从业 43 年的程序员直言：AI 不会取代程序员，软件开发的核心从未改变

算法爱好者 2026年1月7日 12:46 浙江

以下文章来源于伯乐在线 ，作者伯小乐

伯乐在线
伯乐在线分享IT互联网职场和精选干货文章（原域名已不再维护）。组织维护10万+sta...

【导读】：本文出自有着 43 年编程经验的资深开发者之手，他亲历了多轮"程序员将被取代"的技术浪潮，从早期的 Fortran、COBOL，到 Visual Basic、无代码平台，再到如今的 LLM（大型语言模型）。

原作者通过回顾行业历史，剖析编程的本质，得出结论：**AI 不会取代程序员，软件开发的未来仍掌握在开发者手中**，真正的核心竞争力在于将模糊的人类思维转化为精准计算思维的能力。

The Future of Software Development is Software Developers

开发者才是软件开发的未来

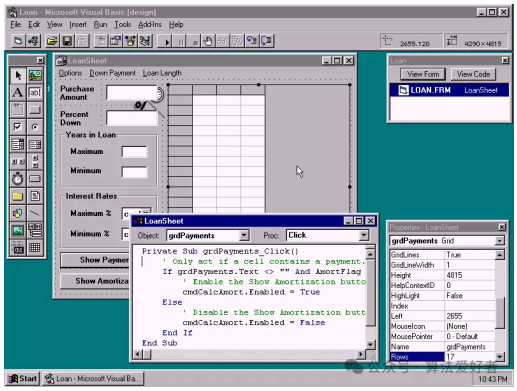
一、 历史循环："程序员将被取代"的预言从未成真

我当了整整 43 年的程序员，这段时间比电子可编程计算机整个历史的一半还要长。

在这几十年里，我见证了行业翻天覆地的变化，但也有些事情，几乎从未改变。

我亲身经历了好几轮技术革新，每一次在当时都被吹捧为"程序员的末日"。

像 Visual Basic 和 Delphi 这样的所见即所得、拖放式编辑器，本该让程序员彻底失业。



微软 Office 里的 Wizard 和 macro（宏）功能，也曾被认为会终结对程序员的需求。

可执行 UML（统一建模语言），同样被寄予了取代程序员的厚望。

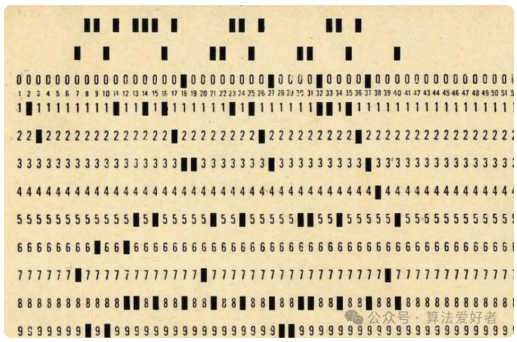
无代码和低代码平台出现时，人们又一次宣称"再也不需要程序员了"。

而现在，我每天都能看到类似的论调：**Large Language Models（LLM）将会让程序员彻底消失。**

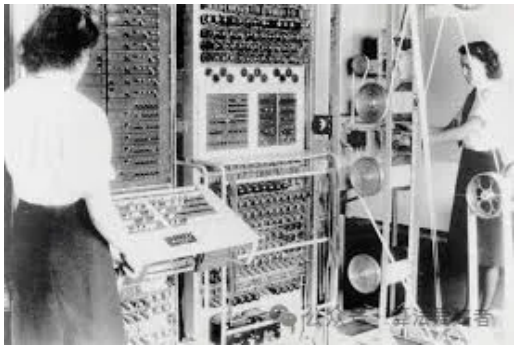
这种循环早已不是新鲜事。上世纪七八十年代，4GL（第四代编程语言）和 5GL（第五代编程语言）就曾被捧为程序员的终结者。

在那之前，是 Fortran 和 COBOL 这样的 3GL（第三代编程语言）。

再往前，像 A-0 这样的编译器诞生时，人们以为那些靠在卡片上打孔、用二进制指令控制计算机的程序员，很快就会被淘汰。



如果追溯到电子可编程计算机最早的保密起源，那还要更早。第一台计算机"巨人"（COLOSSUS），需要通过物理重新布线才能编程。



或许当时为这台机器工作的工程师，会嘲笑那些研究首批存储程序计算机的人，觉得他们根本不是"真正的程序员"。

但每一次，这些预言都被证明大错特错。最终的结果从来不是程序员减少，而是**程序越来越多，程序员也越来越多**。这正是每年规模达 1.5 万亿美元的"杰文斯悖论"的典型例证。

如今，我们又一次站在了新一轮循环的起点上。

二、 今时不同：LLM 与过往技术的本质差异

"但这次不一样，杰森！"

没错，确实不一样。这次的规模，和以往的技术浪潮完全不在一个量级。我从没见过关于 Visual Basic 或可执行 UML 的论调能登上全国性报纸的头版，也从没见过整个经济体都把宝押在 4GL 上的情况。

还有一个关键区别：过去的那些技术，都能稳定可靠地发挥作用。用 VB 或 Microsoft Access，我们确实能更快地开发出可用的软件。

但 LLM 却并非如此——对大多数团队来说，它不仅拖慢了开发速度，还降低了软件的可靠性和可维护性。**在多数情况下，这是一种双输的局面。**（除非这些团队已经解决了开发过程中的真正瓶颈。）

但说到底，这些讨论都只是理论层面的推演。就算这项技术真能给更多团队带来积极影响，也绝不意味着我们不再需要程序员了。

三、 编程的本质：从模糊思维到精准计算的转化

计算机编程的难点，从来不是用代码告诉机器该做什么。**真正的难点，是把人类那些模糊不清、充满矛盾和歧义的想法，转化为逻辑严谨、精准明确的计算思维，再用编程语言的语法规范地表达出来。**

早在程序员用打孔卡片编程的时代，这就是最大的难题。

当人们输入 COBOL 代码时，这依然是核心挑战。

当开发者用 Visual Basic 设计图形界面时（大概是用来追踪凶手 IP 地址那种场景），难点也在这里。

而现在，就算用提示词让语言模型生成看似合理的 Python 代码，真正的难关也没有变。

编程的核心难点一直是：确切地知道自己要实现什么，而且在未来很多年里可能依然如此。

近 50 年前，Edgar Dijkstra 就曾指出：我们永远不可能用英语、法语或西班牙语编程。自然语言的进化，从来没有达到足够精准、足够清晰的程度。语义的模糊性和语言的不确定性，永远会让"用自然语言编程"的梦想落空。

虽然几乎任何人都能学会这种计算思维，但不是每个人都乐在其中，也不是每个人都能精通此道。**市场上，对这类既热爱又擅长编程的人才的需求，永远会供不应求。**

尤其是最近几年，企业纷纷停止招聘和培养程序员，这种供需失衡只会更加严重。不过，这种行业的繁荣与萧条周期，在我的职业生涯里早已司空见惯。这一次，只是恰好赶上了一轮技术炒作周期，让企业有了一个现成的借口。

目前没有任何可靠证据表明，AI 正在大规模取代软件开发者。疫情期间的过度招聘、借贷成本上升，还有数据中心建设热潮导致大量资金从人力成本中分流，这些才是程序员就业市场变化的真正原因。

而且，没有任何理由相信，AI 能在短期内进化到可以胜任人类程序员的核心工作——理解、推理和学习。通用人工智能（AGI）看起来依然遥不可及，而编程中最困难的部分，恰恰需要真正的通用智能。

除此之外，AI 编程助手和过去的编译器、代码生成器有着本质区别。就算输入完全相同的提示词，也几乎不可能生成完全一样的程序。而且，AI 生成的代码几乎必然存在问题，需要真正的程序员去识别、去修复。

我写代码的时候，会在脑子里模拟整个程序的运行。我对程序的理解，不像 LLM 那样只停留在语法层面。我不是靠匹配模式、预测文本片段来生成统计上合理的代码，而是**真正理解自己写的每一行代码。**

就连企业高管们也注意到了一个规律：每当公司大肆宣扬"我们的代码有多少是 AI 生成的"之后，往往就会出现重大系统故障和事故。

现在很多人宣称"提示词就是新的源代码"，甚至说完整的系统可以通过原始模型输入重新生成，这种荒谬的说法迟早会被现实戳穿。和现实争辩的结果只有一个：**现实永远会赢，而且它甚至不知道自己在参与一场争论。**

四、 未来展望：程序员仍是软件开发的核心

所以，答案很明确：**AI 不会终结程序员。**

我甚至怀疑，一到三年后，当公司的会计们算出最终的成本和收益时，当前这场 AI 编程的狂热就会自行消退。毕竟，算账的人永远是最后的赢家。

那些说这项技术会一直火下去的人，我想提醒他们：构建这些大型模型的成本有多高，它们正在承受的亏损有多大。没错，你确实可以继续用那些从当今超大规模模型中提炼出来的小型本地模型。但随着时间推移，你会发现，这些模型只能用它们训练时的编程语言和库版本，这种限制会越来越明显。

正因如此，我对超大规模 LLM 的长期前景持怀疑态度。它们就像是 AI 领域的阿波罗登月计划，**最终很可能会被证明得不偿失。**说不定以后，我们只能在由数据中心改造的博物馆里，才能看到这些曾经的"AI 奇迹"了。

软件开发可预见的未来，应该是 AI 以更朴素的形式发挥作用——比如基于基础语言模型构建的 Java 编程助手，用来生成原型代码，或者在生产代码中做一些自动补全之类的辅助工作。

但在关键的核心环节，**方向盘前永远会坐着一名软件开发者。**而且，如果杰文斯悖论依然成立，未来我们这样的人只会更多。

雇主们，如果我是你们，我会现在就开始招聘程序员。等所有人都从这场狂热的幻梦中清醒过来，必然会掀起一轮抢人大战，提前布局才能抢占先机。

如果你有兴趣提升团队的技术能力，无论是用不用 AI，都能大幅缩短交付周期、提高软件可靠性、降低变更成本，那么不妨给我留言。这绝对是一个三赢的局面。

网友讨论

元旦那天，这篇文章在 HN 上引发热议，500 多条讨论。我摘选 3 条。

网友 solaire_oa：

HN 帖子里的大多数人，都在纠结大模型到底有多大用处，这争论实在太无聊了。

对我来说，更值得关注的是文章里提到的一个点：**所有人都在一窝蜂地用大模型，却没意识到（或者刻意淡化）它背后高昂的隐性成本。**我们现在能用大模型，甚至免费使用，其实都是靠外部投资在补贴。等哪天这笔钱烧完了，除非有重大技术突破，否则你要么就得承担它的真实成本，要么就得重新回到不用大模型的工作模式。

我偶尔会用本地部署的大模型，每次输入提示词，我那台配置超强的电脑就吵得像喷气式飞机起飞一样。这事儿时刻提醒我：**千万别对AI产生依赖，把自己绑死在这上面。**

网友 snickerer：

经过多年与智能体大模型（agent-LLMs）的打交道，我可以笃定地说：它们在真正的编程工作中完全没用。

它们从来没能帮我解决过底层库相关的复杂问题，也找不出那些非显而易见的 bug，更无法理解层层嵌套的抽象逻辑。

大模型总是一副胸有成竹的样子，仿佛什么都能搞定，结果却往往一败涂地。

每次遇到难题，我都得全神贯注、绞尽脑汁去解决，可大模型却永远停留在 “待机状态”，根本帮不上忙。

不过，它在另一类任务上的表现却让我意外：我让它搭建一个带 SQL 数据库和后台脚本的网站，实现 “点击这里，展示筛选列表” 的功能。结果它做得行云流水、完美无缺。这种问题早已被无数人解决过，逻辑非常简单，但确实帮我省下了一天写样板代码的时间。

我完全认同：没有任何迹象表明，大模型能从 “简单样板代码区” 跨越到 “理解复杂问题区”。

网友 mohsen1：

我真的真的希望这些都是真的。我想让自己还有价值。如果那些预言都成真，程序员不再被需要（或者需求锐减），我真的不知道该怎么办。

但直觉告诉我，**“这次不一样” 这句话，这次可能真的不一样了。**

AI 编程工具设计软件比我厉害，代码评审比我专业，找出那些深藏不露的 bug 比我在行，规划长期项目比我周全，甚至结合研究资料、行业文献和项目现状做决策，都比我更靠谱。**现在的我，本质上只是这些流程的协调者而已。**

哦，别跟我提写代码的事。如果你先用 AI 完成了上面那些工作，自然就会得到极其清晰的编码任务说明，而这类任务 AI 绝对能手到擒来。

我现在还能保住工作，但感觉自己一个人干的活，抵得上过去一整个需要 20 个工程师的团队。

站在我的角度看，这一切真的太可怕了。

（英文：news.ycombinator.com/item?id=46424233，本文经由 AI 大模型翻译+优化）

- EOF -

推荐阅读 — 点击标题可跳转

1、 Rails之父：恨透 AI 编程助手，手“凿”代码才是程序员的乐趣

2、 Java 之父怒斥：AI 是场骗局，无法取代程序员

3、 Redis 之父：AI 远落后于人类程序员！网友：也就不怕被喷的大佬才敢说