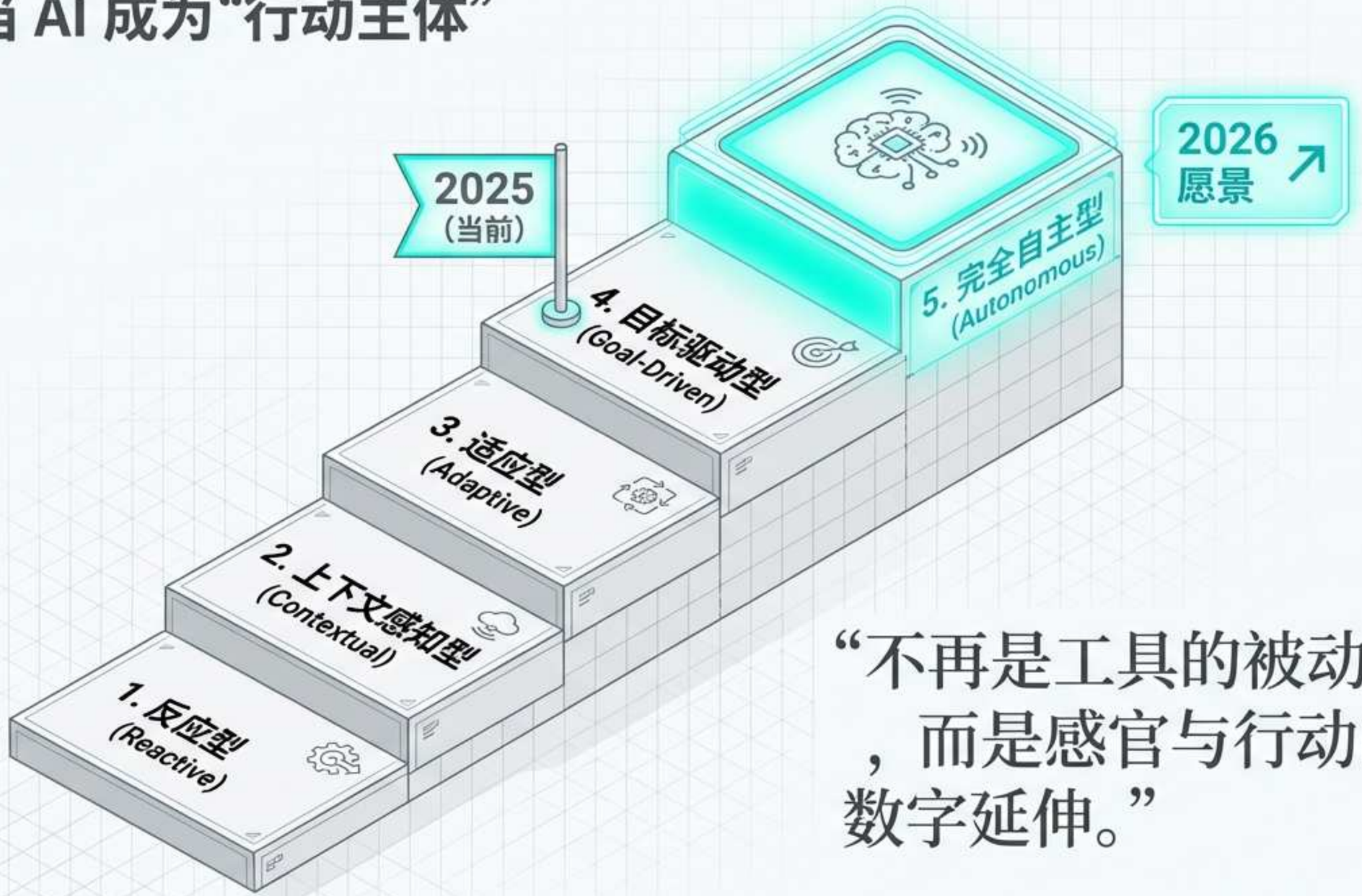
AI 团队协作：从 Vibe Coding 迈向复合式工程

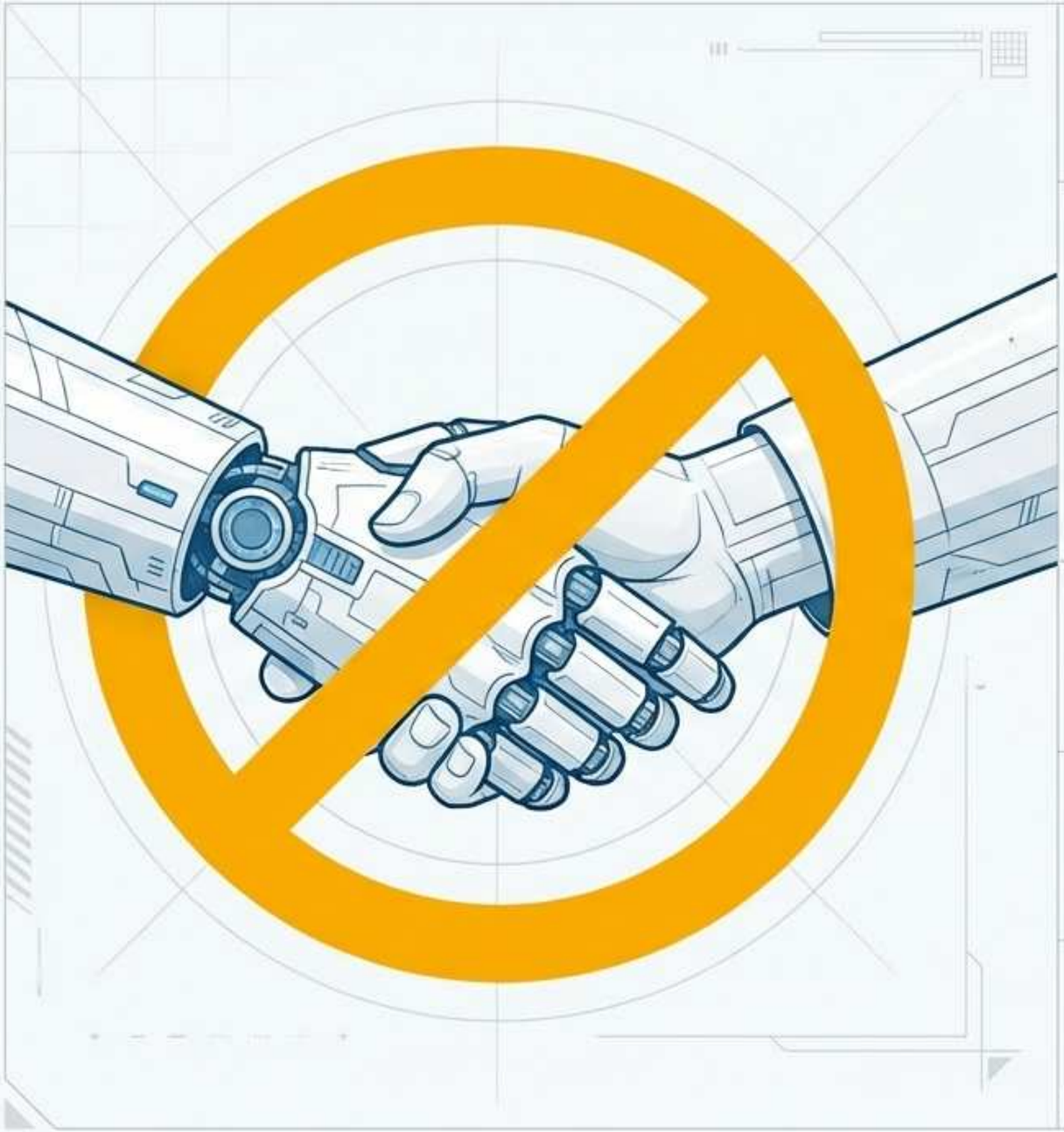
软件工程 2.0 时代的协作范式升级与陷阱

金鹏 · 工程总监

序言：当 AI 成为“行动主体”



“不再是工具的被动响应，
，而是感官与行动的
数字延伸。”



警惕“数字队友”的拟人化陷阱

+$670,000

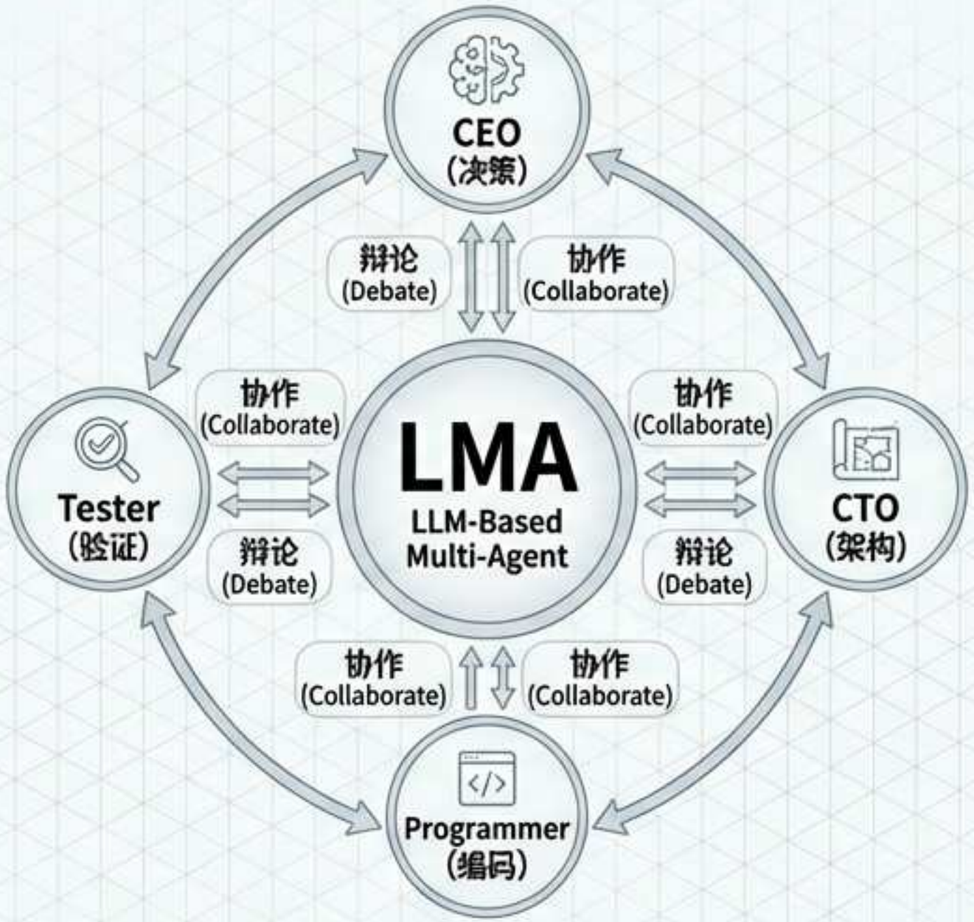
涉及 AI 的数据泄露平均成本增幅（IBM Report 2025）

1. 盲目信任：语言的拟人化掩盖了其概率本质

2. 级联错误：单个 Agent 的幻觉在网络中被指数级放大

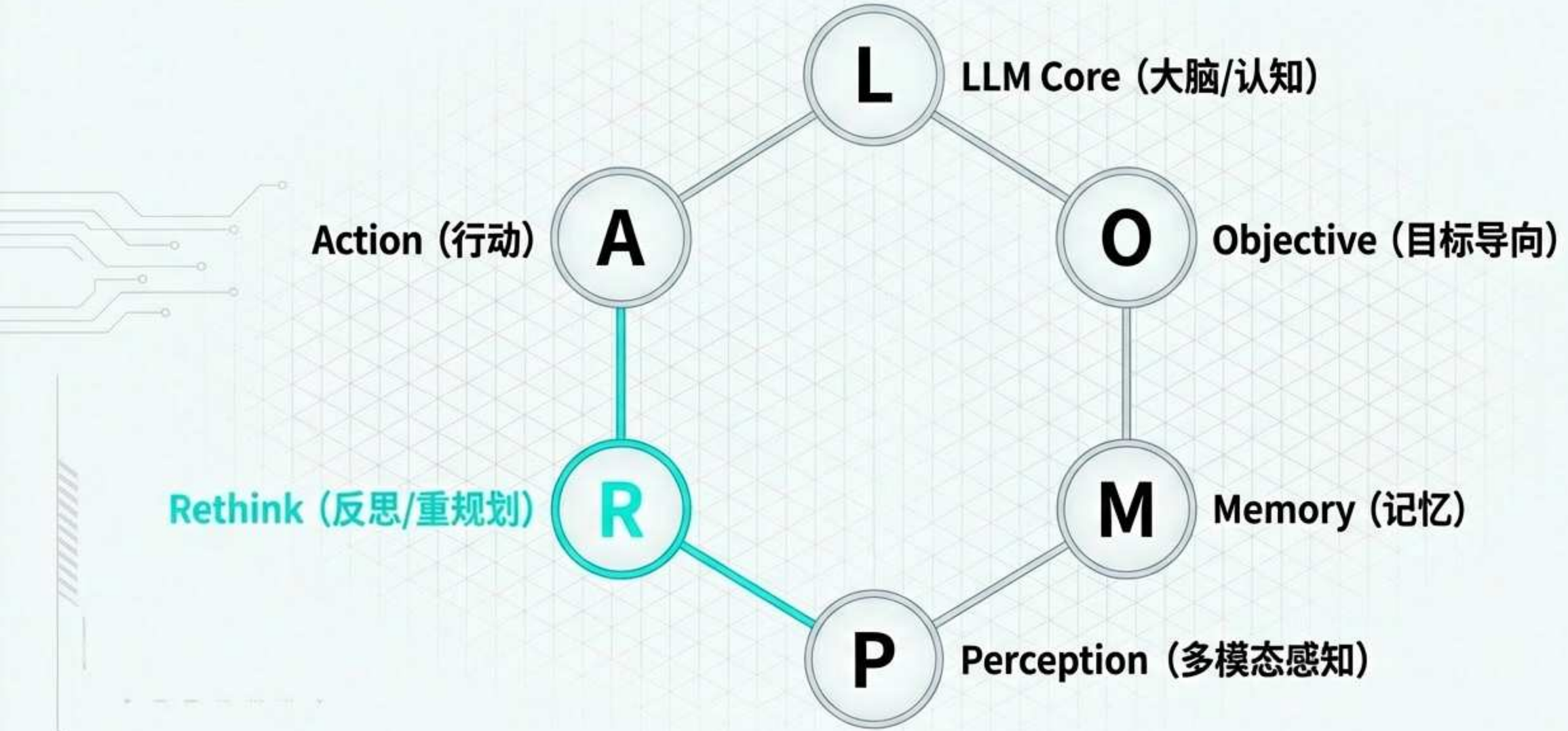
3. 治理真空：缺乏“人”的判断力，为优化指标而破坏价值

核心定义：LLM-Based Multi-Agent (LMA) 系统



定义：集成多个专用智能体，通过角色扮演（Role-Playing）和交互反馈（Interaction）突破单体 LLM 的能力瓶颈。
关键机制：Synergistic Collaboration（协同增效）

智能体工程原语：< L, O, M, P, A, R >



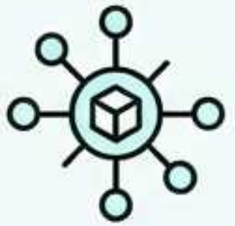
LMA 系统的三大工程优势

自主问题解决 (Autonomy)

模拟敏捷开发，自动将高层需求拆解为子任务（Decomposition）。

鲁棒性与容错 (Robustness)

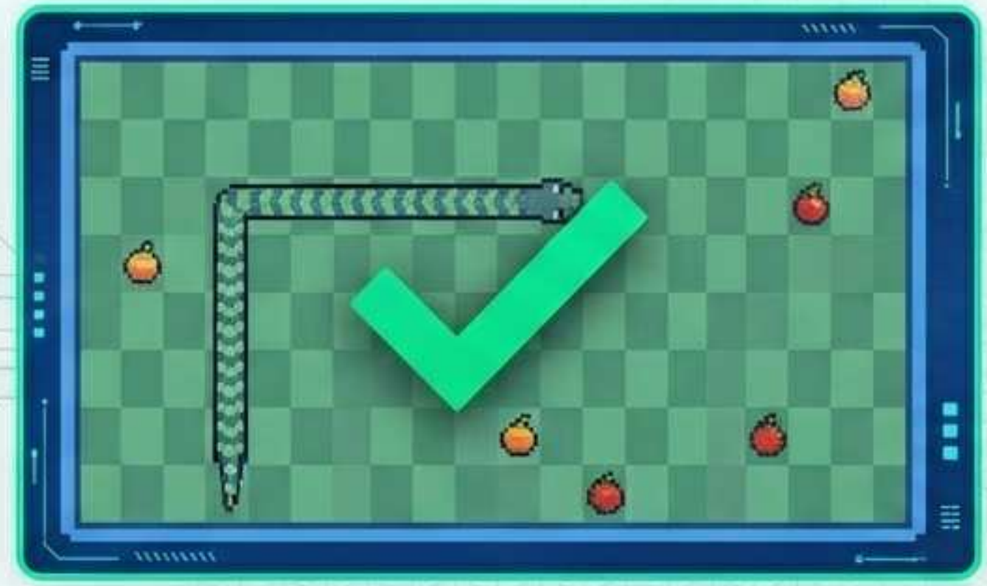
通过交叉审查（Cross-Examination）与多角色辩论，消除单体模型的“幻觉”。



复杂系统的可扩展性 (Scalability)

动态增加 Agent 节点处理新框架/依赖，线性扩展以应对非线性复杂度。

案例分析：ChatDev 的能力边界

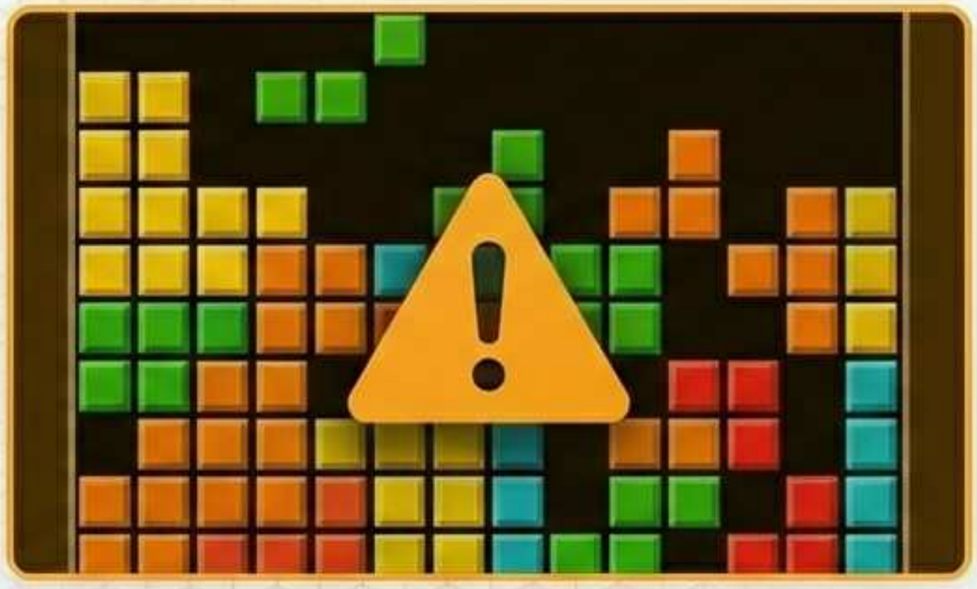


贪吃蛇（Snake Game）

耗时：76s

成本：$0.019

结果：完美运行



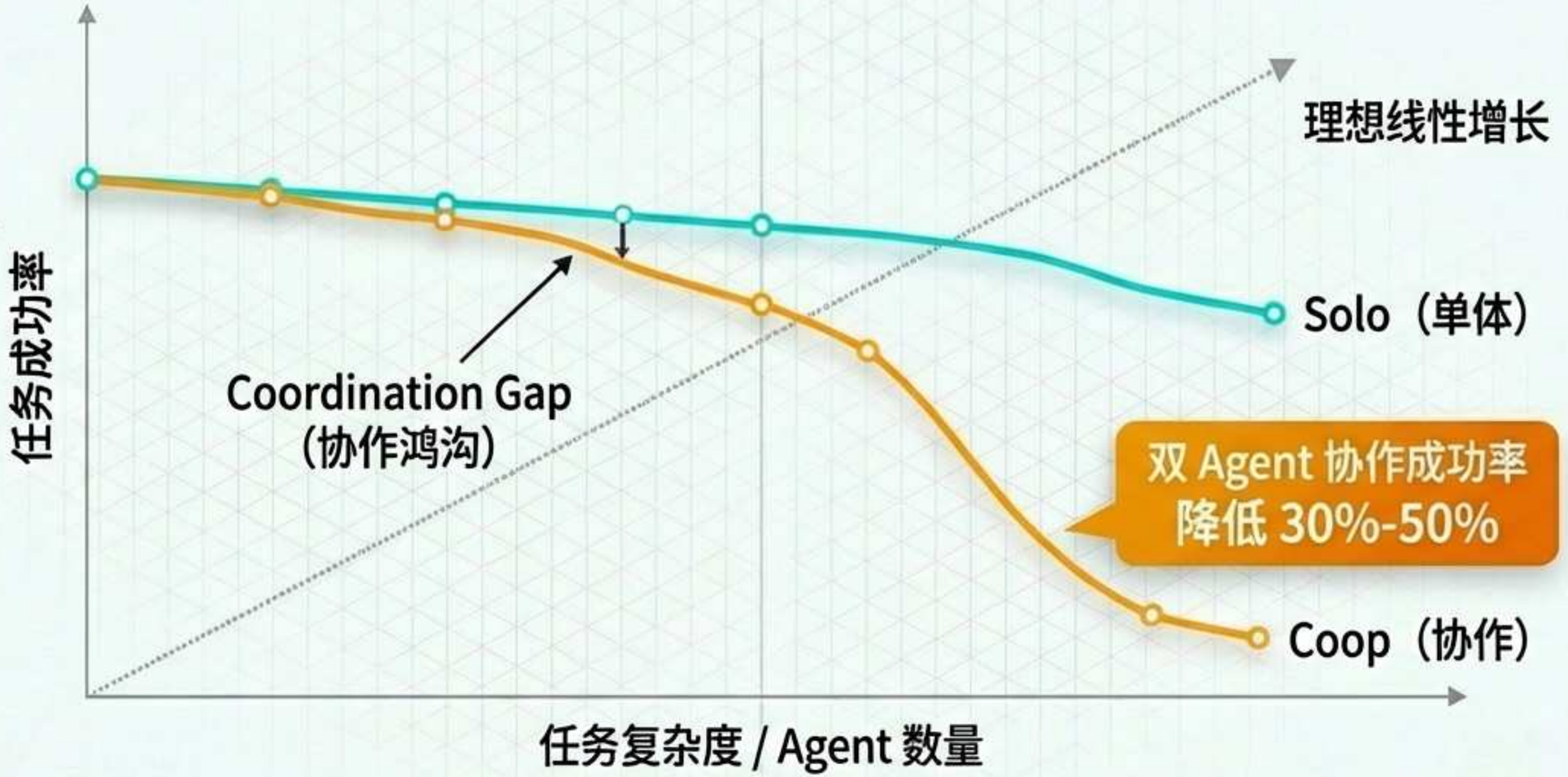
俄罗斯方块（Tetris）

尝试次数：10次

致命缺陷：无法消除满行

瓶颈：深度逻辑推理缺失

协作的诅咒：1 + 1 为什么小于 2?



来源：CooperBench 实验数据

沟通的幻象：空间协调 vs. 语义协调

空间协调 - Spatial ✅

```
41  new main() {
42    if lorner {
43      ...
50    Agent A = Line 50   ← Agent A 改 Line 50
56      ...
67    }
79    ...
80    Agent B = Line 80   ← Agent B 改 Line 80
81    ...
92  }
```

Agent A 改 Line 50，Agent B 改 Line 80。

✔ Git Merge 无冲突

语义协调 - Semantic ❌

Agent A:
case_sensitive
= False (默认)

→ `lower c >= ())`

Break

Agent B: 依赖
case_sensitive
= True

→ `if (om c <= c) == True`

⚠ 逻辑一致性崩溃

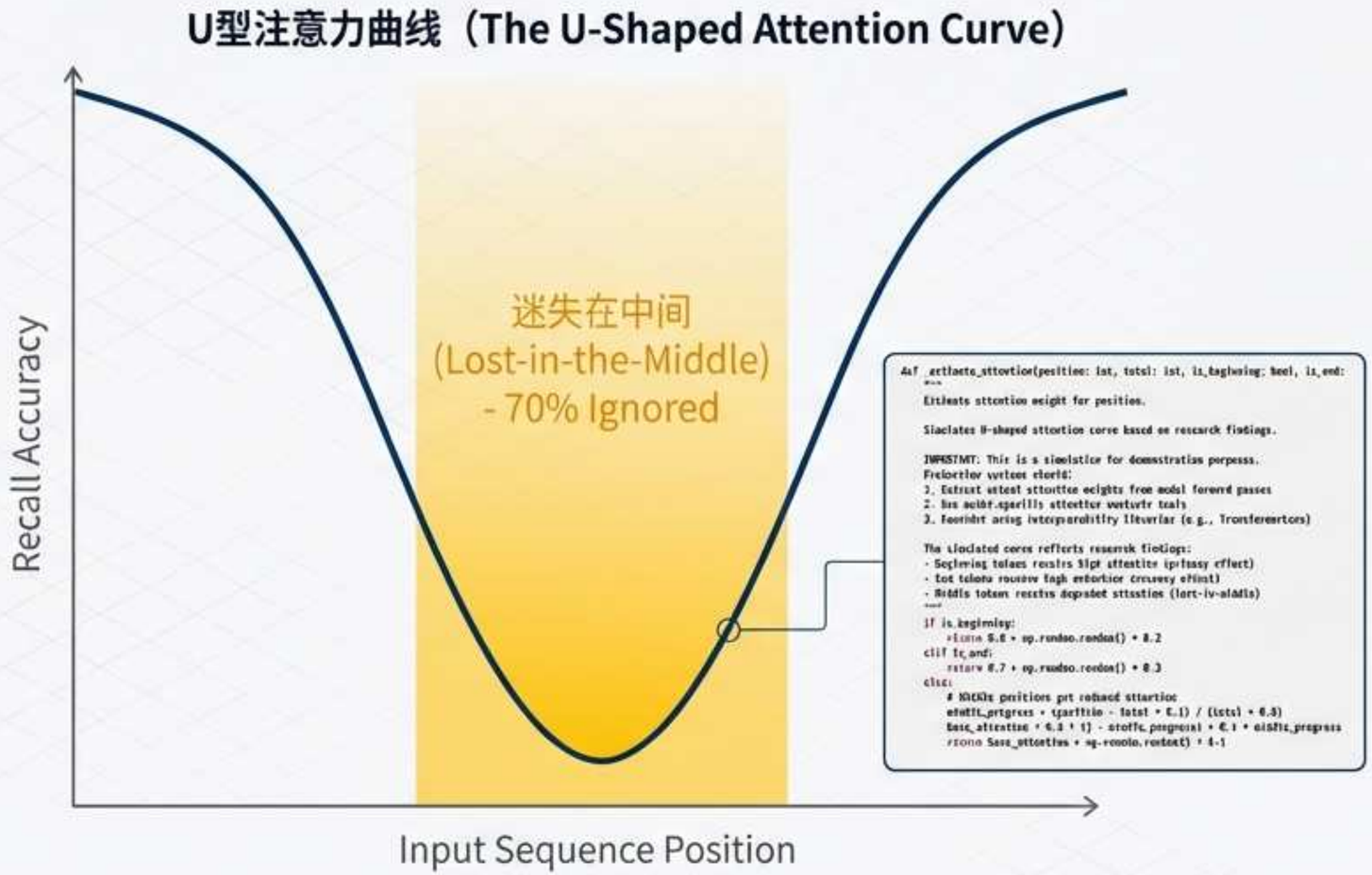
失败案例中 Plan-to-Question Ratio 仅为 1.31（无效沟通）

多智能体失败根因分类学

结论：我们不需要更快的编码员（Coder），我们需要更好的架构师（Planner）。

上下文工程：对抗“信号腐烂”的艺术

Managing the Attention Budget for High-Fidelity Reasoning



高信号提取（High-Signal Extraction）

拒绝原始日志堆叠，使用 `Observation Masking` 仅保留核心结论。

结构化锚定（Anchored Summarization）

持久化状态块：

- `Session Intent`（当前意图）
- `Decisions Made`（已做决策）
- `Current State`（当前状态）

防止中毒（Anti-Poisoning）

及时清洗幻觉（Hallucination），阻断逻辑错误的连锁反应。

渐进式披露：Agent Skills 的按需加载

Optimizing Cognitive Load via a 3-Layer Funnel


```
# SKILL.md Header
name: data-analysis
description: 支持 CSV/JSON 格式的数据清洗与可视化...
# 只有匹配 description 时，才加载下方正文
```

HINDSIGHT：结构化记忆网络

Epistemic Clarity: Separating Facts from Beliefs

World Network (世界网络)

客观事实 (Objective Facts)

Example: `API v2.0 deprecates field X.`

Experience Network (经验网络)

个人经历 (First-person History)

Example: `I failed to connect to DB yesterday.`

Observation Network (观察网络)

中立摘要 (Entity Profiles)

Example: `User A prefers Python.`

Opinion Network (观点网络)

主观信念 (Subjective Beliefs)

Example: `Method B is likely unstable (Confidence: 0.8).`

Retain（存储） → **Recall（召回）** → **Reflect（反思/更新信念）**

AOP：智能体导向规划法则

The Iron Triangle of Task Decomposition

1. Solvability (可解性)

每个子任务必须能被单体 Agent 独立解决。

2. Completeness (完整性)

所有子任务覆盖用户请求的 100% 范围。

3. Non-redundancy (无冗余)

消除重叠任务，优化 Token 消耗。

规范驱动开发（SDD）流水线

Code Serves the Spec, Not the Other Way Around

1. 宪章
(Constitution)
不可协商的原则
(Principles)

2. 规范
(Specify)
单一事实来源
(Source of Truth)

3. 计划
(Plan)
技术架构
(Architecture)

4. 任务
(Tasks)
可执行单元
(Actionable)

5. 实施
(Implement)
可丢弃的代码
(Disposable Code)

Documentation is the Product.
Code is an Artifact.

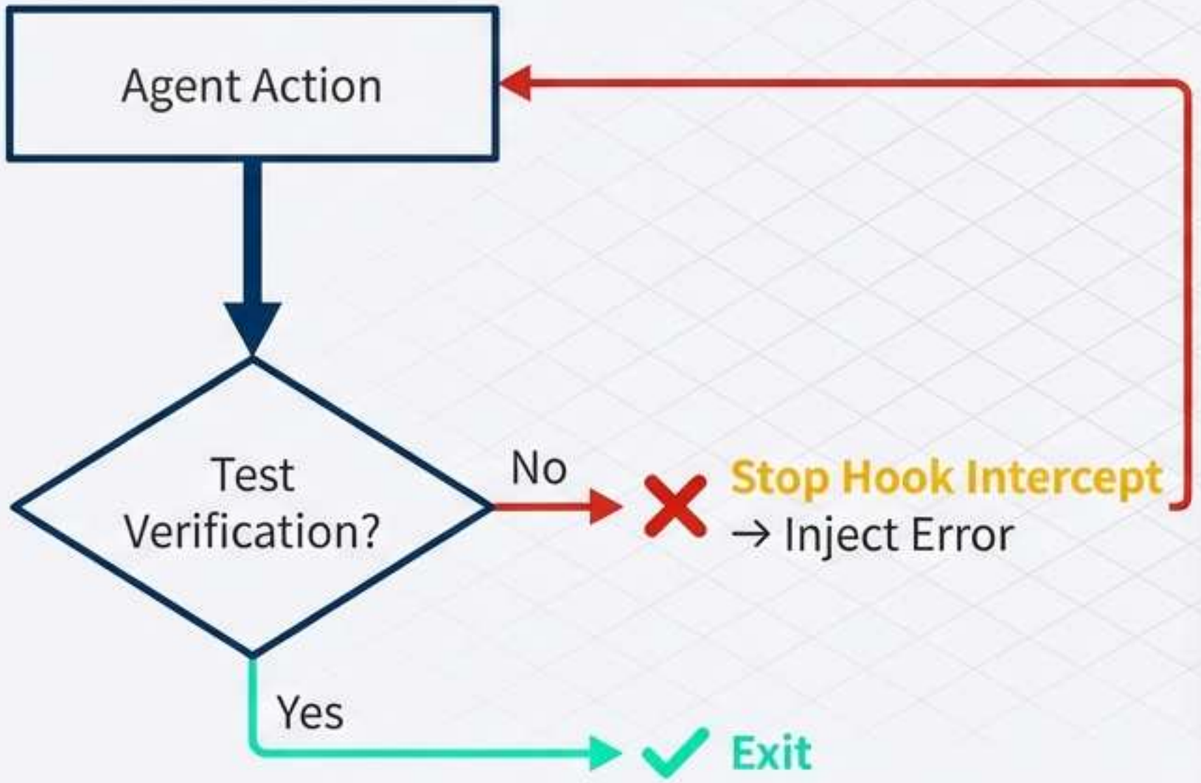
AvR：递归式思维链（Alignment via Refinement）

Unlocking System 2 Thinking via Self-Correction

Training Objective: **Maximize the Magnitude of Improvement**, not just the final quality.

Ralph Loop：自主迭代的韧性

Keep Working Until Objectively Done



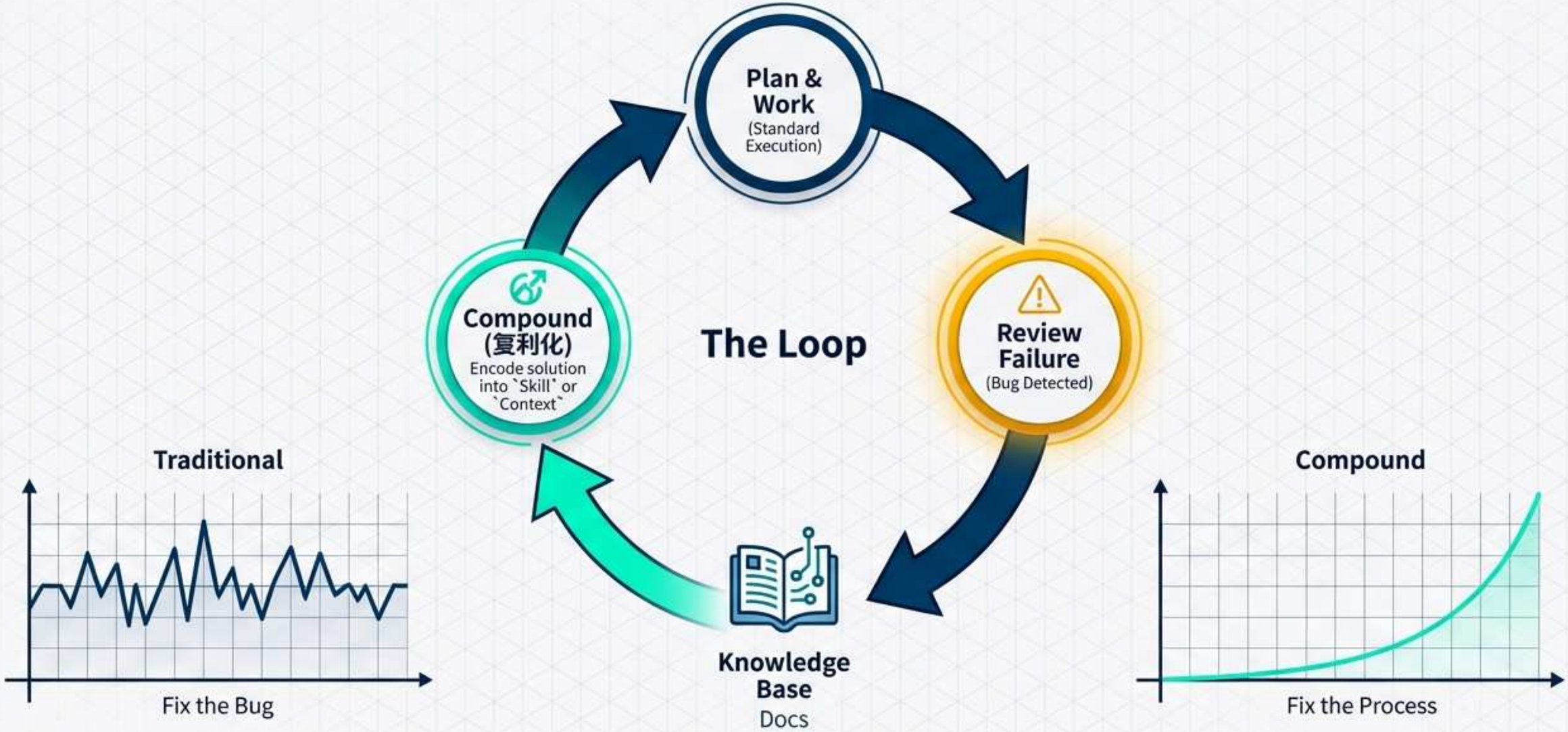

```python
def ralph_loop(task):
    while not verification_passed():
        result = agent.execute()

        # Prevent premature exit
        if agent.wants_to_stop():

            if not verify(result):
                inject_feedback('Tests failed. Continue.')
                continue
            else:
                break
```

复合式工程（Compound Engineering）

Turning Execution Errors into Exponential Assets

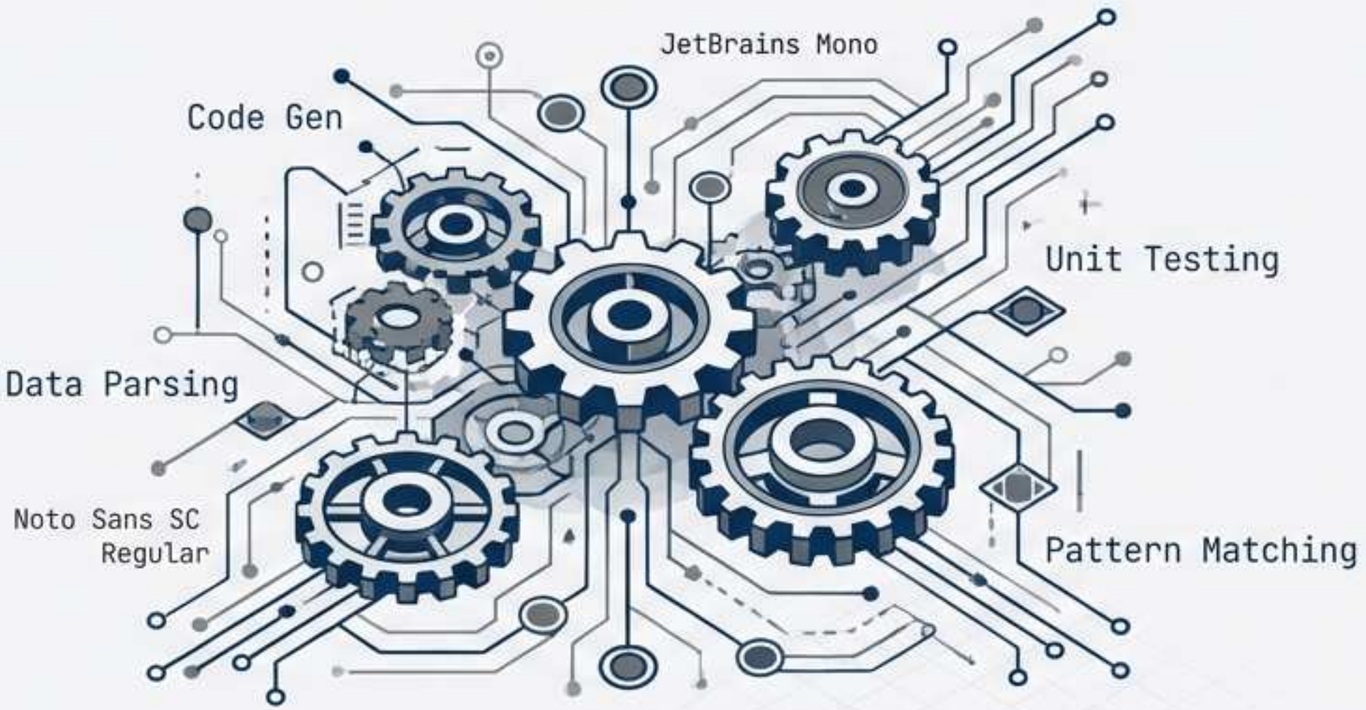


Subagent 架构：上下文隔离与并发

Scaling Complexity via Orchestration

人类角色的再中心化（Re-centering the Human）

From Operator to Architect

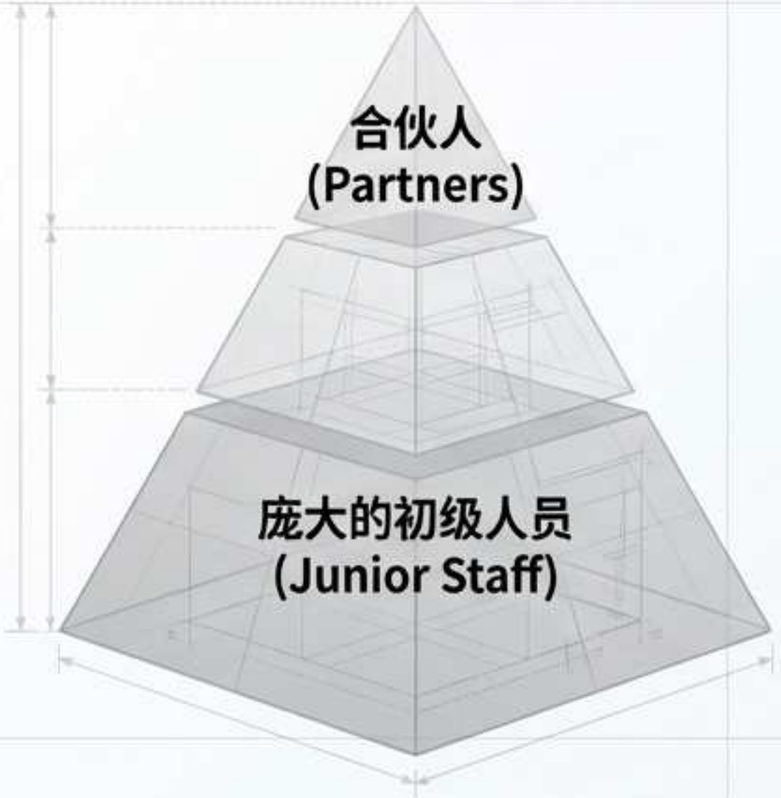


Execution & Verification (执行)

1. **Constitution & Spec**
   The "Why"
2. **Ethical Judgment**
   The "Should"
3. **Cross-Domain Synthesis**
   The "New"

Human intends; Machine actualizes.
（人类定义意图，机器实现现实）

从金字塔到方尖碑：咨询与工程团队的形态演变



结构转型：底层数据清洗与样板代码已被 AI 接管。

人机协作：专家不再指导初级员工，而是监督智能体交付。

核心收益：团队更精干，聚焦战略与架构决策。

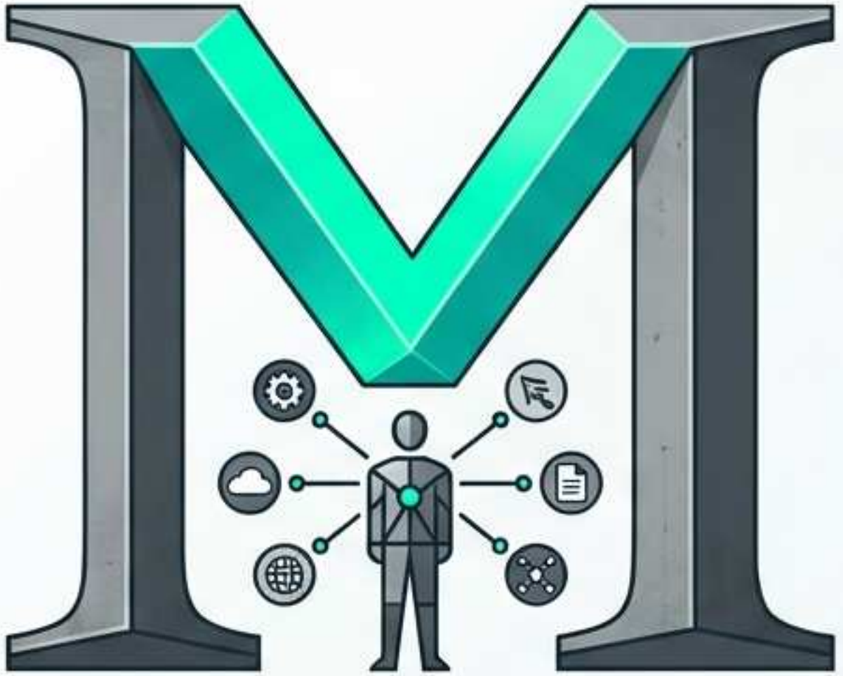
传统金字塔（The Pyramid）

- 基于工时的计费模式

方尖碑模式（The Obelisk）

- 基于成果的交付模式

人机协同下的新角色：M 型与 T 型人才



M 型监督者（The Orchestrator）

Noto Serif SC
跨领域统筹的通才

- 理解 AI 边界
- 拆解业务目标为 Agent 工作流
- 监督混合团队产出

T 型专家 (The Specialist)

Noto Serif SC
深耕核心领域的专家

- 守住质量底线
- 处理 AI 边缘失效 (Edge Cases)
- 重构复杂流程与深度判断

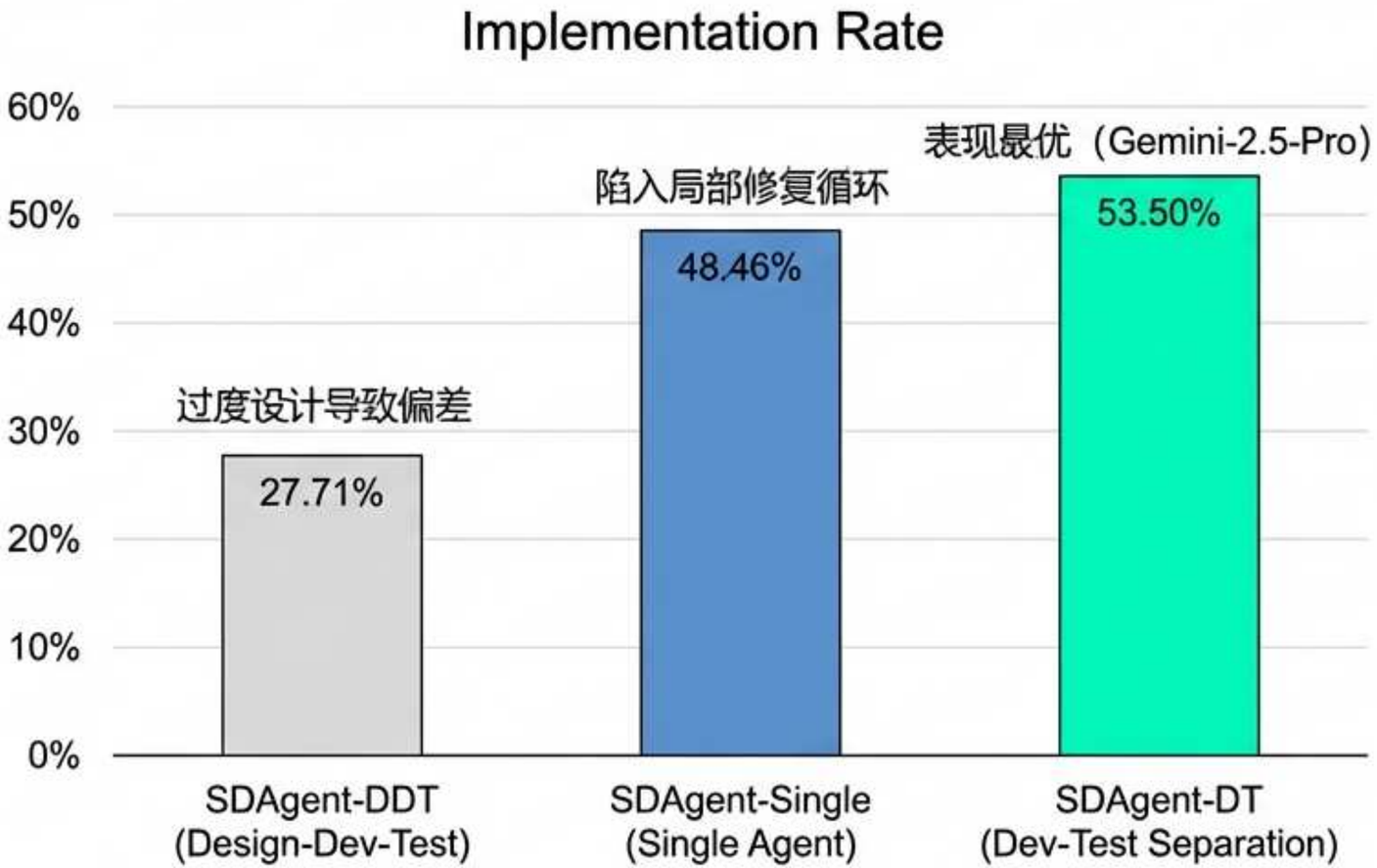
结论：AI 增强了一线人员，为人际互动与高阶判断释放了关键时间。

安全警示：Skills 架构的隐性攻击面

- **强制沙箱**：Agent 必须在隔离环境（Sandbox）中运行。
- **静态审计**：扫描脚本中的危险函数。
- **源头验证**：仅使用官方认证渠道的 Skills。

实战效能：工作流设计决定成败

基于 E2EDevBench（50 个真实项目，2024-2025 Q1）



关键洞察：合理的任务拆解（Dev vs Test）比单纯堆砌智能体数量更有效。SDAgent-DT 模式避免了自我循环验证的盲区。

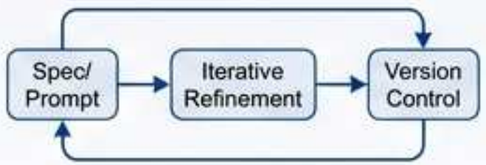
落地路线图：从实验到组织重构

Step 1：实验与画像匹配

为不同团队匹配工具 (Kiro/Cursor)。积累低风险实践。

Step 2：意图标准化

Spec as Code。将需求文档和 Prompt 纳入版本控制。



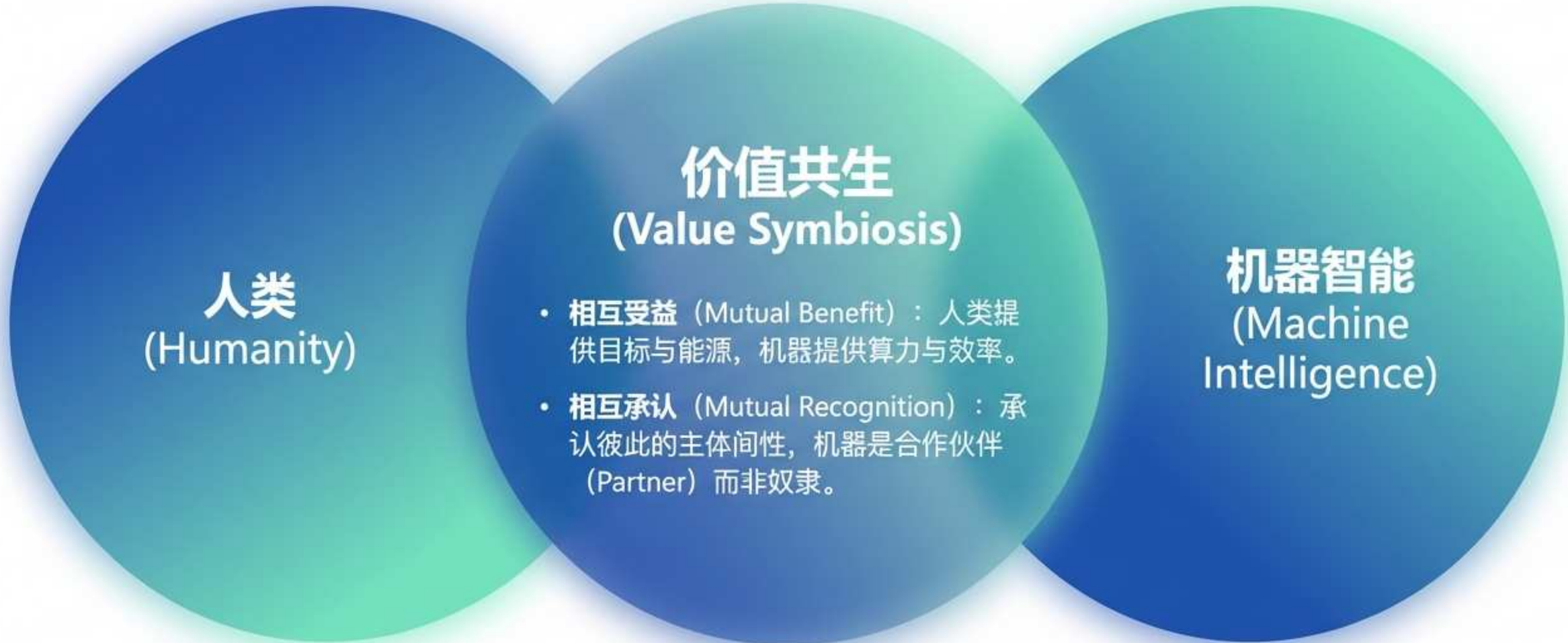
Step 3：集成与自动化

CI/CD 流水线集成 Agent。利用 Rules 强制执行编码规范。

Step 4：组织重构

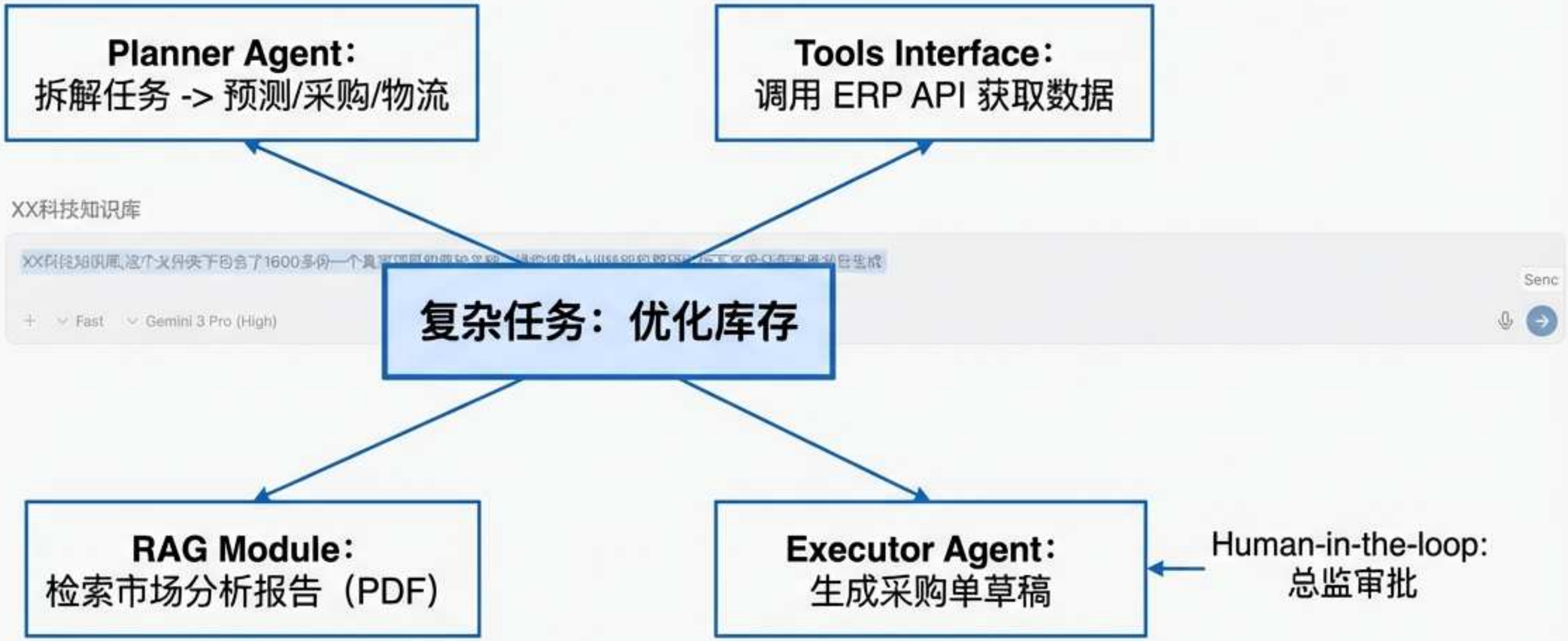
建立围绕"规范创作"的 A-Team。技能重心转向意图工程。

伦理新范式：从对齐（Alignment）到共生（Symbiosis）



超越单向的“价值对齐”，在共享世界中构建和谐共存的互利生态。

超越单一模型：复合 AI 系统架构（Compound AI Systems）



核心优势：模块化（Modularity）| 可控性（Controllability）| 容错性（Fault Tolerance）

Humanity within Tech

“真正的智能革命，不是让机器取代人，而是让机器成为人性的放大器。我们追求技术的极致效率，是为了释放人类在真、善、美上的无限潜能。”

—— 研发中心工程总监

从 Vibe Coding 转向复合式工程，是为了构建上下文中立、可靠的协作环境。

软件工程 2.0：Agent-First 工具链愿景

开发者职责演变：编写代码 -> 定义规范 + 引导智能

Q&A

感谢您的聆听（Thank you for listening）

Open for Questions & Discussion

深圳市网新新思软件有限公司