赛题二

开放原子开源基金会 OPENATOM FOUNDATION | openKylin

基于openKylin桌面环境的多智能体交互技术赛

报名网址：
https://competition.atomgit.com/competitionInfo?id=fed9abe5f32436d32ccdee2198c660d7

赛题介绍

开放原子开源基金会 OPENATOM FOUNDATION | openKylin

实战竞技赛

基于 openKylin 桌面环境的多智能体交互技术

赛题背景

随着大模型和围绕大模型相关的应用不断发展，AI已经不再局限于问答和特定应用场景，具有记忆、能够自主规划和调用系统资源和工具的智能体成为了当前AI应用的主流发展方向，操作系统/桌面环境+智能体也是openkylin在AI操作系统发展规划中的重要课题

赛题任务

本赛题要求开发者能够基于openKylin开源操作系统，在麒麟AI推理框架上，结合UKUI桌面环境实现对openKylin进行操作的智能体应用，此智能体能够自主规划、决策，具备访问系统资源，调用系统、桌面环境接口及工具的能力和记忆能力，同时能够与其它智能体应用进行协作。

openKylin + 桌面环境 + 智能体

技术价值

推动国产化生态智能升级： 将前沿的多智能体技术深度融入国产开源操作系统，是提升国产基础软件智能化水平和竞争力的重要实践，助力构建自主可控的智能计算生态。

探索人机交互新范式： 研究多智能体在桌面环境下的协同、竞争与交互，为探索更自然、高效、个性化的人机协作模式提供了重要试验场，是未来智能桌面交互的雏形。

解决复杂场景实际需求： 针对桌面环境中用户任务日益复杂的痛点，多智能体技术能有效分解任务、分配资源、协同求解，提升用户生产力和体验，具有明确的实用价值。

赛题介绍

开放原子开源基金会 OPENATOM FOUNDATION | openKylin

基于 openKylin 桌面环境的多智能体交互技术

赛题范围

为了实现基于openKylin的多智能体交互，首先需要实现openKylin和智能体的对接，即一个基于openKylin的智能体应用，这个应用可以理解为当前openKylin上的ai助手，围绕这个智能体应用完善对内的智能规划和执行能力，以及对外的协作能力，**可能的解决方案**包括：

- 基于主流模型和推理框架实现智能体应用，适配MCP和A2A协议，实现host能力
- 构建针对openKylin智能体的提示词工程
- 构建基于openKylin桌面环境的mcp server
- …

需要解决的问题

- openKylin智能体应用如何对接现有模型和推理框架
- openKylin智能体如何调用操作系统、桌面环境能力和其它工具
- openKylin智能体如何调用其它智能体，或者被其它智能体调用
- openKylin智能体如何和当前openKylin上的AI框架对接
- …

评分标准

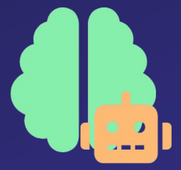
开放原子开源基金会 OPENATOM FOUNDATION | openKYLin

评审细则

1. 参赛作品是否具备理解用户指令，规划决策，调用桌面环境、系统工具接口完成用户指令的能力，20分
2. 参赛作品是否具备记忆、检索用户历史操作的能力，10分
3. 参赛作品是否具备和其它智能体交互协作能力，能够将任务和请求派分到其它智能体，20分
4. 参赛作品是否兼容主流大模型和麒麟AI框架，15分
5. 参赛作品是否具备MCP Server配置能力，15分
6. 参赛作品的原创性，10分
7. 参赛团队是否提供详细的说明文档，可以指导其他开发者运行/使用这个项目， 10分

综合评定

最终团队成绩，将在总决赛中的作品答辩环节的综合评分后得出，评出各类奖项



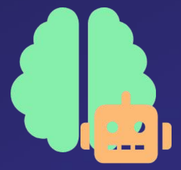
大模型AI Agent智能体技术与应用

智能体开发、MCP协议与多智能体协作

智能体开发

MCP协议

多智能体交互



核心概念介绍

智能体开发、MCP协议与多智能体协作

智能体开发　　MCP协议　　多智能体交互

AI Agent智能体概述

开放原子开源基金会 OPENATOM FOUNDATION | openKYLin

定义与特征

AI Agent（智能体）是人工智能发展到当前阶段的必然产物，它超越了传统AI（如聊天机器人）的被动响应模式，具备了自主感知、规划、决策和执行复杂任务的能力。

与仅能根据预设规则进行简单交互的聊天机器人不同，AI Agent能够理解用户意图，分解任务，调用工具，甚至进行自我学习和反思，从而在更广泛的场景中提供智能服务。

google定义：https://cloud.google.com/discover/what-are-ai-agents?hl=zh-CN

核心能力

自主感知

理解环境状态和用户需求

任务规划

分解复杂任务为可执行步骤

决策制定

基于上下文做出最优选择

执行行动

调用工具完成实际任务

与传统AI的区别

传统AI（如聊天机器人）	AI Agent智能体
被动响应式交互	主动规划与执行任务
预设规则驱动	理解意图与自我学习
简单交互能力	复杂任务分解与执行
无自主决策	能动性与自主决策

从工具到伙伴的演进

被动响应 → 规则驱动 → 主动协作 → 自主决策

智能体发展与价值

开放原子开源基金会 OPENATOM FOUNDATION | openKYLin

哲学启蒙

起源于古老的哲学思辨：

- 亚里士多德描述智能实体（约公元前350年）
- 《道德经》中"道生一"的演化思想
- 庄子"庄周梦蝶"的意识思考
- 狄德罗"聪明的鹦鹉"学说

早期探索

AI Agent概念逐渐成形：

1950s图灵将概念扩展到AI领域

1980sWooldridge引入Agent范式

- 从规则系统到决策系统的演进
- 基于规则的自主式软件程序

大模型引爆

作为"大脑"提供推理	提供历史参考
分解复杂任务	与世界交互

AI Agent区别于AI Workflow：能自主决策并循环解决问题，展现更强的自主决策能力

AI Agent的关键价值与影响

从**被动响应**的"工具"转变为能够**自主思考、规划和行动**的"伙伴"

正在**重塑个人生活和企业运营**的方方面面，从自动化日常任务到提供专业级服务

降低企业数字化门槛，通过**自然语言指令实现零学习成本**，解决传统软件平均3个月的员工培训周期问题

有效填补员工技能断层，使非IT员工的技能获取效率**提升n倍**，企业知识库复用高

未来发展前景

从**"暴力计算"到"精准效能"**的技术跃迁，小模型和低能耗架构将使AI Agent能够在端侧设备上运行

多智能体协作模式将成为主流，通过**多智能体协作规划(MCP)**和**Agent间通信(A2A)**协议实现复杂任务协同

AI Agent将从目前的L3阶段逐步进化到**L4/L5更高自主性等级**，实现从工具执行到自主决策的能力升维

将从**工具属性向更具自主性和智能化的方向迈进**，开启人机协同的新纪元，彻底改变我们与数字世界乃至物理世界的互动方式

MCP概念

模型上下文协议（Model Context Protocol，简称MCP）是一种开放标准，旨在解决AI模型在缺乏实时信息和与外部工具交互能力方面的局限性

定义与来源

由Anthropic公司推出并开源的开放标准，为AI模型提供了与外部世界交互的统一接口

核心目标

让AI模型能够方便、安全地访问外部数据源和工具，克服传统LLM在实时信息获取和工具调用方面的局限

价值与影响

被誉为AI世界的"USB-C接口"，鼓励业界广泛采用和参与开发，构建统一和繁荣的AI应用生态

安全隔离　统一接口　高效交互　可扩展性

MCP协议为AI模型与外部工具和数据源的交互提供了标准化方式，促进了AI应用生态的发展

MCP核心架构

开放原子开源基金会 OPENATOM FOUNDATION | openKYLin

MCP协议采用Host-Client-Server模式，为AI模型与外部世界的交互提供安全、高效的接口

Host（宿主应用）

运行LLM或智能体的应用环境，如Claude桌面应用或IDE插件。负责发起与数据源的连接，是用户与AI模型进行交互的界面。

MCP Client（客户端）

由Host管理的中间代理，每个客户端与一个特定的MCP服务器通信，确保了安全隔离。充当LLM的"接线员"，协调LLM与服务器之间的通信。

MCP Server（服务器）

实现MCP标准的轻量级程序，为特定的数据源或工具提供标准化接口。包括数据资源、函数工具和提示模板等能力。是数据与工具的"翻译官"。

MCP架构优势

- 确保AI模型与外部系统交互的安全性
- 提供统一、标准化的接口
- 便于集成和扩展外部能力

A2A介绍

开放原子开源基金会 OPENATOM FOUNDATION | openKYLin

A2A协议核心机制

- **Agent Card**：标准化的服务发现机制，每个Agent公开元数据文件，通常放置在路径
- **消息格式**：基于JSON的统一消息格式，包含身份验证、请求意图、数据负载等字段
- **安全协议**：支持多种身份验证机制和权限控制，确保智能体间安全通信
- **状态管理**：提供会话持久化和上下文跟踪机制，支持长时间复杂交互

跨框架互操作性

LangChain　AutoGen　CrewAI

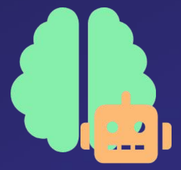
A2A协议作为中间层桥接不同框架，使智能体无需了解其他系统的内部实现即可高效协作

多智能体交互模式

- **点对点通信**：直接的双向交互，智能体间可直接发送查询和响应
- **广播通信**：一对多消息分发，用于公告和信息同步
- **分布式协作**：多智能体组成网络共同解决复杂任务，支持角色分配
- **共享内存**：支持基于键值对的跨智能体知识共享和状态同步

实现与集成方法

- 使用A2A客户端库简化协议实现（Python、JavaScript等多语言支持）
- 通过内置代理实现与不支持A2A的旧系统集成
- 提供Docker容器化部署模板，简化多智能体系统构建
- 支持与云服务平台API无缝集成，实现更丰富的功能扩展



智能体实践开发介绍

智能体开发、MCP协议与多智能体协作

智能体开发　MCP协议　多智能体交互

系统开发环境搭建

开放原子开源基金会 OPENATOM FOUNDATION | openKYLin

操作系统版本

- 下载地址

```
https://www.openkylin.top/downloads/
```

Node.js环境安装

- **下载安装包**：从Node.js官网下载适用于Linux的.tar.xz或.tar.gz包
- **解压安装包**：将包移动到目标目录并解压

```
   sudo tar -xvf node-vXX.YY.Z-linux-
x64.tar.xz -C /usr/local/
```

- **创建软链接**：创建全局访问的软链接

```
   sudo ln -s /usr/local/node-vXX.YY.Z-
linux-x64/bin/node /usr/local/bin/node
   sudo ln -s /usr/local/node-vXX.YY.Z-
linux-x64/bin/npm /usr/local/bin/npm
```

- **验证安装**：在终端运行以下命令检查版本

```
node -v
npm -v
```

Python环境配置

- **检查系统Python版本**：

```
   python3 -V
```

- **安装pyenv（可选）**：用于管理多个Python版本

```
   sudo apt updatesudo apt install -y build-essential
libssl-dev zlib1g-dev libbz2-dev libreadline-devcurl
https://pyenv.run | bash
```

- **安装特定Python版本**：

```
   pyenv install 3.9.18pyenv global 3.9.18
```

- **创建虚拟环境**：

```
   mkdir my_ai_agent_project
   cd my_ai_agent_project
   python3 -m venv venv
```

- **激活虚拟环境**：

```
   source venv/bin/activate
```

其他语言的开发环境搭建，请参考
https://github.com/orgs/modelcontextprotocol/repositories?type=all

大模型选型与API注册

开放原子开源基金会 OPENATOM FOUNDATION | openKylin

大模型能力对比

模型名称	推理能力	代码能力	工具调用能力	情感理解
GPT-4	领先	领先	领先	优秀
Claude	优秀	较强	优秀	优秀
GPT-3.5	良好	良好	良好	良好
Inflection-1	良好	较弱	良好	优秀
LLaMA 2	较弱	较弱	较弱	一般
Baichuan	较弱	较弱	较弱	一般

目前，国内对 MCP 工具调用支持相对比较好,并且稳定的大模型是阿里刚发布的 Qwen3-Plus。建议直接把 Qwen3-Plus 作为默认基座，在它的对话接口里开启 MCP 功能即可顺滑完成工具调用。

API Key申请与注册流程

1 选择云服务商

根据项目需求选择提供大模型API的云服务商，如OpenAI、Anthropic、阿里云、腾讯云等。

2 注册账号并登录

在选定的云服务商平台注册并登录账号。

3 导航至API管理页面

通常在控制台的"API管理"、"密钥管理"或"开发者设置"等部分。

4 创建新的API Key

点击"创建新密钥"或类似按钮，系统会生成一个唯一的API Key。

5 配置权限

根据需要为API Key配置访问权限，限制其可调用的模型或服务范围，以增强安全性。

6 妥善保管API Key

API Key是访问大模型服务的凭证，应像密码一样妥善保管，避免泄露。

选择合适的大模型作为智能体的"大脑"至关重要，不同模型在推理、代码生成和工具调用等方面表现各异。根据您的应用需求，选择最适合的模型。

智能体开发流程

开放原子开源基金会 OPENATOM FOUNDATION | openKYLin

1 定义需求与目标

- 明确智能体需要完成的具体任务和预期目标
- 将复杂任务拆解为更小的子任务

2 设计Agent架构与功能

- **角色分配**：为每个子任务设计智能体角色
- **工具选择**：为每个角色选择合适的工具
- **规划能力**：利用大模型的提示工程赋予任务分解能力
- **记忆机制**：设计短期记忆和长期记忆支持

3 编写代码与集成

- 使用编程语言编写智能体的执行逻辑
- 选择合适的开发框架，如LangChain、Microsoft AutoGen、Crew AI等
- 集成外部工具和数据源，使智能体能够感知环境、执行动作

4 测试与调试

- 使用测试案例验证智能体的功能
- 根据测试结果调整代码和参数
- 优化智能体表现

智能体集成MCP协议

开放原子开源基金会 OPENATOM FOUNDATION | openKYLin

1 理解MCP架构

MCP采用Host-Client-Server模式。智能体作为Host应用，通过MCP Client与MCP Server通信。MCP Server为数据源或工具提供标准化接口。

2 选择MCP客户端实现

根据开发语言和部署需求，选择合适的MCP客户端库。例如，Spring AI提供了Java的MCP客户端实现，支持stdio和SSE两种传输层。

3 配置MCP客户端

在智能体应用中配置MCP客户端，指定其连接的MCP Server信息（如命令、参数、URL等）。例如，可通过配置文件设置stdio或SSE模式的MCP Server连接。

4 设置MCP启动命令

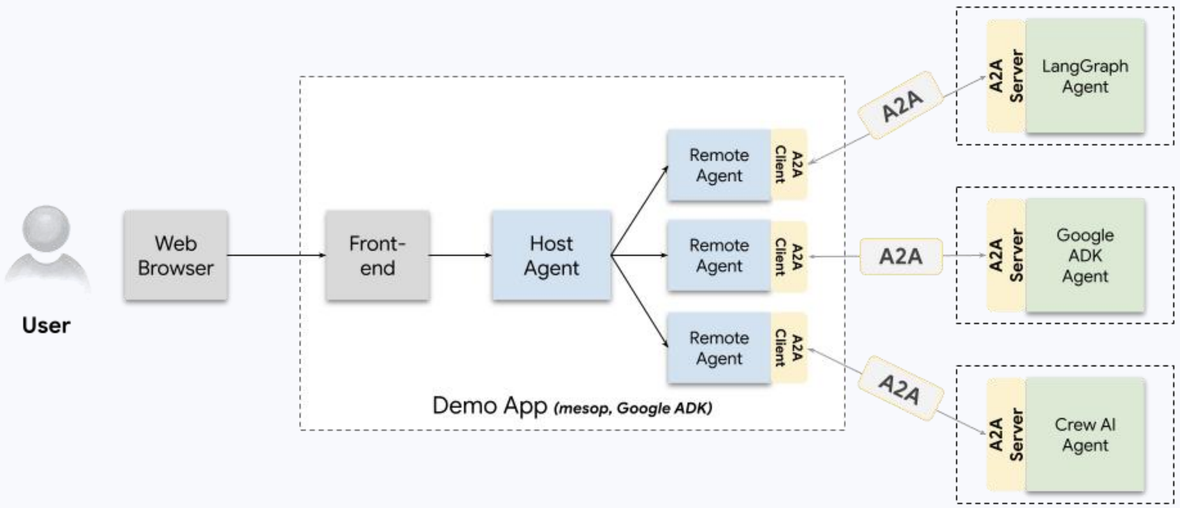
MCP配置启动命令，对于不同的服务使用不同的启动命令：例如使用npx 或者uv来启动

MCP 配置文件

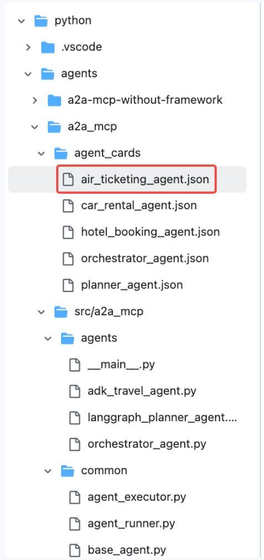
```
"mcpServers": {
            # 文件系统访问工具
            "filesystem": {
                "command": "npx",
                "args": ["-y", "@modelcontextprotocol/server-filesystem",

                "/Users/apple/demo_mcp/Agent_Server_mcp/"]
            },
            # blender使用
            "blender": {
                "command": "uv",
                "args": [
                    "tool",
                    "run",
                    "blender-mcp"
                ]
            },
            # 远程MCP服务器示例 - Streamable HTTP模式
            "remote-http-server": {
                "type": "streamable-http",
                "url": "https://your-remote-server.com/mcp",
                "sse_read_timeout": 300
            }
}
```

A2A协议与多智能体交互



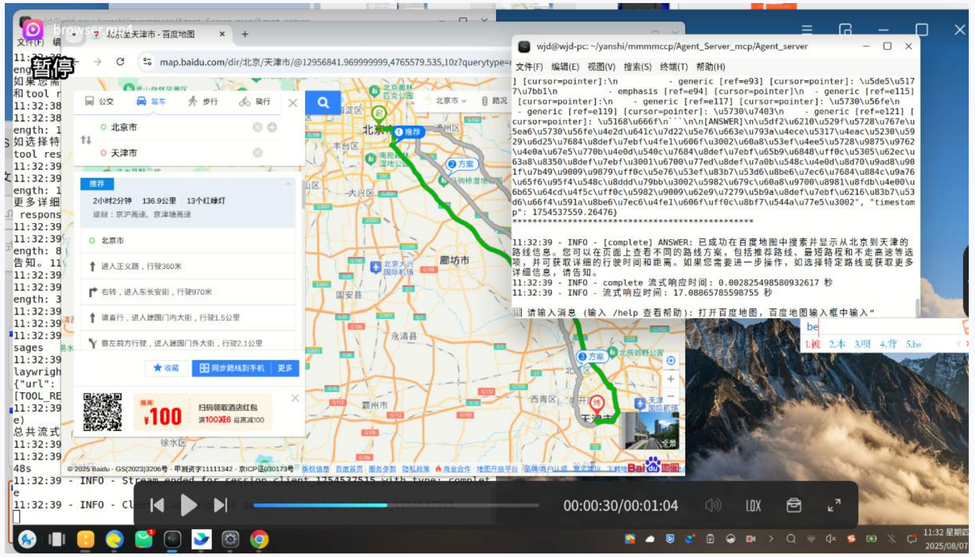
上图是一个A2A智能体的架构图

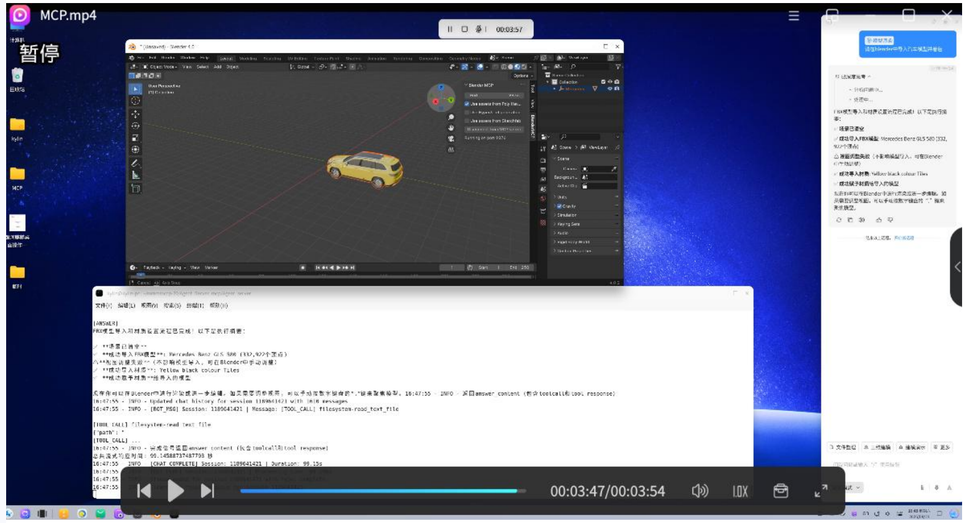


```
{
    "name": "Air Ticketing Agent",
    "description": "Helps book air tickets given a criteria",
    "url": "http://localhost:10103/",
    "version": "1.0.0",
    "capabilities": {
        "streaming": true,
        "pushNotifications": true,
        "stateTransitionHistory": false
    },
    "defaultInputModes": [
        "text",
        "text/plain"
    ],
    "defaultOutputModes": [
        "text",
        "text/plain"
    ],
    "skills": [
        {
            "id": "book_air_tickets",
            "name": "Book Air Tickets",
            "description": "Helps with booking air tickets given a criteria",
            "tags": [
                "Book air tickets"
            ],
            "examples": [
                "Book return tickets from SFO to LHR, starting June 24 2025 and returning 30th June 2025"
            ]
        }
    ]
}
```

上图是每个智能体下需要实现的智能体卡的配置文件

实践展示





参考demo

单智能体：https://github.com/wjd6910502/Agent_mcp_demo/tree/main

多智能体交互：https://github.com/a2aproject/a2a-samples/tree/main/samples/python/agents/a2a_mcp

业内应用案例

开放原子开源基金会 OPENATOM FOUNDATION | openKYLin

案例一：AutoGPT自动化市场调研

用户向AutoGPT下达"对防水鞋进行市场调查，并给出Top 5公司及其竞争对手优缺点报告"的任务。

任务分解与规划：AutoGPT将大目标分解为"获取当前位置"、"确定匹配餐厅"等子任务。

1. **信息收集**：利用谷歌搜索工具，查找防水鞋的综合评估和Top 5公司。
2. **数据分析与验证**：分析其他网站，结合谷歌搜索更新查询，判断评论真实性。
3. **子智能体协作**：衍生子智能体执行分析网站任务，寻找解决方案。
4. **报告生成**：生成详细的Top 5防水鞋公司报告，包含各公司优缺点。

成果：仅用时8分钟，费用约10美分

案例二：虚拟小镇社会模拟

斯坦福大学研究者于2023年4月发表论文，展示了一个由25名AI Agent组成的虚拟小镇"Smallville"。

特点：类似人类的特质、独立决策和长期记忆能力

核心架构：记忆流

每个Agent的核心架构是记忆流，存储所有经历记录，通过最近性、重要性和相关性三个因素进行检索。

社会互动与涌现行为

- 咖啡店长伊莎贝拉计划举办情人节派对，并自主传播派对邀请
- 伊莎贝拉的闺蜜玛利亚得知派对后，偷偷邀请暗恋对象克劳斯一同前往
- 年近六旬的汤姆因对市长选举感兴趣而拒绝了派对邀请

结论：展示了多智能体协作的涌现能力，系统整体行为超越了单个Agent的设计规划

2025

开放原子开源基金会 OPENATOM FOUNDATION | openKylin

欢迎报名

扫一扫 进入赛事页面报名

添加好友，并发送“大赛报名”入群交流

www.openKylin.top

contact@ openKylin.top