第三章 敏捷软件开发

一、关于敏捷

- 敏捷是什么？为什么敏捷开发会迅速发展？

过程和方法对于项目的结果只有次要的影响。首要的影响是人。软件工程师有共同的观点：唯一真正重要的工作产品是在合适时间提交给客户的可运行软件增量

1、敏捷联盟简介

2001年初，一批业界专家聚集在一起概括出了一些可以让软件开发团队具有快速工作、响应变化能力的价值观和原则，他们称自己为敏捷(Agile)联盟。在随后的几个月中，他们创建出了一份价值观声明。也就是敏捷联盟宣言

敏捷联盟宣言

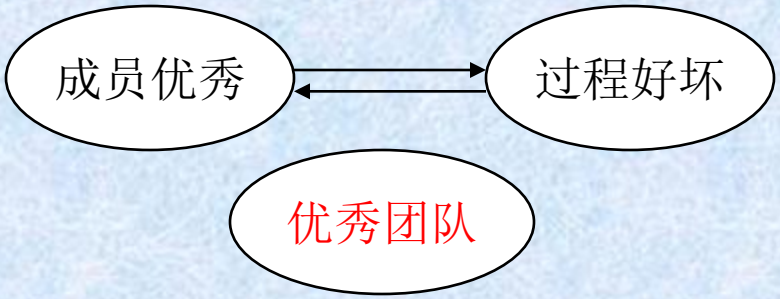
我们正在通过亲身实践以及帮助他人实践，揭示更好的软件开发方法。通过这项工作，我们认为：

个体和交互	胜过	过程和工具
可以工作的软件	胜过	面面俱到的文档
客户合作	胜过	合同谈判
响应变化	胜过	遵循计划

虽然右项也有价值，但我们认为左项具有更大的价值。

个体和交互胜过过程和工具

- 人是获得成功的最为重要的因素。
- 合作、沟通以及交互能力要比单纯的编程能力更为重要。



可以工作的软件胜过面面俱到的文档

- 没有文档的软件是一种灾难。
- 过多的文档比过少的文档更糟。

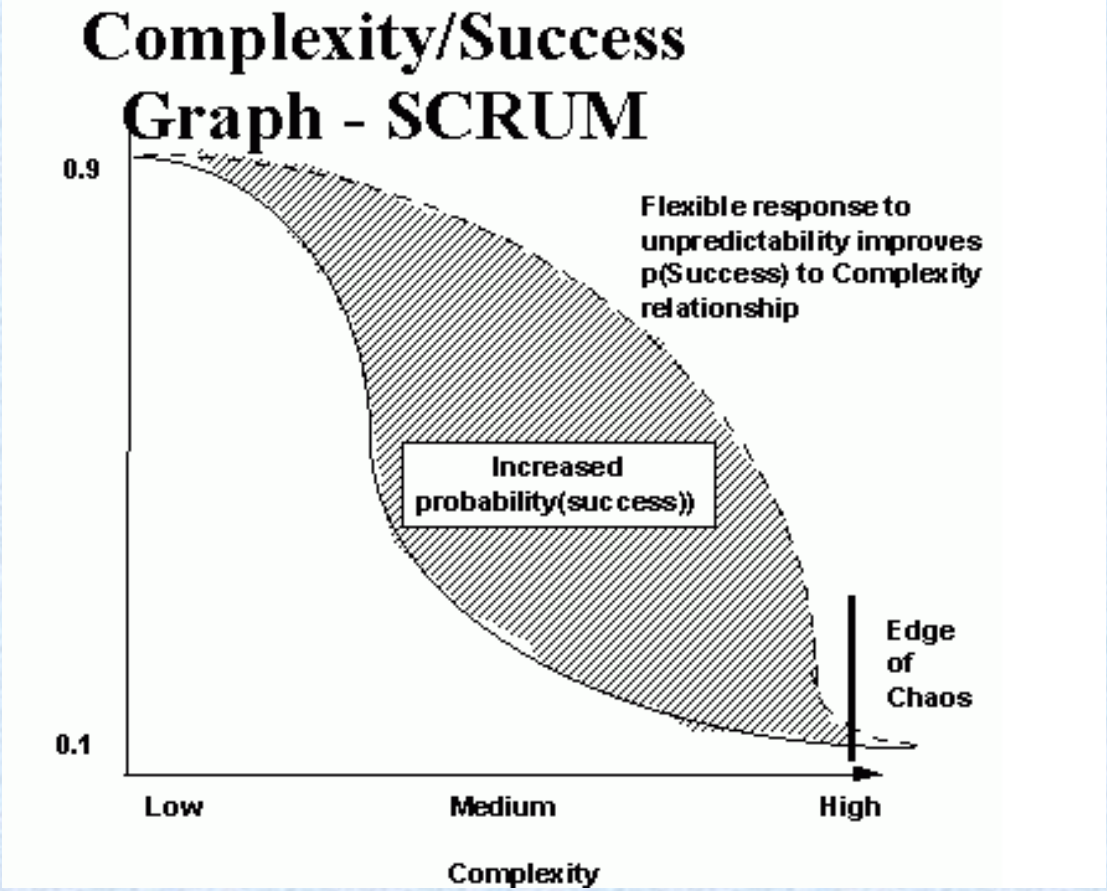
客户合作胜过合同谈判

- 指明了需求、进度以及项目成本的合同存在根本上的缺陷。
- 成功的项目需要频繁有序的客户反馈。为开发团队和客户的协同工作方式提供指导的合同才是最好的合同。

响应变化胜过遵循计划

- 计划赶不上变化。响应变化的能力决定一个项目的成败。
- 较好的做计划策略是：为下两周做详细的计划，为下三个月做粗略的计划，再以后就做极为粗糙的计划。

与传统模型的对比


Complexity/Success Graph - SCRUM
0.9
0.1
Flexible response to unpredictability improves p(Success) to Complexity relationship
Increased probability(success))
Edge of Chaos
Low
Medium
High
Complexity

2、敏捷实践原则

- 我们最优先要做的是通过尽早的、持续的交付有价值的软件来使客户满意。

可以工作的软件胜过面面俱到的文档

- 即使到了开发的后期，也欢迎改变需求。敏捷过程利用变化来为客户创造竞争优势。

响应变化胜过遵循计划

- 经常性的交付可以工作的软件，交付的间隔可以从几个星期到几个月，交付的时间间隔越短越好。

可以工作的软件胜过面面俱到的文档

- 在整个项目开发期间，业务人员和开发人员必须天天都在一起工作。

个体和交互胜过过程和工具

个体和交互胜过过程和工具

- 围绕被激励起来的个体来构建项目。给他们提供所需的环境和支持，并且信任他们能够完成工作。
- 在团队内部，最具有效果并且富有效率的传递信息的方法，就是面对面的交谈。

个体和交互胜过过程和工具

可以工作的软件胜过面面俱到的文档

- 工作的软件是首要的进度度量标准。

不鼓励加班

- 敏捷过程提倡可持续的开发进度。责任人、开发者和用户应保持一个长期恒定的开发速度。

- 不断关注优秀的技能和好的设计会增强敏捷能力。
- 简单—使未完成的工作最大化的艺术—是根本的。

不对未来做猜测，只对当前负责

- 最好的构架、需求和设计出自于自组织的团队。

自组织的团队不担心人员流动

- 每隔一定时间，团队会在如何才能更有效的工作方面进行反省，然后相应地对自己的行为进行调整。

结论

- 每一位软件开发人员、开发团队的职业目标，都是给雇主和客户交付最大可能的价值。
- 虽然在项目中采用过程方法是出于好意，但是膨胀的过程方法对于项目的失败至少应该负一些责任。

膨胀的过程方法会导致效率的降低

- 敏捷软件开发的价值观和原则构成了一个可以帮助团队打破过程膨胀循环的方法，它关注的是可以达到团队目标的一些简单的技术。

常见的敏捷过程有：

SCRUM、Crystal、ADP、 XP

SCRUM和XP的相同点：都是通过迭代进行开发
不同点：XP有结对编程，SCRUM有一个Master，有会议时间，趋向于结构化

二、极限编程实践

极限编程(eXtreme Programming)是敏捷方法中最著名的一个。它由一系列简单却互相依赖的实践组成。这些实践结合在一起形成了一个敏捷开发过程。

1、极限编程实践

●完整团队

XP项目的所有参与者(开发人员、业务分析师、测试人员等等)一起工作在一个开放的场所中，他们是同一个团队的成员。这个场所的墙壁上随意悬挂着大幅的、显著的图表以及其他一些显示他们进度的东西。

●计划游戏

XP习惯把一次迭代发布的内容称之为“planning game”，在迭代之处确认阶段称为“Iteration Planning Game”迭代计划游戏，而确认可发布内容范围称为“Release Planning Game”发布计划游戏；

计划是持续的、循序渐进的。每2周，开发人员就为下2周估算候选特性的成本，而客户则根据成本和商务价值来选择要实现的特性。

●客户测试

作为选择每个所期望的特性的一部分，客户定义出自动验收测试来表明该特性可以工作。首先XP希望对于所开发的代码都要有单元测试、通过持续的集成和测试来保证代码的质量。XP的测试一般特别指功能上的自动测试，和客户的验收测试；

●简单设计

团队保持设计恰好和**当前的系统功能**相匹配。它通过了所有的测试，不包含任何重复，表达出了编写者想表达的所有东西，并且包含尽可能少的代码。

由于XP方法的特征，无法全面把握用户的最终需求，所以不对不存在的功能做猜测，在代码层面简洁、尽量的小粒度的类或方法；

- 结对编程

  所有的产品软件都是由两个程序员、并排坐在一起在同一台机器上构建的。开发两人可以互为观察员和开发者，这样写出的代码具有较少的缺陷，相互学习使得代码具有更好的可读性

- 测试驱使开发

  程序员以非常短的循环周期工作，他们先增加一个失败的测试，然后使之通过。测试用例循序渐进的对代码的编写进行指导。

- **改进设计**

  随时改进糟糕的代码。保持代码尽可能的干净、具有表达力。

- **持续集成**

  团队总是使系统完整的被集成。单独服务器对所有代码进行不间断的功能构建

- **集体代码所有权**

  任何结对的程序员都可以在任何时候改进任何代码

●系统隐喻

人员的交流采用系统隐喻

为了便于设计沟通，XP开发设计人员和客户用约定的方式来描述特定的系统功能。

●可持续的速度

团队只有持久才有获胜的希望。他们以能够长期维持的速度努力工作。他们保存精力，他们把项目看做是马拉松长袍，而不是全速短跑。

所有的预定工作都应在工作时间内完成，个人时间和工作时间应该有清晰的分界，不能相互影响，提倡“没有加班”；

●编码标准

系统中所有的代码看起来就好像是被单独一个非常值得信任的人编写的。

2、计划游戏

当你能够度量你所说的，并且能够用数字去表达它时，就表示你了解了它；若你不能度量它，不能用数字去表达它，那么说明你的知识就是匮乏的、不能令人满意的。

－ 凯尔文勋爵(英国物理学家)

计划游戏

- 初始探索
- 发布计划
- 迭代计划
- 任务计划
- 迭代
- 结论

初始探索

- 在项目开始时，开发人员和客户会尽量确定出所有真正重要的用户素材。然而，他们不会试图去确定所有的用户素材。开发人员对素材进行估算。
- 分解、速度和探究。

发布计划

客户选择素材

- 客户根据开发速度确定每个素材的成本，再根据每个素材的商业价值和优先级别做出最优选择。让客户来选定那些会给他们带来最大利益的素材，属于商务决策范畴。
- 开发人员和客户对项目的首次发布时间达成一致后客户挑选发布中他们想实现的素材，大致确定素材的顺序。

迭代计划

- 开发人员和客户决定迭代规模。客户决定首次迭代中要实现的素材。
- 迭代期间用户素材的实现顺序属于技术决策范畴，开发人员采用最具技术意义的顺序来实现这些素材。
- 迭代开始后客户不能再改变该迭代期内需要实现的素材。
- 迭代要在指定的日期内结束，合计所有已经完成的素材估算值，计算本次迭代的开发速度，用于计划下一次迭代。

任务计划

- 开发人员在客户帮助下尽可能把素材分解成开发任务。
- 开发人员逐个签订他们想要实现的任务，并以任务点数对那项任务进行估算。
- 迭代的中点：在迭代进行到一半时，应该完成本次迭代中所安排的半数素材。

迭代

每两周，本次迭代结束，下次迭代开始。每次迭代结束时会给客户演示当前可以运行的程序，要求客户对程序的外观、感觉和性能进行评价。客户会以新的用户素材的方式提供反馈。

结论

- 通过一次次迭代和发布，项目进入了一种可以预测的、舒适的开发节奏。
- 开发人员看到的是他们自己估算、自己度量的开发速度控制的合理计划。选择舒适的任务并保持高的工作质量。
- 管理人员从每次迭代中获取数据，使用这些数据来控制和管理项目。
- 使用敏捷方法并不意味着客户一定可以得到他们想要的。它只不过意味着他们能够控制着团队以最小的代价获得最大的商业价值。

3、测试驱使开发

- 测试驱使的开发方法
- 验收测试
- 结论

测试驱使的开发方法

- 程序中的每一项功能都有测试来验证它的操作的正确性。
- 迫使我们使用不同的观察点，可以设计出便于调用的软件。

开发者、测试者不同的观察点

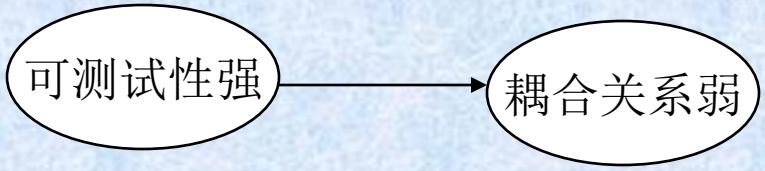
- 迫使我们解除软件中的耦合。
- 测试可以作为一种无价的文档形式，这份文档是可编译、可运行的。它保持最新，不会撒谎。

验收测试

- 验收测试是关于一项特性的最终文档。程序员可以阅读验收测试来理解这项特性。
- 验收测试用来验证系统是否满足客户需求。
- 验收测试可促使你在大的方面做出优良的系统构架决策。

结论

- 系统绝对不允许倒退。
- 单元测试和验收测试是一种准确和可靠的文档形式。
- 测试最重要的好处就是它对于构架和设计的影响。全面地考虑验收测试和单元测试的行为对于软件的结构具有深远的正面影响。



4、重构

用餐后清理厨房值得我们花时间吗？

- 每一个软件模块都具有三项职责。第一个是它运行起来所完成的功能。第二个是它要应对变化。第三是要和阅读它的人进行沟通。

重构主要应对后两个职责

- 重构是在不改变代码外在行为的前提下对代码做出修改，以改进代码的内部结构的过程。

三、Scrum

Scrum 软件开发模型是敏捷开发的一种。Scrum的基本假设是：开发软件就像开发新产品，无法一开始就能定义软件产品最终的规程，过程中需要研发、创意、尝试错误，所以没有一种固定的流程可以保证专案成功。

Scrum 开发流程通常以 30 天(或更短时间)为一个阶段，由客户提供新产品的需求规格开始，开发团队与客户于每一个阶段开始时挑选该完成的规格部分，开发团队必须尽力于 30 天后交付成果，团队每天用 15 分钟开会检查每个成员的进度与计划，了解所遭遇的困难并设法排除。

1、Scrum一些基本概念

- backlog：可以预知的所有任务， 包括功能性的和非功能性的所有任务。
- sprint:一次迭代开发的时间周期，一般最多以30天为一个周期.在这段时间内，开发团队需要完成一个制定的backlog,并且最终成果是一个增量的，可以交付的产品。
- sprint backlog:一个sprint周期内所需要完成的任务。
- scrumMaster：负责监督整个Scrum进程，修订计划的一个团队成员。

- time-box：一个用于开会时间段。比如每个daily scrum meeting的time-box为15分钟。
- sprint planning meeting：在启动每个sprint前召开。一般为一天时间（8小时）。该会议需要制定的任务是：产品Owner和团队成员将backlog分解成小的功能模块， 决定在即将进行的sprint里需要完成多少小功能模块，确定好这个Product Backlog的任务优先级。另外，该会议还需详细地讨论如何能够按照需求完成这些小功能模块。制定的这些模块的工作量以小时计算。

- Daily Scrum meeting：开发团队成员召开，一般为15分钟。每个开发成员需要向ScrumMaster汇报三个项目：今天完成了什么？ 是否遇到了障碍？即将要做什么？通过该会议，团队成员可以相互了解项目进度。
- Sprint review meeting：在每个Sprint结束后，这个Team将这个Sprint的工作成果演示给Product Owner和其他相关的人员。一般该会议为 4 小时。
- Sprint retrospective meeting：对刚结束的Sprint进行总结。会议的参与人员为团队开发的内部人员。一般该会议为 3 小时。

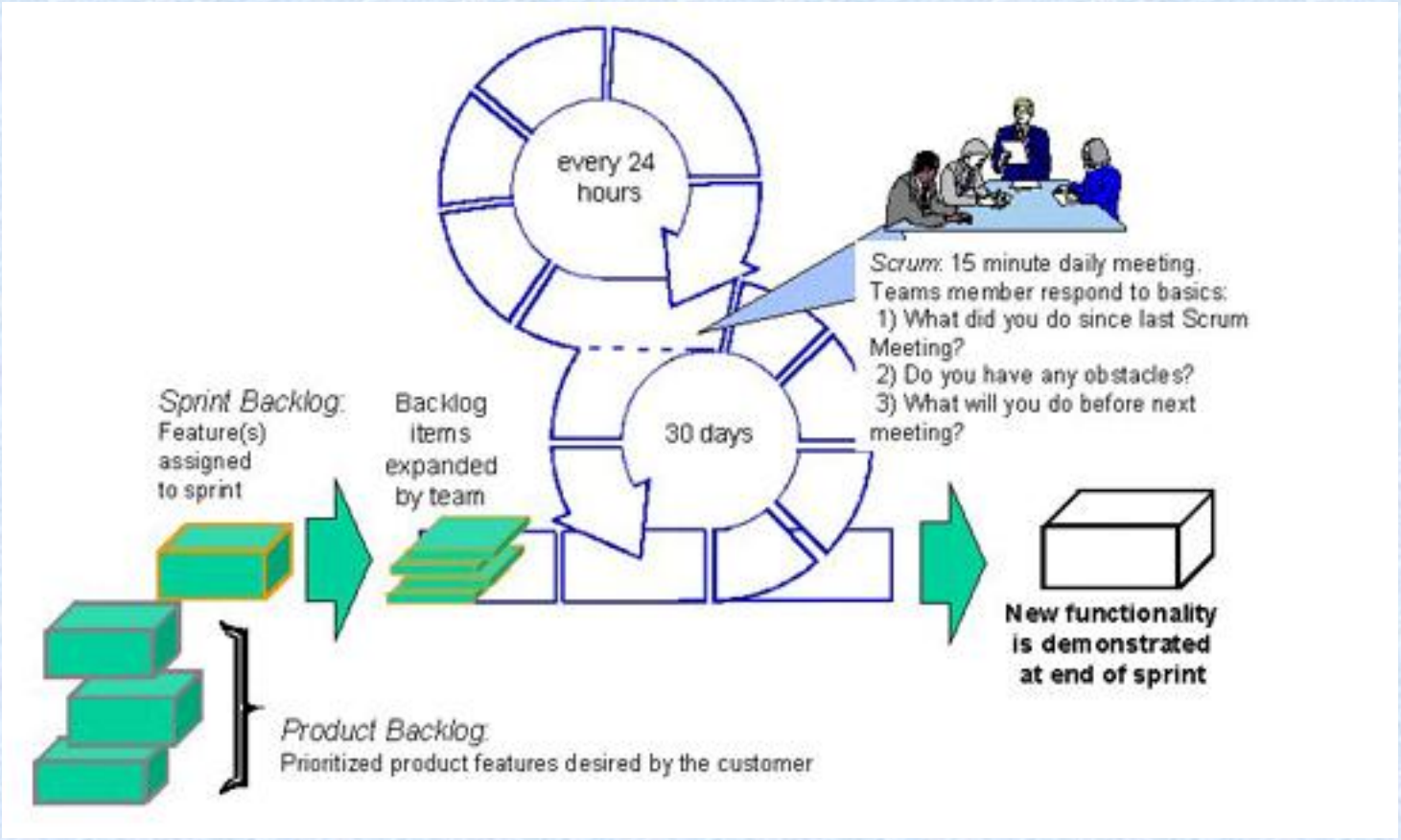
2、Scrum过程

- 将整个产品的backlog分解成Sprint Backlog,这个Sprint Backlog是按照目前人力物力条件可以完成的。

- 召开sprint planning meeting，划分，确定这个Sprint内需要完成的任务，标注任务的优先级并分配给每个成员。注意这里的任务是以小时计算的，并不是按人天计算。

- 进入sprint开发周期，在这个周期内，每天需要召开Daily Scrum meeting。

- 整个sprint周期结束，召开Sprint review meeting，将成果演示给Product Owner.
- 团队成员最后召开Sprint retrospective meeting，总结问题和经验。
- 这样周而复始，按照同样的步骤进行下一次Sprint.


every 24 hours
30 days
Scrum: 15 minute daily meeting.
Teams member respond to basics:
1) What did you do since last Scrum Meeting?
2) Do you have any obstacles?
3) What will you do before next meeting?
Sprint Backlog:
Feature(s) assigned to sprint
Backlog items expanded by team
New functionality is demonstrated at end of sprint
Product Backlog:
Prioritized product features desired by the customer

Bob Martin曾经碰到过一个人。此人声称他的组织正在使用XP。Martin问他如何看待结对编程，此人答道：“我们不结对编程。”Martin又问他重构进行得怎么样，此人答道：“我们不重构。” Martin又问他计划游戏的效果如何，那人答道：“我们没有计划游戏。” 嗯，”Martin问道，“那你们做了什么呢？”得到的回答是：“我们只是不编写任何文档！”