# 从架构设计侧剖析： MCP vs A2A 是朋友还是对手？

Anthropic 推出的 MCP（模型上下文协议）取得了成功，这显然激发了 AI 行业里的其他参与者，大家都想来定义一些开放协议，好用在 AI Agent 系统（Agentic Systems）的集成里。

上周周，谷歌公开发布了一个叫 A2A（Agent2Agent）的开放协议，目标是规范多 AI Agent 系统通信的实现方式。很多人（可能有点误解）说这两种协议是竞争关系，而不是互补关系。

**谷歌公开说 A2A 和 MCP 是互补的**。这话挺合理的。但会不会有隐藏的长期竞争目标呢？我们会不会很快看到协议之间的竞争开始呢？

很多人都问我，我觉得这两种协议未来会不会变得有竞争？本文从架构设计来剖析 MCP vs A2A 是朋友有还是对手？

本文重点剖析：

* + **A2A 架构设计**
  + **MCP 架构设计**
  + **A2A 如何与 MCP 互补，反之亦然？**
  + **长期来看，A2A 是否会在长期内取代 MCP？**

A2A 架构设计

第一、为什么会有 A2A？

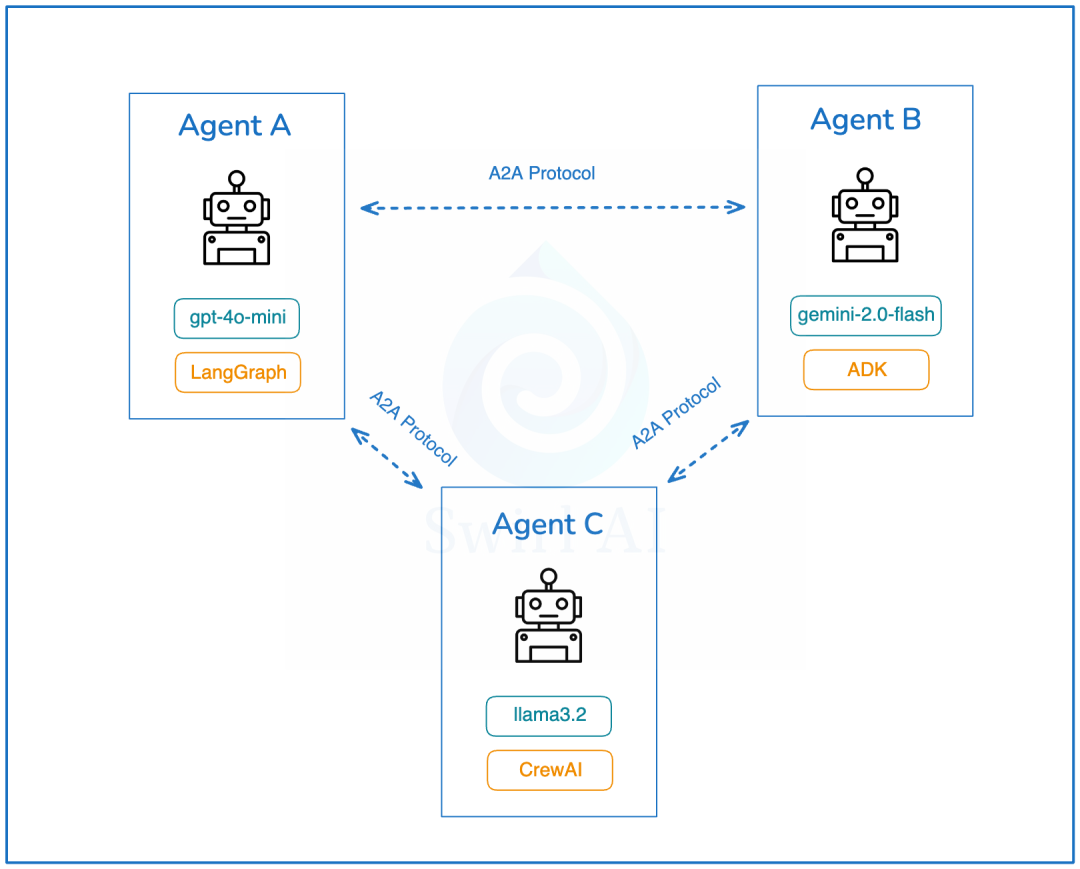
现在越来越清楚，未来的 AI Agent 系统（Agentic Systems）将是多 AI Agent 的。而且，这些 AI Agent 会在彼此之间远程协作，每个 AI Agent 都可能使用不同的 AI Agent 框架（比如：LangGraph、AutoGen、CrewAI、Agent Development Kit 等）来实现。

这里面有**3个固有的问题**：

1. 不同框架实现的 AI Agent 系统之间，不支持系统状态的转移和交换。
2. 远程 AI Agent 之间也无法转移系统状态。
3. 离线的 AI Agent 不共享工具、上下文和内存（包括系统状态）。

第二、A2A 解决方案

A2A 是一个开放协议，它为 AI Agent 之间提供了一种标准方式，无论底层开发框架或供应商如何，都可以进行协作。



根据谷歌的官方文档： A2A 协议促进了“客户端”和“远程” AI Agent 之间的通信。简单来说，“客户端” AI Agent 创建任务并与“远程” AI Agent 沟通，期望执行某些工作或返回数据。

第三、A2A 架构设计

**1、能力发现**：所有实现 A2A 的 AI Agent 都通过“**Agent Card**”公开其能力目录。这有助于其他 AI Agent 发现给定 AI Agent 实现的潜在有用功能。

**2、任务管理**：通信协议，时代短期和长期任务变得更容易。它帮助通信中的 AI Agent 保持同步，直到请求的任务完成并返回答案。这很重要，因为有些 AI Agent 可能需要很长时间来执行工作，而且目前没有统一标准如何等待这种情况发生。

**3、协作**：AI Agent 可以相互发送消息以传达上下文、回复、工件或用户指令。

**4、用户体验协商**：这是一个很有趣的功能。它允许协商数据返回的格式，以符合用户界面的期望（比如：图像、视频、文本等）。

通过 A2A 公开的 AI Agent 的发现是一个重要话题。谷歌建议使用统一的位置来存储组织的“**Agent Card**”。

https://<DOMAIN>/<agreed-path>/agent.json

我喜欢 A2A 强调无需重新发明轮子，并且建立在现有标准之上：

1. **该协议建立在现有、流行的标准之上**，包括：HTTP、SSE、JSON-RPC，这意味着它更容易与企业日常使用的现有 IT 堆栈集成。
2. **2、默认安全** - A2A 旨在支持企业级身份验证和授权，与 OpenAPI 的身份验证方案相当。

MCP 架构设计

MCP（模型上下文协议）是由 Anthropic 定义的一个开放协议，标准化应用程序如何为大语言模型（LLM）提供上下文。更具体地说，它试图标准化基于 LLM 的应用程序与其他环境集成的协议。

在 AI Agent 系统（Agentic Systems）中，上下文可以通过多种方式提供：

1. **外部数据**：这是长期记忆的一部分。
2. **工具**：系统与环境交互的能力。
3. **动态提示词**：可以作为系统提示词（System Prompt）的一部分注入。
4. **为什么要标准化？**目前，AI Agent 应用的开发流程很混乱：
5. **有许多 AI Agent 框架存在细微差异**。虽然看到生态系统蓬勃发展令人鼓舞，但这些细微差异很少能带来足够的价值，但可能会显著改变你的代码编写方式。
6. **与外部数据源的集成通常是临时实现的**，并且使用不同的协议，即使在组织内部也是如此。对于不同公司来说，这显然是如此。
7. **工具在代码库中以略微不同的方式定义**。如何将工具附加到增强型 LLM 上也是不同的。

目标是提高我们创新 AI Agent 应用的速度、安全性以及将相关数据带入上下文的便利性。

