

SOFTWARE REQUIREMENTS ENGINEERING

LECTURE # 2

AGILE METHODOLOGIES

Instructor Information

2

- ❑ Course Instructor: **Engr. Ali Javed**
Assistant Professor
Department of Software Engineering
U.E.T Taxila
 - ✓ Email: **ali.javed@uettaxila.edu.pk**
 - ✓ Website: **<http://web.uettaxila.edu.pk/uet/UETsub/perSites/mySite.asp?frmEmail=ali.javed@uettaxila.edu.pk>**
 - ✓ Contact No: **+92-51-9047747**
 - ✓ Office hours:
 - **Monday, 09:00 - 11:00, Office # 7 S.E.D**
- ❑ Lab Instructor's: **Engr. Asra, Engr. Sobia**

Course Information

3

- ❑ **Course Name: Software Requirements Engineering**
- ❑ **Course Code: SE-203**
- ❑ **CMS Link:** <http://web.uettaxila.edu.pk/CMS/SP2013/seSREbs/>

Presentation Outline

- **Agile Methodologies**
 - **Scrum**
 - **Extreme Programming**
 - **Adaptive Software Development**
 - **Feature Driven Development**
 - **Lean Development**

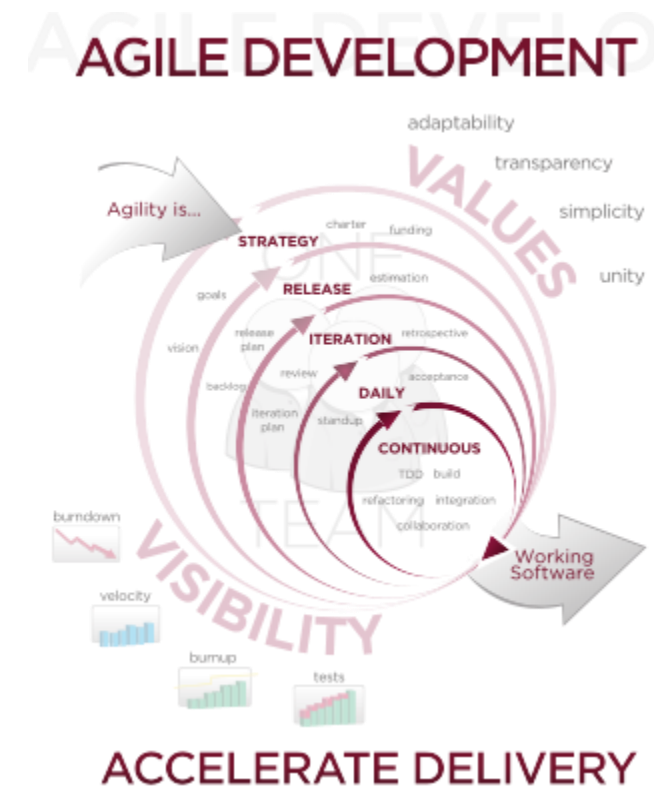
What is Agile?

5

- ❑ **Agile software development** is a group of software development methodologies based on iterative and incremental development, where requirements and solutions evolve through collaboration between self-organizing, cross-functional teams. It promotes adaptive planning, evolutionary development and delivery; time boxed iterative approach and encourages rapid and flexible response to change.

- ❑ **Common Practices**

- ✓ Regular Deployment of Working Software
- ✓ Pair Programming
- ✓ Refactoring [14]
- ✓ Continuous Integration
- ✓ Test Driven Development (TDD)
- ✓ Shared Code Ownership
- ✓ Active Stakeholder Participation



Some Agile Methodologies

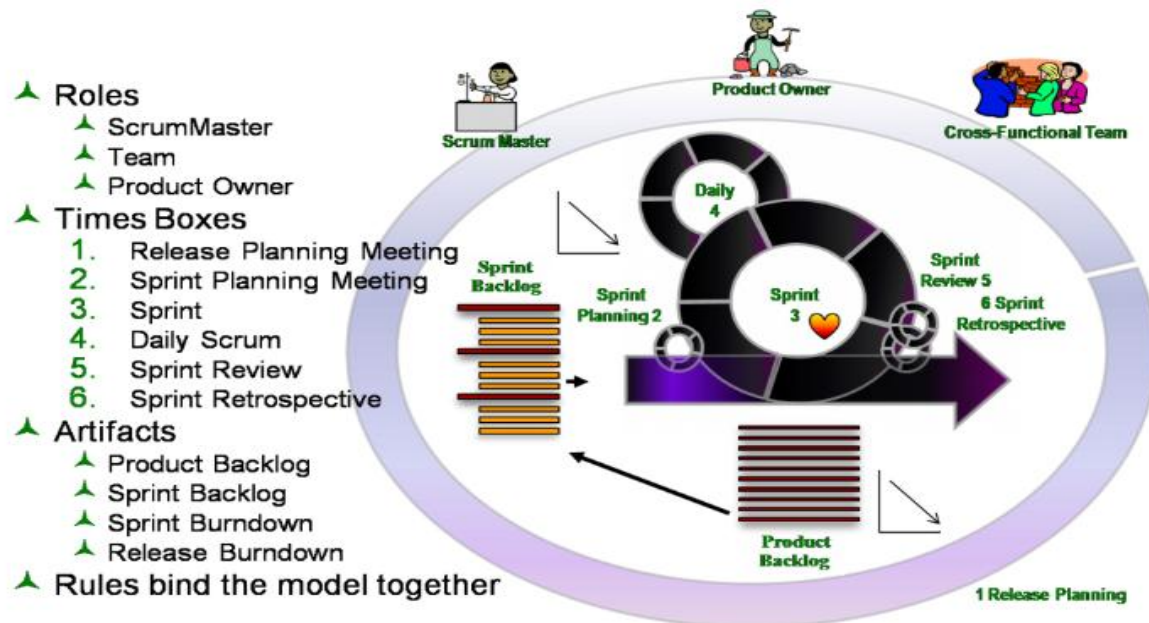
6

- ❑ **Scrum**
- ❑ **Extreme Programming (XP)**
- ❑ **Adaptive Software Process**
- ❑ **Feature-Driven Development (FDD)**
- ❑ **Lean Development**

SCRUM [10,11]

7

- ❑ **Scrum** is an iterative, incremental framework for project management often seen in agile software development
- ❑ It defines a set of activities that can help your team deliver more value to your customers faster.
- ❑ These activities provide your customers with the opportunity to review, guide and influence your team's work as it progresses.
- ❑ This approach does not attempt to define everything at the start of a project. Instead, your team works in short iterations (also called sprints) and refines the plan as the team makes progress.



Scrum Roles

- ❑ Scrum Master
- ❑ Team
- ❑ Product Owner

The SCRUM Master [3,5]

9

- ❑ In the Scrum process, Scrum Master has a role of coach , fixer and gatekeeper
- ❑ The job of the scrum master is to make sure that the project is progressing smoothly
- ❑ He sets the meetings, monitor the work and facilitate release planning
- ❑ Two important task of scrum master are:
 - ✓ Protecting the team from outside disturbance
 - ✓ Clears the ways for the team by helping them to solve their problems



The SCRUM Team [4]

10

- In Scrum, an ideal team would include seven members, plus or minus two. Usually, teams are comprised of cross-functional members, including software engineers, architects, programmers, analysts, QA experts, testers, UI designers, etc. It is recommended all team members be located in the same room, called the team room.
- The team has the autonomy to determine how and when to complete its work. As long as the team finishes its work by the deadline and under budget, it is entirely up to the team to determine how that happens.



Tuotteen Omistaja
(Product Owner)



Tiimi
(Team)

sysart



Talkkari
(Scrum Master)

The Product Owner [6]

11

- ❑ In Scrum, the Product Owner is the one person responsible for a project's success.
- ❑ The Product Owner outlines work in the Product backlog
- ❑ Product Owner makes sure that right features to be included in the product backlog
- ❑ Of course, he or she must also consider the stakeholders and the team



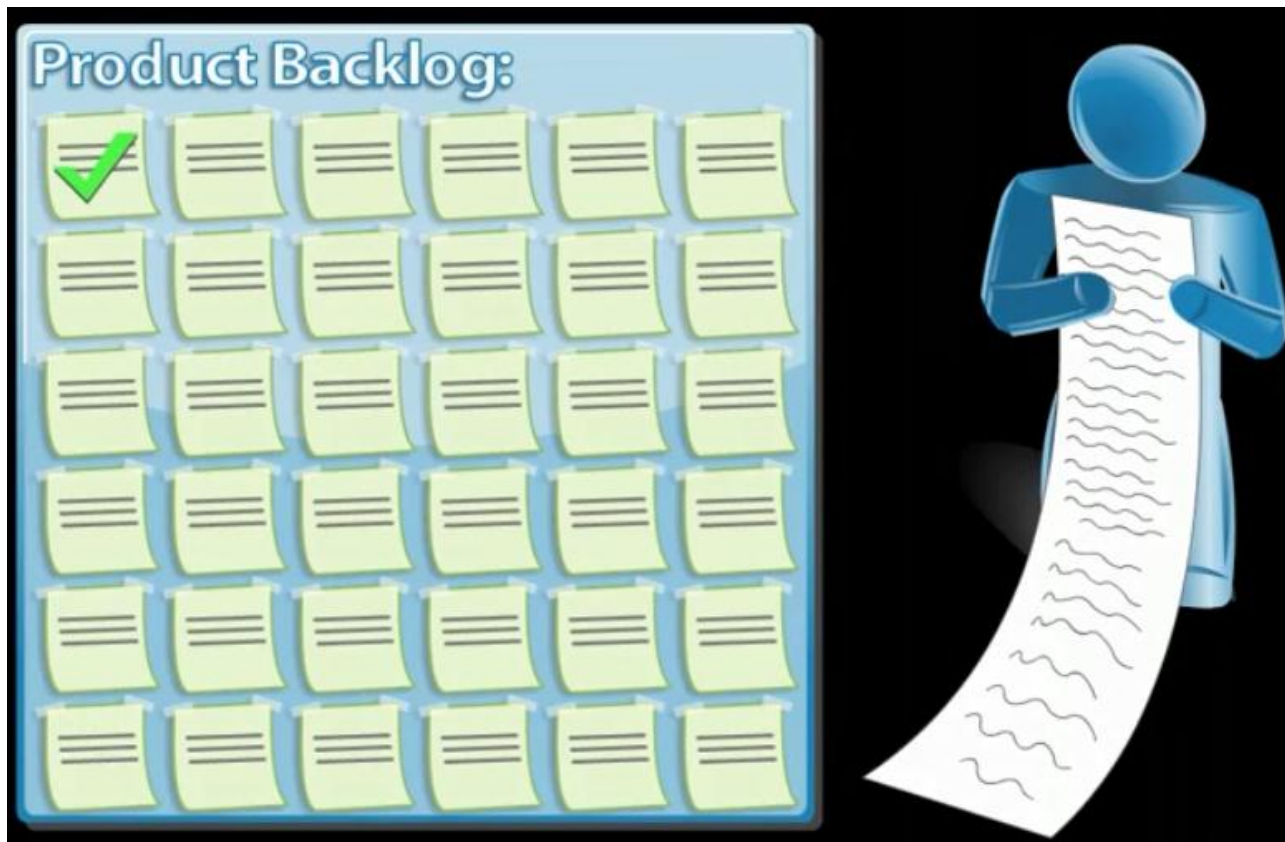
Scrum Artifacts

- ❑ Product Backlog
- ❑ Sprint Backlog
- ❑ Sprint
- ❑ Burn down Chart

Product Backlog

13

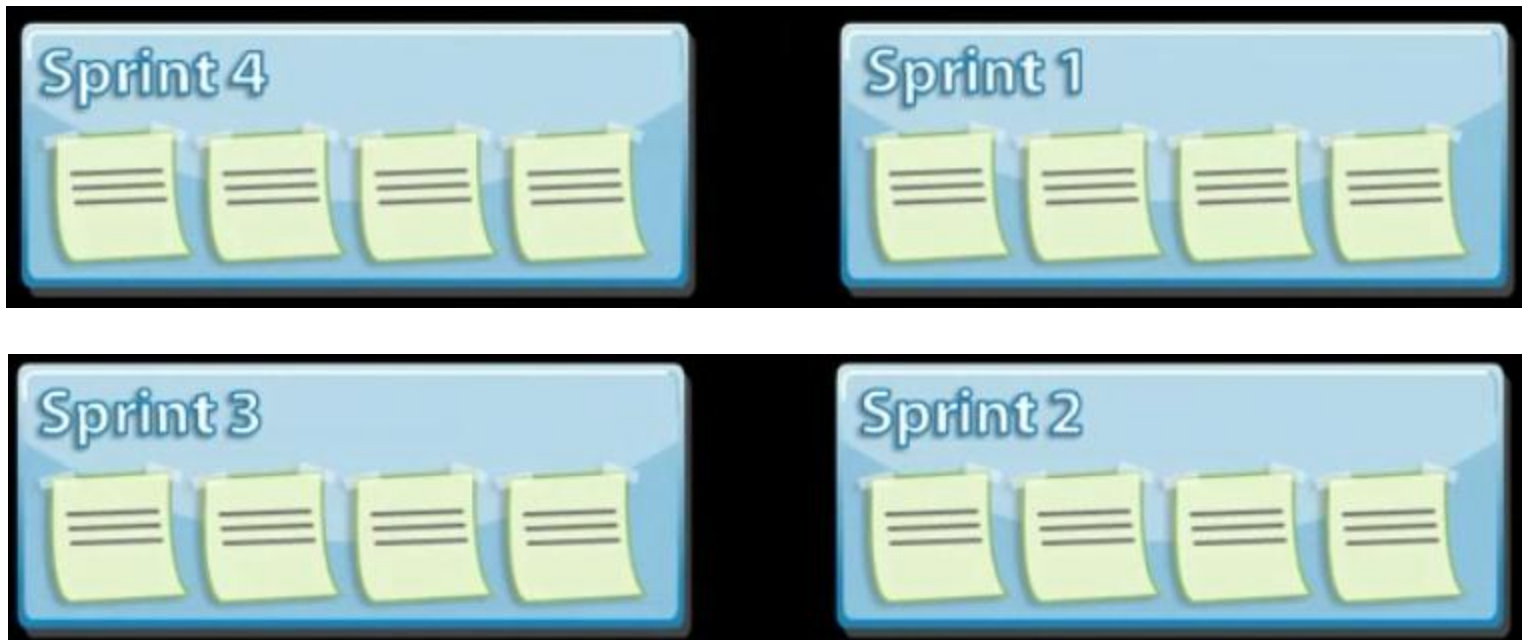
- ❑ Contains all the currently known requirements for a product
- ❑ Is managed by the Product Owner and can change as needed



Sprint Backlog

14

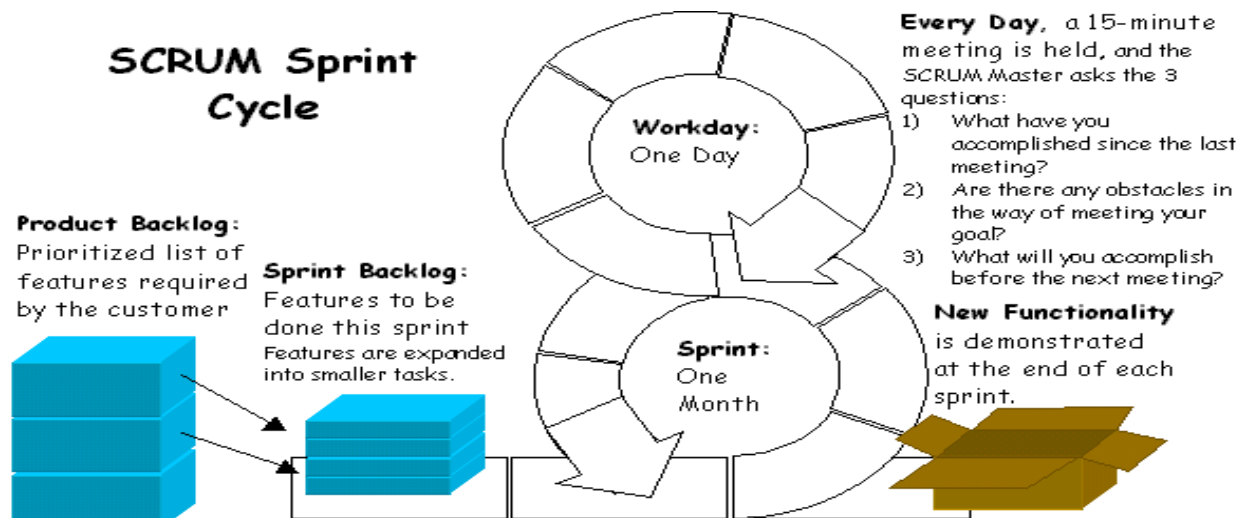
- Contains the set of prioritized Product Backlog items that are currently being worked on



Sprint [7]

15

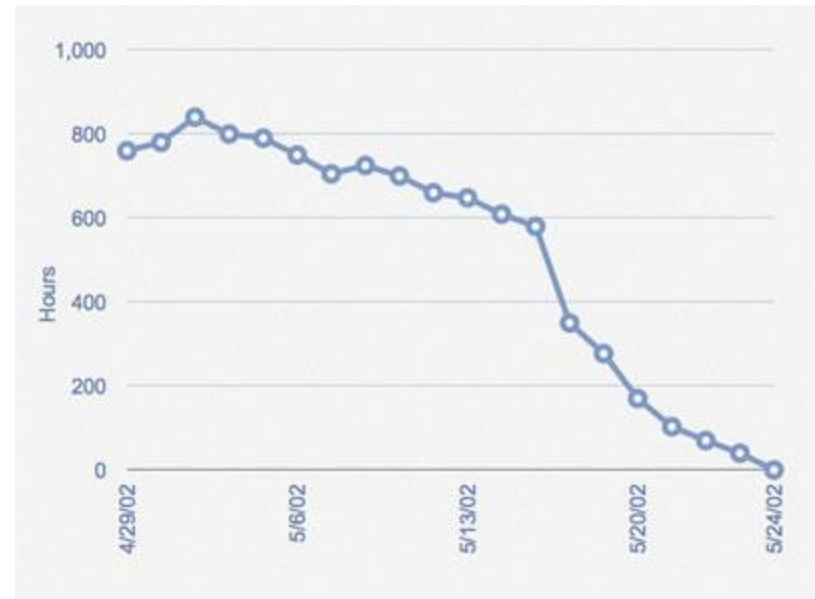
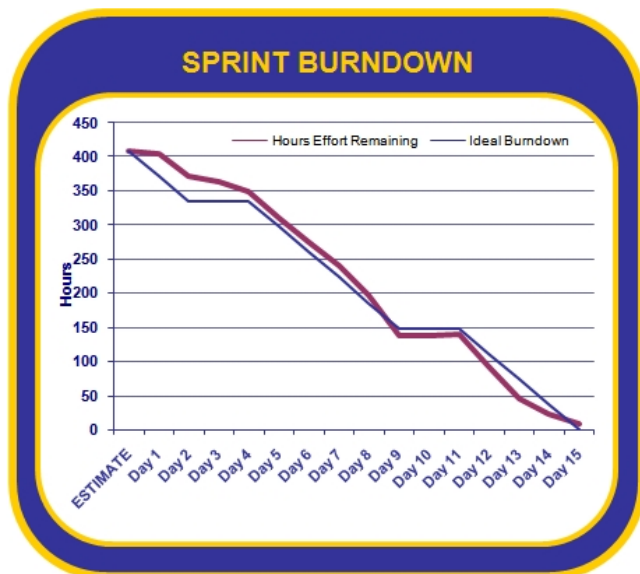
- ❑ The product is developed in a series of 1-to-4-week iterations, or sprints.
- ❑ The sprint has 4 major steps:
 - ✓ **Develop** the product further.
 - ✓ **Wrap** up the work - get it ready to be evaluated and integrated.
 - ✓ **Review** the work done in this sprint.
 - ✓ **Adjust** for any changes in requirements or plans.
- ❑ Results in an incremental delivery of usable product



Sprint Burn down Chart [9]

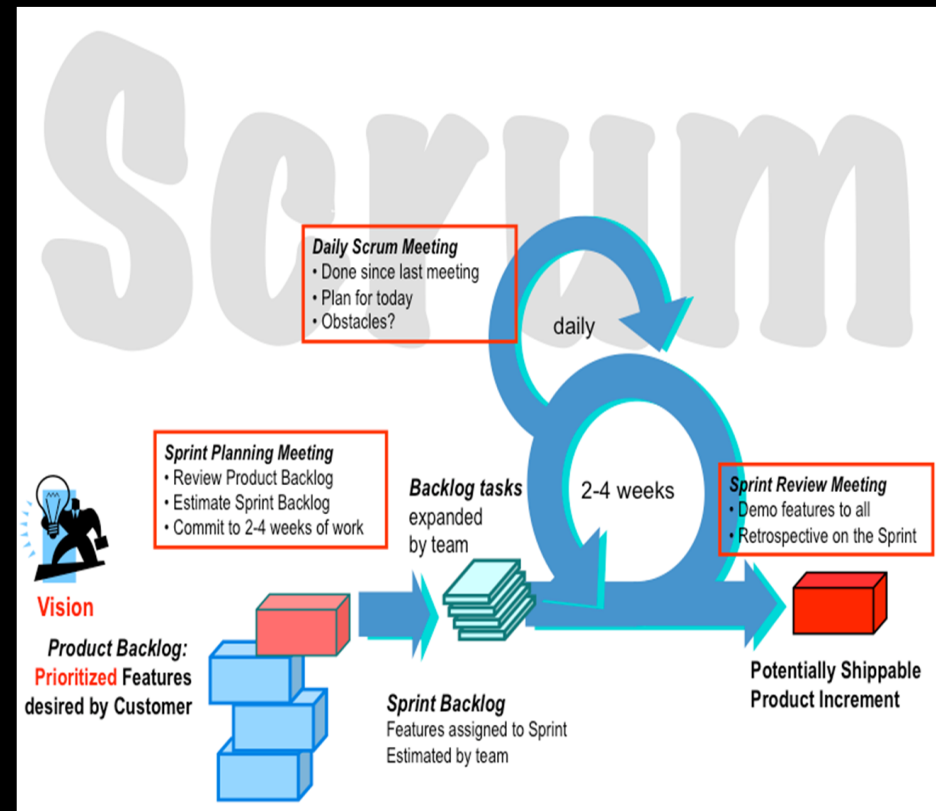
16

- ❑ The estimated work remaining in the sprint is calculated daily and graphed, resulting in a Sprint Burn down Chart
- ❑ The vertical axis displays the hours of effort remaining for the Sprint.
- ❑ The horizontal axis displays the days of the Sprint.
- ❑ The burn down is supposed to be shown by the line of descent from the start of the Sprint with the starting hours, down to the end of the Sprint with no hours remaining.



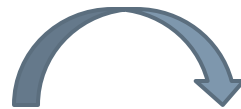
Scrum Meetings

- ❑ Release Planning Meeting
- ❑ Sprint Planning Meeting
- ❑ Sprint Review Meeting
- ❑ Sprint Retrospective Meeting
- ❑ Daily Scrum Meeting



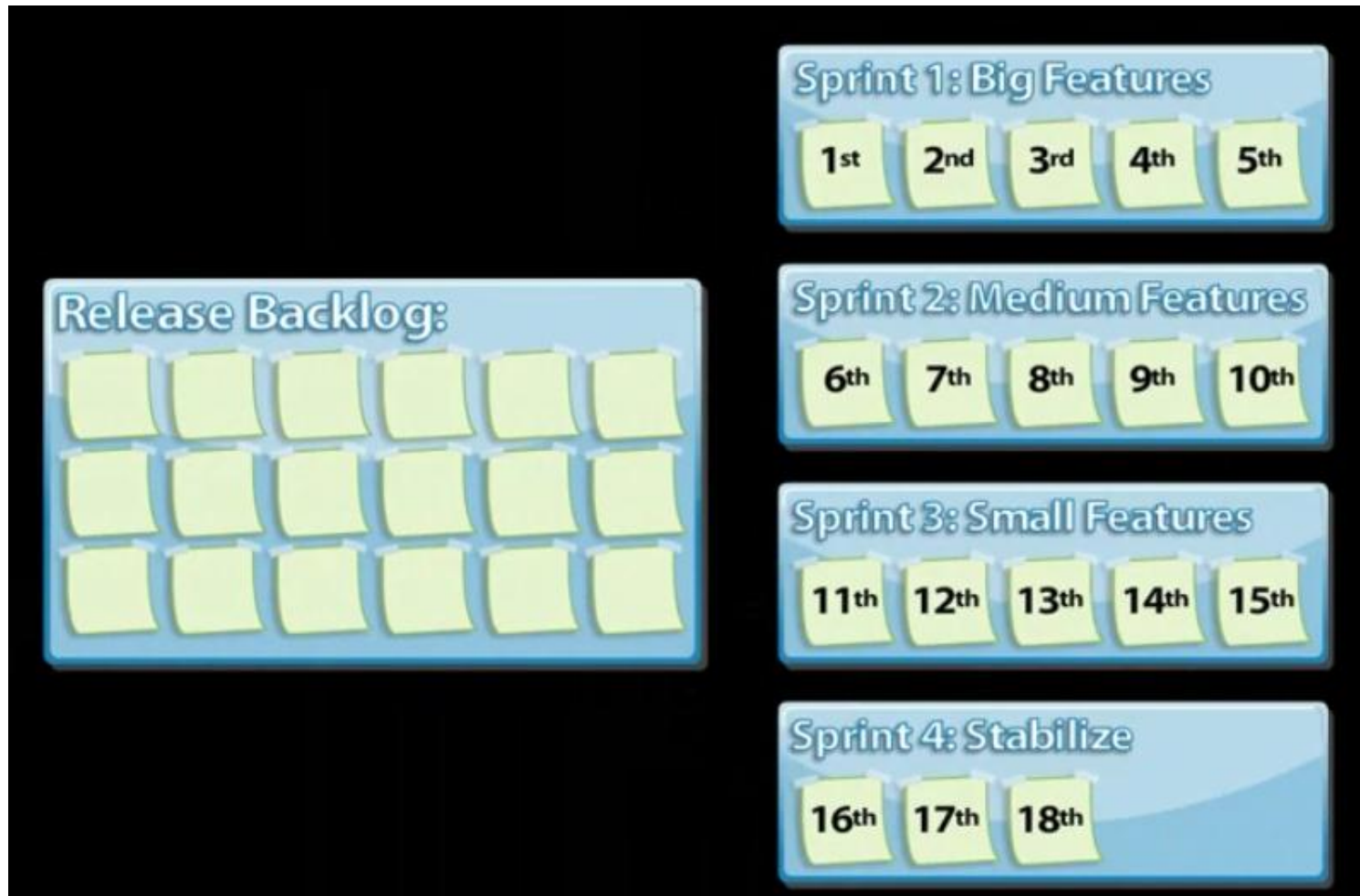
Release Planning

18



Release Planning

19



Sprint Planning Meeting

20

- ❑ A meeting at the beginning of a sprint where the sprint is planned.
- ❑ Items from the Product Backlog are selected to be completed in the sprint, based on the priorities set by the Product Owner. Eight hour time limit
 - ✓ (1st four hours) Product Owner + Team: dialog for prioritizing the Product Backlog
 - ✓ (2nd four hours) Team only: plan for the Sprint, resulting in the Sprint Backlog



Sprint Review Meeting [7]

21

- ❑ Review the work that was completed and not completed
- ❑ Present the completed work to the stakeholders (a.k.a. “the demo”)
- ❑ Four hour time limit



Sprint Retrospective Meeting [8]

22

- ❑ The sprint retrospective meeting is time boxed to 3 hours.
- ❑ It is attended only by the team, the scrum master and the product owner. The product owner is optional.
- ❑ **Make continuous process improvements**
- ❑ Start the meeting by having all team members answer two questions;
 - ✓ What went well during the sprint?
 - ✓ What could be improved in the next sprint?



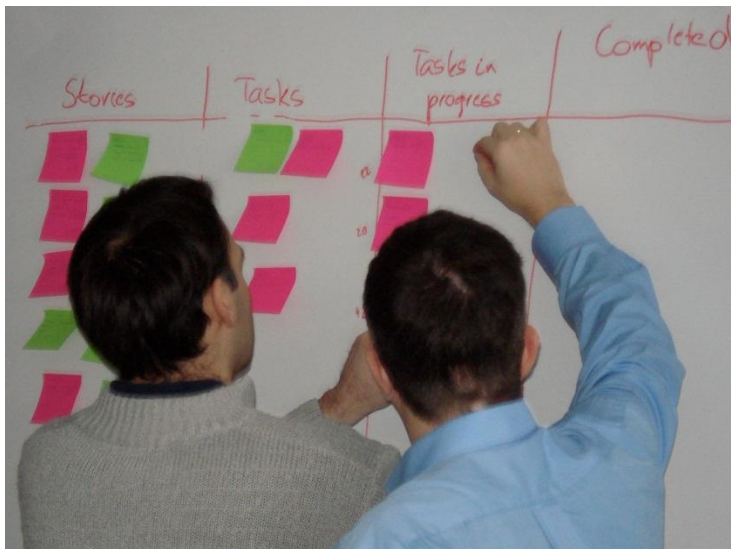
Daily SCRUM Meeting

23

- ❑ Brief 'Stand-up' meeting each morning with SCRUM Team only
- ❑ Duration is 15 min
- ❑ Three questions are asked
 - ✓ What value did you add yesterday?
 - ✓ What value will you add today?
 - ✓ What will stop you making progress?



Team members in the scrum meeting

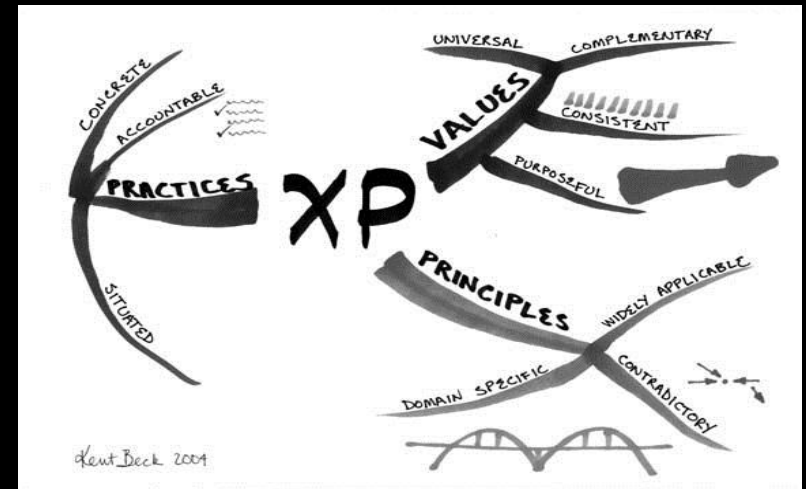


SCRUM

24

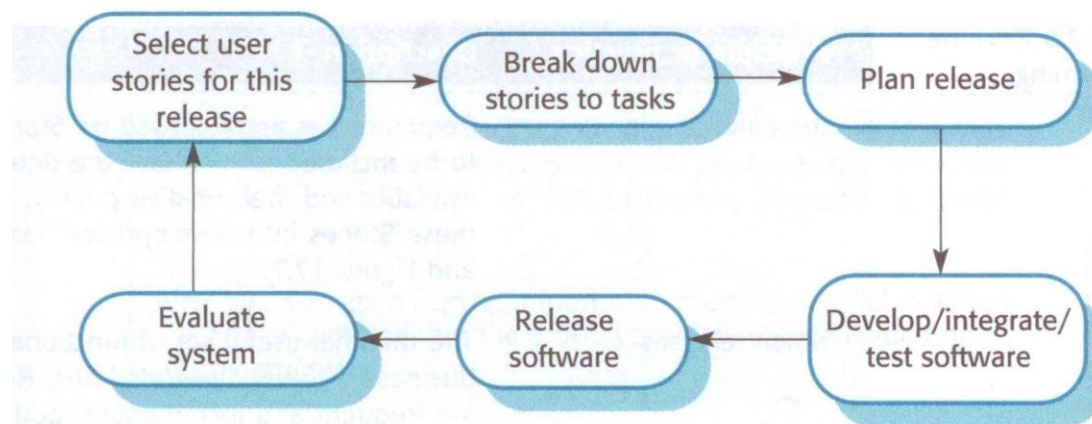
DEMO

- ❑ XP
- ❑ XP Release Cycle
- ❑ Requirement Scenarios
- ❑ Testing in XP
- ❑ Pair Programming



Extreme programming [21]

- **Extreme Programming (XP)** is a software development methodology which is intended to improve software quality, productivity and responsiveness to changing customer requirements. As a type of agile software development, it advocates frequent "releases" in short development cycles (time boxing).
- Other elements of extreme programming include: programming in pairs, doing extensive code review, unit testing of all code, avoiding programming of features until they are actually needed, a flat management structure [22], simplicity and clarity in code, expecting changes in the customer's requirements as time passes and frequent communication with the customer and among programmers



The XP release cycle

Requirements scenarios

- ❑ In XP, user requirements are expressed as scenarios or user stories.
- ❑ These are written on cards and the development team break them down into implementation tasks.
- ❑ The customer chooses the stories for inclusion in the next release based on their priorities and the schedule estimates.

Story card for document downloading

Downloading and printing an article

First, you select the article that you want from a displayed list. You then have to tell the system how you will pay for it—this can either be through a subscription, through a company account or by credit card.

After this, you get a copyright form from the system to fill in. When you have submitted this, the article you want is downloaded onto your computer.

You then choose a printer and a copy of the article is printed. You tell the system printing has been successful.

If the article is a print-only article, you can't keep the PDF version, so it is automatically deleted from your computer.

Testing in XP

- ❑ Test-first development [25]
- ❑ Incremental test development from scenarios.
- ❑ User involvement in test development and validation.
- ❑ Automated test harnesses [12] are used to run all component tests each time that a new release is built.

Task cards for document downloading

Task 1: Implement principal workflow

Task 2: Implement article catalog and selection

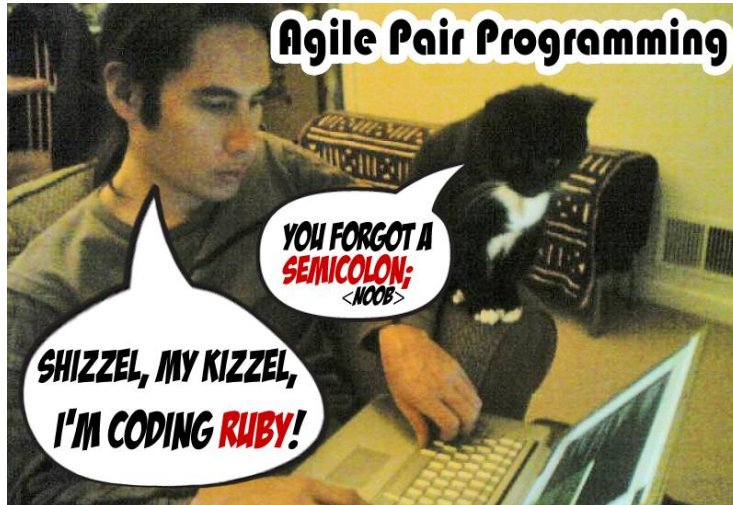
Task 3: Implement payment collection

Payment may be made in 3 different ways. The user selects which way they wish to pay. If the user has a library subscription, then they can input the subscriber key which should be checked by the system. Alternatively they can input an organisational account number. If this is valid, a debit of the cost of the article is posted to this account. Finally they may input a 16 digit credit card number and expiry date. This should be checked for validity and, if valid a debit is posted to that credit card account.



Pair programming

- ❑ In XP, programmers work in pairs, sitting together to develop code.
- ❑ This helps develop common ownership of code and spreads knowledge across the team.
- ❑ It serves as an informal review process as each line of code is looked at by more than 1 person.
- ❑ It encourages refactoring as the whole team can benefit from this. [14]
- ❑ **Measurements suggest that development productivity with pair programming is similar to that of two people working independently but**



Adaptive Software Development

- Introduction
- ASD Cycle



Adaptive Software Development [15][16]

31

- ❑ **Adaptive Software Development** is a software development process that grew out of rapid application development work by Jim Highsmith and Sam Bayer.
- ❑ ASD replaces the traditional waterfall cycle with a repeating series of *speculate*, *collaborate*, and *learn* cycles.
- ❑ The characteristics of an ASD life cycle are that it is mission focused, feature based, iterative, time boxed [13], risk driven, and change tolerant.



The adaptive lifecycle

Adaptive Software Development Cycle [1,16]

32

- ❑ **Speculation-** During this phase coders attempt to understand the exact nature of the software and the requirements of the users. This phase relies on bug and user reports to guide the project.
- ❑ **Collaboration** phase is when the individual developers solidify what they are each doing and how to combine their portions. This phase is generally completely in-house.
- ❑ **Learning** cycles results in releasing the newest version of the software to users. Either they can accept it without any modifications or wants some change.



Adaptive Software development

33

DEMO

Feature Driven Development

- ❑ Introduction
- ❑ FDD Activities
- ❑ Practices in FDD

Feature-driven development [2,17]



35

- ❑ **Feature-driven development (FDD)** is an iterative and incremental software development process.
- ❑ A feature is a small, client-valued function. For example, “Calculate the total of a sale”, “Validate the password of a user”.
- ❑ Features are to FDD as **user stories** are to **XP** – they’re a primary source of requirements and the primary input into your planning efforts.
- ❑ The Activities of FDD are
 - ✓ Develop Overall Model
 - ✓ Build Feature List
 - ✓ Plan By Feature
 - ✓ Design By Feature
 - ✓ Build By Feature

Feature-driven development [17]

36

❑ Develop Overall Model

- ✓ The project starts with a high-level walkthrough of the scope of the system and its context. Next, detailed domain walkthroughs are held for each modeling area.

❑ Build Feature List

- ✓ The knowledge that is gathered during the initial modeling is used to identify a list of features. This is done by functionally decomposing the domain into subject areas.
- ✓ Subject areas each contain business activities, the steps within each business activity form the categorized feature list.
- ✓ Features should not take more than two weeks to complete, else they should be broken down into smaller pieces.

❑ Plan By Feature

- ✓ Now that the feature list is complete, the next step is to produce the development plan. Class ownership is done by ordering and assigning features as classes to programmers.

Feature-driven development [17]

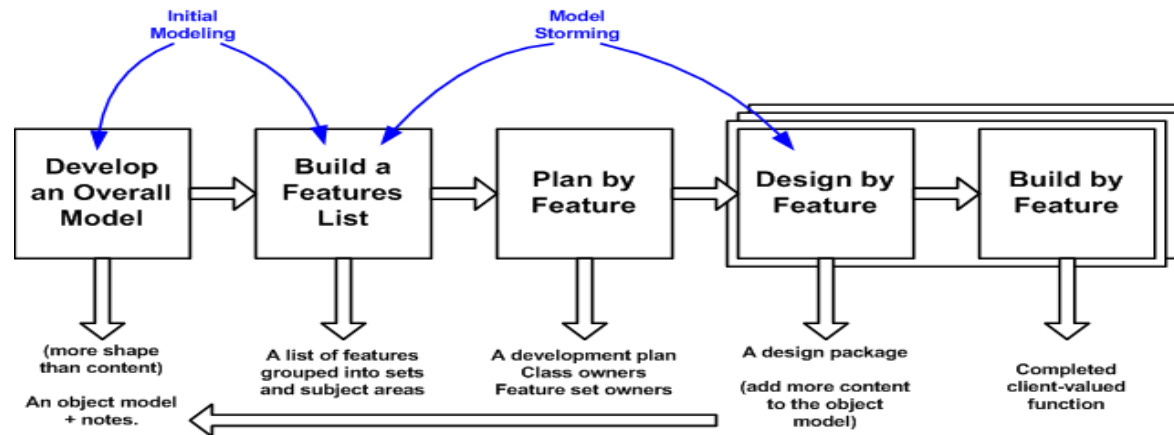
37

□ Design By Feature

- ✓ A design package is produced for each feature. A chief programmer selects a small group of features that are to be developed within two weeks. Together with the corresponding class owners, the chief programmer works out detailed sequence diagrams for each feature and refines the overall model followed by design inspection.

□ Build By Feature

- ✓ After a successful design inspection, a client-valued function (feature) is being produced. The class owners develop the actual code for their classes. After a unit test and a successful code inspection, the completed feature is promoted to the main build.

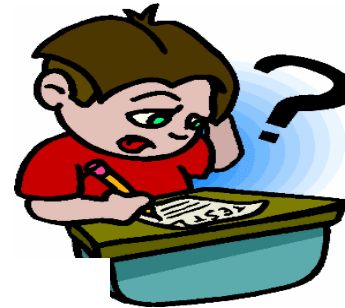


Feature-driven development [17]

38

■ Practices in FDD

- ✓ Domain Object Modeling



- ✓ Developing by Feature



- ✓ Individual Class Ownership



- ✓ Feature Teams



Feature-driven development [17]

39

■ Practices in FDD

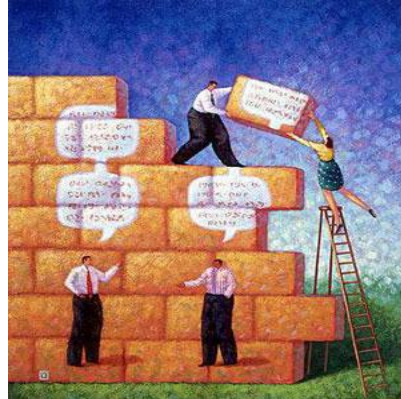
✓ Inspection



✓ Configuration Management



✓ Regular Builds



✓ Visibility of Progress and Results



Feature-driven development

40

DEMO

Lean Software Development

- Introduction
- Lean Principles

Lean software development [18]

42

- ❑ **Lean software development** is a translation of Lean manufacturing and Lean IT principles and practices to the software development domain. Adapted from the Toyota Production System.
- ❑ Lean is an Agile methodology which can also be seen as a philosophy
- ❑ The core idea is to maximize **customer value** while minimizing waste. Simply, lean means creating more value for customers with fewer resources. [19]



Lean principles [18]

43

- ❑ Eliminating Waste



- ❑ Amplify Learning



- ❑ Decide as late as possible



- ❑ Deliver as fast as possible



Lean principles [18]

44

- ❑ Empower the Team



- ❑ Build Integrity in

- ❑ See the Whole



Lean Software Development

45

DEMO

References

1. <http://www.wisegEEK.com/what-is-adaptive-software-development.htm>
2. <http://www.agilemodeling.com/essays/fdd.htm>
3. <http://3back.com/scrum/certified-scrummaster-training/>
4. <http://scrummethodology.com/the-scrum-team-role/>
5. YouTube - Scrum Master in Under 10 Minutes (HD) by @hamids.wmv
6. <http://scrummethodology.com/scrum-product-owner/>
7. <http://www.codeproject.com/KB/architecture/scrum.aspx>
8. <http://abrachan.net/scrum/scrum/sprint-retrospective-meeting/>
9. http://epf.eclipse.org/wikis/scrum/Scrum/workproducts/sprint_burndown_chart_F647C347.html
10. http://en.wikipedia.org/wiki/Scrum_%28development%29
11. <http://msdn.microsoft.com/en-us/library/dd997796.aspx>
12. http://en.wikipedia.org/wiki/Test_harness
13. <http://en.wikipedia.org/wiki/Timeboxing>
14. http://en.wikipedia.org/wiki/Code_refactoring
15. http://en.wikipedia.org/wiki/Adaptive_Software_Development
16. <http://productdevelop.blogspot.com/2011/07/what-is-adaptive-software-development.html>

References

47

17. http://en.wikipedia.org/wiki/Feature-driven_development
18. http://en.wikipedia.org/wiki/Lean_software_development
19. <http://www.lean.org/whatslean/>
20. <http://cseweb.ucsd.edu/users/wgg/CSE131B/Design/node6.html>
21. http://en.wikipedia.org/wiki/Extreme_Programming
22. <http://www.learnmanagement2.com/flat%20structure.htm>
23. http://books.google.com.pk/books?id=8o1eom6ifIMC&pg=PA17&lpg=PA17&dq=difference+between+perceived+integrity+and+conceptual+integrity&source=bl&ots=m1VuEK93KM&sig=5TcbxyjCdO57OJ7VlvC4TLk3ELI&hl=en&ei=VXmWTruiLITKhAeng62EBA&sa=X&oi=book_result&ct=result&resnum=3&ved=0CC0Q6AEwAg#v=onepage&q=difference%20between%20perceived%20integrity%20and%20conceptual%20integrity&f=false
24. <http://scrumalliance.org/articles/466-selforganizing-teams-what-and-how>
25. <http://www.extremeprogramming.org/rules/testfirst.html>

For any query Feel Free to ask



48

