



SkillKit 技术架构全景

重新定义 Agent 运行时: Skills are New Software

个人 AI 的操作系统

核心范式：Agent 作为新一代操作系统

Interface

~~API Endpoints~~

CLIs

// Bash 是万能的通用接口。

Interaction

~~App UI~~

Multi-modal

// 自然语言与视觉的融合。

Architecture

~~Operating System~~

Agents

// Agent 是新的 OS，负责资源调度与任务编排。

// Vibe coding is new product management.

Agent 设计哲学象限

自主性 (Low Autonomy) → 高自主性

Conversation First (AutoGen)
多角色辩论

Skill First (SkillKit)

- 裸 ReAct 循环
- 完全靠模型推理
- Token 高效

Workflow First (LangGraph)
生产级, 限制强

Tool First (LangChain)
易混乱, Token 爆炸

结构化程度 (Fixed Structure) → 灵活性 (Flexible)

什么是 Skill? (vs Traditional Tools)



Traditional Tool

JSON Schema



Model Selection



Function Call

每个工具占用大量 Token



SkillKit Skill

自然语言知识 (Markdown)



Model Learning



Bash Execution

利用 Prompt Caching 降低 80% 成本

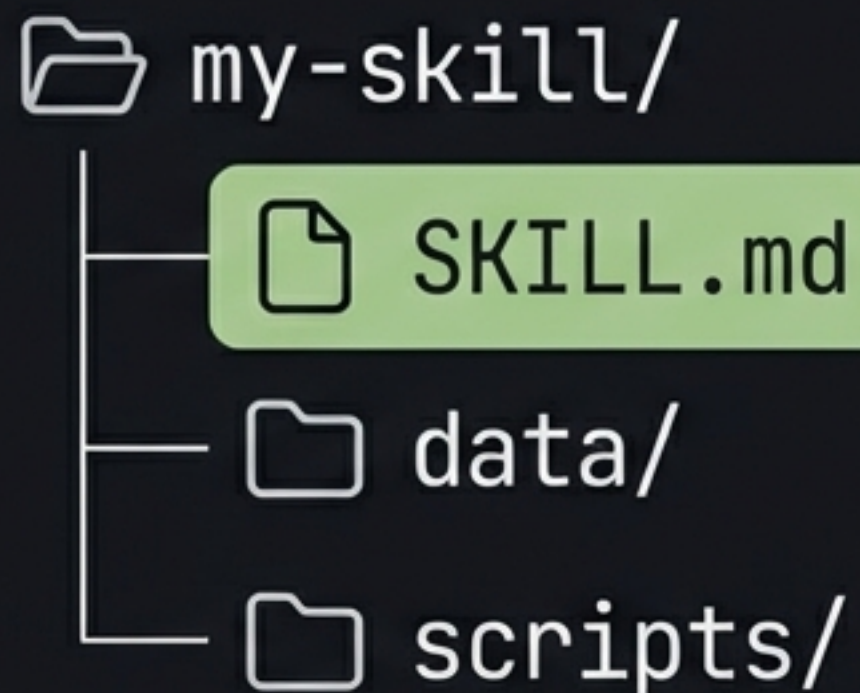
Skill 不是函数调用, 是注入 System Prompt 的 ‘指导知识’。

虚拟机架构隐喻

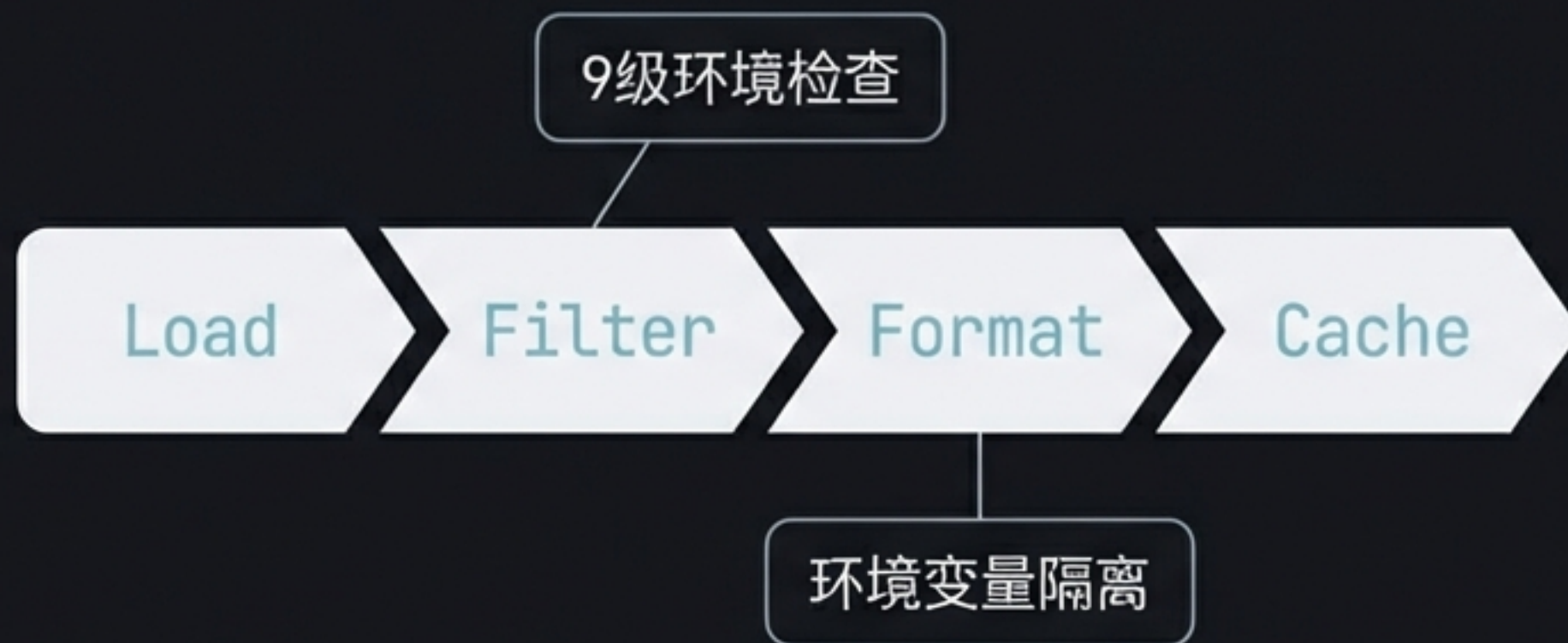


Skills 体系详解

Structure



Pipeline

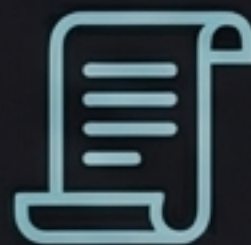


优先级：用户配置覆盖内置默认

极简工具哲学 (Primitive Tools)



execute



execute_script



read

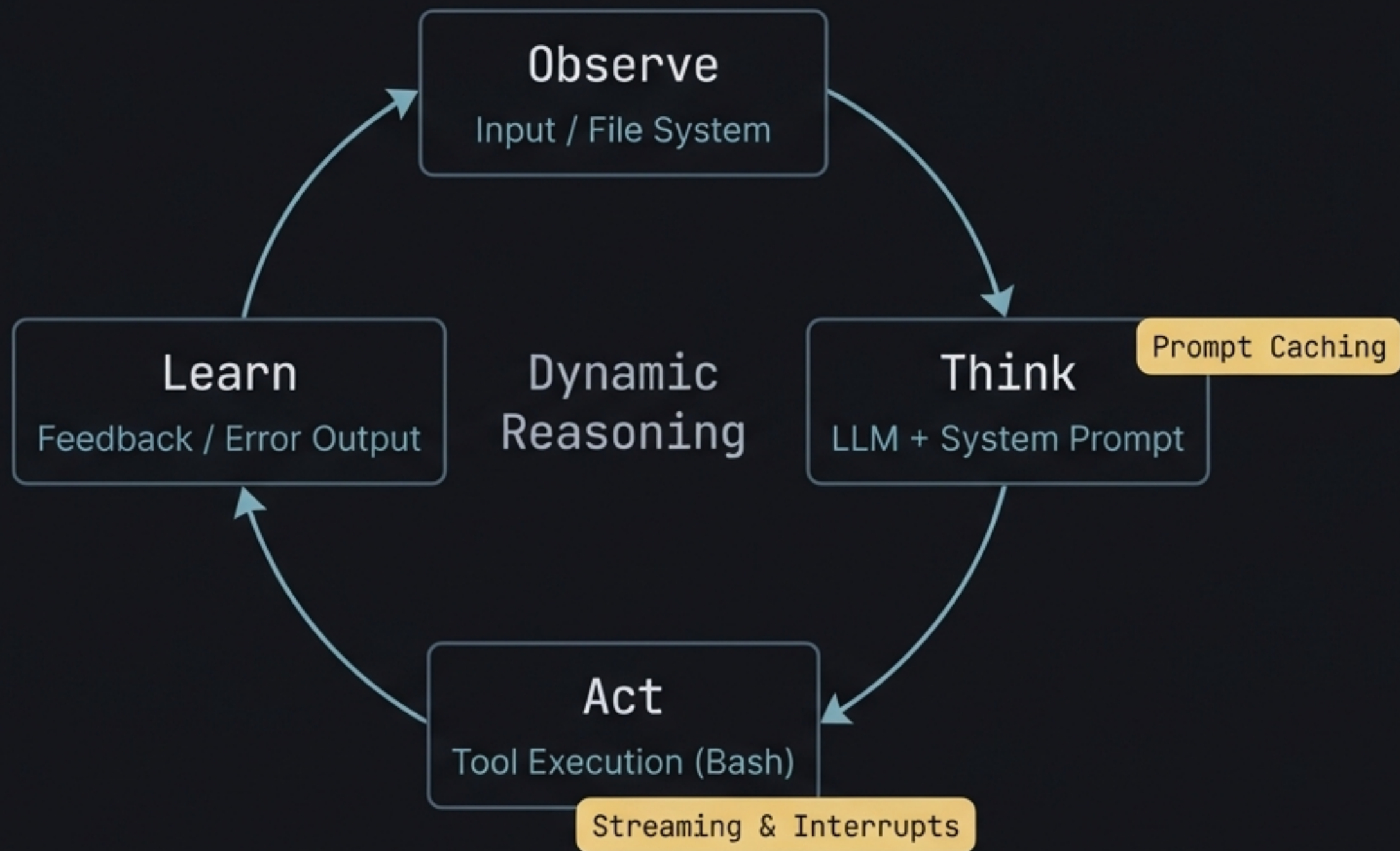


write

“Bash is universal. Don't wrap APIs, teach the model CLI.”

TUI 扩展工具: Edit, Find, Grep, Ls (仅用于交互优化)

Agent 执行循环 (The ReAct Loop)



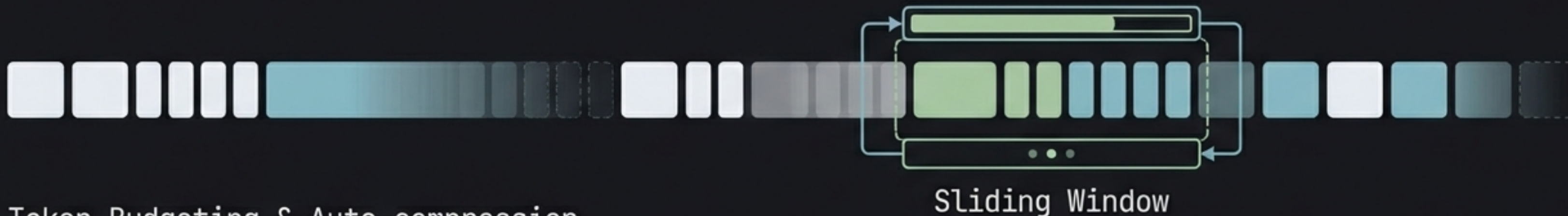
流式输出与协作式中断

事件总线 (EventBus) 与 Hooks

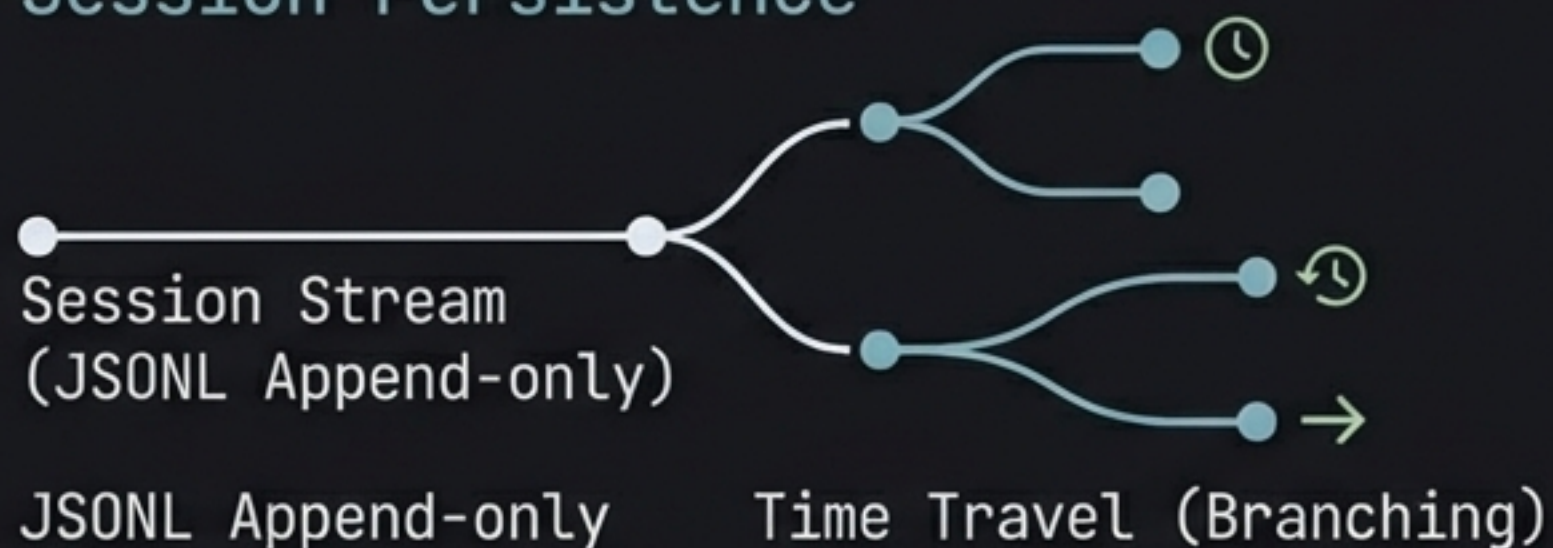


记忆与上下文管理

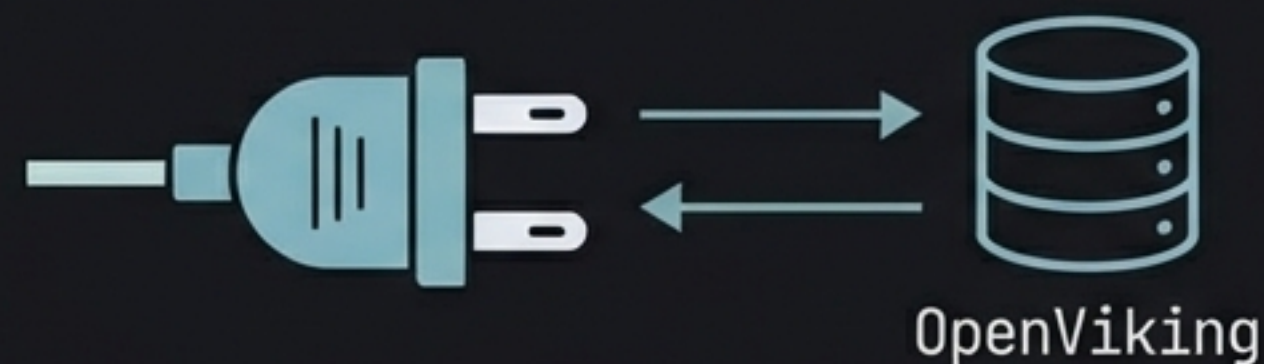
Context Manager



Session Persistence



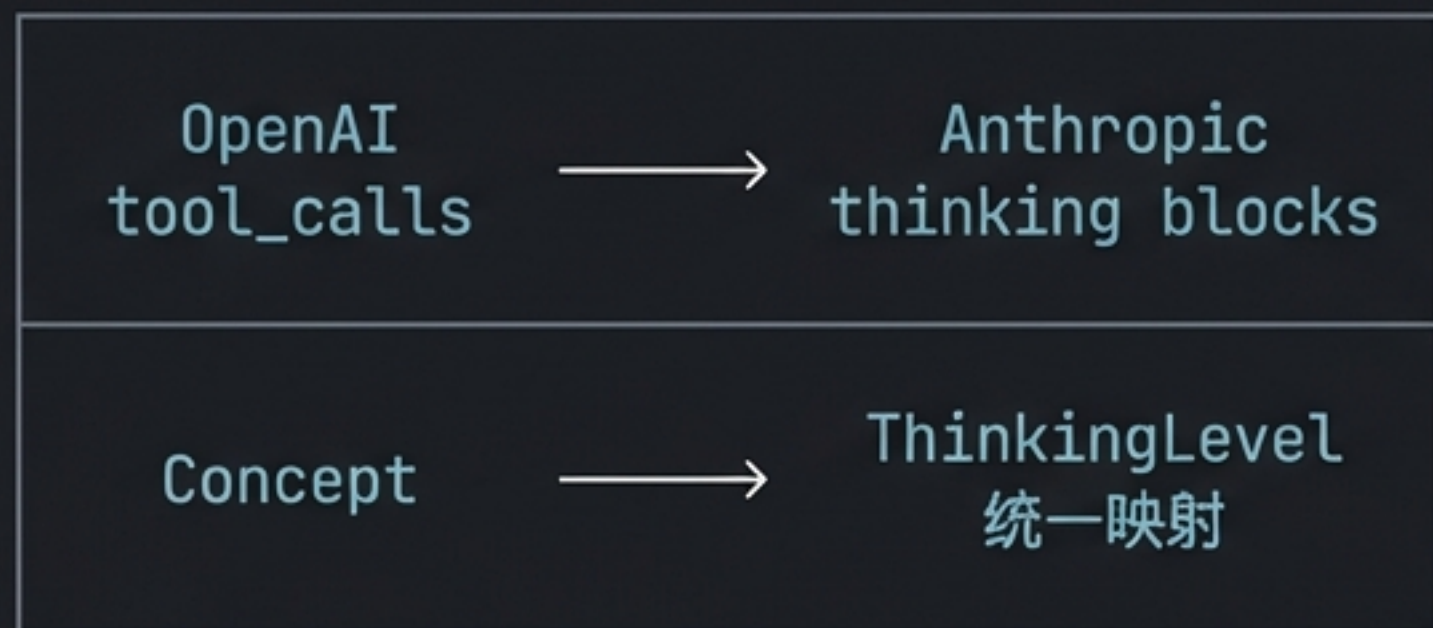
Long-term Memory



Recall / Save / Explore tools

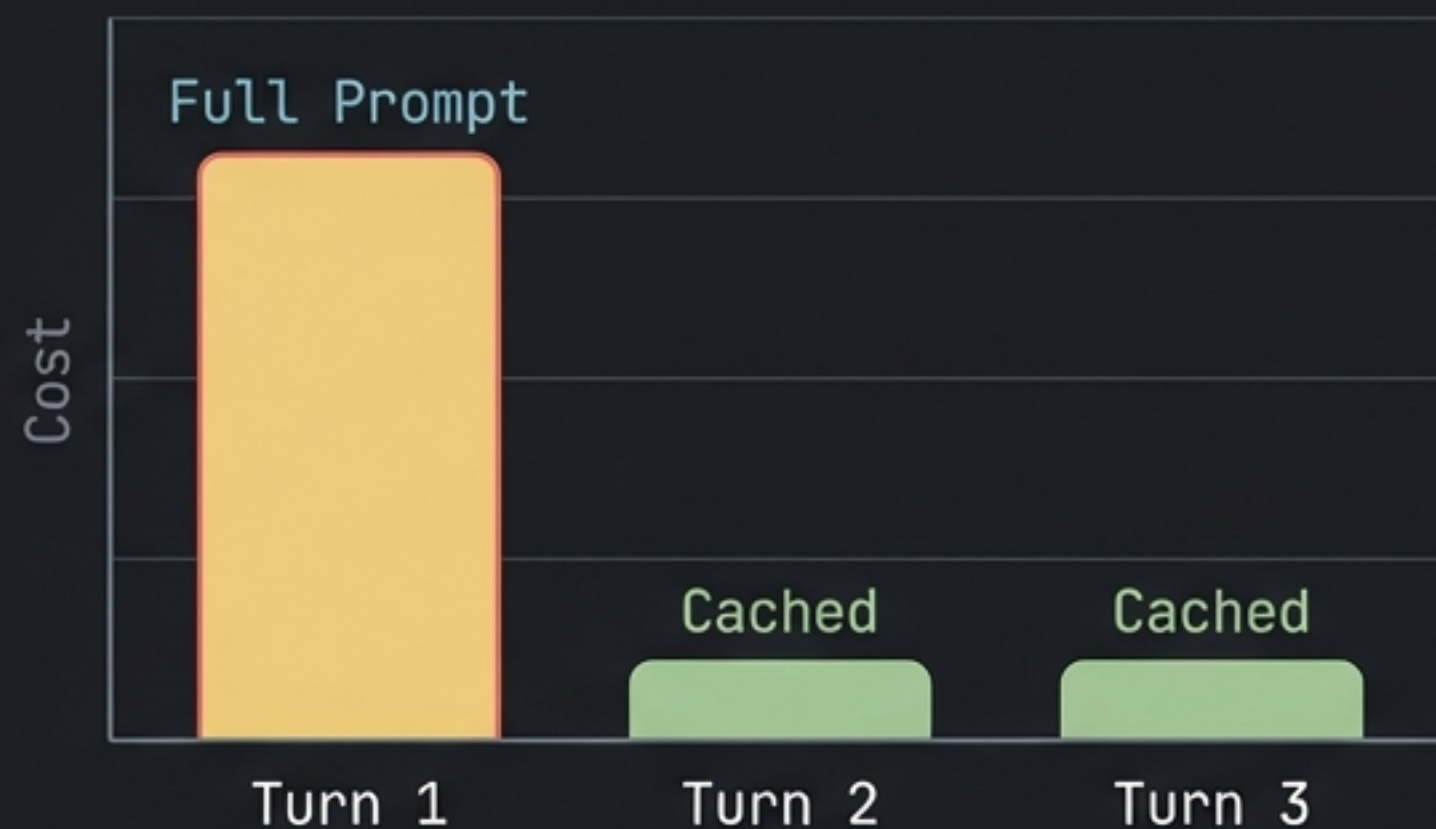
跨模型适配与 Prompt Caching

Adapters



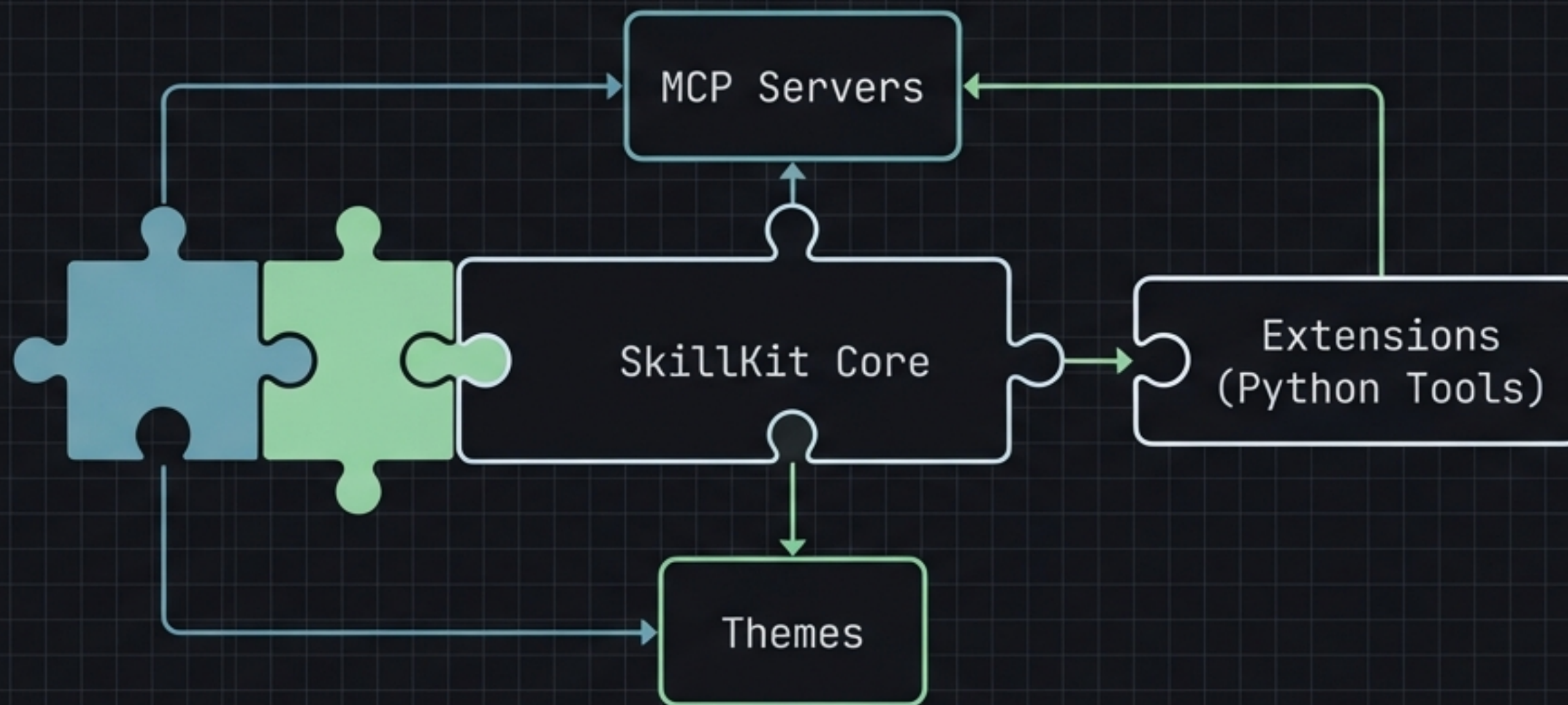
Token Economics

Cost per Turn



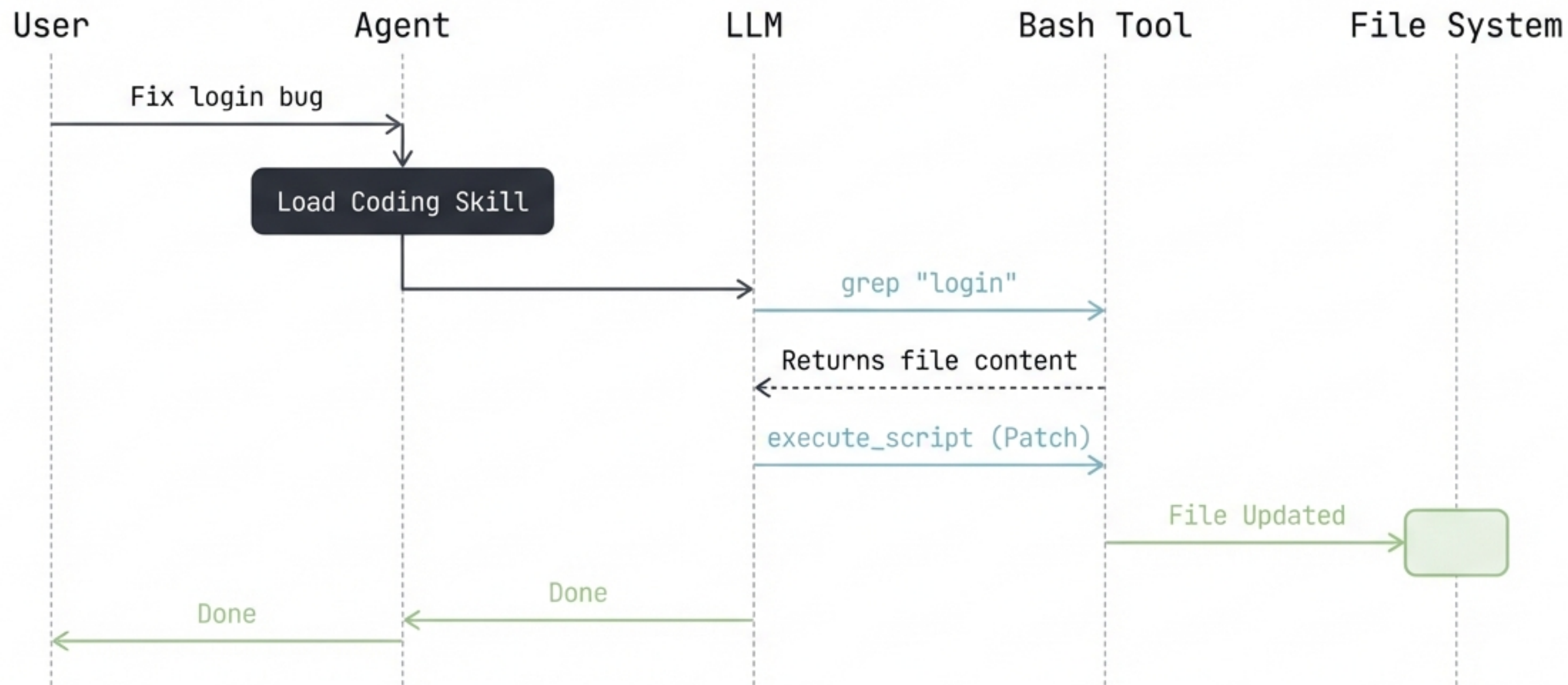
System Prompt (Skills) 缓存利用率 > 90%

扩展体系与 MCP

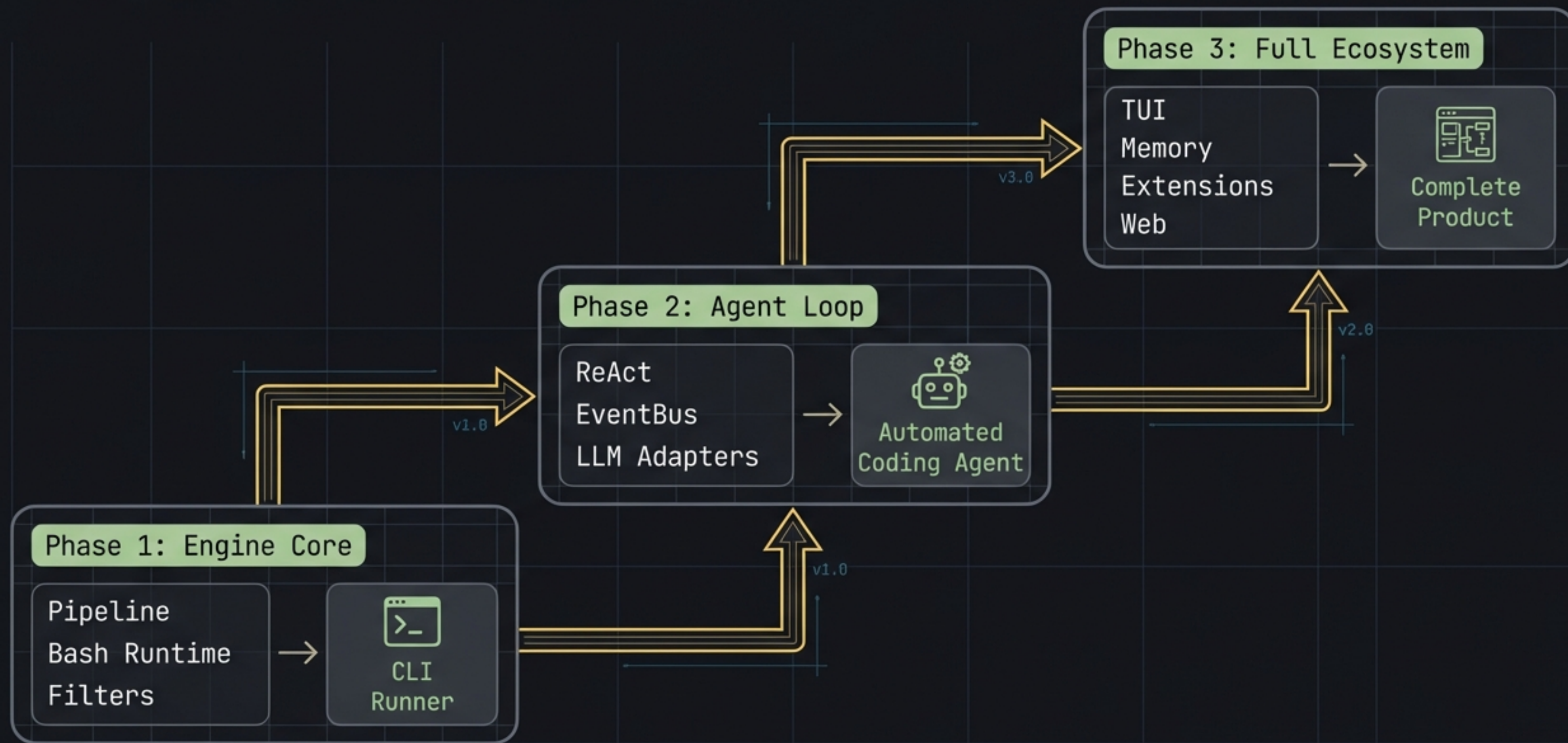


```
# 一行代码接入记忆扩展  
extension.setup_memory()
```


端到端执行案例：Fix login bug



三阶段实现路线图



Knowledge Injection over Tool Stacking 以知识注入替代工具堆砌

Simple enough to understand, Complete enough to use.
足够简单以致完全理解，足够完整以致真正可用。

SkillKit

The OS for your personal AI.