WHITE PAPER

多智能体协同架构白皮书

六层架构、四个关键机制与一条从任务下达到可复核成稿的完整时序

版本	出版	页数	适用读者
v1.0	2026-10	约 6 页	技术负责人 / 方案负责人 / 信息化与数据管理部门

编制单位

写逸网络科技（上海）有限公司

电话咨询

公众号

摘要：多智能体不是把任务拆给几个机器人。这份白皮书给出我们实际交付使用的六层架构与四个决定成败的关键机制。

摘要

多智能体不是“把任务拆给几个机器人”，而是一套让复杂任务能稳定跑完、出错能查、成本可控的工程结构。本白皮书给出我们实际交付使用的**六层架构**（交互 / 规划 / 智能体 / 能力 / 记忆 / 治理 + 底座）、**四个决定成败的关键机制**，以及一条从任务下达到可复核成稿的完整时序，最后说明什么时候**不该**上多智能体。

电话咨询

公众号

一、单体架构什么时候会撞墙

“再加一个提示词模板”是单体智能体最常见的续命方式。它在轻量场景里很有效，但[illegible]面三类信号时会开始结构性失效——不是效果差一点，而是无论怎么调提示词都做不到。

三条结构性信号

信号	表现	为什么单体做不到
一件事要跨角色	同一个任务既要查资料、又要比对、又要成文	三种角色的行为约束互相冲突（有的要求“只取不判”，[illegible]“只判不取”），塞进一段提示词会互相干扰，模型会自[illegible]
要跨多个系统	数据散落在多个业务库与文件通道	取数本身是若干可并行的子任务，单体只能串行做完，一次超时整条链路失败
步骤多且需中途判断	下一步做什么取决于上一步结果	流程无法预先写死，单体只能在一个上下文里反复自我对话，越走越偏且无法定位偏差起点

满足其中任意一条，就该考虑多智能体；三条都不满足，多智能体只会带来更高的成本与更长的排查路径。**判断的时机在方案阶段，不在上线之后。**

二、六层架构

下面这六层加一个底座，是我们每个多智能体项目的设计底稿。重点不在于“用了大模型”，而在于每一层都有明确的工程机制，且每一层都可被审计。

层	职责	关键设计点
交互层	多模态统一入口：语音、文本、图片、视频、表格	输入先落库再处理，原始材料与解析结果分开存；入口即记录来源与时间
规划层	任务拆解 → 角色分配 → 步骤编排 → 失败重试 → 结果仲裁	规划结果必须可读、可改、可保存；不把“想清楚”这一步留给模型的临场发挥
智能体层	各角色智能体：检索、分析、写作、复核、主控	角色按认知动作切分，依赖单向；每个角色有固 schema 与步数上限
能力层	工具与模型：只读查询、文档拉取、OCR、语音、向量检索、写入接口	工具白名单按角色下发；参数校验、超时、幂等 具层实现，不写在提示词里
记忆层	会话记忆、长期事实、明细存储与按需读回	进上下文的只有摘要与索引，明细落库；事实带 效期，可追溯、可失效
治理层	权限、留痕、审计、回滚、评测、巡检	四类事件全留痕（输入、模型调用、工具调用、 预）；回滚按单任务 / 版本 / 数据写回三种粒度
底座	算力、模型服务、向量库、数据库、私有化与信创环境适配	模型可替换；国产 CPU / 操作系统 / 数据库的 验室先跑通再进现场

电话咨询
公众号

分层最直接的好处是**定位问题变快**：结论不对，先看是能力层的检索没命中，还是智能体层的角色越界，还是记忆层把过期版本喂了进去。没有分层，这些判断只能靠猜。

三、四个关键机制

架构决定上限，机制决定能不能稳定跑到上限。下面四个机制是我们踩过坑之后固化成配置的。

机制一：失败重试与降级

- **只对幂等操作重试**：读操作指数退避重试，写操作靠幂等键保证安全，重试次数从严。
- **并发限制在工具层**：同一个外部系统最多几个并发由工具层管，不靠模型自觉。
- **部分失败不拖垮整体**：一条材料取不到，标记该条并继续，最后汇总成“未覆盖清单”交给人工。
- **降级要显式**：换小模型、走模板、返回“已入队稍后处理”都可以，但必须在界面上标注当前为降级结果。悄悄降级比不降级更糟。

机制二：结论仲裁

多个角色给出不一致结论是常态，关键是不能让模型自己“挑一个看起来对的”。我们的做法是：仲裁规则在方案阶段与客户共同确定并写成配置，运行期只按规则判定。

- **有出处的优先**：无出处的结论在仲裁阶段直接淘汰，不进入成稿。
- **时效新的优先**：同主题多版本时按生效期取值，失效版本只用于对比。
- **分歧保留并上报**：规则无法判定的分歧不做“平均”，而是标记为待确认，连同双方起交人工。

机制三：上下文成本控制

- **预算硬约束**：单次调用与单任务各有一个上限，超预算先压缩历史、再限制命中片段后才截断，每一步都留痕。
- **明细落库**：上下文里只放摘要与引用 ID，需要明细时用工具读回。这一条决定了成本不会随材料量线性增长。
- **步数上限**：单角色与主控各有步数上限，超限不是“再试一次”，而是停下转人工并交出中间产物。

机制四：幂等与副作用隔离

- **写操作统一幂等键**：按“任务 ID + 工具名 + 参数哈希”生成，重复调用直接返回上次结果。
- **副作用集中收敛**：所有写操作走少数几个受控接口，便于配额、审计与补偿。
- **先干后干分开**：分析阶段一律只读，人工确认之后才进入写入阶段，两个阶段用不同的工具集。

电话咨询
公众号

四、一条完整时序：从任务下达到可复核成稿

下面这条链路是我们最常见的交付形态，六个步骤加一条贯穿全程的审计线。

1. **任务下达（交互层）**：接收目标与材料，登记来源、时间与发起人，原始材料落库。
2. **拆解与编排（规划层）**：把目标拆成若干子任务，指定由哪些角色承担、依赖顺序如何、失败怎么退。规划结果保存，可人工修改。
3. **取数（能力层 + 智能体层）**：只读工具按白名单取数，命中材料进入候选集并带出处 ID；超时或失败按机制一处理。
4. **分析与比对（智能体层）**：对候选集做要素归一与关系判断，输出带依据与置信度的结构化中间产物。此阶段**不写回任何业务系统**。
5. **复核（人工 + 仲裁机制）**：按机制二做结论仲裁，人工在关键节点确认。确认记录痕，注明改动与原因。
6. **成文与交付（智能体层）**：按单位体例成文，每句结论带出处标记；输出可复核的成 未覆盖/待确认”清单。

贯穿全程的审计线：每一次用户输入、模型调用、工具调用与人工干预都挂在同一个任 下，记录耗时、token 消耗、提示词版本与决策理由。验收时可以现场演示“从一条结 到原始材料”的完整链路——这是我们把“可审计”写进交付标准的具体含义。

电话咨询
公众号

五、什么时候不该上多智能体

下面三种情况，我们会直接建议用单智能体甚至纯规则方案：

- **单一数据源 + 单步判断**：一次检索加一次生成就能完成的场景，多智能体只是把一次调用拆成三次。
- **不需要逐句溯源**：内部草稿、批量摘要这类不进入正式流程的产出，不必为溯源付出结构成本。
- **业务量极小且低频**：一个月跑几次的任务，人工加模板往往比平台更便宜，也更可控。

把这些判断放在方案阶段讲清楚，比事后解释“为什么多花了预算却更难维护”要容易得多。

六、落地路径与投入判断

多智能体的投入不是一次到位，我们通常按三个阶段推进，每一阶段都有独立可验收的产出：

阶段	范围	验收产出
第一期 · 单闭环	一个场景、两到三个角色、只读工具为主	评测集与基线、灰度报告与错例清单、可复核成稿
第二期 · 扩角色	补齐复核与仲裁、增加数据通道、接入写入前的确认环节	角色与工具编排图、权限矩阵、回归结果对比
第三期 · 平台化	治理层完整落地：留痕审计、巡检、评测流水线、按需扩容	运维手册、巡检脚本、回归基线、培训与交接文档

每一期结束后都回答同一个问题：**在当前业务量下，这套结构的收益是否覆盖它带来的**答案是否定的，就停在上一期，把已有的闭环做扎实——这比继续堆角色更值得。

关于架构落地中更细的做法（角色怎么切、记忆怎么分层、工具权限怎么写、审计留什段），我们在官网技术博客栏目有对应的工程笔记，可以与本白皮书对照阅读。

电话咨询
公众号

FAQ

关于这份白皮书