SOFTWARE TESTING

Software Testing

Agile Modeling & SLCP (1)

School of Computer Science & Engineering
Sun Yat-sen University

Instructor: Guoyang Cai
email: isscgy@mail.sysu.edu.cn

Approaches & Technologies

中山大學
SUN YAT-SEN UNIVERSITY

OUTLINE

- 1.1 软件与软件危机
- 1.2 软件开发与软件工程
- 1.3 软件生命周期模型
- 1.4 软件质量标准
- 1.5 敏捷开发
  - 敏捷开发概述
  - Scrum
  - eXtreme Programming
  - TDD
  - 规模化敏捷开发
- 1.6 软件生命周期过程

敏捷开发概述

敏捷开发概述

敏捷开发的起源

- 敏捷建模 (Agile Modeling, AM) 源于 *Scott W. Ambler* 的 Extreme Modeling (XM, 2000)。2001年以 *Kent Beck*, *Alistair Cockburn*, *Ward Cunningham*, *Martin Fowler* 等人为首在 Snowbird, Utah 发布《敏捷宣言》，决定将 Agile 作为新的轻量级软件开发过程的家族名称。
- 敏捷建模是一种态度，而不是一个说明性过程。它是从软件开发过程实践中归纳总结出来的一些价值观、原则和实践。
- 敏捷建模不是一个完整的方法论，而是对已有生命周期模型的补充，在应用传统的生命周期模型时可以借鉴敏捷建模的过程指导思想。
- http://agilemanifesto.org/

敏捷开发概述

- Manifesto for Agile Software Development (敏捷宣言)

敏捷开发概述

- Manifesto for Agile Software Development
  - We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value
    - **Individuals and Interactions** *over processes and tools*
    - **Working Software** *over comprehensive documentation*
    - **Customer Collaboration** *over contract negotiation*
    - **Responding to Change** *over following a plan*
  - That is, while there is value in the items on the right, they value the items on the left more.

敏捷开发概述

- Manifesto for Agile Software Development
  - 我们一直在实践中探寻更好的软件开发方法，身体力行的同时也帮助他人。由此我们建立了如下价值观：
    - **个体和互动** 高于 流程和工具
    - **可工作的软件** 高于 详尽的文档
    - **客户合作** 高于 合同谈判
    - **响应变化** 高于 遵循计划
  - 也就是说，尽管右项有其价值，我们更重视左项的价值。

敏捷开发概述

- Manifesto for Agile Software Development
  - As *Scott Ambler* elucidated:
    - Tools and processes are important, but it is more important to have competent people working together effectively.
    - Good documentation is useful in helping people to understand how the software is built and how to use it, but the main point of development is to create software, not documentation.
    - A contract is important but is no substitute for working closely with customers to discover what they need.
    - A project plan is important, but it must not be too rigid to accommodate (过于死板而无法适应) changes in technology or the environment, stakeholders' priorities, and people's understanding of the problem and its solution.

敏捷开发概述

- Manifesto for Agile Software Development
  - 12 agile principles for Agile Development

(1) Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

(2) Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

(3) Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

(4) Business people and developers must work together daily throughout the project.

(5) Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

(6) The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

敏捷开发概述

敏捷开发概述

Manifesto for Agile Software Development

12 agile principles for Agile Development (cont.)

(7) Working software is the primary measure of progress.

(8) Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

(9) Continuous attention to technical excellence and good design enhances agility.

(10) Simplicity—the art of maximizing the amount of work not done—is essential.

- Don't build tomorrow's software, but focus on the easiest way that need to be solved now.

(11) The best architectures, requirements, and designs emerge from self-organizing teams.

(12) At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

敏捷开发概述

- 敏捷宣言遵循的12条原则：
  - (1) 我们最重要的目标，是通过尽早和持续地交付有价值的软件来使客户满意。
  - (2) 欣然面对需求变化—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) 经常地交付可工作的软件，相隔几个星期或几个月不等，倾向于较短的时间周期。
  - (4) 在整个项目过程中，业务人员与开发人员必须日常一起工作。
  - (5) 激励项目人员，以他们为核心构建项目，为他们提供需要的环境和支持，并相信他们能够完成任务。
  - (6) 无论团队内还是团队间，最有效的沟通方法是面对面的交谈。

敏捷开发概述

- 敏捷宣言遵循的12条原则：(续)
  - (7) 可工作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。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) 对技术的精益求精以及对设计的不断完善将提升敏捷性。
  - (10) 以简洁为本。简洁是尽可能减少不必要的工作量的艺术。
  - (11) 最佳的架构、需求和设计出自于自组织团队。
  - (12) 团队要定期反省如何才能提高成效，并以此调整团队的行为。

敏捷开发概述

敏捷开发概述

敏捷开发的目标

- 敏捷开发的总体目标是通过“尽可能早地、持续地交付有价值的软件”，使客户满意。
- 敏捷开发强调软件开发应当能够对未来可能出现的变化和不确定性作出全面反应。
- 敏捷开发主要用于在需求模糊或快速变化的前提下，支持小型开发团队的软件开发活动。

敏捷开发概述

- 敏捷开发概述
  - 敏捷开发的管理原则
    - 敏捷开发是一种以人为核心、迭代、循序渐进的开发过程指导思想。
    - 在敏捷开发过程中，软件项目的构建被切分成多个子项目分别实现。各个子项目之间相互联系、独立运行，子项目的成果经过测试，具备集成和可运行的特征。

敏捷开发概述

- 敏捷开发概述
  - 敏捷开发的核心实践
    - 项目关键利益方 (Project Stakeholder) 的积极参与
    - 正确使用工件
    - 集体所有制 (对代码、工件、模型的共有，包括使用和修改)
    - 测试性思维 (比如“测试优先”)
    - 并行创建模型 (为一个问题同时建立多种模型)
    - 创建简单的内容
    - 简单地建模
    - 公开展示模型 (使用 modeling wall 向项目参与各方展示)
    - 切换到另外的工件 (遇到困难时的敏捷切换)
    - 小增量建模
    - 和他人一起建模
    - 用代码验证模型
    - 使用最简单的建模工具

敏捷开发概述

- 敏捷开发概述
  - 敏捷开发的补充实践
    - 使用建模标准 (比如 UML)
    - 逐渐应用模式 (pattern)
    - 丢弃临时模型
    - 合同模型要正式
    - 为外部交流建模
    - 为帮助理解建模
    - 重用现有的资源
    - 不到万不得已不更新模型

敏捷开发方法分类

- XP
  - XP (极限编程) 提倡测试先行，目的在于将后面出现缺陷的概率降至最低。
- Scrum
  - Scrum 是一种迭代的增量化过程，用于产品开发或工作管理。
- Crystal Methods
  - Crystal Methods (水晶方法系列) 与 XP 一样，都是以人为中心，但不同类型的项目需要不同的实践方法。
- FDD
  - FDD (特性驱动开发) 是一套针对中小型软件开发项目的开发模式，采用模型驱动的快速迭代开发过程。

敏捷开发方法分类

- ASD
  - ASD (Adaptive Software Development, 自适应软件开发) 从复杂自适应系统理论派生出来，用于对需求多变、开发周期短项目的管理。
- DSDM
  - DSDM (动态系统开发方法) 倡导以业务为核心，快速而有效地进行系统开发。
- RUP
  - RUP 是一个过程框架，它可以包容许多不同类型的过程，但核心还是面向对象过程。

Scrum

- Scrum is an *agile framework* for managing knowledge work, with an emphasis on software development.
  - designed for teams of *3 to 9 members*
  - their work can be break into actions that can be completed within time-boxed (有时间限制的) iterations, called *sprints*, no longer than one month and most commonly two weeks
  - then track progress and re-plan in 15-minute time-boxed stand-up meetings, called *daily scrums*

Scrum

- Scrum is a *lightweight*, *iterative* and *incremental* framework for managing product development.
  - defines “a flexible, holistic (整体的) product development strategy where a development team *works as a unit* to reach a common goal“.
  - challenges assumptions of the “traditional, sequential approach” (*Hirotaka Takeuchi* and *Ikujiro Nonaka* 竹内弘高和野中郁次郎, 1986) to product development.
  - enables teams to *self-organize* by
    - encouraging physical co-location or close online collaboration (紧密的在线合作) of all team members.
    - daily face-to-face communication among all team members and disciplines involved.

Scrum

Scrum

History of Scrum

- *Hirotaka Takeuchi* and *Ikujiro Nonaka* described a new approach to commercial product development that would increase speed and flexibility, based on case studies from manufacturing firms in the automotive, photocopier and printer industries.
  - They called this the *holistic or rugby approach* (整体方法，或橄榄球方法), as the whole process is performed by one *cross-functional team* across multiple overlapping phases, where the team "tries to go the distance as a unit, passing the ball back and forth".
- In rugby football, a scrum refers to the manner of restarting the game after a minor infraction. In the early 1990s, *Ken Schwaber* used what would become Scrum at his company, Advanced Development Methods, and *Jeff Sutherland*, with *John Scumniotales* and *Jeff McKenna*, developed a similar approach at Easel Corporation, and were the first to refer to it using the single word *Scrum*.

Scrum

- History of Scrum
  - In 1995, *Sutherland* and *Schwaber* jointly presented a paper describing the *Scrum methodology* at the Business Object Design and Implementation Workshop held as part of Object-Oriented Programming, Systems, Languages & Applications '95 (OOPSLA '95) in Austin, Texas, its first public presentation.
  - *Schwaber* and *Sutherland* collaborated during the following years to merge the above writings, their experiences, and industry best practices into what is now known as Scrum.
  - In 2001, *Schwaber* worked with *Mike Beedle* to describe the method in the book *Agile Software Development with Scrum*.
  - Its approach to planning and managing projects is to bring decision-making authority to the level of operation properties and certainties.

Scrum

- History of Scrum
  - Although the word is not an acronym (缩略词), some companies implementing the process have been known to spell it with capital letters as SCRUM. This may be due to one of *Ken Schwaber*'s early papers, which capitalized SCRUM in the title.
  - While the trademark on the term *Scrum* itself has been allowed to lapse, so that it is deemed as owned by the wider community rather than an individual, the leading capital is retained—except when used with other words (as in *daily scrum* or *scrum team*).
  - Hybridization of scrum is common as scrum does not cover the whole product development lifecycle; therefore, organizations find the need to add in additional processes to create a more comprehensive implementation.
    - For example, at the start of the project, organizations commonly add process guidance on requirements gathering and prioritization, initial high-level design, and budget and schedule forecasting.

Scrum

- Key Ideas
  - A key principle of Scrum is the dual recognition.
    - Customers will change their minds about what they want or need (often called *requirements volatility* 需求波动).
    - There will be unpredictable challenges—for which a predictive or planned approach is not suited.
  - Scrum adopts an evidence-based empirical approach (Scrum 接纳一种基于证据的经验主义的方法)—accepting that the problem cannot be fully understood or defined up front, and instead focusing on how to maximize the team's ability to deliver quickly, to respond to emerging requirements, and to adapt to evolving technologies and changes in market conditions.

Scrum

- Roles
  - There are three core roles:
    - Product Owner: 产品负责人 / 项目经理
    - Scrum Master: Scrum 主管 / 敏捷专家
    - Development Team: 开发团队

    and a range of ancillary roles (辅助角色).
  - Core roles are often referred to as *pigs* and ancillary roles as *chickens* after the story *The Chicken and the Pig*.
    - The chicken asks the pig if he is interested in jointly opening a restaurant. The chicken says they could call it, "Ham-and-Eggs." The pig answers, "*No thanks. I'd be committed, but you'd only be involved!*"
  - It specifies a concrete set of roles, which are divided into two groups:
    - Committed
    - Involved.

- Scrum
  - Roles
    - Committed
      - those directly responsible for production and delivery of the final product. These roles include the team as a whole, its members, the scrum master, and the product owner.
      - The core roles are those committed to the project in the Scrum process—they are the ones producing the product (objective of the project). They represent the *scrum team*.
    - Involved
      - represents the other people interested in the project, but who aren't taking an active or direct part in the production and delivery processes. These roles are typically stakeholders and managers.

- Scrum
  - The Product Owner
    - The product owner represents the stakeholders and is *the voice of the customer*. He or she is accountable for ensuring that the team delivers value to the business.
    - The product owner writes (or has the team write) customer-centric items (typically user stories), ranks and prioritizes them, and adds them to the product backlog (产品积压工作项).
    - Scrum teams should have one product owner, and while they may also be a member of the development team, this role should not be combined with that of the *scrum master*. In an enterprise environment, though, the product owner is often combined with the role of *project manager* as they have the best visibility regarding the scope of work (products).

- Scrum
  - The Development Team
    - The development team is responsible for delivering potentially shippable product increments at the end of each *Sprint* (the Sprint Goal).
    - A development team is made up of 7 +/- 2 individuals with cross-functional skills who do the actual work:
      - analysis, design, develop, test, technical communication, document, etc.
    - The development team in Scrum is self-organizing, even though there may be some level of interface with project management offices (PMOs).

Scrum

- The Scrum Master
  - Scrum is facilitated by a scrum master (Scrum 项目由一位敏捷专家推进), who is accountable for *removing impediments* (消除障碍) to the ability of the team to deliver the sprint goal/deliverables.
    - The scrum master is not the team leader, but acts as a *buffer* between the team and any distracting (分心的) influences.
    - The scrum master ensures that the *Scrum process* is used as intended. The scrum master is the enforcer of the *rules of Scrum*, often chairs key meetings, and challenges the team to improve.
  - The role has also been referred to as a *servant-leader* (仆人式领导) to reinforce these dual perspectives.
  - The scrum master differs from a project manager in that the latter may have people management responsibilities unrelated to the role of scrum master. The scrum master role excludes any such additional people responsibilities.

Scrum

- Workflows in the Scrum framework
  - (1) A *Sprint* (or iteration) is the basic unit of development in Scrum. The sprint is a time-boxed effort; that is, it is restricted to a specific duration. The duration is fixed in advance for each sprint and is normally between one week and one month, with *two weeks* being the most common.
    - Each sprint starts with a *sprint planning event* that aims to define a sprint backlog, identify the work for the sprint, and make an estimated forecast for the sprint goal.
    - Each sprint ends with a *sprint review* and *sprint retrospective*, that reviews progress to show to stakeholders and identify lessons (教训) and improvements for the next sprints.
    - Scrum emphasizes *working product* at the end of the sprint that is really done. In the case of software, this likely includes that the software has been fully integrated, tested and documented, and is potentially releasable.

Scrum

- Workflows in the Scrum framework

  (2) At the beginning of a sprint, the scrum team holds a *Sprint Planning* event to:

  - Mutually discuss and agree on the *scope of work* that is intended to be done during that sprint.
  - Select *product backlog* items that can be completed in one sprint.
  - Prepare a *sprint backlog* that includes the work needed to complete the selected product backlog items.
  - Once the development team has prepared their sprint backlog, they forecast (usually by voting) which *tasks* will be delivered within the sprint.

Scrum

- Workflows in the Scrum framework
  - (2) At the beginning of a sprint, the scrum team holds a *Sprint Planning* event to:
    - The recommended duration is *four hours* for a two-week sprint, pro-rata (按比例) for other sprint durations.
      - During the first half, the whole scrum team (development team, scrum master, and product owner) selects the *product backlog* items they believe could be completed in that sprint.
      - During the second half, the development team identifies the *detailed work* (tasks) required to complete those product backlog items; resulting in a confirmed *sprint backlog*.
      - As the detailed work is elaborated, some product backlog items may be split or put back into the product backlog if the team no longer believes they can complete the required work in a single sprint.

Scrum

Scrum

- Workflows in the Scrum framework

  (3) Each day during a sprint, the team holds a *Daily Scrum* (or stand-up) with specific guidelines:

  - All members of the development team come *prepared*. The daily scrum:
    - starts precisely *on time* even if some development team members are missing
    - should happen at the *same time and place* every day
    - is limited (time-boxed) to *fifteen minutes*
  - Anyone is welcome, though only development team members should contribute.

Scrum

- Workflows in the Scrum framework
  - (3) Each day during a sprint, the team holds a *Daily Scrum* (or stand-up) with specific guidelines:
    - During the daily scrum, each team member typically answers three questions:
      - What did I complete yesterday that contributed to the team meeting our sprint goal?
      - What do I plan to complete today to contribute to the team meeting our sprint goal?
      - Do I see any impediment that could prevent me or the team from meeting our sprint goal?
    - Any impediment identified in the daily scrum should be captured by the *scrum master* and displayed on the team's scrum board or on a shared risk board, with an agreed person designated to working toward a resolution (outside of the daily scrum). *No detailed discussions* should happen during the daily scrum.

Scrum

Scrum

- Workflows in the Scrum framework
  - (4) At the end of a sprint, the team holds two events: the *Sprint Review* (评审) and the *Sprint Retrospective* (回顾).
    - At the sprint review, the team:
      - reviews the work that was completed and the planned work that was not completed.
      - presents the completed work (aka the demo) to the *stakeholders*.
      - collaborates with the stakeholders on what to work on next.
    - Guidelines for sprint reviews:
      - Incomplete work cannot be demonstrated.
      - The recommended duration is *two hours* for a two-week sprint (proportional for other sprint-durations).

Scrum

Scrum

Workflows in the Scrum framework

(4) At the end of a sprint, the team holds two events: the *Sprint Review* (Sprint 审查) and the *Sprint Retrospective* (Sprint 回顾).

- At the sprint retrospective, the team:
  - Reflects on (反省) the past sprint.
  - Identifies and agrees on continuous process improvement actions.
- Three main questions are asked in the sprint retrospective:
  - What *went well* during the sprint?
  - What *did not go* well?
  - What could *be improved* for better productivity in the next sprint?
- The recommended duration is *one-and-a-half hours* for a two-week sprint (proportional for other sprint duration(s)).
- This event is facilitated by the *scrum master*.

Scrum

- Artifacts in the Scrum framework
  - Product Backlog
    - The product backlog is *a model of work to be done* and contains an ordered list of product requirements that a scrum team maintains for a product.
    - The *format* of product backlog items varies, common formats include user stories, use cases, or any other requirements format the team finds useful. These will define features, bug fixes, non-functional requirements, etc.
    - The *product owner* prioritizes product backlog items (PBIs) based on considerations such as risk, business value, dependencies, size, and date needed.
    - The product backlog is what will be delivered, ordered into the sequence in which it should be delivered. It is visible to everyone but may *only be changed with the consent (同意) of the product owner*, who is ultimately responsible for ordering product backlog items for the development team to choose.

Scrum

- Artifacts in the Scrum framework
  - Sprint Backlog
    - The sprint backlog is *the list of work* the development team must address during the next sprint.
    - The list is *derived by the scrum team* progressively selecting product backlog items in priority order from the top of the product backlog until they feel they have enough work to fill the sprint. The development team should keep in mind its past performance assessing its capacity for the new-sprint, and use this as a guideline of how much 'effort' they can complete.
    - The product backlog items may be broken down into *tasks* by the development team. Tasks on the sprint backlog are never assigned (or pushed) to team members by someone else; rather team members sign up for (or pull) tasks as needed according to the backlog priority and their own skills and capacity. This promotes self-organization of the development team and developer buy-in.

- Scrum
  - Artifacts in the Scrum framework
    - Product Increment
      - The potentially releasable increment is the sum of all the product backlog items completed during a sprint, integrated with the work of all previous sprints.
      - At the end of a sprint, the increment must be complete, according to the scrum team's *definition of "done"*, fully functioning, and in a usable condition regardless of whether the product owner decides to actually release it.

- Scrum
  - Artifacts in the Scrum framework
    - Some Extensions
      - Sprint burn-down chart (燃尽图)
      - Release burn-up chart (燃起图)
      - Definition of done (DoD 完成标准)
      - Velocity (速度)
      - Spike (探针 / 长钉)
      - Research (研究)
      - Tracer bullet (曳光弹).

- Scrum
  - The following terminology is used in Scrum
    - Scrum Team
      - Product Owner, Scrum Master and Development Team.
    - Product Owner
      - The person responsible for maintaining the Product Backlog by representing the interests of the stakeholders, and ensuring the value of the work the Development Team does.
    - Scrum Master
      - The person responsible for the Scrum process, making sure it is used correctly and maximizing its benefits.
    - Development Team
      - A cross-functional group of people responsible for delivering potentially shippable increments of Product at the end of every Sprint.

Scrum

- The following terminology is used in Scrum
  - Product Backlog (产品积压工作列表)
    - A prioritized list of high-level requirements.
  - Sprint
    - A time period (typically 1–4 weeks) in which development occurs on a set of backlog items that the team has committed to. Also commonly referred to as a Time-box or iteration.
  - Sprint Burn Down Chart
    - Daily progress for a Sprint over the sprint's length.
  - Release Burn Down Chart
    - Sprint level progress of completed stories in the Product Backlog.
  - Sprint Backlog (Sprint 积压工作列表)
    - A prioritized list of tasks to be completed during the sprint.

Scrum

- The following terminology is used in Scrum
  - (User) Story (用户描述)
    - A feature that is added to the backlog is commonly referred to as a story and has a specific suggested structure:

      ```
      As a <user type> I want to <do some action> so that
      <desired result>
      ```

    - This is done so that the development team can identify the user, action and required result in a request and is a simple way of writing easily understanding requests.
      - Example: As a wiki user I want a tools menu on the edit screen so that I can easily apply font formatting.
    - A story is an *independent, negotiable, valuable, estimable, small, testable* requirement ("INVEST").
    - Stories may be clustered into epics when represented on a product roadmap or further down in the backlog.

Scrum

- The following terminology is used in Scrum
  - Theme (主题)
    - A theme is a top-level objective that may span projects and products.
    - Themes may be broken down into sub-themes, which are more likely to be product-specific.
    - Themes can be used at both program and project level to drive strategic alignment (推动战略一致性) and communicate a clear direction .
  - Epic (史诗)
    - An epic is a group of related stories, mainly used in product roadmaps and the backlog for features that have not yet been analyzed enough to break down into component stories, which should be done before bringing it into a sprint so to reduce uncertainty.
    - Epics can also be used at both program and project level.

Scrum

- The following terminology is used in Scrum
  - Spike
    - A spike is a time boxed period used to research a concept and/or create a simple prototype.
    - Spikes can either be planned to take place in between sprints or, for larger teams, a spike might be accepted as one of many sprint delivery objectives.
    - Spikes are often introduced before the delivery of large epics or user stories in order to secure budget, expand knowledge, and/or produce a proof of concept.
    - The duration and objective(s) of a spike will be agreed between the Product Owner and Delivery Team before the start. Unlike sprint commitments, spikes may or may not deliver tangible (切实的), shippable, valuable functionality.
    - E.g., the objective of a spike might be to successfully reach a decision on a course of action. The spike is over when the time is up, not necessarily when the objective has been delivered.

Scrum

- The following terminology is used in Scrum
  - Tracer Bullet (曳光弹)
    - The tracer bullet is *a spike* with the current architecture, current technology set, current set of best practices which results in *production quality* code.
      - It might just be a very narrow implementation of the functionality but is *not throw away* code.
      - It is of production quality and the rest of the iterations can build on this code.
    - The name has military origins as ammunition that makes the path of the weapon visible, allowing for corrections. Often these implementations are a 'quick shot' through all layers of an application, such as connecting a single form's input field to the back-end, to prove the layers will connect as expected.
  - Impediment
    - Anything that prevents a team member from performing work as efficiently as possible.

Scrum

- The following terminology is used in Scrum
  - Point Scale/Effort/Story Points (难度尺度)
    - Relates to an abstract point system, used to discuss the difficulty of the story, without assigning actual hours. The most common scale used is a rounded Fibonacci sequence (1,2,3,5,8,13,20,40,100), although some teams use linear scale (1,2,3,4...), powers of two (1,2,4,8...), and clothes size (XS, S, M, L, XL).
  - Task
    - Added to the story at the beginning of a sprint and broken down into hours. Each task should not exceed 12 hours, but it's common for teams to insist that a task take no more than a day to finish.
  - Definition of Done (DoD)
    - The exit-criteria to determine whether a product backlog item is complete. In many cases the DoD requires that all regression tests should be successful.

Scrum

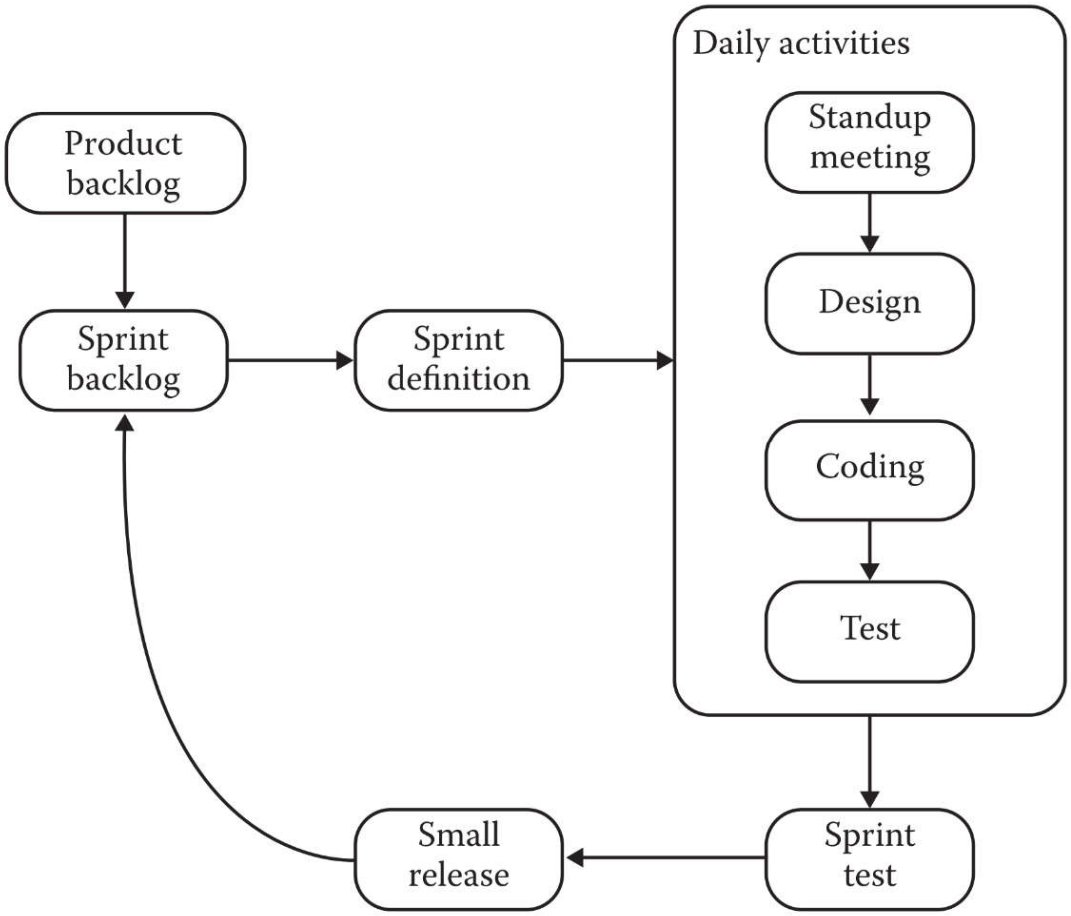
- The following terminology is used in Scrum
  - Velocity (团队速率)
    - The total effort a team is capable of in a sprint. The number is derived by evaluating the story points completed from the last few sprint's stories/features. This is a guideline for the team and assists them in understanding how many stories they can do in a future sprint.
  - Sashimi (成功的定义/生鱼片)
    - A report that something is "done". The definition of "done" may vary from one Scrum team to another, but must be consistent within one team.
  - Planning Poker (计划扑克游戏)
    - In the Sprint Planning Meeting, the team sits down to estimate its effort for the stories in the backlog. The Product Owner needs these estimates, so that he or she is empowered to effectively prioritize items in the backlog and, as a result, forecast releases based on the team's velocity.

Scrum

- The following terminology is used in Scrum
  - Abnormal Termination
    - The Product Owner can *cancel a Sprint* if necessary. The Product Owner may do so with input from the team, Scrum Master or management.
      - For instance, management may wish to cancel a sprint if external circumstances negate the value of the sprint goal.
    - If a sprint is abnormally terminated, the next step is to conduct a new Sprint planning meeting, where the reason for the termination is reviewed.
  - ScrumBut
    - A ScrumBut (or Scrum But) is an *exception* to the "pure" Scrum methodology, where a team has changed the methodology to adapt it to their own needs.

Scrum

- The Scrum life cycle.



Lecture 6. Agile Modeling & SLCP (1)

End of Lecture