人智协同常见问题

从 Harness Engineering 到 OpenClaw 的工程实践

汇报人：金鹏

深圳市网新新思软件有限公司 AI 研究院

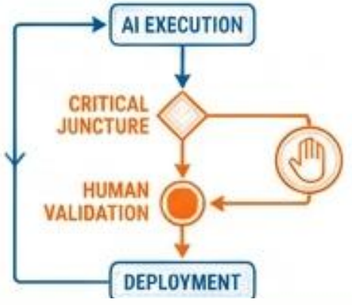
核心标尺：让人放心，把人放大

真正的协同并非取代人类，而是通过各自优势弥补短板，
将 AI 无缝集成到工作流中。

ACTION METRIC

行动标尺

真正的信任源于“人机回环(Human-in-the-loop)”。确保在关键的合并与发布决策点，人类始终拥有最终的“刹车权”。



CAREER FORESIGHT

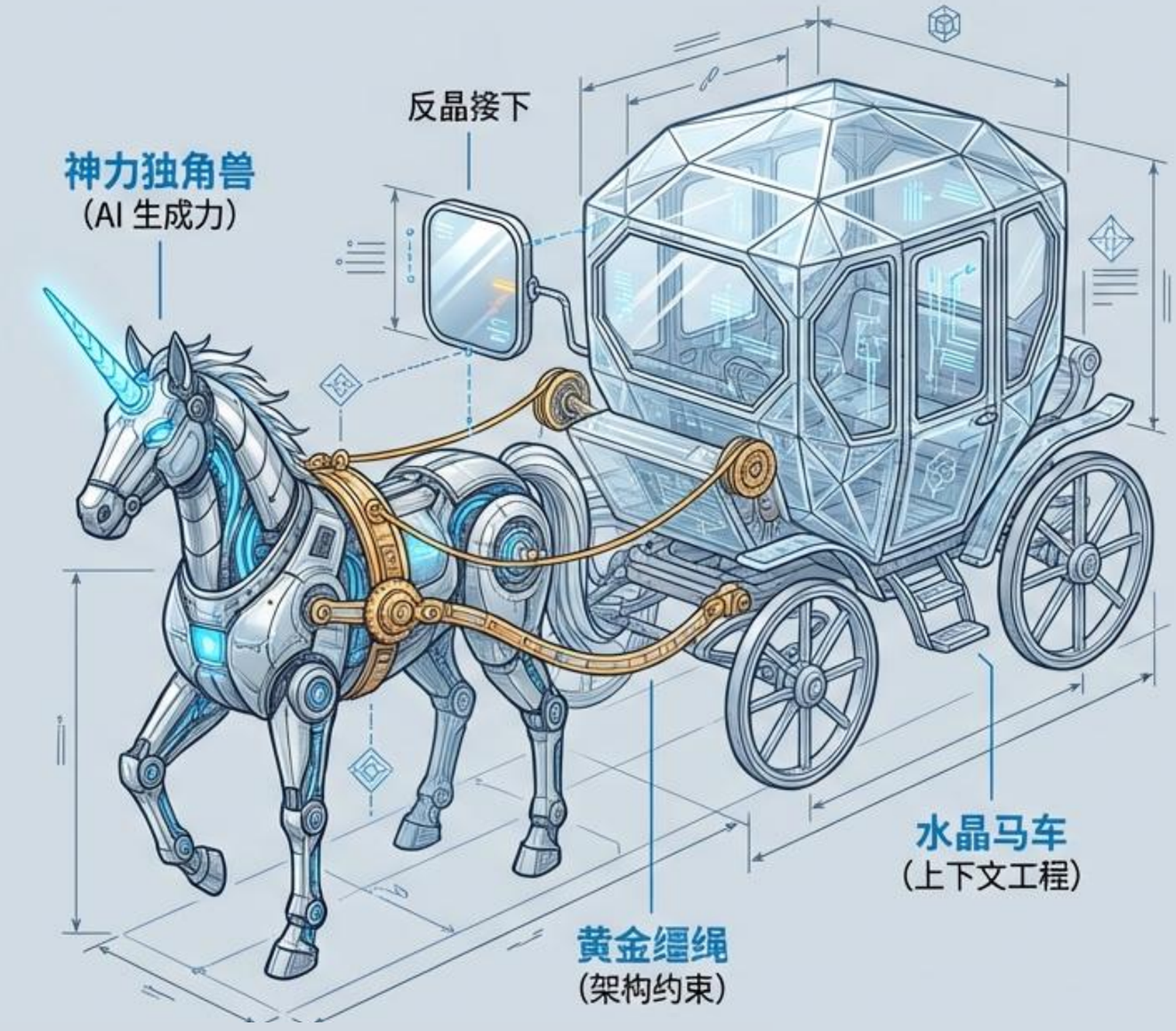
职业前瞻

工程师的核心价值正在发生根本性转移——从“亲手编写每一行代码”转向“设计系统”和“定义边界约束”。

COLLABORATION GOAL

协同目标

让人类掌舵，让智能体执行。



AI 生产力爆发的隐性代价与破局之道

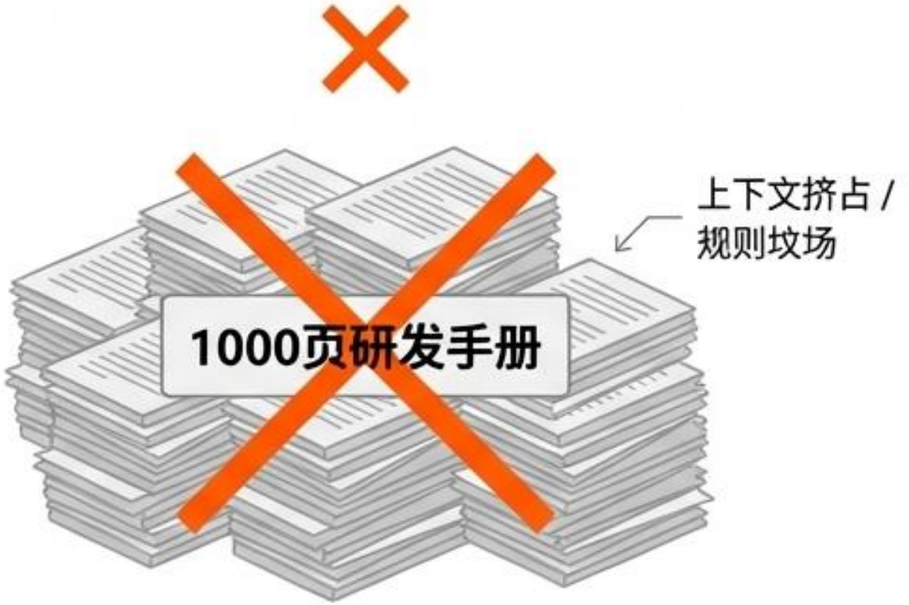
现状困境： 代码生成极快，但系统熵增飙升、坏味道被忠实复现、不可控错误放大。

核心隐喻：
不拔掉独角兽的角，而是打造黄金缰绳与水晶马车。

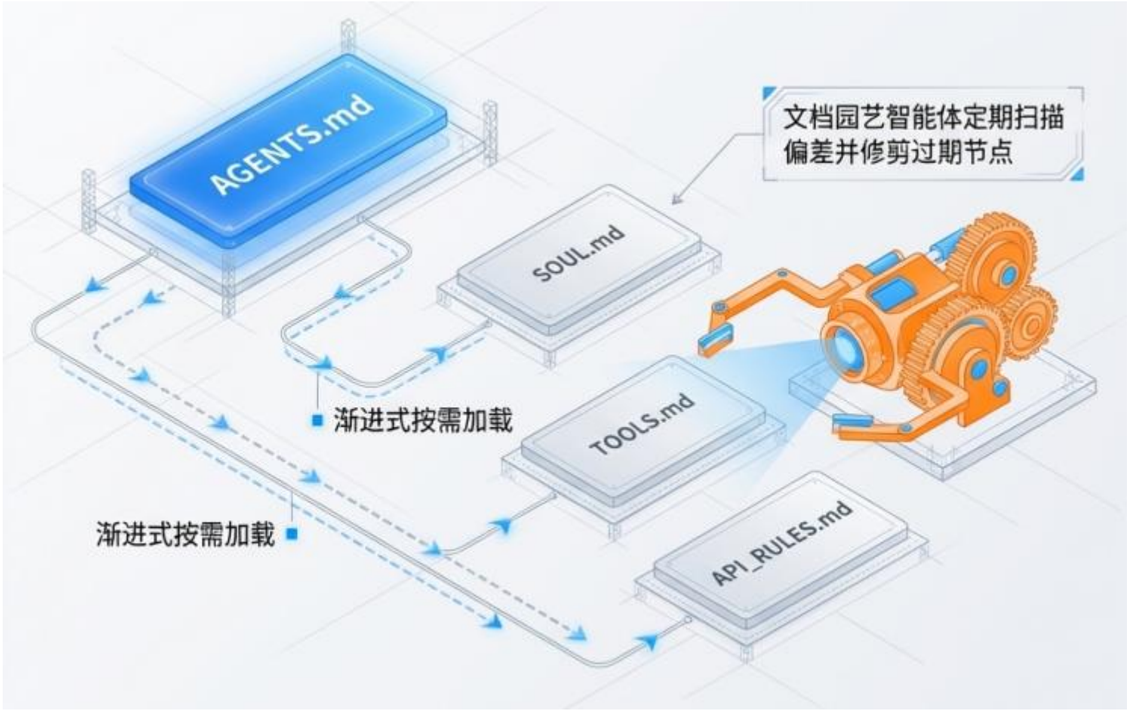
破局理念：
Harness Engineering（驭缰工程）——人类掌舵，智能体执行。

工程师的产出，从“编写代码”变成了“构建约束系统”。

问题 1：文档失效与“规则坟场”



智能体每天生成的百万行代码会让传统长篇手册迅速腐烂。陈旧、矛盾的规则成为了 Agent 故障的源头。



驭缰方案：地图而非手册。将 AGENTS.md 视为极简的入口目录，采用“渐进式披露（Progressive Disclosure）”。引导智能体读取深层约束。

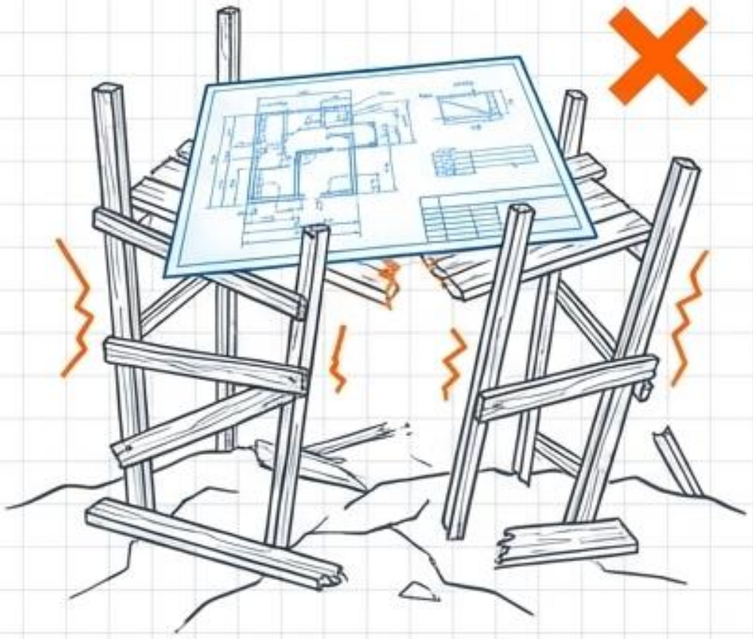
文档园艺（Doc-Gardening）

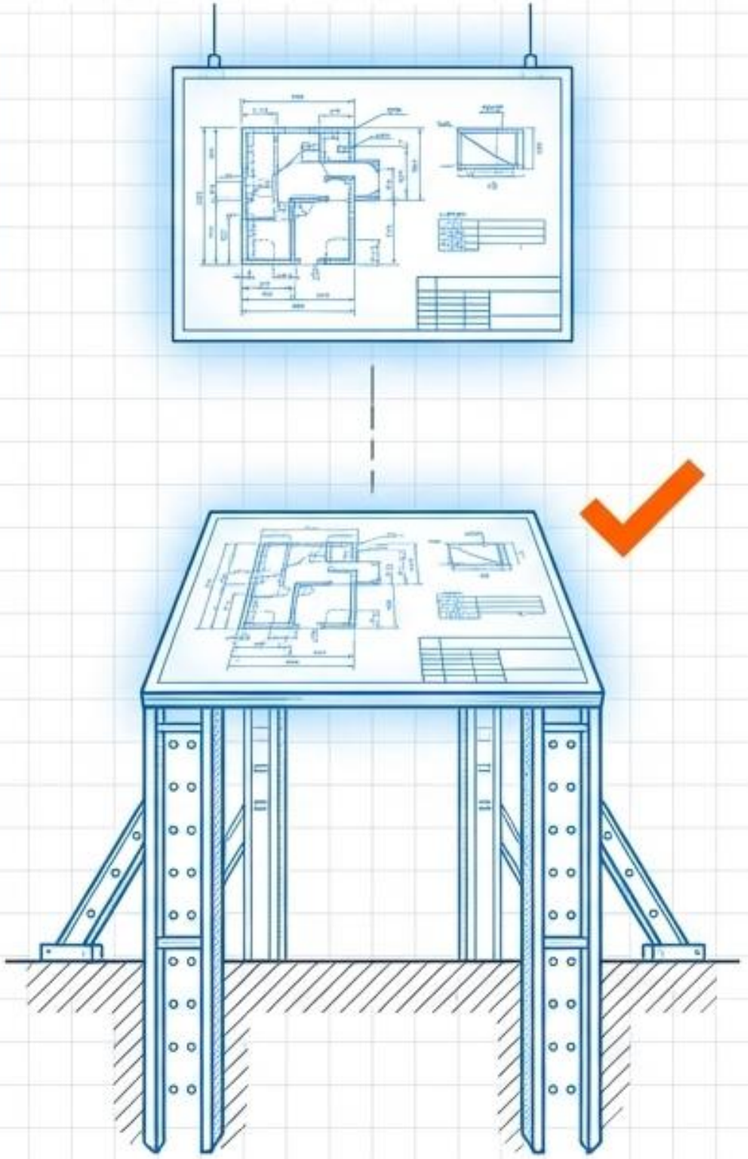
部署后台常驻智能体，定期扫描代码库，自动发起 PR 修复或删除不再反映真实代码行为的过时文档。

问题 2：环境脚手架不稳导致的错误

认知误区：当 Agent 犯错时，人类的第一反应往往是“盲目优化 Prompt”或“更换更强大的大模型”。

解决思路：环境胜过模型。为 Agent 提供明确的数据边界、自定义 linter 和实时验证回路。





68.3%↑

6.7%↑

实战案例证明：仅通过改变代码的编辑格式（采用结构化的 Hashline 锚点格式），底层模型一个参数未改，Grok Code Fast 1 的基准测试得分直接飙升。

拦截不可控错误：“坚固环境”永远胜过“精妙提示词”

第一层防御：提示词工程（Prompt Engineering）

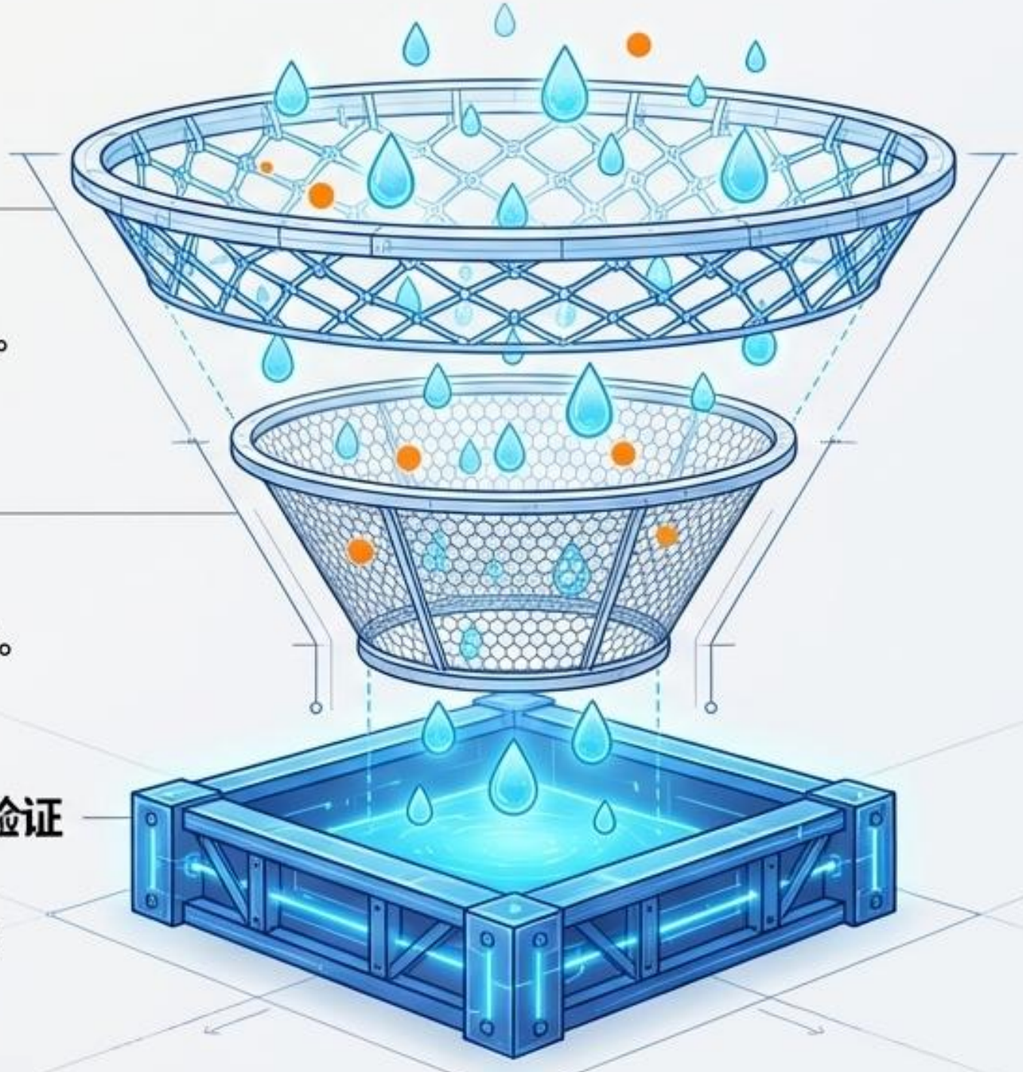
关注“说什么”，容易被遗忘或绕过边界。

第二层防御：基础大模型能力（Foundation Models）

存在偶发幻觉与统计学偏差的固有缺陷。

第三层兜底：环境脚手架与机械验证（Harness Engineering）

关注“在什么条件下运行”。增加工具拦截、明确类型边界、强制结构测试。

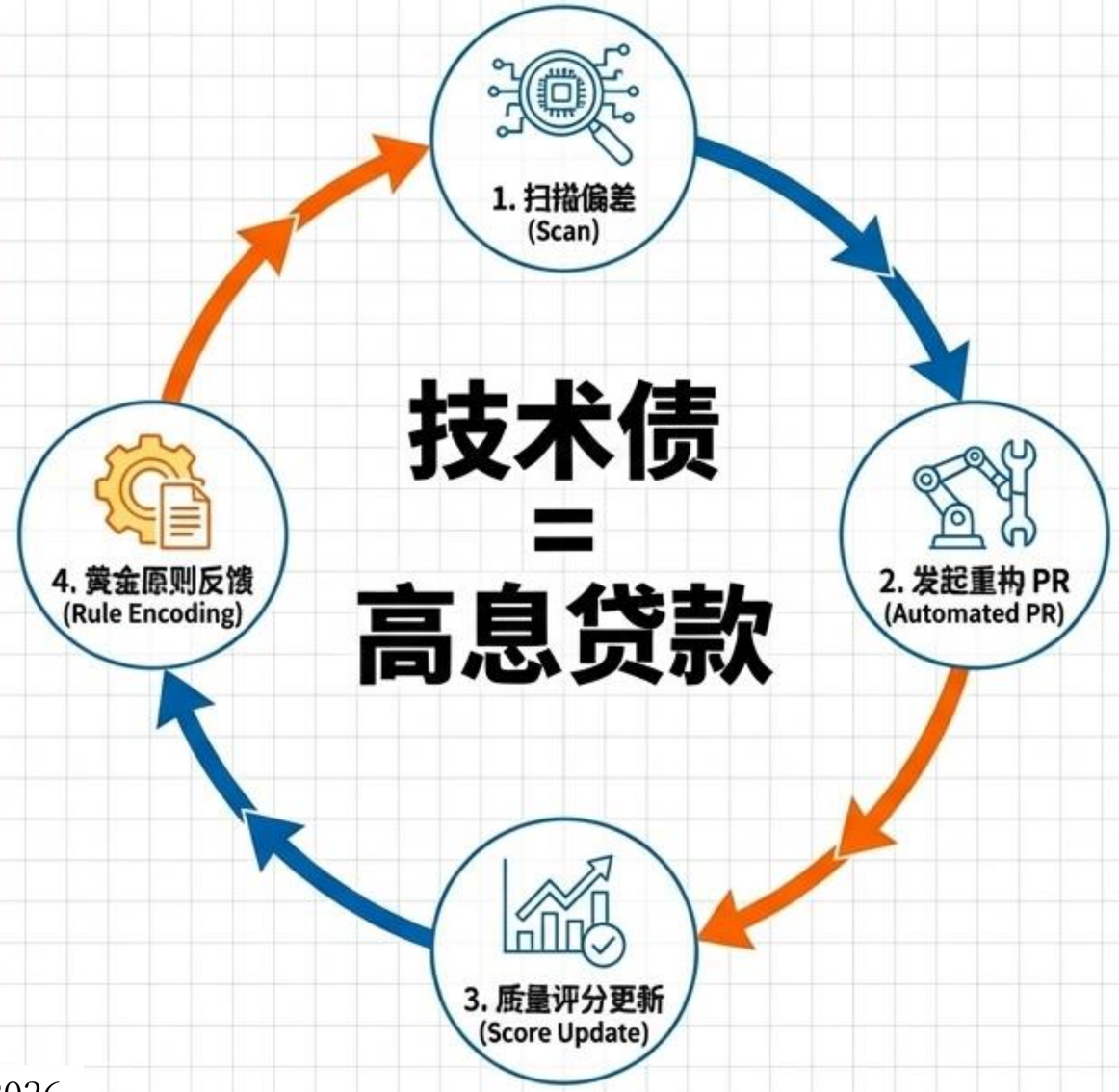


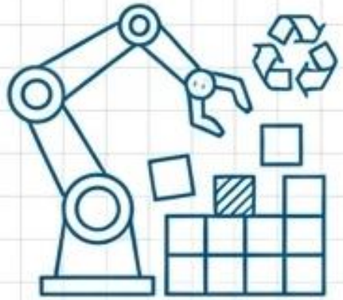
LangChain Agent 仅通过优化外部环境，基准得分即可从 **52.8%** 飙升至 **66.5%**。

问题 3：技术熵增与模式复现

风险挑战：

智能体极其“忠诚”。它们会完美地复现仓库中已存在的坏模式和历史技术债，导致系统的熵值随自动化的加速而急剧膨胀。





治理机制：垃圾回收（Garbage Collection）

将“黄金原则”直接编码进代码仓库。部署常驻的“清道夫”智能体，定期执行偏差扫描，发现不良模式后自动发起微型重构 PR，将系统熵值压制在可控范围内。

对抗极速技术熵增：将“垃圾回收”编码进代码仓库

步骤 1：识别代码坏味道
将系统黄金原则编码为
机械检查脚本

步骤 2：后台智能体启动
Garbage Collection Agent
利用夜间或周末空闲算力巡检

步骤 3：自动重构与验证
清理过时代码模式，进行小额
债务偿还并完成机械验证

步骤 4：发起合并与合流
生成高覆盖率的小额重构
PR，交由系统合流

总结：将人工每周 20% 的代码清理时间，转化为系统的持续、自动化偿还机制。

问题 4：吞吐量过载带来的协作瓶颈

现状倒逼：当团队通过 Agent 达到极高吞吐量时，人工逐行审查代码成为最大的物理瓶颈。

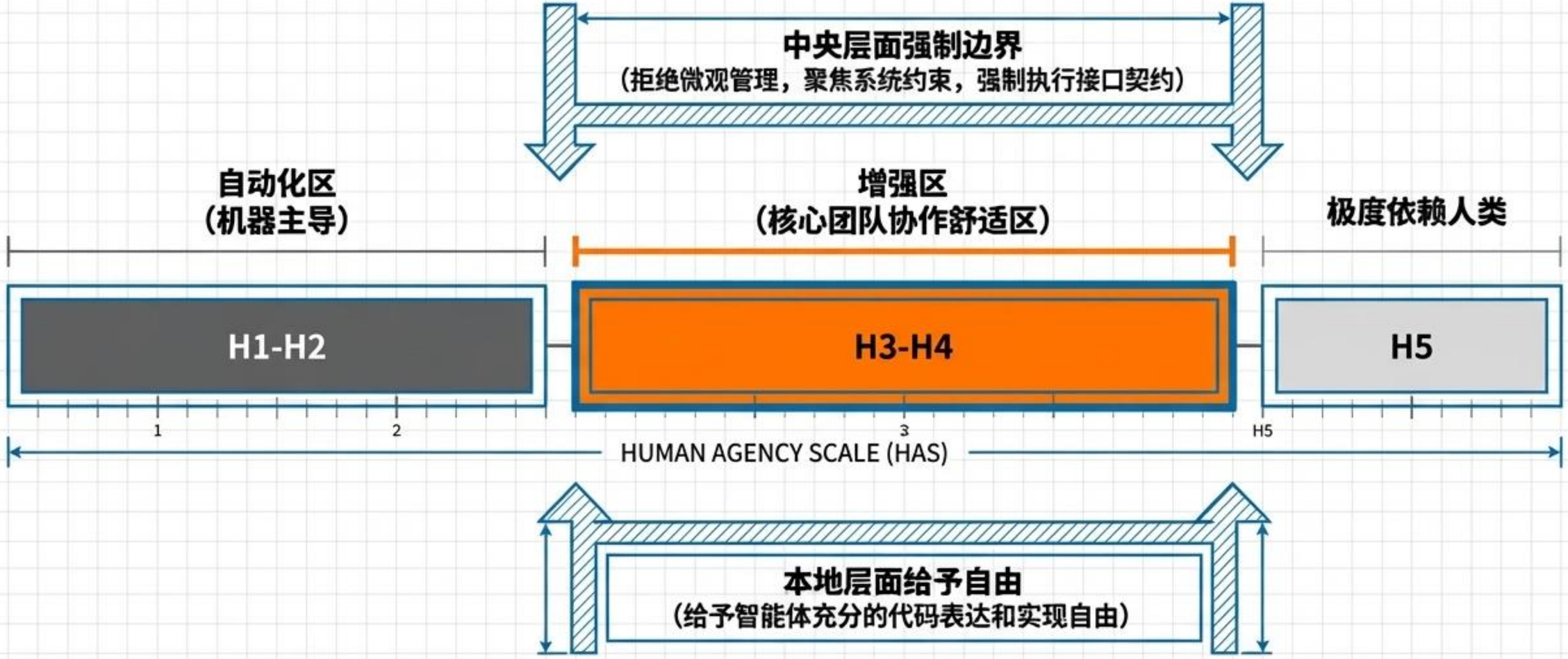
合并理念必须转变：接受短生命周期的 PR，测试偶发失败通过自动重跑解决。

绝大多数常规错误由扮演 Reviewer 角色的 Agent 自动驳回并要求修正。

仅在需要最终价值判断或架构转向时，才交由人类“刹车”。

问题 5：工程师角色的认知失调

人类自主性量表（Human Agency Scale）：并非所有任务都需要完全自动化。真正的工程红利存在于增强协作区。

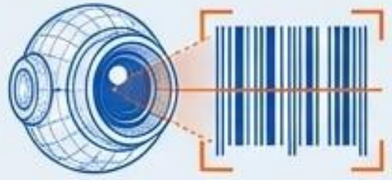


问题 6：可读性冲突与“无聊”技术栈

架构取舍：为了最大化 Agent 的推理与执行能力，我们必须放弃部分传统的“人类代码美学”。

人类工程师美学（Human Aesthetics）

- 追求“酷技术”
- 高度抽象的设计模式
- 复杂的语法糖
- 隐式的系统行为



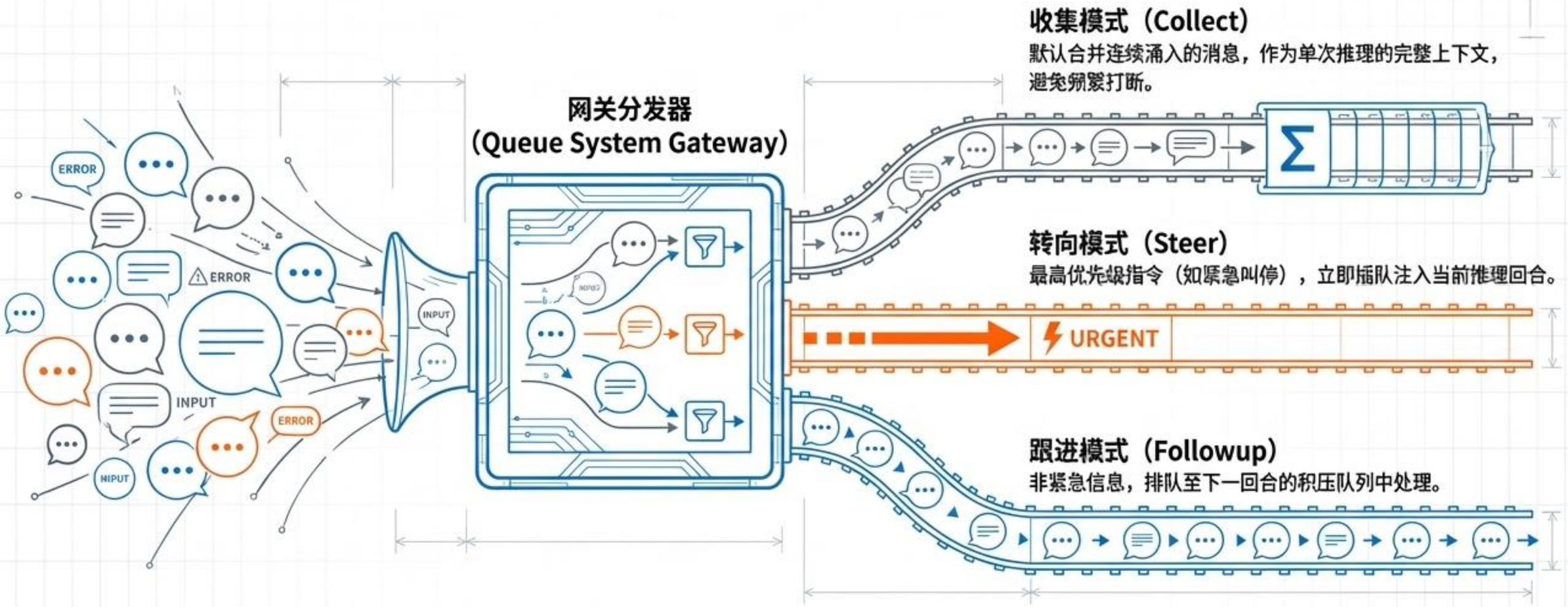
机器可读性（Agent Readability）

- 偏好“枯燥/稳定”的技术栈
- 覆盖广泛的公共训练集数据
- 显式的 API 契约
- 通过 Zod 等工具实现的绝对结构预测

只要代码是正确、可维护的，并且对未来的智能体运行清晰易读，即为达标！

问题 7：并发交互中的消息竞争

痛点：在高频协同中，多个指令和报错同时涌入会导致 Agent 状态极度混乱。
OpenClaw 破局方案：建立精密的底层状态队列系统，终结消息互斥与竞争。



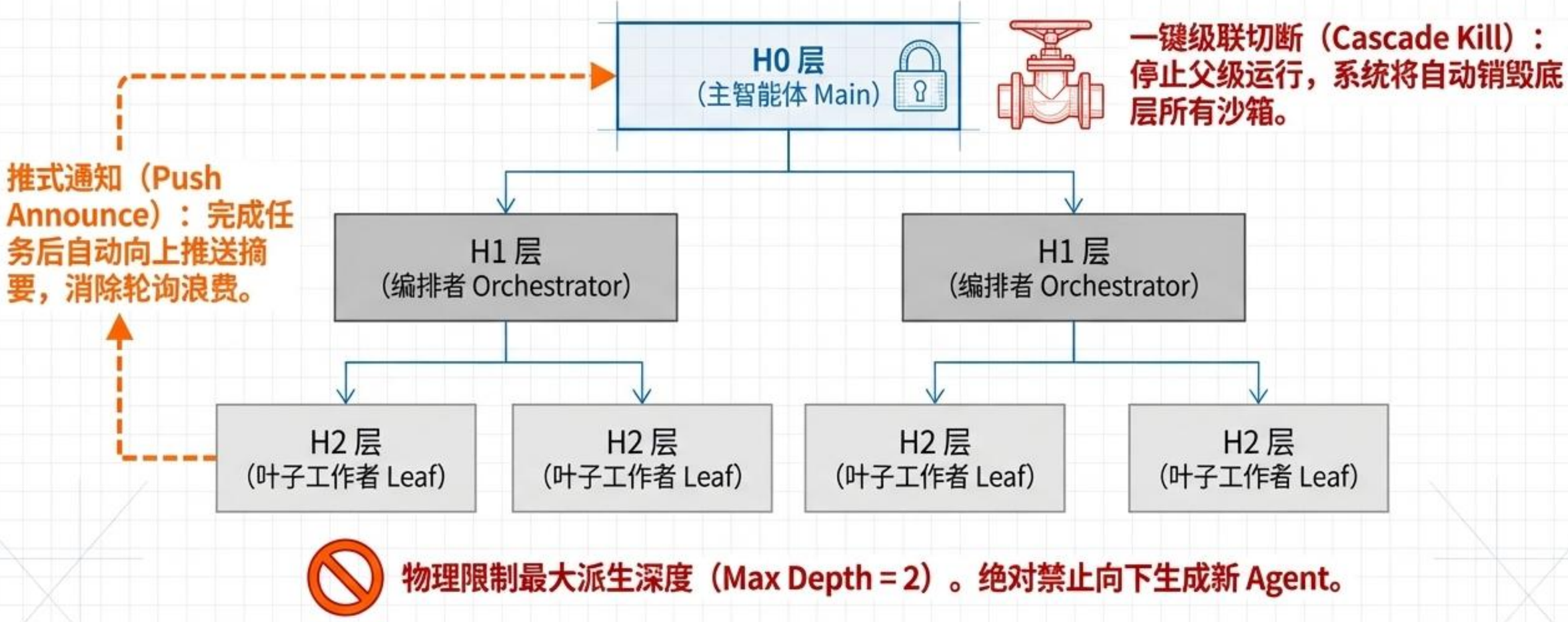
终结群组协同失控：“消息竞争”治理与并发状态管理

收集模式（Collection）：
动态聚合多用户后续回复，避免打断当前推理。

注入模式（Injection）：
系统级最高优先级，立即抢占并注入当前执行回合。

问题 8：嵌套智能体的资源失控

灾难预警：子智能体的无限派生会导致计算资源枯竭。OpenClaw 采用严格的级联控制机制保证资源绝对回收。



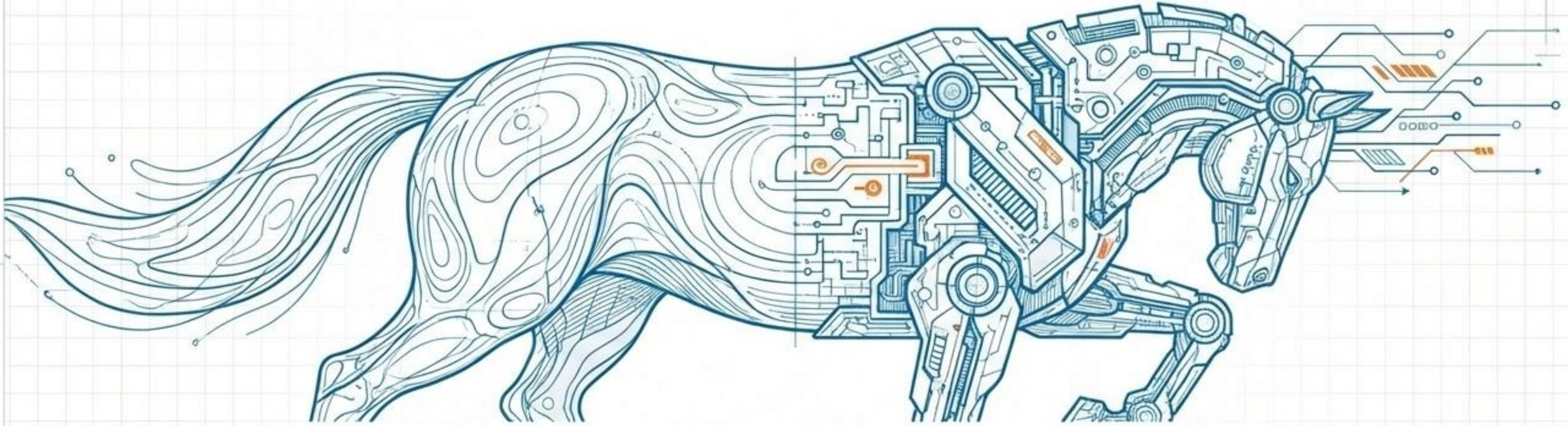
从今天起：我们的日常人智协同行动契约

Design a definitive checklist layout styled as a digital manifesto.

- **仓库即世界：**凡是不在代码仓库里的文档与架构约束，对 AI 均不存在。全团队拒绝隐性知识流转。
- **环境重于模型：**出现不可控系统错误时，第一优先级修补运行环境与规则边界，最后再调优 Prompt。
- **拥抱结构可预测：**容忍“非人类风格”的代码，只要逻辑清晰、结构高度可预测，优先满足 AI 的机器推理需求。
- **敬畏算力资源：**严守 SubAgent 嵌套派生的物理红线，时刻关注并发限制与系统级技术熵增。

总结：构建人机共生的新文明

各尽所能，各取所需。超越单纯的“自动化”，迈向深度的“系统增强”。



1. 提升智能密度

优化工程脚手架，让单位算力（瓦特）产出最大化的高质量智能。

2. 确立角色协议

从依赖个人工程师的“代码美德”，转向建立数字等价的严格角色协议（法官、审计员、执行者）。

3. 重塑基础设施

将 Harness Engineering 内化，建设下一代以 Agent 为中心的 AI Coding Infrastructure。

Tokens
词元

未来的数据中心不再是存储文件的仓库，而是生产 Token 的工厂。

在智能体优先（Agent-first）的世界中，每瓦特电力产出的高质量 Token 吞吐量，将是决定未来数字经济竞争力的唯一核心指标。

深圳市网新新思软件有限公司 AI 研究院（A.I. Research Institute）

算力经济时代的硬通货：词元（Token）

欢迎提问与探讨（Q & A）

感谢聆听。让我们共同探索智能体优先世界的无限可能。

深圳市网新新思软件有限公司 AI 研究院