GCG GBS CTO Office

Nourishing Technical Vitality for Better Client Innovation

IBM

项目经理如何在万物互联时代生存下来？

赵卫

敏捷教练及顾问

IBM 大中华区Agile/DevOps卓越中心

GBS INNOVATION CENTER

INNOVATION INCUBATION PROGRAM

Bluemix PoC

赵卫 ——敏捷教练及顾问

- **专业技能：**
  - 大型敏捷转型实施辅导与咨询（4年）
  - Scrum、看板及精益方法（9年）
  - 敏捷培训和工作坊（4年）
  - 研发经理和软件开发（16+年）
- **近期客户:**
  - 百联全渠道电子商务敏捷转型，敏捷教练及顾问
  - 汇丰银行，敏捷培训和辅导
  - 工商银行敏捷转型，敏捷教练及顾问
  - 西门子楼宇安防&电器集团，精益敏捷培训
  - 华为IT组织敏捷转型，敏捷教练及顾问
  - 华为研发产品级敏捷转型，敏捷教练及顾问
  - 爱立信Modem BU组织级敏捷转型，敏捷教练及顾问
- **近期演讲及培训:**
  - 高效的组织敏捷转型框架 – TiD 2016, Agile China
  - Evolution and Revolution Change – 北京敏捷之旅2015
  - Agile Transition from Project Management to Product Development Flow – ProMAC 2015, Sapporo, Japan
  - OMG最新软件工程标准Essence和状态卡游戏嘉年华 —— ScrumGathering ShangHai 2014
  - 大规模敏捷工作坊 —— TiD 2014, Agile China
- **相关认证:**
  - CSM， CSP，Certified SAFe Program Consultant (SPC)
  - Certified Kanban Training Program

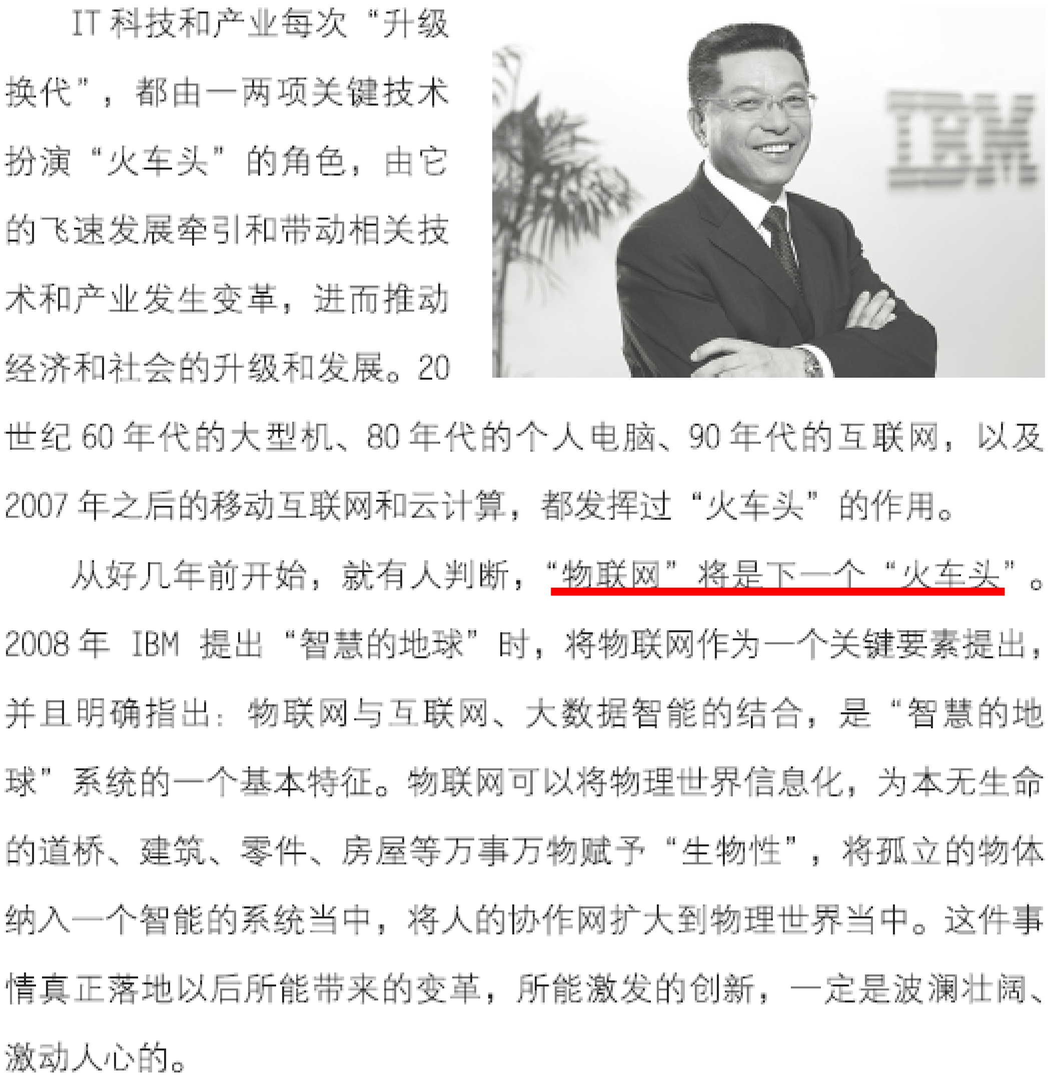
联系信息

- 邮箱: weizbj@cn.ibm.com
- 微信: zhaoweiDavid001
- Skype: zhaoweiok
- Linkedin: http://cn.linkedin.com/in/zhaoweiok

未来

未来

物联网是科技和产业升级换代的火车头



IT科技和产业每次“升级换代”，都由一两项关键技术扮演“火车头”的角色，由它的飞速发展牵引和带动相关技术和产业发生变革，进而推动经济和社会的升级和发展。20世纪60年代的大型机、80年代的个人电脑、90年代的互联网，以及2007年之后的移动互联网和云计算，都发挥过“火车头”的作用。

从好几年前开始，就有人判断，“物联网”将是下一个“火车头”。2008年IBM提出“智慧的地球”时，将物联网作为一个关键要素提出，并且明确指出：物联网与互联网、大数据智能的结合，是“智慧的地球”系统的一个基本特征。物联网可以将物理世界信息化，为本无生命的道桥、建筑、零件、房屋等万事万物赋予“生物性”，将孤立的物体纳入一个智能的系统当中，将人的协作网扩大到物理世界当中。这件事情真正落地以后所能带来的变革，所能激发的创新，一定是波澜壮阔、激动人心的。

来自跨界竞争的颠覆性创新

“New competitors, many of them from outside the industry, are entering the picture. What they’re doing will disrupt the market and our customer base.”

CEO, Banking and Financial Markets, Canada

Redefining Competition

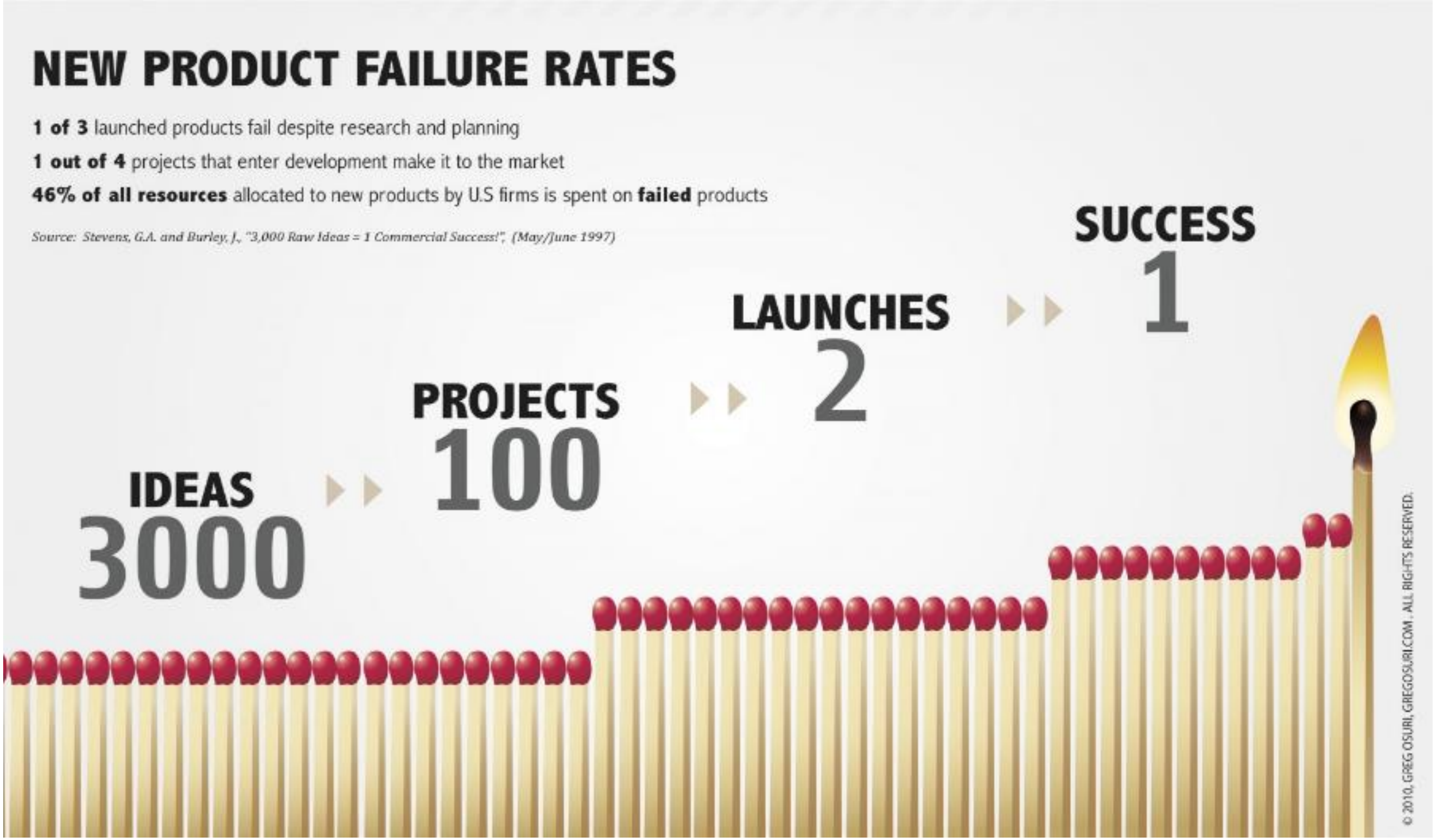
Insights from the Global C-suite Study – The CEO perspective

IBM Institute for Business Value

没有人会说这很容易

当部署新的商业模式的时候，CxO们面临很多障碍

新产品开发失败率



NEW PRODUCT FAILURE RATES

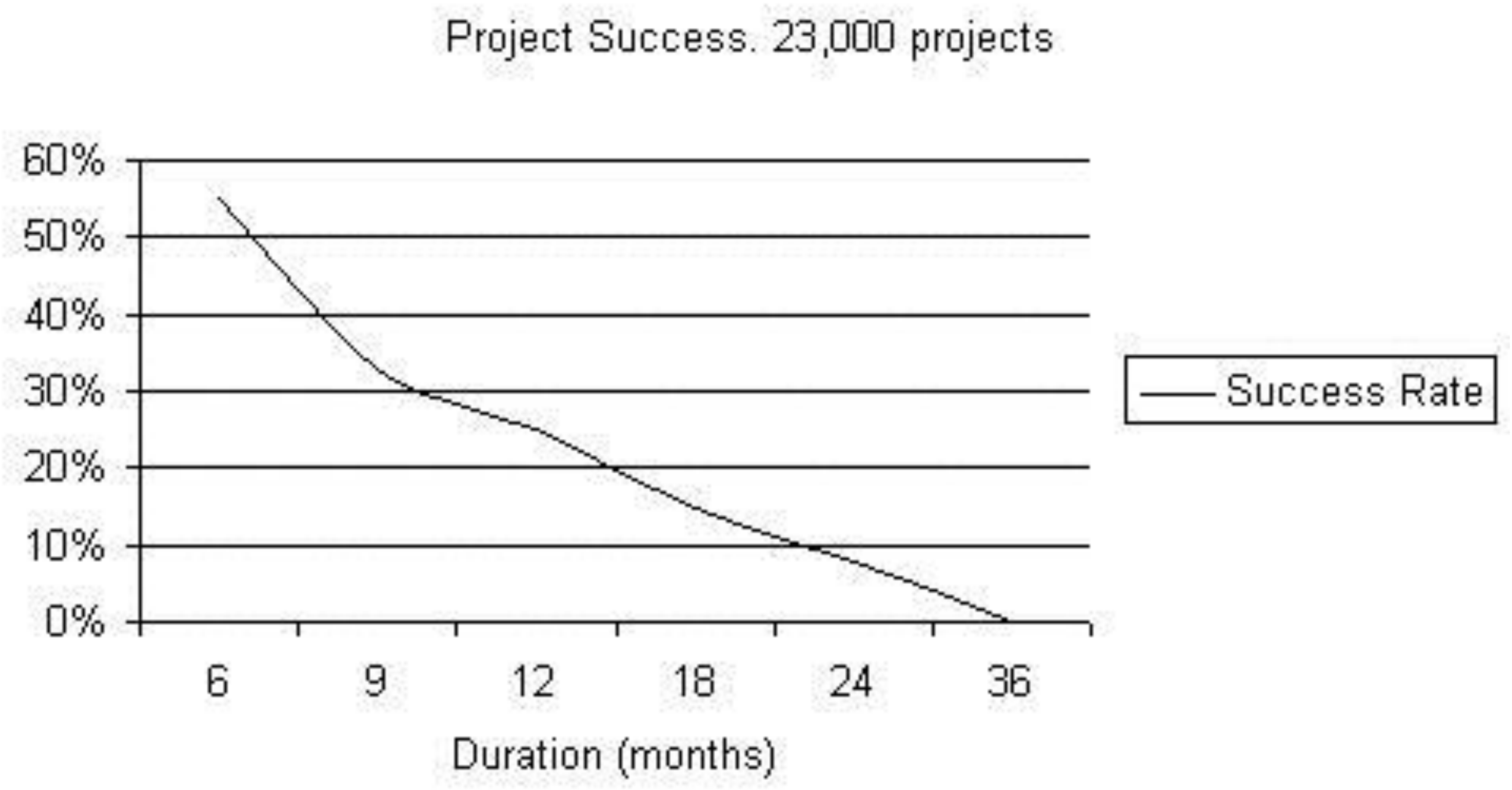
1 of 3 launched products fail despite research and planning

1 out of 4 projects that enter development make it to the market

46% of all resources allocated to new products by U.S firms is spent on **failed** products

Source: Stevens, G.A. and Burley, J., "3,000 Raw Ideas = 1 Commercial Success!", (May/June 1997)

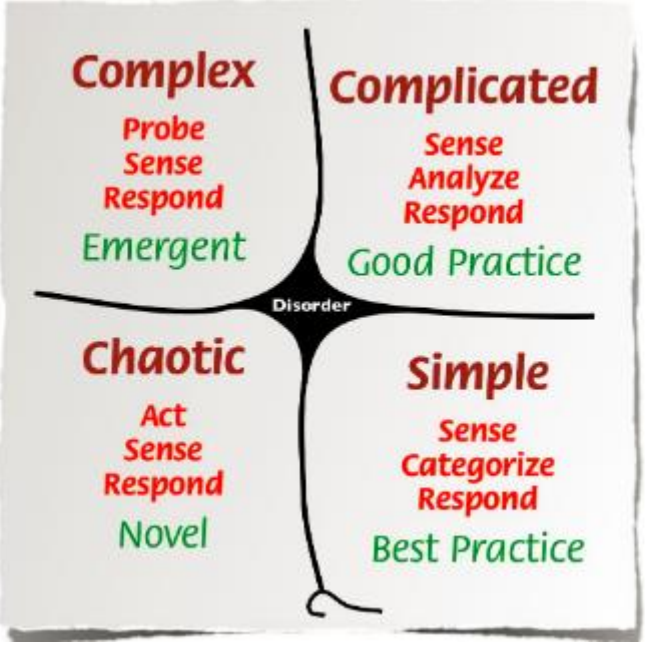
项目失败率



Turning Chaos into Success, Jim Johnson, 1999
http://www.softwaremag.com/focus-areas/application-focus/featured-articles/turning-chaos-into-success/

项目经理如何生存？

Dave Snowden Cynefin framework



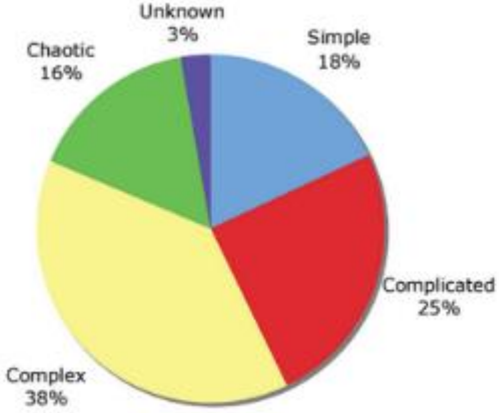
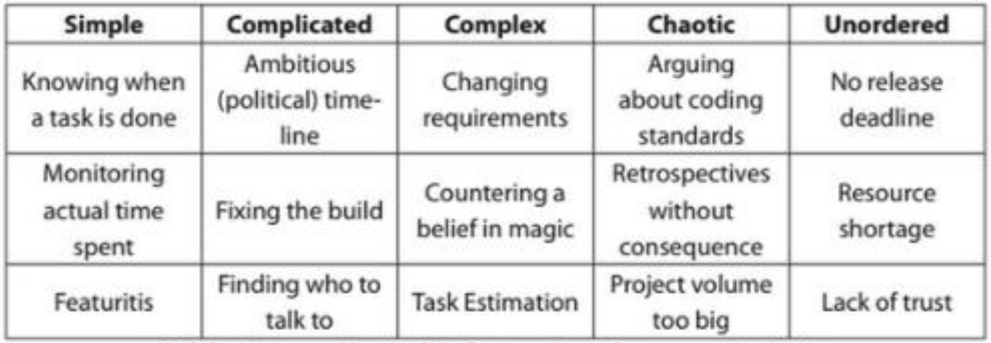


Figure 3 *Breakdown Of Typical Activities In Software Development*

Table 1 offers a sample of typical activities provided by participants, together with their sense-making results:

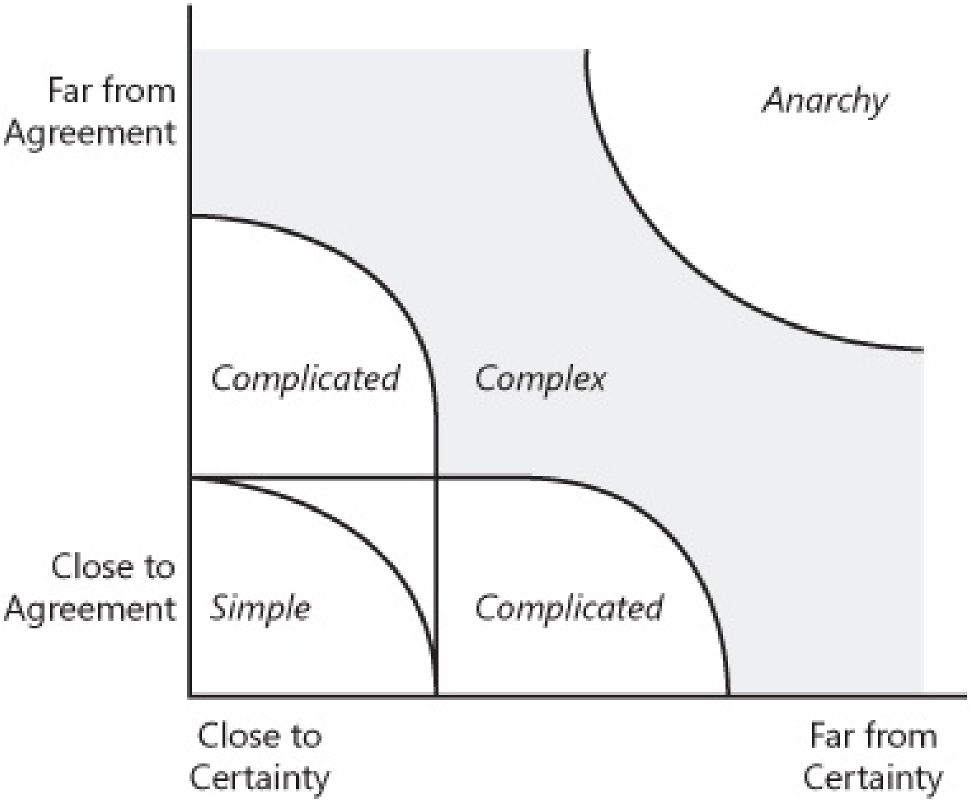


Simple	Complicated	Complex	Chaotic	Unordered
Knowing when a task is done	Ambitious (political) time-line	Changing requirements	Arguing about coding standards	No release deadline
Monitoring actual time spent	Fixing the build	Countering a belief in magic	Retrospectives without consequence	Resource shortage
Featuritis	Finding who to talk to	Task Estimation	Project volume too big	Lack of trust

Table 1 *Some Typical Software Development Activities*

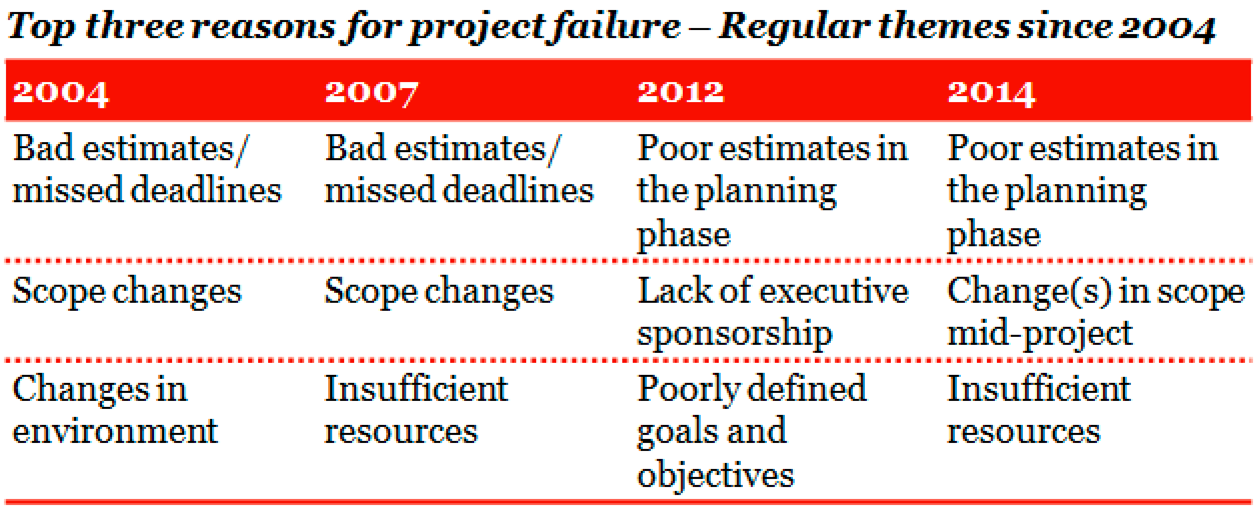
http://cognitive-edge.com/articles/on-understanding-software-agility-a-social-complexity-point-of-view/

Ralph Stacey Matrix



项目失败的主要原因

Portfolio and Programme Management 2014 Global Survey

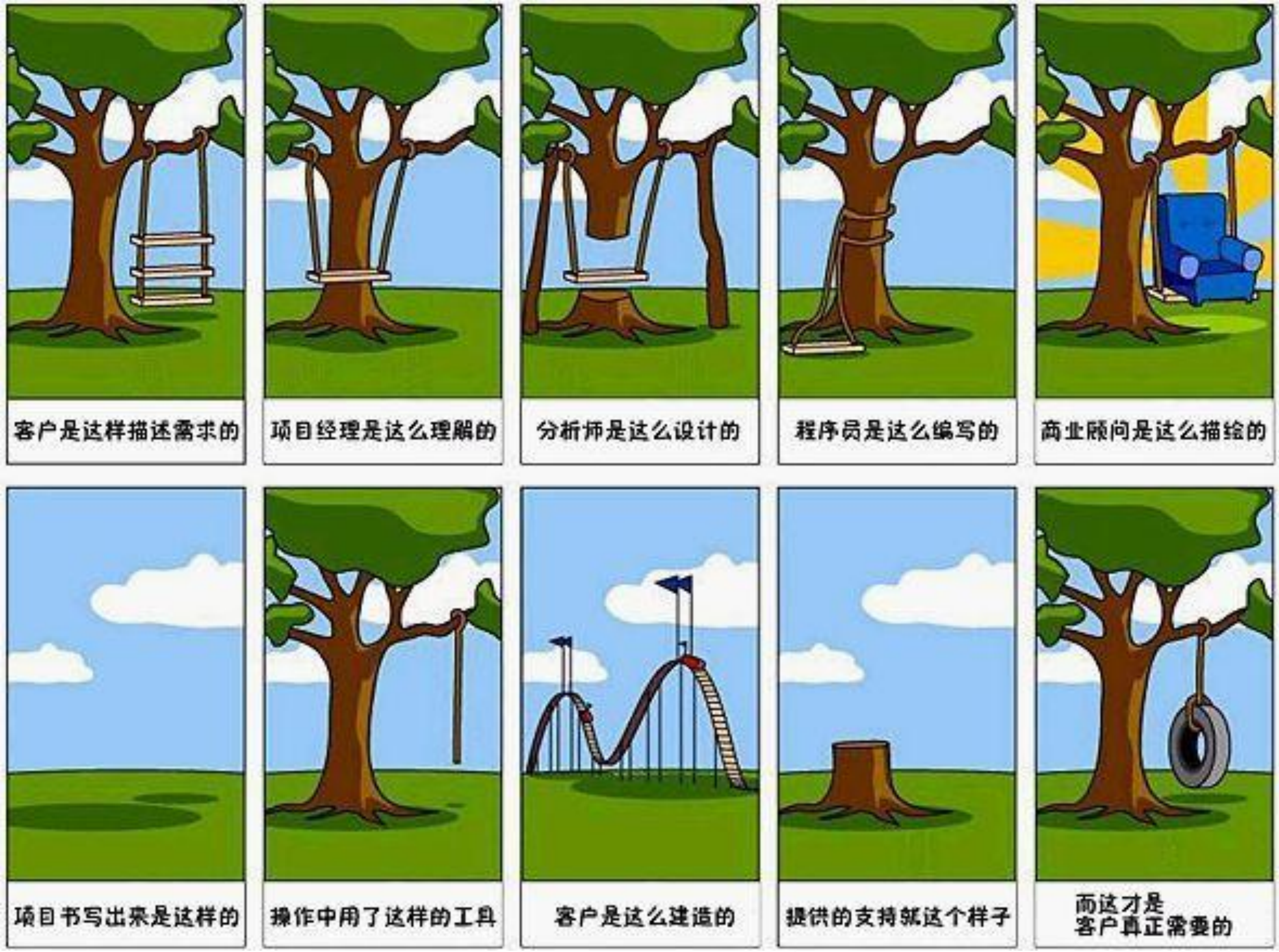


Top three reasons for project failure – Regular themes since 2004

2004	2007	2012	2014
Bad estimates/ missed deadlines	Bad estimates/ missed deadlines	Poor estimates in the planning phase	Poor estimates in the planning phase
Scope changes	Scope changes	Lack of executive sponsorship	Change(s) in scope mid-project
Changes in environment	Insufficient resources	Poorly defined goals and objectives	Insufficient resources

http://www.pwc.com/gx/en/services/advisory/consulting/portfolio-programme-management/global-ppm-survey-2014.html

需求挑战


客户是这样描述需求的
项目经理是这么理解的
分析师是这么设计的
程序员是这么编写的
商业顾问是这么描绘的
项目书写出来是这样的
操作中用了这样的工具
客户是这么建造的
提供的支持就这个样子
而这才是
客户真正需要的

需求变更

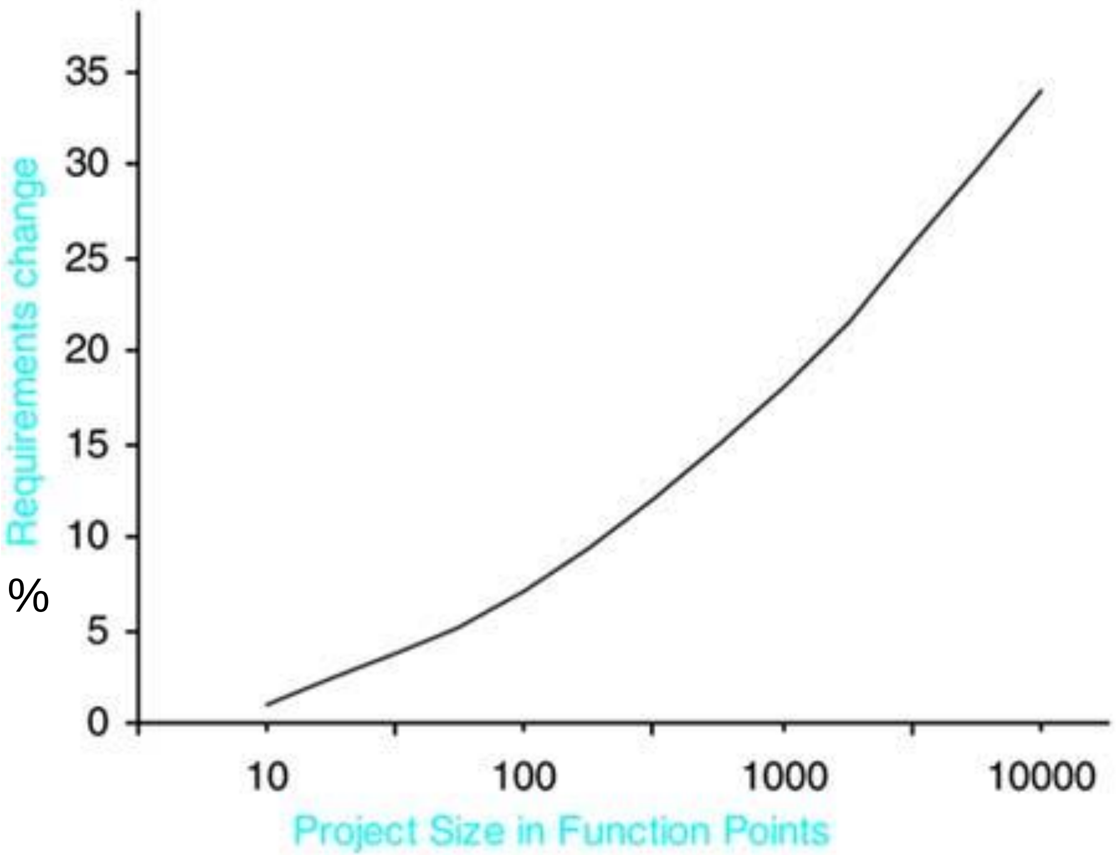
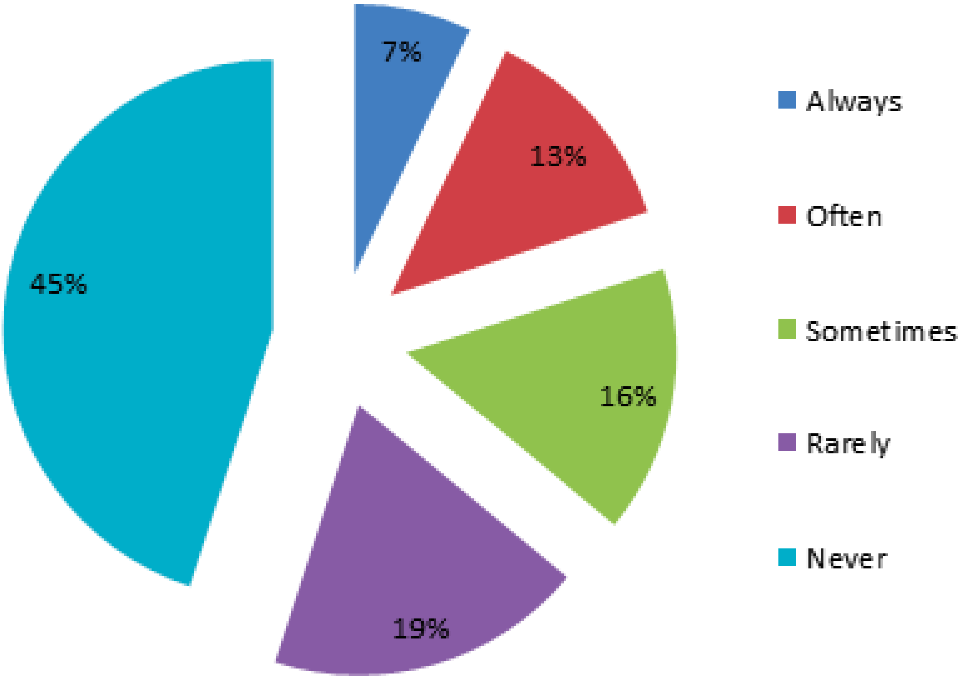


Figure 5.1. rates of change on software projects

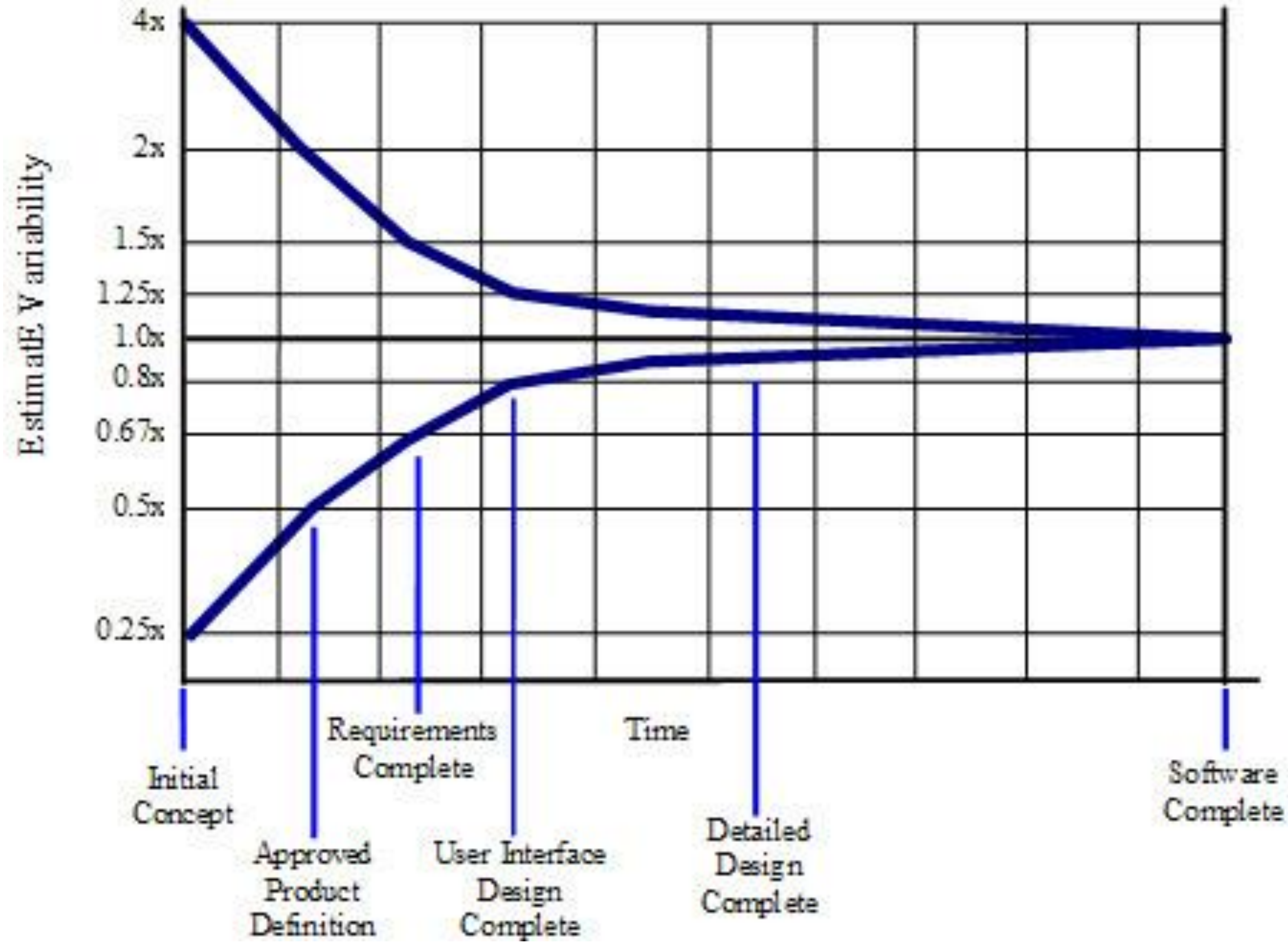
Agile and Iterative Development: A Manager's Guide, by Craig Larman, 2003

功能满足用户的需要了么？



Jim Johnson. The Standish Group International Inc. 2002.

不确定锥形——我们是在预测未来么？



http://www.construx.com/Page.aspx?cid=1648

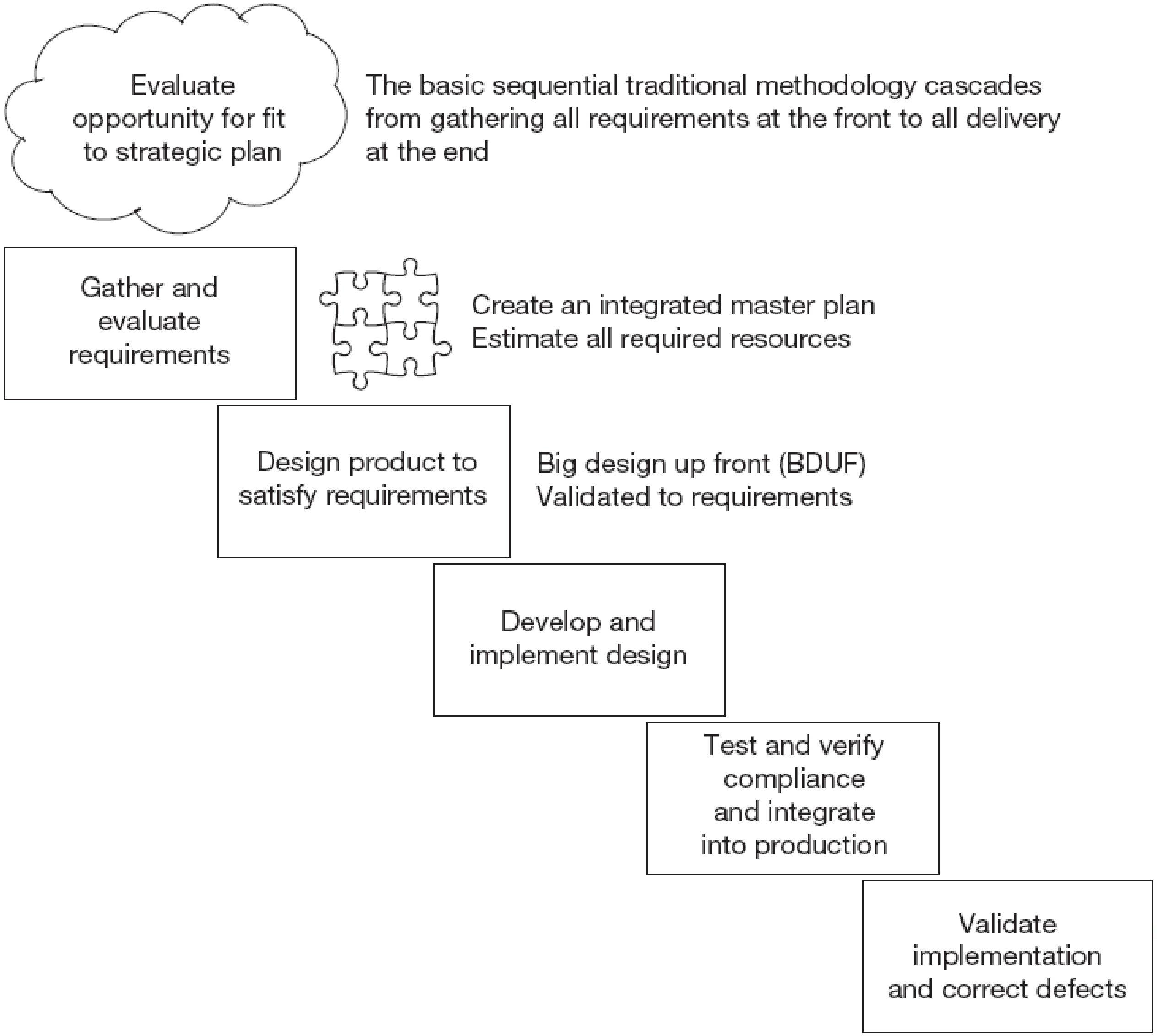
Chemical Industry

Gorey 1958
Bauman 1958

Software Industry

Barry Boehm 1981

瀑布模型



Evaluate opportunity for fit to strategic plan

The basic sequential traditional methodology cascades from gathering all requirements at the front to all delivery at the end

Gather and evaluate requirements

Create an integrated master plan
Estimate all required resources

Design product to satisfy requirements

Big design up front (BDUF)
Validated to requirements

Develop and implement design

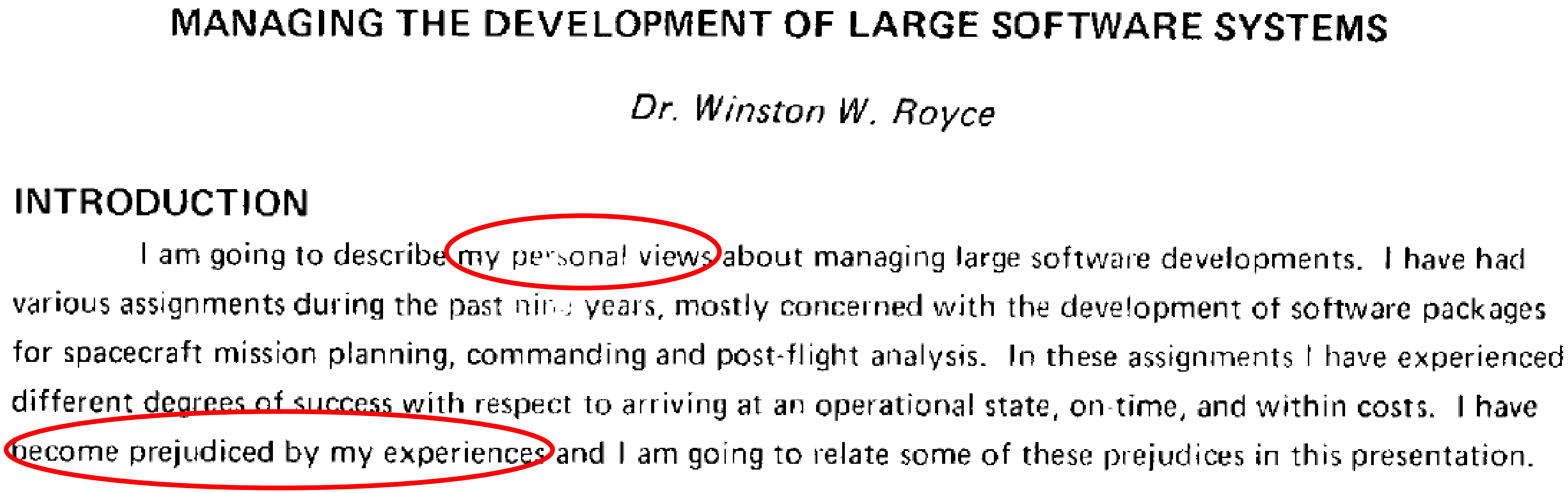
Test and verify compliance and integrate into production

Validate implementation and correct defects

瀑布模型

August, 1970, Dr. Winston W. Royce

IBM首席软件经济学家



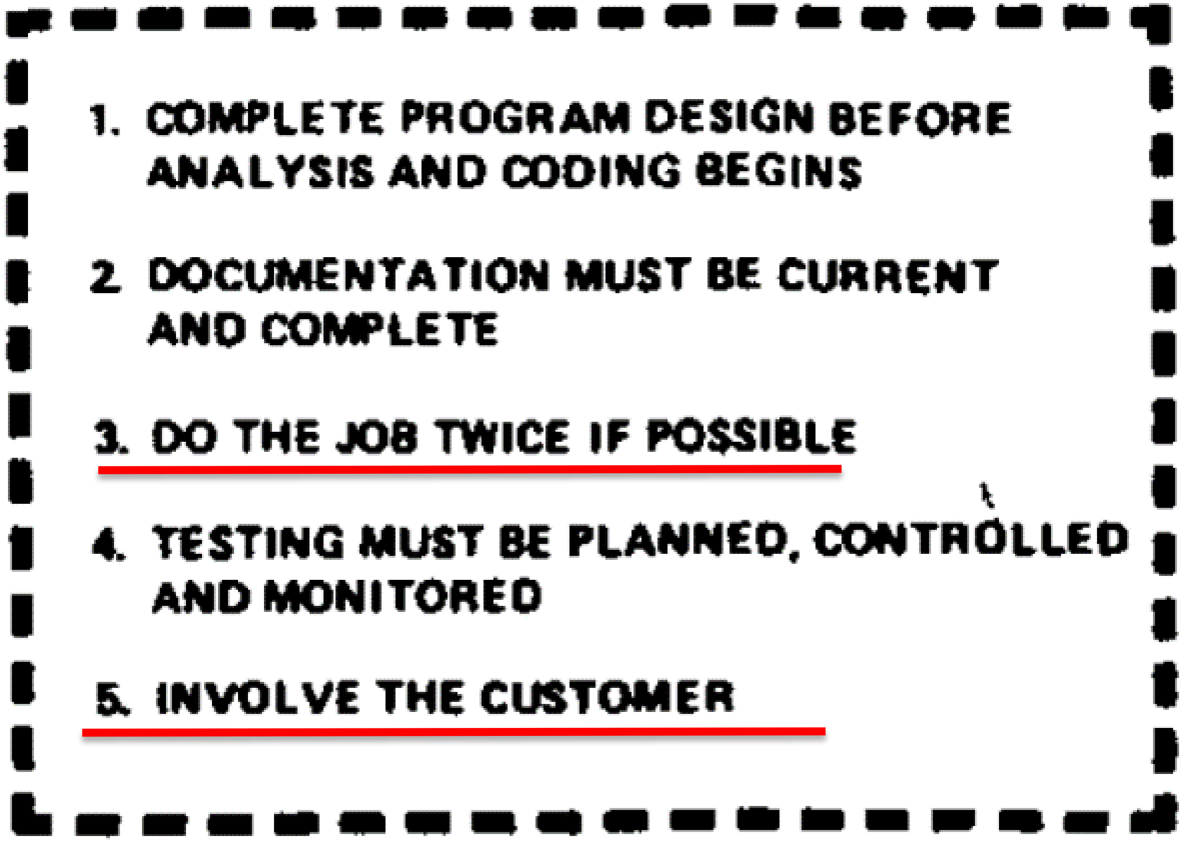
MANAGING THE DEVELOPMENT OF LARGE SOFTWARE SYSTEMS

Dr. Winston W. Royce

INTRODUCTION

I am going to describe my personal views about managing large software developments. I have had various assignments during the past nine years, mostly concerned with the development of software packages for spacecraft mission planning, commanding and post-flight analysis. In these assignments I have experienced different degrees of success with respect to arriving at an operational state, on-time, and within costs. I have become prejudiced by my experiences and I am going to relate some of these prejudices in this presentation.

瀑布模型



1. COMPLETE PROGRAM DESIGN BEFORE ANALYSIS AND CODING BEGINS
2. DOCUMENTATION MUST BE CURRENT AND COMPLETE
3. DO THE JOB TWICE IF POSSIBLE
4. TESTING MUST BE PLANNED, CONTROLLED AND MONITORED
5. INVOLVE THE CUSTOMER

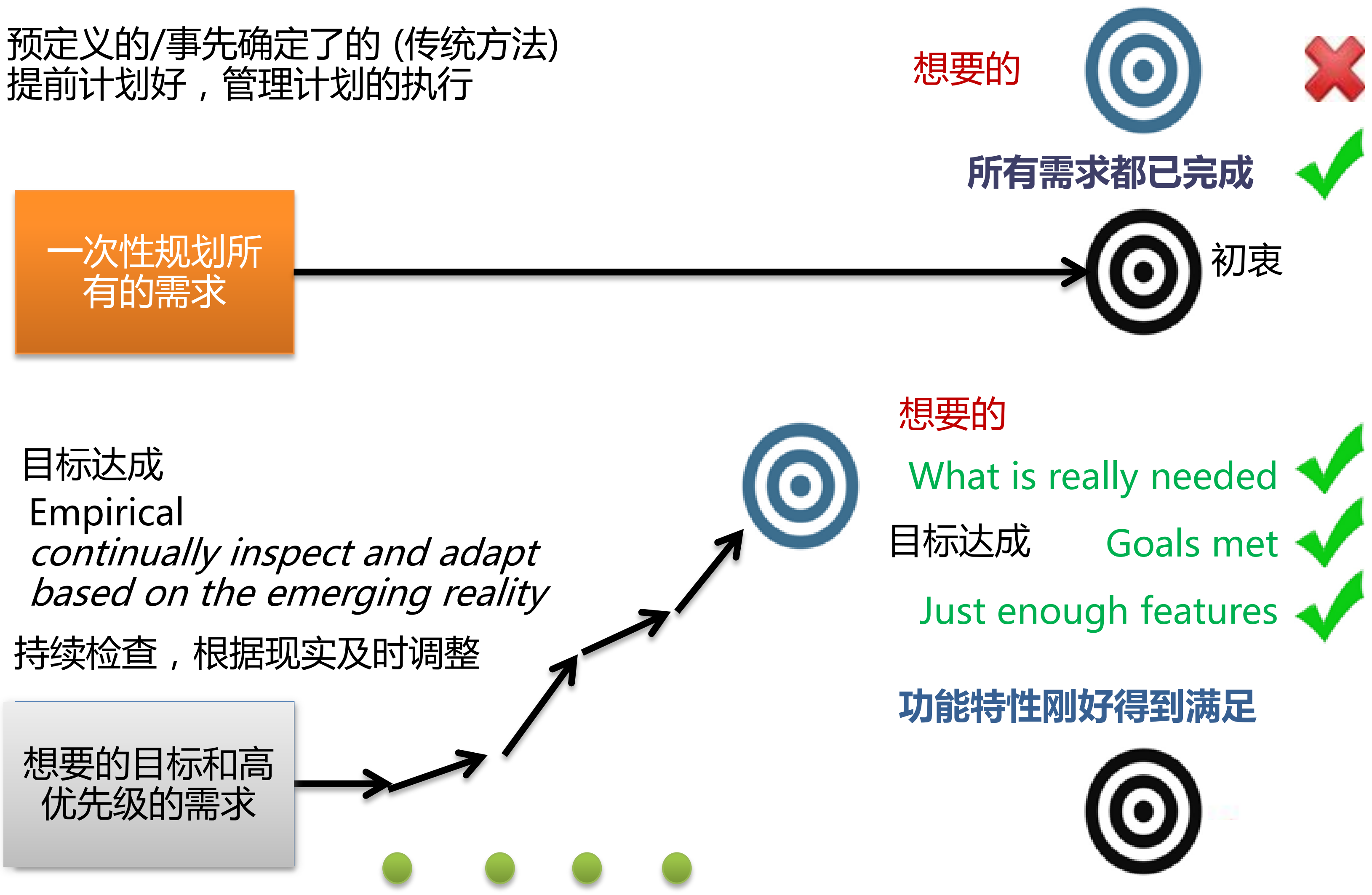
瀑布模型

> He was always a proponent of iterative, incremental, evolutionary development. His paper described the waterfall as the simplest description, but that it would not work for all but the most straightforward projects. The rest of his paper describes [iterative practices] within the context of the 60s/70s government-contracting models (a serious set of constraints).

Walker Royce,
speaking of his father
Winston Royce
IBM首席软件经济学家

LB03 Larman, C., and Basili, V. 2003.
"Iterative and Incremental Development: A Brief History." IEEE Computer, June 2003.

复杂自适应系统：试探-感受-响应的经验过程



预定义的/事先确定了的 (传统方法)
提前计划好，管理计划的执行

一次性规划所有的需求

想要的 ✗

所有需求都已完成 ✓

初衷

目标达成
Empirical
continually inspect and adapt based on the emerging reality

持续检查，根据现实及时调整

想要的目标和高优先级的需求

想要的

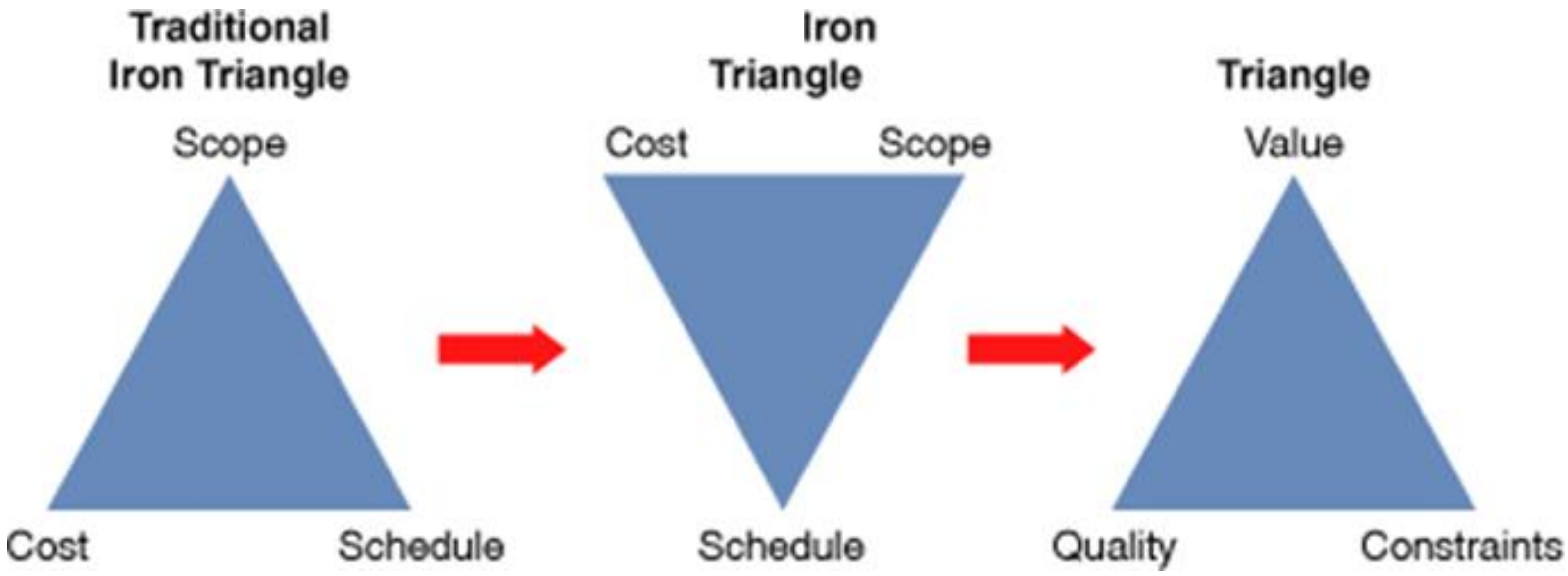
What is really needed ✓

目标达成 Goals met ✓

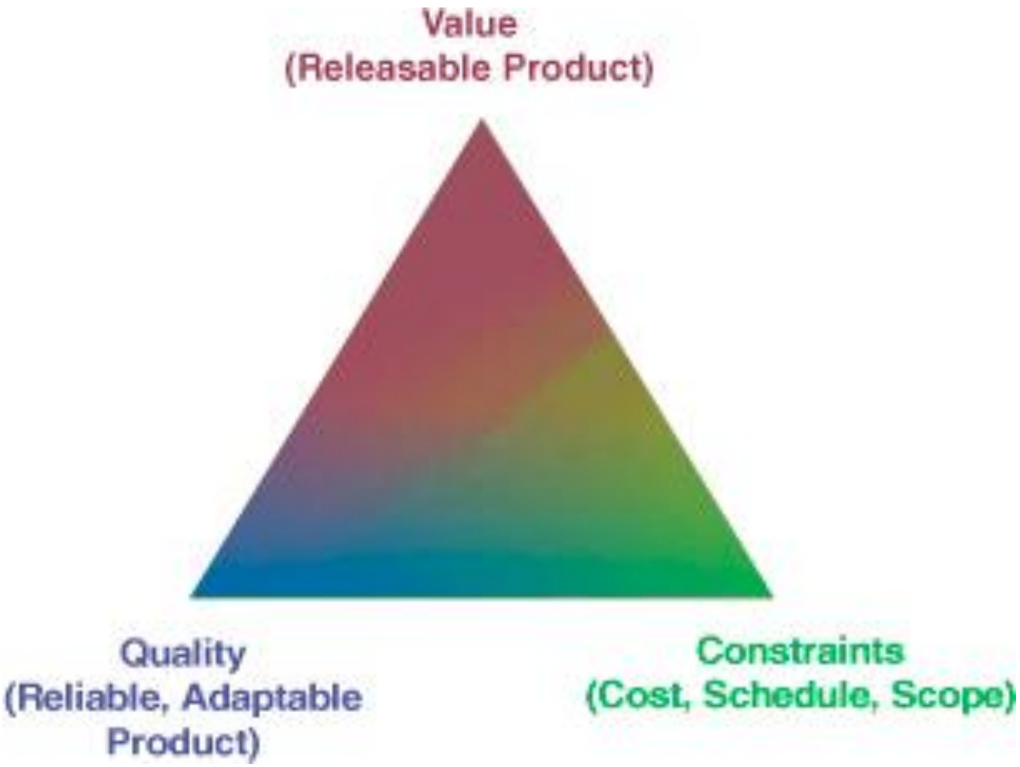
Just enough features ✓

功能特性刚好得到满足

传统铁三角的演进


Traditional
Iron Triangle
Scope
Cost
Schedule
Iron
Triangle
Cost
Scope
Schedule
Triangle
Value
Quality
Constraints

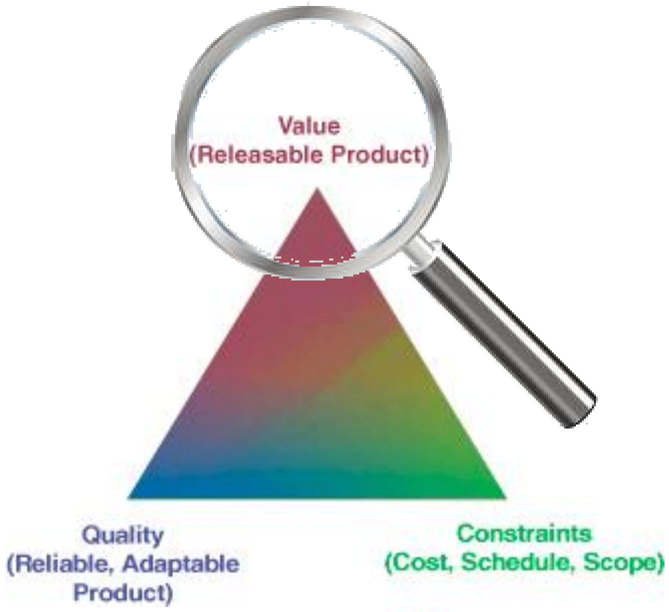
项目管理新的三角


Value
(Releasable Product)
Quality
(Reliable, Adaptable Product)
Constraints
(Cost, Schedule, Scope)

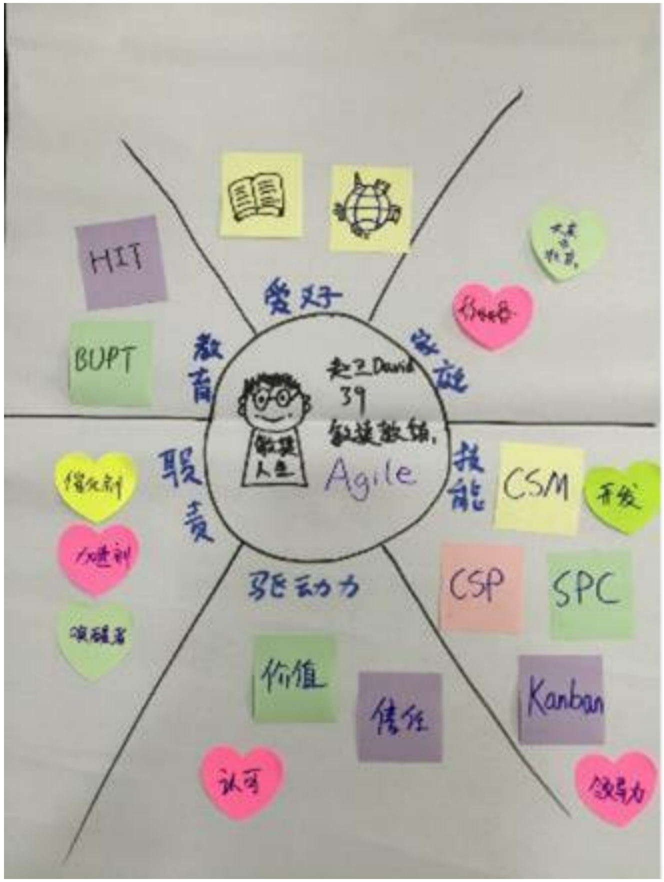
最大化业务价值，而不是按计划交付

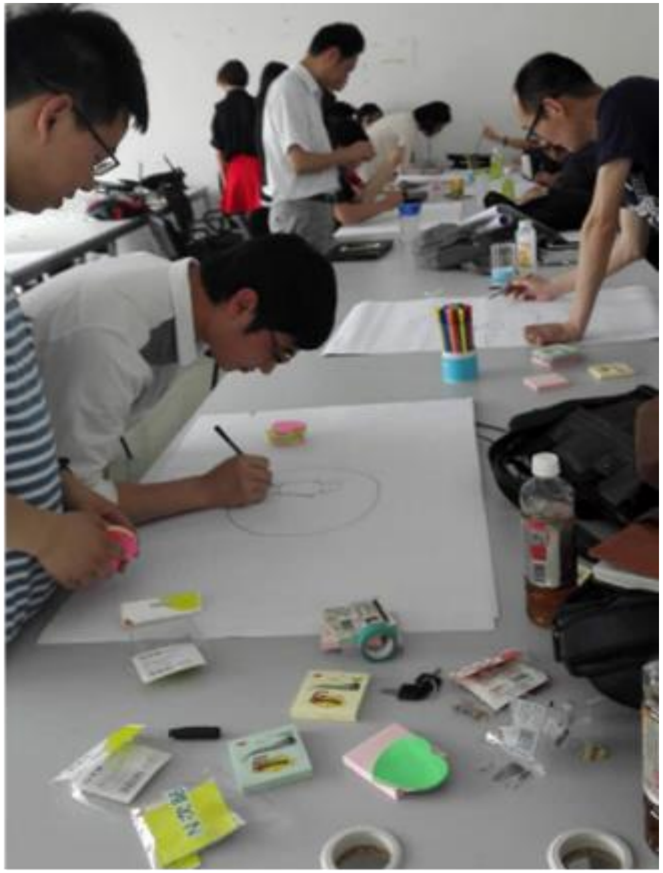
价值

- **所有的需求都是假设，如何定位？**
  - 目标用户，同理心地图，精益画布，价值定位，产品愿景，影响地图，用户故事
- **价值是假设，如何验证？**
  - 验证板
  - 关键业务指标
- **价值如何尽早获得？**
  - 需求拆分
  - 小批量上线

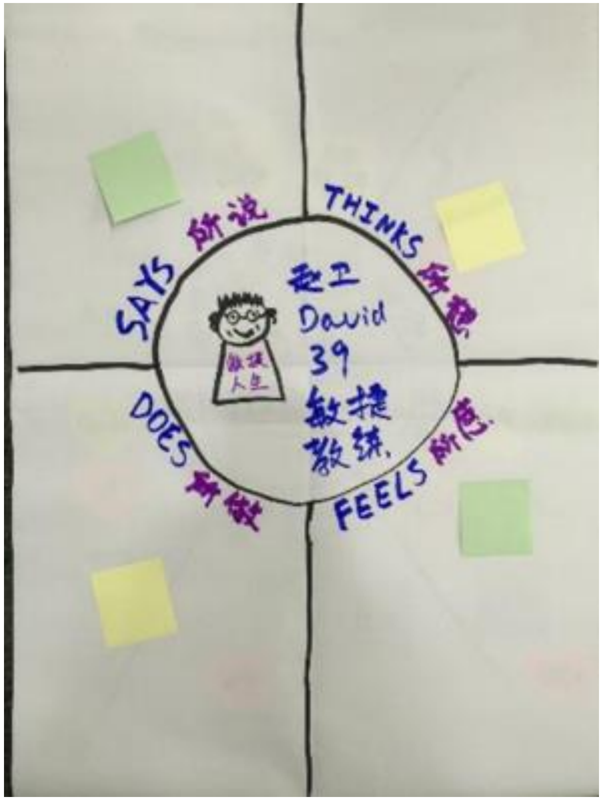


用户画像


爱好
家庭
技能
驱动力
职责
教育
赵卫David
39
敏捷教练
Agile
HIT
BUPT
CSM
开发
CSP
SPC
Kanban
价值
信任
认可
领导力
催化剂
加速剂
破碰者

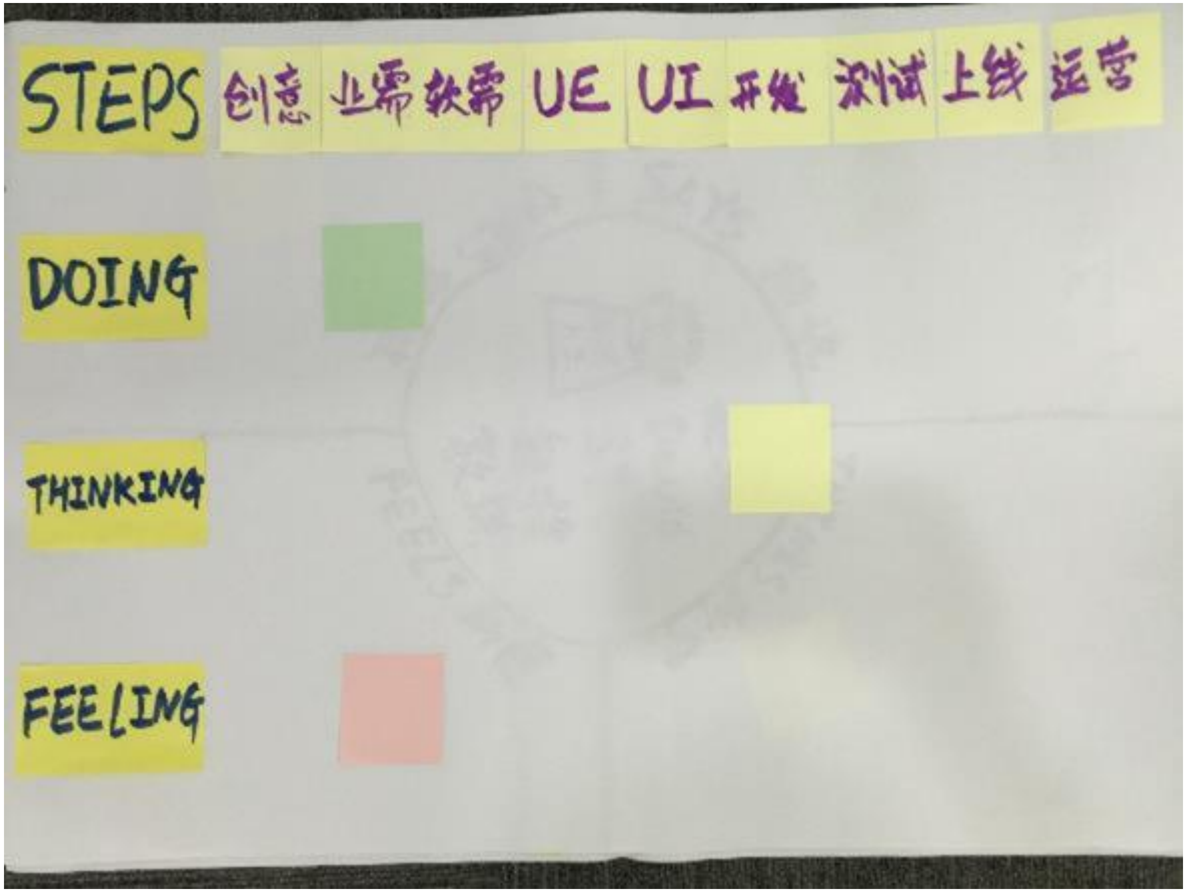


同理心地图

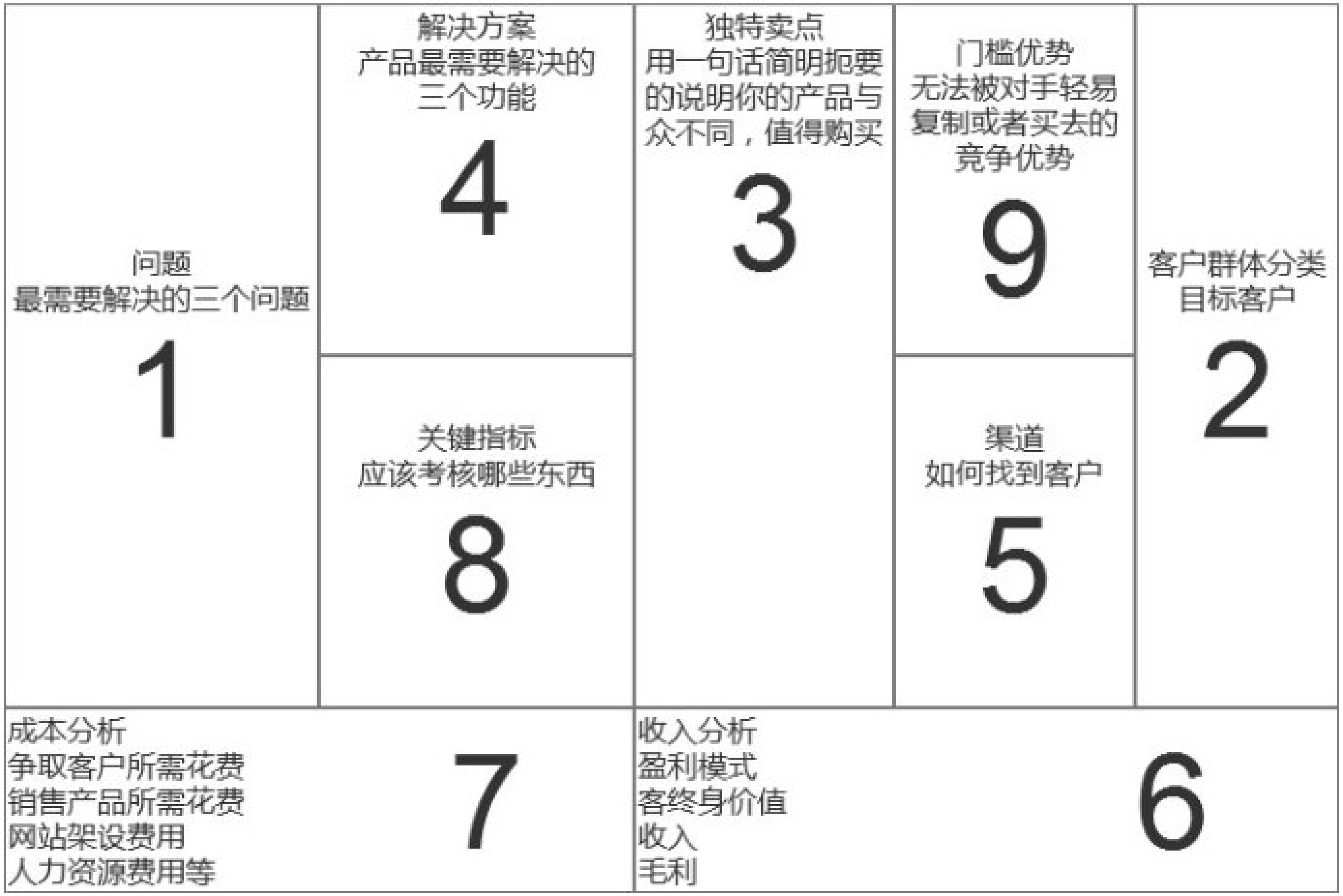

SAYS 所说
THINKS 所想
DOES 所做
FEELS 所感
赵卫
David
39
敏捷
教练.
敏捷
人生

SAYS
THINKS
DOES
FEELS
Marketer
APP
合作

场景地图



精益画布



问题
最需要解决的三个问题
1

解决方案
产品最需要解决的
三个功能
4

关键指标
应该考核哪些东西
8

独特卖点
用一句话简明扼要
的说明你的产品与
众不同，值得购买
3

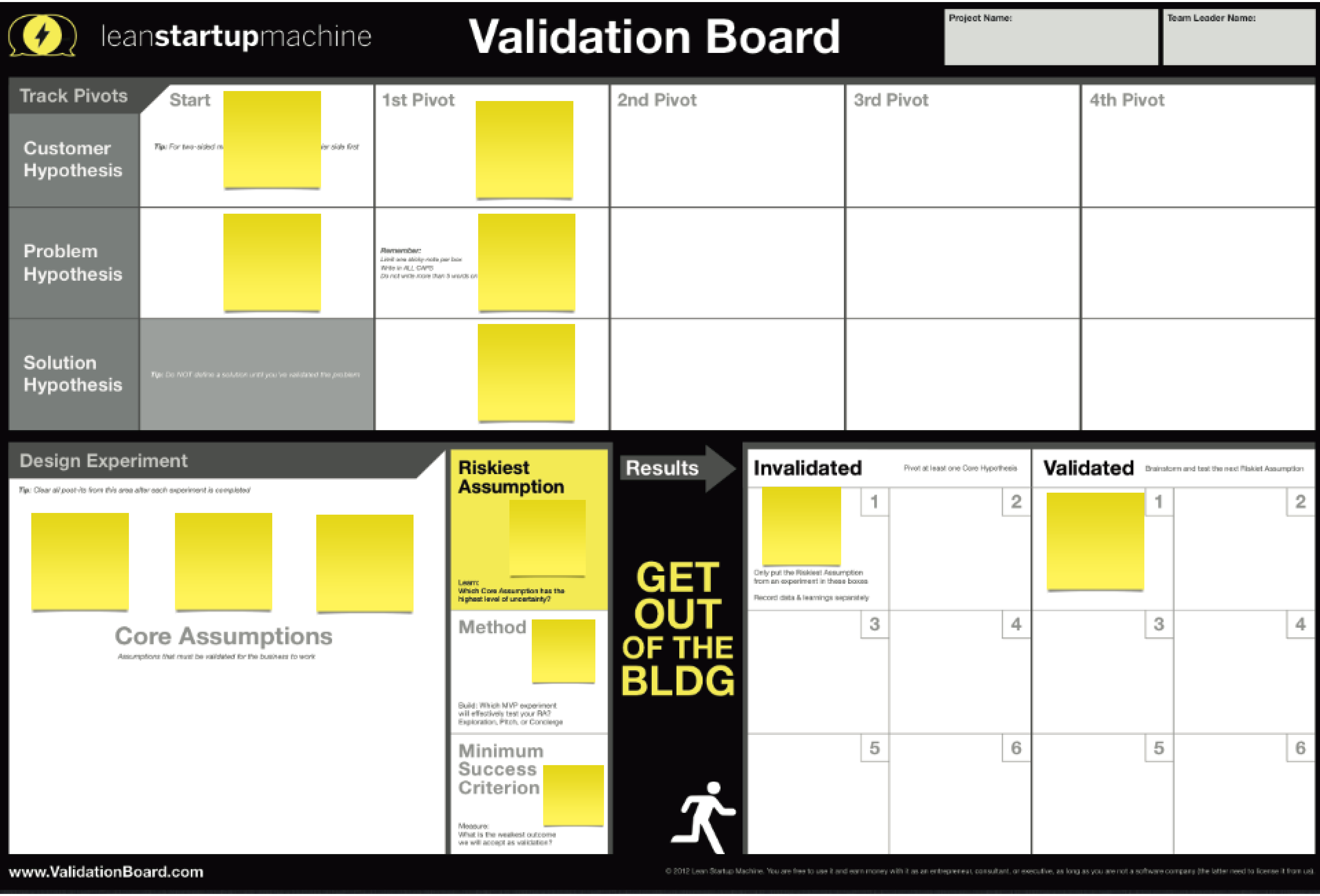
门槛优势
无法被对手轻易
复制或者买去的
竞争优势
9

渠道
如何找到客户
5

客户群体分类
目标客户
2

成本分析
争取客户所需花费
销售产品所需花费
网站架设费用
人力资源费用等
7

收入分析
盈利模式
客终身价值
收入
毛利
6



leanstartupmachine

Validation Board

Project Name:

Team Leader Name:

Track Pivots

Start | 1st Pivot | 2nd Pivot | 3rd Pivot | 4th Pivot

Customer Hypothesis

Tip: For two-sided m ... er side first

Problem Hypothesis

Remember:
Limit one sticky note per box
Write in ALL CAPS
Do not write more than 5 words on

Solution Hypothesis

Tip: Do NOT define a solution until you've validated the problem

Design Experiment

Tip: Clear all post-its from this area after each experiment is completed

Core Assumptions

Assumptions that must be validated for the business to work

Riskiest Assumption

Learn:
Which Core Assumption has the highest level of uncertainty?

Method

Build: Which MVP experiment
will effectively test your RA?
Exploration, Pitch, or Concierge

Minimum Success Criterion

Measure:
What is the weakest outcome
we will accept as validation?

Results

GET OUT OF THE BLDG

Invalidated Pivot at least one Core Hypothesis

1 2 3 4 5 6

Only put the Riskiest Assumption
from an experiment in these boxes

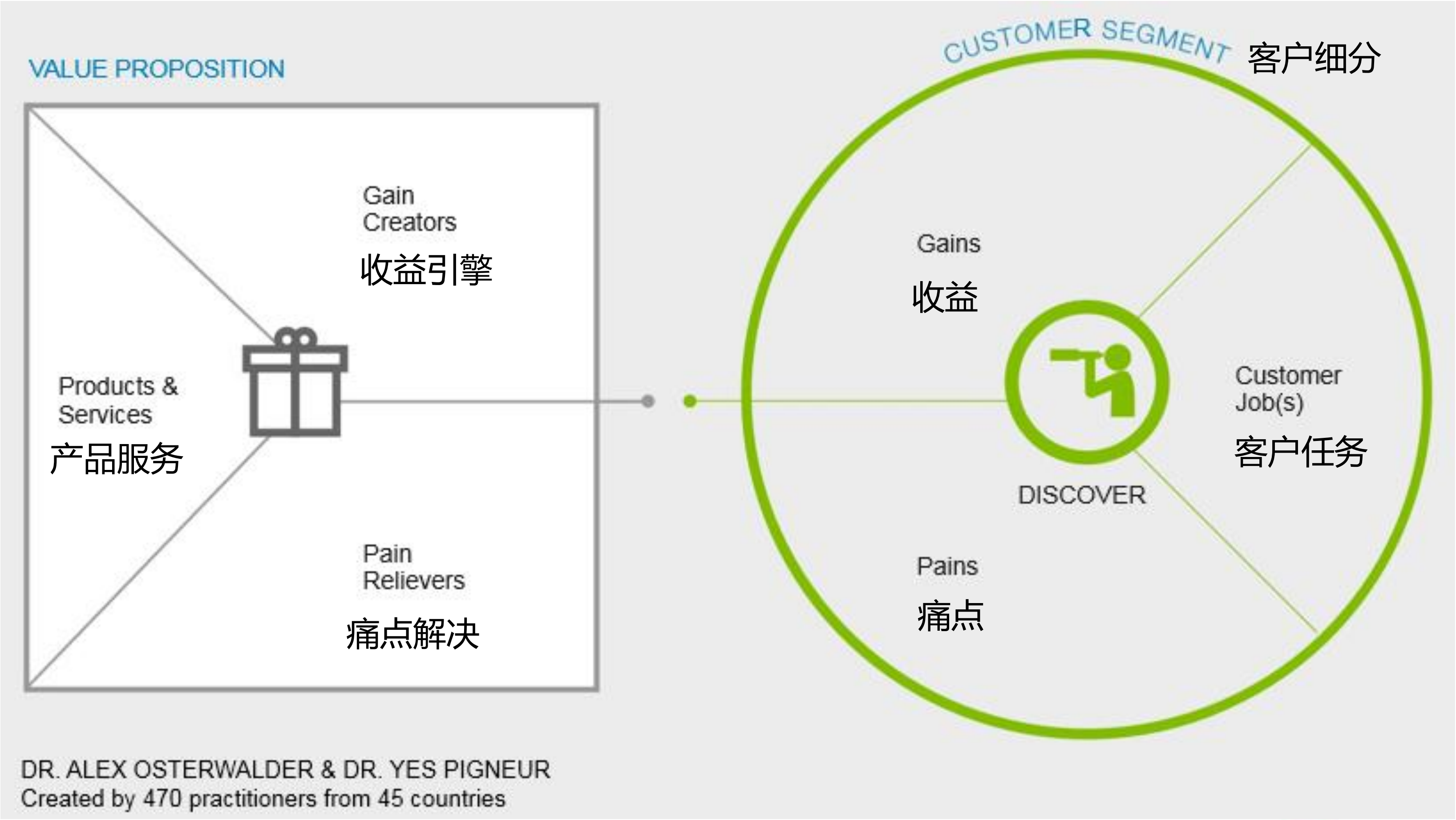
Record data & learnings separately

Validated Brainstorm and test the next Riskiest Assumption

1 2 3 4 5 6

www.ValidationBoard.com

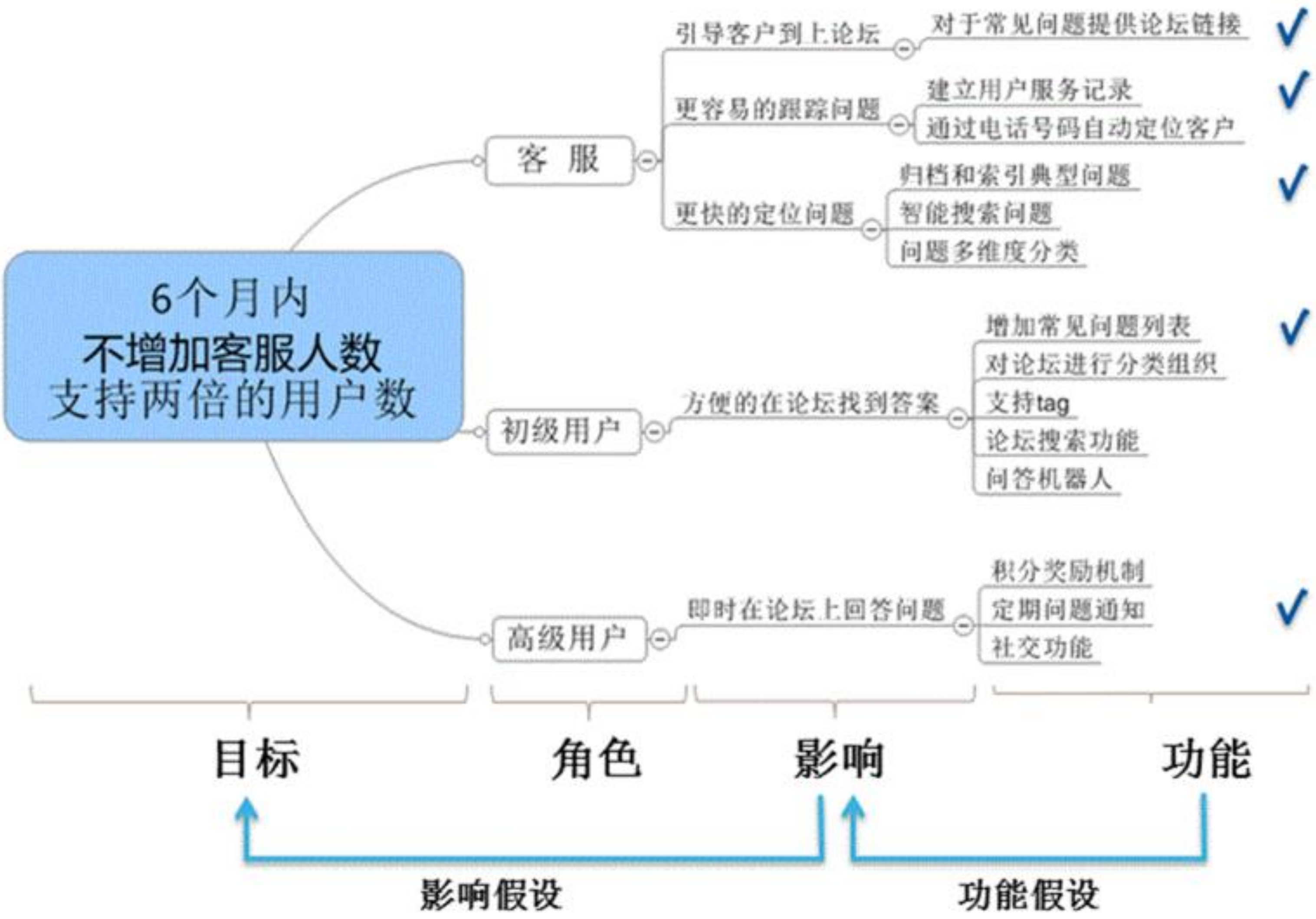
价值定位画布


VALUE PROPOSITION
CUSTOMER SEGMENT
客户细分
Gain Creators
收益引擎
Products & Services
产品服务
Pain Relievers
痛点解决
Gains
收益
Customer Job(s)
客户任务
DISCOVER
Pains
痛点
DR. ALEX OSTERWALDER & DR. YES PIGNEUR
Created by 470 practitioners from 45 countries

产品愿景

For（为谁/给谁）	
Who（有什么问题、影响）	
The（产品/特性/方案）	
Is a(它是什么类型产品)	
That（它有什么关键价值/收益声明，购买或者使用系统的引人入胜的原因）	
Unlike（不是/不像竞争对手或可替代方案）	
Our product（产品/方案的关键差异特性）	

影响地图



6个月内
不增加客服人数
支持两倍的用户数

- 客 服
  - 引导客户到上论坛
    - 对于常见问题提供论坛链接 ✓
  - 更容易的跟踪问题
    - 建立用户服务记录 ✓
    - 通过电话号码自动定位客户
  - 更快的定位问题
    - 归档和索引典型问题 ✓
    - 智能搜索问题
    - 问题多维度分类
- 初级用户
  - 方便的在论坛找到答案
    - 增加常见问题列表 ✓
    - 对论坛进行分类组织
    - 支持tag
    - 论坛搜索功能
    - 问答机器人
- 高级用户
  - 即时在论坛上回答问题
    - 积分奖励机制
    - 定期问题通知 ✓
    - 社交功能

目标 角色 影响 功能

影响假设

功能假设

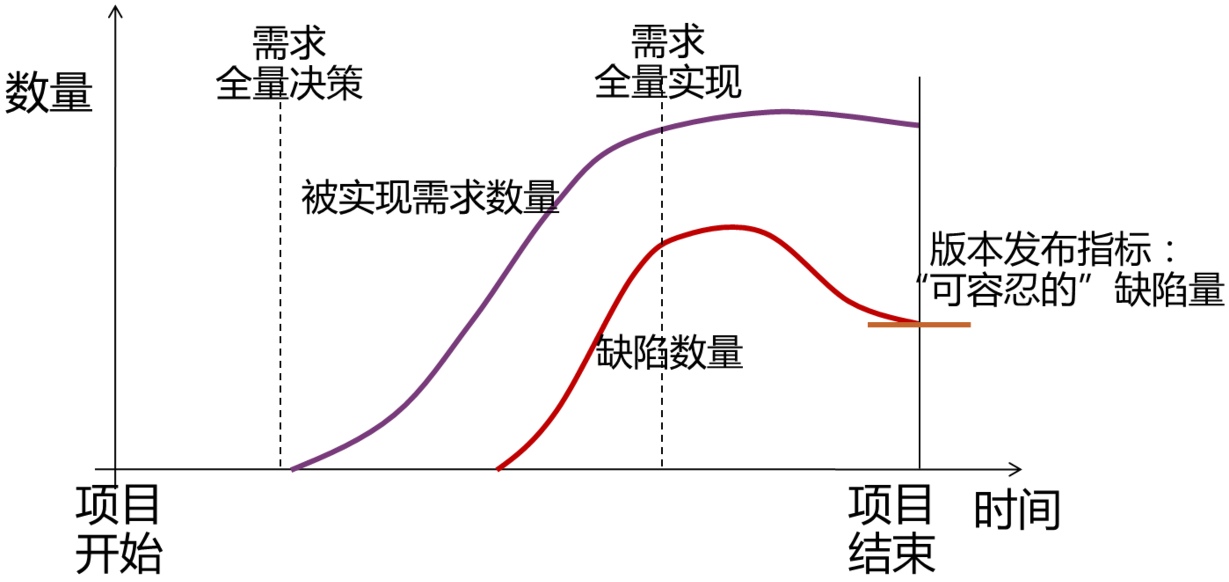
用户故事

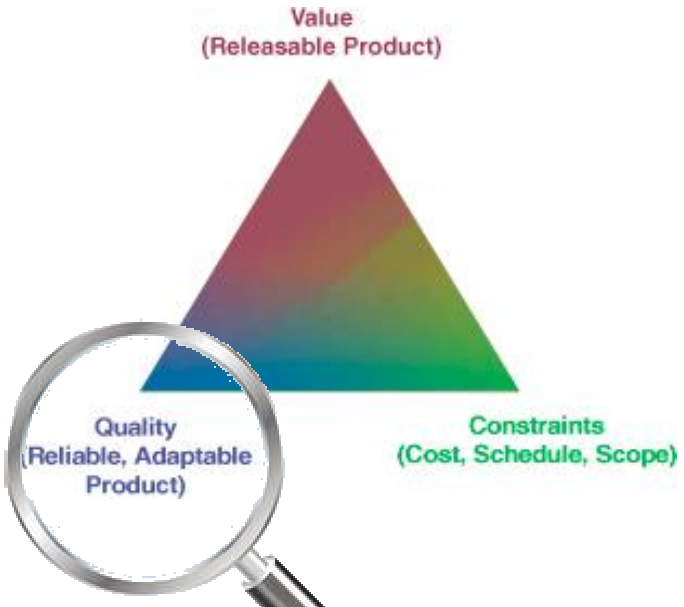
- **Who**
  - 作为（用户画像/角色）
- **What**
  - 我希望/想要（目标系统提供的行为或功能）
- **Why**
  - 从而/以便（这样我就能/实现什么业务价值或目标）

作为个人用户，我希望工作搜索功能，以便帮我找到符合我要求的工作。

质量

- 传统开发（海啸）强调**功能的完整性**胜于质量

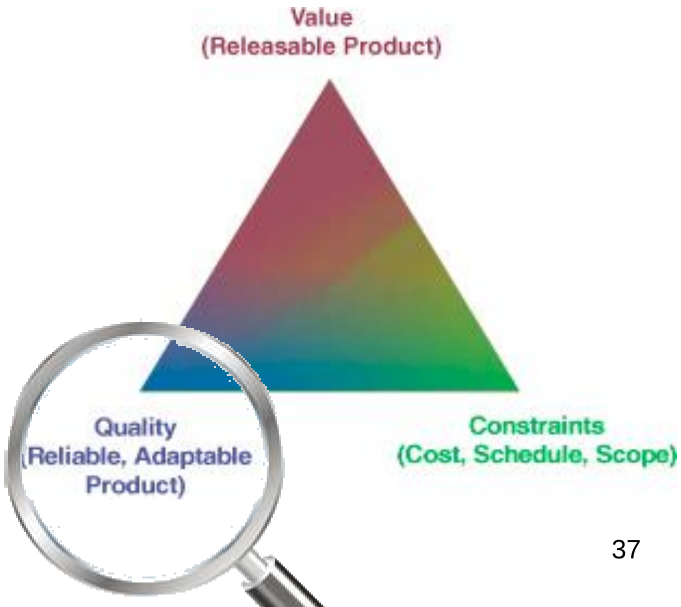




质量

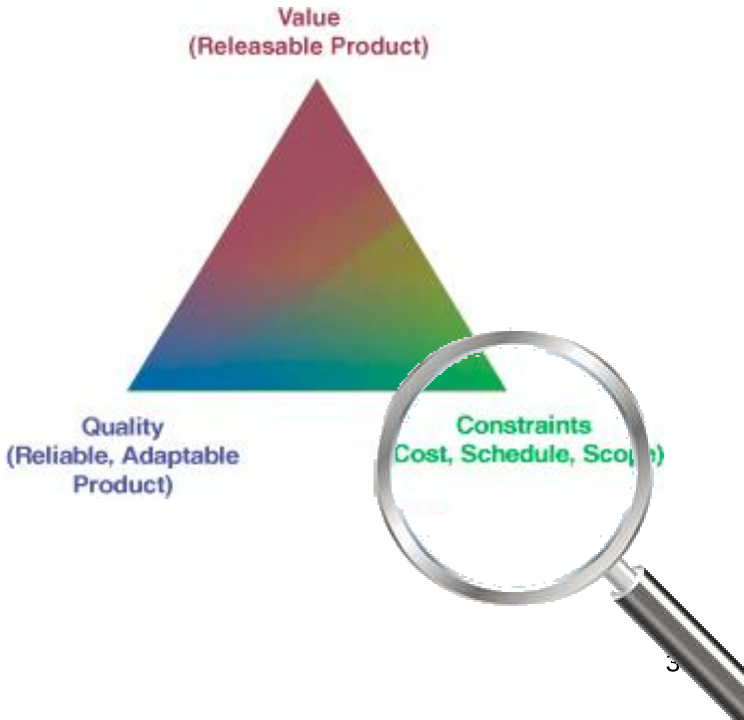
- **持续全员的质量内建和前移**
  - 持续集成
  - 测试先行
  - 重构
  - 结对工作
  - 集体所有制

Stop & Fix



约束——①成本

- **关注人而不是计划**
  - 传统项目管理，是以人为本么？
    - 人头，人力资源，劳工等，可以随时组建、解散和替换
  - 尊重人，授权和赋能的自组织的高绩效团队
    - 自我激励驱动的全员参与的理解需求、设计、估算
    - 对齐有意义的业务目标
    - 持续改进的团队
- **降低成本手段**
  - 提高效率
    - 减少技术债务
    - 持续集成和持续交付
  - 知识工作者的持续学习和反馈改进
    - 持续创新
    - 持续降低不确定性
  - 减少浪费



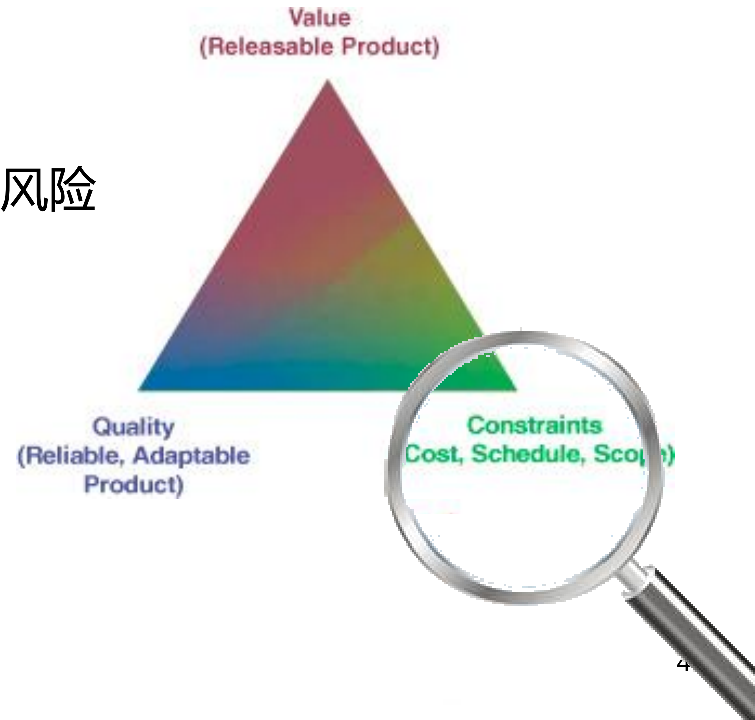
约束——①成本

软件开发中的浪费

- **部分完成的工作**
  - 已完成但尚未签入的代码
  - 没有相关说明文档的代码
  - 未测试的代码
  - 没人使用的代码
  - 被注释掉的代码
  - 分析完的需求发生变更
- **额外的步骤**
  - 过多细化的文档，过度分析
  - 过多的预防性代码，过度设计
- **额外的功能**
  - 镀金的、很少使用的功能，分析、设计、编码、集成、测试，系统复杂度，维护
  - 项目立项早期就固定需求，开发的时候时过境迁，已经发生变化
- **再学习/返工**
  - 糟糕的计划
  - 低劣的质量
  - 不足的沟通和知识积累
  - 没有相关说明的代码
- **交接移交**
  - 开发人员之间的代码交接
  - 开发人员和测试人员间软件的交接
  - 软件从开发到部署的交接
- **等待**
  - 各环节的等待
  - 大批量等待一起测试、上线
- **多任务切换**
  - 多任务切换带来的额外的工作量损耗
- **缺陷**
  - 所有的缺陷都是浪费

约束——②计划

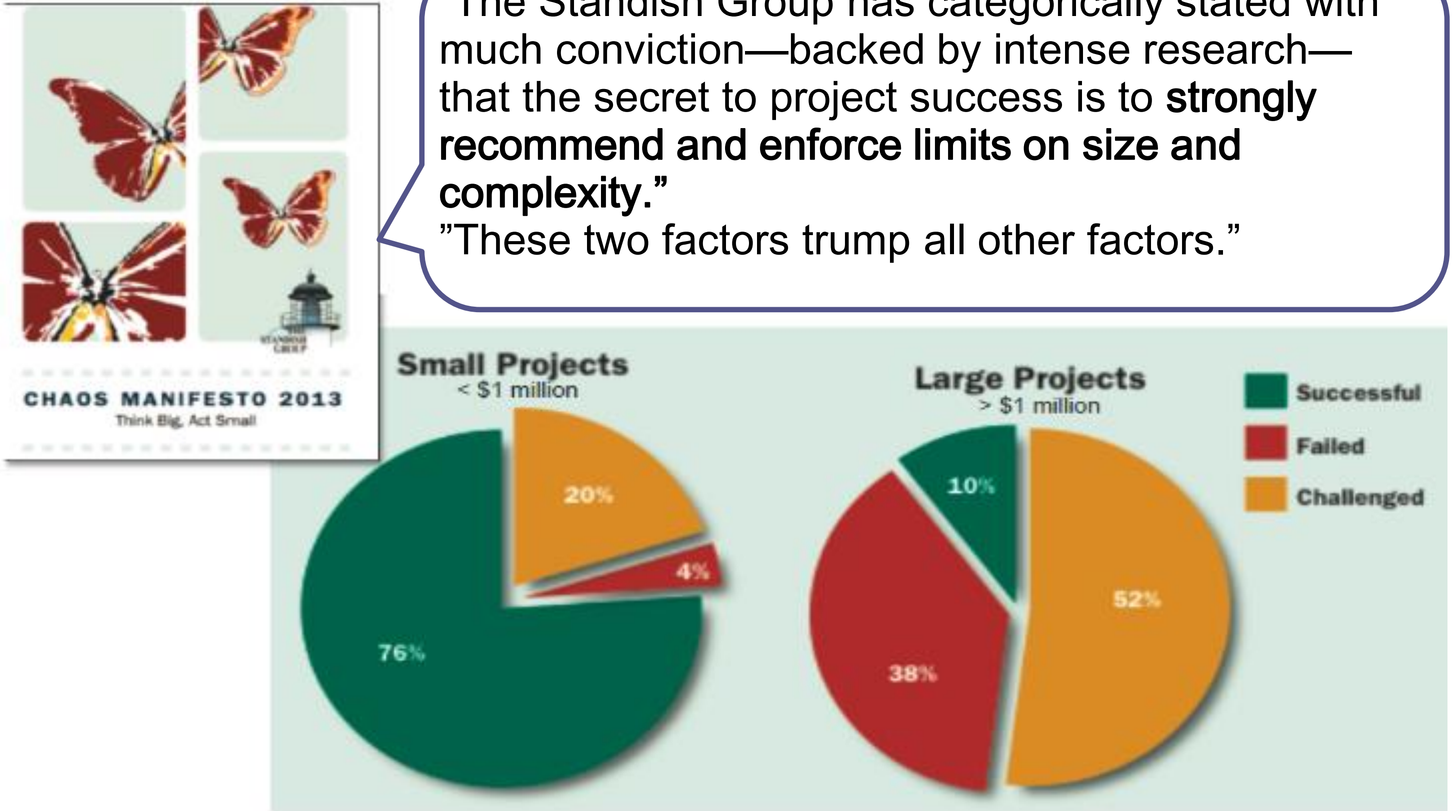
- **适应和响应变化而不是遵循计划**
  - 以可以预测未来算命的方式遵循计划
    - 过载，加班，死亡行军，质量低下，需求变更项目后期代价高
  - 按团队容量和实际进展来不断调整计划
    - 相对的以不变应万变， 快速开始完成，停止不断开始新的开发
- **欢迎需求变更即使是在项目的后期**
  - 避免前期大量的详细计划
- **检验项目的实际进展和风险管控**
  - 传统的基于阶段式的检查点管控不能很好的降低风险
  - 实际的可工作的软件是检验进度的唯一标准
- **承诺计划的时间里程碑而不是范围**
  - 永远没有延期的说法



约束——③范围

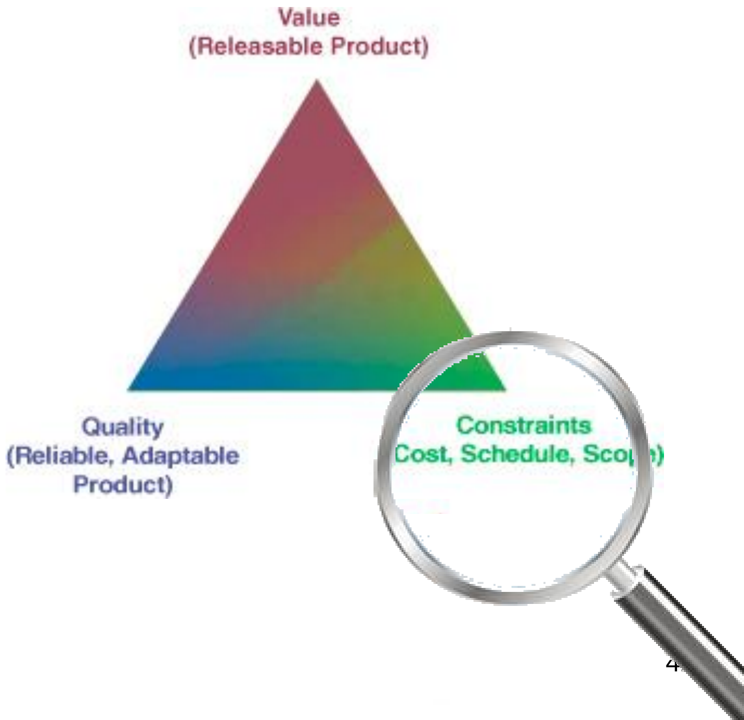
- **无论何种流程，大项目通常意味着失败**

”The Standish Group has categorically stated with much conviction—backed by intense research—that the secret to project success is to **strongly recommend and enforce limits on size and complexity.”**
”These two factors trump all other factors.”

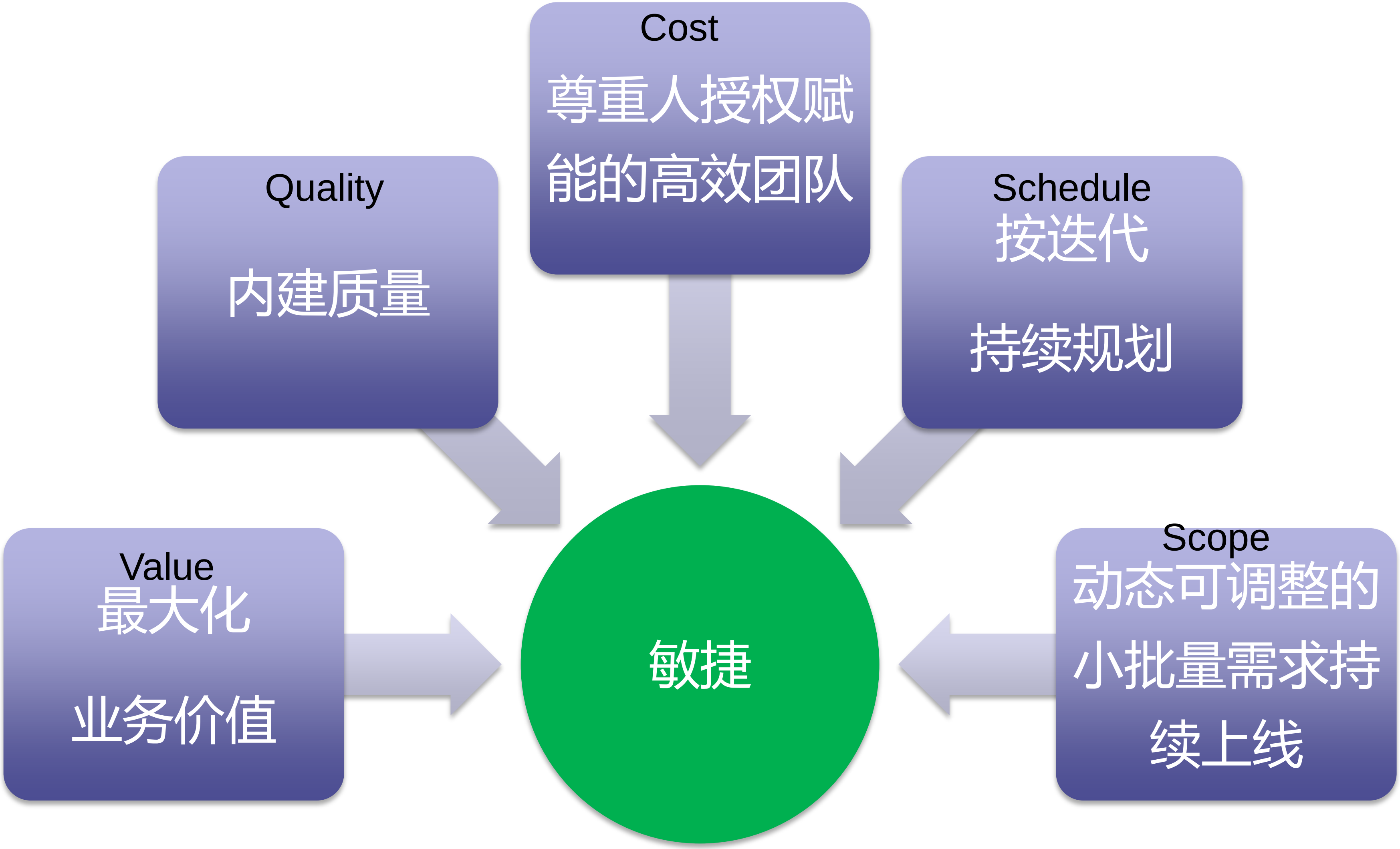


约束——③范围

- **根据业务目标的适应性范围调整**
  - 将大的需求拆小，并按优先级管理，持续小批量分析和开发
  - 小批量需求的持续开发，使得需求变化，对已开发的需求影响和代价小
  - 新增需求可以排队或者替换已有需求
  - 小批量需求，可以被尽早测试，确认和验收，提高反馈速度
  - 不断打磨和调整产品方向



IoT时代的管理新范式



Cost
尊重人授权赋能的高效团队

Quality
内建质量

Schedule
按迭代
持续规划

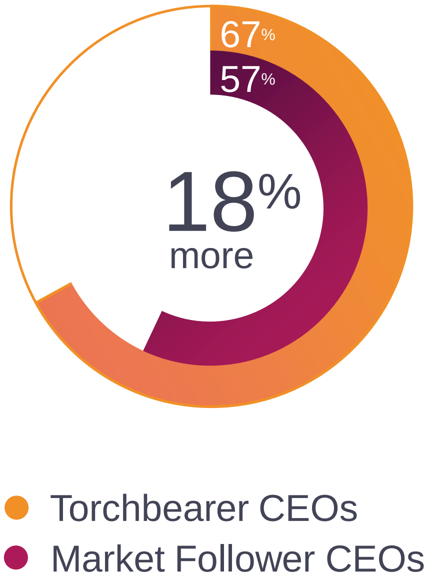
Value
最大化
业务价值

敏捷

Scope
动态可调整的小批量需求持续上线

Torchbearer CEOs believe that agility is essential to rapid, disruptive innovation

Agility for successful innovation



"We need to shorten our new service development cycle and become more agile. As soon as we identify a trend, we should prepare a prototype, gain customer feedback and refine it continuously."

Yutaka Nagao, CEO, Yamato Transport, Japan

Redefining Competition

Insights from the Global C-suite Study – The CEO perspective

Agility is essential to rapid, disruptive innovation

Speed

Experiment extensively and aggressively
Adopt an agile approach in all aspects of innovation and execution
Make it to the finish line first

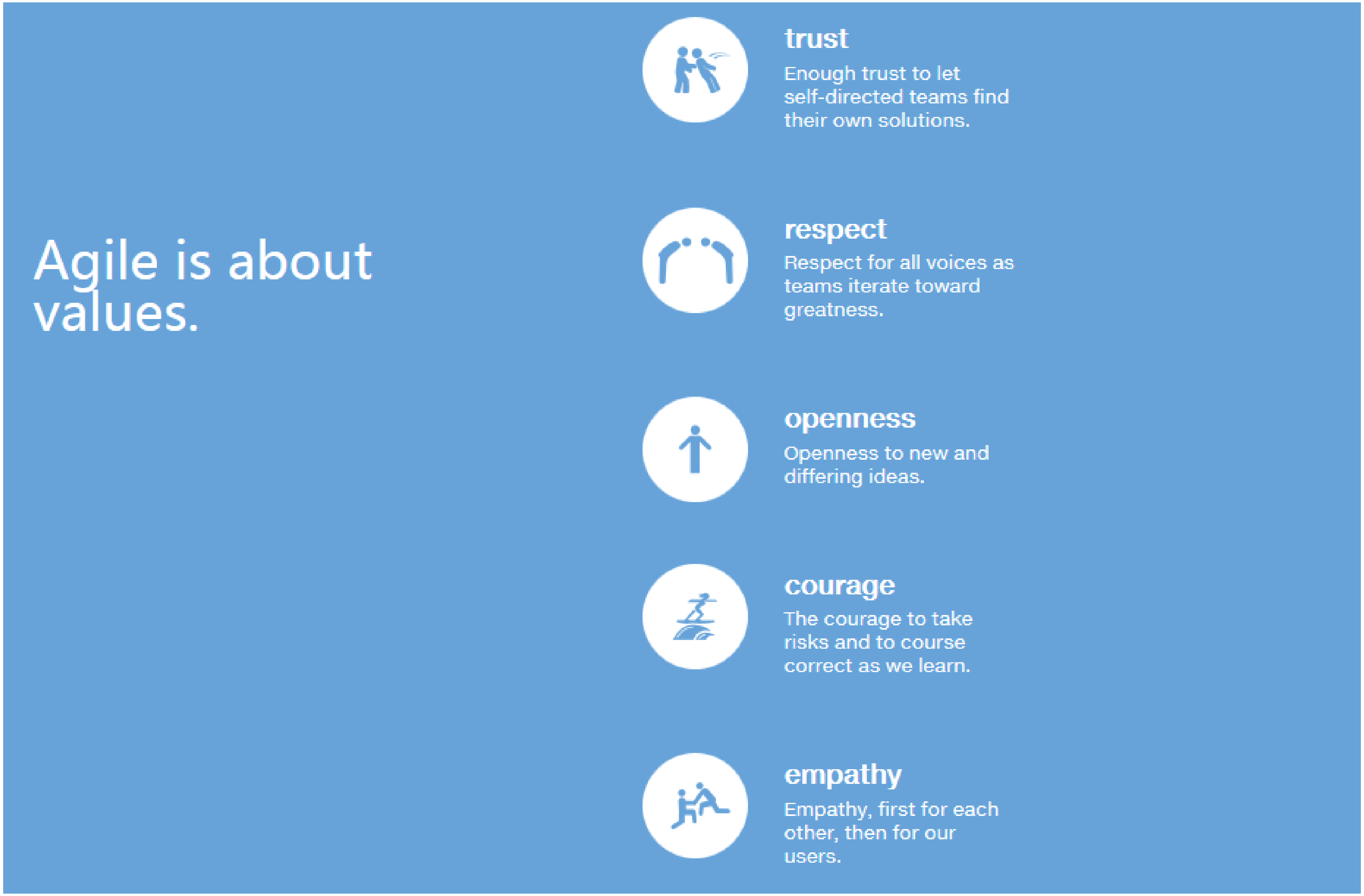
Redefining Boundaries
Insights from the Global C-suite Study

IBM Institute for Business Value

精益敏捷企业

整个组织的敏捷性

IBM Agile - Values



Agile is about values.

trust
Enough trust to let self-directed teams find their own solutions.

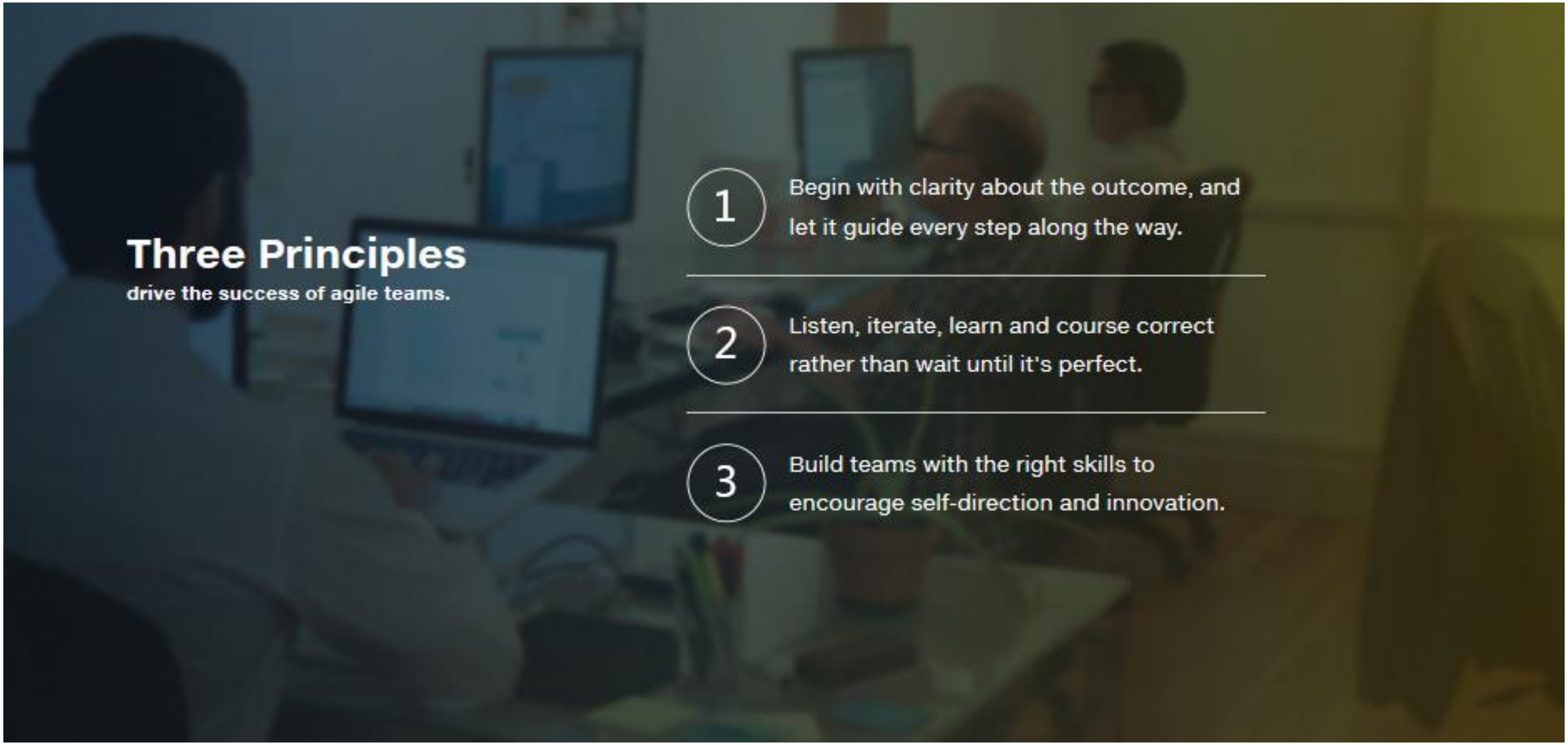
respect
Respect for all voices as teams iterate toward greatness.

openness
Openness to new and differing ideas.

courage
The courage to take risks and to course correct as we learn.

empathy
Empathy, first for each other, then for our users.

IBM Agile - Values



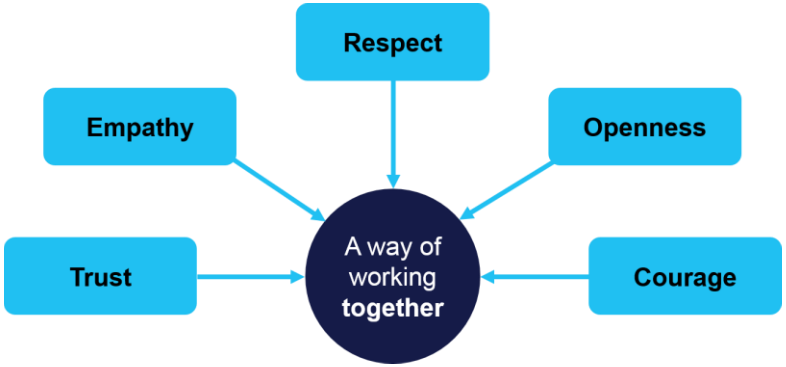
Three Principles

drive the success of agile teams.

1. Begin with clarity about the outcome, and let it guide every step along the way.
2. Listen, iterate, learn and course correct rather than wait until it's perfect.
3. Build teams with the right skills to encourage self-direction and innovation.

IBM Agile

Values



Respect

Empathy

Openness

Trust

Courage

A way of working together

Principles

Clarity of outcome

Iteration and learning

Self-directed whole teams

Methods

IBM Design Thinking

Agile Methods: Scrum, XP, SAFe etc.

IBM DevOps

IBM Bluemix Garage Method

Core Practices

Whole-team approach, Visual management, Roadmapping, Hills, Stand-ups, Playbacks, Retrospectives, Backlog prioritization, User research, Competitive research, Sponsor users, Experience mapping, Prototyping, Story writing, Relative size estimation, User feedback, Automated testing, Code reviews, Continuous delivery, User analytics, User Story, Daily Standup, TDD, ATDD, Agile Architecture, Beyond Budgeting, SoS, Agile Release Train

IBM

Agile Devops COE

GCG GBS CTO Office

Nourishing Technical Vitality for Better Client Innovation