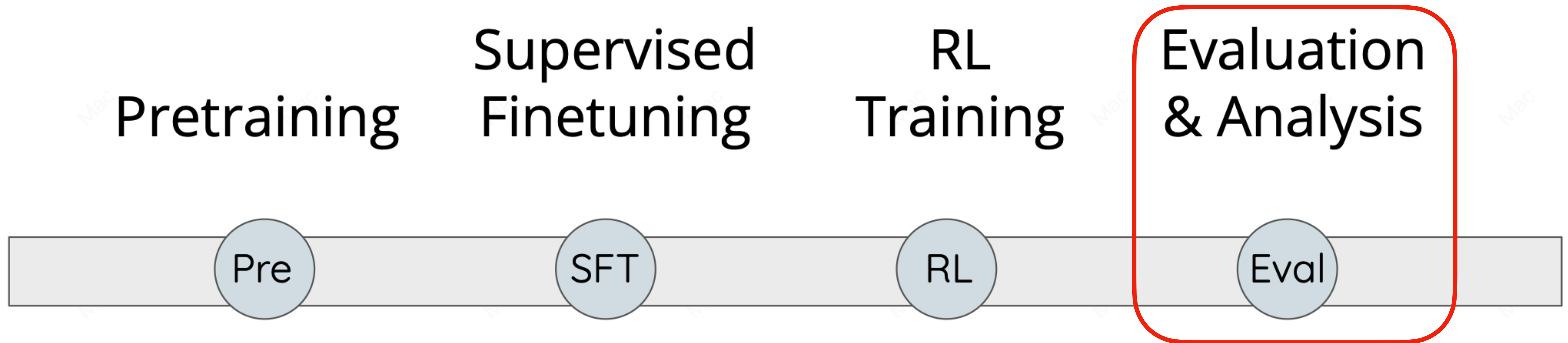


Recap: How Do We Use Synthetic Data



Generating Adversarial Eval Data

For example, we can deliberately train a model to generate prompts that are likely to elicit toxic behaviors, and use the generated prompts to test safety of other LLMs

Synthetic Data for Analyzing LLMs

- Modern example: Kassner, Krojer, and Schutze (2020) generated a synthetic pretraining corpus generated by *synthetic facts* (e.g. “jupiter is big”) and logical rules (e.g. “jupiter is not small”)
- Pretrained BERT from scratch on this corpus
- Discovered that BERT struggles with two-hop reasoning



香港科技大學
THE HONG KONG
UNIVERSITY OF SCIENCE
AND TECHNOLOGY

COMP 4901B

Large Language Models

Language Agents and Tools

Junxian He

Nov 7, 2025

Part of slides are adapted from CMU 11711

LMs are powerful for text generation tasks.

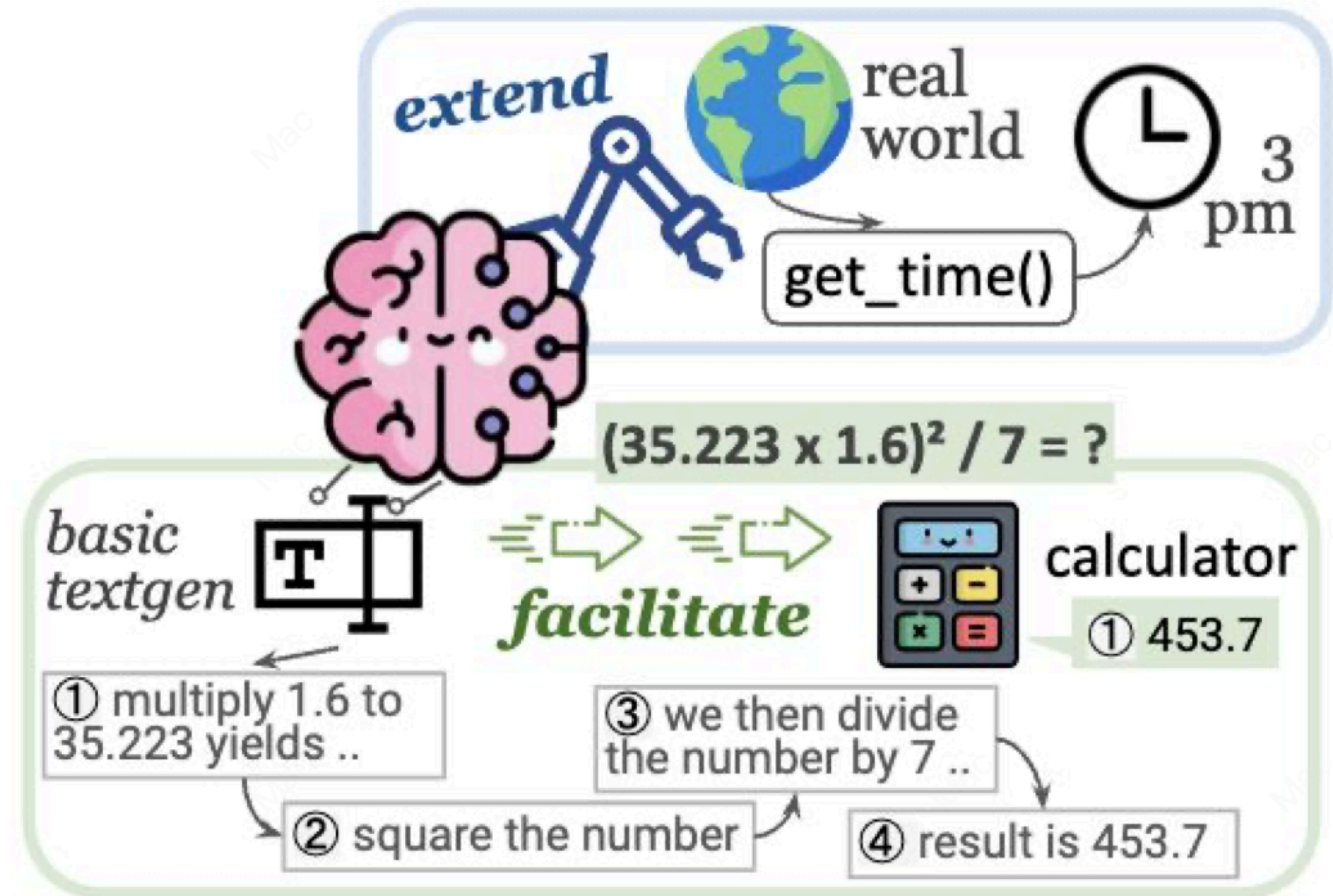
But ...

- Complex reasoning?

Struggle

- Access real-world information?

***Fundamentally
unable***



Tools can Benefit LLMs a Lot

Just imagine, if LMs can use calculators, it can help a lot for mathematical reasoning **How next token predictions elicits external tools?**

The New England Journal of Medicine is a registered trademark of [QA("Who is the publisher of The New England Journal of Medicine?") → Massachusetts Medical Society] the MMS.

Out of 1400 participants, 400 (or [Calculator(400 / 1400) → 0.29] 29%) passed the test.

The name derives from "la tortuga", the Spanish word for [MT("tortuga") → turtle] turtle.

The Brown Act is California's law [WikiSearch("Brown Act") → The Ralph M. Brown Act is an act of the California State Legislature that guarantees the public's right to attend and participate in meetings of local legislative bodies.] that requires legislative bodies, like city councils, to hold their meetings open to the public.

Tools can be Very Diverse

APIs



Search for APIs

Create TeamAdd Your APIDocsLog InSign Up

Video, Images APIs

Video and Images APIs are tools that provide developers with access to various functionalities related to managing, processing, and integrating video and image content into applications, websites, or services.

Search

Collections

[Recommended APIs](#)[Popular APIs](#)[Free Public APIs for Developers](#)[Top AI Based APIs](#)[Tax APIs](#)[View All Collections](#)

Categories

[Sports](#)[Finance](#)[Data](#)[Entertainment](#)[Travel](#)[View All Categories](#)



Cars image background removal

Remove background and blur license plates in vehicle photos. [Examples](#)

Verified 

9.5

6,764 ms

100%



HTML to ZPL

Convert HTML/PDF to ZPL for Zebra label printers

9.6

2,396 ms

100%



YouTube Mp3 Downloader

Update: 19 Apr 2024: API is fixed & it is now 100% working. Now supports YouTube Music Premium

9.7

24,858 ms

92%



Toonify

Tools for face transformation

9.4

12,135 ms

100%



TikTok - Download video no watermark

Get TikTok/Douyin video with/without watermark, high quality, fast, simple. TikTok Downloader No

9.6

10,003 ms

100%



Background removal

Automatic background/foreground removal service. Without resizing.

9.4

7,162 ms

100%



Girls Nude Image

Girls Nude Sexy images API get pussy, boobs, ass, 4k, anal, gone-wild images of girl , this api can make you cum, best porn image, sexy

9.4

391 ms

100%



Toonme API

The Toonme API is a powerful tool that transforms ordinary images into fun cartoon-style images with just a few

9.4

2,056 ms

100%

7

Tools can be Very Diverse

Terminal commands

```
mark@linux-desktop: /tmp/tutorial

File Edit View Search Terminal Help

mark@linux-desktop:~$ mkdir /tmp/tutorial
mark@linux-desktop:~$ cd /tmp/tutorial
mark@linux-desktop:/tmp/tutorial$ mkdir dir1 dir2 dir3
mark@linux-desktop:/tmp/tutorial$ mkdir
mkdir: missing operand
Try 'mkdir --help' for more information.
mark@linux-desktop:/tmp/tutorial$ cd /etc ~/Desktop
bash: cd: too many arguments
mark@linux-desktop:/tmp/tutorial$ ls
dir1  dir2  dir3
mark@linux-desktop:/tmp/tutorial$
```


Tools can be Very Diverse

Actions in Games



Tools can be Very Diverse

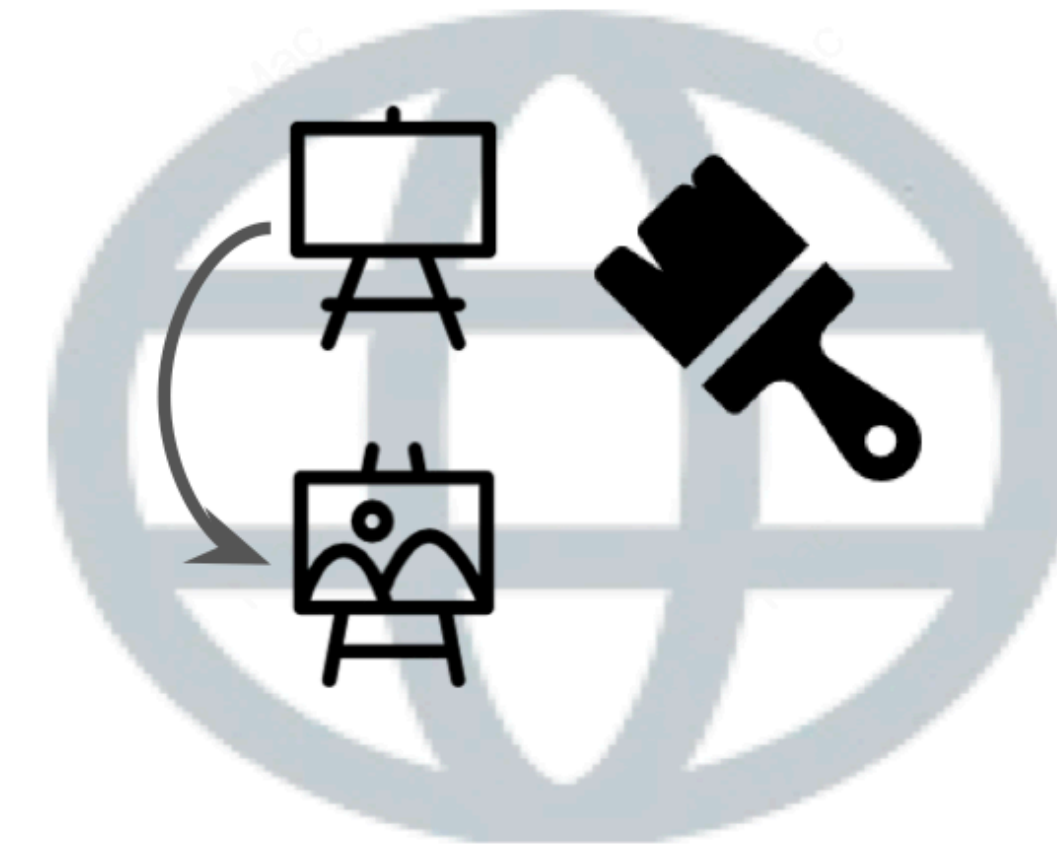
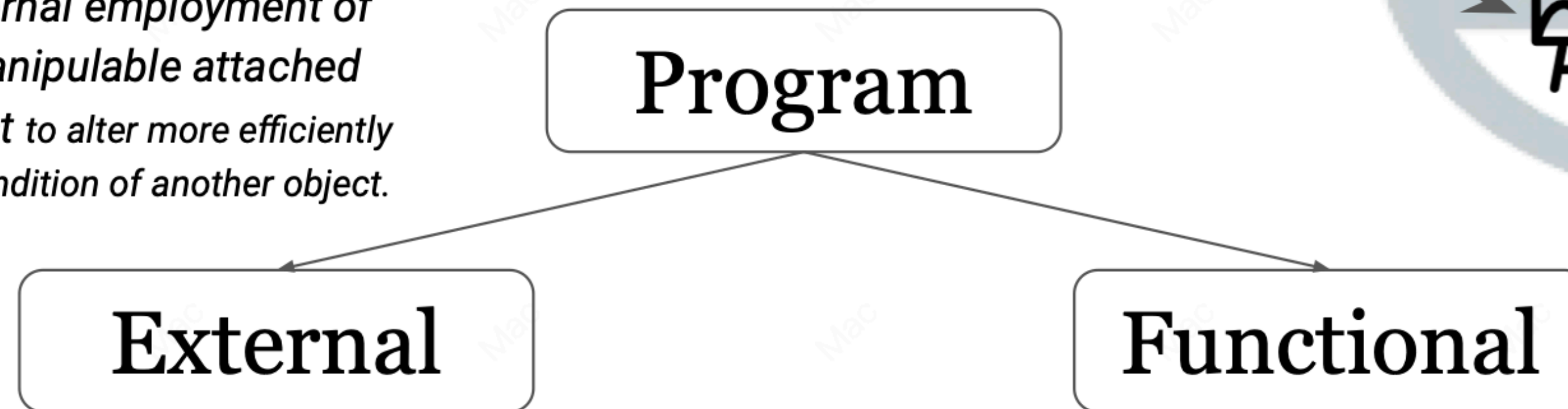
Tools can be another neural model

Call another model to perform translation, describe images, etc.

What is a Tool Anyway?

Tool Basics: Definition

[1] Animal tool: the external employment of an unattached or manipulable attached environmental object to alter more efficiently the form, position, or condition of another object.



*An LM-used tool is a **function** interface to a computer **program** that runs **external** to the LM, where the LM generates the function calls and input arguments in order to use the tool.*

[1] Shumaker et al. Animal tool behavior: the use and manufacture of tools by animals. JHU Press, 2011.

Tool Basics: Functionality

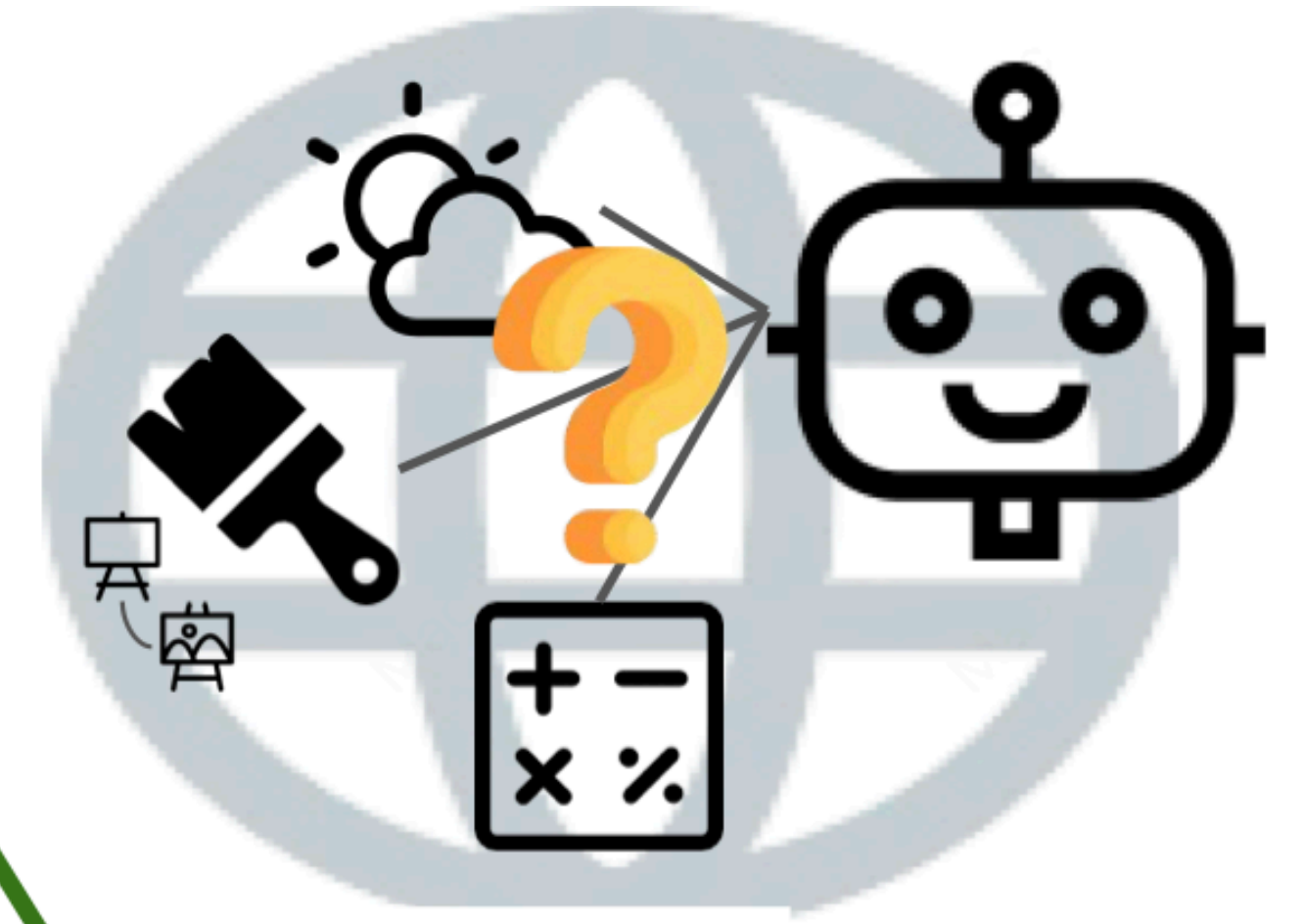
👁️ Perception: collect data from the env



Action: exert actions, change env state



Computation: general acts of computing



Tools

*Agents: anything that can be viewed as **perceiving** its environment through sensors and **acting** upon that environment through actuators^[1].*

[1] Russell, Stuart J., and Peter Norvig. *Artificial intelligence: a modern approach*. Pearson, 2016.

The Basic Tool Use Paradigm

Tool Use: switching between

- text-generation mode
- tool-execution mode

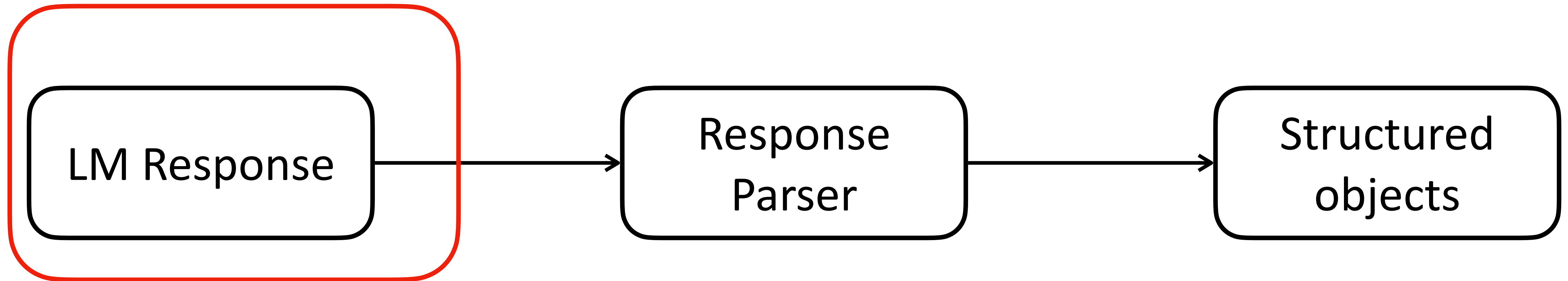
The New England Journal of Medicine is a registered trademark of [QA("Who is the publisher of The New England Journal of Medicine?") → Massachusetts Medical Society] the MMS.

Out of 1400 participants, 400 (or [Calculator(400 / 1400) → 0.29] 29%) passed the test.

The name derives from "la tortuga", the Spanish word for [MT("tortuga") → turtle] turtle.

The Brown Act is California's law [WikiSearch("Brown Act") → The Ralph M. Brown Act is an act of the California State Legislature that guarantees the public's right to attend and participate in meetings of local legislative bodies.] that requires legislative bodies, like city councils, to hold their meetings open to the public.

What is Exactly Happening



Sure, I'll check the current weather for you.

[thinking] I need real-time conditions so the user's route recommendation is accurate. [/thinking]

<tool_call>

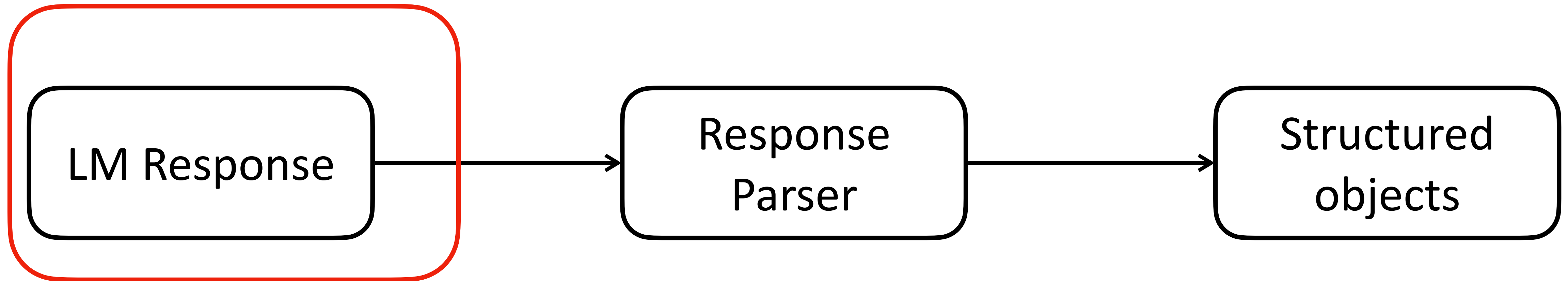
{"name": "get_weather", "arguments": {"location": "San Jose, CA, US", "date": "2025-11-07"}}

</tool_call>

Qwen3

Different Models can have different templates

What is Exactly Happening



Sure, I'll check the current weather for you.

<thinking>

I need real-time conditions so the user's route recommendation is accurate.

</thinking>

<get_weather>

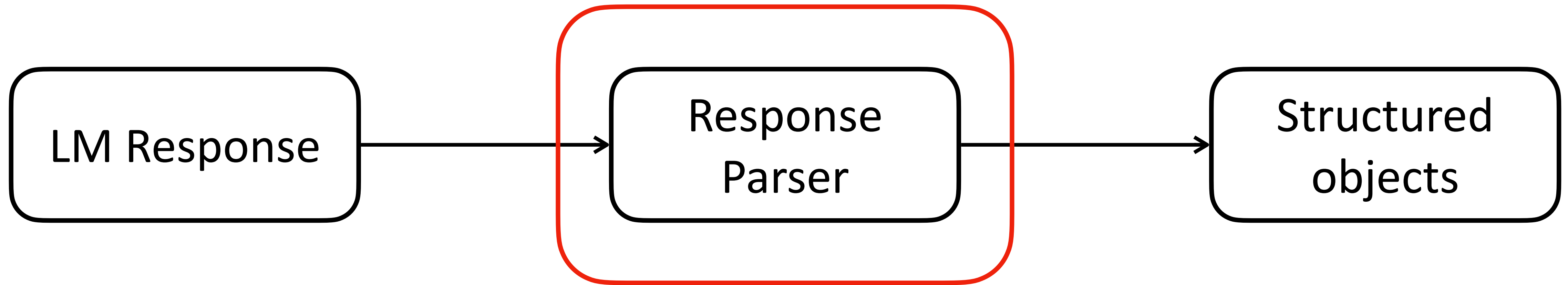
<location>San Jose, CA, US</location>

<date>2025-11-07</date>

</get_weather>

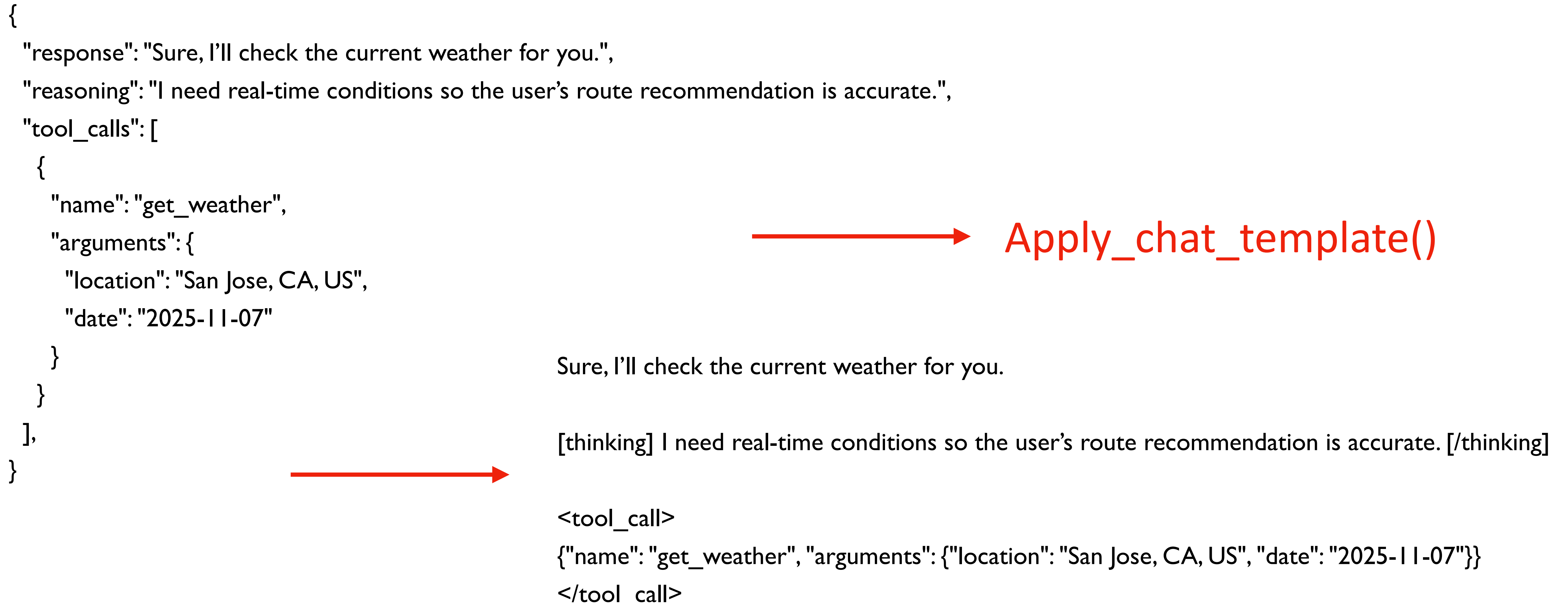
DeepSeekv3.2

What is Exactly Happening



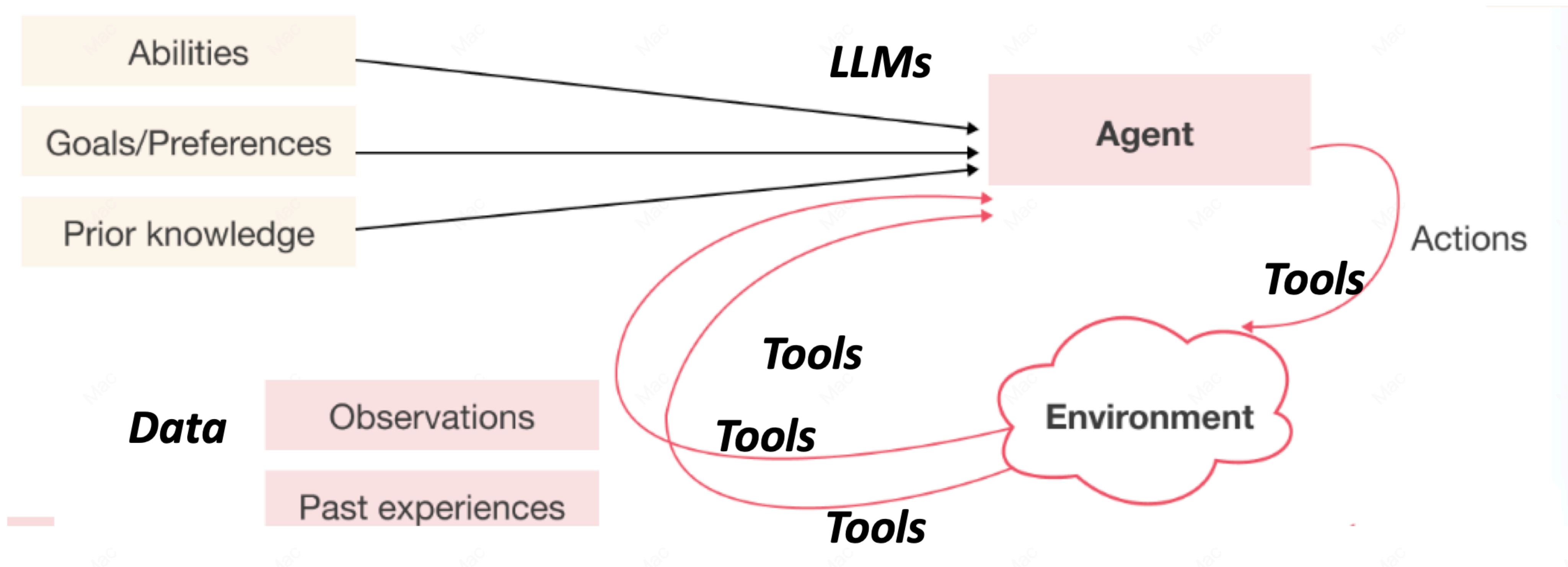
Typically a model is released along with its parser implementation

What is Exactly Happening

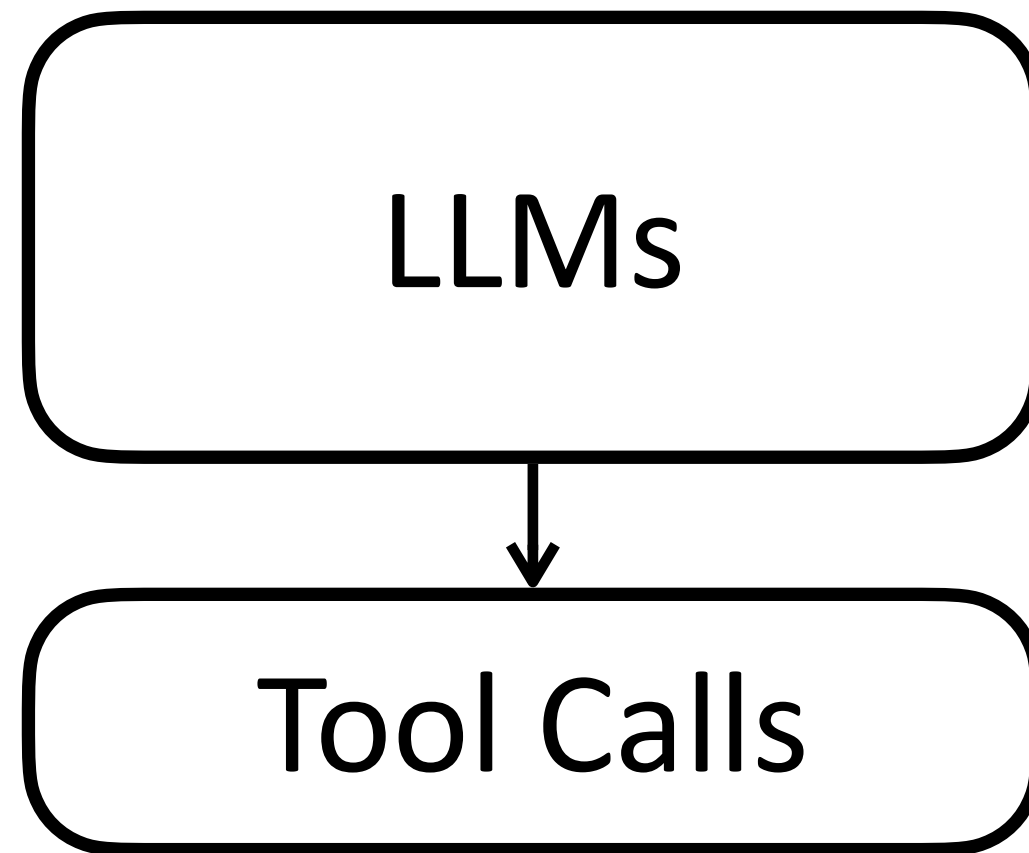


What are Agents

Anything that can be viewed as **perceiving** its environment through sensors and **acting** upon that environment through actuators.

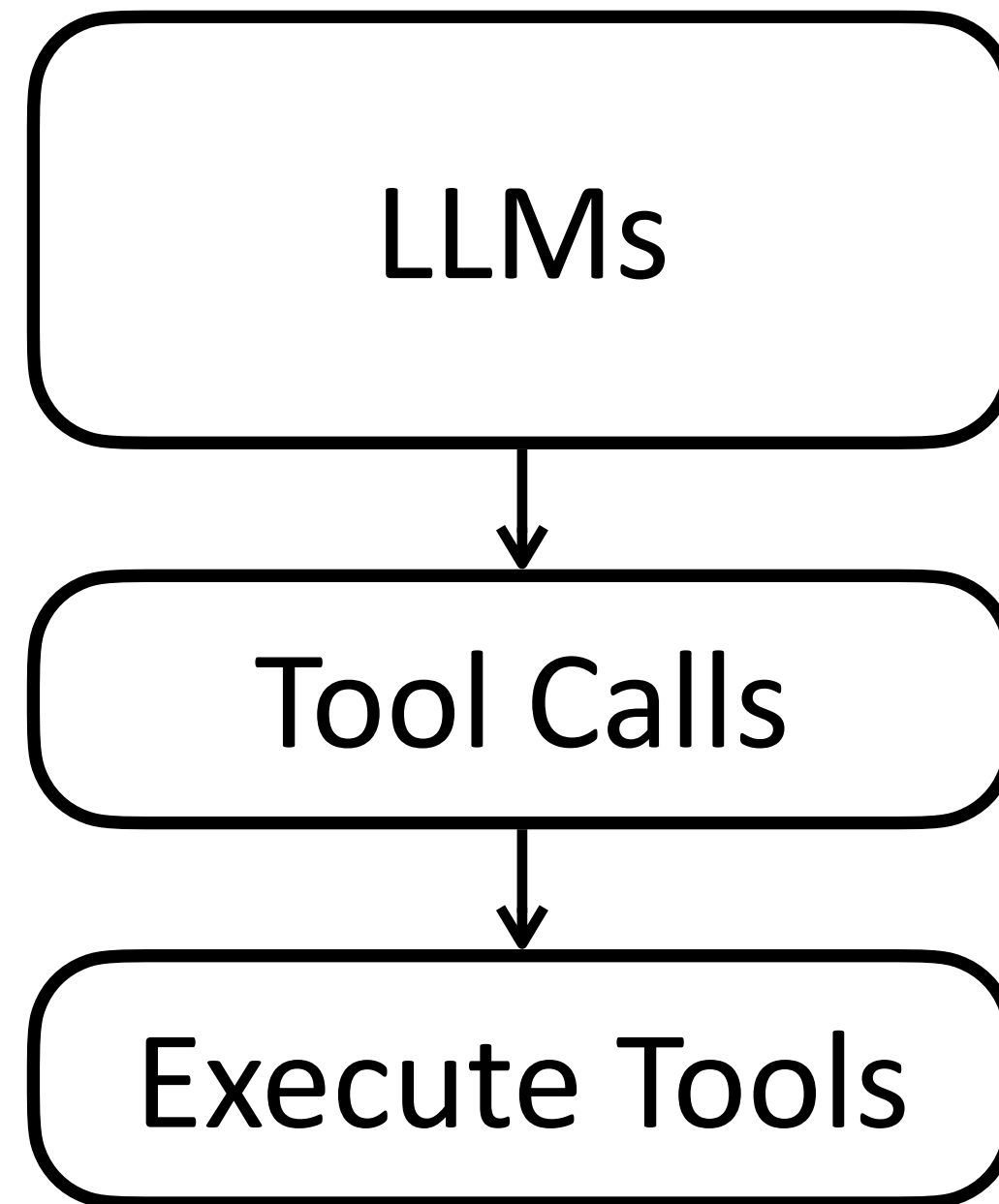


One-Step Tool Call -> Agents

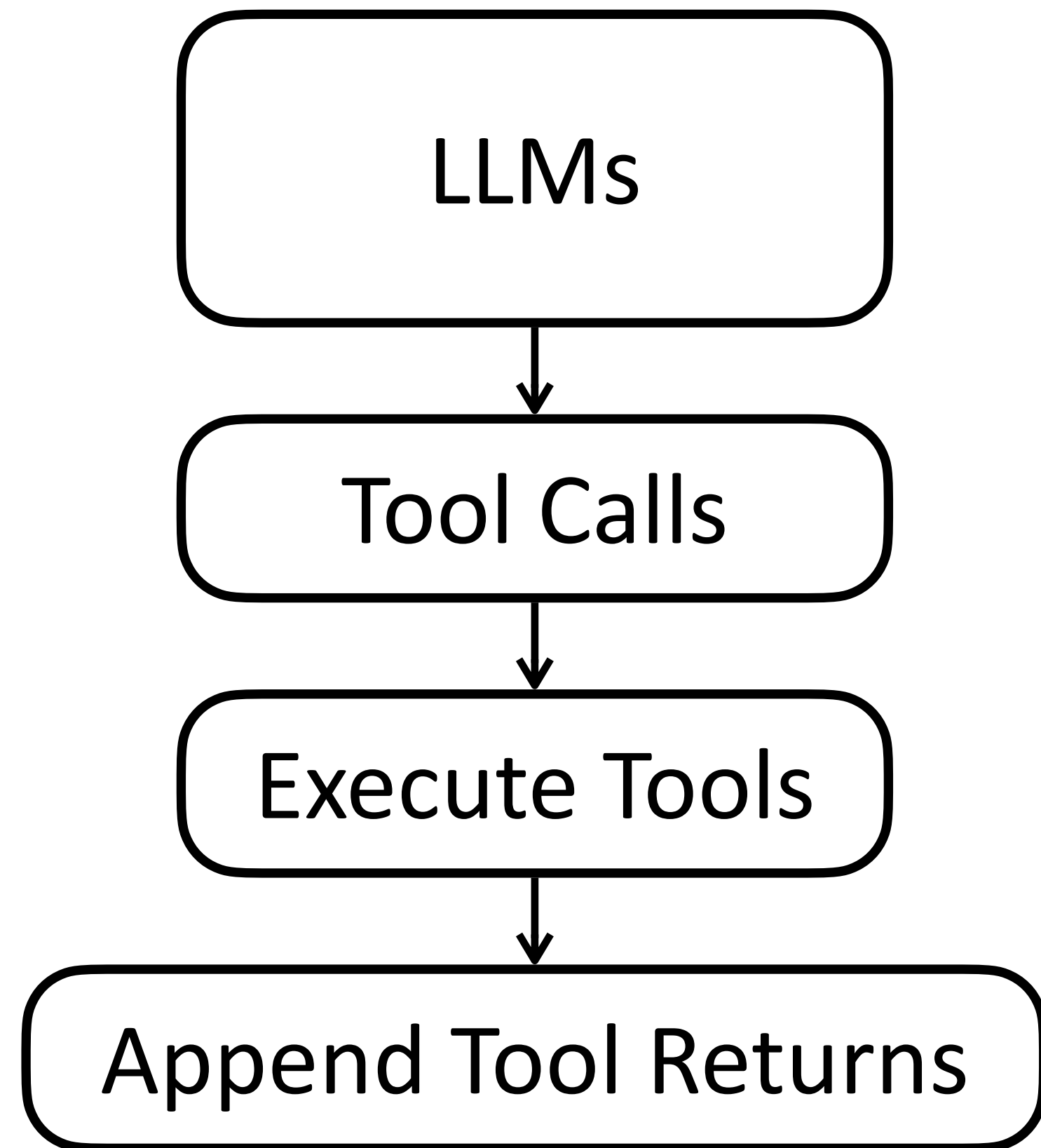


```
{
  "response": "Sure, I'll check the current weather for you.",
  "reasoning": "I need real-time conditions so the user's route recommendation is accurate.",
  "tool_calls": [
    {
      "name": "get_weather",
      "arguments": {
        "location": "San Jose, CA, US",
        "date": "2025-11-07"
      }
    }
  ],
}
```

One-Step Tool Call -> Agents

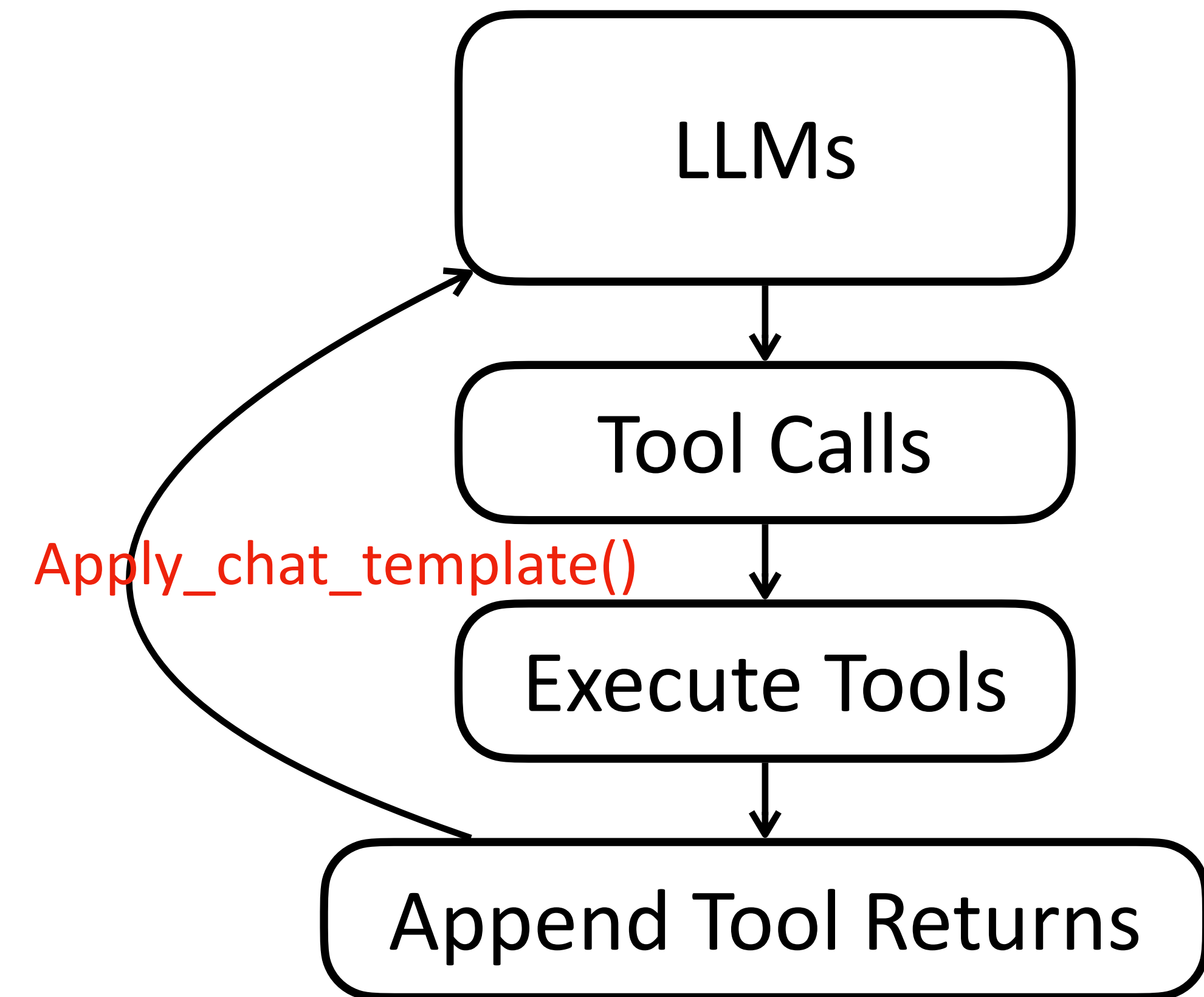


One-Step Tool Call -> Agents



```
{
  "response": "Sure, I'll check the current weather for you.",
  "reasoning": "I need real-time conditions so the user's route recommendation is accurate.",
  "tool_calls": [
    {
      "name": "get_weather",
      "arguments": {
        "location": "San Jose, CA, US",
        "date": "2025-11-07"
      }
    }
  ],
  "tool_return": {
    "temperature": 21.5,
    "condition": "clear",
    "humidity": 60,
    "wind_speed": 10,
    "location": "San Jose, CA, US",
    "date": "2025-11-07"
  }
}
```


One-Step Tool Call -> Agents



```
{
  "response": "Sure, I'll check the current weather for you.",
  "reasoning": "I need real-time conditions so the user's route recommendation is accurate.",
  "tool_calls": [
    {
      "name": "get_weather",
      "arguments": {
        "location": "San Jose, CA, US",
        "date": "2025-11-07"
      }
    }
  ],
  "tool_return": {
    "temperature": 21.5,
    "condition": "clear",
    "humidity": 60,
    "wind_speed": 10,
    "location": "San Jose, CA, US",
    "date": "2025-11-07"
  }
}
```

Sure, I'll check the current weather for you.

[thinking] I need real-time conditions so the user's route recommendation is accurate. [/thinking]

<tool_call>

{"name": "get_weather", "arguments": {"location": "San Jose, CA, US", "date": "2025-11-07"}}

</tool_call>

<tool_return>

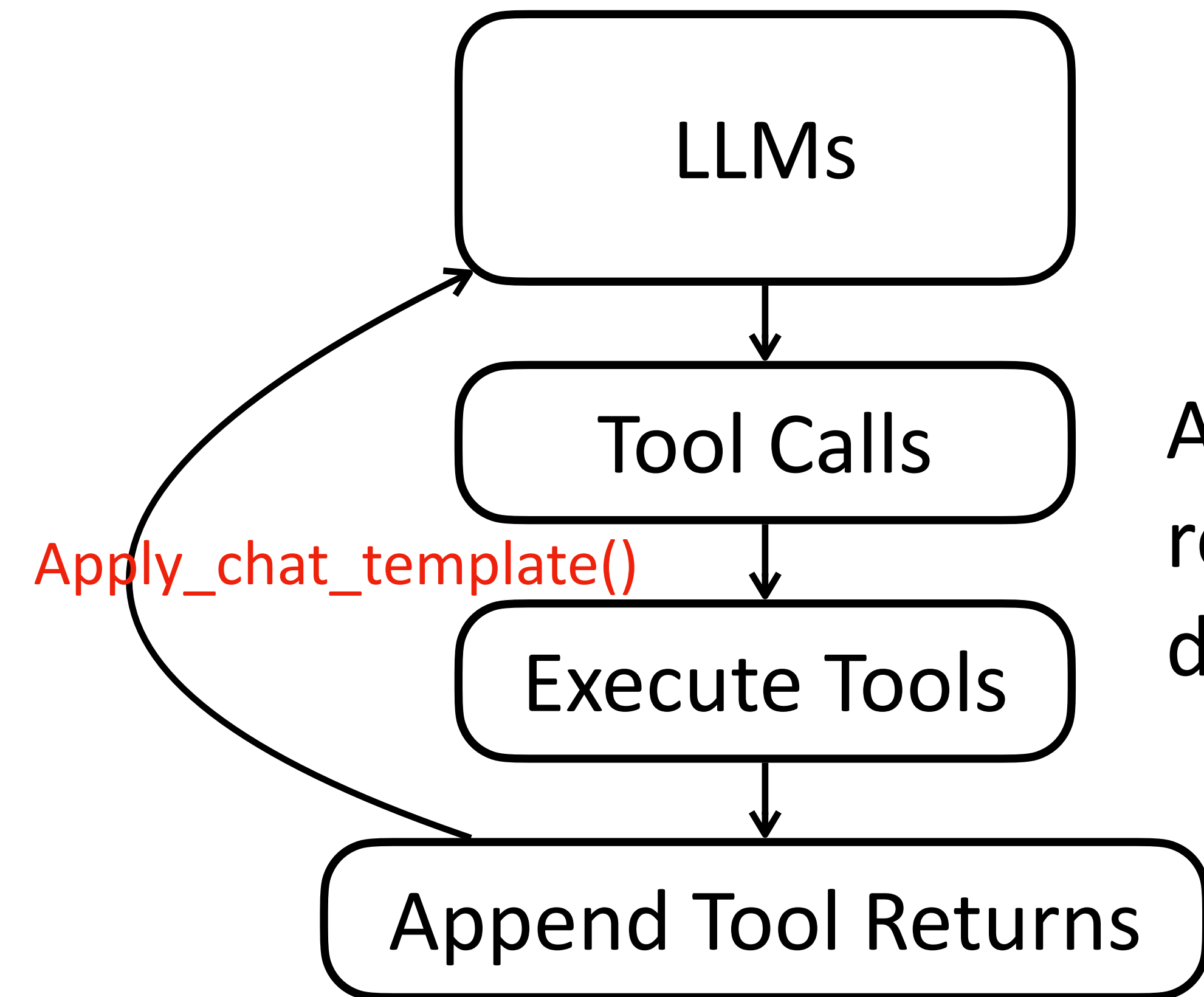
```
{  
  "temperature": 21.5,  
  "condition": "clear",  
  "humidity": 60,  
  "wind_speed": 10,  
  "location": "San Jose, CA, US",  
  "date": "2025-11-07"  
}
```

</tool_return>

This is the context fed back to the
model to continue generation

One-Step Tool Call -> Agents

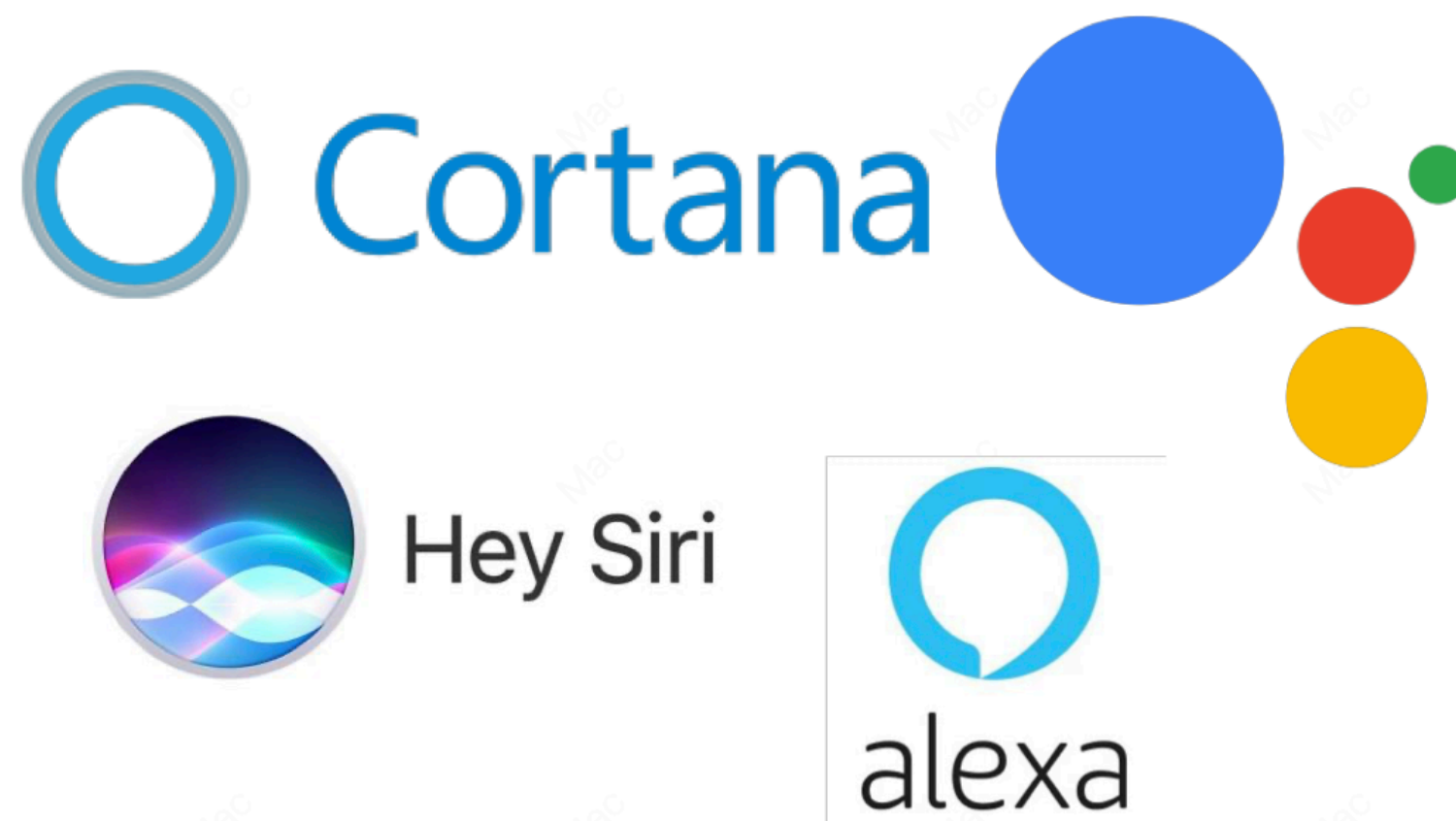
Agent Loop



Agent loop ends when the model generates a text response without tool calling, or sometimes we define a “done” tool for the model to call

Why Do We Want Agents

Imagine if things get done by just talking...



Virtual Assistants

-  *Set an alarm at 7 AM*
-  *Remind me for the meeting at 5pm*
-  *Play Jay Chou's latest album*

A screenshot of a Python IDE window titled 'Untitled-1'. The window has a purple title bar with standard macOS window controls (red, yellow, green buttons). The code editor area shows a Python script with five lines of code, numbered 1 to 5 on the left margin. Line 1: `my_list = [3, 5, 1]`. Line 2: `sort in descending order →`. Line 3: `sorted(my_list, reverse=True)`, which is highlighted with a light green background. Line 4: (empty). Line 5: (empty). The bottom status bar is purple and displays 'master*' on the left, a refresh icon in the center, and 'Python 3.6.5 64-bit' on the right, followed by a close button icon.

Natural Language Programming

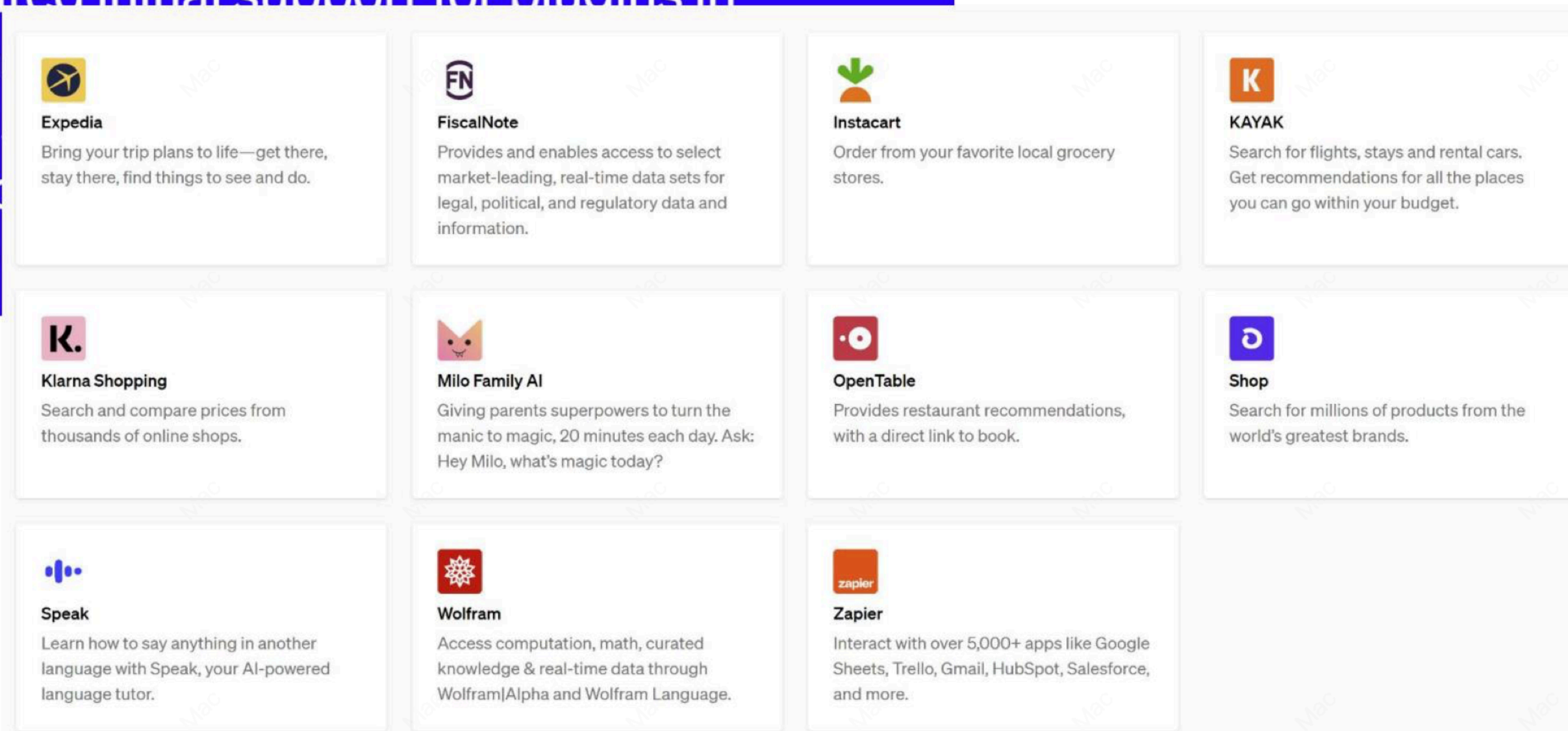
-  *Sort my_list in descending order*
-  *Copy my_file to home folder*
-  *Dump my_dict as a csv file output.csv*

Tool Integrations into ChatBots

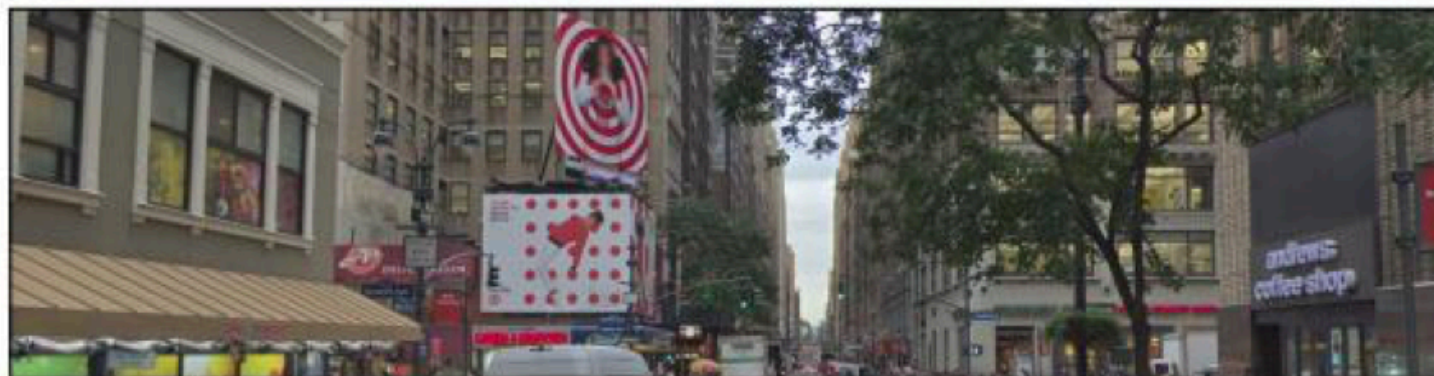
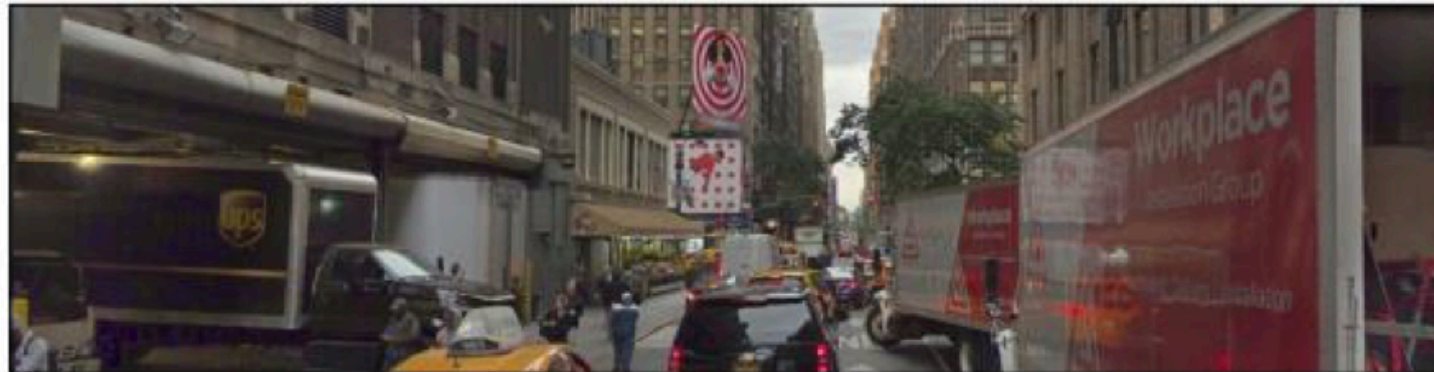
ChatGPT plugins

We've implemented initial support for plugins in ChatGPT. Plugin language models help ChatGPT access computations, or

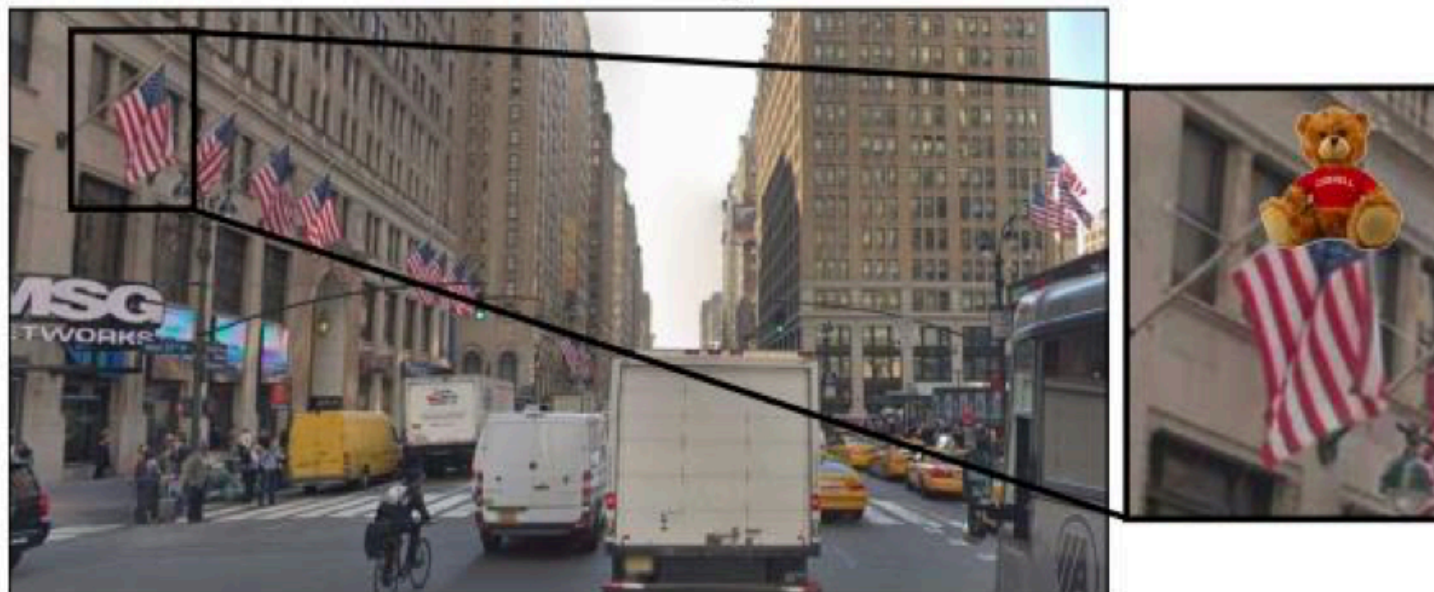
[ChatGPT plugins](#)



Robots



⋮



Turn and go with the flow of traffic. At the first traffic light turn left. Go past the next two traffic light, As you come to the third traffic light you will see a white building on your left with many American flags on it. Touchdown is sitting in the stars of the first flag.

Chen et al., 18'

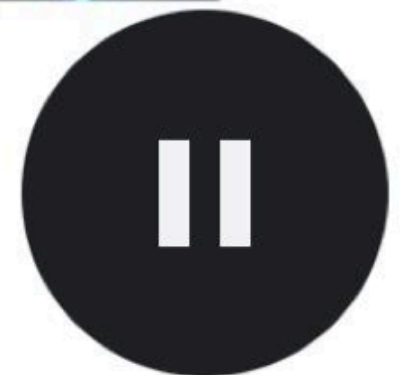
Games



No Man's Sky: Go to spaceship



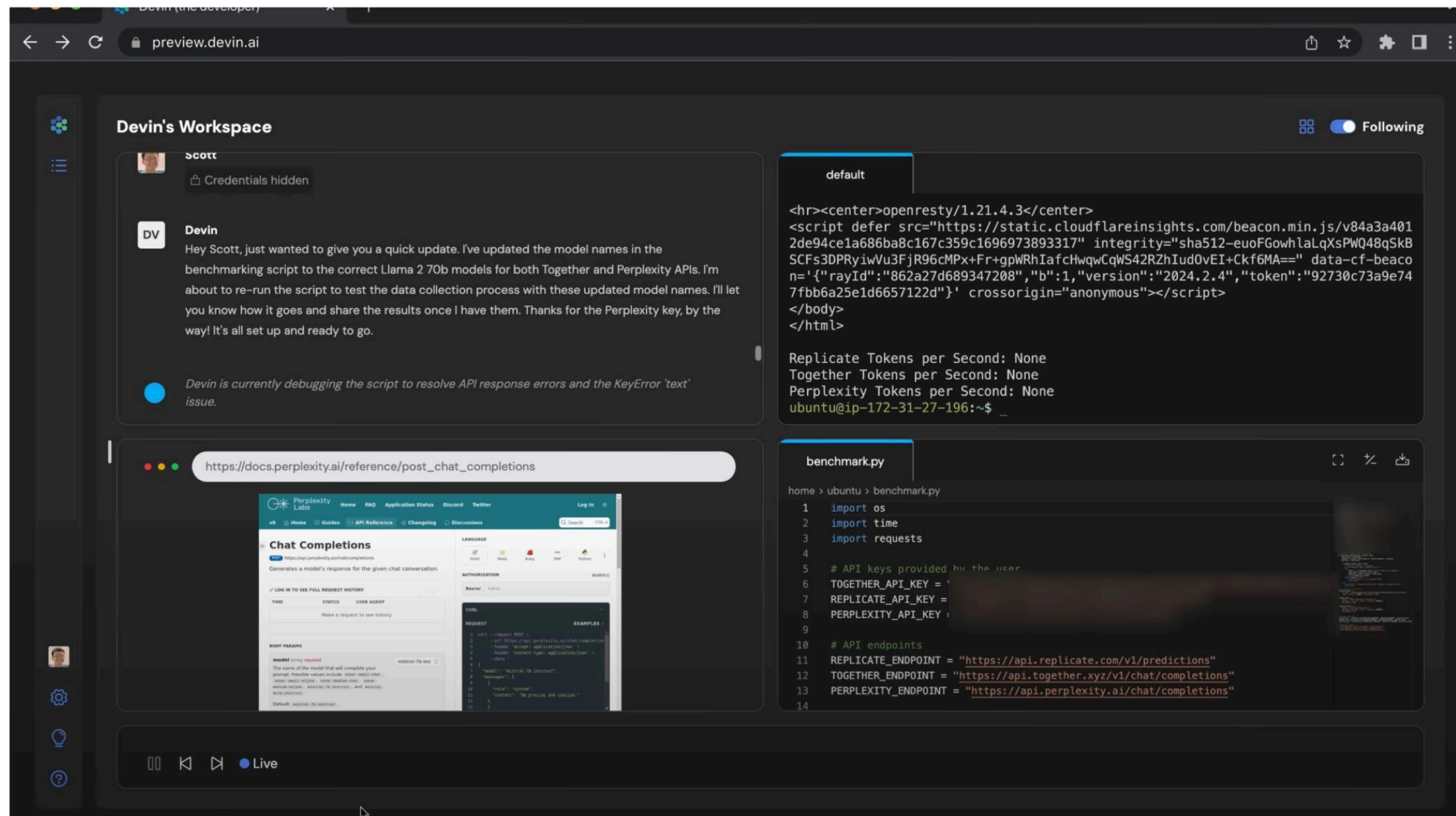
No Man's Sky: Shoot Asteroid



SIMA, DeepMind SIMA Team 24'

SoftWare Development

Some nice examples at: <https://agent.minimax.io/>



Training-free Methods for Building Agents

How to Let LLM Become an Agent

Sure, I'll check the current weather for you.

[thinking] I need real-time conditions so the user's route recommendation

<tool_call>

{"name": "get_weather", "arguments": {"location": "San Jose, CA, US", "date": "2023-10-27"}}

</tool_call>

The New England Journal of Medicine is a registered trademark of [QA("Who is the publisher of The New England Journal of Medicine?") → Massachusetts Medical Society] the MMS.

Out of 1400 participants, 400 (or [Calculator(400 / 1400) → 0.29] 29%) passed the test.

The name derives from "la tortuga", the Spanish word for [MT("tortuga") → turtle] turtle.

The Brown Act is California's law [WikiSearch("Brown Act") → The Ralph M. Brown Act is an act of the California State Legislature that guarantees the public's right to attend and participate in meetings of local legislative bodies.] that requires legislative bodies, like city councils, to hold their meetings open to the public.

We just need the LLMs to output certain formats of tool calls that we can parse

Just Prompt

Supposing you have the following APIs, answer the given question.

Available APIs:

1. weather(location): for getting the weather information given a location
2. location(): for getting the location of the user.
3. bus_route(start, end): for finding the current bus route from the start position to the end position.
4. count_characters(s): for counting the number of characters in a string s.

Question: Is it okay to go hiking today?

To know the answer, we first need to know the current location of the user.

API call: location() = Seattle.

Next, we need to know the current weather in Seattle.

API call: weather(Seattle) = Cloudy with a chance of rain.

Based on this information, it is not recommended to go hiking today in Seattle.

Available APIs:

1. weather(location): for getting the weather information given a location
2. location(): for getting the location of the user.
3. bus_route(start, end): for finding the current bus route from the start position to the end position.
4. count_characters(s): for counting the number of characters in a string s.

Question: how long does it take to take a bus to Vancouver?

To answer this question, we first need to know the current location of the user.

API call: location() = Seattle.

Next, we need to find the bus route from Seattle to Vancouver.

API call: bus_route(Seattle, Vancouver) = 4 hours.

Therefore, it takes 4 hours to take a bus from Seattle to Vancouver.

We just need the LLMs to output certain formats of tool calls that we can parse

Thank You!