《设计原本》

图书基本信息

书名：《设计原本》

13位ISBN编号：9787111325574

10位ISBN编号：7111325575

出版时间：2011-1-1

出版社：机械工业出版社

作者：Frederick P. Brooks, Jr.

页数：268

译者：InfoQ中文站,王海鹏,高博

《设计原本》

内容概要

无论是软件开发、工程还是建筑，有效的设计都是工作的核心。《设计原本:计算机科学巨匠Frederick P. Brooks的思考》将对设计过程进行深入分析，揭示进行有效和优雅设计的方法。
本书包含了多个行业设计者的特别领悟。Frederick P. Brooks, Jr.精确发现了所有设计项目中内在的不变因素，揭示 了进行优秀设计的过程和模式。通过与几十位优秀设计者的对话，以及他自己在几个设计领域的经验，作者指出，大胆的设计决定会产生更好的结果。
作者追踪了设计过程的演进，探讨了协作和分布式设计，阐明了哪些条件造就了真正卓越的设计者。他探讨了设计过程的具体细节，包括多种预算约束条件、美学考虑、设计经验主义及工具。同时，他将这些讨论与现实中的案例结合起来，这些案例从房屋建造到IBM的Operating System/360。成功的关键因素贯穿全书，每个设计者、设计项目经理和设计研究者都应该知道。

《设计原本》

作者简介

Frederick P. Brooks 北卡罗来纳大学计算机科学系的Kenan教授。他因领导开发IBM System/360计算机家族以及Operating System/360而荣获美国国家技术奖，并因对计算机体系结构、操作系统和软件工程作出了里程碑式的贡献而获得A. M.图灵奖。他是畅销书《人月神话》的作者。

《设计原本》

书籍目录

Frederick P. Brooks Jr. 论设计的本质
译者序
前言
作者简介
第一部分 设计之模型
第1章 设计之命题
1.1 培根所言是否正确
1.2 什么是设计
1.3 何为真实？设计的概念
1.3.1 价值何在
1.4 对于设计过程的思考
1.5 设计类别
1.5.1 系统设计与艺术设计
1.5.2 常规，适应性，原创设计
第2章 工程师怎样进行设计思维——理性模型
2.1 模型概览
2.2 该模型的构思从何而来
2.3 理性模型有哪些长处
第3章 理性模型有哪些缺陷
3.1 我们在初始阶段并不真正地知道目标是什么
3.2 我们通常不知晓设计树的样子——一边设计一边探索
3.3 （设计树上的）节点实际上不是设计决策，而是设计暂定方案
3.4 有用性函数无法以增量方式求值
3.5 必要条件及其权重在持续变化
3.6 约束在持续变化
3.7 对于理性模型的其他批评
3.8 但是，尽管有这些缺陷和批评，理性模型仍然不屈不挠地存在
3.9 那又如何？我们的设计过程模型真的那么事关紧要吗
第4章 需求、罪念以及合同
4.1 一段恐怖往事
4.2 殊为不幸，无独有偶
4.3 抵制需求膨胀和蠕变
4.4 罪念
4.5 合同
4.6 一种合同模型
第5章 有哪些更好的设计过程模型
5.1 为什么要有一个占主导地位的模型
5.2 共同演化模型
5.3 Raymond的集市模型
5.3.1 运作机理
5.3.2 模型优势
5.3.3 什么时候可以采用集市模型
5.4 Boehm的螺旋模型
5.5 设计过程模型：第2～5章的讨论小结
第二部分 协作与电信协作
第6章 协作设计
6.1 协作是在本质上是好的吗

6.2 团队设计是现代标准
6.2.1 为什么工程设计从个人转向团队
6.3 协作的成本
6.4 挑战是概念完整性！
6.4.1 异议
6.5 如何在团队设计中获得概念完整性
6.5.1 现代设计是各学科间的协商吗
6.5.2 系统架构
6.5.3 一名用户界面设计师
6.6 协作何时有帮助
6.6.1 确定利益相关人的需求和愿望
6.6.2 概念探索—激进的可选方案
6.6.3 设计复查
6.7 协作何时无用—对设计本身
6.7.1 概念设计尤其不应该协作
6.8 两人团队很神奇
6.9 对于计算机科学家意味着什么
第7章 电信协作
7.1 为什么要电信协作
7.1.1专业化
7.1.2 家
7.1.3整天工作不停
7.1.4成本
7.1.5政策
7.2 到那里，做那事—IBM System/360计算机系列的分布式开发，1961 – 1965
7.3 让电信协作有效
7.3.1 面对面的时间很重要
7.3.2 干净的接口
7.4 电信协作的技术
7.4.1 低科技常常足够
7.4.2视频会议
第三部分 设计观点
第8章 设计中的理性主义与经验主义
8.1 理性主义与经验主义
8.2 软件设计
8.3 我是个铁杆的经验主义者
8.4 其他设计领域中的理性主义、经验主义与正确性
第9章 用户模型——错误胜过含糊不清
9.1 明确的用户与用例模型
9.2 果真如此吗
9.3 团队设计
9.4 假如事实不可用该如何是好
9.4.1 猜测
9.4.2 错误胜过含糊不清
第10章 英寸、盎司、位与美元——预算资源
10.1 何谓预算资源
10.2 美元并非万灵丹
10.3 即便美元也有不同，替代品剖析
10.4 预算资源是可变的

10.5 那又如何
10.5.1 明确确认
10.5.2 公开跟踪
10.5.3 严格控制
第11章 约束是我们的朋友
11.1 约束
11.2 不完全如此
11.3 设计悖论：通用的产品要比特定用途的更难以设计
11.4 小结
第12章 技术设计中的美学与风格
12.1 技术设计中的美学
12.2 何谓逻辑美
12.2.1 简约
12.2.2 结构清晰
12.2.3 一致性
12.2.4 什么才是好的计算机架构
12.4.5 一致性的更多优点
12.6 技术设计中的风格
12.7 何谓风格
12.8 风格的属性
12.9 要想获得一致的风格——记录下来
12.10 如何获得良好的风格
第13章 设计中的范本
13.1 很少会有全新的设计
13.2 范例的角色
13.3 计算机与软件设计呢
13.3.1 你使用何种范本
13.4 学习范本的设计原理
13.4.1 第一代计算机
13.4.2 第三代计算机
13.4.3 虚拟内存
13.4.4 小型计算机的变革
13.4.5 微型计算机与RISC的变革
13.5 如何训练才能改进基于范本的设计
13.5.1 范例集合
13.5.2 超越集合
13.5.3 软件设计怎样呢
13.6 范本——懒惰、创意与自满
13.6.1 一些观点
13.6.2 懒惰
13.6.3 创意与自满
第14章 专业设计者缘何犯错
14.1 错误
14.2 曾经最糟糕的计算机语言
14.2.1 何谓JCL
14.2.2 JCL到底怎么了
14.2.3 JCL缘何是这个样子的
14.3小结
第15章 设计的分离

15.1 设计与使用和实现的分离
15.2 为什么分离
15.3 分离的结果
15.4 补救措施
15.4.1 补救措施1：用户场景体验
15.4.2 补救措施2：通过增量式设计和增量式交付与用户密切交互
15.4.3 补救措施3：并发工程
15.4.4 补救措施4：设计者的教育
第16章 展现设计的演变途径和理由
16.1 简介
16.2 知识网线性化
16.3 我们的设计演变途径记录
16.4 我们研究房屋设计过程的过程
16.4.1 什么是设计树
16.5 深入设计过程
16.5.1 设计不只是满足需求，也是发现需求
16.5.2 设计不是简单地选择可选方案，也是意识到它们的存在
16.5.3 设计变化时树也变化—如何展现演进过程
16.6 决策树与设计树
16.7 模块化与紧密集成的设计
16.8 Compendium和可选工具
16.8.1 Task Architect
16.8.2 项目管理工具
16.8.3 IBIS和它的衍生品
16.8.4 Compendium
16.9 DRed：一个诱人的工具
第四部分 计算机科学家设计房屋的梦想系统
第17章 计算机科学家的建筑设计理想系统——从思维到机器
17.1 挑战
17.2 一个设想
17.2.1 渐进完善
17.2.2 模型库
17.2.3 渐进完善模式的不足
17.3 从思维到机器输入的设想
17.3.1 名词-动词组合
17.4 说明动词
17.5 说明名词
17.6 说明文字
17.7 说明助词
17.8 说明视点和视图
17.8.1 内部视图
17.8.2 外部视图
第18章 计算机科学家的建筑设计理想系统——从机器到思维
18.1 双向通道
18.2 视觉显示——多并发窗口
18.2.1 制图桌和绘图视图
18.2.2 2D内容视图
18.2.3 3D视图
18.2.4 外部视图

18.2.5 工作手册视图
18.2.6 规格视图
18.3 声音展示
18.4 触觉展示
18.5 泛化
18.6可行性
第五部分 卓越的设计师
第19章 卓越的设计来自卓越的设计师
19.1 卓越的设计和产品过程
19.2 产品过程：优点和不足
19.2.1 产品过程抑制了卓越的设计吗
19.2.2 为什么要有产品过程
19.3 观点碰撞：过程抑制，过程不可避免；怎么做
19.3.1 卓越的设计来自卓越的设计师，去找到他们
19.3.2 卓越的设计需要大胆的领导者，他们要求创新
19.3.3 如何设计一个鼓励卓越设计的过程
19.3.4寻求概念完整性：信任一名主设计师来完成设计
第20章 卓越的设计师从哪里来
20.1 我们必须教他们设计
20.2 我们必须为卓越设计而招募人才
20.3 我们必须深思熟虑地培养他们
20.3.1 让两架梯子真实而体面
20.3.2 规划正式的教育经历
20.3.3 规划不同的工作经历
20.3.4 规划离开组织机构去休假
20.4 管理他们时必须发挥想象力
20.5 必须严密地保护他们
20.5.1 防止他们分心
20.5.2 保护他们不受管理者干扰
20.5.3 防止他们去做管理
20.6 把自己培养成一名设计师
20.7 不断画设计草稿
20.8 寻求有知识的人对您的设计的批评
20.9 研究教学示例和先例
20.10 一个自我教育项目：1000平方英尺房屋的建筑平面图
第六部分 设计空间之旅：案例研究
第21章 案例研究：海滨小屋“ View/360 ”
21.1 亮点和特性
21.2 背景介绍
21.3 目标
21.4 机会
21.5 约束条件
21.7 设计决定
21.8 考虑正面
21.9 小屋的尺寸
21.10 设想的开始
21.11 在设计之后，构建之前的设计改动
21.12 在框架和外墙完成和初次入住之后的设计改动
21.13 评估（在37年后）

21.13.1 乐事
21.13.2 实用性
21.13.3 牢固性
21.13.4 如果我“ 废弃一个计划 ”呢
21.14 学到的一般经验
第22章 案例研究：增加厢房
22.1 亮点和特性
22.2 背景介绍
22.2.1 背景
22.3 目标
22.3.1 最初目标
22.3.2 后来发现的目标
22.4 约束条件
22.5 非约束条件
22.6 事件
22.7 设计决定和迭代
22.7.1 考察
22.7.2 分割设计问题
22.7.3 东部
22.7.4 西部一半的功能安排
22.7.5 方式变化：忘掉预算是设计约束条件
22.7.6 新发现的需求：
22.7.7 功能安排的会合
22.7.8 构建期间的改动
22.8 评估——成功与未解决的缺点
22.8.1 新功能
22.9 学到的一般经验
第23章 案例研究：厨房重新建模
23.1 亮点和特性
23.2 背景介绍
23.2.1 背景
23.3 目标
23.4 机会
23.5 约束条件
23.6 关键宽度预算的推理
23.6.1 从北到南需要的宽度
23.6.2 试验性的设计
23.6.3 另一些宽度解决方案
23.6.4 最终的宽度设计
23.7 长度预算的推理
23.8 其他设计决定
23.8.1 照明
23.9 评估
23.10 满足的其他迫切需求
23.11 在设计中使用图纸、CAD、模型、仿真模型和虚拟环境
23.11.1 虚拟环境的发现
23.12 学到的一般经验
第24章 案例研究：System/360体系结构
24.1 亮点和特性

24.2 项目介绍和相关背景
24.2.1 相关背景
24.3 目标
24.3.1 主要目标
24.3.2 其他重要目标
24.4 机遇（截至1961年6月）
24.5 挑战和限制
24.6 最重大的设计决策
24.7 里程碑事件
24.8 结果评估
24.8.1 稳定性
24.8.2 有用性——竞争力，各个市场的分析
24.8.3 闪光点
24.9 取得的经验教训
第25章 案例研究：IBM Operating System/360操作系统
25.1 亮点和特性
25.2 项目介绍和相关背景
25.2.1 System/360系列机型
25.2.2 1961年的软件格局
25.3 接受挑战
25.4 设计决策
25.4.1 系统架构
25.5 结果评估
25.5.1 成功之处
25.5.2 设计中的不足
25.5.3 流程中的不足
25.6 设计师团队
25.7 取得的经验教训
第26章 案例研究：《Computer Architecture: Concepts and Evolution图书设计
26.1 亮点和特性
26.2 项目介绍和相关背景
26.2.1 相关背景
26.3 项目目标
26.4 机遇
26.5 约束
26.6 设计决策
26.7 结果评估
26.8 经验教训
第27章案例研究：联合计算中心组织：三角区大学计算中心
27.1 亮点与特性
27.2 介绍与内容
27.2.1 内容
27.3 目标
27.3.1 主要目标
27.3.2 其他目标
27.4 机会
27.5 约束
27.6 设计决策
27.7 董事会所考虑的投票方案

27.7.1 权力均衡的限定
27.8 测量评估
27.8.1 牢固性
27.8.2 实用性
27.8 经验总结
第28章 推荐阅读
致谢
参考文献

《设计原本》

章节摘录

版权页：插图：理性模型是一种自然的思维模型。理性模型，如上所述、如上所评，似乎看上去相当幼稚。但它是人们能够自然而然地想到的一种思维模型。其思维自然程度可以从Simon版本、瀑布模型版本以及Pahl和lBeitz版本分别独立地提出而得到强烈的印证。然而，从最早的时候开始，设计界就有了对于理性模型有说服力的批评。设计师们根本不那样做事。也许对理性模型最具解构性的批评——尽管也许亦是最难以证明的——就是经验最丰富的设计师根本不那样做事。虽然已经发表出来的批评只是偶尔才会有“皇帝没有穿衣服耶！”这样指出该模型并未反映出专业实践做法的呛声，但是人们还是可以感觉到不厌精细的分析背后的这种压倒性的主张。。 Nigel Cross，其绅士般的言论，也许是最具张力的不同意见。引经据典之下，他毫不讳言地说：有关问题求解的习惯思维，往往看起来和专家级设计师们的行为背道而驰。不过，设计活动和使用习惯思维进行问题求解的活动有着相当多的不同之处……我们必须在从其他领域引入设计行为模型时倍加小心。对于设计活动的实证研究经常会发现，“有直感力的”设计能力乃是最有成果的，也是和设计的内廪性质最密切相关的。不过，设计理论的有些方面就是企图针对设计行为开发出反直觉的模型和处方来。又及，在绝大多数设计过程的模型中都忽视了性质上处于同等地位的设计推理。在有着公论的关于设计过程的模型中，比如说由德意志工程师协会颁布的模型中（VDI，1987）……就主张设计活动应该分成一系列的阶段进行……在实践中，设计活动似乎是以在子解域和子问题域的区间振荡的方式进行，同时也是将问题分解，尔后合并所有的子解决方案的过程。我发现，这个争鸣意见本身以及其实证材料都很具说服力。此处提及的振荡的确可以说是我所有设计经验的特点所在。“外套在哪里摆放？”这样的需求发掘了我们房屋设计过程的深层次内容就是个典型例证。Royce对于瀑布模型的批评。Royce在他的论文原稿中描述了瀑布模型，以便他能够演示其不足之处。”他的基本观点在于，尽管在毗邻方块之间有反向箭头表示逆向的工作流，但是该模型仍然行不通。不过，他的对策仅仅是采取了容许逆工作流箭头指回两个前向方块的模型罢了。治标不治本。

《设计原本》

编辑推荐

《设计原本:计算机科学巨匠Frederick P.Brooks的思考》：如果说《人月神话》是作者刚刚完成若干个改变了全球计算系统格局的重大项目，在人生和事业的巅峰时期的激情之作，那么《设计原本:计算机科学巨匠Frederick P.Brooks的思考》则是作者功成名就之后。在研究和教学中将先前在设计领域中的探索心得和实践经验切磋琢磨、去伪存真、取其精华的反思之作。可以说，比起锐气有余的《人月神话》，《设计原本:计算机科学巨匠Frederick P.Brooks的思考》更多了几分高屋建瓴的大局观以及数十年如一日积淀而成的丰富材料，是设计领域真正的大师之作。《设计原本:计算机科学巨匠Frederick P.Brooks的思考》几乎涵盖所有有关设计的议题：从设计哲学谈到设计实践，从设计过程到设计灵感，既强调了设计思想的重要性，又对沟通中的种种细节都做了细致入微的描述，并且谈到了因地制宜做出妥协的具体准则。其中，作者特别强调的是设计的概念完整性，这不仅对于设计过程中步骤流转时的信息传递十分关键，并且也是沟通中最需要重点注意的地方。全书的案例研究是另一大亮点，在进行抽象的模型和思想论述时，作者时时注意运用图表和案例说话，深入浅出地表达复杂艰涩的思想。并通过层次丰富的案例。展示了设计既能治大国，又可烹小鲜的强大力量和无穷魅力。我们能够从作品的字里行间感受到计算机体系结构刚刚诞生的黄金年代充满了怎样的设计思想和工程实践的生机和活力。而作者也十分擅长专业史料的记载和整理，并且以他独有的方式为读者展示出极为清晰的脉络。无论是软件开发、工程还是建筑，有效的设计都是工作的核心。《设计原本:计算机科学巨匠Frederick P.Brooks的思考》将对设计过程进行深入分析，揭示有效和优雅设计的方法。《设计原本:计算机科学巨匠Frederick P.Brooks的思考》包含了多个行业设计者的特别领悟。作者精确发现了所有设计项目中内在的不变因素，揭示了优秀设计的过程和模式。通过与几十位优秀设计者的对话，以及他自己在几个设计领域的经验，作者指出，大胆的设计决定会产生更好的结果。作者追踪了设计过程的演进，探讨了协作和分布式设计，阐明了哪些条件造就了真正卓越的设计者。他解释了设计过程的具体细节，包括多种预算约束条件、美学考虑、设计经验主义及工具。同时，他将这些讨论与现实中的案例结合起来，这些案例从房屋建造到IBM的Operating System／360。贯穿全书的成功的关键因素，是每个设计者、设计项目经理和设计研究者都应该知道的。《人月神话》作者最新力作，计算机科学大师探究设计原本。

《设计原本》

精彩短评

1、理解高层次的设计理念，对以后设计系统架构，多方面多角度理解问题，发现潜在问题，很有帮助。
2、用心看，用心理
3、概念完整性
4、非常牛逼，非常没用。太能东拉西扯了，我的蛋好疼。
5、刚开始读
6、我的年度最佳之一
7、看了一点 不敢妄论
8、:
TP311.5/4224-6
9、设计之设计。
10、这种上流的书我档次太低体会不到精髓...
11、计算机科学巨匠的思考，真本书真正的教会我们如何思考。虽然书写的不错，但是看过之后真心感觉中文版翻译的真是太烂了。
12、这本书并不是介绍具体的计算机领域的设计案例，而是站在更高的维度，从工程师设计的角度，讲解了设计过程中需要面临的过程挑战、约束、目标、协作等问题的探讨
13、我想一把火烧了机械工业出版社。书中案例对我来说有些遥远，评价一般。
14、讲了很多os360开发,IBM开发这个系统的故事, 讲了开源的集市开发,是人月神话之后, 又一力作,项目经理和管理者以及设计人员更应该读些书
15、应该说The Design of Design是一本很有趣味的书， Frederick P. Brooks, Jr.在多个领域中进行的架构设计精彩而又有价值，揭示了设计的本质。很多设计的经验、技巧都是来自于作者的开发经历，而且作者本人在架构中承担了重要的角色，从中看到了作者对纯粹的架构设计和计算机架构、建筑设计、组织架构设计的思想，感觉受益匪浅。架构设计并不只是在纸上画出一个草图甚至知识大脑中的一个结构，就像德鲁克对管理的描述类似，它是一种实践的过程，通过优秀的设计，实实在在的发现需求，并且满足需求，确定的开发出系统；不但要考虑所有可能的属性和约束，而且要能够确定无疑的实施，是通过确定的工具和方法论系统真实的被完成，或者证明错误，对过程也要有管理和设计；而且动态的应对变化。
16、作者是个通材，硬件，软件，建筑，写作。每章的内容很少，都是泛泛而谈，很难有深入的认识，自己太外行，或者能力不足以参透。但我猜一个同时喜欢这么多东西的读者应该会读得很爽。
17、没看呢。不知道怎么就让我写评论了。。。汗
18、书不错，要想有收获的话，比较费心。翻译水准实在不敢恭维，编辑很随意地选择了几个人翻译，似乎是只要学校不错，翻译过东西，或者有热心，就ok了。等写完，看翻译成这样，将就着出版了。只能说，一本好书，被一个不负责任的编辑糟践了。
19、以前看过人月神话，现在又看着本设计原本感觉真的很不错，如果几年前看几好了。呵呵
20、一般吧。。。
21、书比较薄，不过内容很实在。
22、读过关于设计非常有见解，深入浅出的解析设计、框架的基本要素
23、万事皆系统，系统须设计。如何设计、设计纬度……各学科通用，大有助益。
24、都不知道该如何归类。严格意义上。我不能算是看过
25、纯属收藏了
26、很不错的一本书，不过理论性很强，适合细细品读，反复阅读。
27、从s/360的经验中总结了需求和最终程序实现之间如何进行需求分析和设计决策，经验很实在
28、难得的好书，受益匪浅。
29、有深度，不是那种随便看看的总结、观点，讲了作者对软件设计的深刻理解。
30、纸质一般，帮老公买的，内容无法评论。
31、网站设计现在很重要
32、自己还暂时不能理解完全，以后再看

33、冲着《人月神话》作者去的，看完有点小失望。
34、翻译还是有点问题 但总体来说不错了;如何衡量系统的权重, 对于系统要考虑的方方面面 不错的参考散列点.应当会是对于"系统"经验越多 感受越深
35、内容好，印刷质量一般
36、好看，一下午读完不是梦
37、不过书的质量还是很不错的，内容也挺充实
38、精华需要沉淀，很好，值得持续品读
39、大师的作品，值得阅读！
40、它让我第一次面对广义系统设计的具体内容。很少有这样跨领域的设计师以跨领域地态度描写设计活动了。
41、看了前面，看了有点难懂
42、很好 业余时间看看 好好
43、对于理解设计
还是有一定帮助的！
44、在迷途中摸索，不如借鉴下前辈们的想法。
45、读了一半，目前没有什么感觉。
46、帮朋友买的，感觉不错哈，包装很好
47、这本书比想象中生动有趣，以房屋建造等策划案例，来描绘出一个项目从设计到实现的流程。
48、观点很精辟，推荐订阅该书
49、大概是因为自己缺乏经验吧，难以跟作者产生共鸣。作者自己都承认是经验主义者。纸上得来终觉浅，绝知此事要躬行。
50、似懂非懂要再读一遍。

51、路上用零散时间读的，平时不大从这个层面看问题，还是有些启发的，
52、没有多少比《人月神话》新鲜的东西，最后的几个案例也无太大价值。培根说，将对一门艺术的领悟联系并应用到另一门艺术中，经过若干次这样的经历而有所悟，脑海里自然就孕育出了新思想。作为一项工程，总是喜欢和建筑类比，但建筑自人类伊始就有了，而软件工程不足百年，而且建筑的设计可以数字化控制了.....
53、光看英文的序我就知道要买了，大牛的经验之作
54、大师之作 值得拥有 慢慢阅读
55、我这个小白看不大懂。。。
56、“ 设计师的主要任务乃是帮助客户发现他们想要的设计 ”（18页）
我们干的更多只是工匠活而已
57、和预想的不同，名为设计原本但并不是对设计思想的专门论述，而只是一本专题文集。
58、螺旋模型也许是最好的模型，因为需求在一开始就确定是很少见的。团队设计最重要的是设计概念的完整性。最佳的设计也需要范本来超越，卓越的设计需要卓越的设计师。
59、其名其妙，不知所云。字面意思不难，但看不出跟设计的联系。大家都说软件不像建筑，作者偏偏爱用自家房子举例。
60、书不错，正在看，值得一读
61、只能说大师的杰作确实不同凡响，感觉国内的顶尖人物还是和国外有很大差距的，在国内这个浮躁的环境是不可能有这样的作品的，很难有人能够像作者一样沉淀下来。理论科学的研究是一切科学技术的基本，中国现在只能说是和人家越来越远，差距越来越大。这和整体的社会风气和科学风气是分不开的，那些只知道向钱看的所谓专家和教授不知道还要祸害到什么时候。在社会上混的越久越能体会那种中国的浮躁和自己的无奈。希望我们国家能够沉淀下来，不要再像暴发户。
62、书是好书，但是模仿《几何原本》（Elements of Geometry）把《The Design of Design》翻译成《设计原本》有故意拔高的嫌疑，而且大概是出于这个原因，翻译上也显得文邹邹的，搞清楚读者群好不好，俺们理工科出身不好那口之乎者也，别东施效颦了！另外，把莱特（Wright）兄弟翻译成“ 怀特兄弟 ”，您哪儿的口音？摘要与心得：http://book.douban.com/review/6231417/
63、1、有高度。

《设计原本》

2、读后让你内心很平静，有提升超脱的感觉。
64、每种提出都表现出作者的专业水平
65、还正在看 不过风格的确别具一格 开篇就比较吸引人
66、Brooks的一些很珍贵的项目经验分享
67、还没看完 质量不错
68、当到货，大师的作品，应该错不了
69、有点吊，还得看几遍
70、很好，很形象，正在读。
71、图文并茂，讲设计，而不单单是软件的设计
72、还行，设计的本质是约束条件下，对外来前瞻性和对当前的兼顾。这一点没写出来
73、看不懂
74、还没工作，没啥感触，先买了没事看看
75、事实上拜读之后，我获得的最大的启发是在项目管理而不是设计本身。设计毕竟是个过于庞大的话题，不可能通过这样薄薄的一本书简单的陈述完整，我认为这也超出了作者的能力（毕竟即使只是单纯的软件/硬件设计现在也有巨大的分化和差别，这一点作者也提到了）。但宗师毕竟是宗师，每次提到一些项目运行的话题和一些底层设计的时候，还是总能提出非常有启发性的观点。

第八章读得我晕头转向，不清楚是翻译还是哲学名词的问题，或者是当时状态不够清醒，试看原文：仅仅依靠深思熟虑就能正确设计好复杂的对象吗？……经验主义者认为可以；理性主义者认为不行。
理性主义者认为……在经过……足够仔细的思考后……可以做出完美无暇的设计。
经验主义者认为人类天生就是有瑕疵的……人类做的任何事都是有瑕疵的……任务……根据实验找出瑕疵。
笛卡尔……理性主义者观点；John Locke……提出了经验主义者观点。
（以上均出现在 P.71）
笛卡尔拥抱经验主义，而 Locke 则认为理性主义是数学的基石。（P.74）
以上是我觉得前后矛盾的文字。

PS. 作者在装修厨房的时候居然动用虚拟现实实验室……羡慕啊
76、举例虽然琐碎，但是意在揭示本质。
77、经典且易懂
78、确实翻译得很糟糕。。完全有失原作的风采。读起来磕磕绊绊的，而且晦涩难懂。看完中译本感觉心情很不好，所以又去看原本了。结果原本很惊艳，这下子心情更不好了。。唉。。什么时候关于计算机的优秀丛书可以翻译得好一点呢？
79、设计的设计是什么？
80、大师作品，引人思考啊！何时中国才能有类似的研究呢？
81、经典杰华，值得阅读！
82、不愧为巨著，非常好，强烈推荐
83、《人月神话》的作者有一大作
84、软件设计的新经典
85、书的内容真的很不错。我相信，如果你有这本书就喜欢上的。
86、这个版本翻译的太差，还是要看原版
87、看不太懂 又或者是本书讲的层次比较高一点 难以运用到实际工程之中
88、作者就是写《人月神话》那个。书名很唬人，其实主要还是讲软件设计的，只不过推广了一下。应该是文集，结果很散没有中心思想。
89、我的看法是参考下吧，还行。
90、2014-08-15：没怎么看懂，层次太高了，海滨小屋的设计对买房子有参考价值。//2014-07-01：公司借的，还没看几页
91、为什么不给积分啊？？？？？？？？？？？？？？？？？？？？？？？？？？？？？
92、大师的书，工作一段时间后再回头看一遍

93、设计方面的high level的书籍，并不局限于软件设计
94、有过一定工程经验的人看应该会更有感触
95、翻译得不好。内容也比较碎。
96、好书需要慢慢读的
97、物流快
质量好
还没看
98、软件工程必须多个人完成是一个骗局
99、通用的设计思维方式。
100、可能高度不够，看得迷糊
101、没怎么看懂

《设计原本》

精彩书评

1、注：【】部分为笔者心得，非原文摘抄。* 新思想来自于将对一门艺术的领悟联系并应用到另一门艺术中，历经若干次这样的经历而有所悟，脑海里自然就孕育出了新思想。——《The Two Books of the Proficience and Advancement of Learning》* 设计代表着：** 概念性构思的形成；* 在真实的媒体中实现；* 在真实的体验中与用户交互。* 对大多数的创作者来说，构思的不完整性和bu'yi'zhi'xing 只有到了实现的时候才变得明显起来。* 构思、实现和交互这三个阶段的操作是循环进行的。* 【想法需要实践才能得以完善。】* 良好的设计j具有概念的一致性——统一、经济、清晰。* 【不要相信项目经理说的“ 我们前期抓紧点儿，后期就不用太辛苦 ”，这是句笑话！不过也不要因此而放松对困难的警惕，我的意思是全程都要抓得很紧。】* 现实情况是，设计师只把理性模型视为一种理想化的东西。它以某种方式描述了在我们的认识中设计过程应该如何运作，但在现实生活中，并不是那么一回事。* 设计中最困难的部分在于决定要设计什么。* 设计师的主要任务是帮助客户发现他们想要的设计。* 要在设计过程的一开始就明确地列出已知的约束，作为架构师所谓的设计任务书的组成部分。* 任何在项目伊始就规划所有的可能需求的企图都会失败，并以可观的延误告终。——《Engineering Design》* 需求激增现象必须予以钳制，方法是防患于未然或将这种势头遏制于摇篮之中。* 及早任命手腕强硬、经验丰富、领域知识到位的经理，并要求他们在整个初始系统交付期间全程参与。尔后，授权他们“ 以其认为必要的方式度身定制标准流程和步骤 ”。* 创新性设计并不是先把问题定死，再去寻找一个令人满意的概念解决方案这么一回事；它似乎更像是对于问题的构造本身以及解决方案的思路这两者同时进行研发和完善的工作。——“ 创造型设计中问题和解空间的共同演化 ”* 所有设计过程都共有的关键要素在于对用户的需求、希望和验收标准的发掘。* 与具有代表性的用户保持经常性的而非连续不断的互动，沟通的手段则是相继改进的原型。* 线性地按部就班的理想模型具有很大的误导性。* 【智力密集型工作不遵循“ 人多力量大 ”。】* 在团队设计中，确保概念完整性最重要的方式就是授权给一名系统架构师。此人必须在相关的技术领域具有能力。他必须对要设计的这类系统拥有经验。最重要的是，他必须对系统的特点和目的具有清晰的愿景，必须真正关心系统的概念完整性。* 头脑风暴的标准规则是：关注数量、不要批评、鼓励发散思维、组合并改进想法和勾画出所有想法让大家能看到。* 概念设计尤其不应该协作。* 真实的设计有着更复杂的目标和更复杂的约束条件要满足，对于满意程度的测量也更复杂。真实的设计总是爆发出无数的细节。* 真实的团队设计总是需要一个设计变更控制过程，以免我们左手弄坏右手创造的东西。* 无论多少协作都不能消除对“ 枯燥无味的劳作和孤独的思考 ”的需要。* 远程协作技术让越来越多的专业人士能够住在他们喜欢的地方，并在别处工作。* 【远程团队协作必须保证所用的所有设计和开发软件版本一致。】* 最成功的远程协作建立在大量的面对面沟通的历史基础之上。* 有用的工具创造总是从用户和任务开始的，最好是在工具创造者有一个真正的用户和一个真正要完成的任务时。* 理解力有两种形式：直觉与推演。无论哪一种都需要以知识作为后盾。——《Rules for the Direction of the Mind》* 分析得越精细，就越能精确地度量出必要条件的满足程度以及约束的遵从程度。* 真理来源于错误而非混乱。——《新工具》* 假设越详细、越明确就越有助于设计者们提早进行具体的思考。* 缜密的猜测胜于无言的假设。* 总架构师必须具备率领团队前进的能力。* 明确的假设即便是错误的也要好于含糊的。错误的假设起码会受到质疑，而含糊的则不会。* 如果希望设计，特别是团队设计具备概念完整性，那么你应该明确地指定好稀缺资源，公开地跟踪并严格控制它。* 设计者的必要行为是有意识地预算。* 整个团队需要清楚关键资源的当前预算。* 只能有一个人来控制预算与重新预算，这是必须的。* 通用产品要比特定用途的产品更难于设计。* 一般来说，目的越具体，设计任务就会越容易。* 风格是思想的外衣，良好的思想就像是穿着考究的绅士一样更具优势。——Chesterfield勋爵* 好的建筑架构要满足“ 坚固、效用与情趣 ”的要求。——Vitruvius* “ 优雅 ”需要简约。* 【真正的简约并不束缚发展空间。】* 技术设计中的“ 优雅 ”要求设计中基本的结构性概念要易于为人所理解，否则逻辑就应该是直截了当且易于解释的。* 没有包含所需功能的架构是错误的架构。* 不要将独立的东西耦合起来。* 不要引入无形的东西。* 【组件设计应该保持与整体风格一致。】* 要想获得一致的风格，就得写成文档。* 清晰的风格未必是好的风格，但混乱的风格则绝非好的风格。* 有抱负的设计者必须要为风格的一致性而奋斗。* 设计团队必须将设计恰当地记录下来，团队还必须在维护阶段保持概念上的完整性，将管理与构成可视化设计的各种微观决策记录下来。* 有目的地学习其他设计者的风格。* 为你的产品选择拥有清晰风格与高品位的设计者，根据他们之前的工作作出判断。* 在任何设计领域中都不能懒惰，都需要高度的热情

与勤勉才能掌握大量的范本。* 7天的惊奇很快就会过去。随着新颖的退却，情趣也将一并消失。寻找新奇事物的人永远都是被动的。没有永恒不变的情趣。* 把欲望当做动力会让很多作品走向堕落。* 困扰专业设计者的错误不在于错误地设计了东西，而是设计了错误的东西。* 对于专业设计者来说，成功是一件危险的事情。失败会促使人们去分析、仔细审查以及重新思考。成功会导致设计技术与设计者过度自信。这两种信任可能会发生错位。* 个人产品的设计者总是其用户。* 在修复你不懂的东西时，要小心。* 设计不只是满足需求，也要发现需求。* 设计不是简单地选择可选方案，也需要意识到存在潜在的可选方案。* 金字塔、大教堂以及火箭之所以存在，并不是拜几何、结构理论或者热力学所赐，而是因为这是它们的创造者灵光一闪的产物。——《Engineering and the Mind's Eye》* 渐进完善模式真正的危害在于范本库本身。拙劣的样本、太少的模型、过于狭隘的范围——这些不足之处都将极大地限制呈现出来的设计。* 卓越的设计来自卓越的设计师，而不是来自卓越的设计过程。* 从过程的本质上来说，产品过程是“ 面向否决 ”的，目标是阻止不好的想法并且抓到疏忽。* 对于创新而言，人们必须跳到过程之外。* 卓越的设计需要大胆的领导者，他们要求创新。* 经理本身一定不能对设计放马后炮。* 每个超出一般水平的人都接受了两种教育：第一种来自他的老师；第二种更私人也更重要，来自他自己。——《Memoirs of My Life and Writings》* 必须为卓越设计而招募人才。* 培养设计师与培养经理不同，第一件事就是要为他们打造一条职业发展道路，让他们的报酬和社会地位能够反映他们对创新企业的价值。* 寻求有知识的人对你的设计进行批评。* 即使是诚实的、有能力的、尽职的架构师也会犯错误。* 任何成功的设计都需要你维护很长时间。* 架构总是可以看成是设计过程的原型。——《The Sciences of the Artificial》* 大多数项目需要将总时间表的更多部分投入设计工作之中。* 从相同的体系结构出发，给出多个并存的实现。* 对于全新的设计，而非后续产品而言，从一开始就要将设计工作的一部分投入在性能及其它必要属性指标的建立上，并做好成本估算。* 市场预测的方法论是为后续产品，而不是为全新的创新产品所设计的。* 【让一线人员参与决策。】* 给予系统架构师以充分的设计授权。

2、2011年是IBM 诞生一百周年，IBM全球董事长及首席执行总裁彭明盛SamJ.PalmIS ano于2011年2月在美国约翰霍普金斯大学发表了IBM百年系列活动的开幕演讲。这100多年的历史告诉我们了些什么呢？记得，在IBM 50周年庆上，小汤姆沃森，也就是IBM公司的董事长和其创始人的儿子在在另一所伟大大学的讲堂里，向未来的领导人们发表了演讲。当年演讲时，全球财富500强企业的前25家企业，在2010年只有四家依然存在。而IBM100年的历史人物中，有一个人物不得不提，那就是IBM 360系统之父计算机科学巨匠Brooks。或许中国的很多读者还不认识他，但提及这本刚推出的《设计原本》可以说是作者功成名就之后，从业60年的巅峰之作。作者在从业60年里，扮演过软件架构师、建筑设计师、企业家、作者等几个的重要角色，作者发现所有行业背后的设计过程都是出乎一致的一样。于是将自己60年的经验集结成书，告诉大家只要你能做好一件事，就能做好任何一件事情背后的设计过程。

3、译者的翻译水平很差。译文里虽然没有错别字，但充满了佶屈聱牙式的直译，阅读的体验极差。一句话需要看上三五遍才能看懂；再看一遍，基本上就可以直译成英文了。如果译者还有计划翻译类似的专著，建议译者仔细研究一下翻译理论，免得误人子弟。

4、因为很fan作者，所以很快买了此书。说老实话，前面几章还凑活，经验之谈，也未见有何理论体系可言，后面讲几十年前的老故事，还写了自己如何设计自己的屋子的故事，也许在米国有很多共鸣，我看得索然无味。牛人写了本一般的书，我个人表示失望了

5、1000copy Review for“ the design of design".从论坛和微博和同事等渠道，我知道fredicks brooks，那个《人月神话》的作者有出书了。同事买了一本《设计原本》，昨天晚上清净的时候，就顺便看看，所幸厚度不大，花了N个小时通读，这本书的架构基本上比较清晰了。可是从阅读以来就不太平静。想要说说我的看法。首先是阅读感受。“ 这不是一本可读性比较强的书，并且不值得回头在看 ”。说实在的，这和我原来的看法相左，有点难受——毕竟这是一个前辈，不说老而弥坚，至少文字应该更加容易阅读，更加能够照顾读者的阅读顺流的习惯。这本书给我的感觉是不停的切换，不停的跳转，如同代码中很多goto语句一样，让本来可以优美的设计之源变得非常的犬牙差互。其次是读者范围。我认为如果你刚刚有一些设计经验的话，不需要考虑购买这本书。因为它的英文名字是the design of design ,这个题名还是比较切合内容的。它的着眼点是对设计进行设计，就是说，这是元设计。也许你会说，元设计难道不是设计吗？OK，programing 和metaprograming是完全不在一个数量级的，正如设计和元设计的复杂度不在一个数量级一样。meta打头的没有不复杂的，你不妨看看任何一个语言都可以用helloworld还快速开始，而MPS(Meta programing System——jetbrain的一个产品）这样的元编程就复

杂的太多了。mps的 tutorial你试着阅读下，看看几天内是否可以把它搞得清楚明白。我不认为一个退休了，忙着装房子的老人能够有能力驾驭这个主题。如果你设计经验非常丰富，比如5年以上，我建议你也不要购买。因为太过空泛，难以让人可以从中得到比较直接的指导。继续从副题名看，Essays from computer scicent。这个Essays，就是我们常常说的格言，想想看，格言拿来干什么，通常是给中学老师教育小孩子的时候，通过断章取义灌输自己的私货的。尽管我上面一段提到他的着眼点在于meta design，可是他的论据是什么呢？不外乎是20年多前的IBM sys 360，和功成名就后自己建立的一个house的过程。作为一个程序员或者程序设计师，你真的关心他的house如何建立，他如何分配厨房，卧室给他的老婆Nancy和几个孩子，以及如何采光等等吗？论据的不足必然说明这是一个他想要做design of design 的一个尝试，也必然是非常不成功的一个尝试。Fredicks不是天才，如此大的跨界（从程序项目管理到建筑设计）我没有见过真正成的先例。不然你让克里斯托弗·亚力山大（建筑设计模式的作者）谢谢软件界的设计模式看看。即使本领域内的，以他举出的JCL（任务控制语言）的设计为例，就说的不明不白，一个项目经理是不可能对设计提出真正的意见的。依我看，从JCL设计失误的评价上看，他感叹人生多于对设计本身原则的提炼。想要计算机考古的人，不妨买来看看，反正做考古的人有的是时间，大可以顺着所谓的信手拈来的典故，后面如麻般的参考文献顺藤摸瓜。你还别说，参考文献这么多的，比如你还有兴趣了解IBM JCL 语言规范的，可以真的去看看。当然你这样做还有一个机会成本，那就是你可能需要摸到IBM的档案柜前面，而置手边的google和wikipedia这样的宝库而不顾。说不定档案柜内的纸板的handbook摸起来就会手感很好啊。和brooks一个年代的人可以看看，当然我随便说说的，我这年龄，还无法设想这样的场景：“ 坐在炉子边上，抚摸着一条老黄狗，和孙子说，你爷爷年轻的那会，设计个高级的ＪＣＬ语言，最后和汇编的开发效率差不多哈 ”。喜欢复杂的,绕脑筋而不求功利的人可以看看。我是说真的，“ 我有一个很厉害的程序员，花费了几个月时间研究hashell，我问他为什么看这个几年内也用不上的东西呢？他说，因为他复杂 ”。再说一遍，我是说真的，我很佩服这个程序员的。他尽管hashell没有用上，但是他做事情的缜密思维大约和喜欢研究复杂的实物有关。不过这样的人我工作多年，也只看到一个。如果你认为你是这个百里挑一的人，你可以试试。话说回来，你要试试，不如试试hashell，因为，你知道，我又要说机会成本了。关于案例。这本书的案例中的house建立完全就是败笔，可以直接拿掉。他在这个领域是不专业的，他在给自己建房子，他没有客户，没有人评价他，没有人为这个事情买单，他就是自己的用户。你真的可以相信一个给自己建了一个房子的人可以在建筑领域大谈经验吗？我看了amazon的本书评价，其中有几个给了3星或者以下评价的都有类似的看法。另外一个案例里面他提到了一个劳斯莱斯的什么牛的设计系统。说说我刚刚有点感觉，他马上补充了：”这个东西有专利的，劳斯莱斯不让看“。所以——你知道，就没有下文了。这是21世纪啊，还有这么神秘的、刚强的，不为人知的、但是有非常实用的一个软件存在，真的吊起了我的胃口。然而通过继续看这本书找答案显然是死路一条的。等我哪天有兴趣了，我用万能的wikipedia和google查查，是不是真有其事。关于写作风格。而brooks 的案例的编写看起来不是一本书的章节，而更加像是powerpoint的keyword。大量的dot bullet，大量的跳转和上下文的切换。如果把人的大脑比作cpu的话，这样多的切换真的如同一门语言的名字：brainfuck。在powerpoint这样写不完全是问题，因为powerpoint是为了讲课服务的，讲课内容是主体，speaker是主体，ppt只是一个纲要，很多时候，这不是问题。但是，拜托，这是一本书啊，你brooks不在我的身边啊。不知道译者的原因还是本来如此（应该不是本来如此），其中看到的摆在章节开头的一些英文的essays都存在明显的大小写问题，比如sOFTwAre这样的单词。有点像是过山车，突然来到一个年轻的、hacker的领域。我知道一个朋友做hacker的，他的网名就是这样的奇形怪状。至于其中内部存在的一些重复的句子，重复的单词，让我想起《忐忑》。计算机科学巨匠啊，这样急迫的出中文版，是不是有点像是一代宗师非要和巨鲸帮火并呢。这样的书籍在国内的书评几乎千篇一律，在amason上也只有区区19条，总觉得本来是华山论剑的谱，突然变成了岳不群的君子剑法。不妨看看 amason的主页：http://www.amazon.com/Design-Essays-Computer-Scientist/dp/0201362988/ref=cm_cr_pr_product_top这里的评价还是很中肯的。我试着列出持推荐态度的人的观点如下：1. 要买。因为这是《人月神话》的作者，IBM360会是其中的案例。可是《人月神话》成功，不意味着这本书也会成功。我常备多年的书籍包括winberg的系列，人件，cc2，重构，都是反复阅读的常常可以获得新知的，但是《人月神话》，我认为只是对IBM360这样的史前巨兽好奇而偶尔翻翻。2. 要买，因为他鼓励的个体的天才而不是委员会的臃肿。可是，Apple这么红不就是因为如此吗？福特当然汽车行业老大不也是因为如此吗。60年代的时候这一点并不显然（也许），而今互联网时代，Agile大行其道的年代，这一点不是太显然了！何

况和jobs那样鲜活的经历相比，brooks的ppt体书籍也实在不能支持这样的论点。真的谈到人性的价值的话，还是要看《人件》，这是经典。3.要买。因为No-silver bullet。这个论据我不喜欢——未必是因为论据本身，而是因为总是说这个的人如同背诵语录，让人觉得已经大脑麻木了，何谈让别人相信呢。正如引用《堂吉诃德》的人必然说战风车一样，让人怀疑说者是否真的看过《堂吉诃德》。在我看来《堂吉诃德》的吸引人之处不是这个傻骑士的不传奇的经历，而是贯穿其中的几个美人的传奇的爱情故事——还有异域风情的，比如摩尔美女。有美人、有传奇、有异域，好不好？4. 提出了设计约束的存在。好吧，这么显然的东西说出来可不能算是发现吧。为什么一个类似笔记体的内容，会被当成书籍来出版呢？名人的光辉，常常让大多数人主动的带上眼镜，然后。。。失真，在然后。。。失贞。失去自己的判断能力，然后既无所得，然后不敢质疑权威，然后默然。每念于此，常常想起那个牛人卢梭说过：“ 人生而自由，却无往不在枷锁之中。这是一本大失水准、远低于期望的书。这说明有些老人该腾出地方，好好的搞自己的房子，过自己的生活，比如去瓦尔登湖那里钓鱼什么的，然后给其他新的人杰一些出版和发言的机会。感谢达尔文的进化论。毕竟“ 个体的力量无法承载进化的能量 ”。

6、好多句子不通。“ 但却失之过分繁复 ” 这种明显影响阅读效率的句子就不能翻译的舒服点？“ 经常地，多个模块以及对于同一个缺陷的多个修复会同时提供给社区 ”。现在的翻译都直接保留原句句型的？总言之，写的不错，看得很累。

7、“ 有没有一种方法，可以作为设计的基本原则，在我每次进行设计的时候都给与我帮助？ ” 苦于抓不到设计本质的我请教一位学习平面设计的好友。他给我的建议是在每次设计时回答三个问题：1.Who：这个设计的受众是谁？2.What：这是哪方面的设计？3How：具体应该如何去做设计？Brooks在本书中提供了一个更加详细的理想模型：1.目标2.必要条件3.效用函数4.约束5.资源分配，预算和关键预算6.设计树遗憾的是，这只是一个理想中的模型，在现实中并不一定适用，例如很难确保设计树是完备的，没有遗漏最合理的设计方案。作者推崇的Boehm模型虽然没有详细论述，但无疑更自然些，毕竟我们的自然界正是一个不断演化的产物。关于卓越设计师的教育是我非常赞同的，在其他领域也有越来越多的人在反思类似的问题。我们的知识体系已经太过庞大，庞大到如果先去学习相关的内在原理需要花费数年的时间，而与实践相脱节的理论学习往往不能让学生把握要点。诚然现代教育体系如同流水线培育了大批具有专业知识人才，但卓越的设计师永远不是流水线能够锻造出来的。个人认为最合理的培养机制应该是类似以往手工业流行的学徒制，教授者本来就是实际工作的参与者而不是仅仅是理论研究者。师傅能把自己的经验教训和理论知识倾囊相授，而徒弟能在掌握理论知识的同时具体地去参与真实项目的设计工作。这样的培养方式对师傅的要求很高，徒弟的数量也上不去，但为了得到卓越的设计师，这一切都是值得的。

8、我是冲着Brooks的名头去买这本书的。读的第一本Brooks的书是汪颖翻译的《人月神话》，翻译、审核都不错。然而这本《设计原本》质量应该说还有待提高。今天在飞机上花2小时仔细读了几章，发现了一些难以理解的地方。我看能否找到原文，对比一下才知道是翻译的问题还是原文就如此。

9、这部巨作是《人月神话》作者布鲁克斯最新力作。译者说，《诗》有之：“ ‘ 高山仰止，景行行止 ’。虽不能至，然心向往之。 ”在此引用一下，果然是大师之作啊。在书中，布鲁克斯尖锐地指出：” 很多设计者从心理上和实践上都没有对设计过程进行很好的理解。大部分的设计成本是返工，即纠正错误，这通常达到总成本的1/3。平庸的设计肯定是浪费了世界的资源，破坏了环境，影响了国际竞争力。“ 良好的设计不仅可以工作，而且能带来快乐。本书指引工程师关心效用和功能的同时也注重效率和优雅。设计的最高境界：不是一个技术活儿，而是一个艺术活儿。作者通过桥梁的设计来类比，确实，软件设计也是一个工程类的设计，这个和我们的建筑设计、桥梁设计等等都是一样的。这个本身可以是一个创造美的过程！没有一个学科是可以独立的，布鲁克斯还是个不错的建筑设计师呢。所以说，触类旁通，学术间的交叉融合可以带来不一样的惊喜！我个人的感受是，软件设计原来不是那一小片天！这是布鲁克斯告诉我的。在第二章《工程师怎样进行设计思维——理性模型》中，布鲁克斯以在海边盖房子这个生活化的例子来阐述设计者考虑设计过程，深入分析了设计模型背后的工程思想，通过将工程师的思维过程程序化地展现，批判了这一模型。深刻分析了它的本源，它为什么能够生存，从而自然而然地指出它的缺陷，指导我们在什么情况下用比较和谐。同样，作者在书中对主流开发的各个方面进行了描述,但是缺乏解决的具体手段和思考策略,作者也希望的是读者自己根据书中的线索去领悟自身需要的设计思考方式，因为设计是没有界定的吧。布鲁克斯鼓舞我们：大胆的设计决定会产生更好的结果。此外，书中附有不少的设计插图，便于读者理解，不错。是一本值得收

藏的好书！
10、弗朗西斯·培根在《学术的进展》里提到：将对一门艺术的领悟联系并应用到另一门艺术中，历经若干次这样的经历而有所悟，脑海里自然就孕育出了新思想。换言之，在各种领域里，无论是艺术设计还是工程设计，是存在着对于设计流程、设计方式都广泛适用的属性和经验。故此，我们也应该能从各个领域和行业里得出一种进行设计的理想模型。借鉴于悠久的建筑历史，Brooks在《The Design of Design》勾勒出工程师进行思维设计的一种理性模型。在这个理想的思维模型之下，进行设计首要的是确定目标，比如要建造一间可以用来做什么的房子，要开发一个用于做什么的工程项目。然后是必要条件：目标工程需要具备什么样的特性，如房子的大小、扛震能力，项目有什么样的功能等。效用：对于每个要达成的特性，无可避免地存在着类似”边际价值递减“的状况，故此要确定我们需要足够好的东西，但何谓之足够好？房子有窗户是必要的，但是需要多大？太小了满足不了要求，太大显现是浪费，那么这个中间点在哪里？同样地，在各个领域的设计和执行里面，包括”数量“，”速度“等相关的效用，需要确定如何才算足够。约束：约束的考虑在于资源本身的限制，根据成本来确定执行的目标。尽管在某些地方可以设计出非常好的东西，但超出资源的情况，这种设计是无意义的。决策树：在明确各种目标和条件之后，对每一项设计进行判读和决策，最终形成一个要怎么做，不要怎么做的设计方案。　这就是Brooks得出的理想模型。一言以蔽之，总结一下这个理想模型，其实就是确立目标，明确达成目标所需要具备的条件已经受到的限制和约束，在这些状况之下进行一个做什么不做的什么的分析和决策。那么，这就是一个行之有效的设计过程吗？在回答这个问题之前，先来具体化一下我们的场景：作为一名执行人员（或者管理人员），你被任命带领一个团队去实施客户的需求，这个客户希望你的团队来帮他建造一座房子，并且给出了房子的大小以及房子的特性（比如要有多少个窗户，多少个灯，多少间房子等），也告诉了你他可以承担的支出，你据此跟你的团队进行设计，确定房子要怎么布局等各种分析和决策问题。但是问题出现了，当客户看到你建造出来的房子时发现并不是他想要的那种，可能是他没有表达好，也可能是你的团队没有理解好，沟通与理解上的差异使得实际与需求并不符合。客户在看到实物之后才发现自己需要的不是这样的，极有可能是客户站在你做好的窗户面前，才觉得他不需这样的窗户，即便他之前是已经明确告诉你他需要有一个窗户在这里，在初始阶段其实客户也还没弄清楚自己真正需要一个什么样的东西。设计中最困难的部分是决定要设计什么。客户的要求改变了，他不需要这样的窗户，要求你换一下。于是你发现更大的问题出现，因为窗户的变更使得原来的设计不适用，这可能不单单是一个窗户的问题，包括这个房子的其他结构都需要改变，原本的设计需要重新推倒。你可能还会更加悲观地发现，这个需求的变更，使得之前的约束条件也发生变法，不但使得设计需要变更从而需要更多的时间去实施，花费也大大超出了预算。在理想的模型之下，我们要做的东西是很明确的，但在实际上，变化是经常发生的；约束条件是事先知道的，但真实的情况是我们不可能考虑到所有的情况，尤其是约束条件也可能在某个不确定的情况发生变化（比如政府突然出台政策使得地价更高，超过了客户的预算。或者某种地域的原因，对建筑面积有限制）。也就是，理想的情况是没有变化，而变化却是实际的情况。变化带来的一个问题是需要做的东西更多，因此需要的时间变长了，这可以通过增加人力来解决吗？正如《人月神话》里所讨论的，如果不能很好地分割成各个小的目标，增加人员于事无补。事实上，由于需要花费时间向新加入来的人员进行沟通传达，加上新人也会因为不熟悉而产生问题，简单地增加人员效果是恰得其反。变化带来的另一个问题是需要推翻之前所做的，重新进行设计和决策。对既定目标进行全面的设计，并在之后完善的执行这个设计，只在很小的项目或者那些已经做过很多次、具备足够经验的项目上面才是可行的。为了能够应对变化，就需要放弃企图事先能够做出完善设计的想法。螺旋模型相对于瀑布模型（类型理想模型），对于需求和设计，采取的是渐进式的方法，在不断地迭代之下去推进项目的进程。敏捷的思想，同样是不进行前期的全面设计，采用快速的去构建的方式来处理，一但有变，马上变更，而不固守旧设计（或者是根本没有旧设计这么一回事）。理想模型的困境是在变化发生的情况下，需要付出巨大的成本来去解决。而对理想模型进行改变的螺旋模型以及敏捷的思想，都是一些企图最小化变法带来的问题。而之所有会有不断的变化，更为根本的问题，是在于没能完善地确定目标，在实施之前，需求方其实并不清晰自己的所需，以及对各种条件的把握不足，故此也就不能掌控约束。重新回到培根的话，我们能否借鉴和学习各种范本来确切地知道当我们需要达成某个目标的时候具体需要做的是什么？我们能否借鉴和学习各种范本，在这些经验之下切确地知道面临的各种东西，得到完善的设计方案？这种Design of Design能够找到吗？它会是银弹吗？
11、设计师的首要任务乃是帮助客户发现他们想要的设计。我们确实就是这样工作的，本书如一股清

泉，从山间趟过，冲走泥沙，留下了关于设计及其过程的真实描述:目标不是固定在墙上的飞镖靶，需要我们在设计过程中不断明晰。一个好的工程模型需要对此提供自然的解决方法。
12、1.一些复杂句因为太长，得看好几遍才能找出主谓宾来，看起来费劲。2.翻译太生硬，多数的句子就是直接按照英语那种句式结构直译过来，看了几天，感觉就是个半洋文书。
13、看这翻译水平，联想到译者目的，基本上就是为了在外面说起来，自己翻译过书，完全把这本书当回事，不把读者当回事，胡乱翻译。不知道这本书的编辑是干什么吃的摘抄几句：A style consistently applied is a component, even if only the “ dress, ” of the conceptual integrity of a product.一致的风格就是组件，即便产品的概念完整性只是个“ 裙子 ”而已。作者在书中一再强调设计的一致性，我们看看翻译是怎么保持一致性的：第13章 Exemplars in Design设计中的范本The Roles of Exemplars范例的角色Collection Of Exemplars范例集合Studying Design Rationales Of Exemplars学习范本的设计原理妈蛋，这组队翻译，难不成是一人翻译一段啊！
14、很少有人像Frederick P. Brooks, Jr. 一样对软件开发的实践（而不是学术理论）产生那么大的影响。他在《人月神话》中记录了他作为IBM System/360计算机以及Operating System/360项目领导者的经验，这些经验不断地促使我们形成项目管理的观念。很难找到比他的“ 没有银弹：软件工程中的根本问题和次要问题 ”（《Information Processing 1986, Proceedings of the IFIPS Tenth World Computer Conference》，H.-J. Kugler编辑。Amsterdam: Elsevier Science, 1069-1076）一文被引用更多的文章。“ 银弹 ”的概念代表了对困难的软件和编程问题的某种近乎神奇的解决方案，这是我们的词汇表中不可缺少的一个词。Brooks的这本新书《设计原本》拓展了他以前的思想，添加了对设计的本质和重要性的新领悟。毫无疑问，这本书将会成为所有专业开发者书架上必备的书籍。这本书的副书名是“ 计算机科学巨匠Frederick P. Brooks的思考 ”。书中包含了可以作为独立的文章进行阅读的20章，每一章与其他章的耦合都比较松散。这是好事，因为大多数读者都希望阅读本书的每一章（不一定要按顺序），然后在继续下一章阅读之前思考一下所讲的内容。除了这些文章外，这本书还提供了8个案例，涉及的范围从海滩小屋的设计到IBM System/360的架构。这些案例阐释了本书中的一些重要概念。什么是设计？Dorothy Sayers（英国作家和戏剧家，Brooks引用了他的话）说设计有3个阶段：概念构造的形成，在真实媒质上的实现，与真正用户的交互。Brooks在他的“ 银弹 ”一文中指出，第一个阶段（即概念构造）是软件工程最困难的阶段。但在1986年时，“ 概念构造 ”似乎更侧重计算机在执行指令时内部发生了什么。在这本新书中，关注的重点更多地转移到架构、外观，以及程序工件与环境的交互上。有趣的是，Brooks在第1章的开始引用了培根的话：（新思想来自于）将一门艺术中的领悟联系并应用到另一门艺术中，历经若干次这样的经历而有所悟，脑海里自然就孕育出了“ 新思想 ”。这清晰地表达了跨学科的灵感和锻炼。Brooks没有深入这个概念，即使在后面的内容中讨论如何“ 创造 ”了不起的设计师时，他也没这样做。有几章讨论了设计和设计过程的模型。Brooks在这里严厉批评了流行的“ 理性主义者 ”设计模型（读者可能最熟悉的就是“ 瀑布方法 ”），并断言“ 理性模型过于简单 ”。书中指出了理性模型的诸多缺陷，包括“ 对理性模型的最具破坏性的批评可能是，最有经验的设计完全不是这样工作的 ”。（在本书中提到David Parnas多次，但没有提到他的著名文章“ The Rational Design Process: How and why to fake it ”，这篇文章也对理性模型提出了严厉批评。）本书对其他一些设计模型也进行了讨论，包括：· 迭代演进式开发，与此最接近的是Brooks对敏捷方法的讨论。· Boehm的螺旋模型。· 开源的、Raymond的“ 集市模型 ”。这些讨论的结论是：设计需要某种过程以及该过程的模型（主要是为了沟通并支持协作），Barry Boehm的螺旋模型是Brooks认为最有希望的模型。《设计原本》中的其他章节关注了以下几个问题：· 协作与团队设计，包括远程协作所引发的问题。· 关于“ 理性主义 ”与“ 经验主义 ”的讨论，Brooks自认是一个经验主义者。· 约束条件的价值。其中印证了许多“ 设计 ”文献，这些文献来自工业、产品和图形设计师。设计草案（用于启动设计过程的文档）必须足够模糊，允许自由思考和表达，但必须清楚定义所有的约束条件。约束条件勾画出边界，创造性的、有想像力的、创新的设计将在这个边界之内发生。· 讨论美与风格。· 一些关于设计技术和实践的讨论，Brooks发现它们是有价值的。· 需求、罪过与合约。· 一个案例，讲述了为什么任务控制语言（Job Control Language, JCL，如果你没有在大主机上工作的愉快经历的话）可能是“ 有史以来最糟糕的编程语言 ”！本书末尾用两章的篇幅讨论卓越的设计和卓越的设计师的问题。Brooks果断指出，卓越的设计来自卓越的设计师，而非卓越的设计过程。Brooks说，我们教育和培训软件专业人员的方式无助于培养出卓越的设计师。他指出，培养卓越的设计师的一个主要障碍就是缺少持续的实践和批评。InfoQ有机会对Brooks提出了一些跟进问题，本文包括了这次访谈的内容。问题与回答如下：

开发项目有一个生命周期，要么是经典的线性瀑布式，要么是迭代增量式（敏捷）。设计是在生命周期特定阶段发生的事情，还是分布在所有阶段？它集中发生在迭代开发的前面几次循环，但有时候发生在所有迭代中。如果设计发生在所有的开发阶段中，是否在每个阶段中具有不同的形式、实质或要点？当然是这样的！在第一次迭代中，总体架构是中心问题。在接下来的迭代中，设计工作集中于更精细的层面，除非是在满足最后期限之后的回溯，或者人们意识到需求的改变或新的机会。约束条件在设计过程中扮演什么角色？（本问题的背景知识：其他领域的设计师常常依赖一份“草案”，他们预期/需要草案中有模糊之处，以便能够自由地“设计”，但他们需要清楚表达约束条件，这些约束条件定义了目标区域，最优设计只能在这个区域中产生。）它们既促成了整体架构，又在较小程度上促成了细节设计。是否有明显可以确定的设计“错误”，它们是否能在犯下时就可以意识到？（或者，这样的错误是否像代码中的缺陷，通常需要在犯下之后许久才发现？）大错是在开始时犯下的，而且很少意识到。如果最终被发现，通常是在现场实施之后。较小的错误是在编码开始时出现，或者是在第一个真正的用户测试原型时被发现。大多数设计师认为他们的活动是高度协作的，至少是客户与设计师之间的协作，但设计更多时候涉及一个团队。您是否同意设计是一种协作？如果是这样，设计团队的地理分布或临时分布会带来什么影响？（显然，在今天的离岸外包环境中寻求并行设计，以应对软件开发的一般挑战。）我用了两章讨论协作和远程协作。是的，今天大多数的设计都涉及团队。通用产品的设计不涉及与客户的协作，如iPad。对于为一个客户设计定制的产品，我非常喜欢对原型和代用品（如建筑的虚拟环境模型）尽早地、激烈地、频繁地、不断地进行用户测试。但是，我不认为这是与用户和客户进行协作设计。您是否有一份清单，例如6点建议，可以总结您的书向设计者提供的最重要的经验？1）专心研究以前设计者的工作，看看他们如何解决问题。2）尝试弄明白他们为什么做出那样的设计决定，这是对你自己最有启发性的问题。3）仔细研究以前设计者的风格。最好的方式是尝试用他们的一些风格勾画设计草图。4）保存一本“草图本”，将您的想法、设计和局部设计记录下来，不论使用何种媒质。5）在开始设计时，写下您对用户和使用方式的假定。6）设计、设计、设计！在您的“银弹”文章中，您谈到了“概念构建”和人类在头脑中完成这项工作时遇到的巨大困难。您觉得在这本书中一些思想是否关注并解决了概念构造这一基本困难？肯定关注了，肯定没解决。在第3章中，您漂亮地批评了理性设计过程，特别是瀑布式方法所包含的理性设计思想，并指出这种思想是有害的，必须抛弃。您对这种有害模型的持续存在有何看法？是否人们就是很倔强？或者尽管开发者更了解，但管理层会犯错？是否有一些文化上的偏见（尤其是西方文化上的偏见）阻碍人们抛弃瀑布模型？第4章讨论了软件工程中的瀑布模型的持续存在（在其他学科中并不常见）是因为设计者过早期望得到有约束力的合同和确定的需求。在第20章中，您批评了我们的教育系统（温和，但确是事实），并建议设计者需要参与“批评性实践”。Richard Gabriel长期以来一直主张计算机科学/软件工程大纲应该采用他在取得诗歌硕士学位时一样的大纲（他在很久之前获得了计算机科学博士学位）：大量的练习（每天至少一首诗）、大量的批评（来自同学和导师）、勤奋学习大师和大师的诗歌、不断自省、周期性的反省。这似乎与您的建议相似，那么您是否主张大学提供一个“软件好艺术硕士”学位？不。熊恩在《the Education of the Reflective Practitioner》一书中提出了同样的建议，还有例子，更适合设计。所有的工程学大纲都应该强调这种方式。哪些其他领域的研究将有助于毕业生变成卓越的软件设计师？1）算法和数据结构是最重要的基础课程。2）计算机硬件架构。3）应用领域，特别是商业数据处理、数据库技术和数据挖掘。4）心理学，特别是知觉心理学，因为用户是最重要的。理性设计过程，实际上是所有计算机科学和软件工程，有意识地采用了宇宙的基本模型（20世纪的物理学），即确定性的、机械的、理性的宇宙。如果那就是自然或现实，那么理性的、搜索树式的设计过程模型就很有意义。如果宇宙实际上是复杂的适应性系统，那么理性模型就会失效。您是否走得更远，奠定了复杂系统的设计或修改过程的基础，如文化或商务企业？我不会自大到建议这样的东西。IDEO是一家非常有名的设计公司，它的总裁Tom Kelly写了一些关于设计、设计过程和设计思考的书。他最好的一本书是《Ten Faces of Innovation》，其中他确定了每个设计团队需要的10个角色（人类学家、实验员、嫁接能手、跨栏运动员、合作者、指导者、用户体验设计师、布景师、专业护理人员和讲故事的人）。如果您知道这本书，是否类似的角色应该出现在软件设计团队中？怎样做到？我不了解这本书。听起来有趣，而且有用。您提到了Christopher Alexander和他的影响并在注释中提到了《The Synthesis of Form and A Pattern Language》一书。那代表了“理性的Alexander”，但“Timeless Way of Building”和他的杰作“Opus, Nature of Order”却更为“神秘化”。“神秘的Alexander”是否对您的设计和设计过程思想有所影响？我受到了《The Timeless Way of Building》一书的影响。软件领域

的设计将大多数注意力放在人工制品上，即执行程序的计算机。越来越多的实践（但学术界还没开始）开始关注“用户体验设计”，即计算机所处的系统和生态环境。对于用户体验设计者如何从本书中获益，您是否有一些建议？我觉得前面给出的建议同样适用于用户体验设计，但这个领域与软件工程领域相比，自由式教育更为重要。您能列举出值得所有人学习的3名设计大师（任何领域）和3名软件设计大师，4至5个设计杰作，并简单说明为什么吗？·巴赫，作曲家·伦勃朗，画家·Seymour Cray，超级计算机设计师·Christopher Wren，建筑，特别是他在伦敦设计的教堂·Nicholas Wirth，计算机语言·Donald Knuth，算法我不认为所有的人都应该学习同样的例子。人们需要接受范例的设计原则方面的基本教育，这样才能深入研究一个范例，欣赏这些设计问题和解决方案。但是，即使是门外汉也能欣赏一致性和设计概念的统一性。原文网址
：http://www.infoq.com/articles/brooks-design-book-review
15、计算机大师心中的建筑设计情怀Frederick P.Brooks 在计算机领域可谓是无人不晓。他因领导开发IBM System/360计算机系列以及操作系统而荣获美国国家技术奖，因为对其计算机体系结构研究作出的里程碑式的贡献而获得了A.M.图灵奖。布鲁克斯在他60多年的职业生涯中涉及了计算机架构、软件、房屋、图书和组织机构五个领域的工作，而且作者发现这些领域里，设计过程都极为相似，比如思维过程、人与人的交互、迭代、约束条件和劳动等。而他在IBM公司100年之际推出了他的新作《设计原本》，也就是要让人们反思这些隐藏在这些设计活动背后不变的设计过程。在建筑设计方面，布鲁克斯设计过程的演进，探讨了协作和分布式设计，阐明了哪些条件造就了真正的卓越设计者。他解释了设计过程的具体细节，包括多种预算约束条件、美学考虑、设计经验主义及工具，同时他将这些讨论与现实中的案例结合起来，这些案例从房屋建造到计算机的系统设计。一本品位独特的图书。
16、本书中国互动出版网china-pub正在低价火热预订中，网址：http://www.china-pub.com/197442。《设计原本:计算机科学巨匠frederick p. brooks的思考》对设计过程进行深入分析，揭示进行有效和优雅设计的方法，包含了多个行业设计者的特别领悟。作者精确发现了所有设计项目中内在的不变因素，揭示了进行优秀设计的过程和模式。通过与几十位优秀设计者的对话，以及他自己在几个设计领域的经验，指出，大胆的设计决定会产生更好的结果。作者追踪了设计过程的演进，探讨了协作和分布式设计，阐明了哪些条件造就了真正卓越的设计者。他探讨了设计过程的具体细节，包括多种预算约束条件、美学考虑、设计经验主义及工具。同时，他将这些讨论与现实中的案例结合起来，这些案例从房屋建造到ibm的operating system/360。成功的关键因素贯穿全书，每个设计者、设计项目经理和设计研究者都应该知道。
17、第8章71页正文第一段最后一句：应该是“理性主义者认为可以；经验主义者则认为不行”翻译把理性主义和经验主义翻译反了原文Rationalists believe I can; empiricists believe I cannot.
18、标题的这句话来自于本书，这让我想起了爱因斯坦的那句话：天才是99%的汗水+1%的天分。没有1%的天分，再多的汗水也无助于你成为天才。Brooks作为项目经理，对于真正的人才是非常渴求的。本书难得的地方在于：对于一个很宽泛的主题（设计的设计）能够说出不少有价值的话题，这一点我个人感觉是很难的。越是宽泛，形而上的主题越是没什么什么实际的意义。而作者居然能写这么多内容，可见其自身经历确实相当丰富。本书的论题着眼于人文上的设计，作者不会教你如何设计一个类或是一个框架。而是一种非常大局观的设计原则，设计实践---如何在一个组织（公司，或项目组）里进行最佳的设计实践。经验主义的理念贯穿全书。书的开始部分讨论了理性主义和经验主义，无非是瀑布模型和迭代（敏捷）模型的优劣对比。作为经验主义者的作者还是以理性主义的思维对两者进行了一个对比。接下来对于设计的约束条件花了不少笔墨，这个约束不光是设计自身的（技术上的）还有一个很大的因素来自于公司政治，在跨国公司工作过的想必都有感触了。当约束发生改变的时候重新审视设计是一个极其重要的设计要素，这一点也是敏捷非常看重的review。最后部分花在了项目的管理上：如何找到一个卓越的设计师并保护他让他对于项目产出应有的价值。Brooks在这一点上是很到位的，因为只有设计师才是项目的实际构筑人。成功的项目一定依赖于设计师的优秀工作，而让这成为可能的则需要项目经理的优秀工作。以上都来源于我看书的记忆，只着重了我个人印象比较深的内容。书里有一些计算机架构相关的内容，虽然作者尽力让这些内容显得易懂，但不得不说即便对于计算机工程师，有些东西也稍显遥远了点。
19、翻译的很有趣，很文学，有一两句给我印象较深：（1）诚哉斯言（2）殊为不幸这种用词，在计算机科学类图书中很少见，可见译者的苦心。－－－－－－－－－－－－－－－－－－－－－系统提示：抱歉，你的评论太短了评论本就三言两语，何必强求以影响用户体验？如果你真的需要这种限制，可用

较醒目的方式事前提示。－－－－－－－－－－－－－－－－－－－－－－
20、豆瓣上原来也有很多五毛，马勒隔壁的！好评一大堆，你大爷的，你看过这本书没？Fuc k!译者他大爷的就一个啥X，翻译的太差劲，连个最基本的语句都不会组织。这书看得人头疼，几乎每个句子都要重新组织才能明白。给你的评价就是比谷歌翻译强点!你看看<启示录，打造用户喜爱的产品>人家怎么翻译的？垃圾翻译！
21、目标（核心需求）必要条件（其他需求）和主要目标相关的是一组必要条件或者说是次要目的效用函数（需求之间的制约）就是说满足a条件满足到一定程度的时候，会抑制b条件，比如：如果增加窗户的面积，那么冬季的保暖性必然受到影响，如果加了窗户的厚度和层数那么预算会增加约束（限制）就是限制啦，比如：北方的房屋需要考虑保暖和防风，南方则需要防潮、资源分配、预算和关键预算（资源限制）这个就是要求你抓重点，抛起一些不核心或者性价比不高不那么必须的需求设计树（最后的具体的决策）这个是作者提出来的一个观点，他认为按照理性模型的思路，设计师们可以形成设计却测，然后在之后的设计空间中，又形成另一个决策后记：这本书看完了，最后觉得这本书主要的思想，或者核心的东西，其实是在讲设计前要考虑的一些东西：1、核心需求2、其他需求3、需求之间有约束和限制——不能同时满足或者同时达到最大值4、设计本身的限制5、资源的限制——对于一个复杂的设计而言（例如：房屋装修的设计，或者房子本身的设计），由于①需求多 ②需求之间的相互制约 ③限制多... ...所以比较好的方式是真实的去体验用户目前的解决方法和使用情景和环境，根据在试用的过程中的用例，来考虑核心需求、需求、限制，并将这些item列出来以保证无遗漏和综合考虑，最后再来设计就如书中的设计任务书——设计任务书是一个文档，需要与客户共同完成，他规定了目标、必要条件&约束。这样才能围绕约束来满足需求，而非靠臆想或者其他卓越的设计在于过程，而不在于人主要观点：产品过程在实质上会阻碍卓越产品的诞生：——因为每个专家拿工资是为了避免错误，而非创造卓越的产品；→。→这导致大家偏向于找理由不前进，这样即使新产品没有被否决，也会被抹去棱角强制妥协产品过程的必要性在于：低级的可以因此迈向中级的产品过程的重复循环是：产品定义、市场预测、成本预估、价格预估卓越的设计过程的特征：①明确基本重点&约束条件 ②简单轻便的例外机制=。=给设计师充分自由：①、不要越俎代庖 ②尊重设计师的专业性 ③不要分散精力 ④资源支持卓越的设计师从哪里来？好的培训体制：多实践、少阅读招募：经理常常招聘一些新的设计师，然后下意识地以精力自己的额工作为标准对他们进行评判：“ 他能做好我现在的工作么？ ”这样会倾向于能说会道的领导者以及善于开会的人。同事忽略内向的、说话慢的、尤其是不合常规的人。但... ...优秀的设计师也可能来自这些人吧在项目早期就要给出目标的显式陈述、相关的必要条件以及约束说明，这有助于避免让团队陷入举棋不定的局面应用需求追踪矩阵以确保每一个被精化、定义和列出的需求都的确是从一个或者多个出事的总体需求中派生出来的——确保它不是从某个用户代表的要求或者是设计师的愿望出发而被悄悄混入的需求经验丰富的设计者经常一开始就将他们对于用户的了解、使用目的以及使用模式等信息精确滴记录再按。聪明的设计者还会降对用户不了解以及假设的地方明确地记录下来。每个设计决策都是以设计者与用户的假设作为指导的，无论一致与否都是如此。这事实上意味着，含糊的设计者会吧他自己当做用户，按照自己的想法去设计，就好像他就是用户一样，但他不是用例模型是用例的集合并且带有权重约束是我们的好朋友——通用产品比特定用途的产品更难以设计由于设计者往往不是用户，因此会产生一些问题，比如其实不了解用户需求神马的，这时候的参考解决办法：①用户场景体验：用户体验的模拟或者类似场景的体验都可以，去感受其中的不便和需求等等 ②：从一开始就和用户保持密切接触的最佳方式，先构建一个能工作的最小功能版本，然后让用户使用，或者至少以测试来驱动开发 ③设计者需要身故到实际的体验过程和实现过程中 ④、需要有一门课程，让设计者更好的理解用户的需求和期望，并进行一定实践

《设计原本》

章节试读

1、《设计原本》的笔记-第63页

同一地点的团队工作中，非正式沟通渠道的巨大重要性沟通的艺术，不是会议、电话和邮件所能达到和替代的。

2、《设计原本》的笔记-第17页

设计中最困难的部分在于决定要设计什么。

3、《设计原本》的笔记-第3页

(新思想来自于)将对一门艺术的领悟联系并应用到另一门艺术中，历经若干次这样的经历而有所悟，脑海里自然就孕育出了(新思想)

4、《设计原本》的笔记-第31页

若是以为用户方式指定需求，不免由它的本性所驱使，有着将产品研发过度的趋势尤其是在缺乏一线技术人员的委员会中。

5、《设计原本》的笔记-第17页

设计中最困难的部分在于决定要设计什么。

6、《设计原本》的笔记-第29页

Chapter 4 Requirements, Contracts and Sins
Any attempt to formulate all possible requirements at the start of a project will fail and would cause considerable delays.
-- Pahl and Beitz [2007], engineering design
这句开篇的话说的极是。任何的尝试都会导致2个潜在的可能性：成功完成和延迟乃至难产死亡。做好心理准备之外，还要做好应急方案。
Boehm的 螺旋模型 （v.s.）瀑布模型

7、《设计原本》的笔记-第37页

共同演化模型：
共同演化模型进行设计并不是先把问题固定下来，再寻找一个满意的解决办法。而是根据现有的问题寻找解决方案，再找出新的问题跟新的解决方案，问题和解决方案是共同演化的。强调对需求进行递进式探索和构造。
缺点：处于原始的构造形态中，并且其几何上也没有收敛的过程。

集市模型
类似于linux社区，由一群志愿者，不断的发现新需求和新问题；对新需求进行开发，对新问题进行完善；不断的添加新功能，完善问题；这样集成化越来越高。
缺点：只有在特定的条件下才能进行。

螺旋模型

将同一活动的连续和反复关联起来，构建一个原型不添加新功能，在此基础上对原来的功能不断的进行完善和改进。
缺点：没法作为合同达成的基础

8、《设计原本》的笔记-第18页

快速原型是一种进行精确的需求配置的必要工具。在设计师无力独自构建原型之处，工程师就成为设计过程的一部分。

9、《设计原本》的笔记-第11页

很困，但是趁着清早看两眼。

这一章（工程是怎样进行设计思维 -- 理性模型）只能尝试着去从概括性的语言去读了，专业的一概不懂。
……因为设计师的理论即一般的搜索理论……对象是巨大的组合空间

… [F]or the theory of design is that general theory of search … through large combinatorial spaces.
-- HerBert Simon [1969], The sciences of The arTificiaL, 54
模型概览包括（21章）：目的、必要条件、效用函数、约束、资源分配和预算、设计树。
The Model : Goal. Desiderata. Utility Function. Constraints.
闹钟的设计树（部分）软件开发的瀑布模型（spiral model of software），个人觉得“瀑布模型”这个译法有点问题

10、《设计原本》的笔记-第33页

有个悖论，想精确界定合同总价，必须清晰明确目标需求；而同时，在订立合同的开始阶段没有迭代几轮的情形下，很难明确需求。
建筑设计过程这样解决：
先请好的设计师，按人月charge，购买服务先搞方案；
设计师完成概念设计，并在预算、工期及可实现性方面得到利益相关方的确认，形成首个原型设计。
经过几轮迭代，形成教详细方案，3D等，形成规格说明书。
依据规格，形成固定价格合同。

11、《设计原本》的笔记-第31页

在IBM OS/360的研发过程中，其需求是由来自市场部的一个大型委员会初步定稿的。身为项目经理，我不得不将这份需求文档断然拒绝，因为它完全不切实际，并且用以获取项目实质的、由架构师和市场一线人员以及实现人员组成的工作组的体积过小。
需求需要由产品、市场、设计、开发共同完成。这也是我为什么一直试图让开发者尽早参加产品设计初期的原因。不仅可以提高开发者对产品的认可度，还可提高产品需求和设计的质量。

12、《设计原本》的笔记-第25页

瀑布模型是错误的、有害的，我们必须发展并摈弃之。
幸好我很早就认识到了这一点 :)

13、《设计原本》的笔记-第4页

《设计原本》

第一章晚餐的时候读完了第1章 设计之命题

p.4 对大多数的创作者来说，构思的不完整性和不一致性只有到了实现的时候才变得明显起来。因此，记录、实验、和 ”解决“ 成为了理论家们得关键原则。
For most human makers of things, the incompletenesses and inconsistencies of our ideas become clear only during implementation. Thus it is that writing, experimentation, “ working out, ” are essential disciplines for the theoretician.
构想、实现和交互这三个阶段的操作时循环进行的。... ...
The phases of Idea, Implementation, and Interaction operate recursively.

14、《设计原本》的笔记-第18页

设计师的主要任务乃是帮助客户发现他们想要的设计。

15、《设计原本》的笔记-第7页

只看了序就受不了了！！
好烂好拗口的翻译啊。。。。无数的括号算什么意思啊！！
要知道能看懂和能翻译之间还有很长的距离啊！！
怪不得英文版比中文版贵了。。。。

16、《设计原本》的笔记-第5页

就

17、《设计原本》的笔记-第19页

研习设计史的最有力的原因是去了解怎么样的设计方案是行不通的以及为什么这些设计方案行不通。
所以需要大量阅读失败设计。

18、《设计原本》的笔记-第6页

良好的设计具有概念性的完整性-统一、经济、清晰。

19、《设计原本》的笔记-第49页

由于技术复杂性的爆炸式增长和time to market周期的不断缩短，在大规模工业化背景下，使得产品设计逐步向团队协作转变。
而协作设计最大的挑战在于概念的完整性，这就是为什么伟大的作品往往是个人或pair完成的。
如何保证概念完整？
必须是民主与专制的辩证统一：
民主（协作）：需求获取、概念探索（头脑风暴、设计竞赛）、设计评审
专制（专职）：概念设计、系统架构、使用交互体验（如UI）
而且这个人必须贯穿以上整个过程并对最终结果负责。人们看到这种模式（物理隔离、小团队、注意力高度集中和一人领导）反复出现在真正有创意的产品设计中。正如诸如神偷电影＜＜十三罗汉＞＞一样

20、《设计原本》的笔记-第29页

理性模型设计，是根据需求文档进行的，需求文档的定义使得约束不变、各种必要条件极其权重不变；实际情况是设计需求文档的人凭空想象提出不切实际的需求或者需求定义的不是那么清楚，设计者根据这些需求，按照需求提出人权重对需求进行评估，对设计约束条件进行定义，结果设计出的东西实用性并不是很高。造成这样的结果往往是因为骄傲、贪婪及惰性造成的。
实际中签订合同，目的是在早期对设计的目标、需求和约束进行定义，这也是合同的要素。但是在组织内部或者组织与组织之间里所有的人都清楚，这些约束条件在实际当中会发生变化，这就为不良的行为打开了方便之门，现在合同上让一步等需求发生变化时再抬价。这种签订合同的要素也就导致了瀑布模型在设计复杂系统时那么长久。

21、《设计原本》的笔记-第35页

今年之前一直没有弄清楚一个问题，在建筑设计或者装修布局的时候，到底是怎么样的一个过程，是不是那种准确无误的，已经考虑了所有细节的设计；想要给任何复杂系统通过指定的方式完成一组完备的、精确的需求实质上是不可能的，除非是通过设计过程中的迭代式交互才有可能完成，这才是实际发生的事情。一直假设在设计的时候已经考虑了一个桌椅组合放置到哪个具体的位置，只有那个位置或者某个区间附近才是满足各种需求的，最舒服，但是实际上不可能实现的，就算是有那个桌椅组合的各种不同的人的使用统计，估计也并不可能存在这样一个区间；好像应该是先大致放在一个经验上或者习惯的位置，然后开始按照系统化的流程和步骤，直到接近可以看到原型的时候或者可以看到原先设计和进行到的步骤矛盾的时候，再重新迭代（当然是限量版桌椅卖完了，收购太贵超出成本也不是奇怪的事情），进行调整才是Brooks表达的精神。
在需求获取中，更大的问题其实应该已经超越了软件工程原有的边界，是任何类型的系统构建中都有的问题，沟通和博弈的问题。看到P30的愿望列表的部分，马上想到的居然是一幅博弈论和权力斗争的图景，一部漫画Liar Game很好的描述了这样的场景，利益相关者的博弈和权衡取舍，各自利益的斗争，也难怪有些软件系统后门多，客户本身就有这样的需求，虽然客户不一定是最终使用者。
罪念：骄傲、贪婪以及惰性。这些阻碍沟通和信任建立的因素也是，很多时候理想的系统的确是非常值得期待的，当时可以必须屈服于现实

22、《设计原本》的笔记-第19页

那么，设计师该怎么做？估算！理所当然，正式的也好，非正式的也罢，都要做估算。在求精的步骤中，人们必须对设计树进行剪枝。
估算是设计师基本能力之一。

23、《设计原本》的笔记-第3页

什么是设计：概念性构想的形成；真实媒体中的实现；真实体验中与用户交互；
设计的概念：良好的设计具有概念性的完整性，统一、经济、清晰。
设计的类别：分为系统设计与艺术设计，本书主要讲述系统设计。

24、《设计原本》的笔记-第31页

需求评估必须在核心成员间展开，由架构师、市场一线人员和核心实现人员组成。决策需要民主集中。

任命项目经历很重要，人选需要手腕强硬、经验丰富、领域知识到位。并授权以其认为必要的方式量

身定制标准流程和步骤。

25、《设计原本》的笔记-第48页

许多人手会让工作变得轻松——通常如此。
但许多人手会让工作变得更多——总是如此

26、《设计原本》的笔记-第18页

设计师的主要任务乃是帮助客户发现他们想要的设计。

27、《设计原本》的笔记-第15页

瀑布流不是最优的原因
原来如此,用户根本就不知道自己真正的需求.

28、《设计原本》的笔记-第25页

瀑布模型是错误的，有害的，我们必须发展并摒弃之。

29、《设计原本》的笔记-第47页

现代农民？？？靠

30、《设计原本》的笔记-第52页

竞争是另一种协作。——正如说互斥是特殊的同步一样

31、《设计原本》的笔记-第1页

设计有三个阶段：概念构造的形成，在真实媒介上的实现，与真正用户的交互。

32、《设计原本》的笔记-第7页

现代软件构建 拆分成 构思 （essence）和实现 （accident）

由于构思存在不完整性和不一致性 实现时会暴露出来

Idea Energy Interaction 是为「胸有成竹」的三个阶段

嗯 很好理解

事实上 「设计概念」在设计活动开始前就已经成型了

33、《设计原本》的笔记-第7页

向设计者提供的最重要的经验
1、专心研究以前设计者的工作，看看他们如果解决问题。
2、尝试弄明白他们为什么做出那样的设计决定，这是对自己最有启发性的问题。
3、仔细研究以前设计者的风格，最好的方式是尝试用他们的一些风格勾画设计草图。
4、保存一本“草图本”，将想法、设计和局部设计记录下来，不论使用何种媒质。
5、在开始设计时，写下对用户和使用方式的假定。
6、设计、设计、设计！

34、《设计原本》的笔记-第21页

将约束明确列出，是把丑话说在前面，这就可以避免日后突然爆发令人不快的局面。这同时也是在设计师的脑海中烙下对于这些约束的印象，从根本上提高某一约束消失时被设计师发现的可能性。后者更重要。

35、《设计原本》的笔记-第1页

[New ideas would come about] by a connexion and transferring of the observations of one Arte, to the uses of another, when the experience of several misteries shall fall under consideration of one mans minde.
-- Sir FranciS Bacon [1605], The Two Books of The Proficience and advancemenT of Learning, Book 2, 10

（新思想来自于）将对一门艺术的领悟联系并应用到另一门艺术中，历经若干次这样的经历而有所悟，脑海里自然就孕育出了（新思想）。

Few engineers and composers ... can carry on a mutually rewarding conversation about the content of the other’s professional work. What I am suggesting is that they can carry on such a conversation about design, . . . [and then] begin to share their experiences of the creative professional design process.
-- HerBert Simon [1969], The sciences of The arTificiaL, 82

很少有工程师和创作者……能够通过探讨对方的专业领域而互有所得。我的建议是，他们可以相互探讨设计……（然后）彼此分享在这种转行业的创造性设计过程中的经验。
任何的两种专业都可以发生交互和碰撞。只是欠缺交流的媒介。搜索引擎和问答类的网站在某种程度上起到这个“探讨”的桥梁作用。
Dorothy Sayers, the English writer and dramatist, in her magnificent and thought-provoking book The Mind of the Maker, breaks the creative process out further into three distinct aspects. She calls them the Idea, the Energy (or Implementation), and the Interaction Dorothy Sayers : 1893 至 1957[Dorothy Sayers]
http://movie.douban.com/celebrity/1019682/

36、《设计原本》的笔记-第45页

人类绝大多数的工作都是一个人完成的，或者是两个人紧密合作。

37、《设计原本》的笔记-第1页

作者提到的文章，计算机科学家的使命是制造工具 II（The Computer Scientist as Toolsmith II），中文翻译在这里：
[http://www.newsmth.net/nForum/article/SoftEng/77938]

38、《设计原本》的笔记-第1页

看了你们摘录的一些句子，这本书的翻译好像很糟糕阿－－

39、《设计原本》的笔记-第40页

尽管有关开源过程的“开放”和“自由”的文字早已是铺天盖地，但是整幢 Linux 大厦的建成却很难看做是只砖片瓦的随机堆砌 – Linus Torvalds 始终作为一个保持其概念完整性的关键首脑力量存在。此外，对于 Linux 来说，功能规格说明早已有了：这就是 UNIX。同样重要地，总体系统设计也是现成的。

40、《设计原本》的笔记-第13页

设计的理性模型是设计过程系统结构化。
设计空间可以表达为树形结构，参考mind manager

41、《设计原本》的笔记-第25页

论原型的重要性：我晚饭在肯德基排队到我时还不太确定我的需求到底是什么 一份打印着各种汉堡的单子是多么重要啊

一边设计一边探索

节点不是决策，而是暂定方案 等着组合爆炸吧

必要条件和权重、以及约束持续变化 促使发生迭代

42、《设计原本》的笔记-第30页

隐隐觉得现在已经是「需求在设计开始之前就确定了」

开队是否是种变相的瀑布模型呢？

好在工程师和pm都人少，还看不出臃肿的需求集合

抵制需求膨胀和蠕变

「进度的紧迫性是最后的挡箭牌」

建议：新的技术队的开发 首个和第二个milestone间完成设计初始表达的定义

43、《设计原本》的笔记-第18页

设计师的主要任务乃是帮助客户发现他们想要的设计。
这说明了设计本身是个动态的过程，对于软件更是如此。那么，对于当前大型的软件系统，其开发周期本就非常漫长，当完成开发之时，也正是其开始修改之日。

因此，倒不如从一开始就以修改为目标设计，将变更作为基础需求目标来实现更为合适。

44、《设计原本》的笔记-第48页

许多人手会让工作变得轻松---------通常如此
许多人手绘让工作变得更多---------必然如此

45、《设计原本》的笔记-第13页

设计过程的隐含模型：有序过程的有序模型

深度优先寻找每个节点最棒的方案 否则回溯尝试别的路径
预感、经验、连贯性和审美旨趣引导着每一次的方案选择瀑布模型仍然是设计的理性模型的基础性表述之一
理性模型的作用：

步骤清晰 估点方便
有助于避免团队举棋不定

46、《设计原本》的笔记-第3页

（新思想来自于）将对一门艺术的领悟联系并应用到另一门艺术中，历经若干次这样的经历而有所悟，脑海里自然就孕育出了（新思想）。----培根

通过探访对方的专业领域而互有所得。彼此分享在这种专业的创造性设计过程中的经验。

47、《设计原本》的笔记-第11页

理性模型概览：目标、必要条件、效用函数、约束、资源分配、预算和关键预算、设计树。
理性模型的由来：讲述理性模型的由来，有一定的历史，而设计空间表达为树形结构的概念，是在simon的著作中隐含的提出的。
理性模型的长处：它为设计项目规划提供了清晰的步骤，为日程规划和进度评估定义了明确的阶段里程碑，为项目组织和人员配备指明了方向，改进了设计团队的内部沟通。
缺点：太过简化。

48、《设计原本》的笔记-第51页

懂得系统所有部分的主架构师？

49、《设计原本》的笔记-第17页

设计师只是把理性模型视为一种理想化的东西，它以某种方式描述了我们在设计过程是应该如何运作，但实际过程中这种理想主义很天正，以下是现实与理想之间的区别：
我们在设计阶段并不真正地知道目标是什么：
理性模型的严重缺陷在于，设计师们往往只有一个模糊不清的，不完成的既定目标，或者树是主要的目的。在此情形之下：设计中最困难的部分是在于决定要设计什么。
作者列举他还是学生时为一家兵工厂打工，在设计系统的时候，做一些给客户演示之后再修改一些，最后达成兵工厂所要的样子。使作者明白，他提供的最有用服务是帮助，客户发现什么是他们真正想要的。
设计师的主要任务是帮助客户发现他们想要的设计，现在大多数设计师都是先模拟或者构建出原形，以促成目标的收敛。因此目标的迭代必须作为设计过程的固有组成部分考虑。

我们通常并不知晓设计树的样子——一边设计一边探索：
设计树是很少能够一次性绘制出来，没有任何设计师具有一次性设计的全部知识绘制出该领域的决策树，设计者们都是一边做，一边对设计书进行探索，做出某个决策，然后查看由它启动或否决的备选方案，继而依此做出排在下一个的设计决策。

（设计树上的）节点实际上不是设计决策，而是设计暂定方案：
特定的设计树只是在树形结构中搜索的简化模型，节点处有并列的属性分支，也有备选分支。在设计树的每一个节点处，设计师所要面对的不仅仅是为单独一个设计决策准备的若干简单备选方案，而是为多个设计暂定方案准备备选方案了。并且设计树中的决策排序顺序也是官重大。以此以属性模型结构表示的设计模型，其复杂性带来的组合爆炸是人们思维中的难以承受之重。

有用性函数无法以增量方式求值：
理性模型的假定是吗设计是对于设计树的搜索，并且在每个节点处人们可以对若干下一级分支的有用性函数求值。事实上，除非探索到所有分支的所有页节点的程度，否则人们就很难做到这一点，因为大量的有用性指标对于随后的设计细节有着强烈的依赖。因此，虽然对有用性函数的求值在原则上是可行的，但是在实践上人们就会在这里再次遭遇组合爆炸。
一般都会根据
经验：很多辅助信息都能够促进该过程的直接判断。、
简单估算量：设计是们经常在进行设计树探索的早期就烈性地采用简单估算量。

必要条件极其权重在持续变化：
整体设计是诸多因素以错综复杂、彼此牵制而又彼此交互的方式组成的结构；许多项必要条件的权重计算方式发生变化，整体设计就需要重新考量。

约束在持续变化：
设计任务书是一个规定了目标、必要条件以及约束的文档，在设计开始之前即便是围绕着设计任务书进行设计的。然而然而设计过程中，这些约束不是一成不变的。

设计空间之外的约束在变化：
有时候设计的突破性进展是来自设计空间之外；设计空间内遇到棘手的问题，因设计空间之外的条件改变，在设计空间之内的约束就会自动消除。

对理性模型的其它批评：理性模型是一种设计师很自然想到的一种模型，但是他们常常不那么做。

50、《设计原本》的笔记-第19页

建筑设计师需要研究一定数量的优秀设计和失败设计，才会出师。工业设计也是如此。唯有软件

行业的设计师，少有人也少有机会去研究优秀的设计。一方面是软件不是处于开发状态就是进入维护阶段，而进入维护阶段的软件又少有人愿意深入研究。软件设计师大多都在追求新的技术，而忽视设计本身。另一方面，软件本身的设计更多的隐含在其自身，大多无法由明显的形式表现出来，这无形中增加了研究的复杂度，从而极大的加大了研究的时间成本。