

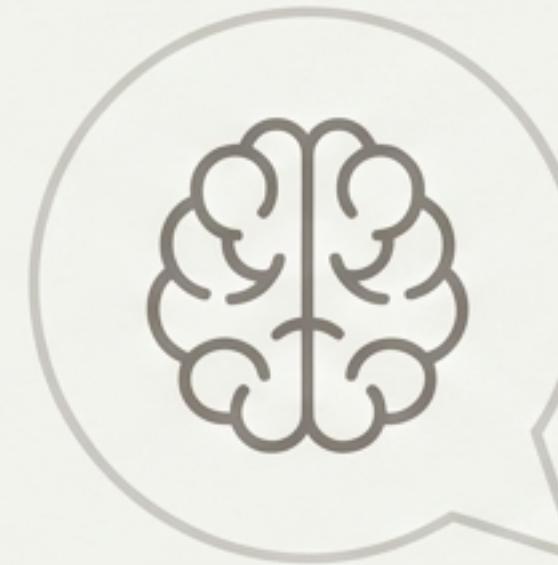
从“陪聊”到“干活”：构建AI友好的世界

一份面向行动智能体的数据、应用与API架构蓝图

AI的下一步：不止会说，更要会做

关键不是让模型更聪明，而是为模型构建一个它能理解、能协作的“世界”。

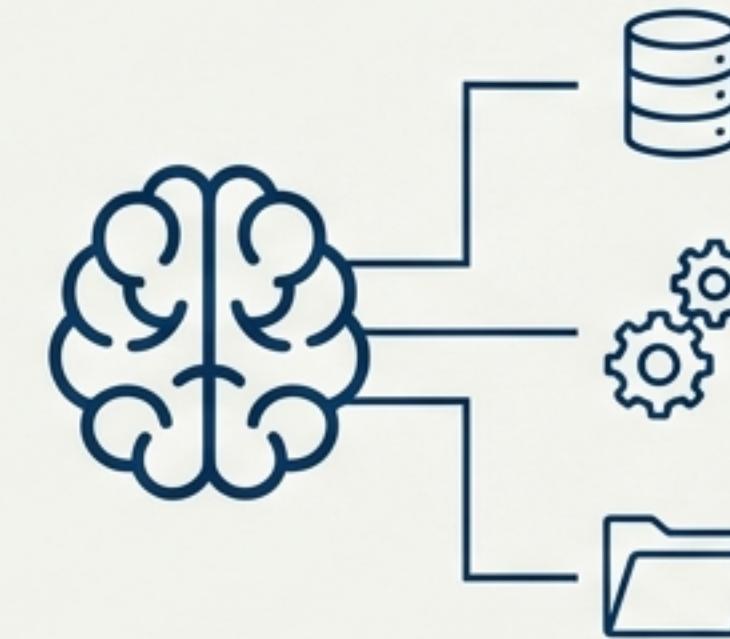
现状：语言模型（Language Model）



聊天
推理
生成

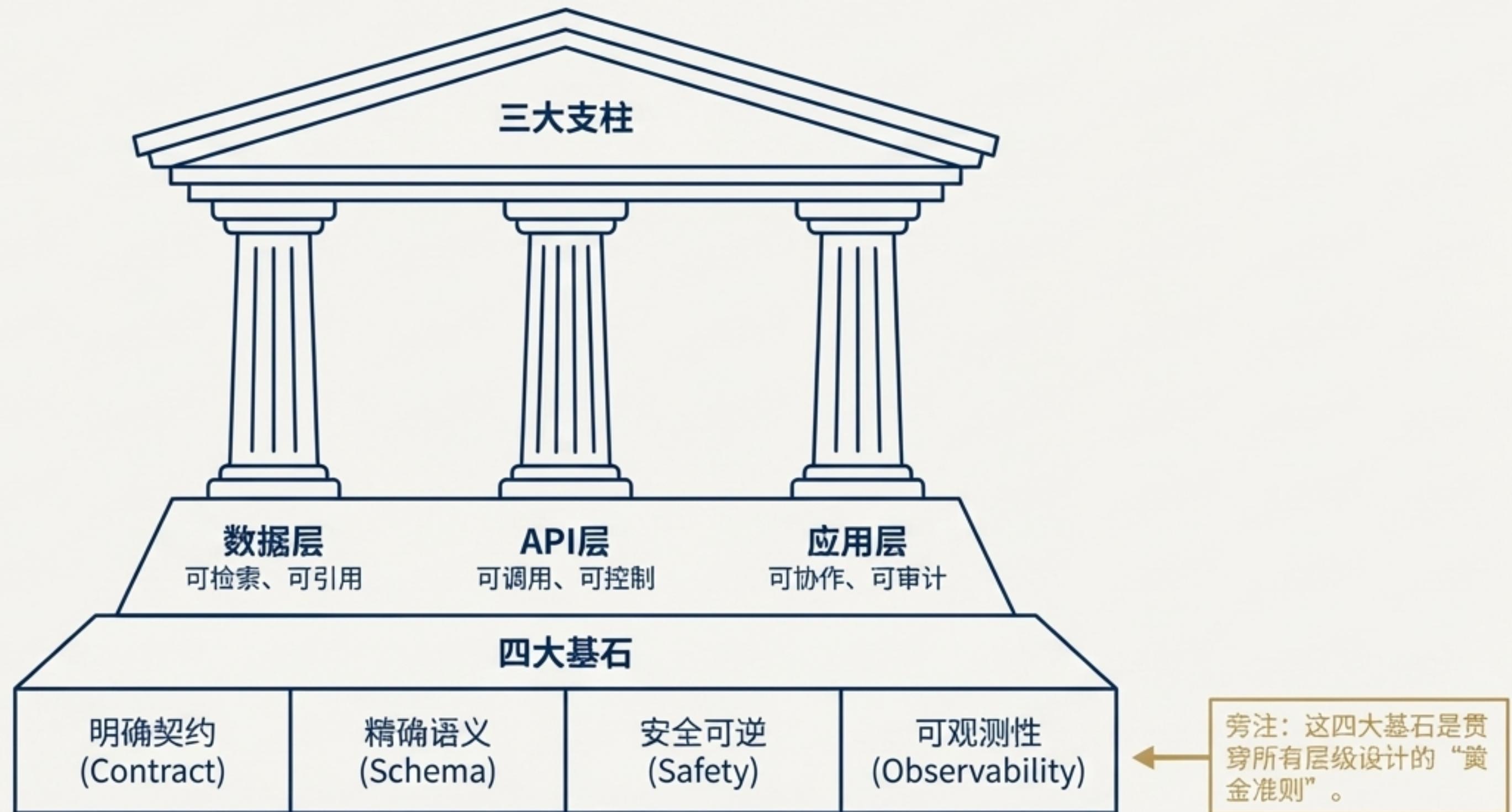


未来：行动智能体（Action Agent）



查询
计算
修改
执行

我们的蓝图：一个可控、可信的AI协作框架



支柱一：让数据成为AI可靠的“事实来源”

可检索

语义与结构化分块，向量+关键词混合索引，丰富的元数据（来源，时间，权限，版本）。

可引用

稳定的`source_uri`，`record_id`，和`version`，实现结果的可溯源性。

可计算

拒绝“自由文本驱动”。强制使用结构化字段、枚举值、标准单位与时区。

附加考虑

新鲜度：通过TTL与变更事件，支持“结果可能过期”的提示与刷新。

权限：在数据层实现实行级脱敏与敏感字段保护。

代码即证据：`search_customer_orders`

```
{  
  "name": "search_customer_orders",  
  "description": "当需要查询指定客户的历史订单状态、发货时间或退款记录时使用。不要用于库存查询。",  
  "parameters": {  
    "customer_id": { "type": "string",  
      "pattern": "^[A-Z0-9]{8}$" },  
    "date_range": {  
      "type": "object",  
      "properties": {  
        "start": { "type": "string",  
          "format": "date" },  
        "end": { "type": "string",  
          "format": "date" }  
      },  
      "required": ["start", "end"]  
    },  
    "status": { "type": "string",  
      "enum": ["pending", "fulfilled", "refunded"]  
    }  
  }  
}
```

精确指令

“精确语义”的实证

支柱二：让API成为AI安全的“行动手臂”

契约即一切

使用 JSON Schema 定义输入、输出、错误。返回值包含 `data`，`source`，`confidence`。

控制副作用

使用幂等键 (`idempotency_key`)。将高风险操作设计为“草稿 → 模拟执行 → 正式执行”的状态机。

错误可修复

返回结构化错误 (e.g., `error_code: missing_parameter`) 和可行的修复建议。

附加考虑

性能: 支持分页 (next_cursor) 和流式响应。在响应中暴露速率限制信息 (rate_limit_remaining)。

代码即证据: `send_email`

```
{  
    "name": "send_email",  
    "description": "向指定接收人发送电子邮件。仅  
    仅在用户明确确认发送内容后调用。",  
    "parameters": {  
        "to": {  
            "type": "string",  
            "format": "email"  
        },  
        "subject": {  
            "type": "string",  
            "minLength": 1  
        },  
        "body": {  
            "type": "string",  
            "minLength": 1  
        },  
        "idempotency_key": {  
            "type": "string"  
        }  
    }  
}
```

强调“安全可逆”基石

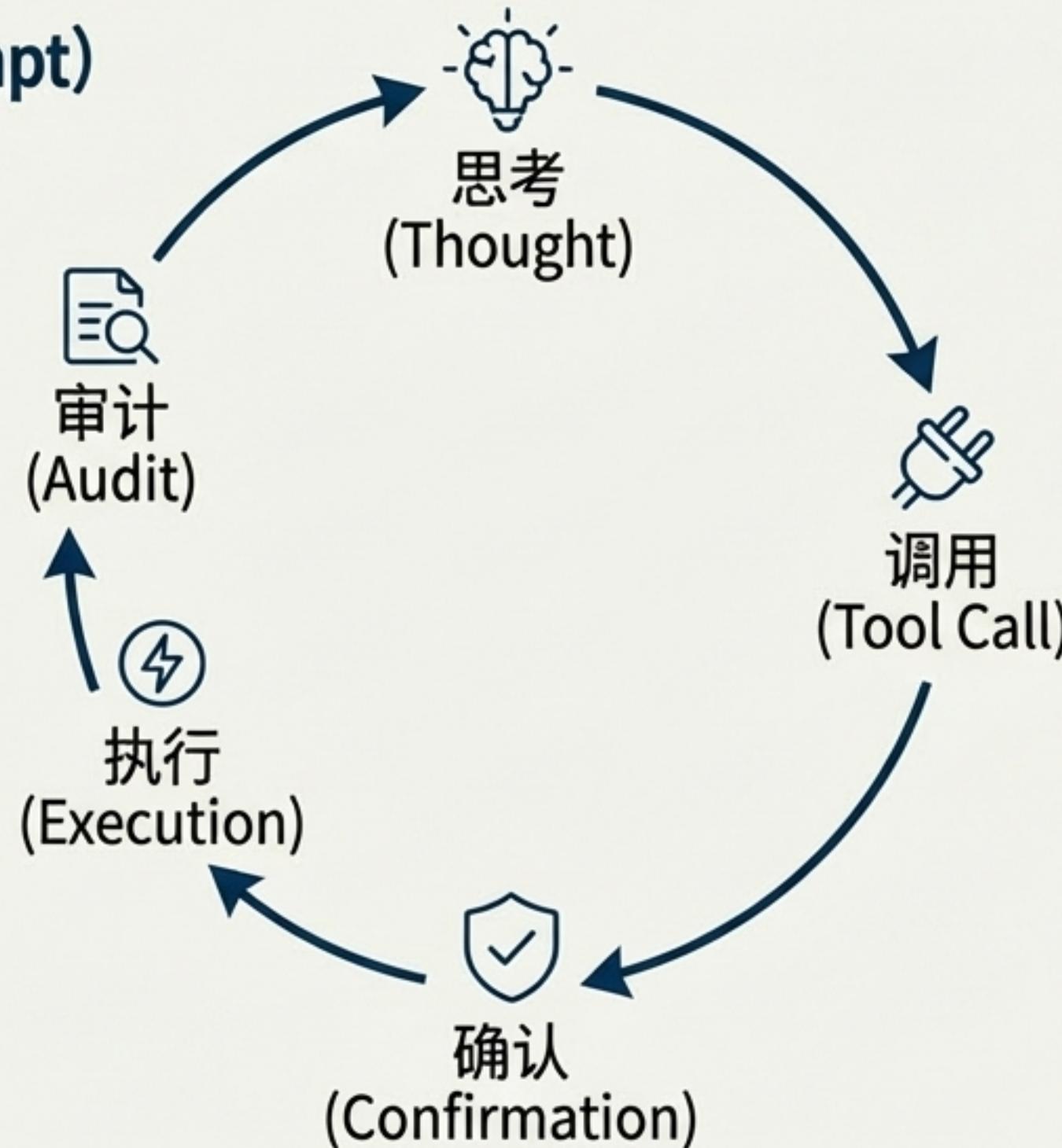
支柱三：让应用成为AI高效的“操作系统”

系统提示词 (System Prompt)

设定“思考-调用-执行-回答”的元认知循环。要求缺参时反问，禁止编造信息。

弹性执行 (Resilient Execution)

具备失败降级策略（如：换工具、缩小范围、请求用户补充信息）。



技能目录 (Agent Skills)

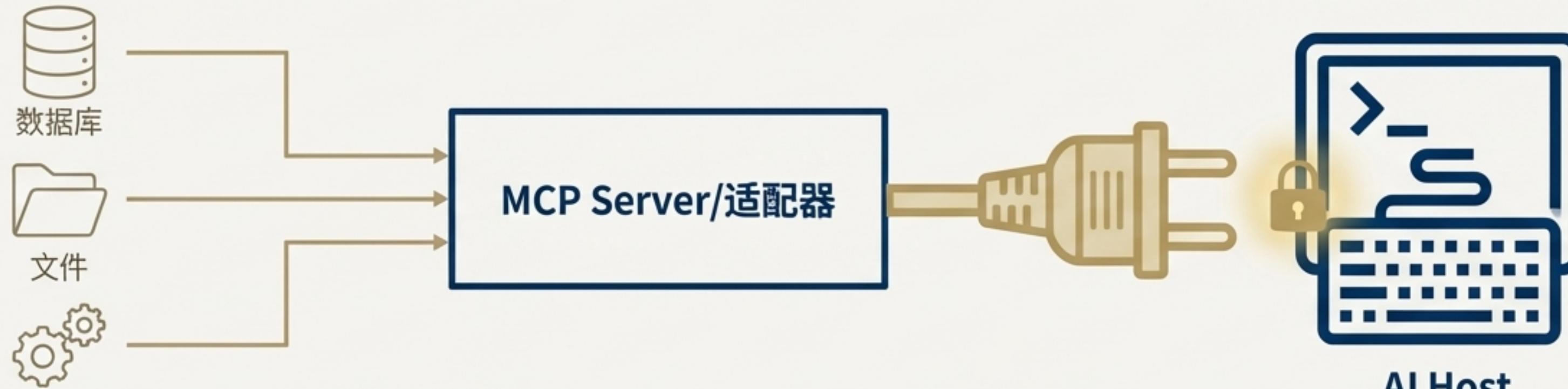
每个工具都附有清晰的“何时使用/不使用”描述、前置条件和权限要求。

人在环路 (Human-in-the-Loop)

关键操作必须有关闭、草稿预览和可回滚机制。

统一协议 (MCP) : 把世界做成即插即用的“插头”

核心理念：将任何数据或API（Resources）用标准适配器（Server）封装，供对话端（Host）即插即用。



示例：Windows 配置

```
{  
  "mcpServers": {  
    "filesystem": {  
      "command": "npx",  
      "args": ["-y", "@modelcontextprotocol/server-filesystem", "0:\\\\grc"]  
    },  
    "sqlite": {  
      "command": "npx",  
      "args": ["-y", "@modelcontextprotocol/server-sqlite", "0:\\\\data\\\\sales.db"]  
    }  
  }  
}
```

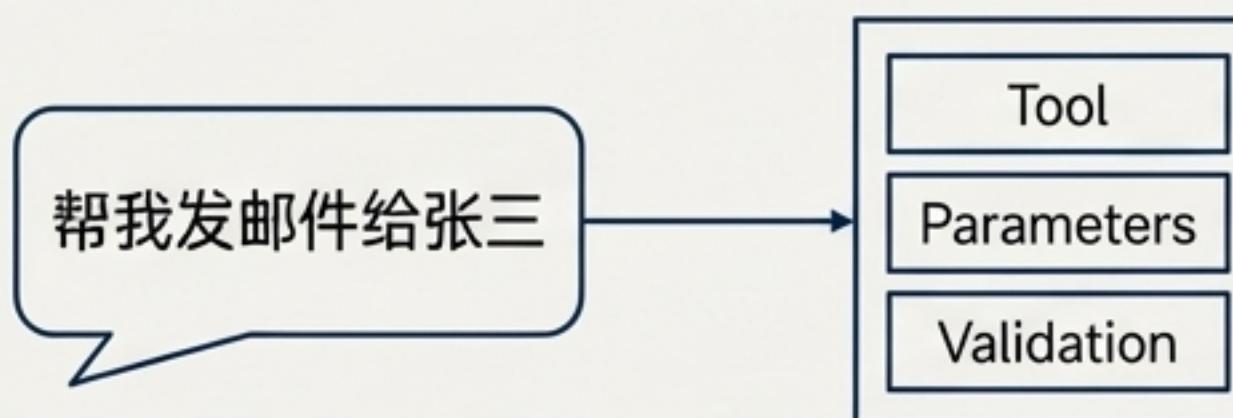
接入本地文件系统和数据库就是如此简单。

技术核心：用结构化约束战胜语言模糊性

函数调用 (Function Calling)

作用：将自然语言意图精确映射到结构化的工具调用。

目的：杜绝“下周一”这类模糊指令，强制参数校验。



自然语言请求

“帮我发邮件给张三”

约束解码 (Constrained Decoding)

作用：使用 JSON Schema 或语法（如 EBNF/PEG）直接“锁定”模型的输出格式。

目的：确保输出100%可被机器解析和消费。

语法约束示例 (EBNF)

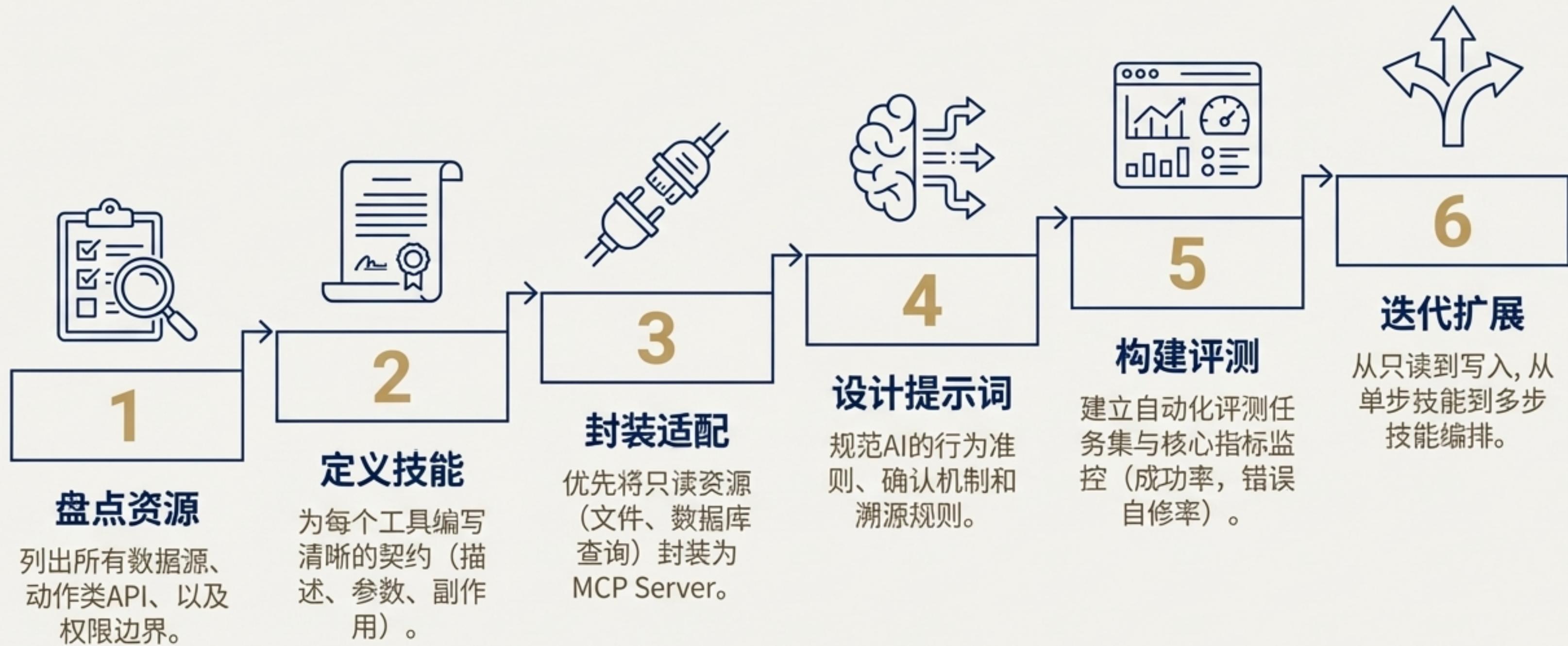
```
request = "{ fields }"
fields = to "," subject "," body
to = "\"to\" ":" string
subject = "\"subject\" ":" string
body = "\"body\" ":" string
...
```

严格结构化输出

```
{
  "to": "zhang.san@example.com",
  "subject": "...",
  "body": "..."
}
```

约束解码

您的实施路线图：从0到1的六个步骤



AI友好架构的设计十原则



描述即契约：清晰到足以降低误用。



错误可修复：返回“原因+建议”。



明确参数：类型、格式、约束越细越好。



溯源强制：每个结果都携带来源。



幂等优先：动作类工具支持安全重复调用。



权限内嵌：在数据和工具层实现安全。



确认关口：副作用操作默认先生成草稿。



可观测：保证全链路日志与重放能力。



可分页可流式：大结果支持增量消费。



逐步扩展：先接入只读技能，再扩展动作技能。

警惕！这些是常见的失败之源



描述模糊，导致误用

工具描述和参数定义随意。



返回自由文本，无法复用

API返回无结构、无引用的文本。



动作无确认，后果失控

动作类工具没有幂等设计或用户确认环节。



大结果一次性返回，导致超时

API不支持分页或流式传输。



错误信息模糊，无法恢复

只返回泛泛的 "500 Error"，无修复提示。

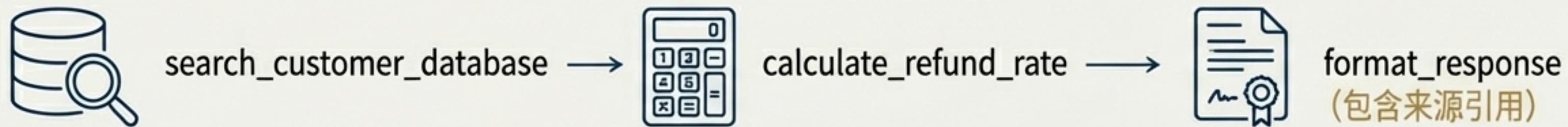


权限管理混乱，导致泄露

权限控制在应用层，而非数据/工具层。

三个典型场景，看懂完整流程

场景 1：查—算—说 (Read-Compute-Report)



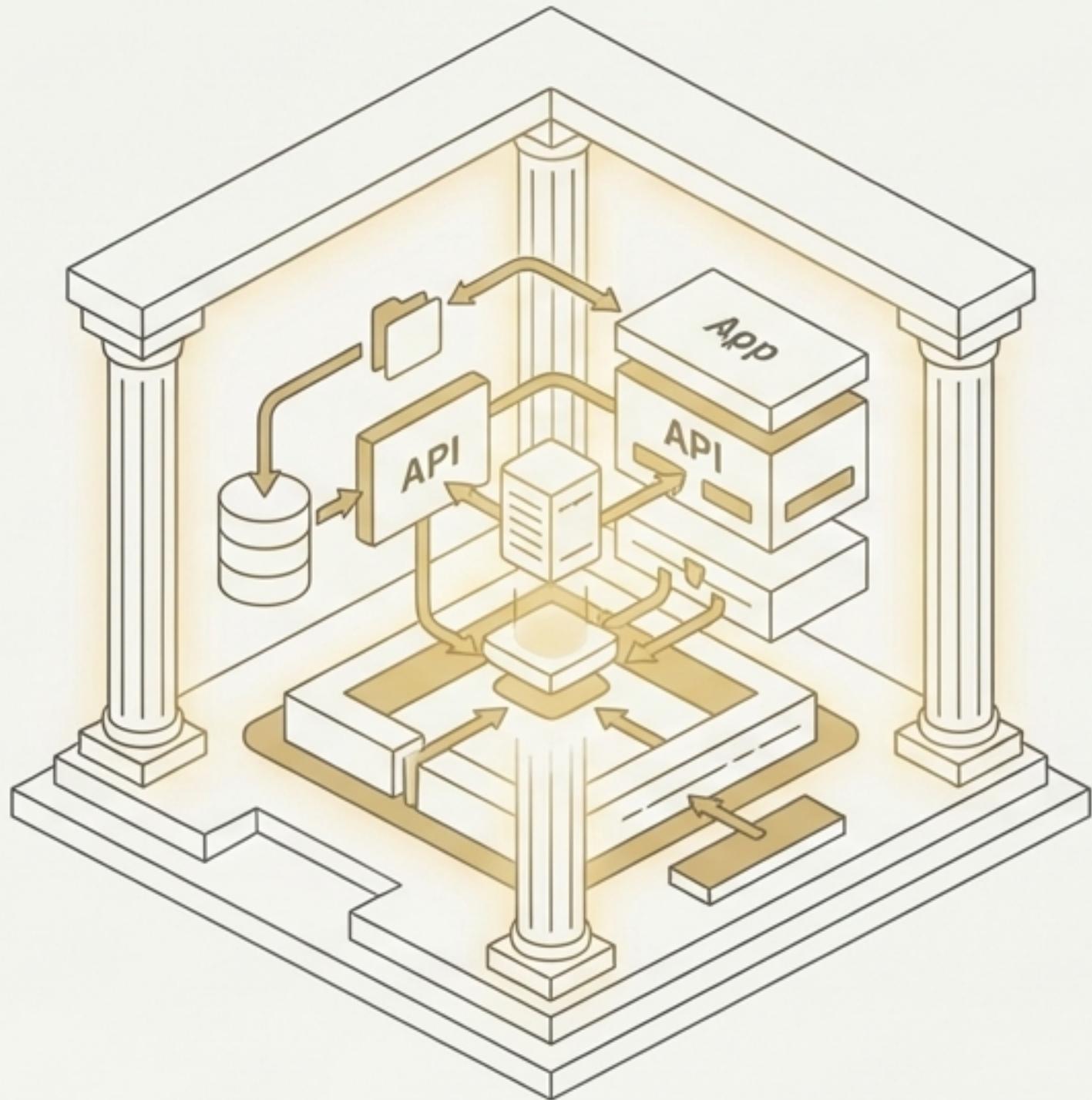
场景 2：写—审—发 (Draft-Review-Send)



场景 3：读—评—改 (Read-Propose-Apply)



核心总结：AI友好不是“更会说话”，而是“更好合作”



三大核心行动

- 让数据可取：把世界做成 MCP 插头。
- 让 API 可用：把工具写成技能契约。
- 让应用可审：用约束解码守住结构与安全。

当你的系统变得“AI友好”，模型才能从“聊天”真正进化到“干活”，并且“可控、可审、可扩展”。

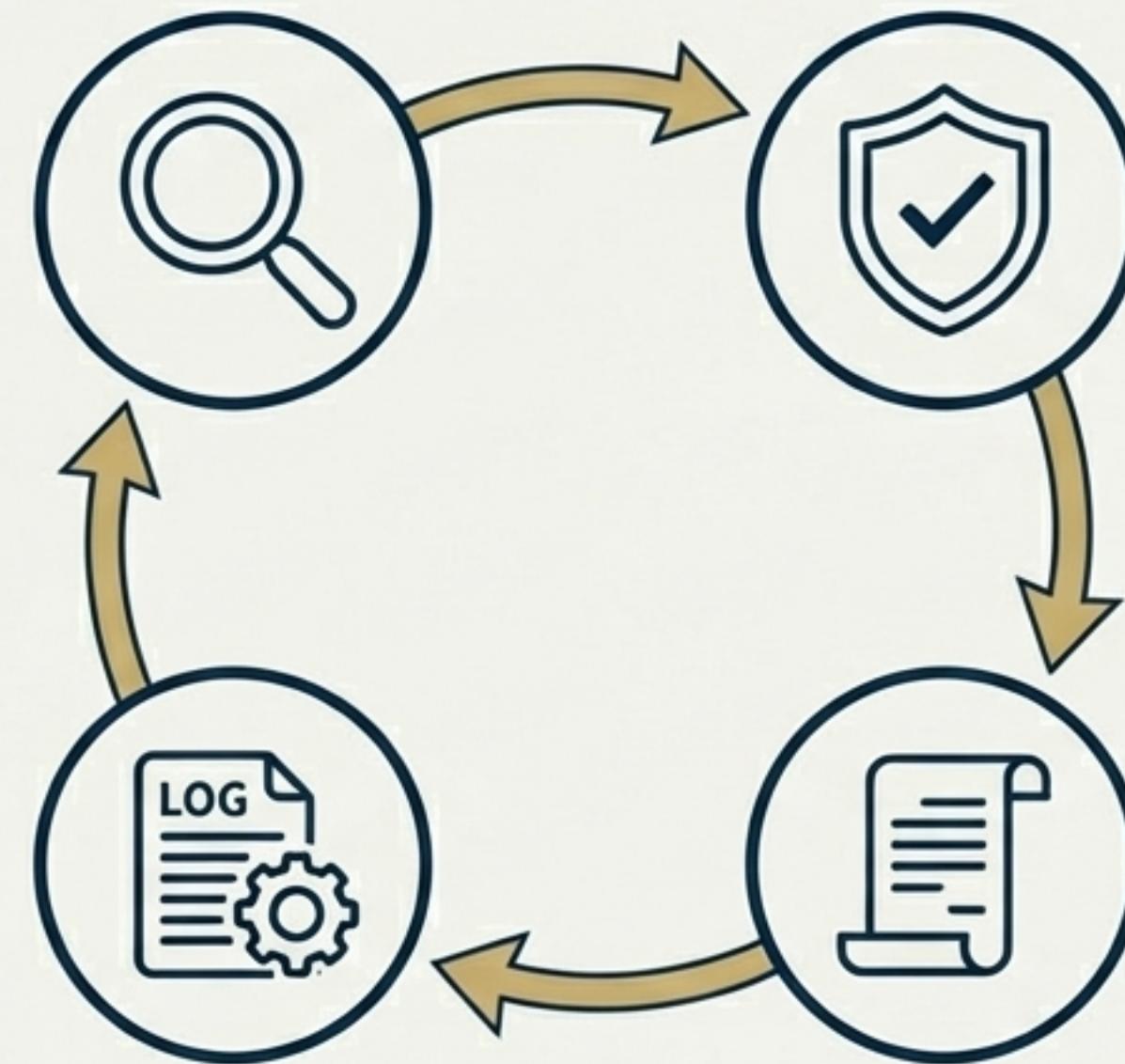
万里长征，始于足下：从你的第一个最小闭环开始

第一步：从一个“只读”技能开始（例如，查询数据库或读取文件）。

第二步：增加一个有“确认机制”的轻量级“动作”技能（例如，创建草稿）。

第三步：为它们编写一个明确的系统提示词，规范其行为。

第四步：建立最基础的可观测日志，记录调用与结果。



立即开始构建你的第一个可控、可审、可扩展的AI Agent。

谢谢 & 交流

演讲团队：AI架构设计组

联系方式：arch-contact@example.com

