敏捷方法与过程

主要内容

1 敏捷过程模型

2 极限编程(XP)

3 Scrum

4 与传统开发过程模型的对比

5 敏捷案例分析

敏捷方法与过程

为何要“敏捷”？

- 最初的软件 (1960-70 年代) 的顾客都是大型研究机构、军方等，他们需要软件系统来搞科学计算、军方项目、登月项目等，这些系统相当庞大，对准确度要求相当高
- 1980-90年代，软件进入了桌面软件的时代，开发的周期明显缩短，各种新的方法开始进入实用阶段。但是软件发布的媒介还是CD、DVD，做好一个发布需要较大的经济投入，不能频繁更新版本
- 互联网时代，大部分的服务是通过网络服务器端实现，在客户端有各种方便的推送 (push) 渠道
  - 由于网络的转播速度和广度，知识的获取更加容易，很多软件服务可以由一个小团队来实现
  - 同时技术更新的速度在加快，那种一个大型团队用一个固定技术开发 2-3 年再发布的时代早已经过去了
  - 用户需求的变化也在加快，开发流程必须跟上这些快速变化的节奏

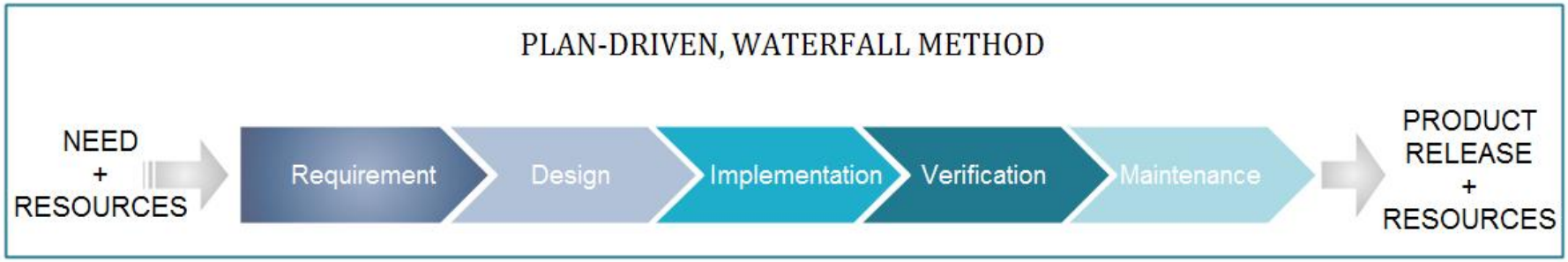
敏捷方法与过程

为何要“敏捷”？

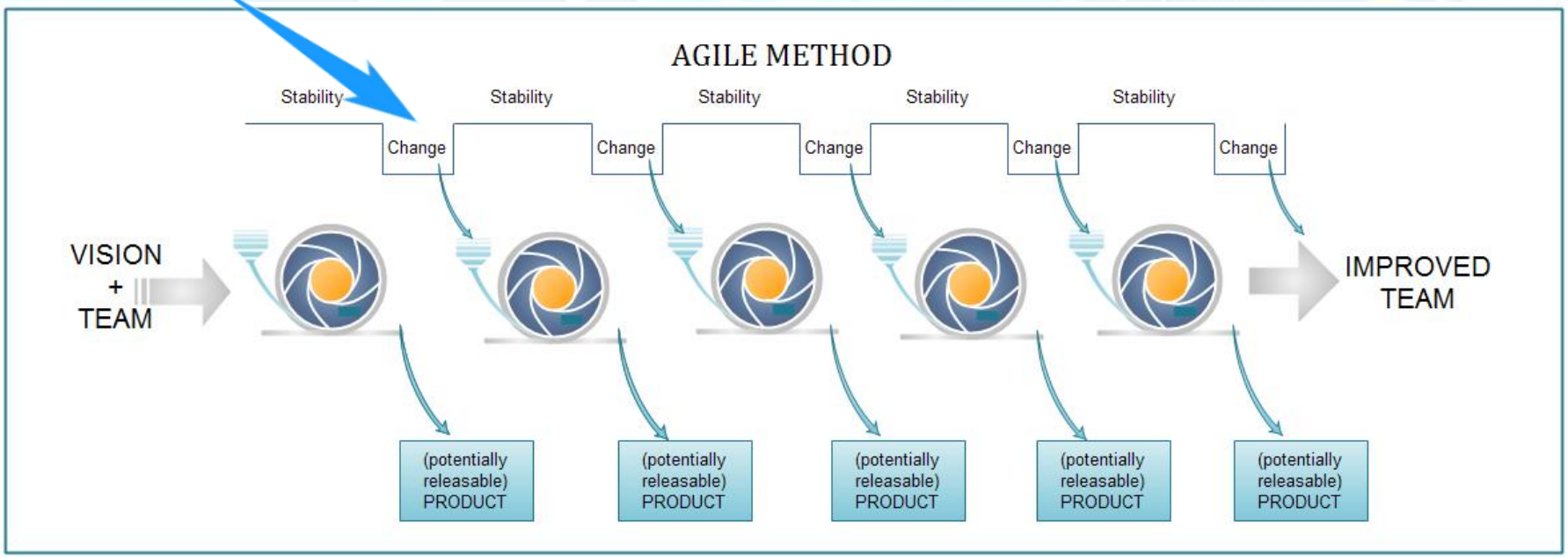
- 开发过程中的“变化”是无处不在的，也是不可避免的
- 在实际项目中，很难预测需求和系统何时以及如何发生变化
- 对开发者来说，应将变化的意识贯穿在每一项开发活动中

敏捷方法与过程

敏捷方法 VS 传统方法

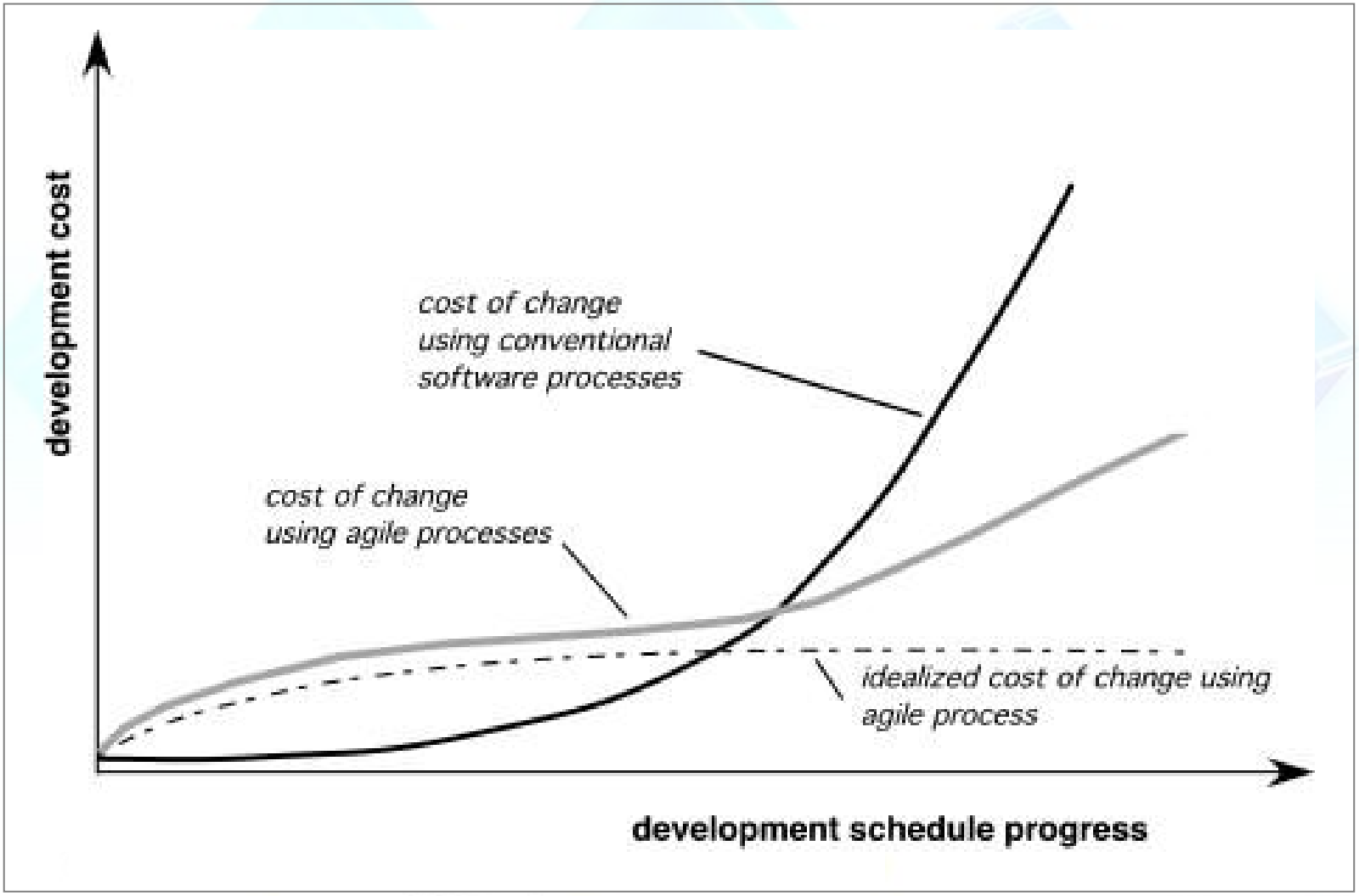


传统方法很难应对频繁的变化、快速的交付！



敏捷方法与过程

成本变化曲线：敏捷方法 VS 传统方法



敏捷方法与过程

敏捷软件开发宣言

- **2001年，Kent Beck等17位编程大师共同发布《敏捷软件开发宣言》：**

“人”以及“人与人的互动” **胜于** “过程”和”工具”

Individuals and interactions over processes and tools

可运行的软件 **胜于** 面面俱到的文档

Working software over comprehensive documentation

客户合作 **胜于** 合同谈判

Customer collaboration over contract negotiation

响应变化 **胜于** 遵循计划

Responding to change over following a plan

敏捷宣言遵循的12条原则(1)

- **准则1：Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.**
  - 我们的最高目标是通过尽早和持续交付有价值的软件来满足客户
- **准则2：Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.**
  - 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
- **准则3：Deliver working software frequently, from a couple of weeks to a couple of mouths, with a preference for the shorter timescale.**
  - 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

敏捷宣言遵循的12条原则(2)

- **准则4：Business people and developers must work together daily throughout the project.**
  - 项目过程中，业务人员与开发人员必须在一起工作
- **准则5：Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.**
  - 要善于激励项目人员，给他们所需要的环境和支持，并相信他们能够完成任务
- **准则6：The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.**
  - 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

敏捷宣言遵循的12条原则(3)

- 准则7：**Working software is the primary measure of progress.**
  - 可用的软件是衡量进度的主要指标
- 准则8：**Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.**
  - 敏捷过程提倡可持续的开发；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
- 准则9：**Continuous attention to technical excellence and good design enhances agility.**
  - 对技术的精益求精以及对设计的不断完善将提升敏捷性

敏捷宣言遵循的12条原则(4)

- **准则10：Simplicity--the art of maximizing the amount of work not done--is essential.**
  - 要做到简洁，即尽最大可能减少不必要的工作，这是一门艺术
- **准则11：The best architectures, requirements, and designs emerge from self-organizing teams.**
  - 最佳的架构、需求和设计出自于自组织的团队
    - 自组织团队也叫做自管理团队、或者被授权的团队。团队被授权自己管理他们的工作过程和进度、并且团队决定如何完成工作。
    - 对于经理领导的团队来说，团队成员被分配任务，团队成员只有执行任务的权利。管理者除了要确定目标、方向，团队的组织结构、团队的组成，还需要监督和管理团队的过程和进度，分配任务即确定谁要做什么。这种团队的管理方式，更多的是命令与控制，以及微观管理。
- **准则12：At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.**
  - 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

敏捷方法与过程

归纳：固定节奏、小步快跑、及时反馈、应对变化、快速交付

- 不强调文档，转向强调可运行的软件片段
- 开发者与客户之间频繁沟通
- 快速开发，快速反馈，快速修改，增量交付
- 连续不断的短周期迭代
- 工作节奏稳定，不提倡加班
- 不看重形式和工具，看重“人”和内容，保持简洁

本质：以快速的增量和迭代方式进行软件开发

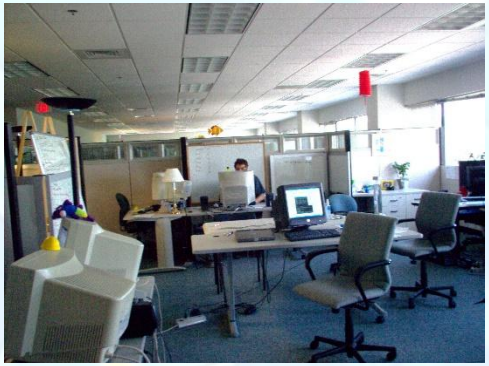
敏捷方法与过程

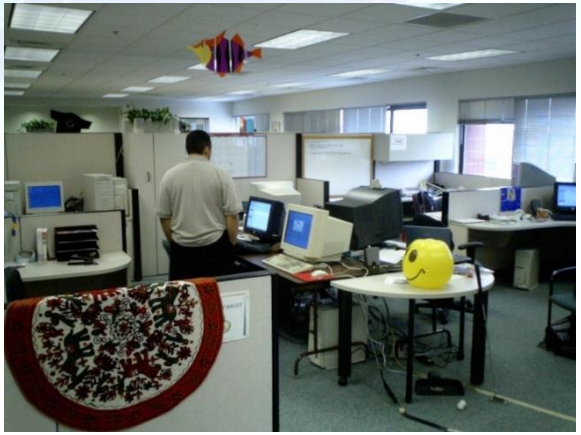
敏捷过程中最重要的因素：人

- Competence 基本能力 (talent, skills, knowledge)
- Common focus 共同目标 (deliver a working software increment)
- Collaboration 精诚合作 (peers and stakeholders)
- Decision-making ability 决策能力 (freedom to control its own destiny)
- Fuzzy problem-solving ability 模糊问题解决能力 (ambiguity and constant changes, today problem may not be tomorrow's problem)
- Mutual trust and respect 相互信任与尊重
- Self-organization 自我组织 (themselves for the work done, process for its local environment, the work schedule)

敏捷方法与过程

敏捷开发的工作场景



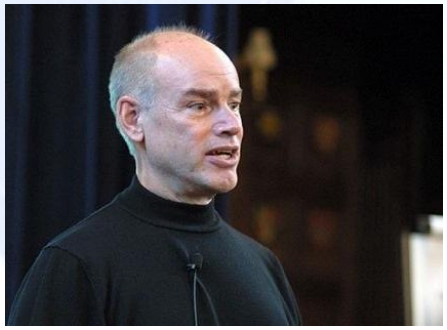


敏捷方法与过程

目前广泛使用的敏捷开发方法论

- XP (eXtreme Programming) (by Kent Beck)
- SCRUM (by Ken Schwaber)
- TDD (Test Driven Development)
- FDD (Feature Driven Development)
- DSDM (Dynamic Systems Development Method)
- Adaptive Software Development (Jim Highsmith)
- Crystal (Alistair Cockburn & Jim Highsmith)
- Pragmatic（实用的） Programming
- AUP (Agile Unified Process)
- Kanban
- ……

Kent Beck



Ken Schwaber

敏捷方法与过程

主要内容

1 敏捷过程模型

2 极限编程(XP)

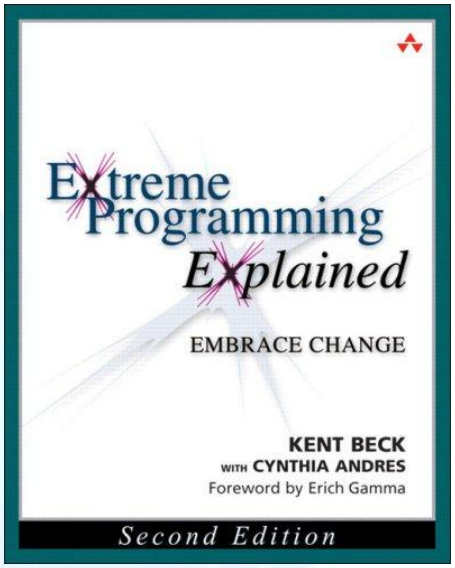
3 Scrum

4 与传统开发过程模型的对比

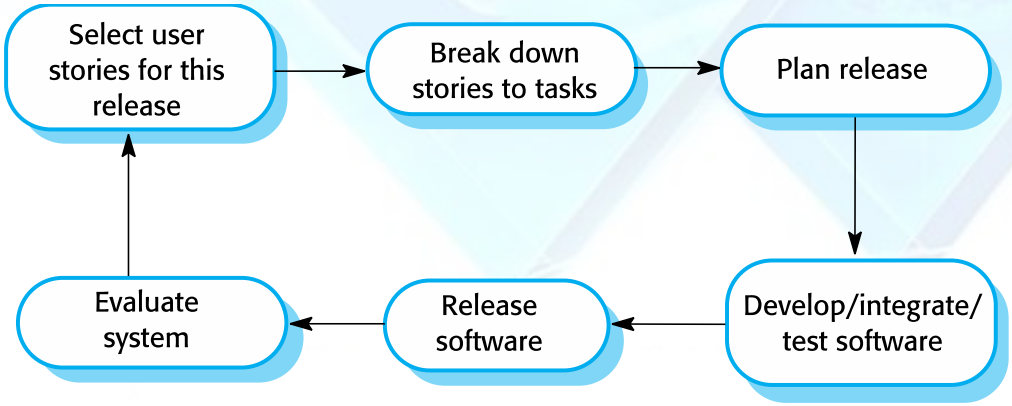
5 敏捷案例分析

敏捷方法与过程

极限编程XP (eXtreme Programming)

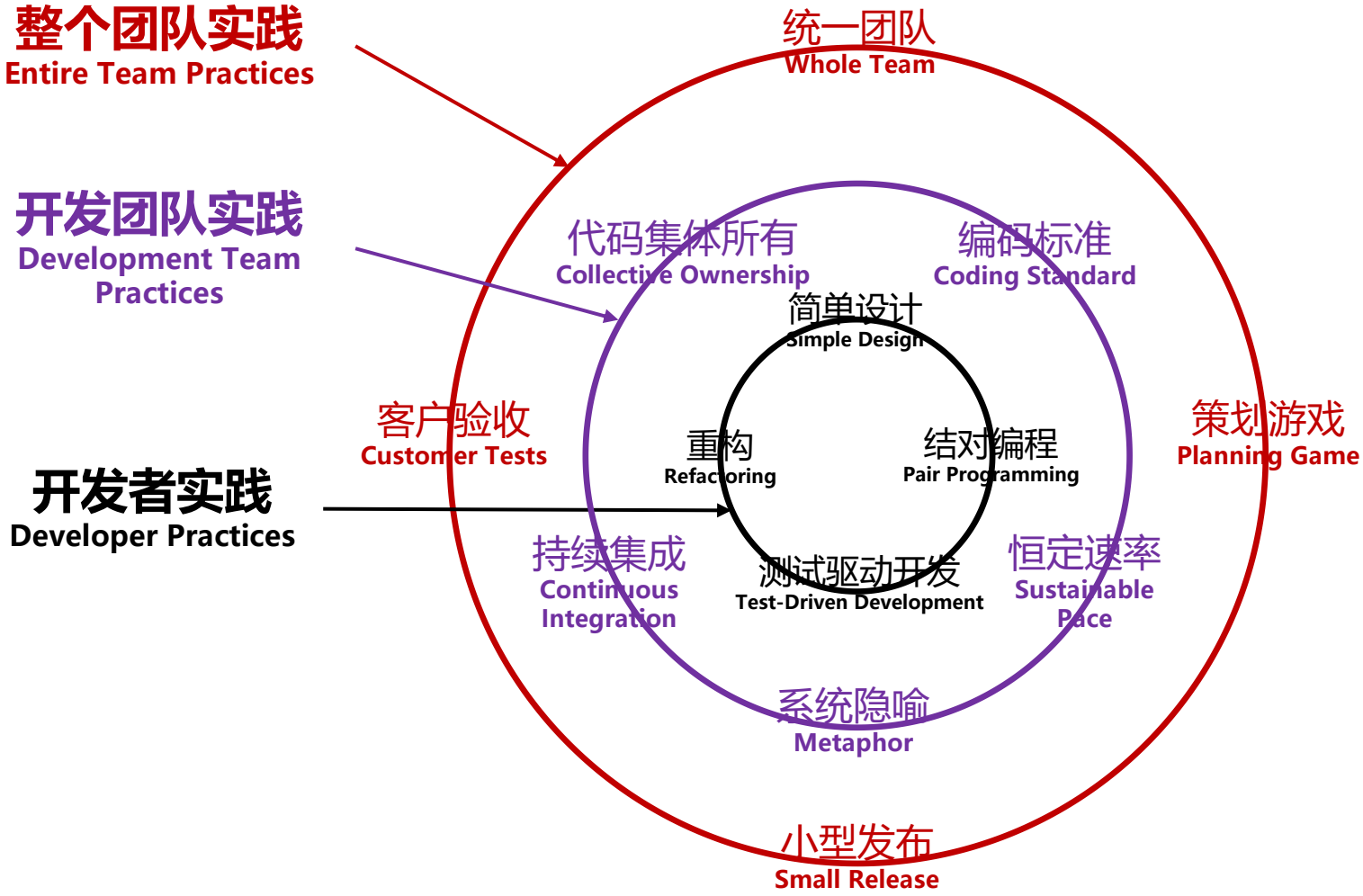


- 极限编程(XP)：一种广泛应用的敏捷开发方法
- XP思想产生于20世纪80年代后期，由Kent Beck完善并实践
  - 一套软件需求与软件开发的实践规则
  - 二者有机融合，高效而保持恒定效率推进软件开发过程
  - 包容客户需求的不断变化，实时响应变化
  - 底层开发人员高效的短期开发行为与项目宏观目标的一致性
- Book:《Extreme Programming Explained: Embrace Change》



敏捷方法与过程

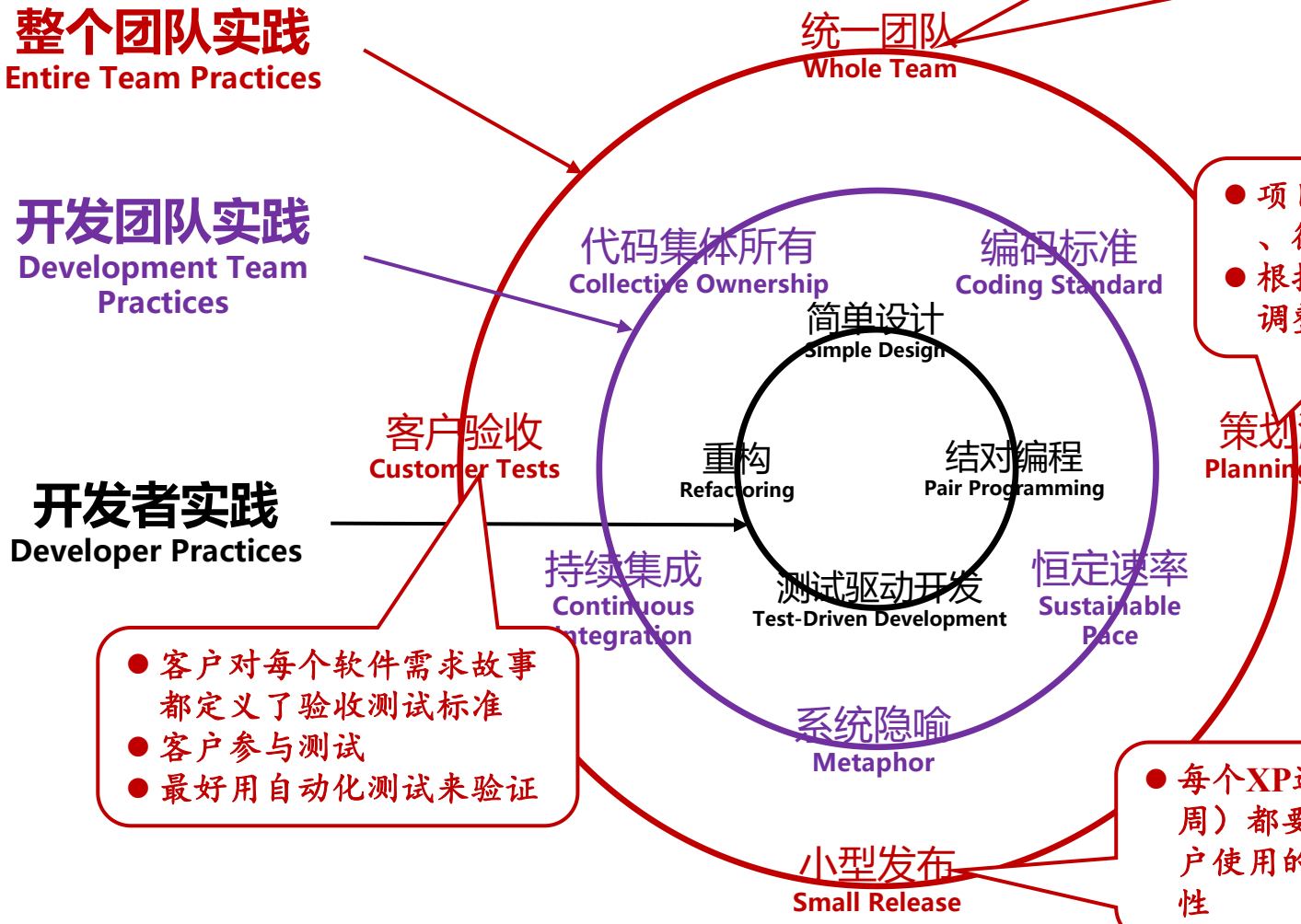
极限编程XP (eXtreme Programming)



敏捷方法与过程

极限编程XP (eXtreme Programming)

- 所有人目标统一，同一开发空间一起工作
- 客户业务代表/程序员/测试人员/系统分析师/项目经理/XP教练
- 团队成员关系平等，无特殊人物



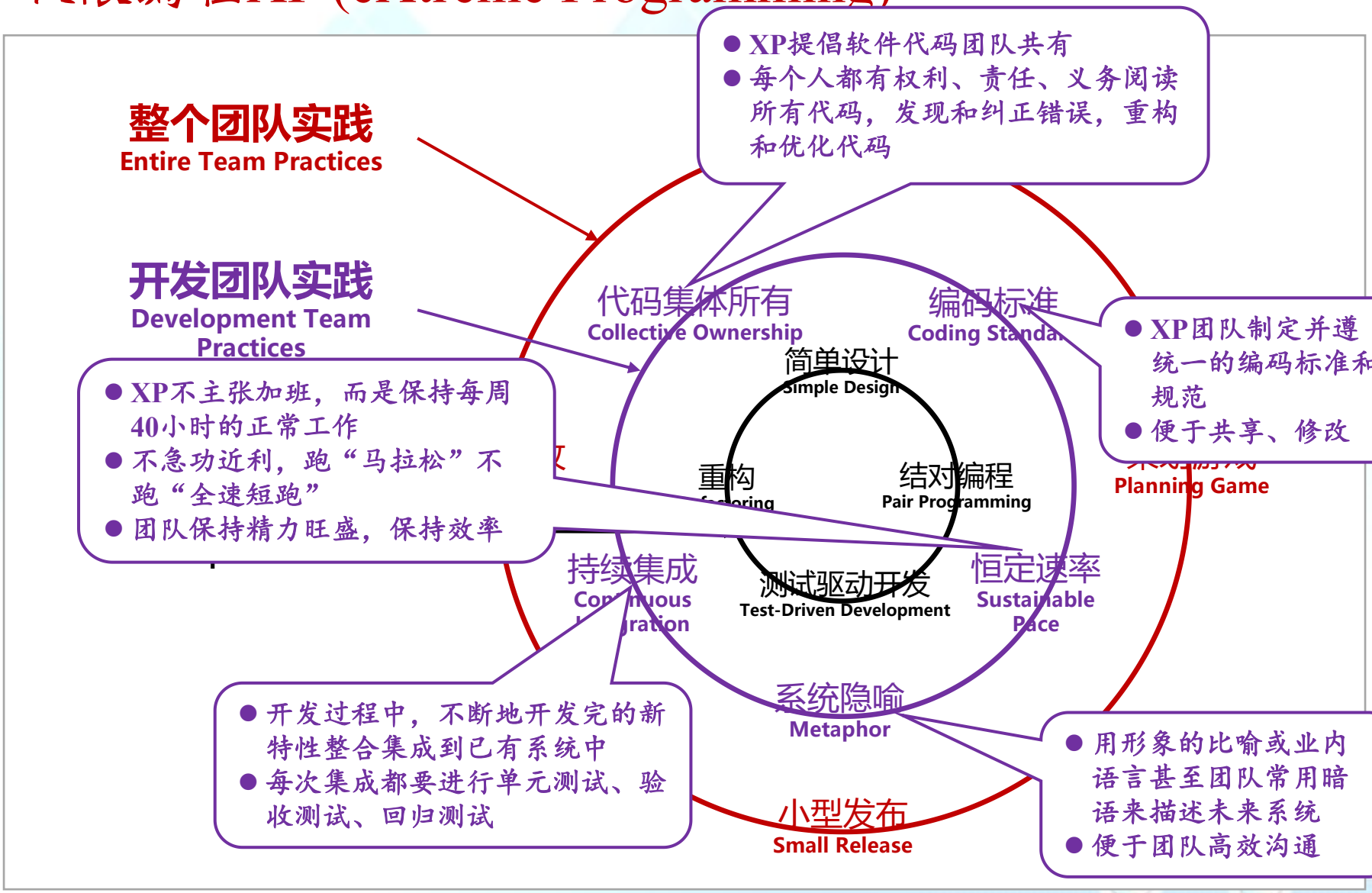
- 项目计划是持续的、循序渐进的
- 根据实施情况不断调整计划

- 客户对每个软件需求故事都定义了验收测试标准
- 客户参与测试
- 最好用自动化测试来验证

- 每个XP迭代周期（1-2周）都要发布可供用户使用的版本或新特性

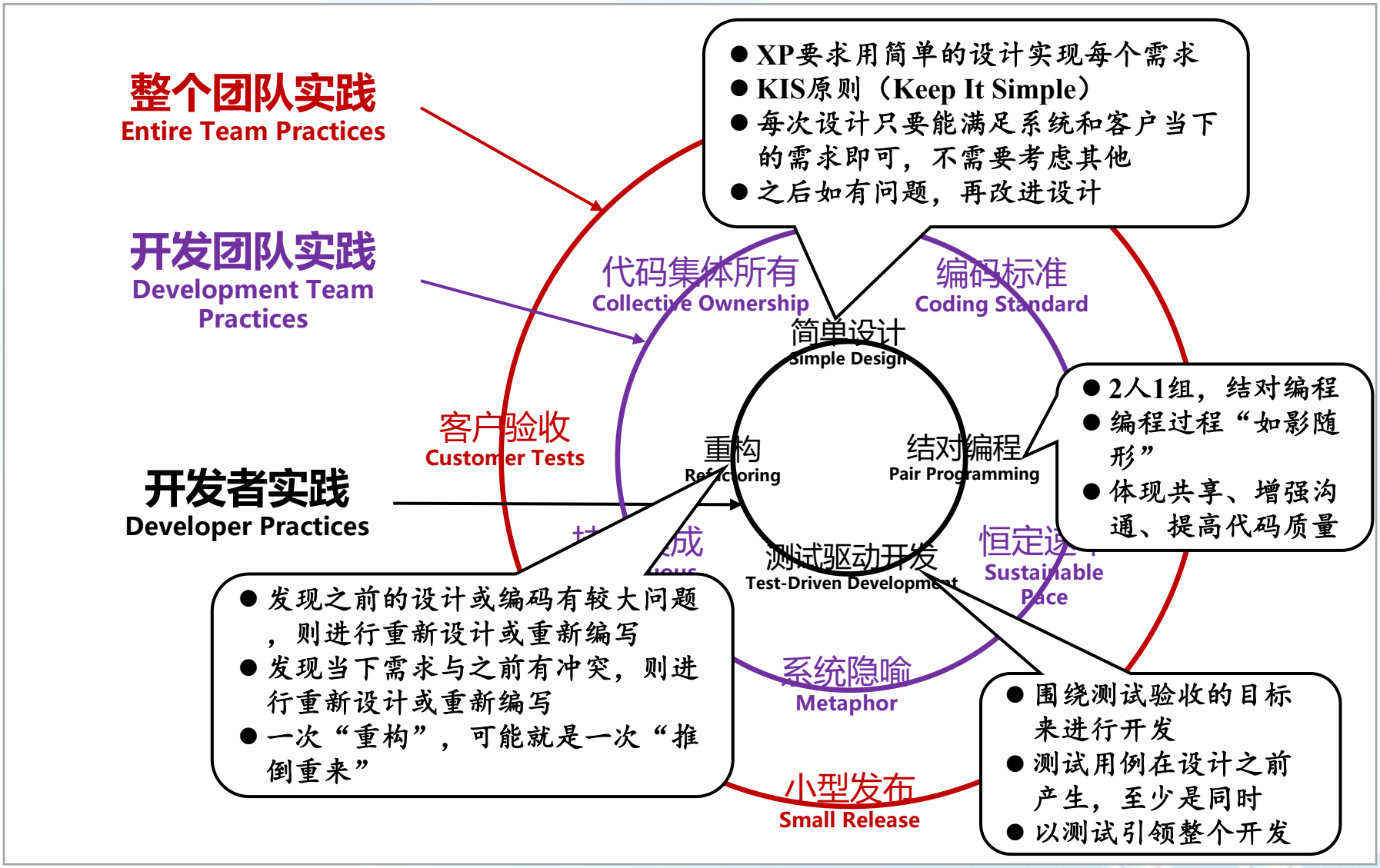
敏捷方法与过程

极限编程XP (eXtreme Programming)



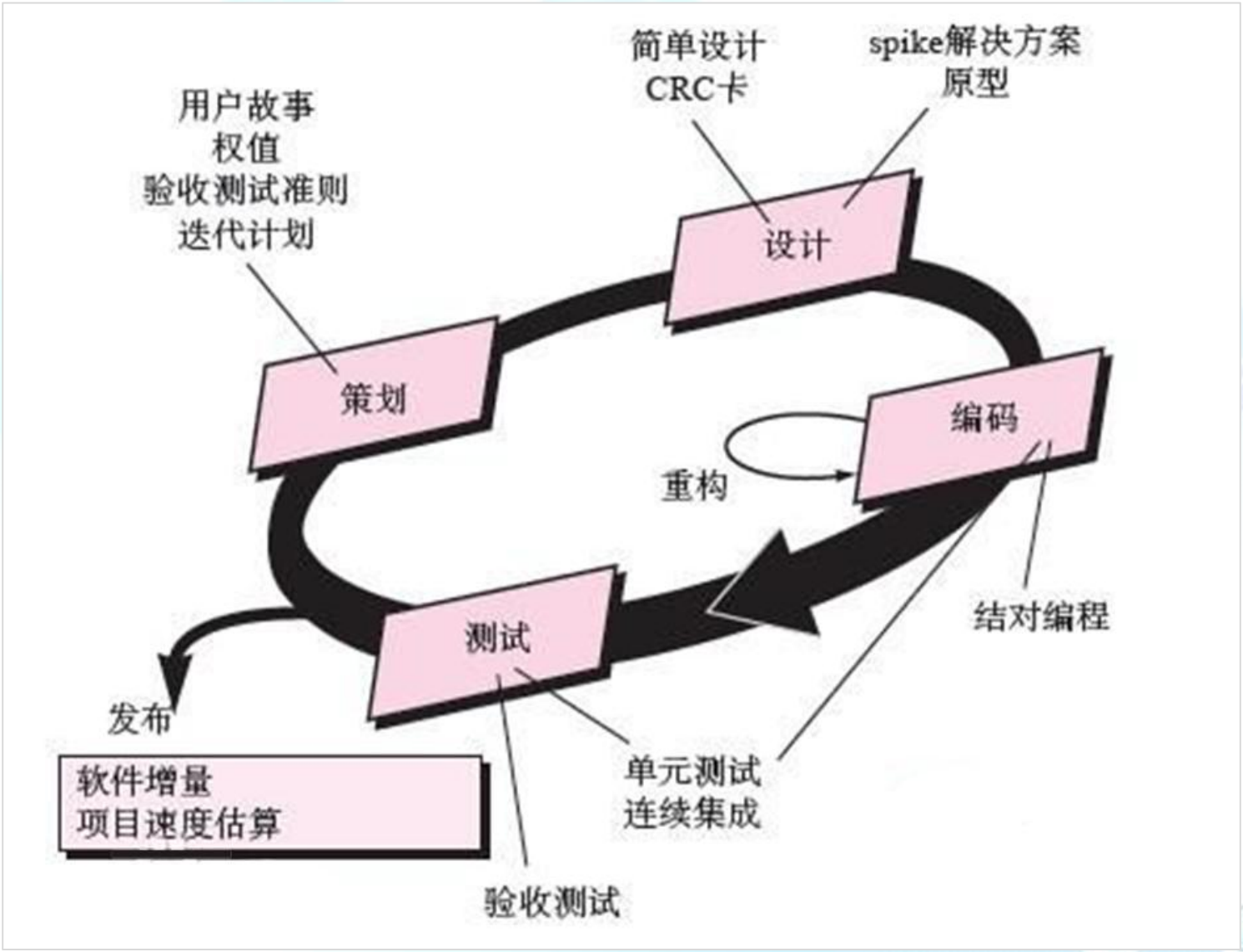
敏捷方法与过程

极限编程XP (eXtreme Programming)



敏捷方法与过程

XP的核心实践方法



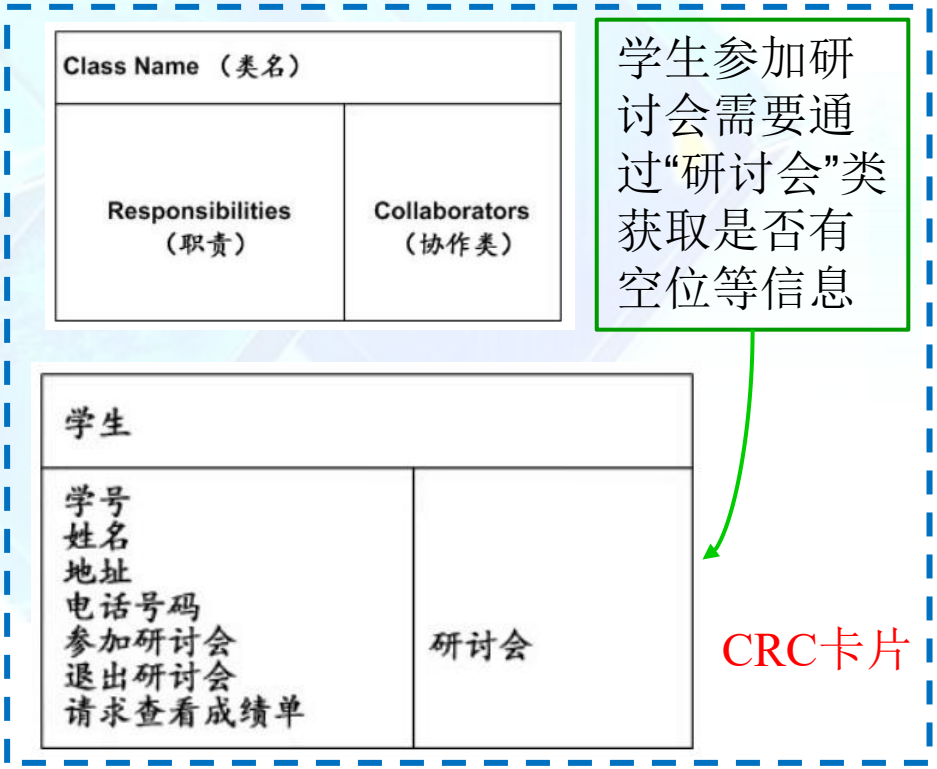
XP Planning：计划阶段

- 倾听客户陈述，形成一组“用户故事(User Stories)”，描述其输出、特性、功能等
- 按照价值或风险排序：客户为每个用户故事指定优先级(Priority)
- XP团队评估各个用户故事，为其指定成本(Cost, 开发周数)，若超过3周，则拆分
- 将若干个用户故事指定为下一次发布的增量，确定发布日期：
  - 所有用户故事
  - 优先级高的用户故事
  - 风险高的用户故事
- 规划整体进度(project velocity)：以怎样的速度开展项目
- 客户可以在开发过程中扩展新故事、去除原有故事、改变优先级、拆分等

敏捷方法与过程

XP Design：设计阶段

- 遵循KIS原则(Keep It Simple)
- 设计模型：面向对象方法，CRC卡片(Class-Responsibility-Collaborator)
- 遇到困难问题，创建“Spike Solutions”(探针原型，通俗的来说就是技术预研)
- 对设计方案不断重构(Refactoring)
  - 遵循用户故事的外特性要求
  - 改善内部结构
  - 消除bug
  - 提高效率
  - 提高易读性



XP Coding & Testing：编码与测试阶段

- **XP Coding**
  - 在编码之前，根据用户故事设计单元测试用例
  - 结对编程(Pair programming)：两人一起编程，实时讨论、实时评审
  - 测试驱动的开发(TDD)：先写测试用例，再写代码
- **XP Testing**
  - 自动化单元测试(Unit test)
  - 持续集成(Continuous Integration)
  - 持续进行回归测试(Regression test)
  - 验收测试(Acceptance test)

Pair Programming：结对编程

- 两个程序员肩并肩地、平等地、互补地进行开发工作
  - 并排坐在一台电脑前，面对同一个显示器，使用同一个键盘，同一个鼠标一起工作
  - 一起分析/设计/写测试用例/编码/单元测试/集成测试/写文档等
- 现实生活中，也存在着类似的搭档关系(Partnership)：
  - 越野赛车(驾驶，领航员)
  - 驾驶飞机(驾驶，副驾驶)
  - 战斗机的编组(长机，僚机)
- 这些任务都有共同点：在高速度中完成任务，任务有较高的技术要求，任务失败的代价很高

敏捷方法与过程

Pair Programming：结对编程

敏捷方法与过程

Pair Programming：结对编程

- 驾驶员(Driver)：控制键盘输入的人
  - 写设计文档，进行编码和单元测试等XP开发流程
- 领航员(Navigator)：起到领航、提醒的作用
  - 审阅驾驶员的文档、驾驶员对编码等开发流程的执行；考虑单元测试的覆盖程度；是否需要和如何重构；帮助驾驶员解决具体的技术问题
- 驾驶员和领航员不断轮换角色，不宜连续工作超过一小时；领航员要控制时间
- 主动参与：任何一个任务都首先是两个人的责任，也是所有人的责任；没有“我的代码/你的代码/他的代码”，只有“我们的代码”
- 只有水平上的差距，没有级别上的差异：尽管可能大家的级别资历不同，但不管在分析、设计或编码上，双方都拥有平等的决策权利

From 《现代软件工程讲义》

Pair Programming：结对编程

- 每人在各自独立设计、实现软件的过程中不免要犯这样那样的错误；在结对编程中，因为有随时的复审和交流，程序各方面的质量取决于一对程序员中各方面水平较高的那一位；这样，程序中的错误就会少得多，程序的初始质量会高很多，这样会省下很多以后修改、测试的时间
  - 在开发层次上，结对编程能提供更好的设计质量和代码质量，两人合作能有更强的解决问题的能力
  - 对开发人员自身来说，结对工作能带来更多的信心，高质量的产出能带来更高的满足感
  - 在心理上，当有另一个人在你身边和你紧密配合，做同样一件事情的时候，你不好意思开小差，也不好意思糊弄
  - 在企业管理层次上，结对能更有效地交流，相互学习和传递经验，能更好地应对人员流动；因为一个人的知识已经被其他人共享

From 《现代软件工程讲义》

小结：为何称为“极限”编程

- 极限：把某件事情做到极致

如果……	发挥到极限就变成……
了解客户的需求很重要	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—TDD
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
其他好方法……	发挥到极限的做法……

http://www.cnblogs.com/xinz/archive/2011/08/07/2130332.html

关于XP的一些反对意见

- **Requirements volatility（需求波动）: customer is an active member of XP team, changes to requirements are requested informally and frequently.客户是XP团队的活跃成员，经常非正式地请求对需求进行更改**
- **Conflicting customer needs（客户需求冲突）: different customers‘ needs need to be assimilated. Different vision or beyond their authority.不同客户的需求需要被理解、采纳，不同的观点、想法或超出他们的权限**
- **Requirements are expressed informally（需求非正规表达）: Use stories and acceptance tests are the only explicit manifestation of requirements. Formal models may avoid inconsistencies and errors before the system is built. Proponents said changing nature makes such models obsolete as soon as they are developed.用户故事和验收测试是需求的唯一明确表现形式；正式模型可以在系统构建之前避免不一致和错误；倡导者们说，自然的变化使得这些模型一经开发就过时了**
- **Lack of formal design（缺乏形式化设计）: XP deemphasizes the need for architectural design. Complex systems need overall structure to exhibit quality and maintainability. Proponents said incremental nature limits complexity as simplicity is a core value. XP不强调架构设计的需要；复杂系统需要整体结构来展示质量和可维护性；倡导者们说，由于简单性是一个核心价值观，渐进性限制了复杂性**

敏捷方法与过程

主要内容

1 敏捷过程模型

2 极限编程(XP)

3 Scrum

4 与传统开发过程模型的对比

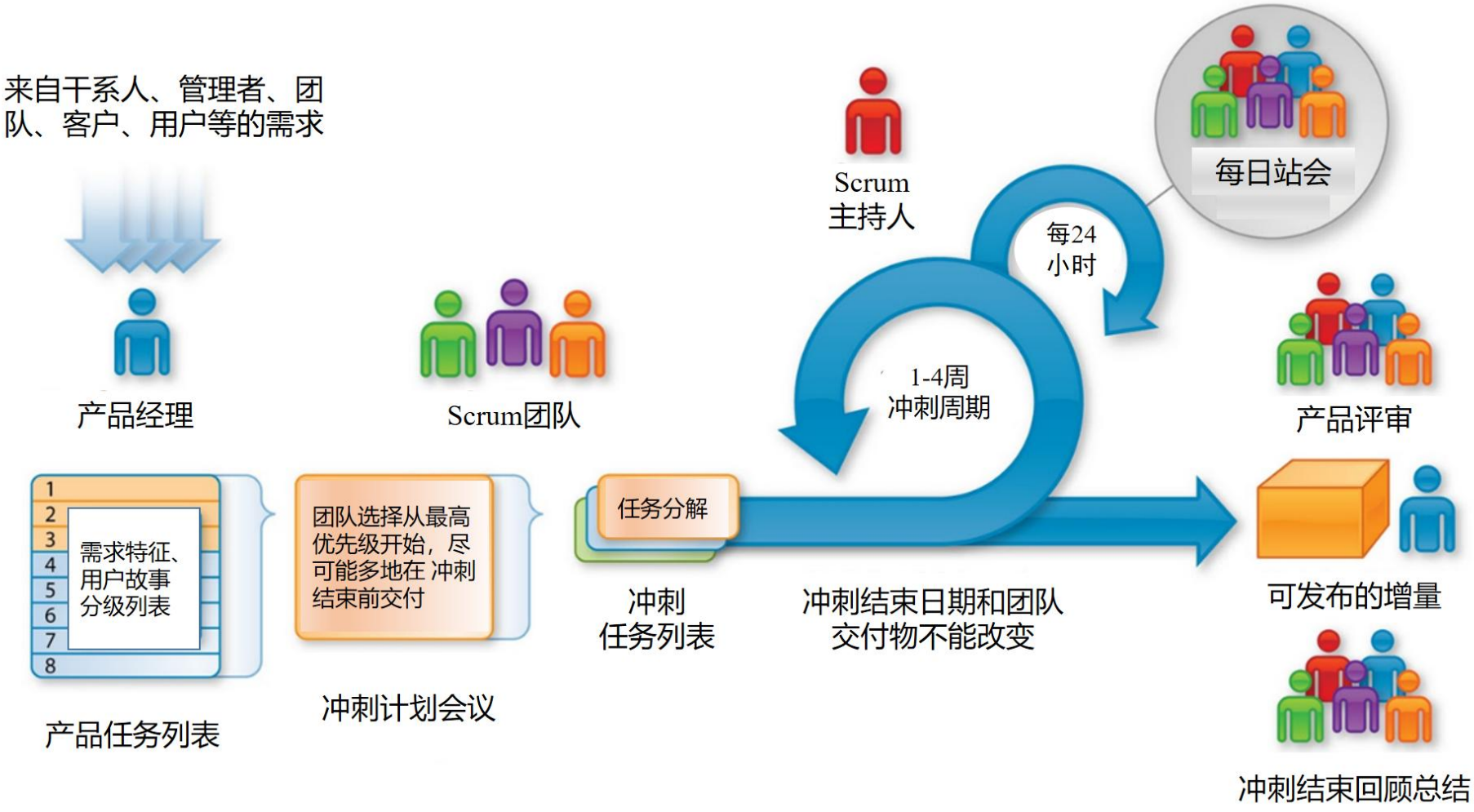
5 敏捷案例分析

敏捷方法与过程

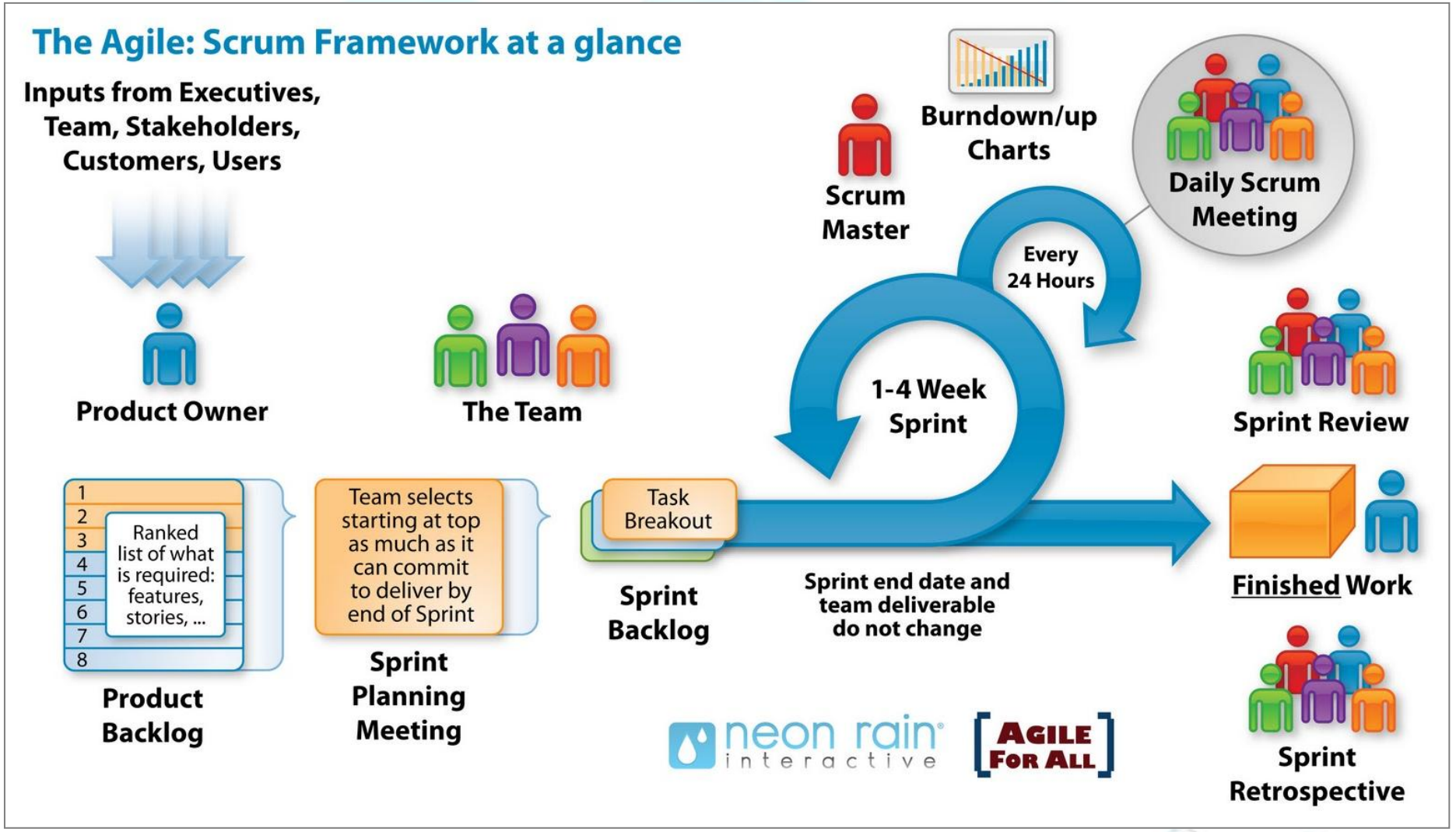
敏捷过程模型的典型代表：Scrum

- **Scrum**
  - 一个敏捷开发框架：增量/迭代的开发过程
  - 1990年代初，Ken Schwaber在其公司使用了一种开发方法Advanced Development Methods（先进开发方法），在90年代中期进一步完善和实践，取名自橄榄球比赛中的术语“争球”（Scrum）
  - 整个开发过程由若干个短的迭代周期组成，一个短的迭代周期称为一个Sprint，每个Sprint的建议长度是1-4周
  - 使用产品Backlog来管理需求，是一个按照商业价值排序的需求列表，列表条目的体现形式通常为用户故事
  - 总是先开发对客户具有较高价值的需求
  - 在Sprint中，Scrum团队从产品Backlog中挑选最高优先级的需求进行开发；挑选的需求在Sprint计划会议上经过讨论、分析和估算得到相应的任务列表(Sprint backlog)
  - 在每个迭代结束时，Scrum团队将提交潜在可交付的产品增量

Scrum的基本过程

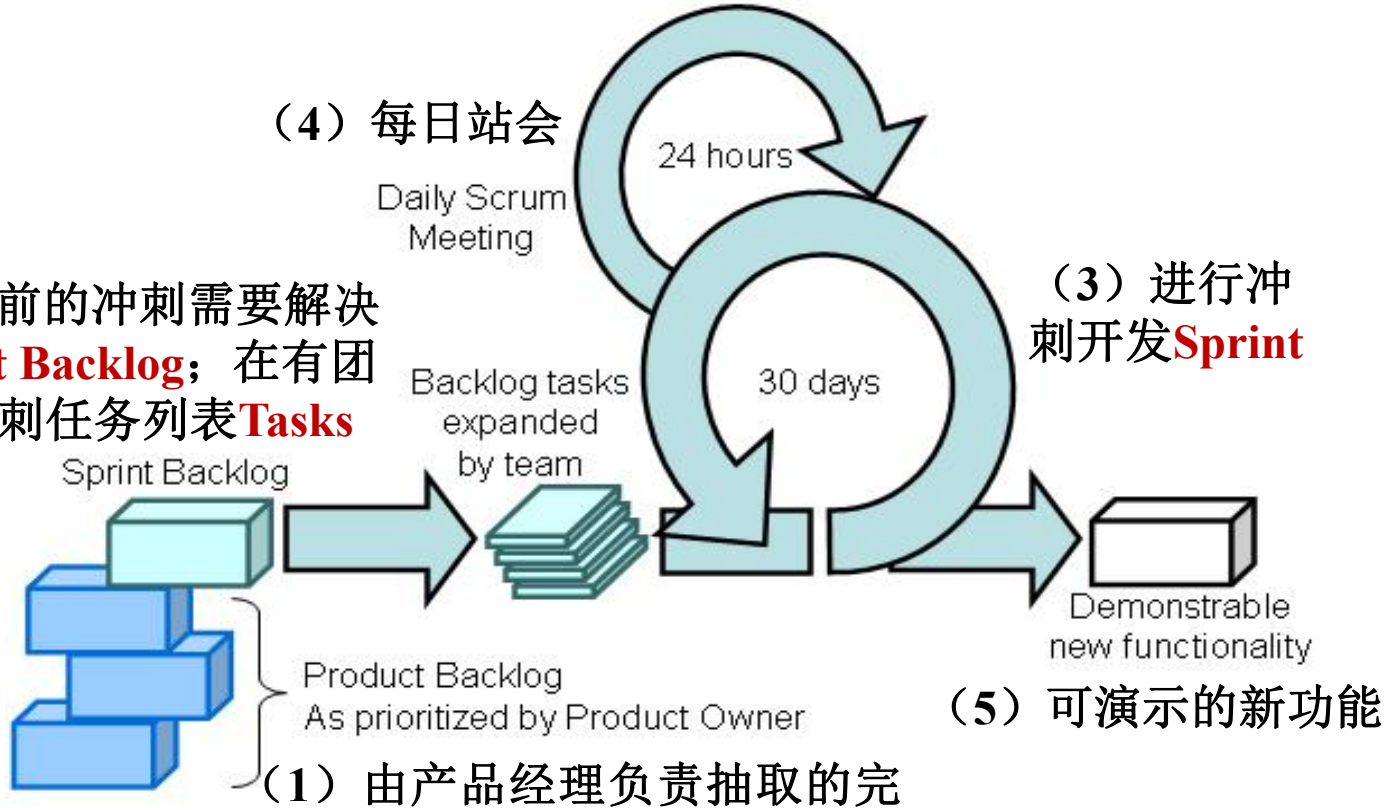

来自干系人、管理者、团队、客户、用户等的需求
产品经理
Scrum团队
Scrum
主持人
每日站会
每24
小时
1-4周
冲刺周期
产品评审
1
2
3
4
5
6
7
8
需求特征、
用户故事
分级列表
团队选择从最高优先级开始，尽可能多地在 冲刺结束前交付
任务分解
冲刺
任务列表
冲刺结束日期和团队
交付物不能改变
可发布的增量
产品任务列表
冲刺计划会议
冲刺结束回顾总结

Scrum的基本过程


The Agile: Scrum Framework at a glance
Inputs from Executives, Team, Stakeholders, Customers, Users
Product Owner
The Team
Scrum Master
Burndown/up Charts
Daily Scrum Meeting
Every 24 Hours
1-4 Week Sprint
Sprint Review
1
2
3
4
5
6
7
8
Ranked list of what is required: features, stories, ...
Product Backlog
Team selects starting at top as much as it can commit to deliver by end of Sprint
Sprint Planning Meeting
Task Breakout
Sprint Backlog
Sprint end date and team deliverable do not change
Finished Work
Sprint Retrospective
neon rain interactive
AGILE FOR ALL

敏捷方法与过程

Scrum的基本过程



（4）每日站会

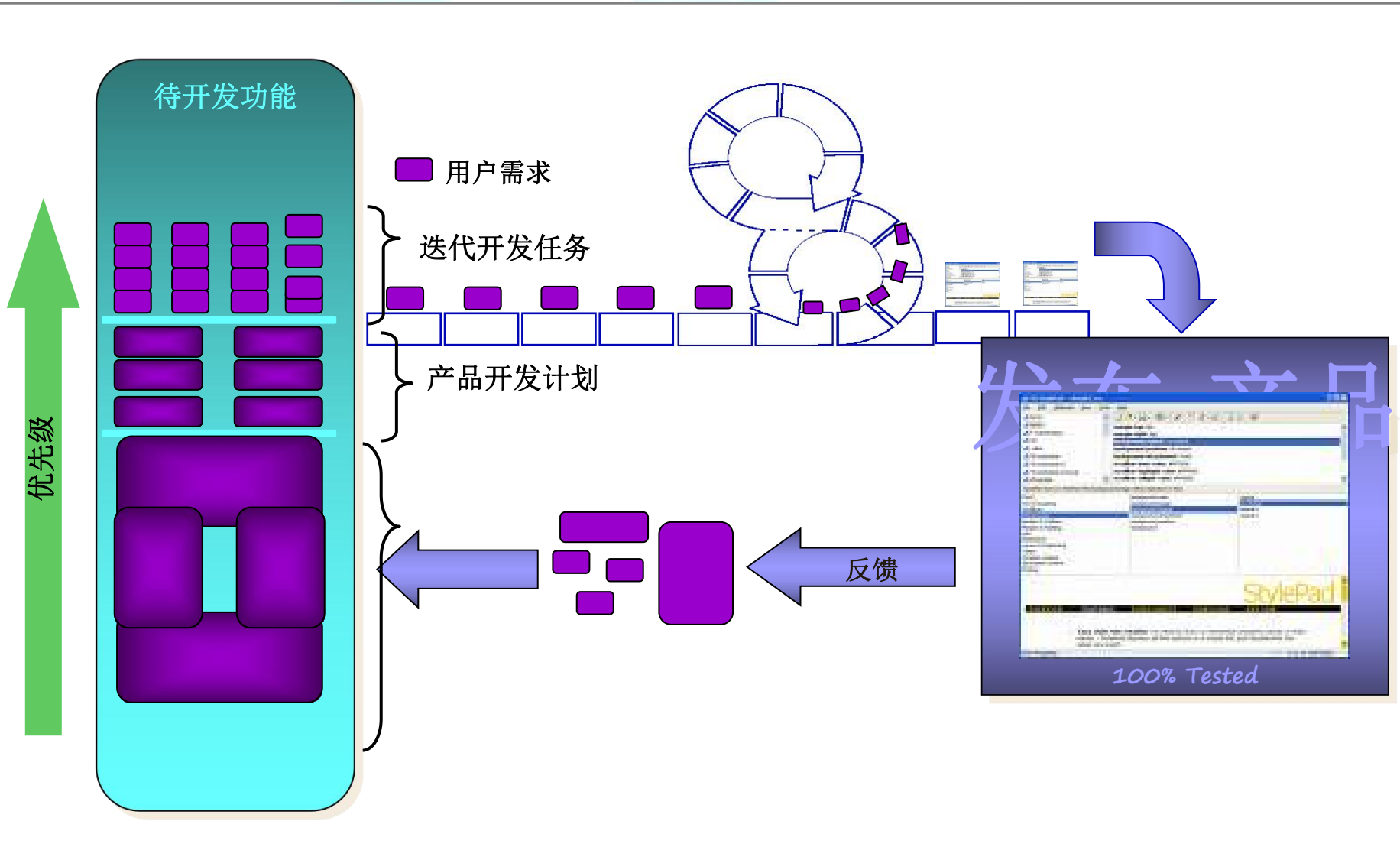
（2）决定当前的冲刺需要解决的事情**Sprint Backlog**；在有团队分解的冲刺任务列表**Tasks**

（3）进行冲刺开发**Sprint**

（5）可演示的新功能

（1）由产品经理负责抽取的完成产品需要做的事情，即按照优先级排序的**Product Backlog**

Scrum的基本过程



Scrum的基本过程

① Product Owner组织会议将计划开发的产品分解成若干开发项（Product Backlog），该列表是有优先级的；该表中的开发项在没有被开发前是可以新增或删除的（引入需求变更）

② Product Backlog中的一个或几个任务项，是一次Scrum Sprint（Scrum冲刺）要开发的任务；1个Sprint一般为2-4周；一旦Sprint启动，在开发完成前是不允许变更需求的

③ 在Sprint开始前，Scrum Master组织Scrum Team会议将Sprint的任务分解为更小的开发单元（Sprint Tasks，列在Sprint Backlog）；Scrum Team成员的开发任务单元就是每个Task

④ Sprint启动后，每天需要召开一次会议（Daily Scrum Meeting），一般不超过15分钟，每人简短陈述3句话：上次Scrum例会后做了什么？遇到了什么问题？下次Scrum例会前计划做什么？（注意：提出的问题在例会上不做任何讨论）

⑤ Sprint结束后，展示产品新功能；并做Sprint评审和回顾，即Sprint Review Meeting和Sprint Retrospective Meeting

上述Sprint过程循环进行，直到Product Backlog列表空了为止

Scrum的3人、4表、5会、6事

- 人 -- 3种角色：
  - Product Owner（产品拥有者/产品负责人）
  - Scrum Master（Scrum主持人）
  - Scrum Team（Scrum团队成员，一般5-10人）
- 表 -- 4种列表：
  - Product Backlog（产品任务列表）
  - Sprint Backlog（冲刺任务列表）
  - Task Board（任务墙）
  - Burn down/up Charts（燃尽图）
- 会 -- 5种会议：
  - 产品计划发布会（产出Product Backlog列表）
  - 冲刺计划会议（产出Sprint Backlog列表）
  - 每日站会（督促激励个体工作）
  - 冲刺结束时的评审会（展示产品新功能）
  - 冲刺结束后的回顾会（总结冲刺过程的经验教训）
- 事 -- 6种活动：
  - Sprint（冲刺）+5种会议

Product Backlog（产品待办事项列表）样例

每个 Sprint 的新估算									
优先级	事项	细节（wiki 链接）	初始规模估算	1	2	3	4	5	6
1	作为买家，我想把书放入购物车（见 wiki 页面用户界面草图）		5						
2	作为买家，我想从购物车中删除书		2						
3	改进事务处理效率（见 wiki 页面目标性能指标）		13						
4	探讨加速信用卡验证的解决方法（见 wiki 页面目标性能指标）		20						
5	将所有服务器升级到 Apache 2.2.3		13						
6	分析并修复订单处理脚本错误（错误号：14834）		3						
7	作为购物者，我想创建并保存愿望表		40						
8	作为购物者，我想增加或删除愿望表中的条目		20						

Sprint Backlog（Sprint待办任务列表）样例

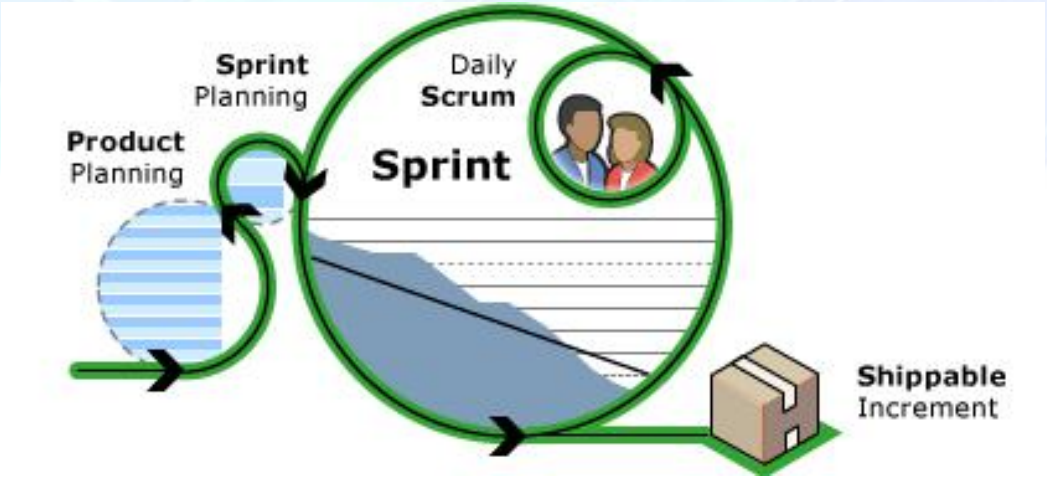
每日结束时所剩余工作量的最新估计									
产品待办事项列表事项	Sprint 中的任务	志愿者	初始工作量估计	1	2	3	4	5	6
作为买家，我想把书放入购物车	修改数据库		5						
	创建网页（UI）		8						
	创建网页（JavaScript 逻辑）		13						
	写自动化验收测试		13						
	更新买家帮助网页		3						
	…								
改进事务处理效率	合并 DCP 代码并完成分层测试		5						
	完成 pRank 的机器顺序		8						
	把 DCP 和读入器改为用 pRank HTTP API		13						

Scrum中的三种角色

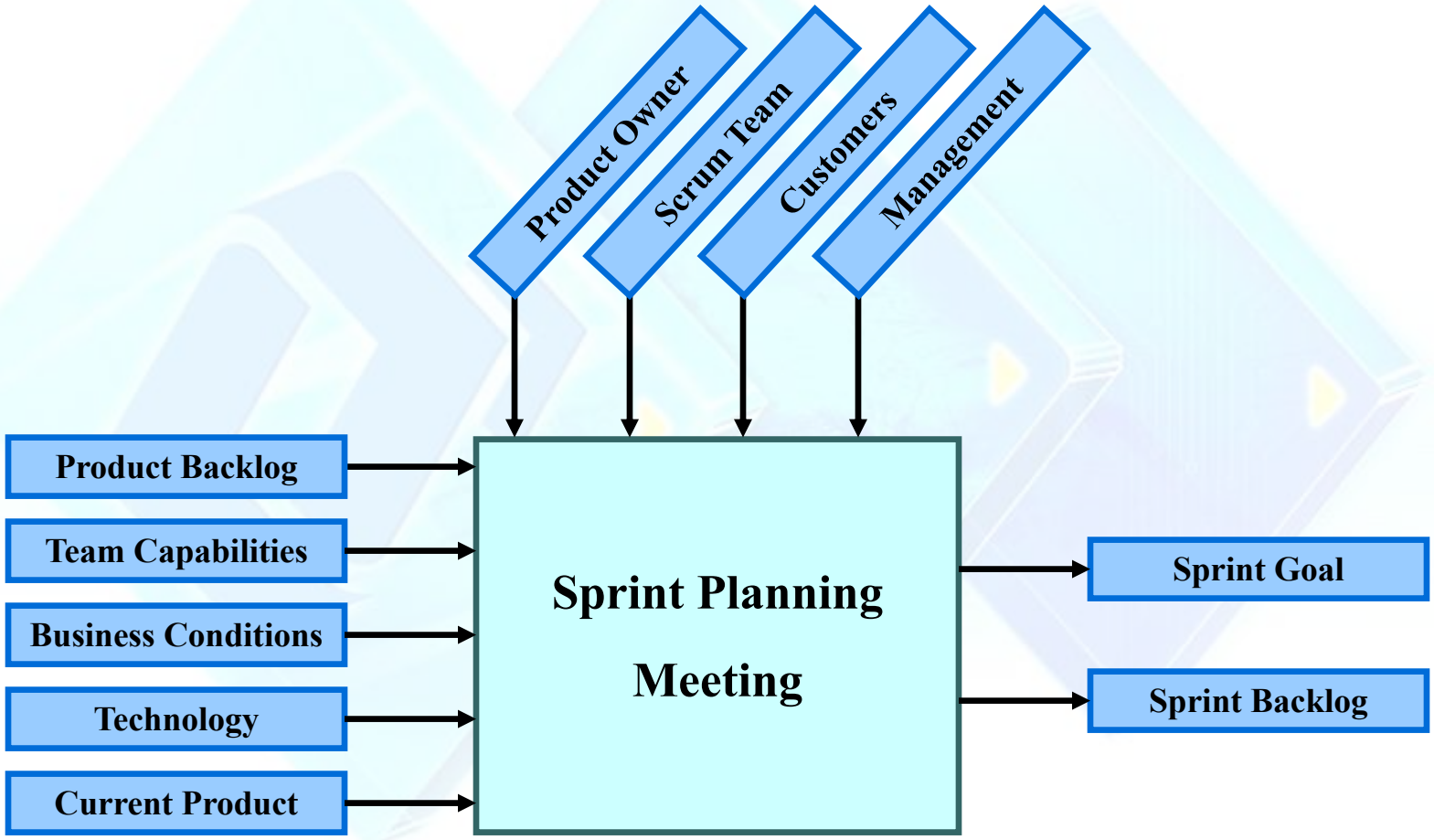
- **Product Owner(产品负责人)：确定产品的功能，负责维护产品Backlog、deadline、priority、ROI（投资回报率）；验收结果**
- **Scrum Master（团队leader）：保证开发过程按计划进行；组织每日站会、Sprint计划会议、Sprint评审会议和Sprint回顾会议；通过外在/内在协调，确保团队资源完全可被利用并且全部是高产出的**
- **Scrum Team（Scrum团队）：在每个Sprint中将产品Backlog中的条目转化成为潜在可交付的功能增量；规模在5-10人；具备交付产品增量所需要的各种技能**

Scrum中的六项活动

- Sprint (冲刺)：代表一个2-4周的迭代
- 发布计划会议(Release Planning Meeting) → Product Backlog
- Sprint计划会议(Sprint Planning Meeting) → Sprint Backlog
- 每日站会(Daily Scrum Meeting)
- Sprint评审会(Sprint Review Meeting)
- Sprint回顾会议(Sprint Retrospective Meeting)



Sprint计划会议


Product Owner
Scrum Team
Customers
Management
Product Backlog
Team Capabilities
Business Conditions
Technology
Current Product
Sprint Planning Meeting
Sprint Goal
Sprint Backlog

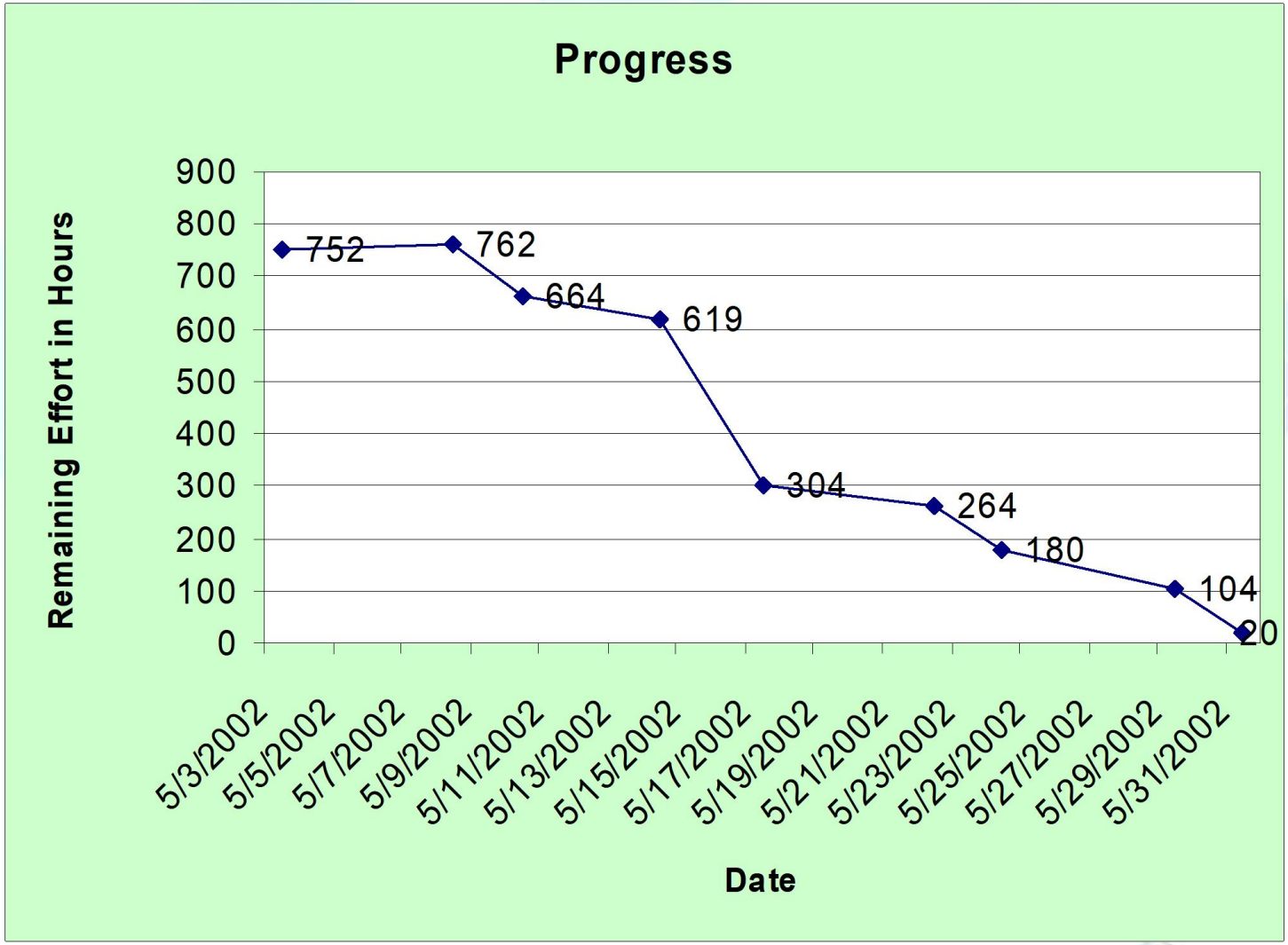
每日站会

- 团队成员站着开会 — 强迫在短时间(15分钟)内高效率讨论问题，保持注意力集中
- 强迫每个人向同伴报告进度, 迫使大家把问题摆在明面上，如果每日站会定于早上开会的话，则每个团队成员应该回答下面问题：
  - 我昨天做了什么(What have you done since yesterday?)
  - 我今天要做什么(What are you planning to do today?)
  - 我碰到了哪些问题(Do you have any problems that would prevent you from accomplishing your goal?)
- 用简明的图表展现整个项目的进度，这个图最好放在大家工作的环境中，或者每天传达给各个成员

敏捷方法与过程

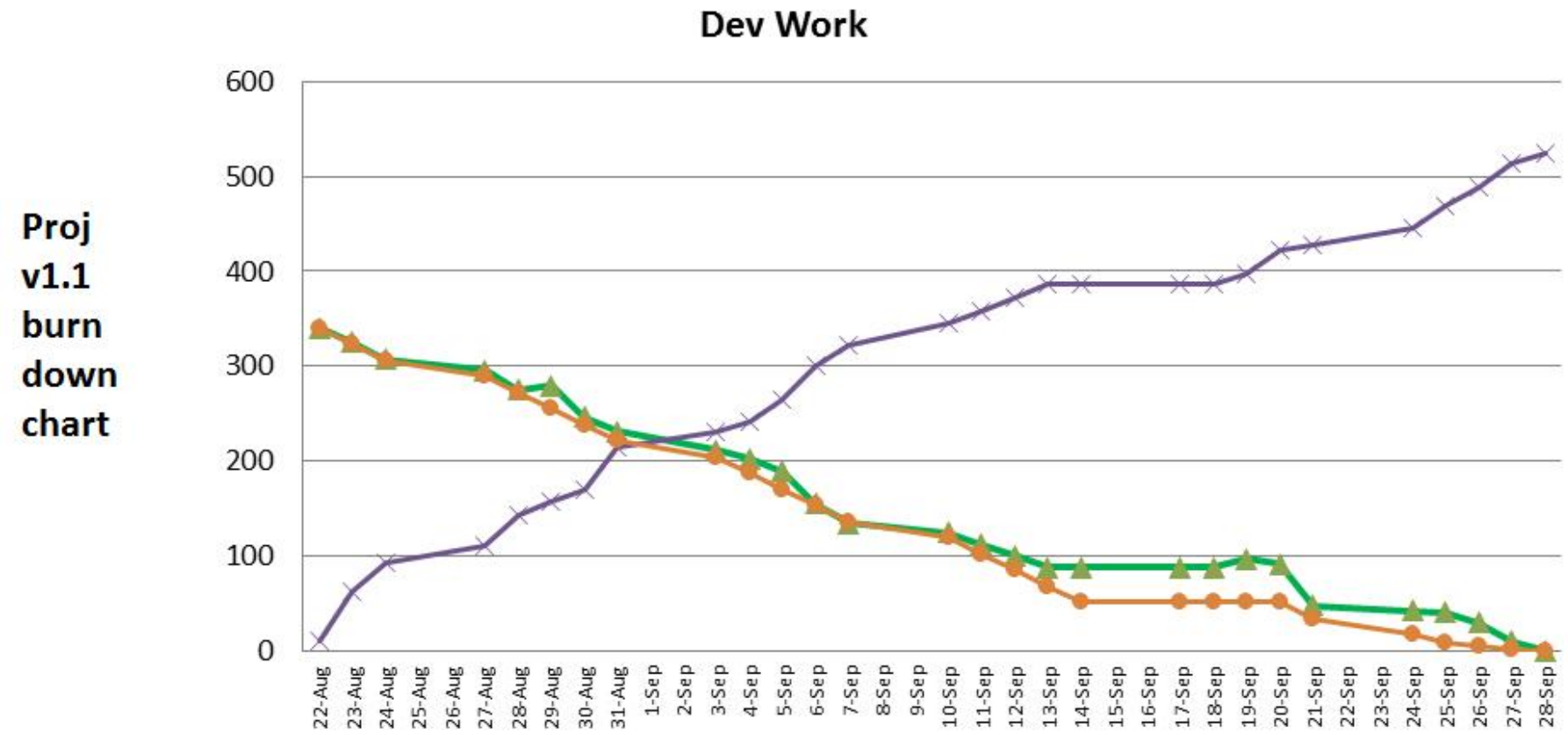
每日站会

Sprint Burndown Chart（Sprint燃尽图）



敏捷方法与过程

Sprint Burndown Chart（Sprint燃尽图）

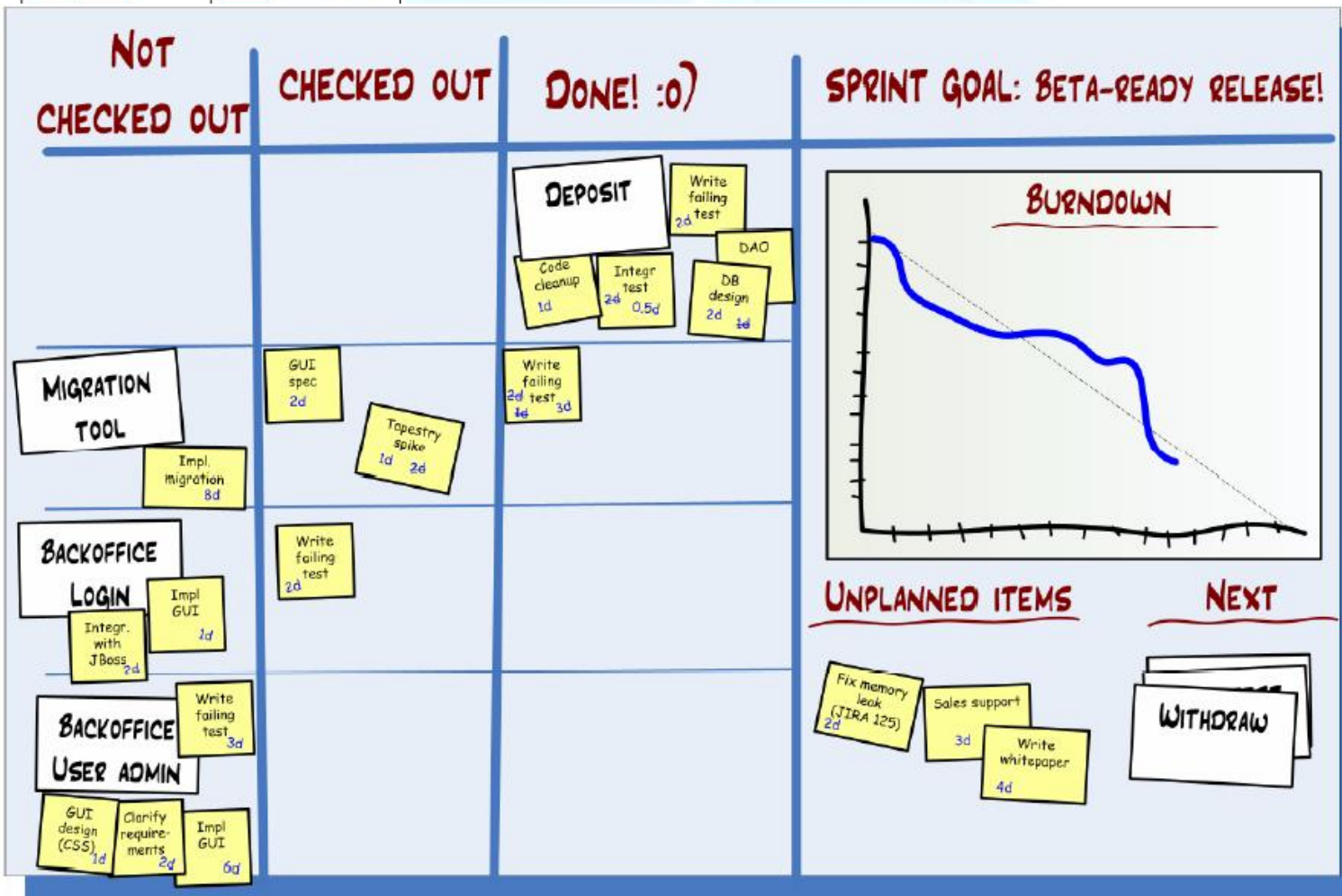


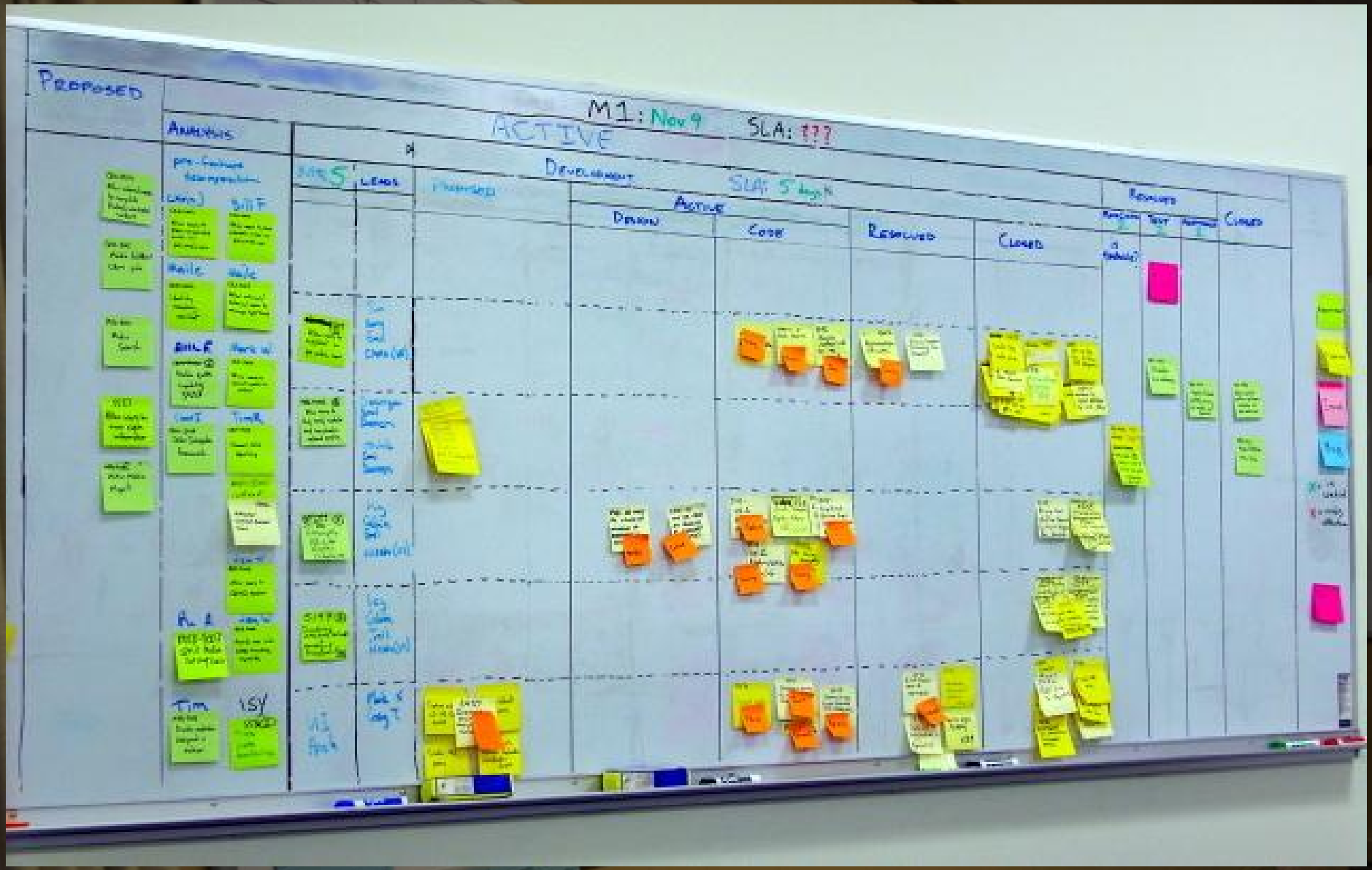
	22-Aug	23-Aug	24-Aug	27-Aug	28-Aug	29-Aug	30-Aug	31-Aug	3-Sep	4-Sep	5-Sep	6-Sep	7-Sep	10-Sep	11-Sep	12-Sep	13-Sep	14-Sep	17-Sep	18-Sep	19-Sep	20-Sep	21-Sep	24-Sep	25-Sep	26-Sep	27-Sep	28-Sep
remain hour	340	325	307	295	275	279	246	231	211	202	189	155	134	124	112	100	87	87	87	87	96	91	47	42	40	29	9	0
complete hour	9	61	93	111	143	157	169	215	230	242	264	301	322	346	358	372	386	386	386	386	397	422	427	445	468	488	513	524
projected remaining hour	340	323	306	289	272	255	238	221	204	187	170	153	136	119	102	85	68	51	51	51	51	51	34	17	8	4	1	0

敏捷方法与过程

Task Board（任务墙）

Feature	Tasks		
	Waiting	In Progress	Done
A			
B			
C			
D			
E			




PROPOSED
ACTIVE
M1: Nov 9
SLA: ???
Analysis
Development
Active
Done
Code
Resolved
Closed
Resolved
Test
Closed

敏捷方法与过程

归纳：敏捷开发过程的最佳实践(XP & Scrum)

- 用户故事(User Story)
- 结对编程(Pair Programming)
- 测试驱动的开发(TDD)
- 持续集成(Continuous Integration)
- 验收测试(Acceptance Testing)
- 冲刺/迭代(Sprint / Iteration)
- 产品清单/冲刺清单(Product backlog / Sprint backlog)
- 燃尽图(Burndown chart)
- 每日站会(Daily Stand-up Meeting)

归纳：敏捷开发过程的最佳实践(XP & Scrum)

XP 与 Scrum的明显区别

编号	区别点	XP	Scrum
1	迭代周期长度	1-2周	1-4周
2	迭代中需求变化	允许	不允许
3	迭代选取的用户故事优先级	严格按照优先级来开发	根据具体情况允许跳跃选取
4	迭代过程控制	严格控制，必须TDD、配对编程等	比较随意

敏捷方法与过程

主要内容

1 敏捷过程模型

2 极限编程(XP)

3 Scrum

4 与传统开发过程模型的对比

5 敏捷案例分析

敏捷方法与过程

一句话归纳各类过程模型

- 瀑布模型：将全部需求以整体方式向前推进，无迭代
  —— 基本模型
- 增量模型：将需求分成多份，串行推进，无迭代
  —— 串行的瀑布
- RAD模型：将需求分成多份，可并行推进，无迭代
  —— 并行的瀑布
- 原型模型：始终结果可见，不断迭代修正原型，直到开发完成
  —— 基本模型
- 螺旋模型：按瀑布阶段划分，各阶段分别迭代(原型+风险分析)
  —— 原型+瀑布
- 敏捷模型：将需求分成尽量小的碎片，以碎片为单位进行高速迭代
  —— 增量+迭代+原型

敏捷方法与过程

各类过程模型

适用方式 客观因素	敏捷 (Agile)	计划驱动 (Plan-driven)	形式化的开发方法 (Formal Method)
产品可靠性要求	不高，容忍经常出错	必须有较高可靠性	有极高的可靠性和质量要求
需求变化	经常变化	不经常变化	固定的需求，需求可以建模
团队人员数量	不多	较多	不多
人员经验	有资深程序员带队	以中层技术人员为主	资深专家
公司文化	鼓励变化，行业充满变数	崇尚秩序，按时交付	精益求精
实际例子	写一个微博网站	开发下一版本办公软件；给商业用户开发软件	开发底层正则表达式解析模块；科学计算；复杂系统核心组件
用错方式的后果	用敏捷的方法开发登月火箭控制程序，前N批宇航员都挂了	用敏捷方法，商业用户未必受得了2-4周一次更新的频率	敏捷方法的大部分招数都和这类用户无关，用户关心的是：把可靠性提高到99.99%，不要让微小的错误把系统搞崩溃！

敏捷方法与过程

主要内容

1 敏捷过程模型

2 极限编程(XP)

3 Scrum

4 与传统开发过程模型的对比

5 敏捷案例分析

敏捷过程的小案例

- 公司有个项目组开发一个网游物品交易平台
- 该平台是典型的互联网项目，在开工的时候客户对功能需求还不明确，但需要快速推出抢占市场，正是最适合敏捷过程的项目
- 在项目伊始，业务分析师和客户做了深入的谈话，了解他的商业构想，他的盈利模式，搞清楚宏观的结构，然后思考并整理获得的结果，花1-2天时间将客户需求大略整理为几十个用户故事
- 这些用户故事并不完善，不足以做好整个系统；但对于我们开始项目的第一阶段，已经足够了

敏捷过程的小案例

- 敏捷方法希望快速交付可用的软件，实现软件的快速交付是通过迭代来完成
- 在迭代开始前，由一组有经验的开发人员大致评估一下用户故事，标记出不同的难度和风险，并提出问题供业务分析师来获得更详细的信息，业务分析师会和相关涉众去讨论
- 然后业务分析师将推荐优先级最高的一组用户故事给客户来挑选，客户可以选择这些用户故事，或者指出从他的视角看到的优先级更高的用户故事；这些将成为下一个迭代的内容
- 客户看到每个迭代交付的可运行的软件后或者得到用户反馈后，常常会有新的想法冒出来；有些想法是好的，有些想法就属于看到别家网站有这个功能，不假思索提出的功能
- 这些不同的需求都需要经过认真的分析，找出哪些是值得我们立即考虑的、哪些是不用急迫的去实现的

敏捷过程的小案例

- 有一次和客户谈话时，他说到希望增加拍卖功能
- 那么，我们为什么需要拍卖呢？客户说希望让用户拍卖物品以获得最高价格
- 经过考虑，我们发现网游物品的实时性和唯一性决定了系统不适合使用拍卖机制；拍卖的时效性无法满足实时交易的要求，因此，用户最终放弃了这个特性
- 另一次，客服人员提出增加一个查询用户交易的功能，而此时我们有其他更加重要的功能需要先去考虑
- 查询用户交易功能可以由技术人员临时通过数据库直接代为查询，因为项目运营初期交易不是很多，暂时还不需要专门的后台功能来支持客服的工作
- 所以把这个需求卡片一直贴在墙壁上，始终没有排到最高的优先级

敏捷过程的小案例

- 用户故事的跟踪和管理是由项目经理来进行：每个迭代跟踪卡片的进展，是否已经开始实现？是否已经完成代码开发？是否已经开始功能测试？
- 不同的卡片在迭代前都会评估为不同的大小，一般分为大中小三级
- 每个迭代内分析好恰好足够下一个迭代开发的需求，就是业务分析师的主要工作内容；业务分析师的需求分析工作在上一个迭代完成，包括需求的了解、分析、评估和排列优先级
- 在每个迭代开始的时候，由业务分析师主持召开迭代计划会议，在会议上向所有的程序员解释这个迭代要完成的用户故事，然后由程序员自由提问，知道他们能够获得足够开始实现该功能的信息
- 在程序员完成一个用户故事后，业务分析师还要来代表客户做功能验收测试，查看是否完成了预计的功能，是否有程序员还没有想到的异常情况；如果存在问题需要退回给程序员继续完成；这在一定程度上保证了系统完成的需求不偏离客户的要求