



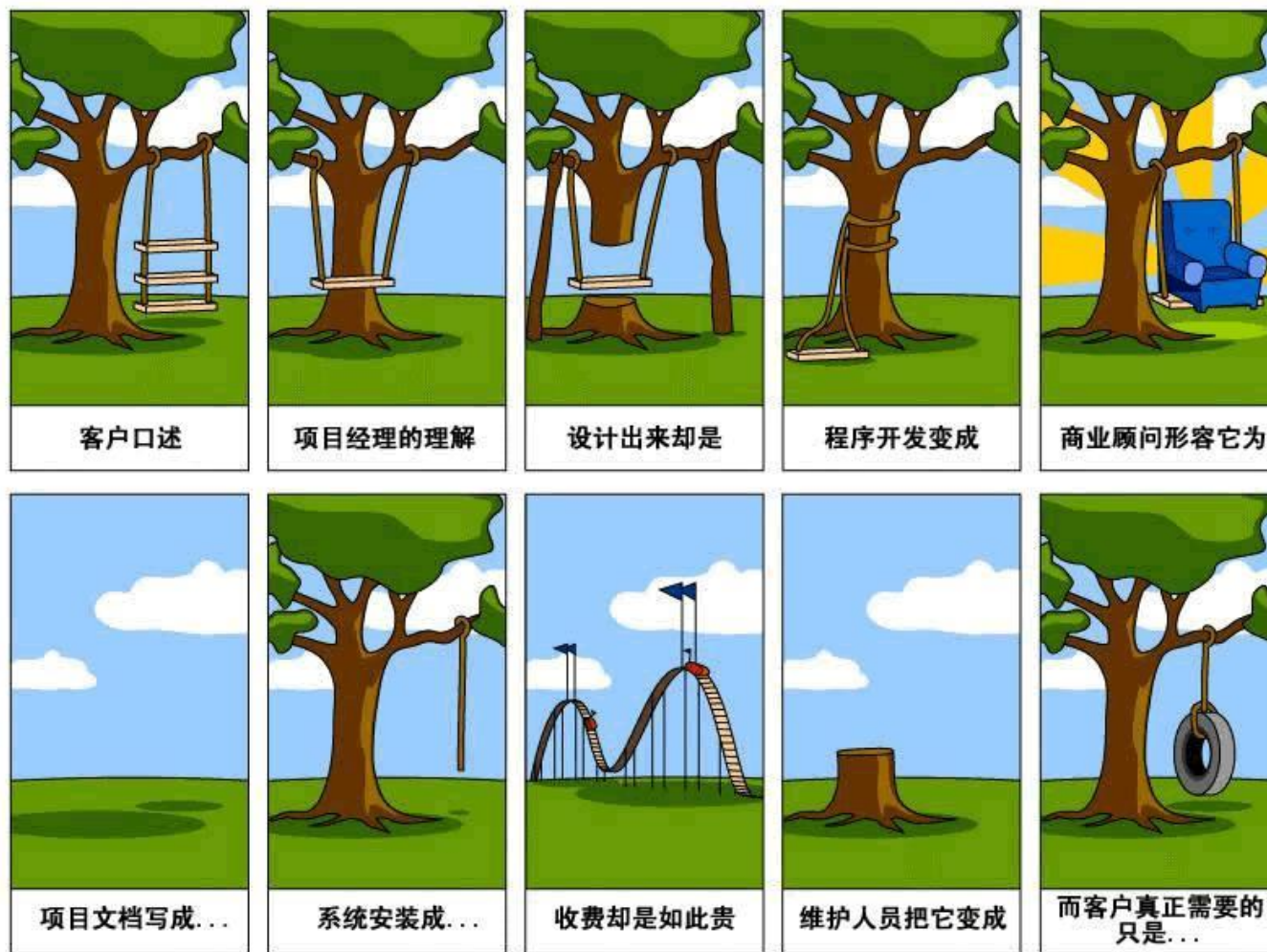
# 需求获取与用户故事

---

清华大学软件学院 刘强

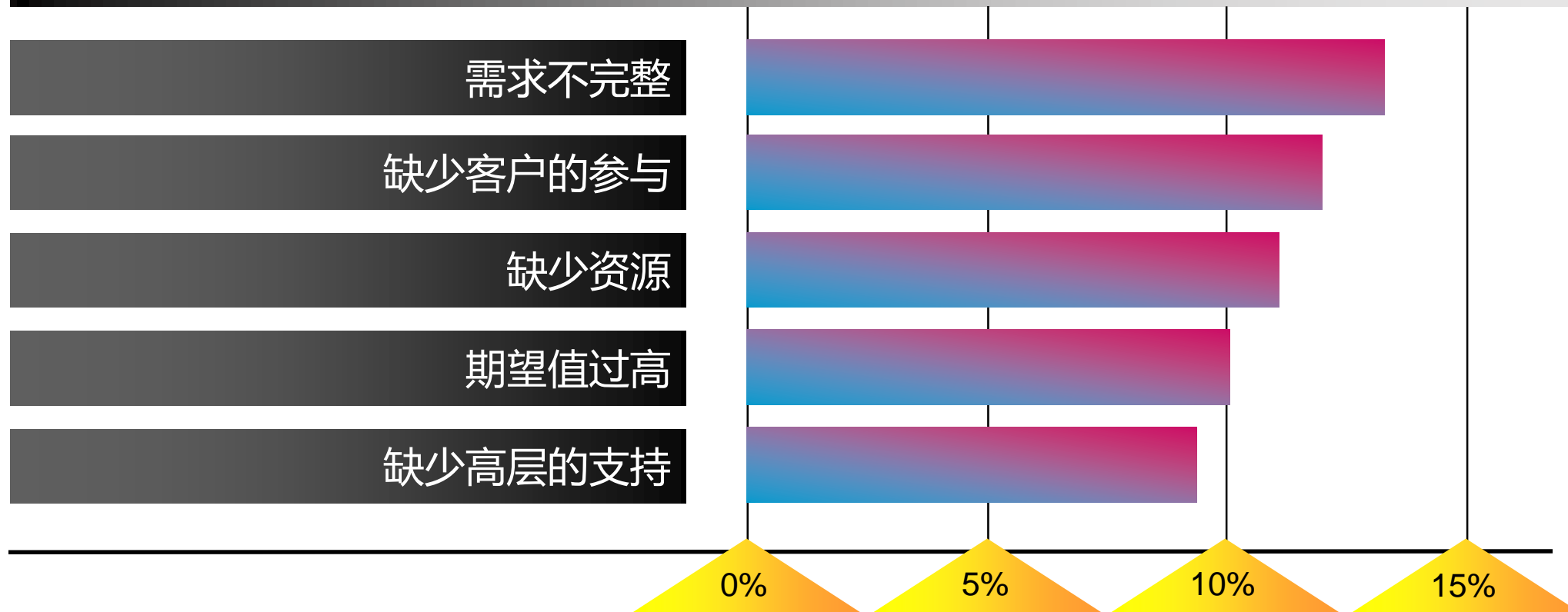


# 需求的重要性



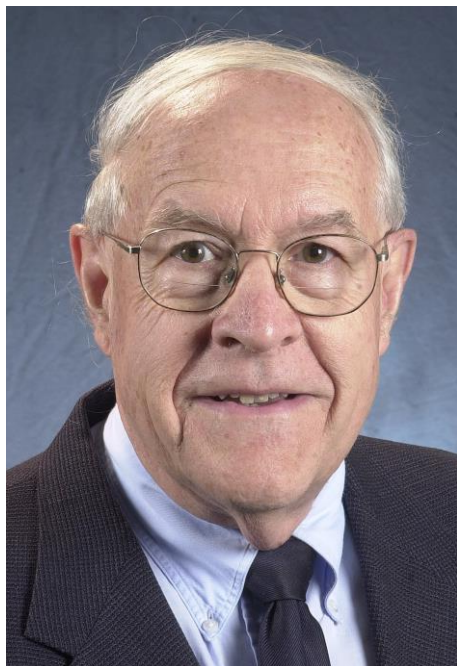
# 需求的重要性

## 软件项目失败的最重要的五个原因



# 需求的重要性

---



建造一个软件系统最困难的部分是确定要建造什么。在概念性工作中，没有其他任何一个部分比确定详细的技术需求更加困难，包括所有的人机界面、与机器和其他软件系统的接口。需求错误对最终系统的影响比其他任何部分的失误都要大，而修补需求也比其他任何部分更困难。

—— Fred Brooks



1

需求的概念与类型

2

需求获取与分析

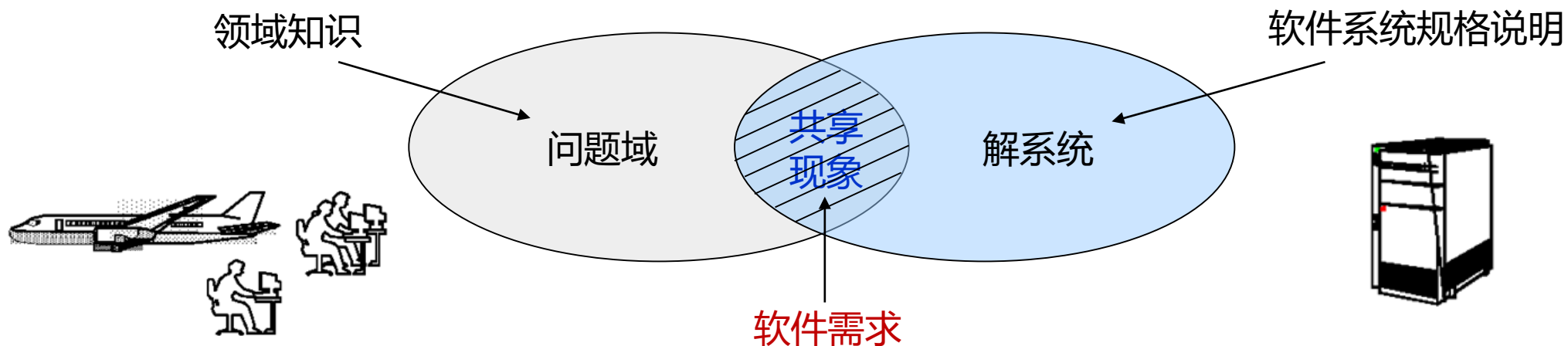
3

用户故事地图

# 需求的定义

软件需求的定义 [IEEE, 1997] :

- ① 用户解决问题或达到目标所需的条件或能力。
- ② 系统或系统部件要满足合同、标准、规范或其它正式规定文档所需具有的条件或能力。
- ③ 一种反映上面 ① 或 ② 所描述的条件或能力的文档说明。



# 需求的定义

---

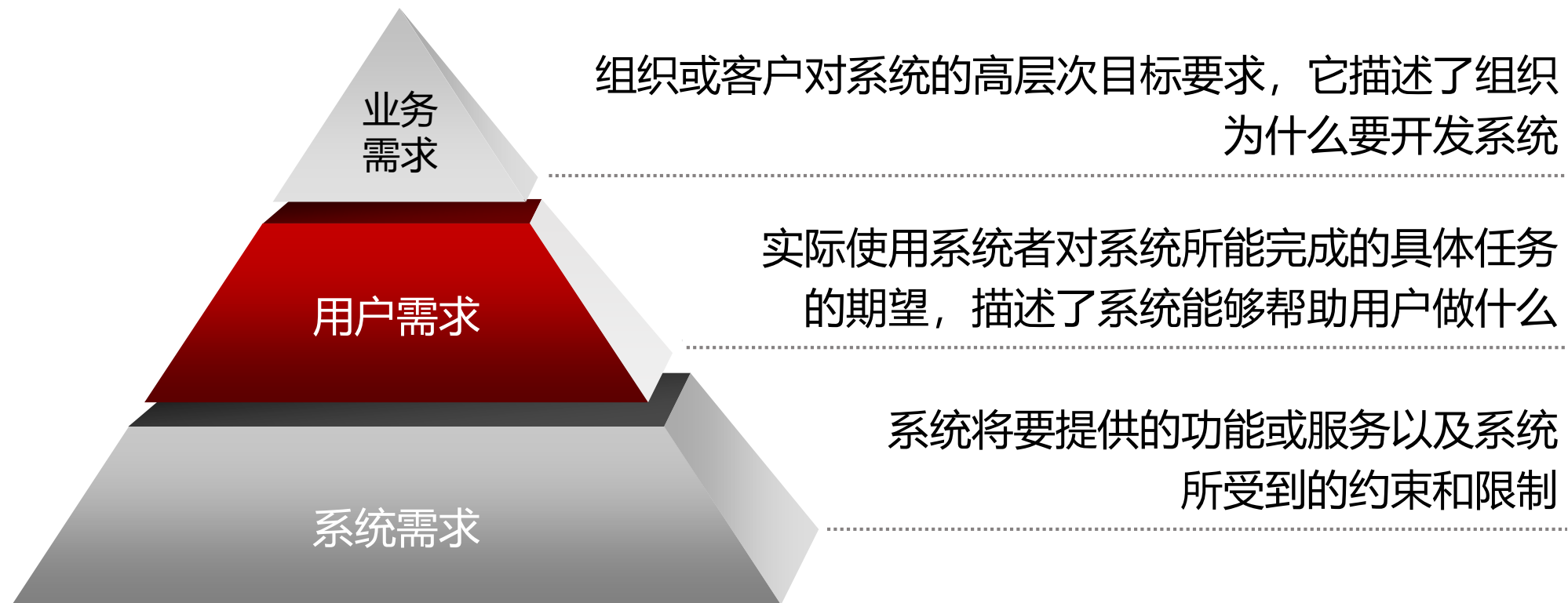
关于需求的两种描述：

- 需求是客户希望在问题域内产生的效果
- 需求是系统为解决问题或完成目标所必须满足的条件或能力

举例：

- 每当电梯停在某一楼层时，电梯门将在开启、等待、关闭三个状态中循环。
- 每当电梯停在某一楼层时，系统将使电梯门在开启、等待、关闭三个状态中循环。

# 软件需求的层次





# 业务需求

**业务需求**是系统建立的战略出发点，表现为高层次的目标。为了满足用户的业务需求，需求工程师需要描述高层次的解决方案，定义系统应具备的特性。



清华大学已成为节假日参观的热门校园景点，原有排队集中进校的管理模式简单粗放，校园秩序受到干扰。学校保卫处希望建设校园预约参观系统，由过去的粗放管理模式向人性化、精准化服务模式转变，通过移动终端的预约参观流程提高对进校人员数量、时间以及校内活动范围的约束，努力做到满足社会公众参观游览校园需求的同时加强管控，将使校园秩序得到整体提升。

# 用户需求

用户需求通常用自然语言、图表和直观的图形进行描述，只描述系统的外部行为。



- 保卫处可以根据校历缺省设置每个学期的参观时段和人数，并且可以临时修改具体日期的设置。
- 游客可以查看一周之内的每天开放情况以及开放参观日期的当前预约情况。
- 在有预约名额的情况下，游客可以选择具体的开放参观日期进行参观预约。
- 预约成功之后，游客在规定的预约时间持有效证件进校。

# 系统需求

系统需求通常使用更专业化的符号系统进行描述，包括结构化语言、需求的图形模型、形式化数学描述等。



1. 在有预约名额的情况下，游客可以选择具体的开放参观日期进行参观预约。



- 1.1 系统应查询游客是否经过实名认证，没有通过认证不允许预约。
- 1.2 系统应查询游客在两周内是否已经有未取消的预约记录。
- 1.3 系统允许游客输入两个同行人的信息（包括姓名和身份证号），并检查同行人的信息是否重复预约以及证件号码是否有效。
- 1.4 在游客身份合法且允许预约的情况下，系统为该游客生成一个新的预约记录，并更新相应日期的余额数量。

# 软件需求的类型

---

- **功能需求：**系统应该提供的服务、如何对输入做出反应以及系统在特定条件下的行为描述。
- **质量需求：**也称非功能需求，描述软件系统必须拥有的质量特性，诸如快速的响应时间、易使用性、高可靠性或低维护代价等。
- **设计约束：**设计约束是已经做出的设计决策或限制问题解决方案集的设计决策，例如平台或构件接口的选择。
- **过程约束：**过程约束是对用于构件系统的技术和资源的限制，例如客户要求使用敏捷开发方法。

# 软件需求的类型

## 功能需求

### 功能

- 系统将做什么？
- 系统什么时候做？
- 有多种操作模式吗？
- 必须执行什么种类的计算或数据转换？
- 对可能刺激的合适反应是什么？

### 数据

- 输入、输出数据的格式应该是什么？
- 在任何时间都必须保留任何数据吗？

## 设计约束

### 物理环境

- 设备安装在什么地方？
- 在一个地方还是多个地方？
- 是否有任何环境的限制，如温度、湿度或者电磁干扰？
- 是否对系统的规模有所限制？
- 是否在电源、供热或空调上有所限制？
- 是否会因为现有的软件而对程序设计语言有所限制？

### 接口

- 输入是来自一个还是多个其他系统？
- 输出是否传送到一个或者多个其他系统？
- 输入 / 输出数据的格式是否是预先规定的？
- 数据是否必须使用规定的介质？

### 用户

- 谁将使用系统？
- 将会有几种类型的用户？每类用户的技术水平如何？

## 过程约束

### 资源

- 开发系统需要哪些材料、人员或其他资源？
- 开发人员应该具有怎样的技能？

### 文档

- 需要多少文档？
- 文档是联机的、印刷的，还是两种都要？
- 每种文档针对哪些读者？

### 标准

- 系统或开发过程应遵循的标准是什么？

# 软件需求的类型

## 质量需求

### 性能

- 有没有执行速度、响应时间或吞吐量的约束？
- 使用什么效率测量方法测量资源使用和响应时间？
- 多少数据将流经系统？
- 数据接收和发送的时间间隔是多少？

### 易用性和人的因素

- 每类用户需要什么类型的培训？
- 用户理解并使用系统的难易程度如何？
- 就一个系统而言，在多大程度上防止用户误用系统？

### 安全性

- 必须控制对系统或信息的访问吗？
- 应该将每个用户的数据或其他用户的数据隔离开吗？
- 应该将用户的程序和其他程序以及操作系统隔离开吗？
- 要采取预防措施以防止盗窃或蓄意破坏吗？

### 精度和准确性

- 数据计算的准确度要求是多少？
- 计算的精确度要达到什么程度？

### 可靠性和可用性

- 系统需要检测并隔离故障吗？
- 规定的平均失效间隔时间是多少？
- 失效后重新启动系统允许的最大时间是多少？
- 系统多久备份一次？
- 备份副本要存放在不同的地方吗？
- 要采取预防措施以防止水灾、火灾吗？

### 可维护性

- 维护仅仅是修正错误？或者还包括改进系统？
- 系统可能会在将来的什么时候以什么方式被改变？
- 给系统增加特性的难易程度如何？
- 从一个平台向另一个平台移植系统容易吗？

### 交付时间和成本

- 有预先规定的开发时间表吗？
- 花费在硬件、软件或开发上的资金有限制吗？



1

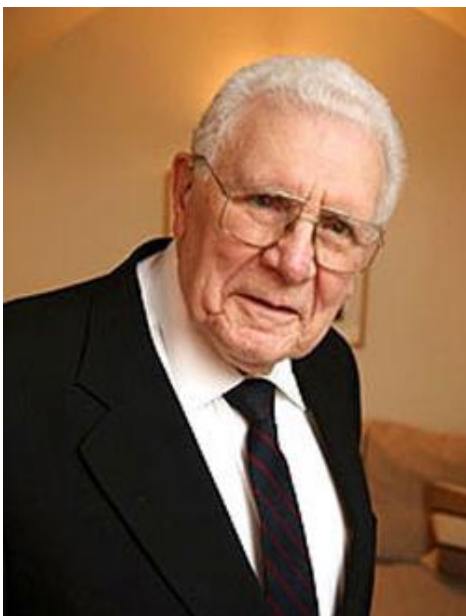
需求的概念与类型

2

需求获取与分析

3

用户故事地图



Doing The Right Things !  
Doing The Things Right !

—— Peter F. Drucker

选择了错误的方向而又正确地执行它，将是一件可怕的事情！



# 需求获取

---

需求获取是需求工程中非常关键的一部分，开发人员需要与客户和用户一起理解所要解决的问题，明确客户和用户到底想要什么。

有人说：获取需求不是很简单吗？让用户告诉我们就是啦！

你认同这种说法吗？你曾经遇到过什么问题？

# 举例：校团委微信应用

## 服务号可以实现的基本功能（包括但不限于此）：

- 搜索引擎式的校内资讯查询服务。例如：输入“社团部”，会返回社团部相关介绍；输入“吉他协会”，会返回吉他协会相关介绍；输入“时事大讲堂最新讲座信息”，会将时事大讲堂的最新活动信息返回给同学。
- 热门活动的抢票服务。例如：除排队领票外，可以开放一部分门票给线上抢票，用户根据学号和姓名等信息进行抢票，后台立即提示是否抢票成功，之后再开放一个集中时间段供同学凭学生证来领票。
- 大家可以录入自己的院系、年级、社团、兴趣爱好等信息，基于这些信息进行个性化的信息推送，并根据这些信息调整查询结果的内容顺序。
- 根据大家的点击行为、查询行为猜测大家的兴趣，并调整推送信息的内容和查询内容的排序。
- 了解不同兴趣点的相关性，了解不同活动参与人群的分布，为活动组织提供数据指导，为新的活动与资讯定点推送到热点人群。

# 举例：校团委微信应用

## 服务号可以实现的基本功能（续）：

- 对不同用户群定向投放调查问卷。例如：青研中心想向体育爱好者投放关于“改善体育场地夜间照明开放时间”的调查问卷，后台可以根据用户之前的点击和查询行为，定点向经常关注体育类相关信息的同学，而不会打扰到其他同学。
- 简单的聊天机器人服务，学习新式词语并给大家提供基本娱乐。（类似图书馆推出的小图）
- 根据每位同学的兴趣和历史行为向大家推荐非时效性内容，如文章、音乐、图片等，形成每个人默认的兴趣频道。
- 实现简单的日历提醒功能。例如：用户输入：“下午五点开会”，那么服务号可以提前30分钟，10分钟，5分钟分别推送一条提醒信息。
- 用户定向反馈功能。例如：用户输入“志愿中心：错过了献血时间怎么办？”，后台将此信息反馈给志愿中心相关联络人员，之后将解决方案反馈给用户。

# 举例：校团委微信应用

## 服务号可以实现的高级功能（包括但不限于此）：

- 收集校园活动的全媒体资料，存放成频道以供大家查询。例如，校内各报刊的图文内容、校歌赛金曲合集、129合唱比赛原声、校园微电影节精选节目都可以形成频道。
- 各院系也可以自己组织提交频道，频道可以是单一媒体形式的，也可以是丰富多彩的。
- 后台利用现有的工业化语音识别引擎，可以尝试进行语音查询服务。

## 益智类游戏功能（包括但不限于此）：

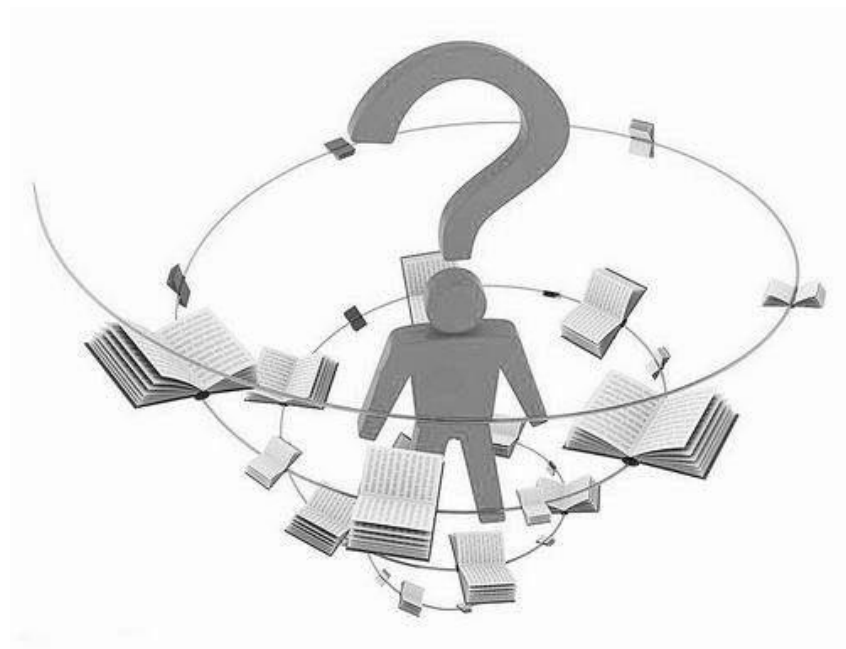
- 利用后台提供益智小游戏，例如可以模仿一站到底，让大家轮流答题PK。题库可以用校内资源，如实验室安全知识竞赛题库等。

## 与其他系统绑定（基于申请情况，包括但不限于此）：

- 可以与教务系统捆绑，让大家能够利用文字消息和语音方便地查询到课程信息、课表信息，下载课程文件，查询成绩等。
- 可以尝试和更多系统联网，例如说一句“打印课程表”，持学生卡就能在校图书馆上打印出课程表。

# 需求获取的挑战

- 问题的复杂性与问题空间
- 理解的不完备性与不一致性
- 用户不知道需要什么，或不知道如何表达
- 开发人员自认为了解用户的需求
- 需求是不断变化的



# 需求获取的挑战



如何理解乔布斯说的「消费者并不知道需要什么，直到我们拿出自己的产品，他们就发现，这是我要的东西」？

乔布斯曾表示“消费者并不知道需要什么，直到我们拿出自己的产品，他们就发现，这是我要的东西”。他的成功证明他的想法是正确的，但一个成功的产品一定是要迎合消费者的需求，而乔布斯的这句话是建立在“消费者不知道自己需要什么”的基础之上，是否与之相悖？如何理解？

人们永远也无法告诉你，他的隐性需求是什么？

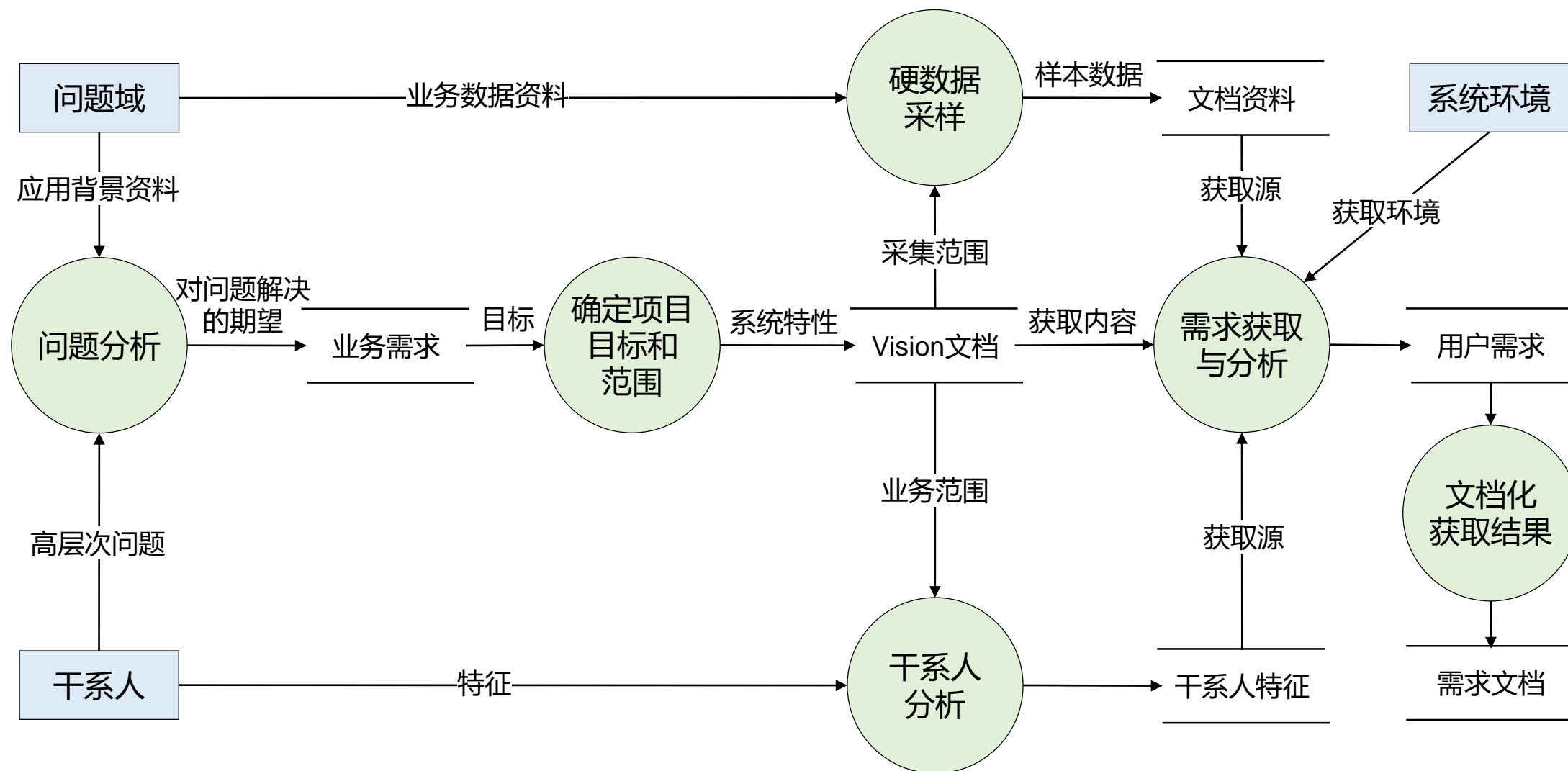
- 很难通过简单的访谈和问卷调查去获取隐性需求

如何定义痛点假设和解决方案假设？



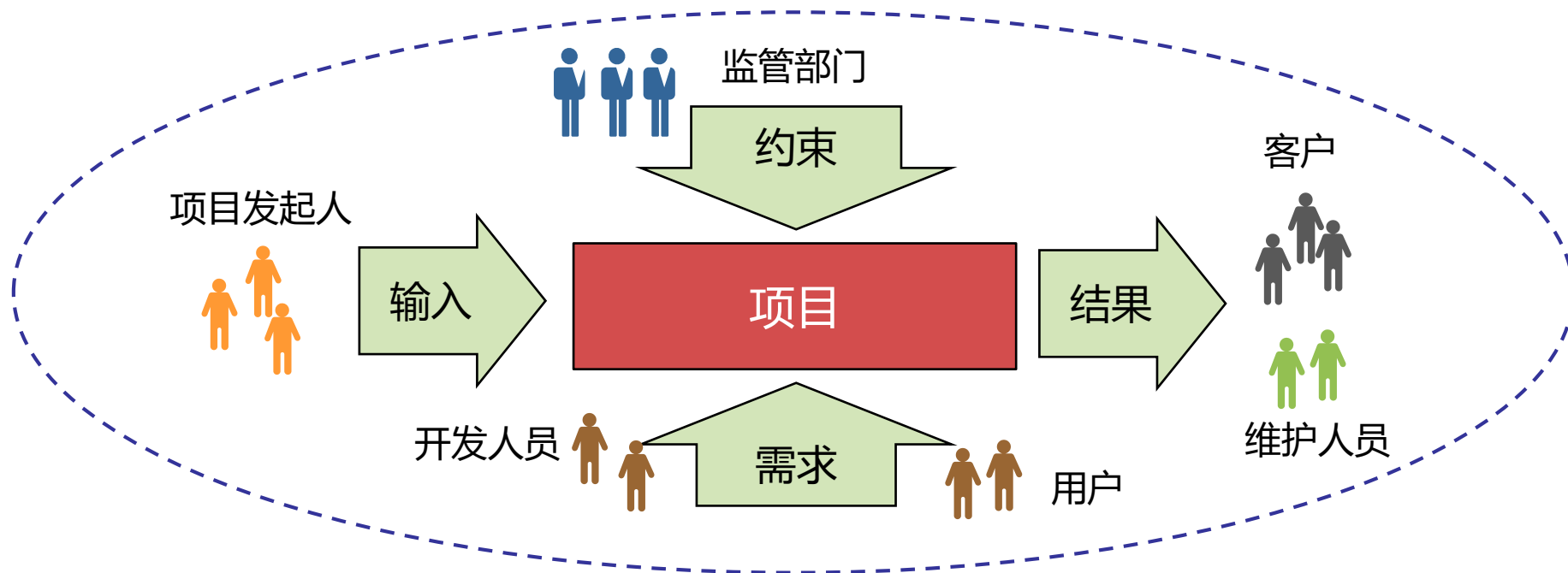
- 使用头脑风暴，开发团队带着强烈的同理心去体验用户体验。
- 参与性的观察和深度访谈，深入观察用户，与用户进行大量的互动，甚至把自己变成用户的一分子。
- 从别人的失败中学习。

# 需求获取过程



# 干系人分析

项目干系人 (Stakeholders) 是指那些与项目有利害关系的个人或组织，他们能够给项目的发展带来潜在的正面或负面影响。



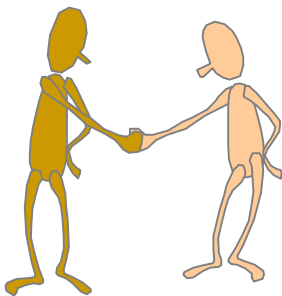


# 干系人分析



用户	职责与特征
学校保卫处	学校保卫处是项目的发起人，承担学校安全管理工作的职能部门，具体负责治安、消防、交通、保密、集体户口、校门守卫等工作。保卫处希望建设校园预约参观系统，以新型服务模式满足校园参观需求，同时加强管控使校园秩序得到整体提升。
保卫处工作人员	保卫处工作人员主要决定校园参观的时段和名额，对团队参观的组织资质和申请进行审核。这类人员熟悉保卫处的日常工作，应该参与需求获取、需求评审和系统评价等。
游客	游客系统能够方便预约校园参观，主要的人群是中年家长以及高中生或者大学生，多数可以熟练使用手机，经常使用微信平台进行联系和交流。

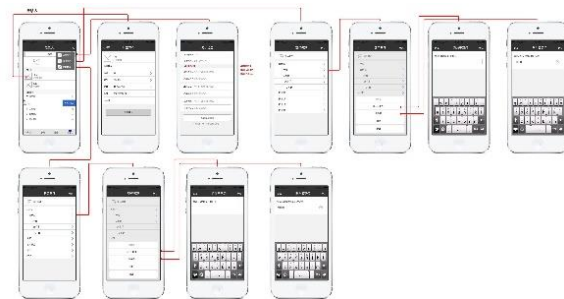
# 需求获取技术



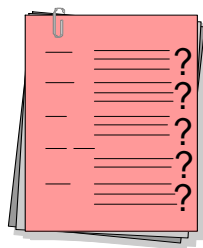
用户访谈



竞品分析



原型系统



问卷调查

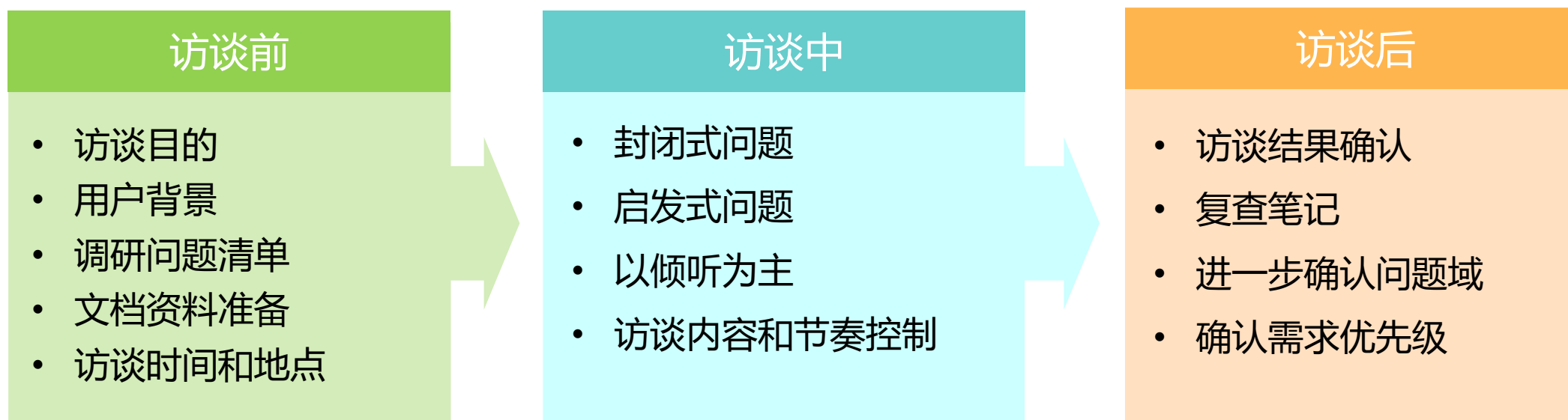


头脑风暴

# 用户访谈

用户面谈是最简单直接的一种需求收集方式，几乎适合任何商务场合。

- **利：** 直接有效、形式灵活、交流深入，应为主要的需求获取技术；
- **弊：** 占用时间长、面窄而且容易造成信息的片面性。



# 竞品分析

竞品分析是指对现有的或潜在的竞争产品的优劣势进行比较分析。

- 了解市场，看清市场的发展趋势，找准市场切入点；
- 了解对手，他山之石可以攻玉，同时发现潜在的竞争对手；
- 了解需求，把握需求对应的功能点和界面结构，并侧面了解用户习惯。

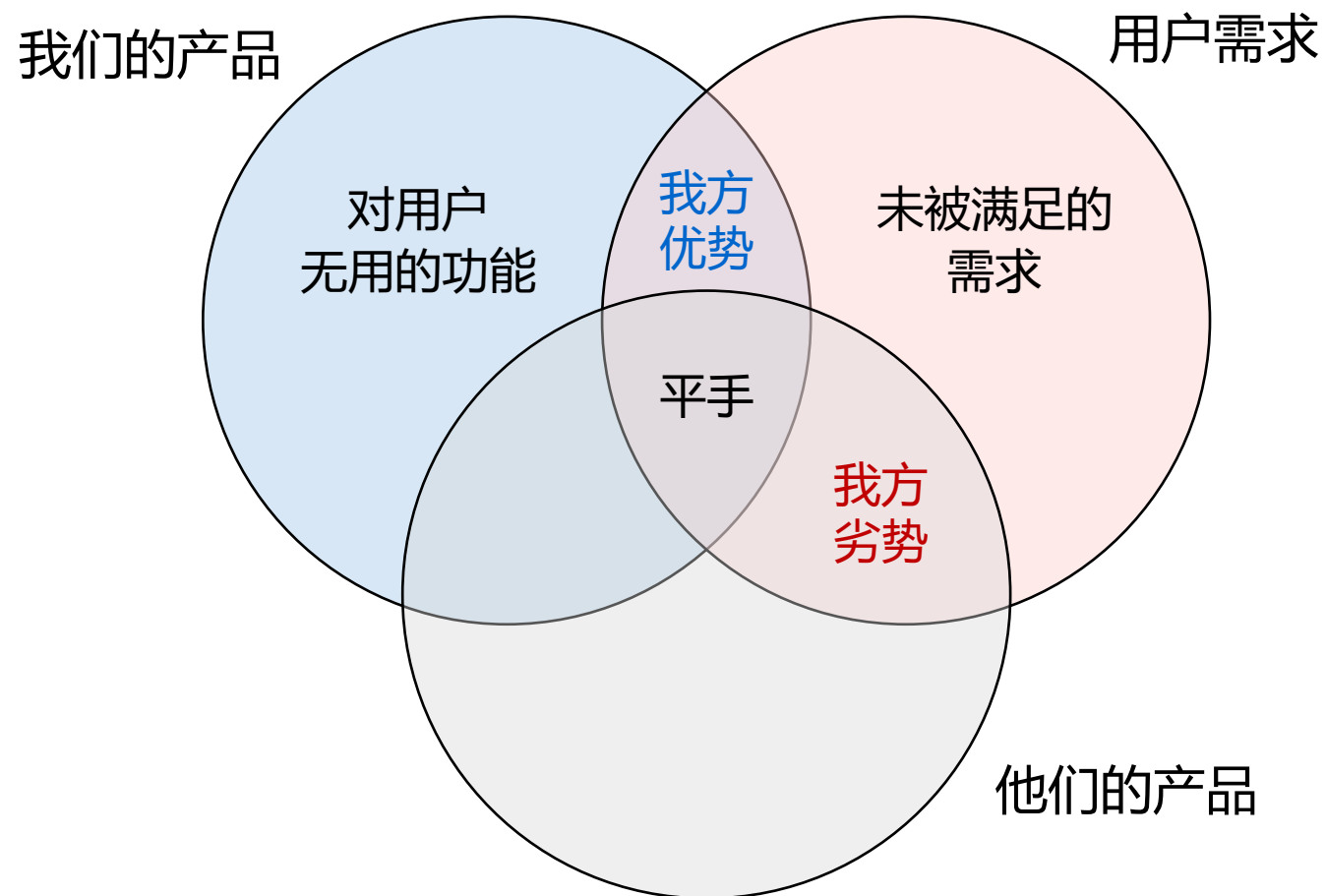
## 竞品选择

- 满足同样的需求
- 具有同样的用户群
- 选择2个左右的竞品
- 通过各种方式搜索

## 竞品分析

- 产品信息：产品定位、用户群、产品形态及产品特点等
- 用户分析：用户的类型和特征、核心需求或要求等
- 业务流程：产品覆盖的主要业务以及具体业务流程
- 页面分析：具体页面功能、信息展示方式和优缺点等
- 归纳总结：产品亮点和不足，可能的新想法或新创意

# 竞品分析



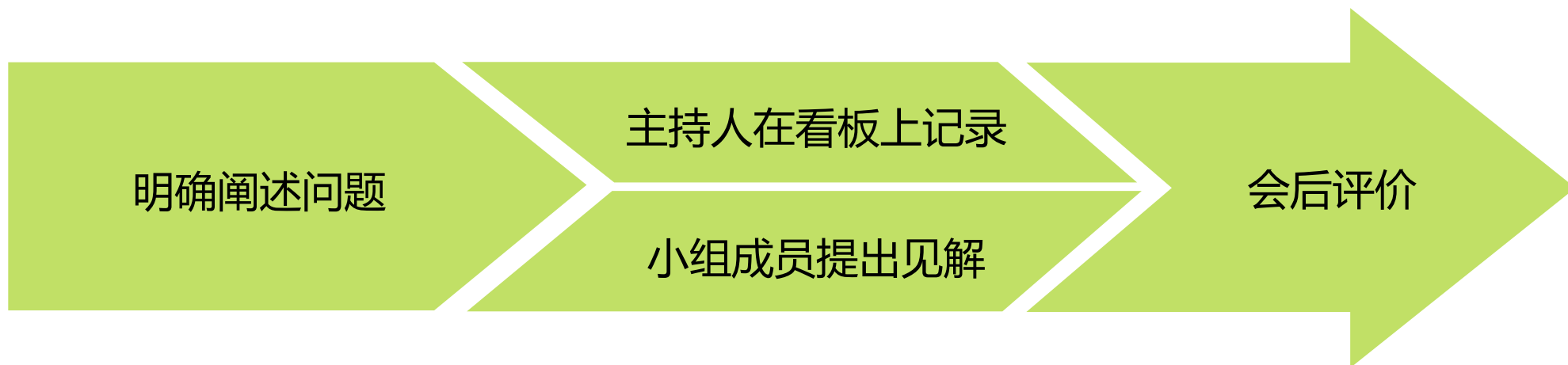
- 圆圈：产品功能
- 我方优势：我方产品有的独特功能满足用户需求
- 我方劣势：我方产品没有的功能
- 平手：双方都有的功能
- 未被满足的需求：用户需求未被任何产品满足

# 头脑风暴法

头脑风暴法

是用于产生和收集对项目需求与产品需求的多种创意的一种技术。

- 产生大量观点或可选方案的方法，要求参加者具有较高的联想思维
- 尝试充分运用所有员工的创造力，可以打破群体思维的方法

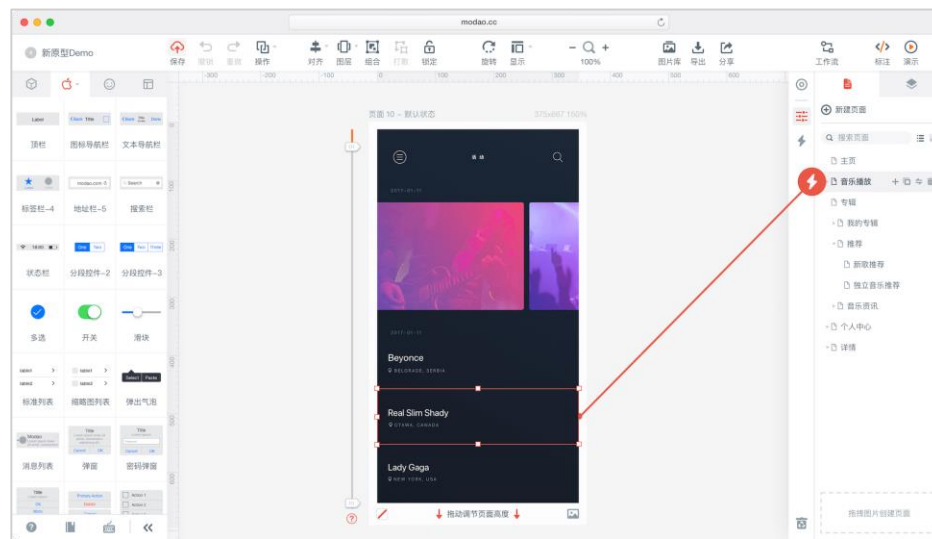


# 原型系统

**原型法**是在实际开发产品之前，先实现该产品的部分原型，并据此征求对需求的反馈意见，它比开发人员常用的技术术语更易于理解。



纸上原型设计示例



<https://modao.cc/>



### 预约参观页面：

- 显示当日开始之后八天的状态
- 状态：关闭、开放、已满
- 点击可预约日期，显示可入校时段
- 时段：9-10，10-11，14-15，15-16
- 点击同行人的按钮，允许输入同行人的信息
- 每个预约人最多可有两个同行人的
- 在选择日期和时段后，点击预约才进入确认预约的页面

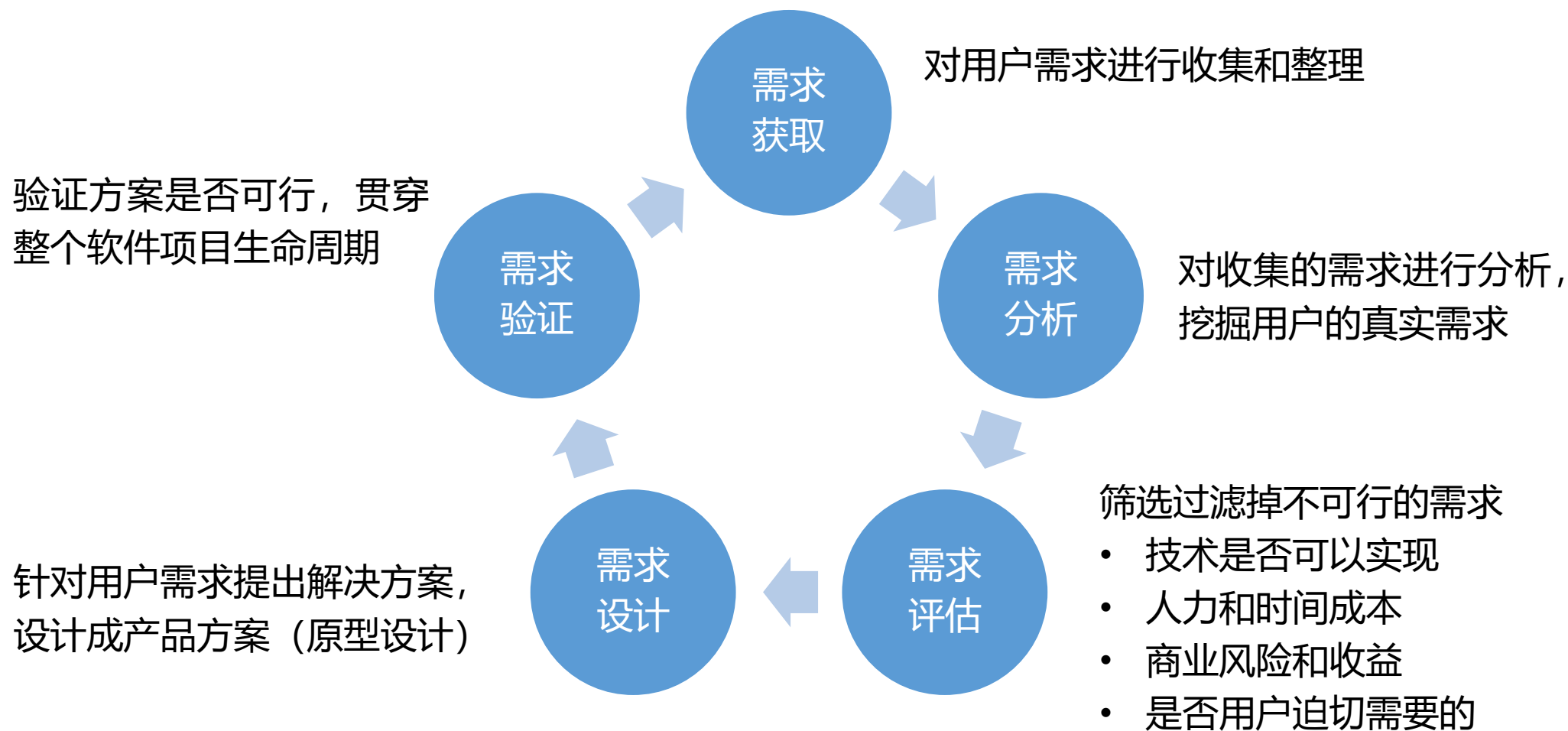


### 添加同行人的页面：

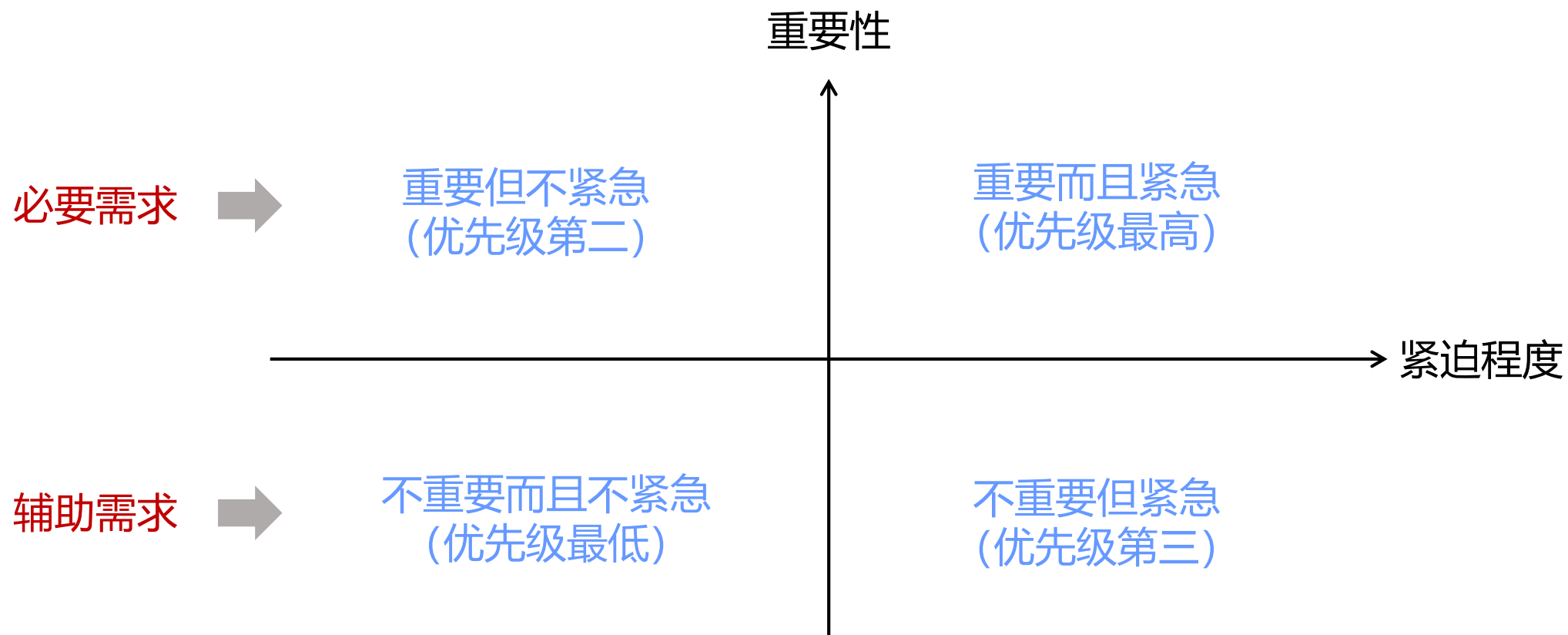
- 验证同行人的身份证号规则，但不进行姓名和身份证号匹配验证
- 验证当前页面的预约人和同行人的证件号是否重复



# 需求分析

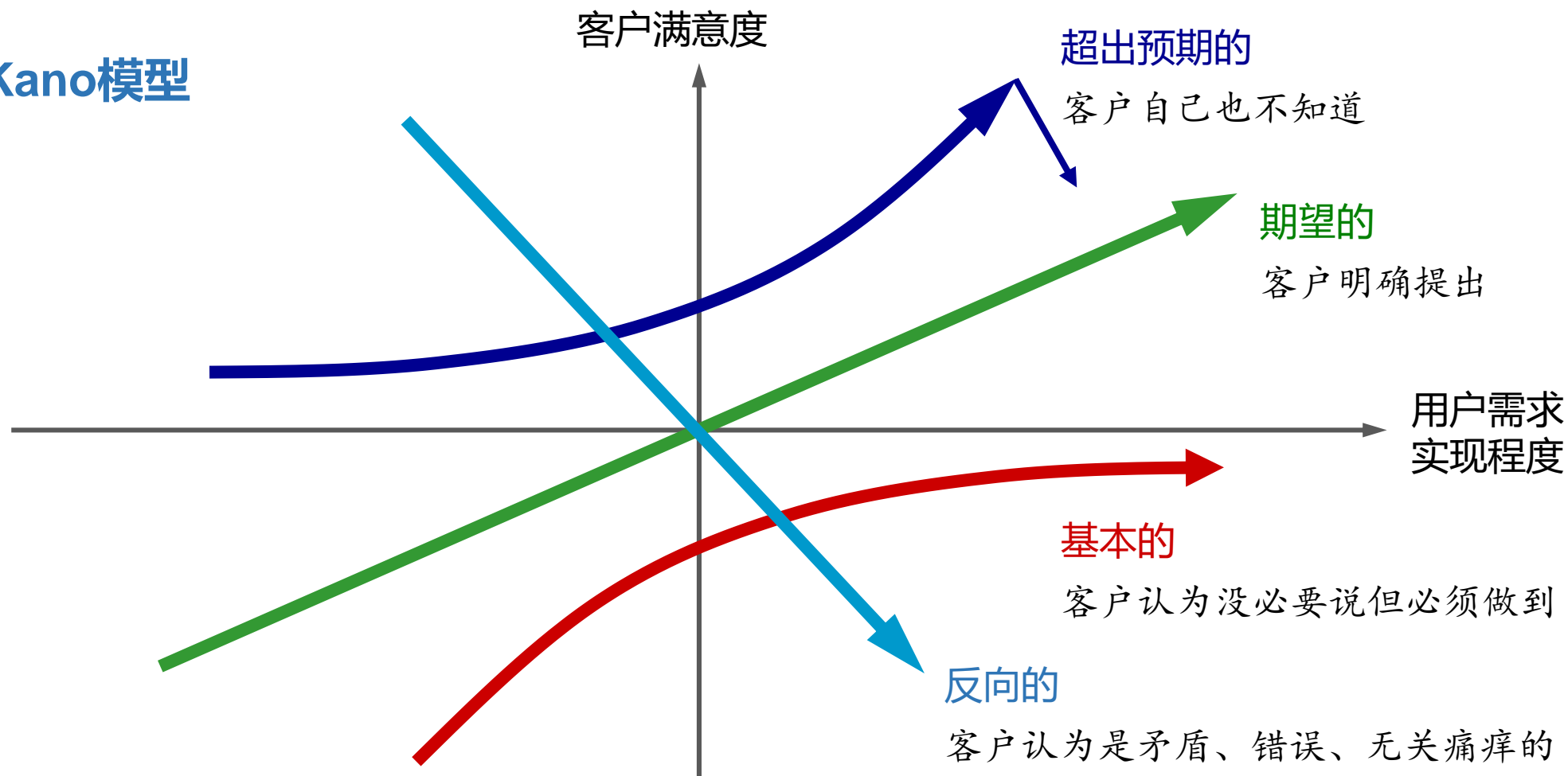


# 需求优先级



# 需求优先级

## Kano模型





1

需求的概念与类型

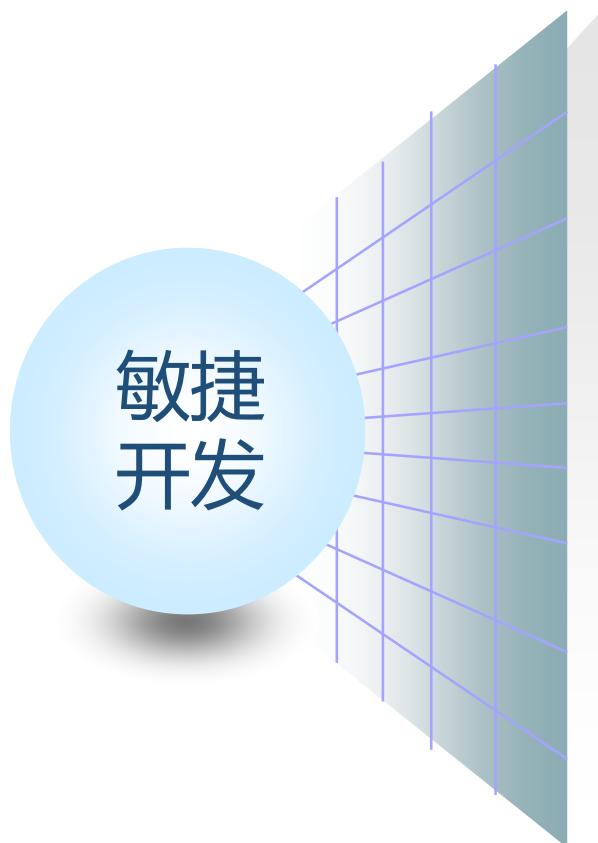
2

需求获取与分析

3

用户故事地图

# 敏捷开发方法



用户故事

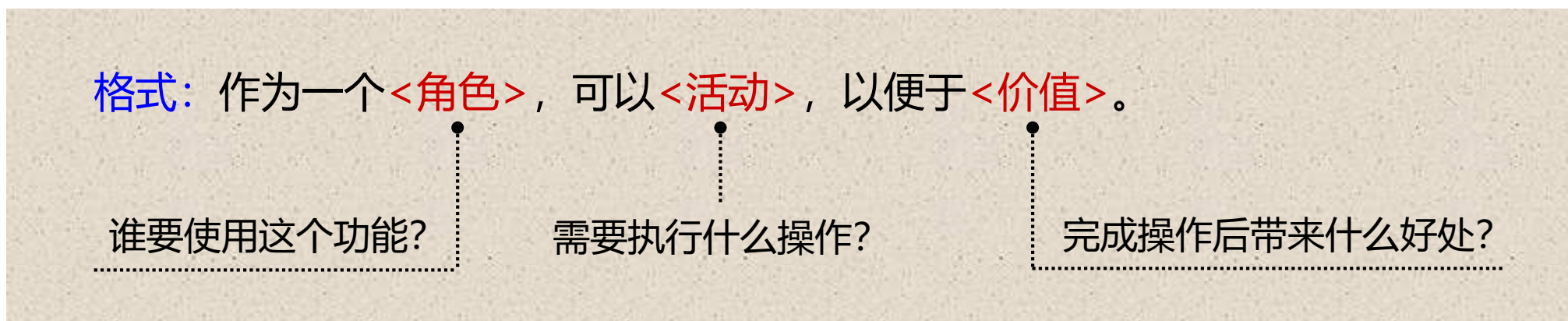
结对编程

测试驱动开发

持续集成

# 用户故事

用户故事（User Story）是从用户角度对功能的简要描述。



卡片：一份书面的故事描述

交流：有关故事的对话，用于具体化故事细节

确认：测试承载和记录了细节，可以用来检验用户故事是否完整

# 用户故事

---

- 用户故事是一种工具，用于以开发人员和用户都能理解的方式定义系统的行为。
- 用户故事将工作重点放在用户定义的价值上，而不是功能分解结构上。
- 用户故事以用户为中心，要求站在他人的角度进行思考并将他们的需求放在焦点上。
- 用户故事简单易懂，可以促进用户与开发人员之间的沟通。
- 用户故事代表了有价值的功能小增量，易于估算和灵活安排。
- 用户故事适合于迭代开发，可以写出一部分就开始开发与测试，然后按需求的节奏重复这个过程。可以先写出一个起始的目标层面及占位意义的故事，在这个故事对于开发进程变得重要时，才用更多对的细节来代替这个简单的描述。


# 如何写好用户故事

---


- **独立性**：尽可能避免故事之间存在依赖关系，否则会产生优先级和规划问题。
- **可协商**：故事是可协商的，不是必须实现的书面合同或者需求。
- **有价值**：确保每个故事对客户或者用户有价值的，最好是让用户编写故事。
- **可估算**：开发者应该能够预测故事的规模，以及编码实现所需要的时间。
- **短小的**：故事尽量短小，最好不超过10个理想人天，至少在一个迭代中完成。
- **可测试**：所编写的故事必须是可测试的。



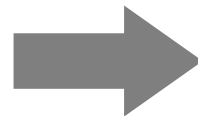
# 如何写好用户故事



作为管理员，我可以设置消费者的密码安全规则，以便消费者需要创建并保留安全密码，保证系统安全。



作为消费者，我需要遵循管理员设置的密码安全规则，以便我可以保持我的帐户的高安全性。



作为管理员，我可以设置密码有效期，以便用户被迫定期更改自己的密码。

# 如何写好用户故事



顾客可以使用信用卡购买购物车中的商品。

注释：接受Visa、Master和American Express信用卡。

# 如何写好用户故事



- 一个用户故事的内容要是可以协商的，它不是特定功能的合同。
- 一个用户故事只是对功能的简短描述，应避免太多的细节，具体细节在沟通时产生。用户故事如果带有太多的细节，容易限制用户、团队的想法和沟通，也会丧失调整实施的灵活性。

# 如何写好用户故事

作为一个玩家，可以通过显示排名，让自己在服务器中的地位获得认可。

- 激发玩家“斗志”，鼓励购买道具；
- 实现有技术问题：实时查看不现实；
- 小玩家对自己的排名不太关心，不会为了提升排名去购买道具，只有少数顶级大佬才会受此蛊惑。



作为一个排名靠前的付费玩家，可以通过显示排名，让自己在服务器中的地位获得认可（以刺激消费）。

系统每周重新排名一次，而且只显示前XXX名玩家。

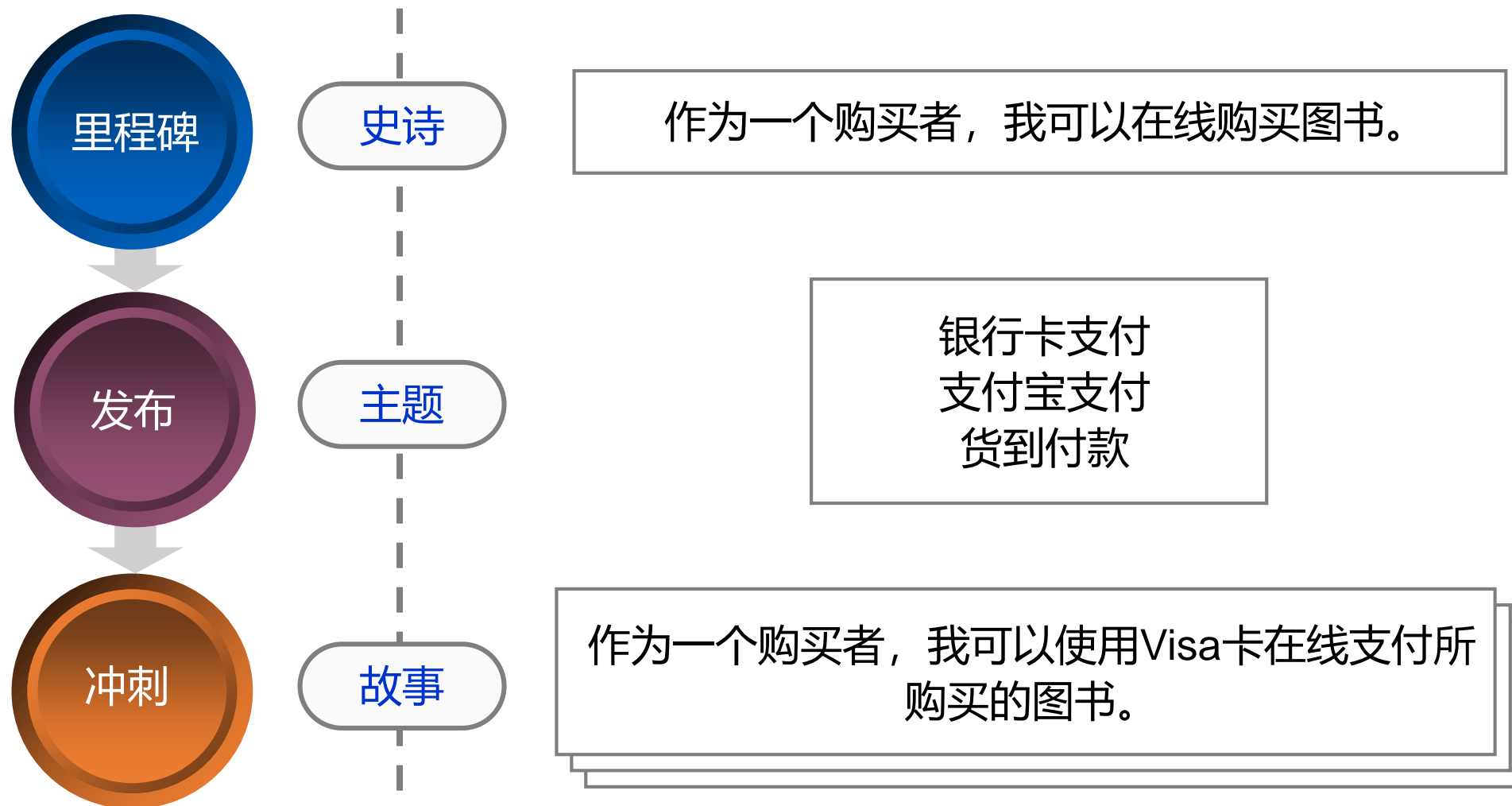
# 如何写好用户故事

顾客可以使用信用卡购买购物车中的商品。

注释：接受Visa、Master和American Express信用卡。

- 用 Visa、MasterCard和American Express进行测试（通过）
- 用 Diner's Club（大莱卡）测试（失败）
- 用 Visa借记卡测试（通过）
- 用正确的、错误的和空的卡号测试
- 用过期的卡测试
- 用不同限额的卡测试（包括超出银行卡的限额）

# 用户故事类型



# 用户故事分解

作为一个读者，我可以查询图书。

根据所处理的不同数据分解



用户可以按照多个  
关键字进行查询

作为一个读者，我可以  
按照书名查询图书。

作为一个读者，我可以  
按照作者查询图书。

作为一个读者，我可以  
按照图书类型查询图书。

# 用户故事分解

作为一个图书管理员，我可以维护图书的信息。

根据操作的类型进行分解



用户可以按照多个关键字进行查询

作为一个图书管理员，我可以增加一本新书。

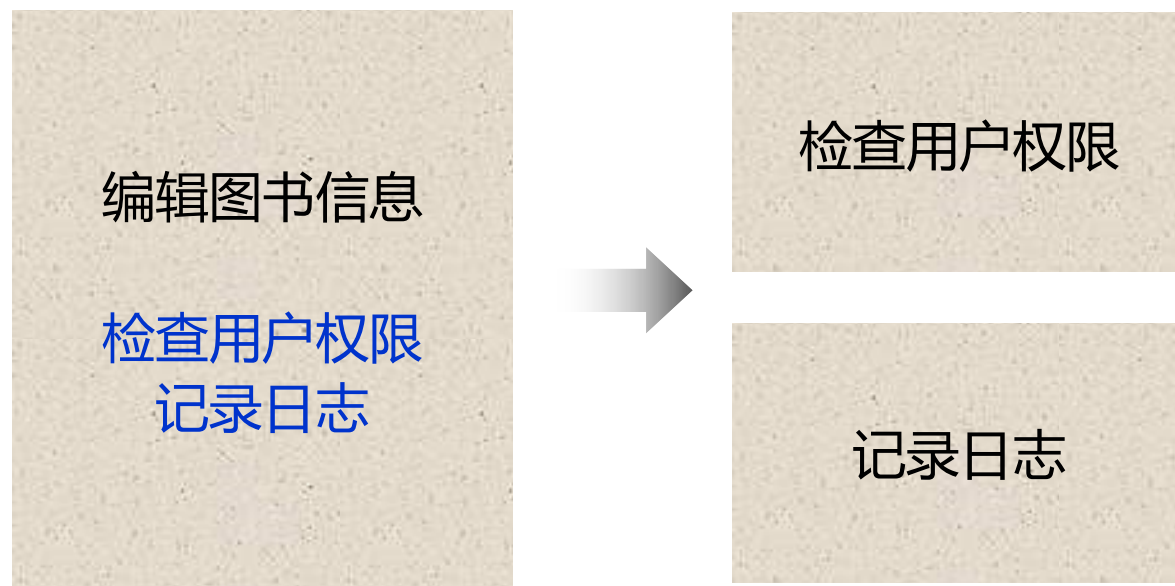
作为一个图书管理员，我可以修改一本现有图书信息。

作为一个图书管理员，我可以删除一本图书。



# 用户故事分解

根据功能的独立性进行分解



# 用户故事分解

在线查询并购买图书

30秒内返回查询结果  
支持100,000个在线用户



30秒内返回查询结果

支持100,000个在线用户

根据功能性与非功能性进行分解



# 用户故事分解

---

作为一个用户，我可以用自己的用户名和密码登录系统，以便可以开始使用系统。



如果输入正确的用户名和密码，则正常登录系统。

如果连续三次输入错误用户名或密码，则该用户账号将被锁住。

如果用户登录被拒绝，则发一条通知信息给该用户，告知有人试图登录系统。

根据优先级进行分解

# 用户故事的问题

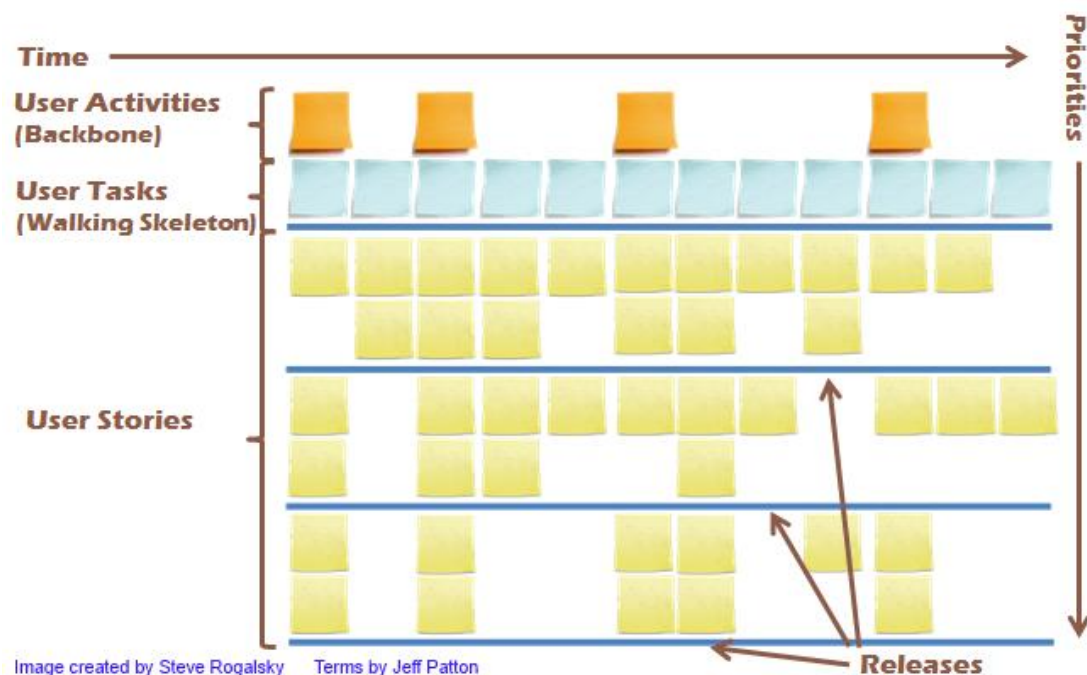


## 只見樹木，不見森林！

- 随着大量故事的产生，容易陷入细节
- 偏离整体目标以及要解决的问题，很可能演变成功能的堆砌
- 对整体缺乏一致性的理解，完成故事开发却仍无法交付

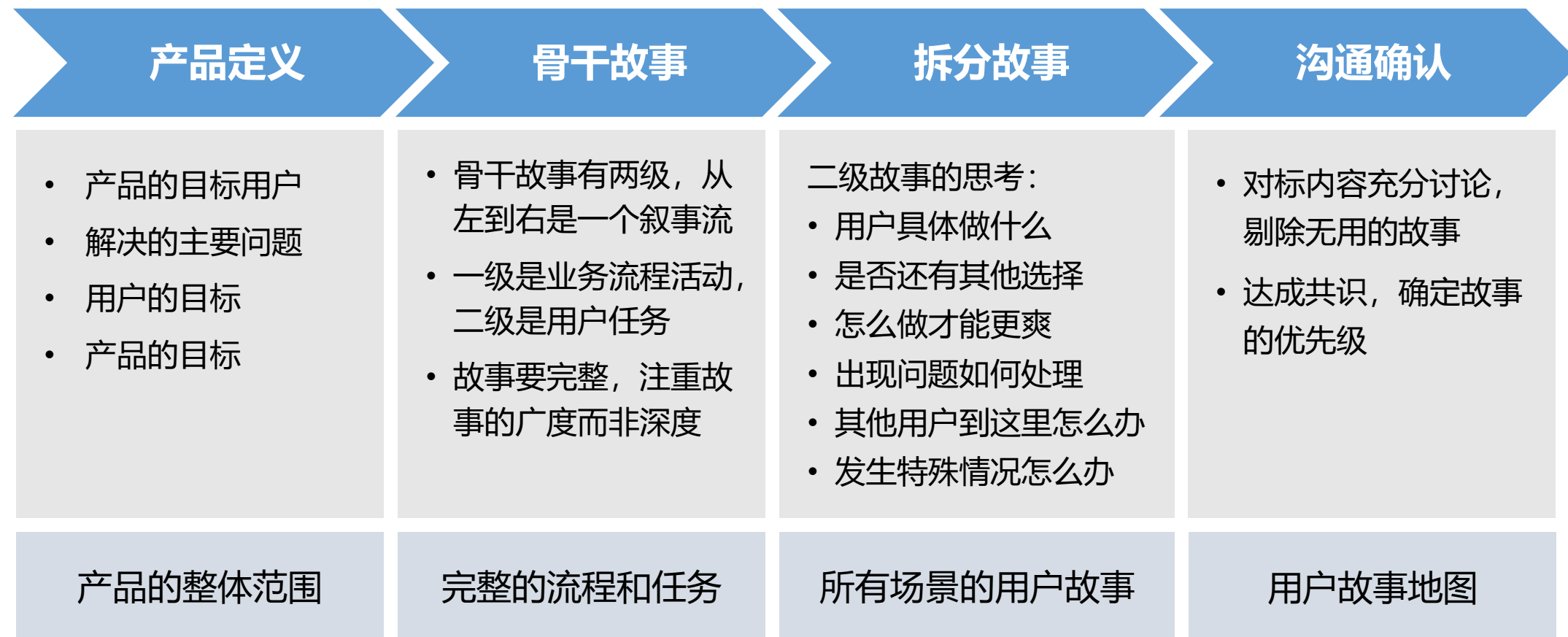
# 用户故事地图

用户故事地图是敏捷需求规划中一种组织和管理用户故事的方法。



- 每个用户故事地图代表一个完整的业务流程
- User Tasks: 从最重要的用户开始, 使用头脑风暴的方法, 描述该用户在一天的使用产品的情景, 把每一件事都写到一张便利贴中, 按照时间顺序从左到右排列
- Backbone: 对所有的便签进行分组, 将类似的任务分为一组, 并对每个组进行命名
- 在每个用户任务下面添加更加细节的用户故事, 借助如Persona和Scenario等方式完成
- 根据用户故事的优先级, 划定产品发布计划

# 用户故事地图



# 举例：参观清华



## 校园参观的痛点：

- 清华大学已成为节假日参观的热门校园景点，原有排队集中进校的管理模式简单粗放。
- 游客：排队等待时间长，体验差
- 学校：排长队、秩序乱、黑导揽客影响学校形象
- 周边：环境脏乱、阻塞交通、存在治安隐患



# 举例：参观清华

## 竞品分析



故宫博物院



首都博物馆

## 最终方案



小程序



人脸识别

- 不用安装，即开即用，用完就走
- 多平台适用，开放成本低
- 小程序实名认证预约，线下闸机刷脸入园参观，安全可靠
- 推广运维更容易，方便触达游客



# 举例：参观清华

## 目标用户



在校学生



一家人

## 特征：

- 18~30岁之间的在校学生，35-45岁之间的父母
- 用户身份繁杂，包括国内的一般游客和军人、国外游客、港澳台游客等
- 比较年轻，乐于接受新事物，喜欢外出旅游，羡慕和向往名牌学校，希望自己或子女将来考入好学校

## 经验：

- 有一定的互联网产品使用经验
- 对微信、QQ、订票等APP比较熟悉

## 举例：参观清华

---

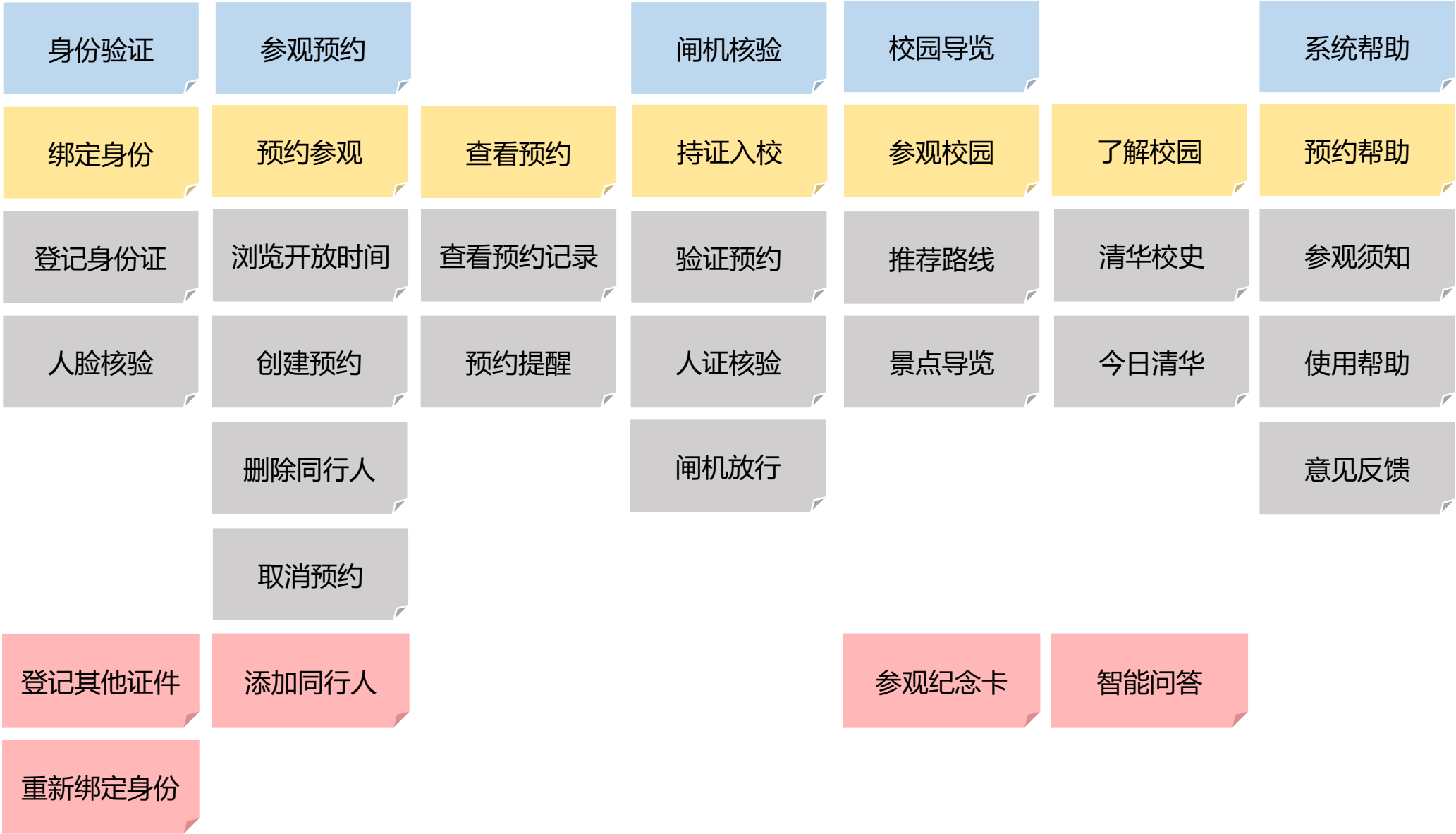
- 每周开放时间：正常学期是周六和周日，寒假和暑假是除周一外都开放，学校有重要活动或恶劣天气时可能会关闭校园开放
- 游客须填写姓名和证件信息，将自己的身份信息与微信号绑定，其中大陆游客应进行实名认证，其他游客应拍摄人脸照片
- 年满10周岁可独立预约；年满16周岁每人可以添加2个同行者，预约者应该与同行者一起进入校园；不满10周岁的儿童须有家长或监护人一起预约入园
- 提前预约的期限是一周，每次预约参观结束两周之后才能再次进校参观；进校时需要检查是否有预约，并且进行人证核验
- 在保证不影响正常作息的前提下，为游客提供经典的景观浏览路线

业务流程（时间线）

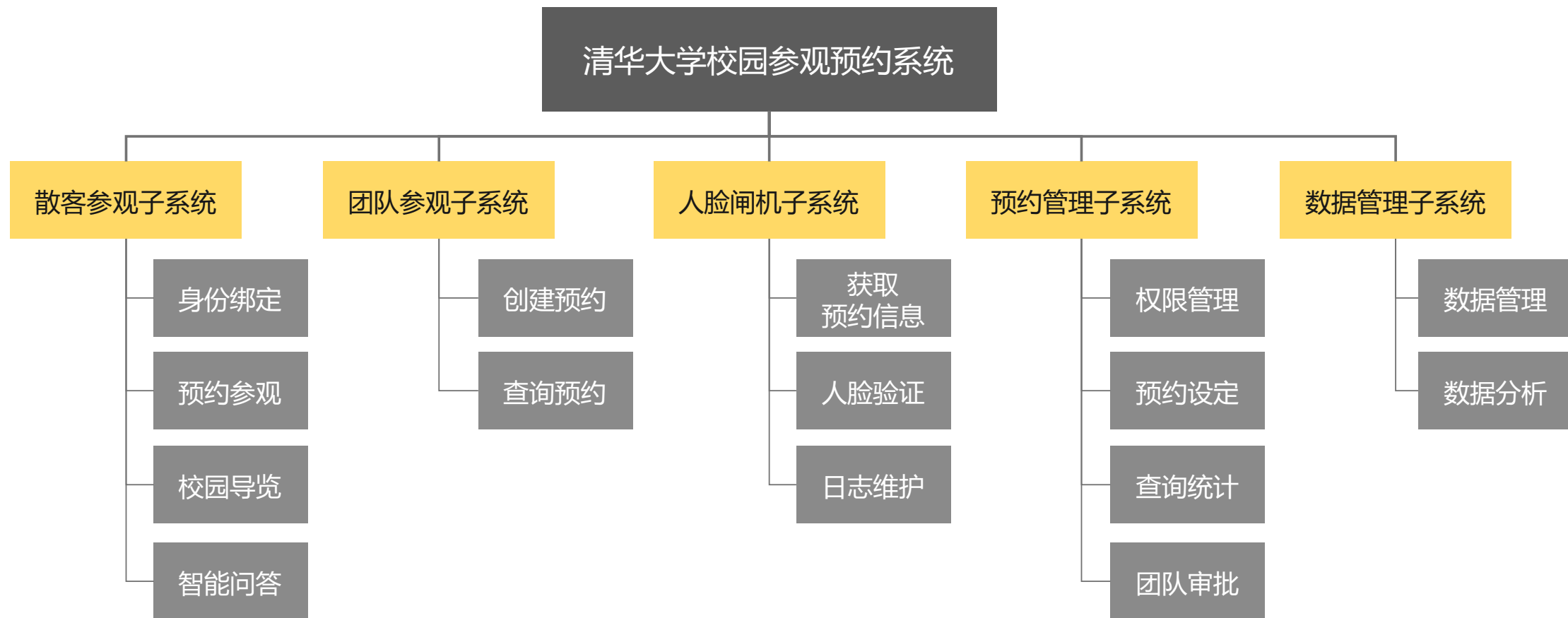
高

商业价值

低



# 举例：参观清华



# 举例：参观清华

## 性能

- QPS达到1000
- 响应时间低于500毫秒
- 合法请求的成功率达到99%

## 安全性

- 敏感数据加密
- 确保身份数据不泄露
- 避免黄牛利用漏洞牟利

## 可靠性

- 入园高峰时段保持高可用性
- 网络中断情况下避免大量游客滞留
- 服务器避免单点故障影响系统运行



参观清华

## 故事点

- 故事点是用于表达用户故事、功能或其他工作的总体规模的度量单位，它是一个相对度量单位。
- 使用时，可以给每个故事分配一个点值；点值本身并不重要，重要的是点值的相对大小。

## 理想日

- 理想日是用于表达用户故事、功能或其他工作的总体规模的另外一种度量单位，它是一个绝对度量单位。
- 理想时间是某件事在剔除所有外围活动以后所需的时间；一般为一天有效工作时间的 60 - 80% 比较合理，但绝不会是全部。

# 敏捷估算

---

## 故事点的基本做法：

- 把一些常见的“标准任务”给出一个“标准点数”，形成比较基线；
- 估算时只要是同一类型任务，直接写故事点数而非天数。

## 使用难点：

- 故事点的项目或产品特征很明显，几乎无法进行跨团队比较；
- 如果没有历史数据，很难设定标准任务；
- 在标准任务没有那么多种类时，很难判断一个新任务到底像哪个标准任务；而太多的标准任务又令人迷惑。

# 敏捷估算

优先级	名称	用户故事描述	故事点
1	浏览商品	作为一名顾客想购买商品而不确定型号时，我希望能浏览网站在售的商品，按照①商品类型和②价格范围进行过滤。	10
2	搜索商品	作为一名顾客在查找某种商品时，我希望能进行不限格式的文本搜索，例如按照短语或关键字。	15
3	注册账户	作为一名新顾客，我希望注册并设置一个帐户，包括用户名、密码、信用卡和送货信息等。	1
4	维护购物车	作为一名顾客，我希望能将指定商品放入购物车（稍后购买）、查看我的购物车内的商品以及移除我不想要的物品。	40
5	结账	作为一名顾客，我希望能完成我购物车内所有商品的购买过程。	28
6	编辑商品规格	作为一名工作人员，我希望能够添加和编辑在售商品的详细信息（包括介绍、规格说明、价格等）。	12
7	查看订单	作为一名工作人员，我希望能登录并查看一段时间内应该完成或已经完成的所有订单。	7



# 敏捷估算

敏捷估算扑克本质上是扑克牌，它基于Delphi估算原理，可以快速地估算出需要的数字。



- 估算扑克牌上的数字：有些牌是自然数排列，有些是斐波纳契数，有些则是不连续自然数。
- 具体选用哪种扑克，要根据被估算的内容的跨度大小而定，如果估算值跨度在10倍以内，那么采用顺序自然数比较好，如果数值跨度较大，达到10倍以上，那么采用斐波纳契数比较好。
- 一般而言，估算软件开发工时的话，自然数可能更好一些，毕竟数值都不大，跨度也不会很夸张。

# 敏捷估算

**分牌：**每名参与估算的成员分得相同花色的一组牌，两张Joker不参与估算。



敏捷扑克和普通游戏扑克一样，也有54张牌，也拥有4种花色（每种各13张）和两张Joker。

敏捷扑克的每种花色均是一组13张牌组成的估算扑克牌，牌正面上印刷有供估算用的数字与符号，数字分别是1/2、1~10和20以及符号“！”（代表一些未知情况，如无法提供准确估算值等）。

- 一副估算扑克可供四人使用，如果参与的人员多于4人，可使用多副扑克。
- 一般推荐4-8人参与估算，人太少会使估算结果偏差很大，而太多会拉长估算时间，降低估算效率。

# 敏捷估算

**讲解订单故事：**产品负责人从Backlog中选择一个条目，为大家详细讲解该条目；团队成员进行讨论并提问，产品负责人逐一解答大家的问题。



在讲解过程中，千万不要制定该条目的负责人或有明显倾向的人来做这个条目，这样会大大降低团队成员的积极性，甚至会扰乱估算秩序与结果。

这个步骤是团队和产品负责人之间的交互环节，帮助团队和产品负责人共同加深对条目的理解；产品负责人也会根据大家的反馈，及时修改和完善条目。



# 敏捷估算

**估算：**当团队成员确认已经对该条目完全了解且无任何重大问题后，大家开始对该条目进行估算，同时选出代表自己估算值的纸牌，但不可立即亮牌。在估算过程中，为避免干扰估算结果，团队成员之间不可以互相商讨。当所有成员选牌完毕，大家可以同时亮牌。





# 敏捷估算



VS



**争论与讨论：**对比每张牌估算值之间的大小，若估算值差距明显，代表大家对该条目的价值没有获得共识，团队需要对该条目价值评估结果进行讨论。

**共识：**对该条目重新进行估算，直到团队的评估数值达成一致。一般情况下，最多三轮就可以得出一个比较统一的意见；如果三轮之后依然没有得到统一的意见，那么Scrum Master应当立即中断该条目的估算，取平均值或其他大家能接受的值作为估算结果。





# 谢谢大家!

---

## THANKS

