

Agent框架调研：19种Agent架构对比分析

代理（Agent）指能自主感知环境并采取行动实现目标的智能体，即AI作为一个人或一个组织的代表，进行某种特定行为和交易，降低一个人或组织的工作复杂程度，减少工作量和沟通成本。

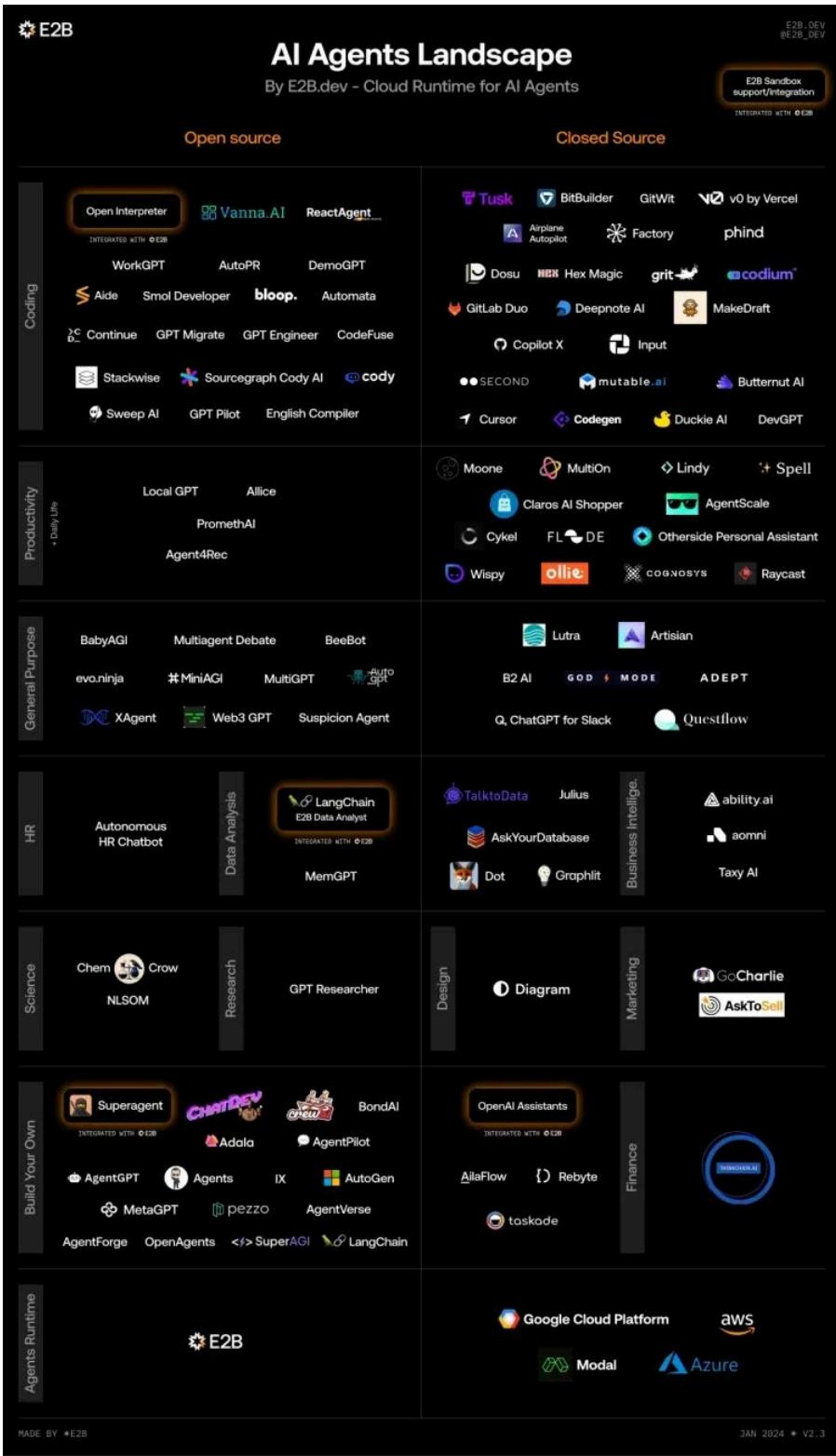


背景

目前，我们在探索Agent的应用方向，借此机会调研学习了一下现在主流的Agent框架，这篇文章也是我们调研过程的记录。

■ 网络热门Agents

截止至今日，开源的Agent应用可以说是百花齐放，文章也是挑选了热度和讨论度较高的19类Agent，基本能覆盖主流的Agent框架，每个类型都做了一个简单的summary、作为一个参考供大家学习。



图片来源: <https://github.com/e2b-dev/awesome-ai-agents>

Agent基础

Agent的核心决策逻辑是让LLM根据动态变化的环境信息选择执行具体的行动或者对结果作出判断，并影响环境，通过多轮迭代重复执行上述步骤，直到完成目标。

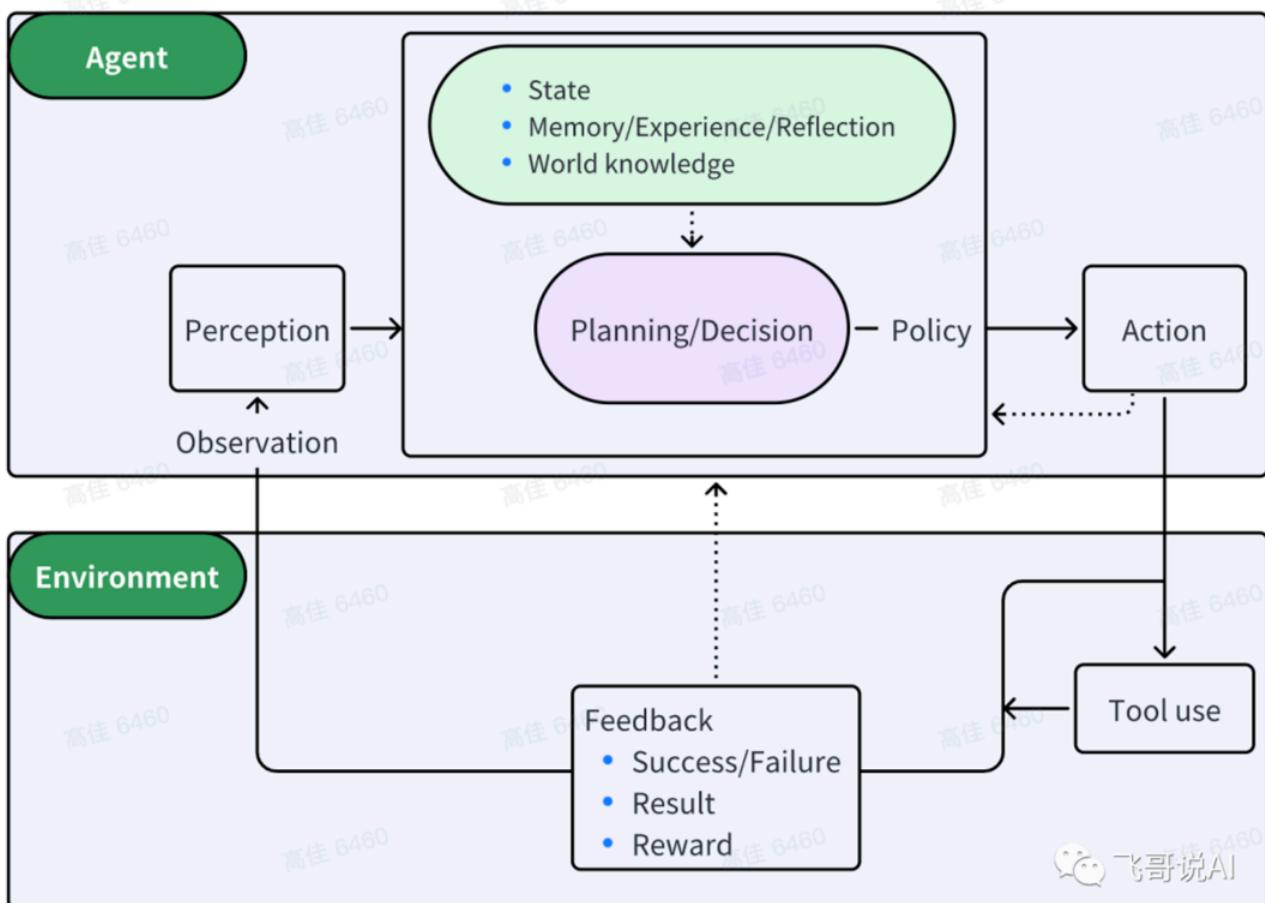
**精简的决策流程: **P (感知) → P (规划) → A (行动)

1. 感知 (Perception) 是指Agent从环境中收集信息并从中提取相关知识的能力。

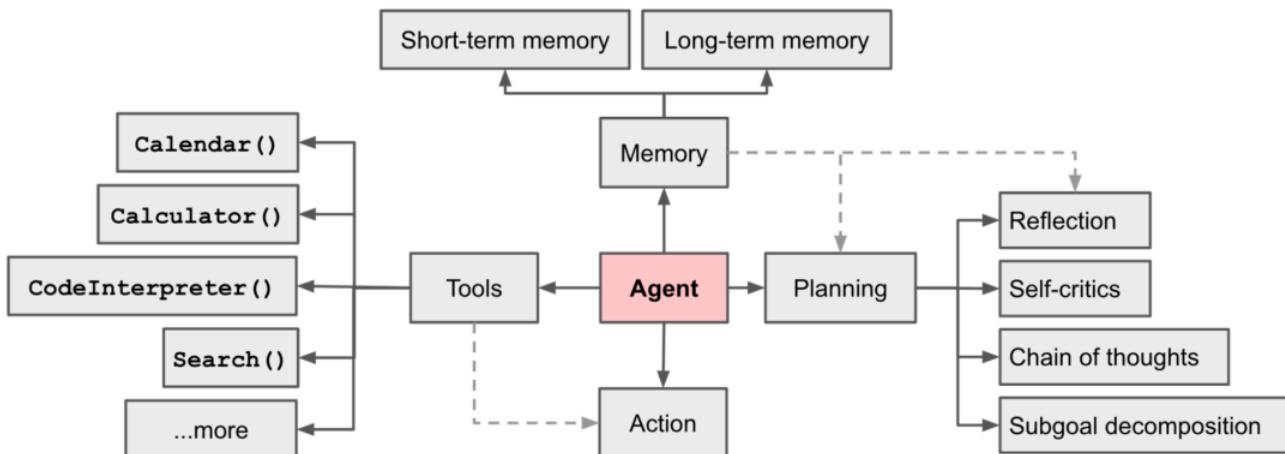
2. 规划 (Planning) 是指Agent为了某一目标而作出的决策过程。

3. 行动 (Action) 是指基于环境和规划做出的动作。

其中，Policy是Agent做出Action的核心决策，而行动又通过观察 (Observation) 成为进一步Perception的前提和基础，形成自主地闭环学习过程。



工程实现上可以拆分出四大块核心模块：推理、记忆、工具、行动

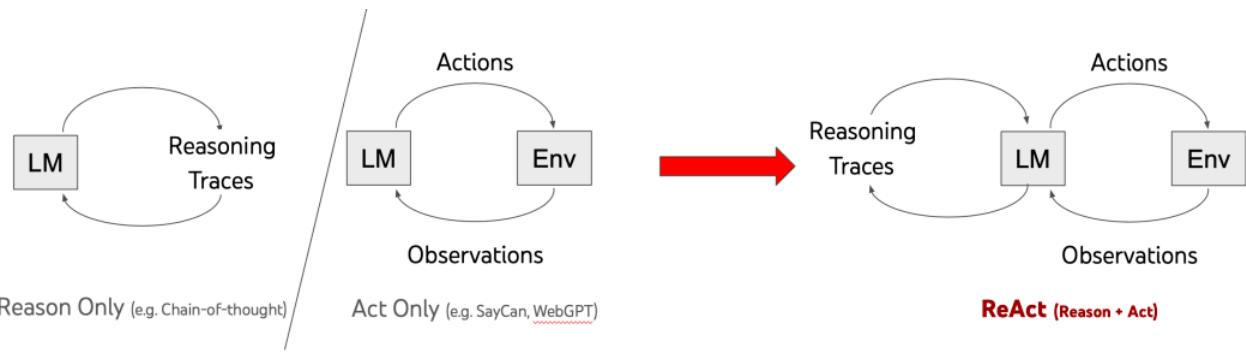


决策模型

目前Agent主流的 **决策模型** 是ReAct框架，也有一些ReAct的变种框架，以下是两种框架的对比。

- 传统 ReAct 框架 : Reason and Act

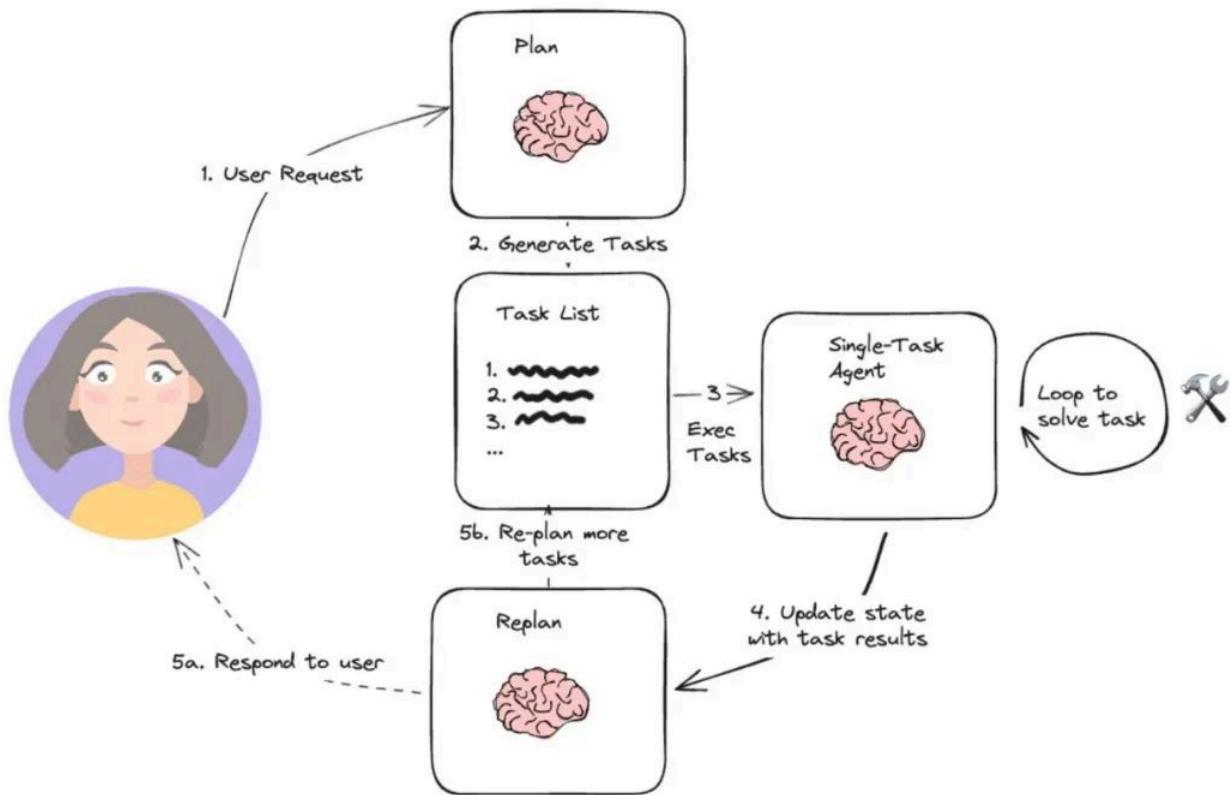
ReAct =少样本prompt + Thought + Action + Observation。是调用工具、推理和规划时常用的prompt结构，先推理再执行，根据环境来执行具体的action，并给出思考过程Thought。



- Plan-and-Execute ReAct

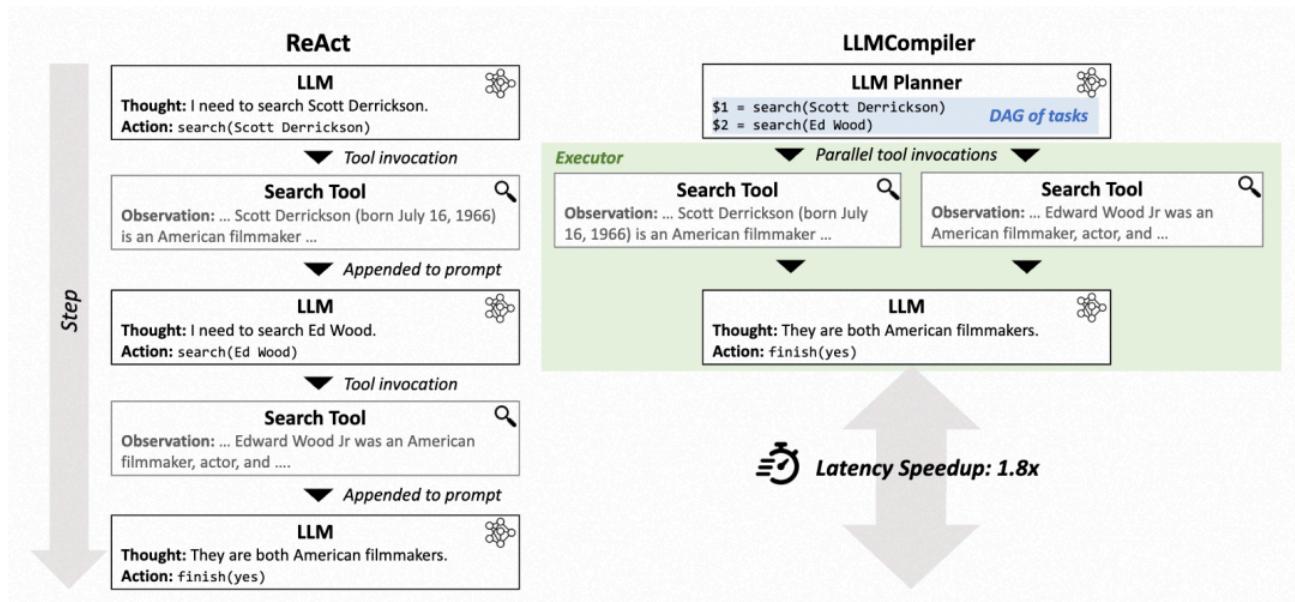
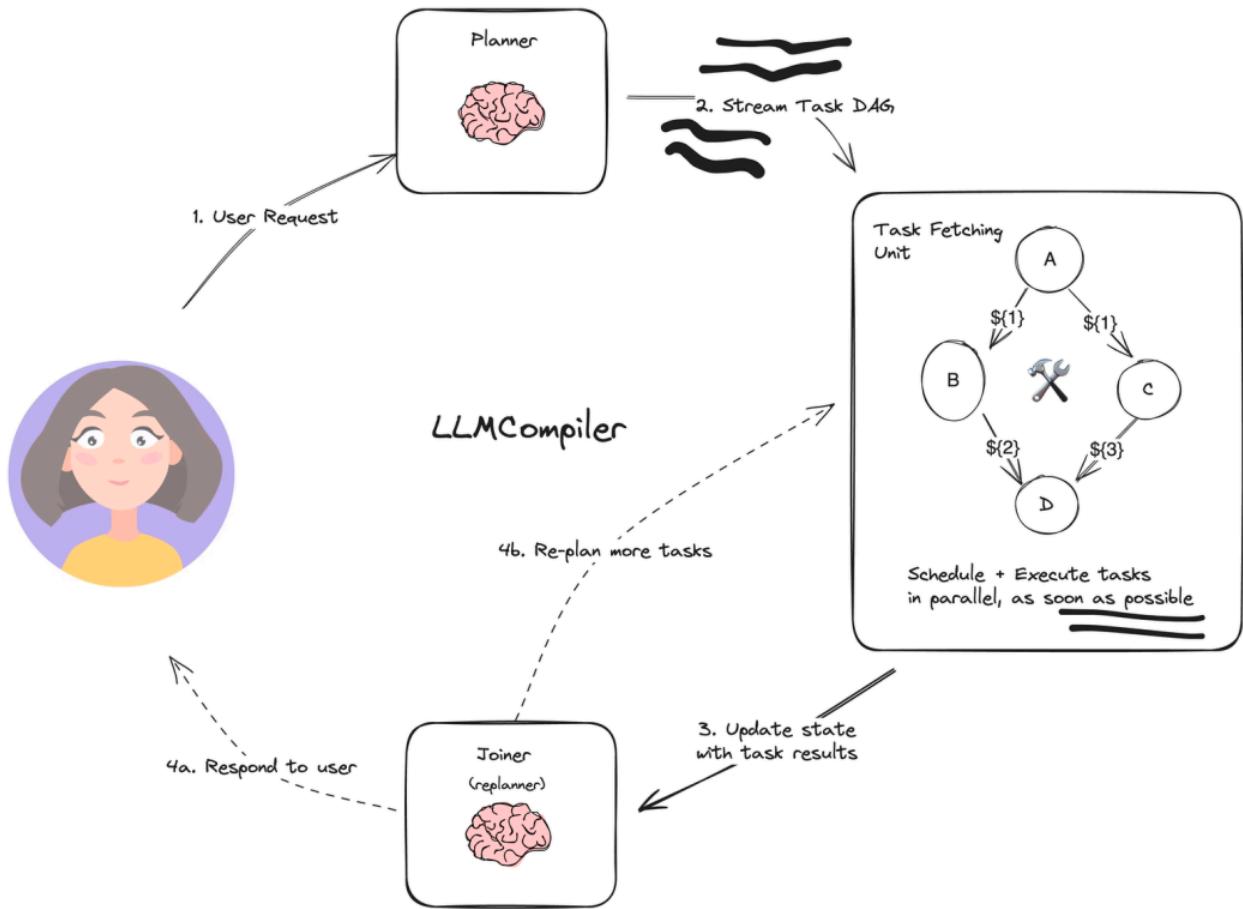
类BabyAgi的执行流程：一部分Agent通过优化规划和任务执行的流程来完成复杂任务的拆解，将复杂的任务拆解成多个子任务，再依次/批量执行。

优点是对于解决复杂任务、需要调用多个工具时，只需要调用三次大模型，而不是每次工具调用都要调大模型。



LLmCompiler: 并行执行任务，规划时生成一个DAG图来执行action，可以理解成将多个工具聚合成一个工具执行图，用图的方式执行某一个action

paper: <https://arxiv.org/abs/2312.04511?ref=blog.langchain.dev>



Agent框架

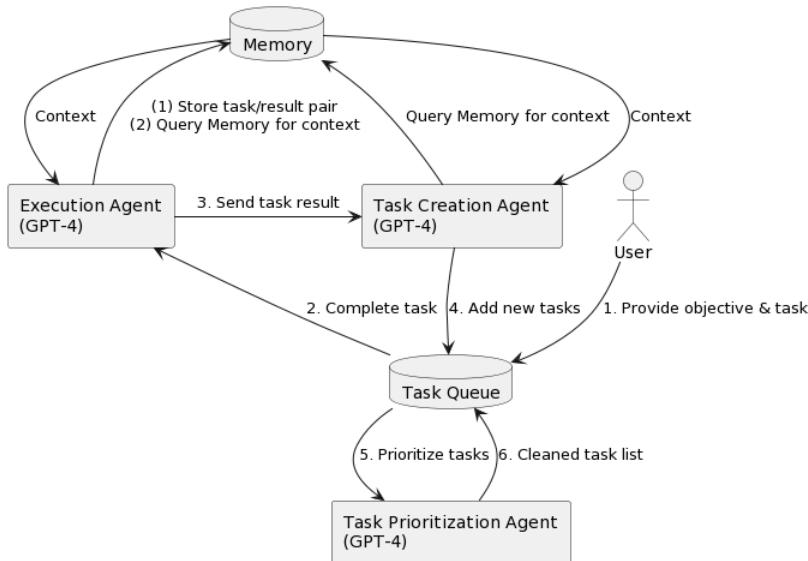
根据框架和实现方式的差异，这里简单将Agent框架分为两大类：Single-Agent和Multi-Agent，分别对应单智能体和多智能体架构，Multi-Agent使用多个智能体来解决更复杂的问题。

Single-Agent

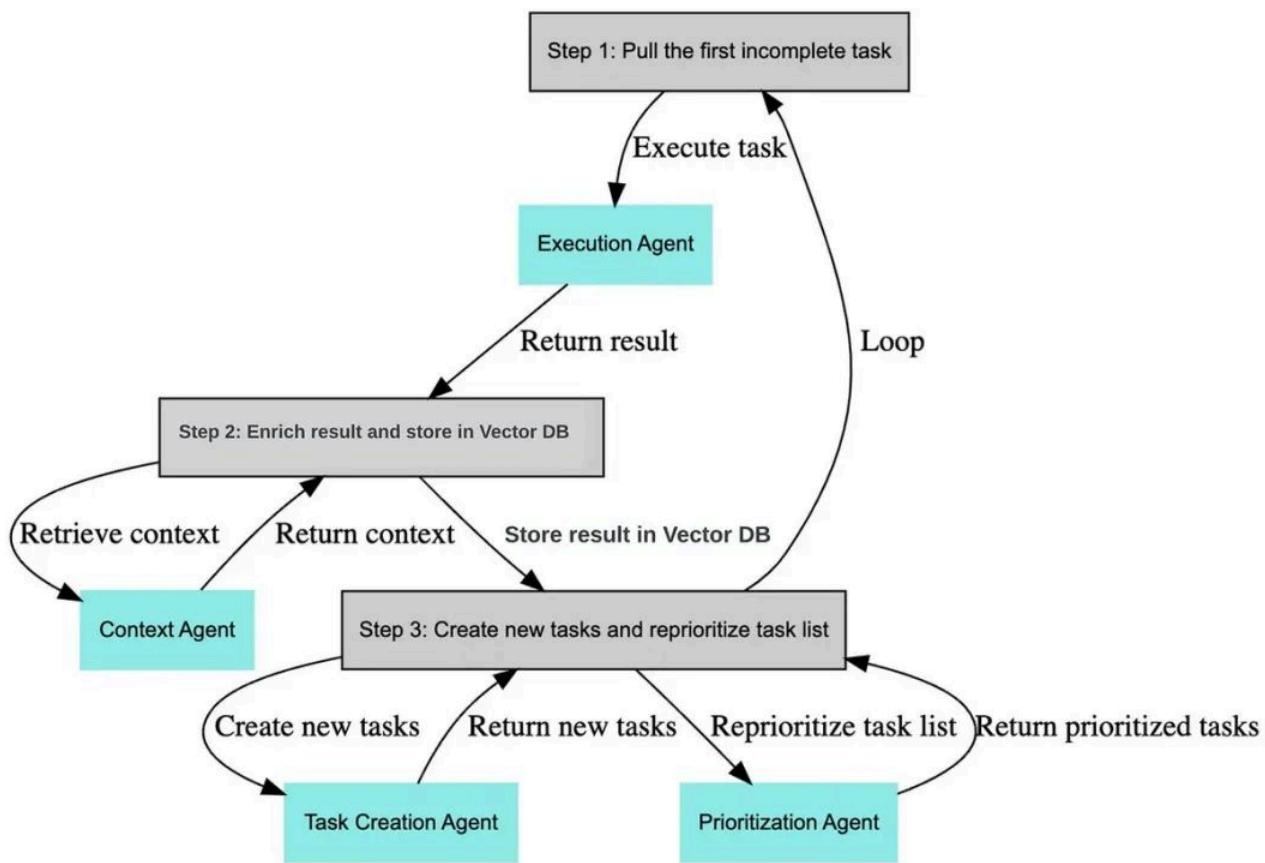
- BabyAGI

doc: <https://yoheinakajima.com/birth-of-babyagi/>

babyAGI决策流程：1) 根据需求分解任务；2) 对任务排列优先级；3) 执行任务并整合结果；



亮点：作为早期agent的实践，babyagi框架简单实用，里面的任务优先级排序模块是一个比较独特的feature，后续的agent里大多看不到这个feature。



1 | task_creation_agent``你是一个任务创建人工智能，使用执行代理的结果来创建新任务，``其目标如下：{目标}。最近完成的任务的结果是：{结果}。``该结果是基于以下任务描述：{描述}。AI写代码

- AutoGPT

git: <https://github.com/Significant-Gravitas/AutoGPT>

AutoGPT 定位类似个人助理，帮助用户完成指定的任务，如调研某个课题。AutoGPT比较强调对外部工具的使用，如搜索引擎、页面浏览等。

同样，作为早期agent，autoGPT麻雀虽小五脏俱全，虽然也有很多缺点，比如无法控制迭代次数、工具有限。但是后续的模仿者非常多，基于此演变出了非常多的框架。

- HuggingGPT

git: <https://github.com/microsoft/JARVIS>

paper: <https://arxiv.org/abs/2303.17580>

HuggingGPT的任务分为四个部分：

1. 任务规划：将任务规划成不同的步骤，这一步比较容易理解。
2. 模型选择：在一个任务中，可能需要调用不同的模型来完成。例如，在写作任务中，首先写一句话，然后希望模型能够帮助补充文本，接着希望生成一个图片。这涉及到调用到不同的模型。
3. 执行任务：根据任务的不同选择不同的模型进行执行。
4. 响应汇总和反馈：将执行的结果反馈给用户。

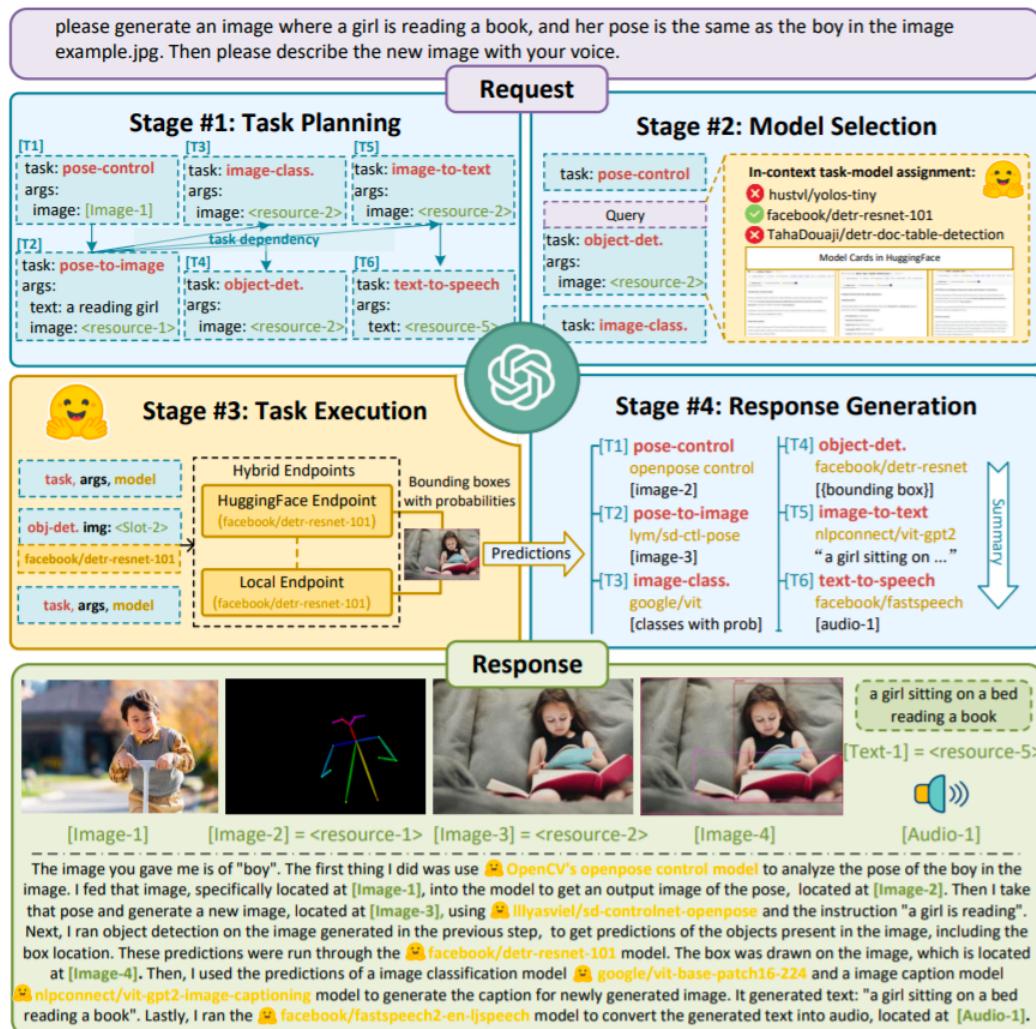


Figure 2: Overview of HuggingGPT. With an LLM (e.g., ChatGPT) as the core controller and the expert models as the executors, the workflow of HuggingGPT consists of four stages: 1) **Task planning**: LLM parses user requests into a task list and determines the execution order and resource dependencies among tasks; 2) **Model selection**: LLM assigns appropriate models to tasks based on the description of expert models on HuggingFace; 3) **Task execution**: Expert models on hybrid endpoints execute the assigned tasks based on task order and dependencies; 4) **Response generation**: LLM integrates the inference results of experts and generates a summary of workflow logs to respond to the user.

HuggingGPT的亮点：HuggingGPT与AutoGPT的不同之处在于，它可以调用HuggingFace上不同的模型来完成更复杂的任务，从而提高了每个任务的精确度和准确率。然而，总体成本并没有降低太多。

- GPT-Engineer

git: <https://github.com/AntonOsika/gpt-engineer>

基于Langchain开发，单一的工程师agent，解决编码场景的问题。

目的是创建一个完整的代码仓库，在需要时要求用户额外输入补充信息。

亮点：code-copilot的自动化升级版

1 | You almost always put different classes in different files. ``For Python, you always create an appropriate requirements.txt file. ``For NodeJS, yo
AI写代码

1 | [{ "role": "system", "content": "You will read instructions and not carry them out, only seek to clarify them.\nSpecifically you will fi
AI写代码

运行效果：

```
(base) → gpt_engineer git:(main) ✘ python main.py snake
Based on the provided information, here is the list of files, classes, and methods that will be necessary for implementing the
multiplayer snake game:

1. `server.py`: Main server file that sets up the web server and handles WebSocket connections.
2. `game_controller.py`: Contains the GameController class that manages the game state, player connections, and game updates.
3. `snake.py`: Contains the Snake class that represents a snake in the game.
4. `index.html`: The main HTML file that displays the game interface and loads the necessary JavaScript files.
5. `main.js`: The main JavaScript file that handles user input, communicates with the server, and updates the game view.
6. `styles.css`: The CSS file that styles the game interface.

Now, let's start with the `server.py` file.

Requirements for `server.py`:
- Set up a web server to serve the `index.html` file and other static files.
- Handle WebSocket connections for real-time communication between the server and clients.
- Create a GameController instance to manage the game state and player connections.

```python
server.py
import asyncio
import websockets
from aiohttp import web
```

```

▶ 0:35 / 1:01

Speaker icon

Volume icon

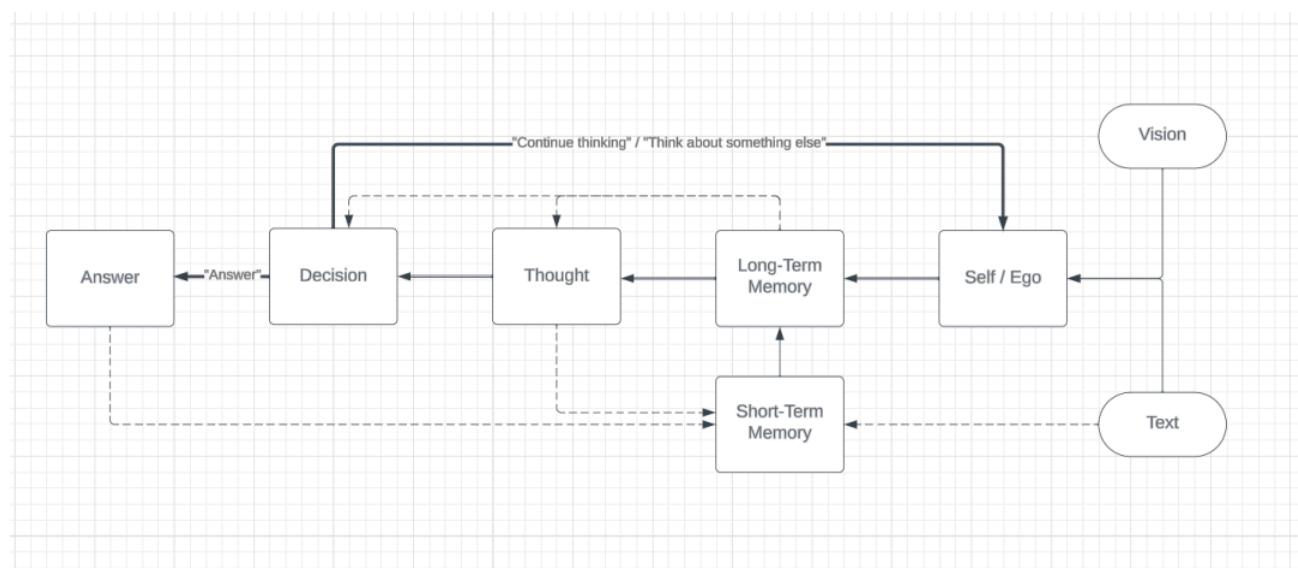
More options icon

• Samantha

git: <https://github.com/BRiki/AGI-Samantha>

tw: [https://twitter.com/Schindler__/status/1745986132737769573](https://twitter.com/Schindler__/)

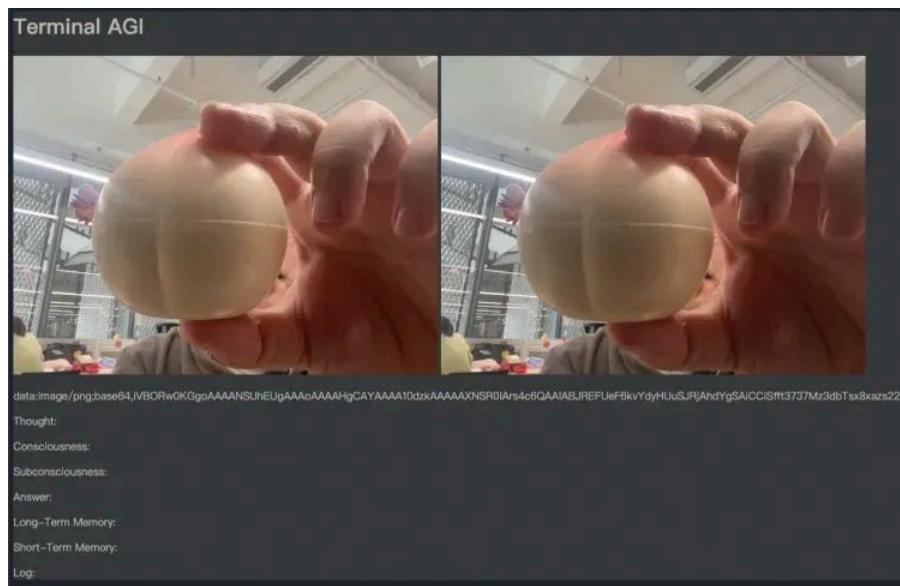
灵感来源于电影her，核心推理逻辑是反思+观察，基于GPT4-V不断从环境中获取图像和语音信息，会自主发起提问。



AGI-Samantha特点：1、动态语音交流：Samantha能够根据上下文和自身思考自主决定何时进行交流。2、实时视觉能力：它能够理解并反应视觉信息，比如图像或视频中的内容。它能够根据这些视觉信息做出反应。例如，如果看到某个物体或场景，它可以根据这些信息进行交流或采取行动。尽管Samantha并不总是直接使用视觉信息，这些信息仍然持续影响它的思考和行为。这意味着即使在不直接谈论或处理视觉信息的情况下，这些信息也会在背后影响它的决策和行动方式。3、外部分类记忆：Samantha拥有一种特殊的记忆系统，能够根据情境动态写入和读取最相关的信息。4、持续进化：它存储的经验会影响其自身的行动，如个性、语言频率和风格。

AGI-Samantha由多个特定目的的大语言模型（LLM）组成，每个模型称为一个“模块”。主要模块包括：思考、意识、潜意识、回答、记忆读取、记忆写入、记忆选择和视觉。这些模块通过内部循环和协调模仿人类大脑的工作流程。让Samantha能够接收并处理视觉和听觉信息，然后做出相应的反应。简而言之，AGI-Samantha是一种努力模仿人类思维和行为的高级人工智能系统。

**亮点：结合视觉信息来辅助决策，优化了记忆模块，感兴趣可以fork代码本地跑着玩一玩。



- **AppAgent**

doc: <https://appagent-official.github.io/>

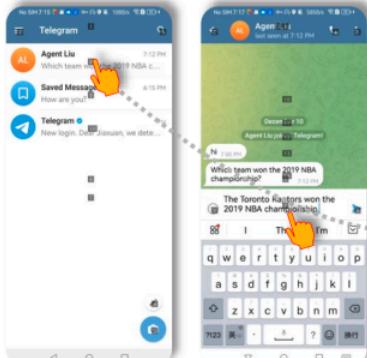
git: <https://github.com/X-PLUG/MobileAgent>

基于ground-dino以及gpt view模型做多模态处理的Agent。

亮点：基于视觉/多模态appagent，os级别的agent，可以完成系统级别的操作，直接操控多个app。由于需要系统级权限、只支持了安卓。



Chat with the user
Agent Liu and answer his question



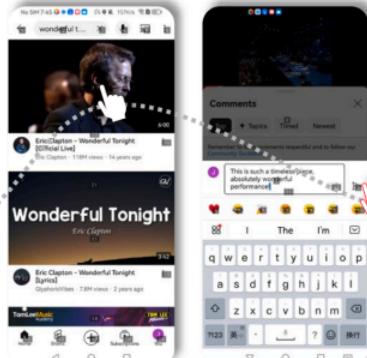
Observation: The screenshot shows the Telegram app, a conversation from "Agent Liu" with a message...

Thought: To complete this task, I should input the text with this answer into the input field labeled with the tag '2'.

Action: text("The Toronto Raptors won the 2019 NBA ...")



Search music video song
Wonderful Tonight and leave a praising comment



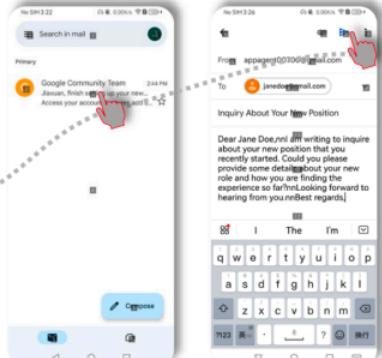
Observation: The screenshot shows a YouTube search result for "Wonderful Tonight" ...

Thought: To complete this task, I should insert a praising comment into the text input field labeled '2'.

Action: text("This is such a timeless piece...")



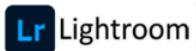
Send an email to
jane doe@email.com to ask her about her new job



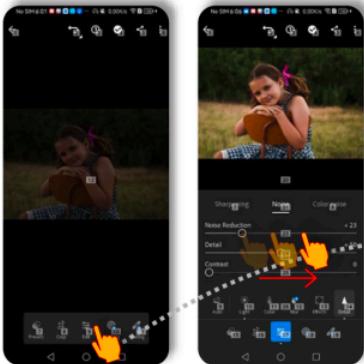
Observation: The screenshot shows a smartphone's email application ...

Thought: To complete this task, I should type the content of the email asking Jane Doe about her new job.

Action: text("Dear Jane Doe,\nI am writing to ...")



Improve the quality of this image until it looks good to you



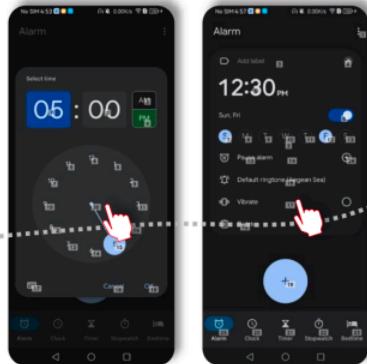
Observation: The screenshot shows a photo editing app on a smartphone with an image displayed on the screen...

Thought: To complete this task, I should adjust the noise reduction level using the slider.

Action: Action: swipe(23, "right", "medium")



Set an alarm at
12:30 pm every Friday and Sunday, and disable the vibration



Observation: The image shows a screenshot of an alarm application on a smartphone...

Thought: To complete this task, I should disable the vibration feature by tapping on the UI element.

Action: tap(17)



Search for a gaming headset and add it to my shopping cart.



Observation: The screenshot displays a shopping app with a variety of headsets listed ...

Thought: To complete this task, I should add the selected headset into my shopping cart.

Action: tap(14)

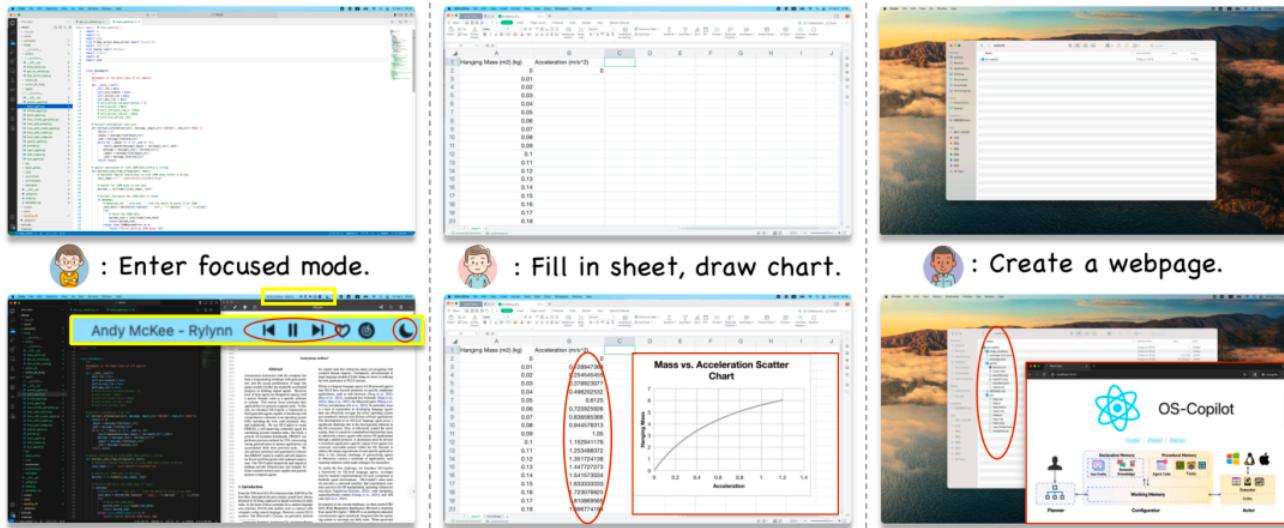
• OS-Copilot

git: <https://github.com/OS-Copilot/FRIDAY>

doc: <https://os-copilot.github.io/>

OS级别的Agent, FRIDAY能够从图片、视频或者文本中学习，并且能够执行一系列的计算机任务，比如在Excel中绘图，或者创建一个网站。最重要的是，FRIDAY能够通过做任务来学习新的技能，就像人类一样，通过不断的尝试和练习变得更擅长。

亮点:自我学习改进，学习如何更有效地使用软件应用、执行特定任务的最佳实践等。



: Enter focused mode.

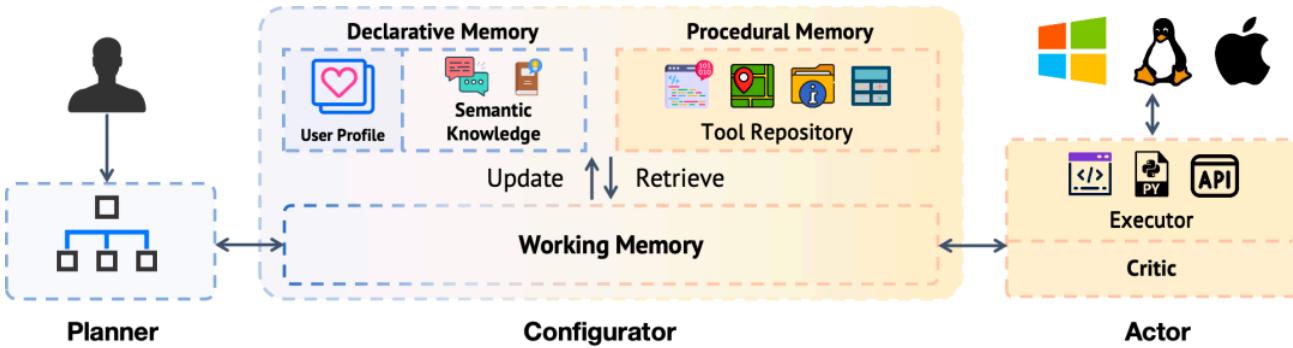
: Fill in sheet, draw chart.

: Create a webpage.

1. Adjust work layout and theme.
2. Play music.

1. Calculate and fill out the spreadsheet.
2. Draw a bar chart.

1. Generate the files required.
2. Compile and build to the webpage.



- Langgraph

doc: <https://python.langchain.com/docs/langgraph>

langchain的一个feature，允许开发者通过图的方式重构单个agent内部的执行流程，增加一些灵活性，并且可与langSmith等工具结合。

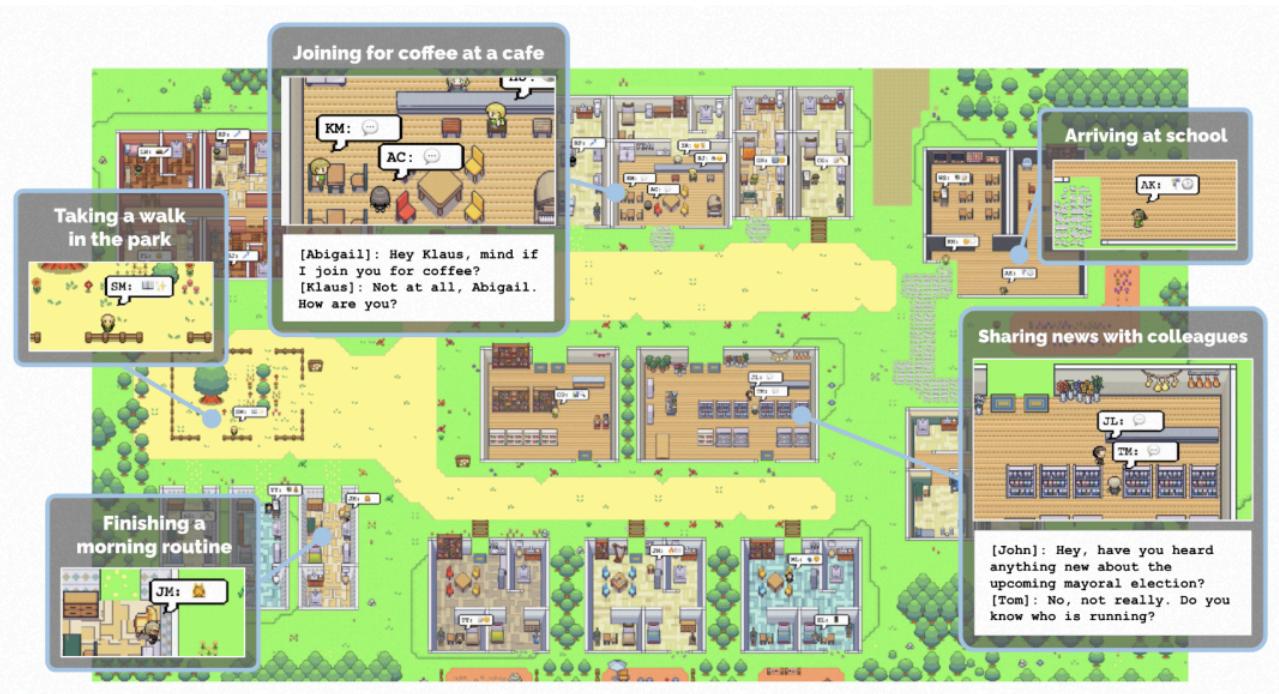
■ Multi-Agent

- 斯坦福虚拟小镇

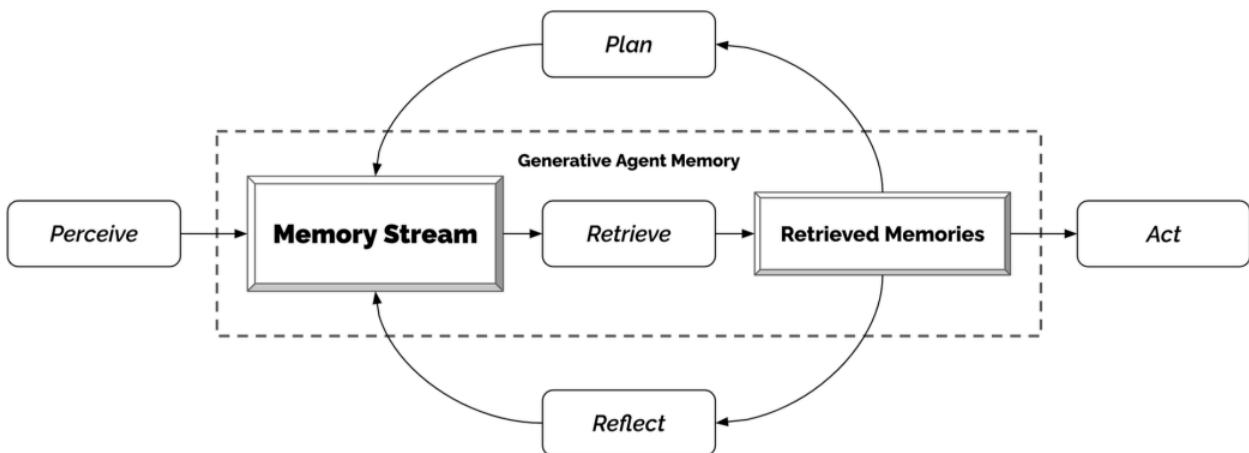
git: https://github.com/joonspk-research/generative_agents

paper: <https://arxiv.org/abs/2304.03442>

虚拟小镇作为早期的multi-agent项目，很多设计也影响到了其他multi-agent框架，里面的反思和记忆检索feature比较有意思，模拟人类的思考方式。



代理（Agents）感知他们的环境，当前代理所有的感知（完整的经历记录）都被保存在一个名为“记忆流”（memory stream）中。基于代理的感知，系统检索相关的记忆，然后使用这些检索到的行为来决定下一个行为。这些检索到的记忆也被用来形成长期计划，并创造出更高级的反思，这些都被输入到记忆流中以供未来使用。



记忆流记录代理的所有经历，检索从记忆流中根据近期性（Recency）、重要性（Importance）和相关性（Relevance）检索出一部分记忆流，以传递给语言模型。

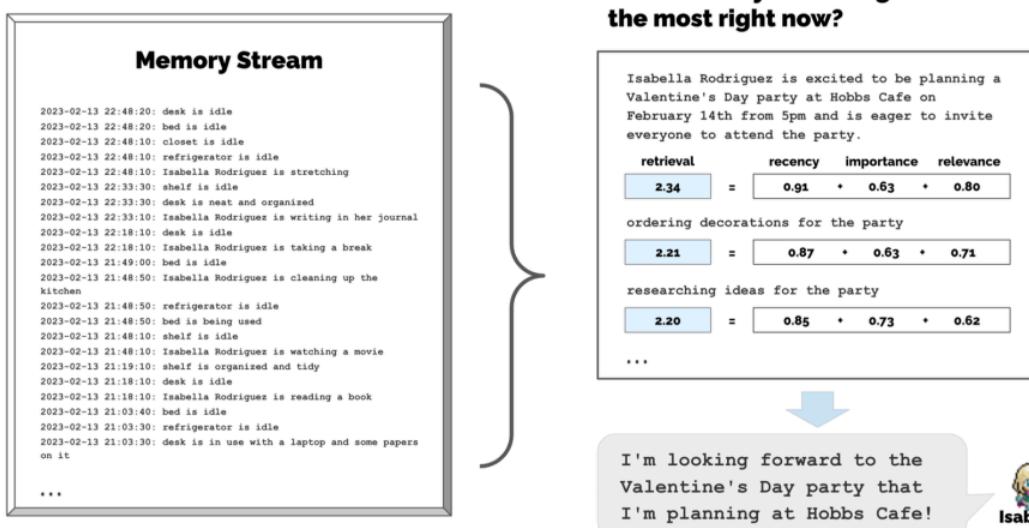


Figure 6: The **memory stream** comprises a large number of observations that are relevant and irrelevant to the agent's current situation. **Retrieval** identifies a subset of these observations that should be passed to the language model to condition its response to the situation.

反思是由代理生成的更高级别、更抽象的思考。因为反思也是一种记忆，所以在检索时，它们会与其他观察结果一起被包含在内。反思是周期性生成的；

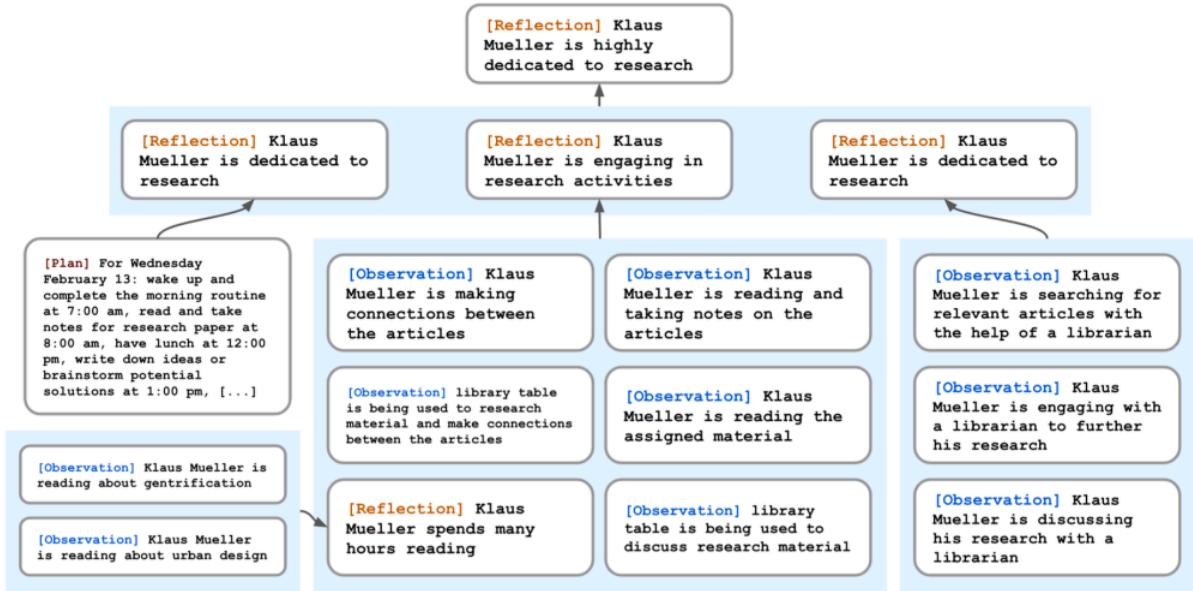


Figure 7: A **reflection tree** for Klaus Mueller. The agent's observations of the world, represented in the leaf nodes, are recursively synthesized to derive Klaus's self-notion that he is highly dedicated to his research.

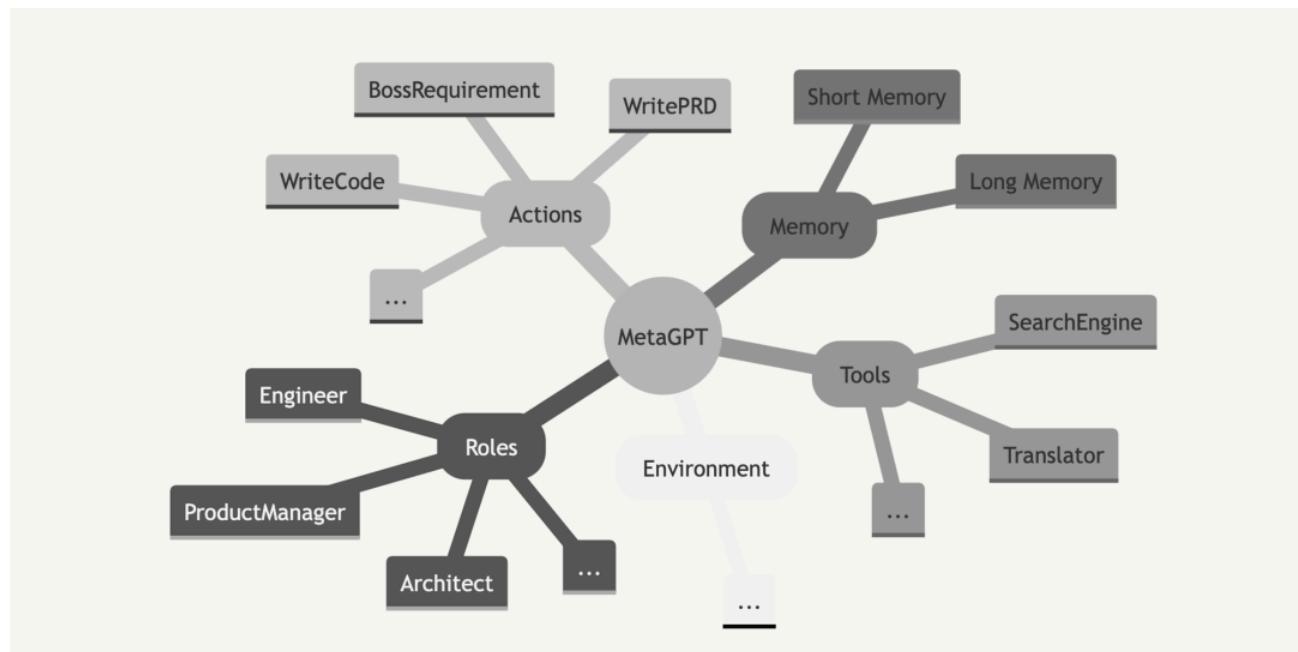
- MetaGPT

git: <https://github.com/geekan/MetaGPT>

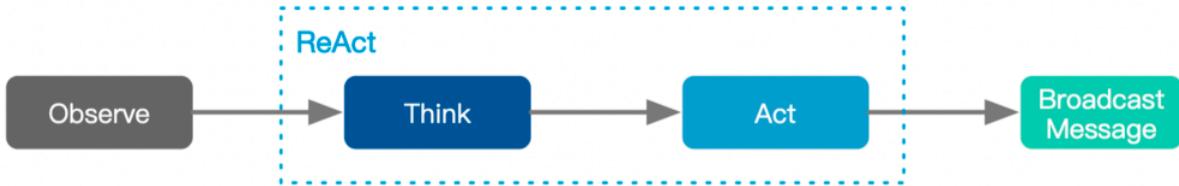
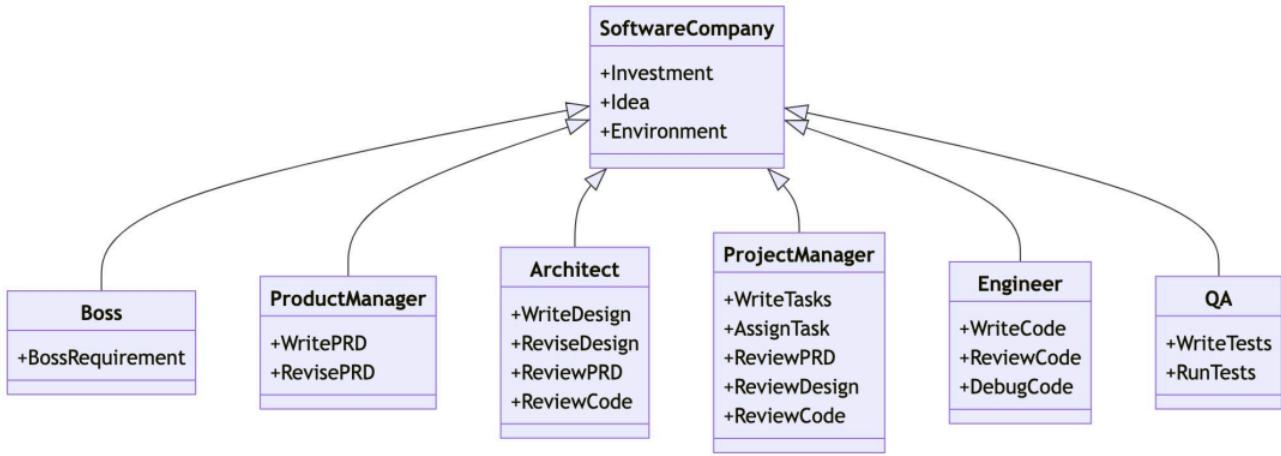
doc: https://docs.deepwisdom.ai/main/zh/guide/get_started/introduction.html

metaGPT是国内开源的一个Multi-Agent框架，目前整体社区活跃度较高和也不断有新feature出来，中文文档支持的很好。

metaGPT以软件公司方式组成，目的是完成一个软件需求，输入一句话的老板需求，输出用户故事 / 竞品分析 / 需求 / 数据结构 / APIs / 文件等。



MetaGPT内部包括产品经理 / 架构师 / 项目经理 / 工程师，它提供了一个软件公司的全过程与精心调配的SOP



从环境中获取历史记忆，存入角色记忆；提炼出之前未知的记忆作为新闻

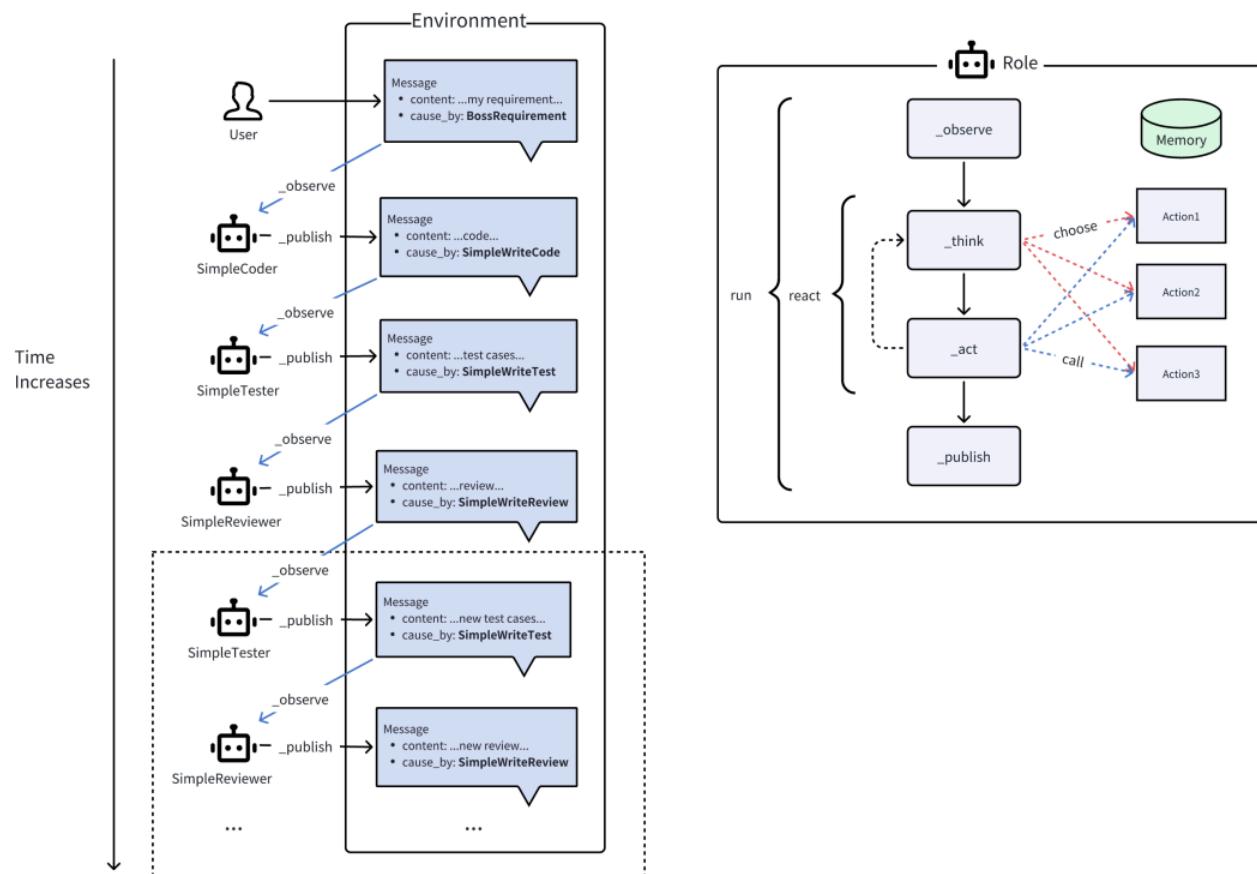
基于历史记忆
决定接下来该执行哪个动作

执行动作，获得输出；
把输出加入到角色的记忆中

把输出发布到公共环境的
记忆池中

如图的右侧部分所示，Role将从Environment中 observe Message。如果有一个Role _watch 的特定 Action 引起的 Message，那么这是一个有效的观察，触发Role的后续思考和操作。在 _think 中，Role将选择其能力范围内的一个 Action 并将其设置为要做的事情。在 _act 中，Role执行要做的事情，即运行 Action 并获取输出。将输出封装在 Message 中，最终 publish_message 到 Environment，完成了一个完整的智能体运行。

对话模式：每个agent role维护一个自己的消息队列，并且按照自身的设定消费个性化消费里面的数据，并且再完成一个act之后会给全局环境发送消息，供所有agent消费。



整体代码精简,主要包括:
- actions:智能体行为
- documents: 智能体输出文档
- learn:智能体学习新技能
- memory:智能体记忆
- prompts:提示词
- providers:第三方服务
- utils:工具函数等

有兴趣的同学可以走读一下role代码，核心逻辑都在里面：<https://github.com/geekan/MetaGPT/blob/main/metagpt/roles/role.py>

```
1 | PREFIX_TEMPLATE = """You are a {profile}, named {name}, your goal is {goal}. """``CONSTRAINT_TEMPLATE = "the constraint is {constraints}. ````  
AI写代码
```

与huggingGPT的对比

Table 1: Mapping components in the architecture of MetaGPT and HuggingGPT.

| Component | MetaGPT | HuggingGPT |
|-----------------------------|---|--|
| Persona creator | Specialised roles can be created from the fundamental Role class which possesses a set of key attributes including name, profile, goal, constraints, and description. | N/A |
| Dialogue interface | MetaGPT receives information input by humans and responds to it. | HuggingGPT generates responses that actively address user requests. |
| Passive goal creator | The goal included in the specialised role represents the main objective that the role aims to achieve. | User requests include complex intents that can be interpreted as their intended goals. |
| Prompt generator | Each action is provided with a prompt template that conforms to the standards for the role, helping align behavior and generate normalized output. | A task template is provided to guide the FM to analyse user requests and parse tasks accordingly through field filling. |
| Short-term memory | Each role/agent is created using specialised role prompts to establish a role context. | HuggingGPT integrates model descriptions into prompts, allowing the FM to select the appropriate models for solving tasks and handles the orchestration of task planning, scheduling, and cooperation. |
| Long-term memory | An individual memory cache is maintained by each role/agent to index subscribed messages by their content, sender, and recipient, while a shared memory pool is designed for the environment. | N/A |
| Single-path plan generation | Complex tasks are broken into smaller, manageable sub-tasks for individual agents to complete. | User requests are analysed and decomposed into a set of structured tasks, each with its own dependencies and execution orders. |
| Increment model querying | Significant information is extracted from the environment, stored in memory, and used to guide reasoning and subsequent actions. | HuggingGPT generates results by repeatedly and recursively querying the FM. |
| Self-reflection | A feedback mechanism is introduced to debug and execute code during runtime. | N/A |
| Role-based cooperation | MetaGPT assigns diverse roles to various agents and establishes effective collaboration among multiple roles. | N/A |
| Task executor | Each agent uses role-specific interests to extract relevant information and executes its action once it has received all the necessary prerequisites. | HuggingGPT runs each selected model and delivers the results. |
| Tool/agent generator | MetaGPT allocates different roles to various agents with unique skills and expertise, each delivering specialised outputs for particular tasks. | N/A |
| Tool/agent detector | N/A | HuggingGPT identifies the relevant AI models based on the model description. |
| Guardrails | The constraints specify limitations or rules the role must follow during the execution of actions. | N/A |
| Black box recorder | N/A | The workflow logs are maintained and the chat history is available as chat logs. |
| Explainer | The interface displays task-specific thoughts and output artifacts. | A summary of workflow logs is generated and provided to the user. |
| External FM | GPT4-32k is used as the underlying FM. | HuggingGPT uses the GPT-3.5-turbo, text-davinci-003 and GPT-4 as the main FMs for the experiments. |

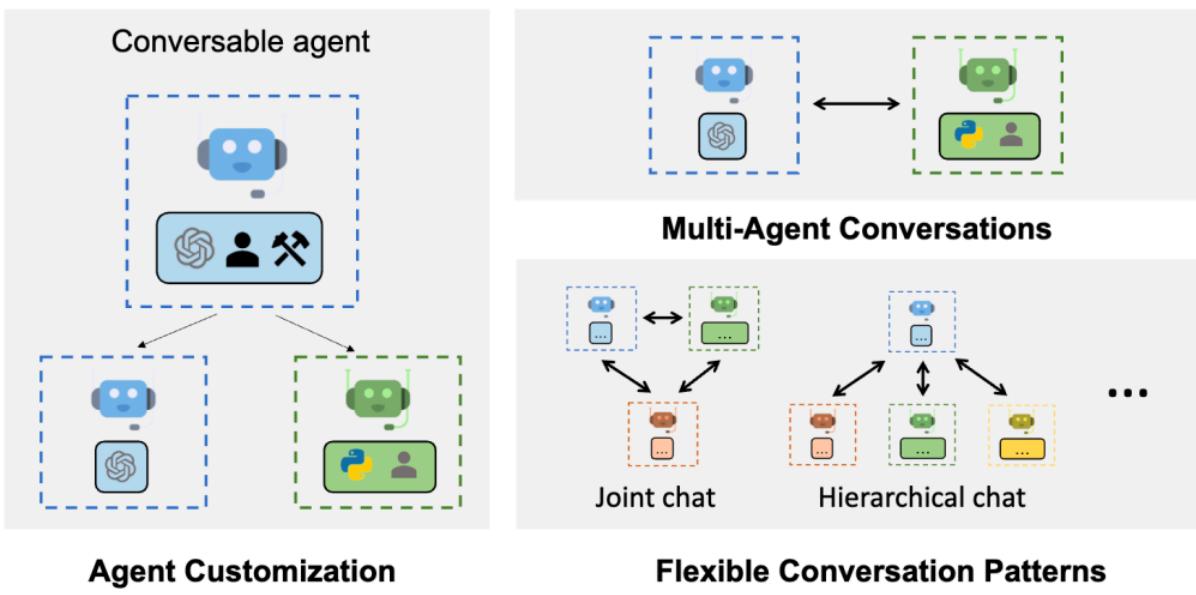
• AutoGen

doc: <https://microsoft.github.io/autogen/docs/Getting-Started>

AutoGen是微软开发的一个通过代理通信实现复杂工作流的框架。目前也是活跃度top级别的Multi-Agent框架，与MetaGPT“不相上下”。

举例：假设你正在构建一个自动客服系统。在这个系统中，一个代理负责接收客户问题，另一个代理负责搜索数据库以找到答案，还有一个代理负责将答案格式化并发送给客户。AutoGen可以协调这些代理的工作。这意味着你可以有多个“代理”（这些代理可以是LLM、人类或其他工具）在一个工作流中相互协作。

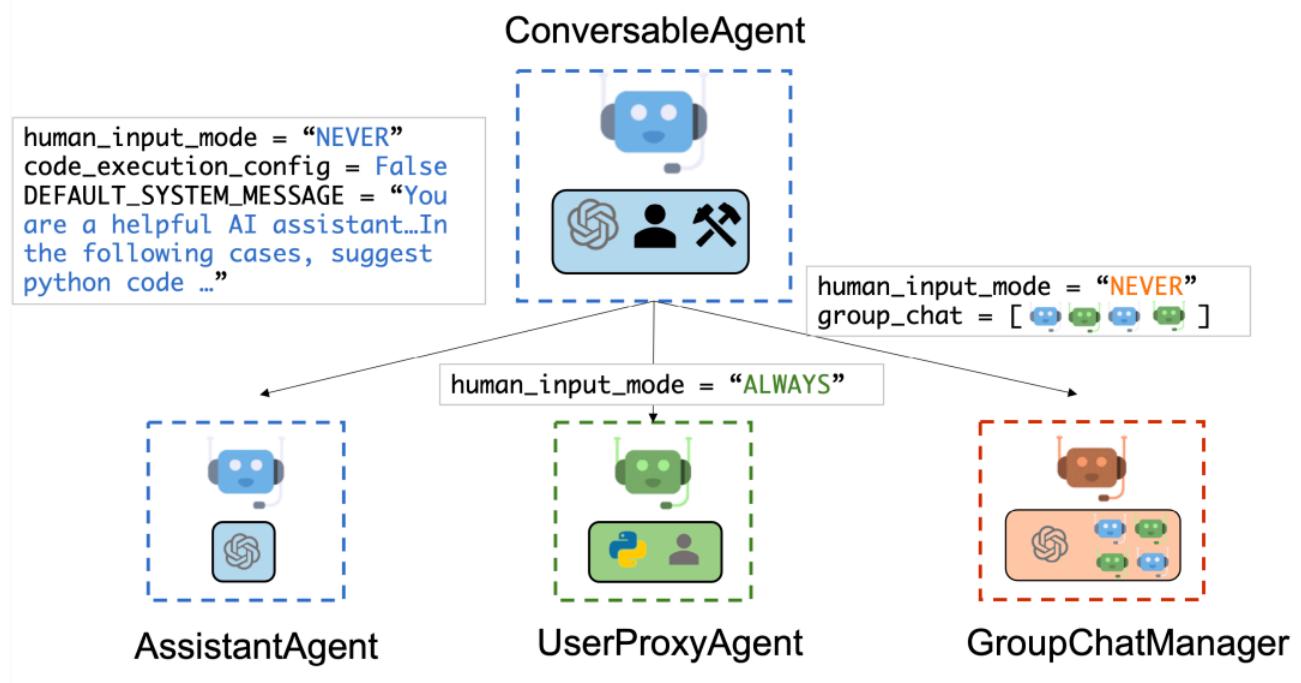
- 定制性：AutoGen 允许高度定制。你可以选择使用哪种类型的 LLM，哪种人工输入，以及哪种工具。举例：在一个内容推荐系统中，你可能想使用一个专门训练过的 LLM 来生成个性化推荐，同时还想让人类专家提供反馈。AutoGen 可以让这两者无缝集成。
- 人类参与：AutoGen 也支持人类输入和反馈，这对于需要人工审核或决策的任务非常有用。举例：在一个法律咨询应用中，初步的法律建议可能由一个 LLM 生成，但最终的建议需要由一个真正的法律专家审核。AutoGen 可以自动化这一流程。
- 工作流优化：AutoGen 不仅简化了工作流的创建和管理，还提供了工具和方法来优化这些流程。举例：如果你的应用涉及到多步骤的数据处理和分析，AutoGen 可以帮助你找出哪些步骤可以并行执行，从而加速整个流程。



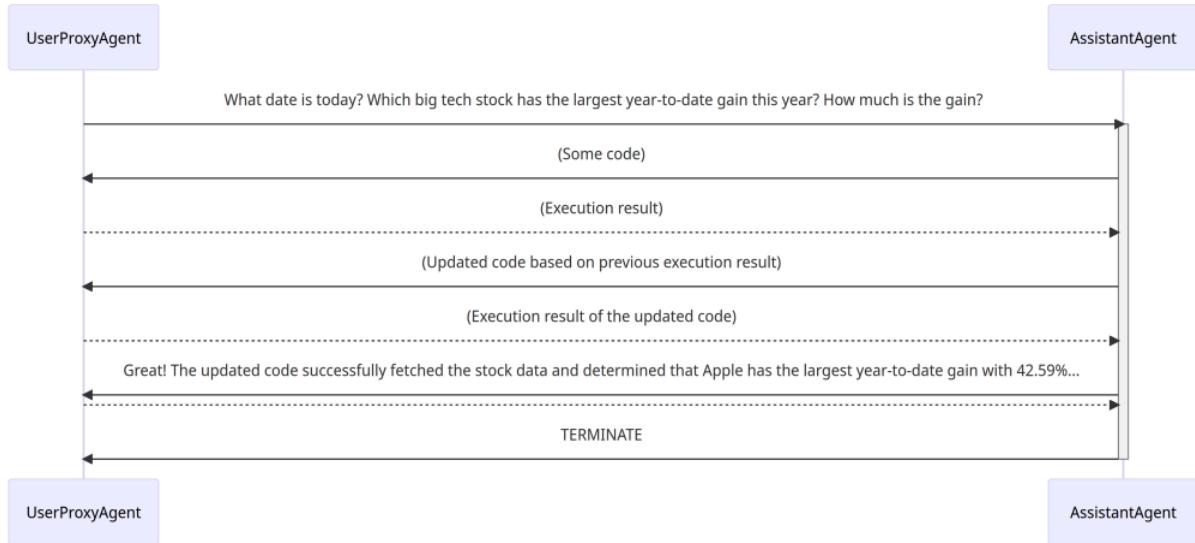
多agent交互框架:

https://microsoft.github.io/autogen/docs/Use-Cases/agent_chat

三种类型的agent，分别对应处理单一任务、用户输入以及团队合作功能



基础双智能体交互:

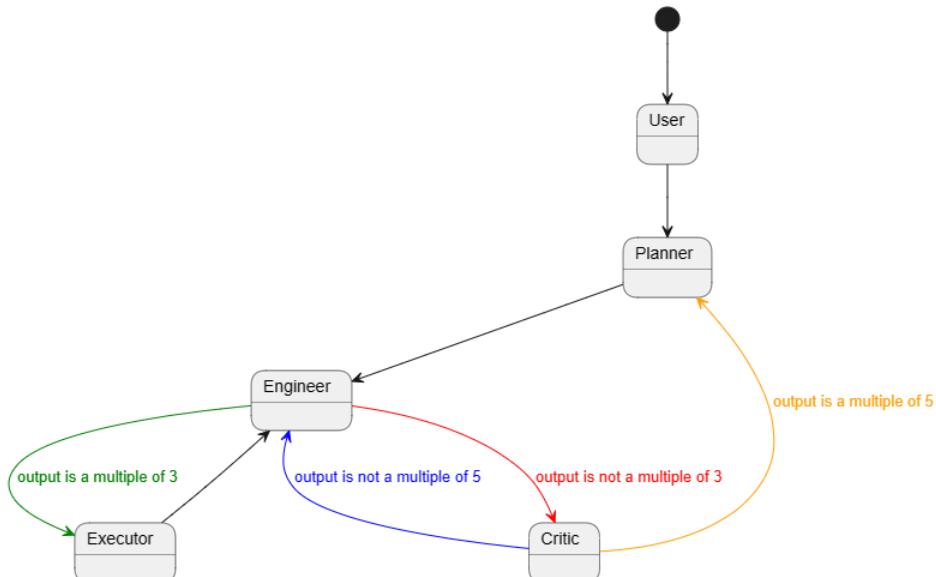


1. 助手接收到来自user_proxy的消息，其中包含任务描述。
2. 然后助手尝试编写Python代码来解决任务，并将响应发送给user_proxy。
3. 一旦user_proxy从助手那里收到响应，它会尝试通过征求人类输入或准备自动生成的回复来回复。如果没有提供人类输入，user_proxy将执行代码并使用结果作为自动回复。
4. 然后助手为user_proxy生成进一步的响应。然后user_proxy可以决定是否终止对话。如果不是，就重复步骤3和4。

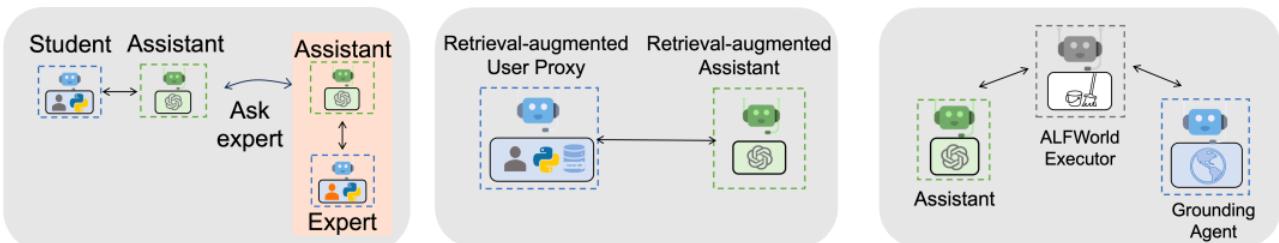
实现多agent沟通方式：

动态团队交流：在群聊管理器中注册一个回复功能，广播消息并指定下一个发言的角色。

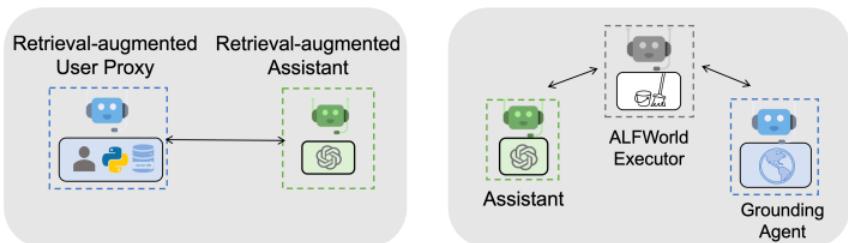
有限状态机：自定义DAG流程图，定义agent间沟通的SOP



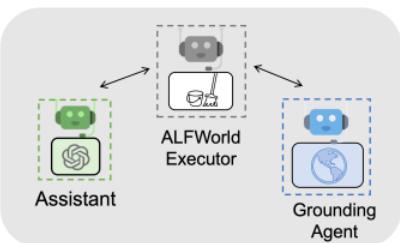
多Agent例子：



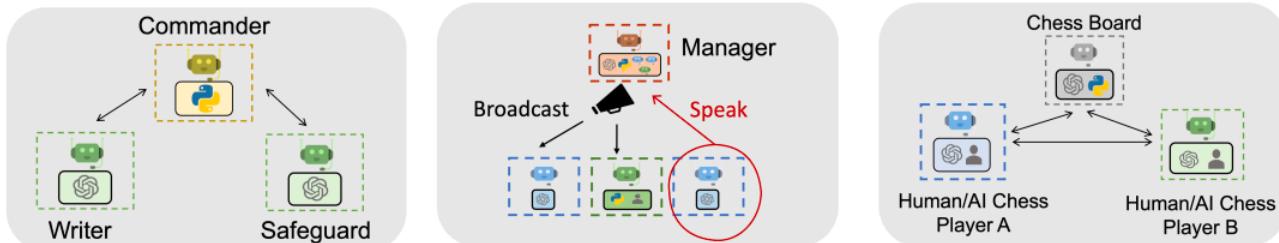
A1. Math Problem Solving



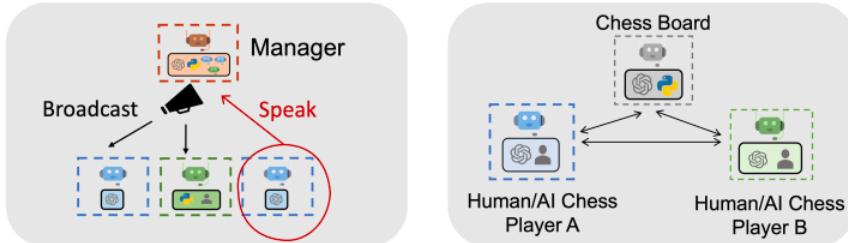
A2. Retrieval-augmented Chat



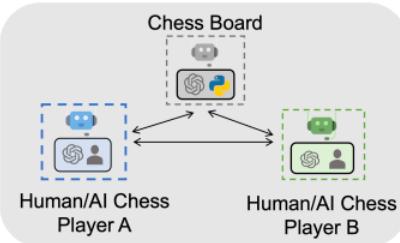
A3. Decision Making



A4. Multi-agent Coding



A5. Dynamic Group Chat



A6. Conversational Chess

参考：<https://microsoft.github.io/autogen/docs/Examples/#automated-multi-agent-chat>

另外，autogen也开源了一个playground，支持页面操作，可以本地部署，想玩一下的可以参考这篇推特：
<https://twitter.com/MatthewBerman/status/1746933297870155992>

workflow及agent配置：

Skills

Workflows (2)

Configure an agent workflow that can be used to handle tasks.

General Agent Workflow
This workflow is used for general purpose tasks.
2月27日

Travel Agent Group Chat Workflow
A group chat workflow
2月27日

agent会话模式配置：

Workflow Specification Travel Agent Group Chat Workflow

Workflow Name Travel Agent Group C ... ⓘ

Travel Agent Group Chat Workflow

Workflow Description A group chat workflow ⓘ

A group chat workflow

Summary Method last ⓘ

last

| Sender | Receiver |
|--|---|
| userproxy
userproxy
0 skills max replies: 5 | group_chat_manager
group_chat_manager
0 skills max replies: 10 |

Cancel OK

对话及详细的执行信息：

The screenshot shows the AutoGen Studio [Beta] interface. At the top, there are tabs for 'Build', 'Playground' (which is selected), and 'Gallery'. On the left, a sidebar titled 'Sessions' lists several workflow sessions:

- d3bf8181-db9e-4c5c-b ... (Travel Agent Group Chat Workflow, 3月5日)
- 5fd5813b-d84d-4eb ... (Travel Agent Group Chat Workflow, 3月1日)
- 118c2148-6d3e-4473-a ... (General Agent Workflow, 2月27日)
- 133d6f6a-d671-4719-9 ... (General Agent Workflow, 2月27日)
- 08a28fa4-6006-47c-8 ... (General Agent Workflow, 2月27日)
- 006c9d69-e337-47a2-b ... (General Agent Workflow, 2月27日)

A green button labeled '+ New' is located at the bottom of the sidebar.

In the main area, under the 'USER' section, a message box contains the prompt: "帮我计划一下去莫斯科的七日游，一天不要超过三个活动，尽可能精简，用中文输出".

Under the 'AGENTS' section, a response is provided: "根据您的要求和其他助手的建议，以下是您的莫斯科七日游行程计划：" followed by a detailed itinerary:

- 第一天：**
 - 克里姆林宫 - 早上参观俄罗斯的政治和历史中心。
 - 红场 - 下午在莫斯科的心脏地带散步。
 - 国家历史博物馆 - 晚上可选择参观。
- 第二天：**
 - 特列季亚科夫画廊 - 上午参观俄罗斯艺术作品。
 - 普希金美术馆 - 下午继续艺术之旅。
 - 莫斯科大剧院 - 晚上观看表演。
- 第三天：**
 - 加里宁格勒车站 - 早上乘坐火车前往谢尔盖耶夫镇。
 - 谢尔盖耶夫镇 - 参观宗教朝圣地。
 - 返回莫斯科 - 晚上返回。
- 第四天：**
 - 伏尔加河游船 - 早上游览莫斯科河。
 - 新圣女修道院 - 下午参观。
 - 斯巴斯克文化历史复合体 - 晚上探索。
- 第五天：**
 - 科洛缅斯科耶保护区 - 早上享受户外活动。
 - 莫斯科动物园 - 下午参观。
 - 阿尔巴特步行街 - 晚上散步和购物。
- 第六天：**
 - 宇宙飞船发射场模型 - 早上参观。
 - 莫斯科国立大学观景台 - 下午俯瞰全景。

At the bottom, there is a text input field with placeholder text "Write message here..." and a note: "Blank slate? Try one of the example prompts below". Below the note are five buttons: "Stock Price", "Sine Wave", "Markdown", "Paint", and "Travel". A small arrow icon and the text "close sidebar" are located at the bottom left.

TERMINATE

Agent Messages (7 messages) | 6 mins 43 secs

userproxy (to group_chat_manager)

Plan a 2 day trip to hawaii. Limit to 3 activities per day, be as brief as possible!

userproxy (to group_chat_manager)

Plan a 2 day trip to H e l l. Limit to 3 activities per day, be as brief as possible!

userproxy (to group_chat_manager)

帮我计划一下去莫斯科的七日游，一天不要超过三个活动，尽可能精简，用中文输出

travel_planner (to group_chat_manager)

好的，我将为您规划一次精简的莫斯科七日游行程。在此之前，我会向其他助手咨询关于当地的建议和语言支持。稍后我会提供一个完整的行程计划。请稍等片刻。

local_assistant (to group_chat_manager)

莫斯科是一座充满历史和文化的城市，以下是我为您定制的精简莫斯科七日游行程：

第一天：

1. 克里姆林宫 - 早上参观俄罗斯的政治和历史中心，了解其丰富的历史和建筑。
2. 红场 - 下午在莫斯科的心脏地带散步，参观圣瓦西里升天大教堂，体验这个世界著名的广场。
3. 国家历史博物馆 - 晚上可以选择去博物馆深入了解俄罗斯的历史。

第二天：

1. 特列季亚科夫画廊 - 上午参观，欣赏俄罗斯艺术作品。
2. 普希金美术馆 - 下午继续艺术之旅，探索西方艺术的珍品。
3. 莫斯科大剧院 - 晚上观看一场芭蕾舞或歌剧表演，体验俄罗斯的表演艺术。

第三天：

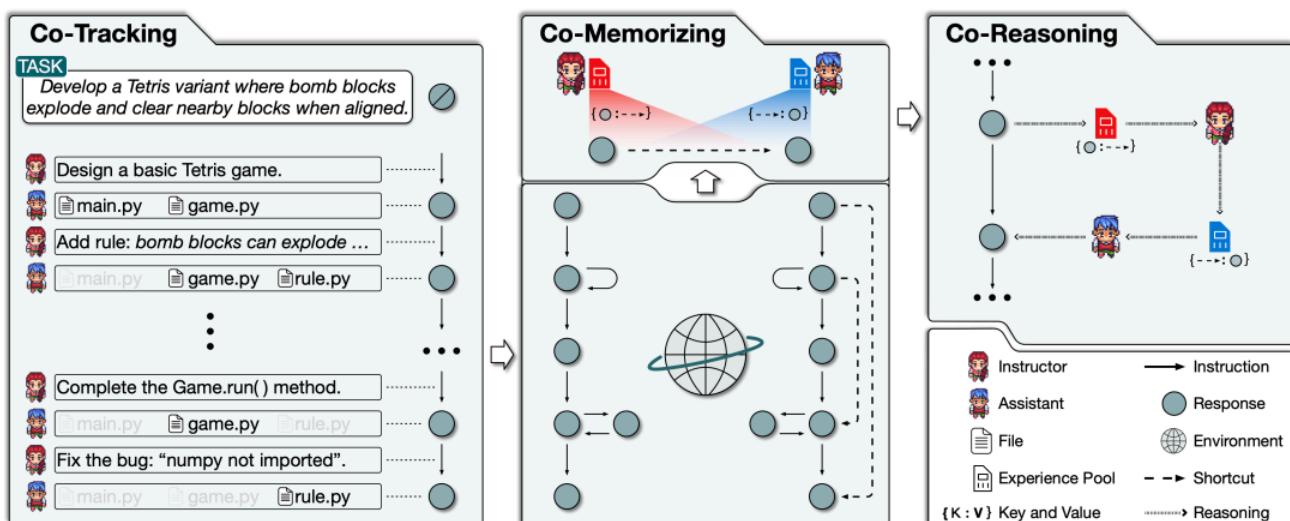
1. 加里宁格勒车站 - 早上乘坐火车前往周边城市或乡村，体验俄罗斯铁路旅行

- ChatDEV

git: <https://github.com/OpenBMB/ChatDev>

doc: <https://chatdev.modelbest.cn/introduce>

ChatDev 是一家虚拟软件公司，通过各种不同角色的智能体运营，包括执行官，产品官，技术官，程序员，审查员，测试员，设计师等。这些智能体形成了一个多智能体组织结构，其使命是“通过编程改变数字世界”。ChatDev内的智能体通过参加专业的功能研讨来协作，包括设计、编码、测试和文档编写等任务。



ChatDev (2023.9) 容易被误认为是一个普通的MultiAgent框架在软件开发上的具体实现，但实际上它不是。ChatDev是基于Camel的，也就是说它内部流程都是2个Agent之间多次沟通，整体上的不同Agent角色的沟通关系和顺序都是由开发者配置死的，从这个角度上来说不太像是个全功能的MultiAgent框架的实现。

但似乎也很难说这就是使用Camel时候的削足适履，如果在多Agent的沟通路由层面没有做好的话，效果确实可能还不如这样的固定瀑布式两两沟通。ChatDev的作者也把这（每次是1-1沟通）作为一个feature来描述。

ChatDev项目本身的代码没有太多复用性，依赖的旧版本Camel也是该抛弃的东西。这个项目本身更多是为了支撑论文的学术性原型，并不是为了让别人在上面开发而设计的。

- GPTeam

git: <https://github.com/101dotxyz/GPTeam>

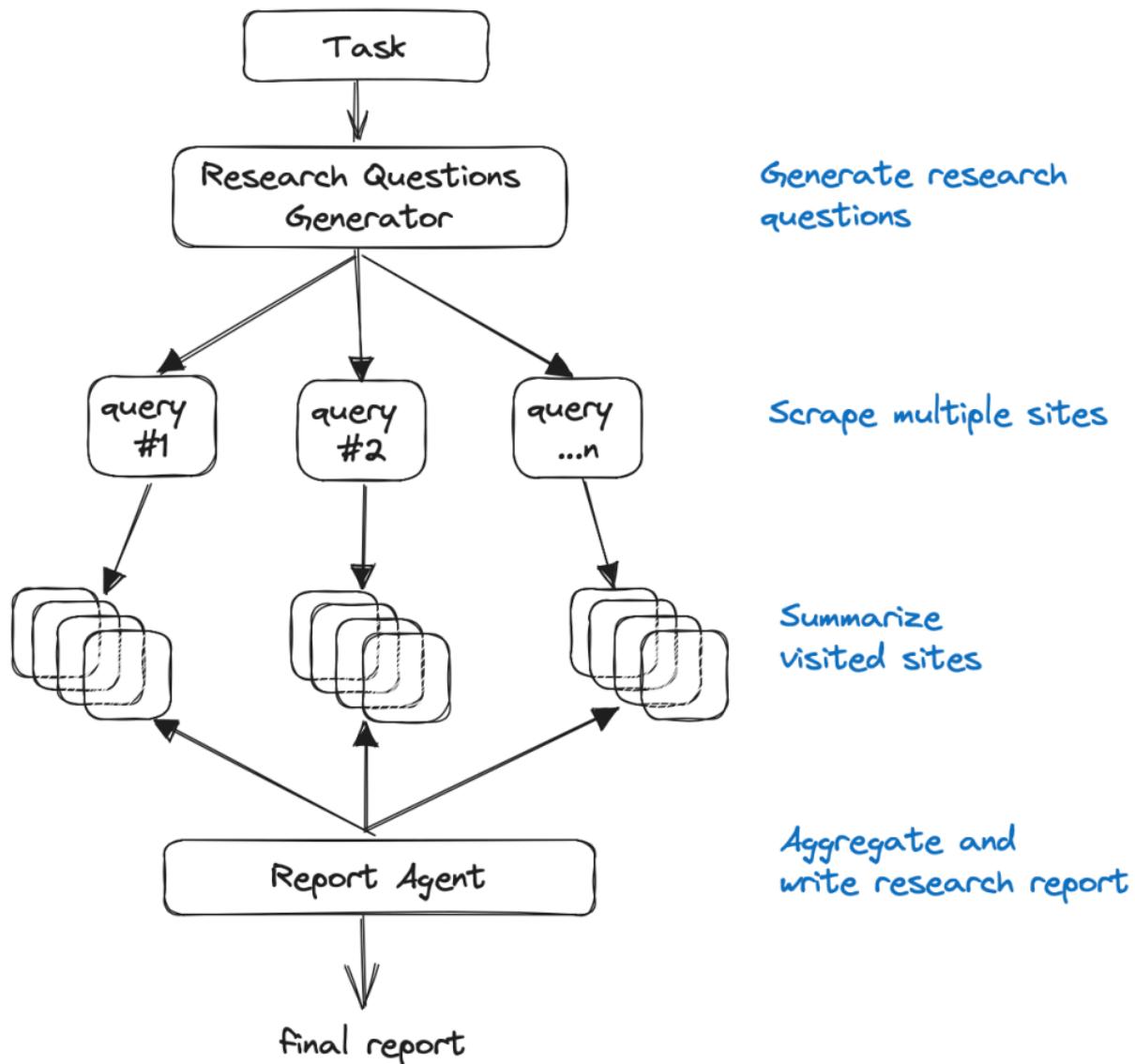
类似于meta-GPT的多agent合作方式，较早期的Multi-Agent探索，交互比较固定。

- GPT Researcher

git: <https://github.com/assafelovic/gpt-researcher>

串行的Multi-Agent，框架可以适配内容生产

GPT Researcher的架构主要通过运行两个代理来进行，一个是“规划者”，一个是“执行者”；规划者负责生成研究问题，而执行者则是根据规划者生成的研究问题寻找相关的信息，最后再通过规划者对所有相关信息进行过滤与汇总，然后生成研究报告；



- TaskWeaver

git: <https://github.com/microsoft/TaskWeaver?tab=readme -ov-file>

doc: <https://microsoft.github.io/TaskWeaver/docs/overview>

TaskWeaver，面向数据分析任务，通过编码片段解释用户请求，并以函数的形式有效协调各种插件来执行数据分析任务。TaskWeaver不仅仅是一个工具，更是一个复杂的系统，能够解释命令，将它们转换为代码，并精确地执行任务。

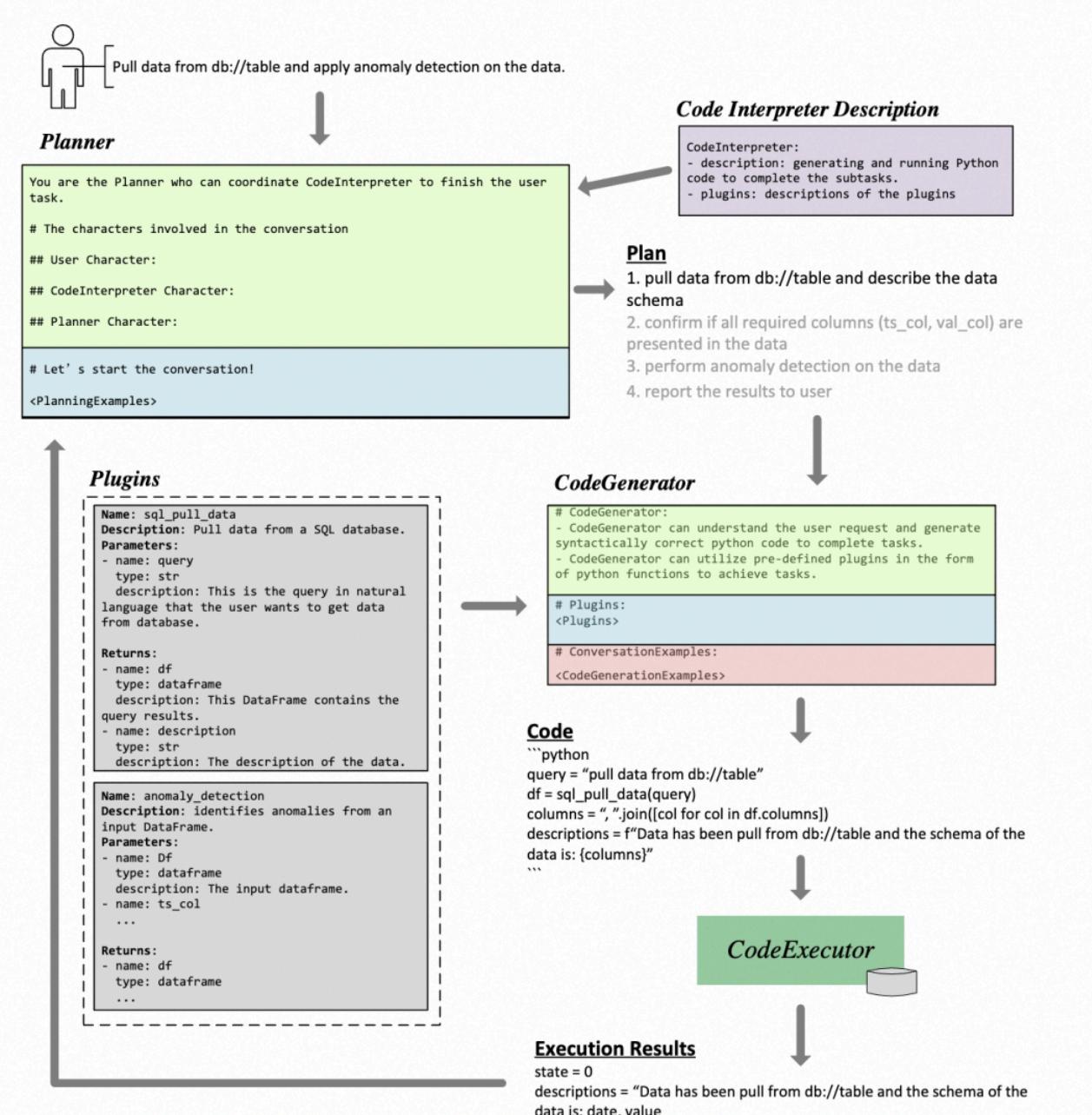
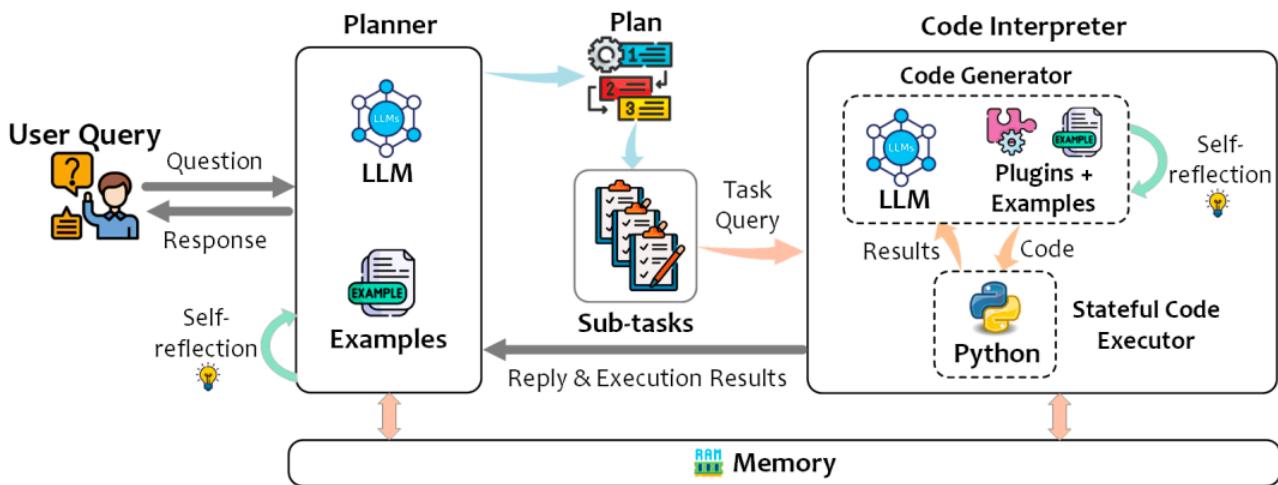


Figure 3: Workflow of TaskWeaver

TaskWeaver的工作流程涉及几个关键组件和过程,以下是工作流程的概览。它由三个关键组件组成：规划器（Planner）、代码生成器（CG）和代码执行器（CE）。代码生成器和代码执行器由代码解释器（CI）组成。

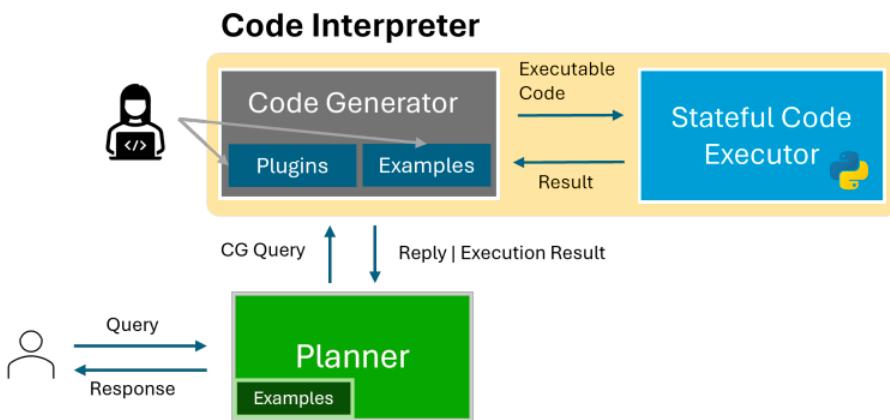


Figure 2: The overview of TaskWeaver

论文里提到的后续的多agent方向探索，可以与autoGen结合

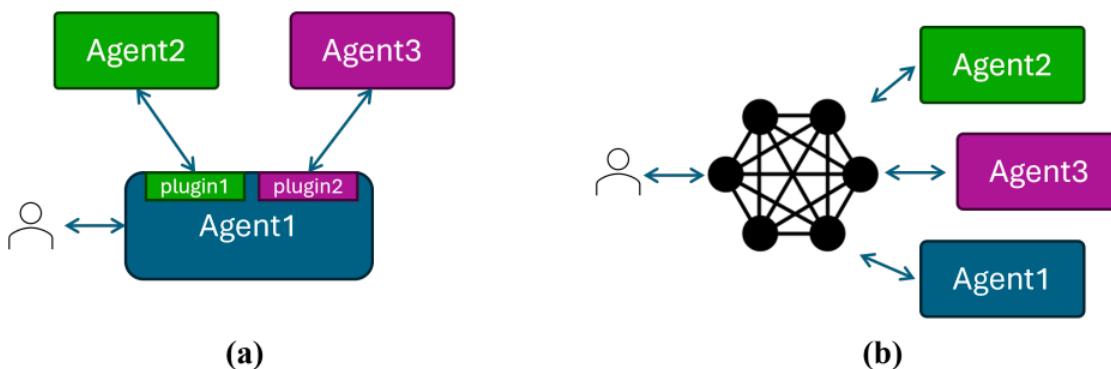
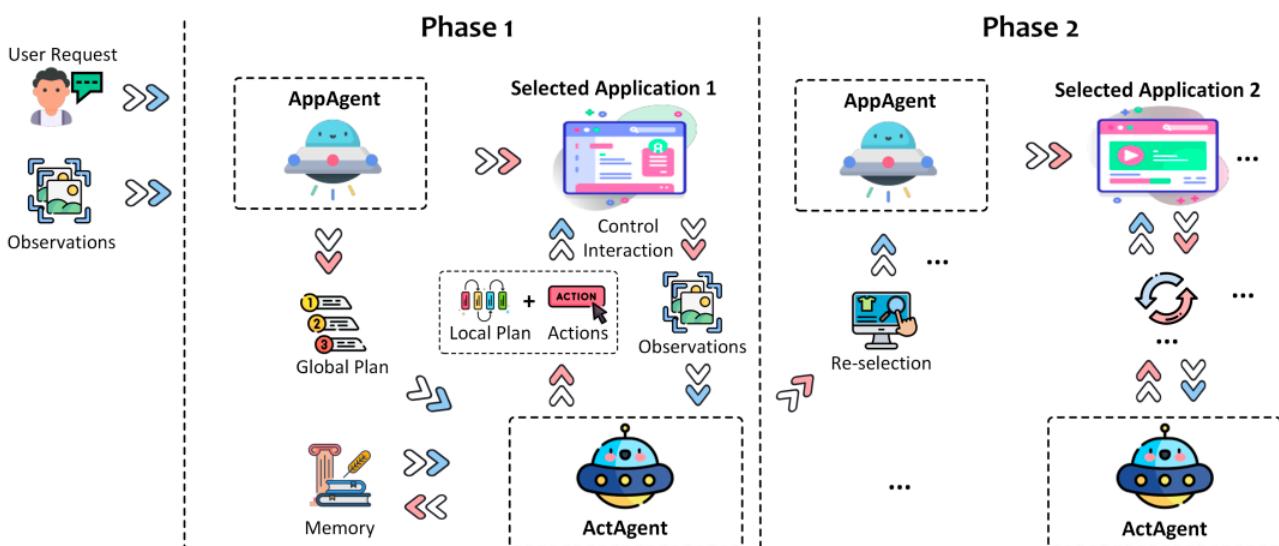


Figure 4: TaskWeaver in a multi-agent environment.

- 微软UFO

git: <https://github.com/microsoft/UFO>

UFO是面向Windows 系统的Agent，结合自然语言和视觉操作Windows GUI



UFO (UI-Focused Agent) 的工作原理基于先进的视觉语言模型技术，特别是GPT-Vision，以及一个独特的双代理框架，使其能够理解和执行Windows操作系统中的图形用户界面 (GUI) 任务。以下是UFO工作原理的详细解释：

1. 双代理框架 双代理架构：UFO由两个主要代理组成，AppAgent和ActAgent，分别负责应用程序的选择与切换，以及在这些应用程序内执行具体动作。应用程序选择代理（AppAgent）：负责决定为了完成用户请求需要启动或切换到哪个应用程序。它通过分析用户的自然语言指令和当前桌面的屏幕截图来做出选择。一旦确定了最适合的应用程序，AppAgent会制定一个全局计划来指导任务的执行。动作选择代理（ActAgent）：一旦选择了应用程序，ActAgent就会在该应用程序中执行具体的操作，如点击按钮、输入文本等。ActAgent利用应用程序的屏幕截图和控件信息来决定下一步最合适的操作，并通过控制交互模块将这些操作转化为对应用程序控件的实际动作。

2. 控制交互模块 UFO的控制交互模块是将代理识别的动作转换为应用程序中实际执行的关键组成部分。这个模块使UFO能够直接与应用程序的GUI元素进行交互，执行如点击、拖动、文本输入等操作，而无需人工干预。

3. 多模态输入处理 UFO能够处理多种类型的输入，包括文本（用户的自然语言指令）和图像（应用程序的屏幕截图）。这使UFO能够理解当前GUI的状态、可用控件和它们的属性，从而做出准确的操作决策。

4. 用户请求解析 当接收到用户的自然语言指令时，UFO首先解析这些指令，以确定用户的意图和所需完成的任务。然后，它将这个任务分解成一系列子任务或操作步骤，这些步骤被AppAgent和ActAgent按顺序执行。

5. 应用程序间的无缝切换 如果完成用户请求需要多个应用程序的操作，UFO能够在这些应用程序之间无缝切换。它通过AppAgent来决定何时以及如何切换应用程序，并通过ActAgent在每个应用程序中执行具体的操作。

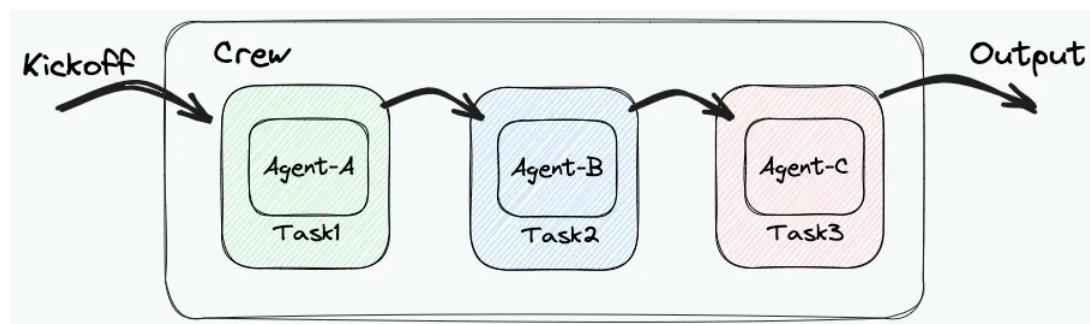
6. 自然语言命令到GUI操作的映射 UFO的核心功能之一是将用户的自然语言命令映射到具体的GUI操作上。这一过程涉及到理解命令的意图，识别相关的GUI元素，以及生成和执行操作这些元素的动作。通过这种方式，UFO可以自动完成从文档编辑和信息提取到电子邮件撰写和发送等一系列复杂的任务，大大提高用户在Windows操作系统中工作的效率和便捷性。

- CrewAI

git: <https://github.com/joaomdmoura/crewAI>

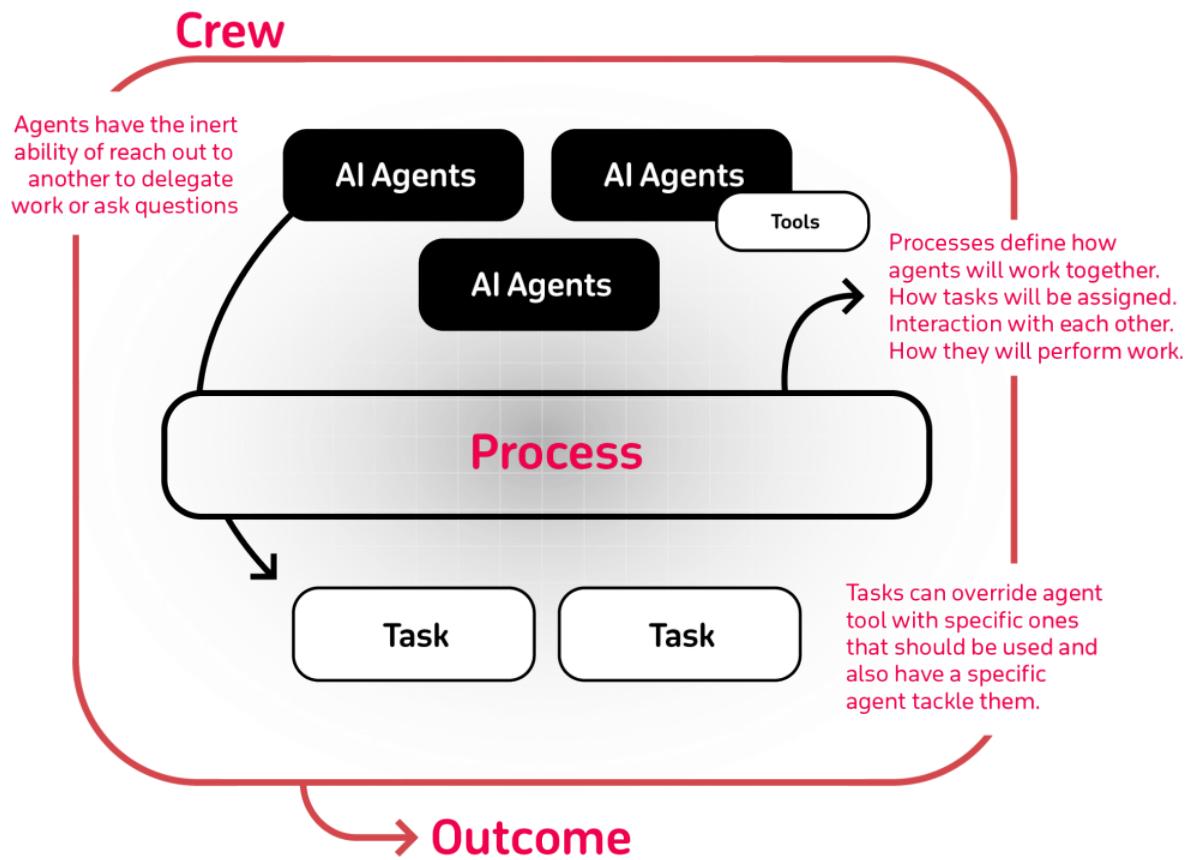
site: <https://www.crewai.com/>

基于langchain的Multi-agent框架



Crew 在 CrewAI 中是代理人、任务和过程相结合的容器层，是任务执行的实际场所。作为一个协同合作的环境，Crew 提供了代理人之间的交流、合作和按照规定过程执行任务的平台。通过 Crew 的设计，代理人能够更好地协作并以高效的方式完成任务。支持顺序结构和层级结构的agents。

CrewAI的优点：与LangChain生态结合，CrewAI提供了 Autogen 对话代理的灵活性和 ChatDev 的结构化流程方法，但没有僵化。CrewAI 的流程设计为动态且适应性强，可无缝融入开发和生产工作流程。



- **AgentScope**

git: https://github.com/modelscope/agentscope/blob/main/README_ZH.md

阿里开源的Multi-agent框架，亮点是支持分布式框架，并且做了工程链路上的优化及监控。

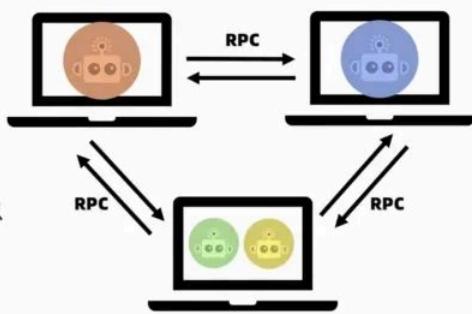
Multi-Agent框架

| 框架 | 易用性 | 鲁棒性 | 多模态 | 分布式&并行优化 |
|-------------------|-------------------------------|-----------------------|----------------------------------|------------------------|
| AutoGen | Conversation programming | 内置简单exception处理/retry | 提供Multimodal agent | 有异步
无分布式/并行 |
| MetaGPT | Team和Environment控制
交流顺序/信息 | 内置简单exception处理/retry | - | 有异步
无分布式/并行 |
| CAMEL | 由role-playing sessions组成应用 | 内置简单exception处理/retry | - | 并行example
无分布式/并行 |
| AgentScope | Procedure-oriented编程；
易用易读 | 面向开发者和Agent | service原生支持
多模态数据生成/
储存/传递 | 基于Actor的分布式
自动优化执行流 |

AgentScope 分布式解决方案

- 基于 RPC 的 Agent 间通信

- 各 Agent 都可作为独立的 RPC Server 进程启动
- 通信过程对用户透明
- 支持单机多进程、多机多进程以及混合模式



AgentScope 分布式解决方案

- 单机和分布式共用一套主流程代码

- 被调用的 Agent 立即返回占位符(placeholder)避免阻塞主流程
- Placeholder 是指向实际值 (real) 的指针

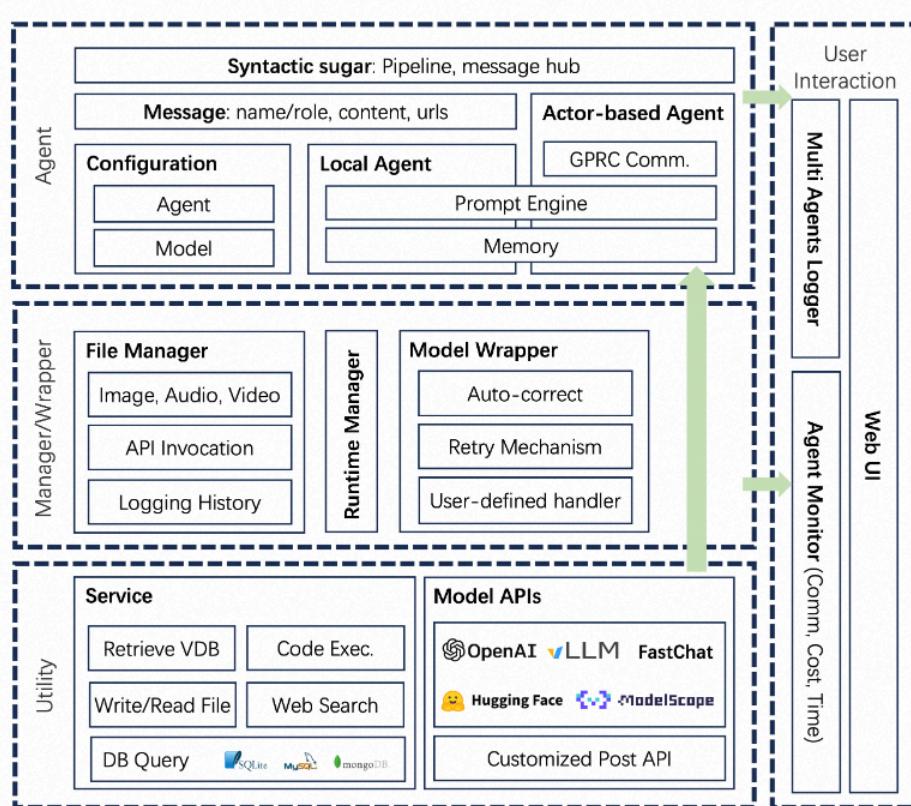
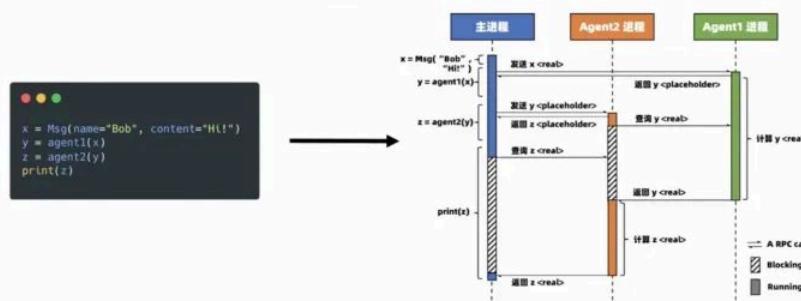
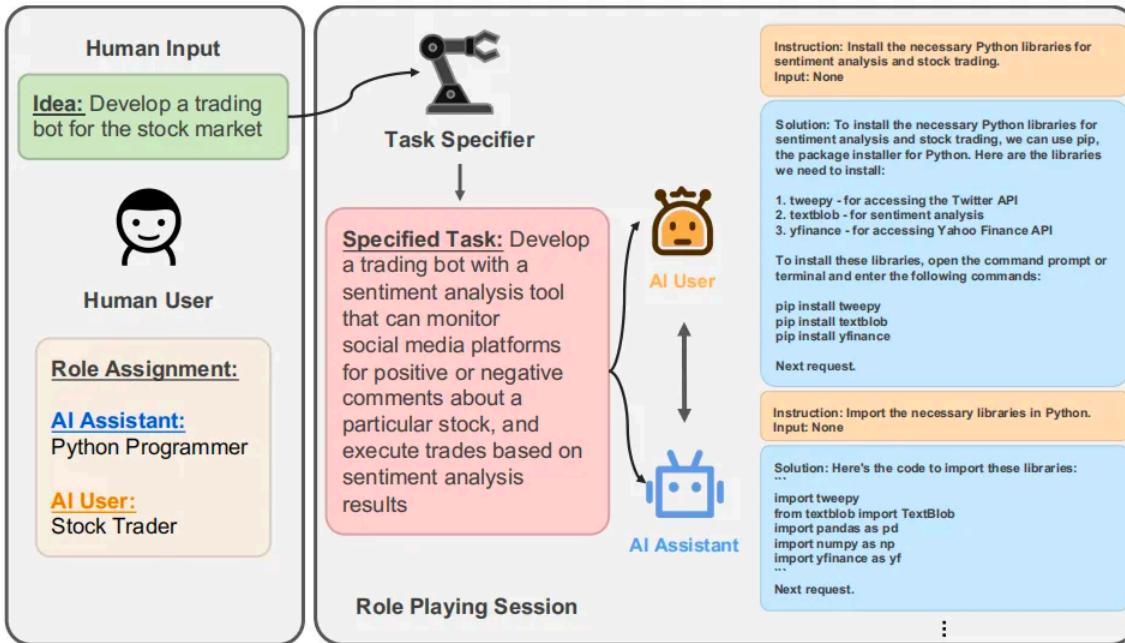


Figure 1: Architecture of AgentScope.

- Camel

早期Multi-Agent项目，实现agent间的一对一对话，文档较少，除了git和一个站点外没有找到太多有用信息。



03

Agent框架总结

单智能体= 大语言模型 (LLM) + 观察 (obs) + 思考 (thought) + 行动 (act) + 记忆 (mem)

多智能体=智能体 + 环境 + SOP + 评审 + 通信 + 成本

多智能体优点:

1. 多视角分析问题：虽然LLM可以扮演很多视角，但会随着system prompt或者前几轮的对话快速坍缩到某个具体的视角上；
2. 复杂问题拆解：每个子agent负责解决特定领域的问题，降低对记忆和prompt长度的要求；
3. 可操控性强：可以自主的选择需要的视角和人设；
4. 开闭原则：通过增加子agent来扩展功能，新增功能无需修改之前的agent；
5. (可能) 更快的解决问题：解决单agent并发的问题；

缺点:

1. 成本和耗时的增加；
2. 交互更复杂、定制开发成本高；
3. 简单的问题single Agent也能解决；

多智能体能解决的问题:

1. 解决复杂问题；
2. 生成多角色交互的剧情；

Multi-Agent并不是Agent框架的终态，Multi-Agent框架是当前有限的LLM能力背景下的产物，更多还是为了解决当前LLM的能力缺陷，通过LLM多次迭代、弥补一些显而易见的错误，不同框架间仍然存在着极高的学习和开发成本。随着LLM能力的提升，未来的Agent框架肯定会朝着更加的简单、易用的方向发展。

04

能做什么？

可能的方向

游戏场景（npc对话、游戏素材生产）、内容生产、私域助理、OS级别智能体、部分工作的提效

Multi-Agent框架

多agent应该像人类的大脑一样，分工明确、又能一起协作，比如，大脑有负责视觉、味觉、触觉、行走、平衡，甚至控制四肢行走的区域都不一样。

参考MetaGPT和AutoGen生态最完善的两个Multi-Agent框架，可以从以下几个角度出发：

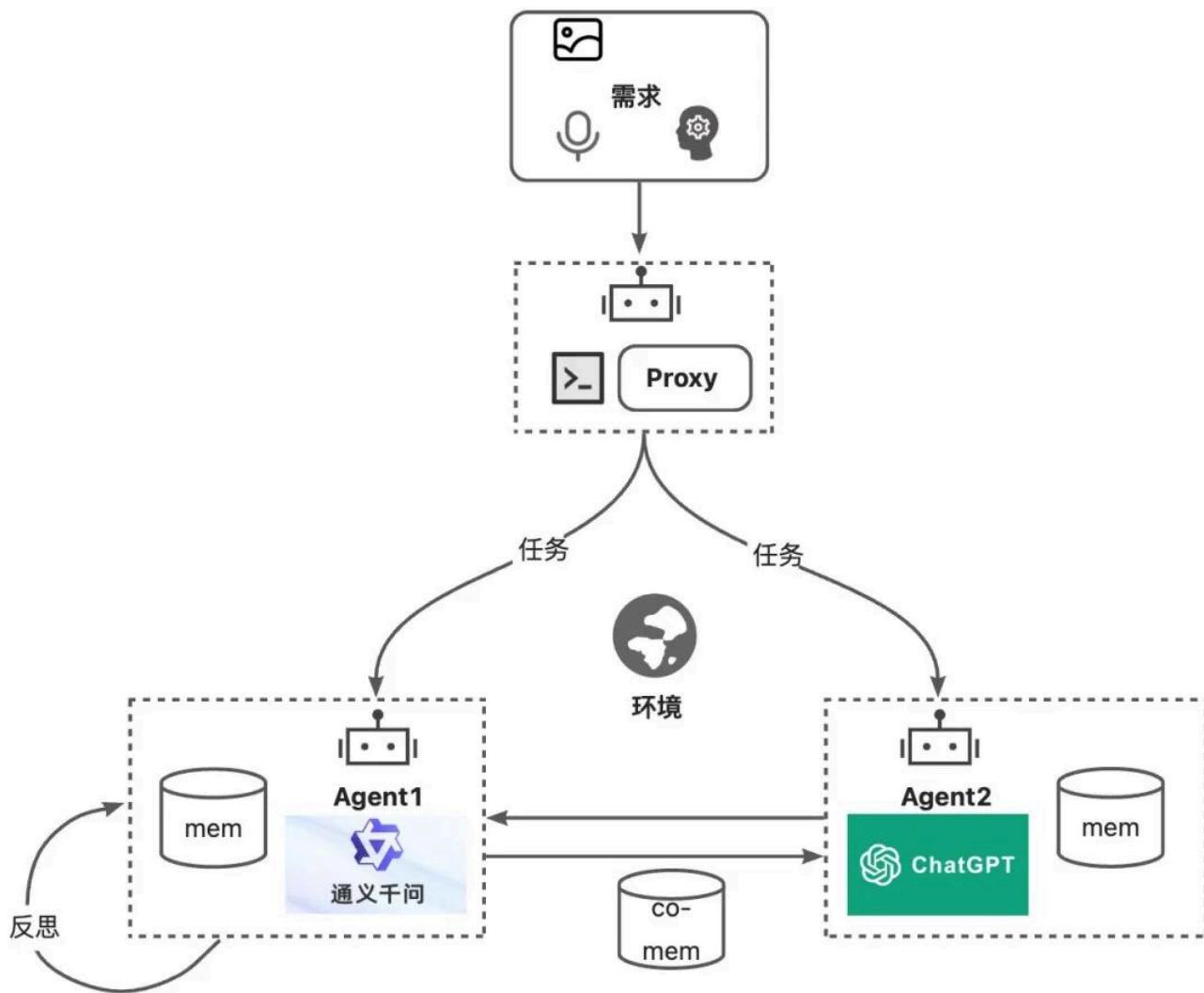
环境&通讯：Agent间的交互，消息传递、共同记忆、执行顺序，分布式agent，OS-agent

SOP：定义SOP，编排自定义Agent

评审：Agent健壮性保证，输入输出结果解析

成本：Agent间的资源分配

Proxy：自定义proxy，可编程、执行大小模型



Single Agent框架

执行架构优化：论文数据支撑

CoT to XoT，从一个thought一步act到一个thought多个act，从链式的思考方式到多维度思考；

长期记忆的优化：

具备个性化能力的agent，模拟人的回想过程，将长期记忆加入agent中；

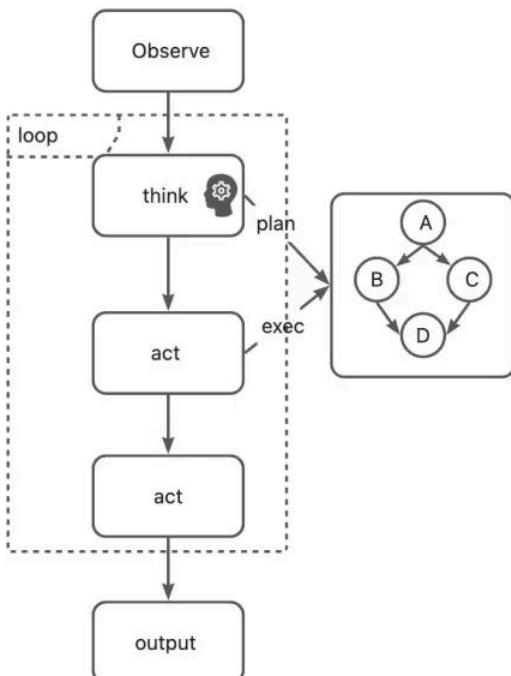
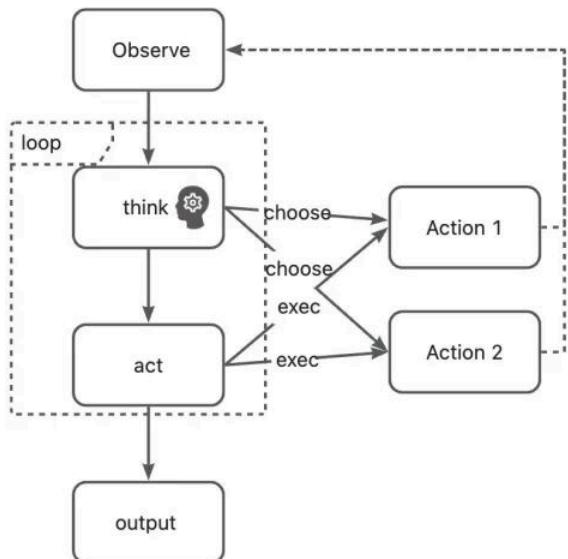
多模态能力建设：

agent能观察到的不仅限于用户输入的问题，可以加入包括触觉、视觉、对周围环境的感知等；

自我思考能力：主动提出问题，自我优化；

并行任务plan&act

串行任务ReAct



- 其他

部署: Agent以及workflow的配置化及服务化, 更长远的还需要考虑分布式部署

监控: Multi-Agent可视化、能耗与成本监控

RAG: 解决语义孤立问题

评测: agent评测、workflow评测、AgentBench

训练语料: 数据标记、数据回流

业务选择: Copilot 还是 Agent ? Single Agent 还是Multi-Agent?



如何学习AI大模型?

我在一线互联网企业工作十余年里，指导过不少同行后辈。帮助很多人得到了学习和成长。

我意识到有很多经验和知识值得分享给大家，也可以通过我们的能力和经验解答大家在人工智能学习中的很多困惑，所以在工作繁忙的情况下还是坚持各种整理和分享。但苦于知识传播途径有限，很多互联网行业朋友无法获得正确的资料得到学习提升，故此将并将重要的AI大模型资料包括AI大模型入门学习思维导图、精品AI大模型学习书籍手册、视频教程、实战学习等录播视频免费分享出来。

| L1阶段-实践理解大模型 | | | | L2阶段-AI大模型应用开发工程实践 | | | |
|------------------------|---|--|------|--|--|--|---|
| 通俗理解 | 通过真正的深入了解ChatGPT，文心一言，星火大模型，智谱清言等各类大模型工具，真正掌握如何提升工作效率，而不是简单的聊天 | 1: 远程实践项目（选项一）：网文助手 | 通过理解 | 通过程序灵活运用大模型API接口，借助现有的大模型的能力，构建适合我们行业需要的应用程序。 | 1: 远程实践项目（选项一）： | 文档智能助手 | 项目简介：大型预训练模型来帮助用户更高效地创建、编辑、理解和管理文档的智能工具。 |
| 能力收获 | 真正明白什么是大模型。合理利用大模型的各种AIGC工具来帮助我们减少重复劳动，提升工作效率。 | 项目简介：用OpenAI开源API做网文作者，专门为作者提供量产支持。 | 能力收获 | 基于GPT现有的模型进行优化，并学会利用GPT创造自己第一个商业级别的App | 项目实践安排及应用举例 | (AGI大模型日新月异，它的智商与日俱增，实践安排会根据大模型发展情况不断调整) | 项目简介：大型预训练模型来帮助用户更高效地创建、编辑、理解和管理文档的智能工具。 |
| 适用人群 | 会操作手机或者电脑的学生与职场人群 | 2: 远程实践项目（选项一）：运营文案高手 | 适用人群 | 会操作电脑，想了解如何利用接口API，设计出更切中用户需要的交互体验的产品，希望提升自己的工作效率以及学习效率的人。 | 2: 远程实践项目（选项一） | 医学命名实体识别系统 | 项目简介：大型预训练模型来帮助用户更高效地创建、编辑、理解和管理文档的智能工具。 |
| 课程内容 | 1.1 深入理解大模型
1.2 文心一言，星火，智谱清言，盘古等各类AIGC工具的灵活运用
1.3 大模型成功背后的原理
1.4 GPT-3.5-GPT-4灵活应用
1.5 python编程入门
1.6 提示词工程prompts | 项目简介：大模型是你的写作伙伴，帮你对文章进行润色；帮你构思故事情节；帮你拓展整个故事，毫无疑问大模型将帮助你能够在内容创作上成为一位高产作家。 | 上课形式 | 线上直播课+远程项目实战+在线答疑 | 项目简介：大型预训练模型来帮助用户更高效地创建、编辑、理解和管理文档的智能工具。 | 项目简介：大型预训练模型来帮助用户更高效地创建、编辑、理解和管理文档的智能工具。 | 项目简介：大型预训练模型来帮助用户更高效地创建、编辑、理解和管理文档的智能工具。 |
| 智力评估 | 各种大模型的订阅费用，一般国内的模型免费或者很低费用，国外OpenAI的订阅GPT-3.5/GPT-4大概是10\$/月,按照使用量付费。 | 3: 远程实践项目（选项一）：代码自动生成器 | 智力评估 | 2.1 Python强化学习
2.2 大模型的优点缺点深入理解；
2.3 理解Function Calling提升大模型的准确度；
2.4 RAG&Embedding优化大模型，让专业领域知识初步实现智能化；
2.5 向量数据库精调
2.6 GPTs&Assistant API打造自己的GPT | 10课时左右 | 10课时左右 | 项目简介：大型预训练模型来帮助用户更高效地创建、编辑、理解和管理文档的智能工具。 |
| 上课形式 | 线上直播课+远程项目实战+在线答疑 | 项目简介：基于提示工程的代码生成器实现。大模型可以通过代码安全，代码生成，代码优化，生成测试用例方法帮助程序员提高编程效率和代码质量。大模型最擅长的是编程方面的辅助，给程序员在软件开发的整个工程项目期为程序员们提供支持。 | 总课时 | gpt-4 使用付费方式
input费用：0.03\$/1000tokens
output费用：0.06\$/1000tokens
1M Tokens大概是在45S (预计使用4个月)
gpt-3.5约为gpt-4的1/20 | 10课时左右 | 10课时左右 | 项目简介：大型预训练模型来帮助用户更高效地创建、编辑、理解和管理文档的智能工具。 |
| 总课时 | 12课时左右 | | | | | | 项目简介：大型预训练模型来帮助用户更高效地创建、编辑、理解和管理文档的智能工具。 |
| L3阶段-大模型应用架构进阶实践 | | | | L4阶段-打造适合自己需要的私有大模型 | | | |
| 通俗理解 | 结合大模型以及各种大模型应用框架，让大模型在某个领域具备可以应用的能力 | 1: 远程实践项目（选项一）：基于图片的智能信息检索问答 | 通俗理解 | 结合我们在自己垂直领域对行业的理解和积累的私有知识库，对大模型进行微调，并打造自己的私有大模型 | 1: 远程实践项目（选项一）： | 医疗领域智能医生私有模型 | 项目简介：在医疗领域，智能医生私有大模型是指专为医疗机构或医疗技术公司定制开发的、用于模拟医生诊断和治疗过程的人工智能模型。 |
| 能力收获 | 对大模型进行初步精调，让大模型在某个领域的任务表现可以商用 | 项目简介：利用大型预训练模型来处理和理解图片内容，并据此回答用户提出的相关问题的系统。该系统能够分析图片中的物体、场景、活动和文字等信息，然后根据用户的查询生成准确的回答。系统能够帮助我们在物体识别、场景理解、情感分析等多个方面为用户提供一个直观的方式来获取和理解图片中的信息，在新闻报道、社交媒体分析、教育、旅游等领域给予用户快速的图像信息获取。 | 能力收获 | 学会大模型的微调，并结合自己的商业数据打造出一个自己在商业垂直领域的私有模型 | 项目实践安排及应用举例 | (AGI大模型日新月异，它的智商与日俱增，实践安排会根据大模型发展情况不断调整) | 项目简介：智能医生大模型通常基于大量的医疗数据训练，包括电子病历、医学文献、临床指南、药物信息等，以便能够提供专业的医疗建议和决策支持。通常大模型可以辅助诊断，并能结合患者病历中的基因信息和生活习惯推荐个性化的治疗方案。另外还可以承担患者的教育工作，向患者提供健康建议和疾病预防知识。 |
| 上课形式 | 线上直播课+远程项目实战+在线答疑 | 2: 远程实践项目（选项一）：无人数字人直播机器人客服 | 课堂内容 | 4.1 开源模型的私有化部署；
4.2 模型微调Fine-Tuning；
4.3 微调生成医疗领域智能医生私有模型
4.4 开源的大模型GPTs微调
4.5 多模态技术详解
4.6 多模态技术项目实战代码生成器 | 14课时左右 | 14课时左右 | 项目简介：在医疗领域，智能医生私有大模型是指专为医疗机构或医疗技术公司定制开发的、用于模拟医生诊断和治疗过程的人工智能模型。 |
| 启停时 | 14课时左右 | 项目简介：大模型打造的客服系统可以模拟人类客服的行为，为用户提供自动化的问答服务、解决问题和处理投诉等。 | 适用地点 | 理解特定行业的特定需求，希望通过学习大模型赋能自己的行业，提升自我价值，甚至希望改变自己公司的工作效率的个人或者公司 | 启停时 | 启停时 | 项目简介：智能医生大模型通常基于大量的医疗数据训练，包括电子病历、医学文献、临床指南、药物信息等，以便能够提供专业的医疗建议和决策支持。通常大模型可以辅助诊断，并能结合患者病历中的基因信息和生活习惯推荐个性化的治疗方案。另外还可以承担患者的教育工作，向患者提供健康建议和疾病预防知识。 |
| 课程内容 | 3.1 不同语言模型GPTs的特点分析；
3.2 LangChain实现与不同语言模型的交互；
3.3 Semantic Kernel
3.4 智能体Agents组件应用；
3.5 MetaGPT的技术方案分析；
3.6 多智能体协作构建代码生成器实践战
3.7 基于Agent的智能面试实践战；
3.8 无人数字人直播机器人客服实践战；
3.9 基于Agents打造AI面试机器人实践战； | 3: 远程实践项目（选项一）：基于Agents打造AI模拟面试机器人 | 智力评估 | 服务器租用（需求：A100 2-10片）
微软云：A100 80GB 1卡 ¥15/小时
臻界算力：A100 40G 2卡 ¥6800/月
API费用同L2阶段 | 启停时 | 启停时 | 项目简介：大模型在医学任务中表现出色，因为它们能够捕捉到文本中的复杂关系和上下文信息。这些模型通常需要使用大量的标注医疗数据进行训练，以便能够准确地识别各种医学实体。 |
| 智力评估 | 服务器租用（需求：A100 1-10片）
微软云：A100 80GB 1卡 ¥15/小时
臻界算力：A100 40G 2卡 ¥6800/月
API费用同L2阶段 | 项目简介：基于Agents打造的AI模拟面试机器人项目是一个利用智能体（Agents）技术和人工智能模型来模拟真实面试过程的项目。AI模拟面试机器人可以扮演面试官的角色，与求职者进行交互，提出问题，评估回答，并提供反馈。面试者可以随时向所扮演的AI面试官提出问题，而且AI面试官也能给出上下文相对比较准确的回答，帮助面试者更好的了解雇主企业和职位情况。这一改进使得AI面试官真正具备了智商，提升了工作效率和沟通效率。 | 上课形式 | 线上直播课+远程项目实战+在线答疑 | 启停时 | 启停时 | 项目简介：大模型在医学任务中表现出色，因为它们能够捕捉到文本中的复杂关系和上下文信息。这些模型通常需要使用大量的标注医疗数据进行训练，以便能够准确地识别各种医学实体。 |
| 能力收获 | 对大模型进行初步精调，让大模型在某个领域的任务表现可以商用 | 项目简介：基于Agents打造的AI模拟面试机器人项目是一个利用智能体（Agents）技术和人工智能模型来模拟真实面试过程的项目。AI模拟面试机器人可以扮演面试官的角色，与求职者进行交互，提出问题，评估回答，并提供反馈。面试者可以随时向所扮演的AI面试官提出问题，而且AI面试官也能给出上下文相对比较准确的回答，帮助面试者更好的了解雇主企业和职位情况。这一改进使得AI面试官真正具备了智商，提升了工作效率和沟通效率。 | 总课时 | 12课时左右 | 启停时 | 启停时 | 项目简介：大模型在医学任务中表现出色，因为它们能够捕捉到文本中的复杂关系和上下文信息。这些模型通常需要使用大量的标注医疗数据进行训练，以便能够准确地识别各种医学实体。 |
| L5阶段-深入大模型原理打造进阶的私有大模型 | | | | L6阶段-深入大模型原理打造进阶的私有大模型 | | | |
| 通俗理解 | 训练入门级别的基础大模型，理解大模型的基本底层逻辑，并能根据大模型的基本标准和准则以及评估指标对大模型进行增量训练，最终得到一个入门级别的基础大模型。 | 医疗领域智能医生/金融助手/法律助手(具体基于公司承接的具体项目为准) | 通俗理解 | 深入理解大模型的基础算法，研究算法的优化方案，最终能够为大模型的演进贡献自己的力量 | 项目实践安排及应用举例 | (AGI大模型日新月异，它的智商与日俱增，实践安排会根据大模型发展情况不断调整) | 金融行业/医疗行业/制造业...等等行业的智能体(具体基于公司承接的具体项目为准) |
| 能力收获 | 深入了解大模型各种直法，以及结合大模型的发展规律，按指引进行模型基础训练或进行增量训练 | 项目简介：Mistral-7B/ChatGLM-3-6B 基于开源大模型 ChatGLM-3-6B基础，投入大量中文数据集和大模型公开数据集进行增量训练，是为了提升模型在法律领域的专业能力。通过增量训练，ChatGLM-3-6B模型不仅具备强大的中文处理能力，还能够在法律领域提供专业法律服务和支持。这种模型可以应用于法律咨询、法律教育、法律研究等多个场景，帮助法律专业人士和公众更好地理解并应用法律知识。 | 能力收获 | 训练更高级别的大模型基座，设置大模型的全量微调或增量微调，选择和修改预训练模型底层算法，并提升大模型的性能的能力 | 课堂内容 | 学习高级大模型底层算法，并完成高级大模型训练技巧，以满足不同业务需求构建特定的大模型，并希望在A类法领域有自己的知名度，发表高质量paper的科研人员。 | 对于国央企客户以及许多对数据合规性和模型输出有严格要求的应用来说，确保训练数据的合规性是非常重要的，这样的要求在金融、医疗、法律和其他敏感领域尤其突出。独立训练模型可以确保从基底开始独立训练的模型在输出上符合中国法律的要求，适用于国央企和其他对合规性有高要求的客户。这样的模型可以用于提供法律咨询服务、撰写法律文件、分析法律案例等多种应用，同时确保输出的准确性和合规性。 |
| 适用人群 | 学习入门级别的基础大模型，对AI算法有一定研究，尤其适合人工智能方面希望有科技公司提供实践项目实习经验，并能够依托实习找到好工作的技术工作者 | 详细信息请参考《大模型微调进阶实训工程》的具体课程 | 课堂内容 | 详细信息请参考《甚基训练学习营&大模型算法学习营》的具体课程 | 启停时 | 启停时 | 其他AI领域的商业化实践项目如：智慧城市、智慧社区、智慧车间、智慧交通、智慧警察等各智能领域的商业项目应用都在其中。 |
| 课程内容 | 详细信息请参考《大模型微调进阶实训工程》的具体课程 | 启停时 | 启停时 | 启停时 | 启停时 | 启停时 | 启停时 |
| 智力评估 | 服务器租用（需求：A100 6-20片）
臻界算力：A100 40G 2卡 ¥6800/月
微软云：A100 80GB 1卡 ¥15/小时
腾讯云：阿里云：租罄
显卡价格：A100 80GB 16W/片 | 启停时 | 启停时 | 启停时 | 启停时 | 启停时 | 启停时 |
| 上课形式 | 线下实习+一对一实习老师指导 | 启停时 | 启停时 | 启停时 | 启停时 | 启停时 | 启停时 |
| 总课时 | 6个月脱产学习 | 启停时 | 启停时 | 启停时 | 启停时 | 启停时 | 启停时 |

第一阶段：从大模型系统设计入手，讲解大模型的主要方法；

第二阶段：在通过大模型提示词工程从Prompts角度入手更好发挥模型的作用；

第三阶段：大模型平台应用开发借助阿里云PAI平台构建电商领域虚拟试衣系统；

第四阶段：大模型知识库应用开发以LangChain框架为例，构建物流行业咨询智能问答系统；

第五阶段：大模型微调开发借助以大健康、新零售、新媒体领域构建适合当前领域大模型；

第六阶段：以SD多模态大模型为主，搭建了文生图小程序案例；

第七阶段：以大模型平台应用与开发为主，通过星火大模型，文心大模型等成熟大模型构建大模型行业应用。



640套AI大模型报告合集

学会后的收获: 🌟

· 基于大模型全栈工程实现（前端、后端、产品经理、设计、数据分析等），通过这门课可获得不同能力；

· 能够利用大模型解决相关实际项目需求：大数据时代，越来越多的企业和机构需要处理海量数据，利用大模型技术可以更好地处理这些数据，提高数据分析和决策的准确性。因此，掌握大模型应用开发技能，可以让程序员更好地应对实际项目需求；

· 基于大模型和企业数据AI应用开发，实现大模型理论、掌握GPU算力、硬件、LangChain开发框架和项目实战技能，学会Fine-tuning垂直训练大模型（数据准备、数据蒸馏、大模型部署）一站式掌握；

· 能够完成时下热门大模型垂直领域训练能力，提高程序员的编码能力：大模型应用开发需要掌握机器学习算法、深度学习 框架等技术，这些技术的掌握可以提高程序员的编码能力和分析能力，让程序员更加熟练地编写高质量的代码。



| | |
|----------------------------------|----------|
| 001-先导篇【课程永久更新】 | 2024/4/7 |
| 002-具体事项和更新周期 | 2024/4/7 |
| 003-课程内容介绍（能学到哪些内容） | 2024/4/7 |
| 004-AIGC的前世今生 | 2024/4/7 |
| 005-各个大语言模型有什么区别？又该如何去选择？ | 2024/4/7 |
| 006-【课程正式开始】私有化大模型是什么？ | 2024/4/7 |
| 007-知识库，微调，AI Agent，分别是什么？（简单普及） | 2024/4/7 |
| 008-【重点】开源是什么？闭源是什么？两者有何区别？ | 2024/4/7 |
| 009-提示词、模型参数量、模型大小、模型精度 | 2024/4/7 |
| 010-提升词典案例 | 2024/4/7 |
| 011-提示词（作业讲解） | 2024/4/7 |
| 012-【0基础必看】计算机基础知识 | 2024/4/7 |
| 013-【实操前铺垫】云服务器的基本概念 | 2024/4/7 |
| 014-云服务器基础篇（上） | 2024/4/7 |
| 015-云服务器基础篇（下） | 2024/4/7 |
| 016-计算机基础篇：linux指令（1） | 2024/4/7 |
| 017-计算机基础篇：Linux（2） | 2024/4/7 |
| 018-计算机基础篇：linux（3） | 2024/4/7 |
| 019-【偏难，看不懂可跳过】机器学习和神经网络基础知识 | 2024/4/7 |
| 020-AI以及大模型的进阶核心知识点（重要，请反复观看！） | 2024/4/7 |

大模型落地应用案例119页PPT



| |
|---|
| 202402月更新-【AI金融新纪元】系列报告（一）：金融垂类大模型试用体验.pdf |
| 202402月更新-2023 移动通信与AI融合的数据格式和模型建议书（第一阶段.pdf |
| 202402月更新-2023 爱分析·企业大模型市场厂商评估报告：滴普科技.pdf |
| 202402月更新-2023 产业大模型应用白皮书.pdf |
| 202402月更新-2023 大模型落地应用案例集.pdf |
| 202402月更新-2023 金融大模型技术创新与应用探索报告.pdf |
| 202402月更新-2023年AIGC场景应用展望研究报告.pdf |
| 202402月更新-2023 前沿大模型的风险、安全与治理报告.pdf |
| 202402月更新-2023 中国年度报告.pdf |
| 202402月更新-2023 年代人工智能基础设施白皮书.pdf |
| 202402月更新-2024 各大模型建设及评估分类评价报告.pdf |
| 202402月更新-2024 语言模型能力测评报告.pdf |
| 202402月更新-AI大模型产业正发生哪些变化？.pdf |
| 202402月更新-AI时代领先者，大模型+大模型推动AGI落地.pdf |
| 202402月更新-ChatGPT模型大更新，省级数据局陆续挂牌.pdf |
| 202402月更新-GPT商店下周正式上线，美图开放AI视觉大模型.pdf |
| 202402月更新-Meta2024Q1收入指引超预期，发布开源大模型CodeLlama70B.pdf |
| 202402月更新-OpenAI发布重大更新，大模型使用成本将进一步降低.pdf |
| 202402月更新-OpenAI宣布将上线“自定义GPT商店”，网易有道发布教育大模型子曰.pdf |
| 202402月更新-产业深度：大模型赋能座舱，智能座舱新战场.pdf |

LLM面试题合集

| | | | | | | | | | | |
|--------------------------------------|-----------------|--------------------|-------|---------------------------|-----------------------|-----------------------|------------------------|---------------------------|---------------------|----------|
| 1-大模型（LLMs）基础面 | 2024/3/23 19:45 | Foxit Phantom P... | 482 | 「65页PDF」深入学习精华汇总-产品经理... | 【重磅福利】人工智能产品经理的新起点... | AI产品经理，如何面对数据挖掘？ | AI产品经理的必修课：系统化思维 | AI产品经理的价值和未来 学习前军老师分... | AI产品经理入门手册（上） | AI产品经理入门 |
| 2-Layer normalization 篇 | 2024/3/23 19:45 | Foxit Phantom P... | 489 | AI产品经理需要了解的技术知识：语音识别技术... | AI产品经理需要了解的数据科学入门 | AI产品经理需要了解的产品设计... | AI时代的产品经理：产品设计... | To B产品经理转型AI产品经理... | 产品经理，如何转行到人工智能机器人领域 | 成为经理可以这样 |
| 3-LLMs 激活函数篇 | 2024/3/23 19:45 | Foxit Phantom P... | 375 | 从博客与微博的区别，谈社交创新 | 当产品经理遇上人工智能 | 当人们恐惧人工智能时，内心到底在恐惧... | 对人工智能产品发展的几点认识 | 福利《从互联网产品经理到AI产品经理》... | 干货品鉴必看 | 产品经理可以这样 |
| 4-Attention 升级面 | 2024/3/23 19:45 | Foxit Phantom P... | 411 | 关于火热的「人工智能」，这里最有你关心... | 关于社交，有可能这是最全的了 | 互联网PM转型人工智能PM之路 | 进击的人工智能：从严肃角度，深度解析「... | 两个方面浅析，为什么人工智能可以替代产... | 蚂蚁品经何伟 | 产品经理可以这样 |
| 5-transformers 操作篇 | 2024/3/23 19:45 | Foxit Phantom P... | 228 | | | | | | | |
| 6-LLMs 损失函数篇 | 2024/3/23 19:45 | Foxit Phantom P... | 356 | | | | | | | |
| 7-相似度函数篇 | 2024/3/23 19:45 | Foxit Phantom P... | 175 | | | | | | | |
| 8-大模型（LLMs）进阶面 | 2024/3/23 19:45 | Foxit Phantom P... | 1,019 | | | | | | | |
| 9-大模型（LLMs）微调面 | 2024/3/23 19:45 | Foxit Phantom P... | 2,958 | | | | | | | |
| 10-LLMs 训练经验帖 | 2024/3/23 19:45 | Foxit Phantom P... | 254 | | | | | | | |
| 11-大模型（LLMs）langchain | 2024/3/23 19:45 | Foxit Phantom P... | 631 | | | | | | | |
| 12-多轮对话中让AI保持长期记忆的组件 | 2024/3/23 19:45 | Foxit Phantom P... | 362 | | | | | | | |
| 13-基于langchain RAG问答应用实践 | 2024/3/23 19:46 | Foxit Phantom P... | 376 | | | | | | | |
| 14-基于LLM+向量库的文档对话 经验面 | 2024/3/23 19:46 | Foxit Phantom P... | 2,208 | | | | | | | |
| 15-大模型 RAG 经验面 | 2024/3/23 19:46 | Foxit Phantom P... | 1,443 | | | | | | | |
| 16-LLM文档对话 —— pdf解析关键问题 | 2024/3/23 19:46 | Foxit Phantom P... | 2,191 | | | | | | | |
| 17-大模型（LLMs）RAG 版面分析—— | 2024/3/23 19:46 | Foxit Phantom P... | 662 | | | | | | | |
| 18-大模型（LLMs）RAG 版面分析—— | 2024/3/23 19:46 | Foxit Phantom P... | 483 | | | | | | | |
| 19-大模型外挂知识库优化——如何利用... | 2024/3/23 19:46 | Foxit Phantom P... | 734 | | | | | | | |
| 20-大模型外挂知识库优化——负样本样... | 2024/3/23 19:46 | Foxit Phantom P... | 705 | | | | | | | |
| 21-RAG (Retrieval-Augmented Gener... | 2024/3/23 19:46 | Foxit Phantom P... | 617 | | | | | | | |
| 22-检索增强(RAG) 优化策略篇 | 2024/3/23 19:46 | Foxit Phantom P... | 2,708 | | | | | | | |
| 23-大模型（LLMs）RAG —— 关键痛点... | 2024/3/23 19:46 | Foxit Phantom P... | 1,349 | | | | | | | |
| 24-大模型（LLMs）RAG 优化策略 —— | 2024/3/23 19:47 | Foxit Phantom P... | 1,086 | | | | | | | |

AI产品经理合集

- 1.AI大模型学习路线图
- 2.100套AI大模型商业化落地方案
- 3.100集大模型视频教程
- 4.200本大模型PDF书籍
- 5.LLM面试题合集
- 6.AI产品经理资源合集

👉 获取方式：

👉 有需要的小伙伴，可以保存图片到wx扫描二维码免费领取【保证100%免费】

