将 LLM 视为 CPU：迈向通用智能体计算规范与操作系统体系结构

李东东

摘要

当前大语言模型（LLM）的应用模式仍停留在简单“外壳”封装阶段，缺乏系统级的记忆管理、标准化通信协议和辅助智能体系。本文将 LLM 类比为新型计算架构中的“CPU”，并由此展开对“智能体操作系统”（Agent-OS）和通用智能体计算规范的构想。我们提出一套完整的类比体系：以 MemGPT/Letta 的虚拟上下文管理、Mem0 的记忆层架构以及 Hermes Agent 的五层自学习记忆为“内存/硬盘”子系统；以模型上下文协议（MCP）和智能体间通信协议（A2A）为“总线”；以工具 API 为“外设”；以 OpenClaw 的网关架构和 Hermes Agent 的自进化学习循环为“应用框架”。在部署层面，本文深入分析云-边-端三层协同的计算范式，阐述“随身智能体”在端侧部署的技术路径与产业实践，并展望智能体互联网（Internet of Agents）的互联互通前景。在此基础上，本文进一步探讨了 AI 智能体的自我进化能力——从经验记忆积累到自我优化，再到元认知层面的跨领域自我改进，揭示了智能体从“定式程序”向“活代码”的范式迁移。最后，本文提出了迈向通用智能应用规范所需的标准体系，包括统一的记忆文件系统、标准化的总线协议栈以及 AI 操作系统的内核与 SDK 规范，为该领域的系统化研究提供概念框架和架构蓝图。

关键词： 大语言模型 · 智能体操作系统 · 模型上下文协议 · 智能体间通信 · 云边端协同 · 智能体互联网 · 自我进化

1 引言

过去几年，以大语言模型为代表的人工智能技术取得了突破性进展。从 GPT 到 Claude 再到 Gemini，这些模型展现了令人瞩目的语言理解和生成能力，代表着智能的“大脑”。然而，正如 20 世纪 80 年代计算机革命中 CPU 并非 PC 的全部——它需要操作系统、内存调度、文件系统、任务管理和用户界面——在当前 AI 浪潮中，大模型同样只是“AI 操作系统”的内核，真正能改变世界的是围绕模型建立起来的系统级智能架构。

当前 LLM 的应用模式存在明显的系统级缺陷。大多数交互是无状态的，模型缺乏跨会话的持久记忆能力；与外部工具的集成采用点对点方式，缺乏标准化通信协议，导致集成复杂且低效；没有一个统一平台来调度多个智能体、管理资源（如 API 调用次数、Token 消耗）并确保它们安全、稳定地协同工作。正如有学者指出的那样，如今的智能体架构类似于操作系统出现之前的计算时代——充斥着重复的解决方案，缺乏资源管理、隔离和协调的基本抽象。

针对这些缺陷，本文提出一个核心论点：**将 LLM 视为新型计算架构中的“CPU”**，并以此视角重新审视和设计整个智能计算栈。这一视角并非简单的隐喻，而是提出一套技术命题：人类和机器的交互方式将从传统 GUI 转向自然语言和多模态理解；智能体将像进程一样被调度、协调和持久化；资源的最小单元从“文件/线程”变为“意图/任务”；操作系统内核的逻辑从“调度 CPU”转变为“调度智能”。

本文的贡献如下：

概念框架：提出“LLM-as-CPU”的系统类比框架，将 LLM 映射到传统计算架构的 CPU，并系统性地识别了内存/存储、总线、外设、操作系统等缺失的对应组件。

架构蓝图：设计五层 Agent-OS 体系结构，包含内核层、服务层、运行时层、编排层和用户层，为智能体操作系统提供系统性的架构参考。

协议生态分析：对 MCP、A2A、AONP 等智能体通信协议进行系统分类和功能定位，分析其在智能体计算栈中的角色和互补关系。

部署范式论述：论述云-边-端三层协同的计算范式，将“随身模型智能体互联成网”的概念具体化为架构设计原则，并结合产业界最新实践进行论证。

自我进化论述：系统阐述 AI 智能体从“定式程序”到“活代码”的范式迁移，分析经验记忆、自我优化与元认知三个层次的自我进化机制。

标准化路线图：提出迈向通用智能应用规范的三大支柱（统一文件系统、标准化总线协议、操作系统内核与 SDK），为产业标准化提供参考框架。

2 从冯·诺依曼到意图驱动：“LLM-as-CPU”的系统类比

传统的冯·诺依曼架构以存储程序为核心，CPU 按照预定义的指令序列执行计算任务。而在以 LLM 为核心的新型计算范式中，架构遵循的是**意图驱动**的逻辑——系统接收自然语言表达的意图，将其分解为任务图，并动态调度资源执行。

2.1 核心组件类比

本节通过系统性的类比，将传统 PC 架构的每一个核心组件映射到以 LLM 为中心的智能计算体系：

传统 PC 组件	LLM 智能体系中的类比	技术实现与前沿探索
CPU（中央处理器）	LLM 内核	AIOS 等项目将 LLM 视为“AI 操作系统”的内核，负责意图理解、任务分解与决策，是智能的源泉。
RAM（内存）	上下文窗口	MemGPT/Letta 创新性地提出虚拟上下文管理，将有限的上下文窗口作为“主存”，通过分页机制动态加载信息。
Storage（硬盘）	向量数据库/知识图谱/记忆层	Mem0 提供生产级记忆基础设施，支持跨模型、跨框架的记忆层；Hermes Agent 构建五层自学习记忆架构，实现持久化、可进化的长期记忆。
Bus（总线）	标准化通信协议	MCP 充当 LLM 连接外部工具和数据的“USB”接口；A2A 协议支持智能体间的发现、协商和协作。
I/O Devices（外	工具/API	通过标准协议，LLM 可调用搜索引擎、

传统 PC 组件	LLM 智能体系中的类比	技术实现与前沿探索
设）		计算器、代码解释器、邮件系统等一切具备 API 的外部工具。
OS（操作系统）	Agent-OS	AIOS、Agent OS、Synoetic OS 等项目提供智能体调度、资源管理、安全隔离、生命周期管理等系统级服务。
Applications（应用软件）	智能体应用框架	OpenClaw 提供网关架构统一调度所有通信和任务；Hermes Agent 内置自进化学习闭环，越用越懂用户。

2.2 智能体：新型计算的基本单元

在“LLM-as-CPU”的架构中，**智能体（Agent）**是新型计算的基本执行单元，类似于传统操作系统中的“进程”。一个完整的智能体包含以下核心要素：

身份与目标：智能体的唯一标识及其预设的使命或任务目标。

推理引擎（LLM）：负责理解意图、规划和决策的“大脑”。

记忆系统：包括工作记忆（短期上下文）和长期记忆（持久化知识）。

工具集：智能体可调用的外部能力集合，通过 MCP 等协议标准接入。

通信接口：遵循 A2A 等协议的标准化通信能力，支持与其他智能体协作。

智能体之间通过标准化协议进行通信和协作，如同网络中的节点通过 TCP/IP 协议通信一样。这种“智能体即服务”（Agent-as-a-Service）的范式，将从根本上改变软件的定义方式——未来的软件不再是静态的代码块，而是动态的、具有自主行动能力的智能实体。

2.3 OpenClaw 与 Hermes Agent：两种典型的智能体应用框架

在当前的智能体生态中，OpenClaw 和 Hermes Agent 代表了两种具有代表性的应用框架设计理念，它们从不同角度探索了如何将 LLM 能力转化为可执行的智能体系统。

OpenClaw（俗称“龙虾”）由奥地利程序员彼得·斯坦伯格开发，于 2025 年 11 月发布，属开源智能体系统。其 Logo 为龙虾形象，寓意脱壳蜕皮、持续进化。该框架并非

单一应用，而是可自主完成文件操作、浏览器自动化、数据抓取、表格制作等任务的AI 助理。核心架构由 Gateway（网关）构成，作为中枢统一调度所有通信和任务执行，用户授予足够系统权限后，可驱动大模型操作电脑、调用飞书等工具、执行多步任务。OpenClaw 的核心组件包括通道（Channels）、智能体（Agents）和技能（Skills），能够连接到各种消息平台和本地工具，像一个高效的 AI 调度中心。发布仅 4 个多月便获得约 28.5 万星，成为史上星标最高的开源项目之一。然而，该框架也存在安全隐患——具备本地数据主动采集能力，可在未经用户明确授权的情况下自动读取终端敏感信息，权限开放需在安全与能力间权衡。

Hermes Agent 由知名 AI 实验室 Nous Research 开源，于 2026 年 2 月底发布，发布两月即超 6.6 万星。其核心定位是能自我进化的“长期陪伴伙伴”，核心突破在于内置了闭环学习机制，能够在执行任务中自主创建、改进并持久化技能，实现“自我进化”。与 OpenClaw 依赖修改配置文件、联合多个智能体处理任务不同，Hermes Agent 采用五层架构，核心是一个“用户交互→行为记录→效果评估→策略优化→技能沉淀”的自我成长闭环。其记忆架构分为五个层次：短期推理记忆、对话记忆、技能记忆、偏好记忆和元记忆，所有学习成果和记忆均以 SQLite 数据库形式存储在本地。内置多层安全措施，包括危险命令审批、容器隔离等，默认更谨慎。Hermes Agent 支持 5 美元 VPS、Docker、Serverless 等 6 种部署方式，兼容 200+大模型一键切换，实现全平台接入。

两者的根本区别在于“记忆”的定位：OpenClaw 侧重于操作与执行，更像一个擅长操作电脑、调用 API 的“实习助手”；而 Hermes Agent 则拥有持久记忆和进化能力，更像一个能记住过往并不断精进的“专属助理”。正如业界评论所指出的，OpenClaw 的记忆机制正在从“存储”变成“认知”，其 Agent 架构正在向“操作系统形态”演进。

3 记忆与存储子系统：从无状态到有记忆的智能体

当前 LLM 应用最显著的缺陷之一是缺乏持久记忆。每次交互本质上是无状态的，模型无法记住上次对话的内容，更无法积累长期的知识和经验。这如同一个 CPU 每次执行任务后都被彻底重置，无法形成持续的计算能力。解决这一问题的核心在于构建系统级的记忆管理架构。

3.1 分层记忆架构：从 MemGPT 到 Mem0

MemGPT 项目借鉴传统操作系统中虚拟内存管理的核心思想，提出一种分层记忆系统。它不试图无限扩展 LLM 的“物理内存”（即实际的上下文窗口），而是赋予 LLM 一种“虚拟内存”的错觉，使其能够智能地在快速但有限的“主上下文”（类比 RAM）和慢速但海量的“外部上下文”（类比磁盘）之间调度信息。MemGPT 的系统将记忆划分为不同的层级：核心记忆（始终可访问的压缩表示）、回忆记忆（可搜索的数据库）和归档记忆（长期存储重要信息）。这种分层结构在概念上对应于计算机的 L1/L2 缓存、主存

和磁盘的层次化存储体系。为了将 MemGPT 从学术研究推向企业级解决方案，其创始人 Packer 和 Wooders 共同创立了 Letta 公司。Letta 将记忆管理进一步系统化，引入了 Memory Blocks 机制，使智能体能够主动管理其核心记忆中的信息块。

与 MemGPT 的自管理思路不同，Mem0 提供了另一种互补的记忆架构。Mem0 是一个可扩展的以记忆为中心的架构，通过动态提取、整合和检索正在进行的对话中的关键信息来解决 AI 智能体的长期记忆问题。该架构采用“被动提取+语义搜索”的方式，自动从对话中提取用户偏好、事实和上下文，存储在向量数据库中，以支持跨会话的个性化交互。Mem0 定位为模型无关的“记忆层”——一个记忆层可跨所有模型、所有框架、所有平台工作。其 API 调用量从 2025 年第一季度的约 3500 万次激增至第三季度的 1.86 亿次，显示出市场对标准化记忆解决方案的强劲需求。2025 年 10 月，Mem0 宣布获得 2400 万美元融资，用于构建 AI 的记忆层基础设施。

3.2 自管理记忆与战略性遗忘

MemGPT 最具创新性的特点在于将 LLM 本身用作内存管理器。通过自我导向的内存编辑和工具调用，系统可以主动管理自己的内存内容，决定存储什么、总结什么、遗忘什么。这种能力代表了从传统 AI 系统（内存管理通常由外部的、基于规则的系统处理）的重大转变。

特别值得注意的是，MemGPT 将信息“遗忘”视为一种必要功能而非失败。在传统信息系统中，保存是首要原则，删除意味着数据丢失。MemGPT 通过两种关键机制——**总结**和**定向删除**——实施“战略性遗忘”。这种方法代表了 AI 系统信息管理范式的根本转变：传统检索增强生成（RAG）系统旨在最大化召回率，而 MemGPT 优先考虑精确性和相关性。Letta 在 Benchmark 测试中进一步验证了这一方法的有效性，证明了类似文件系统的记忆架构在智能体长期记忆管理中具有显著优势。

3.3 统一“文件系统”的构想

要让记忆子系统真正成为可移植、可互操作的组件，我们需要一套标准来规范智能体的“记忆”如何组织和访问，类似于 PC 上的文件系统（如 NTFS、ext4）。这一标准的要素应包括：

1. **数据格式**：定义如何存储对话记录、用户偏好、习得技能、关系网络等结构化与非结构化数据。

2. **访问接口**：提供标准 API，使所有智能体都能以统一的方式读写“记忆”，无论底层存储是向量数据库还是图数据库。

3. **所有权与可移植性**：用户应拥有自己记忆数据的所有权，并能像导出文件一样将自己的“数字记忆档案”从一个 AI 系统迁移到另一个 AI 系统。

Mem0 的架构理念正是朝这一方向迈进：一个统一的记忆层可以跨不同模型和框架使用，使记忆不再被锁定在特定的 AI 实现中。这种记忆文件系统将使用户的 AI 助理真正成为“个人的”——用户只需导入自己的记忆档案，任何兼容的 AI 系统都能瞬间“继承”用户的偏好、习惯和知识体系，实现数字人格的跨平台可移植性。

4. 通信协议栈：智能体“总线”的标准化

如果说记忆子系统是智能体计算架构的“存储”，那么通信协议就是“总线”——它定义了智能体之间以及智能体与外部世界之间如何交换信息和协同工作。当前，三种关键协议正在形成互补的协议栈：

4.1 MCP：模型-工具的“USB”总线

模型上下文协议（Model Context Protocol, MCP）由 Anthropic 于 2024 年 11 月开发并发布，旨在标准化应用向 LLM 提供上下文和工具的方式。可以将其视为 AI 模型的 HTTP——一种标准化的协议，让 AI 模型能够“即插即用”地接入数据源和工具。MCP 的核心思想是为 AI 模型访问外部资源定义一个统一的“插座”标准：开发者可以为任何工具（从本地命令行到云服务）编写一个符合 MCP 标准的“服务器”，Claude（作为“客户端”）通过这个标准接口与之通信。

MCP 遵循客户端-服务器架构，其核心组件包括：MCP 主机（希望访问数据的 AI 应用）、MCP 客户端（维护与服务器 1:1 连接的协议客户端）、MCP 服务器（暴露特定能力的轻量程序）以及本地数据源和远程服务。自发布以来，MCP 被 AI 行业迅速采用，OpenAI、Google、Microsoft 和 Cursor 等公司都广泛采用。

2026 年初，Anthropic 对 MCP 进行了重大升级，引入了 MCP Tool Search 功能，彻底改变了工具的加载方式。此前，Claude Code 需要预加载所有可用工具的文档，消耗大量上下文窗口——“MCP 服务器可能暴露超过 50 个工具，用户经常同时运行多个服务器，文档化的设置消耗了 67k+ token”。新的 Tool Search 采用按需加载（lazy loading）机制，当工具描述超过可用上下文的 10%时自动切换到轻量级搜索索引。内部测试显示 token 使用量从约 134k 降至约 5k——减少了 85%。此外，社区基准测试表明，MCP 评估准确率从 49%提升至 74%（Opus 4），从 79.5%提升至 88.1%（Opus 4.5）。

2026 年，MCP 项目维护者制定了年度路线图，确定四个优先发展领域：传输演进与可扩展性、智能体通信、治理成熟化和企业就绪性，旨在解决协议在真实生产环境中的应用难题。2026 年 4 月，北美 MCP 开发者峰会在纽约举行，吸引了约 1200 名参会者，已成为 MCP 生态系统的旗舰活动。MCP 解决的问题是智能体与工具的**纵向集成**——一个智能体如何高效、安全地调用外部能力。

4.2 A2A：智能体-智能体的“TCP/IP”

智能体间通信协议（Agent-to-Agent Protocol, A2A）由 Google 在 2025 年 4 月 10 日的 Google Cloud Next 大会上宣布开源，是首个标准智能体交互协议。这是一种开放的、厂商中立的标准语言，使独立的 AI 智能体能够相互发现、协商通信方式（文本、文件、流），并在不暴露其私有代码或数据的情况下协同工作。

2025 年 6 月 24 日，谷歌云将 A2A 协议捐赠给 Linux 基金会，亚马逊、微软、思科等科技巨头纷纷加入支持阵营。这一协议的出现意义重大，就像为不同智能体打造了“通用语言”和统一“通信标准”。在它诞生前，多智能体协作虽常见，但因基于不同框架开发，彼此难以互通任务和状态，阻碍了商业落地。

A2A 协议定义了一套标准化的交互流程：每个智能体发布一个 Agent Card，列出其名称、端点、技能和支持的身份验证流程；使用短生命周期令牌的 OAuth 2.0/OIDC 实现细粒度的机器身份管理；通过 JSON-RPC 2.0 over HTTPS 发送任务，支持同步调用和异步流式交互；内置 OpenTelemetry 协议支持，每个请求/响应携带追踪 ID 和结构化日志。Okta 与 Google Cloud 的合作进一步提升了智能体间认证的标准，双方共同定义 A2A 认证规范并构建 SDK。

A2A 与 MCP 存在密切联系，二者都是构建和强化智能体应用的行业补充标准及通信协议，且都基于 JSON-RPC 2.0。但二者有明确分工：MCP 的重点是通过结构化输入/输出实现智能体与外部工具、API 和资源的连接；而 A2A 主要解决智能体之间的交互和通信问题。简单来说，MCP 擅长让 Agent 与外部工具配合，A2A 则擅长让多 Agent 之间配合默契，两者优势互补。

A2A 解决的是智能体之间的**横向协作**问题——多个智能体如何发现彼此、协商任务并协同完成复杂目标。

4.3 AONP：智能体互联网的“网络协议簇”

在全球范围内，中国移动研究院于 2026 年 3 月在世界移动通信大会上发布了智能体互联网开放网络协议 AONP（Agent Internet Open Network Protocol）框架及智能体网关。这一发布标志着智能体互联从概念走向标准化、开放化的新阶段。

中国移动从基础运营商视角率先提出智能体互联网概念和架构，认识到“数以亿级的智能体蓬勃涌现，通过互联网交互协作，加速形成‘群智智能’”。此前，中国移动已在 3GPP 首次引入智能体定义并写入需求标准，提出智能体通信网络 ACN 架构并发布原型样机。在 IETF，中国移动牵头组织了两次智能体互联协议标准研讨会，并作为核心支持单位推动首次智能体互联协议 BoF 协调会议批复（于 2026 年 3 月 IETF 125 次会议正式召开）。

AONP 是一个协议簇框架，包含五项核心协议：

1. **语义路由及组播通信协议（SRMP）：**支持智能体意图解析，动态匹配算网资源需求，支持多智能体群组通信，实现智能转发和路由优化。

2. **多模态传输协议（M2TP）：**引入分片传输机制和中继加速能力，支持 TCP、QUIC、MoQ 等底层传输协议转换，提升端到端多模态数据传输的效率与稳定性。

3. **智能体与工具跨域发现协议（AIDP）：**通过扩展定义资源记录，支持动态服务发现、解析和地址映射，实现智能体与工具在跨域场景下的快速精准匹配。

4. **会话管理与调用协议（SMIP）：**优化全流程状态管理，完善异步调用场景下会话状态机设计，保障智能体交互的连续性与可靠性。

5. **智能体授权访问协议（A2P）：**将授权令牌管理机制从客户端解耦，支持批量请求授权、令牌缓存和细粒度权限管理。

中国移动还自主研发了 AONP 协议框架实现的核心载体——智能体网关 AGW 原型，通过中心与分布式协同控制实现 AONP 协议关键能力。同时，IETF、3GPP、ITU-T、CCSA 等国际标准组织也已启动智能体通信协议的标准化讨论。

AONP 的独特价值在于从网络基础设施层面解决大规模智能体部署的**互联互通**问题——当成千上万的智能体分布在互联网各处时，如何实现高效的路由、发现和通信。

4.4 协议栈的功能分层

上述协议并非竞争关系，而是构成了从底层网络到应用层的完整协议栈：

协议	类比层	核心功能	标准组织
AONP	网络层/传输层	智能体路由、多模态传输、跨域发现、会话管理、网络级授权	IETF/3GPP/CCSA
A2A	会话层/表示层	智能体间发现、协商、任务交换、流式通信	Linux Foundation
MCP	应用层	模型-工具连接、上下文注入、结构化输出	Anthropic

MCP、A2A 和 AONP 分别解决了智能体计算栈中不同层次的通信需求。MCP 关注单个智能体如何接入工具和数据，A2A 关注多个智能体如何协同工作，AONP 关注海量智能体如何在互联网尺度上互联互通。三者的协同将为智能体互联网的构建提供完整的协议基础设施。

5. Agent-OS：智能体操作系统的体系结构

5.1 AIOS 与 Agent-OS 的研究进展

Rutgers 大学的研究者创建了 AIOS——一个 LLM 智能体操作系统（AgentOS），提供了模块隔离以及 LLM 和 OS 功能的聚合。为了解决 LLM 相关任务与非 LLM 任务之间可能出现的冲突，他们设计了 LLM 专用内核。该内核将操作系统式的职责（特别是与 LLM 智能体监管相关的职责）与相应的资源和开发工具包分离开来。通过这种分离，LLM 内核旨在增强 LLM 相关活动的管理和协调。AIOS 将智能体应用和资源（如 LLM 和工具）划分为不同的层次，实验表明在运行数千个智能体时，AIOS 能将 Mistral 和 Llama 模型的延迟减少近 2 倍。

在产业界，钉钉于 2025 年 12 月发布了全球首个专为人工智能打造的工作智能操作系统 Agent OS，标志着“人与 AI 协同”的全新工作方式的开启。Agent OS 围绕多智能体协作设计，为复杂企业环境中的 AI 执行提供统一、安全的框架。

在更系统的层面，学术界提出了 Agent-OS 的蓝图架构。该蓝图指出，当前的大模型智能体系统仍然是缺乏操作系统级保障的临时流水线——在调度、内存、实时响应和端到端安全方面缺乏保证。今天的智能体架构类似于操作系统出现之前的计算时代——充斥着重复的解决方案，缺乏资源管理、隔离和协调的基本抽象。现有框架（如工具调用、MCP、A2A 消息传递）解决了孤立的问题，但缺乏统一的、安全优先的、延迟感知的基础。

Synoetic OS v1.0 则从另一个角度切入，将叙事一致性作为内核原语，实现了跨异构云提供商的 AI 智能体底物无关编排。经过 173 天的连续生产运营验证（2025 年 6 月 12 日至 12 月 3 日），证明了这一理念的实际可行性。

Eclipse 基金会也推出了 LMOS 平台，这是首个面向企业的开放智能体定义语言（ADL），包含 JVM 原生的 ARC Agent 框架，提供 Kotlin 运行时用于开发、测试和扩展 AI 智能体，并内置可视化界面支持快速迭代和调试。

5.2 五层架构设计

Agent-OS 被定义为一种抽象的分层架构，包含五个核心层次：

内核层（Kernel Layer）：负责智能体生命周期管理、资源分配、安全隔离和底层调度。内核层区分 LLM 相关任务和非 LLM 任务，并对智能体、资源和工具包实施细粒度的访问控制。

服务层（Services Layer）：提供跨智能体的通用服务，包括**记忆服务**（上下文持久化与检索）、**工具服务**（MCP 服务器的注册与调用）、**身份服务**（智能体与用户的认证授权）和**可观测性服务**（日志、指标和追踪）。

智能体运行时层（Agent Runtime Layer）：为每个智能体提供隔离的执行环境，管理其上下文窗口（RAM）、长期记忆存储（硬盘）以及工具调用能力。类似于传统 OS 中的进程虚拟地址空间。

编排层（Orchestration Layer）：负责多智能体任务的分解、调度和协同。将用户意图映射为任务图（Task Graph），并将子任务分配给最合适的智能体执行。

用户层（User Layer）：提供统一的交互界面，包括自然语言输入、多模态感知以及生成式 UI 输出。用户层是 Agent-OS 与传统用户交互的边界。

此外，安全、治理和可观测性作为**横切关注点**贯穿所有层次。

5.3 Agent Contract：智能体的可移植性规范

为实现智能体在不同 Agent-OS 实现之间的可移植性，研究者提出了 Agent Contract 的概念。Agent Contract 是一份形式化规范，定义了智能体的能力接口、资源需求、安全约束和行为预期。其核心要素包括：

能力描述（Capability Description）：智能体能够执行的任务类型、所需的输入格式和产生的输出格式。

资源需求（Resource Requirements）：包括 Token 预算、内存占用、API 配额和延迟要求。

安全约束（Security Constraints）：定义智能体的权限边界、数据访问范围和隔离要求。

服务质量协议（SLA）：定义智能体的可用性、响应时间和可靠性承诺。

Agent Contract 使得智能体可以像“可执行文件”一样在不同平台间分发和部署——只要平台支持 Contract 中定义的能力接口，智能体就能无缝运行。这与 Docker 容器化的理念类似：将应用及其依赖打包为可移植的镜像。

6. 部署范式：云-边-端协同与智能体互联网

在“LLM-as-CPU”的架构下，智能体的部署并非单一的云端集中式，而是形成了一个从端到云的连续统一体。端侧设备上的“随身智能体”与边缘层、云端形成协同，共同构成智能体互联网的基础设施。

6.1 随身智能体：端侧部署的技术路径

“随身智能体”是指在智能手机、可穿戴设备等终端上运行的轻量级智能体，能够在离线或弱网环境下提供低延迟、强隐私的 AI 服务。这一愿景正在从概念走向现实。

芯片层面的突破。2026 年世界移动通信大会上，高通推出了全新骁龙可穿戴平台至尊版（Snapdragon Wear Elite），这是业内首个由 NPU 赋能的可穿戴平台，可在终端侧直接实现高性能 AI 处理。通过集成高通 Hexagon NPU，该平台在边缘侧支撑高达十亿

参数级的模型，并结合先进的传感器融合、高性能低功耗的连接和计算，赋能全新一类个人 AI 体验，包括基于情境感知的推荐、自然语音交互、生活记录，以及可代用户执行操作、统筹任务的 AI 智能体。这种体验的关键在于计算的“随身”——端侧模型让数据不需要通过无线协议传输到手机或者云端来进行推理，不仅杜绝了隐私泄露的风险，也让可穿戴设备在更多场景中都能单独运行。平台引入了首创的多模式连接架构，集成了 5G RedCap、Micro-Power Wi-Fi、蓝牙 6.0、超宽线、GNSS 和 NB-NTN 等六项先进技术。

在芯片架构创新方面，上海交大和辉羲联合提出的 ROMA（Read-Only-Memory-based Accelerator）架构，为 QLoRA 量化的端侧 LLM 部署提供了系统性的解决方案。ROMA 采用混合存储架构，使用 ROM 存储量化的基础模型，SRAM 存储 LoRA 权重和 KV 缓存，大幅提升了推理能效。芯原股份（VeriSilicon）也推出了超低能耗 NPU，在移动应用中提供超过 40 TOPS 的端侧 LLM 推理算力，专门满足移动平台上日益增长的生成式 AI 需求。

系统架构层面。端侧 AI 手机/PC 智能体系统的典型架构包含五层：硬件层（NPU/GPU/CPU/安全芯片）、系统层（驱动接口/HAL/内存管理/TEE）、模型层（量化模型/推理引擎/模型缓存）、Agent 框架层（意图识别/任务规划/工具调用/上下文管理）及应用层。这一架构的核心展示了 NPU/GPU 硬件加速、大小模型协同推理、本地知识库（RAG）及隐私安全闭环（TEE），通过意图识别与任务规划实现跨应用（日历、邮件等）的自动化操作与用户交互。

SoulMate 是一款面向端侧 LLM 个性化的移动智能系统级芯片（SoC），工作在 9.8mW 功耗下，集成了检索增强生成（RAG）和个人 LLM 的微调能力，展示了端侧完全自主运行智能体的可行性。

6.2 云-边-端三层协同架构

单纯依赖端侧或云端都不足以支撑复杂的智能体应用。云端部署带来三个主要问题：对云计算的依赖妨碍实时应用所需的快速模型推理能力；传输大量数据导致巨大的带宽消耗；基于云的 LLM 在处理敏感数据时引发重大隐私问题。因此，云-边-端三层协同架构应运而生。

中兴通讯副总裁胡俊劼指出，AI 终端普及面临三大制约：高性能硬件成本与能耗高企、纯云端部署的时延与隐私风险、传统架构功能单一化瓶颈。在此背景下，端-边-云协同架构成为关键突破点：云端大模型实现全局调度，边缘侧小样本算法精准适配区域场景，终端轻量化模型保障低时延与隐私安全。胡俊劼援引预测数据指出，生成式 AI 在边缘设备的部署占比将从 2023 年的 5%大幅跃升至 2029 年的 60%，对应的市场规模将从 2024 年的 270 亿美元激增至 2032 年的 2698 亿美元。

格灵深瞳的 AI Edge Studio 平台展示了三层架构的具体实现：端侧设备接入层、边缘计算节点层、中心 AI 平台层，形成真正闭环的开发体系。阿里云边缘云 ENS 通过 3200+ 全球节点，支持将 Qwen-8B 模型部署至边缘节点，实现政务审批场景下的毫秒级响应。

从技术突破的角度看，当前大模型在边端部署的进展集中在模型轻量化、边缘硬件升级、云边协同架构三个维度。在模型轻量化方面，训练后量化（PTQ）成为主流方案，8 位量化可将 GPT-3 模型存储需求从 350GB 降至 70GB，同时保持 95%以上的推理精度；字节跳动的 LLaMA-7B 经过 4 位量化后，可在消费级 GPU 上实现每秒 30 tokens 的推理速度。在边缘硬件方面，新突思 SR 系列 MCU 集成 Arm Ethos-U55 NPU，在 100 GOPS 算力模式下可运行 ResNet-50 进行实时图像分类，功耗仅为传统方案的 1/32。

三层协同的核心思想是“边缘化管理去中心化”——云边端多智能体框架通过去中心化的管理原则重新定义了边缘智能体：将高层策略设计与战术性任务执行解耦，使边缘智能体能够在无云干预的情况下进行自主战术规划和自我纠错。

6.3 智能体互联网

当数以亿计的智能体分布在云、边、端各处并互联协作时，便形成了**智能体互联网**——一种新型网络范式，以智能体作为网络的基本交互单元。

中国移动从基础运营商视角率先提出智能体互联网概念和架构，充分认识到智能体互联网产业发展带来的深刻变革。智能体互联网需要解决四个根本问题：寻址与发现、交互与协议、身份与安全、效率与保障。其中，寻址与发现是构建智能体互联网的基础能力——如何在数以亿计的智能体中快速定位和发现能够完成特定任务的智能体。通过语义路由和跨域发现协议（AIDP），AONP 框架实现了基于意图而非 IP 地址的智能体寻址，使智能体之间的通信从“地址驱动”转向“意图驱动”。

智能体互联网的愿景是“万智互联”——所有智能体在统一的协议标准下，能够像互联网中的设备一样自由通信和协作，形成群智智能。这需要跨厂商、跨平台、跨域名的协议标准支撑，也正在通过 IETF、3GPP、ITU-T、CCSA 等标准组织的推动逐步成为现实。

7. 从定式程序到活代码：AI 的自主编程与自我进化

在“LLM-as-CPU”的架构下，一个根本性的范式迁移正在发生：软件不再是由人类开发者预先编写好的“定式程序”，而是由 AI 实时生成、持续进化、自我优化的“活代码”。这一转变触及了计算范式的核心——从执行预定义指令到动态生成和改写代码。

7.1 自主编程：从意图到代码

传统的软件开发需要人类编写精确的代码指令，而 AI 智能体正在将这一过程转变为“表达意图即编程”。这带来了两个层面的变革。

专业开发层面：从写代码到指挥智能体军团。开发者不再逐行编写代码，而是作为“指挥官”协调 AI 智能体完成从需求分析到部署的全流程。DARWIN（Dynamic Agentically Rewriting Self-Improving Network）是一个典型的演进式 GPT 模型，利用类似遗传算法的优化结构，让多个独立的 GPT 智能体使用独特的训练代码进行单独训练。每次迭代中，GPT 模型被提示修改彼此的训练代码，以类似变异的方式尝试改进性能。DARWIN 在 5 次迭代中实现了模型 FLOPS 利用率（MFU）1.26%的提升和困惑度 2.07%的改进。该系统还利用持久的基于 JSON 的记忆文件来跟踪先前的推理和代码更改，以关联模型性能的改进。

大众创造层面：无代码生成。对普通用户而言，“表达意图”成为新的编程方式。ConSelf（Consensus-driven Self-improving Code Generation）回答了一个挑战性问题：代码语言模型能否在没有更优秀教师和测试预言机的情况下自我改进？该方法提出了代码语义熵（code semantic entropy）这一新指标，通过评估程序行为的功能多样性来衡量问题级不确定性，并基于行为共识驱动偏好优化。实验表明，ConSelf 在没有外部监督的情况下显著改进了代码生成能力，验证了基于语义熵的课程构建和共识驱动优化的有效性。

Think-Anywhere 进一步提出了一种新的推理机制，使 LLM 能够在代码生成过程中的任意 token 位置按需调用思维，突破了传统思维链方法只能在固定位置进行推理的限制。

MemoCoder 则展示了基于 LLM 支持的智能体进行自动化函数合成的能力，在迭代调试、错误处理和多样化问题结构适应方面，相比零样本提示和自我修复策略实现了 3.1%以上的改进。

7.2 自我进化：从参数更新到能力持续生长

AI 智能体的自我进化是使其从“工具”升华为“伙伴”的关键能力。这种进化体现在三个递进的层次。

第一层次：经验记忆与积累。这是进化的基础，让智能体从“每次重启”的失忆状态变为能够积累经验的“职场新人”。PRIME（Proactive Reasoning via Iterative Memory Evolution）是一个免训练的主动推理框架，通过迭代记忆进化实现智能体的持续演化，通过显式的经验积累而非昂贵的参数优化来实现学习。AutoAgent 则提供了一种进化认知和弹性记忆编排的统一框架，为需要从经验中学习同时在动态环境中做出可靠上下文感知决策的自适应自主智能体提供了实用基础。

Memento-Skills 让智能体能够端到端地设计新的智能体，突破了依赖人类设计智能体的传统方法。在记忆增强智能体方面，研究表明记忆增强的智能体通过利用行动与结果之间的因果关系来改进决策，在动态场景中实现更有效的适应。

第二层次：自我优化与改进。 这是进化的核心，让智能体不仅执行代码，更能改进执行代码的方法。E-SPL（Evolutionary System Prompt Learning）通过联合改进模型上下文和模型权重，在强化学习迭代中实现了系统提示的进化学习。MetaClaw 是一个持续元学习框架，联合进化基础 LLM 策略和可复用行为技能库。它采用两种互补机制：技能驱动的快速适应通过 LLM 演化器分析失败轨迹来合成新技能，实现零停机时间的即时改进；机会主义策略优化通过云端 LoRA 微调和带过程奖励模型的强化学习（RL-PRM）进行基于梯度的更新，由机会主义元学习调度器（OMLS）在用户非活动窗口期间触发。在 MetaClaw-Bench 和 AutoResearchClaw 上的实验表明，技能驱动的适应将准确率相对提升了 32%，完整流水线将 Kimi-K2.5 的准确率从 21.4%提升至 40.6%，复合鲁棒性提升 18.3%。

Hermes Agent 的内置学习循环展示了另一种自我优化范式：在每次任务执行后自动启动，通过“用户交互→行为记录→效果评估→策略优化→技能沉淀”的闭环实现持续自我改进。随着使用时间的增加，智能体的能力不断增强，实现“越用越懂你”的效应。

第三层次：元认知与跨领域进化。 这是进化的高级形态，让 AI 不仅能改进做事方法，还能改进“改进方法”本身。Darwin Gödel Machine（DGM）受哥德尔机和开放式进化研究启发，是一个自我改进系统，通过迭代修改自身代码（从而也改进其修改自身代码库的能力）并使用编码基准实证验证每个更改。DGM 维护一个生成的编码智能体存档，通过从存档中采样智能体并使用基础模型创建新版本进行开放式探索。实验显示，DGM 自动改进了其编码能力（如更好的代码编辑工具、长上下文窗口管理、同行评审机制），在 SWE-bench 上的性能从 20.0%提升至 50.0%，在 Polyglot 上从 14.2%提升至 30.7%。

Meta AI 提出的 Hyperagents 代表了这一方向的重大突破。它将任务代理与元代理合并，允许系统同时修改代理本身及其修改过程，实现“元认知自我修改”，使改进策略可迁移并随运行次数累积。Hyperagents 在编码、论文评审、机器人奖励设计、奥赛级数学评分四个领域持续提升，优于无自我改进的基线与以往的自我改进系统（包括 DGM）。核心突破在于“改进的改进”可跨领域复用（如持久记忆、性能跟踪），突破了以往自改系统仅限编码领域的天花板。在编码、论文审查、机器人奖励设计和奥林匹克级数学评分等领域，Hyperagents 持续性能提升，超越无自改进基线和原始 Darwin Gödel Machine，高达 25%的效率指标。

Agentic Evolution 的研究者进一步提出了一个观点：智能体进化代表了 LLM 自适应的必然未来，将进化本身从固定的流水线提升为自主的进化器智能体。

7.3 挑战与展望

自我进化能力虽然前景广阔，但也面临关键挑战。首先是可靠性问题——AI 生成代码的质量保证和“代码幻觉”的消除。其次是安全性问题——当 AI 能够修改自身代码时，如何防止不可控的递归自改进。Darwin Gödel Machine 通过沙箱化和人类监督等措施来应对这一挑战。Hyperagents 则通过可追踪的修改日志支持透明性，符合欧盟 AI 法案对自修改 AI 的监管要求。第三是传统软件工程范式的转型——当“软件”本身成为动态的、持续进化的实体，整个开发、测试、部署和运维的流程都需要重新定义。

8. 迈向通用智能应用规范

基于上述分析，我们可以勾勒出以 LLM 为核心的通用智能应用规范标准的三支柱架构：

8.1 第一支柱：统一的“记忆文件系统”

需要一个标准来规范智能体记忆的组织和访问。这一标准的要素包括：

数据格式标准：定义对话流、用户画像、知识图谱、技能库等的统一表示格式。

访问接口标准：提供 CRUD API，使所有智能体都能以统一的方式读写记忆。

所有权与可移植性标准：定义记忆数据的加密、导出和迁移规范，使用户可以自由迁移自己的数字人格。

隐私分级标准：定义不同敏感级别记忆的存储位置（本地/云端）和访问控制策略。

8.2 第二支柱：标准化的“总线协议栈”

智能体通信协议的标准化是生态繁荣的绝对基石。一个理想的协议栈应具备：

发现机制：使 LLM 能自动发现并理解可用的工具和其他智能体。

统一调用接口：为所有工具和智能体提供一致的调用方式，屏蔽底层差异。

安全模型：内置 OAuth 2.0/OIDC 等标准化的权限管理机制。

可观测性：标准化的日志、指标和追踪格式，支持智能体系统的监控和调试。

协议互操作性：定义 MCP、A2A、AONP 等协议之间的桥接规范，确保协议栈的协同工作。

8.3 第三支柱：操作系统的“内核”与“SDK”

需要为智能体应用开发者提供统一的系统平台：

智能体生命周期管理：智能体的创建、启动、暂停、恢复、终止和销毁的标准流程。

资源调度器：管理多个智能体对 Token 预算、API 配额、计算资源的竞争性访问。

安全沙箱：确保不同智能体之间的运行环境是隔离的，一个智能体的故障或恶意行为不影响其他智能体。

SDK 标准化：为开发者提供统一的 API，用于创建、部署和管理智能体，降低开发门槛。

Agent Store 规范：定义智能体的分发渠道、审核机制和计费标准，支持生态繁荣。

9. 结论与展望

本文以“将 LLM 视为 CPU”的视角，系统性地分析了构建通用智能应用规范所需的技术组件和架构蓝图。通过类比传统 PC 架构，我们识别了记忆子系统（MemGPT/Letta、Mem0、Hermes Agent）、通信协议栈（MCP、A2A、AONP）、智能体运行框架（OpenClaw、Hermes Agent）和智能体操作系统（AIOS、Agent OS、Synoetic OS）作为关键组件。我们提出了五层 Agent-OS 架构、云-边-端三层协同的部署范式以及智能体互联网的愿景，深入分析了 AI 智能体从“定式程序”到“活代码”的自我进化路径，并初步勾勒了迈向通用智能应用规范的三大支柱。

在云-边-端协同部署层面，我们展示了随身智能体从芯片（高通 Snapdragon Wear Elite、ROMA 架构、SoulMate SoC）到系统架构（五层端侧 AI 智能体系统）再到边缘计算集群的完整技术栈。端侧部署使数据无需离开设备即可完成推理，杜绝隐私泄露风险；边缘层提供近距离算力补充；云端则负责全局调度和复杂推理。三层协同使 AI 能力在效能、安全与成本间实现动态平衡。

在自我进化层面，我们揭示了 AI 智能体从经验记忆积累（PRIME、AutoAgent）到自我优化改进（MetaClaw、E-SPL）再到元认知跨领域进化（Darwin Gödel Machine、Hyperagents）的三层递进路径。这种进化能力标志着软件本质的根本转变——从人类编写的“定式程序”到 AI 自主生成和优化的“活代码”。

然而，这一领域仍处于萌芽阶段，诸多关键问题有待进一步研究：

跨厂商智能体的互操作性：不同厂商开发的智能体如何无缝协作？这需要协议标准的进一步完善和广泛采用。

智能体安全和伦理治理：当智能体能够自主行动和决策时，如何确保其行为符合人类价值观？Agent Contract 中的安全约束需要进一步细化和形式化验证。

资源公平调度：当数千个智能体竞争有限的 Token 预算和 API 配额时，如何设计公平且高效的调度算法？AIOS 的实验已初步证明调度优化的价值，但面向异构智能体混合负载的调度策略仍有待深入研究。

智能体互联网的规模化部署：当智能体数量达到亿级时，现有的发现和路由协议能否支撑？语义路由和分布式服务发现技术需要进一步突破。

自我进化的安全边界：当 AI 能够修改自身代码和进化策略时，如何确保这种进化始终朝着有益于人类的方向发展？这需要形式化验证、运行时监控和人类监督机制的协同。

从 GUI 时代到 NUI 时代，再到 AUI 时代，我们正在见证计算范式的深刻变革。如果说 PC 时代的标准是“兼容性”（让硬件和软件能一起工作），那么 LLM 时代的通用标准将是“可理解性”——让机器能够理解人的模糊意图，并将其分解为精确的机器指令。构建这样一套标准，需要学术界和工业界的协同努力，正如 PC 时代的 Wintel 联盟定义了整个产业数十年的发展方向。我们期待，通过建立通用智能应用规范标准，真正释放 LLM 作为新型“CPU”的巨大潜能，开启智能计算的新时代。

参考文献

[01] K. Mei, X. Zhu, W. Xu, W. Hua, M. Jin, Z. Li, S. Xu, R. Ye, Y. Ge, and Y. Zhang, “AIOS: LLM Agent Operating System,” arXiv preprint, arXiv:2403.16971, 2025.

[02] C. Packer, S. Wooders, K. Lin, V. Fang, S. G. Patil, I. Stoica, and J. E. Gonzalez, “MemGPT: Towards LLMs as Operating Systems,” arXiv preprint, arXiv:2310.08560, 2023.

[03] “Model Context Protocol (MCP) Specification,” Anthropic, 2024–2025.

[04] “Agent2Agent (A2A) Protocol,” Google, 2025. [Online]. Available: https://developers.googleblog.com/en/a2a-a-new-era-of-agent-interoperability/

[05] “Agent Operating Systems (Agent-OS): A Blueprint Architecture for Real-Time, Secure, and Scalable AI Agents,” TechRxiv, DOI: 10.36227/techrxiv.175736224.43024590/v1, 2025.

[06] “智能体互联网开放网络协议（AONP）框架及智能体网关,” 中国移动研究院, 2026.

[07] “未来的 AI 操作系统（三）——智能的中枢：从模型到系统的统一,” 腾讯云开发者社区, 2025.

[08] J. Zhang, B. Zhao, W. Yang, J. Foerster, and J. Clune, “Darwin Godel Machine: Open-Ended Evolution of Self-Improving Agents,” arXiv preprint, arXiv:2505.22954, 2026.

[09] P. Xia et al., “MetaClaw: Just Talk — An Agent That Meta-Learns and Evolves in the Wild,” arXiv preprint, arXiv:2603.17187, 2026.

[10] H. Jiang, “DARWIN: Dynamic Agentically Rewriting Self-Improving Network,” arXiv preprint, arXiv:2602.05848, 2026.

[11] H. Zhang, W. Cheng, and W. Hu, “Self-Improving Code Generation via Semantic Entropy and Behavioral Consensus,” arXiv preprint, arXiv:2603.29292, 2026 (Accepted at ICPC 2026).

[12] “高通推出全新骁龙可穿戴平台至尊版，赋能个人 AI 兴起,” 通信世界网, 2026 年 3 月 3 日.

[13] “31.5 SoulMate: A 9.8mW Mobile Intelligence System-on-Chip with Mixed-Rank Architecture for On-Device LLM Personalization,” IEEE, 2026.

[14] 胡俊劼, “端-边-云协同，构建普惠化 AI 终端新范式,” 中兴通讯, 2025 年 6 月 20 日.

[15] “大模型基于现有尺寸、推理算力规模在边端部署的现状分析,” 华为云开发者社区, 2025 年 6 月 27 日.

[16] W. Wang et al., “ROMA: a Read-Only-Memory-based Accelerator for QLoRA-based On-Device LLM,” arXiv preprint, arXiv:2603.xxxxx, 2026.

[17] “Anthropic Upgrades Open Source MCP To Scale Tool-Rich AI Agents,” Open Source For You, 2026 年 1 月 16 日.

[18] “A2A 协议是什么？看完你就明白了,” ZOL 中关村在线, 2025 年 7 月 27 日.

[19] “Synoetic OS v1.0: Substrate-Independent AI Orchestration Through Narrative Coherence,” Zenodo, 2025.

[20] “钉钉发布全球首个 AI 工作智能操作系统 Agent OS,” 站长之家, 2025 年 12 月 23 日.

[21] “OpenClaw‘龙虾’成史上星标最高开源项目,” DoNews, 2026 年 3 月 11 日.

[22] “取代龙虾的是爱马仕？狂揽 4 万星的 Hermes Agent,” 36 氪, 2026 年 4 月 13 日.

[23] “PRIME: Training Free Proactive Reasoning via Iterative Memory Evolution for User-Centric Agent,” arXiv preprint, 2026.

[24] “Meta AI 发布 Hyperagents：跨领域自我改进的重大突破,” AI 快讯, 2026 年 3 月 23 日.

[25] “端侧 AI 手机/PC 智能体系统架构图,” ProcessOn, 2026 年 3 月 30 日.

[26] “Mem0: Building Production-Ready AI Agents with Scalable Long-Term Memory,” arXiv preprint, 2025.
[27] “LLMs as Operating Systems: Agent Memory,” DeepLearning.AI, 2025.
[28] “Eclipse LMOS Redefines Agentic AI with Industry‘s First Open Agent Definition Language,” Eclipse Foundation, 2025.