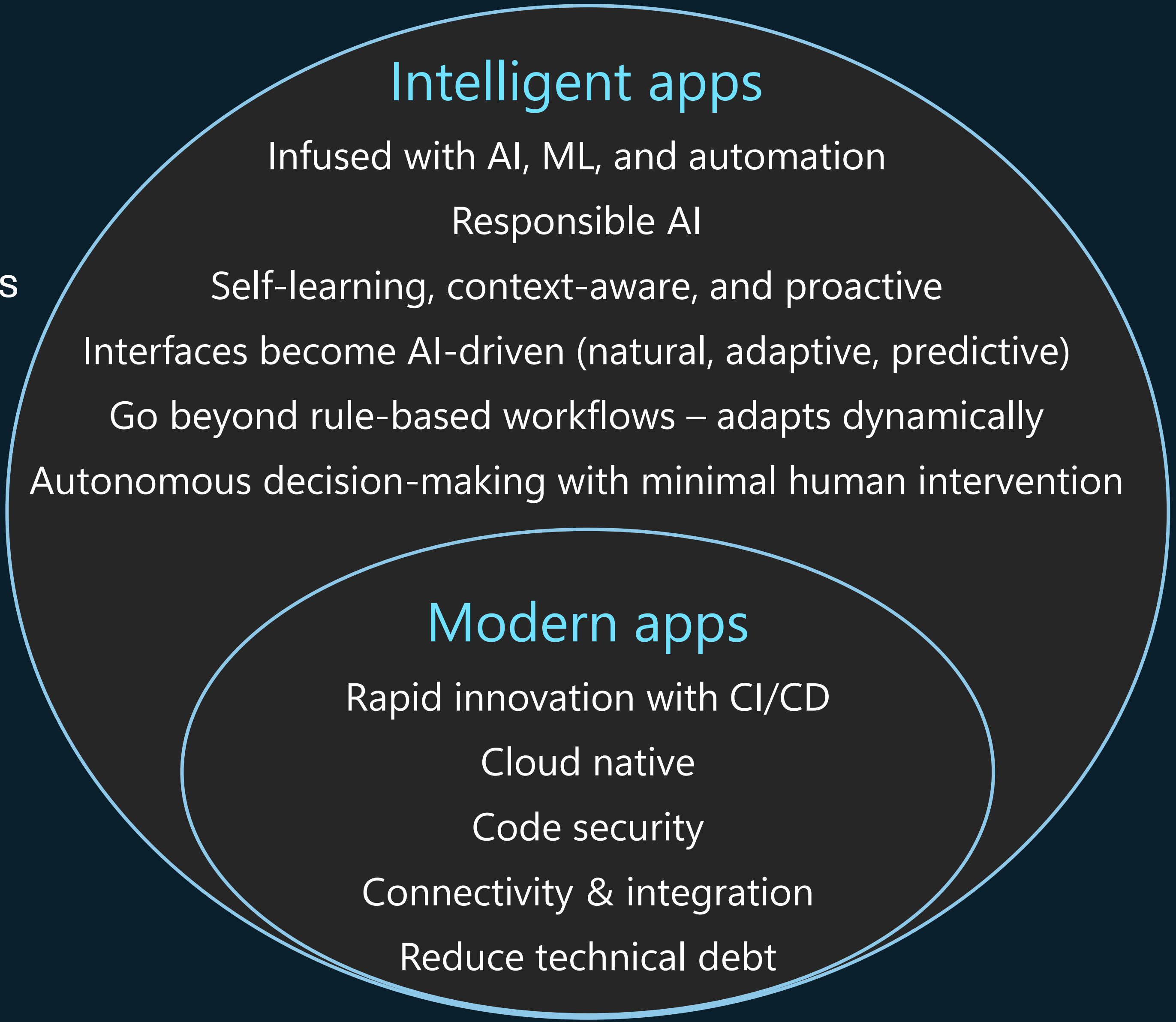
Building AI into software

To enhance the capabilities, improve automation, and enable smarter decision-making.

Intelligence is the new baseline for modern apps

Build AI into software to enhance the capabilities, improve automation, and enable smarter decision-making.

Developers are key to implement the business logic that deliver truly agentic applications.



Intelligent apps

Infused with AI, ML, and automation

Responsible AI

Self-learning, context-aware, and proactive

Interfaces become AI-driven (natural, adaptive, predictive)

Go beyond rule-based workflows – adapts dynamically

Autonomous decision-making with minimal human intervention

Modern apps

Rapid innovation with CI/CD

Cloud native

Code security

Connectivity & integration

Reduce technical debt

Intelligent Application

Responsible AI

Fairness

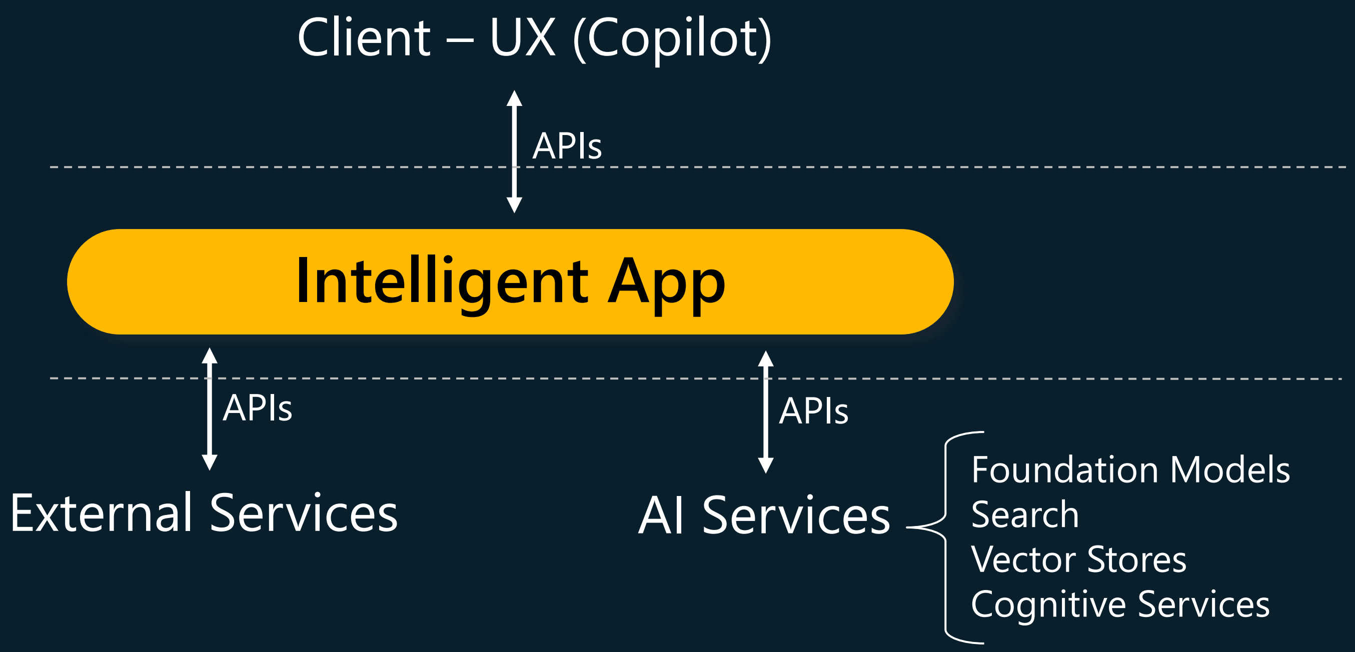
Reliability and Safety

Privacy and Security

Inclusiveness

Transparency

Accountability



Client – UX (Copilot)

APIs

Intelligent App

APIs

APIs

External Services

AI Services

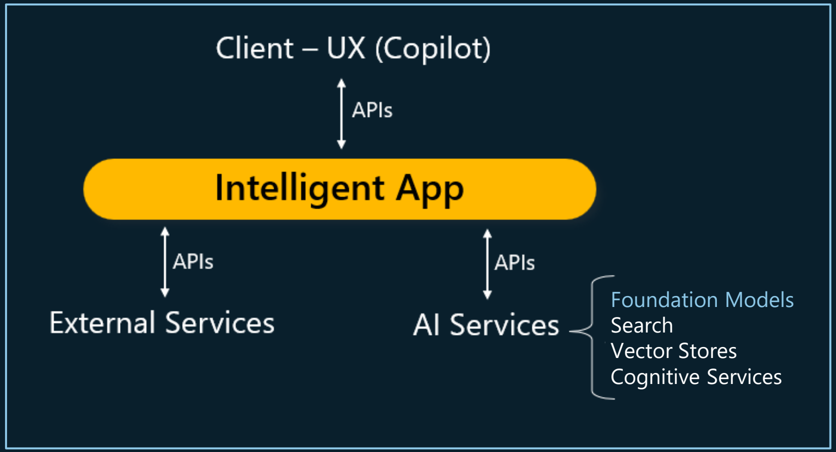
Foundation Models

Search

Vector Stores

Cognitive Services

Foundation Models



Client – UX (Copilot)

APIs

Intelligent App

APIs

APIs

External Services

AI Services

Foundation Models
Search
Vector Stores
Cognitive Services

A large-scale AI system trained on diverse data and can perform various tasks often with minimal additional training.

- Training: Learns from large datasets by adjusting internal parameters to minimize errors.
- Inference: Uses trained knowledge to make predictions or decisions on new data.

Example: Text

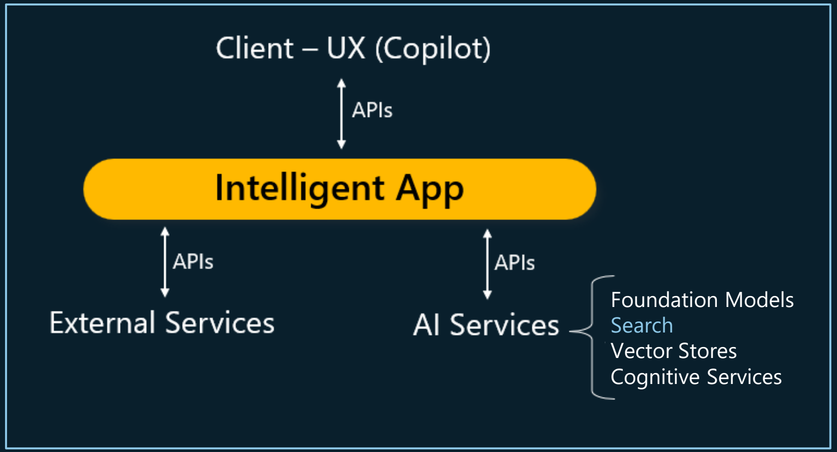
- Trained on massive text data to understand and generate human language.
- Recognizes patterns in grammar, context, and word associations for meaningful responses.
  - Large Language Models (LLMs): Require high computational power, typically run in the cloud.
  - Smaller Models: Optimized versions for local devices with lower computational needs.

Examples:

- Azure Open AI – GPT, text-ada-002, DALL-E
- Open AI – GPT, text-ada-002, DALL-E
- Anthropic – Claude
- Google – Gemini, BERT
- Meta - Llama
- Mistral

Search

Keyword Search v Semantic Search



Keyword search is a traditional method where a system looks for exact words or phrases in documents, databases, or other text sources.

Semantic Search is a broad concept. It refers to any search approach that retrieves results based on the meaning (semantics) of the query, rather than exact keyword matches.

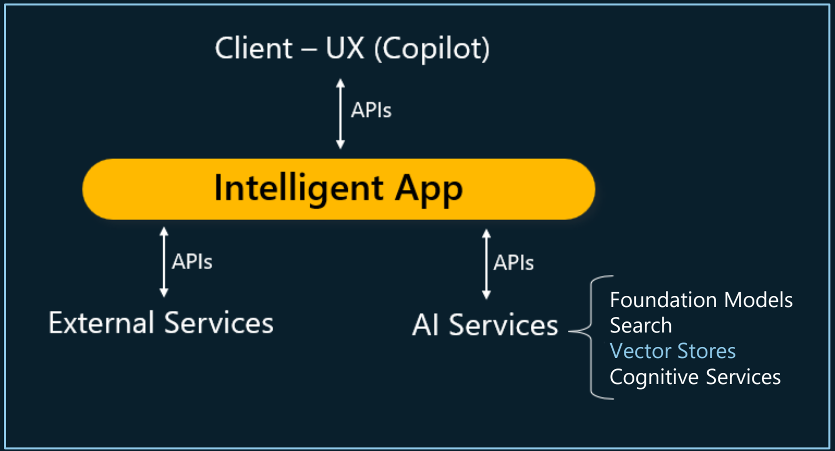
Example: Query: "How does global warming impact nature?"

Keyword Search: Matches documents with "global warming," "impact," and "nature.”

Embedding Search: Matches content about "climate change effects on biodiversity," "ecosystem damage," or "rising temperatures harming wildlife," even if "global warming" isn’t explicitly mentioned.

Embeddings are a specific technique used to implement semantic search.

Embeddings



Client – UX (Copilot)

APIs

Intelligent App

APIs

APIs

External Services

AI Services

Foundation Models
Search
Vector Stores
Cognitive Services

Embeddings encode complex data in a high-dimensional space, where similarity is measured by distance apart (using cosine similarity).

Example: Text … using models such as text-ada-002, BERT, others

Encoded into numerical vectors that capture semantic meaning.

"How do I fix my phone screen?" -> [0.8, 0.1, 0.2, 0.7, …]
"What is the best way to repair a broken phone screen?" -> [0.8, 0.1, 0.3, 0.6 , …]
Close
"What is the weather like today?" -> [0.2, 0.8, 0.7, 0.3, …]

Example: Medical images … using model MedImageInsight (MI2)

Certain conditions in an X-ray image would cluster together

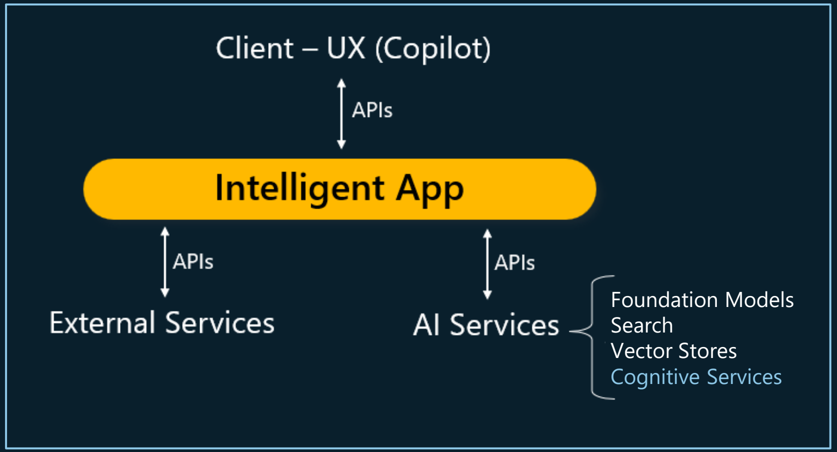
Vector Store is the underlying technology or database where embeddings / representations of data are physically stored and managed.

Examples:

- Azure AI Search
- Pinecone
- Weaviate
- Azure CosmosDb
- Redis
- MongoDB
- SQLServer
- Qdrant
- Elasticsearch

mark.harrison@microsoft.com

Cognitive Services



Cognitive Services are cloud-based APIs that allow developers to integrate artificial intelligence (AI) capabilities into applications without having to be experts in machine learning or AI.

Azure Cognitive Services :

AI Vision: Analyses images and videos for objects, faces, and text.

AI Speech: Converts speech to text, text to speech, and translates audio.

AI Language: Understands text, sentiment, and key phrases.

AI Translator: Provides real-time text and speech translation.

AI Search: Enhances search with AI-powered indexing and relevance.

AI Content Safety: Detects and moderates harmful or inappropriate content.

AI Document Intelligence: Extracts data from documents such as invoices, receipts, and forms.

AI Content Understanding: Derive meaningful insights from diverse data, ranging from text, audio, images, and video.

AI Application

Responsible AI

Fairness

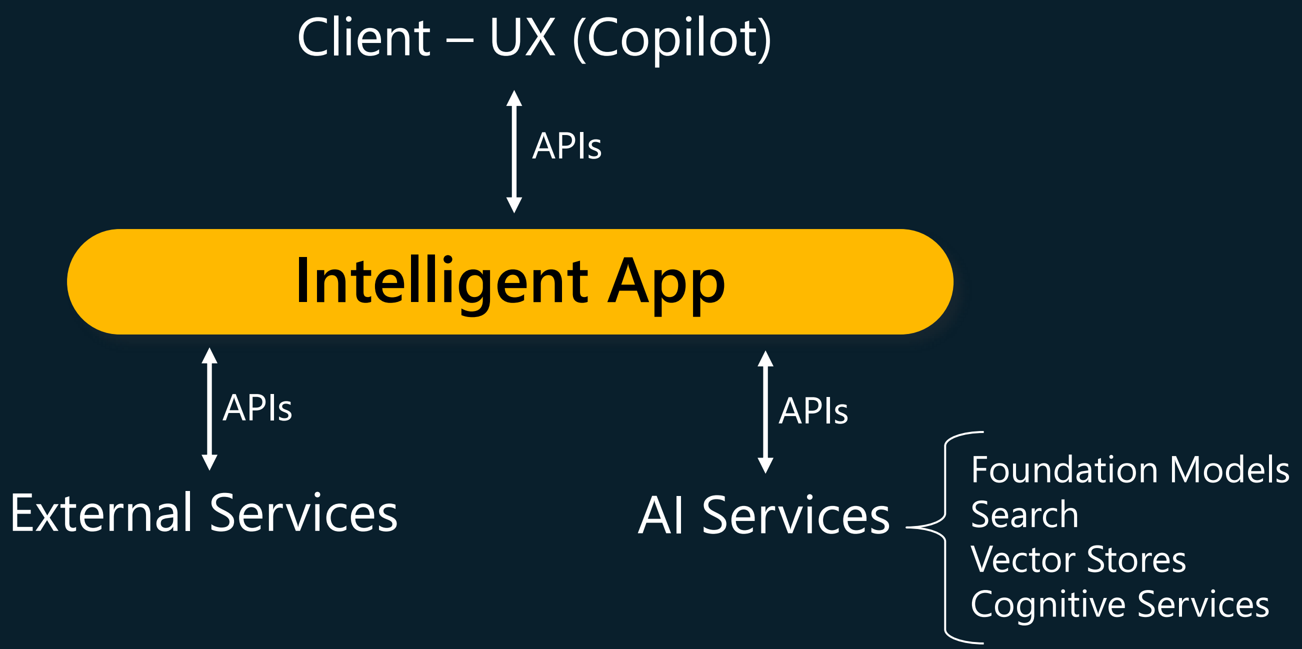
Reliability and Safety

Privacy and Security

Inclusiveness

Transparency

Accountability



Client – UX (Copilot)

APIs

Intelligent App

APIs

APIs

External Services

AI Services

Foundation Models

Search

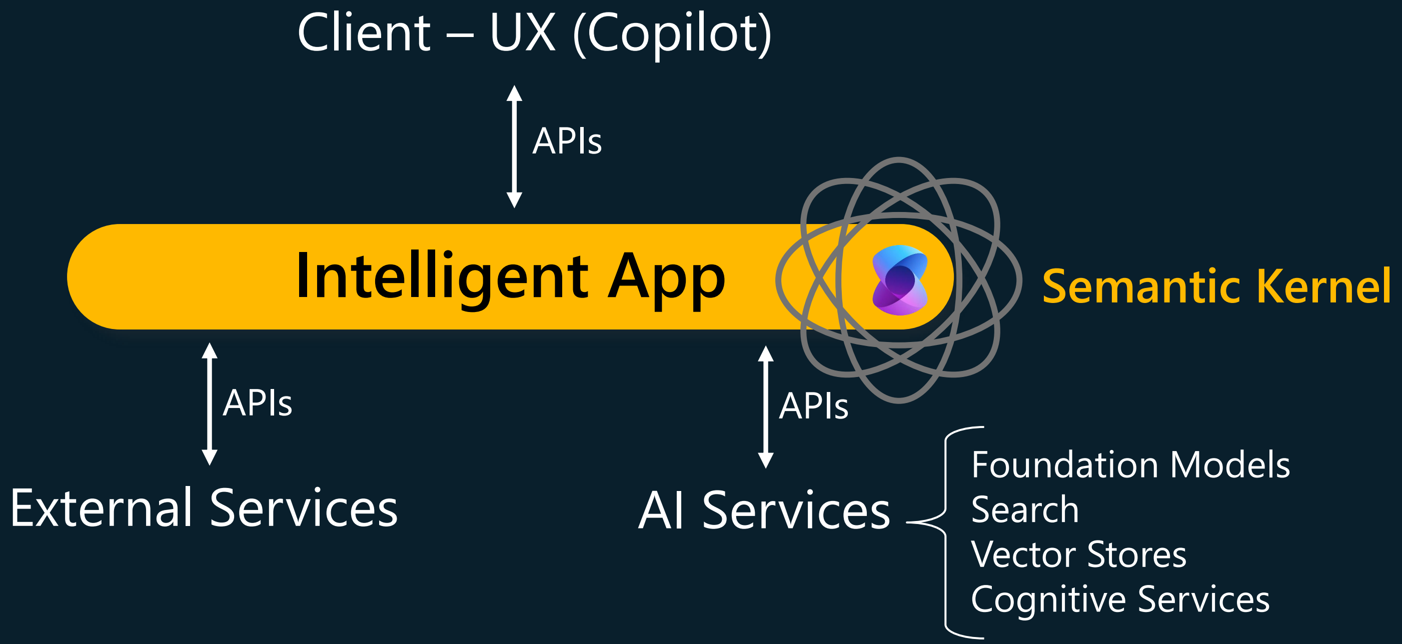
Vector Stores

Cognitive Services

AI Application

Responsible AI

Fairness
Reliability and Safety
Privacy and Security
Inclusiveness
Transparency
Accountability



Client – UX (Copilot)

APIs

Intelligent App

Semantic Kernel

APIs

External Services

APIs

AI Services

Foundation Models
Search
Vector Stores
Cognitive Services

Semantic Kernel

Orchestration middleware that lets you easily add AI to your apps

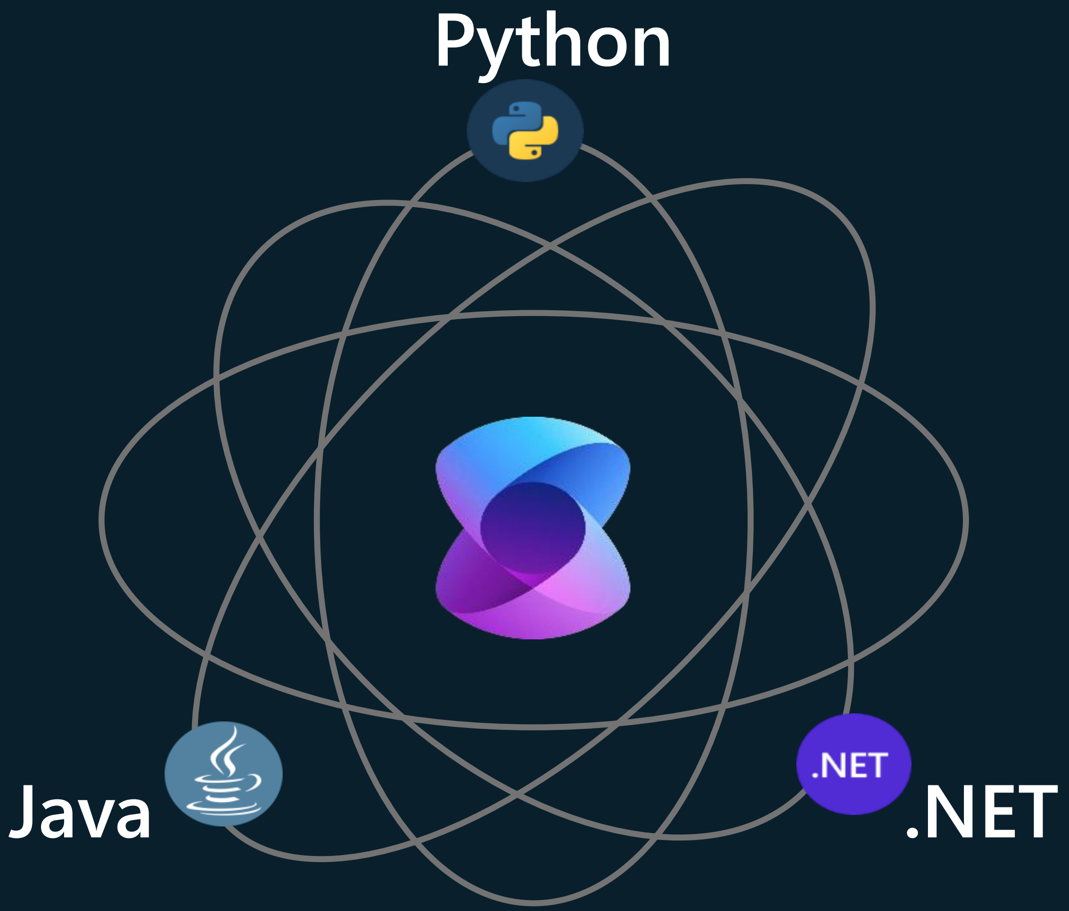
Open-source / Lightweight / Extensible

Built specifically for enterprise app developers

Supported / Trustworthy / Reliable

Includes:

- Connectors to AI Services
- Context / Memories
- Prompts
- Plugins
- Telemetry
- Agent Framework
- Process Framework



https://github.com/microsoft/semantic-kernel

Semantic Kernel / AutoGen

Microsoft have two AI frameworks:

- Semantic Kernel - for enterprise AI applications – fully supported / commitment to stability and non-breaking changes. https://github.com/microsoft/semantic-kernel
- AutoGen - from Microsoft Research intended to ideate and experiment. https://github.com/microsoft/autogen

Working on a mechanism to seamlessly transition from AutoGen to Semantic Kernel

Semantic Kernel v AutoGen - which framework is more appropriate for production?

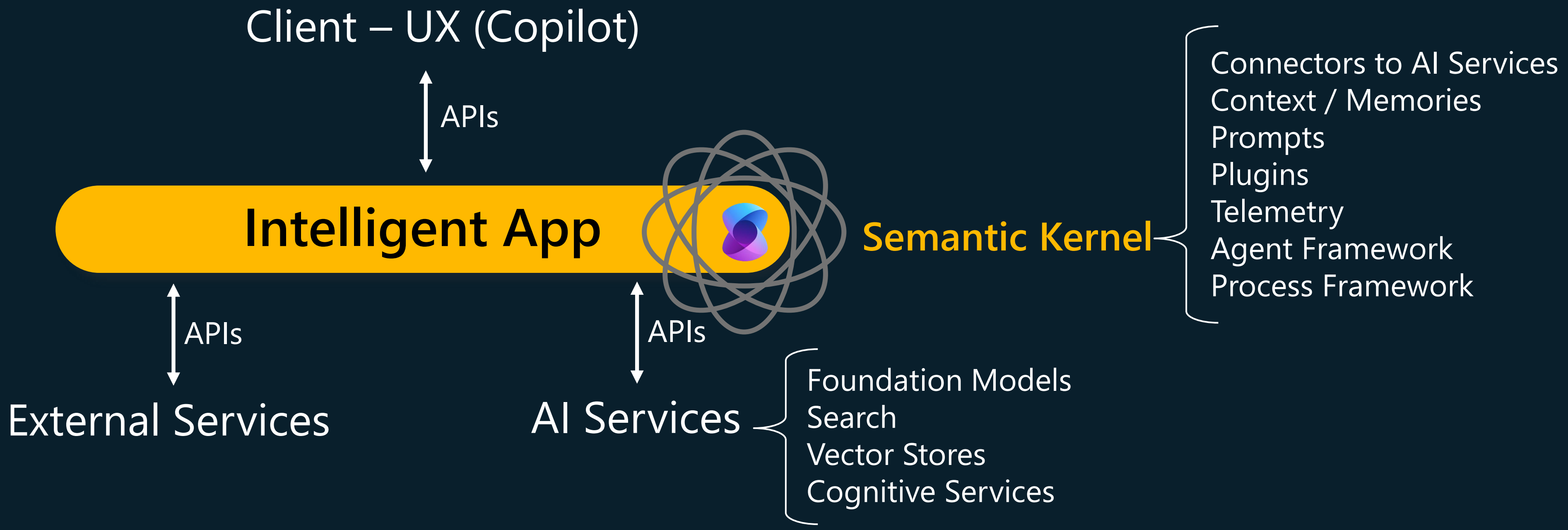
https://devblogs.microsoft.com/semantic-kernel/microsofts-agentic-ai-frameworks-autogen-and-semantic-kernel/

Semantic Kernel Roadmap H1 2025

https://devblogs.microsoft.com/semantic-kernel/semantic-kernel-roadmap-h1-2025-accelerating-agents-processes-and-integration/

Intelligent Application

Responsible AI
Fairness
Reliability and Safety
Privacy and Security
Inclusiveness
Transparency
Accountability



Client – UX (Copilot)

APIs

Intelligent App

Semantic Kernel

- Connectors to AI Services
- Context / Memories
- Prompts
- Plugins
- Telemetry
- Agent Framework
- Process Framework

APIs

External Services

APIs

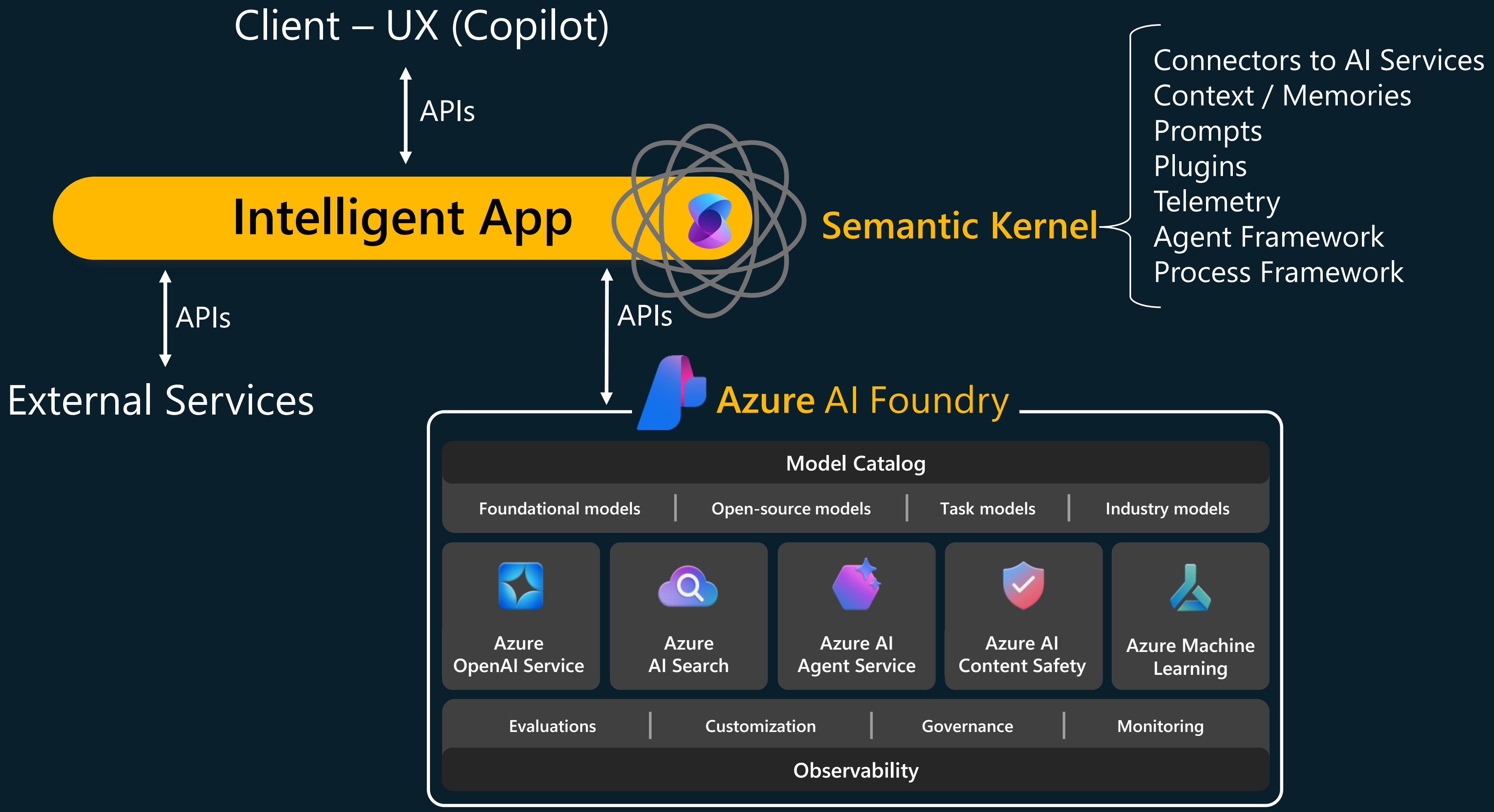
AI Services

- Foundation Models
- Search
- Vector Stores
- Cognitive Services

Intelligent Application

Responsible AI

Fairness
Reliability and Safety
Privacy and Security
Inclusiveness
Transparency
Accountability



Client – UX (Copilot)

APIs

Intelligent App

APIs

External Services

APIs

Semantic Kernel

Connectors to AI Services
Context / Memories
Prompts
Plugins
Telemetry
Agent Framework
Process Framework

Azure AI Foundry

Model Catalog

Foundational models | Open-source models | Task models | Industry models

Azure OpenAI Service

Azure AI Search

Azure AI Agent Service

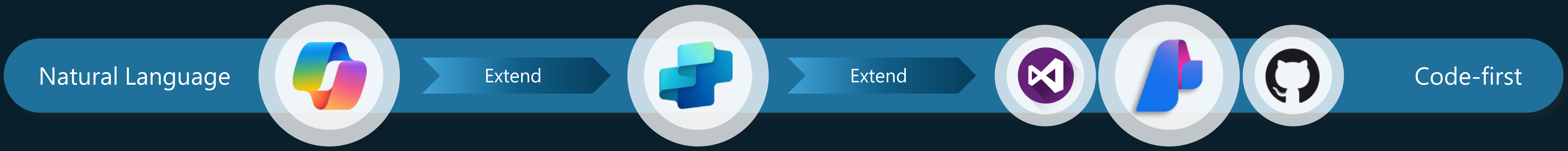
Azure AI Content Safety

Azure Machine Learning

Evaluations | Customization | Governance | Monitoring

Observability

Building AI applications – Compose / Code



Agent Builder

For end-users

Copilot Studio

For creators

Azure AI Foundry

For developers / data scientists

Low-code with a managed SaaS stack
Rapid authoring / time to value
Build agents for business processes
All-in pricing

Pro-code with full flexibility & control
Cutting-edge AI, 1400+ models
Accelerate with a library of SDKs.
Component level pricing

Show some code !

https://github.com/markharrison/sk-Chatter-Bing

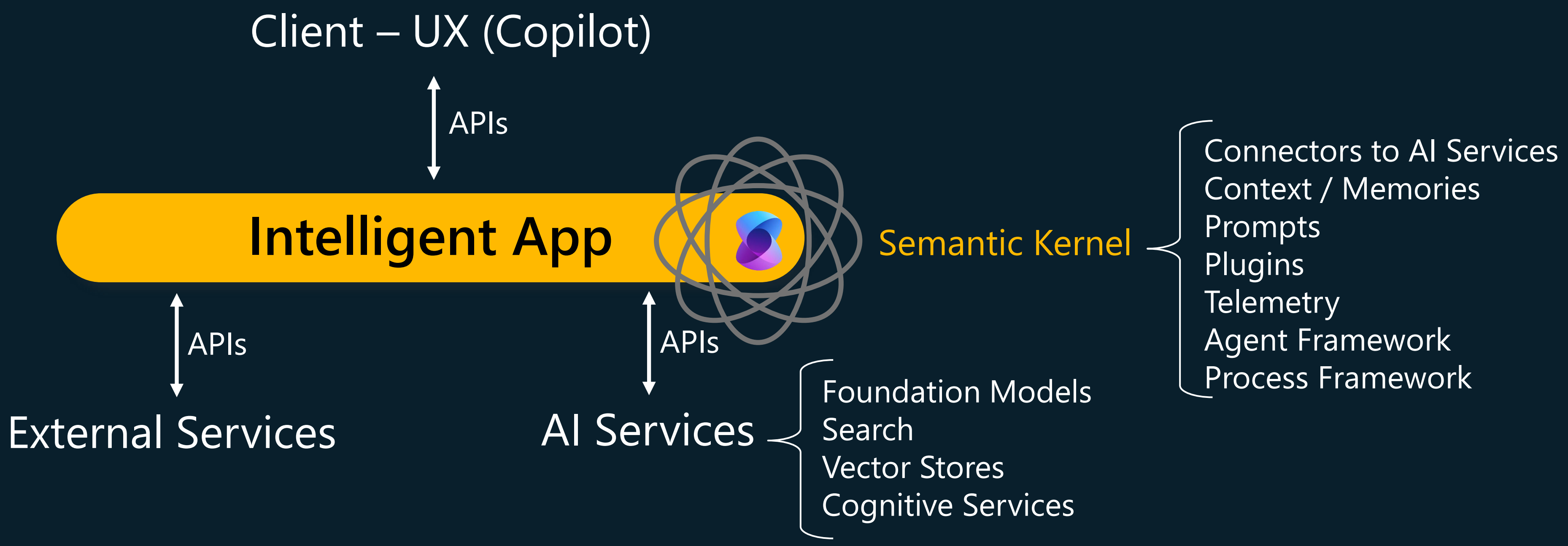
https://github.com/markharrison/sk-DocUploader

https://github.com/markharrison/sk-Chatter-ToDo

https://github.com/markharrison/sk-Chatter-MultiAgent

https://github.com/markharrison/sk-Process-Triage

Intelligent Application



Client – UX (Copilot)

APIs

Intelligent App

Semantic Kernel

- Connectors to AI Services
- Context / Memories
- Prompts
- Plugins
- Telemetry
- Agent Framework
- Process Framework

APIs

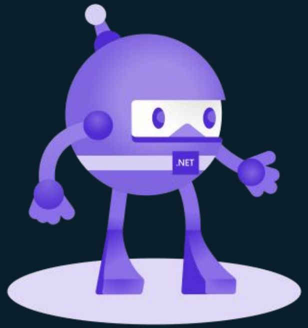
External Services

APIs

AI Services

- Foundation Models
- Search
- Vector Stores
- Cognitive Services

Adding Semantic Kernel to your .NET application



Semantic Kernel for .NET is distributed as a set of NuGet packages.

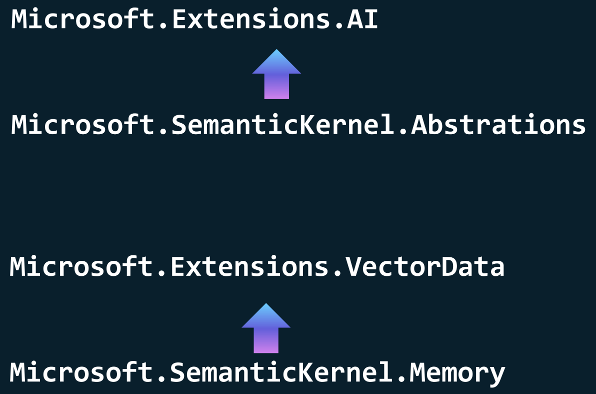
Each package typically focuses on a specific area of functionality, allowing developers to include only the components they need in their applications.

This modular approach helps keep dependencies lightweight and targeted.

```
using Microsoft.SemanticKernel;    // the core package

using Microsoft.SemanticKernel.ChatCompletion;
using Microsoft.SemanticKernel.Connectors.AzureOpenAI;
using Microsoft.SemanticKernel.Connectors.OpenAI;
using Microsoft.SemanticKernel.Plugins.Web;
using Microsoft.SemanticKernel.Plugins.Web.Bing;

// create the kernel instance
var kernel = Kernel.Builder.Build();
```



Microsoft.Extensions.AI

Microsoft.SemanticKernel.Abstrations

Microsoft.Extensions.VectorData

Microsoft.SemanticKernel.Memory

Demo – chat application

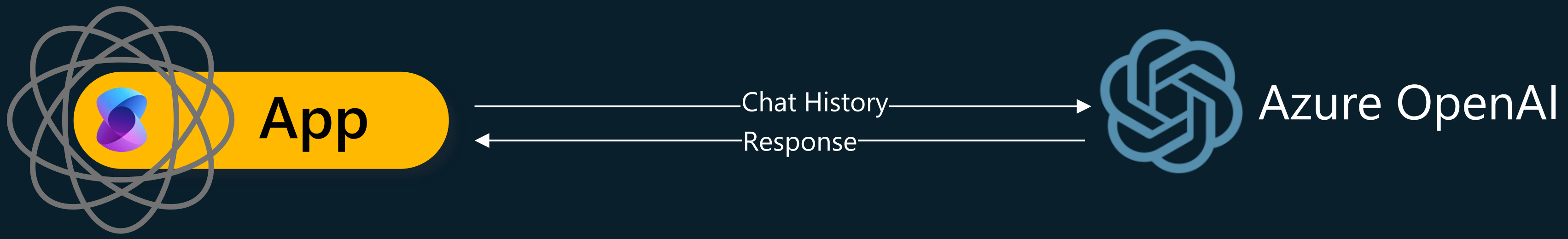
```
using Microsoft.SemanticKernel.ChatCompletion;  # Chat Service / Chat History
using Microsoft.SemanticKernel.Connectors.AzureOpenAI;  # could use alternative models
```

Create: Kernel + Chat Service + Chat History

Add system message to History

Do loop:

- Get user prompt … add user message to History
- Call AI endpoint
- Display response … add response to History



Enhancing generative models

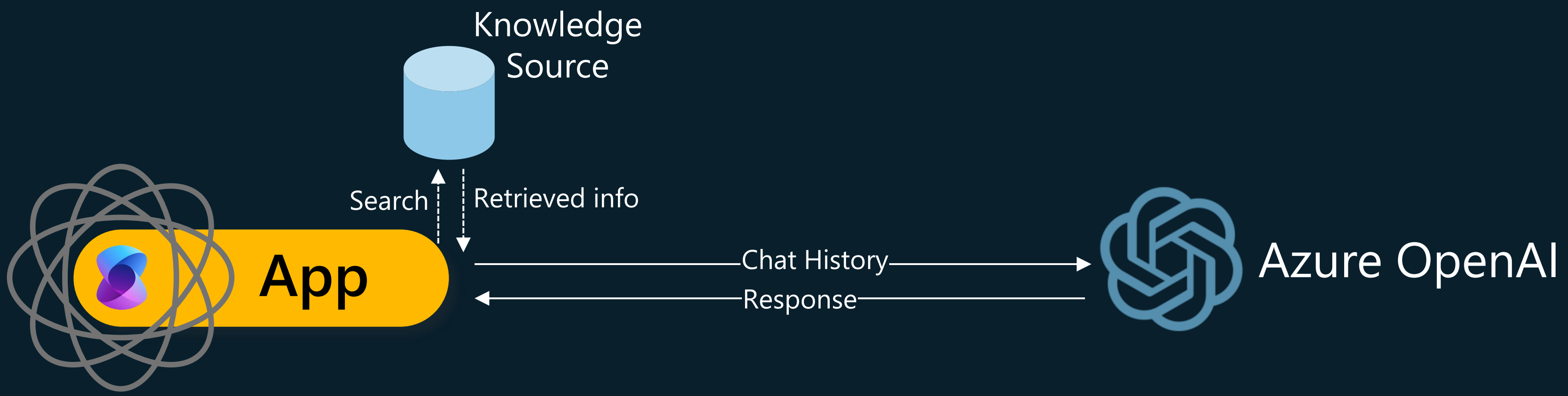
A popular pattern is Retrieval-Augmented Generation (RAG)

Combines generative AI with additional data to overcome reliance on outdated public datasets.

Enables real-time, accurate information access.

Can incorporate private or specialised knowledge.

Use Vectors Stores to enable semantic searches e.g. Azure AI Search.



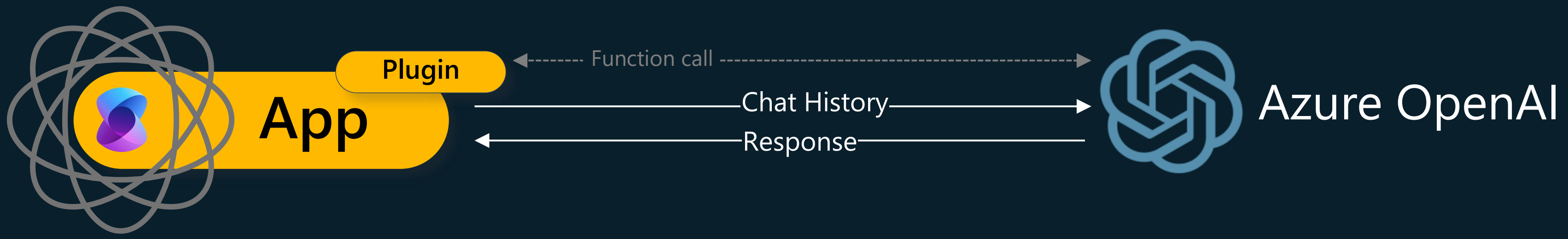
Plugins

A plugin is a self-contained unit of logic for specific tasks and can be used to integrate to other systems or incorporate business logic.

Allows LLMs to invoke your code & APIs.

Semantic Kernel marshals the request to the appropriate function in your codebase and returns the results back to the LLM so the LLM can generate a final response.

Some plugins are prebuilt supplied with SK - but can write your own.

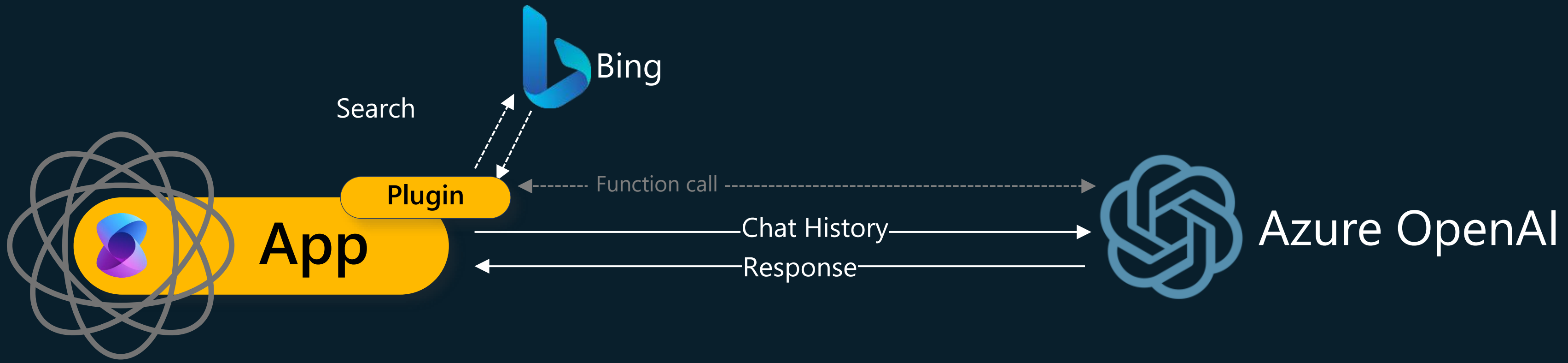


Demo – chat application with Bing plugin

Returning to the RAG pattern

The Bing Search plugin enables the LLM to ground its responses using up-to-date information retrieved from the web..

```
using Microsoft.SemanticKernel.Plugins.Web;
using Microsoft.SemanticKernel.Plugins.Web.Bing;
```



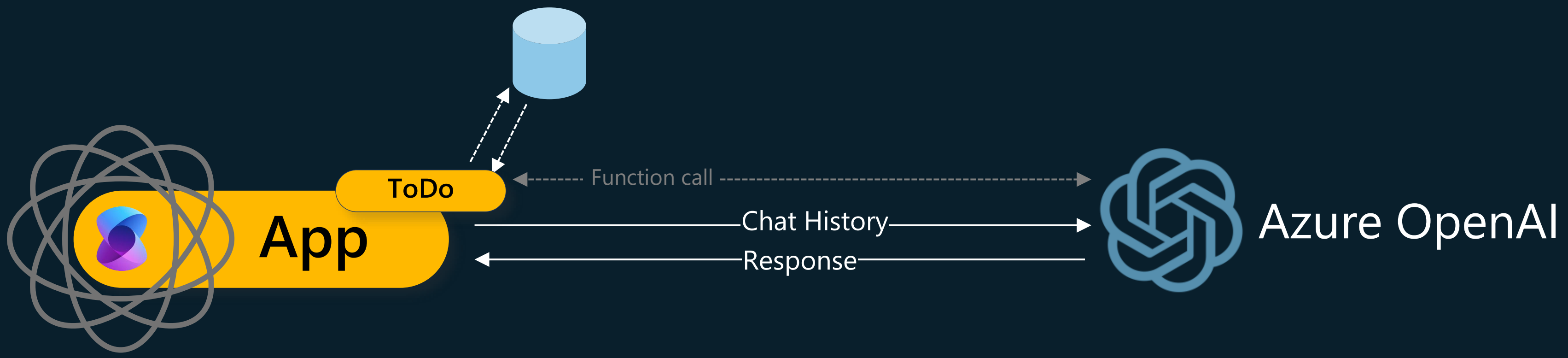
https://github.com/markharrison/sk-Chatter-Bing

Demo – chat application with custom plugin

Custom plugin to handle “To Do” items

Functions:

```
public List<ToDoItem> GetToDoItems()
public string AddToDoItem(string description)
public string RemoveToDoItem(string description)
```



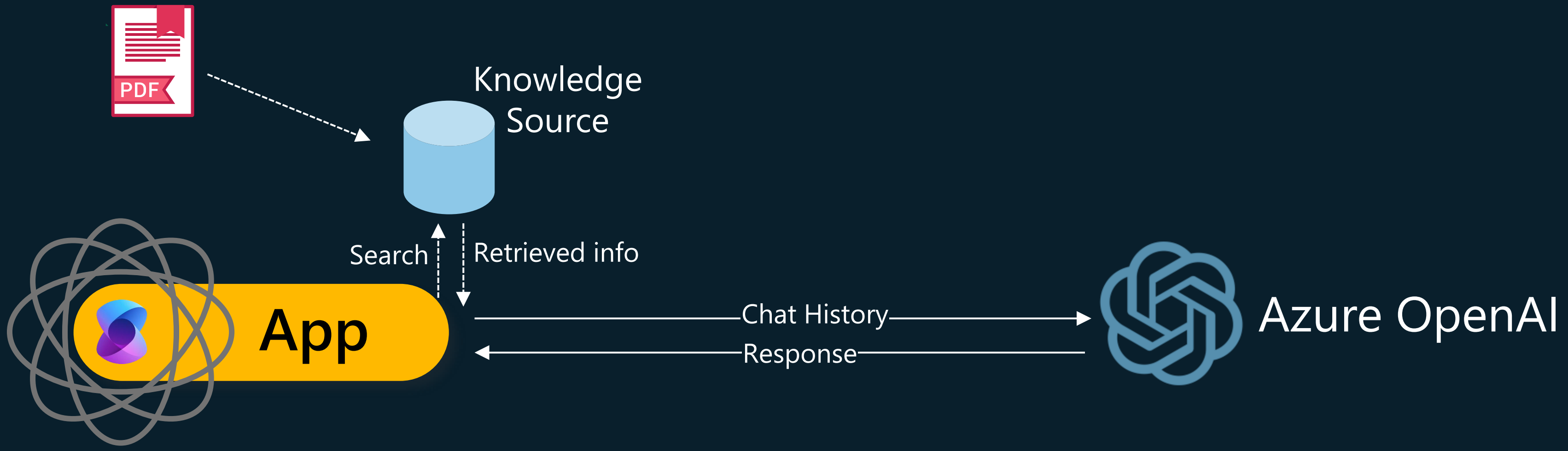
https://github.com/markharrison/sk-Chatter-ToDo

Demo – document uploader

Returning to RAG. What if our information is in documents such as PDF, Word, ….

Need to

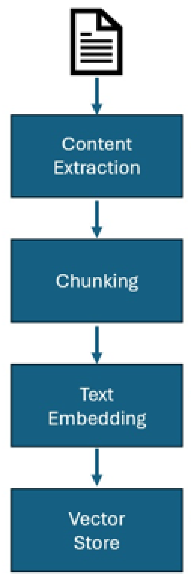
- Upload document / extract the contents.
- Split the document contents up into small chunks.
- Vectorise the chunk
- Add the chunk plus its vectorised version into a vector store

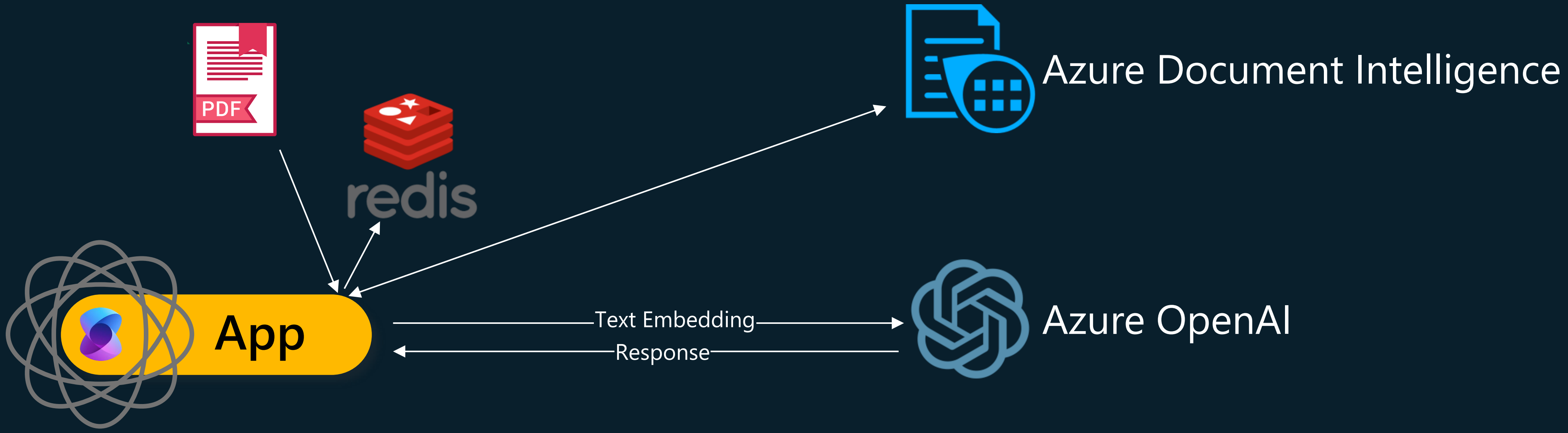


https://github.com/markharrison/sk-DocUploader

Demo – document uploader

```
using Azure.AI.DocumentIntelligence;
using Microsoft.SemanticKernel.Embeddings;
using Microsoft.Extensions.VectorData;
using Microsoft.SemanticKernel.Connectors.AzureOpenAI; # could use alternative text embedding
using Microsoft.SemanticKernel.Connectors.Redis;       # could use alternative vector stores
```





https://github.com/markharrison/sk-DocUploader

mark.harrison@microsoft.com

To implement complex agentic architectures, we now have two frameworks, which can be used independently or combined:

- Agents Framework
- Process Framework

Agent Framework

The SK Agent Framework provides a platform to allow for the creation of AI agents and the ability to incorporate agentic patterns.

Multiple agents can collaborate within a single conversation.

An agent can engage in and manage multiple concurrent conversations simultaneously.

Different types of agents can participate in the same conversation, each contributing their unique capabilities.

Agent Class

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.

Subclass is *Kernel Agent* which connects to an instance of Semantic Kernel.

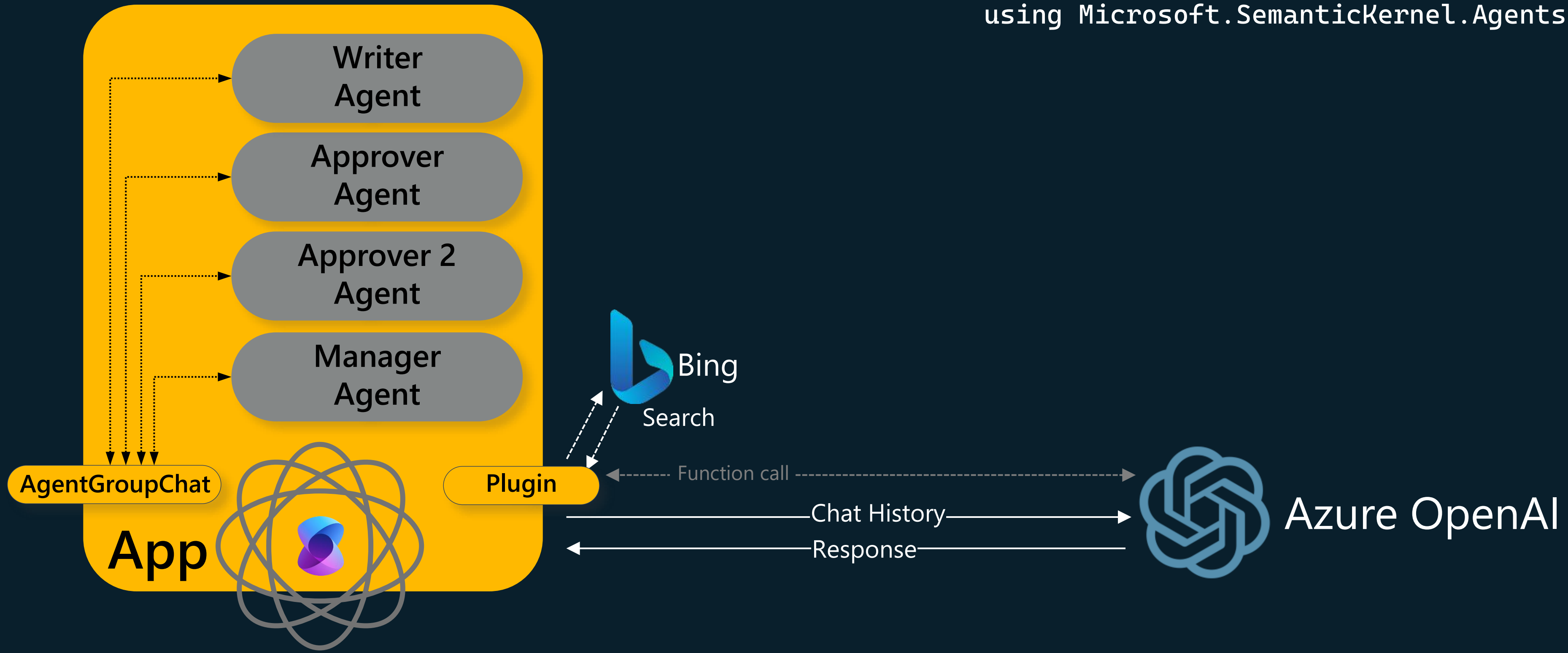
Agents can either be invoked directly to perform tasks or orchestrated within an *AgentGroupChat*, where multiple agents may collaborate or interact dynamically with user inputs.

Demo

Agents work together to achieve an objective.

- In a multi-turn invocation, the system must decide which agent responds next and when the conversation should end
- Selection : define whose turn it is – which agent next.
- Termination : when the dialogue should stop – has the objective been achieved.

```
using Microsoft.SemanticKernel.Agents.Core;
using Microsoft.SemanticKernel.Agents.Chat;
```



https://github.com/markharrison/sk-Chatter-MultiAgent

Process Framework

The integration of AI into business processes has become increasingly important.

The SK Process Framework empowers developers to efficiently create, manage, and deploy business processes while leveraging the powerful capabilities of AI.

Concepts

- Process: A collection of steps arranged to achieve a specific business goal for customers.
- Step: An activity within a process that has defined inputs and outputs, contributing to a larger goal.
- Event: Something has happened – trigger a step.

Process

Process is the overarching container – orchestrates flow/routing of data between the steps

- A process contains multiple steps

A step is a discrete until of work

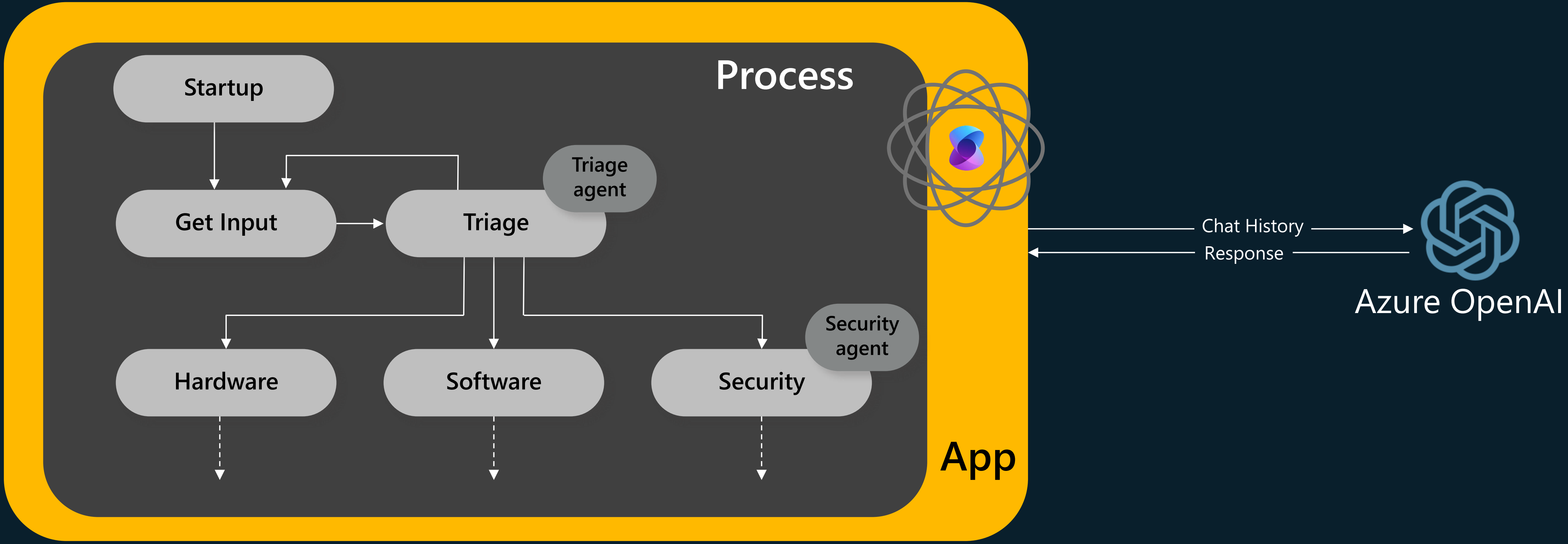
- A step contains one or more functions
- A step emits events that can trigger subsequent steps.

Demo – support ticket

Process to manage support tickets

Triage step uses chat agent to classify the type of ticket and route appropriately to next steps

```
using Microsoft.SemanticKernel.Process.Core;
```



https://github.com/markharrison/sk-Process-Triage

mark.harrison@microsoft.com

Thank you

mark.harrison@microsoft.com

Microsoft

mark.harrison@microsoft.com

Links

Unlocking human potential starts with trust

https://www.microsoft.com/ai/responsible-ai