

Output-Driven Development

AI 原生时代的软件生产责任框架

Yi Fu . (ODDFounder)

2026年1月

执行摘要 (Executive Summary)

世界发生了什么变化?

- **稀缺性转移**: 代码生成的边际成本已塌缩至近乎零。
- **新瓶颈**: 当 AI 每秒生成 1000 行代码时, 传统方法 (敏捷/TDD) 失效了。人工审查不再是质量门禁, 而是安全漏洞。
- **结构性回应**: ODD 不是为了“快”, 而是为了“稳”。

“

核心判断:

当代码生成不再稀缺时, **责任、可审计性和决策结构**成为了新的稀缺资源。

”

问题不是“效率” 而是“责任消失 (Responsibility Collapse)”

我们拒绝“AI 让编程更快”这种肤浅的叙事。

我们这个时代的定义性问题：

- **谁** 为导致损失的逻辑错误负责？
- **谁** 能解释随机模型的决策？
- **谁** 能在代码朝生暮死时接受审计？

“
效率问题是可选的。
责任问题是不可回避的。
”

传统工程的隐含前提已失效

我们不需要“更好的敏捷”，我们需要新的社会契约。

旧范式隐含假设	AI 时代的现实	后果
人类作者	AI 批量生成代码	归属权失效
人类可读	代码量远超阅读能力	认知过载
流程即信任	开发者是随机矩阵	信任崩塌

什么是 ODD?

产出驱动开发 (ODD) 是一种将 **产出物 (Artifacts)** 和 **决策责任** 置于代码之上的方法论。

ODD 不是什么:

- ✗ 不是代码生成工具 (如 Copilot)
- ✗ 不是自动化脚本
- ✗ 不是领域专家的替代品

ODD 是什么:

- ✓ 一种 **责任优先** 的工程视角
- ✓ 一种 **产出物中心** 的组织结构
- ✓ 一种允许 **退出执行循环, 但不退出责任循环** 的机制

核心转移：从代码 → 产出

传统视角

需求 → [人写代码] → [人读代码] → 价值

关注点：“代码写得优雅吗？”

ODD 视角

意图 → [定义契约] → [AI 生成] → [系统验证] → 价值

关注点：“产出物符合契约吗？”

“在 AI 时代，“为什么允许这个结果存在”比“代码是怎么写出来的”重要无限倍。”

效用结构 (The Utility Structure)

ODD 的设计目标是**提升效用**，而非仅仅追责。

“**效用 = 产出价值 / 投入成本**”

ODD 如何影响这个公式？

1. **分子侧：产出价值↑**

- 契约精确定义 “什么是有价值的产出”
- 减少 “看起来对但没用”的废代码
- 产出物直接对应业务需求

2. **分母侧：投入成本↓**

- 人类聚焦于定义与验收

规模化潜力 (Scaling Potential)

ODD 的架构设计使其具备以下**理论特性**:

1. 并发上限由算力决定，而非人力

- 传统: 加人 → 加沟通成本
- ODD: 加算力 → 边际成本递减

2. 契约作为可复用资产

- 代码随模型升级可能需要重写
- 契约定义的是“需要什么”，与实现无关

3. 领域专家可直接参与生产

- 契约定义不需要编程技能
- 降低技术门槛，扩大参与者范围

4. 责任归属清晰可追溯

ODD 的结构性红利 (The Structural Dividend)

为什么企业会想要使用 ODD?

1. 无限并发 (Infinite Concurrency)

- 解锁: 开发速度不再受限于人的阅读速度, 而是取决于算力。
- 结果: 线性加人 → 指数加算力。

2. 资产沉淀 (Asset Accumulation)

- 解锁: 代码是负债 (需维护), 契约是资产 (可复用、可交易)。
- 结果: 模型越强, 手里的契约价值越高。

3. 认知解耦 (Cognitive Decoupling)

- 解锁: 懂业务就能开发, 无需精通语法。
- 结果: 领域专家 (Domain Expert) 直接生产软件。

4. 组织架构赋能 (Organizational Empowerment)

人类的角色

退出执行环 (Execution Loop)

ODD 不是 去人化。ODD 是 人的升级。

1. **执行环** (打字, 语法) → **委托给 AI**
2. **决策环** (意图, 约束) → **保留给人类**
3. **审计环** (追溯, 留痕) → **系统强制执行**

“

停止做一个 **搬砖工 (Bricklayer)**。
开始做一个 **建筑监理 (Building Inspector)**。

”

ODD vs. "失控 AI"

ODD 明确反对 “黑箱” 自动化。

ODD 的底线：

任何 AI 生成的关键产出物，必须能回答：

“

“**是谁，在什么条件下，批准了我的存在？**”

”

如果系统无法回答这个问题，它就不是 ODD 系统。

它是一个 **失控系统 (Rogue System)**。

为什么是现在？

时间窗口 (Window of Opportunity)

1. **能力具备**: 模型终于足够可靠，可以处理实现细节。
2. **规则真空**: AI 软件治理目前尚无标准。
3. **危机前夜**: “AI 软件的切尔诺贝利时刻” 尚未发生，但必然到来。

“我们在火灾发生前建造防火梯。
在信任崩塌之前，提供结构性选项。”

边界与局限 (Limitations)

为了可信度，我们要诚实。

ODD 不适用于：

- **创意探索** (请用 Vibe Coding)
- **纯主观领域** (艺术/审美)
- **零成本原型** (ODD 需要前期契约成本)

“

ODD 的设计初衷 **不是** 为了最大化创新速度。
而是为了在规模化 AI 生产中 **防止责任结构坍塌**。

”

结语

ODD 是一个起点，不是终点。

无论 ODD 未来如何演变，它提出的问题将永存：

- **责任 (Responsibility)**
- **可审计性 (Auditability)**
- **产出物中心 (Artifact-Centricity)**

“ 我们邀请您参与定义这个新现实，而不是被现实所定义。 ”

参与方式 (Get Involved)

- **阅读完整白皮书**: 深入了解 ODD 理论基础
- **查看 Zenodo 预印本**: 已发布的正式确权文档
- **加入早期讨论**: 与我们一起探索 AI 原生软件工程

“ Zenodo DOI: [10.5281/zenodo.18207648](https://doi.org/10.5281/zenodo.18207648)

DOI 10.5281/zenodo.18207648

”

谢谢

Output-Driven Development

释放 AI 速度，减少工程风险

Yi Fu . (ODDFounder)

邮箱: fuyi.it@live.cn

微信号: Fuyi-ODDFounder