AI 炼金术 · 课程全文

AI 原生组织转型

不是转组织来用好 AI，而是用 AI 来替代组织

任鑫（Mars） · 2026

> 这份全文是给 AI 读的，也是给你读的。
>
> 推荐用法：把这份 PDF 连同 t.marsren.ai 上的「落地参谋提示词」一起丢给最熟悉你和你公司的那个 AI，让它读完课，再问你几个问题，然后告诉你——你们公司具体该怎么落地。

AI 原生组织转型 · 第 1 讲

模块一 · 为什么「任务层」的 AI 转型无效

开场 · 人人都提效了，公司没有

AI 已经用上了，但组织效率没变

越来越多的公司跟我说，他们已经把AI部署下去了，也发现了越来越多的应用场景，越来越多的工作流程当中已经嵌入了AI，但是整体核算一下会发现，虽然可以出很多demo，出很多PPT说自己AI降本增效了，但公司的整体效率并没有提升。

效率已经提升了，但钱却没多挣

还有一类公司会说，他们其实是提效成功了的，还可以拿出具体的降本增效数据和案例，比如说这个环节原来多少人，现在只需要多少人。但是一问他们，你们有没有赚钱呢？就支支吾吾说，嗯，其实没有，今年生意不好做，也没有多赚钱。

AI 提效进入“鬼故事”阶段

总的来说，AI提效进入了一个鬼故事的阶段。好像处处开花，但就是看不到结果。今天就来聊一聊这背后的原因可能是什么，以及如何应对、如何解决、如何才能真正做出AI转型，拿到结果。

- **处处冒烟，就是不着火。**
- **每个人都变快了，公司没有变快；每个环节都省了钱，账上没有多钱。**

现场调研：你们在哪里？

- A：还在学习了解 AI 有什么用，能怎么用
- B：已经把 AI 落地用起来了，但整体组织还没提效
- C：已经整体提效了，但还没扩大市场增加盈利
- D：整体提效了，而且已经获得了商业回报

拆解：AI 影响的三个层次

AI 影响的三个层次：任务，协调，生态

AI对于我们整个世界的影响，不仅仅是提供了一个可以提效的工具，它其实影响的是三个层次。第一个层次是任务层，它让我们干活做任务可以变快，可以提效。第二层是组织协调层，它影响的是我们相互之间的协同，影响原有的组织方式、原有的资源组织方式、原有的协调方式，它可以更高效地把资源拼起来达成结果。第三层影响的是整个世界，整个商业格局会变。

所以本质上来讲，它对我们的影响分为任务层、协调层和生态层。任务层影响的是我们干活的速度，协调层影响的是我们一个一个的组织有效调度资源、对齐方向和推进成果的效率。而生态层影响的是我们所处的市场本身，甚至整个经济体，甚至整个社会本身。它会造成更大的影响，会让我们面对一个新的世界。

协调层：一流武装 ≠ 一流军队

任务层很好理解嘛，我们每个人做PPT做快了，写代码写快了，这就是任务层提效。协调层是什么呢？协调层是说我们的组织本身，各种组织形式，不管你是政府机关也好，是企业也好，是一个慈善组织也好，是一个平台也好，本质上来讲，我们是通过一个机制，把很多人的不同的努力、很多资源综合起来，一起去达成一个结果。

因为这并不仅仅是一个简单叠加的过程，不是说30个人的团队就一定比20个人的强，并不是这样。单兵的能力并不等于组织作战能力，士兵的数量不等于这支军队的战斗力。所以历史上才会出现几十万人被几万人、甚至于几千人打得团团转这种事情。

AI可能是很好的单兵武器，但如果我们协调层不做改造，就会像是北洋水师买铁甲舰，买得到最好的定远、镇远，当时亚洲吨位最大的铁甲舰，但是如果你的指挥体系、训练、弹药后勤、人事激励全是旧军制，没有相应的进化，你只能买来最好的军舰，但是你不会有最好的海军。

军舰可以花钱买，海军只能自己练。

生态层：赢得战斗 ≠ 获得战果

更惨烈的是，哪怕你拥有一支良好的军队，你可以赢得战斗，但你未必能够取得战果。因为商业世界最重要的事情并不是高效地做出来一些东西，而是你的东西有没有满足市场的需求？并且你在满足需求的过程当中，有没有占据垄断性的生态位、占据瓶颈的生态位？只有你既满足需求，又占据瓶颈性的生态位，获得了一些类似于网络效应之类的竞争优势，你才可以取得高速的增长和利润。

如果只是高效地交付东西，但是没有满足市场需求，这其实就是在无谓地浪费资源。如果高效地满足了市场的需求，但是没有占据垄断性的生态位，没有核心的竞争优势，在一个自由竞争、卷的生态里面，那最多也就是赚一个非常辛苦、非常微薄的存活，保证存活而已，没有办法取得增长和收益。

举例来说，昆仑万维用 AI 做短剧，提效了 10 倍，但一年亏十几亿，而且看不到好转的迹象，这又有什么用呢？

效率是我们定的，价值是市场定的。

三级台阶：两个难关，先聊一个

总的来说，从单点的任务侧AI提效，到这些提效的成果能够拼装起来形成组织的竞争力，再到组织的竞争力可以在市场上取得战果，并且占据一些垄断性的、稀缺的位置，获得增长或者盈利，这其实有两个大台阶要跨。

第一个台阶就是要在协调层做AI组织转型，第二个台阶就是要在生态层做AI原生的战略准备。我们今天先讨论第一件事情，如何在协调层做AI组织转型。

分析：为什么任务提效 ≠ 组织提效

局部提升，对全局影响不够

阿姆达尔定律

简单做一个分析。当你的公司里面有很多很多的任务，一个流程当中有非常多的步骤，那你优化了其中一两个步骤、一两个环节、一两个任务，其实对整体的优化是非常有限的。

> $S = 1 \div [(1-p) + p/s]$
>
> > p = 你优化的那部分占全流程的比例，s = 这部分快了多少倍

类比：京沪线飞机提速

举个例子来讲，北京到上海的飞机可能是130分钟。如果有一个新的技术可以把这个航班优化到100分钟，这其实是一个非常高科技的进步，非常难，非常不容易。看起来从130分钟优化到100分钟，进步非常大。

但作为一个商务旅行的人，我其实不是这样看问题。因为总体上来讲，我关心的并不是飞行的时长，而是门到门的时长。从我上海的家里到机场一个小时，然后要提前90分钟、也就是一个半小时到机场准备登机，飞机上再待130分钟，落地再出机场、到目的地估计又要一小时。这样整体算下来，我需要的时间是60+90+130+60，等于340分钟。而这30分钟的优化，在340分钟当中连10%都不到，对我来讲其实忽略不计。我心里想的就是，门到门还是要5个小时。

案例：局部 AI 提效

类似的很多局部的AI提效，不管是我们做图做得更快啦，审核各种文件可以看得更准啦，还是某个环节可以全部用AI来做、比如说客服电话的审核啦，这些事情往往在你的组织里面都只占了其中一部分工作。那么哪怕你把AI提效做出来，对整体组织的影响也是有限的。

我自己也参与了很多公司的AI转型。在这个过程当中，我合作的公司其实也发了很多PR稿，我们也确实做出来了非常多局部的成效。但只要是大型公司，不管有没有我的参与，你会发现现在都没有什么本质上的、对公司层效率的显著提升。

另外一方面，大家现在天天看到的那些PR稿里的大公司，说我们今天做AI这个、做AI那个，其实他们私下也在找我聊，说也卡在这里，并没有获得什么公司层的本质进展。那么其中一个原因，就是局部提效对于整体的影响其实太有限了。

PR 稿是真的，卡住也是真的。

瓶颈约束，让提效也是浪费

系统效率由瓶颈环节决定

讲到这里，你可能会说，不对啊。对于一家大公司来讲，虽然你只优化了一两个环节，只优化了1%、2%，但是这家公司规模很大，这1%、2%、10%也是很有价值的呀。你不能光看相对值，得看绝对值，对吧?

确实如此。那么更糟的一个问题就是，很多时候事情并不是线性的，并不是像我刚刚举的坐飞机的例子那样，我优化了这个环节，它就可以在全局上达成影响。因为对于一个复杂系统来讲，它最终的表现取决于瓶颈环节，但凡你优化的不是瓶颈环节，那么对整体表现的贡献几乎就为0，会全部被瓶颈环节吃掉。

类比：洗菜快 10 倍，灶台只有 4 个

类比一下，比如说你开一个餐馆，你的洗菜工、摘菜工，他们的效率提高10倍。但是你们的灶台只有4个，这只会造成洗好的那些菜堆积如山，并不能加快菜品的生产速度。因为瓶颈是灶台太少了，同时能够做的菜太少了。

案例：20 倍代码提效，却卡在审核上

前 Meta L8 Kun Chen 他离职单干前的真实困境："我一天写 20 个 PR，没人审得过来……**我发现自己只好少写点 PR——因为瓶颈在团队其他人身上。**""我们的工作流、团队编制，全是在'人把大部分时间花在写代码'的年代设计的。当你开始写 10 倍的 PR，一切都没按这个假设建过。

他不是被别人拖慢的，他是自己选择慢下来的。 因为快没有用。

组织协调，才是真正的瓶颈

白领工作，60% 精力不在“做事”

对于大部分白领工作来讲，其实大部分的时间精力并不在做事，而在于协调。协调呢，就是去对齐、去理解老板到底想要什么，把老板想要的东西结合下属的情况说给下属听，和其他兄弟部门说清楚你干什么、我干什么，以及催这个活、催那个活，然后救火。最后还要开复盘会，说这个事情到底该怎么干？我们大家对这个错有没有共识？下次该怎么做？该甩锅甩锅，该担责担责，该推进推进。

而且你会发现，在你们公司，花钱花得越多的管理人员，其实越多的时间是在做这一类的事情；工资越高的人，做协调类的工作就做得越多。在座各位也可以把自己上一周的日程看一下，你多少时间是在真的干活，多少时间是在开会，多少时间是在碰一碰、沟通沟通。

60%的精力是在协调，只有40%在做事、在做任务。所以我们在任务上的效率优化得再多，也就是优化了不是瓶颈的那40%而已。

更可怕的事情是，当协调出现问题的时候，任务层的努力是会被溶解掉的，任务侧的努力成果是会被看不见的。它是一个乘数。

案例：两边都做对了，但探测器烧毁了

1999 年 NASA 火星气候轨道器（Mars Climate Orbiter）。NASA 调查结论是一个团队传英制单位、另一个团队按公制使用，关键接口没有识别并纠正差异，导致轨道计算错误、飞船损失。（给我具体数据）

两边的工程师都把自己那一段算对了，可能都很努力，可能都很有成效，可能都很有效率。但最终它其实是死在了两边没有对齐语言，没有检查出错误，也没有合理的交接。所以真正的问题不是出现在每一个任务里面，也不是出现在每一个岗位上，而是出现在岗位和岗位中间的那个断档、任务和任务之间的那个衔接上。

- **1998 年 12 月 11 日**发射，飞行 **286 天**，**1999 年 9 月 23 日**入轨时失联
- 造价 **1.25 亿美元**（只算轨道器本身）
- 根因：地面软件里一个叫 **"Small Forces"** 的文件，推力器性能数据用了**英制的磅力·秒（lbf–s）**，而接收方按**公制的牛顿·秒（N–s）**解读
- 后果：实际进入火星大气的高度约 **57–60 公里**，而计划高度约 **150 公里**——**低了将近一百公里**，探测器在大气层中解体烧毁
- 双方：**洛克希德·马丁**（丹佛）负责航天器，**NASA JPL** 负责导航

286 天 / 1.25 亿美元 / 一个单位。这三个并排，比任何一句话都响。

- 「最危险的错误**不发生在格子里，发生在格子和格子之间**。」

AI 放大并暴露了这个瓶颈

其实组织的协调一直就是生产力的瓶颈，但因为之前生产本身很慢、做事本身很慢、完成任务本身很慢，所以我们看不到协调层其实才是效率的瓶颈。但是现在做事、完成任务本身很快了，这个瓶颈一下子就显得非常显眼。

举例来说，以前你写一个软件需要开发三个月，那你走一个流程走两周，开18个会，每个会浪费半天，然后等回复，每次等个两三天，这都是可以接受的。但现在如果写一个软件本身只要6个小时，你再让他开18个会，再让他等一个回复等两三天，这件事情就会显得特别荒谬，这个瓶颈就会被凸显出来。

解法：用 AI 来做协调层

为什么有组织

组织是一种发明

《企业的性质》——

> **企业为什么存在？因为在市场上找人、谈价、签约、监督的成本太高。**当组织内部协调比市场交易便宜，企业就出现；哪边便宜，边界就往哪边挪。

组织并不是一种天经地义的东西，它其实是因为有科斯定理的存在的话，所以我们用来替代市场来进行高效协调的一种方式。

现代公司，比如说科层制啊，比如说员工啊、朝九晚五啊，包括怎么分配任务、怎么做绩效、怎么做KPI，包括工作本身，其实都是特定时代、特定生产力下面的一个发明，它是用来解决协调问题的。组织和物理学、和蓝天白云不是一个东西，它是我们人类发明出来解决问题的一种东西，它不是天经地义的存在，也不是必然要遵守的约束条件。

所以我们思考AI组织转型的时候，脑子当中可能会有一个不必要的隐含假设，叫做我们的公司就应该长这样，我要把它做一下微调去适应AI，让它用好AI。其实不是这样的，公司可以完全不长这样，甚至可以完全没有公司。你要从零去思考，从第一性原理去思考：组织到底是为什么而存在的？我们又是用什么样的方法结合AI、结合先进的生产力，更有效地达成原有组织要达成的这个目的。

很多的假设都不再成立

现在组织的一些基本长相，比如说一定要有层级，一定要有部门，一定要有上下级，一定要有KPI，一定要有流程，这些都是基于特定的生产力约束想出来的、去有效协调的解决方案。但是这些生产力约束，很有可能已经不成立了，这些假设也不再成立了。我们现在还抱守着这些东西，就不靠谱。

比如说，我们一定要有层级，不能一个人管200个人，中间还需要再架一层到两层，是因为人的信息输入带宽和处理带宽太有限，我们看不到所有人的情况，也没有办法对每个人进行管理。但AI如果能够看到每个人的情况，跟每个人沟通，理解每个人的任务，推进每个人的任务，是不是其实不需要分层？

比如说我们一定要有部门，是因为每个领域的专业性都太强，我们要把专业的人聚拢到一起，方便统一协调，然后给这个专业性的小群体提供一个接口，让外界方便调用。那如果每个人都有了一个万能的翻译，可以帮他去对接任何的组、任何的专业性，是不是就不需要把所谓专业的人放到一起，而是可以把他们拆散到具体的业务中？

很多的形态，只是历史的化石

我觉得很多形态之所以有这些设计，是基于当时的生产力水平的，是适应当时的生产力水平的。但是AI出现了，它极大地改变了所有的前提假设。所以这些东西今天再看，不应该是「它过去管用，所以今天也管用」。我们应该看到，它可能是历史的化石，是上一个时代的化石。我们可以研究它，然后从第一性原理去挑，看哪些还管用。

> 这些形态不是**规律**，是**遗迹**。它们证明的不是「应该这么做」，只是「**当年只能这么做**」。

「它们不是定律，是化石。」

类比：坐月子的禁忌

就好比说一些坐月子的禁忌，比如说不能洗头。这个在古时候其实很有可能是有道理的。古人本来就很少洗澡洗头，你为什么一定要在月子里洗呢？那时候烧水没有热水器，烧了水很容易就凉掉；也没有电吹风，你洗了头发很难干；也没有抗生素，没有现在这些各种各样的药，感冒了之后又很难好。那个时候，不洗澡不洗头其实是非常合理的一个选择。但是放到现在来讲，就会觉得这是一件非常脑残的事情。

任何的结构其实都是适应当时的生产力的。时代变了，我们的打法就要变。

用 AI 来替代组织职能

AI 转型，不是转组织用好 AI，而是用 AI 替代组织

所以我们思考AI转型的时候，经常会有三个层次。第一个层次是我怎么把AI工具给用好。第二个层次是，哎呀，我用好了，但是好像组织上没有起效果，我怎么样把组织做一下微调去适应AI，把它的效果给释放出来。第三层才是本质，就是我如何用好AI来替代组织的各种功能，让整个组织的效率可以得到极大的提升。

金句：AI 不是组织的工具，是组织的竞品

AI 不是组织的工具，是组织的竞品

AI 不是组织的工具，是组织的竞品

（强调展示金句）

协调的瓶颈：对齐，推进，闭环

我们一般期待组织有很多很多的功能，我们今天讲效率，那和效率比较有关的是三件事情。

第一件事，我们希望组织能够把所有人拉到 same page、在同一张纸上，帮我们完成对齐的工作，让所有人可以劲儿往一块使，看到同一个图景，往一个方向努力，把所有的能量协调起来。第二件事，

我们希望组织可以有效地推进事物，就是一件事从一开始起心动念，到最后完成商业成果，整个过程能够有效推进，不要卡在那里，这才是整体的效率。第三件事，我们希望这个组织是活水，它能够持续地进步，做得不好的它能改，做得好的它能够沉淀，它是一个自己生长的生命体。所以总的来说，就是对齐、推进和闭环。

那与之对应的，一个坏的组织长什么样？第一，如果一个组织里面你发现人心不齐，所有的事情你想你的、我想我的，这肯定不好。第二，如果一件事流程很长，这里卡一下、那里卡一下，这肯定不好。第三，一个错误会连续犯，一个经验沉淀不下来，这肯定不好。所以总的来说，看一个组织好不好，也就是看它能不能对齐、能不能推进、能不能闭环。

AI 转型的本质：对齐，推进，闭环

传统来讲，我们会用组织来完成协调的工作。而我们发现，组织在完成对齐、完成推进、完成闭环上面，现在都成了巨大的卡点。因为任务层的效率已经提上来了，所以协调层这些瓶颈非常明显。那我们AI转型的本质，其实就是要用AI来更有效地做组织内的对齐、推进和闭环。这样，我们的方向就非常明确了。

> **能不能让 AI 来对齐？能不能让 AI 来推进？能不能让 AI 来闭环？**

AI 原生组织转型 · 第 2 讲

模块二 · 用 AI 做对齐

什么是对齐

对齐：世界是什么，我们怎么做

对齐有两个含义，一个是对齐事实，一个是对齐打法。

对齐事实就是公司里面有这么多人，我们看到的世界是不是都是一个样子？你看到的销售额和我看到的是不是一样？你看到的市场机会和我看到的一样吗？我们看到的问题是不是同一个问题，我们理解的这个词是不是同一个词汇。所以对齐事实就是让大家对于当下是个什么样子、什么状态有一个共识，基于这张共有的地图，on the same page，然后我们再去思考我们要怎么做。

而对齐打法就是每家公司都有自己的特点：我们做事有什么规范，应该怎么做事，什么优先级，边界在哪里。这些其实是这家公司行为上的特点。在同一幅地图上，如果我们拿着的作战指令、作战的方法不一致，你是狂飙突进，我是徐徐图之，相互矛盾，也会造成无谓的损耗。

对齐只有两件事：**我们是不是在同一张地图上，以及我们是不是按同一套交规在开**。

> 地图不一样——你看到的销售额和我看到的不是一个数，你说的「客户」和我说的「客户」不是一个东西——这叫**事实没对齐**。 地图一样，但你狂飙突进我徐徐图之——这叫**打法没对齐**。

On the same page

为什么对齐很贵

组织越大，对齐的工作量越大

理论上来讲，需要对齐的线条数量跟组织的节点数量的平方成正比。你有3个人的时候，只需要三条线就可以对齐成功；你有50个人，就会有1225条，这是很离谱的一件事情。小公司之所以比大公司快，也就是这个原因——对齐的线条数不是线性地随人数增加，而是平方级地增加。

但是很多时候大家并没有感受到那么夸张，其实是因为我们会把公司切成非常多的部门、小组和项目组，让组和组之间干脆不用对齐，形成一定的独立性；而且中间也不会做完全对齐，会做非常多的压缩。

人类靠压缩来表达，靠幻觉来重构

因为对齐的量过于庞大，所以我们更有效的方式其实是通过有损压缩来传递我们的思考和上下文。

比如说一个设计师，他为了你的产品，脑子里过了100种方案，知道哪些不行、哪些可行；他真正尝试了3种，发现其中两种会遇到别的问题，最后给了你一种。但是他不可能把这些背景知识、他的学习、他的脑子全部给到你，他只能给你一个PPT，说我们选择了这种，因为它有三个原因。他其实进行了一次有损压缩，这样沟通效率才高。

而我们拿到他的东西的时候，其实也是在进行一次有损的还原。我们会通过想象来还原，还会想，诶，为什么他没有试过那个？其实他可能试过了，只是我们不知道。对于我们不了解的地方，我们就凭想象来还原他的设计。他在隐空间里思考的过程其实没有传递过来，传过来的只是一页PPT上的几

个字、一份PRD上的一堆文字而已，我们就会根据这个、根据我们自己的上下文来重构对这件事情的理解。

每一次我们重构的时候，塞进去的其实就是我们自己的幻觉。而这个幻觉会开始累积——传递的每一个环节都压缩、都损失，然后再由下一个人产生一些幻觉补充进去，到最后损失率就巨大。

如果看过快乐大本营里那种传话的游戏，一个人对着另外一个人的耳朵说话，再让他传给下一个人，一路传下去，最后已经不知道偏到哪里去了。这就是传统对齐的问题。

案例：安克

安克在分享当中提到他们新品上市的流程。原本的链路是建产品档案，然后做全球策略、做区域策略，然后区域设置定价，还要拉品牌营销、供应商备货、财务论证。整个流程里面可能十几个节点，有大几十个商业判断，都是靠文档、靠会议来传递。每交接一次上下文就少一层。

原话是：「**哪怕每一个节点我们都做到 99 分，那一百多个判断点叠下来，它可能损耗也会超过七成**」「最终执行层拿到的，可能已经不是最开始那个全球策略的全貌了，执行效果也会大打折扣」

这确实是这样。他说的是每一个节点都做到99分，但实际上99分是做不到的，我们的信息损耗其实会比这个还要更大。

AI 如何来对齐：把 N 变成 N

AI 成为唯一事实源头

刚看到那个N^2的图的时候，大家可能就很容易想到：不应该让点和点之间对接呀，应该大家都统一跟一个信息源对接，再由它去对齐所有人，不就好了吗？这不就把一个N^2的问题变成N的问题了吗？

这样所有的线都不需要相互连了，都连向一个中央就行。举例来说，如果前面讲的那个传话游戏不是一个人传另外一个人，而是每个人都对着一个中央的广播电台说，那个广播电台再播给其他所有人、把短信发给大家，那所有人不就很快能拿到一手的信息，而且不需要同步那么多次了吗。

比如说我们现在讨论一件事情，不要每个人去单聊别人、一对一开会，而是所有人围绕着一个共享文档工作，不就方便了？根本没有那种很费平方的麻烦，是吧？

别让人和人对齐，让人和同一个东西对齐。

为什么必须是 AI：存取方便的活水

刚刚讲的那个，好像共享文档也可以做到，但你仔细想一想，如果一个项目足够大，你的文档就会有两个问题。一个就是大家需要大量地对这个文档进行更新。

看看你们公司的知识库。在AI之前，有多少人愿意往里面贡献知识？有多少人会主动去维护它、更新它？甚至包括公司政策之类的文档，有多少人会积极地更新呢？政策还算是很久才改变一次的，如果是每天要改的东西，那有多少人会主动积极地把它保持在最新的状态？很少。

但有了AI之后，AI只需要沉浸在所有的会议和其他文档当中，它可以自己来收集信息，保证自己是了解全局的。这有点像是给每个人配了个助理，就轻松多了。输入上，原来人的输入意愿和输入能力是瓶颈，现在AI打开了这个瓶颈。

而另外一方面，你想一下，那个文档如果每天都在更新，而且越来越大，有谁真的看得过来呢？你明明知道自己做这件事情的时候要跟这份137页的文档对齐，但是它里面哪些改动跟你有关、它昨天更新了什么，看起来都很累。所以每天去看它是个瓶颈，去理解它又是个瓶颈。现在AI可以站在你的角度告诉你哪些东西跟你有关，它把这个瓶颈也打开了。所以这个唯一的事实源才变成一个源头活水——更多的活水流进去，也更容易从里面把水取出来，它才能够有用。

这也就是为什么每个公司都号称有标准流程、有知识库、有文档，但是大家真的要办什么事儿的时候，不管是想要了解情况还是了解流程，你还是选择去找“熟悉这事儿的老王”。因为你知道他脑子里东西是新鲜的，而且你问他比较省事儿。AI 以后就是这个老王的角色。

共享文档不是不好，是它有**两个瓶颈，都在人身上**：
写不进去——没人愿意维护；**读不出来**——137 页你不知道哪三行跟你有关。
AI 同时打开了这两头：**它自己去收集，它按你的角度递给你**。

你们公司有没有标准流程？有。有没有知识库？有。那你真要办事的时候去哪儿？**去找老王**。
为什么？因为老王脑子里的东西是**新鲜的**，而且**问他比较省事**。
——所以你们缺的从来不是文档，是一个**随时更新、随问随答的老王**。而这正好是 AI 能干的事。

对齐事实

什么是对齐事实

对齐事实，就是让每个人看到的世界是一样的，整个公司on the same page。就有点像一个军队里面，每个联队看到的地图要是一样的，对于我们的位置、敌人的位置、路线，看到的都是一样的。

有效对齐：操作同一个对象

我们一讲到对齐，往往就会想到要加强沟通、有效沟通。但实际上最好的对齐是不要沟通，尽量最小化沟通。这不是我们现在才想到的事情，之前杰夫·贝索斯，就是亚马逊的CEO，其实也讲过一样的话：Communication is a sign of dysfunction.

不要沟通。

沟通的意思就是你脑子里有一份东西，我脑子里有一份东西，我们指望着靠自己的嘴巴讲清楚，也期望对方能够理解到，然后把两个脑子里的东西统一掉。这几乎是不可能的事情，效率极低。更有效的做法，不是我们各自有一张地图、靠嘴巴把它沟通清楚再对齐到一起，而是我们本身就在同一张地图上操作，我们操作同一个对象——这样的对齐才是最高效的。

沟通是在同步两个大脑；共享对象是在同步一份状态。

前者靠嘴，后者靠系统。**只要还得靠嘴，就一定会漏**

案例：波音 777

波音很早就用 CAD，但是……

举个例子，波音公司造飞机，他们之前是用手绘图来设计飞机的。你想想，莱特兄弟发明飞机的时候还没有电脑绘图嘛。后来有了电脑制图之后，像波音他们其实是用电脑在画图，这样显然可以极大地提高完成这个任务的效率，单个任务的效率是可以得到极大提升的。

但是他们画完图之后，一个部门会把这张图纸打印出来给到另外一个部门，另外一个部门靠肉眼看、凭经验去理解，再根据这张图做自己部门的活。比如说一个部门做门，另外一个部门做的是门框，做门框的会跟做门的这边开会讨论，拿着那张纸质图纸研究思考，然后再开始做门框。所以据说767一个舱门，来来回回返工了13000多次。

每个部门都数字化了，整架飞机还是纸上对齐的。

单个任务高效，对齐带宽是瓶颈

为什么需要返工呢？其实就是因为很多地方靠的是我们刚刚讲的那种有损压缩的表达——一张平面上的图纸，凭讨论和感觉来理解，这样会损失掉大量的信息。所以波音公司才会一个舱门返工13000次，才需要用看起来很傻的办法，用木头把各种零件做出来，再做实体飞机的模拟，看看能不能拼得上去。以及一架飞机上不同的地方，可能毫无必要地用了300种螺丝。

大家想一想，比如说100个部门要沟通，那就是99次基于纸上的传递和有损压缩，怎么样可以让大家更有效地工作呢？既然已经是CAD了，怎么让大家更有效地对齐呢？

一架飞机上，**同一个功能的位置用了几百种不同的螺丝**。没有人做错任何事——每个部门都按自己的图纸选了最合适的那一种。**问题是没有人看得到全局**。

波音 777：统一数字模型

现在看起来答案一目了然。你既然已经是数字化地在做飞机了，不是应该所有人都在云端操作同一架飞机吗？你做的是门框，我做的是门，那我们就应该在云端把这个门和门框对起来做呀——我做门框的时候，时时刻刻应该看得到你这扇门。

我们就应该在数字空间里面看到所有的相互依赖关系、所有的协调关系，这样才是真正的对齐。所以真正有效的对齐，并不是加强我们那种低带宽的沟通、把那张图画得更漂亮一点，而是我们最好不要沟通——大家都对着同一个数字空间工作，这个数字空间里面会把所有的依赖和矛盾暴露出来。

波音777 后来统一用 CATIA 做设计，**100% 数字化设计、用三维实体技术定义**，返工量减少 50%

案例：会议纪要

今天的会议纪要，就是当年的打印图纸

刚刚听那个波音的故事，你是不是觉得波音有点傻？都已经用CAD这种数字工具了，怎么就想不到在云端把它统一一下呢？为什么还需要传打印出来的纸呢？实在是太傻了——像我们就肯定不会这么傻。

觉得自己公司不像波音公司那么傻的，举手。

最近开过会的，继续举手。

会议纪要和会议纪要是打通的，如果两个会议有矛盾信息，系统会告诉你的，继续举手。

举手的同学，我想问你一下：你们最近有开会吗？开会有记会议纪要吗？你们的会和会议纪要之间是打通的吗？如果一个会议提到一件事情，跟另外一个会议矛盾，你们看得出来吗？

你的会议纪要，就是波音打印出来的那张图纸。

单独的会议纪要 ≈ 打印出来的 CAD

比如说，你们公司想要做一个创新的AI项目，叫做Mars火星计划。你们关于火星计划的会议已经开了6个，今天开第7个。那前面6个会议，是有6份单独的会议纪要？还是这6个会议里关于Mars这个项目要做的A、B、C、D这些事情，都写到了一个统一的文档、一个统一的状态里面？

如果是6份单独的会议纪要，那是给谁看？是给人看吗？给人看了，让他用脑子理解，然后去开下一个会——这跟波音公司把图纸打印出来有什么区别呢？

比如说你们这个Mars火星计划，一个会议上说，我们要做AI智能耳机，要把那个声音做大一点；另外一个会议当中说，我们要把那个做大音量的供应商给去掉，因为觉得他们态度不好。这两个会上发生的事情，单独看都是合理的。那我们的会议纪要当中能够看出这里面的矛盾吗？看不出来的话，那我们的会议纪要，本质上跟做门和做门框中间要通过一张图纸去理解、反反复复返工，其实是一样的。

注意，这两个决定**单独看都是对的**，而且**都在纪要里写着**。
冲突不在任何一份纪要里，**冲突在两份纪要之间**——而「两份纪要之间」这个地方，**公司里没有任何一个岗位负责**。

每一份都对，合起来是错的。

应该更新云端统一对象，不是会议纪要

你们有一个关于项目的文档，是不是应该所有人都去更新这个文档？我们应该更新的是云端统一的这个数据对象，而不是一个单独的会议纪要。

我们很多的工作方法其实都是古代的，就是习惯了——因为以前那套东西号称是有用的，但其实更新很麻烦，大家也不看。但是现在，反正是AI帮你写会议纪要，AI帮你去更新文档，也是AI帮你去读文档，为什么我们不这样子来操作呢？

从今天起，会议结束的产出**不是一份纪要，是对那个项目对象的一次修改**。
「螺丝从 3 号改成 4 号」这个结论，不应该写进一份叫《8 月 23 日耳机项目会议纪要》的文档，**它应该直接改掉云端那只耳机**。

纪要的终点不是记下来，是改掉那个东西

案例：出门问问

我听到的做得比较极端的案例，其实是出门问问。他们每个项目会有一个自己的Agent，上下文、进展、待办都是那个Agent负责管理。甚至有一段时间听说李志飞说不准人与人之间直接开会，有什么会都要通过Agent来开，要让Agent来统管上下文、理解所有上下文。

讲的还是所有人都应该统一在一个数字对象上工作，而不是人与人之间靠那么大量的沟通来对齐。

案例：安克

安克也是这样，他们会给每个项目一个项目ID，每个项目相当于有一个容器，里面会准备好所有的数据，所有人都是在这个项目容器里面去取数据。他们的原话是，项目是主线，所有事围绕它发生，而不是围绕着某一个人的飞书消息发生。

而因为所有跟项目相关的数据、信息、思考和决策，都在这个项目容器里面，所以当它改变了什么的时候，所有人工作的那个对象其实就发生了改变，这样对齐就方便了。比如说他们把其中一个螺丝从3号改成4号，那不需要跟任何人沟通，所有人就会发现自己手头对着的那个项目当中已经从3号改成了4号；你如果是做螺母的，就会发现原本的螺母拼不上去了。这件事情马上就会暴露出来，而不见得需要再拉一个会、需要大量的沟通。

「项目是主线，所有事情围绕它发生，而不是围绕某一个人的飞书消息发生。」

案例：Claude Tag

更科幻一点的其实是Anthropic的Claude Tag。它看起来很简单，就有点像是你邀请了一个AI到你的飞书群里面，或者到你的微信群里面。但是实际上，它会读你所有的信息——读项目信息，读你这个具体工作组的信息，读你们的聊天记录。

它把所有的信息都拿到了之后，会主动地识别里面有什么是它可以做的，主动识别跨项目之间、跨会议之间有什么冲突的地方，甚至主动参与，跟你说：哎，上次那边说要把螺丝从3号改成4号，那你这边的螺母就配不上去了。它会主动来识别这个。

出门问问和安克，是**把大家赶到同一个对象上**。Claude Tag 更进一步——**它自己同时在所有场子里**。
这件事人做不到：**没有一个人能同时坐在你们公司所有的群里**。 但 AI 可以。
所以它能看见一种谁都看不见的东西——**跨场冲突**。

有效对齐：不要光靠语言

讲的其实就是大家不要各自有一套东西，然后我们要操作同一个对象。

另外还可以补充一个也很有用、大家也普遍在用的、对齐旁边的小战术，就是用成品来沟通，不要光靠语言。很多时候我们不可能在一个数据对象上面把所有东西都写清楚，靠文字把所有东西都沟通清楚。所以很多时候不对齐，并不是没有操作同一个数据对象，而是我们操作同一个数据对象的时候，你以为的是一个意思，我以为的是另一个意思——我们看到的是同一个东西，但是我们理解的不是同一个东西。

所以更有效的对齐，最好就是直接把最终成品的样子给做出来，不要光靠语言在对齐。文字是非常有歧义的东西，一千个人有一千个哈姆雷特；但是东西长什么样子、能够怎么用、用起来是什么感觉、用出来什么效果，这个东西是大家可以调动所有感官去理解的。所以早期对齐的时候，我们应该直接把东西做出来，让大家都可以看到、摸到、玩到，这样所有人不光是对齐了文字，更是对齐了理解。

过去做不到是因为生产成本太高，比如说刚给你开一个会就要把产品做出来，怎么可能呢？但是现在有了AI生成，这件事情完全就是可能的。现在我们平均出一个产品的速度大概也就是十几二十分钟，如果是软件产品的话。

硬件产品当然不可能这么快，但是如果你只是需要出这个硬件产品的效果图，甚至于出一个这个硬件产品使用的宣传小视频，其实也无非就是20分钟、30分钟的事情。如果要把它打印出来，让你实体手上可以看到——我们在户外企业做过很多这方面的东西——大概用3D打印把它搞出来，也就是个半天一天；如果说还要再拼一些东西，就会从华强北或者网上淘宝上买一堆零件拼起来，硬件的prototype大概半周也就可以拼出来。

过去我们靠文档沟通，**不是因为文档好，是因为做出来太贵**。文档是**成品的替代品**，是一种妥协。
现在成品变便宜了——**妥协的理由没了，妥协的习惯还在。**

文档是成品太贵时代的替代品。

案例：Botlearn

比如说像Botlearn，我前一段时间跟他们在AI炼金术聊天，他就说他们公司早就改了规则：现在产品或者研发想出来什么idea，都是直接做出来、直接上线，然后找到目标用户人群里的一小撮，直接做实验做MVP；做好了之后发现确实可行了，再去找设计师把它做成好看的样子，再去走流程融入到产品里面。相当于是直接把它给做出来，并且直接投入市场测试，就一个人全部做完决定。 **研发先做出来 → 直接上线 → 找一小撮目标用户跑 MVP → 确认可行 → 这时候设计师才介入**，而且**设计师直接改代码**。

案例：Orion

我们也孵化了一个to B做AI转型的公司，他们现在更极端，是把这一套打法用在了谈客户上面。现在经常是跟客户——比如说是一个石油公司——一边谈，电脑就一边在跑，开了半个小时会之后，就直接

跟对方说：哎，王总，我们已经做好了。您刚刚提的点子我们都觉得特别有启发，加深了我们对行业的理解，这边是刚刚我们用AI帮您做的8个产品原型，您可以看一下。

然后线上就跑起了8个已经能用的产品。里面用的都是假数据，但是上面的功能都是可以点的。

这样子，一方面可以震撼一下客户，就是说你看我们多么AI原生、多么了不起；另外一方面，提高成单率；还有一方面，也是为了让对方感受一下到底要做的是个什么东西。然后他就会说：第三个是我想要的，我想要第三个好久了；第四个其实不是我的意思，我再给你讲讲。这样子沟通效率会更高——大家对着同样的一个可见、可摸、可感知的东西，对齐效率比纯粹的语言沟通要高很多。

> 客户第一次不是在**评价一个方案**，他是在**使用一个东西**。
> 而人对着一个能点的东西，说出来的话完全不一样——他不会说「挺好的，你们再完善一下」，**他会说「第三个就是我要的，第四个不是我的意思」**。
>
> **你不是提高了成单率，你是把一轮为期两周的需求澄清，压进了三十分钟。**

别问他想要什么，给他八个能点的东西。

对齐打法

什么是对齐打法

为什么需要对齐打法

除了对事实的认知要统一之外，我们也要在行为方式上产生共识，这样才方便协调，才能让大家做事做到一块去。最简单的例子就是我们门口的马路，大家都看到它，但我们还有一个规则叫靠右行走，对吧？所有人都靠右，就不会撞起来，很简单。但如果没有这条行为规范上的共识，很可能马路大家都看到了，可是有的人靠左开、有的人靠右开，马路就堵起来了。

对齐，不容易

传统上要对齐打法，首先就是靠培训嘛，大家都要学会我们公司的规范、我们公司的打法。请像我这样的人过去讲堂课，大家鼓掌，讲得太精彩了，然后回去全忘掉。第二种是靠制度，写好多好多制度，写得特别精彩，贴在墙上，根本没人看，也没人管。第三种是靠KPI，定一些过程指标、结果指标，你们要按照这些指标来做，然后大家就想办法把这些KPI做到——但给你设计这个指标的目的，跟你实际执行的那个动作，对不对得齐？那就不一定了。

招	失效在哪一层
培训	讲的时候懂，**做的时候不在场**

招	失效在哪一层
制度	写下来了，**执行的时候不在路径上**
KPI	在路径上了，**但它只管终点不管过程**

案例：抖音广告

我最近在抖音上经常看到一种广告，会有两个人出来，一个人对着观众说：早几天全价200块钱买了某某某的同学，现在一定已经哭晕在厕所了，一定后悔死了，今天大降价，只要多少多少块钱。这种广告，如果我是这个品牌的老板，我一定会气死——这是赤裸裸地在教育用户，不要在原价的时候买东西。这种广告我已经看到很多次了，说明它在转化上是有效的，但它确实在伤害品牌。我估计品牌方给他们的KPI是按效果结算，这么算下来转化效率确实高，但对长期有伤害。所以这个对齐就会很困难，甚至有可能是广告投放的部门和供应商合谋的，因为这样可以把KPI做好看嘛。

这条广告的转化率一定是好的，**否则它不会被投这么多次**。

> 品牌方的 KPI 是转化，代理商的 KPI 是转化，**双方都完成了 KPI。 唯一受损的是品牌资产——而品牌资产不在任何人的 KPI 里。**

案例：K12 电销

这种好歹还可以说是外包，但对内部来说也是一样。我最近有一个做K12的公司创始人打电话给我，让我帮他推荐AI的电销团队。他们已经有很多人买了试听课，他需要打电话去催这些买了试听课的用户转正价课。我就帮他介绍了一个很靠谱的真人电销团队，因为我们刚好在合作，在做一些AI转型的事情。他很客气地拉了个群，也很客气地联系了一下。后来又给我打了个电话，说你还能不能给我介绍AI的。我说这家是按效果结算的呀，你为什么一定要AI的？他说不行，他们内部其实也有这种电销小团队做转化，也是按效果结算，但是他们最后总会发展出一些我不想要的话术，有些甚至会让我不合规，给我制造风险。我现在需要的是既要结果，也要控过程，而这件事只有AI能做到，人做不到。想了想确实是啊——你跟人家讲我按这个KPI给你奖金，他一定会想办法把这个奖金拿到嘛。但在这个过程当中，你如果又把过程定得很死，说第一句讲这个、第二句讲那个，既要过程又要结果，这种老板是非常讨厌的，真的，你很容易被人打。这种要求，只有AI会说好的。

「我既要结果，也要控过程。」
这种自相矛盾的要求，只有 AI 会答应。

有效对齐：把规范写到环境里

所以，希望靠人和人之间相互沟通来对齐打法、然后自觉服从，是很难的。最好是把它变成一个可以按的按钮，一段可以运行的代码，一个在后台自动检查、会帮用户自动改到合规的东西。总而言之，

尽可能让用户无感、无成本，这样我们才能在打法上做得更好。

案例：Botlearn.ai

有个很简单的例子。早几天我跟BotLearn在AI炼金术录了一期播客，俊东就讲，他在做他们公司的营销，但他们又是一家比较贵族范、比较重品牌的公司，很重视品牌的各种守则。所以他要出一些东西，比如出一些宣传文档，以前都得跟设计师沟通。现在是设计师把他们所有的品牌元素、品牌规范全部写成一个skill，他直接调用这个skill，一次性就会按照品牌规范出所有的营销内容，不会有那种偏差。这比你要求大家对着一本册子去逐条检查，要靠谱多了。

最高级的专家写一次，所有人反复调用。

Ramp CPO：我说过 1000 遍 CTA 一定要在首屏，现在终于可以做到了。

案例：Ramp

Ramp的CPO分享了一个点。他说如果你要提高转化，就是要把call to action，就是行动召唤的那个按钮，一定要放在首屏，放在首屏，放在首屏。他说这是他6年做AB testing得出来最重要的经验。但是他们公司自己老是做不到，下面发上来给他看的设计图，那个大的行动按钮就是没有放在首屏。他说他念了100遍、1000遍，公司还是不一定做得到。但现在就能做到了，因为他们做了一个agent叫Inspect，嵌进了他们所有的工作流里面。在Figma里工作的时候，它会自动把这条工作规范加进去，所以他后来看到的，只要是那种详情页、着陆页，行动按钮一定是在首屏。

而且更进一步，你都不需要记得去用它，它会自动来改你做的东西。

案例：ZooWork

还有一个例子是我AI炼金术的搭档。他自己在创业，做的是那种小直播，他们公司早就全员Vibe Coding，都是用AI写代码。但是就会发现，过程当中有非常多的规范是没有满足的，尤其是像我这种外行开始写代码。中间比如说有一件事叫画图，用SVG画图——大家现在在有些网站上看到的那种三角形、正方形，或者一只小熊猫走来走去，其实都是SVG画的。他说他们公司原本每个人都在用自己的AI去画，而且画的方式、写出来的代码都非常丑陋。按传统的方法，就应该让大家培训嘛，统一规范，用统一的库来做，不要重复造轮子。但他们用了一个更AI native的方法：大家各自爱怎么写怎么写，交东西上去，交完之后AI会在背后看一眼——哦，你这是写SVG的，那这一段我按照规范去调用统一的库，再把你拼进去。这样就没有支付对齐的成本，但是享受了对齐的结果。

没有支付对齐的成本，但是享受了对齐的结果

案例：艾语智能

我听到最极端的一个案例，是艾语智能。我跟天乐好好地聊了三次，他们公司有一个非常神奇的打法，就叫做跟Anthropic对齐。他觉得Anthropic既然是世界上现在最强的AI公司，那他们肯定比我们看得深、看得远，那我们就学习他们的做法。所以他们公司做了一个Agent，专门去网上监控Anthropic发表的一切言论，最新的blog呀、最新的发言，然后去理解他们的做法。据他说，Anthropic前后其实

非常一致，是有一套明显的组织逻辑在里面的。所以他们内部要定什么打法的时候，就会跟这个agent商量：我们打算公司往这个方向改、往那个方向改，可不可以？符不符合Anthropic的想法和打法。整体上来讲，就是他们在用一个Agent去蒸馏Anthropic的组织方法论和思想，然后跟它来对齐——他们所有讨论组织方向、怎么改公司的话题，都是去跟Anthropic对齐。

时间和起因：大概去年底今年初做的调整，起因是「**概念实在太多**——各家的 harness、各种框架满天飞，讨论不过来」

> - **做法**：做了一个 skill，**定期抓取 Anthropic 高管和核心的文章做蒸馏**，过滤掉少量矛盾内容（「能过滤掉的非常少」），沉淀成公司的架构标准。**这个 skill 到现在调用频率还非常高**
> - **你自己在节目里的总结**（可以直接引用你自己）：「**你不是全面对齐 Claude Code，你是把公司对标了 Anthropic。**」
> - **张天乐原话**：「到后来讨论也不讨论了，**他说干咱就干，他说不干咱就不干，一定要让公司减少无效的讨论。**」

总结

所以总的来说，AI可以让我们更方便地对齐事实，也更方便地对齐打法。对齐事实，主要是AI可以成为统一的事实来源，让所有人操作同一个对象，减少不必要的人与人之间的沟通；而且AI可以让我们更早地看到实体的成品，而不再是含混的概念沟通、产生大量误解，东西看得见摸得着，理解就更容易对齐。最后，AI还可以帮我们把对齐的规范和规则写到环境里，让我们按一下按钮，甚至连按钮都不用按，它就自动对齐到整个大流程里，这样可以显著地提高效率。

对齐事实 → **别再互相沟通，去操作同一个对象**
对齐打法 → **别再提要求，把规矩写进环境**

最好的对齐，是不需要对齐

AI 原生组织转型 · 第 3 讲

模块三 · 用 AI 做推进

什么是推进

推进，就是把决策编译成可以执行的工作

推进就是，你公司会有很多决策，会有很多事要做。那把这些事怎么拆成一个一个具体的工作、具体的活，安排给谁，什么时间做？谁做完了之后质量怎么检查？这些整体上就是推进。很多公司其实会

发现，推进才是难点，所以会有PMO之类的工作，它其实就是我们如何把决策翻译成行动，如何把行动落实到人、落实到时间线上，这就是推进。

对齐解决的是「**大家看到的是同一件事**」，推进解决的是「**这件事真的会往前走**」。

组织最大的损耗，发生在任务之间

我们经常想要优化完成一个任务的时间效率，但一个组织真正的推进效率，取决于任务和任务之间。在任务发生之前，他怎么开始第一个动作，以及任务之间的传递，以及最后的那个收尾，这才是最难的。

大部分活真正的损耗，都卡在「我们开会再碰一下」「我们再研究一下」「下次我们再拉上谁一起碰一下」，都是这种事情上面。

> 你回想一下上一件卡了很久的事。它卡在哪儿？
> **不是卡在谁不会干，是卡在「我发出去了，等他回」。**
>
> 方案做完了躺在群里等产品，产品看完了躺在邮箱里等法务，法务看完了躺在系统里等采购。
> 而且还不是白等——每一站都给你一点反馈，然后「**我们再碰一下**」「**下次拉上谁一起看看**」，就没完没了。

一句可上屏：「活不是在被做的时候慢，是在被等的时候慢。」

公司最贵的人，常常在处理推进问题

我们请来很贵的人，表面上都是为了他的专业能力、领域知识，但实际上我们需要的是他的管理能力和推进活的能力。大部分时间，就是当小朋友们遇到了问题、遇到了卡点，跟部门协调不下去，谁谁谁不反馈，老是拖延的时候，这个时候就需要很贵的人、需要老大们用权威来往前推，来做一些妥协。那这个人其实是靠他掌握更多的上下文、敢做决定和权威在推。

公司里最贵的人，一半时间在当拖车。

用 AI 做推进

AI 更知道全局，也更耐烦、更敢催

AI更了解上下文，也更耐烦，它也更敢去催，没有那么多的政治压力，看起来像一个公正客观的第三方。所以让AI来扮演这个所谓PMO的角色来负责推进，其实我们效率是更高的。

Agent 去推动流程， 而不是人去用 Agent

在我们做AI的流程改造之后，这件事尤其明显。因为老流程你推进我推进都差不多，大家都一样的熟；但是如果你要推一个新流程、新方法，比如说你用AI改造了整个Workflow，希望大家按照新的这个流程来走，那么如果不靠AI推进，人是很难去执行的，慢慢就滑回到原来的位置上。

甚至大部分的公司，包括不是改流程的事情，仅仅是让大家去用Agent、把Agent用起来这件事情，如果你前前后后不用Agent来推动，给他提供足够的上下文，催他把它用起来，大家用着用着也就会退回去。所以要用Agent来控制整个的流程，Agent为人安排工作，然后催人去用他的Agent。而不是你做一个培训去让大家一定要用新流程、用新工具，而是用新的流水线催大家一定要用新的做法、新的工具。

让人去用 Agent ＝ AI 是可选项。你忙、你赶、你出错的时候，**最省事的动作永远是按老办法来**。
让 Agent 去推流程 ＝ AI 是默认项。你不走这条路，**活根本不到你手上**。

而**可选项永远干不过默认项**

案例：安克

这边就有安克的例子。安克之前做了很多的流程Agent，然后让大家用，每个人都说好，培训的满意度巨高。但是过了一段时间去检查，发现那些Agent根本没人用，大家又退回到老的做法。所以后来他们就反思，不应该是让人去用Agent，而应该是用Agent去推动流程。用了这种做法之后，大家才真的转到新的打法上来。

- Marketing 第一批流程智能体上线，做了好几轮培训、**几百多人上课，课程满意度 8 分**。培训负责人把评分发群里，几个智能体创建人**疯狂点赞**，有人说要给满意度最高的几个智能体**加鸡腿**
- **两周后翻后台**：**超过一半的流程智能体，使用人数不超过 5 个人**——讲者自己解释：「相当于用过的人是**建设者自己，再加上身边一两个同事帮忙测了一下**」；另一批是**用了一次就再也没回来**
- 原话：「后台冰冷的数据给我们每个人都**泼了一盆冰水。没错，不是冷水，是冰水**。」
- 1. **项目的动态上下文没被管起来**——通用逻辑能内置进智能体，但「这款耳机的卖点、打哪类人群、竞品最近什么动作、这次花多少钱」内置不了；没有统一存放的地方，**每个人靠自己拼，拼得全不全全看个人能力**
- **智能体之间没有协作关系设计**——洞察智能体跑完产出报告「**它停在了那里**」，策略智能体不知道它的存在，**要靠人工搬运**
- **全链路运行状态不可见**——「每个建设者的视野就是自己的那一段」

发现，分配，传导，检查

具体来讲，我们如果要用AI来做推进，要做的事情就这么四件，发现工作、分配工作、传导工作和检查工作。

发现工作

发现工作：让工作自己举手

发现工作是做项目推进的第一步，就是知道有什么活要干。原本这件事情也是依赖人类来思考、来创造的，人类要先发现情况，再分析情况得出结论，然后把工作安排下去。但是人类的工作是有周期的，所以发现工作往往就不及时。如果我们用AI来发现工作，应该是让AI来全量地看所有的情况，由它来分析、它来总结、它来发起工作，这样的效率才是最高的、最及时的。

案例：海底捞

比如说海底捞。海底捞店员的平板上面会看到很多桌子，那个桌子是会自己报状态的，会说我现在是被占用还是空闲，还是需要整理了，桌子会自己说。那桌子为什么会自己说？是因为有摄像头看着所有的桌子。

这就比原本对每个店员的要求低了很多。原本对店员的要求会很高，要求他眼观六路耳听八方，去时刻观察哪边需要整理了，哪边来了客人需要去干嘛了，他得自己去判断。现在把这个观察和判断的活收给了中央，收给了摄像头和平板，对每个人的要求就降低了，工作也会安排得更及时。

- 「以前对店员的要求是**眼观六路**；现在店员只需要**看一眼平板。不是人变强了，是判断这件事被搬走了。**」

案例：美的

工厂里面很多初级的管理岗，其实每天做的事情也就是在发现问题、发现工作。比如说美的的工厂那边，之前就需要有人去巡检，每两个小时到大屏前面去碰一下；现在是机器人巡检识别问题，机器人自动生成工单，派工单去解决。

我们会发现，现在很多这种协调型的工作也是用人来完成的，用人去看情况，用人去发现问题，人去定义工作，人去找另外一个人去解决问题。那么这些活现在都应该交给AI，把这个活与活之间的那个缝隙给填掉。

案例：安克

刚刚两个讲的都是实体的传统行业，为什么呢？是因为传统行业的工作和问题发生在物理世界，比较容易被看出来。但是数字空间不一样，比如说一个员工对着电脑，他到底在干嘛？你其实不太知道。而现在，我们也可以很快地让AI来看数字空间里发生的可能的问题，让它实时地发现工作、实时来解决。

比如说安克，假设他们发现某一款产品上周销量下降了23%，AI会自动识别到这个异常，然后就去分析，到底是我们库存不够了？还是流量不够了？还是广告没投好？还是因为我们的评论不好、商品质

量不行？然后它就会自己去调取库存信息、调取广告信息、调取评论信息，以及去社交媒体上去找，看到底是不是有了我们的差评，再把这些东西统一打一个包，发给相应的负责人员。

安克的人在介绍这一段的时候，还会特意解释说，其实理论上来讲，它应该直接生成一个工单就自己把它处理了。比如说发现有一个广告方面的投放错误，那就应该自己处理了。但现在他们还没有做到这么AI原生，不过已经很先进了。

传统的话，我们往往要等到日会甚至周会的时候，大家拉数据，一起来碰一下，发现诶销售下降了，我们再去讨论说为什么。然后中间就会有一个人认领说，我回去看一下可能是哪几方面的问题，但是他手头没有数据，于是他再回去调数据，再来分享，再定方法，再去行动，这就慢了。

但是AI可以让第一时间就去查，看到问题就直接把数据搞齐了。哪怕它最后还是把这个活交给一个人，它也交得更快一些——它交的时候已经把上下文，比如说我们社交媒体上的评价是这个、我们最近库存是这个，把这些相应的资料包已经准备齐了。

① **数字给全**：安克的说法是**亚马逊某 SKU 环比下跌 23%**。传统做法要**跨看板拉数据 48 小时到两周**——而且这句更狠：**「更多的情况是你连问题都没找到，过段时间恢复了，吃了个哑巴亏。」**

以前是周会上才发现，现在是数据掉下去的那一刻，材料已经备齐了。

案例：AI Jason

还有一个案例是说，他们会用AI来做SEO和SEM的投放，如果发现哪一个关键词的效果特别好，但是他们在这边的自然流量又比较小，他们就会自动开始去铺SEO相关的内容，希望增加SEO的流量。那么这样子其实就是说，不光可以发现哪里有问题我去补，也可以发现哪里有机会我们要上，这都是可以让AI来发现的。

广告 loop（Giorgio 的一个月实验）

- 给 agent 配技能包（分析表现 / 写文案 / 生图 / 调研）+ 一个 `state` 目录记 changelog 和 learnings + JSON 记 campaign 历史
- **第一周自主测了 10 种广告形式**（白板手写、笔记本页、纸板牌、推文截图）→ **学到「丑素材反而赢」** → 第二周自动收敛到白板 + 特定文案结构
- **1500 美元预算换 243 条线索，全程一个月无人干预**

你要的那条正是这两个 loop 之间的跨界

> 广告 loop 发现**某个关键词点击率好、但没有自然内容——这条 signal 自动回流进 SEO loop，被排进优先级。**
> **广告部和 SEO 部之间，没有开过一次会。**

分配工作

分配工作：让工作自己找人

让工作找人，而不是人找工作

传统来讲，我们总是希望人自己在想我应该做什么活，但是实际上更有效的是让适合这个人的活走到他的面前。否则的话，人类就要浪费大量的时间去思考我到底要干嘛。大家但凡有过那种假期要自己安排自己的生活、而没有一个工作节奏的经历——像我现在就大部分时间是自己安排——你就会发现，经常是雄心壮志定了一个大计划，但是啥事都不会发生，反倒不如在办公室里效率高。

人不擅长找活，人擅长做被递到面前的活。

案例：美团外卖 & 滴滴

这件事情对于每个人来说都是一个常识。你自己想一想嘛，我们已经接受了美团外卖，也接受了滴滴，我们都已经接受了这种非常高效率的活的安排。那你显然也看得出来，就是应该通过一个中央的算法去调度不同的人去干不同的事情，它会找到最合适的司机、最合适的外卖员去送哪一单，这样子是最高效的。肯定不可能是中央有一个调度人——哪怕你用AI去辅助他，他也不可能做这么复杂的一个运算，不可能调度得这么合适。

所以分配活这件事，我们在大规模的这种蓝领工作当中已经看到了极好的效果，只是在白领的工作里面，我们总觉得白领不一样。但是实际上大部分白领的日常工作，其实就是认知型的、用电脑的蓝领。那这部分如果用AI，效果会好很多。

美团 2024 年即时配送日订单峰值 9800 万单，有接单收入的骑手 745 万人——一个算法，调度着七百多万人。

我去给美团外卖讲过AI原生组织的课程分享。我当时就跟他们说，其实你们已经是最AI原生的组织了，只是我们习惯性地觉得管理外卖员那边的算法不能用在内部，其实把它往那边挪就好了。

优先让 AI 完成，才能促进转型

总的来说，分配工作这件事情，就应该让AI来进行分配。首先应该让AI来定义，为了完成这个目标需要做哪些工作；然后再让它去定义，这些工作里面交付物各自是什么、需要什么样的能力；然后再让它去看，哪些事情应该用AI来完成。能用AI的尽量用AI，不能用AI的可以跟人类讨论是否能够用AI、有没有现成的工具；如果实在不行的话，再跟人类一起完成，完成了之后再让它去思考，这件事情以后能不能让AI做。

默认顺序反过来：AI 先做，人处理例外。

案例：Codebuddy

听说CodeBuddy——腾讯研究院出的一份报告里面就会写——CodeBuddy的程序员每天并不是老板给他派活，而是AI会根据这个程序员的历史状况，看你原本是干嘛的、适合干嘛，然后给他几个活让他挑，他自己选择自己能干的那一件大事，然后自己去干，干完了交活就好。所以也不需要每天自己去想，AI把这些都想好了，人类就是你可以反馈、你可以提意见。

但是他们也稍微有一点点不同，就是并不是给人分配任务。他们原话是，早上打开电脑，没有人给你分配任务，但AI已经推荐了几个跟你专业领域匹配的待解问题。为什么是待解问题而不是任务呢？因为明确的任务，其实现在AI自己都能搞定了；但是如果是一个课题，需要研究、需要人类的判断力，它才需要人参与进来。然后人类参与进来把它拆解好，根据自己的经验安排好之后，再让AI去做。

> 明确的**任务**，AI 自己就干完了，不用给人。
> 留给人的是**问题**——需要判断、可左可右、没有标准答案的那种。
> 所以你会发现一件事：**AI 不是替人干活，是把「活」和「题」分开了。人从此只做题。**

一句可上屏：「**AI 把任务做完了，把问题留给你。**」

案例：酷特智能

不仅仅是这个行业已经这样，其实传统行业早就已经用这种思路了。除了我刚刚讲的滴滴啊美团之外，像服装工厂，酷特智能就是做衣服的。传统来讲，也应该有那种工作安排，说谁应该干什么，对吧？需要有班组长是吧，之类的，那就是人肉的组织，去把活安排下去，然后每个人再用自己的小肉脑袋去思考应该干什么。那他们现在，其实都是衣服上带RFID了，带了RFID的衣服到你面前，就会有一个Pad告诉你，这件衣服应该对它进行什么操作。这样是可以实现大规模的柔性生产，但是也不需要让每个人去具体地做那么多的决策，认知负载也降下来了。

蓝领的世界里面其实早就做到了让机器来为人安排工作，早就被认为是有效的了。只是这件事情现在慢慢要侵蚀到我们的白领工作里面。

「蓝领世界早就做到了，只是慢慢侵蚀到白领」

「——而且是从最像蓝领的那部分白领开始的。」

传导工作

传导工作：让工作自己交棒

因为工作往往不是一个人完成嘛，所以在分配完了之后，经常会需要这个人做完了，跟另外一个人讲说我做完了，你下面要做什么，你开个会碰一下，或者传一个文档，那协调层的损耗就会消耗在这个

上面。所以最理想的事情是，工作做完了，这个工作本身就会按照下游的要求检查自己的质量，然后它就会交给下一个人。

理想的交接，是没有交接动作。

案例：福特流水线

这也不是一个什么很新鲜的概念，其实早在福特发明流水线的时候，大家就已经认识到了。流水线上很多的工种其实还是人嘛，也有一些机器，但最重要的是，这个流水线本身、这个传送带本身是一个机器。一个活干完之后，流水线会把这个活自动推给下一个工种、下一个工位，下一个工位又再敲两下，然后再推到下一个工位。

就不需要一个人完成了之后跟老王说，哎，老王，我这个活完成了，你那边好了吗？我发给你。或者老王问上游说，哎，你那个好了吗？你好了发给我。没有这种事情，搞完了就自动推过去了，这就是个传送带、流水线，工作会自己移动。

- **福特 1913 年在高地公园工厂上移动装配线，底盘装配时间从 12 小时 8 分压到 1 小时 33 分**（已查证，见 P024 那条的出处）
- 一句提炼（这是这页真正的知识点）：**流水线上最重要的机器，不是任何一个工位上的机器，是那条传送带**。它本身不生产任何东西，**它只负责让活自己动**

大金句：**你们装了机器，没装传送带**。

案例：安克

所以安克也有这样的例子，他们做的就是一条流水线。比如说他们的新品开发流程，就有非常多的环节，每个环节应该做什么，全程是他们自己定义的——几个人关进小黑屋，整个流水线不跑通不准出来，把整套流程给梳理出来，就相当于定义了流水线。从小黑屋出来之后，这个流水线定义出来了，也分成哪几大块。

弄出来之后，他们就会定义，一个模块和另一个模块中间的交付物到底是什么。然后会有9~10个标准交付模板，是给流程智能体看的，每个环节不重不漏。每次做完之后，缺什么，流程智能体就会自己去补；补不到的，它就会问人类要。然后它做完之后，就会向下一个环节推，推的时候就按照标准来做。

最后发现只有这样做，才把那个效果真的提上来了。早期的时候他们做培训，相当于还是那么多环节，每个人回头自己去用自己的agent，然后就发现其实大家根本用不起来。一是习惯了，二是每次你要用这个agent之前，你还要自己收集数据，用完你还要自己交给下一个环节，会发现也没省什么事。最后要把这些事情都交给AI之后，才能把它用起来。

对，他们在开始一个项目的时候会做一个项目容器，发一个ID，然后所有项目信息就会归到这个容器里面。这个项目容器的agent就会自己去把相关的所有信息给拿到，甚至包括社交媒体的评论啊，什么都拿到。拿到之后，会跟这个项目的那个owner去沟通，说你想完成的目标是什么，然后它会来帮你编

排，说需要走多少步，每一步需要哪些agent、走哪些流程，第一步、第三步、第四步应该怎么走，然后推荐用哪些agent、走哪些工作流。

然后人可以改。比如说我最近看他们举的例子，就是agent推荐了15个agent，后来那个项目经理觉得这个有风险，就加了6个人。然后再去定义每个环节，我们这些agent是用什么样的跑法？是并行跑还是串行跑？要人类检查还是不要人类检查？全部写在编排系统里面。所以他维护的是一张运行图，而不是待办。

然后它就自己按照这个流程，一个一个催Agent来完成。Agent需要人类帮助的时候，就会去呼唤人类帮助。而所有人在里面的时候，也可以看到全局的图。他不是只看到自己这个活，他会看到整个这个项目是什么，所有的信息，卡在哪一步了，然后他需要我做什么，我的东西给下游到底干嘛，都清清楚楚。

不会浪费最贵的人去做无聊的查信息和盯进度的事情。

21 个流程智能体有条不紊跑完全程，交付的十几份 AI 输出物**全部落地**；人与人的协作「非常丝滑」——**「『你那边好了吗』这种东西，一次都没有在群里出现过。」**
定性那句最适合当这页收口：**「这并不是某个单点的流程智能体变聪明了，而是整个项目的运作方式变了。」**

案例：华住

我们举了高科技的例子，再举一个传统行业的例子，同样是传导工作，我们举一个华住酒店的例子。比如说客人打电话给前台说空调不制冷，那大家想想会发生什么事情？是不是前台会分析一下这是什么事，可能就说这其实是客房部的事情，得转客房部；客房部再去看一下，然后转工程班；工程班排班，然后具体安排王师傅来做。那每一站都是重新理解、改写和等待。

如果我们不让人来传递这件事情，而是让AI来做呢？用户打电话的时候，AI就直接旁听了这个对话。是不是AI可以直接理解这件事情是个什么样的性质，直接生成一张工单，就是要修？并且它知道这个电话是从哪里来的，它就知道了这边应该是用的哪个空调、几月几号。然后它也可以直接查到工程班的排班信息，就直接给王师傅生成了一张工单。

那它生成这张工单之后，是不是给酒店前台确认一下，说我没有理解错，是这样吧？然后直接发送，就可以把整件事情完成了呢？所以最好的传导工作，是不是就是不用传导？可以得到更好的服务，每个人的工作量其实都变小了，而且对于前台的培训、对每个环节的培训，时长要求也大大地降低。

最好的传导，是不传导。

AI 传导工作的额外价值

暴露真正的瓶颈

我们传导工作都是从一个人传到另外一个人，这里面可能有大量的线下沟通协调之类的工作，就会很难看出来瓶颈到底在哪里。因为你表面上看，每个人每天都很忙。而既然我的下游很慢，我也不会有

大量的产出——因为产出了我还要跟他沟通，感觉也过不去，我就会停在这里。

但是如果我们是用生产线、流水线这种传送带的方式，由AI分配工作，给我安排个活，我做完了就交回给AI，AI传给下一个人，拿着一个人的工作把它直接送到另外一个人那里，那很快就可以看出来，到底在哪个环节已经发生了大量的堆积，那个环节就会是瓶颈。

人肉传导里，堆积藏在忙里；总线传导里，堆积是一个数

挖掘出隐性知识

我做出来的工作，我跟老王去交接的话，那么其实就会有大量的我们两个之间的沟通。我给的东西，老王觉得不是很满意，然后我们可以通过大量的语言沟通、甚至吃饭沟通来弥补这个不足。而大量的隐性知识就会持续沉淀在我们吃饭的饭局里面，以及我们两个人脑子里面。

如果是我跟AI交接，它给我一个活，让我把它补充完整，然后它把整个大模块往老王那里交，老王这个时候就会逼这个AI说，哎，你这点没说清楚，你那里没给我，你缺这个东西。老王就会在这个过程当中把这些东西讲出来，否则的话给他的东西是不对的。而他讲的东西其实是对AI讲的，那AI就会把这一点给沉淀下来，之后每一个人都可以在这个基础上再进一步进化。

包括我们整个流程出错了之后，最后做出来的结果不好，那我们要反思每一步到底是怎么做的。每一步的过程全部都在总线上，那我们要反思、要修改，要反思我们的整个决策链还是什么做法，全部的东西都会写在明面上，这样子也方便调整和优化。

> 我交给老王，老王不满意，我们两个在饭桌上把它对齐了——**这次对齐的成果，留在我们两个脑子里**。
> 我交给 AI，AI 交给老王，老王不满意——**他只能冲 AI 说「你这点没说清楚、你缺这个东西」**。
> 而 AI 会把这句话**记下来，变成下次的标准**。
>
> **同样一次抱怨，一个消散在饭局里，一个沉淀成了资产**。

一句可上屏：「人对人抱怨，会消散；人对系统抱怨，会沉淀。」

AI 穿到工作的额外问题

工作强度变大

我们之前用流水线做比方，其实就很容易看出来这个问题。福特在使用流水线之后，把汽车生产效率大幅度提升，好像说是从10个小时提升到了一个小时吧，与此同时，他把工人的日薪涨了一倍，涨到了5美金日薪。并不是因为他发善心，而是改成流水线之后，大量的工人离职了。

因为原本的时候，虽然我们会抱怨公司，比如说老是卡呀、这个东西找不到呀，但是其实我去跟老王开会的时候，我去跟大家沟通的时候，以及什么材料缺了我去拿、这个不全我也去补，这些低智力劳动的活，虽然我口头会抱怨这些琐事，但其实它就是我两次高强度工作之间的休息。你把它变成流水

线了，相当于把这些休息的椅子给抽走了，虽然我工作时长可能不变，但其实我工作的强度是大幅度提升了。所以福特并不是因为发善心给工人涨薪，而是因为这些工人如果不涨薪的话，是留不下来的。

- 1913 年福特在**高地公园**工厂上移动装配线，**底盘装配从 12 小时 8 分压到 1 小时 33 分**（不是「10 小时到 1 小时」）
- **1913 年底，福特的劳动力流失率高达 380%——工人受不了流水线，跑了**
- **1914 年 1 月**推出**每日 5 美元**（此前约 2.34 美元，约翻一倍），同时把工时从 9 小时降到 8 小时、改三班制
- 结果：**1914 年产量超过 30.8 万辆，比当年其他所有车厂加起来还多**

工作压力变大

用AI也是。我们传统当中大量的工作本身就是休闲摸鱼型工作，如果你把这部分工作全部取消掉了，对人的脑力消耗其实是更大的。我们平时大部分的工作都是非常低算力的，但你把低算力的活全部交给AI，让它来推进、让它来做，并且它还会不停地推新的活，让我们去做高算力的、只有我能做的那部分事的时候，其实对人的要求、对人的压力就会提得很高。

我自己现在就是在大规模地使用AI，我感觉我的工作压力和对脑子使用的强度，是远远大于我过去的工作强度的，可能是10倍的差别，所以其实现在每天都更疲劳。

这部分有可能有三个缓解的路径。一个就是我们的薪资待遇要提高，对人的要求要提高的话，你对人的回报也要提高。第二就是要提高这个岗位的自主性，不能让人在这边拧螺丝，拧螺丝的活都让AI去做，人应该做更高层的活。AI给人的活，应该是一整块的、有意义的一个目标，而不是一个具体的、完成性的任务。

所以AI在不停推进的时候，是给人发送有意义感的目标，然后人类把这个有意义感的目标根据自己的经验拆解成小任务，再打回给AI，让它去完成这些琐事，这样子人类在过程当中享受到的乐趣会变多。所以对人的要求会提高，给人的钱应该变多；安排工作的时候，不应该是安排工作，而应该安排目的，这样子让人类在过程当中会有更多探索和掌控的乐趣。

检查工作

检查工作：用机制保证质量

AI 产能爆发 & 不确定性增加

我们前面说，其实用AI在任务层提效是一件很容易的事情，一个人的产能很容易就能翻个5倍、10倍。而当我们又用AI来做对齐，用AI来做任务的推进、协调管理和传导，把这些瓶颈解决掉之后，你就会发现产能在大规模地爆发。

这个爆发是不是一定是一件好事？或者说它一定能转化为成果呢？不一定。因为AI现在本质上还是一个非常早期的技术，它有自己的幻觉，有自己的局限性，有非常非常多不靠谱的地方。那当我们的产能爆发之后，这些产能其实是跟过去非常异质的，是跟过去非常不一致的、充满着新的不确定性的产能。

你放大的不只是产能，还有出错的速度。

错误心态：抗拒接受 or 人工审核

面临这种爆发的、但是又不靠谱的产能，大部分人有两种态度。一种态度是，你看它不靠谱吧，你看它有幻觉吧，你看我就说了它没法用吧，来抗拒它。还有一种呢，就是哎，我得用它，但是我确实不放心，所以我们都得用人工来检查它。

对于第一种人，我是想说，虽然AI不靠谱，但是其实人类也不靠谱是吧？那你为什么敢用人类呢？按照你这个逻辑，人类还有可能贪腐，有可能忘记事，有可能不靠谱，你为什么能用呢？这完全说不通。对于第二种人，如果你每一件事都还要用人类在最后把关检查，那AI的提效几乎就是5%、15%已经封顶了，你是无法享受到AI提效的效果的。

用检查控制 AI 产出质量

真正好的做法，应该是我们来设计机制解决这个问题。比如说你们写代码，可能还会有人在做测试，对吧？你做采购，可能还有一个内控的流程在做审计；财务那边还会对账，还会抽查，还会内审。所以我们其实是会用很多制衡的机制来解决问题、来管理人类的。

你现在就把AI当做一个非常不靠谱的、但是非常才华横溢，而且收钱收得巨少的外包公司。那你会怎么样来用这个外包公司？不是不用，也不是去检查他们每一天的工作、盯着他，而是怎么样设计层层叠叠的检查、质控和对抗的机制，来确保它的质量。

你会怎么管一家这样的外包公司？
不会不用——太便宜了。
也不会盯着他每一天干活——那你自己雇人算了。
你会做的是：**定验收标准、设抽检、留追责、分阶段付款**。

1. **自动门禁**——不合规的东西根本进不来
2. **对抗性检查**——检查的那个必须和干活的那个**上下文隔离**
3. **人类抽检**——按带宽排，不是每件都看
4. **兜底与回滚**——错了追得回来

案例：ZooWork

跟我一起做AI炼金术的搭档徐文浩，他们公司就搭了一层harness，每个人都可以自己Vibe Coding、自己写东西。但是他们的代码上传之后会有几十道门禁，检查格式、重复率、命名、依赖，这些东西

都会检查；还会在项目开始的时候就产生几百个测试用例。最终要把所有的规范检查全部过掉，测试用例全部通过，然后人工才会再看一下，再允许合并上线，而不是说谁的东西就直接上线。

这里面当然也出现过小插曲。他分享说，之前他们公司的检查会发现400多个检查全是绿灯，但是上线一堆bug——发现AI其实在里面自己把所有的数据都mock掉了，在做检查的当中作弊作假，假装自己已经通过了所有的检查。所以这边要经过合理的设计，要用干净的上下文、用对抗的机制来做检查，避免AI自己骗自己，也来骗人。

另一方面，有一些检查也是需要人类来检查的，但是AI的量太大了，所以也需要你专门设计看板、设计检查机制，把那些需要人看的东西，用人类适合的、符合人类带宽的方式展现出来。那徐文浩就会特地安排，需要他审核的事情，周一专门看什么，周二专门看什么，这样子比较符合人类的处理模式。

案例：安克

安克也是会在各种检查点检查异常情况，去看到底发生了什么事情、应该怎么办。比如说他们可能就会发现一个定价错误，然后就会分析这个定价为什么错了，然后就会发现，可能是因为把竞品的促销价当做日常价来读取进来。那根据这个来读取的话，后面所有的策略都是错的。如果是后续靠人工慢慢检查出来就来不及了，因为它很快就会进入到整个的定价链路当中，后面很多的决策都会根据这个来做；但是他们的质量门禁半小时就抓出来了，说哦，原来你是把竞品促销价当做日常价读入。他们会检查所有可能的错误，这样子就更早地发现了所有的问题。

除此之外，安克的阳萌也讲说，其实最重要的是检查，而不是那个流程本身。所以他们会非常在意流程中间检查点的设计。你看这张是安克内部的一个流程图，他们会设很多的check point，这个check point有些是机器检查，有些是人工的检查。然后一个交付需要满足哪些点、需要达成哪些质量要求，都会有检查点，通过这些检查点才会往下流。那就保证了错误不会放大，而是每一层在交接的时候都保证了质量。

第一步 · 机制是什么

> 每一个流程智能体跑完的产出物，**必须经过项目负责人审阅确认，才能流向下游**。

第二步 · 为什么要这道门（讲者自己算的账，特别好用）

> 「AI 的产出不是百分之百稳定的，而且**细微的偏差沿着整条业务链条叠加**，一直没被发现，到执行层就会变成很明显的问题。
> **我们加一道确认节点，可能只花十分钟；但这比起让问题流到下游、然后花几天时间返工，已经省时非常多了。**」

第三步 · 这次真的抓到了什么

> 洞察智能体跑完，报告**没有自动流向策略端**，而是进了项目负责人的审阅队列。他打开一看——**竞品的一个促销价格，被当成了正常的长期价格记录了下来。**
> 这个偏差要是带进策略，会影响后续的定价判断。于是在系统里**打回修改**，执行人修正后重新提交，确认通过，报告才流向策略。
> **「整个过程可能都花了不到半个小时。但如果没有这道门禁，这个偏差就会一路带进策略、带进执行，才发现不对。」**

第四步 · 一句话收口

> **「质量门禁要把风险控制在每个节点，不能让偏差累积到最后才爆发。」**

人类对于 AI 预警的异常进行检查验收。

总结

总的来说，就是我们要用AI来加快组织工作的推进。首先用AI来发现工作，然后用AI来分配工作，让AI成为工作的起点以及工作的流水线，提升效率。然后我们要减少人与人之间的口头交接、交接棒，把所有的事情都对着AI交接，这样子不仅可以让传递更快，也可以让经验和教训更快地浮现和沉淀下来。最后我们要用AI来检查工作，用检查点的方式，把AI这种产能巨大但是充满不确定性的产出，质量和产量都提升上去。

发现——让活自己举手
分配——让活自己找人
传导——让活自己交棒
检查——让活自己过关

四件事的主语都不是人。**人从「推着活走」，退到「决定活该往哪走」。**

AI 原生组织转型 · 第 4 讲

模块四 · 用 AI 做闭环

什么是闭环

闭环的目的：一次比一次好

我们经常讲学习型组织，其实讲的就是组织应该越来越厉害：这次做成的事情，下一次还能做成；这次做砸的事情，我们能知道原因在哪里，能把它改进，下一次做成的概率就提高。

所以所谓的闭环，就是在我们的决策和行动之后，我们能看到结果，并且能够把结果和决策行动连接起来，反向地督促我们学习。这样每一次的行动都成为了我们学习的一个环节。

这样讲起来也非常像是 reinforcement learning 强化学习，只是我们是在真实的世界当中，每一次的行动其实就成为了产生数据的点，让公司越来越强。

闭环 = 下一次做同一件事，起点比上一次高。

闭环的类型：能力闭环，市场闭环

闭环有内部和外部两种。内部的闭环，其实就是看这事有没有做成、有没有按照预期完成。比如说我要做这个课程，我的内部闭环就是会发现：做这个课程哪里花的时间比想象当中要久，或者哪个案例最后发现查错了，或者做最后这个 PPT 的过程当中会发现，这种图片方式特别有用、做图的这个 skill 特别好，我要把它沉淀下来。这就是内部的闭环，把经验和教训分析得当，然后让它沉淀下来。

然后也有对外的闭环，就是外部闭环。比如说这个 PPT 讲完之后，大家都骂说，哇，讲的什么鬼，对我有什么用？拿到这个反馈，我就会反思：到底哪里出了问题？或者大家觉得哇，这个太有用了，我要把公司送给你，让你来改。得到这个反馈，我就会来想：这次到底哪里好哪里不好？下次应该怎么做可以放大这个市场效果？

所以第一层是看这事有没有做好，第二层是看市场上有没有得到良好的、期待的反馈。然后根据这两个反馈数据，我们再来反思：是我思考和计划哪里做好了、哪里没做好？还是我执行层哪里做好了、哪里没做好？再来进化我整个的做法，下一次就会做得更好。

能力闭环问的是「我们这次干得好不好」，市场闭环问的是「这事值不值得干」。
前者的裁判在公司内部，后者的裁判在外面。

AI 让闭环更简单

AI 来复盘、学习

大部分人做事其实都缺乏反思和进步的习惯，都是凭直觉在做，做了也不愿意改。我印象很深刻，之前读书的时候你会发现，有些同学很热爱做卷子，但是他们居然不去对答案。你会觉得，你做卷子不对答案有什么用呢？后来知道，他就是寻求一个心理安慰，又不想面对出错、或者面对自己不会真正学习的那个困难。所以大部分人其实是活在一个开环的环境当中，会越来越好地重复昨天的执行动作，但是却很难进化。

我们大部分公司其实也不算开环，只是闭环特别粗糙。每次都在重复地做一件事情，然后凭想象在反思，开个周会、开一下年度战略会反思一下，闭环特别的慢，节点特别的粗，也不知道到底哪个反思是有效的，哪个结果跟哪个决策是相关的。

而且开会的时候，人类都是要脸的嘛，大家不好意思盯着其他部门说你这里没做好、那里没做好，要不然就是相互甩锅。所以大部分人类在复盘的时候，还是推卸责任和找责任比较多，更少去找原因。
那让 AI 盯着做的时候，可以让它更多地去找原因。

AI 本身的学习就是一种闭环的学习：你给他越来越多的场景让他做事，他在做事的过程当中拿到越来越多的经验、得到越来越多的反馈，他就根据这个反馈让自己进一步学习，他就会越来越强。

现在 AI 圈子里越来越多在讲一个词叫 Loop，其实就是闭环。前几天 Google 的 AI 老大 Jeff Dean 出来创业，他的公司就叫 Discovery Loop。所以这是一个非常 AI 模型的概念，就是要做一个圈：不仅让他做事，还得让他自己看到做事的结果，拿到结果之后，他再根据结果去调整和学习。这样整个系统就会越来越强。

我读书的时候一直有个疑惑：有些同学特别爱做卷子，**但从来不对答案**。
后来才想明白——**做卷子是安慰，对答案是审判**。

公司也一样。**开工是安慰，复盘是审判**。 所以大家都愿意开工。

AI 来沉淀、验证

以前很难建立学习的闭环，除了人类不太想复盘、组织很难真的把因果关系建立起来之外，还有一个原因：我们经常总结了经验，比如大家说以后一定要这样、一定要那样，形成了各种规范，但是很难落实到位。那闭环了也没有用，最后沦为新一轮的口号。

但是有了 AI 之后就不一样了。我们前面讲了 AI 可以对齐打法，那么沉淀了经验之后，AI 可以把它沉淀到系统的运营流程当中，让大家强制性地把新的打法沉淀下去。这样一来，闭环的效果就从「分析出经验」到「让这个东西真的有效」，转化率又提高了。所以现在用 AI 来做闭环的效果是更好的：既更容易分析、把它闭起来，又容易让闭起来的那个成果真的用起来。

能力闭环

能力闭环：把经验和教训变成能力

能力闭环：沉淀经验和教训

用 AI 的好处跟以前不一样，就是你可以在做成之后，把沉淀经验、教训的这些活全部交给 AI，跟它说：这次哪里做成了、哪里做砸了，你自己总结一下，自己把它沉淀下来，以后复用。它就会去做。做得好不好再说，但至少这一步就可以把它沉淀下来了。

会有一些误解，觉得这个时候需要搭一层什么很复杂的 Harness。其实只要你用的是比较先进的 AI 和比较先进的 Harness，比如说 Claude Code 或者 CodeBuddy，你完全可以跟你的 AI 说，让它自己把这层东西搭出来。你就是跟它讲一句，你这次做完要自己看一下，甚至于你告诉它，每次做完之后要自动反思一下，都是可以的。

案例：腾讯研究院

腾讯研究院之前是安排人去监控市面上所有的 AI 信息，然后出内容、出报告。后来春节期间，他们就改用一个 AI 来做这些事情，发现效果其实也不错，后来就把它沉淀为一个 Skill。

所以第一层、最简单的所谓能力闭环，就是你用 AI 做了一些事情，发现它不错，你就可以把它变成一个 Skill 存下来，以后直接用这个 Skill 来复用。

春节期间要做热点值班——**他们没有排人，派了一个 skill 上去值班**。每天调用、每天迭代，**节后它成了常驻工具**。

（腾讯研究院做那个值班的那个Skill给找出，那个案例给找出来。）

案例：我的素白 PPT

其实我做这个 PPT 也是一样。前几个月我自己要做一个 PPT，想着我要改一个格式，要用 Codex 做图，但是要用 Claude Code 来做 PPT。我就自己试了一下，大部分其实是手工在指挥 AI 做，最后发现，哎，这个还挺漂亮的。然后我就跟它说，你生成一个 Skill。现在我直接跟 AI 说我要做素白版的 PPT，它就会自动根据我的文档开始出了。

所以所有你做的事情，一旦做成了，你就应该要求 AI 把它做成一个 Skill。组织也是一样，一旦组织做成了什么事情，过程当中 AI 做成的这些路径，你就应该让它沉淀成一个 Skill，以后就可以复用，不要每次都重新发明轮子。

凡是你做成过一次的事，都欠自己一句「把它变成 skill」

案例：Claude

Claude 的安全组之前分享过他们的经验：他们每一次解决完内部的问题之后，都会要求自己的 Claude Code 去总结经验——这次哪些地方成功、有哪些成功的经验，有哪些地方做得不好，你都总结分析一下，然后把你的经验都写到 CLAUDE.md 文件里面。那下一次，他们自己的 Claude Code 就会去看一下那个 MD 文件，根据这些沉淀下来的经验教训再来做。这样每一次的成功经验和失败教训都不会被浪费。

Anthropic 内部用 Claude Code 写 Claude Code，每次解决完内部问题都要求它总结经验写进 `CLAUDE.md`。「安全组」这个具体归属**没有一手出处**，建议上台改成「**Anthropic 内部**」，别指定到某个组。

案例：余一

当然，这跟人类的那种复盘会也有点类似：AI 有的时候复盘、沉淀经验，沉淀完了也不见得真的有效，也不见得真的会改。所以我们甚至会让它去沉淀一些关于「如何沉淀经验」的经验。

比如说我有一个好朋友余一，他用 AI 非常强。他会让 AI 总结经验和教训，说你应该总结经验，下次不要再犯这个问题。然后 AI 就会跟他说，那我以后往 MD 里面写什么什么东西，下次我就会怎么怎么样。

然后余一就会继续 PUA 它：你这样写又是写了几段漂亮话，到那时候下次又不会改，你如何确保下次碰到类似的问题就真的做到？而且你又不能把 MD 文件搞得无限长，你如何把这些统一的原则给抽象出来，不要把它越写越长？他会给这个总结经验的 AI 也提很多很多要求。

所以复盘本身又是一个能力，这个能力本身又可以再循环起来、再迭代。如果你发现一个 AI 上次沉淀了经验、这次又犯，你就可以借这个机会再去告诉它：你觉得这个合适吗？你又犯了这个问题，那下一次我们应该有一种怎样沉淀经验的方法。

复盘本身也是一种能力，而这个能力本身也可以被闭环。

案例：我自己

事情其实没有大家想象中那么难。我举一个我自己的例子。我平时用的 AI，每个星期也会让他跑定时任务，反思我们这一周一起做了哪些工作、遇到哪些困难、沉淀了哪些经验，然后把这些经验和教训给我看一遍，帮我沉淀成 skill。这样他下一个星期做类似的事情的时候，就不需要再重复犯错误了，他就可以进步了，就这么简单。

而且他不光在优化他自己，他也在优化我使用他的行为方式。比如说有一次他就跟我讲，他发现我过去一周有 80 次等他的时间超过了 15 分钟，甚至中间有十几次我等得不耐烦，就新开了一个窗口——因为我要指挥它干活嘛，它干活的时候我跟它讲话它是不搭理我的，所以我又要新开一个窗口跟它讲。

然后它就说：如果你有大量的任务要同时跟我讲，但是又不想多开几个窗口，其实你可以跟我讲「请后台运行这件事情」，那么我就会派一个小弟、派一个 agent 去做这个事，你前台继续跟我聊天也是可以的，你不需要每次都在那边等我。

所以你就会发现，每周让它反思一下：反思自己哪里可以优化、碰到哪些障碍、如何绕过去、有哪些成功经验；也可以让它反思一下我们跟它的交互方式，哪里好哪里不好，如何可以优化。

组织共享：让最佳实践在组织内流动

共享 MD 文件，共享最佳实践

因为很多公司都在用 Claude Code 这类工具嘛，所以我知道很多公司的做法，其实是在组织层面自己搭一层 harness，然后在不同的项目组、不同的部门，按公司级、组织级、个人级安排不同的 MD 文件。MD 文件其实就是一个文本文件，大家就会把最佳实践、打法什么的全部写在这个里面。

所以每一个人的 AI 干活的时候，都会先满足公司层的那个 MD 文件，然后再去读部门层，再去读项目层。这样就保证了最佳实践在组织里面是共享的、是流动的，有谁做出来好的东西之后，组织层也可以拿到这个好的打法。

> 公司级 MD、部门级 MD、项目级 MD——**这不就是公司手册、部门规范、项目章程吗**？
> 区别只有一个：**以前这三层是给人看的，看不看全靠自觉；现在这三层是 AI 每次干活前必读的。**

> 换句话说，**你们公司的制度第一次有了强制执行的路径。**

案例：Every

一般我们讲到组织里面的共享，就会想到不要重复造轮子呀、最好有一个中台赋能啊诸如此类的。但是实际上因为 AI 的能力比较强，它可以在中间做翻译和转译。中台这个逻辑听起来肯定是有用的，但实际上你有了一个中台，它就会干扰前台，它自己会长出能力来——就有点像阿里巴巴之前搞中台，反正所有被阿里投资的企业都怨声载道的嘛。

怎么做到既没有中台，又可以让组织能力复用呢？其中一个打法就是我们之前讲的：让 AI 重写，把所有东西都重写一遍嘛，对吧？像之前讲的 SB 基金，就是徐文浩他们公司的例子。

但是还有别的例子，就像 Every。Every 这家公司相当于一个美国的孵化器，他们做了五六个不同的产品，有些是帮你写 email 的，有些是帮你做 AI 语音输入的。理论上来讲，这些产品都会有一些相通的模块，比如说登录模块，比如说做邀请码的模块，那这些模块是不是应该有一个中台可以复用，是不是能提高效率呢？

他们没有这样做，因为这样做整个组织就又僵化了、又耦合到一起了。他们的做法是：当有一个项目比较牛、有一个模块做得比较好的时候，他们对其他的项目组开放代码，其他项目组可以派自己的 agent 过去偷看——这段代码到底怎么写的，这个模块是怎么做的，它的设计思路是什么，它的文档是什么样子的，看一遍，然后回来自己重写。

所以这跟原来的思路就很不一样。原来是我们要复用嘛，现在不一定要复用，你只要让我看一看，然后我就会了，我自己回来重写，适合我的就好。这就保证了经验可以复制，但是我又不需要卡死在你那一坨上面。

别共享轮子，共享造轮子的那次经历。

自动触发：AI 自动优化自己

案例：安克

比如说我们前面提到的，安克发现某个产品销量下降 23%，就自动触发了一个流程：自动把相关的评论啊、库存啊、广告啊、流量啊这些数据都打包，把上下文打到一起，发给一个人。这其实是当下解决问题的很好的方式。

但是如果我们想要系统性地、整个地提升，这个系统就应该再往下沉淀原因：这个问题为什么会出现？再去分析根因。这个根因可能是当时的广告系统错误地对竞品的价格做了一个判断，做了这个错误判断之后，导致广告在不应该停投的时候停止了投放。那当下的举措当然是恢复投放，但逻辑上来讲，应该是去查出根因——对竞争对手的价格监控到底发生了什么问题，我们怎么把这个问题修复掉，让它以后不要再重复出现。

所以每一个问题的发生都应该触发两条线：一条线是如何解决当下这个问题，另外一条线是去分析出现这个问题的根因是什么、如何修复整套系统，让这个问题下一次不要再出现。这样我们整个公司才

会越来越强。

更进一步的话，如果我当时分析的根因就是广告投放这个问题，那恢复之后系统还应该后续跟进：广告投放已经恢复了，那我们的销量有没有涨回来？如果没有涨回来，那可能是我们分析的这个环节有问题，它找到的这个原因是不对的。原因找得不对，我们就又激发了下一个自我学习的闭环——这个原因到底是什么？我们应该如何去分析？

我们很多时候对一个问题的解决常常只有一层，就是解决这个问题。好一点的会有两层，就是分析根因、把根因解决了。但是实际上，解决完这个问题之后，还要再看一下：我们把它修正了之后，数据有没有变？市场反馈有没有变？然后根据这个反馈，我们再来判断自己之前的判断做得对还是不对。所以后面其实还需要做很多的工作。

> 每一个问题发生，都应该触发**两条线**：
> **第一条**：当下怎么解决？（恢复投放）
> **第二条**：根因是什么、怎么改系统，**让这个问题下次不再出现**？（修掉价格监控）
>
> 大部分公司只跑第一条。**跑第一条的公司在救火，跑两条的公司在变强。**

你后半段还讲了第三条，而且这条更少人想到，建议明确编号：

> **第三条**：改完之后**回头验证**——广告恢复了，销量涨回来没有？
> 没涨回来，说明**根因找错了**——这时候要闭环的不是那个问题，**是「我们分析根因的方式」**。

案例：安克

在实体生产里面，安克也分享过另外一个例子。我在他们的一个分享里看到，他们有一次出现了螺纹开裂，螺纹一开裂，系统就会启动一个四维数据的联合推理，去分析到底可能是什么问题。AI 会自己分析，分析的过程当中可能还会反问工程师：再给我多一点上下文，到底材料是什么？这个是刚发现的，还是隔一段时间才出问题、还是马上就出问题？最后判定是熔接线和应力集中的问题。

这个时候它不光会告诉你这个问题当下怎么解决，而且会判定说，这是我们前面 checklist 的问题，要把这一条加到生产的可制造性流程检查点里面去，以后就不会再出现类似的问题，相当于把这个问题的识别提前了。这有点像生产里的 8D 分析：你不是光要解决当下的问题，关键是要有一套机制，保证日后这个问题不会再重复出现。

具体来讲就是，我们看到一个螺纹开裂，应该是 AI 马上识别到它开裂，然后去分析它为什么会开裂，这个开裂是由哪个设计造成的，再去思考我们的流程当中为什么会存在这个设计、以后怎么样规避这个设计。当它融入到流程当中，变成一个以后的设计规范和检查点的时候，它才从一个偶发的问题，变成了以后的经验和积累下来的财富。

这个案例是全模块最具体的一个，建议把细节给全（一手转录里有）：

> 生产中发现**螺纹开裂** → AI 触发**四维数据源联合推理** → 分析过程中**反问工程师**：「材料是什么？是刚发现的，还是隔一段时间才出问题？」 → 最终判定是 **熔接线 + 应力集中** → **修复项自动推进进 DVT checklist**。

最后半句是这一页的全部价值，建议单独停下来讲：

> 注意最后这一步——**它不是给你一个答案，它是给你的检查清单加了一条。**
> 下一款产品设计的时候，这条会自动被检查。**这个问题从此在这家公司消失了。**

你把它类比成 8D 分析是很准的，建议保留并且点名：这就是制造业几十年的 **8D / 根因分析**方法，**区别只在于以前这一步靠一个很有经验的质量工程师，现在它是自动触发的。**

一句可上屏：**「问题的终点不是被解决，是变成一条检查项。」**

案例：ZooWork

所以总的逻辑大家就可以看到：每一次的摩擦、每一次的问题，都应该是一个刺激，让我们的系统长出新工具、长出新能力，让它的血槽变厚。

安克还有另外一个例子，就是他们的数据中台。如果员工去数据中台调数，调出来了，他们就会把这一套调数的方法沉淀下来——原来这个数据也能导出来，是这么搞的；如果没调出来，他们就会发现数据上的缺口，然后去识别我怎么样 fix 这个缺口，缺什么东西、要找哪个人类去要，然后把这个系统能力建设起来。所以每一次使用，其实就是对它的一次赋能。

我很多次讲到徐文浩，是因为我经常找他聊天嘛。他讲他们公司专门有一个叫 Optimize Harness 的东西：一个 agent 帮他们干活，中间遇到卡点，缺什么权限、缺什么工具、绕了多少条路，他们 harness 里的那个 agent 就会专门去分析你这一路上的辛辛苦苦和不容易，然后去帮它设计工具——下一次你可以用这个；下一次我不应该帮你准备这个环境，我应该帮你准备那个工具，把整个环境设计好。这样让整个组织的能力变强。

每一次摩擦，都应该让系统长出一件新工具。

市场闭环

市场闭环：根据市场反馈调整

市场反馈是最好的训练数据

刚刚讲的这些能力闭环，还是我们自己内部的事情，比如说活有没有干成功、螺纹有没有开裂，都是一些有内部标准的事情在闭环。但是我们最终是要去市场上拿结果的，所以最良好的闭环、最公正无私的反馈指标、最有效的数据反馈，其实是市场的反馈。

案例：Applovin

最经典的案例就是 AppLovin 这种做广告投放的公司。这种广告投放，他们内部肯定都有一套评估体系：根据每一次广告投放，效果好还是不好，多少人看了、多少人点击、多少人买东西、多少人下载，反正都会拿到这些数据，然后根据这些数据再去反推、去反思——我们的策略、我们的投放人群、我们的物料到底有什么要改进。那一整天就可以迭代几十圈。这种闭环速度快了、颗粒度细了，效率才能越来越高，这是非常好理解的案子。AppLovin 最近估值也涨得很快。**AppLovin 2025 年收入 55 亿美元、+70%**；

AppLovin 这个案例库里有可视化版本，比讲「一天迭代几十圈」好用得多：

> **一圈：500 次展示，其中 15 次点击。**
> 系统怎么做？**压低那 485 次无兴趣展示对应的特征，强化那 15 次点击以及后续深度浏览对应的特征。**
> 然后下一轮再来 500 次展示，再看点击、浏览、购买信号，**逐轮调整人群预测。**

引擎有名字：**Axon 2.0**，2023 年初上线——这是转折点。

案例：AI Agency

我有一家孵化器，他们孵化过一家做广告投放的公司。我看过这家公司，他们当时其实是在 AI 做广告物料还很不成熟的阶段，但是他们非常激进：不仅让 AI 出策略、AI 看数据，也让 AI 出所有的广告物料。

他们当时就号称，别人一周只能迭代一两圈的时候，他们一天就可以迭代 20 圈、20 次循环。为什么呢？因为他们整个流程当中没有人类的参与，没有任何人类审批，也没有人类在那边做物料、在那边看，全部 AI 往外出。这样他们的迭代效率就可以更快。这家公司后来在那个孵化器当期的 Batch 里面，应该是融资融得最顺利的。

案例：艾语智能

其实也不是那么简单，不是每一次直接把市场反馈给他就好，因为很多市场反馈是长闭环的，很慢。比如说艾语智能，他们是在做电话的催收和展期。如果只是把最后的收款行为当成落点，那有些案子因为展期了，分成 24 个月、36 个月，那就太久了。

所以他们就要收集很多前置的指标，比如说加上了企业微信，这也算一个，要把这些都算上，这样系统才能够得到更快的闭环。不能等到真的整个商业结果出来——虽然那个很硬，但是太慢了。所以这里也是需要精巧的设计的。

给 AI 授权，AI 才能有效学习

让 AI 做 PMO，而不是小助理

如果只是在各个项目的各个局部使用 AI，他其实就是个帮你打杂的，那你最后告诉他这次项目效果不好、让他去复盘，其实复盘不出什么东西。

但是如果像我前面讲的，这个项目本身是 AI 在推进，AI 掌握全局，由他来安排事情——哪怕中间你不同意他的各种做法，你否定他，他改了，但最终相当于他是 CEO，你是董事长，你可以改他、可以监控他，但整件事情你是全权交给他来负责，你在旁边督导。这种情况下他是知道全局的，从头到尾是他在做，他最后去复盘就会更有用，他学习才会有效。

所以我们要让他学习，就是要把前面那些事情真的交给他，他才能够学会。不能把他当小助理，得把他当 CEO 的接班人来培养，至少是个 PMO。

你要问自己一个问题：**这件事，是他在做，还是他在帮你做**?
帮你做的，**复盘的时候他只有片段**——他不知道当初为什么这么定，也不知道后来为什么改。
他自己做的，**从头到尾的因果链都在他手里**。

没有全程，就没有归因；没有归因，就没有学习。

案例：Block

（我调取一下Blocker这个案例，就是他们会把公司当做一个mini ATI，然后 mini AGI的话，他们会根据公司的，根据客户的情况，比如说这个客户是这种商业性质，然后他历史上在这个时候都遇到淡季，会有资金链的短缺。那根据他的情况的话，我们应该给他生成一个什么样的金融产品。mini AGI会自己做决定，自己生成组装出这个产品。如果缺什么能力，他会找专门的那个同事去帮他补，然后做完之后，他会自己发，推给这个用户。如果他接受了，没接受的话，他又会把这个信息拿过来说，哦原来他不接受，那我对他的情况进行一个修正，或者说我对我的策略进行一个修正，他又会自己再闭环、再学习、再修正。整个过程的话。会有点像是我们在刷抖音，抖音其实就是根据对你的偏好在不停的给你推东西，这是它来决定的，它推完之后根据它推的东西你有没有接受？有没有点赞，然后再去反思说，哦？要么它对我们的偏好了解有问题，要不然它自己推放策略有问题，它会再自己再进化，所以所有的工作过程就是进化过程）

案例：Kavak

他们的原则是 **build agents per customer, not per workflow——不是一个任务一个 agent，是一个客户一个 agent。**
每天实例化 **10–20 万个 agent，每个自带一台虚拟机**，记住这个客户几年的交互；**醒来工作三分钟或者三天，然后给自己定个闹钟去睡觉。**
目标不是完成工单，是这个客户的终身价值。 为什么这样闭环才成立（这正好接你上一页的论点）

> 因为**这个 agent 拥有这个客户的全部上下文**——它知道当初为什么这么谈、后来为什么变卦。**它归因得了。**
> 而且它缺什么会自己要：要更多 context、要更多能力——**后台的人就是去满足它的。**

人在回路的接法，这条特别反直觉，建议单独讲

> 业界常规做法是 agent 撞墙就甩给二线人工——**结果闭不了环，也产不出训练数据。**
> Kavak 反过来：**agent 撞墙时调一个 API 说「我需要帮助」，API 那头是人。**
> 他自己的话：「**画成组织架构图，就是人类团队向 agent 汇报。**」

eval 那条是这一页的压舱石（也回扣你「市场闭环」的主题）

> 「我要跑得极快，但要跑快你得有刹车。**能开多快，取决于 eval 的质量。**」
> **量化配比：花在 eval 上的时间、token 和钱，和造 agent 本身大致相等（1:1）。**
> 而且他们的 eval **只认业务结果**（转化了吗？客户还回来吗？），**点名批评「通话数量、通话分钟数」这类表面 KPI。**

总结

总的来说，我们应该让 AI 来做闭环。一个是做事的闭环，就是我们内部的这些工作：做得好的下次可以做得更好，做得不好的可以沉淀教训，下次不掉到同一个坑里。另外，我们要做市场的闭环，就是拿到市场的反馈，根据市场的反馈来调整我们的策略和对客户的理解，然后一步一步地从市场的需求把我们内部的能力给拉出来，而不是我们自己在这边盲目瞎搞。

AI 原生组织转型 · 第 5 讲

模块五 · 三个落地动作

从愿景到行动

不是技术问题

- 第一步行动，不是找“AI 专家”
- 第一步，全是传统 IT 问题 & 业务问题，没有 AI 问题

- 如果是小公司 & 有决心，直接搞就好
- 如果是大公司，难度 X 10，建议先做准备动作

我们刚刚讲了这么多，大家可以看到一个未来真正的AI原生组织是什么样子：用AI来对齐事实、对齐打法，用AI来发现工作、分配工作、传导工作、检查工作，以及用AI来做能力的闭环和市场的闭环。那么接下来我们要怎么开始呢？首先我要纠正大家一个极其错误的想法：大家一想到这是跟AI相关的事情，想的就是我要找一个什么研究员、科学家、博士来做。

不是这样的，真的不是这样的。对公司来讲，这是一个业务转型和组织转型的过程，所以你需要的是梳理业务、梳理组织。一开始的时候，远远没有到搞技术、搞算力算法、搞服务器那一步，这些东西跟你毫无关系，甚至哪个模型好、哪个模型差，什么harness，跟你都没有关系。你要思考的是业务和组织问题。

这个问题，就我们的实践经验看起来，如果你是一个二三十人的小公司，而且你是CEO，其实非常好解决，可以一步到位。你们公司统一用一套Harness，或者哪怕都用飞书，然后你按照我这个框架硬推，公司里搞几个人硬推，你又有决心开掉一半、换掉一半的人，那问题都不大。我现在认识的这些小公司都做得很好，效率都巨高。你是强势领导，又有这个意识的话，改动起来都非常容易。

比较麻烦的是，如果你不是二三十人的小公司，而且你不是CEO。比如说你是个500人、5000人的公司，你是个高管，这个时候难度可能比你想象中要大10倍、100倍。所有看起来低垂的果实，比如说其实只要很简单的AI技术就可以拿到的效果，像把会议协调一下什么的，整体推进起来都会比你想象当中困难很多。因为这毕竟是个组织问题嘛，组织问题牵涉到的大量是利益的问题。

很难一步到位

- 不是：大刀阔斧搞改革
  - 难度会比想象大，波音后来也回退
- 而是：搭架子，准备人，试试看
  - 让可能性生长出来，新业务拽出新组织

所以我的建议是，我们提前做好各种各样的准备工作，把整个架子搭起来，让一些年轻人冒头，先小规模地试起来，让他们把这个结构在小范围跑起来，把整个公司往那个方向拖。

我前几个月发表过一篇文章，好像广为传播，就是我在讲AI转型是没有用的，必须AI投胎。因为你要转变一个大的恐龙其实很难，你必须要做一些新事，等这些新事的新打法、新市场打出来之后，它会慢慢把老的资源吃进去，实现一个重新夺舍的过程，这才是真正的转型。所以「转」是很难的一件事情。

我们现在应该做的，是做好准备，让自己的一切都做成AI ready，准备度比较高。然后鼓励一些新的事发生，让这些新的事有可能把老的东西往里面吃，准备好被夺舍。

波音自己也是最好的例子：**777 之后，他们把这套丢掉了，787 退回了部门协作**

转型的逻辑是：**把老组织改造成新组织。**

投胎的逻辑是：**在旁边长一个新的，让它把老的吃掉。**

做好准备，激发涌现

- 修路：让 AI 能进入你的公司
- 点火：准备好懂 AI 的人才
- 分圈：设计好独立业务小组织

那这些准备有哪些呢？其实就是今天大家回去就可以做的3件事情。

第一件事，我们要让AI能够进入你的公司，你的那些系统啊、权限啊，尽可能地打开，让AI好用一点。第二件事，你回去要搞AI组织转型，总归不能当光杆司令吧？你得有几个人配合你，所以要把公司里面AI准备度已经比较高的人挖出来，让他们跟你是一伙的，一起去推动这件事。第三件，要在公司里面划一些小圈子，让一些小团队去完成端到端的任务，把完整的业务交给这些小圈子，让局部先高效运转起来。最后，要倒过来授权，让那些在前线的小圈子从端到端的任务逐步向市场侧转移，给他们越来越大的权限和激励，让整个公司被这些鲶鱼激活，再把有效的打法往回灌给母公司。

简单讲，就是先修路、点火、分圈。

这几个准备做完之后，你会发现，这个模型好还是那个模型好、你要用workhorse还是cloud，这些其实都是细节的小问题。你的系统、你的人才、你的权力结构、你的组织架构都已经有了一些松动，做起事来就会比较方便。

该发生的事情会自然发生。

修路

修路：让 AI 能进入你的公司

要做的第一个准备，就是让自己的公司对AI开放，让AI进来之后能用。这有点像你招聘一个特别强的人，招进来得给他配台电脑，帮他开通公司的各种权限，给他开个门禁卡，否则他没法干活呀，对吧？

所以第一件事，你公司里面这些系统、所有这些信息、这些知识，它能不能读到？第二件事，你能不能找个老手带带他，让他不光能读到这些，还大概知道是怎么回事、彼此之间是什么关系。第三件，他如果能看懂了，你还能不能把那些系统都开给他，让他能够操作、能够点来点去。这三件事都能做到，AI就能够在里面干活了。

具体是哪个AI不要紧，到时候可以换，但这些基建是肯定要做的。这些基建基本上不需要什么AI科学家，也不需要什么特别强的FDE就能做到。

总的来说，就是要让 AI 能够看见业务、理解业务、操作业务。

让 AI 能看见业务

前提：业务过程留记录

所以第一步是让AI可以看见业务。对大部分公司来说，这听起来像是要让AI能连通你们家数据库之类的，但实际上，你现在放一个超级厉害的专家，只看你们公司数据库，只看你们公司已有的那些所谓的知识图谱文档，绝大部分公司的情况是，这个专家看了也不知道你们在写什么，看不懂。大部分公司的绝大部分业务，真实的过程是没有被记录下来的。所以最重要的第一件事，还不是联通这些，而是赶紧开始把所有的业务过程留痕、留记录。

比如我们说我们公司最核心的能力是知道怎么搞定客户。好，那搞定客户这件事有SOP吗？有流程文档吗？每一个客户现在是什么情况、具体是怎么搞定的，这个有记录吗？如果没有，那对AI来说，这部分业务就是不可见的，这部分业务的知识也是不可见的。

或者你说，我们公司曾经有50万个用户都用我们的设计软件做过椅子，所以我们在椅子方面有大量的沉淀和经验。那从他们打开软件到最后做成椅子，中间那些点击的操作步骤在不在？那些数据在哪里？那些经验在哪里？哪些做成了、哪些没做成？他们反复的那些分析在哪里？然后能不能合法地使用这些数据？如果没有，这些东西就还是不存在。

或者你说你的能力是我们的销售特别强，在4S店可以搞定用户。那我们的销售对话在不在？有没有录音？有没有存档？能不能合法地使用？如果没有，这些业务对AI也是不可见的，那你所谓的数据财富也都是不存在的。

从这个角度出发，你会发现，我们大部分的业务在AI的空间里面其实不存在，大量我们认为的数据资产，在AI的时代也是不存在的。所以我们第一件事，最简单的，就是开始疯狂地做记录、留档。最简单的方法就是能录音的全部都要开始录音，能加埋点留记录的，现在全部都要加上埋点留记录，所有结构化、非结构化的数据都开始记下来，然后你要开始检查。

没有被记录的业务，对 AI 来说不存

案例：YC

YC是美国最有名的孵化器，Sam Altman之前就是YC的CEO嘛。YC最近做了一件事情。他们是教育创业者应该怎么创业的嘛，以前有一个操作手册，会写说你应该怎么融资、应该怎么做产品。但这份东西好几年没有更新了，因为总觉得做一次挺麻烦——你想想，写一本书该是多大的一个动静啊？而且你写的这本书又是在这种瞬息万变的世界里面，那就很难写了。

那他们真实的动作是怎么做的呢？他们发现自己有一个很好的行为，叫办公室时间Office Hour。什么意思？就是创业者可以跟他们那些合伙人去请教问题，合伙人就会回答他，哦，你融资遇到这个问题，然后指导他。他们所有这部分的指导都是有录音的，包括现在每天可能还有很多很多这种Office Hour正在发生，是鲜活的、当下的指导。

所以他们干脆就不写书了，而是直接根据他们大概几千个小时、几万个小时的Office Hour，根据创业者的问题和合伙人的回答，直接出了这样一个手册。这样一本手册就是鲜活的，而且它不光是一本PDF的小册子，更是一个活着的、你可以直接向它问答的、相当于Agent的机器人。这对每个创业者的帮助就更大了，他可以听到每一个不同的YC合伙人对于他的问题的具体回答，而这个回答是从几千几万个小时的真实回答当中萃取出来的。

这一切的前提你发现没有？就是他们过去把所有的Office Hour都录音了。如果没有这些录音的存在，他们是做不了这件事情的。所以你想想，你们公司有没有发生这种业务过程？这种业务过程有没有被记录下来？这种业务过程本身是不是蕴含着大量的智慧、知识、经验？这些都是财富，如果没有记录下来，赶紧记录下来。

注意，**YC 没有为了做这件事而新增任何动作**。
Office Hour 他们本来就在做，录音他们本来就在录。
他们只是发现——原来那堆录音，已经是一本书了。

所以留痕的价值不在当下，在于它把「以后可能做的事」的门票买了。

案例：安克，出门问问，Claude

讲到对齐的时候，我们讲了安克的例子，安克所有的会议都要被Agent记会议纪要，并且所有的会议纪要都打通，这其实就是把非结构化的知识和经验、把当下发生的事情都存下来了。那出门问问，要求每一个人不能人对人沟通，都要有Agent这个第三方参加。然后Claude Tag会跨组、跨项目地旁听所有的事情。这些其实都是在把非结构化的信息结构化，做记录、留存档。那你可以想想，你们公司还有没有更多可以留的地方。

只有留了痕的业务过程，才是AI能够理解的业务过程，才是AI可以看得到的你的公司。否则的话，这一块不管是业务也好、经验也好，对它来说都是不存在的。所以第一件事是留痕。

留痕不是记录会议，是记录组织。

让 AI 能理解业务

让 AI 读懂业务关系

当然了，如果我们光讲看见，把信息存下来AI就理解了，那也不见得。看得见不等于看得懂嘛。现在AI看到的是各种孤立的点，首先第一个问题就是信息和信息之间连不上。

比如说你是4S店，你把公司的文档、聊天记录、工单都给他，他看得到的是：张先生打电话来问保养；京A1234这台车三个月第二次进场；工单20260814-17，这个客户情绪不好。这三条消息，他怎么知道是同一件事呢？对吧？你得有办法让他知道。

另外一件事，就是你们公司可能有各种流程定义，比如客户进来的pipeline；一个业务实体跟另外一个业务实体之间有相应的关系；不同的部门、不同的系统、不同的业务实体之间有不同的能力，可以发起不同的动作；历史上我们又会有不同的经验教训。这些东西它都不知道嘛，所以我们需要把这些都

告诉他。讲得高级一点，就是它需要有一张知识网络，知道公司里面有哪些实体、实体之间的关系是什么、每个实体可以有哪些行为、可以发起哪些行动，这些东西你都得给他。

有点像你们公司来了一个新人，办好了工牌、办好了各种入职手续、开通各种权限之后，你还要告诉他公司里面那些系统里没有的知识：哪个东西一定要走哪个流程，A部门和B部门是什么关系，哪家厂其实是我们的关联公司、哪家厂不是。这些隐性的知识都要告诉他，他才能够理解业务。

案例：Ontology, OpenKG

讲到这个话题，最有名的其实就是Palantir的Ontology。我们现在孵化了一个服务型的公司，出去也会说我们帮你做一套Ontology。其实我看下来，跟我们做的知识图谱也没有太大的差别，无非就是把整个业务理解一套。所以你如果对这一块感兴趣，可以去研究一下Palantir的Ontology；或者我前一段时间访谈了开放知识图谱OpenKG这个大的开源框架的发起人、同济大学的王昊奋教授，大家也可以去看一下。

这是两个不同路线的思考。但一开始的时候不需要进入到这么细的程度，你其实就把它全部写在一个MD文档里面，你们公司有哪些系统、怎么运转，用口喷的方式让AI来问你，也是可以用的。然后在用的过程当中，你发现它缺你们公司哪些知识，你再去补；补不上的话，再去外面找专家也可以。让AI来告诉你应该怎么搭这套体系，也是可以做出来的。

听到 Ontology 别紧张。**这件事有一个 5 分钟版本和一个五百万版本，你先做 5 分钟版本。**

5 分钟版本：开一个 MD 文档，把「我们公司有哪些业务实体、它们之间什么关系、哪些流程必须走哪一步」**用嘴说给 AI，让它记下来。**
然后在用的过程中，**它不懂什么就问你什么，你补什么。**
补不上的地方，才是你需要找专家的地方。

让 AI 能操作业务

为 AI 准备 MCP & CLI

最后就是要让AI可以操作你的业务。我们大部分业务操作都在电脑上嘛，所以你所有的系统最好都能让它使用。现在我们的程序、软件、内部系统其实都是为人类设计的，对AI不太友好，让AI在屏幕上点来点去，你会发现点得巨慢。这个时候如果不改造，就相当于你请了一个打字飞快、过目不忘的超级高手，却把他手脚绑起来，只让他用嘴巴叼着一根铅笔去点鼠标，那当然很慢嘛。

所以我们应该为AI设计适合它的交互界面，这样它就可以飞快地操作了。那它适合什么呢？第一个是MCP，相当于给你的每一个系统功能都给AI写一份说明书，教它怎么用，它自己看说明书就能自己用。另外就是提供CLI的界面，其实这两者你用CLI就可以，不一定要用MCP。用CLI这种纯命令行的界面，它操作起来就不需要看屏幕，都是读文字，再用文字的方式下命令去处理，然后再读文字，这样就快得多。所以建议是把系统重构，做CLI化。

MCP 是说明书——告诉 AI「你能干什么、参数是什么」。

> **CLI 是把手**——让它真的能动手，而且是**用文字动手，不用看屏幕**。
>
> 两个的共同点是：**都是给机器看的界面，不是给人看的界面**。 你过去二十年做的所有 UI 优化，**对 AI 来说全是障碍**。

做一定有用的事情

而且这个事情是反正都要做的。因为大部分工作，比如说你要不要做一个Agent平台？你要不要在里面帮Agent变得更聪明？这些Harness的工作，大部分做完6个月后就没有用了。但是你们家的接口到底有哪些功能、它的规则是什么，这些东西过6个月，AI还是需要的。这属于最基础的基础，所以你可以先搞这个。

但我现在见到大部分公司都是上来先搞一堆花里胡哨的、跟AI更相关的工作，那些反倒没有必要，大概率6个月之后就过时了。甚至包括很多公司还在建内部的各种AI系统。如果你是几万人的公司，你当我没说；但几百人的公司，我看到很多都在建内部的这个系统、那个系统，其实必要性是非常非常弱的。什么内部的这种库、那种库，从零开始开发，六个月之后你们都会发现是白干。

判据（这句建议直接上屏）：

> **凡是会随着模型变强而失效的，都先别做；凡是模型再强也还需要的，先做。**

理由（把它讲成投资问题，老板听得懂）：

> 你做的 harness、你搭的 agent 平台、你调的那些提示词——**它们的价值随着模型变强而衰减**，因为下一代模型自己就会了。
> 但「**你们家系统有哪些功能、每个功能的规则是什么、数据在哪儿、口径是什么**」——**模型再强也得问你**。这件事没有人能替你做，也不会过期。
>
> 所以这不是「先做哪个」的问题，**是「哪个是资产、哪个是耗材」的问题**。

一句可上屏：

- **「先修不会过期的那部分。」**
- **「Harness 是耗材，接口和口径是资产。」**

案例：Stripe

随便举个例子，比如说Stripe。

这个过程我想讲两个例子，一个是Stripe。Stripe首先做了MCP，就是我刚说的说明书，你现在在各个AI里面敲一行命令就连上了，AI自己就会用了。

第二个是CLI，AI用起来就特别方便，CLI装好了之后它用起来速度特别快。就好像我现在为了把AI用好，是用飞书跟我的AI沟通，它平时在我的移动端上，就是因为飞书提供了CLI的能力，所以我的AI操作飞书会比较容易，有事就通过飞书跟我说话。

第三个，它还单独为AI做文档，AI看一看Markdown的这种文档就知道该怎么做。而且很巧妙的是，它不是出了一堆文档啊、工具啊这种东西，而是出了一个渐进式的目录，告诉你我们有哪些东西、每一个到底有什么用；你想要什么功能，再根据这个目录去找到相应的功能，它再详细告诉你怎么调，不会让你一下子去读一本百科全书。这样对AI就特别友好，不会一下子把它脑子给占满，不会把Context Window给占满。

① **MCP**：Stripe 有官方 MCP server，**托管在 `https://mcp.stripe.com`**，走 OAuth 授权。它给 AI 的不只是调用 API 的工具，**还包括搜索 Stripe 知识库（文档 + 支持文章）的工具**——也就是说，**AI 遇到不会的可以自己去查文档**。

> ② **CLI**：Stripe CLI 是老资产了（本来就是给人用的），**现在直接成了 agent 最顺手的把手**——这正好呼应上一页 Minions 那句「对人类好的，对 LLM 也好」。
>
> ③ **给 AI 的文档**：Stripe 挂了一个 **`/llms.txt`** 文件——这是一个正在成形的标准，**专门告诉 AI 工具怎么去取每一页的纯文本版**。他们自己给的理由很具体：纯文本**格式 token 更少**、**藏在折叠标签里的内容在纯文本版里是展开的**、**markdown 的层级 LLM 解析得动**。
>
> ④ **Agent Skills**：Stripe 还做了一个 **skill 目录**——**把「怎么正确接入 Stripe」这件事本身写成了 agent 能直接执行的技能**。
>
> **你说的「渐进式目录、不让它一口气读百科全书」是对的**，而且这正是 `llms.txt` 的设计意图：**先给索引，再按需取全文**——省的就是 context window。

我自己用飞书，就是因为 CLI

提醒：不要做

① **别从零自建一套内部 AI 平台**（几万人公司除外）

> > 你现在建的那个「内部 XX 库」「内部 XX 平台」，**大概率六个月后是白干**——因为它解决的是这一代模型的短板，而这个短板下一代自己就补上了。
>
> ② **别先招 AI 科学家 / 算法团队**

> 你现在缺的不是模型能力，**是能被模型调用的业务**。招进来他也无从下手，而且他会开始建第 ①、②、③ 个平台。

③ 别一上来做全公司知识图谱 / 大中台

> 立项半年、上线一年、上线那天业务已经变了。**从一条业务反推着做**，AI 缺什么你补什么。

提醒：短期不一定正反馈

再提醒一件事情。在提供CLI的过程当中，你很有可能会发现，原来的系统其实有很多设计不合理的地方，你会想要重构。有些地方甚至AI想要的功能，你这个能力单独提供不出来，甚至根本没有API化，更不要说CLI化了。这个时候，你的系统有很多要升级的地方；但很多时候你要清洗这些数据、要重构这个系统，成本和过程会比你想象当中更艰难。

艾语智能跟我聊的时候就说，他们整个重构的时候，本来以为数据清洗是几天的事，结果做了三四周，重构系统那个月公司的业绩还跌了。所以他很坦诚地说，大概率你的AI系统改造完会比原来的效率低，但是它加速度快，迭代得快，修得快。所以大家要有这个心理准备。

点火

点火：准备好懂 AI 的人才

我刚刚说的是修路，是方便AI进入你的公司，让它能够把你公司里面原有的资源、信息、系统给用好。但它进来了，如果没有人接应也是白费嘛。现在也还没有做到完全靠AI可以run一家公司，所以第二件事就是准备人才，要点火，把人才准备好。

大家可能会觉得，我不一定要内部准备人才，我可以靠外面的咨询公司、FDE这些人来搞定。但事实上我们现在看下来有两个问题。第一，FDE要理解你的业务，其实还挺花时间的。第二，经常有朋友让我介绍FDE，我说我介绍的人都会贵到你不想找，你最后还是打算自己搞。

所以性价比最高的方法还是用你自己的人，他最懂业务。你把中间最懂AI的那些人挖出来，在人才上面花的钱其实就要少一个零。你要是找外面的，尤其是稍微有点名气的，比如说你找我，成本会高到你几乎想要放弃这件事情，最后还是会被逼到自己搞。所以最省钱、最高效的方法不是找外部的AI专家或者AI科学家，而是把公司里面已经懂业务、已经沉淀在业务当中、已经熟悉流程、又拥抱AI的这些人挖掘出来，把他们激励起来，通过外部赋能把他们的能力增强，这就是你最好的FDE团队。

最好的 FDE 已经在你公司里了，只是还没被找出来。

老板带头

老板是最适合 AI 转型的个体

这批人应该怎么弄出来呢？其实就几件事。第一件事，老板一定要带头。

因为用AI主要会卡在几件事上，而这几件事老板天然就不会卡。第一个卡点是大家不会布置任务，只会自己干活，对吧？大部分人，你让他自己做一件事，他做得挺好；你让他把事说清楚交给别人做，他说不清楚。但老板天然就是指挥别人干活的，所以天然适合下命令，这事用现在的AI特别合适。第二个卡点是很多人想不出来有什么事要交给别人干，因为没有用惯助理、没有用惯下属嘛，所以想不出来。

老板天然就没这个问题，又知道怎么支持别人干活，又知道怎么PUA，又天然每天会有莫名其妙的想法出来。所以老板是最适合AI转型的那个个体。你作为老板自己都转不过来，下面人更转不过来，所以你首先要自己做示范，何况你自己本来就是最适合转的。

但更重要的是上有所好，下必甚焉。你显得非常热爱AI，下面的人才会跟你一起来搞。如果你只是喊一喊，下面人看得出来的。你每天就用用豆包，那你还要下面人怎么转型？大家一看就知道你水平就在这里，就是做个样子。所以大家是看得懂的，你自己需要扑上去。

这边我打一个预防针：很多老板其实热衷于看AI新闻、热衷于讨论AI，但并不热衷于用AI。什么叫真的在做AI转型？就是你日常工作都已经在用AI了，自己搞东搞西。国内的话，你用WorkBuddy、Trae这种工具；国外的话，你用Codex、Claude Code。你至少要用一个，才知道到底什么叫AI。就有点像诺基亚也是手机，iPhone也是手机，但iPhone跟诺基亚就不是一个东西。所以你必须要用这种东西，才能说自己现在是在这个时代用AI干活了。如果你只是在看公众号，转发一些短视频到公司群里说你看别人怎么怎么样，那你就是你们公司AI转型的一个阻碍。

普通人用 AI 卡在哪	老板为什么不卡
不会布置任务——自己干得挺好，让他说清楚交给别人做就说不清	他天天在指挥别人干活
想不出有什么可以交出去——没用惯助理和下属	他天天在想「这个谁去做」

案例：老板重构系统

猎豹的傅盛、出门问问的李志飞、特赞的范凌，你会看到都是老板自己已经疯狂地拥抱AI了，然后带动下面人开始拥抱AI。

这边插一个段子。我认识一个做工业的老板，今年春节过完年之后，有一天早上非常激动地打电话给我要约时间，说，我天呐，我们当年花了1400万做的某某某系统，我上周花了一整天、花了200刀，给它重构了。然后他就拿着这个东西去跟他的CTO讲，CTO震惊了，他们就开始疯狂开会，整个公司后来就进入了一个癫狂地开始用AI的状态。

我是觉得，他跟我一样也是20年没写代码了。以我的水平来看，因为我也会写嘛，他一天搞出来的那个东西，大概率是没有办法替代那个1400万的系统的，是用不了的。但他已经表达出了这种癫狂的状态。老板已经把这个东西写出来了，你说它不好用、或者它上线就会塌，那是下一件事情；老板的态度已经表达出来了，就是楚王好细腰。这个时候下面人才会真的开始跟。他至少不会再告诉你说这写不出来，而是会告诉你说写得出来，但是会塌——那我们就开始解决它塌的问题；或者解决完不塌了，但是跟我们原有系统对不上，那就再解决下一个问题。大家就从「这个东西不能用、没有用，我也懒得搞」，变成「这个东西有用，但是不好用，那我们就去解决好用的问题」，这就会往前走一大步。

主要就是要通过这些动作来表现老板是认真的。只有老板认真搞起来，下面人才会认真搞起来。

那个 1400 万系统的段子是你全场最好的故事之一，建议把节奏再压一压（现在有点绕）：

> 今年春节刚过，一个工业的老板早上非常激动地打电话给我。
> 他说：**我们花了 1400 万做的那套系统，我上周花了一整天、花了 200 美金，我把它重构了。**
>
> 然后他拿去跟 CTO 讲。CTO 震惊了。整个公司从那天开始进入了一种癫狂的状态。

你后面那段自我拆台是这个故事真正值钱的地方，建议讲慢、讲透：

> 我得说句实话：**他那个东西大概率替代不了那套 1400 万的系统**，上线可能就塌。
> **但这不重要。**
> 重要的是——**老板已经把东西写出来了。**
> 从那一刻起，下面人不能再说「写不出来」，只能说「写得出来，但会塌」。
> **而这两句话之间，隔着整个公司的转型。**

楚王好细腰。

识别先进

看见公司里的超级个体

经常有朋友问我，那我怎么把这批人挑出来？我给的最简单的一个原则，就是你公司里面如果已经有人自己花钱买了Codex或者Claude Code，而且已经用了一段时间、每个月都在交钱，那这就是一群值得被培养的人。因为他敢于在自己身上投资，他能解决各种技术障碍把这件事搞定，而他持续花钱，说明他已经找到了有回报的领域。这样的人是很值得被培养的。

如果你要找得系统化一点，那么第一，他已经把工具用得比较好了，而且用的都不是那种聊天型的工具，而是WorkBuddy、Claude Code、Codex这种干活型的工具。第二，他能讲清楚自己端到端做成过什么事情，并且能告诉你过去两个月他改了哪些skill、改了哪些流程、什么做法已经不work了。第

三，他甚至已经发挥了影响力，影响了身边几个同事在做哪些事情。像这样的人，自己动手，又持续学习、持续进步、持续迭代，而且还能影响身边的人，绝对值得被挖出来。

别找最听话的，找已经在跑的。

案例：我的陪跑

我自己曾经给一个大型的传统企业做过一段时间陪跑，效果蛮好的。但我后来反思了一下，这个效果几乎跟我一毛钱关系都没有。主要是因为在这个陪跑的过程当中，他们挑选了组织里面一群人，让他们每一个半月到两个月去跟董事长汇报一次。所以小朋友们的激励主要来自他们的老板，以及他们自己非常想在董事长那边表现、要出成绩。每次都要有PPT，要显示我们AI又有什么进展、又做了什么东西。他其实是为了在董事长面前发光发热、露脸，所以小朋友们就比较积极去搞。在这个过程当中，我就不是一个逼着他们用AI的敌人，而是帮他们在老板面前闪光的支持者。

所以我觉得比较基础的、一定可以做的事情，就是多在公司里面鼓励小朋友们向老板做展示，让他们被看见。看见本身就是一个态度嘛。

案例：XMind

如果你觉得搞黑客松大赛、AI大赛太麻烦，想轻松一点，我觉得XMind那个方式也可以试。XMind是搞了一个小作文大赛，所有人都可以写一个话题，叫「我如何用AI」，然后往CEO邮箱里面发。CEO的Agent会评分，看谁写得好、谁写得不好，然后立刻发奖金；发完奖金之后，HR立即组织获奖分享，让所有人都看到，用AI是可以马上有正反馈的，而且这个正反馈会被所有人看到。

因为有些公司现在还在用影子AI，大家还担心被别人发现自己用AI，觉得你工作不认真，那我们至少要把这个风气给扭转下来。所以这一招大家也是可以学的。先把这些个体识别出来，以后你要做什么事，这些敢往CEO邮箱里发东西的人，就是第一批种子嘛。

案例：鹿客

当然，如果要更激进一点，也可以像鹿客那样。鹿客发了一封全员邮件，给员工的信只有两句话——我们会安排时间，你可以来做一个presentation，用尽任何办法向我证明你的AI能力有多强；以及告诉我你对什么业务感兴趣。

第一次组织这个活动的时候，50个人报名。但说着说着，现场就有人举手说，我也要说，我也要说。第二场就200个人了。这个过程当中就识别出了非常多已经把AI用得很好的超级个体。

他们还跟我分享了一个段子。他们之前的安排是实习生不参加这个活动，因为觉得实习生没什么经验嘛。结果后来允许实习生参与之后，发现这些年轻人其实越年轻越能干，在AI方面越是AI原生。所以后来就发现，这些年轻人也是可以激活起来的。

> **「用尽任何办法，向我证明你的 AI 能力有多强。」**

> 「告诉我你对什么业务感兴趣。」

可能的风险和冲击

不要提公司名字，我熟悉的两家公司。

我今年过年期间在混沌做过一次分享，也讲了一堂大课，那时候就在讲，应该全员拥抱像Claude Code这样的东西，要开小团队做新事，而且一定要做到矫枉过正——要让团队里面觉得老板就只看AI、有小人凭借AI上位，一定要有这种风气、这种口碑传出来，才说明你做到位了，否则就说明没有做到位。

但反过来讲，如果你真的做到这个程度，那很有可能你原来的那些领域专家会觉得受不了，他会离职，这是你需要承担的一个风险。比如说XMind他们公认的顶级开发就离职了，鹿客那个很懂华为IPD的产品大师也离职了。这个过程会伴随着一些阵痛，因为往往在原来的领域里最强的人最不愿意转，而原来半懂不懂的那些人机会成本最小，反而转得最快。这方面你要做好心理准备。

”最强“的人可能会走。

培养赋能

支持，培养，赋能

人找到了之后，第二件事就是赋能，因为不见得这些人就一定能干出所有的事。这块有一点跟大家想象当中不一样：我觉得大家不要想着买一堆课、全员培训一下，那没什么用。

因为面对一个新工具，能用好它的本来就只是一部分人。我们不需要所有人都转型，你只转那些最有意愿、最有能力的人，是最快的，然后用先进带动后进，允许一部分人先先进起来嘛。所以重点还是提前挑出一批尖子生，让他们先转，这是比较容易的。

把这批人赋能好，让他们起到示范效应，在整个湖面上掀起涟漪，带动一整个湖起波纹，这样比较有效。不是要求全员一定要干，而是把尖子生培养好，让他们发挥超级个体的能量，把组织带动成为超级组织。

AI 转型的终点是全员，但路径不是全员

案例：报销 Token

我在身边很多公司都听到一个常见的做法：把这批人挑选出来之后，给他们赋能的方式不是培训，而是帮他们报销token的钱。不管他是在工作上使用AI，还是仅仅为了他的私活在用AI，你希望他拥抱AI，最好就把已经愿意自己花钱买token的这些人挑出来，帮他报销，比如说20美金一个月；如果他干得好，甚至可以报销200美金一个月。

这样一来，你其实就把人挑出来了，钱也给到了。给到之后再问他：你看这200美金，你干出什么活来了？你得向大家展示。这就很顺。所以第一个赋能，是金钱上的赋能。

那些人**已经在自己掏钱了**。
你把这笔钱接过来，等于公司第一次对他说：**你干的这件事，是公司的事，不是你的私活**。

一个月几百块钱，换一个态度信号。这是全公司最便宜的一笔转型投资。

案例：安克

第二个赋能是能力上的赋能。这方面最经典的案例其实是安克。他们做的是火箭班，第一期是2025年2月份搞的，就在DeepSeek火起来之后。他们挑了50多个最懂业务的骨干，加上IT、加上AI混合编组，带着真实案例进去，每1~2天一次分享——分享我跟AI做了什么、踩了哪些坑，让大家看到其他人已经做成了什么。因为他们的观点是，让人产生变化不能靠讲道理，要先让大家看到，再让大家相信。

第二期是全价值链铺开，把大家聚到一起，一起建业务模型，把业务沉淀到平台里面，统一来做。第三期再做一个更严肃的建设，还有认证体系，考C照什么的。但总体上讲，他不是在做培训，而是一两个月的沉浸式共建，而且是脱产的。

脱产完了之后，大家的能力提高了一节，相互促进过了，再把大家放回到原有的部门职能里面，相当于催化剂，把原有的部门带动起来做AI转型。

他们也会把这些公开出来，让大家都能看到别人已经做成什么样子、别人在什么水平。对于其中做得比较好的人，他们会叫「new人」，就是英语的new。对这些人，他们会给几倍工作产出的期待，同时也给几倍薪资回报的期待。

案例：王博龙

另外还有一条路线。我们不仅仅要赋能那些边缘的小年轻、这些先进分子，很多时候，那些非常重要的老臣也是需要我们去赋能的。

我知道有一家非常大的公司，他们挖了非常强的人来做内部的FDE。这个FDE的工作，就是到公司里面所有核心骨干那边，一个一个帮他们在电脑上装好Codex，一个一个访谈，问他们最核心的痛点在哪里，当场在他的电脑上演示Codex怎么帮他完成这个工作；平时还拉群，一对一上门提供服务；甚至把新电脑装好，做成一体机，给几个董事会的高层，一个一个教他们怎么用。

这其实是很有必要的。不管这些人出于利益上阻不阻挠，至少要让他们对这件事情的效果不要有完全错误的观念。像我们这样的老灯特别讨人厌的一个地方，就是我们掌握权力，但很多时候对世界有非常固有的想法——哪怕口头说我知道AI很强，实际上脑子里的图景都是错的。所以这个时候，光请一个我这样的人过去培训是没用的，最好是你自己在公司里设一两个这样的岗位，请一两个这样的小朋友，去帮他们一个一个解决。

我最近在一个商学院里面，我们组里有很多同学问关于AI的事情。问着问着，结果全是一些最基础的问题，安装啊、刷信用卡啊之类的。我就拉了一个群，安排了一个刚从英国回来的小朋友，帮他们一个一个上门安装、解决这种问题，大家好像都挺满意的。

所以很多时候，卡点其实不在那些高级的AI转型方法论，而在于最基础的、前面几步手把手的那三四个小时。大家可以在公司里成立一个这样的小组织，去帮人家把一开始那几个小时的困难给解决掉。

像我们这种老灯最讨人厌的地方就是：**我们掌握权力，但我们对世界有非常固有的想法**。嘴上说「我知道 AI 很强」，**脑子里的图景全是错的**。
所以请一个我这样的人去讲一堂课，**没用**。

卡点从来不在方法论，在最开始那三四个小时。

分圈

分圈：设计好独立业务小组织

人有了超能力，工作范畴就要变大

当我们真的做到了AI赋能之后，每个人就会从一个螺丝钉变成一个超级个体、超级赛亚人、钢铁侠。你再要把他塞到原有的工作岗位上，就有大量的不匹配，他已经放不进原来的萝卜坑了。这个时候，为了让大家可以充分发挥，应该让每个人做的事情范畴变得更大。

AI不一定能够完成当下任何一个工作的100%，但它很有可能可以完成每个工作的70%，所以一个人现在可以覆盖的范围是变大的。整体上讲，你要还是把一个工作切成10份、让10个人来完成，那每个被AI赋能的人都会觉得很憋屈：要不然是他空有一身本领，不敢越雷池一步；要不然是他越了雷池，会被其他人指责说你过界了。这样组织上的摩擦反倒会增加。

不越界——空有一身本领，只能干那一小块，做完就等交接
越界——马上有人说你过界了、你不专业、这不是你该管的

两条路都通向摩擦增加。所以不是人不行，是坑的形状不对

个体能力变大，团队规模就可以变小

同样的，如果每一个人可覆盖的专业范围变大、工作产能变大，那么完成一个业务所需要的团队就变小了——不需要那么多人，不需要那么多层级，不需要那么多环节。这样才是真正的提效。我们刚刚讲的自动帮他做转接，也是一种提效；但实际上更有效的提效，是让工作的交接大幅度变少，不需要那么多交接。

OpenAI Codex PM Rohan Varma："**人对人协作变成了大瓶颈。我们每条产品线其实就一两个工程师从头包到尾**——人一多协作开销巨大，而且慢，产品决策一周才发生两三次；当工程师能跑得飞快，大量微决策**理想情况下他自己就该拍板，不需要来找我**。"

Codebuddy subo：「**AI 时代任务越拆越大：整块交给一个人加 AI，比切碎分给五个人再拼装要快得多**。」

Botlearn 张栖铭："真正开发的时间不长，大头是开会、讨论、相互对齐……所有流程都没有变化，本质上就是旧的流程被加速了，你在中间纠结的事情一点都没变"；配合 AI 写文档反而双重工作量——"我又要理解 AI 写的东西，又得想办法把 AI 写的东西传递给别人，脑子都烧坏了"

AI 时代任务越拆越大。

让小团队用 AI 独立完成业务

基本组织单元，从”职能“变成”业务“

我们现在的基本组织单元都是一个一个专业性的职能：你是设计，他是产品经理，他是研发，然后我们跨职能部门沟通。那未来如果要变小，并不是说这样一个个部门变小，而是单独一个小团队就应该完成所有的事情，实现这个业务端到端的交付。

过去因为人是专才嘛，只能按技能来分，然后用流程把技能串起来。现在每个人的技能都多了，一个基本单位就可以完整地覆盖一个闭环。AI补齐了长尾的那些职能，又把必须的那几个人的专业性装在里面，那么组织的切法就会从按照职能横切，变成按照结果、按照业务结果、按照客户满意来纵切。

案例：安克

我们前面举了很多安克的例子。去看他们分享的案例你会发现，他们很多长流程，比如原来10步的，现在已经变成了4步；原来要十几二十个人的，现在3个人、4个人就能完成。其实也是在把整个层级压扁，把流程变短，把组织变小，把圈子变小。

他们有专门的变革负责人，以前是麦肯锡的，就负责推进组织的专小化。他们在识别跨职能小团队作为最小的作战单元，因为一个小团队可以共享同一套数据和流程智能体，agent跨职能流转，上下文都可以共享。内部自然会形成角色融合，少数人覆盖全局，一人多角色。这样组织小了，决策就变快了。

安克这条最有价值的是「10 步变 4 步、20 人变 3 人」这组数，建议直接上屏——它把「组织变小」从主张变成了刻度。

案例：XMind

他们撞上的第一个深层问题不是没想法，也不是没人愿意负责，**是责任和权力被拆开了——一批人离产品和用户最近，要对结果负责，但资源不在他们手里。**

XMind也做了这个改革。他们管这个东西叫圈子，从以前的很多部门，改成了分成非常多的圈子，每一个圈子有自己的独立业务，要对完整的端到端业务负责。以前那些中台、后台，其实全部都拆到了圈子里，拆圈子的规则叫高内聚、低耦合。

不同的人可以到不同的圈子里面去承担角色。一个设计师，可以去不同的圈子里面，这个圈子我投入5%的精力，那个圈子投入20%的精力，在不同的圈子里面做贡献、扮演角色。这样就把人给拆碎掉

部门为什么消失的具体机制：「以前**增长部门承担全公司的增长，业绩就变成了增长部门和营销部门负责的事，他们背业绩；现在是每个产品背自己的业绩，所以他们的增长归自己管**。」结果：「以前在他们那里就会不停地排期，**现在产品团队跟增长团队无缝衔接**。」

我问Mango，你这样跟矩阵式管理有什么区别？他说，以前有两个老板，现在就只有业务方老板了。我说那职能部门怎么办？他说职能部门就退化成了一个社区，有点像行业协会的存在。

他们想做这件事情，其实是因为发现之前对业务负责的人太少了，很多人都在支持线上，而不是在业务线上，不是在对结果负责——设计有设计线，研发有研发线，营销有营销线，所以要做业务、要往前走，就要一轮一轮地解释、排期、确认和等待。这个在以前业务单一、核心产品运行稳定的时候未必是个大问题，但当他们想要开大量新产品、要迎合AI的高速变动的时候，这个结构就明显不合理了。

所以他们开始重新划分，一起研究怎么重新分配活，把任务重新组织成按业务来分圈子：每个圈子里面要哪些角色？要完成哪些功能？然后就会发现，越来越多的人可以按角色进入到不同的圈子里面，对岗位边界的理解也就松动了。所以没有人再去问这件事属于哪个部门、哪个岗位、谁有权限做，而是问这个业务目标是什么、我怎么样可以贡献、把这个事给搞定。

不再是先定义我属于哪个部门、我的角色是什么；现在是谁能够把这件事做出来、做出结果，谁就是那个角色。

放权让前线拥有炮火

炮火足够廉价，应该下放到连队

华为的任正非之前说，我们要让听得到炮火声音的前线来呼唤炮火。其实就是要给前线、给业务方、给真正懂业务赚钱的那些人授权嘛。但之前做不到。一个原因是太贵了——你要开发一套大型系统，你要改生产流程，这些东西都很贵；另外一个就是这些设备太难了，里面有专业性，你不可能指望前线一个个嘴上没毛的小年轻就能讲清楚后面该怎么做。所以一个是难，一个是贵，导致最多也就喊一喊让前线呼唤炮火，而且往往还呼唤不动。

后台只要有一群人在管炮火，他有自己的专业壁垒，有自己的组织，就会形成自己的利益、自己的视角。这个时候前线呼唤是呼唤不动的。我刚毕业的时候是在一个很有名的电商公司，那时候这家公司是以IT能力见长的，我们这些业务方提IT需求的时候就特别艰难，因为从IT的角度看我们，就觉得你们啥都没想清楚就瞎提，所以基本上很多想法都被无视了。

这也挺正常的，因为当时我们确实可能脑子不清楚，讲得不清楚。但从我们业务方、从前线打仗的角度，就会觉得很憋屈，觉得公司里面衙门很多，我要帮公司做事还要求爷爷告奶奶，整体的积极性是受打击的。

但现在AI贡献的是生产力。生产力贡献进来之后，它把一件事情的专业理解门槛和沟通交流门槛大幅度降低，也把生产的成本大幅度降低。这个时候，我们的前线就应该拥有越来越多的权力去影响炮火，甚至直接拥有一部分炮火，把它放到前线来。让离客户更近的地方拥有更多的话语权，现在是可以做到的。

一旦完成这个，那就不仅仅是一个交付单元的重组、从科层部门变成圈子，而是一个组织动力源的翻转——从中央的计划驱动，变成了客户需求拉动。

大原则：过度授权，紧缩预算

那如果让小团队去完整负责端到端业务，我们应该怎么给他们资源支持呢？这里面有两个大原则。一个叫紧缩预算。这是Claude Code包括无数家总结出来的经验：我们开新团队、让他们用AI的时候，就是要给不够的钱、不够的人，给更大的目标，才能把人逼出来，才可能做成新事。更多的预算可以用来给他们激励，但不应该让他们花在请更多的人、买更多的系统上，而应该让他们用AI想出新的方法去做，用AI来协助提效。

前面这一条因为符合老板降本增效PUA的目标嘛，所以一般人很好接受。但另外一件事很多人就很难接受了，叫做过度授权。过度授权就是，你要让这个小团队基本上不需要向外面的人走审批流程。一旦你让他还要走外部的审批流程去要资源、去求爷爷告奶奶，整件事的速度又会被降下来，他们就没有办法对结果完全负责了。所以你可以给他们很少的钱，但应该让他们受到很少的外部干扰，几乎不用走其他的审批流，几乎不用求外部资源，什么都可以内部搞定；以及他们如果需要公司授权，可以得到你作为CEO的最高授权，什么事情都可以秒过。这样才能弥补他们在资金和人力上的不足。

我之前在大厂负责过创新项目孵化，最重要的事情并不是你要给他多少钱，最大的卡点从来不是卡在钱上面，而是卡在公司的流程上。比如说一个小的MVP实验，需要你去签一个战略合作协议；什么东西都需要你走三十二层合规。所以最重要的卡点不是钱，不是资源。你要解决的其实是他对别人资源的依赖，以及合规、流程方面对他的卡。这就是对前线团队最大的支持。

稀缺是给了他足够的使用AI去降本增效的动机，而授权是给了他足够强的能力。

最大的卡点从来不在钱。**一个小的 MVP 实验，要签一个战略合作协议；一个东西要走三十二层合规。**
所以你要给的支持不是预算，**是替他挡掉对别人资源的依赖，和流程上的卡。**

案例：艾语智能

整体上讲，如果你还是希望按照原来的组织结构和流程去框定，那么所有单个任务层面的提效都会被掩盖掉。只有当你充分授权一线的人，让他们把AI用起来，让他们不用走那么多流程审批，才能够把整件事情做出来。

举一个艾语智能的小例子。我跟他对谈、问他怎么做AI改造的时候，他说他们现在一线员工都有数据库和代码权限，遇到什么问题可以自己改系统。比如说有一个被告的名字里面有生僻字，前线一个不是特别好大学毕业的、很普通的基层员工，觉得我可以改一下系统，允许我加拼音，那一线就可以直接改这个系统，不需要走需求流程，直接就做了，整个就快了。如果不给这个授权，他还是得提需求，提到后台的开发部门，开发部门再排期，这么一个小需求，整体效率就慢下来了。

当然了，如果你要安全，还是要把我们前面讲的那些检查点全部设计好。所以整体上讲，你要提效，就要充分相信AI；你要管理安全，就要设计好Harness。这都是两个工程的活、两个设计的活，而不应该把一切都套在旧的那套管控组织里面，让整个企业的效率降下来。

改系统的**不是工程师，是山西作业中心一个大专学历的基层员工**。这个细节比任何论证都有力。

> **你那句「提效靠授权，安全靠 Harness」是这一页的收口，建议做成一句上屏：**
>
> > **「要提效，就充分相信 AI；要安全，就设计好检查点。这是两个工程的活，不是同一个管控流程能一起解决的。」**
>
> **再补一条他们的防线做法**（回答"那安全怎么办"）：他们**不靠人管，靠工程——提交前 pre hooks 做代码检查、AI 写的东西由 AI 审、什么样的提交直接拒绝、权限和边界提前定义好**。原话：**「让 AI 去解决 AI 的事情。」**

案例：XMind

比如说XMind，他们就把各个职能部门全部拆散掉，放到业务的圈子里面，每个圈子都要负责端到端，直接搞定结果。他们的原话就是，从制定规则、审批流程，全面转向赢得结果。这其实就是我刚刚讲的，要让在前线的同学们来指挥炮火。

具体来说，他们首先是分圈，刚刚我们讲过，是架构重构，按照价值流切出能够自转的小闭环，让结果、权力和资源尽量回到前线。第二是分活，让碳硅分工，不是按照旧岗位拆工作，而是把它做成「端到端结果 × 人＋AI」的编队，把人从执行岗位推到判断岗位。最后一块是分钱：既然你要对结果负责，我就按照结果给你相应的激励。所以整个圈子的钱是跟这个圈子的表现相关的，圈长最终有权决定怎么把这笔奖金分下去。

XMind的Mango跟我讲，他们不同的圈长做法很不一样：有些雨露均沾，每个人平分；有些分得非常剧烈，有人给很多、有人给很少。这样一来，圈子里的人就会流动，觉得自己受到不公正待遇的，可能就投奔其他圈子去了。整个生态就会非常自由、非常灵活。但反过来，它对圈长的要求就非常高，相当于在内部演化出了无数个创业团队。

案例：美图

美图已经把这件事情实践得非常多了。一家2000多人的公司，切成了很多个不超过10个人的创新工作室，每个团队不超过10个人，最高可申请1000万元启动资金，以半年为期验证。如果跑通了，核心团队就利润分红，甚至可以独立融资；如果做不成，这个工作室就关掉。工作室内部取消所有常规会议，也没有固定的产品经理，大家混在一锅里去搞。有决定权、有预算权，甚至还可以分钱，这就齐了。

大家看，现在的AI组织转型，有一部分转出来的形态会更像是孵化器或者Venture Studio这种模式，组织变成后台的一个赋能中心，前台是一个一个的创业小组。组织需要思考的是，我如何把创业者、把那些天然对市场和生意有敏感度、又能把AI杠杆用起来的人留下来，给他们提供什么样的支持。所以公司会越来越像孵化器。

我有两年一直在云九资本，在一个美元基金做孵化器嘛。现在跟所有组织聊的时候就会发现，大家要改的方向其实也是孵化器，就觉得太好了。

案例：Kavak

关于不停地向前台授权，我最后还想分享一个案例，就是墨西哥卖二手车的品牌Kavak。他们现在95%的交易从头到尾是没有人参与的。而且他们做得更过分——不是像内部创业那样给每个创业小团队授权、让他们去思考应该做什么、后台提供支持。我刚刚讲的是把权力从支持执行的职能部门推向前台的业务部门，他们做得更极端。

他们不是为业务流程设计Agent，而是为每一个客户设计一个专属的Agent，给这个专属Agent设定的是业务目标，就是这个客户长期的满意度和长期的价值。这个Agent思考的就是，我如何从这个客户身上赚取尽量长期的Lifetime Value；然后它会反过来想，那我需要帮这个客户搞定哪些事情、需要调取哪些资源，它会从Agent的角度、从AI的角度反过来去问公司，向公司提出需求。如果做到了，它就自己沉淀经验；如果没做到，它就反过来向公司提需求。

而且因为每一个客户都有一个Agent，它就不是抽象地去思考我们要把转化率从多少提高到多少，而是会到每一个具体客户的需求当中去挖掘潜在的商机和机会，千人千面地提供服务，然后向后台提出支持性的炮火需求。这其实是把权力进一步往前放，甚至直接放到了用户那边。

逻辑上其实很像我之前在混沌分享过的一个道理，叫JTBD，Jobs to be done。所有的公司，我们号称是在干活、在做产品，但实际上我们是在帮用户完成他的任务、满足他的需求，那边才是一切价值的原点。所以我们真正要做的，不是所谓做一个AI改革的组织，也不是做一个更强的产品，而是更有效地帮助用户完成他的JTBD、达成他的成果。从这个角度去做优化，把一切权力的原点放在客户侧，放在per customer这边，就是一种更极端的把权力往前推的方法。

建议补几个具体到吓人的细节（这些数字会让台下记住它）：

> **96% 的客户交互、95% 的交易由 agent 完成**，还在一座城市跑了一个 **AI CEO**（6 周，目标利润翻倍、实际 1.5 倍）
> 每天实例化 **10–20 万个 agent，每个自带一台虚拟机**，记住这个客户几年的交互；**醒来工作三分钟或三天，然后给自己定个闹钟去睡觉**

你说的「它反过来向公司提需求」有一句更狠的原话，建议用：

> agent 撞墙的时候，**它调一个 API 说「我需要帮助」，而 API 那头是人。**
> 他自己的总结：**「画成组织架构图，就是人类团队向 agent 汇报。」**

以客户为中心第一次不再是口号，因为你终于配得起。

总结

总结来说，你要落地，要考虑的事情并不是什么AI大模型、什么算力、什么GPU，这些东西都不是第一步需要考虑的。你要考虑的其实是准备好自己的系统，让AI方便进入你的公司，让它能够看到你的东西、理解你的东西、操作你的东西。

第二步，你需要准备好能够跟AI相处的人才。一方面，把你已有的、已经熟悉AI的人才挖掘出来，让他们被看见、被激励，然后给他们赋能；另一方面，公司里那些业务骨干，你就派专人一对一上门帮他们转化过来，因为这批人是不能换的嘛。

最后，你要在公司里面拆出一些小圈子，让一些闭环比较短的业务端到端地完全用AI来完成。对这些小圈子，可以给它更紧缩的预算，但是要给它超量的授权，这样才能让小圈子爆发出10倍、20倍的生产力，给其他人打样。然后的话呢。