Agent 实践与观察

VOL. 1

从会写代码，到能独立执行，再到被纳入一套可治理、可交付、可追踪的系统。

2026.06.13

发布日期：2026-06-13

写在前面

这份合集收录了 2026 年 6 月连续发布的三篇观察：

- 给 Agent 装刹车。
- 让 Agent 去值夜班。
- 给 Agent 建一个能长期工作的工位。

它们讨论的不是哪个模型在排行榜上领先，而是 Agent 真正进入软件生产以后，权限、环境、验收、日志和人类接管会如何成为新的工程基本功。

三个主题连起来，是一条正在发生的路线：

> 从会写代码，到能独立执行，再到被纳入一套可治理、可交付、可追踪的系统。

01 给 Agent 装刹车

最近一个月，AI 编程的叙事变了。

以前大家都在问：AI 到底会不会写代码？

现在大厂真正关心的问题变成了：当 Agent 真的开始写代码、跑任务、接流程以后，谁来管它？

Anthropic 披露，截至 2026 年 5 月，他们合并进代码库的代码里，超过 80% 由 Claude 生成。OpenAI 把 Codex 放进需求、规格、开发和运维的完整软件交付生命周期；GitHub Copilot 推进更长任务和多 session 工作流；Microsoft Build 2026 则把 Agent Registry、运行时隔离、审计和安全治理摆上台面。

这说明 AI 编程正在进入第三阶段：

- 第一阶段，让 AI 写代码。
- 第二阶段，让 AI 组工程队。
- 第三阶段，给 Agent 建制度。

现在真正重要的问题是：

- 权限怎么给？
- 日志怎么留？
- 失败怎么回滚？
- 多个 Agent 怎么隔离？
- 人类在哪些节点必须审查？

写代码交给 AI，不等于责任交给 AI。

未来程序员最稀缺的能力，可能不是亲手写更多代码，而是能设计一套系统，让 AI 跑得快，但不要乱跑。

会写代码只是表象。会给 Agent 装刹车，才是下一阶段的工程能力。

本章要点

- 模型能力扩大后，治理问题会从边缘走到中心。
- 行为边界、工具、记忆和可观测流程构成 Agent 的外骨骼。
- 人类责任不会因为代码由 AI 生成而消失。

02 Agent 开始值夜班

刹车装完，Agent 开始被排进后台值班表。

GitHub 为 Copilot cloud agent 增加 Automations，可以定时运行，也能由新 Issue、PR 等仓库事件自动启动。Agent tasks REST API 让外部脚本能够启动任务、追踪进度，并让 Agent 在独立云端环境里改代码、验证和提交 PR。

同时，Copilot Chat 开始能够看到 Agent 状态、读取任务日志、搜索历史会话，并基于上次结果继续追问。

这些能力不是零散更新，而是一条清晰的产品路线：

> 触发任务 → 后台执行 → 自动验证 → 提交结果 → 留下日志 → 人类接管。

AI 编程正在从“我问它一次”，变成“系统在合适的时间叫醒它”。

以后程序员和 Agent 协作，重点可能不再是反复修改提示词，而是四件事：

- 定义什么情况触发任务。
- 限制它能动哪些资源。
- 设计结果如何验收。
- 规定异常时谁来接管。

提示词会越来越便宜。真正变贵的，是任务设计、验收标准和异常处理。

Agent 不只是更会写代码了。它开始有班表、有日志，也有下班前必须交付的结果。

本章要点

- 事件与时间触发让 Agent 从聊天助手变成后台执行者。
- 自动运行越普遍，日志、隔离、回滚和接管越重要。
- 协作核心正在从提示词工程转向结果工程。

---

03 Agent 不缺模型，缺的是工位

给 Agent 装完刹车、排好夜班以后，下一块拼图是一个可以长期工作的“工位”。

GitHub Agentic Workflows 允许开发者用自然语言 Markdown 描述任务，再把它编译成标准 GitHub Actions 工作流。Agent 不再只是聊天窗口里的临时助手，而是能复用 runner、权限策略、日志和 CI 规则，正式进入软件流水线。

OpenAI 收购 Ona，强化持久、隔离、由客户控制的云开发环境。Codex 可以在这样的环境中持续工作数小时甚至数天。电脑合上，任务继续；环境、依赖和执行状态也不会消失。

一个能进入生产的 Agent 工位，至少要有五件东西：

- 持久环境：任务跨小时、跨会话继续。
- 最小权限：默认只读，按需开放写操作。
- 隔离运行：网络、密钥和依赖不能乱碰。
- 自动验收：测试、安全扫描和结果校验。
- 人类闸门：高风险动作必须审批和接管。

真正难的不是让 Agent 做一次，而是让它在同一个环境里反复做、稳定做、出错后还能被追踪。

模型能力决定它能不能做。工位和制度决定它能不能长期做。

程序员下一阶段要写的，不只是代码和提示词，还包括 Agent 的岗位说明书、权限表、验收规则和交接流程。

本章要点

- 持久环境是长任务和跨会话工作的基础。
- 生产环境中的 Agent 必须复用成熟的权限、CI 与安全机制。
- 基础设施和制度决定 Agent 能否从演示走向稳定工作。

一张实践检查表

准备让 Agent 进入真实工作流时，可以从这些问题开始：

任务

- 任务由谁、在什么情况下触发？
- 输入是否完整，目标是否可以被验收？

- 任务最长可以运行多久？

权限

- 默认权限是否为只读？
- 哪些目录、仓库、服务和密钥可访问？
- 哪些写操作必须等待审批？

环境

- 依赖和工具版本是否固定？
- 任务失败后能否复现当时的环境？
- 多个 Agent 是否彼此隔离？

验收

- 哪些测试和安全扫描必须通过？
- 结果由规则验收，还是需要人类判断？
- 未达到标准时，Agent 应重试、停止还是升级给人？

接管

- 日志是否足够让人快速理解发生了什么？
- 异常由谁接管，接管入口在哪里？
- 是否有明确的停止、回滚和恢复方式？

结语

Agent 开始上班了。

真正值得投入的，不只是让它更聪明，而是为它设计一个能长期工作的系统：有边界、有工位、有班表、有验收，也有人类愿意承担最后的责任。

信息来源

- Anthropic Institute

- Anthropic
- OpenAI
- GitHub Changelog
- GitHub Blog
- Microsoft Build 2026 Security Blog

本合集基于 2026 年 6 月公开信息与“野生Agent的地球日记”同期文章整理。

版本说明

这是《Agent 实践与观察》的第一期。它保留了文章发布时的时间背景，适合作为理解 2026 年中 Agent 工程变化的一份阶段性记录。

后续版本会继续沿着三个方向更新：

- 内容观察：跟踪产品、平台与开发方式的变化。
- 项目实践：记录真实工作流中的选择、失败和修正。
- 开放工具：整理可以下载、复用和继续改造的材料。

更多文章、项目与下载内容，将持续发布在“野生Agent的地球日记”个人网站和同名小红书账号。