软　件　工　程

西南交通大学电气工程学院

1.1 软件危机

1.2 软件工程

1.3 软件生命周期

1.4 软件过程

敏捷开发的12条原则

1、尽早和持续地交付有价值的软件来满足客户

2、欢迎对需求提出变更——即使是在项目开发后期。要善于利用需求变更，帮助客户获得竞争优势。

3、要不断交付可用的软件，周期从几周到几个月不等，且越短越好。

4、项目过程中，业务人员与开发人员必须在一起工作。

5、要善于激励项目人员，给他们以所需要的环境和支持，并相信他们能够完成任务。

6、无论是团队内还是团队间，最有效的沟通方法是面对面的交谈。

敏捷开发的12条原则

7、可用的软件是衡量进度的主要指标。

8、敏捷过程提倡可持续的开发速度，项目方、开发人员和用户应该能够保持恒久稳定的进展速度。

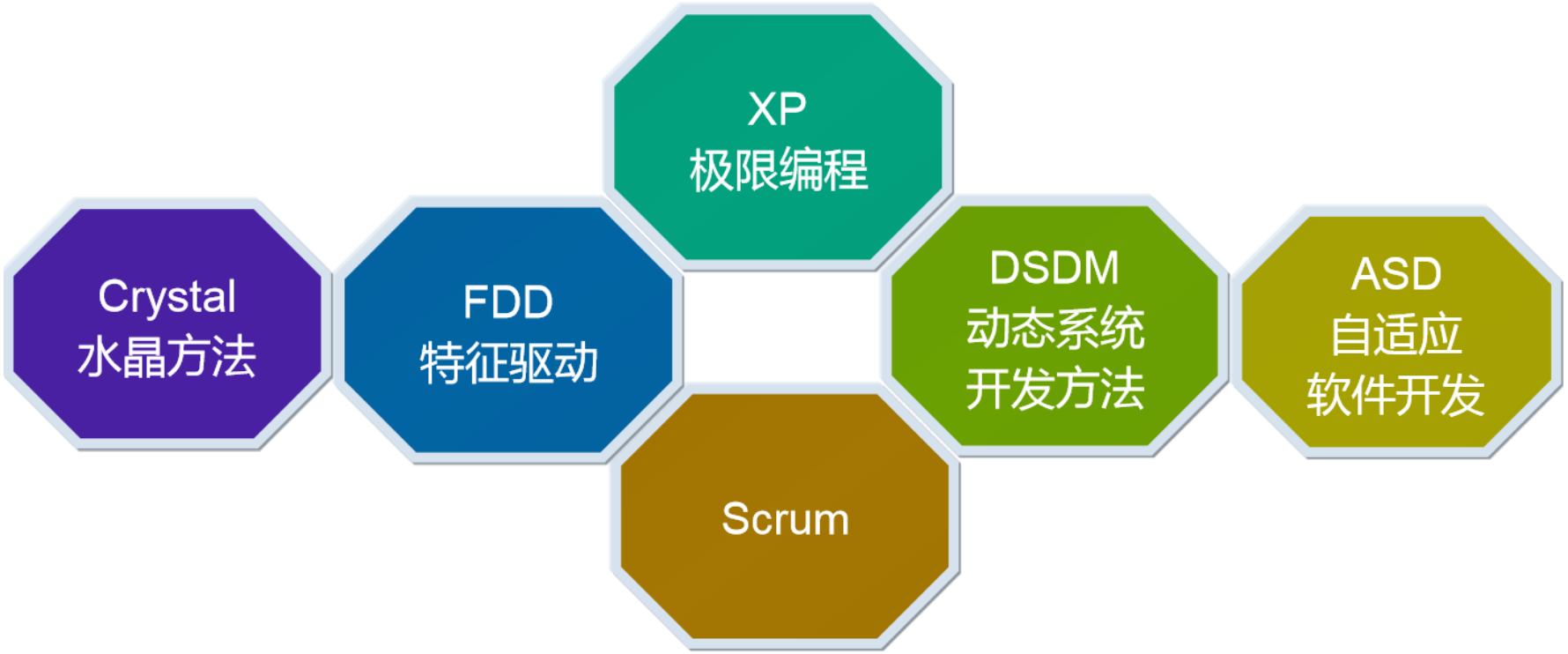
9、坚持不懈地追求技术卓越和良好设计，这将提升敏捷能力。

10、要做到简单，即尽最大可能减少不必要的工作。

11、最佳的架构、需求和设计出自于自组织的团队。

12、团队要定期反省如何能够做到更有效，并相应地调整团队的行为。

敏捷开发方法



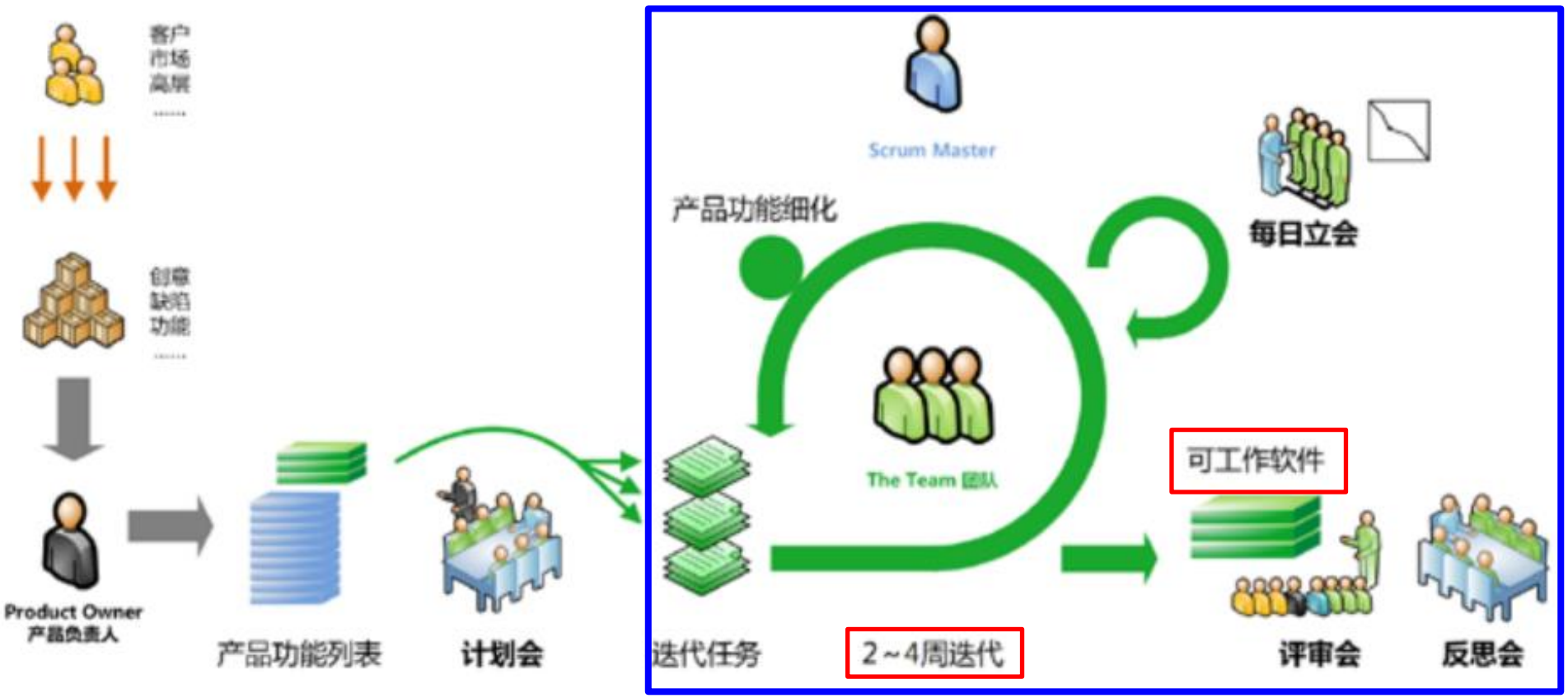
敏捷开发方法是一组轻量级开发方法的总称，包含很多具体的开发过程和方法， 最有影响的两个方法是Scrum开发方法和极限编程（XP）。 Scrum偏重项目管理，XP偏重编程实践

Scrum 框架

- Scrum是橄榄球运动的专业术语，表示“**争球**”
- 在敏捷开发系列中，把一种开发流程命名为Scrum，意味着开发团队在开发一个项目时，大家像打橄榄球一样迅速、富有战斗激情、人人你争我抢地完成，将目标需求进行快速的开发和迭代。

Scrum 框架

兼顾计划性与灵活性的敏捷开发过程，将整个开发过程分为若干次更小的迭代，每个迭代周期称为一个冲刺Sprint 。



Scrum 步骤

第一步：找出完成产品需要做的事情—Product Backlog

- 产品负责人主导大家分析和细化Backlog中的条目，估计工作量（以天为时间估计单位）。

第二步：决定当前冲刺需要解决的事情—Sprint Backlog

- 整个产品的实现被划分为几个互相联系的冲刺。细化产品订单上的任务(分解为以小时为单位)。

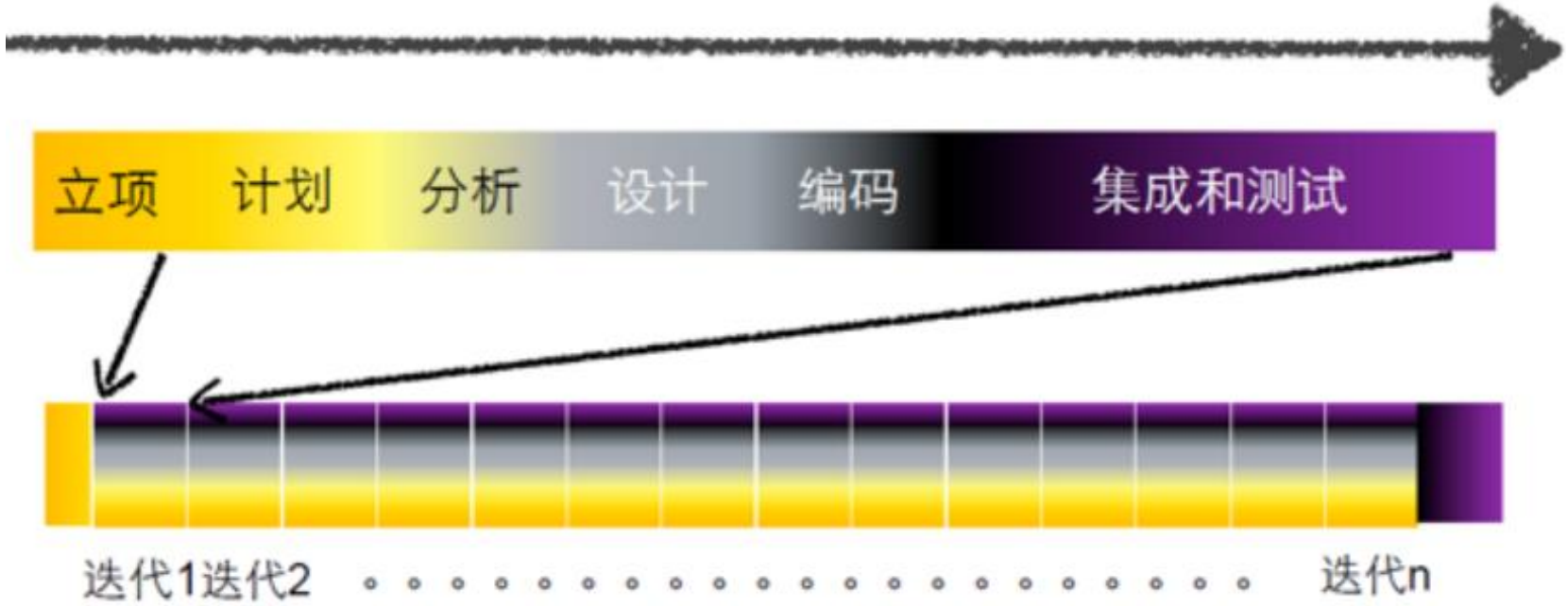
第三步：冲刺(Sprint)

- 本阶段外部人士不能直接打扰团队成员。一切交流通过Scrum Master完成。较好地平衡了"交流”和“集中注意力”的矛盾

第四步：发布一个增量版本。然后开始下一个迭代。

Scrum 迭代开发

将整个软件生命周期分成多个小的迭代（一般2-4周），每次迭代就是一个小的瀑布模型，包括需求分析、设计、实现和测试等活动，**结束时都要生成一个稳定和被验证过的软件版本**。



Scrum 迭代开发的关键点

- 每一次迭代都建立在稳定的质量基础上，作为下一轮迭代的基线，系统的功能随着迭代稳定地增长和不断完善。
- 每次迭代要邀请用户代表验收，提供需求是否满足的反馈。
- 在一次迭代中，一旦团队作出承诺，就不允许变更交付件和交付日期；若发生重大变化，产品负责人可中止当次迭代。
- 在迭代中可能会出现“分解”和“澄清”，但是不允许添加新工作或者对现有的工作进行“实质变更”。
- 对于“分解”和“澄清”，如果存在争议，那么将其认定为变更，放到产品订单中下一次达代再考虑。

Scrum 框架

三个角色

产品负责人

Scrum 主管

团队成员

三个制品

产品订单

迭代订单

可工作的软件

四个会议

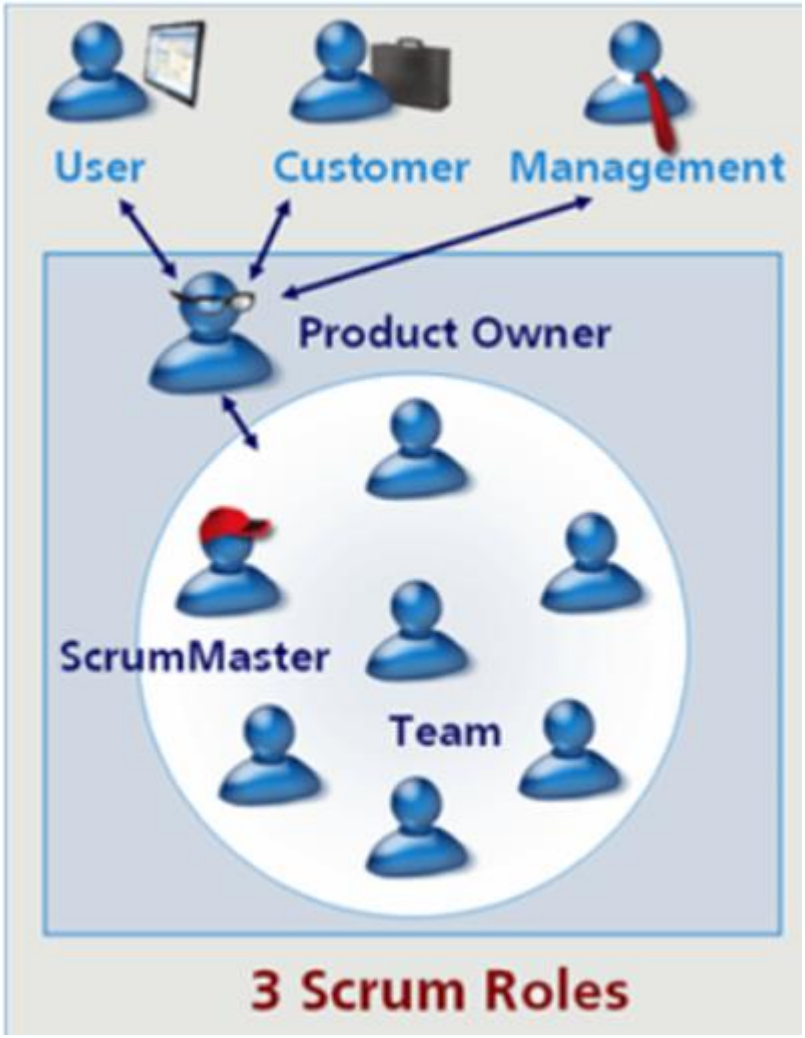
迭代计划会议

每日站会

迭代演示会议

迭代回顾会议

Scrum 团队角色



产品负责人（Product Owner）

职责：定义开发目标以及需要实现的特性和优先级。

Scrum主管（Scrum Master）

职责：保证团队高效而不受打扰地工作，优化工作条件和过程。

团队成员（Team）

职责：自组织地完成项目开发，使用一切可行手段保证进度和质量。

团队中的“猪”和“鸡”

自主管理
自我组织
多功能型

猪角色：团队中的核心成员，产品负责人、Scrum主管和开发团队

鸡角色：用户、客户或提供商，经理等。

敏捷的团队

- **自主管理**：不是领导布置任务，团队实现。而是自己挑选任务，每次冲刺结束后要总结不足，提出改进，并且自己要实施这些改进。"自主管理"不等于"没有管理"
- **自我组织**：不是只做好自己的事，安心下班。而是每个人要联合起来对项目负责，有人工作落后要帮助他改进
- **多功能型**：以前规格说明书有项目经理写，测试由测试人员做。现在每个人都全面负责，自己搞定规格说明书，和别人沟通，同时自己搞定测试。

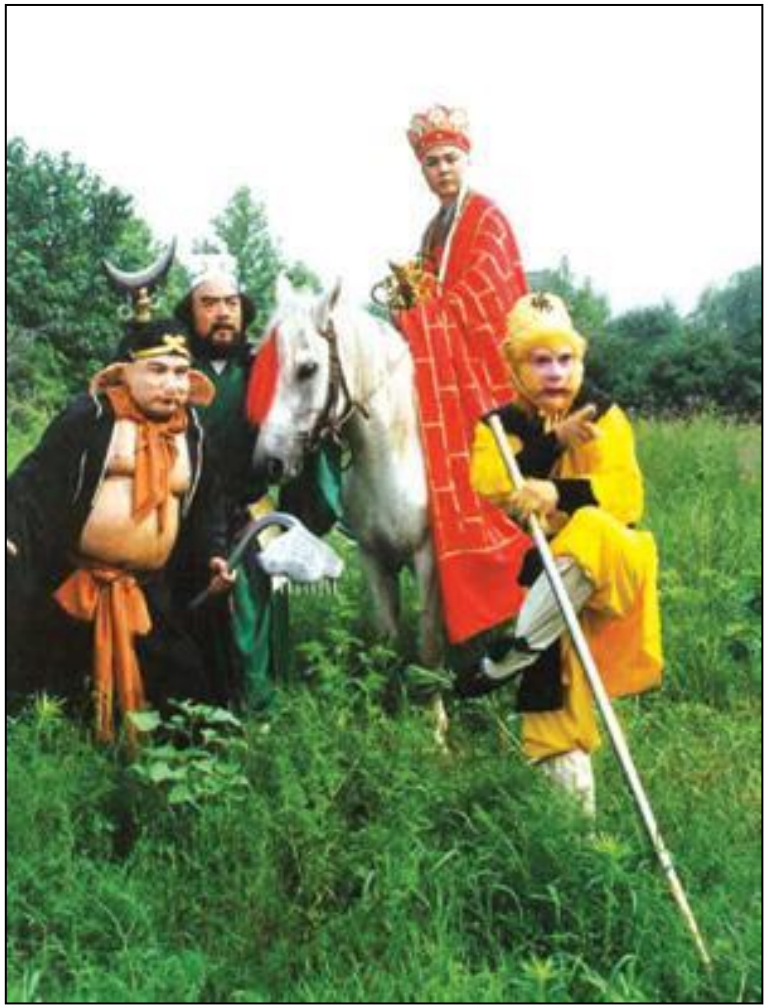
人员的选择

例：假设你是软件公司项目经理，公司希望进入帮助独立生活的老年人和残疾人的技术服务市场。你需要组建一个6人的开发团队，他们可以基于公司现有的报警处理技术开发新产品。

你如何选择团队成员?为什么?

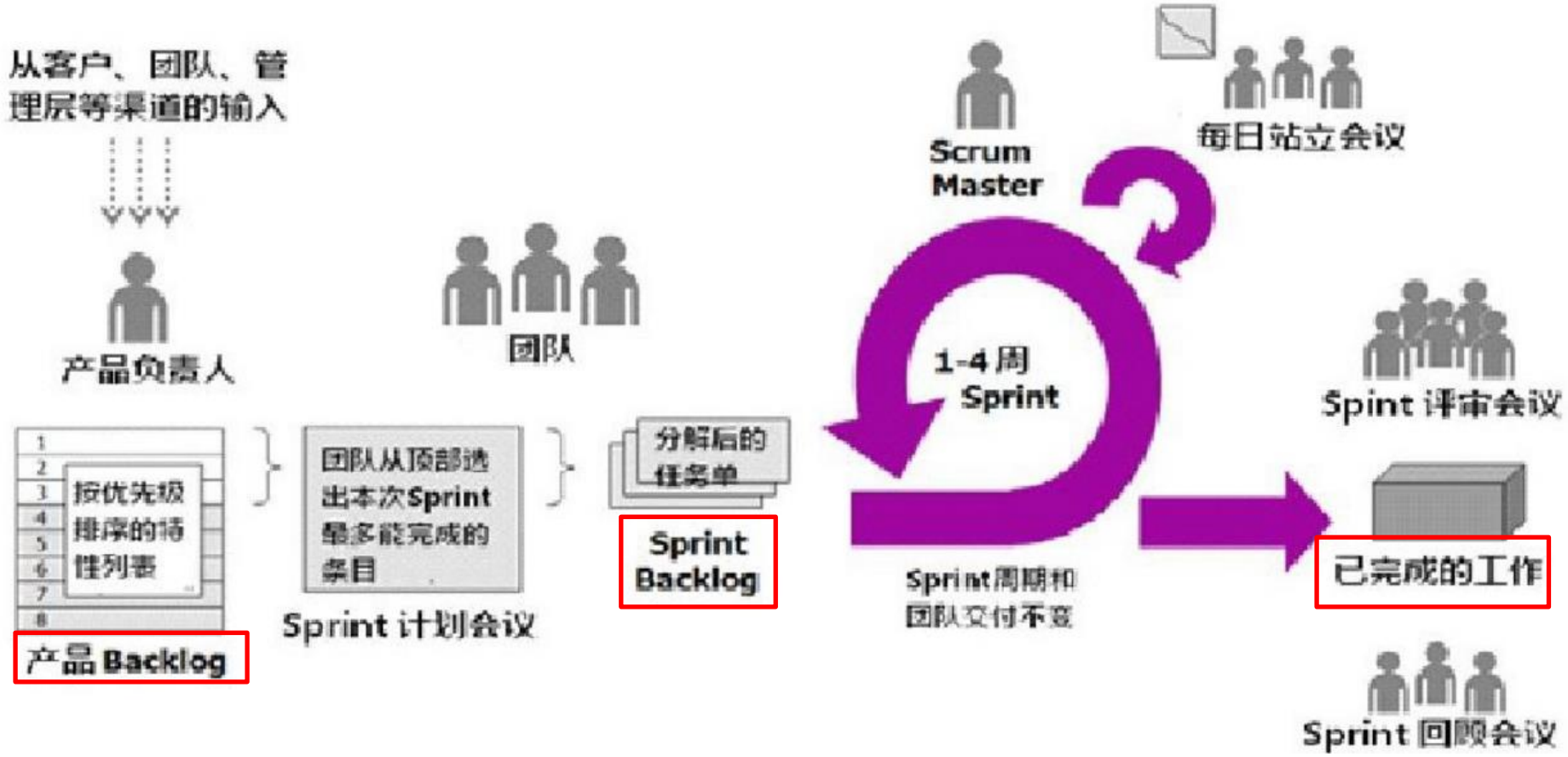
- 应用领域经验
- 平台经验
- 编程语言经验
- 教育背景
- 解决问题能力
- 沟通能力
- 适应性
- 工作态度
- 个性

人员的选择



- 应考虑团队中的技术、经验和个性是否整体均衡
- 选择性格互补的成员组成的团队可能比仅仅根据技术能力选择成员的团队更有效率
- 团队的领导力来自于成员的尊重，而不是名义上的头衔

Scrum 框架流程


从客户、团队、管理层等渠道的输入
产品负责人
团队
Scrum Master
每日站立会议
1-4周 Sprint
Spint 评审会议
1
2
3
4
5
6
7
8
按优先级排序的特性列表
产品 Backlog
团队从顶部选出本次Sprint最多能完成的条目
Sprint 计划会议
分解后的任务单
Sprint Backlog
Sprint周期和团队交付不变
已完成的工作
Sprint 回顾会议

Scrum 制品

Product Backlog

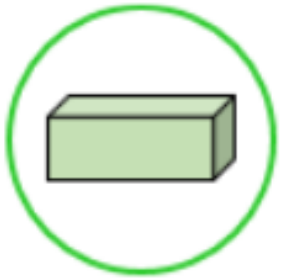
产品订单

- 从客户价值角度理解的产品功能列表
- 整体上从客户价值进行优先级排序

Sprint Backlog

迭代订单

- 从开发技术角度理解的迭代开发任务



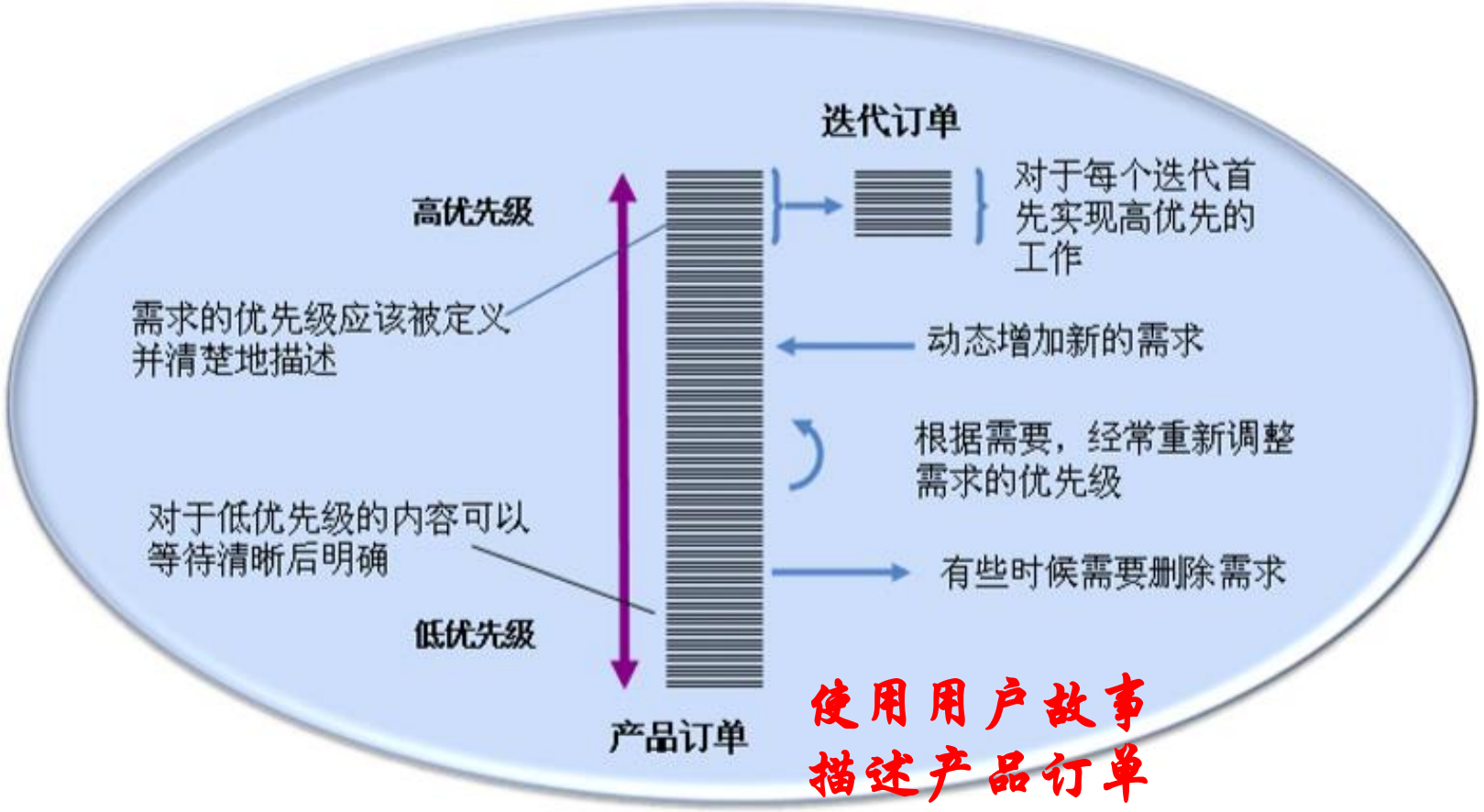
Working Software

可工作软件

- 可交付的软件产品

产品订单

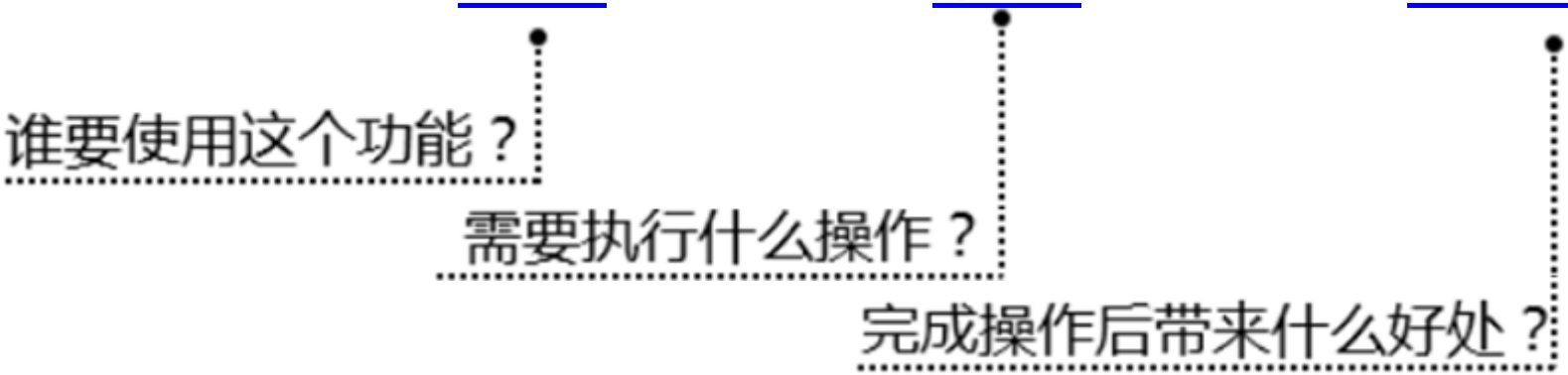
迭代规划时，产品负责人告诉团队需要完成产品订单中的哪些订单项，开发团队决定在下一次迭代中能完成多少订单项。迭代过程中不允许变更迭代订单，即**一次迭代中需求是被冻结的**



用户故事(User Story)

- 用户故事是从用户角度对功能的简要描述。
- 格式：

作为一个<角色>，我想要<活动>，以便于<商业价值>



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。

用户故事(User Story)

如下网络游戏的排行榜功能用户故事描述是否合适

作为一个玩家，通过显示排名让自己在服务器中的地位获得认可

- 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。

排行榜功能修改为：系统每周重新排名一次，而且只显示前XXX名玩家。

修改后的用户故事：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）

用户故事类型

- 作为一名**维基用户**，我希望上传一个文件到维基，以便可以和同事进行分享。
- 作为一名**客服代表**，我希望为客户问题创建一个记录卡，以便记录和管理客户支持请求。
- 作为一名**网站管理员**，我想统计每天有多少人访问网站，以便赞助商了解网站会给他们带来什么收益
- 系统必须支持IE、360、Firefox 和Google浏览器。
- 修复缺陷#256，这样可以使客户在搜索项中输入特殊字符不会出现异常。

例：购物网站的产品订单

优先级	名称	用户故事描述
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。
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。
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。
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。
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。
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）。
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。

必须按照商业价值进行优先级排序

产品订单——>迭代订单

产品订单任务

- 作为用户，希望这个英语单词学习软件能在各个移动平台上实现我的学习进度同步。

可执行的迭代订单任务

- 什么是用户学习进度？用户如何衡量这个功能的优劣？
- PC平台和各个移动平台分别用什么来表示“学习进度”？
- 同步是通过什么样的技术手段实现？
- 如何解决可能的同步冲突问题？
- ……

团队日历

四个会议（以2周迭代为例）

1	2	3	4	5
计划会议故事 计划会议任务	站立会议	站立会议	站立会议	站立会议

6	7	8	9	10
站立会议	站立会议	站立会议	站立会议	演示会议 回顾会议

计划会议

- 参加人员：产品负责人、 Scrum Master、 Scrum团队和其他感兴趣的人。
- 产品负责人从Backlog中挑选高优先级的故事，并与Scrum团队一起决定在这个 Sprint中需要完成多少功能
- 将任务分解成小的功能模块。
- 团队成员详细讨论如何按需求完成这些功能模块，并估计完成每个功能模块所需的大概时间。
  - 团队评审并确认“完成”的定义
  - 团队决定承诺完成多少，以及如何实现承诺

敏捷估算

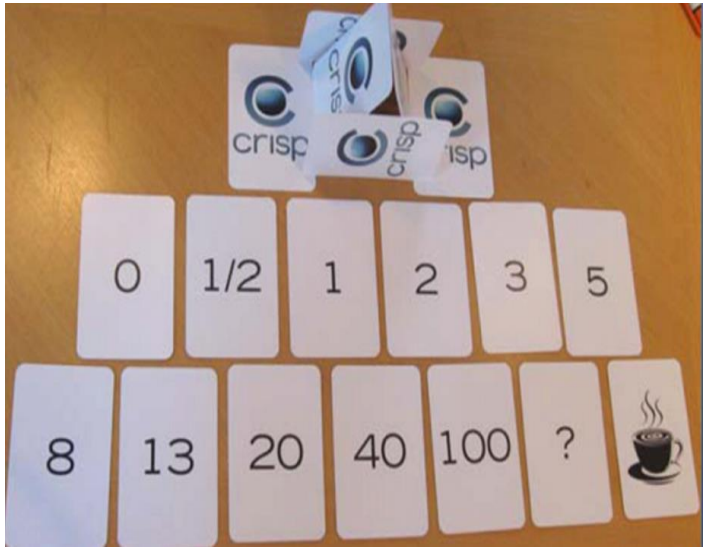
- 故事点
  - 一个相对度量单位，可给每个故事分配一个点值
  - 点值本身并不重要，重要的是点值的相对大小。
- 故事点的基本做法：
  - 给一些简单的“标准故事”设定一个“标准点数”，形成比较基线
  - 其他故事与标准故事进行比较，给出一个相对的比例，得到该故事的一个估计值。

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

故事点：表达用户故事的一个相对度量单位。把一些常见的“标准任务”给出一个“标准点数”，形成比较基线。

敏捷估算扑克



分牌：每名参与估算的成员分得相同花色的一组牌。

- 0 代表条目已完成或太小没有估算意义
- 1/2 代表微小条目，1, 2, 3 代表小条目
- 5, 8, 13 代表中等大小条目
- 20, 40 代表大的条目，100 代表非常大的条目
- ？ 代表对条目不理解或不知道如何估算

计划纸牌（敏捷估算）

产品负责人逐个讲解产品订单，团队成员进行估算

团队成员了解条目后进行估算，选出代表自己估算值的纸牌，为避免干扰，不可以商讨，所有成员估算完毕，大家**同时亮牌**

计划纸牌（敏捷估算）

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。

每日站立会议

- 做计划，协调其每日活动，报告和讨论遇到的障碍
- 任务板帮助团队聚焦于每日活动上，应在这个时候更新任务板和燃尽图。

每日站立会议

- 每个团队成员需要回答三个问题：
  - 我昨天做了啥?
  - 我今天要做啥?
  - 遇到了什么困难？(不讨论)

- 可能流于形式：
  - 我昨天写代码
  - 我今天继续写
  - 我没碰到困难

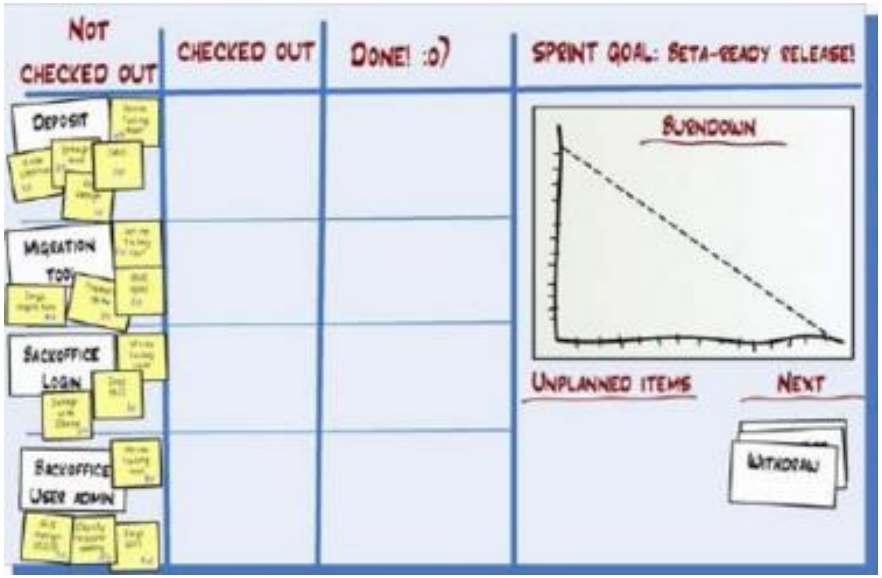
改进1：明确定义每个人的任务。

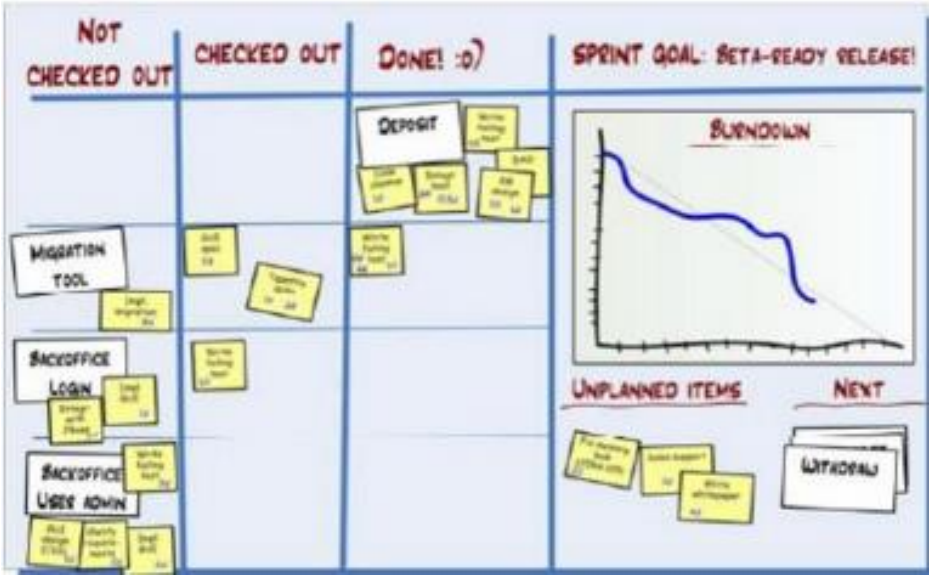
说明：编号为X的任务现在是什么状态？做好之后交给了谁？

改进2：完成这个任务还需要多少时间？

说明：已经花了多少时间虽然重要，但关键是还要花多少时间

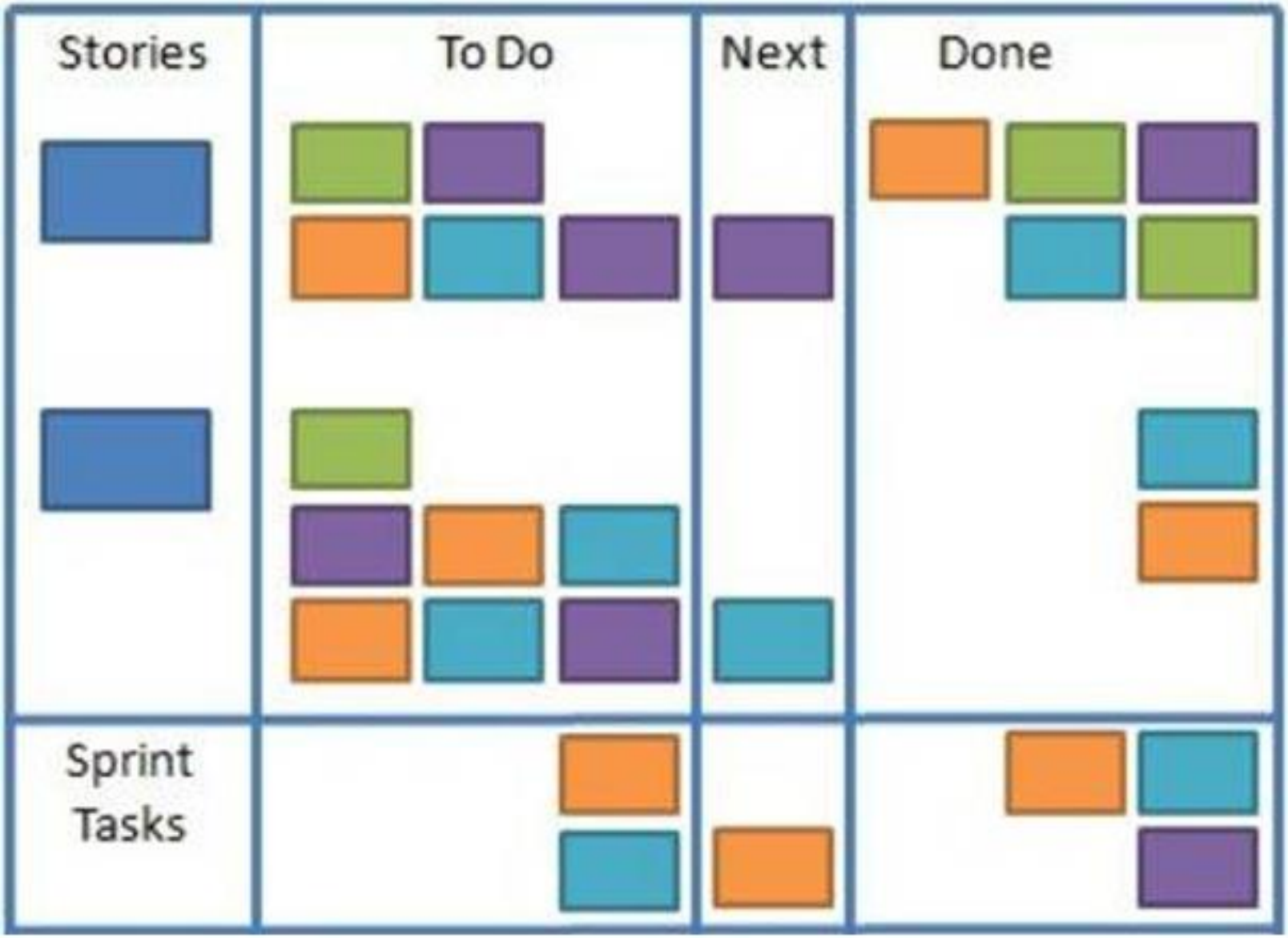
可视化管理——任务看板





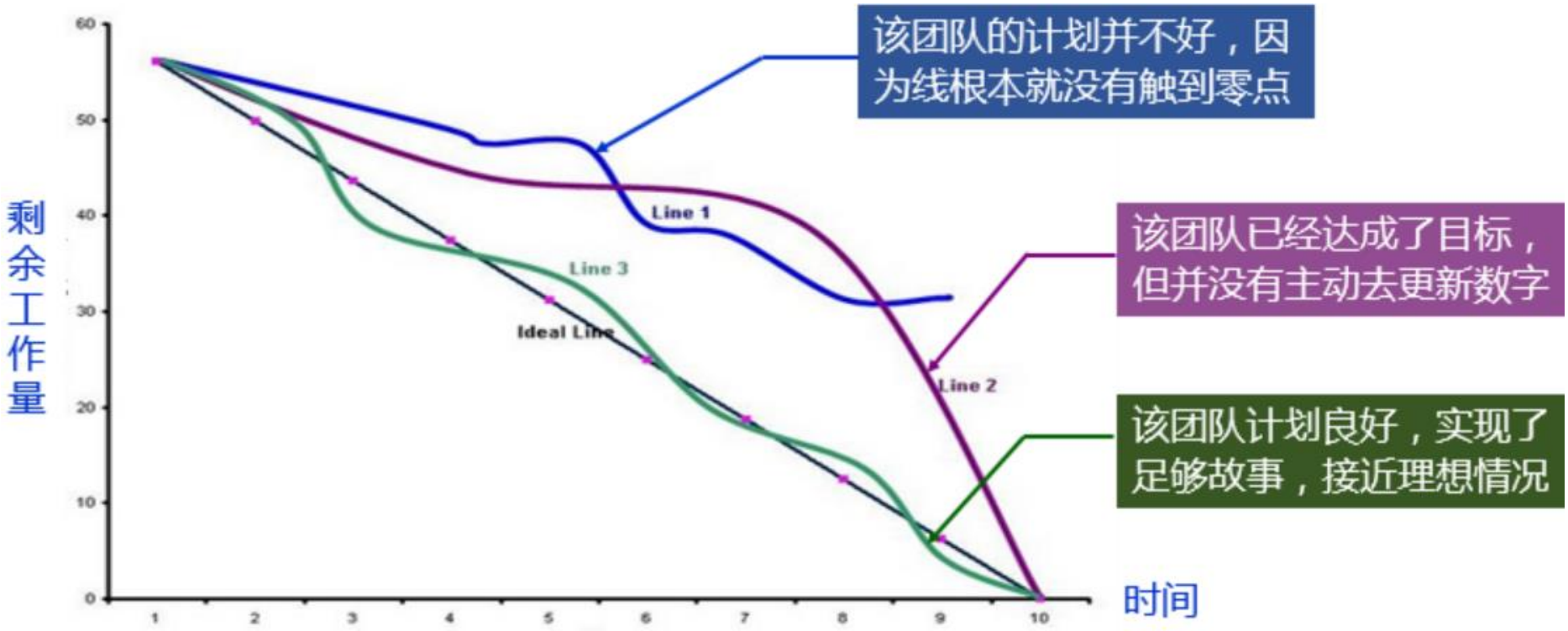
任务看板包含未完成、正在做、已完成的工作状态，假设你今天把一个未完成的工作已经完成，那么你要把小卡片从未完成区域贴到已完成区域。

可视化管理——任务看板



可视化管理——燃尽图

燃尽图：以图形化方式展现了剩余工作量(Y轴)与时间(X轴)的关系。



迭代演示会议和迭代回顾会议

迭代演示(评审)会议

- 团队在会议中向**最终用户**展示工作成果并希望得到反馈，并以之创建或变更订单条目。

迭代回顾会议

- 每一次迭代完成后举行一次迭代总结会议，所有成员都要反思这个迭代。其目的是为了持续过程改进。

敏捷视频观看

极限编程

如果…	发挥到极致就变成……
了解顾客的需求很重要	每时每刻都有客户在身边，时时了解需求
测试/单元测试能帮助提高质量	那就先写单元测试，从测试开始写程序——测试驱动开发
代码复审可以找到错误	从一开始就处于“复审”状态——结对编程
计划没有变化快	那就别做详细的设计，做频繁的增量开发、重构和频繁地发布
如果提高用户的体验是最重要的目标	那就从体验出发，发掘用户需求，而不是从技术出发，说服用户使用最新的技术
(其他好方法……)	(发挥到极限的做法……)

微软解决方案框架MSF

具体
实践

Microsoft® Solutions Framework

1994年正式问世，把技术、人和过程结合起来

2010年后，MSF对于敏捷的流程有更多的支持

MSF基本原则

(1) 推动信息共享与沟通

- 保留并公开所有信息，讨论要包括所有涉及的角色
- 告知所有人项目进度以及项目中存在的各种问题。

(2) 为共同的远景而工作

- 明确项目的目标是什么，没有二义性；
- 不是当前就能达到，必须是通过努力才能达到的；
- 目标不空泛，项目成员每天的工作都有指导作用。

MSF基本原则

(3) 充分授权和信任

- 平等协作，成员之间、团队之间是平等协作关系
- 充分授权给团队和成员。成员有权力在自己的职权范围内按照他们自己的承诺完成任务，同时，他们也充分信任其他同事也能实现各自的承诺

MSF团队模型是网状，而不是层次结构

MSF基本原则

(4) 各司其职，对项目共同负责

关键质量目标	MSF小组角色	出口条件
按约束条件交付产品	项目管理	我们的项目是在时间/资源的条件内交付的么?
按产品规格说明交付产品	开发	我们是否按照功能说明完成了各项功能?
保证所有问题都得到处理	测试	我们发现了所有的问题，而且都有处理方案吗?
产品部署和后续管理	发布管理	客户是否能快速方便地部署产品和进行后续管理?
让产品更好用	用户体验	产品是否适应用户的使用习惯？易学易用?
让客户满意	产品管理	客户是否（在总体上）满意我们的项目?

与“信息共享与沟通”原则相呼应，让所有人明确自己的职责，同时有大局观：知道别人在做什么，为什么，以及整个项目的目标

MSF基本原则

(5) 重视商业价值

如果你还没能说清楚你的产品解决了什么问题，为谁解决问题，为什么你的产品会解决这些问题，以及客户怎样付钱让你解决问题，那你就不应该贸然创业。

(6) 保持敏捷，预期和适应变化

软件工程，唯一不变的是变化。微软的一些成功项目的各个版本之间一般间隔18~24个月。

MSF基本原则

(7) 投资质量

不是质量第一，而是解决用户的问题第一。及时发布能够解决用户问题的软件，并且能及时修改软件中的问题

(8) 学习所有的经验

- 让团队成员从别人的成果和失败的例子中学习
- 帮助新项目重复以往成功的做法
- 培育团队总结的习惯和批评与自我批评的文化。

(9) 与顾客合作

- 不是“我觉得用户会喜欢”，而是“用户觉得”