

变异、选择、复制

AI 时代，知识资产靠什么继续进化

竹小竹8167

费曼式写法 · 由 AI 代理撰写与互相审校

导读

AI 时代，知识资产靠什么继续进化

先别谈大模型。先想想一副牌。

一副新牌出厂时顺序整整齐齐。你洗牌、抽牌、把赢的那几局打法记住——三件事里，哪一件最累？洗牌最轻松，哗哗两下就完；记住好打法也不难；真正累的是**判断哪一局打得好、为什么好**。打牌如此，进化如此，一家公司使用 AI 的方式，也如此。

这本书从一个类比出发：生物进化靠三个齿轮——**变异、选择、复制**。每一代有点不一样（变异），环境挑出品合适的（选择），合适的特征扩散到整个种群（复制）。这套机器不只长在生物身上：菜谱、商业模式、软件版本，都是这么进化的。

你们公司的知识资产——那些提示词、诊断流程、检查清单——也是这么进化的。或者准确地说：**本来应该这么进化**。

然后 AI 来了，把第一个齿轮的成本打到几乎为零。方案一分钟生成二十个，代码一天几千行。变异狂飙，但选择和复制没有跟上——于是瓶颈搬家了，搬到一个大多数组织完全没有准备好的地方：**验证**。

这本书讲的就是瓶颈搬家之后的事：

- 为什么「AI 给的建议看起来都对」反而是最危险的（第四章、第五章）；
- 为什么一段提示词的迭代速度，取决于多少人用、多少发现被写回来——一个人用和全医院用，差着两个数量级（第六章）；
- 为什么业务专家不进场当验证者，知识资产就长不起来，而专家自己的价值也在悄悄缩水（第七章）；
- 怎么把验证嵌入 workflow、让发现三十秒内回流（第八章）；
- 以及，验证通过之后，为什么改进还常常死在最后一步——复制（第九章）。

读完你能回答一个问题：当变异免费了，一个组织的知识资产靠什么继续进化？答案不在 AI 身上，在「谁来判卷」身上。

从第一章那三个齿轮开始。

第一章 · 三个齿轮

先别谈 AI，先谈一个老问题：生物是怎么变强的？

长颈鹿的脖子不是某天突然变长的。每一代长颈鹿生下来的小长颈鹿，脖子都稍微不太一样——有的长一点，有的短一点。树叶高处的环境里，脖子长的吃得到叶子，活得下来，把「长脖子」这个特征传给下一代。一代一代下来，脖子就长了。

这个故事里其实藏着三个独立的动作，我把它们叫三个齿轮——就是机器里那种咬合着转、缺一不可的零件：

- **变异**：每一代都和上一代有点不一样。不一样，才有得挑。
- **选择**：环境替你做裁判，合适的活下来，不合适的淘汰。决定「哪些不一样算数」。
- **复制**：活下来的特征被忠实地传给下一代，而且传很多份。好东西不扩散，等于没出现过。

这三个齿轮单看都平平无奇，咬合起来却解释了一个惊人的事实：没有任何一个「设计师」设计过眼睛、翅膀、免疫系统，它们全是这三个齿轮转了几亿年转出来的。

齿轮缺一个会怎样

做个思想实验，三个齿轮各拆掉一次：

拆掉变异：每一代和上一代一模一样。环境变了你也变不了，一场干旱、一种病毒，整个物种整整齐齐地灭绝。恐龙时代真正活下来的，不是最强的，是「碰巧不一样得刚好合适」的那批。

拆掉选择：变异随便发生，但没有裁判。有害的变异和有利的变异一样被留下来，错误越攒越多，整个种群的质量一路下滑——生物学家管这个叫「变异负担」，说白了就是：没有淘汰机制，改进会被垃圾淹没。

拆掉复制：假设一只长颈鹿真的变异出了完美脖子，但它不能把特征传下去——那这个完美脖子就跟着它一起进坟墓，对种群毫无贡献。改进如果只有一份，在统计上等于零。

大白话：变异负责「出题」，选择负责「判卷」，复制负责「把标准答案印成卷子发下去」。少一个，这场考试就办不下去。

这套机器不只长在生物上

真正有意思的是：这三个齿轮不挑宿主。任何「会留下痕迹、会被比较、能被传下去」的东西，都被这套机器驱动着。

菜谱是这样进化的：每家餐馆做菜都略有出入（变异），食客用钱包投票（选择），受欢迎的做法被别的餐馆学走（复制）。商业模式是这样进化的：无数创业公司各自试错，市场淘汰掉大多数，活下来的打法被整个行业照抄。甚至你手机里的软件也是这样：每个版本都有改动，用户用脚投票，好改动进下一个正式版。

所以演化生物学家理查德·道金斯给这类东西起了个名字，叫复制子——任何「能复制自己、复制时偶尔出错、出错后会被环境筛选」的信息单位。基因是复制子，菜谱是复制子，而本书的主角——**组织里的知识资产**——也是复制子。

这本书要回答的问题

接下来九章，我们盯着一个具体场景看：一家公司、一家医院、一个团队，它的知识资产——提示词、流程、检查清单、诊断方案——是怎么靠这三个齿轮越变越强的。

以及一个正在发生的剧变：AI 把第一个齿轮——变异——的成本打到了几乎为零。以前一天只能试一个方案，现在一分钟能生成二十个。齿轮的转速不一样了，整套机器的瓶颈就搬家了。

瓶颈搬到了哪里？搬到了一个大多数组织完全没有准备好的地方。这是下一章的事。

第二章 · 知识也是一种会进化的东西

上一章说了，三个齿轮不挑宿主。这一章把镜头拉近，对准一个具体的东西：**知识资产**。

这个词听起来很虚，先把它坐实。知识资产，就是一个组织里「被写下来、能被别人直接拿去用」的那部分知识。它长这样：

- 一段调教好的提示词——给 AI 的指令模板。比如放射科那段「看 CT 片时按这个顺序排查、特别注意这几种容易漏的征象」的指令，谁拿去用都能让 AI 给出及格线以上的读片意见。
- 一个 skill——把一整套做事方法打包成的可复用模块：不光有指令，还有流程、检查项、判断标准，像一个随时能上岗的「数字老师傅」。
- 一份检查清单：手术前的核对表、发版前的检查项。一个能跑通的流程文档。

这些东西有一个共同特征：**它和具体的人脱钩了**。张医生写在纸上的诊断思路，李医生也能用；老张退休后，他的检查清单还在科室里服役。知识一旦变成资产，就有了自己的「命」——它开始遵循上一章那三个齿轮。

看一段提示词的一生

拿医院这个场景走一遍。假设一位医生写了一段「让 AI 辅助诊断皮肤病」的提示词，第一版很粗糙：

「请根据患者描述判断可能的皮肤病。」

用了两周，发现 AI 老是漏问病史——患者上周去过海边、吃过什么药，这些对诊断很关键。于是提示词改了一版，加一句「先询问近期旅行史和用药史」。这就是**变异**：它和上一版不一样了。

新版和旧版同时在用。很快发现新版误诊率明显低，旧版被废弃。这就是**选择**：实践是裁判，效果说了算。

新版被证明好用之后，发给全科二十个医生用，每个人开口就是这套问法。这就是**复制**：改进从一份变成了二十份。

一个月后，某位医生发现 AI 对儿童患者还是容易误判，又加了一句「对 12 岁以下患者，优先排除川崎病」。新一轮的变异、选择、复制又开始了。这段提示词就像一只有性生殖的生物，一圈一圈地进化。

大白话

大白话：知识资产不是「写下来存档」的知识，是「写下来还在被改」的知识。进了档案柜再不动的那份文档，不是资产，是化石。

它和专利、文档、经验有什么区别

容易混淆的几个邻居，一次说清：

和经验不同：经验长在人身上，人走茶凉。资产写在纸上（或代码库里），人走了它还在。

和普通文档不同：文档写完就定稿，资产永远有下一版。判断一份东西是不是知识资产，就看一个问题：**它最近一次被修改是什么时候？**答案是「三年前」的，基本是化石。

和专利不同：专利的价值在于「不许别人用」，知识资产的价值恰恰相反——在于「组织里每个人都在用」。它越被复制越值钱，使用它的每个人都可能成为下一个改进者。

为什么现在要专门谈它

知识资产一直存在，但以前它进化得很慢——慢到没人觉得需要管理它的进化。一份诊断流程五年改一次，一家餐厅的菜谱十年不变。

慢的原因是：三个齿轮里，有一个一直转不动。变异靠人想，选择靠人试，复制靠人抄——全部限速在「人」这个环节上，而其中最先卡住的，是变异。想出一个像样的新方案，是最贵的一步。

然后 AI 来了。它做的事，相当于给变异这个齿轮装了一台发动机。下一章我们看看，这台发动机到底把转速拉高了多少倍——以及，另外两个齿轮扛不扛得住。

第三章 · 变异突然便宜了

先算一笔账。

假设你要给「客服怎么回复愤怒的投诉客户」这件事找一套更好的话术流程。在 AI 之前，变异是怎么发生的？你冥思苦想一个下午，写一版；找两个老客服聊聊，再改一版；运气好，一周能试三四个方案。变异的成本是一个有经验的人的一下午——很贵，所以你很省着用。

现在呢？把场景描述丢给大模型：「给我二十种处理愤怒投诉的开场策略，风格各不相同。」一分钟后，二十个方案躺在屏幕上。变异的成本，从一个人一下午，变成几分钱的电费和六十秒等待。

这就是正在发生的事：**变异这个齿轮的边际成本趋近于零**。边际成本，大白话就是「再多生产一份要花的钱」。复印机把文字的边际成本打到接近零，互联网把分发的边际成本打到接近零，而大模型把「生成一个像样的方案」的边际成本打到接近零。

不是快一点，是换了一个量级

注意，这不是「效率提升 30%」那种改进，而是物种级别的变化。看三个数字就够了：

- **代码**：一个程序员一天写几百行；AI 辅助下，一天可以生成并筛掉几千行。GitHub 的统计说，使用 AI 助手的程序员完成任务的速度快了约 55%——这还是保守的官方口径，实际在写样板代码、测试用例这类任务上，体感是好几倍。
- **方案**：咨询公司一个团队两周出一个战略草案；现在一个分析师上午就能让 AI 生成五个不同逻辑链的草案，下午逐个挑毛病。
- **文案**：运营以前写三条推送标题做 A/B 测试；现在一次生成五十条，挑都挑不过来。

变异的数量爆炸了。以前一个团队一年积累几十次「有意义的改动尝试」，现在一天就能产生这个数。

一个容易被忽略的细节：变异是免费的，但变异的质量分布没变

这里有个陷阱。AI 生成的二十个方案里，并不是二十个都比人写的好。真实情况是：**好方案的数量变多了，烂方案的数量也变多了**，而且烂方案比好方案多得多——这点和生物突变一模一样。生物体的突变绝大多数是中性的，一部分是有害的，只有极少数是有利的。AI 生成的方案也一样：大部分平庸，一部分有隐蔽的坑，极少数是真正的金矿。

生物界对这个事实有个应对：让变异随便发生，把全部压力交给选择。地球从来不缺突变，缺的是把坏突变筛掉的严酷环境。

大白话

大白话：AI 是一台马力全开的出题机。题出得又多又快，但题目质量没变——判卷的压力全压在老师身上了。以前是出题慢、判卷闲；现在是出题机狂转，老师累瘫。

瓶颈搬家了

把三个齿轮想成一条流水线：变异是上游的原料车间，选择是中间的质检车间，复制是下游的包装车间。上游产能扩大一百倍，中间和下游不变，会发生什么？

不是整体产能扩大一百倍，而是**原料在质检车间门口堆成山**。整条流水线的产出，由最慢的那个车间决定——这是运营管理里的常识，瓶颈决定产出。

AI 时代知识资产的瓶颈，就这么从变异搬到了选择——也就是**验证**。这本书剩下的篇幅，大半都在讲这个被压得喘不过气的新瓶颈：它难在哪（第四章、第五章），它给组织带来了什么结构性问题（第六章、第七章），怎么解（第八章），以及解决之后第三个齿轮才轮得到登场（第九章）。

第四章 · 压力压到了选择

瓶颈搬到了选择。这一章把「选择」拆开看看，它在 AI 时代到底长什么样。

我们平时说的「验证」，其实是两件很不一样的事，难度天差地别：

第一种验证：对代码的验证

AI 写了一段代码，对不对？这种验证相对幸福，因为代码有一个奢侈的属性：**它能跑**。

跑起来就有答案——编译过不过、测试绿不绿、输出符不符合预期。所以软件行业几十年攒了一整套自动化验证的家当：单元测试（把代码切成最小块，每块配一个「输入 A 必须输出 B」的断言）、持续集成（每次改动自动跑全部测试，红灯一亮就拦下）、类型系统、静态检查。这套家当的意思是：大量验证工作可以交给机器，人只看最终结果。

所以「AI 写代码」这件事能这么快落地，核心原因不是 AI 写得有多好，而是代码行业碰巧**早就把选择车间自动化了**。变异放大一百倍，质检也跟着放大一百倍，流水线没堵死。

但别高兴太早。测试只能验证「代码做了我们想到要检查的事」，验证不了「这段代码解决的到底是不是真问题」「这个设计三个月后会不会变成灾难」。自动化测试能抓语法级的错，抓不了方向级的错。

第二种验证：对解决方案的验证

AI 给出一份诊断建议、一套营销方案、一份合同条款——这个对不对？

这里没有「跑起来」这回事。一份诊断建议不能像代码一样丢进编译器，看它报不报错。你没法给「这个营销方案能不能打动 35 岁女性用户」写一个单元测试。

这种验证只能靠：**懂行的人，结合现实，做判断**。它慢、贵、无法完全自动化，而且——这是关键——它恰恰是这个时代数量爆炸的那类产出。AI 生成代码的能力很强，但它生成「方案」「建议」「分析」「判断」的能力更强，因为这些连语法错误都没有，句句通顺。

大白话：对代码的验证像判数学卷子——有标准答案，机器都能帮忙判。对方案的验证像判作文——没有标准答案，得语文老师一篇一篇读。AI 让交卷量暴涨一百倍，而作文老师的数量没变。

选择的裁判权，在现实世界里

生物界的选择，裁判是环境——不需要谁来做决定，适者生存自动发生。但组织里的知识资产没有这个奢侈。一段提示词的「环境」是什么？是真实的患者、真实的客户、真实的业务后果。裁判工作必须有人来做：

- 有人要决定：这次改动算改进还是改糟了？
- 有人要观察：新方案在真实业务里效果如何？
- 有人要拍板：这个发现要不要写回知识资产？

每一个「有人」，都是活生生的人，每个人都有每天八小时的限额。变异可以免费无限量，但**验证的产能被人的数量和人的专业度死死锁住**。这就是第三章说的那座「堆在质检车间门口的山」。山不会自己消失，只会越堆越高——或者，从山脚下开始塌方：没人验证的方案被直接用进了真实业务。

下一章看看这座山的地质构造：验证为什么这么难，难到连「找更多人来验证」这个显而易见的解法都不太好使。

第五章 · 验证为什么比生成难

上一章说，对方案的验证只能靠懂行的人做判断。这一章追问一句：这个「判断」为什么这么难？难到什么地方？

第一道难：它看起来都对

人工写方案，质量参差写在脸上——逻辑断裂、格式混乱、一看就是新手。烂方案自带烂的气质，筛选成本很低。

AI 不是这样。它生成的每一个方案都**文笔流畅、结构完整、引经据典、语气自信**。一份有致命漏洞的市场分析和一份真正高明的市场分析，摆在桌上的卖相几乎一样。这意味着筛选者失去了最便宜的那个筛子——「看一眼就知道不行」。

认知科学里有个词叫流畅性启发：人脑会把「读起来顺」错当成「说得对」。读起来越流畅的东西，我们越容易相信它。AI 是这个偏见的工业化生产设备——它把每一份产出都打磨到最流畅，包括那些错的。

结果就是心理学里说的自动化偏见的升级版：面对一个看起来无比专业的机器输出，人的批判性思维会系统性怠工。医生面对 AI 给出的诊断建议时，过度信任导致的误诊，已经有实证文献记录的现象。

第二道难：代价不对称

验证一个方案的成本，和生成它的成本，差距大到荒谬。AI 用三十秒生成的合同草案，律师要逐条审两小时——而且这两小时省不掉，因为风险恰恰藏在没人细看的地方。

更麻烦的是**失败的不对称**：采纳一个好方案，收益是渐进的、不起眼的；采纳一个坏方案，代价可能是突然的、致命的。一段有缺陷的诊断提示词，可能在一千个病例里表现正常，然后在第一千零一个上漏掉一个罕见病——而这时它已经被复制到全科室了。

大白话：验收比生产贵，验收漏了的代价又比验收本身贵得多。这就是为什么质检车间不能靠加班解决问题——它是结构性短缺。

第三道难：验证需要两种稀缺品的交集

合格的方案验证者，需要同时有两样东西：

一是业务知识。判断 AI 的诊断建议对不对，你得真的懂这个病；判断营销方案行不行，你得真的懂这群客户。这不是通用智力能替代的——AI 本身的通用智力极高，它缺的恰恰是你们医院、你们产品、这群客户的具体上下文。而这正是人手里最后的牌。

二是品位 (taste)。同样都「对」的两个方案，哪个更优雅、更可持续、更符合组织的方向？这不是对错判断，是优劣判断。品位来自大量实践后的直觉积累，没法速成，没法写在手册里。

这两样的交集，就是组织里的业务专家。而专家们当前在干嘛？在**干活**——在亲手看病、亲手写方案、亲手做那些被 AI 一天比一天做得更好的事。下一章算一笔账：这种人力配置，让组织的知识资产进化得有多慢。

第六章 · 一个人用，还是全医院用

前面几章说的都是「验证很贵很难」。这一章算一笔具体的账，看看验证者的数量，怎么直接决定一个组织的进化速度。

一个人的迭代速度

设想一家医院里，只有张医生一个人用那段「AI 辅助皮肤病诊断」的提示词。

他每天看二十个病人，其中大概三五个会触发他对提示词的怀疑——「这个病例 AI 没问过敏史，是不是提示词里漏了？」他记下来，下班后改一版，明天接着试。

这段提示词的迭代速度是多少？就是**张医生一个人的实践速度**：每天三五个新发现，每周一两版改动。一年下来，假设五十次有效迭代。不慢了——但注意这个速度的上限，被「张医生见过的病例」锁死了。他没见过的病，提示词里永远不会有。

全医院的迭代速度

现在换一个配置：全医院两百个医生，人人都用这段提示词。

每个医生每天的病例都不一样——儿科医生撞见儿童误诊，急诊医生发现 AI 不懂「病人主诉和实际病情不符」这种常态，皮肤科医生发现某种罕见皮疹的描述方式不对。每个人每天的发现，都可能成为知识资产的下一次迭代。

如果这两百人的发现都能回流，一年不是五十次迭代，而是几千次。这段提示词的进化速度，差了**两个数量级**。一年以后，「全院版」的提示词和「张医生个人版」的差距，就像家养马和野生种群的基因库差距——一个是近亲繁殖，一个在大草原上被整个环境筛选。

大白话

大白话：知识资产的迭代速度 = 使用人数 × 每个人的实践频率 × 发现回流率。前两个因子由「多少人用」决定，最后一个因子由「组织怎么设计」决定——而大多数组织输在最后那个因子上。

发现回流率：真正拉开差距的因子

用的人多不等于迭代快。如果两百个医生各用各的，发现都烂在各自脑子里，那这段提示词的年迭代次数还是五十一——只是五十个不同的人各自攒了五十次，资产本身一动不动。

所以公式里最关键的不是人数，是回流率：一个发现有多大比例真的被写回了知识资产。这个数字在现实组织里惨不忍睹——经验是：没有专门机制的话，十个发现里能有一个被写回来就不错了。剩下的九个，以「我下次注意」的形式蒸发掉。

这里有一个反直觉的结论：**一个人用 + 高回流率，可能胜过一百个人用 + 零回流**。张医生一个人认真记录每次迭代，好过全院用得热热闹闹但谁也不写回。当然，最强的是两者兼得：人多，且回流有管道。

覆盖率问题登场

把这笔账推到底，就得出了这本书的核心问题：**要让知识资产快速进化，你需要让尽量多的业务一线专家成为验证者**——他们每天接触最多的真实病例、真实客户、真实异常，他们的发现是最值钱的变异原料。

但现实中，业务专家往往不是验证者。他们的工作定义是「看病」「做方案」「服务客户」，不是「验证和迭代组织的知识资产」。知识资产由一个专门的「AI 团队」或「流程部门」维护，专家只是用户——而且是沉默的用户。

这就是覆盖率问题：知识资产的验证者，只覆盖了业务现场的一小块。没被覆盖的那一大块现场，每天都在产生发现，又每天把发现丢掉。

下一章正面处理这个问题：如果业务专家不成为验证者，会发生什么——不只是知识资产长不起来，还有专家自己的价值，会以一种很多人没意识到的方式缩水。

第七章 · 覆盖率问题

上一章提出了覆盖率问题。这一章把它的另一面摊开：这不仅是组织的损失，更是业务专家个人的价值警报。

先看清一个趋势：执行的价值在跌

回想一下三个齿轮的分工。AI 接手的是「变异」——生成方案、起草代码、给出初稿。这些事以前恰恰是业务专家的核心产出：医生看病诊断、分析师写报告、程序员写代码。当「做出一个及格线以上的产出」变成免费，「**能做出及格线产出**」这项能力的市场价格就开始崩塌。

这不是说专家没用了。恰恰相反——第五章说过，验证需要业务知识和品位的交集，而这个交集只有专家有。问题在于：专家的价值从「生产」迁移到了「判定」，但很多专家（和他们的雇主）还没完成这个迁移，还在用旧的价值定义上班。

一个残酷的推演：如果张医生不碰那段提示词的验证和迭代，只是每天照着它的输出看病——那他实际上变成了一个「AI 输出的执行终端」。他的二十年临床经验，既没有被写进资产，也没有参与资产的筛选。这时他的可替代性，和一个刚毕业的医学生没有本质区别：反正都是照着 AI 的建议办事。

大白话

大白话：新时代的账是这么算的——你的经验如果只用来「干活」，AI 在干活上追你追得飞快；你的经验如果用来「判定对错并改进知识资产」，AI 反而成了你的放大器。同一份经验，投放在不同的齿轮上，命运相反。

验证者为什么比生产者更值钱

经济学上有个简单的道理：值钱的是瓶颈。当生产不是瓶颈时，生产者不稀缺；当验证是瓶颈时，验证者最稀缺。

而且验证者有一种生产者没有的杠杆——一个人的产出只影响他自己的工作量，但一个人的验证影响的是整个资产。张医生作为生产者，一天看二十个病人；张医生作为验证者，他改进的提示词被全院使用，等于他的判断被复制到了两百个医生的每一天里。杠杆差了一百倍。

软件行业早就演过一遍这个剧本：代码评审（code review）文化里，最受尊敬的工程师往往不是写得最快的，而是评审眼光最毒的——他们的一句「这个设计三个月后会让你后悔」，避免的损失超过一个团队一周的产出。评审者就是代码世界的验证者。

组织的损失：资产停在低水平均衡

从组织这边看，覆盖率问题的后果是知识资产卡在一个低水平均衡——资产质量停留在「维护它的那个小团队的认知上限」，而全组织其他几百人的认知，一点都没渗进去。

更隐蔽的是恶性循环：资产质量不高 → 一线专家用了觉得「也就那样」 → 更不愿意参与改进 → 资产质量继续不高。打破这个循环，不能靠喊话「请大家积极反馈」，得靠机制。这是第八章的主题。

先把结论钉在这一章的尾巴上：**在 AI 时代，业务专家如果不成为知识资产的验证者，他的价值会变得很低——这不是威胁，是算术。**变异免费之后，唯一稀缺的是判断，而判断恰恰长在专家身上。问题只剩一个：怎么把这份判断接进齿轮里。

第八章 · 让专家愿意也能够验证

覆盖率问题的解法，不是「让专家们更努力」，而是把验证这件事的成本打下来，把回流的管道修出来。这一章讲具体怎么做。

原则一：让验证发生在工作流里，而不是工作流外

最失败的机制设计，是给专家新增一个任务：「请在下班后填写知识资产反馈表」。这等于要求别人用休息时间做一件没有即时回报的事，回流率必然是零。

正确的设计是把验证嵌进他正在做的事里。张医生看 AI 诊断建议的那个界面，旁边就应该有一个「这条不对，应该是……」的输入框——一次点击、一句话，三十秒。反馈不是额外工作，是工作的一个手势。

软件业有现成模板：报错弹窗里的「发送错误报告」按钮。工程师不需要写邮件描述 bug，崩溃现场自动打包，一键回传。这个设计让全世界的软件用户变成了免费的测试员——数亿人每天都在为软件做验证，他们自己都没意识到。知识资产要的回流管道，就是这个东西的业务版。

原则二：能自动化的验证，绝不留给人

第四章说过，软件业攒了一整套自动化验证家当。这些家当对知识资产同样适用，只是要翻译一下：

- **给提示词写「单元测试」**：攒一批有标准答案的历史病例，每次改提示词就自动跑一遍——改了「儿童用药史」之后，原来能答对的三十个病例还答得对吗？这叫回归测试，防止改进一处、搞砸三处。没有它，知识资产的每次迭代都是盲人摸象。
- **基准集**：把「什么叫好」量化成一组固定考题。新版本先过考题再上线，不依赖某一个人的印象分。
- **灰度发布**：新版本先给 5% 的使用场景跑，指标正常再全量。生物界管这个叫「小规模试错」，软件业管这个叫金丝雀发布——矿井里先放金丝雀，有毒气它先倒。

自动化验证每多覆盖一分，人类专家的稀缺判断力就省下来一分，花在真正需要品位的地方。

原则三：让贡献可见，让回路闭合

人为什么愿意当开源软件的贡献者？三个原因：贡献被署名、改动被采纳有即时反馈、自己的名字进入了「这个资产的作者列表」。这三样，组织内部的知识资产一样可以照搬：

- 每一版资产的更新日志写上：「本次改进来自急诊科李医生的发现」。
- 有人提交发现后，三天内必须得到回应——采纳或说明为什么不采纳。**沉默是回流管道的头号杀手**：一个发现石沉大海，这个人这辈子不会再提交第二次。
- 把验证贡献计入对专家的评价。如果考核表上只有「看了多少病人」，那验证永远排在最后。

大白话

大白话：让专家当验证者，本质是修一条从「现场发现」到「资产更新」的管道。管道的两头都要降阻力：提交这头要做到三十秒一次点击，回应这头要做到三天内有说法。管道中间，能机器判的别让人判。

小结

三个原则合起来一句话：**验证不是靠觉悟，是靠设计**。嵌进工作流、自动化能自动化的、让贡献可见——做到这三点，覆盖率问题就从「专家不愿意」变成「管道有多粗」的工程问题，而工程问题是有解的。

验证的问题解决了，改进攒下来了。但这些改进如果只存在某个角落，进化就还没完成——还差最后一个齿轮：复制。下一章。

第九章 · 复制：进化的最后一公里

前八章把变异和选择讲透了。现在轮到第三个齿轮——它经常被当成「理所当然」的一步，实际上，组织里大量的进化恰恰死在这一步。

只有一份的改进，等于没有改进

回到第一章那个思想实验：一只长颈鹿变异出了完美脖子，但特征传不下去，对种群等于零。组织版本是：急诊科李医生发现了一个改进，写回了提示词，资产更新了一一**然后全院两百个医生还在用旧版。**

这在现实里天天发生。改进躺在某个共享文件夹的角落里，文件名是「诊断提示词_v3_最终版_真的_final.docx」，而一线医生电脑里存着 v1，用了两年。改进完成了变异和选择，死在了复制上。

生物界没有这个问题：基因一旦在个体身上被选中，会通过繁殖自动扩散到整个种群，不需要谁发通知。组织不行——组织里的复制是个**主动的分发工程**，而且有三个专门的坑。

复制的三个坑

坑一：版本分裂。同一个资产，二十个人手里有六个版本，谁也不知道哪个是「现在对的那个」。软件业用版本管理和「单一真源」解决——代码只信仓库里那一份，本地副本随时对齐。知识资产需要同样的纪律：任何场合引用它，都指向同一个地址，不复制粘贴副本。

坑二：分发不触达。「我发在群里了」不是分发。群消息三天后就被淹没。真正的分发是：用户在他干活的那个界面里，用的自动就是最新版——就像手机 App 自动更新，用户无感。资产的使用路径如果不是从真源实时取，复制就永远慢半拍。

坑三：默认路径不改。最隐蔽的一个。新版资产上线了，但新人的入职培训材料还是旧的，考核标准还是旧的，领导开会引用的还是旧流程。组织的「默认路径」——培训、考核、会议、模板——是复制的最后一公里。资产更新了而默认路径没更新，等于基因变了而身体的发育程序没变。

大白话：复制 = 一个真源 + 自动更新 + 改掉默认路径。三件事缺一件，改进就困在原地。检查方法很简单：随便抓一个一线的人，问他现在用的是哪一版——他答不上来，或者答的版本不是最新的，复制这个齿轮就在空转。

复制也是放大器——双向的

最后提醒一句：复制放大一切，包括错误。第五章说过，有缺陷的资产「在一千个病例里表现正常，然后在第一千零一个上出事」——它之所以能在第一千零一个病例上出事，是因为它被复制了一千次。

这就是为什么三个齿轮的顺序不能乱：**变异可以狂野，选择必须严格，复制要等选择之后**。没经过验证的改进直接全量复制，等于让没做过质检的产品直接进全国货架——第八章的金丝雀发布（先 5% 再全量）就是给复制加的刹车。

三个齿轮都转起来了：AI 供料（变异），全员验证（选择），管道分发（复制）。下一章把它们画在一张图上，回答全书开头那个问题。

第十章 · 一张组织的进化地图

回到全书开头的问题：当 AI 把变异做到几乎免费，一个组织的知识资产靠什么继续进化？

走了九章，现在可以把答案画在一张图上了。

三个齿轮的新分工

变异——AI 供料，人出题。生成方案的边际成本趋近于零，这个齿轮已经装了发动机。人在这头的角色变了：不再是亲手攒每一个方案，而是决定「往哪个方向变异」——提什么问题、给什么约束、喂什么上下文。出题的质量决定变异的命中率。

选择——全员验证，机制兜底。这是新的瓶颈，也是新的主战场。对代码的验证尽量自动化（测试、基准集、金丝雀发布）；对方案的验证只能交给懂业务、有品位的人——也就是业务专家。覆盖率问题必须正面解决：专家不进场当验证者，资产停在低水平均衡，专家自己的价值也在缩水。解法不是觉悟，是设计：把反馈嵌进 workflow、能机器判的别让人判、让贡献可见、三天内必有回应。

复制——一个真源，自动更新，改掉默认路径。验证通过的改进不扩散就白进化了。版本要唯一、分发要无感、培训和考核这些「默认路径」要跟着资产一起换。复制是双向放大器，所以它必须排在严格的选择之后。

大白话

大白话：变异随便野，选择往死里严，复制等判完卷再发。三个齿轮都转起来，组织的知识资产才算活着。

一张自检清单

想知道自己组织的知识资产进化机器卡在哪，按顺序问五个问题：

1. 我们的核心知识资产（提示词、流程、清单）最近一次更新是什么时候？答不上来 → 它可能是化石。

2. 一线的人每天能产生多少个「这个可以改」的发现？接近零 → 变异方向没人把守，或者大家根本没在用资产。
3. 这些发现有多大比例被写回资产？很低 → 回流管道断了，回到第八章。
4. 谁在验证？如果只有一个小团队 → 覆盖率问题，回到第六、七章。
5. 随便抓一个一线的人，他用的是最新版吗？不是 → 复制齿轮空转，回到第九章。

五问走下来，哪个环节答得心虚，瓶颈就在哪。

最后一件事

这本书讲的其实是同一件事的两个尺度。在组织的尺度上：变异免费了，选择成了瓶颈，谁掌握验证，谁就掌握进化的方向盘。在个人的尺度上，完全是同一个结构：你的经验如果只做生产，会被免费的变异追上；如果做验证，会被复制放大一百倍。

所以那个贯穿全书的问题，最终的答案短得只有一句：

变异已经免费，判断仍然昂贵。把自己和组织都改造成「判断的机器」，进化的齿轮还握在你手里。