

第 1 章



原型设计的价值

- 原型是生成的
- 原型设计——展示、讲述和体验之源
- 原型可减少误解
- 原型可节省时间、精力和费用
- 原型可减少浪费
- 原型可提供真实价值
- 小结

每

年，都有几百万人通过概念车来窥见未来。汽车厂商在这些独一无二的概念车上投入数年时间和数百万美元。但大多数概念车都没有被量产。量产的往往也只是原始想象中的一小部分。

在汽车行业，竞争相当激烈。创新不只是保持领先的手段，而往往是一种生存手段。每辆概念车都是一次设计练习，都是对可能性、可行性和市场的探索——这就是原型。

数十年以来，这种原型方法一直是汽车行业的核心。概念车虽然昂贵，但重新配备所有必要机器和上线失败的代价更高，风险实在太太。

原型是整体设计过程的必要组成部分，比如汽车或导弹导航系统设计，这无可非议。但对于软件开发而言，为创建原型而争取支持有些难。事实上，这是我们面临的最典型的挑战之一。

第 1 章重点说明把原型融入现有设计/开发流程时所面临的一些挑战。本章还要给出一些要点，以说明原型在提早识别问题、减少风险进而最终节省时间、精力和费用等方面所发挥的价值。

如果客户或者管理层不熟悉原型，往往会把原型视为“烧钱”的行为，即使有好处，也微不足道。我收到过一个最常见的问题：“怎样才能使老板或者客户认同原型设计？他们都说没有这方面的时间和预算。”

如果你的业务涉及到用软件和硬件构建网站、软件应用程序或者系统，就能做原型。随着系统复杂程度的提高，原型设

计的成本/效益比也会显著提高。

原型设计的确是有成本的。它不是免费的。但如果你没有做过原型，就已经失去了很多创新、显著降低成本的机会。原型的收益远远超过它的初始成本。

原型是生成的

原型的根本价值之一是它是生成的（也称创成的），也就是说，在原型设计过程中，你会得到几百个（甚至上千个）想法。有些才华横溢，有些一般。我发现，即使那些一般的想法也可以催生优秀的解决方案。

作为生产过程，原型设计流程中往往也会引发创新，显著节约时间、精力和成本。原型过程有助于激发灵感，能帮你得到更具体的东西——可以感觉、体验、参与、娱乐和测试的东西。

原型设计——展示、讲述和体验之源

如果说一图胜千言，那么我要说原型抵万言。原型的威力远远超过展示和讲述——原型能让你体验自己的设计。

“讨论、建立故事板是一回事，看到真实的东西则是另外一回事。”

—— Robert Hoekman, Jr.

交流设计构想或者编写设计文档有很多方法，包括需求文

档、线框图、视觉组件和原型。

设计文档的三大常见模型

需求文档。需求文档通常是描述系统技术或者功能需求的书面文档。需求文档更关注书面说明，较少关注视觉呈现——更侧重于讲述，而不是展示。缺少视觉模拟常常会导致对需求的误解。页面截屏可以减少误解，但静态截屏的能力有限。

线框图。有没有见过房屋建筑蓝图？它有点像软件线框图。线框图是功能页面结构的视觉呈现。线框图从视觉上传达页面上展现哪些功能以及功能之间的关系。线框图一般以黑白或者灰度形式呈现。

线框图结合详细的交互行为注释，比需求文档更有利于展示和描述。但线框图常常会在设计中留下空白。这些空白会导致细节遗漏或者误解，显然并不是好事。

原型。原型是最终系统的代表模型或者模拟。不同于需求文档和线框图，原型比演示和讲述走得更远，原型能让你真正地体验设计。

有些技术需求，例如页面不超过 100KB，也不能在原型上明显地体现出来。一份不到 60~200 页的补充文档足以轻松说明这类需求。

需求文档或者线框图本身都不足以展示和讲述复杂系统。对于简单系统，采用需求文档或者线框图也许就够了，但对于复杂系统，采用需求文档或者线框图最终会使你陷入麻烦。

需求文档或者线框图通常结合使用以创建一个“完整画面”。但对于实际体验设计，需求文档或者线框图仍然无法达成目标。

带批注的线框图，再加上需求文档，能达到原始想法 70% ~ 80% 的准确度。但对于本书而言，出错空间仍然太大了。

AJAX 和 RIA 电动扳手

如果把 AJAX 或者其他富互联网应用 (RIA) 加入设计文档中，会发生什么？势必崩溃——而且很快崩溃。无论是需求文档，还是带注释的线框图，都不擅长于讲述富交互和动态变化的故事。

不同于基于页面的传统交互，AJAX 和 RIA 常常使用基于状态的交互。页面或者界面可以有多个块或多个小部件，彼此独立操作而又内部关联。更新页面的 RSS 源时，不再需要刷新整个页面。只需要更新 RSS 源小部件，页面其他部分则保持不变。

这促使设计社区中很多人宣称旧的页面范式已死——新的范式是界面或者状态。

变化和动画是另一项挑战。你是否曾经想过要描述一个自我修复的 AJAX 变化？我用过最好的描述，配合策略性手势和魔杖仿真，但仍然有很多人皱起眉头、满脸狐疑。

随着 AJAX 和其他 RIA 技术的持续发展，原型作为设计传达工具的需求和价值在显著增长。其实，我认为原型已经成

为 决 定 成 败 的 关 键。

原型可减少误解

拿一份 60 页的需求文档。召集 15 个人到一个房间。把文档发给每个人。请每个人都阅读文档。然后问他们你想做什么。你会得到 15 个不同的答案。用一份 200 页的需求文档做相同的试验，想象一下——情况会更糟。

对于打算建立的系统，原型是更形象、更容易感知的表现形式。原型能提供真实的体验。

我们公司经历了从需求依赖型流程到原型依赖型流程的转变，转变后，我们发现解释和返工量马上减少了。解释一致性从 60% ~ 80% 上升到 90%，甚至更高。

我们还发现，制作原型所需的全部时间和精力少于创建一份详细的规格文档和带注释的线框图。

我发现下面这些原因更容易导致书面文档被误解。

没人想读一份长达 60~200 页的书面规格说明。实在没有丝毫乐趣。

如果你无法让人们阅读文档，更无法让他们完全理解。

书面文档无法让人看到“全景”，而且一次只能看一行。

文字描述会造成很多种理解。

从另一个角度来说，原型有很多优点，有助于减少误解。

可以实际体验系统是如何工作的，而不是只读相关的

文档。

原型鼓励人们积极使用。如果能让其他人使用原型，
他们可能更能理解原型。

原型可节省时间、精力和费用

下面这样的话，你曾听客户、老板甚至是团队设计师和开发人员说过多少次？

“我们没有时间做原型。”

“我们没有钱做原型。没有这方面的预算。”

这些话，我听过几十次了。老实说，他们这么说也并不是没有依据。前面说过，原型设计并不是说不花钱，但原型的好处远远超过其成本，或者说，最重要的是好处远远超过原型本身。

和那些从无原型开发流程转为含原型开发流程的人谈一谈，他们会告诉你原型帮他们节约了大量时间，省去了大量麻烦。原型不仅能让你实现并体会到更快的设计，而且最终会减少其他设计及开发流程所造成的浪费。

原型可减少浪费

在典型的设计和开发流程中，我们的做法是先写好需求，然后把它交给设计师或者开发人员。随后，设计师或者开发人员解读需求，并根据他们自己的理解来构建东西。

理论而言，需求驱动的设计流程应该可以减少浪费。其整体目标就是所有人达成共识。如果大家的理解是一致的，最终浪费就少。听起来很不错。

这种流程在理论上非常好。但经验表明，理论和现实是两码事。需求驱动的传统设计和开发流程有很多缺点，产生了很多浪费。

1. 需求由错误的人编写。设计师和开发人员很少参与需求编写流程。相反，需求通常由业务分析师或者类似人员来编写。这些人缺少设计师和开发人员所具备的技术和设计常识，经常导致很多需求屡屡被重写。
2. 耗费大量时间和精力。编写、审阅和修订详细需求说明所花的时间相当多。对于复杂系统，我见过花 3 到 9 个月甚至更长时间才完成的情况。如此漫长的时间里，很多事情早就物非人非。
3. 没有最终的最终文档。在理论上，需求是最终文档。然而在事实上，需求一直在变，甚至在“定稿”后还在变。
4. 曲解文档。对于一份 60~200 页的需求文档，往往会产生很多误解。误解会导致数周、数月的返工，同时延误产品发布。
5. 非根本特性。需求文档经常充斥着没有太大价值的特性。人们不得不耗费时间和精力建立并测试这些特性。其后果是，浪费时间去为这些没有多大价值、往往不会有人使用的特性写需求、构建和测试特性。
6. 发现错误太晚。需求驱动的流程一般在进入生产阶段之后才会出问题。开发流程中发现问题的时间越晚，修复

成本越高。

以上任何问题都会造成时间和精力浪费。在很多情况下，需求驱动的流程经常受到以上问题困扰，导致效率低下，浪费严重。从另一个方面讲，流程中包含原型有助于减少浪费，并带来以下好处。

1. 由正确的人来做决策。设计师和开发人员能充分利用经验和知识为流程做出贡献，并最终确保正确的人做出正确的设计决策。
2. 适者生存。产生并测试多个想法，以确保留下最强大的解决方案。
3. 适应。可以快速更新原型，以弥补软件开发持续变化的特性。
4. 减少误解。原型是系统的视觉展现，有时是有形展现。视觉展现和有形展现比 60~200 页的书面文档产生的误解要少。减少了误解，就可以减少返工数量。返工数量减少意味着成本更低，上市时间更快。
5. 专注。原型能产出更专注的产品。产品更专注，就可以减少设计、开发和返工所造成的浪费。
6. 提前发现错误。原型有助于在设计和开发流程早期发现问题。发现问题越早，解决问题的成本越低。
7. 减少风险。误解减少，在设计和开发周期中提前发现问题，如此一来，原型便减少了风险。

原型虽然不能解决困扰着需求驱动型流程的所有问题，但肯定有助于减少普遍存在的很多低效和浪费的问题。

原型可提供真实价值

Jonathan Baker-Bates 领略过通过原型所得到的可度量的好处。Jonathan 就职于一家英国咨询公司，这家公司的设计和开发流程非常典型。他的开发团队会定期收到长达 200 页的规格文档，并根据文档进行报价和开发。这是他们过去的做法。

他们公司最近转型为由原型驱动的流程。开发团队现在拿到的不是 200 页文档，而是一个高保真原型，附带一份 16 页的支持文档。

自从流程改变后，他们公司看到了很多显著的改进。

相较于编写 200 页文档，制作原型和编写 16 页补充文档所花的时间和精力不到一半。

估算开发时间和成本的准确度提高了 50%。

开发团队的“澄清”要求减少了 80%。

发布后的返工和缺陷修复数量减少到以前类似项目的 25%。

所有团队成员都认同用原型流程来做设计比老流程简单。

案例研究：在预算紧张和期限紧张时采用原型方法

作者：Jonathan Baker-Bates

我们要用不到四个月的时间为一家大型电脑游戏开发商设计开发一个社交网站，我们拿到了强大的视觉设计概念、很多广泛的内容和功能需求，还有一个内部开发团队准备开发一套内容管理系统。但是预算紧张，而且项目范围还有很多不确定的地方。这也是我们第一次与这位特殊客户合作，他们不需要我们提供大型文档来进行研究（类似项目的文档常常多达 200 页）。我们需要所有人都参与，并从一开始就清楚自己要做什么。

我们决定，从设计阶段的第一天起，所有人都以功能性 HTML 原型为中心，这个原型用 Axure 创建和维护，能演示整个网站。记住了这点，我们希望能非常详细地向所有人展示所有必要需求，并且每个人都要领会，不管是 CEO 还是内容管理系统集成人员。

当然，这个原型也不可能展示所有东西。非功能性元素、一些条件性界面、异常流程，还有实施过程的各种注释，都需要单独记录。这些内容可以写成大约 20 页的文档。有条原则适用该文档，即文档应当只包含原型中可能会正常无法理解的内容。这对于减少费用和避免文档失控非常重要。

我们首先发现原型消除了冗长的“介绍”交流和讨论。看过原型的人很快在短短几分钟内熟悉了我们要做的东西，

案例研究：在预算紧张和期限紧张时采用原型方法
(续)

根本不用花几个小时甚至几天时间。这样一来，我们可以转而讨论细节。我们能在多个方面受益。首先，估算开发的准确度非常高。其次，整体设计沟通(或者问题)大大减少——可能比通常情况减少了80%。最后，在整合和测试阶段，设计也进行得相对顺利，根据规格文档报告的缺陷数量减少了25%。预算和范围限制仍然意味着困难还是有，但有原型意味着能够快速说明我们打算如何缩减部分需求并最终让客户签收。

对我们而言，这种方法成功了。但我想提醒读者：所有项目都不同，在其他环境下它也许并不是最理想的选择。同时值得指出的是，主站上线后，我们又回到了基于文档的传统流程，以应对不断增长的变化。但如果没有原型，我们可能无法交付。

Jonathan 的故事并不鲜见。我为本书写作采访过很多人，他们也有同样的故事。参加过我的原型工作坊或者讲座的人也给我讲过类似的故事。本书写作过程中，我进行过多次调查，有好几百人在回复中也讲述了同样的故事。

小结

原型真的能够减少几个月甚至几年的开发时间。你还在等什么？

知道原型的价值后，你会得到客户或者管理层的支持。只需记住以下要点。

原型是生成式的。

原型可通过展示和讲述进行传达。

原型可减少误解。

原型可节约时间、精力和费用。

原型可创建快速反馈循环，最终能减少风险。

下一章将介绍我所使用的快速、迭代的原型设计流程。