Machine Programming

Lecture 9 – Agentic Frameworks and Model Context Protocol

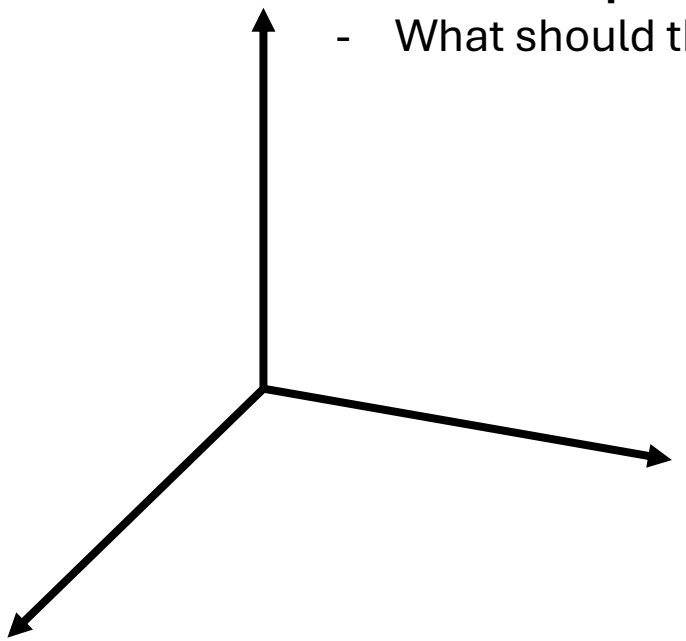
Ziyang Li

Logistics – Week 5

- Assignment 2
  - https://github.com/machine-programming/assignment-2
  - Due next Thursday (Oct 2nd)
  - Expected to take quite some time, so please start working on it early
- Feedback form
  - Follow this link to fill the questionnaire; worth 1% of extra credit
  - Thanks to those who submitted already!
  - Help us make the course better!

Dimensions in Program Synthesis

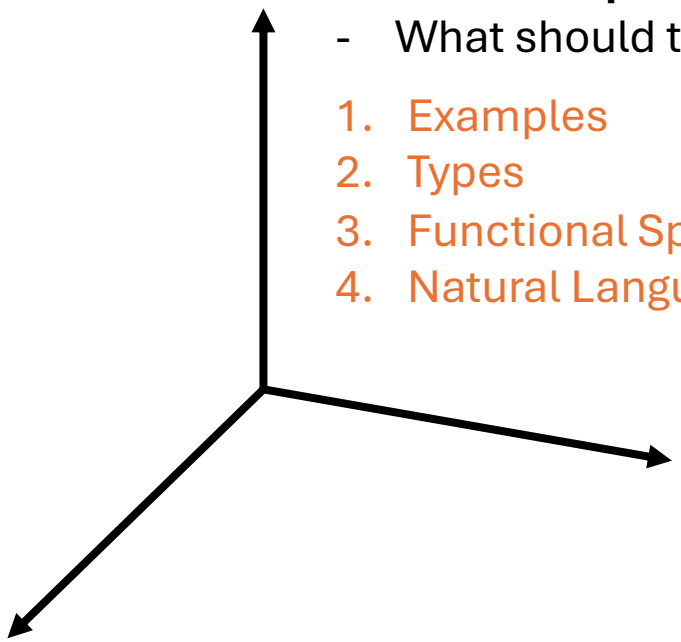
Behavioral Specification
- What should the program do?



Synthesis Strategy
- How do we find such a program?

Structural Specification
- What is the space of the programs?

The Course So Far



Behavioral Specification

- What should the program do?

1. Examples
2. Types
3. Functional Specifications
4. Natural Language

Synthesis Strategy

- How do we find such a program?

Enumeration

- Enumerating all programs with a grammar
- Bottom-up vs top-down

Data-Driven Approaches

- Next token prediction, greedy decoding, controlled decoding
- Prompting, Iterative Refinement

Structural Specification

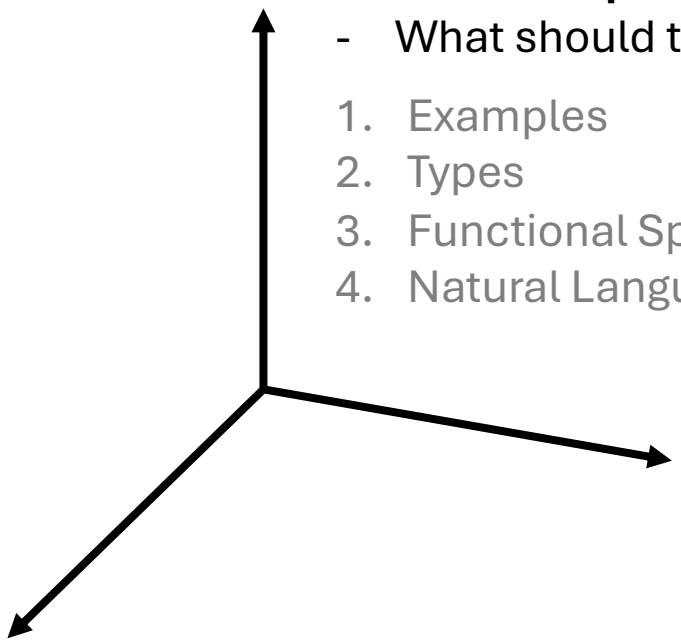
- What is the space of the programs?

General Purpose Programming Language

Python / Java / C / Rust / …

Domain Specific Languages

The Course So Far



Behavioral Specification
- What should the program do?

1. Examples
2. Types
3. Functional Specifications
4. Natural Language

Synthesis Strategy
- How do we find such a program?

Enumeration
- Enumerating all programs with a grammar
- Bottom-up vs top-down

Data-Driven Approaches
- Next token prediction, greedy decoding, controlled decoding
- Prompting, Iterative Refinement

Structural Specification
- What is the space of the programs?

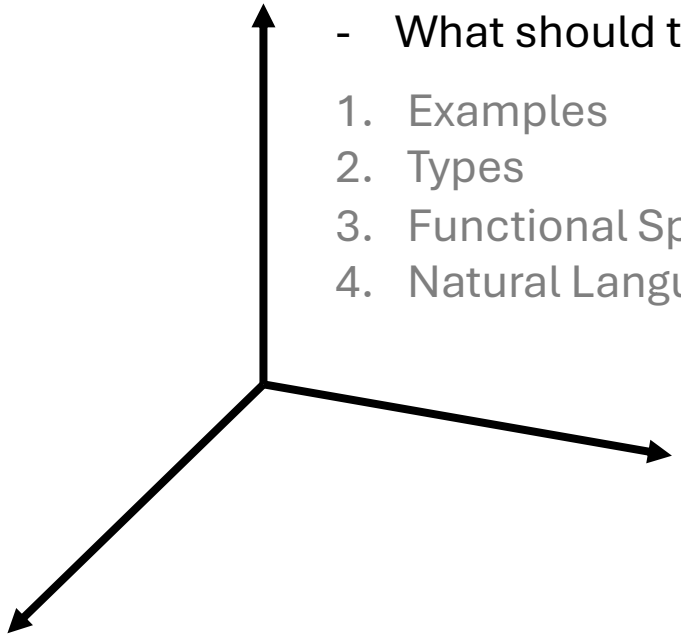
General Purpose Programming Language
Python / Java / C / Rust / …

Domain Specific Languages

Today

Behavioral Specification
- What should the program do?

1. Examples
2. Types
3. Functional Specifications
4. Natural Language



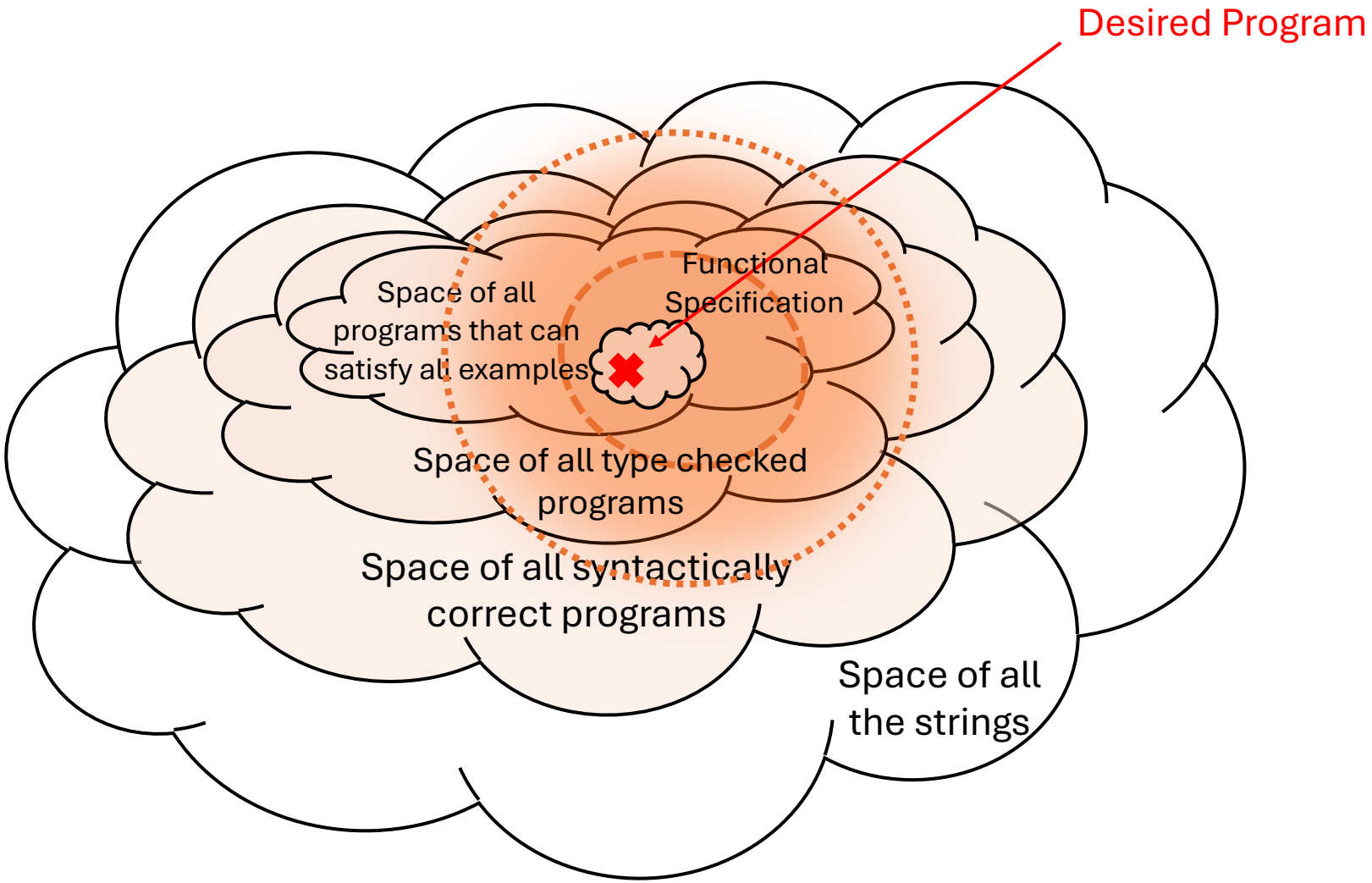
Structural Specification
- What is the space of the programs?

General Purpose Programming Language
Python / Java / C / Rust / …

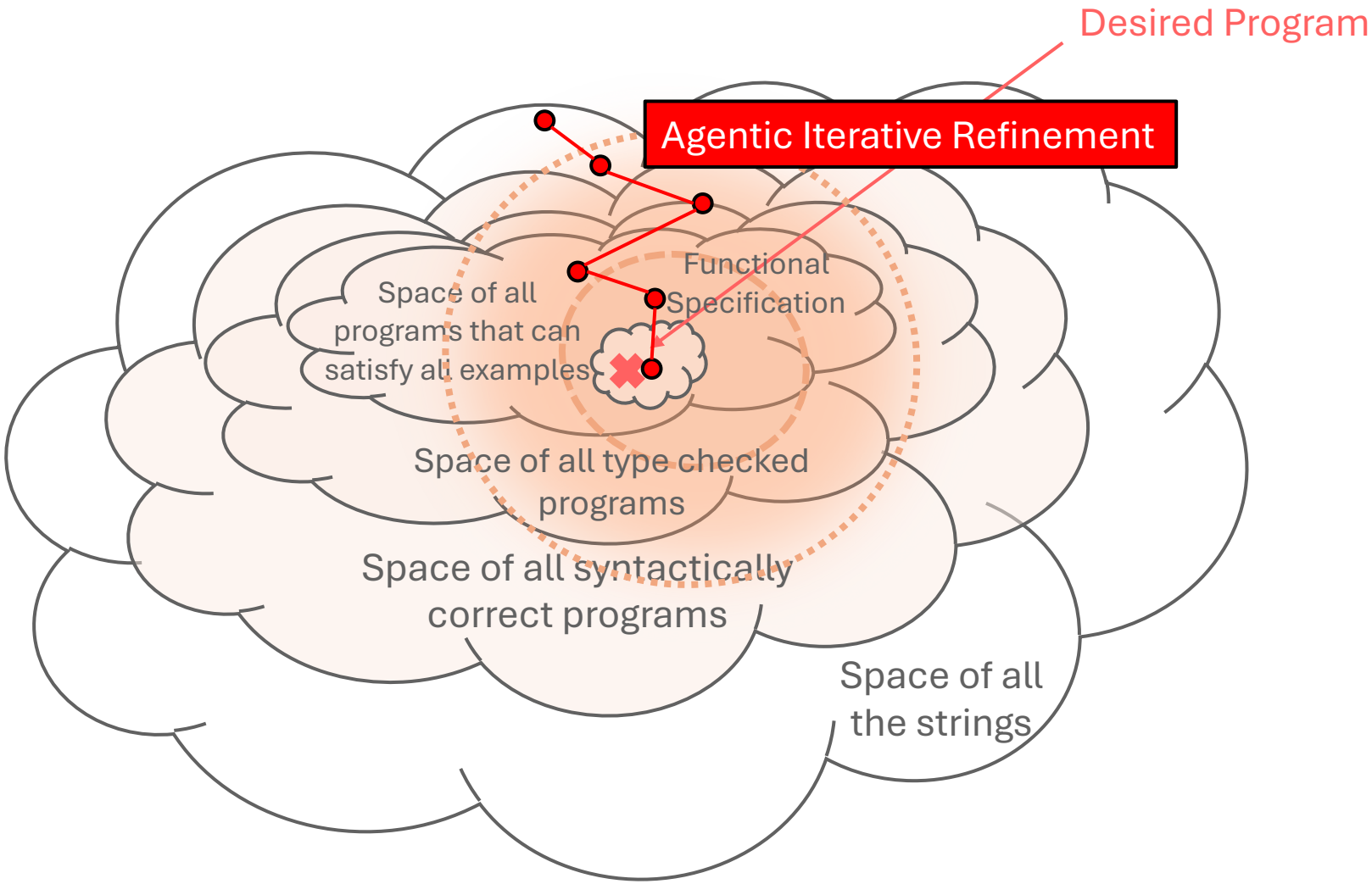
Synthesis Strategy
- How do we find such a program?

Agentic Frameworks
- Tool Use, MCP

High Level Picture

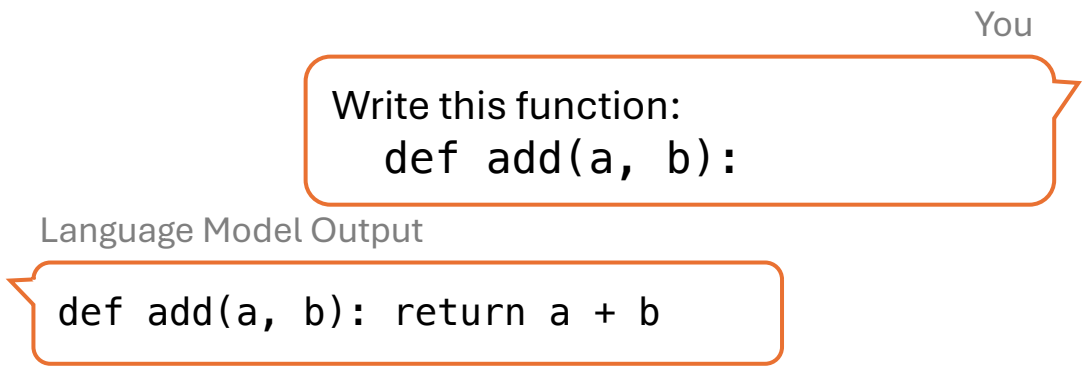


High Level Picture



High Level Picture

Single-step Prompting



Prompting with a Conversation

```
f"""
Write this function:
  def add(a, b):
Output:
  def add(a, b): return a + b

Write this function:
  def reverse(s: str) -> str:
Output:
  def reverse(s: str) -> str:
    output = ""
    for c in s:
      output = c + output
    return output

Write this function:
  def is_palindrome(s: str) -> bool:
"""
```

You

Write this function:

```
def add(a, b):
```

Faked Language Model Output

```
def add(a, b): return a + b
```

You

Write this function:

```
def reverse(s: str)…
```

Faked Language Model Output

```
def reverse(s: str) -> str:
  output = ""
  for c in s:
    output = c + output
  return output
```

You

Write this function:

```
def is_palindrome(s: str)
```

Prompting with a Conversation

You

Write this function:

```
def add(a, b):
```

Faked Language Model Output

```
def add(a, b): return a + b
```

You

Write this function:

```
def reverse(s: str)…
```

Faked Language Model Output

```
def reverse(s: str) -> str:
  output = ""
  for c in s:
    output = c + output
  return output
```

You

Write this function:

```
def is_palindrome(s: str)
```

➔

```
{
  "messages": [
    { "role": "user",
      ”parts": ["Write this function:\ndef add(a, b):”] },
    { "role": "assistant",
      ”parts": ["def add(a, b): return a + b”] },
    { "role": "user",
      ”parts": ["Write this function:\ndef reverse(s: str)…”] },
    { "role": "assistant",
      ”parts": ["def reverse(s: str) -> str:\n output = \"\"\n
                  for c in s:\n output = c + output\n
                  return output”] },
    { "role": "user",
      ”parts": ["Write this function:\n
                  def is_palindrome(s: str)”] }
  ]
}
```

Prompting with a Conversation

System

You are a senior software engineer from a top-tier company...

You

Write this function:

```
def add(a, b):
```

Language Model

```
def add(a, b): return a + b
```

...

You

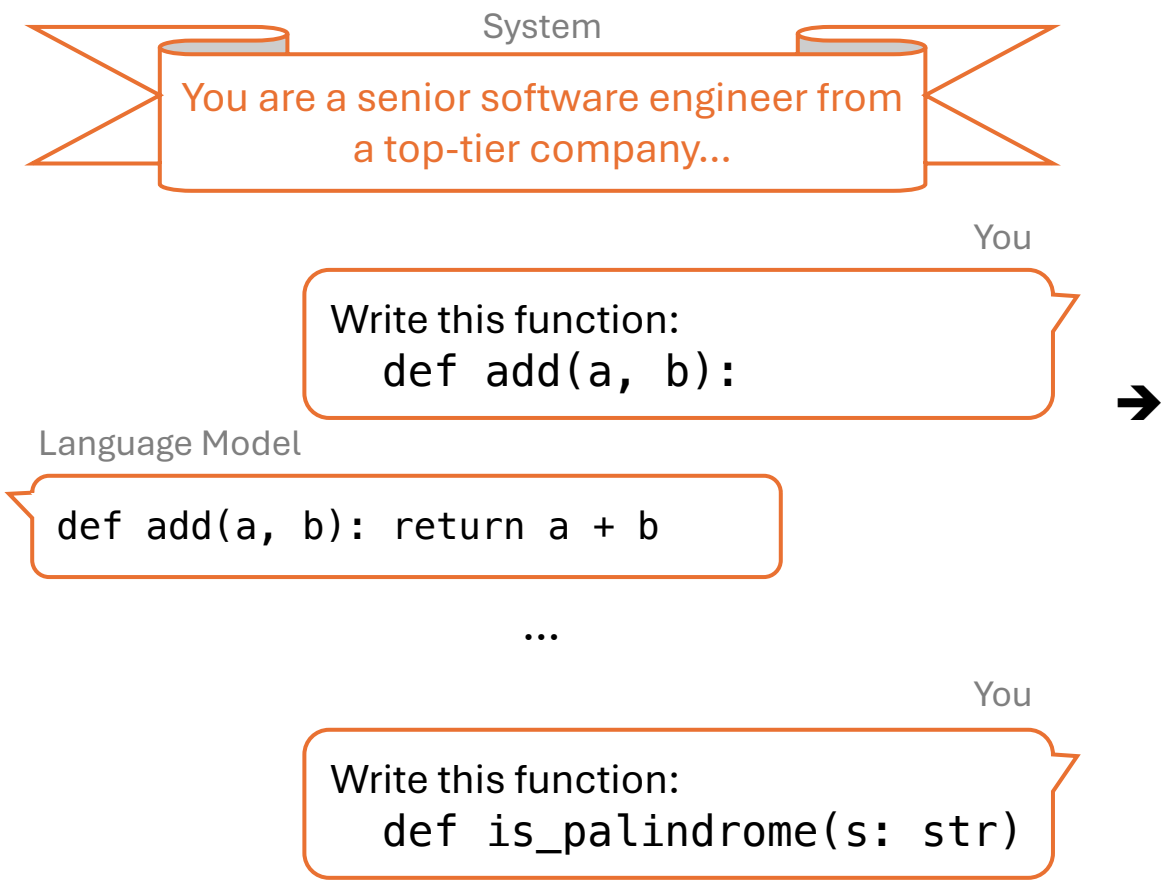
Write this function:

```
def is_palindrome(s: str)
```

➔

```
{
  "messages": [
    { "role": ”system",
      ”parts": [”You are a senior software engineer from…”] },
    { "role": "user",
      ”parts": ["Write this function:\ndef add(a, b):”] },
    { "role": "assistant",
      ”parts": ["def add(a, b): return a + b”] },
    { "role": "user",
      ”parts": ["Write this function:\ndef reverse(s: str)…”] },
    { "role": "assistant",
      ”parts": ["def reverse(s: str) -> str:\n output = \"\"\n
                  for c in s:\n output = c + output\n
                  return output”] },
    { "role": "user",
      ”parts": ["Write this function:\n
                  def is_palindrome(s: str)”] }
  ]
}
```

Prompting with a Conversation



➔

```
{
  "messages": [
    { "role": ”system",
      ”parts": [”You are a senior software engineer from…”] },
    { "role": "user",
      ”parts": ["Write this function:\ndef add(a, b):”] },
    { "role": "assistant",
      ”parts": ["def add(a, b): return a + b”] },
    { "role": "user",
      ”parts": ["Write this function:\ndef reverse(s: str)…”] },
    { "role": "assistant",
      ”parts": ["def reverse(s: str) -> str:\n output = \"\"\n
                  for c in s:\n output = c + output\n
                  return output”] },
    { "role": "user",
      ”parts": ["Write this function:\n
                  def is_palindrome(s: str)”] }
  ]
}
```

```
client.chat.completions.create(
  model="gpt-4o-mini",
  messages=messages
)
```

Prompting with a Conversation: Behind the Scene

Special Tokens denoting Roles

```
{
  "messages": [
    { "role": ”system",
      ”parts": [”You are a senior software engineer from…”] },
    { "role": "user",
      ”parts": ["Write this function:\ndef add(a, b):”] },
    { "role": "assistant",
      ”parts": ["def add(a, b): return a + b”] },
    { "role": "user",
      ”parts": ["Write this function:\ndef reverse(s: str)…”] },
    { "role": "assistant",
      ”parts": ["def reverse(s: str) -> str:\n output = \"\"\n
                for c in s:\n output = c + output\n
                return output”] },
    { "role": "user",
      ”parts": ["Write this function:\n
                def is_palindrome(s: str)”] }
  ]
}
```

➔

```
<system>
You are a senior software engineer from…
<user>
Write this function:
  def add(a, b):
<assistant>
def add(a, b): return a + b
<user>
Write this function:
  def reverse(s: str)…
<assistant>
def reverse(s: str) -> str:
  output = ””
  for c in s:
    output = c + output
  return output
<user>
Write this function:
  def is_palindrome(s: str)
```

Chain-of-Thought Prompting

You

You are tasked to write this function:

```
def min_cost_k_perio…(x, …):
```

Before writing it, please first think step-by-step. analyze what needs to be done and sketch out a piece of pseudo code…

Language Model (Actual Generation)

I will write the function `min_cost_k_periodic_segmentation` to the specified requirements. The problem appears to be a dynamic programming challenge.

You

Good. Please go ahead and write the function.

Language Model (Actual Generation)

```
def min_cost_k_perio…(x, …):
  …
```

Chain-of-Thought Prompting

You

You are tasked to write this function:
`def min_cost_k_perio…(x, …):`
Before writing it, please first think step-by-step. analyze what needs to be done and sketch out a piece of pseudo code…

Language Model (Actual Generation)

I will write the function `min_cost_k_periodic_segmentation` to the specified requirements. The problem appears to be a dynamic programming challenge.

You

Good. Please go ahead and write the function.

Language Model (Actual Generation)

```
def min_cost_k_perio…(x, …):
  …
```

➔

```
<system>
You are a senior software engineer from…
<user>
You are tasked to write this function:
  def min_cost_k_perio…(x, …):
Before writing it, please first think step-by-step. analyze what needs to be done and sketch out a piece of pseudo code…
<assistant>
I will write the function min_cost_k_periodic_segmentation to the specified requirements. The problem appears to be a dynamic programming challenge.
<user>
Good. Please go ahead and write the function.
```

Self-Reflection Prompting

You

You are tasked to write this function:
```
def min_cost_k_perio…(x, …):
```
Please write the function according to the specification and the input/output examples.

Language Model (Actual Generation)

```
def min_cost_k_perio…(x, …): …
```

You

Please now reflect what you have written. Try simulate what would happen if we pass in an example and see whether the result is expected.

Language Model (Actual Generation)

I will analyze the function that I just wrote…

You

Now please use your analysis to correct the function that you wrote; if nothing wrong, just copy and paste the function you had generated.

Self-Reflection Prompting

You

You are tasked to write this function:
`def min_cost_k_perio…(x, …):`
Please write the function according to the specification and the input/output examples.

Language Model (Actual Generation)

`def min_cost_k_perio…(x, …): …`

You

Please now reflect what you have written. Try simulate what would happen if we pass in an example and see whether the result is expected.

Language Model (Actual Generation)

I will analyze the function that I just wrote…

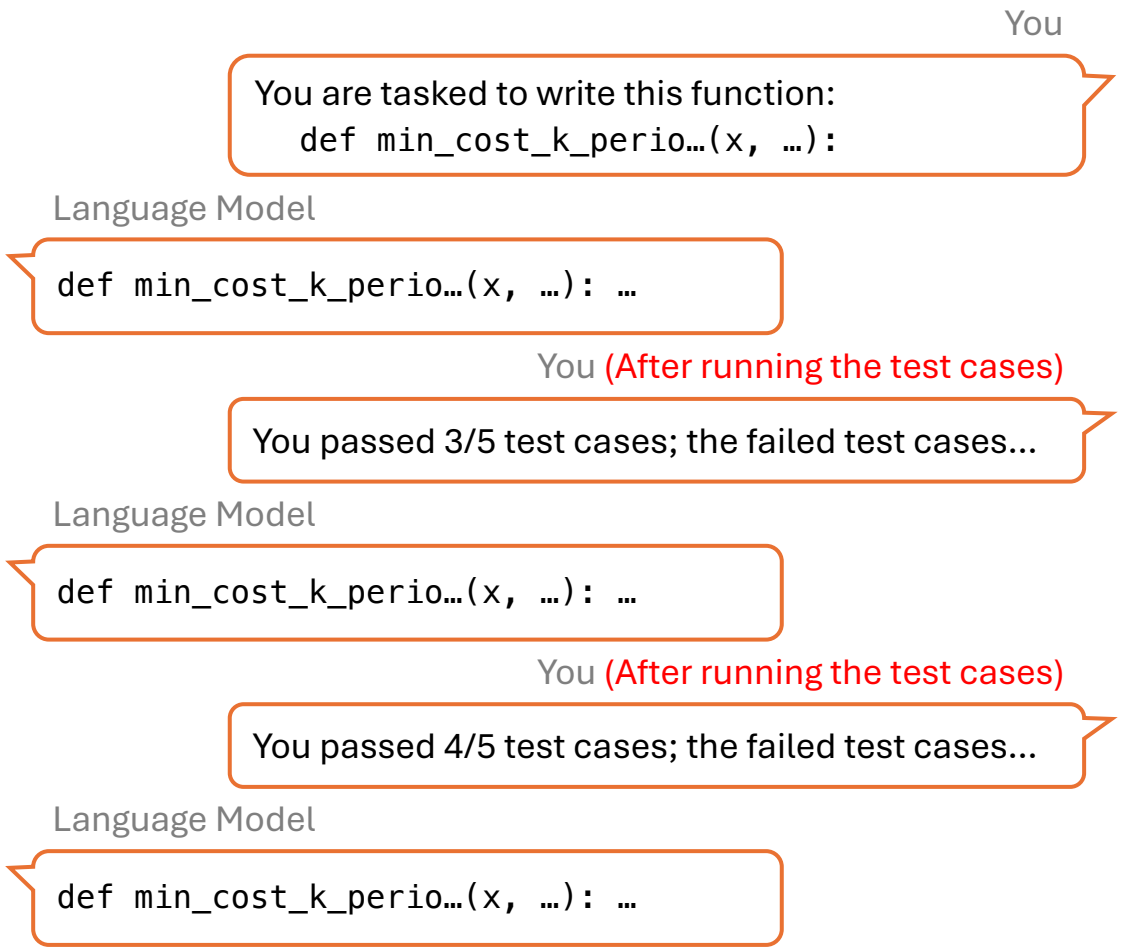
You

Now please use your analysis to correct the function that you wrote; if nothing wrong, just copy and paste the function you had generated.

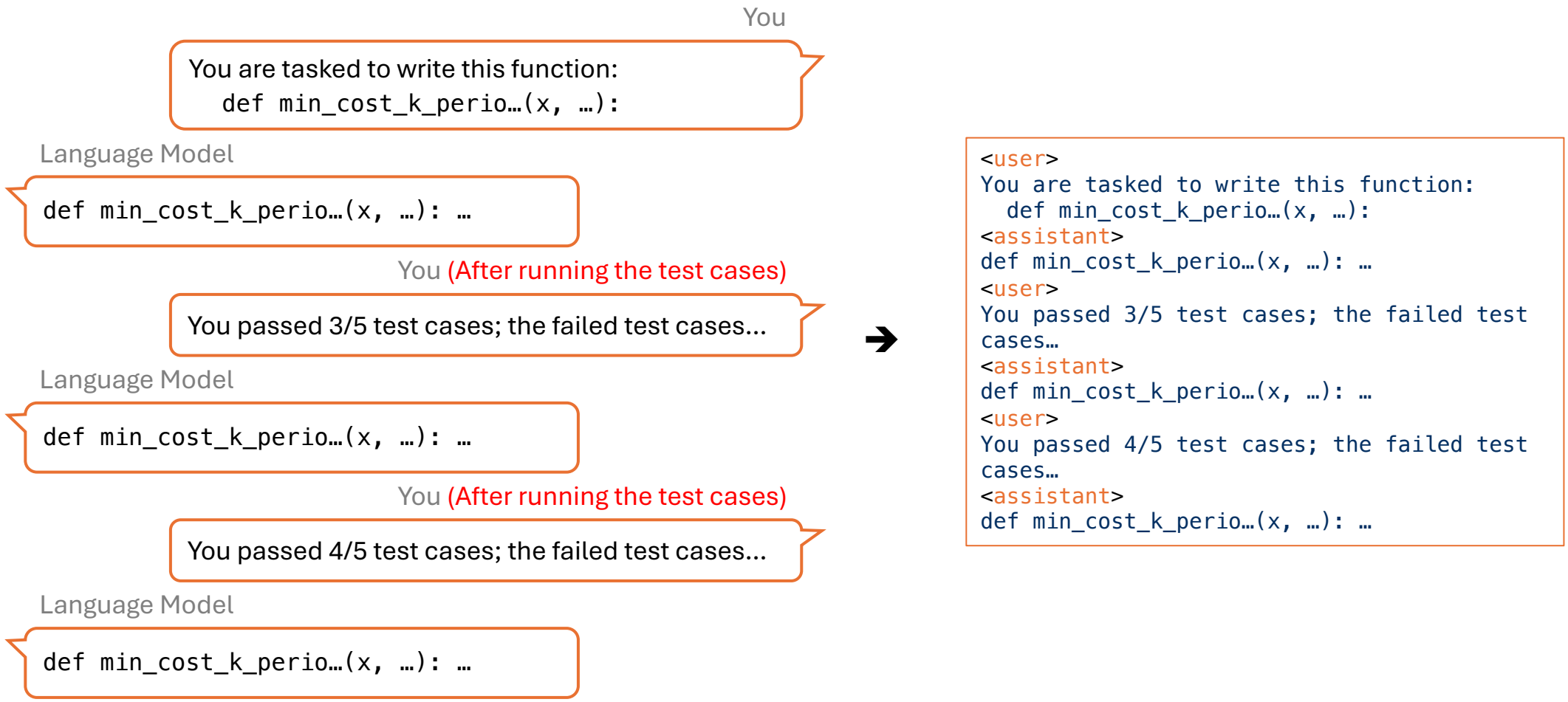
➔

```
<user>
You are tasked to write this function:
  def min_cost_k_perio…(x, …):
Please write the function according to the
specification and the input/output
examples.
<assistant>
def min_cost_k_perio…(x, …): …
<user>
Please now reflect what you have written.
Try simulate what would happen if we pass
in an example and see whether the result is
expected.
<assistant>
I will analyze the function that I just
wrote…
<user>
Now please use your analysis to correct the
function that you wrote; if nothing wrong,
just copy and paste the function you had
generated.
<assistant>
...
```

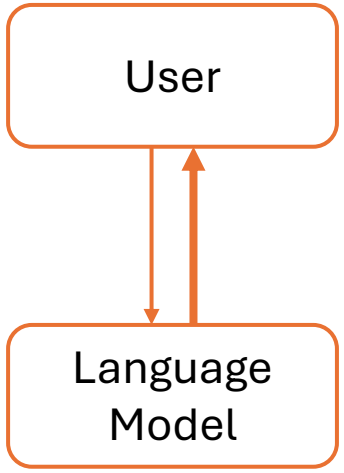
Iterative Refinement



Iterative Refinement

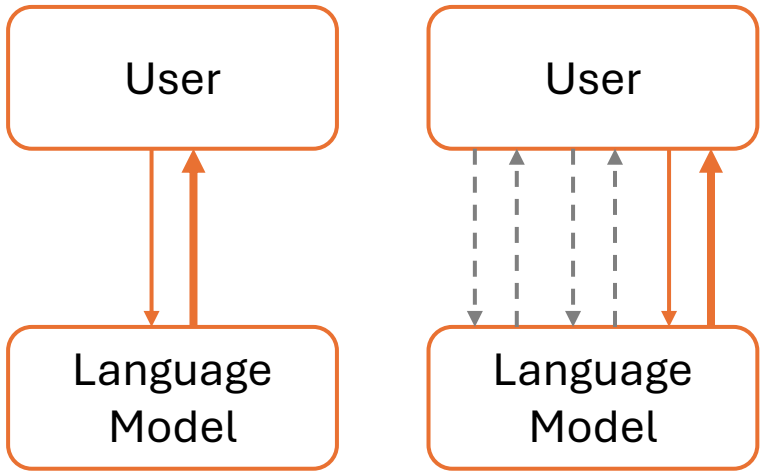


Prompting Pipelines



Zero-shot Prompting

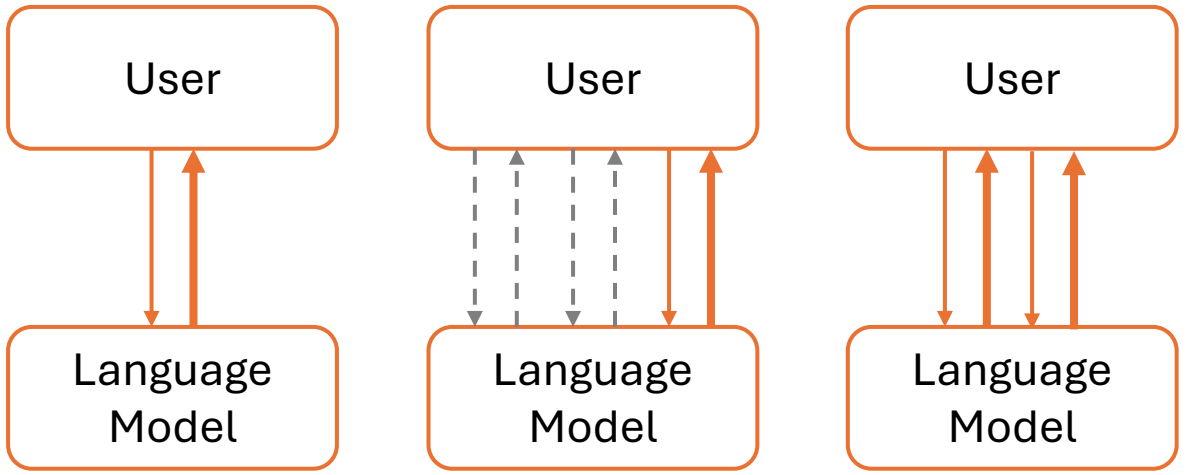
Prompting Pipelines



Zero-shot Prompting

Few-shot Prompting

Prompting Pipelines

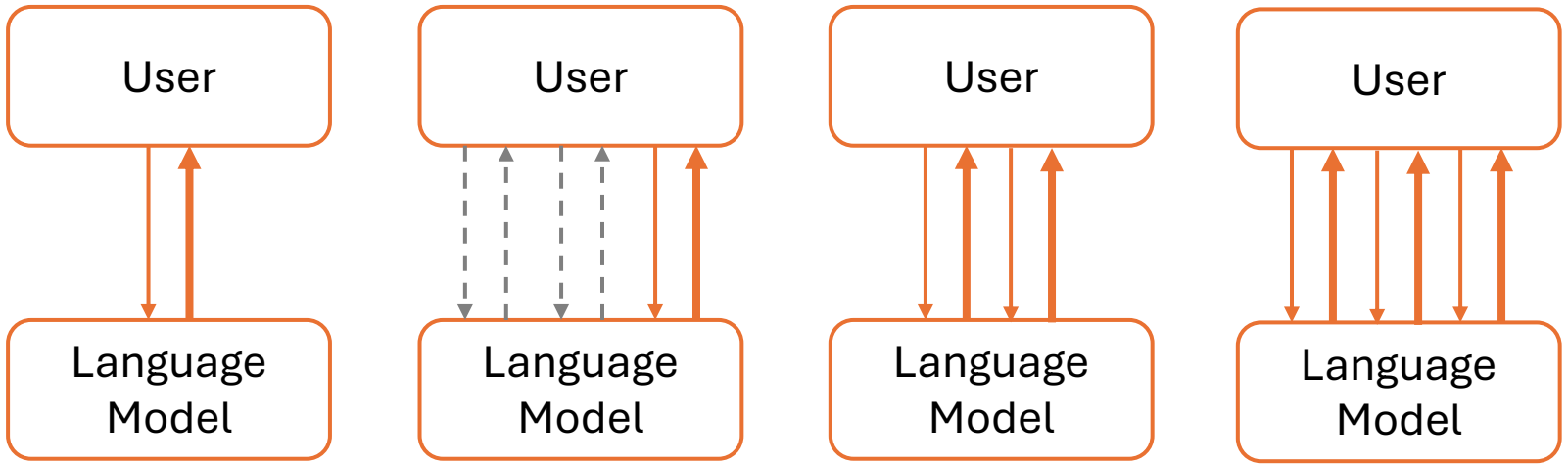


Zero-shot Prompting

Few-shot Prompting

Chain-of-thought Prompting

Prompting Pipelines



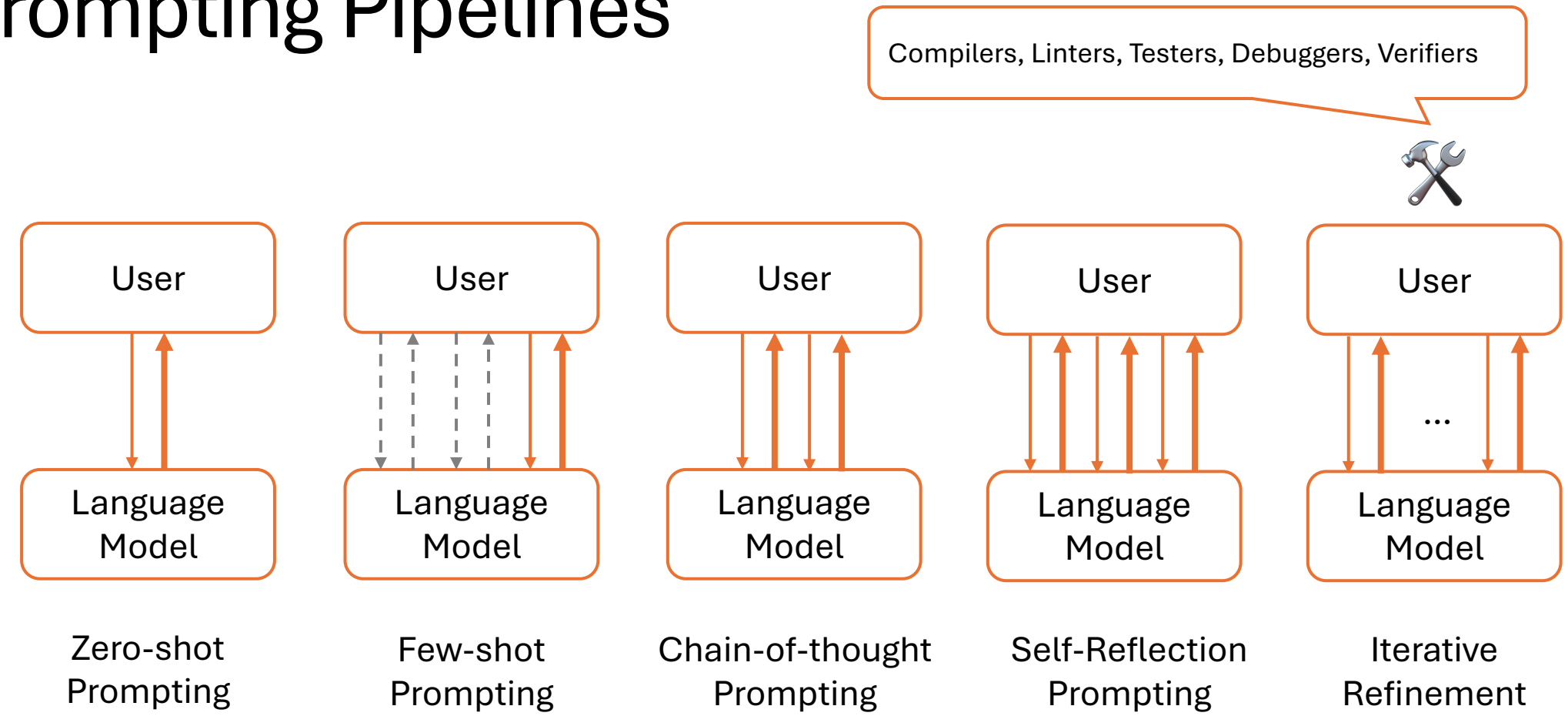
Zero-shot Prompting

Few-shot Prompting

Chain-of-thought Prompting

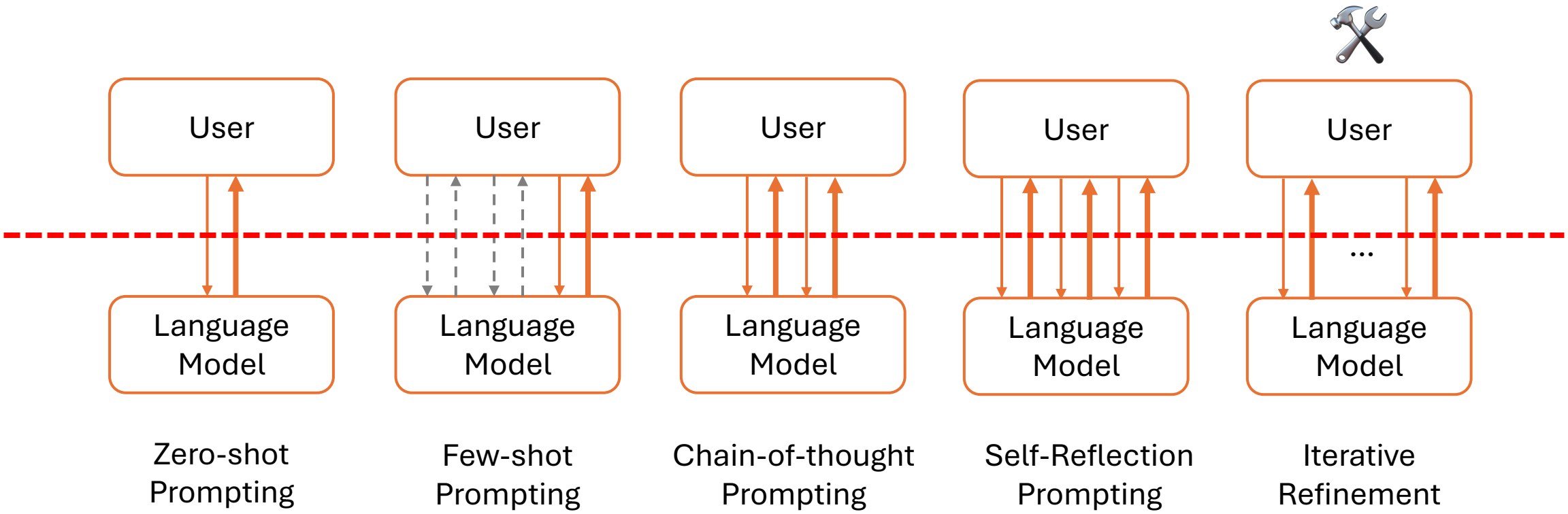
Self-Reflection Prompting

Prompting Pipelines



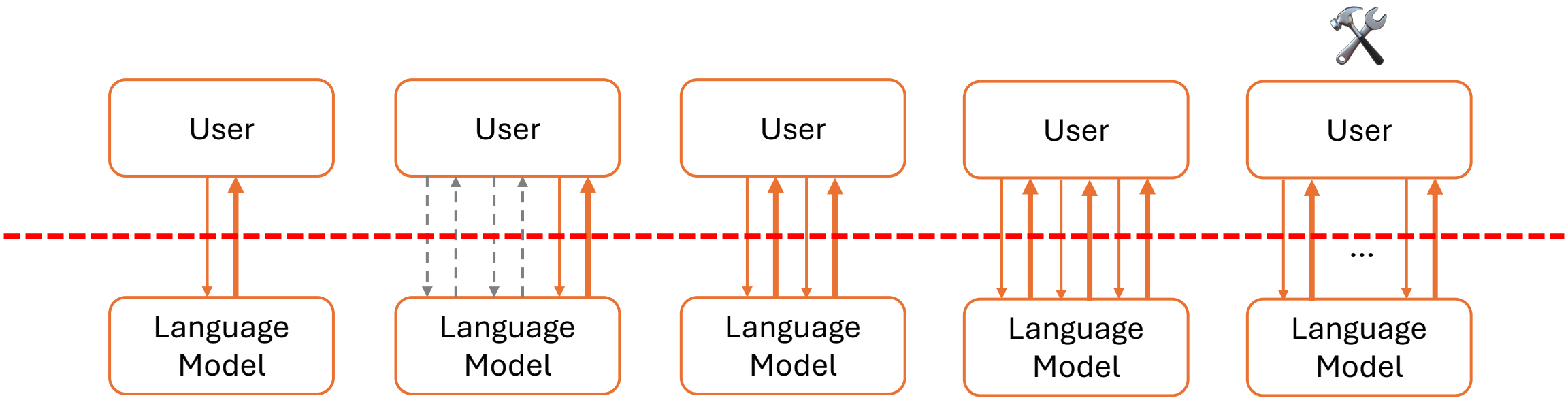

LLM
← User

Prompting Pipelines



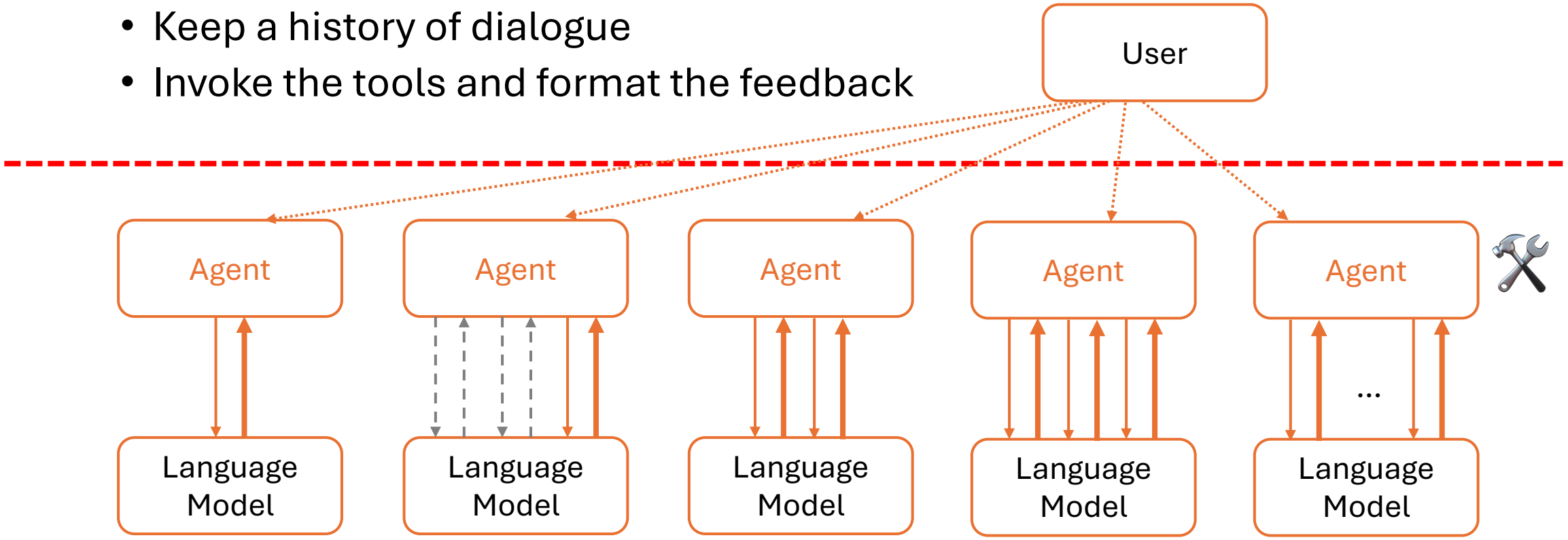
Prompting Pipelines

- User holds the responsibility to
  - Send requests to LLM
  - Keep a history of dialogue
  - Invoke the tools and format the feedback



Agentic Pipelines

- Agent holds the responsibility to
  - Send requests to LLM
  - Keep a history of dialogue
  - Invoke the tools and format the feedback

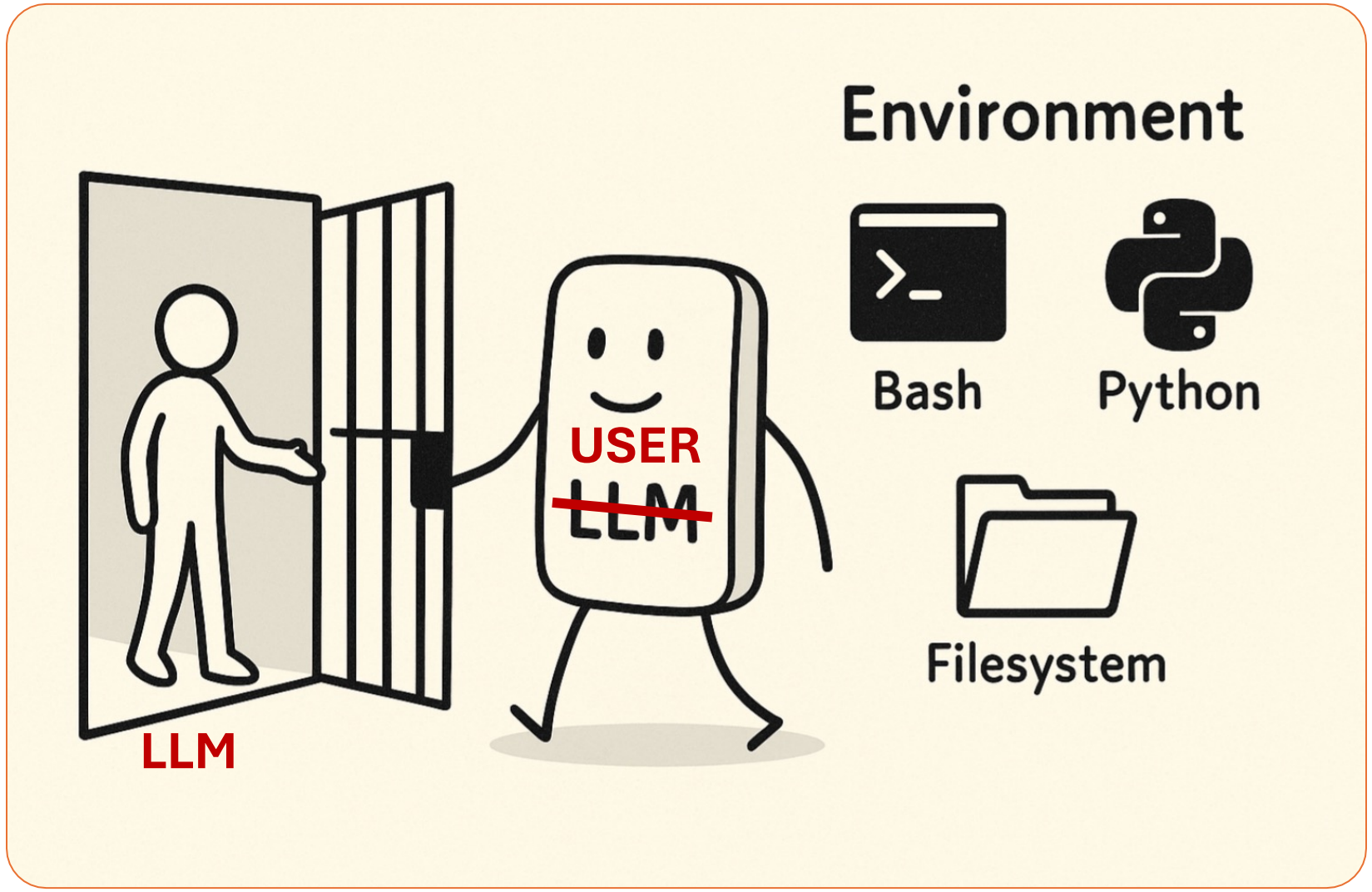


agency (noun)

- The capacity to act:
  - Definition: The ability of an individual or entity to make choices and act on them independently.
  - Example: “Children gradually develop their own agency in deciding what they want to study.”

LLM's agency (noun)

- The capacity to act:
  - Definition: The ability of an **LLM** to make choices and act on them independently.
  - Example: "**LLM** gradually develop their own agency in deciding what they want to study."



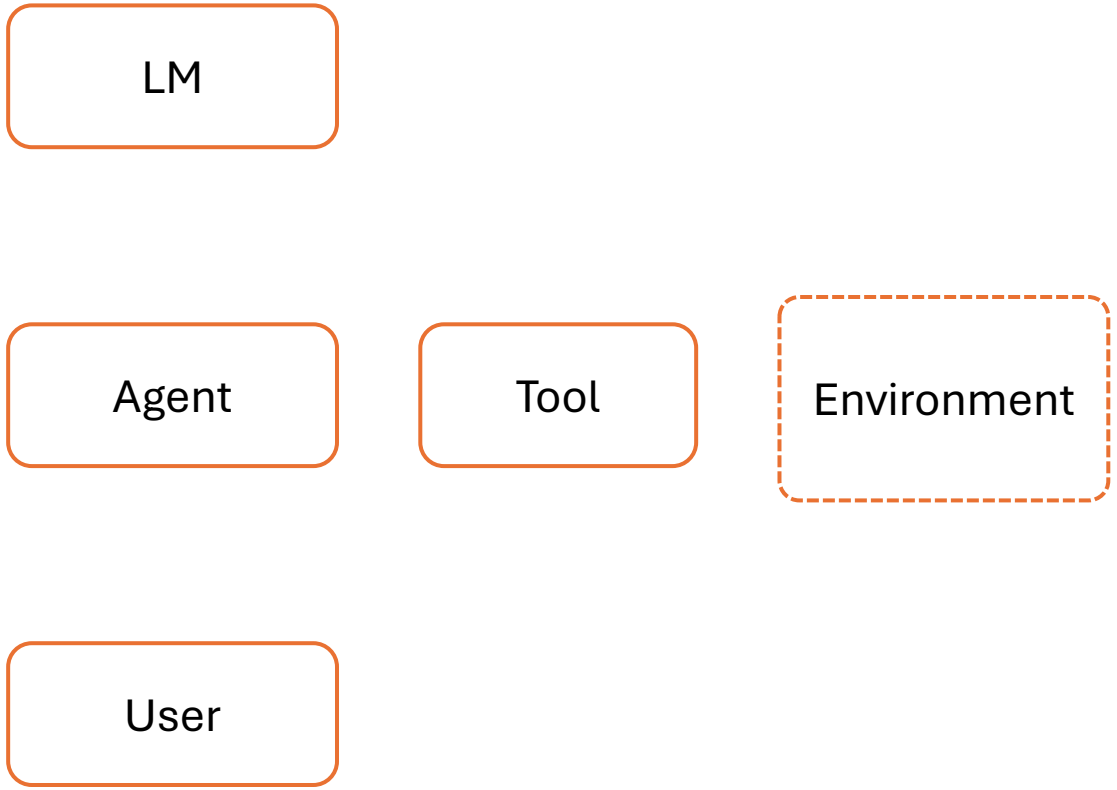
(GPT-5 generated figure, which needs some corrections...)

Definitions of Entities

- User (human, end-user)
- Agent (a computer program)
- LM (a neural language model)
- Environment (the world)
- Tool (access point to the world)

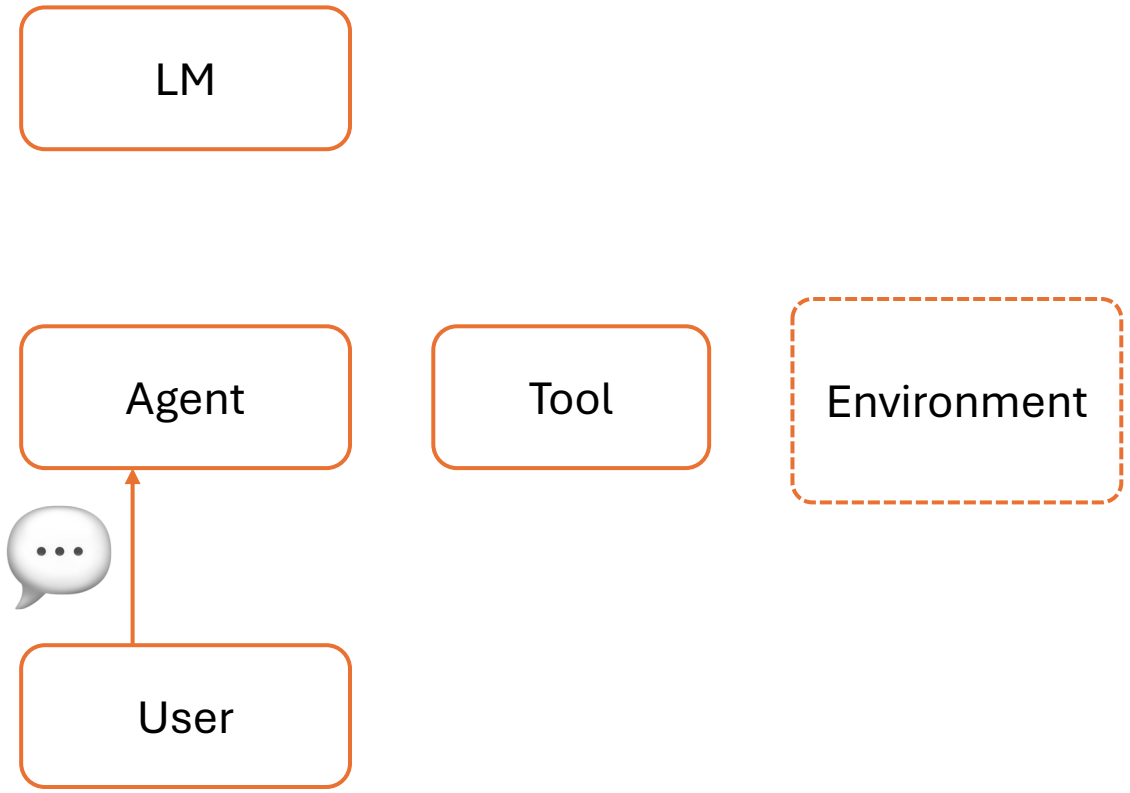
Definitions of Entities

- User (human, end-user)
- Agent (a computer program)
- LM (a neural language model)
- Environment (the world)
- Tool (access point to the world)



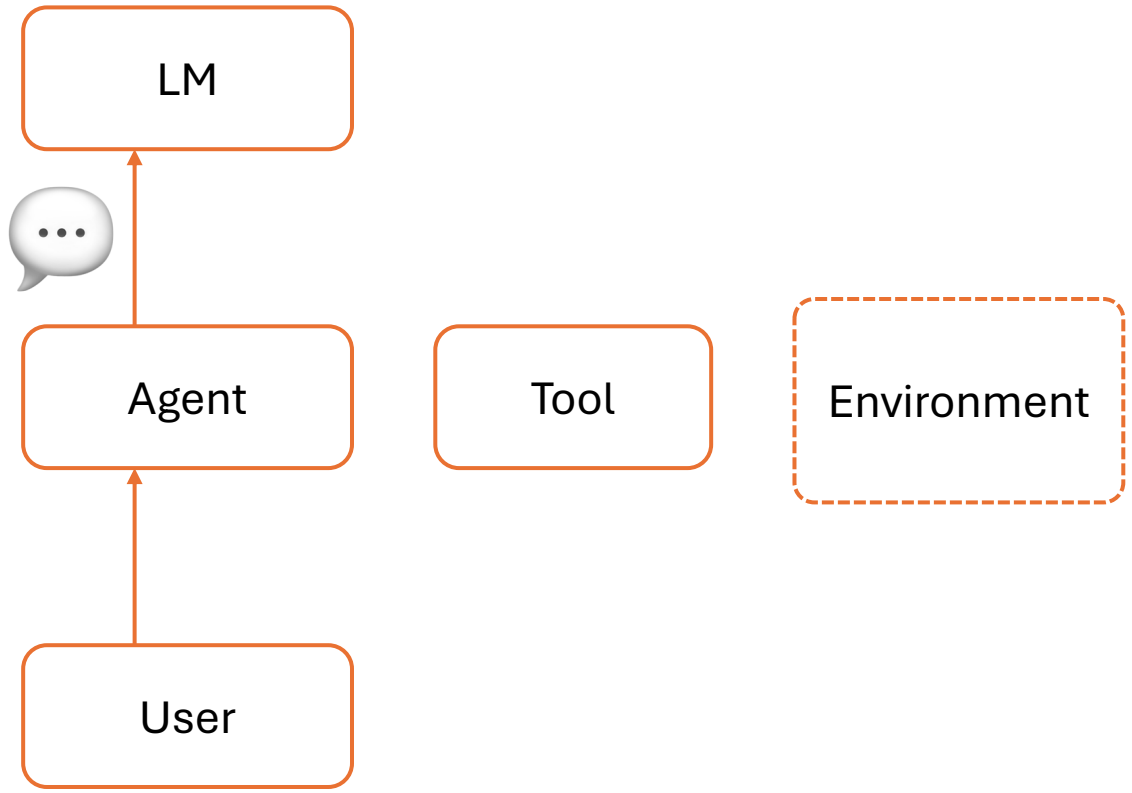
Definitions of Entities

- User (human, end-user)
- Agent (a computer program)
- LM (a neural language model)
- Environment (the world)
- Tool (access point to the world)



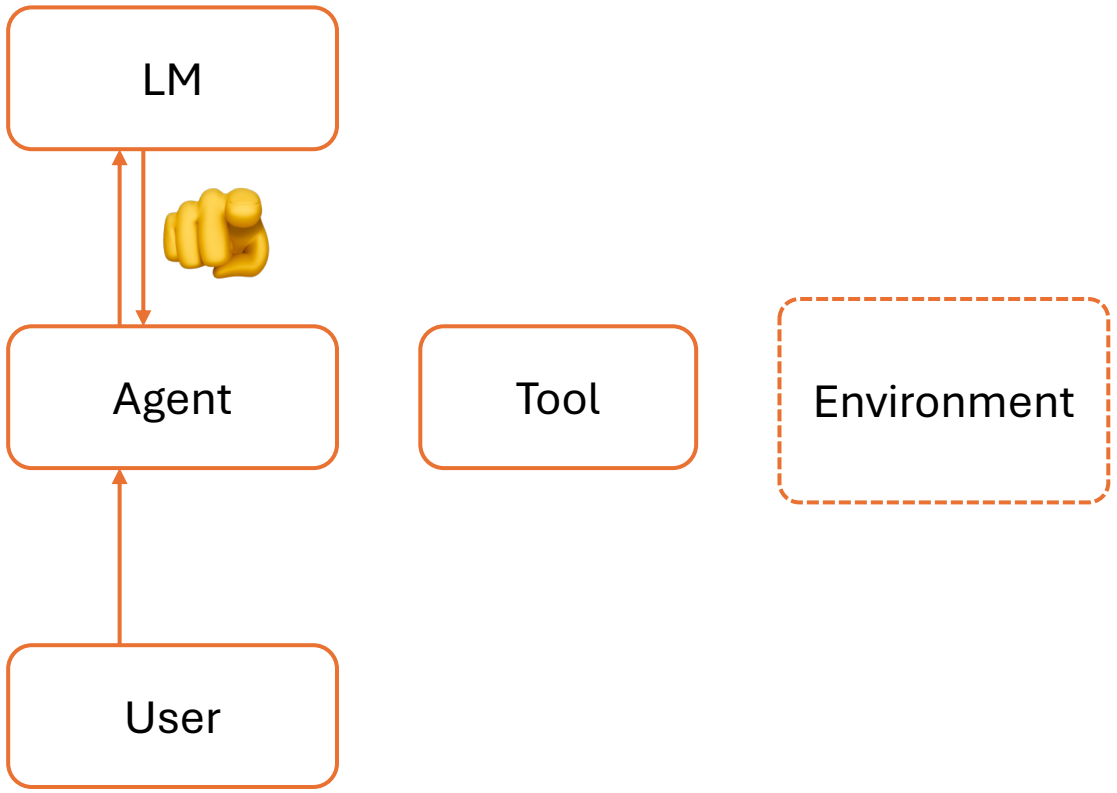
Definitions of Entities

- User (human, end-user)
- Agent (a computer program)
- LM (a neural language model)
- Environment (the world)
- Tool (access point to the world)



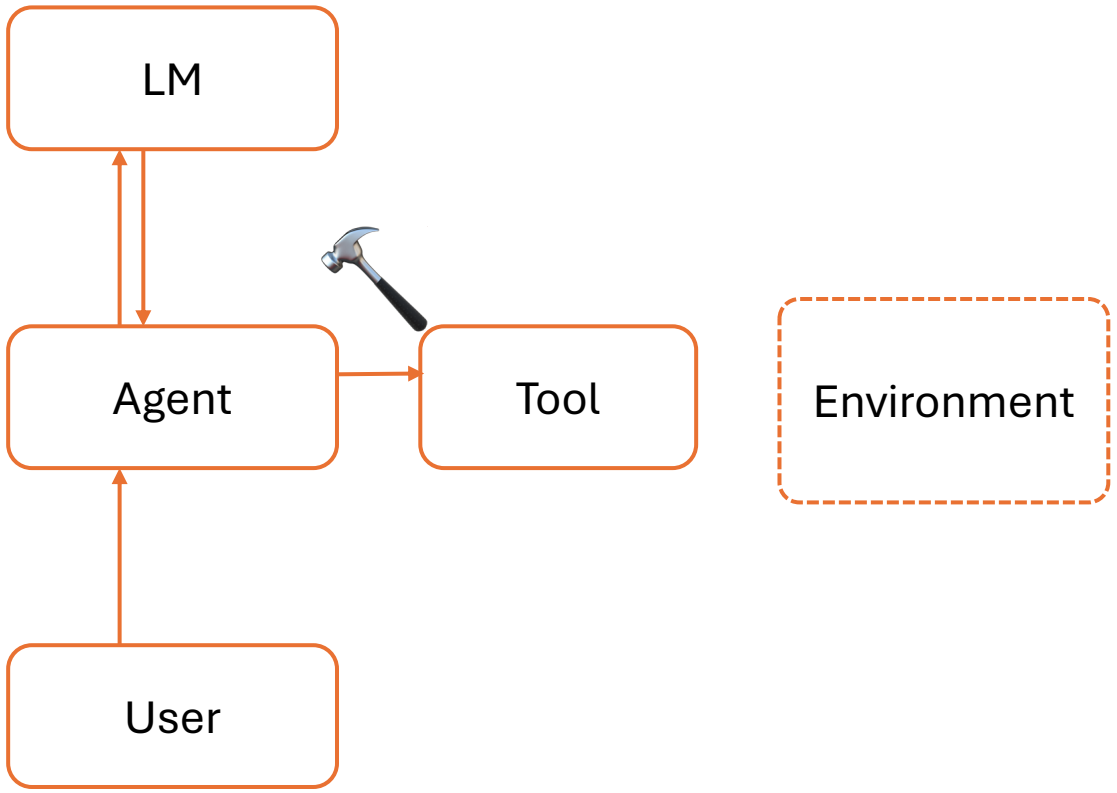
Definitions of Entities

- User (human, end-user)
- Agent (a computer program)
- LM (a neural language model)
- Environment (the world)
- Tool (access point to the world)



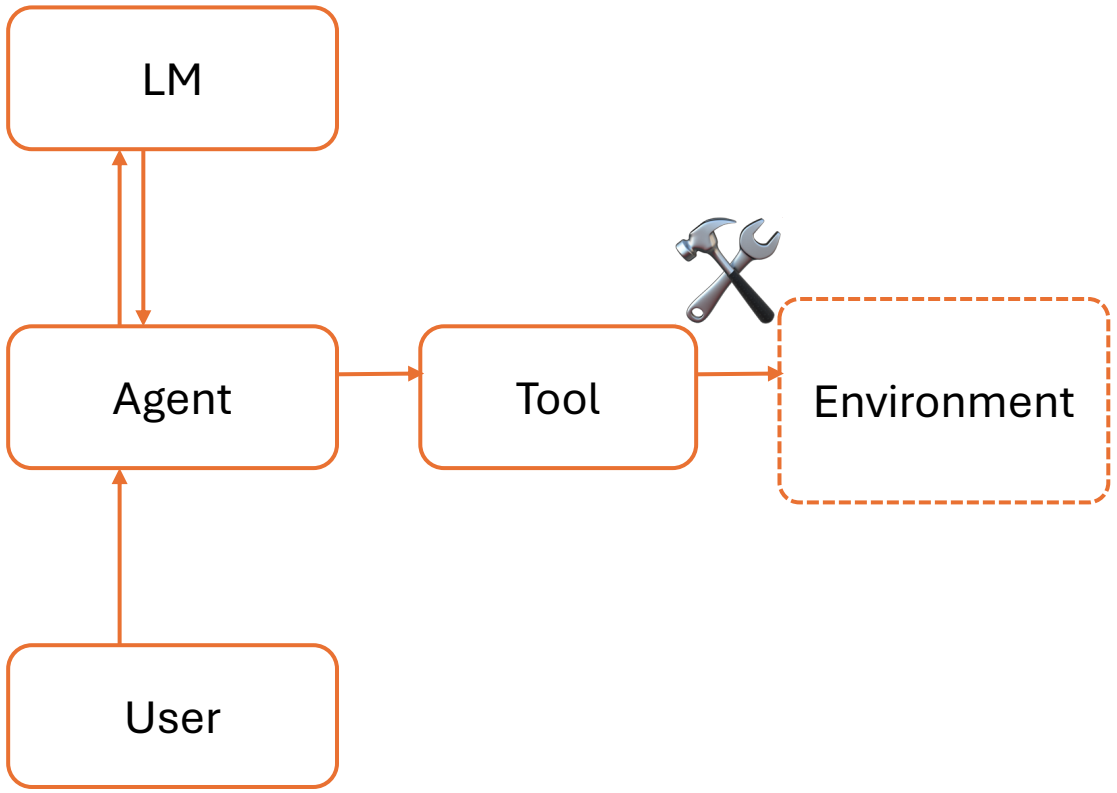
Definitions of Entities

- User (human, end-user)
- Agent (a computer program)
- LM (a neural language model)
- Environment (the world)
- Tool (access point to the world)



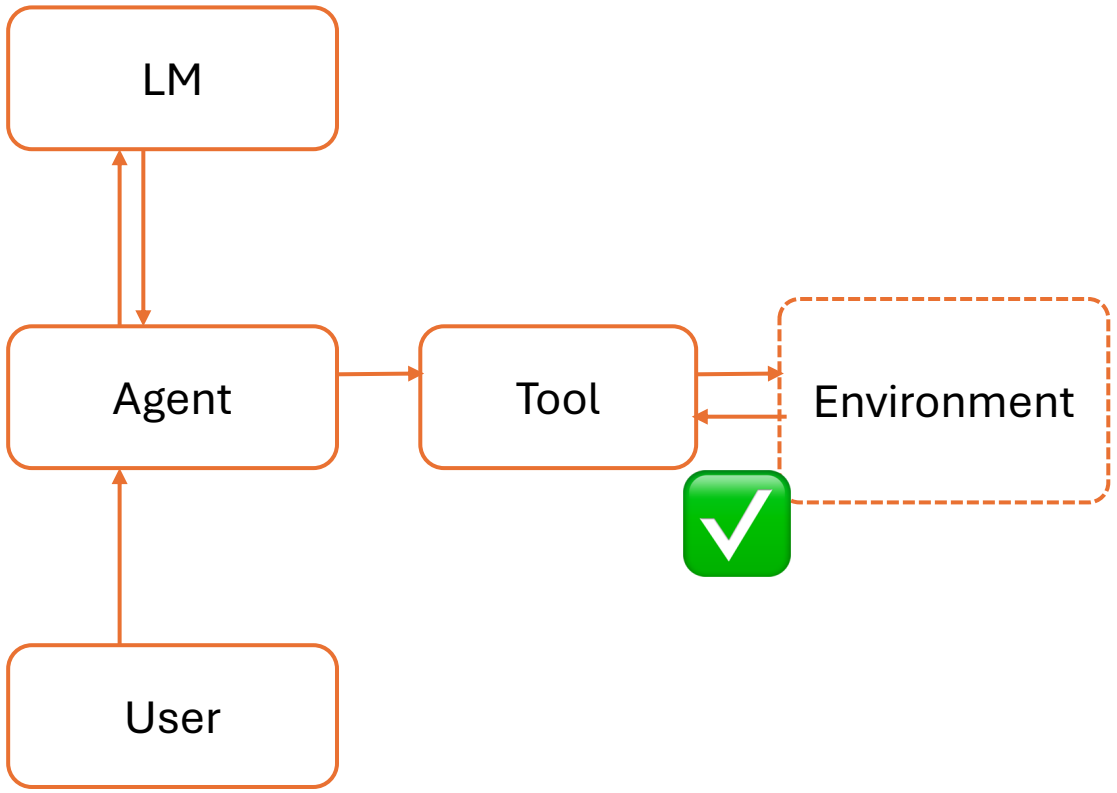
Definitions of Entities

- User (human, end-user)
- Agent (a computer program)
- LM (a neural language model)
- Environment (the world)
- Tool (access point to the world)



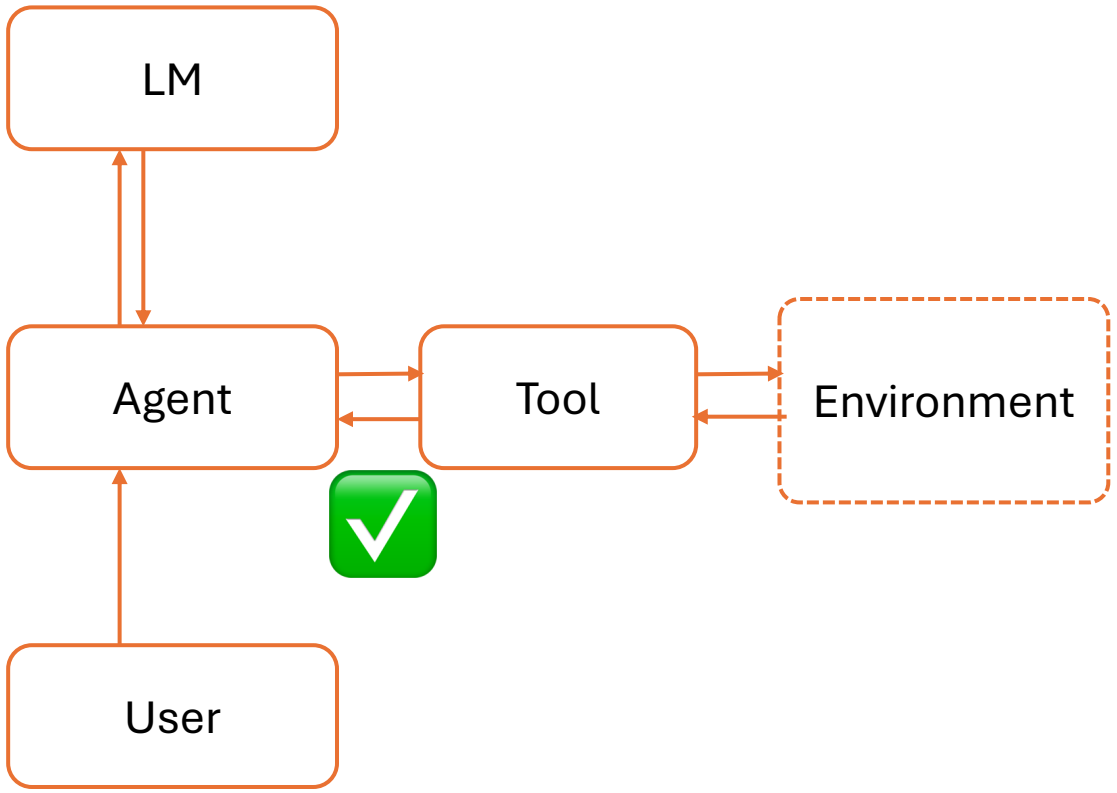
Definitions of Entities

- User (human, end-user)
- Agent (a computer program)
- LM (a neural language model)
- Environment (the world)
- Tool (access point to the world)



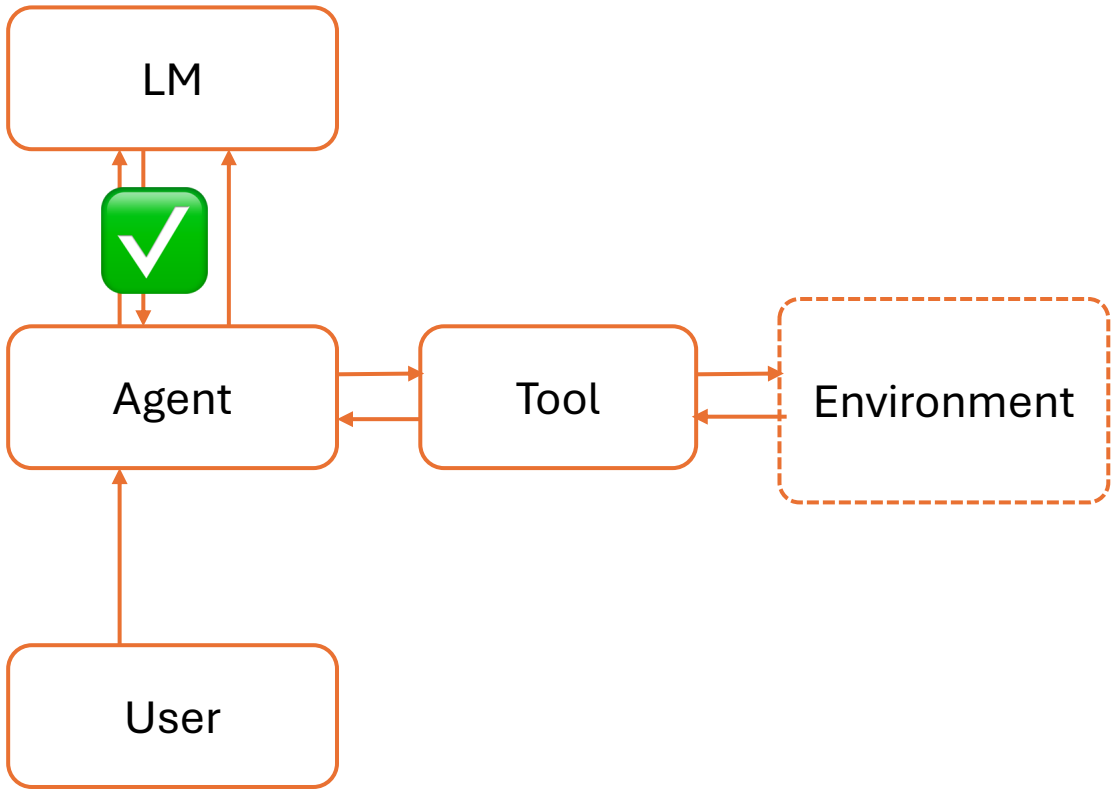
Definitions of Entities

- User (human, end-user)
- Agent (a computer program)
- LM (a neural language model)
- Environment (the world)
- Tool (access point to the world)



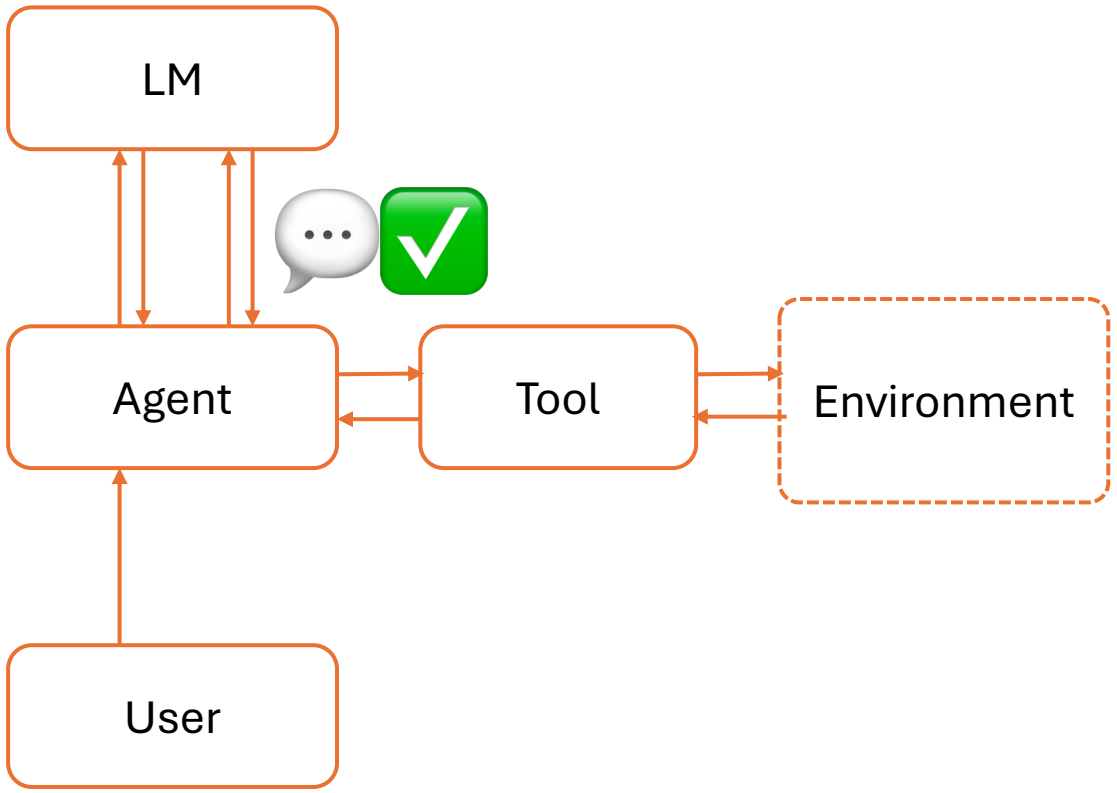
Definitions of Entities

- User (human, end-user)
- Agent (a computer program)
- LM (a neural language model)
- Environment (the world)
- Tool (access point to the world)



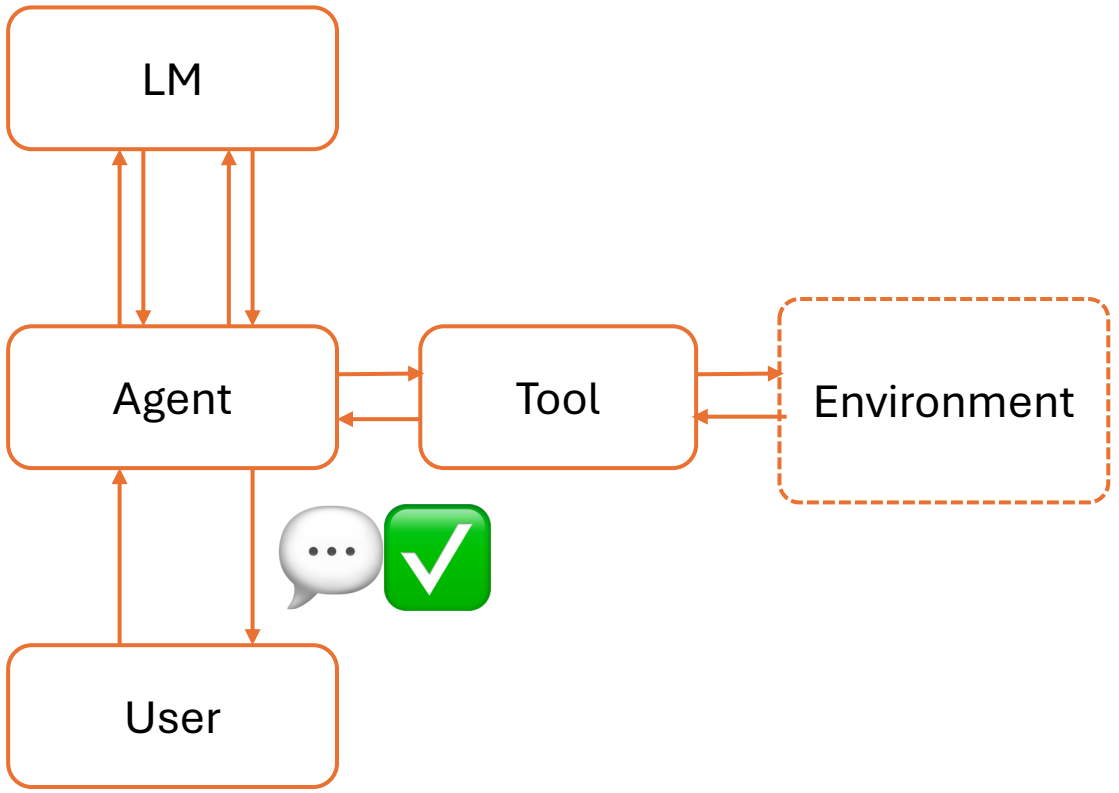
Definitions of Entities

- User (human, end-user)
- Agent (a computer program)
- LM (a neural language model)
- Environment (the world)
- Tool (access point to the world)



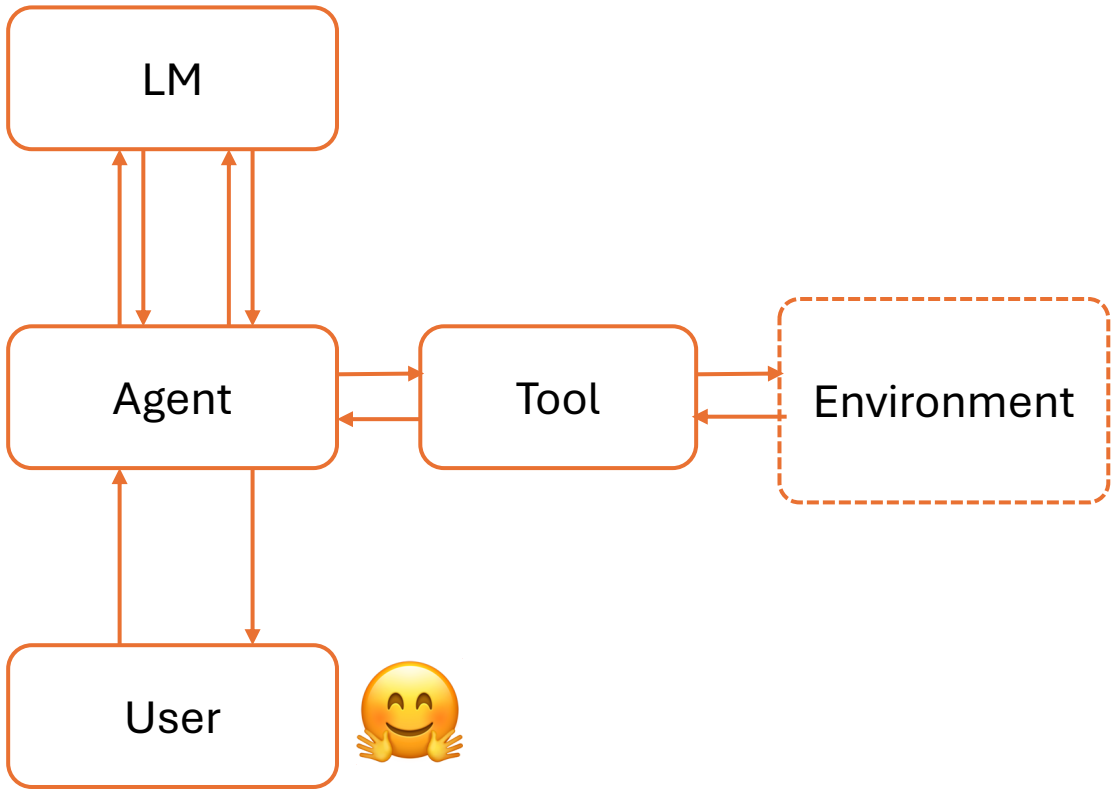
Definitions of Entities

- User (human, end-user)
- Agent (a computer program)
- LM (a neural language model)
- Environment (the world)
- Tool (access point to the world)



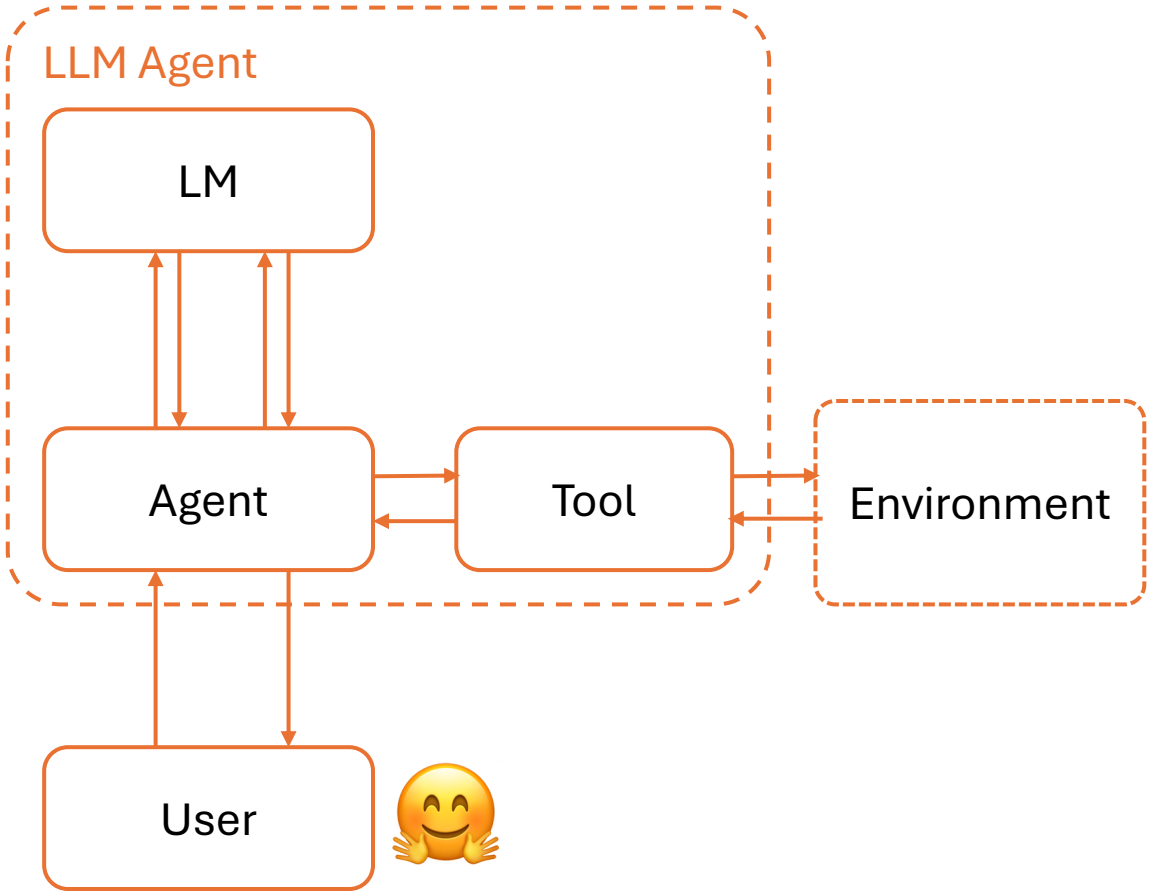
Definitions of Entities

- User (human, end-user)
- Agent (a computer program)
- LM (a neural language model)
- Environment (the world)
- Tool (access point to the world)



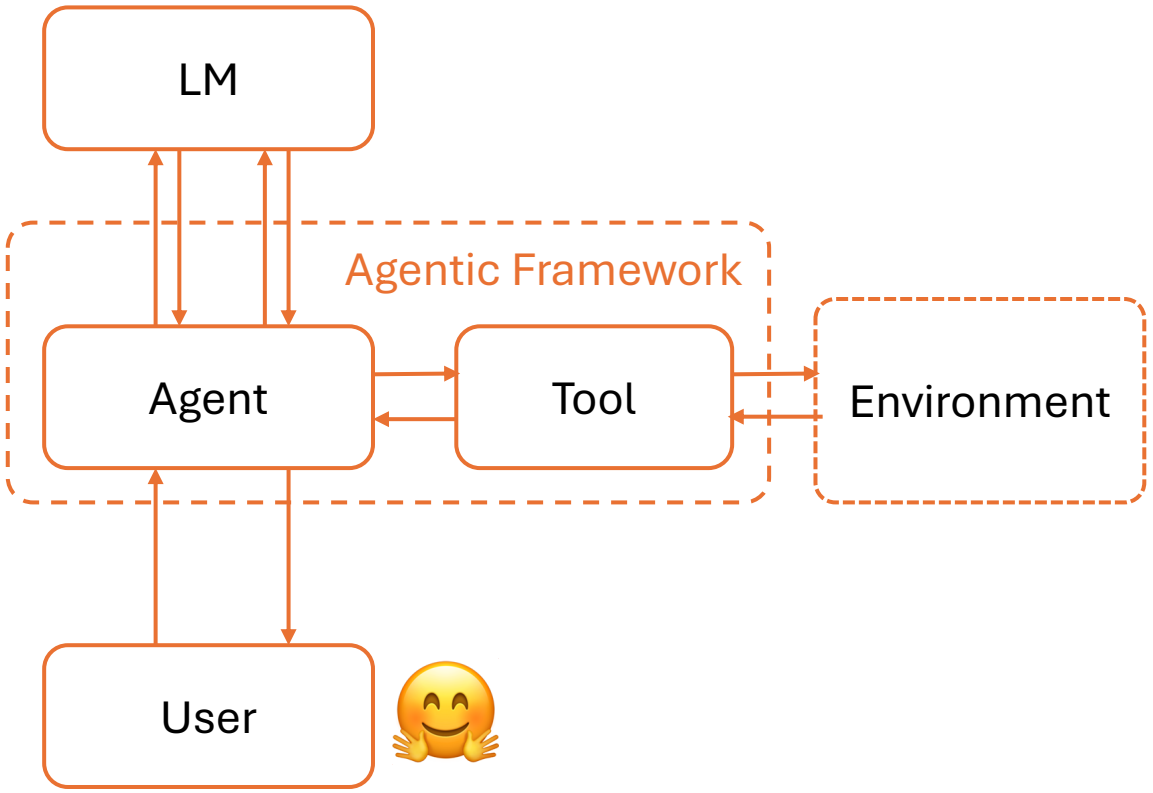
Definitions of Entities

- User (human, end-user)
- Agent (a computer program)
- LM (a neural language model)
- Environment (the world)
- Tool (access point to the world)



Definitions of Entities

- User (human, end-user)
- Agent (a computer program)
- LM (a neural language model)
- Environment (the world)
- Tool (access point to the world)



Foundations of Tool Use

Toolformer: Language Models Can Teach Themselves to Use Tools

Timo Schick **Jane Dwivedi-Yu** **Roberto Dessì**[†] **Roberta Raileanu**
Maria Lomeli **Luke Zettlemoyer** **Nicola Cancedda** **Thomas Scialom**

Meta AI Research [†]Universitat Pompeu Fabra

Toolformer: Language Models Can Teach Themselves to Use Tools, Schick et. al., 2023

Toolformer: Language Models Can Teach Themselves to Use Tools

Timo Schick **Jane Dwivedi-Yu**
Maria Lomeli **Luke Zettlemoyer**

Meta AI Research †Ur

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.

Toolformer: Language Models Can Teach Themselves to Use Tools, Schick et. al., 2023

Toolformer: Language Models Can Teach Themselves to Use Tools

Timo… …leanu
Maria … …Scialom

The New England Journal of Medicine is a registered trademark of [QA("Who is… England Journal of Medi… Medical Society] the MM…

Out of 1400 participants, … → 0.29] 29%) passed the…

The name derives from "l… [MT("tortuga") → turtle] …

The Brown Act is Californ… Act") → The Ralph M. Br… California State Legislatu… right to attend and partic… legislative bodies.] that r… city councils, to hold their…

Your task is to add calls to a Question Answering API to a piece of text. The questions should help you get information required to complete the text. You can call the API by writing "[QA(question)]" where "question" is the question you want to ask. Here are some examples of API calls:

Input: Joe Biden was born in Scranton, Pennsylvania.

Output: Joe Biden was born in **[QA("Where was Joe Biden born?")]** Scranton, **[QA("In which state is Scranton?")]** Pennsylvania.

Input: Coca-Cola, or Coke, is a carbonated soft drink manufactured by the Coca-Cola Company.

Output: Coca-Cola, or **[QA("What other name is Coca-Cola known by?")]** Coke, is a carbonated soft drink manufactured by **[QA("Who manufactures Coca-Cola?")]** the Coca-Cola Company.

Input: x

Output:

Toolformer: Language Models Can Teach Themselves to Use Tools, Schick et. al., 2023

Toolformer: Language Models Can Teach Themselves to Use Tools

The New England Journal of Medicine is a registered trademark of [QA("Who is the publisher of The New England Jou
Medical Soci

Out of 1400 p
→ 0.29] 29%)

The name de
[MT("tortuga

The Brown A
Act") → The
California St
right to atten
legislative bo
city councils,

*Your task is to add calls to a Question Answering API to a piece of text. The questions should help you get information r
API by writing
question you
calls:*

Input: Joe Bi

Output: Joe
Biden born?"
Scranton?")]

Input: Coca-
manufacture

Output: Coca
Coca-Cola k
manufacture
the Coca-Col

Input: x

Output:

oberto Dessì† **Roberta Raileanu**

API Name	Example Input	Example Output
Question Answering	Where was the Knights of Columbus founded?	New Haven, Connecticut
Wikipedia Search	Fishing Reel Types	Spin fishing > Spin fishing is distinguished between fly fishing and bait cast fishing by the type of rod and reel used. There are two types of reels used when spin fishing, the open faced reel and the closed faced reel.
Calculator	27 + 4 * 2	35
Calendar	ε	Today is Monday, January 30, 2023.
Machine Translation	sûreté nucléaire	nuclear safety

Toolformer: Language Models Can Teach Themselves to Use Tools, Schick et. al., 2023

Toolformer: Language Models Can Teach Themselves to Use Tools

The New England Journal of Medicine is a registered trademark of [QA("Who is the publisher of The New England Journal of Medicine?")] → 0.29] 29%

Out of 1400 p

The name de [MT("tortuga

The Brown A Act") → The California St right to atten legislative bo city councils,

Your task is to add calls to a Question Answering API to a piece of text. The questions should help you get information r API by writing question you calls:

Input: Joe Bi

Output: Joe Biden born?" Scranton?")]

Input: Coca-Cola, or Coke, is a manufactured by the Coca-Cola

Output: Coca-Cola, or [QA("Wha Coca-Cola known by?")] Coke, i manufactured by [QA("Who ma the Coca-Cola Company.

Input: x

Output:

oberto Dessì† Roberta Raileanu

n

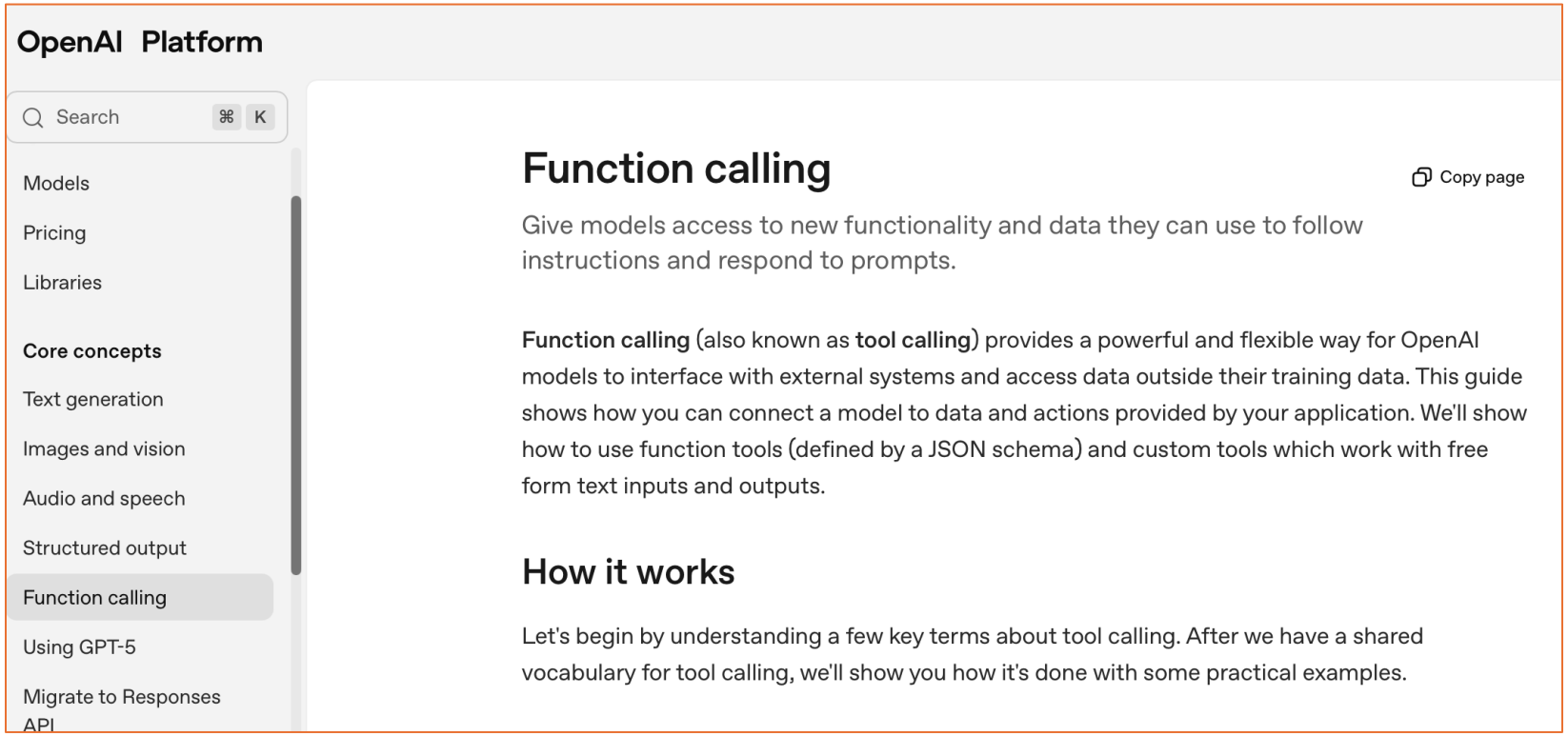
API Name	Example Input	Example Output
Question Answering	Where was the Knights	New Haven, Connecticut
Wikipedia Search		
Calculator		
Calendar		
Machine Translat		

Model	ASDiv	SVAMP	MAWPS
GPT-J	7.5	5.2	9.9
GPT-J + CC	9.6	5.0	9.3
Toolformer (disabled)	14.8	6.3	15.0
Toolformer	**40.4**	**29.4**	**44.0**
OPT (66B)	6.0	4.9	7.9
GPT-3 (175B)	14.0	10.0	19.8

Table 4: Results for various benchmarks requiring mathematical reasoning. Toolformer makes use of the calculator tool for most examples, clearly outperforming even OPT (66B) and GPT-3 (175B).

Toolformer: Language Models Can Teach Themselves to Use Tools, Schick et. al., 2023

Foundations of Tool Use



OpenAI Platform

Search ⌘ K

- Models
- Pricing
- Libraries

Core concepts

- Text generation
- Images and vision
- Audio and speech
- Structured output
- Function calling
- Using GPT-5
- Migrate to Responses API

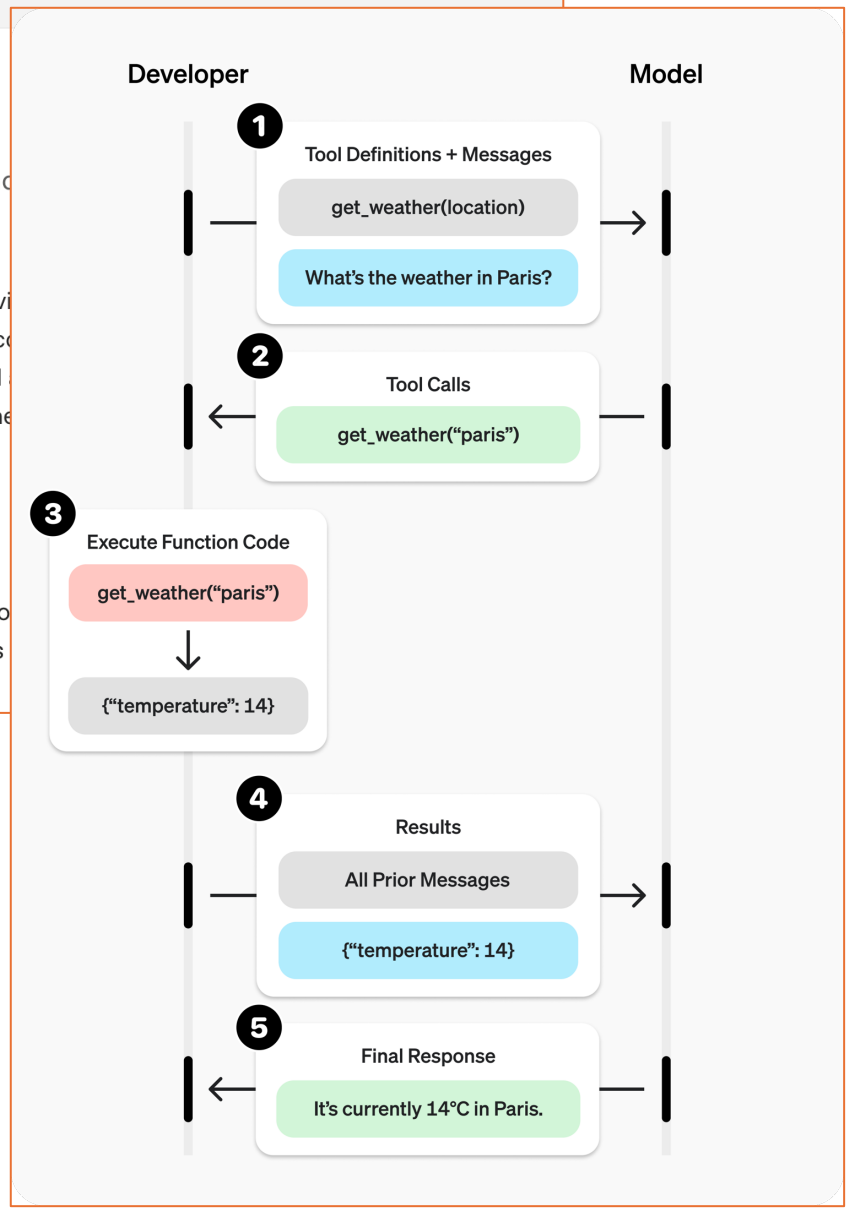
Function calling

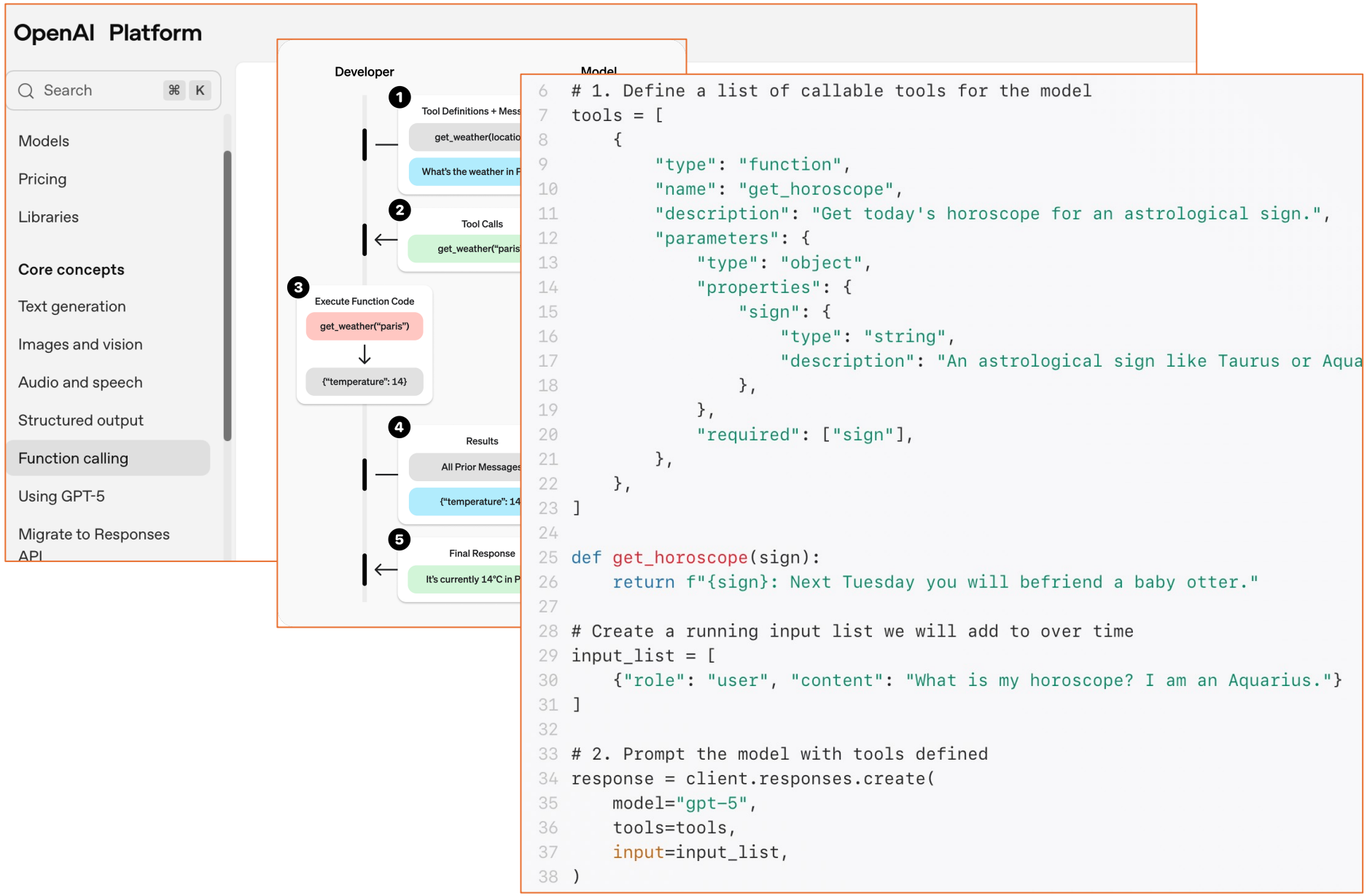
Give models access to new functionality and
instructions and respond to prompts.

Function calling (also known as **tool calling**) provi
models to interface with external systems and acc
shows how you can connect a model to data and
how to use function tools (defined by a JSON sche
form text inputs and outputs.

How it works

Let's begin by understanding a few key terms abo
vocabulary for tool calling, we'll show you how it's



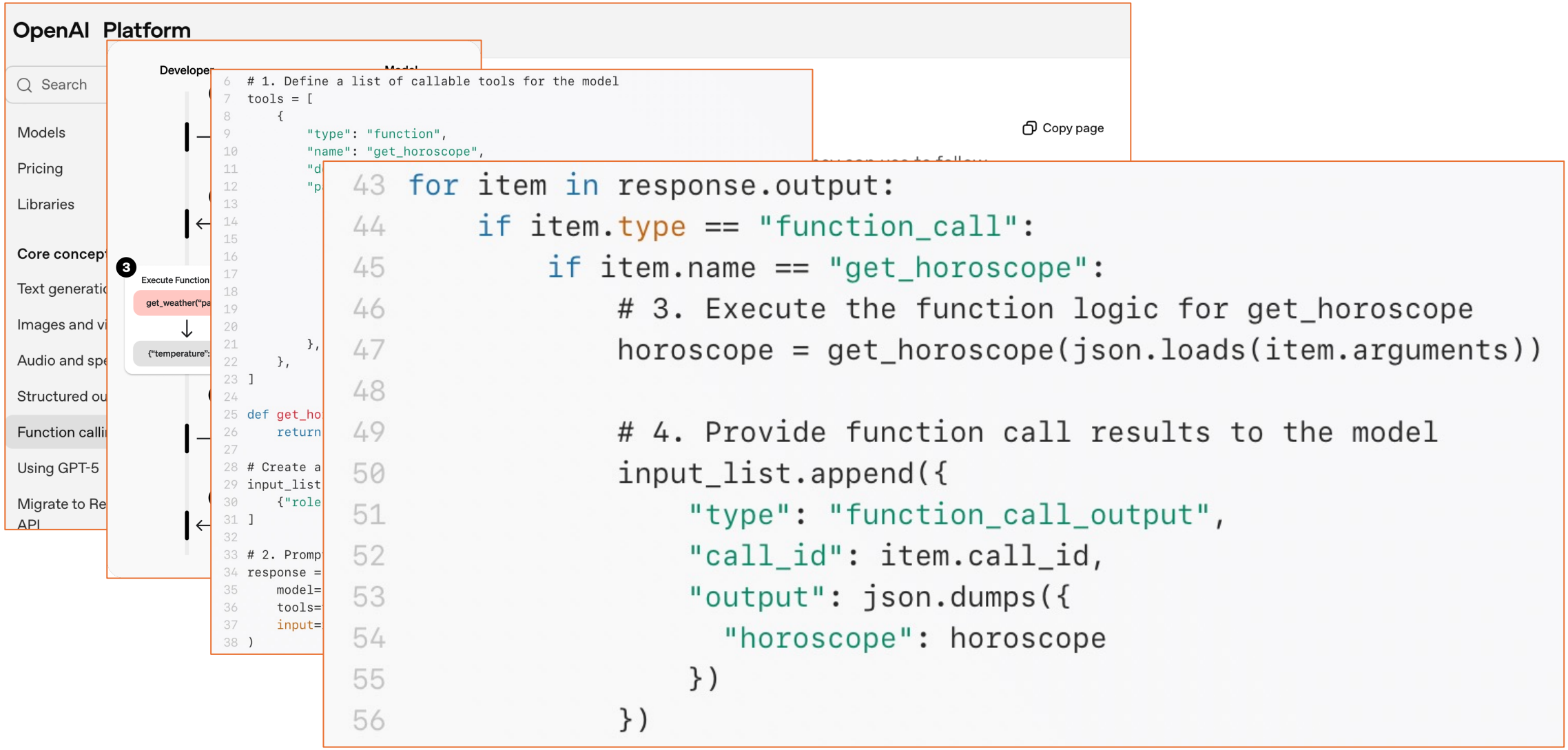



```python
# 1. Define a list of callable tools for the model
tools = [
    {
        "type": "function",
        "name": "get_horoscope",
        "description": "Get today's horoscope for an astrological sign.",
        "parameters": {
            "type": "object",
            "properties": {
                "sign": {
                    "type": "string",
                    "description": "An astrological sign like Taurus or Aqua
                },
            },
            "required": ["sign"],
        },
    },
]

def get_horoscope(sign):
    return f"{sign}: Next Tuesday you will befriend a baby otter."

# Create a running input list we will add to over time
input_list = [
    {"role": "user", "content": "What is my horoscope? I am an Aquarius."}
]

# 2. Prompt the model with tools defined
response = client.responses.create(
    model="gpt-5",
    tools=tools,
    input=input_list,
)
```


OpenAI Platform
Search
Models
Pricing
Libraries
Core concep
Text generatio
Images and vi
Audio and spe
Structured ou
Function calli
Using GPT-5
Migrate to Re
API
Copy page
3
Execute Function
get_weather("pa
{"temperature":
# 1. Define a list of callable tools for the model
tools = [
{
"type": "function",
"name": "get_horoscope",
"d
"p
},
},
]
def get_ho
return
Create a
input_list
{"role
]
2. Prompt
response =
model=
tools=
input=
)
for item in response.output:
if item.type == "function_call":
if item.name == "get_horoscope":
# 3. Execute the function logic for get_horoscope
horoscope = get_horoscope(json.loads(item.arguments))
# 4. Provide function call results to the model
input_list.append({
"type": "function_call_output",
"call_id": item.call_id,
"output": json.dumps({
"horoscope": horoscope
})
})

Exemplars of Tool Use

VOYAGER: An Open-Ended Embodied Agent with Large Language Models

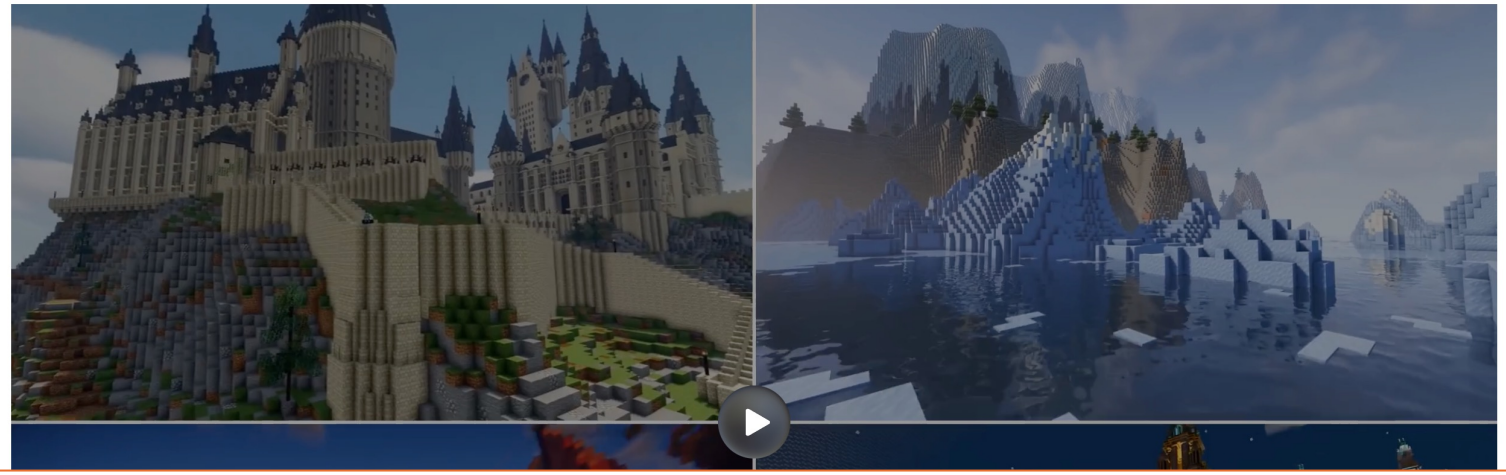
Guanzhi Wang[1 2]✉, **Yuqi Xie**[3], **Yunfan Jiang**[4*], **Ajay Mandlekar**[1*],
Chaowei Xiao[1 5], **Yuke Zhu**[1 3], **Linxi "Jim" Fan**[1†]✉, **Anima Anandkumar**[1 2†]
[1]NVIDIA, [2]Caltech, [3]UT Austin, [4]Stanford, [5]UW Madison
*Equal contribution †Equal advising ✉ Corresponding authors
https://voyager.minedojo.org

MINEDOJO

Building Open-Ended Embodied Agents with Internet-Scale Knowledge

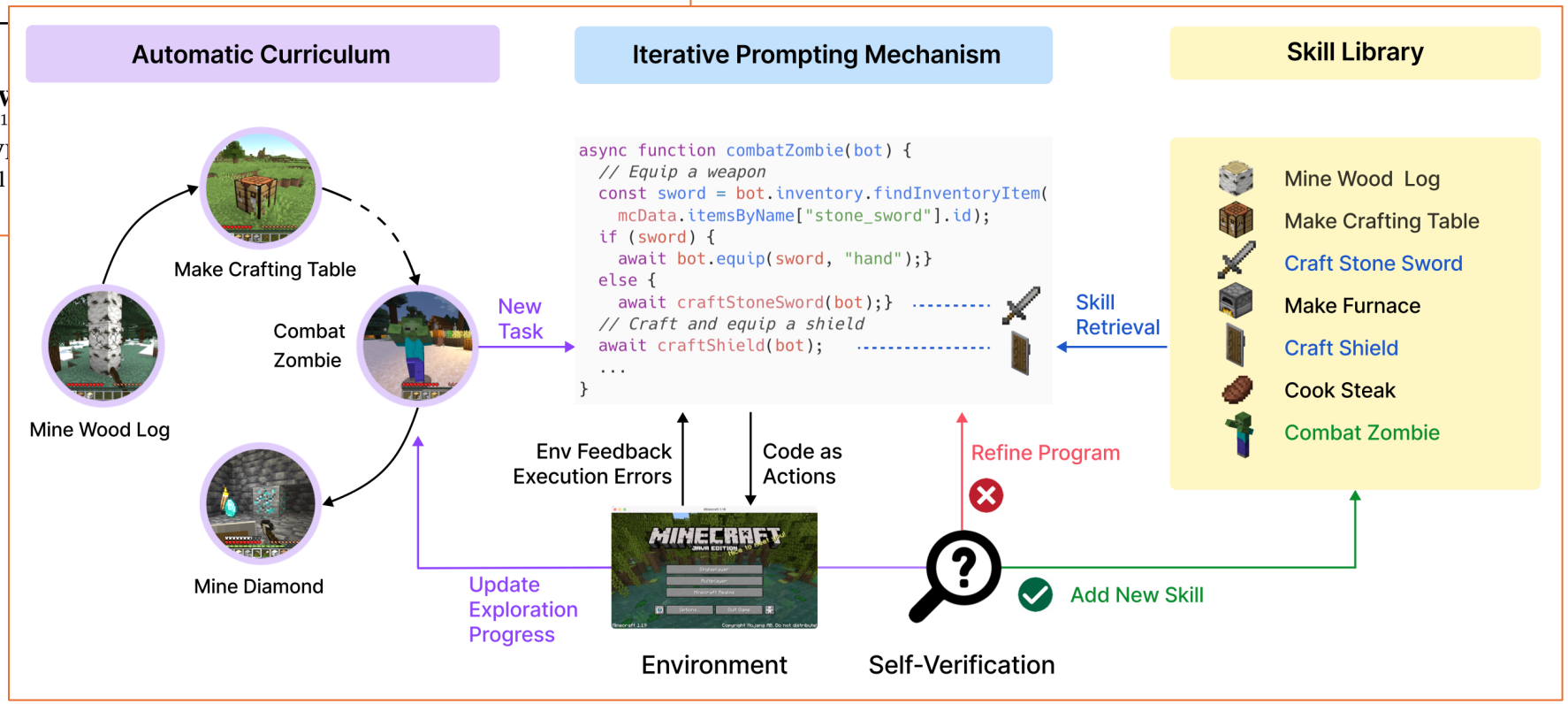
✨ NeurIPS 2022 Outstanding Paper Award ✨

Overview | Paper | NeurIPS 2022 | Knowledge Base | GitHub | Docs | Team



VOYAGER: An Open-Ended Embodied Agent with Large Language Models

Guanzhi V
Chaowei Xiao[1]
[1]NV
*Equal



VOYAGER: An Open-Ended Embodied Agent

Automatic Curriculum | Iterative Prompting Mechanism | Skill Library

Mine Wood Log

Inventory (5/36): {'oak_planks': 3, 'stick': 4, 'crafting_table': 1, 'stone': 3, 'wooden_pickaxe': 1}

GPT-4

Reasoning: Since you have a wooden pickaxe and some stones, it would be beneficial to upgrade your pickaxe to a stone pickaxe for better efficiency.
Task: Craft 1 stone pickaxe.

Biome: river
Inventory (4/36): {'sugar_cane': 3, 'paper': 3, 'fishing_rod': 1, 'wooden_pickaxe': 1}

GPT-4

Reasoning: Since you have a fishing rod in your inventory and you are near a river biome, it would be a good opportunity to catch some fish for food and experience.
Task: Catch 1 fish.

Nearby entities: pig, cat, villager
Health: 12/20
Hunger: 0/20

GPT-4

Reasoning: Your hunger is at 0, which means you need to find food to replenish your hunger. Since there are pigs nearby, you can kill one to obtain raw porkchops.
Task: Kill 1 pig.

Inventory (6/36): {'furnace': 1, 'stone_pickaxe': 1, 'oak_planks': 7, 'crafting_table': 1, 'raw_iron': 4, 'coal': 1}

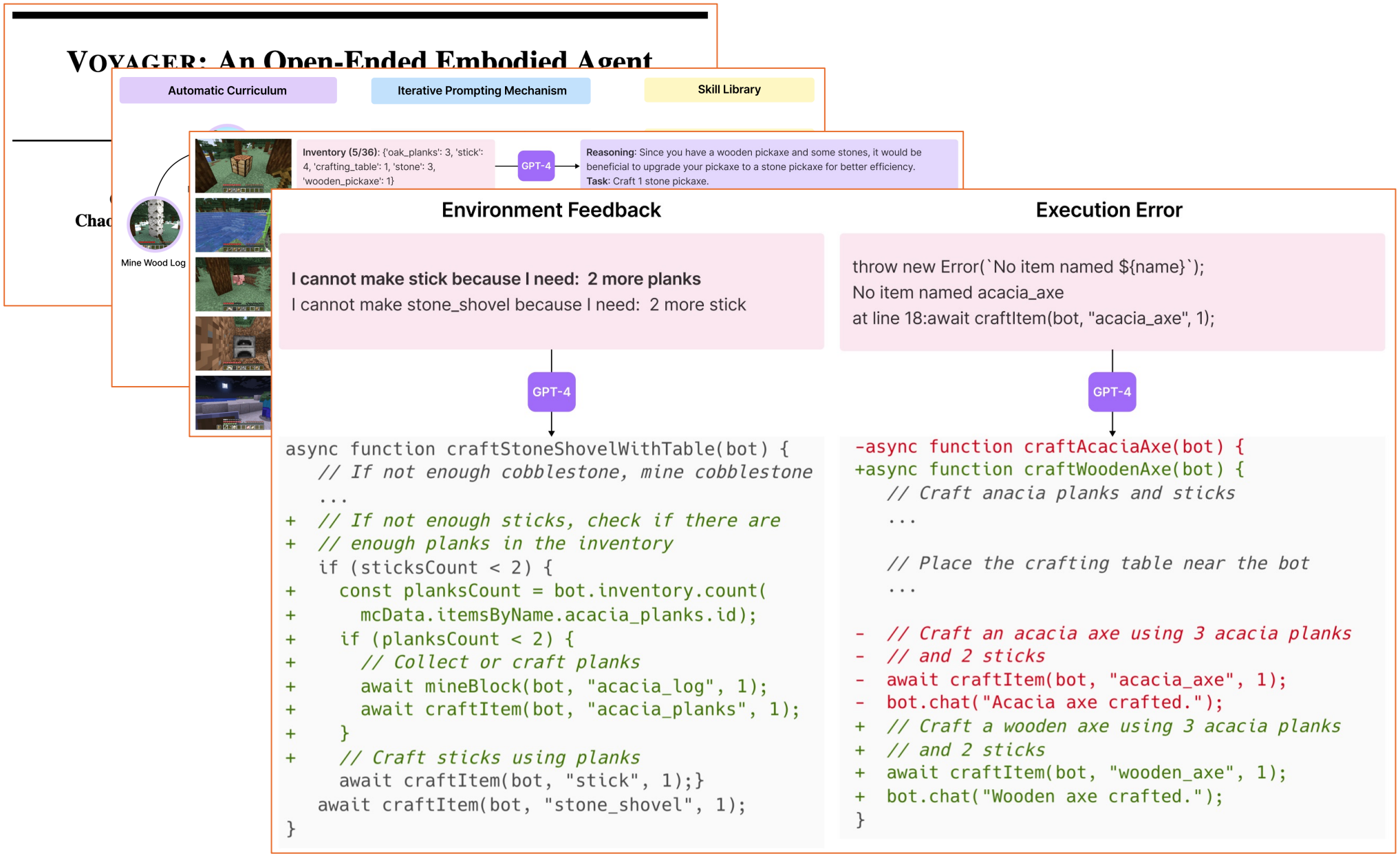
GPT-4

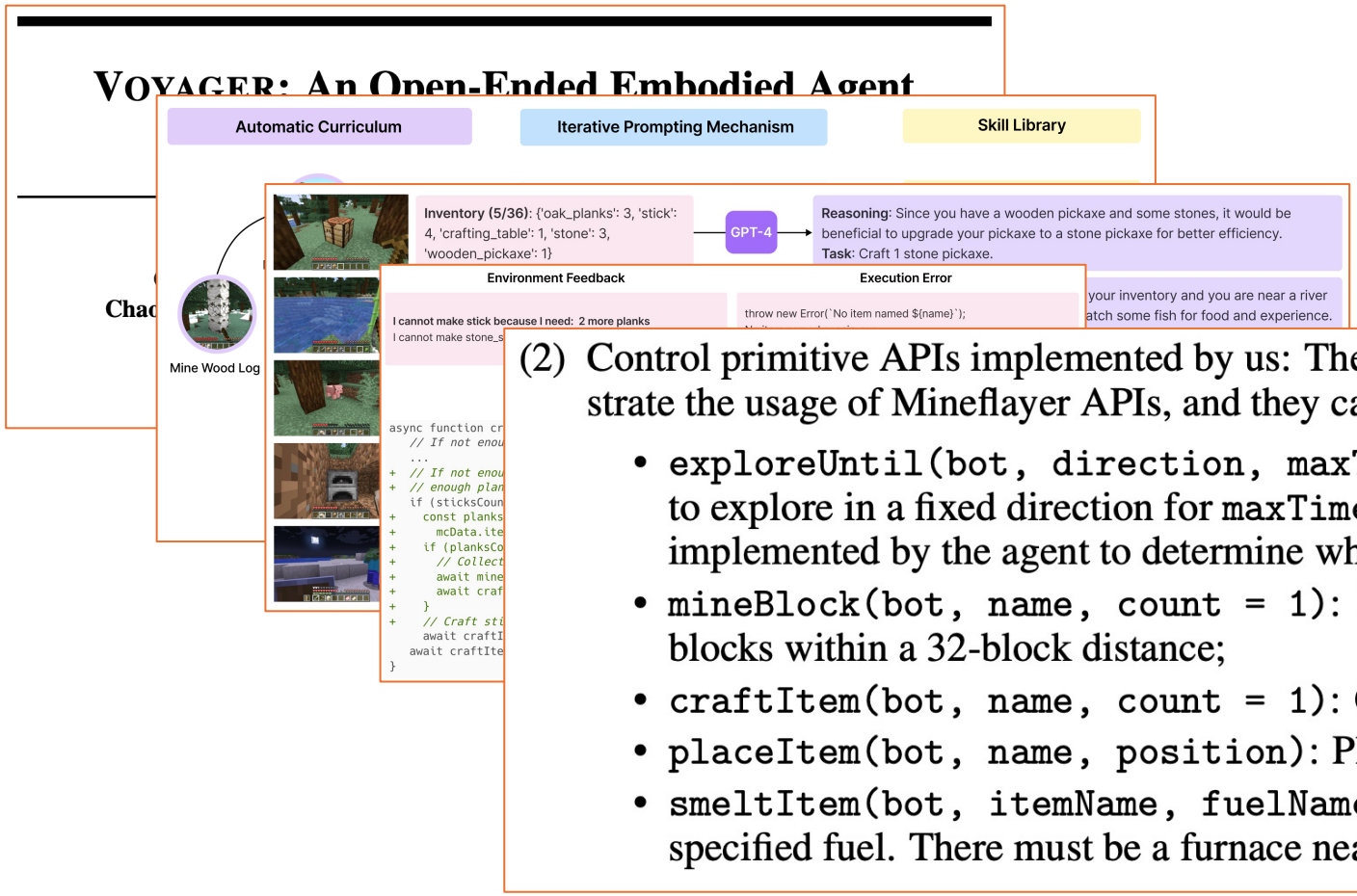
Reasoning: You have raw iron and coal, and you have a furnace. It's time to smelt the iron to make iron ingots, which can be used to craft better tools and armor.
Task: Smelt 4 raw iron.

Time: night
Nearby entities: zombie
Equipment: 'stone_sword', 'shield'

GPT-4

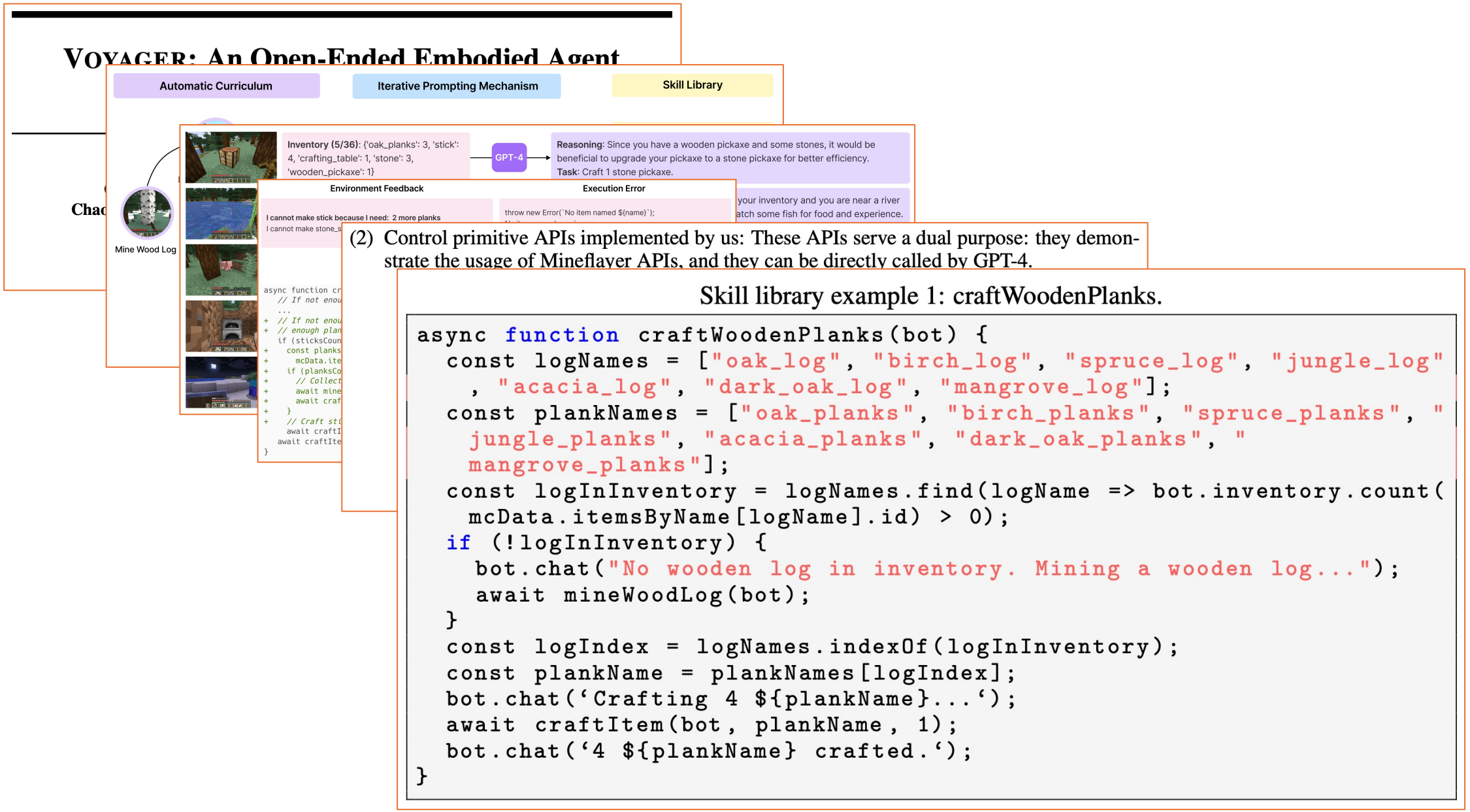
Reasoning: Since it's night and there's a zombie nearby, it's a good opportunity to try killing the zombie now that you have a stone sword and a shield equipped.
Task: Kill 1 zombie.

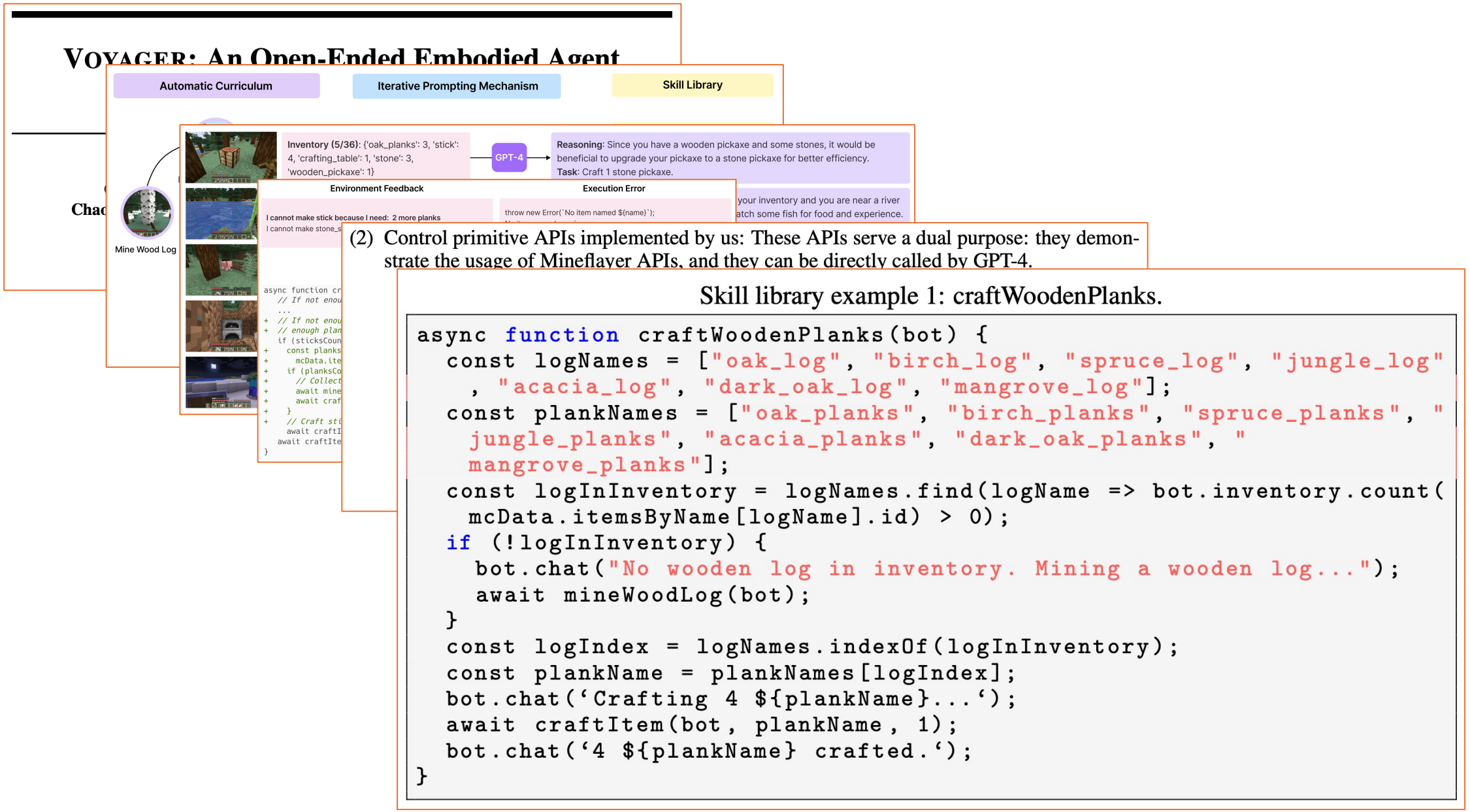

VOYAGER: An Open-Ended Embodied Agent
Automatic Curriculum
Iterative Prompting Mechanism
Skill Library
Mine Wood Log
Inventory (5/36): {'oak_planks': 3, 'stick': 4, 'crafting_table': 1, 'stone': 3, 'wooden_pickaxe': 1}
GPT-4
Reasoning: Since you have a wooden pickaxe and some stones, it would be beneficial to upgrade your pickaxe to a stone pickaxe for better efficiency.
Task: Craft 1 stone pickaxe.
Environment Feedback
I cannot make stick because I need: 2 more planks
I cannot make stone_shovel because I need: 2 more stick
GPT-4
async function craftStoneShovelWithTable(bot) {
// If not enough cobblestone, mine cobblestone
...
+ // If not enough sticks, check if there are
+ // enough planks in the inventory
if (sticksCount < 2) {
+ const planksCount = bot.inventory.count(
+ mcData.itemsByName.acacia_planks.id);
+ if (planksCount < 2) {
+ // Collect or craft planks
+ await mineBlock(bot, "acacia_log", 1);
+ await craftItem(bot, "acacia_planks", 1);
+ }
+ // Craft sticks using planks
await craftItem(bot, "stick", 1);}
await craftItem(bot, "stone_shovel", 1);
}
Execution Error
throw new Error(`No item named ${name}`);
No item named acacia_axe
at line 18:await craftItem(bot, "acacia_axe", 1);
GPT-4
-async function craftAcaciaAxe(bot) {
+async function craftWoodenAxe(bot) {
// Craft anacia planks and sticks
...
// Place the crafting table near the bot
...
- // Craft an acacia axe using 3 acacia planks
- // and 2 sticks
- await craftItem(bot, "acacia_axe", 1);
- bot.chat("Acacia axe crafted.");
+ // Craft a wooden axe using 3 acacia planks
+ // and 2 sticks
+ await craftItem(bot, "wooden_axe", 1);
+ bot.chat("Wooden axe crafted.");
}



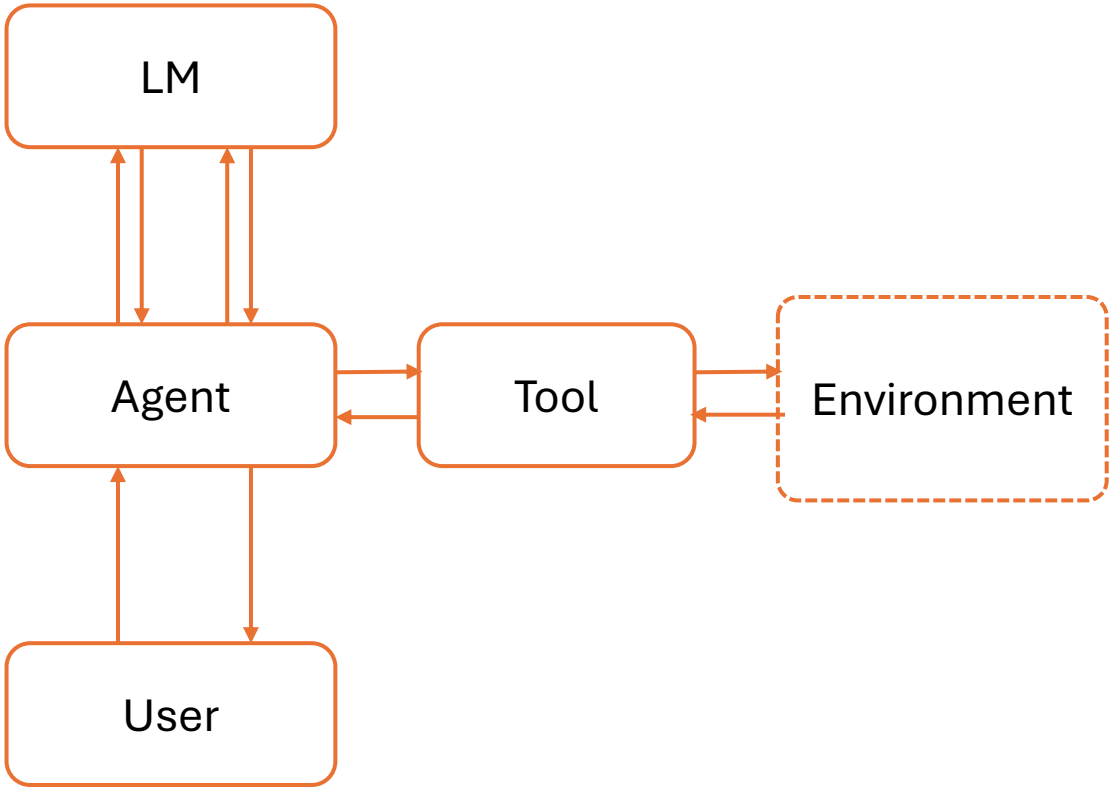
(2) Control primitive APIs implemented by us: These APIs serve a dual purpose: they demonstrate the usage of Mineflayer APIs, and they can be directly called by GPT-4.

- `exploreUntil(bot, direction, maxTime = 60, callback)`: Allow the agent to explore in a fixed direction for `maxTime`. The `callback` is the stopping condition implemented by the agent to determine when to stop exploring;
- `mineBlock(bot, name, count = 1)`: Mine and collect the specified number of blocks within a 32-block distance;
- `craftItem(bot, name, count = 1)`: Craft the item with a crafting table nearby;
- `placeItem(bot, name, position)`: Place the block at the specified position;
- `smeltItem(bot, itemName, fuelName, count = 1)`: Smelt the item with the specified fuel. There must be a furnace nearby;

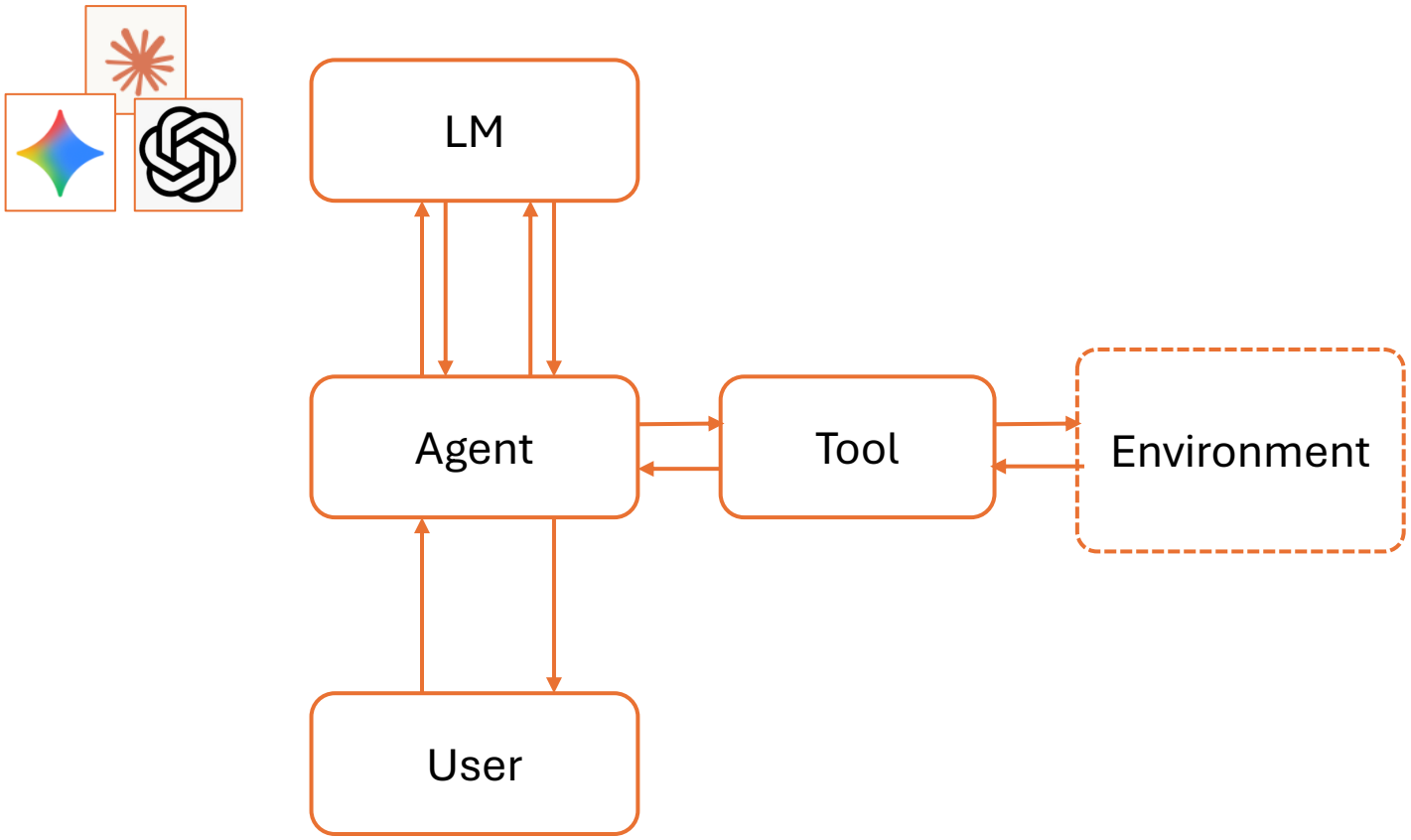

VOYAGER: An Open-Ended Embodied Agent
Automatic Curriculum
Iterative Prompting Mechanism
Skill Library
Mine Wood Log
Inventory (5/36): {'oak_planks': 3, 'stick': 4, 'crafting_table': 1, 'stone': 3, 'wooden_pickaxe': 1}
GPT-4
Reasoning: Since you have a wooden pickaxe and some stones, it would be beneficial to upgrade your pickaxe to a stone pickaxe for better efficiency.
Task: Craft 1 stone pickaxe.
Environment Feedback
Execution Error
I cannot make stick because I need: 2 more planks
throw new Error(`No item named ${name}`);
(2) Control primitive APIs implemented by us: These APIs serve a dual purpose: they demonstrate the usage of Mineflayer APIs, and they can be directly called by GPT-4.
Skill library example 1: craftWoodenPlanks.
async function craftWoodenPlanks(bot) {
const logNames = ["oak_log", "birch_log", "spruce_log", "jungle_log", "acacia_log", "dark_oak_log", "mangrove_log"];
const plankNames = ["oak_planks", "birch_planks", "spruce_planks", "jungle_planks", "acacia_planks", "dark_oak_planks", "mangrove_planks"];
const logInInventory = logNames.find(logName => bot.inventory.count(mcData.itemsByName[logName].id) > 0);
if (!logInInventory) {
bot.chat("No wooden log in inventory. Mining a wooden log...");
await mineWoodLog(bot);
}
const logIndex = logNames.indexOf(logInInventory);
const plankName = plankNames[logIndex];
bot.chat(‘Crafting 4 ${plankName}...‘);
await craftItem(bot, plankName, 1);
bot.chat(‘4 ${plankName} crafted.‘);
}


VOYAGER: An Open-Ended Embodied Agent
Automatic Curriculum
Iterative Prompting Mechanism
Skill Library
Mine Wood Log
Inventory (5/36): {'oak_planks': 3, 'stick': 4, 'crafting_table': 1, 'stone': 3, 'wooden_pickaxe': 1}
GPT-4
Reasoning: Since you have a wooden pickaxe and some stones, it would be beneficial to upgrade your pickaxe to a stone pickaxe for better efficiency.
Task: Craft 1 stone pickaxe.
Environment Feedback
Execution Error
I cannot make stick because I need: 2 more planks
throw new Error(`No item named ${name}`);
(2) Control primitive APIs implemented by us: These APIs serve a dual purpose: they demonstrate the usage of Mineflayer APIs, and they can be directly called by GPT-4.
Skill library example 1: craftWoodenPlanks.
async function craftWoodenPlanks(bot) {
const logNames = ["oak_log", "birch_log", "spruce_log", "jungle_log", "acacia_log", "dark_oak_log", "mangrove_log"];
const plankNames = ["oak_planks", "birch_planks", "spruce_planks", "jungle_planks", "acacia_planks", "dark_oak_planks", "mangrove_planks"];
const logInInventory = logNames.find(logName => bot.inventory.count(mcData.itemsByName[logName].id) > 0);
if (!logInInventory) {
bot.chat("No wooden log in inventory. Mining a wooden log...");
await mineWoodLog(bot);
}
const logIndex = logNames.indexOf(logInInventory);
const plankName = plankNames[logIndex];
bot.chat(‘Crafting 4 ${plankName}...‘);
await craftItem(bot, plankName, 1);
bot.chat(‘4 ${plankName} crafted.‘);
}

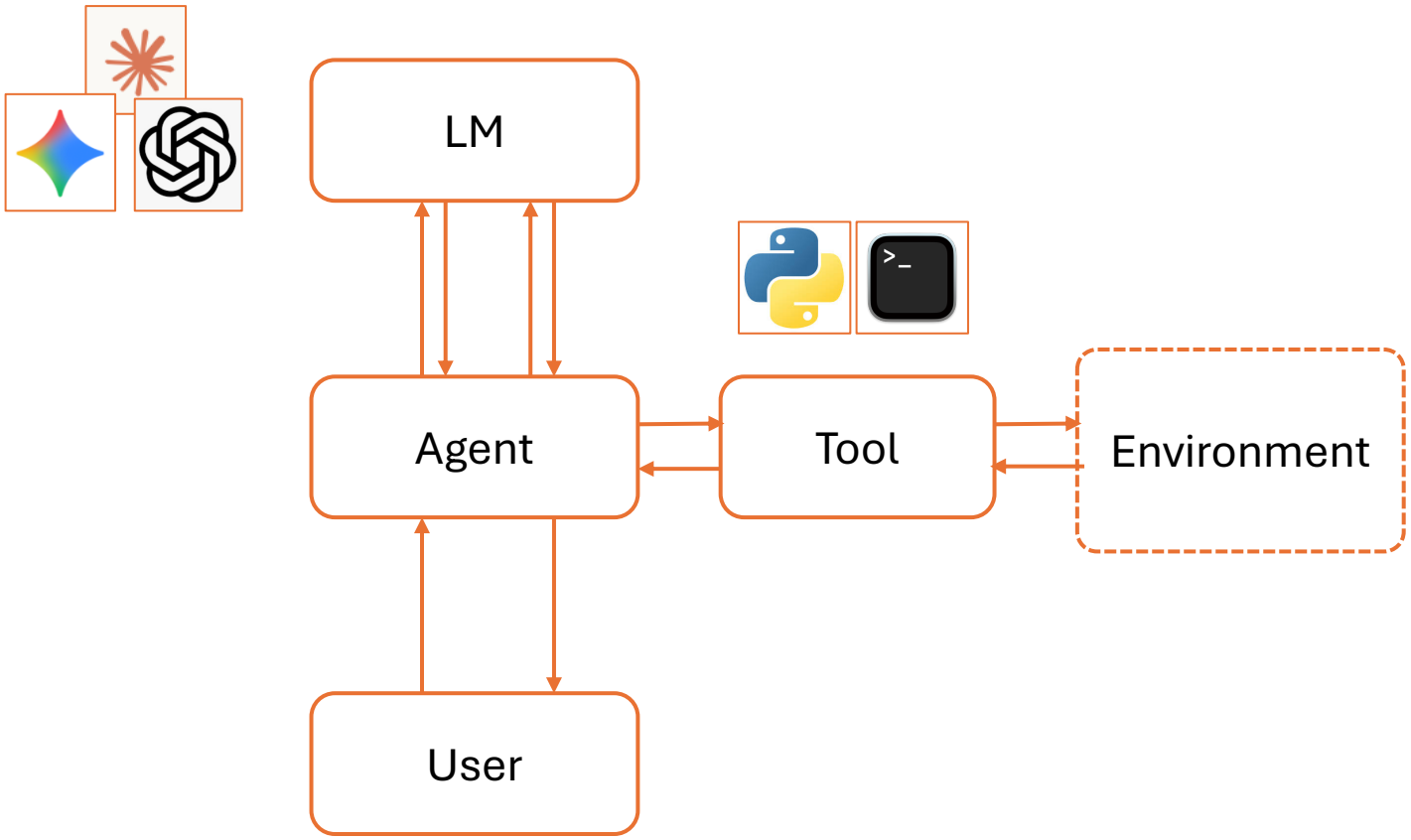
Implication for Program Synthesis



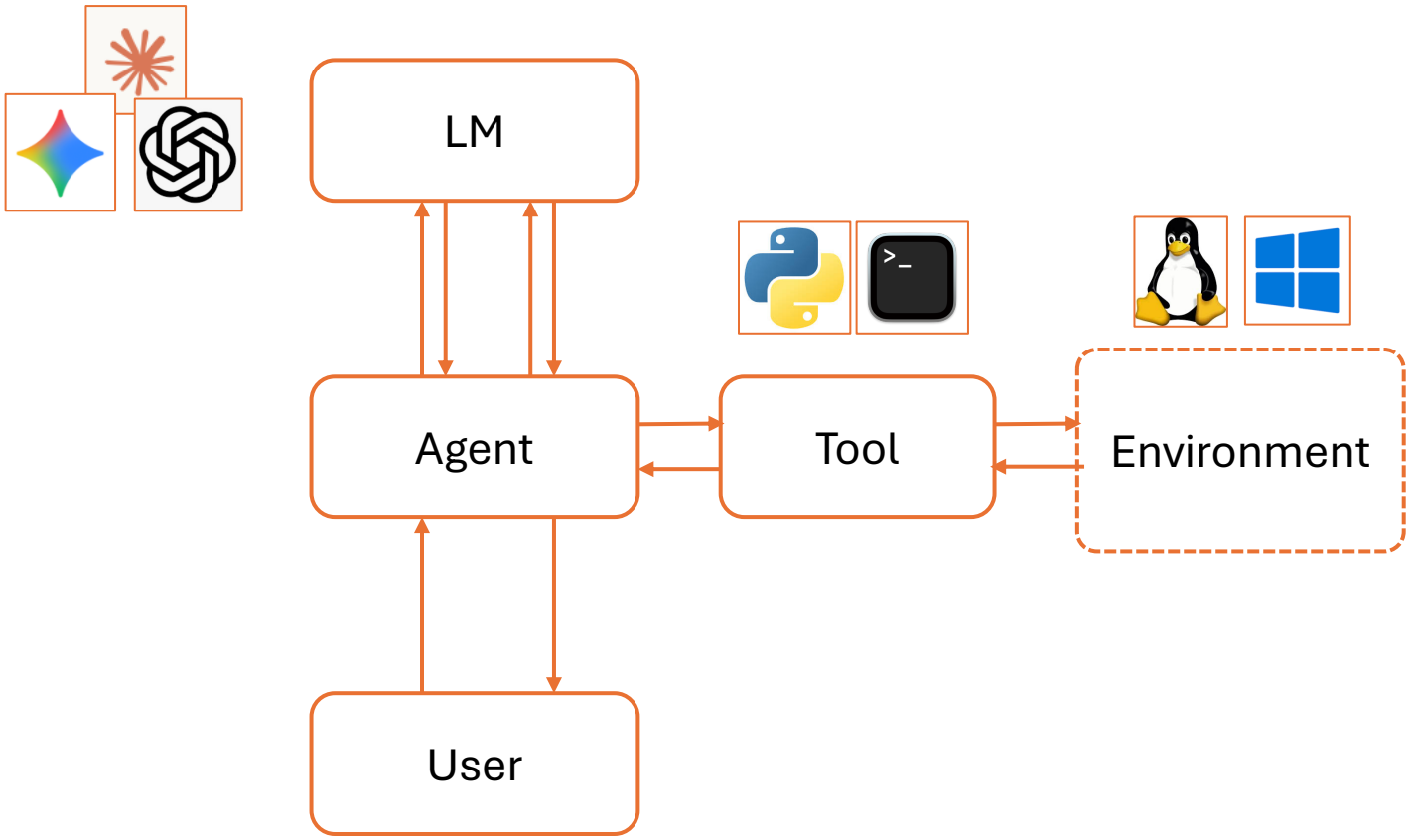
Implication for Program Synthesis



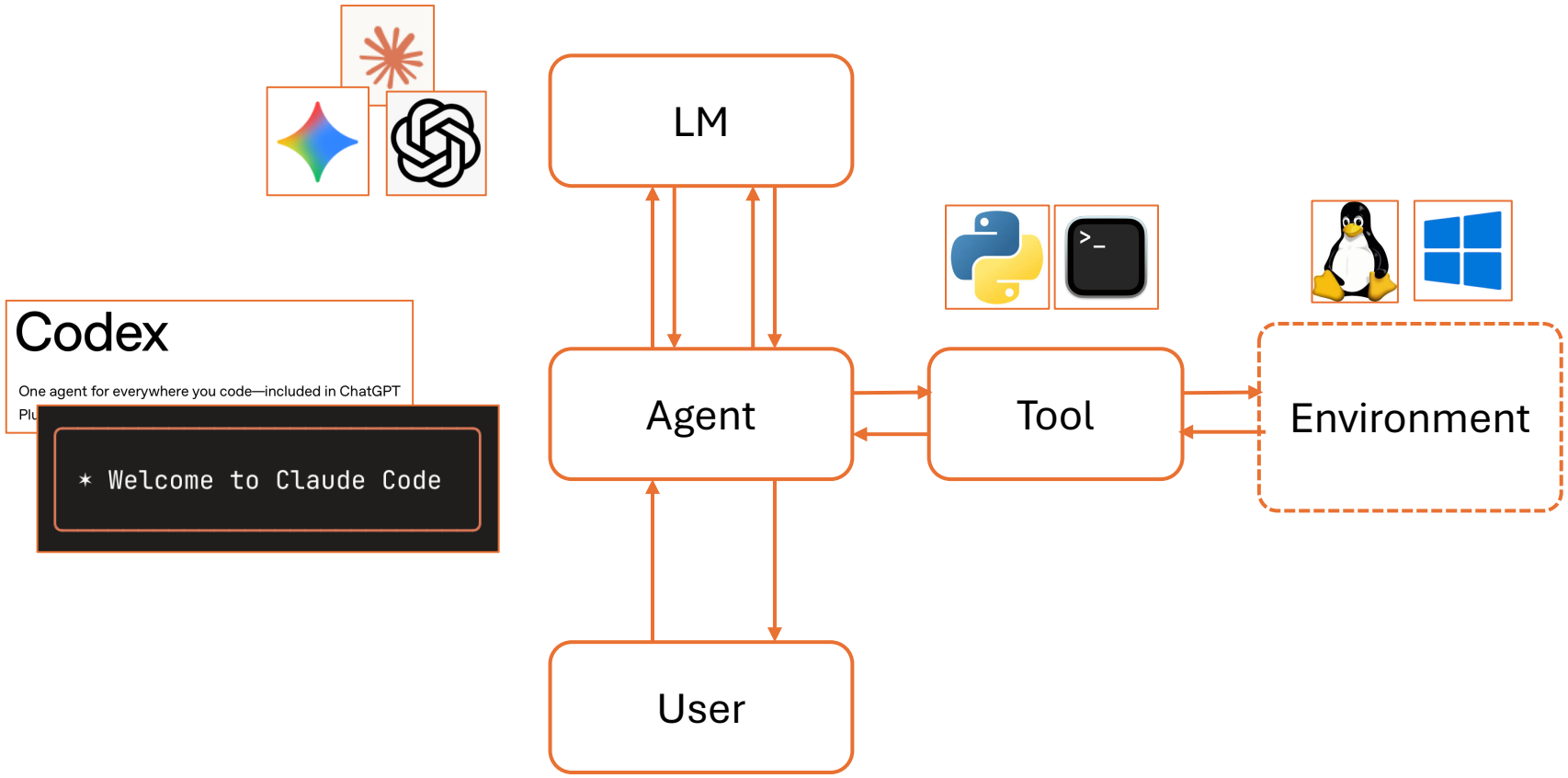
Implication for Program Synthesis



Implication for Program Synthesis

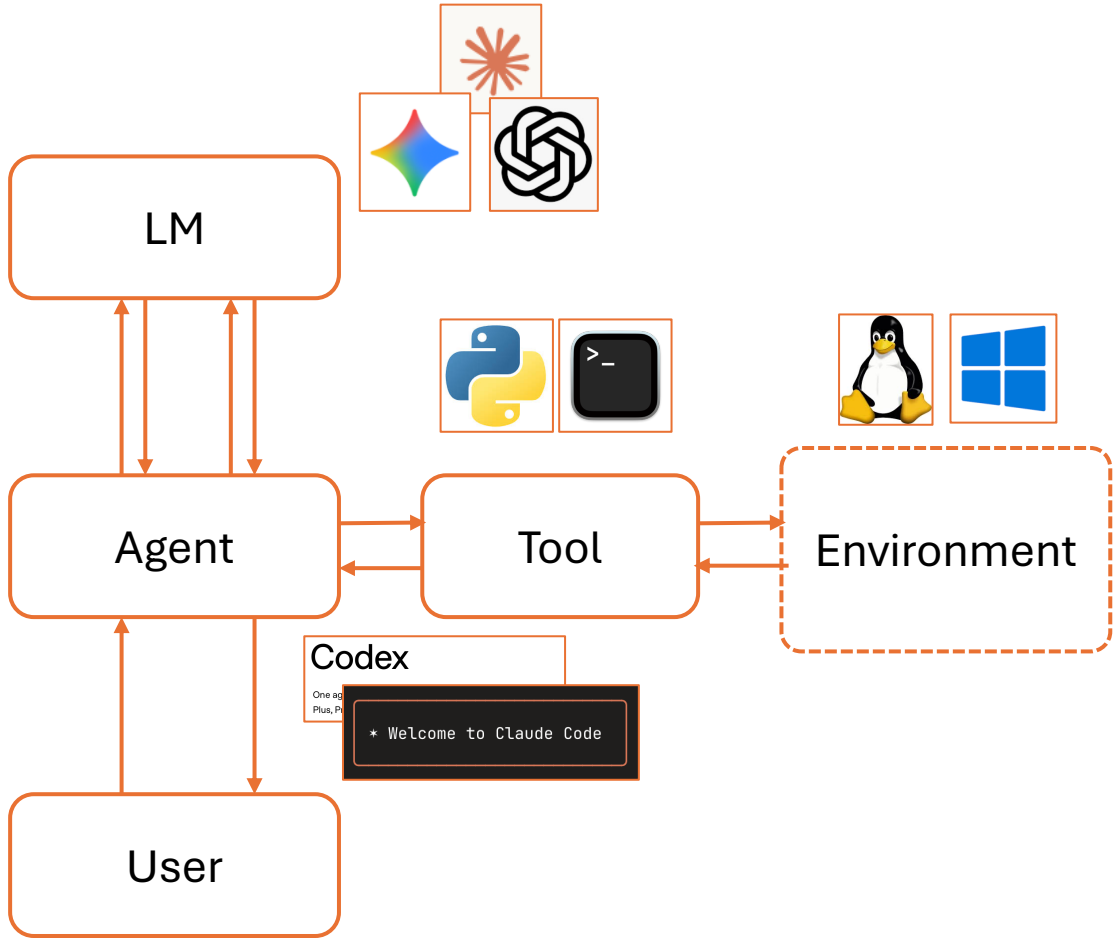


Implication for Program Synthesis



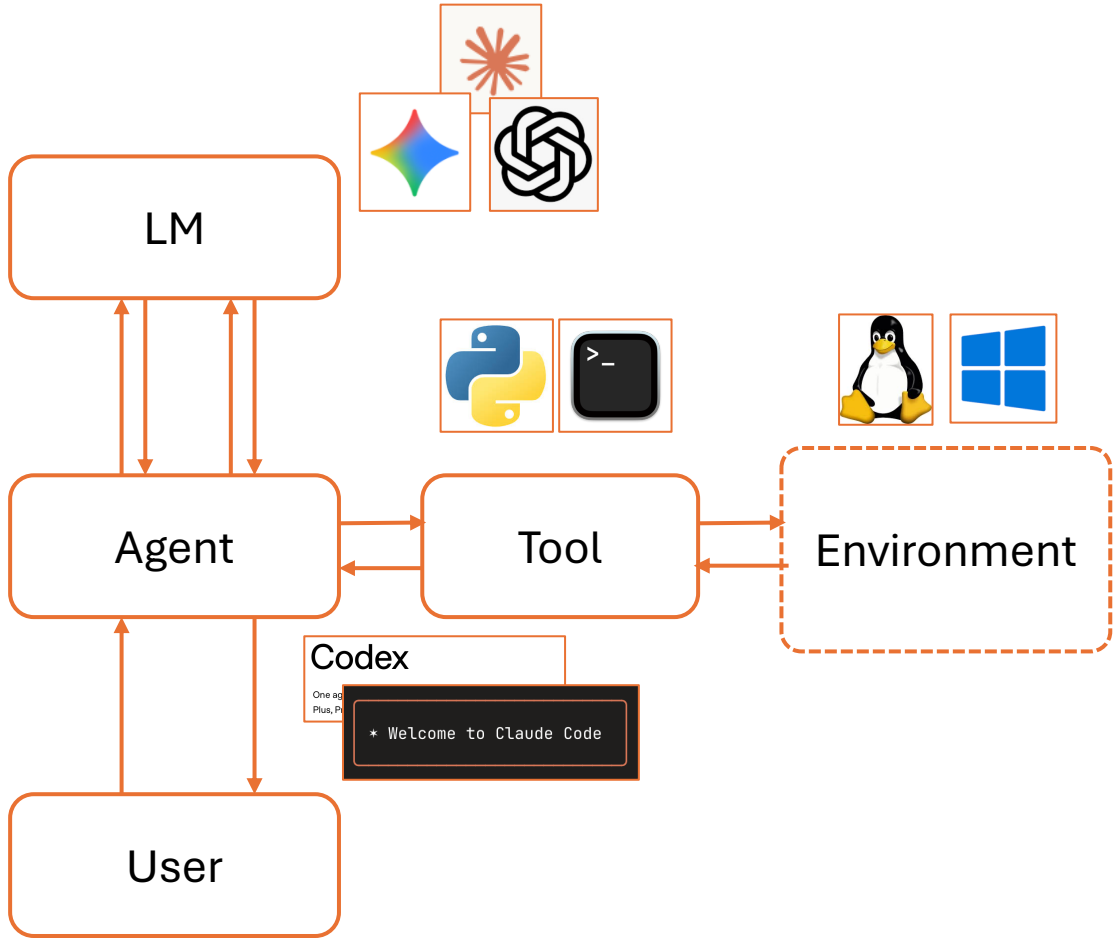
Implication for Program Synthesis

- Less input required from user
  - Agent pre-defines a set of general prompts for typical synthesis use cases;
  - Feedback can come from tools and environments.
- Less responsibility from user
  - Agent exposes the set of tools to language models;
  - Agent takes the responsibility to invoke the tools and process the results.

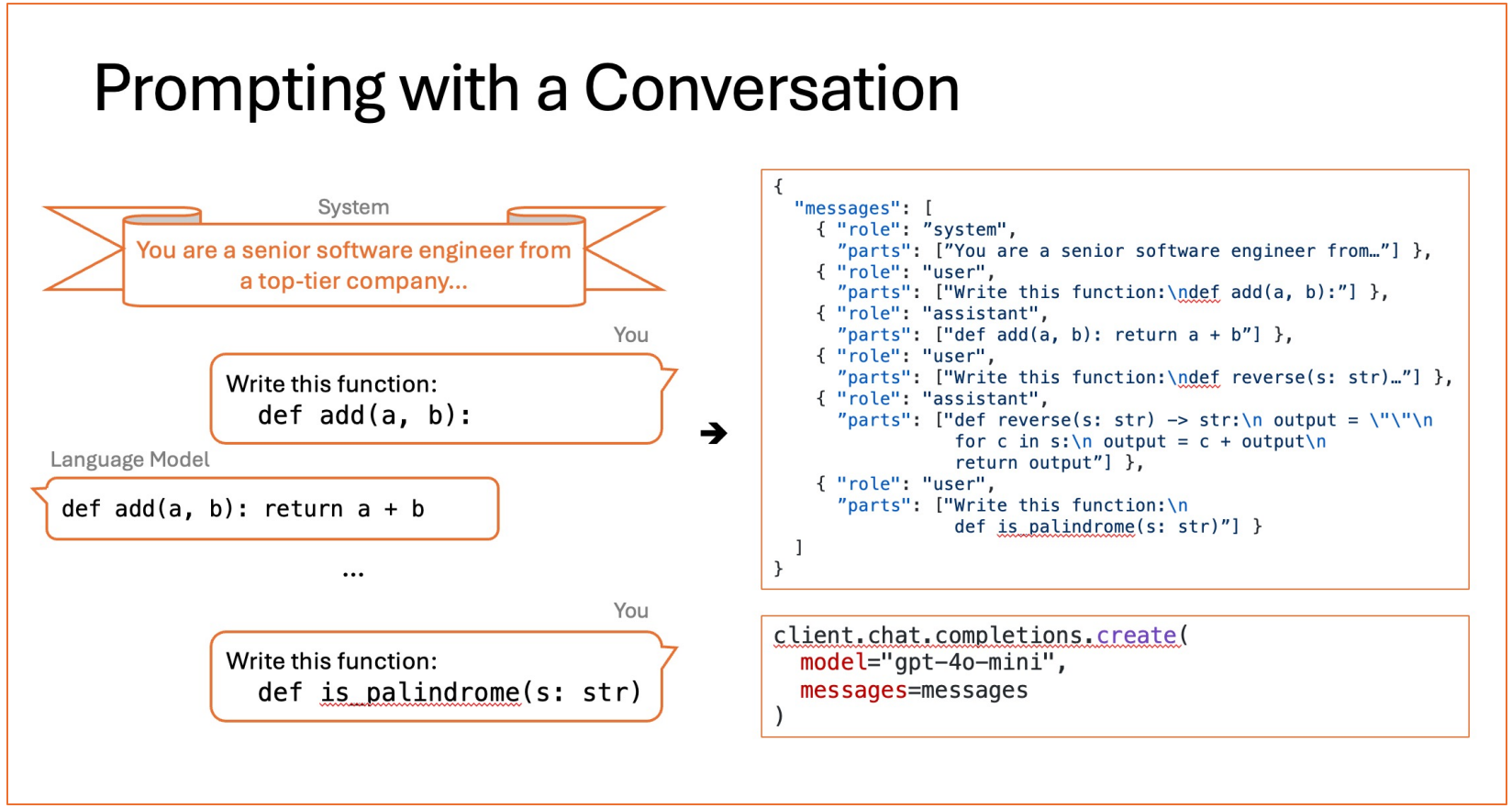


Potential Achievements for Program Synthesis

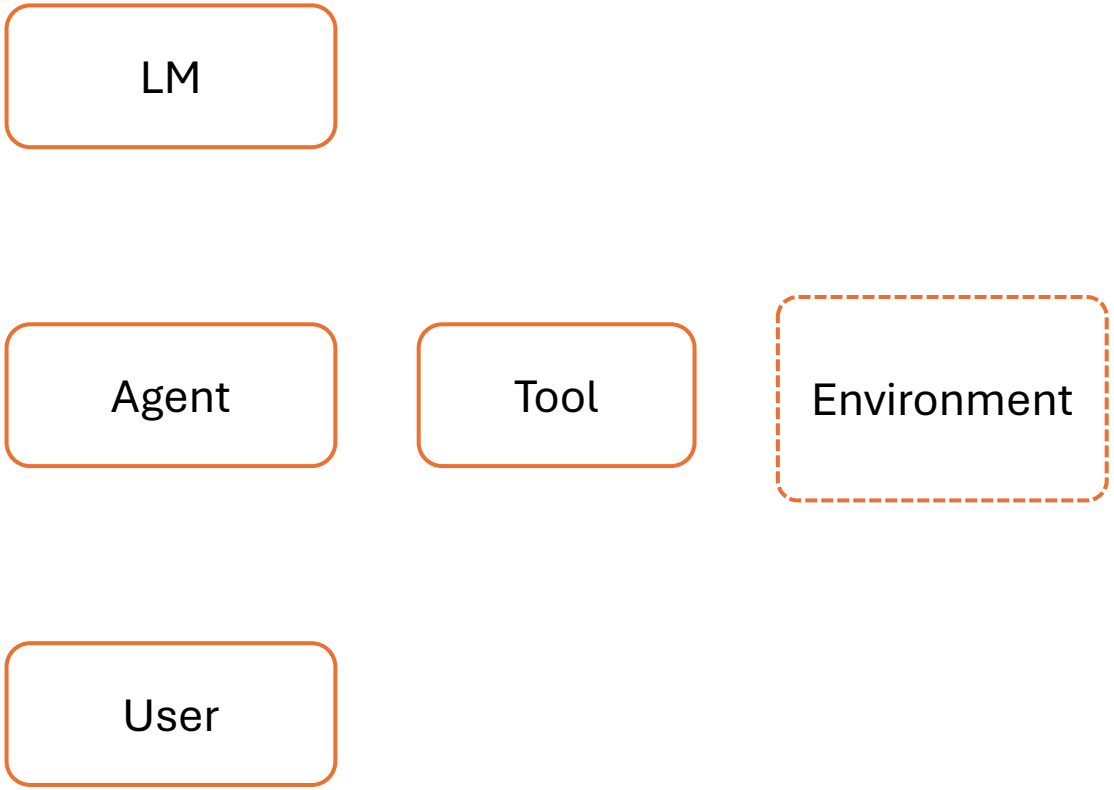
- Automated testing
  - Writing and run new test cases
- Repository level reasoning
  - Through file systems, read multiple files in the repository
- Adaptive iterative refinement
  - Dynamically choose which tool to run (compiler, linter, test, etc.)
  - Creative ways to debug: delta debugging, tracing, etc.



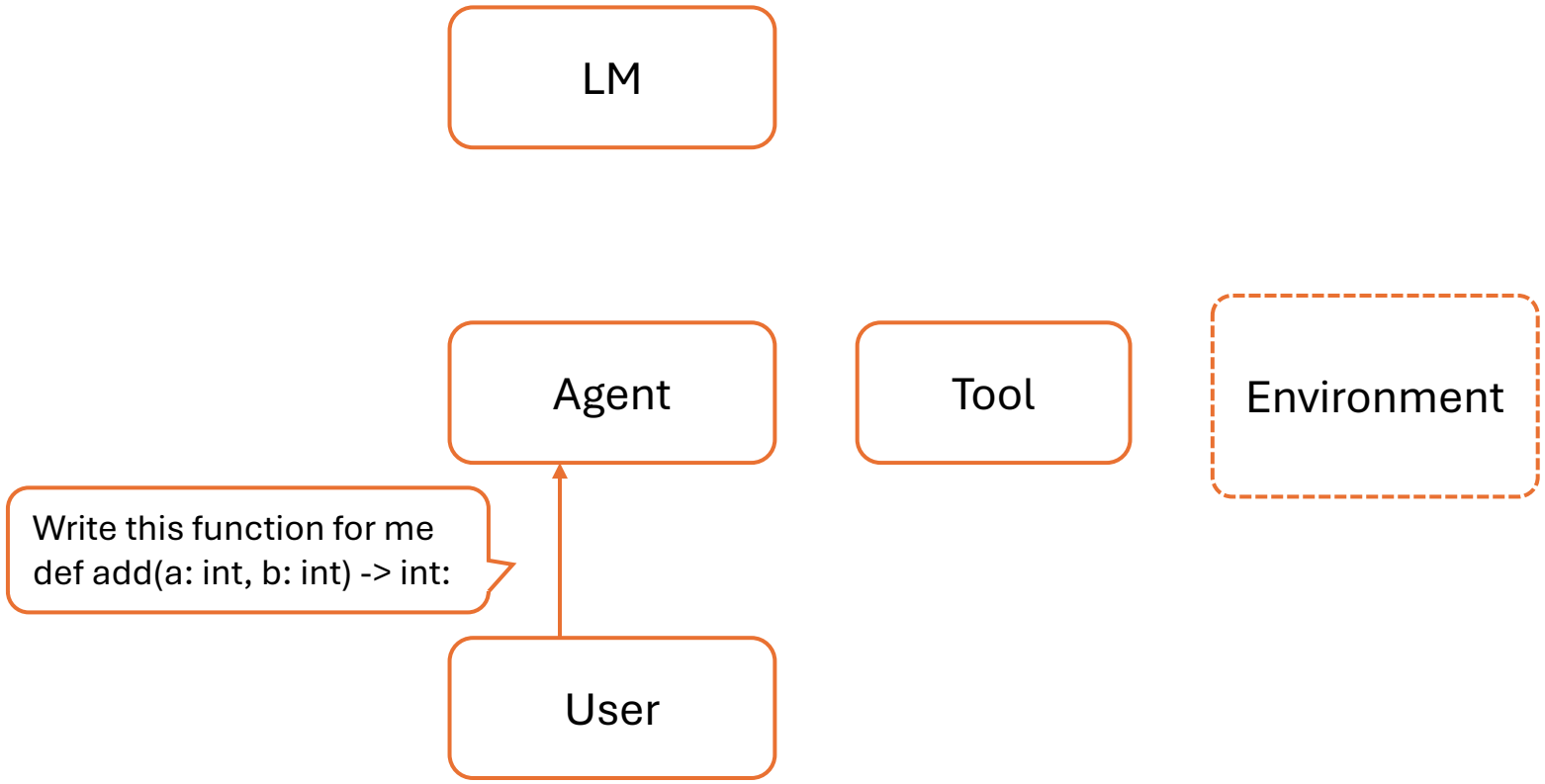
Design and Implementation of Agentic Frameworks



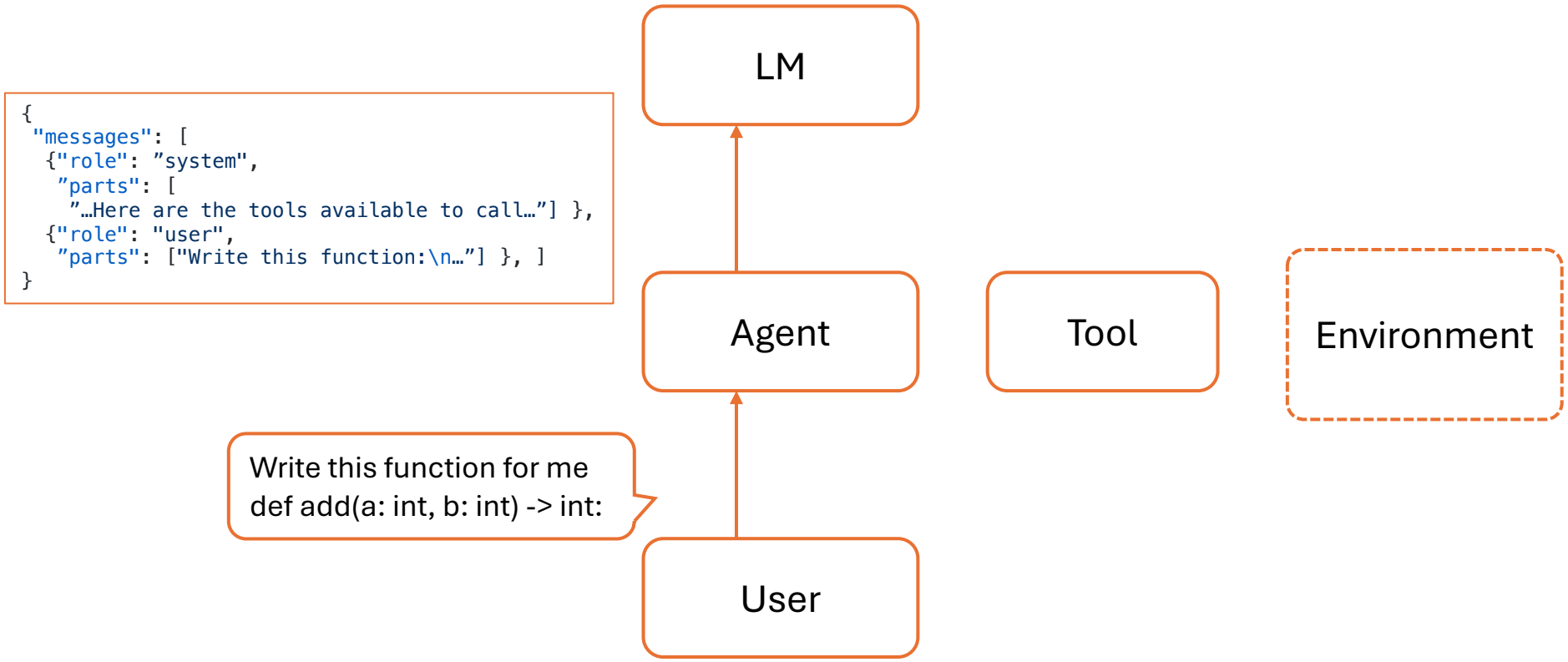
Design and Implementation of Agentic Frameworks



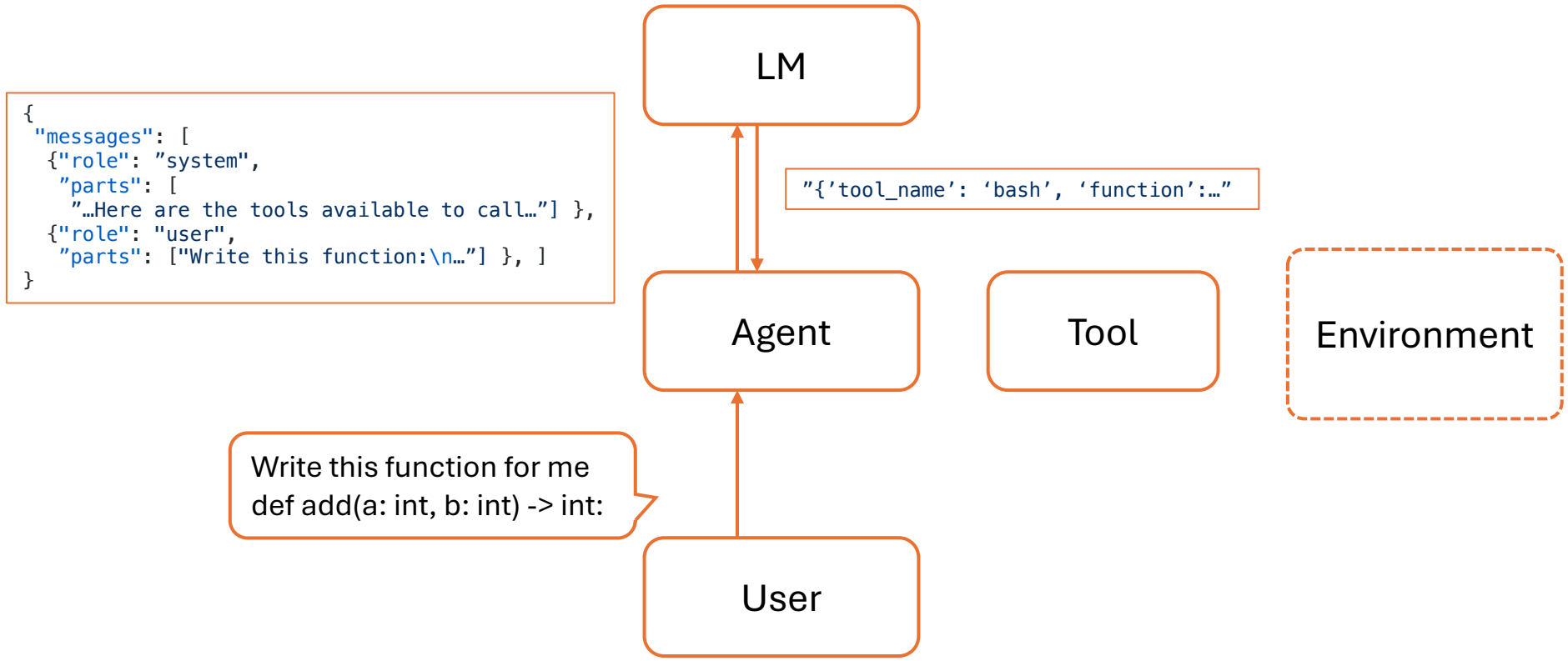
Design and Implementation of Agentic Frameworks



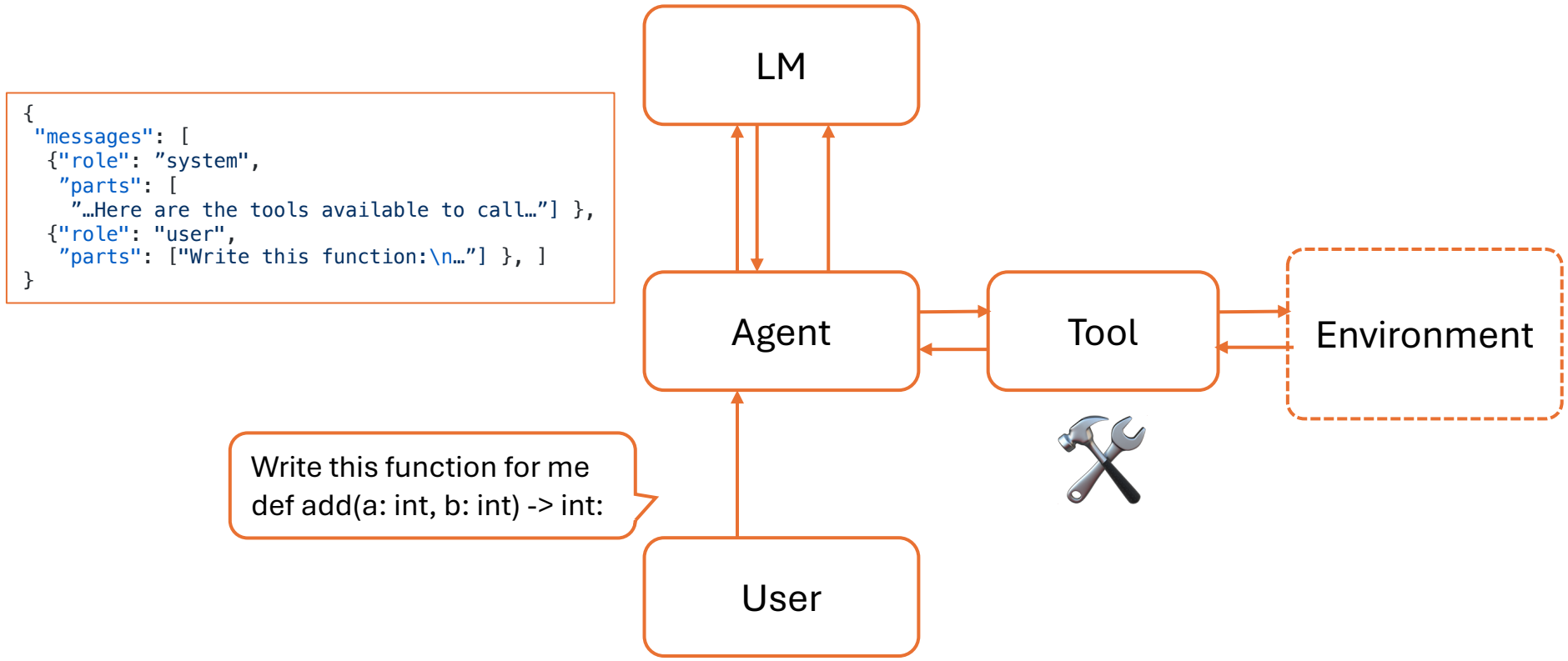
Design and Implementation of Agentic Frameworks



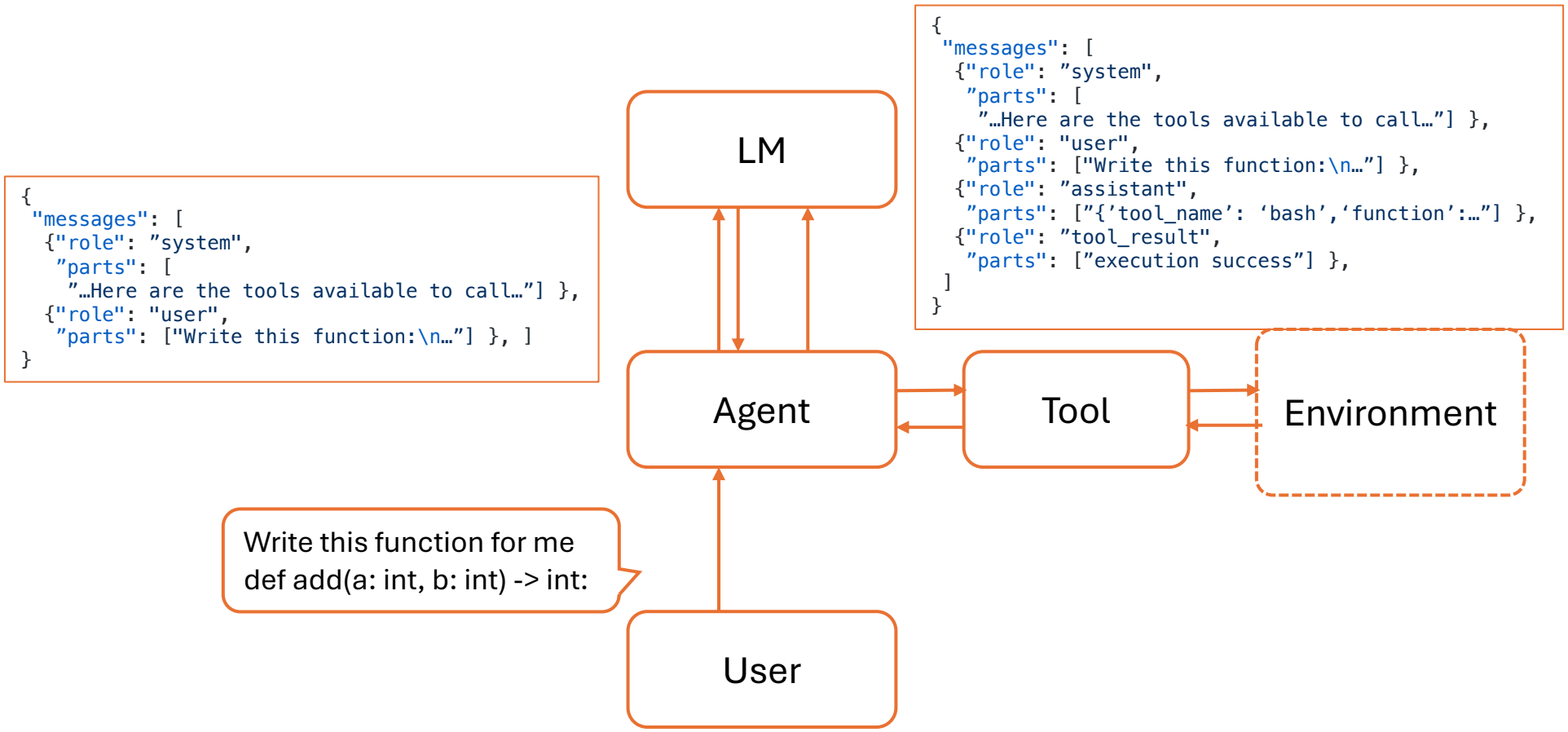
Design and Implementation of Agentic Frameworks



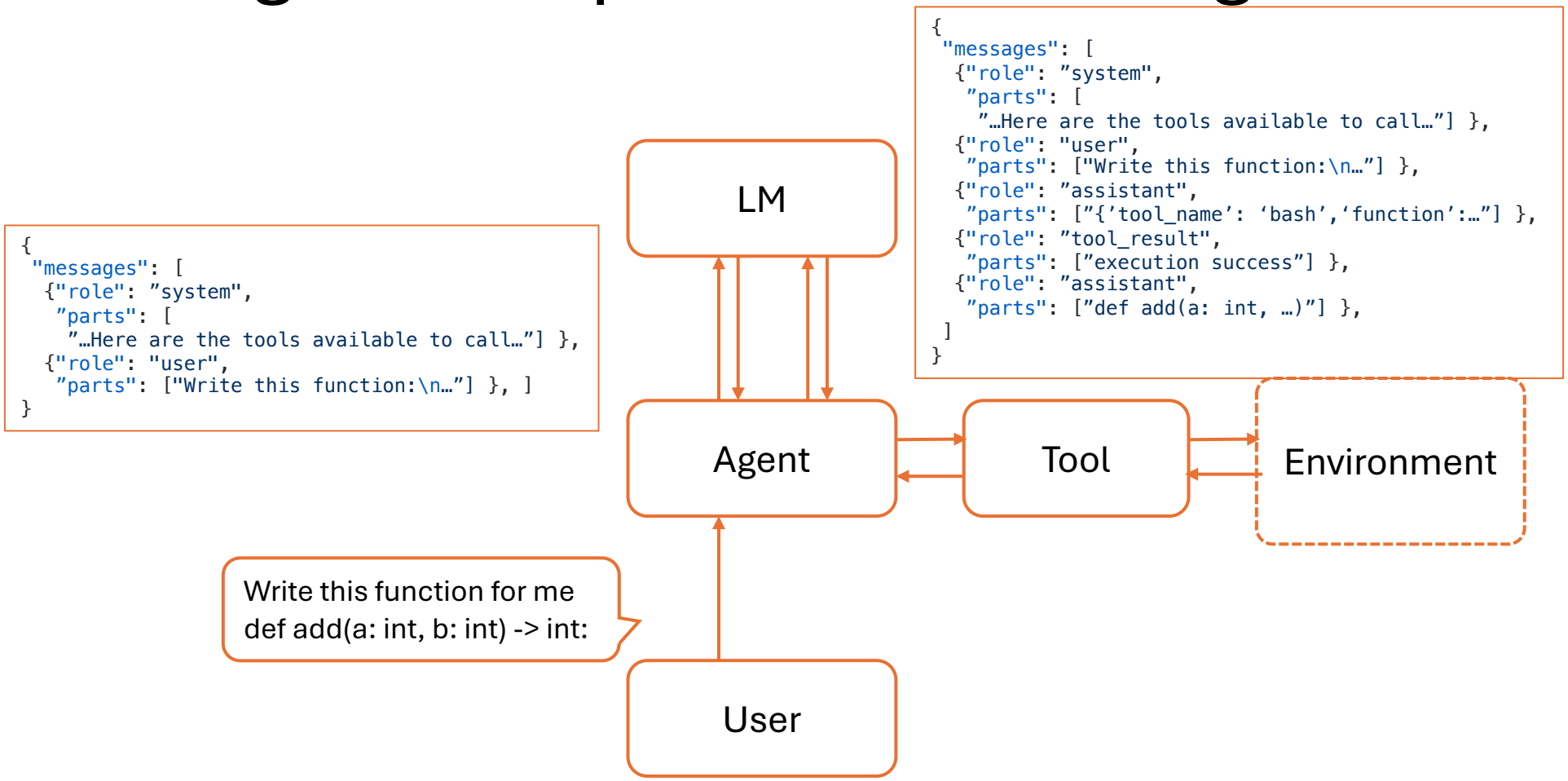
Design and Implementation of Agentic Frameworks



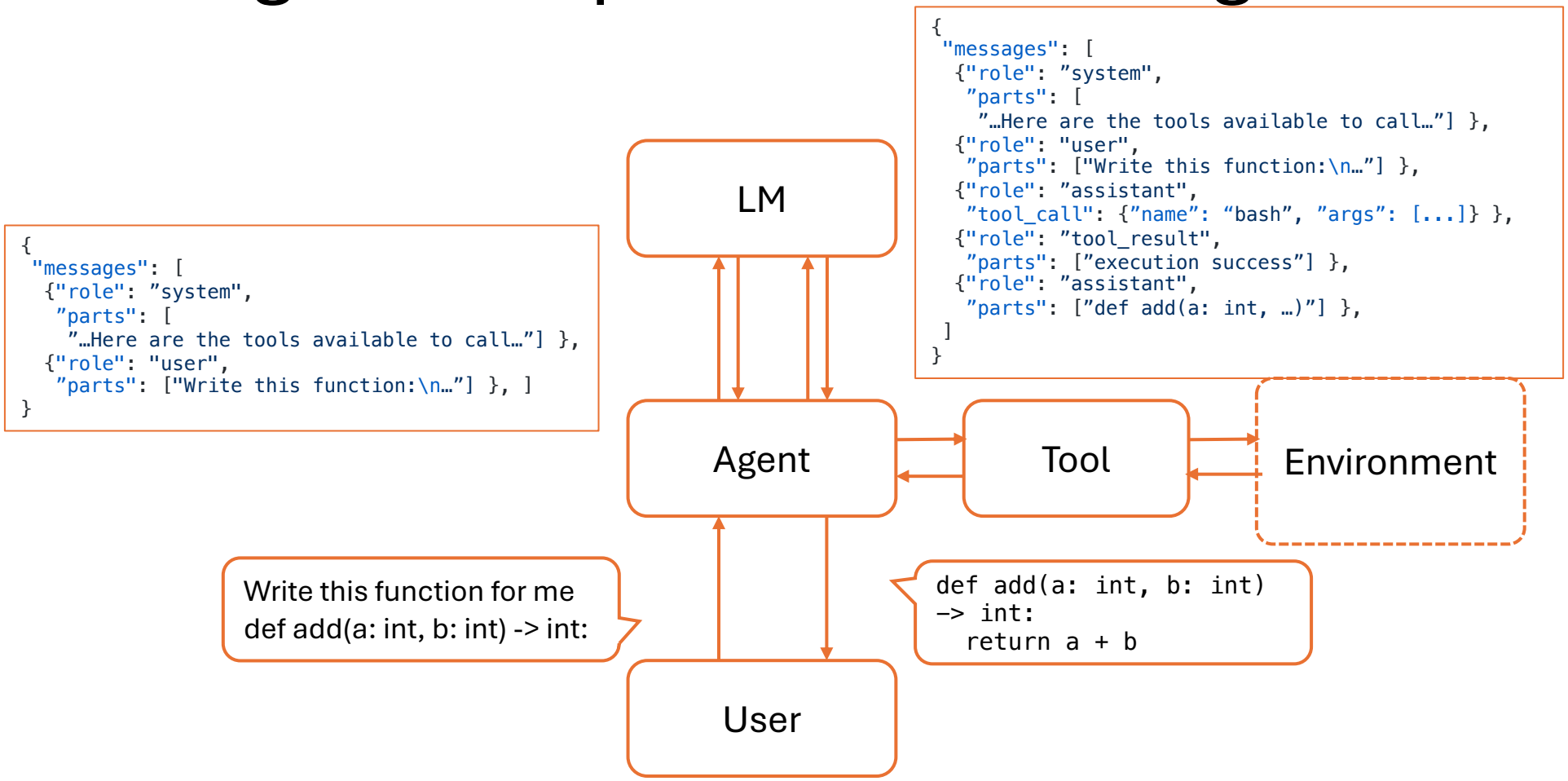
Design and Implementation of Agentic Frameworks



Design and Implementation of Agentic Frameworks



Design and Implementation of Agentic Frameworks



Common Protocol for Agentic Framework

- Protocol
  - A protocol is a precisely defined set of rules and formats that govern how entities in a system communicate and interact.
- Having a protocol provides the following benefits
  - Tools are easier to develop
    - All tools expose themselves to the agent via a unified format
    - All tools respond to the agent via a unified format
    - Online registry of tools; Similar to pip (Python's package manager)
  - Models are easier to develop
    - Can be trained/fine-tuned on this unified format to provide maximum adaptability
  - Agents are easier to develop
    - Assume all tools and models agree on the same format
    - Prompts and pipelines designed for this format

modelcontextprotocol

Type / to search

Overview | Repositories 25 | Discussions | Projects 7 | Packages | People 22

Model Context Protocol

An open protocol that enables seamless integration between LLM applications and external data sources and tools.

Verified

36.5k followers https://modelcontextprotocol.io

README.md

Model Context Protocol

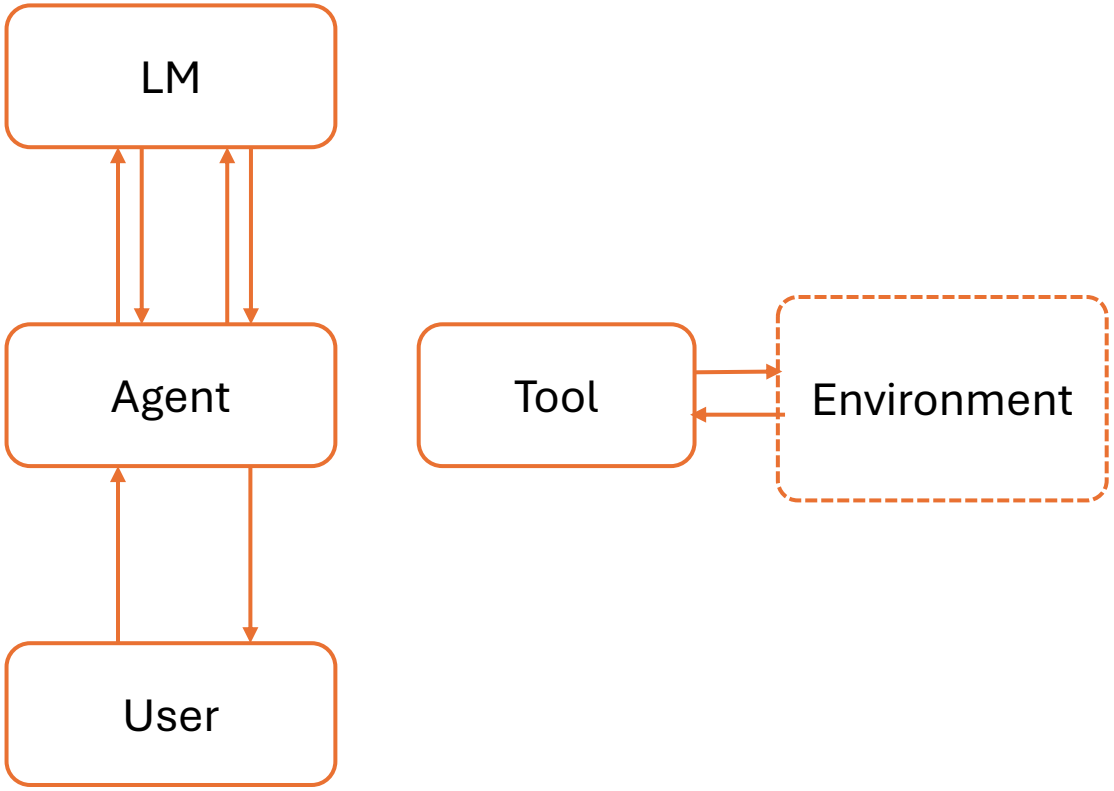
Model Context Protocol

A protocol for seamless integration between LLM applications and external data sources

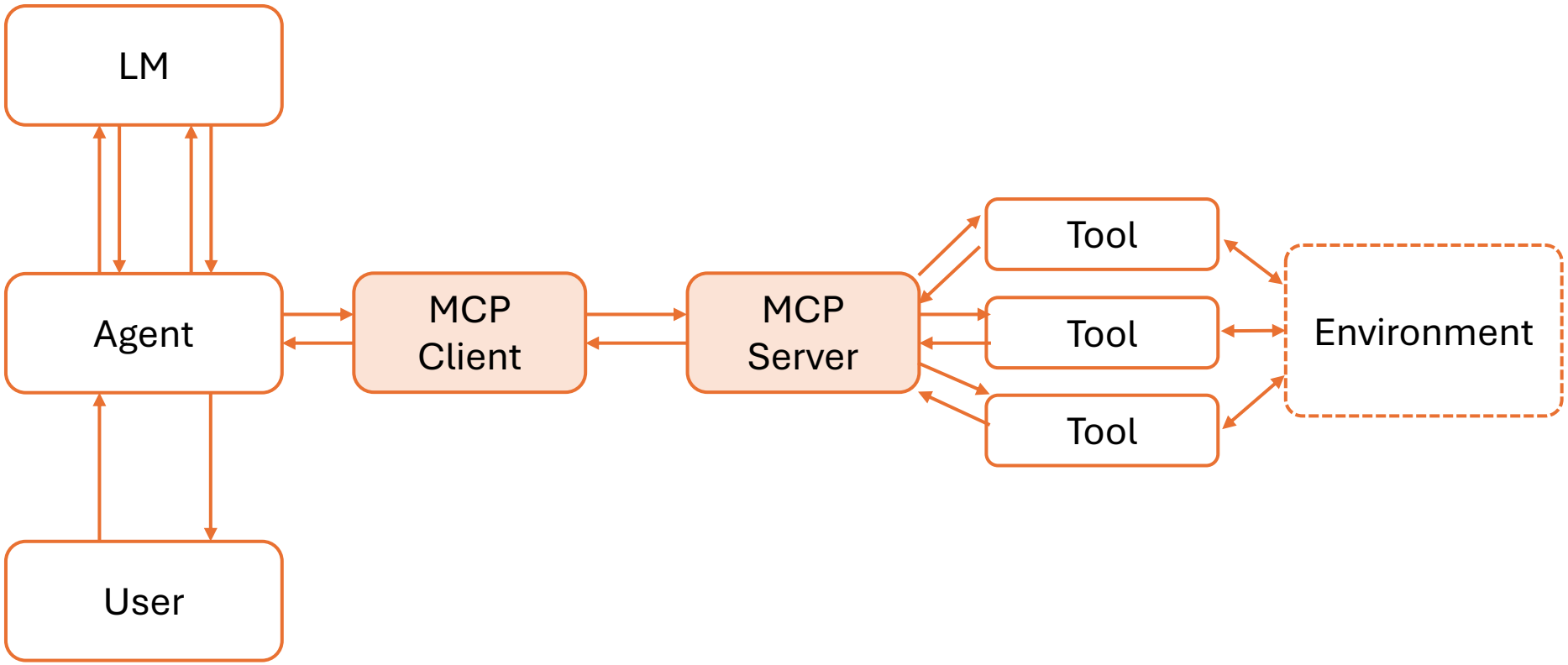
Documentation | Specification | Discussions

The Model Context Protocol (MCP) is an open protocol that enables seamless integration between LLM applications and external data sources and tools. Whether you're building an AI-powered IDE, enhancing a chat interface, or creating custom AI workflows, MCP provides a standardized way to connect LLMs with the context they need.

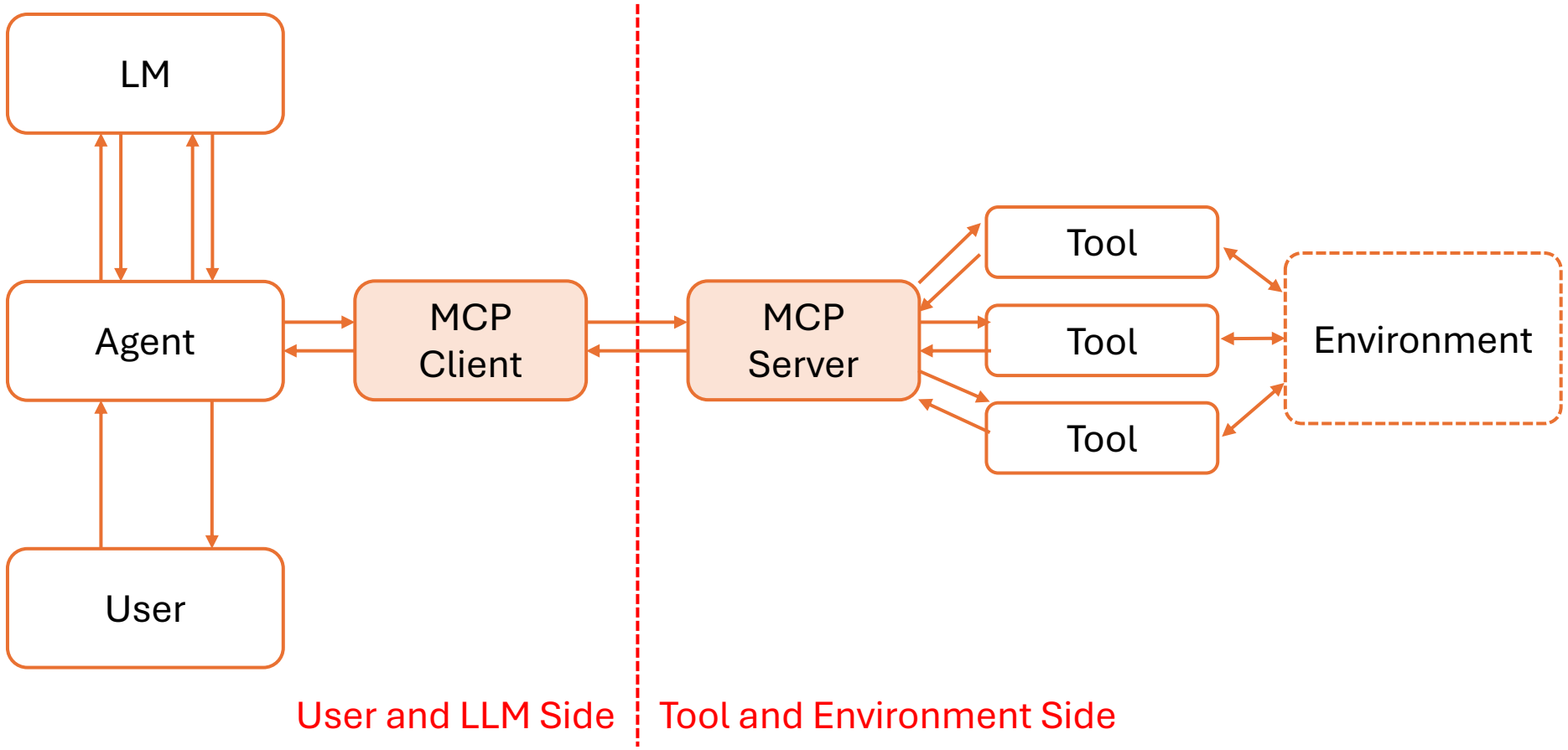
Model Context Protocol (MCP)



Model Context Protocol (MCP)



Model Context Protocol (MCP)



Model Context Protocol servers

This repository is a collection of *reference implementations* for the Model Context Protocol (MCP), as well as references to community-built servers and additional resources.

The servers in this repository showcase the versatility and extensibility of MCP, demonstrating how it can be used to give Large Language Models (LLMs) secure, controlled access to tools and data sources. Typically, each MCP server is implemented with an MCP SDK:

Model Context Protocol servers

This repository is a collection of *reference implementations* for the Model Context Protocol (MCP), as well as references to community-built servers and additional resources.

The servers in this repositor
to give Large Language Mo
server is implemented with

🌟 Reference Servers

These servers aim to demonstrate MCP features and the official SDKs.

- **Everything** - Reference / test server with prompts, resources, and tools.
- **Fetch** - Web content fetching and conversion for efficient LLM usage.
- **Filesystem** - Secure file operations with configurable access controls.
- **Git** - Tools to read, search, and manipulate Git repositories.
- **Memory** - Knowledge graph-based persistent memory system.
- **Sequential Thinking** - Dynamic and reflective problem-solving through thought sequences.
- **Time** - Time and timezone conversion capabilities.

Model Context Protocol servers

This repo ... reference ...

The serve ... to give La ... server is i ...

🌟 Reference Servers

These serve ...

- Everyth ...
- Fetch - ...
- Filesys ...
- Git - To ...
- Memor ...
- Sequer ...
- Time - ...

Archived

The following reference servers are now archived and can be found at servers-archived.

- **AWS KB Retrieval** - Retrieval from AWS Knowledge Base using Bedrock Agent Runtime.
- **Brave Search** - Web and local search using Brave's Search API. Has been replaced by the official server.
- **EverArt** - AI image generation using various models.
- **GitHub** - Repository management, file operations, and GitHub API integration.
- **GitLab** - GitLab API, enabling project management.
- **Google Drive** - File access and search capabilities for Google Drive.
- **Google Maps** - Location services, directions, and place details.
- **PostgreSQL** - Read-only database access with schema inspection.
- **Puppeteer** - Browser automation and web scraping.
- **Redis** - Interact with Redis key-value stores.
- **Sentry** - Retrieving and analyzing issues from Sentry.io.
- **Slack** - Channel management and messaging capabilities. Now maintained by Zencoder
- **SQLite** - Database interaction and business intelligence capabilities.

https://github.com/modelcontextprotocol/servers

🤝 Third-Party Servers

🎖️ Official Integrations

Official integrations are maintained by companies building production ready MCP servers for their platforms.

- **21st.dev Magic** - Create crafted UI components inspired by the best 21st.dev design engineers.
- **ActionKit by Paragon** - Connect to 130+ SaaS integrations (e.g. Slack, Salesforce, Gmail) with Paragon's ActionKit API.
- **Adfin** - The only platform you need to get paid - all payments in one place, invoicing and accounting reconciliations with Adfin.
- **AgentOps** - Provide observability and tracing for debugging AI agents with AgentOps API.
- **AgentQL** - Enable AI agents to get structured data from unstructured web with AgentQL.
- **AgentRPC** - Connect to any function, any language, across network boundaries using AgentRPC.
- **Agentset** - RAG for your knowledge base connected to Agentset.
- **Aiven** - Navigate your Aiven projects and interact with the PostgreSQL®, Apache Kafka®, ClickHouse® and OpenSearch® services
- **Alation** - Unlock the power of the enterprise Data Catalog by harnessing tools provided by the Alation MCP server.
- **Alby Bitcoin Payments** - Connect any bitcoin lightning wallet to your agent to send and receive instant payments globally with your agent.
- **Algolia** - Use AI agents to provision, configure, and query your Algolia search indices.
- **Alibaba Cloud AnalyticDB for MySQL** - Connect to an AnalyticDB for MySQL cluster for getting database or table metadata, querying and analyzing data. It will be supported to add the OpenAPI for cluster operation in the future.
- **Alibaba Cloud AnalyticDB for PostgreSQL** - An MCP server to connect to AnalyticDB for PostgreSQL instances, query and analyze data.
- **Alibaba Cloud DataWorks** - A Model Context Protocol (MCP) server that provides tools for AI, allowing it to interact with the DataWorks Open API through a standardized interface. This implementation is based on the Alibaba Cloud Open API and enables AI agents to perform cloud resources operations seamlessly.
- **Alibaba Cloud OpenSearch** - This MCP server equips AI Agents with tools to interact with OpenSearch through a standardized and extensible interface.
- **Alibaba Cloud OPS** - Manage the lifecycle of your Alibaba Cloud resources with CloudOps Orchestration Service and Alibaba Cloud OpenAPI.
- **Alibaba Cloud RDS** - An MCP server designed to interact with the Alibaba Cloud RDS OpenAPI, enabling programmatic management of RDS resources via an LLM.
- **AlipayPlus** - Connect your AI Agents to AlipayPlus Checkout Payment.
- **AllVoiceLab** - An AI voice toolkit with TTS, voice cloning, and video translation, now available as an MCP server for smarter agent integration.
- **Alpaca** – Alpaca's MCP server lets you trade stocks and options, analyze market data, and build strategies through Alpaca's Trading API
- **AlphaVantage** - Connect to 100+ APIs for financial market data, including stock prices, fundamentals, and more from AlphaVantage
- **AltTester®** - Use AltTester® capabilities to connect and test your Unity or Unreal game. Write game test automation faster and smarter, using AltTester and the AltTester® MCP server.
- **Antom** - Connect your AI Agents to Antom Checkout Payment.
- **Anytype** - An MCP server enabling AI assistants to interact with Anytype - a local and collaborative wiki - to organize objects, lists, and more through natural language.
- **Apache Doris** - MCP Server For Apache Doris, an MPP-based real-time data warehouse.
- **Apache IoTDB** - MCP Server for Apache IoTDB database and its tools
- **Apache Pinot** – MCP server for running real - time analytics queries on Apache Pinot, an open-source OLAP database built for high-throughput, low-latency powering real-time applications.
- **Apify** - Use 6,000+ pre-built cloud tools to extract data from websites, e-commerce, social media, search engines, maps, and more
- **APIMatic MCP** - APIMatic MCP Server is used to validate OpenAPI specifications using APIMatic. The server processes OpenAPI files and returns validation summaries by leveraging APIMatic's API.
- **Apollo MCP Server** - Connect your GraphQL APIs to AI agents
- **Aqara MCP Server** - Control Aqara smart home devices, query status, execute scenes, and much more using natural language.
- **Archbee** - Write and publish documentation that becomes the trusted source for instant answers with AI. Stop cobbling tools and use Archbee — the first complete
- **Astra DB** - Comprehensive tools for managing collections and documents in a DataStax Astra DB NoSQL database with a full range of operations such as create, update, delete, find, and associated bulk actions.
- **Atla** - Enable AI agents to interact with the Atla API for state-of-the-art LLMJ evaluation.
- **Atlan** - The Atlan Model Context Protocol server allows you to interact with the Atlan services through multiple tools.
- **Atlassian** - Securely interact with Jira work items and Confluence pages, and search across both.
- **AtomGit** - Official AtomGit server for integration with repository management, PRs, issues, branches, labels, and more.
- **Audiense Insights** - Marketing insights and audience analysis from Audiense reports, covering demographic, cultural, influencer, and content engagement analysis.
- **Auth0** - MCP server for interacting with your Auth0 tenant, supporting creating and modifying actions, applications, forms, logs, resource servers, and more.
- **Authenticator App · 2FA** - A secure MCP (Model Context Protocol) server that enables AI agents to interact with the Authenticator App.
- **AWS** - Specialized MCP servers that bring AWS best practices directly to your development workflow.
- **Axiom** - Query and analyze your Axiom logs, traces, and all other event data in natural language
- **Azure** - The Azure MCP Server gives MCP Clients access to key Azure services and tools like Azure Storage, Cosmos DB, the Azure CLI, and more.
- **Azure DevOps** - Interact with Azure DevOps services like repositories, work items, builds, releases, test plans, and code search.
- **Backdocket** - Search, Retrieve, and Update your Backdocket data. This currently includes Claims, Matters, Contacts, Tasks and Advanced Searches. To easily use the Remote Mcp Server utilize the following url: **https://ai.backdocket.com/mcp**
- **Baidu Map** - Baidu Map MCP Server provides tools for AI agents to interact with Baidu Maps APIs, enabling location-based services and geospatial data analysis.
- **Bankless Onchain** - Query Onchain data, like ERC20 tokens, transaction history, smart contract state.
- **Baserow** - Query data from Baserow self-hosted or SaaS databases using MCP integration.
- **BICScan** - Risk score / asset holdings of EVM blockchain address (EOA, CA, ENS) and even domain names.
- **Bitrise** - Chat with your builds, CI, and more.
- **Boikot** - Learn about the ethical and unethical actions of major companies with boikot.xyz.
- **BoldSign** - Search, request, and manage e-signature contracts effortlessly with BoldSign.
- **Boost.space** - An MCP server integrating with Boost.space for centralized, automated business data from 2000+ sources.
- **Box** - Interact with the Intelligent Content Management platform through Box AI.
- **BrightData** - Discover, extract, and interact with the web - one interface powering automated access across the public internet.
- **Browserbase** - Automate browser interactions in the cloud (e.g. web navigation, data extraction, form filling, and more)
- **BrowserStack** - Access BrowserStack's Test Platform to debug, write and fix tests, do accessibility testing and more.
- **Buildkite** - Exposing Buildkite data (pipelines, builds, jobs, tests) to AI tooling and editors.
- **Buildable** (TypeScript) - Official MCP server for Buildable AI-powered development platform. Enables AI assistants to manage tasks, track progress, get project context, and collaborate with humans on software projects.
- **BuiltWith** - Identify the technology stack behind any website.
- **Burp Suite** - MCP Server extension allowing AI clients to connect to Burp Suite
- **Cal.com** - Connect to the Cal.com API to schedule and manage bookings and appointments.
- **Campertunity** - Search campgrounds around the world on campertunity, check availability, and provide booking links.
- **Canva** — Provide AI - powered development assistance for Canva apps and integrations.
- **Carbon Voice** - MCP Server that connects AI Agents to Carbon Voice. Create, manage, and interact with voice messages, conversations, direct messages, folders, voice memos, AI actions and more in Carbon Voice.
- **Cartesia** - Connect to the Cartesia voice platform to perform text-to-speech, voice cloning etc.
- **Cashfree** - Cashfree Payments official MCP server.
- **CB Insights** - Use the CB Insights MCP Server to connect to ChatCBI
- **Cloudinary** - Exposes Cloudinary's media upload, transformation, AI analysis, management, optimization and delivery as tools usable by AI agents
- **Cloudsway SmartSearch** - Web search MCP server powered by Cloudsway, supporting keyword search, language, and safety options. Returns structured JSON results.
- **Codacy** - Interact with Codacy API to query code quality issues, vulnerabilities, and coverage insights about your code.
- **CodeLogic** - Interact with CodeLogic, a Software Intelligence platform that graphs complex code and data architecture dependencies, to boost AI accuracy and insight.
- **CoinGecko** - Official CoinGecko API MCP Server for Crypto Price & Market Data, across 200+ Blockchain Networks and 8M+ Tokens.
- **Comet Opik** - Query and analyze your Opik logs, traces, prompts and all other telemetry data from your LLMs in natural language.
- **Conductor** - Interact with Conductor (OSS and Orkes) REST APIs.
- **Composio** – Use Composio to connect 100+ tools. Zero setup. Auth built-in. Made for agents, works for humans.
- **Confluent** - Interact with Confluent Kafka and Confluent Cloud REST APIs.
- **Contrast Security** - Brings Contrast's vulnerability and SCA data into your coding agent to quickly remediate vulnerabilities.
- **Convex** - Introspect and query your apps deployed to Convex.
- **Cortex** - Official MCP server for Cortex.
- **Couchbase** - Interact with the data stored in Couchbase clusters.
- **CRIC Wuye AI** - Interact with capabilities of the CRIC Wuye AI platform, an intelligent assistant specifically for the property management industry.
- **CrowdStrike Falcon** - Connects AI agents with the CrowdStrike Falcon platform for intelligent security analysis, providing programmatic access to detections, incidents, behaviors, threat intelligence, hosts, vulnerabilities, and identity protection capabilities.
- **CTERA Edge Filer** - CTERA Edge Filer delivers intelligent edge caching and multiprotocol file access, enabling fast, secure access to files across core and remote sites.
- **CTERA Portal** - CTERA Portal is a multi-tenant, multi-cloud platform that delivers a global namespace and unified management across petabytes of distributed content.
- **Cycode** - Boost security in your dev lifecycle via SAST, SCA, Secrets & IaC scanning with Cycode.
- **Dart** - Interact with task, doc, and project data in Dart, an AI-native project management tool
- **Databricks** - Connect to data, AI tools & agents, and the rest of the Databricks platform using turnkey managed MCP servers. Or, host your own custom MCP servers within the Databricks security and data governance boundary.
- **DataHub** - Search your data assets, traverse data lineage, write SQL queries, and more using DataHub metadata.
- **Daytona** - Fast and secure execution of your AI generated code with Daytona sandboxes
- **Debugg.AI** - Zero-Config, Fully AI-Managed End-to-End Testing for any code gen platform via Debugg.AI remote browsing test agents.
- **DeepL** - Translate or rewrite text with DeepL's very own AI models using the DeepL API
- **Defang** - Deploy your project to the cloud seamlessly with the Defang platform without leaving your integrated development environment
- **Detailer** – Instantly generate rich, AI-powered documentation for your GitHub repositories. Designed for AI agents to gain deep project context before taking action.
- **DevCycle** - Create and monitor feature flags using natural language in your AI coding assistant.
- **DevHub** - Manage and utilize website content within the DevHub CMS platform
- **DevRev** - An MCP server to integrate with DevRev APIs to search through your DevRev Knowledge Graph where objects can be imported from diff. Sources listed here.
- **DexPaprika (CoinPaprika)** - Access real-time DEX data, liquidity pools, token information, and trading analytics across multiple blockchain networks with DexPaprika by CoinPaprika.
- **Dolt** - The official MCP server for version-controlled Dolt databases.
- **Dot (GetDot.ai)** - Fetch, analyze or visualize data from your favorite database or data warehouse (Snowflake, BigQuery, Redshift, Databricks, Clickhouse, ...) with Dot, your AI Data Analyst. This remote MCP server is a one-click integration for user that have setup Dot.
- **Drata** - Get hands-on with our experimental MCP server—bringing real-time compliance intelligence into your AI workflows.
- **Dumpling AI** - Access data, web scraping, and document conversion APIs by Dumpling AI
- **GitGuardian** - GitGuardian official MCP server - Scan projects using GitGuardian's industry-leading API, which features over 500 secret detectors to prevent credential leaks before they reach public repositories. Resolve security incidents directly with rich contextual data for rapid, automated remediation.
- **GitHub** - GitHub's official MCP Server.
- **GitKraken** - A CLI for interacting with GitKraken APIs. Includes an MCP server via `gk mcp` that not only wraps GitKraken APIs, but also Jira, GitHub, GitLab, and more.
- **Glean** - Enterprise search and chat using Glean's API.
- **Globalping** - Access a network of thousands of probes to run network commands like ping, traceroute, mtr, http and DNS resolve.
- **gNucleus Text-To-CAD** - Generate CAD parts and assemblies from text using gNucleus AI models.
- **Google Cloud Run** - Deploy code to Google Cloud Run
- **GoLogin MCP server** - Manage your GoLogin browser profiles and automation directly through AI conversations!
- **Google Maps Platform Code Assist** - Ground agents on fresh, official documentation and code samples for optimal geo-related guidance and code..
- **gotoHuman** - Human-in-the-loop platform - Allow AI agents and automations to send requests for approval to your gotoHuman inbox.
- **Grafana** - Search dashboards, investigate incidents and query datasources in your Grafana instance
- **Grafbase** - Turn your GraphQL API into an efficient MCP server with schema intelligence in a single command.
- **Grain** - Access your Grain meetings notes & transcripts directly in claude and generate reports with native Claude Prompts.
- **Graphlit** - Ingest anything from Slack to Gmail to podcast feeds, in addition to web crawling, into a searchable Graphlit project.
- **Gremlin** - The official Gremlin MCP server. Analyze your reliability posture, review recent tests and chaos engineering experiments, and create detailed reports.
- **GreptimeDB** - Provides AI assistants with a secure and structured way to explore and analyze data in GreptimeDB.
- **GROWI** - Official MCP Server to integrate with GROWI APIs.
- **Gyazo** - Search, fetch, upload, and interact with Gyazo images, including metadata and OCR data.
- **Harper** - An MCP server providing an interface for MCP clients to access data within Harper.
- **Heroku** - Interact with the Heroku Platform through LLM-driven tools for managing apps, add-ons, dynos, databases, and more.
- **HeyOnCall** - Page a human, sending critical or non-critical alerts to the free HeyOnCall iOS or Android apps.
- **Hiveflow** - Create, manage, and execute agentic AI workflows directly from your assistant.
- **Hive Intelligence** - Ultimate cryptocurrency MCP for AI assistants with unified access to crypto, DeFi, and Web3 analytics
- **Hologres** - Connect to a Hologres instance, get table metadata, query and analyze data.
- **Homebrew** Allows Homebrew users to run Homebrew commands locally.
- **Honeycomb** Allows Honeycomb Enterprise customers to query and analyze their data, alerts, dashboards, and more; and cross-reference production behavior with the codebase.
- **HubSpot** - Connect, manage, and interact with HubSpot CRM data
- **Hugging Face** - Connect to the Hugging Face Hub APIs programmatically: semantic search for spaces and papers, exploration of datasets and models, and access to all compatible MCP Gradio tool spaces!
- **Hunter** - Interact with the Hunter API to get B2B data using natural language.
- **Hyperbolic** - Interact with Hyperbolic's GPU cloud, enabling agents and LLMs to view and rent available GPUs, SSH into them, and run GPU-powered workloads for you.
- **Hyperbrowser** - Hyperbrowser is the next-generation platform empowering AI agents and enabling effortless, scalable browser automation.
- **IBM wxflows** - Tool platform by IBM to build, test and deploy tools for any data source
- **Inbox Zero** - AI personal assistant for email Inbox Zero
- **Inflectra Spira** - Connect to your instance of the SpiraTest, SpiraTeam or SpiraPlan application lifecycle management platform by Inflectra
- **Inkeep** - RAG Search over your content powered by Inkeep
- **Integration App** - Interact with any other SaaS applications on behalf of your customers.
- **IP2Location.io** - Interact with IP2Location.io API to retrieve the geolocation information for an IP address.
- **IPLocate** - Look up IP address geolocation, network information, detect proxies and VPNs, and find abuse contact details using IPLocate.io

MCP Safety Audit: LLMs with the Model Context Protocol Allow Major Security Exploits

Brandon Radosevich*
John T. Halloran*
Leidos
halloranjt@leidos.com

MINDGUARD: Tracking, Detecting, and Attributing MCP Tool Poisoning Attack via Decision Dependence Graph

Zhiqiang Wang*, Junyang Zhang*, Guanquan Shi†, HaoRan Cheng*, Yunhao Yao*, Kaiwen Guo‡, Haohua Du†, and Xiang-Yang Li*

sa21221041@mail.ustc.edu.cn, zhangjunyang@mail.ustc.edu.cn, shiguanquan@buaa.edu.cn, chenghaoran@mail.ustc.edu.cn, yaoyunhao@mail.ustc.edu.cn, kevinguo@ouc.edu.cn, ustc.edu.cn

iversity ‡Ocean University of China

Model Context Protocol (MCP) at First Glance: Studying the Security and Maintainability of MCP Servers

MOHAMMED MEHEDI HASAN, Queen's University, Canada
HAO LI, Queen's University, Canada
EMAD FALLAHZADEH, Queen's University, Canada
GOPI KRISHNAN RAJBAHADUR, Queen's University, Canada
BRAM ADAMS, Queen's University, Canada
AHMED E. HASSAN, Queen's University, Canada

Logistics – Week 5

- Assignment 2
  - https://github.com/machine-programming/assignment-2
  - Due next Thursday (Oct 2nd)
  - Expected to take quite some time, so please start working on it early
- Feedback form
  - Follow this link to fill the questionnaire; worth 1% of extra credit
  - Thanks to those who submitted already!
  - Help us make the course better!