Microsoft Azure

intel

CLOUD NATIVE COMPUTING FOUNDATION

When Your App Begins to Talk

Engineering behind “GPT-ize” enterprise applications and beyond

Lei Zhang

Partner Software Engineer @Azure Dev Compute

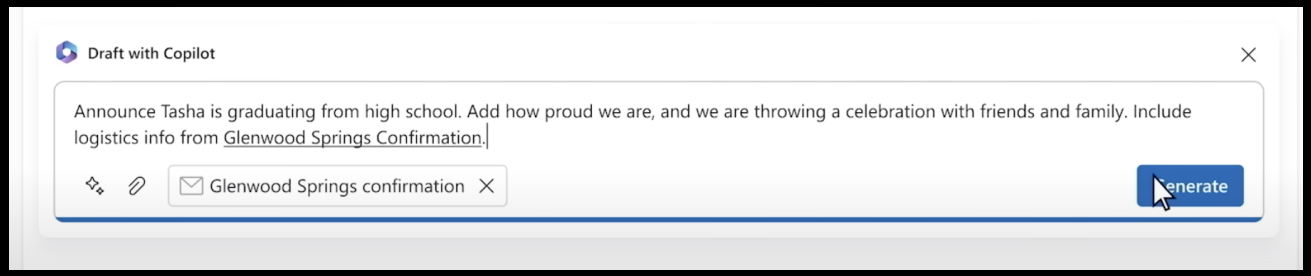
Mar 2023

目录

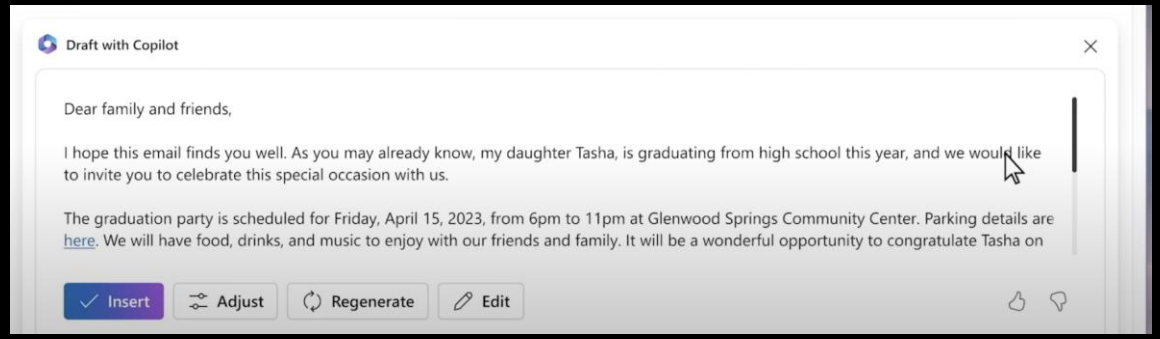
CATALOGUE

Microsoft 365 Copilot

You say:



Copilot gives you:



How is This Implemented?

"GPT-ize" Microsoft Office apps!

Explain ChatGPT Like a 3-Year-Old

GR

I want you to act as a Linux terminal. I will type commands and you will reply with what the terminal should show. I want you to only reply with the terminal output inside one unique code block, and nothing else. Do no write explanations. Do not type commands unless I instruct you to do so. When I need to tell you something in English I will do so by putting text inside curly brackets {like this}. My first command is pwd.

Copy code

```
/
```

GR

cd ~

Copy code

```
```

GR

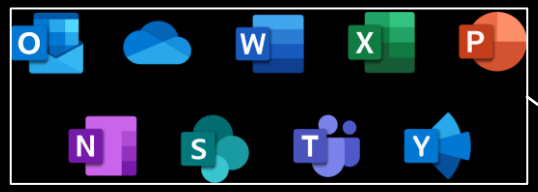
{Please make a file jokes.txt inside and put some jokes inside}

Copy code

```
touch jokes.txt
echo "Why did the chicken cross the road? To get to the other side." >>
jokes.txt
echo "Why couldn't the bicycle stand up by itself? Because it was two-
tired." >> jokes.txt
```

OK, it's all about ***Prompts*** and ***Responses!***

“GPT-ize” Your App – What Dose it Really Mean?



Microsoft 365

1. User prompt

Copilot

3. Modified prompt

LLM
(e.g., ChatGPT)

Like a “black box”

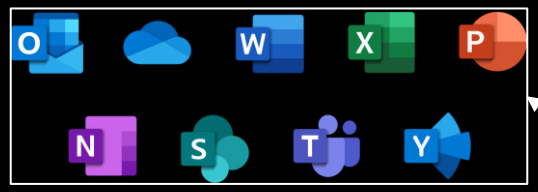
2. Modify prompt with domain knowledge

Data
(e.g., emails, files, meetings, calendars, contacts)

Microsoft Graph

Request Path

“GPT-ize” Your App – What Dose it Really Mean?



Microsoft 365

3. Response + App commands

Copilot

1. LLM response

LLM
(e.g., ChatGPT)

Like a “black box”

2. Post processing

Data
(e.g., emails, files, meetings, calendars, contacts)

Microsoft Graph

Response Path

“GPT-ize” Your App – Explain Like a 3-Year-Old

- Treat ChatGPT/LLM as a “black box”
- Pre-process
  - Modify prompt with domain knowledge
- Post-process
  - Modify responses with:
    - domain knowledge
    - app commands/DSL

See? Still all about ***Prompts*** and ***Responses!***

“GPT-ize” Your App – Explain Like an Engineer

- The essentials of a LLM app
  - A chain of ***one or multiple prompted calls*** to LLM models or external services (API or data) in order to achieve a particular task based on natural language input
- Key traits
  - The calls are normally sent to ***remote endpoints***, and the endpoints could be slow/dead
  - ***Add-hoc “grounding” activities*** in-between the calls, most for “pre/post-processing” purposes
  - Multiple calls/activates are chained together, executed in parallel, and needs certain ***orchestration***

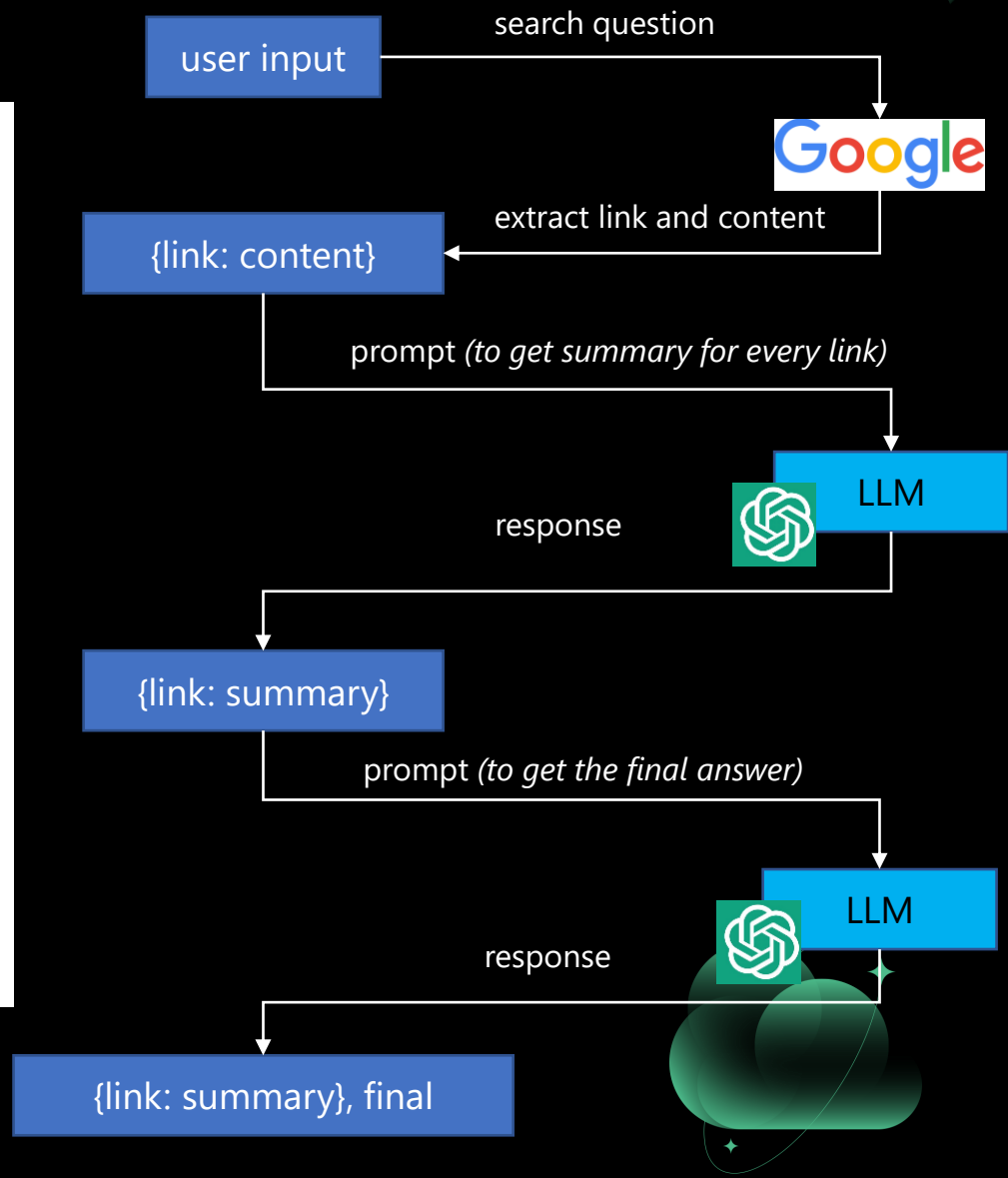
“GPT-ize” Your App – DIY a Search Assistant

question	How can different animals perceive different colors?

summaries	[
	  {
	  "link": "https://crosstalk.cell.com/blog/5-things-you-didnt-know-about-how-animals-see-color",
	  "summary": "Different animals perceive different colors because they have different numbers of color-receptor cones in their eyes. Insects, for example, can see in ultraviolet."
	  },
	  {
	  "link": "https://askabiologist.asu.edu/colors-animals-see",
	  "summary": "Different animals perceive different colors because they have different types of cones in their eyes. Humans have three types of cones, which allow us to see a range of colors, but some animals only have two types of cones, which limits the colors they can see."
	  },
	  {
	  "link": "https://www.colormatters.com/color-matters-for-kids/how-animals-see-color",
	  "summary": "Different animals have different abilities to see colors, with some animals seeing a very limited range of colors while others have a much wider range. In some cases, animals can see colors that humans cannot see. Good color vision in animals helps them to find food, identify potential mates, and spot predators. One example of an animal with good color vision is the bee, which can see colors in the ultraviolet range. Another example is the pit viper, which can see heat in an object."
	  }
	]

final	Different animals perceive different colors because they have different numbers[0] and types[1] of color-receptor cones in their eyes. As a consequence, some animals may see a very limited range of colors while others have a much wider range[2]. Humans have three types of cones but other animals may have few types of cones which limits the range of colors they can see[1]. Finally, some animals may see colors humans cannot see, as an example, the bee, which can see colors in the ultraviolet range, or the pit viper, which can see heat emitted from an object[2].

A Search Assistant *(sample from dust.tt)*



The New Challenges For Developing LLM Apps

- Resiliency & Durability
  - What if a prompt call failed?
  - Do I need to redo everything if one step failed?
- Resource & Developer Efficiency
  - Do I need to provision a full VM/container for each activity?
  - Do I need to learn another full platform to achieve elastic compute?

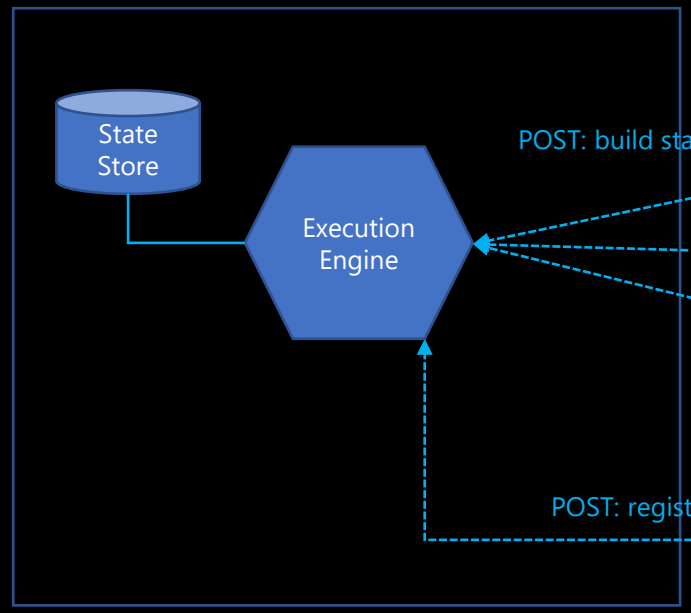
AI Infra concerns

- Compossibility
  - Can I modify and reuse existing prompts/activities from others?
- Best Practices for
  - Prompt chaining, recursive reasoning, summarization, zero/few-shot learning, contextual memory, long-term memory, embeddings, semantic indexing, planning, accessing external knowledge stores/data …

Dev SDK/framework concerns

Resiliency and Durability - The Durable Executions

- Durable Tasks Framework (https://github.com/microsoft/durabletask-go)
  - OSS Engine: Dapr Workflow



Durable Execution Engine (hosted on Cloud)

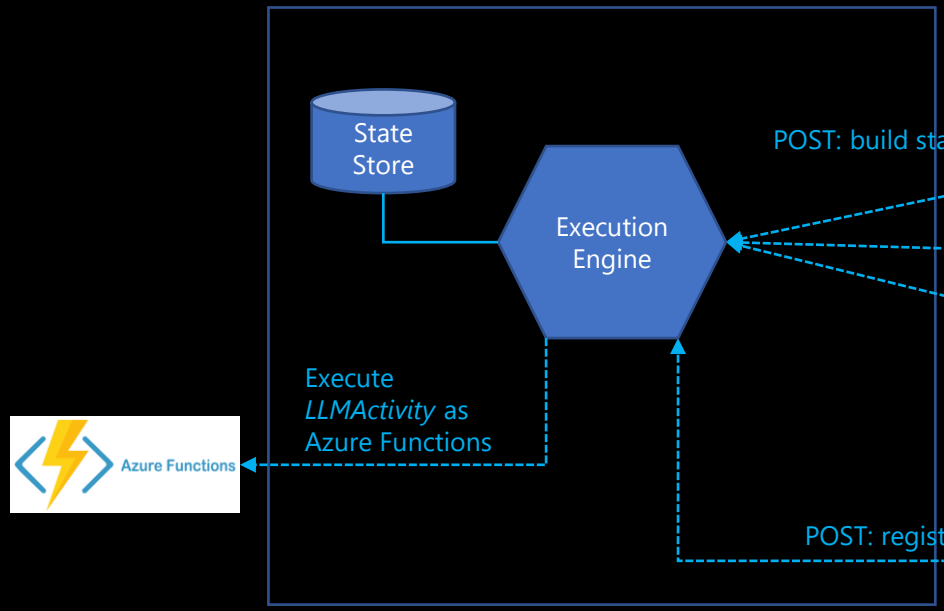
```go
// ActivitySequenceOrchestrator makes three activity calls in sequence and results the results
// as an array.
func ActivitySequenceOrchestrator(ctx *task.OrchestrationContext) (any, error) {
    var helloTokyo string
    if err := ctx.CallActivity(LLMActivity, "Tokyo").Await(&helloTokyo); err != nil {
        return nil, err
    }
    var helloLondon string
    if err := ctx.CallActivity(LLMActivity, "London").Await(&helloLondon); err != nil {
        return nil, err
    }
    var helloSeattle string
    if err := ctx.CallActivity(LLMActivity, "Seattle").Await(&helloSeattle); err != nil {
        return nil, err
    }
    return []string{helloTokyo, helloLondon, helloSeattle}, nil
}

// LLMActivity can be called by an orchestrator function and will feed LLM with a friendly greeting.
func LLMActivity(ctx task.ActivityContext) (any, error) {
    var input string
    if err := ctx.GetInput(&input); err != nil {
        return "", err
    }
    return llm.Feed("Hello, %s!", input), nil
}
```

Your App (runs anywhere)

Resource & Developer Efficiency – Remote Activities

- ~~Your app as an Azure Functions~~
- Your activity as remote/scheduled Azure Functions



Durable Execution Engine (hosted on Cloud)

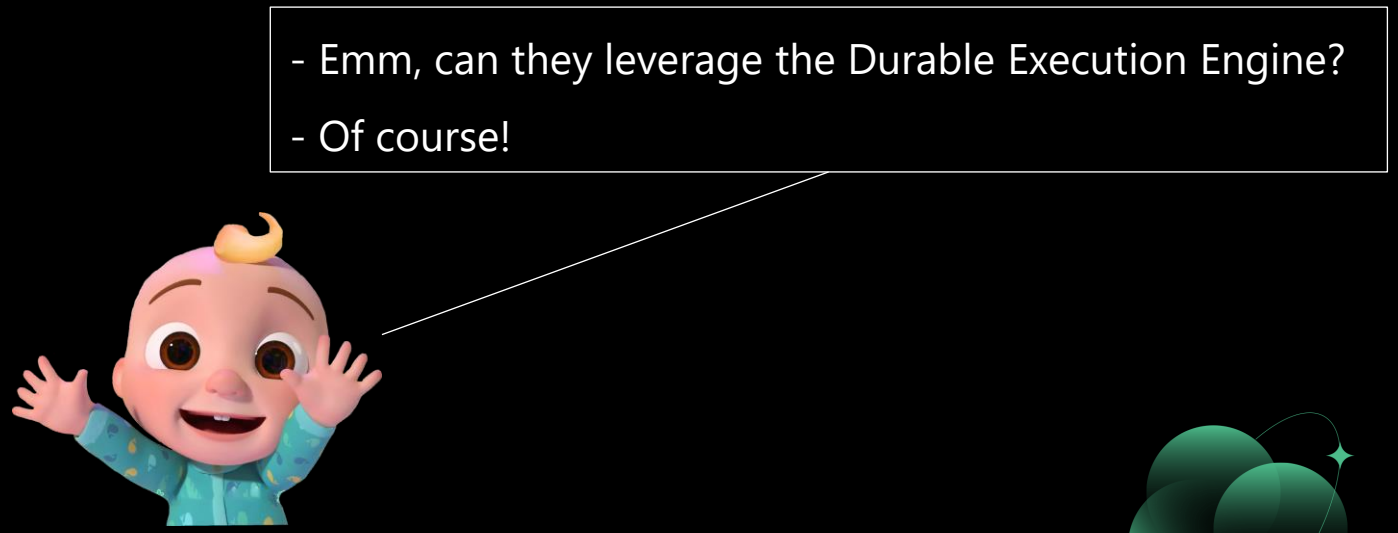
```go
// ActivitySequenceOrchestrator makes three activity calls in sequence and results the results
// as an array.
func ActivitySequenceOrchestrator(ctx *task.OrchestrationContext) (any, error) {
    var helloTokyo string
    if err := ctx.CallActivity(LLMActivity, "Tokyo").Await(&helloTokyo); err != nil {
        return nil, err
    }
    var helloLondon string
    if err := ctx.CallActivity(LLMActivity, "London").Await(&helloLondon); err != nil {
        return nil, err
    }
    var helloSeattle string
    if err := ctx.CallActivity(LLMActivity, "Seattle").Await(&helloSeattle); err != nil {
        return nil, err
    }
    return []string{helloTokyo, helloLondon, helloSeattle}, nil
}

// LLMActivity can be called by an orchestrator function and will feed LLM with a friendly greeting.
// @runner(provider=functions, sku=A10)
func LLMActivity(ctx task.ActivityContext) (any, error) {
    var input string
    if err := ctx.GetInput(&input); err != nil {
        return "", err
    }
    return llm.Feed("Hello, %s!", input), nil
}
```

Your App (runs anywhere)

Best Practices & Compossibility – LLM Dev Framework/SDK

- Semantic Kernel
  - https://github.com/microsoft/semantic-kernel
- Langchain
  - https://langchain.readthedocs.io/en/latest/
- Dust
  - https://dust.tt



What Are Still Missing?

- Abstraction and encapsulation
    - Most apps are not built around “a chain of LLM calls”.
    - Proper abstraction and encapsulation to hide LLM specific details would help the business logic call into LLM capability as needed, not verse versa.
        - Workflow could be the essential model for such encapsulation.

- Quantitative evaluation of the outputs
- Security, audit, tests/validation and versioning of the final prompts and model configs, before pushing directly to the LLM
- Observability of the LLM chain

Thanks