Software in the Age of Agents | The a16z Show – 图文解说

> 一张关键画面对应一段完整解说，按原视频时间推进。

文章摘要

本期围绕《Software in the Age of **Agent**s | The a16z Show》展开，按时间顺序讨论开场：软件为什么粘，以及“无头软件”要讨论什么、Headless Software：从界面访问转向 agent 访问、企业软件为什么粘：UI、流程、依赖和单一事实来源、SAP 不会因为数据库加 API 就消失。**以下内容保留完整问答、例子、数字、画面信息和限定条件，适合先读这段摘要建立框架，再结合每个关键帧逐段阅读。**

01. 开场：软件为什么粘，以及“无头软件”要讨论什么（00:00–02:14）

场景 1

本节主旨 开场：软件为什么粘，以及“无头软件”要讨论什么

完整内容

Steven Sinofsky **先给出一段预告式判断：软件之所以粘，很多时候不是因为表面的界面，而是因为人如何与软件互动，以及软件下面保存的数据和业务逻辑。**很多人低估了企业软件里被编码进去的东西，以为把最琐碎的流程自动化之后，事情就已经结束了；但每次自动化一个旧问题，都会出现新的问题。Oracle 的 Larry Ellison 曾经批评企业软件“太蠢”，因为每家公司都把它定制得不一样，但这恰恰说明企业软件不是一个简单的数据库。

他接着指出，现在常见的误解是：只要有一个 Postgres 数据库和一组 API，就可以直接替代 SAP。**这完全不成立，因为 SAP 里保存的逻辑和业务规则，比数据碰巧存在哪个数据库里重要得多。**技术发生指数级变化时，人们往往等到变化已经发生，才意识到自己一直在用线性方式外推。最大的机

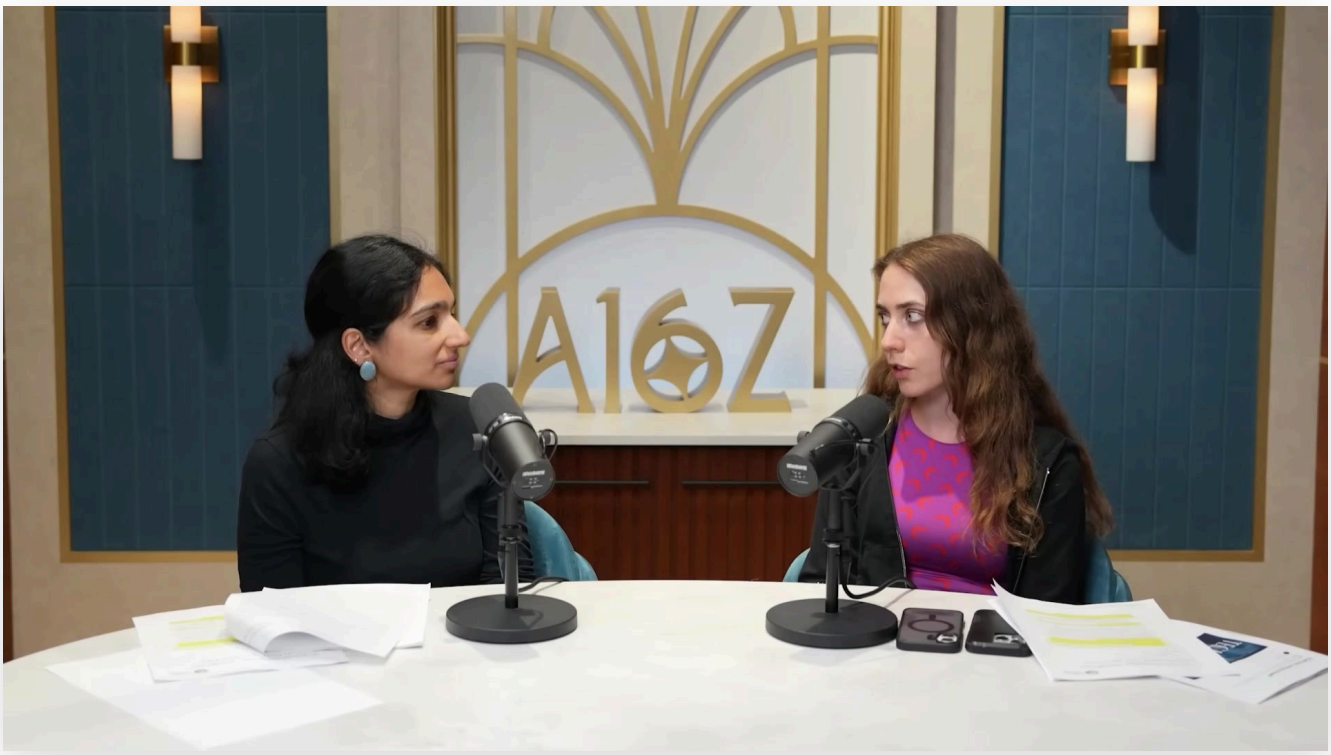
会，也会出现在这种变化过程中。

随后 Elena Burger 介绍本期节目。她邀请 a16z Enterprise team 的 Partner Seema Amble，以及 a16z Board Partner、前 Microsoft 高管 Steven Sinofsky，讨论 Seema 写的文章《Is Software Losing Its Head》。**Salesforce 宣布推出 Headless 360，这引出了本期问题：当软件不再主要由人通过界面访问，而是由 agent、API 或其他接口访问时，SaaS 产品和软件本身会发生什么变化?**

画面说明

开场先出现 Steven Sinofsky 的远程单人机位，随后切到 Elena Burger 与 Seema Amble 的 a16z 棚内双人机位；画面中的姓名条确认了三位说话人。**该段主要是节目介绍与问题设定，没有 PPT 或需要单独解释的图表。**

02. Headless Software：从界面访问转向 agent 访问（02:14–10:00）



场景 2

本节主旨 Headless Software：从界面访问转向 agent 访问

完整内容

Seema 解释，“headless software”并不是一个全新的技术名词，但最近因为 Salesforce 的 Headless 360 宣布而重新进入公共讨论。传统软件围绕人的访问方式来设计：人通过 UI 操作工作流，填写数据，软件再把这些数据记录下来。**在 agent 主导的世界里，agent 可能根本不会访问 UI，因此界面的重要性会下降；真正需要保留的是数据、业务逻辑，以及存放在软件内部的那一层规则。**

她认为 Salesforce 的 Headless 360 更像是对市场变化的承认，而不是产品本身突然发生了根本变化。**原来暴露出来的 API 被重新包装和命名，Salesforce 只是意识到 agent 需要访问 CRM 数据。**Notion 的情况可能更自然，因为它的用户更可能自己构建 agent。访问方式也不一定非要是 API 或

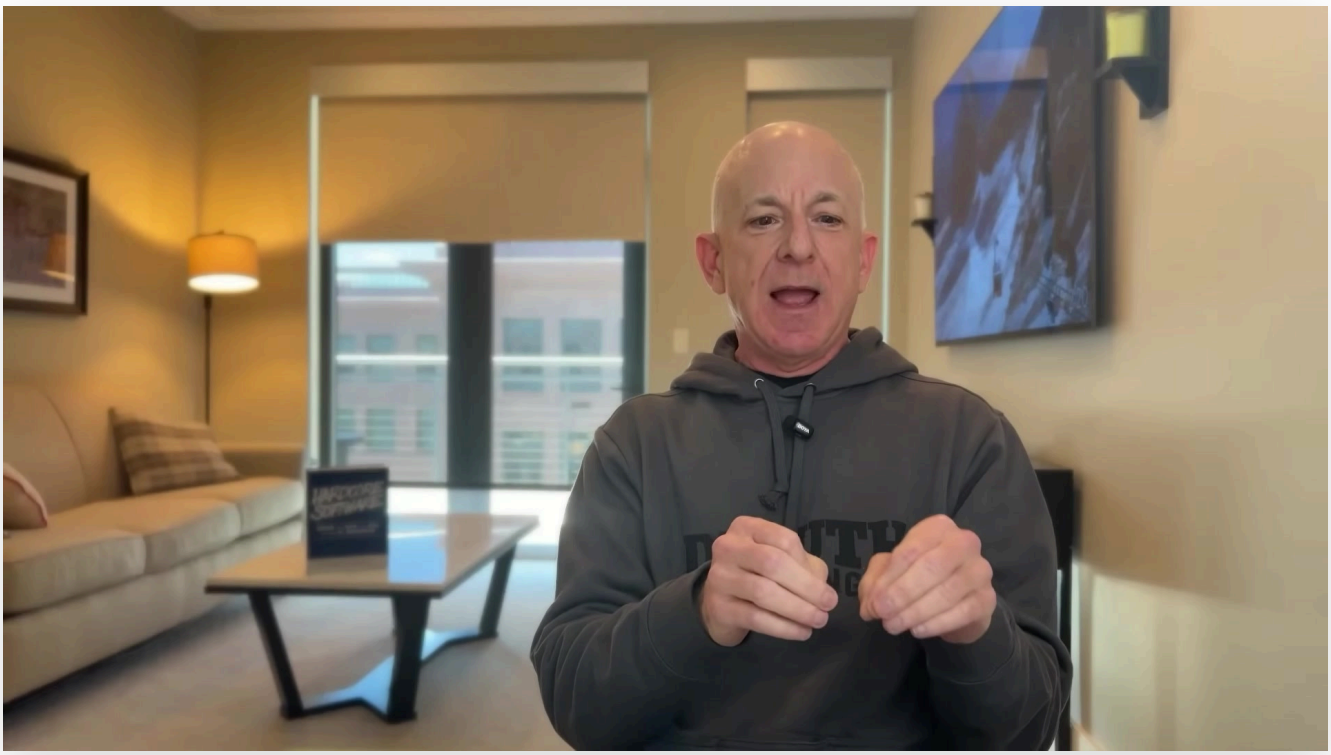
MCP server，Slack 里的 chatbot 同样可以让人不登录 Salesforce，就读取机会、客户和进展信息。Seema 提到，Slack 中 agent 的使用量曾经出现很大的增长，这说明“人必须进入 UI”正在变成可选项。

Steven 说，行业现在处于“定义地狱”：新一轮技术变化会带来很多新词，但有些词只是给旧东西换了一个更酷的名字。他把 agent 的工作拆成三类。第一类是 lookup，即查一个东西，这其实是很多系统一直都能做的事；第二类是 do，agent 要代表某个具体的人改变系统，这就涉及凭证、权限、是否需要一个付费席位等企业软件问题；第三类是 analyze，agent 要跨多个系统查找资料、反复迭代、比较不同模型的结果。**分析最有潜力，但也最需要验证，因为每一步都可能产生幻觉，必须能够确认整个分析链条是正确的。**

画面说明

画面以 Seema 的棚内近景、Elena 的近景和 Steven 的远程机位交替出现，偶尔出现三人分屏。**这里没有额外的图表，关键视觉信息是三人的固定身份与对谈轮次。**

03. 企业软件为什么粘：UI、流程、依赖和单一事实来源 (10:00–18:02)



场景 3

本节主旨 企业软件为什么粘：UI、流程、依赖和单一事实来源

完整内容

Elena 把讨论推进到下一个问题：历史上究竟是什么让软件变得粘，agent 又会怎样破坏这种粘性？Seema 先回答说，很多企业软件的粘性来自它围绕人的工作方式建立起来。用户高频读写，周边有下游工作流，组织里还有没有写进软件的 SOP、经验和肌肉记忆。**Salesforce 之所以粘，不只是销售人员习惯每天打开它，还因为财务要依赖 Salesforce 的输出做账单，市场团队也要依赖这些数据。**

企业还需要一个单一事实来源：某个客户是否成交、谁负责它、收入和付款数字是多少，都必须在一个被组织认可的地方记录。**到了 ERP、薪资和合规场景，这种要求更强，因为审计不能接受多个版本互相冲突的数字。**于是软件会因为使用频率、上下游依赖、法律合规、组织默认选项和长期习惯而

变得耐久。

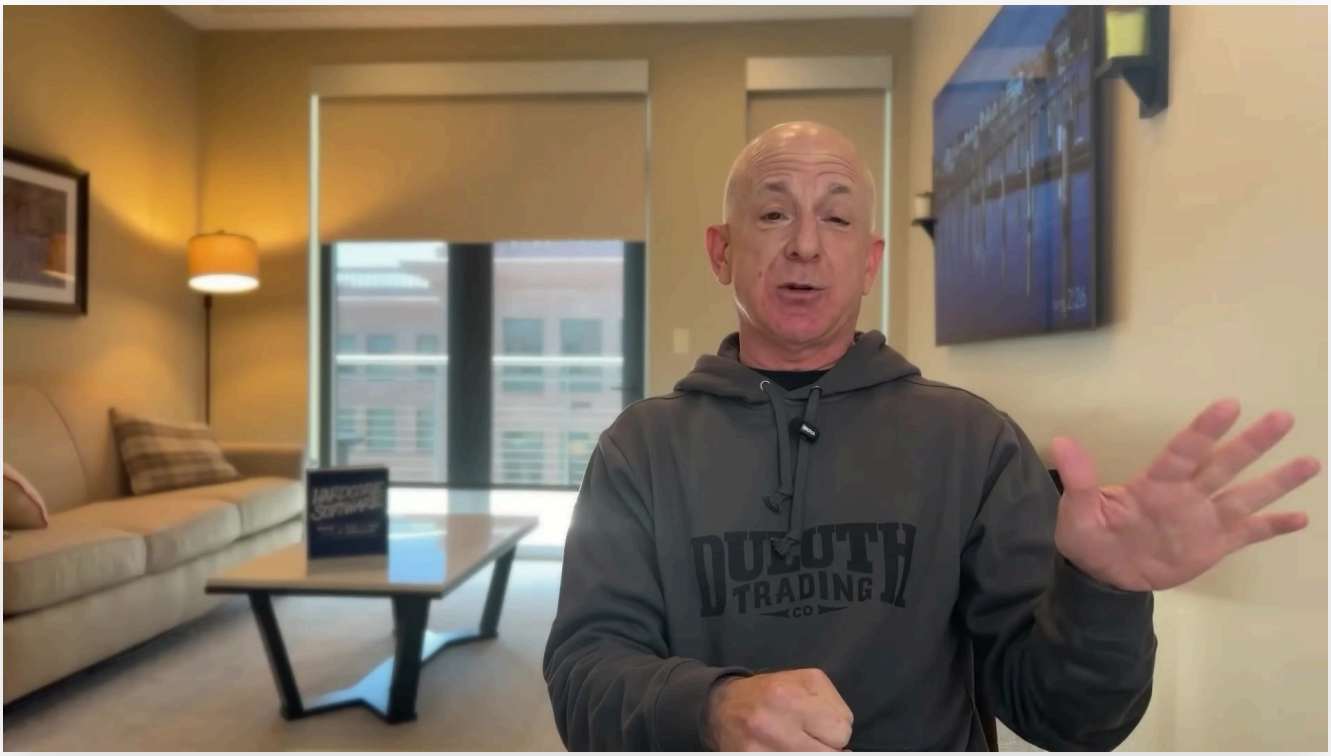
Steven 补充说，最粘的软件往往是能帮公司收钱的软件。不同部门会给出不同的“粘性”解释：合规人员说是 HIPAA 合规，管理员说是新用户开通，普通用户说是肌肉记忆、快捷键，甚至工会规则也可能成为替换障碍。**卖软件时最重要的是先让它真正被用起来；当客户威胁要替换时，销售团队应该认真听他们到底舍不得什么，这些反复出现的原因才是真正的粘性来源。**

他用 Outlook 举例：没人专门开会决定要把“多人共享日历、委托访问、重复会议例外处理”做成护城河，但这些细节会让大公司无法轻易迁移。Seema 说，软件一旦延伸到组织各处，就会渗入人们做事的方式，形成嵌在软件和人脑中的隐性理解，很难抽出来。**Elena 随即引出 Steven 对《The Death of Software》的观点，追问为什么“软件末日”被夸大了。**

画面说明

选用 Seema、Steven 和双人全景的代表性画面，重点是保留问题、回答和插话关系；没有用镜头切换来制造额外段落。

04. SAP 不会因为数据库加 API 就消失（18:03–22:55）



场景 4

本节主旨 SAP 不会因为数据库加 API 就消失

完整内容

Steven 说，比 SAP 更粘的，可能是保险公司几十年前写下的后台软件。那些系统编码了每个国家、地区、税法、汇率和边界条件，甚至需要 COBOL 程序员继续维护。**它们之所以不会消失，是因为它们把外部监管力量和公司的运营规则固化进了软件。**SAP 对大型汽车制造商和 Walmart 也是类似的：拿走 SAP，不只是拿走一个工具，而是拿走了公司定义业务规则的方式。

Seema 强调，企业创业者常见的误解是：只要建一个 Postgres 数据库、接几组 API，就可以替换 SAP。**SAP 之所以需要多年实施，不只是系统集成商效率低，而是它必须按照企业真实运行方式定制。**Steven 进一步用费用报销说明规模差异：40 个人的报销可以让一个人手工处理，或者拍照、OCR、分类；但到了 20 个国家、10 万名员工，就会叠加不同国家法律、公司政策、货币和组织规则，整个业务已经被编码进去。

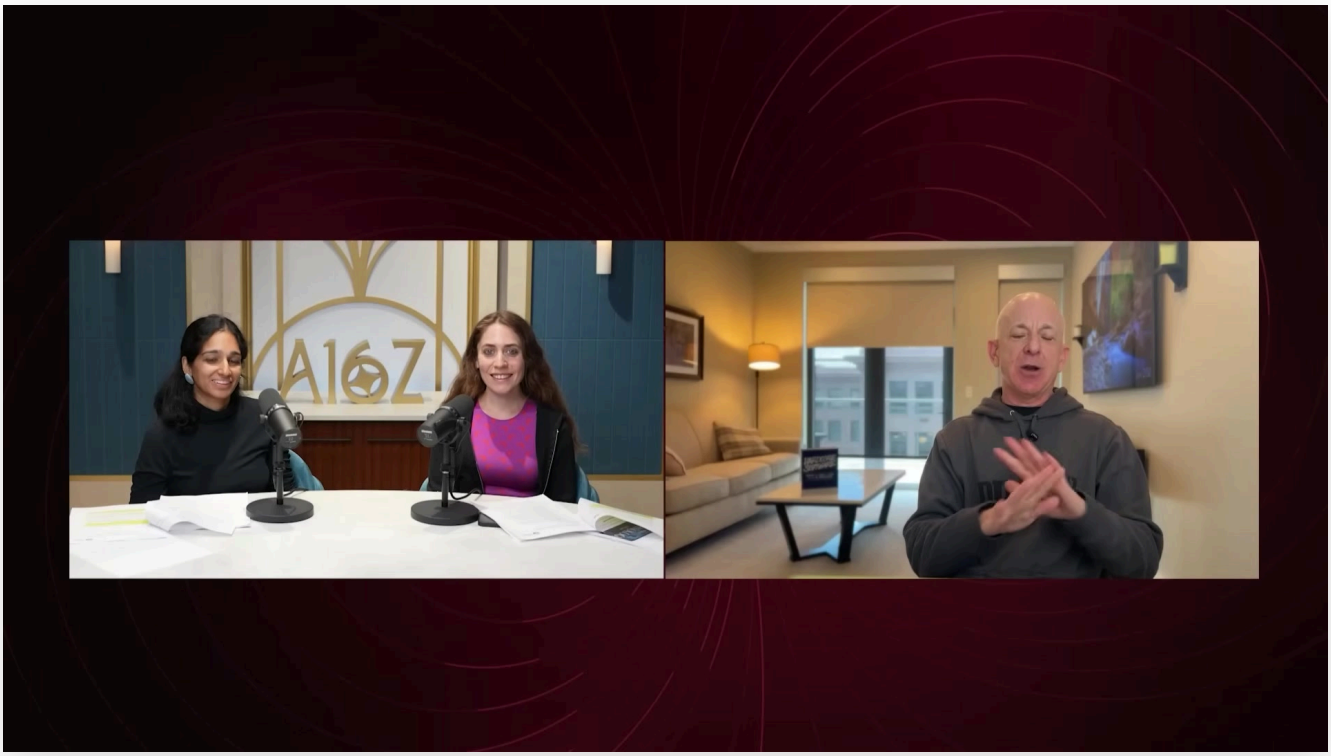
他回到 Larry Ellison 的“**80%** 解决方案”观点。汽车公司看起来都在造车，但真正区分 Ford、Toyota、GM 和 Daimler 的，是它们如何决定生产什么车、采购多少材料、对冲哪些货币、何时招聘和推出产品，而这些决策都进入了企业资源规划系统。大家可能看着同样的 SAP 屏幕，但选择的屏幕、定制的流程和操作方式不同。

Steven 又讲了 Goldman Sachs 的 Excel 故事。早期他们去推销 Excel，认为 Excel 比 Lotus 1–2–3 好；Goldman 的人却说：“我们从 Excel 赚的钱比你们卖 Excel 赚的钱还多。”**因为 Goldman 在 Excel 上写了插件、代码和工作流程，它的应用方式已经高度差异化。**这个例子说明，企业软件的价值不只是功能列表，而是客户把自己的运营方式编进了软件。

画面说明

本段主要使用 Steven 的远程近景和 Seema 的棚内近景；视频没有展示 SAP 界面或 Excel 图表，因此画面说明只保留人物、机位和姓名条，不虚构不可见的系统画面。

05. Vibe coding 不能直接重建企业软件（22:55–28:34）



场景 5

本节主旨 Vibe coding 不能直接重建企业软件

完整内容

Seema 讲到，她听说一家千人以上的成长型公司让负责人内部重建 Salesforce。**对方以为只要知道字段、导入数据就够了，但真正困难的是：什么应该被记录？**组织边界如何围绕它建立？谁长期维护？业务变化后谁来调整？很多人可以 vibe code 一个 CRM 或项目，但项目一旦过时、没人维护，就会变成新的负担。

她指出，现实中很多创业公司并没有把 SAP 完全拿掉，而是在 SAP 之上和周围解决令人头疼的问题。**AI 的一个重要用途，是让人用自然语言查询 SAP、连接不同地理区域和数据表、自动生成定制报告，而不必自己跑 SQL、翻很多屏幕或重新走一遍 SAP 定制流程。**UI 变成可选入口，但数据和业务逻辑仍然必须存在。

Steven 接着说，企业软件几乎总能完成客户想做的事，只是客户不知道如何让它完成。**最常用的功能往往不是系统内置分析，而是“导出到 Excel、CSV 或 PDF”。**导出是一个逃生阀：用户把系统里处理不了的资料拿出来，再自己分析。语言模型现在能更容易地消费这些导出的文件，例如把 20 份 PDF 放进模型，分析一个特殊时间区间、不同货币或系统 UI 处理不了的异常。

这些临时的业务流程，可能就是下一批产品的起点。CRM 早期也只是客户经理用 Excel 记录客户，后来才出现 Siebel 和 Salesforce。**今天一些通过聊天访问 SAP 或 Salesforce 的应用，本质上是在利用语言模型合成和编排非结构化信息。**Elena 补充说，Salesforce 还像销售团队的执行机制，要求团队把业务状态记录下来；当 agent 参与外呼和消息发送时，系统不仅要拿到字段，还要理解更隐性的业务上下文。

画面说明

场景中出现棚内双人和 Steven 的分屏，视觉作用是区分“谁在提出企业重建问题”和“谁在解释导出数据、语言模型与新产品机会”；没有额外证据图，因此不增加重复头像。

06. Agent 真正需要的是上下文和例外规则（28:35–32:49）

场景 6

本节主旨 Agent 真正需要的是上下文和例外规则

完整内容

Seema 说，在传统 Salesforce 里，销售人员负责把数据填进去，记录客户状态。但在 agent 世界里，agent 可能要根据 CRM 里的信息发起外呼或发送消息。**它不太在意字段在 UI 里怎样排列，也不在意需要点多少次，但它必须知道每一种情况该怎么处理。**

这就是所谓的 context graph：**不是简单地再加几个字段，而是把边界情况、权限、政策以及组织里没有被正式写入 Salesforce 的规则带进来。**比如同样是发送一封邮件，亚洲客户和美国客户可能有不同的回应方式；这些规则可能只在某个人的经验里，而不在 CRM 字段中。agent 要代表业务行动，就必须理解这些上下文，而不是只读出一组数据。

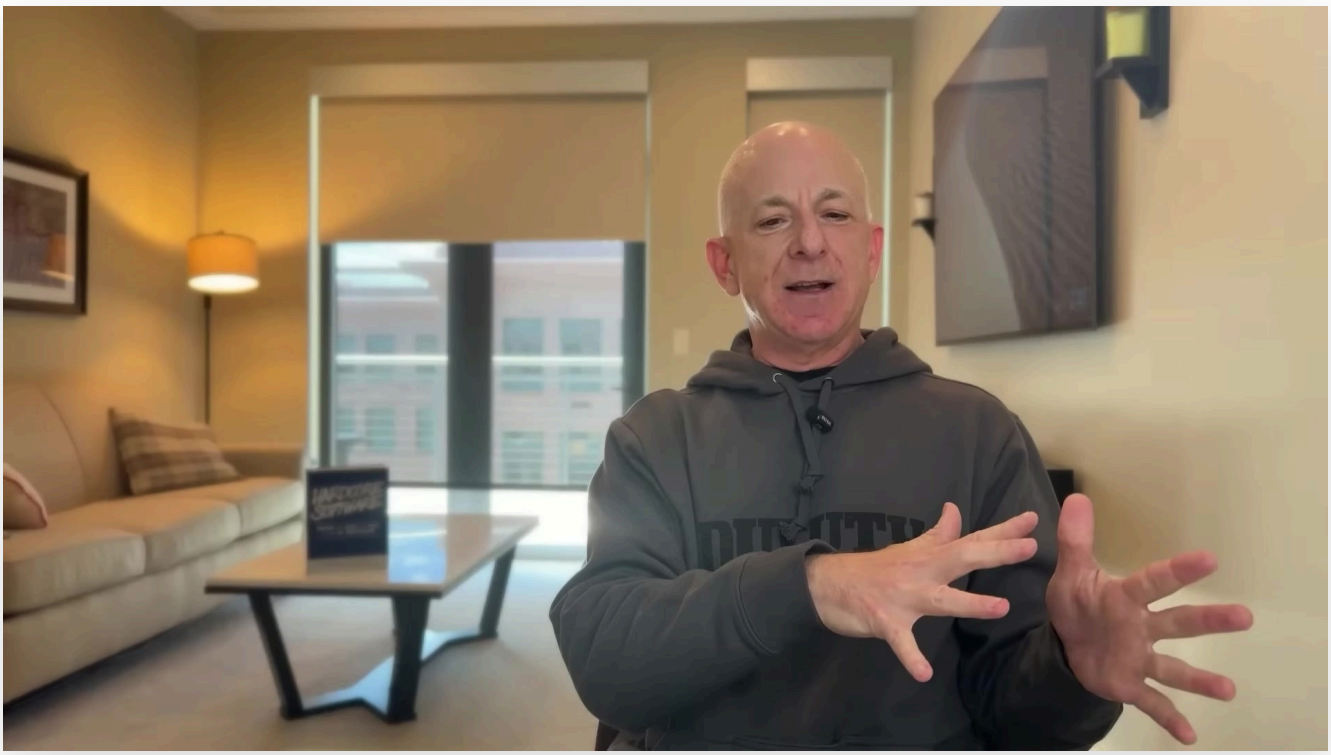
Steven 说，销售人员从来不会认为 CRM 的默认答案适合所有客户。即使语言写对了、付款提醒写对了，具体销售代表仍然会按自己的客户关系处理。**企业里真正有价值的事情几乎都是例外：在 McDonald's 看 15 分钟，就会发现顾客不是按菜单默认选项行动，而是要求混合口味、改变组合。**企业定价也是一样，“每席位多少钱”往往要打电话、谈判和做例外处理。

Seema 补充，voice agent 可以在合规检查、货运等场景中记录过去没有被记录的例外；观察人类如何点击软件、如何回应，也可以收集上下文。**但这不是几天就能完成的知识提取，因为企业销售周期很长，例外出现频率低，还需要积累足够多的交互，让买方相信系统已经理解了规则。**

画面说明

画面主要在 Seema、Steven 和双人分屏之间切换。**图文中的核心信息来自口头概念，不把普通人物画面误写成 context graph 的可视化图表。**

07. 自动化的难点在长尾：权限、验证和异常处理（32:50–36:17）



场景 7

本节主旨 自动化的难点在长尾：权限、验证和异常处理

完整内容

Steven 说，工程师很容易把 headless、API 和 agent API 当成同一件事，以为业务流程都可以直接写成代码。但非工程师甚至很难把自己解决客户问题的过程完整说出来。**Amazon 是一个很好的例子：它不希望客户打电话，因此长期学习如何自动处理问题，并把“倾向于为客户解决”作为核心原则。**客户说收到错误商品，聊天机器人可能直接补发，而不是让人证明每一个细节。

这种异常处理会反过来改变公司的流程。Amazon 收集到退货、错误商品和客服交互的数据后，可以改进物流、仓库、商品描述和评论系统。**AI 让企业对异常的定义和处理方式发生变化，短期内可能因为不再默认偏向客户而让客服体验变差，但长期目标是让公司更有预测性和可重复性。**

Elena 追问，自动化长尾是不是最难的问题。Seema 回答说，权限同样困难：哪些人或 agent 可以抽取哪些数据？什么时候可以读，什么时候可以写？**多个 agent 同时访问一个系统记录时，谁先访问、谁可以修改、谁对结果负责？**这些问题并非不可解决，只是需要时间和大量真实交互。

这段讨论的结论是：让 agent 查到一个字段很容易，让它在一个真实企业里可靠地代表人做决定很难。**企业软件的复杂性并不只在数据表，而在各种例外、权限和责任链里。**

画面说明

本段使用 Steven 的远程机位、Seema 的近景和双人分屏，适合用少量代表性人物图承载完整问答，不按 Amazon 案例的每个句子增加截图。

08. 生产力不会只消灭工作，而会创造新场景（36:18–41:09）

场景 8

本节主旨 生产力不会只消灭工作，而会创造新场景

完整内容

Steven 把话题从异常处理转到生产力。他说，人们通常看着今天已经存在的工作，问怎样把它做得更快，于是担心所有工作都会变成 API，最后只剩 agent。**但生产力提升后，大家会发明新的事情去做，所以“把固定数量的人替换掉，工作就结束了”的线性想象不成立。**

Amazon 的客服是一个例子：自动化让退货更容易，也让公司可以设计出“消耗品错发就不用寄回”的新规则；旧的长尾问题被解决后，后台又会出现新的分析工具和新流程。法律也是这样，合同不会因为 AI 变短，反而可能覆盖更多场景，形成更多条款、诉讼和商业服务。**关于 AI 会不会减少放射科医生的说法，Steven 也提醒这只是相关性而不是因果性，但它至少说明需求和市场不会静止不动。**

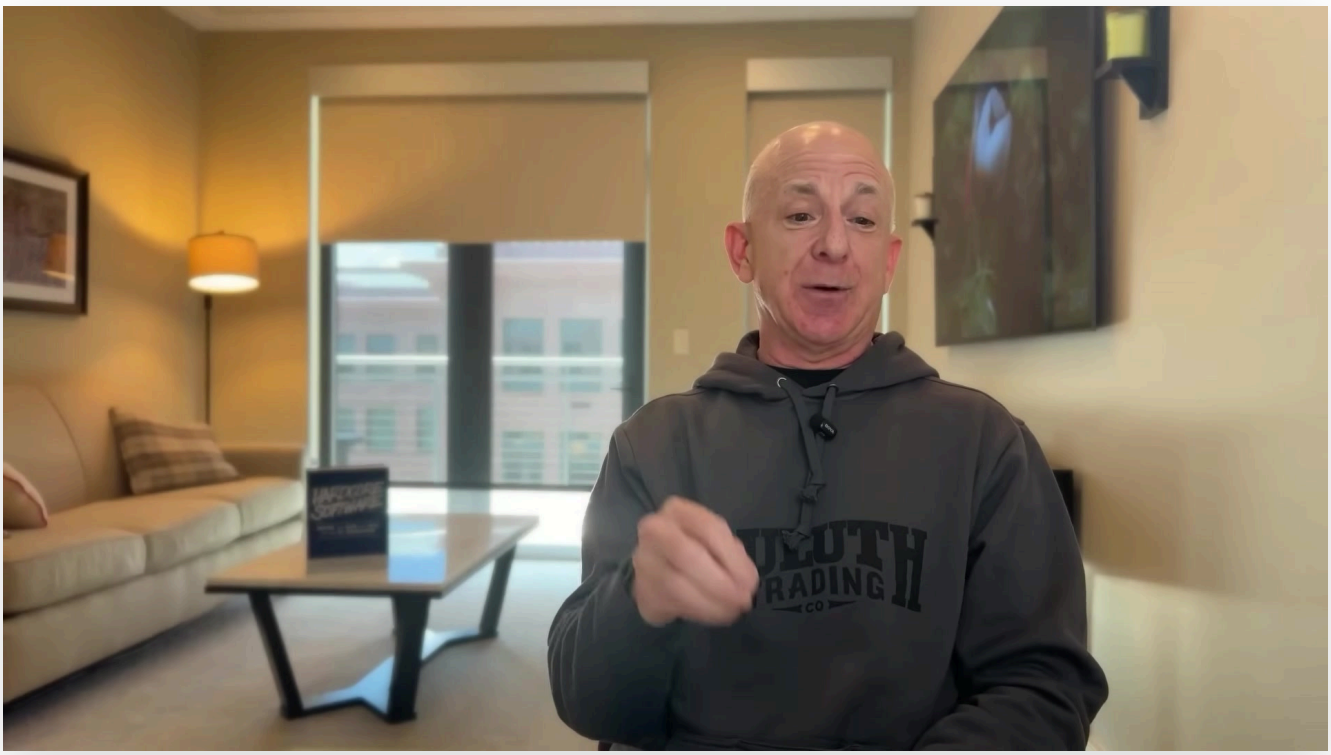
在企业内部，自动化最琐碎的费用报销后，会出现业务旅行分析：公司可以比较不同航班、信用卡和里程方案，把旅行申请路由到更好的价格，还能分析出差对业务表现的影响。**于是原来只是订机票和报销，逐渐变成旅行绩效优化，工作不会消失，而是转到了更高一层的分析和决策。**

Elena 说，数字世界和物理世界也不会完全分开。销售还要和客户见面，人还要乘飞机，只是他们可能不再花时间向 Salesforce 填数据。之后会留下新的数据排放，软件可以用这些数据做优化。**Steven 认为，真正重要的是理解生产力如何不断打开新场景，而不是把经济看成一张固定大小的工作清单。**

画面说明

画面以 Steven 的远程机位为主，中间穿插 Elena 和 Seema 的回应。**没有展示 Amazon、旅行或法律的外部资料画面，因此图注只描述人物与分屏，不把口头案例伪装成屏幕证据。**

09. 企业的本质仍是人做决定，数据来自真实世界（41:10–44:05）



场景 9

本节主旨 企业的本质仍是人做决定，数据来自真实世界

完整内容

Elena 继续说，未来仍然会有人在线下完成销售、谈判和其他工作，问题是这些人不一定要手工把每个步骤输入 Salesforce。现场发生的事情、邮件、通话和结果，会形成新的数据排放，之后再被软件用来做分析和优化。

Steven 用开源软件开发说明“没有 API 的部分”。一个项目必须在某个时间点完成，大家才能把它当成稳定版本使用；但真正困难的是停止修改代码、让一群人同意现在可以发布。**开发者可能会设想用投票、讨论和情绪分析自动完成这件事，但最后仍然需要人对“修还是不修”达成共识。**

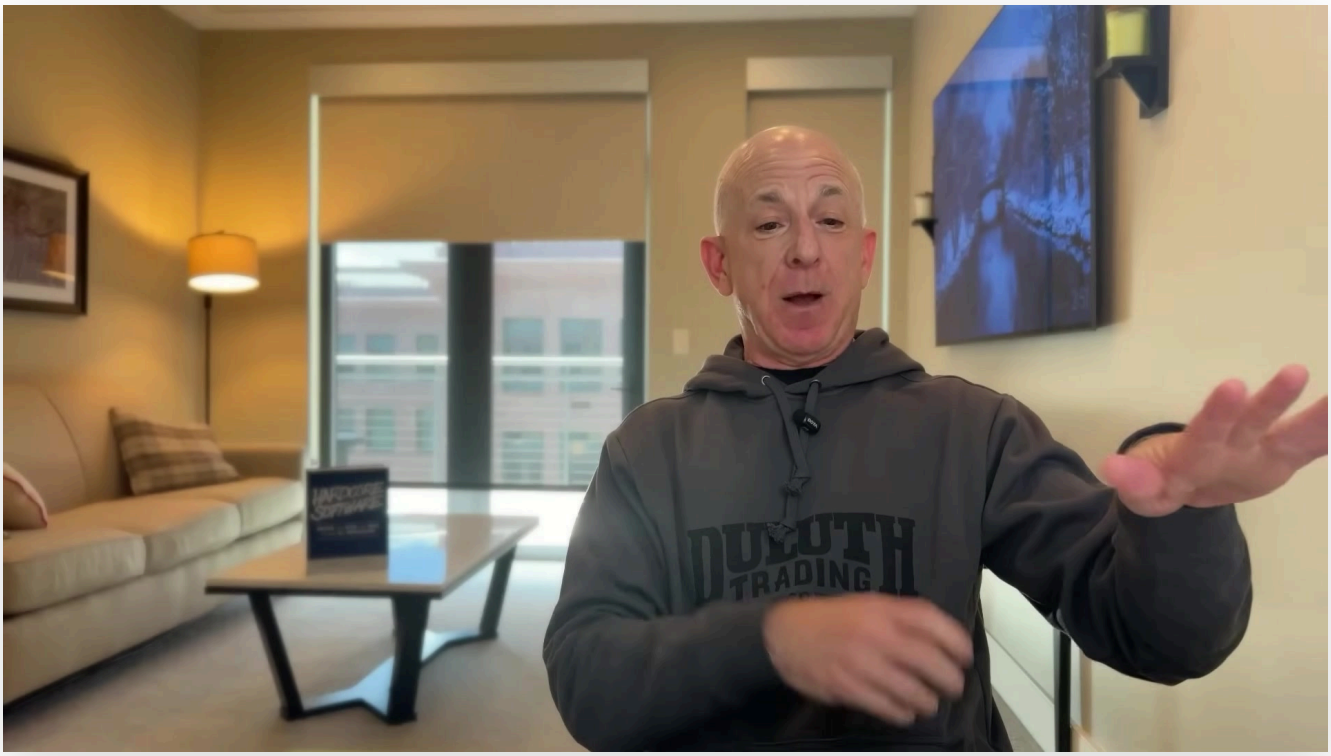
他认为，企业里的很多事情都类似：比如结账、确定哪些销售计入本期、解释发生了什么、留下谁做了什么的审计轨迹。**软件只是把人的决策提升到更高层次、抽象出工具和流程，并没有让决策本身消失。**

Elena 提到，如果能记录人们真实做过的事情，模型就能更好地捕捉线下成交、出差和现场工作。另一位棚内发言者 Elena 也明确补充：这不是在倡导“全景监控”，但录音、邮件、文件和其他书面材料正在被持续摄取，企业会越来越依赖这些真实工作痕迹。

画面说明

此段出现三人分屏、Steven 的远程近景以及棚内两位女性的机位。**画面是对谈记录，不存在需要截取的操作结果；因此保留一张代表性群像即可。**

10. 组织最有价值的知识藏在文档和经验里（44:06–45:12）



场景 10

本节主旨 组织最有价值的知识藏在文档和经验里

完整内容

Steven 接着说，企业里的专业知识像一朵云，散落在不同人的经验、Word 文档、Excel 表格、PowerPoint 演示和过去的工作记录中。这是现代企业尚未充分利用的资源。**困难不只是把文件找出来，而是判断哪些文件重要、哪些版本可信、哪些模型真的被团队依赖。**

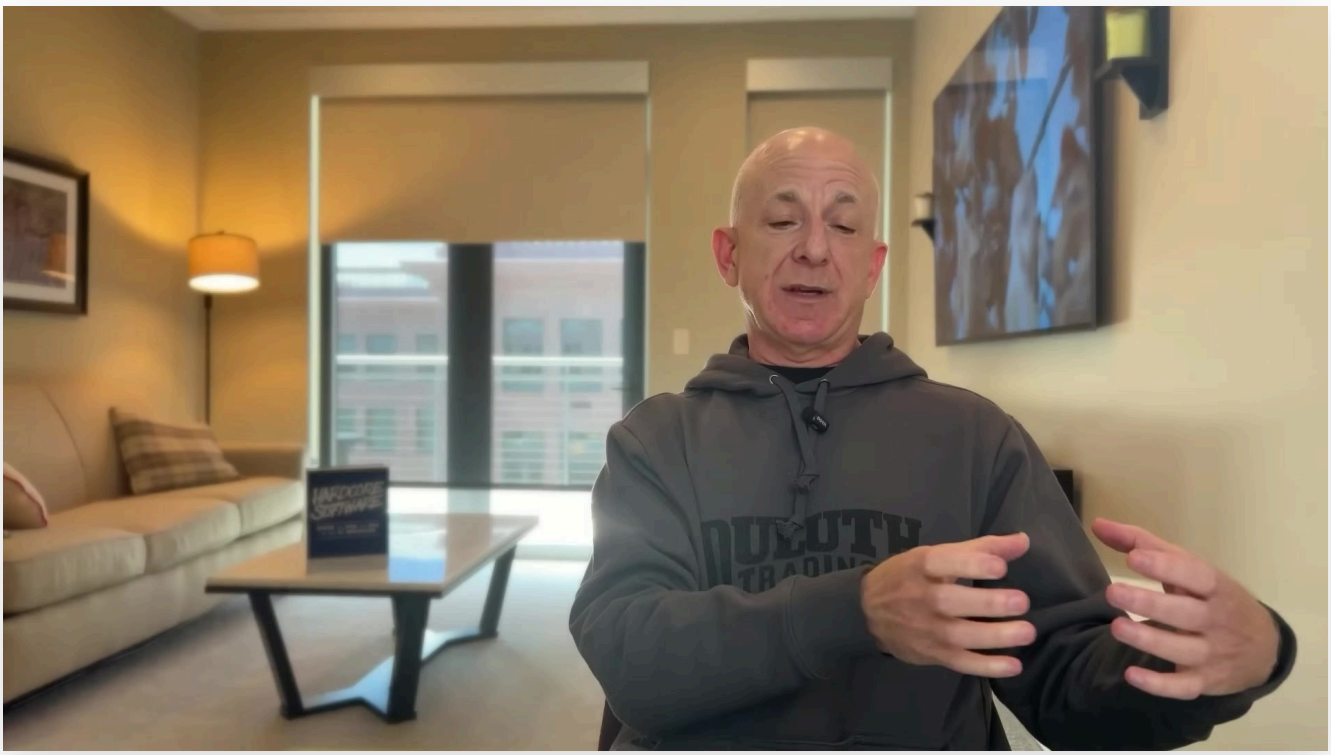
他引用 Box 的观察：客户会用 Box 反过来找出哪些销售演示真正有效、哪些表格和模型被人反复使用。文化和组织经验决定了“应该相信哪一份文件”，这种判断过去很难被软件捕捉。AI 第一次有机会大规模处理这些非结构化信息，并把它们与企业实际行为联系起来。

Elena 在这里暂时收束前面的生产力讨论，准备进入下一段：Headless Software 在更长的技术历史里到底是什么，MCP 和早期 Microsoft middleware 的争论之间是否存在相似之处。

画面说明

场景以 Steven 远程近景为主，配合 Seema/Elena 的棚内回应；没有出现 Box 文件界面或具体文档，因此不新增虚构的文档截图。

11. MCP 与 middleware：工程架构不等于真实世界（45:13–49:15）



场景 11

本节主旨 MCP 与 middleware：工程架构不等于真实世界

完整内容

Elena 引出 Steven 之前写过的文章：MCP 的兴起让人想起微软早期诉讼中关于 middleware 的讨论。不同技术浪潮会重复某些产品层面的模式。**Steven 说，今天关于 MCP 的很多讨论都由工程师驱动：工程师希望每个工具都有干净的 API，最好像命令行一样输入和输出文本。**

但真实世界并不希望一切都被这样抽象。**安全、合规、权限只是部分原因，更根本的原因是：没有一个软件愿意被放到中间层下面，只负责储存一张 SQL 表，然后把所有高价值工作交给别人。**一个费用软件如果只提供费用记录，任由另一个中间层完成分析和业务流程，它就会变成一个不断衰退的业务。

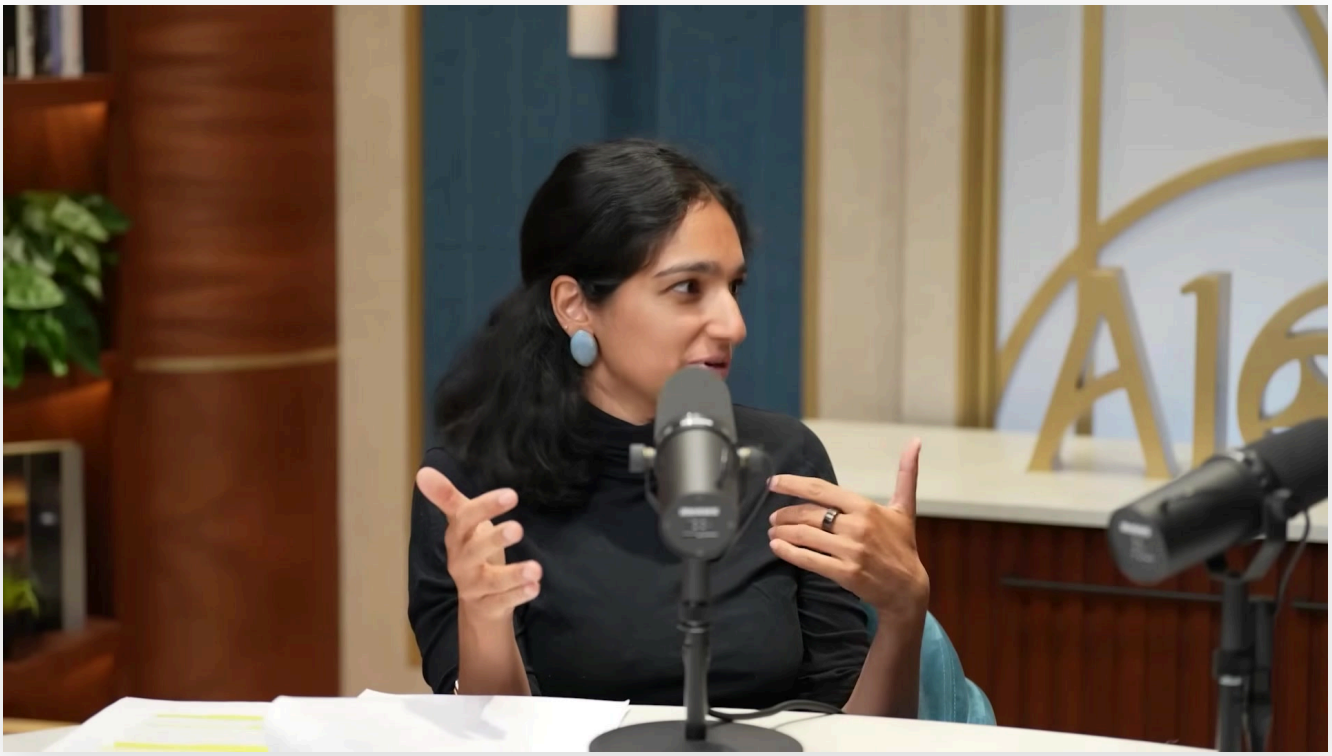
客户也不一定希望自己从多个供应商拼装完整场景。整个系统的稳定性取决于最不稳定的部分；只要其中一个费用服务商倒闭，整个流程就会失效。**所以客户会抱怨原有软件越来越复杂、UI 改版令人烦躁，但又希望这个供应商继续生长、继续承担更多责任。**

Steven 认为，middleware 在网络层级图上看起来很漂亮，但现实中通常不稳定。既有系统会观察用户在做什么，然后自己向左、向右扩展。**SAP 周边生态会继续长出来，但 SAP 也会把其中一些能力吸收进自己的产品，而且不一定做得很好。**大公司即使只和竞争对手打成平手，也可能通过捆绑和免费赠送来守住客户，这会进一步压缩中间层的空间。

画面说明

画面在 Elena 提问、Steven 远程回答和 Seema 的棚内镜头之间切换。**视频没有展示 MCP 架构图，所以说明只保留可见人物和分屏，不把口头类比扩写为不可见的图表。**

12. 企业采用 AI 的三条路径（49:17–52:10）



场景 12

本节主旨 企业采用 AI 的三条路径

完整内容

Seema 回到 Salesforce 和 Workday 的现实：它们即使有 API，也不一定让客户干净地取出全部数据，更不会自然地鼓励别人把自己降级成一个“哑数据库”。**因此企业面对 AI 软件时，大致有三条路。**

第一条是在现有 Salesforce 上打开 **Agent**force，或在 Salesforce 之上构建 agent，把 Salesforce 当成后台系统。这种方式一部分可行，但 Salesforce 不会希望自己只剩后台数据，结果会是混合的。**第二条是完全 DIY，企业拥有最大控制力，但重建真正的企业软件极其困难。**为 Fortune 500 公司重建 CRM，不是把字段和表格重新做一遍，而是要重新捕捉几十年业务逻辑、权限、协作和组织习惯，像在病人还活着时做“开胸手术”。

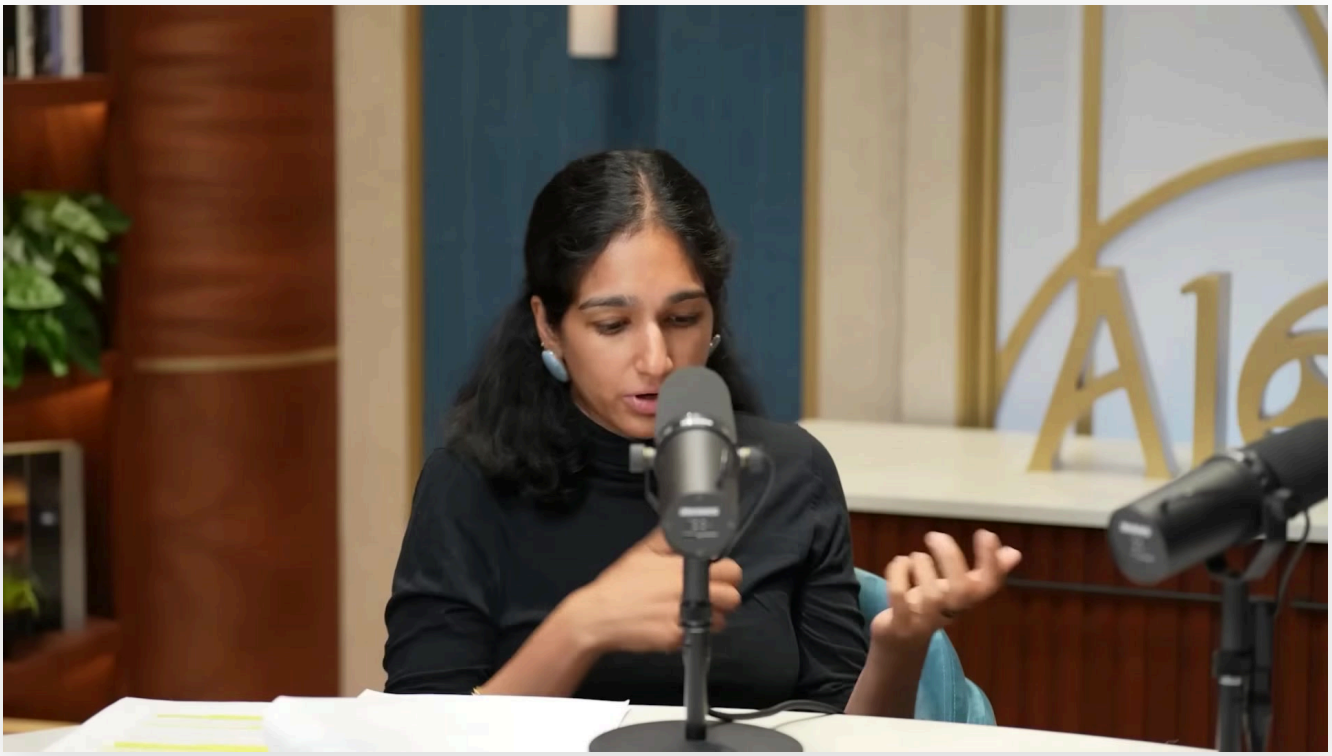
第三条是让 AI 软件在现有系统旁边工作，作为 SAP 等系统的可见性层和体验增强层。数据可以在后台被吸收，业务用户可以在现有逻辑之上使用 agent，又不必把原有系统记录全部扔掉。与此同时，语音 agent 的录音、转写和文档摄取会产生新的数据，未来这些 AI 初创公司也许会替代旧系统记录，但更可能先通过持续观察企业如何运行来逐步完成替代。

Seema 的结论是，企业不会一夜之间抛弃系统记录；更现实的变化，是 AI 先把原有数据变得可用、可行动，再慢慢建立新的系统记录。

画面说明

候选图以 Seema 的棚内近景、Steven 远程机位和双人分屏为主。**这里的“三条路径”是口头结构，不是片中出现的可见列表，所以不额外制作不存在的流程图。**

13. 创业机会：从收集数据走向采取行动（52:10–54:04）



场景 13

本节主旨 创业机会：从收集数据走向采取行动

完整内容

Seema 认为，创业公司最有机会做的是既有厂商还没有做好的部分：不要停在收集数据，而要在数据之上采取行动。**以 CRM 为例，系统不应只记录通话，还可以帮助团队判断哪些线索优先、哪些客户有流失风险，然后直接发送外呼。**

这会形成一个合成循环：agent 发出消息、观察对方回复、判断什么有效、什么无效，再把不同地区、语言和客户类型下的效果积累成基准数据。系统不只是完成一次动作，还会持续知道哪种开场白在亚洲更有效、哪种表达在欧洲更有效。**这样产生的数据排放，会成为新的业务资产。**

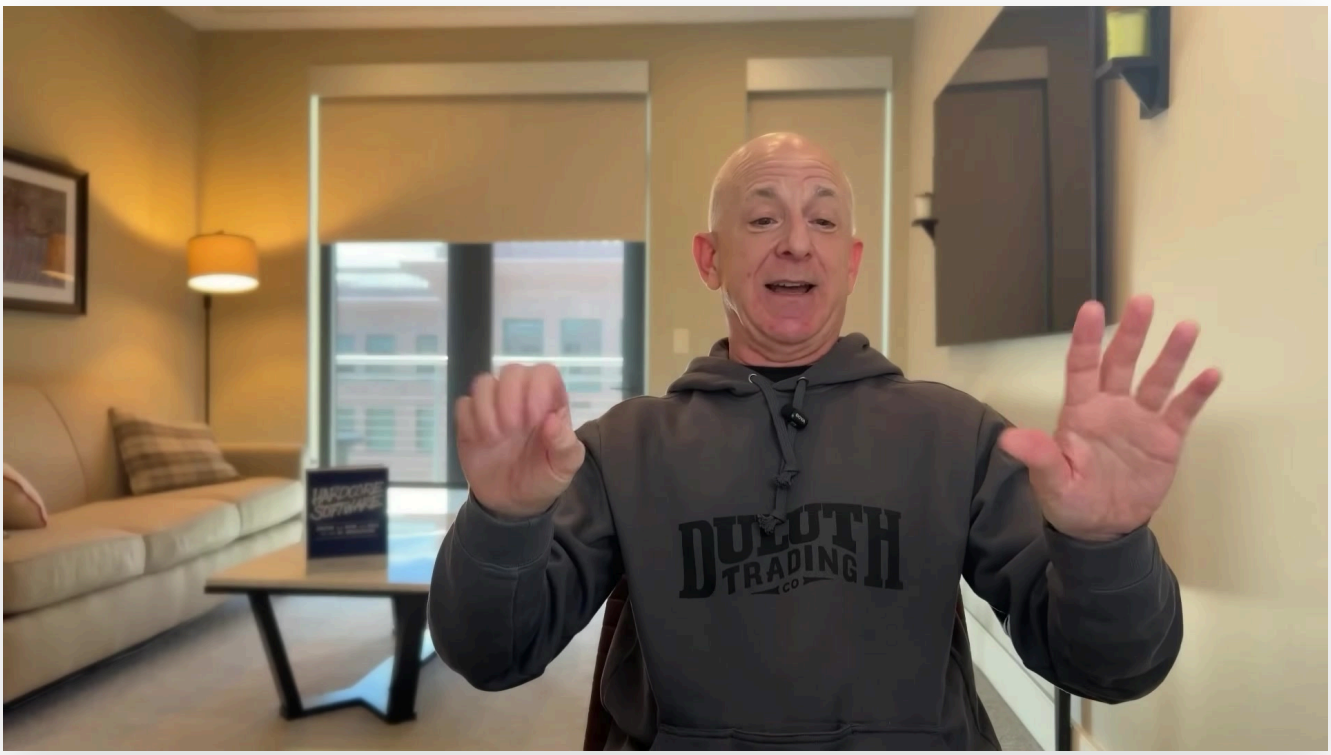
她提出的第二类机会来自物理世界。建筑、制造等垂直软件处理的是过去很难捕捉的数据：人们在现场做了什么，机器发生了什么，实际生产和施工如何推进。**agent 不能只操作软件里的数字，还要把现场的人和机器产生的信息收回来，连接到企业系统中。**

这里的重点不是“把所有旧软件替换掉”，而是让 AI 在业务行动、反馈循环和物理世界之间形成新的闭环。创业公司可以先从一个现有系统不擅长的行动入口进入，再逐步建立自己的数据和规则。

画面说明

画面仍是稳定的棚内/远程对谈，没有出现 CRM 界面或建筑现场。**关键视觉是 Seema 的回答机位和三人分屏，避免把口头案例误写成实际产品演示。**

14. 不要正面复制成熟类别，要在两个系统之间做新东西 (54:05–56:22)



场景 14

本节主旨 不要正面复制成熟类别，要在两个系统之间做新东西

完整内容

Steven 给创业者一条普遍建议：在技术大变革中，最困难、也最不聪明的事情，是正面进入一个已经存在的类别，而且还用同样的方式做。**更好的机会是看企业软件的既有地图，找到两个成熟玩家之间的位置。**

成熟厂商不会轻易破坏自己的产品线和销售渠道。他们会在旧产品旁边加 AI、暴露一个 API、把 agent 能力贴上去，但不会主动停止原有产品，也不会让多年积累的框架失效。创业公司如果直接面对成熟客户，客户会要求它先满足几千条旧要求；而处在中间、用新方式解决新问题的公司，可以把问题改成：“你为什么存在？”**这个问题虽然困难，但创业公司自己能够控制答案。**

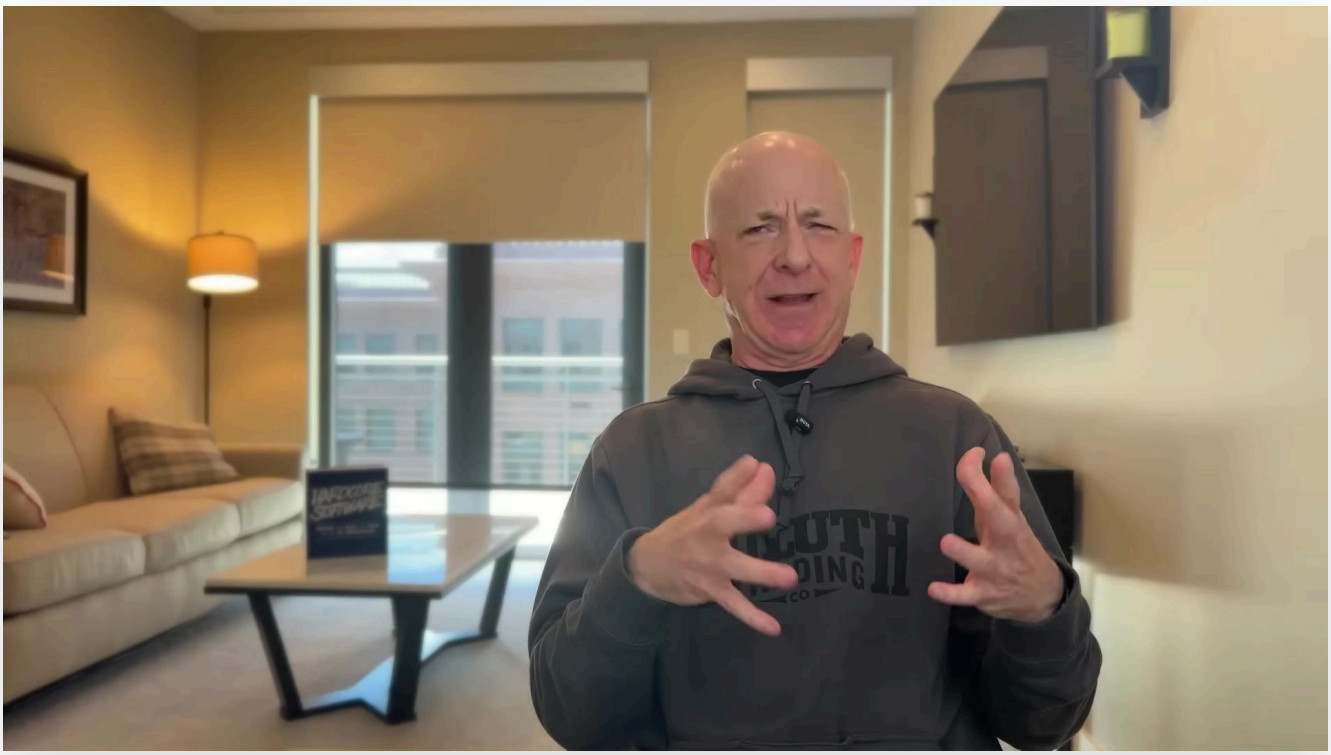
Steven 用 HTTP 和 HTML 取代 client–server 的历史作类比。**Web 并没有把 client–server 时代的所有能力原样实现，甚至很多事情都没有实现；它的优势是用完全不同的方式完成了新的任务。**正因为它不需要先满足旧系统的全部框架，才有机会在既有厂商拥有巨大投资的情况下建立新类别。

Elena 补充，机会不只存在于两个旧厂商之间，也存在于企业内部两个职能之间。**软件过去通常分别卖给销售团队或财务团队，但真正的业务发生在销售、财务、产品等职能的交接处。**AI 可以成为翻译层，把原本互相不沟通的职能连接起来。

画面说明

本段使用 Steven 的远程机位承载核心建议，随后切回 Elena 的提问和双人画面；没有可见的 HTTP、HTML 或企业类别图，因此保持一张代表性画面即可。

15. 企业内部的网络效应：让不同职能真正互相沟通（00:56:22–01:00:46）



场景 15

本节主旨 企业内部的网络效应：让不同职能真正互相沟通

完整内容

Elena 最后追问，企业软件过去很少形成消费互联网式的网络效应，Salesforce 是否会让买方和卖方都进入 CRM，从而建立新的网络效应？**Steven 认为，企业外部的网络效应受合规和安全限制，但企业内部正在出现更重要的网络效应。**

大多数员工并不会每天思考怎样把工作做得更好，他们只是希望完成任务、不要出错。但总有一小群人会主动寻找更快、更好的方法。**Steven 回忆，早期 Goldman Sachs 的银行家用 Excel 做更复杂的模型，而其他人还在使用 Lotus 1–2–3。**当别人看到同事在做什么、发现这些工具确实能让工作更好时，使用就会在组织内部扩散。

他把今天的 Chat 与当年的 Excel 类比。Steven 曾经问 SAP 的朋友正在写什么问题，然后用模型帮她生成了一份白皮书；这让团队里的其他人看见一种新的工作方式，形成了类似内部病毒循环的扩散。它不一定是技术意义上的 viral loop，但确实是“看到别人如何让工作变好，于是自己也开始用”的组织网络效应。

更大的机会，是做连接器：让两个原本无法对话的职能互相理解。企业软件整合过去依靠手工和高层流程完成；如果软件能用 AI 连接销售与财务、设计与产品开发、IT 与预算团队，就可能形成新的类别。**Steven 认为，能把组织中不常沟通的部分连接起来，是企业软件里非常有价值的方向。**

Elena 以这句话结束节目，感谢 Steven 加入，也感谢 Seema 参与讨论。**全文的最终落点是：AI 不只是把旧软件变成 API，更可能把企业中原本分开的数据、经验和职能连接起来。**

画面说明

末段使用 Steven 的远程机位、棚内双人机位和最后的分屏，展示三位嘉宾完成收束；画面没有实际的 Excel 广告或产品图，不额外添加不可见证据。