Microsoft

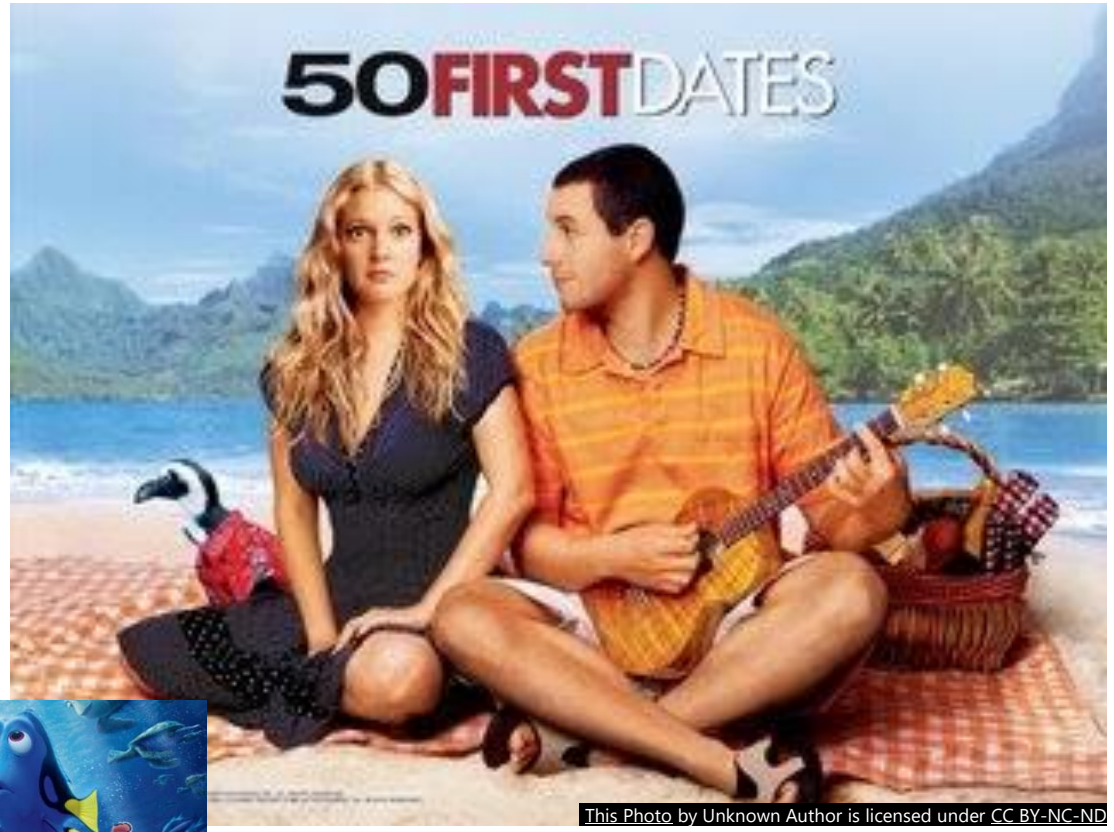
How to Build Chat GPT with Semantic Kernel?

Yuexin Mao
Cloud Solutions Architects

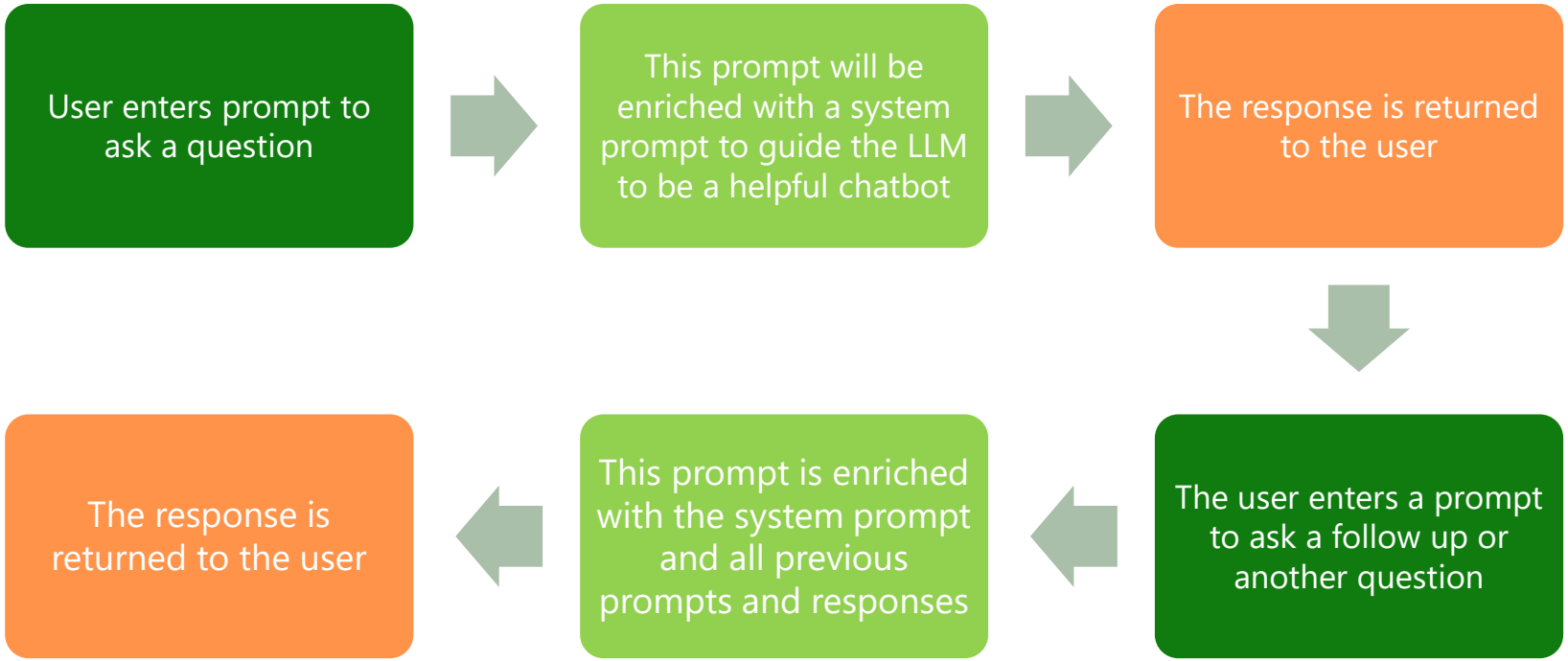
Walid Amro
Cloud Solutions Architects – Developer Advocate

Facts about Large Language Models (LLMs)

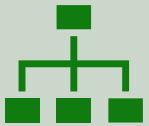
- LLMs Do Not Chat!
- They take a prompt and return a response.
- They don't understand previous prompts.
- They don't understand any kind of history behind what they do.
- You simply give them some text and they give some text back with no other context than the text you give them.
- If you want them to chat, then they need guidance.



How does Chat GPT Work?



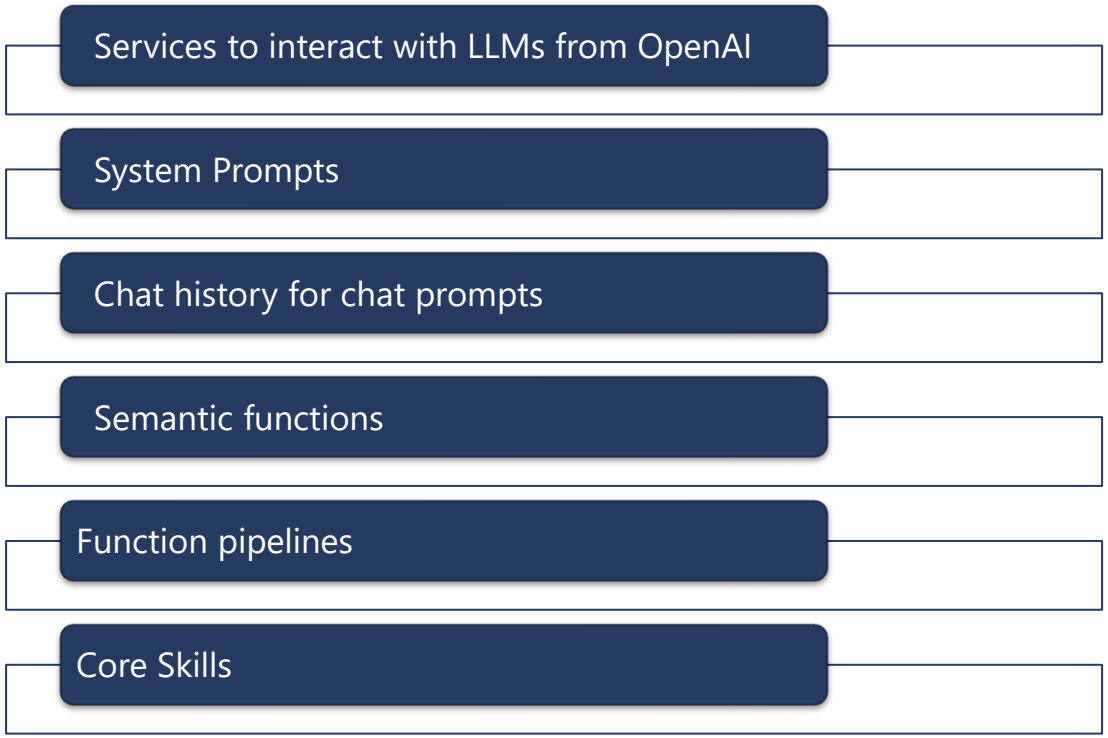
What do we need to design a Chat GPT?



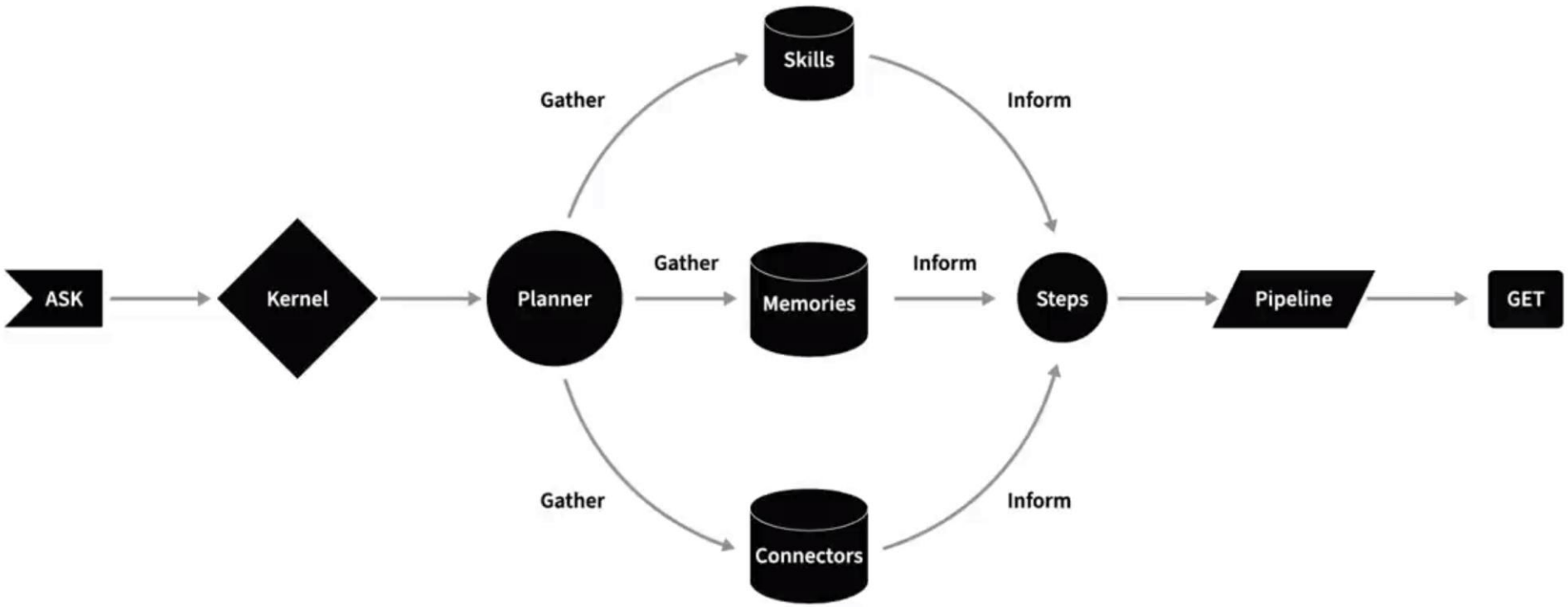
LLM (i.e. GPT 3, 3.5, or 4)

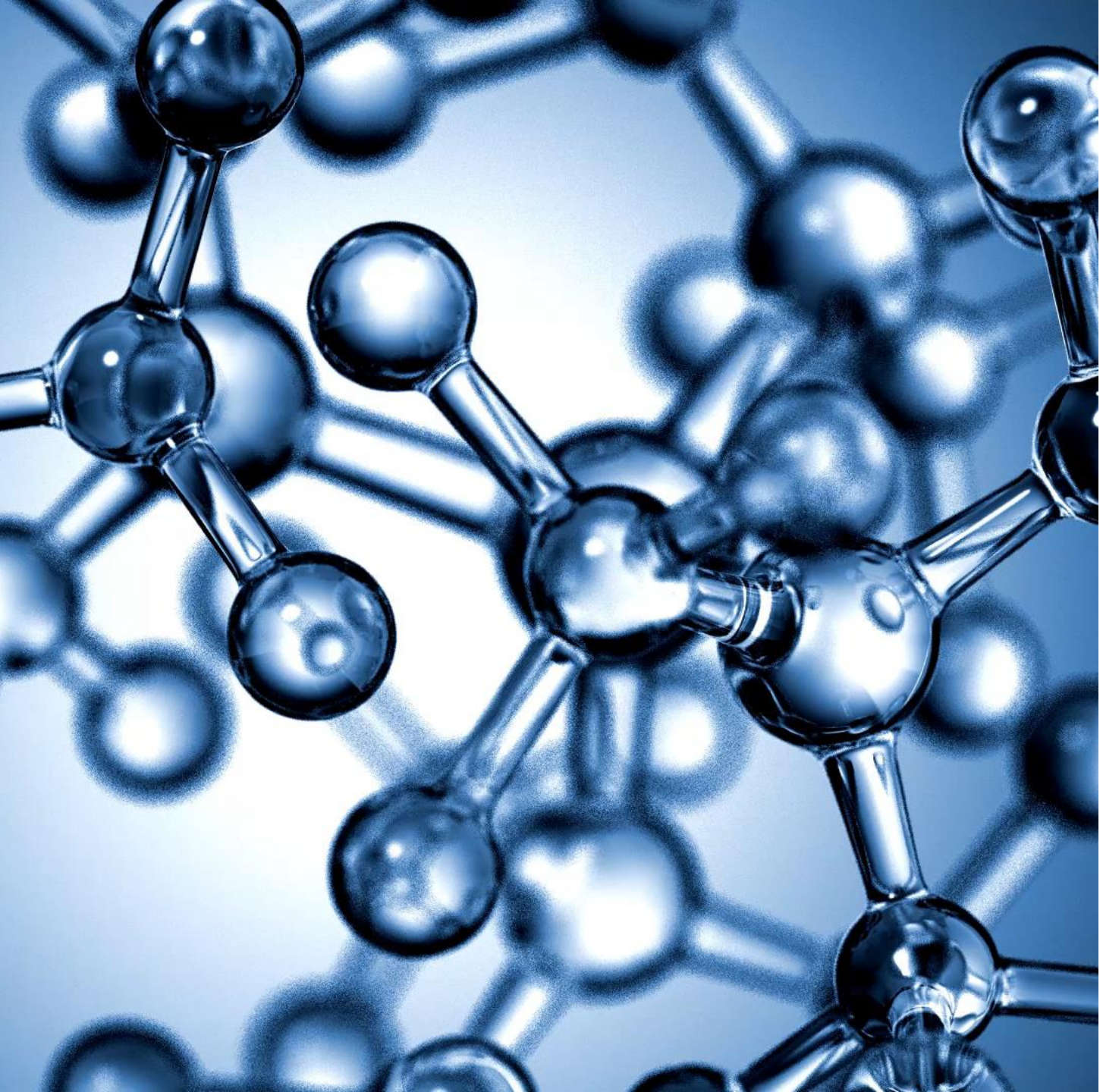
Gathering prompts from user, to build memory for the chat history for LLM

How Semantic Kernel Can Help?



How Semantic Kernel Works?





Semantic Kernel Definitions

Semantic Kernel is a framework for building natural language understanding systems. It consists of the following components:

- **Ask**: It starts with a goal being sent to Semantic Kernel by either a user or developer.
- **Kernel**: The kernel orchestrates a user's ask. To do so, the kernel runs a pipeline / chain that is defined by a developer. While the chain is run, a common context is provided by the kernel so data can be shared between functions.
- **Skills**: These are the tasks that the system can perform, such as answering questions, summarizing texts, generating captions, etc.
- **Memories**: With a specialized plugin, a developer can recall and store context in vector databases. This allows developers to simulate memory within their AI apps.
- **Connectors**: To get additional data or to perform autonomous actions, you can use out-of-the-box plugins like the Microsoft Graph Connector kit or create a custom connector to provide data to your own services.
- **Planner**: This is the component that decides which skills and connectors to use for a given input and output.
- **Steps**: These are the intermediate representations that the system produces and consumes, such as queries, embeddings, labels, etc.
- **Pipeline**: This is the sequence of steps that the system executes to perform a skill.
- **Response**: Once the kernel is done, you can send the response back to the user to let them know the process is complete.

Why should I use Semantic Kernel?

- Fast integration.
- Common integration layer for many AI technologies.
- Most advanced in the market.
- Open source.


Microsoft
Thank you