# 智能体总体架构

## 一、 核心概念：Agent 的双层记忆架构 (Dual-Memory System)

在流程图开始前，我们先明确 Python 智能基座层需要实现的两层记忆：

1. **短期记忆 (Short-term / Working Memory)：**
   - **载体：** Redis (`ChatSession` + `MemoryBuffer`)。
   - **作用：** 维持当前这“一趟”对话的连贯性。通常保留最近的 N 轮对话历史（如最近 10 轮或 4000 个 Token 内的历史）。超出窗口的老旧对话会被丢弃或压缩。
2. **长期记忆 (Long-term / Episodic Memory)：**
   - **载体：** ES Serverless (Vector 存储) + MySQL (`UserProfile`)。
   - **作用：** 跨会话的用户画像与事实留存。
   - **实现机制（也是一种工具）：** Python 后台实现一个 `update_memory` 工具。当大模型在聊天中敏锐地捕捉到用户的关键事实（例如：“我平时主要用 Java 和 React”、“我有个下个月要交的 PPT”），模型会主动调用这个工具，将事实提取、向量化并持久化到 ES 中。下次用户再来时，系统会先通过向量检索把这些长期事实捞出来，作为系统 Prompt 注入。

## 四、 架构师详细逻辑说明 (针对 Agent 与 Memory)

### 1. 记忆系统的运作闭环 (Memory Lifecyle)

- **读记忆（前置动作）：** 用户一句“帮我把那张图做成汇报”，模型怎么知道是哪张图？系统会在调用 LLM 前，去 Redis 读取最近 10 轮对话（知道是上一轮发的架构图），同时去 ES 长期记忆库拉取用户偏好（知道用户的汇报对象通常是技术高管）。
- **写记忆（工具动作）：** 这是一个极其巧妙的设计。大模型的工具清单里有一个名为 `update_long_term_memory(key_fact, confidence_score)` 的函数。在多次对话中，如果模型发现用户反复提及某个核心项目，模型会自主决定调用这个工具，将该事实永久写入 ES。

### 2. “While 循环”的破局 (Agentic Loop)

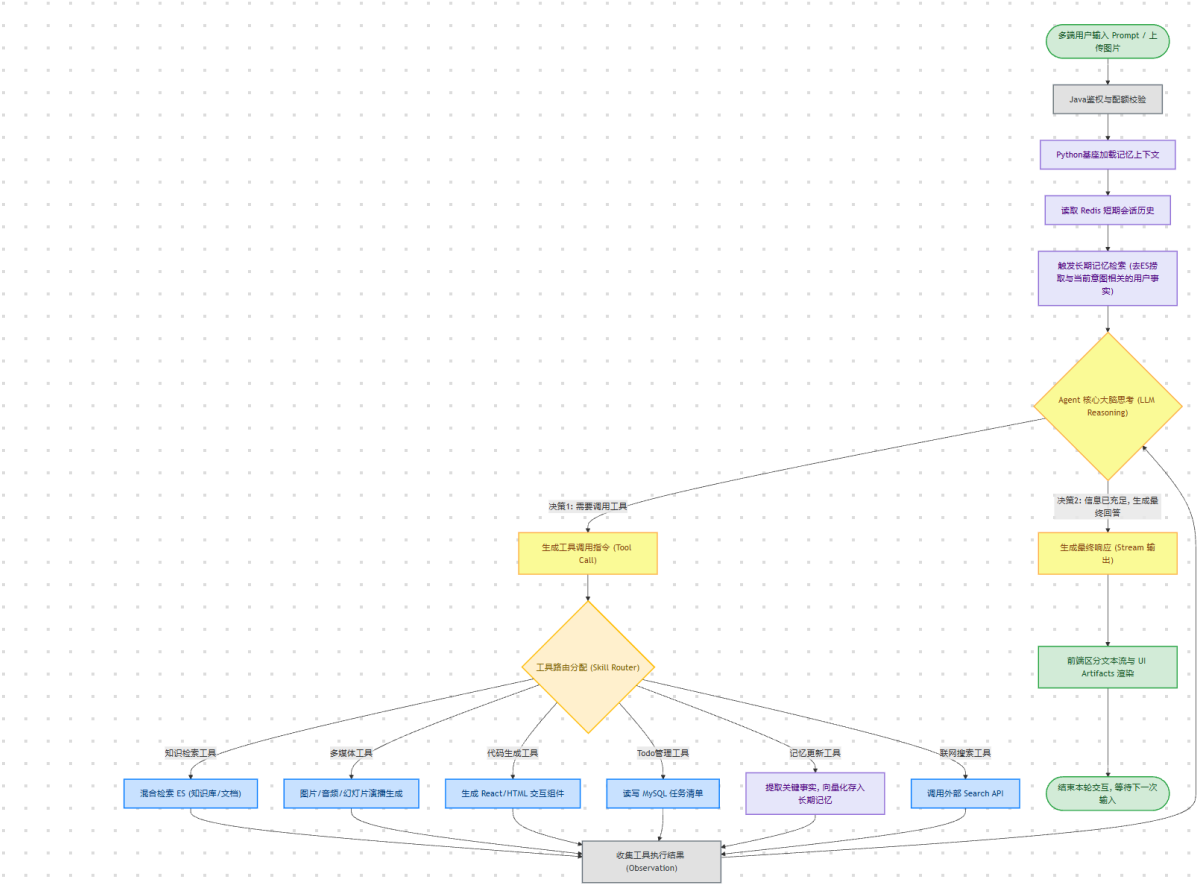
传统的接口是：`Request -> LLM -> Response`。
现在的 Agent 接口是：`Request -> While(LLM.wants_to_use_tool) { Execute_Tool ->`

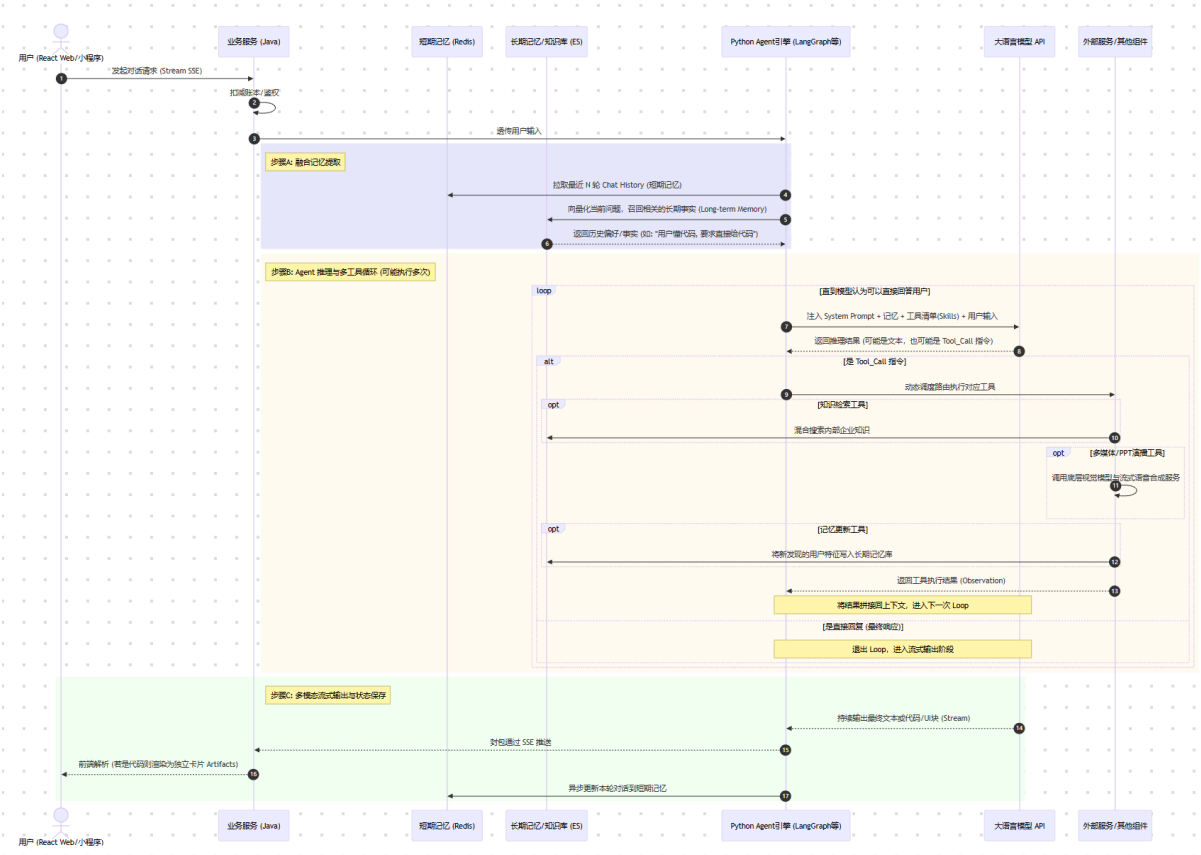
`Feed_back_to_LLM } -> Final Response`。
在这个循环中，大模型可能会自己跟自己“聊”好几个回合。比如：先调用 **“Todo 工具”** 发现有任务，再调用 **“知识检索工具”** 查相关的资料，最后再调用 **“多媒体工具”** 生成一段总结音频。所有的内部思考过程对用户是透明的，前端只会显示一个优雅的“Agent 思考中：正在检索知识... 正在生成图表...”。

## 3. 展现层的多模态渲染 (Artifacts UI)

由于我们赋予了模型 `ToolCode` (生成交互代码) 和多媒体等能力，前端（React 19）收到 SSE 流时，不能仅仅把 Markdown 当文本丢给解析器。前端需要实现**自定义渲染拦截器**：

- 收到纯文本：走正常的打字机效果。
- 收到特定标记的 React/HTML 代码块：在页面右侧或对话流中展开一个沙盒视窗（Iframe 或 Shadow DOM），直接将代码编译渲染为可交互的 UI 组件。
- 收到音频流链接：渲染原生的 Audio 播放器。




用户 (React Web/小程序)
业务服务 (Java)
短期记忆 (Redis)
长期记忆/知识库 (ES)
Python Agent引擎 (LangGraph等)
大语言模型 API
外部服务/其他组件
发起对话请求 (Stream SSE)
扣减账本/鉴权
透传用户输入
步骤A: 融合记忆提取
拉取最近 N 轮 Chat History (短期记忆)
向量化当前问题，召回相关的长期事实 (Long-term Memory)
返回历史偏好/事实 (如: "用户偏代码，要求直接给代码")
步骤B: Agent 推理与多工具循环 (可能执行多次)
loop
[直到模型认为可以直接回答用户]
注入 System Prompt + 记忆 + 工具清单(Skills) + 用户输入
返回推理结果 (可能是文本，也可能是 Tool_Call 指令)
alt
[是 Tool_Call 指令]
动态调度路由执行对应工具
opt
[知识检索工具]
混合检索内部企业知识
opt
[多媒体/PPT渲染工具]
调用底层视觉模型与流式渲染合成服务
opt
[记忆更新工具]
将新发现的用户特征写入长期记忆库
返回工具执行结果 (Observation)
将结果拼接回上下文，进入下一次 Loop
[是直接回复 (最终响应)]
退出 Loop，进入流式输出阶段
步骤C: 多模态流式输出与状态保存
持续输出最终文本或代码/UI块 (Stream)
对包通过 SSE 推送
前端解析 (若是代码则渲染为独立卡片 Artifacts)
异步更新本轮对话到短期记忆