(11224) INTRODUCTION TO SOFTWARE ENGINEERING

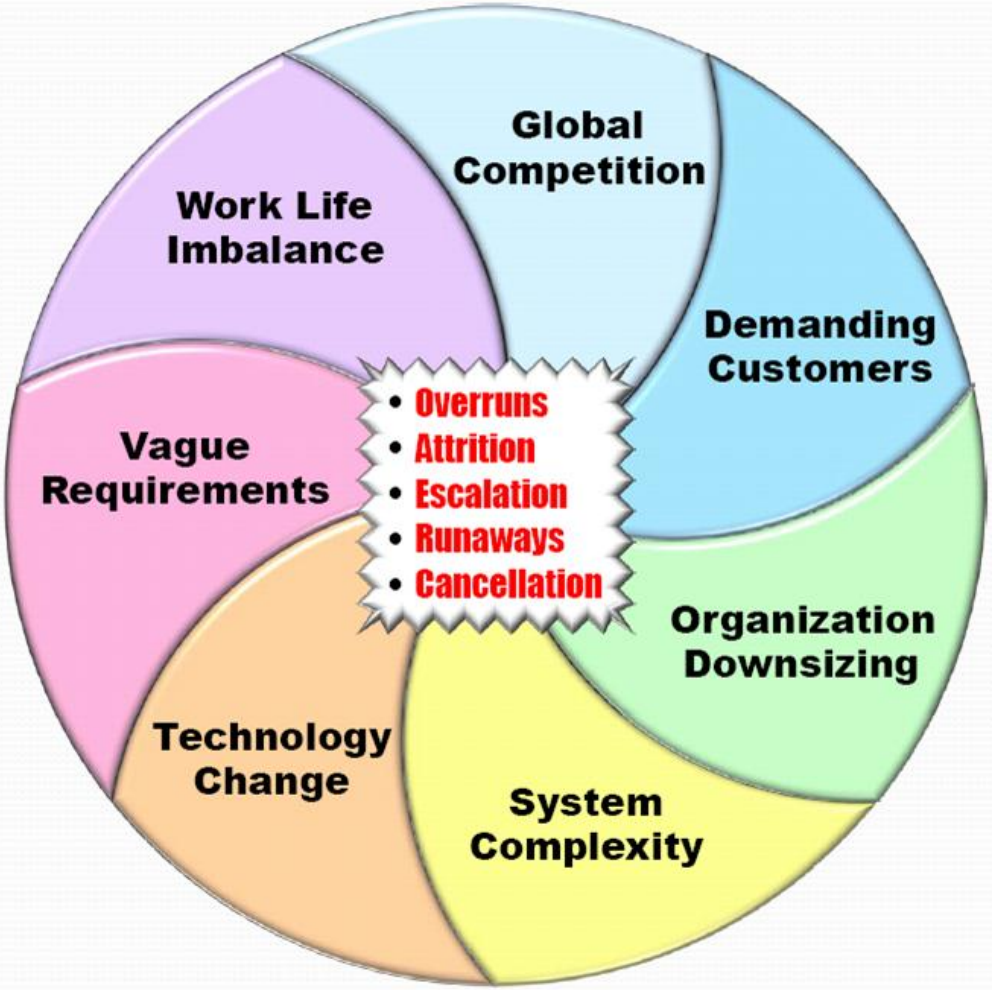
Lecture 15: Agile Methods

Shereen Fouad

Motivation

- Large projects requires a plan driven approach with precise and well analysed and documented specifications and requirements.
- High cost in preparation of heavy requirements capture document and system specification
- Low speed of development and long delivery times.
- Today’s business applications cannot wait long to deploy results of software development projects or they will lose business to their faster competitors.
- You need to be competitive in today’s fast moving market place

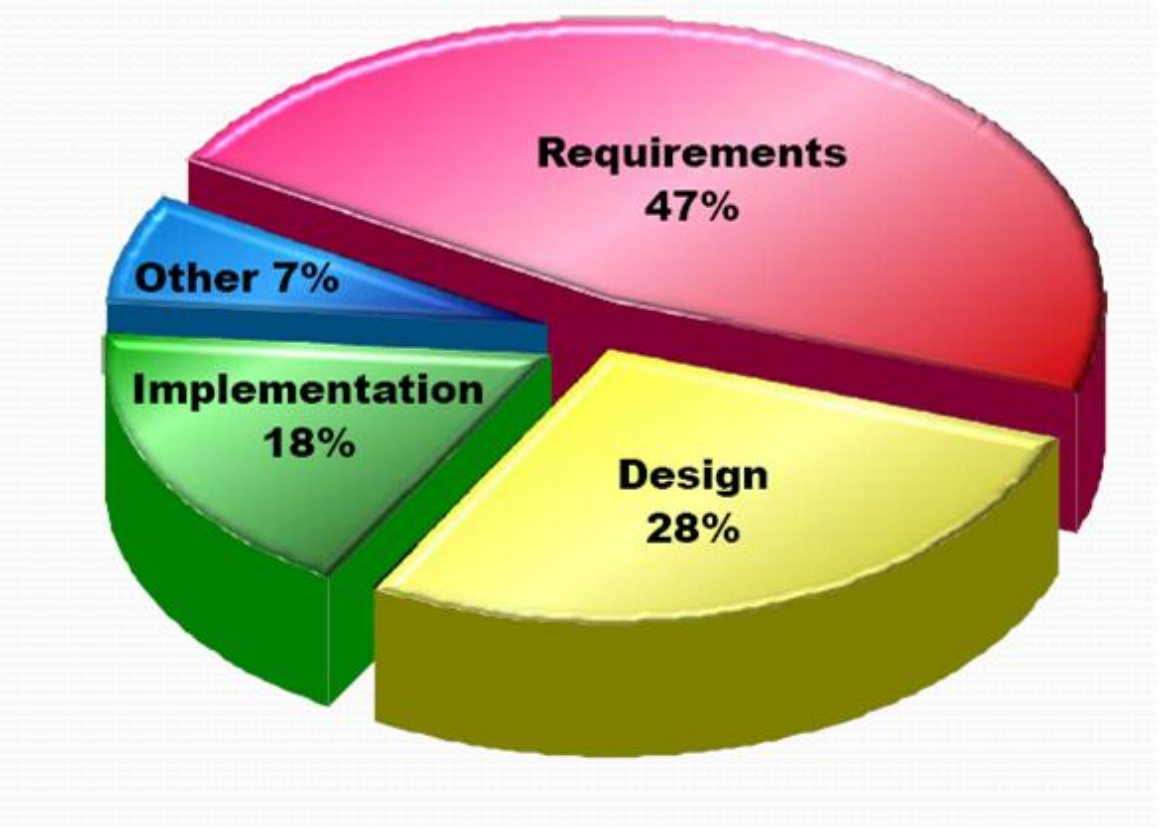
Today's Business Environment



Pine, B. J. (1993). *Mass customization: The new frontier in business competition*. Boston, MA: Harvard Business School Press.

Wasted Requirement specifications

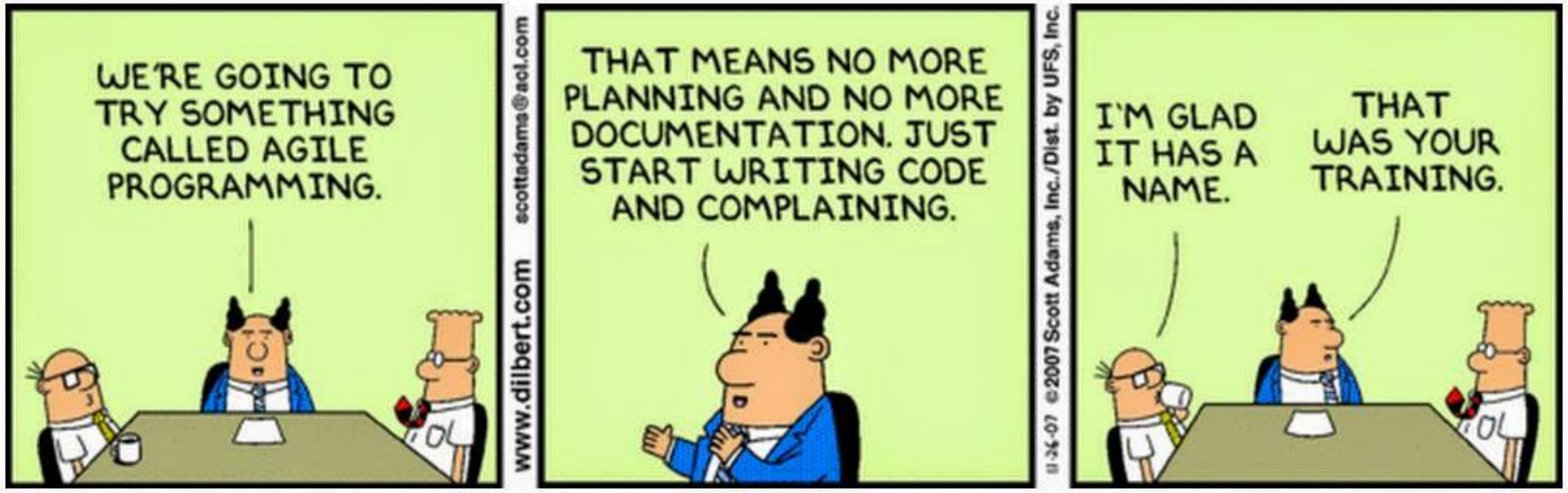
Business environment changes very quickly.
A full bore planning and specification model, when it finally delivers its result, may do so into a significantly different environment than that for which it was planned.



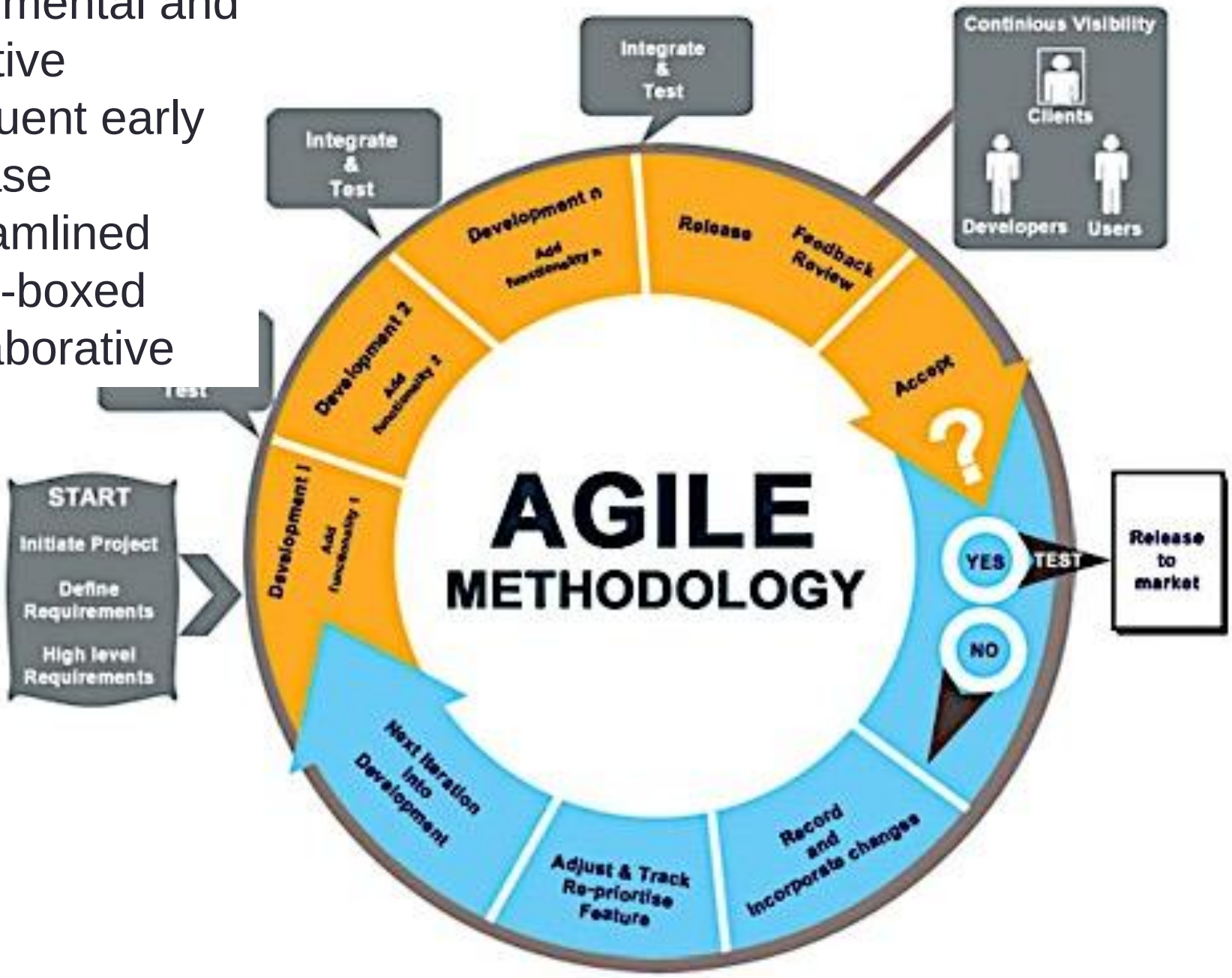
RUP Criticism

- ‘High ceremony methodology’
- Bureaucratic: process for everything
- Slow: must follow process to comply
- Excessive overhead: rationale, justification, documentation, reporting, meetings, permission

Agile methods have grown specifically to cope with such needs.



- Incremental and Iterative
- Frequent early release
- Streamlined
- Time-boxed
- Collaborative



https://www.interaction-design.org/ux-daily/110/problems-associated-with-integrating-agile-methods-in-ucd-

Features of different agile methods

- No detailed system specification is produced
- Design documentation is minimised or generated automatically by Computer Aided Software Engineering (CASE) tools
- Specification, design and implementation is interleaved
- The user requirements document only defines the most important aspects of the system rather than the detailed requirements

Features of different agile methods

- The system is developed incrementally in versions
- End users are heavily involved in specifying and evaluating each version of the system
- System interfaces are typically developed using interactive CASE tools.
- Increments are short: typically a new version is released every two to three weeks
- Documentation is minimised

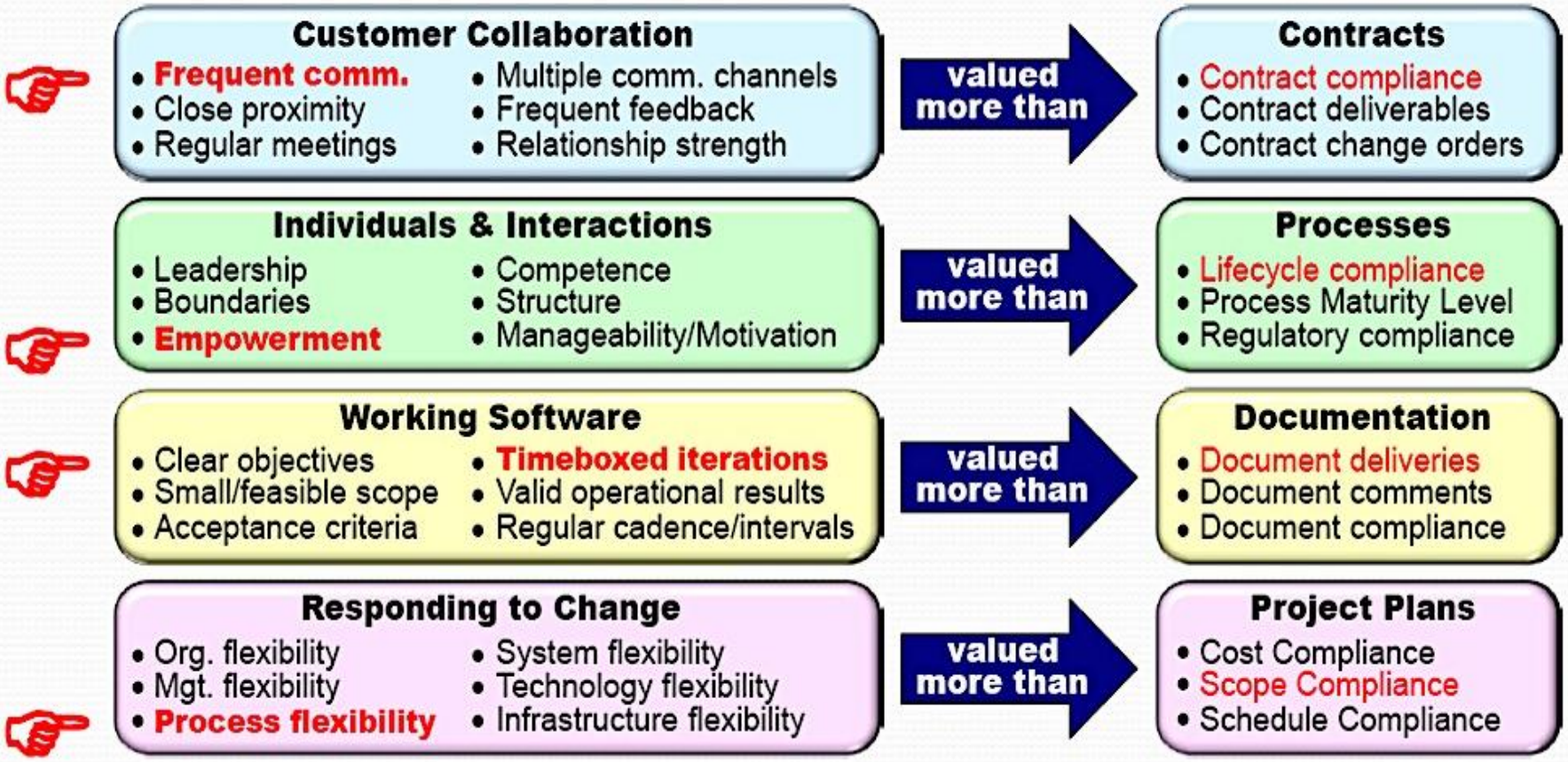
Manifesto for Agile Software Development

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, we value the items on the left more.

Kent Beck
Mike Beedle
Arie van Bennekum
Alistair Cockburn
Ward Cunningham
Martin Fowler
James Grenning
Jim Highsmith
Andrew Hunt
Ron Jeffries
Jon Kern
Brian Marick
Robert C. Martin
Steve Mellor
Ken Schwaber
Jeff Sutherland
Dave Thomas



Agile Manifesto. (2001). *Manifesto for agile software development*. Retrieved September 3, 2008, from http://www.agilemanifesto.org

The Agile Manifesto is based on the following 12 principles

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

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

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

04 Business people and developers must work together daily throughout the project.

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

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

07 Working software is the primary measure of progress.

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

09 Continuous attention to technical excellence and good design enhances agility.

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

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.

Agile Methods

- Agile methods:
  - Scrum
  - Extreme Programming
  - Adaptive Software Development (ASD)
  - Dynamic System Development Method (DSDM)
  - …

What is Scrum?

Definition from rugby football:

a scrum is a way to restart the game after an interruption, where the forwards of each side come together in a tight formation and struggle to gain possession of the ball when it is tossed in among them

Scrum in 100 words

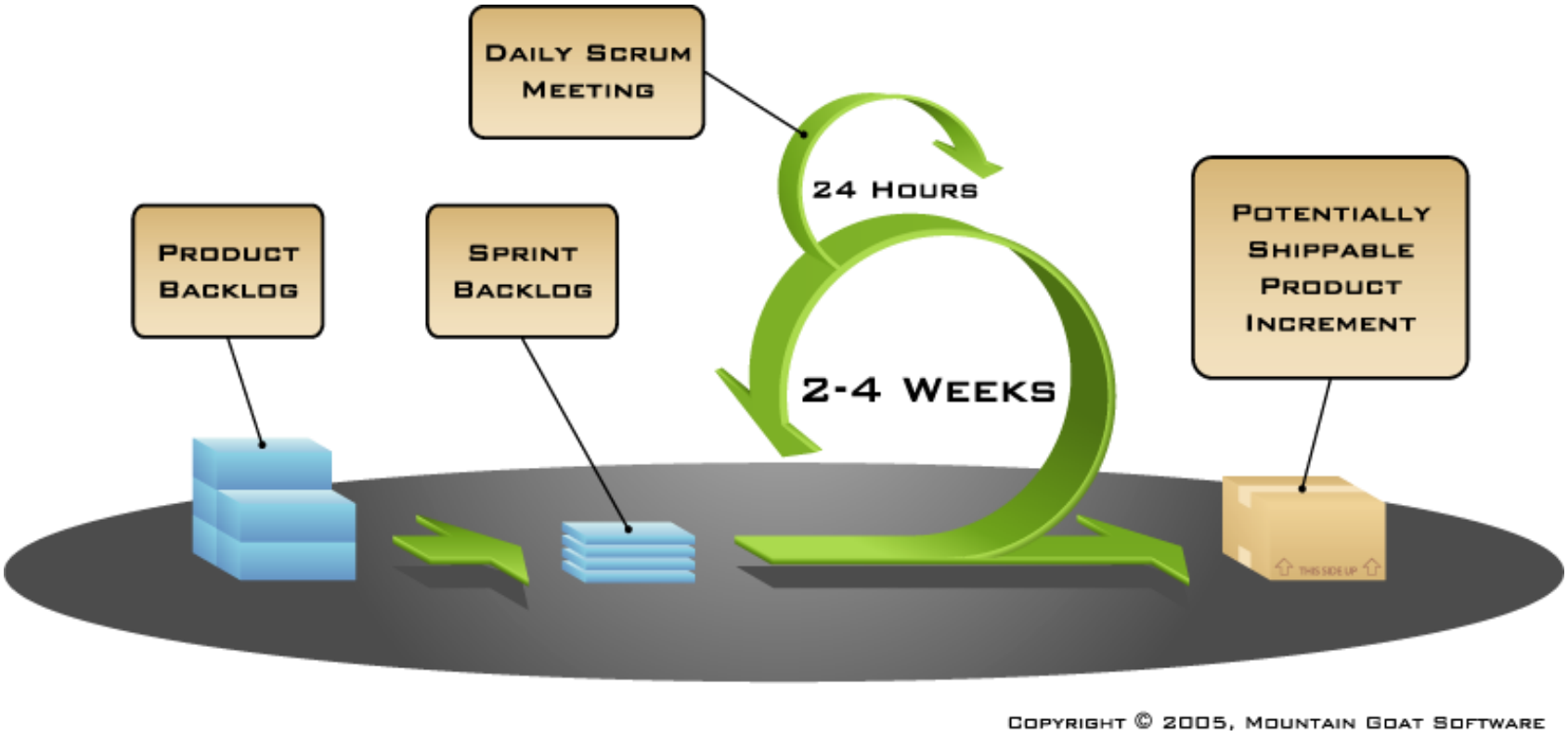
Scrum is an agile process that allows us to focus on delivering the highest business value in the shortest time.

It allows us to rapidly and repeatedly inspect actual working software (every two weeks to one month).

The business sets the priorities. Our teams self-manage to determine the best way to deliver the highest priority features.

Every two weeks to a month anyone can see real working software and decide to release it as is or continue to enhance for another iteration.

How Scrum Works?



Backlog – prioritize list of tasks to be completed.

Scrum – Meeting to discuss sprint status. Attended by all and managed by ScrumMaster

Sprint – A few days of intense work to produce a product increment.

Characteristics

- Self-organizing teams
- Product progresses in a series of month-long “sprints”
- Requirements are captured as items in a list of “product backlog”
- No specific engineering practices prescribed
- Uses generative rules to create an agile environment for delivering projects
- One of the “agile processes”

Quiz!!

- **Agile Software Development is based on**
  a) Incremental Development
  b) Iterative Development
  c) Linear Development
  d) Both a and b
- **Which of the following does apply to agility to a software process?**
  a) Uses incremental product delivery strategy
  b) Only essential work products are produced
  c) Eliminate the use of project planning and documentation
  d) All of the mentioned

Quiz!!

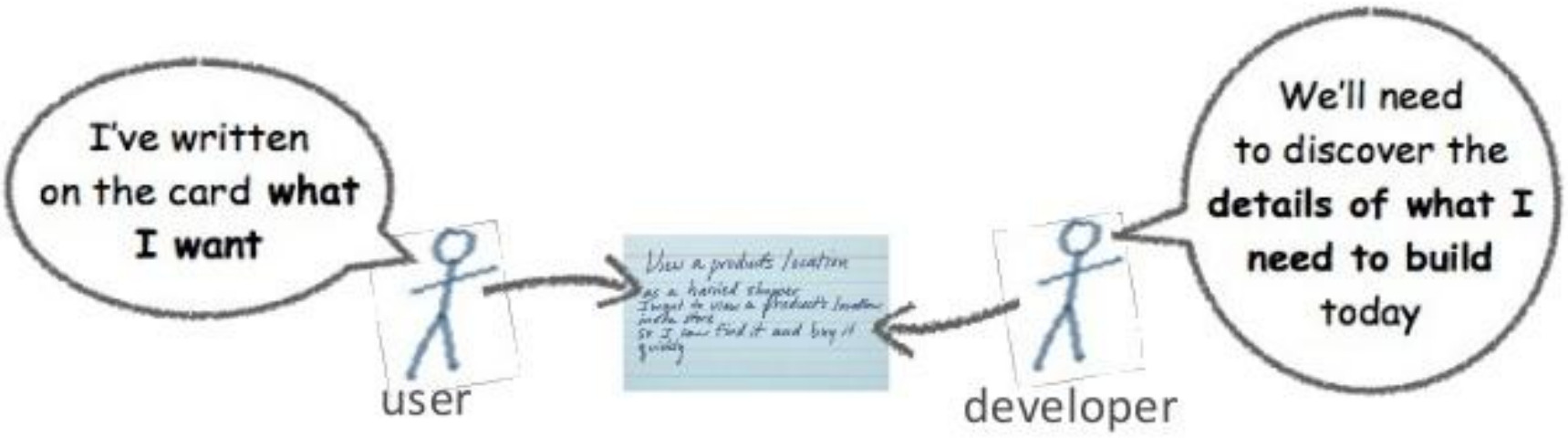
- **In scrum the team activity is monitored and coordinated on ..... basis.**
  A) hourly
  B) daily
  C) weekly
  D) monthly

- **The first step in Scrum is for the Product Owner to articulate the product vision. Eventually, this evolves into a refined and prioritized list of features called the .....**
  A) Sprint Backlog
  B) Whiteboard
  C) Product Backlog
  D) All of above

Extreme Programming (XP)

- The best known and most commonly employed agile process
- The process is cyclic, with a cycle usually taking 2 weeks.
- The planning and each iteration is based on User Stories.
- In many ways, user stories serve the same purpose as Use Cases in the Rational Unified Process, but they are different.

Stories facilitate a conversation with a goal of shared understanding

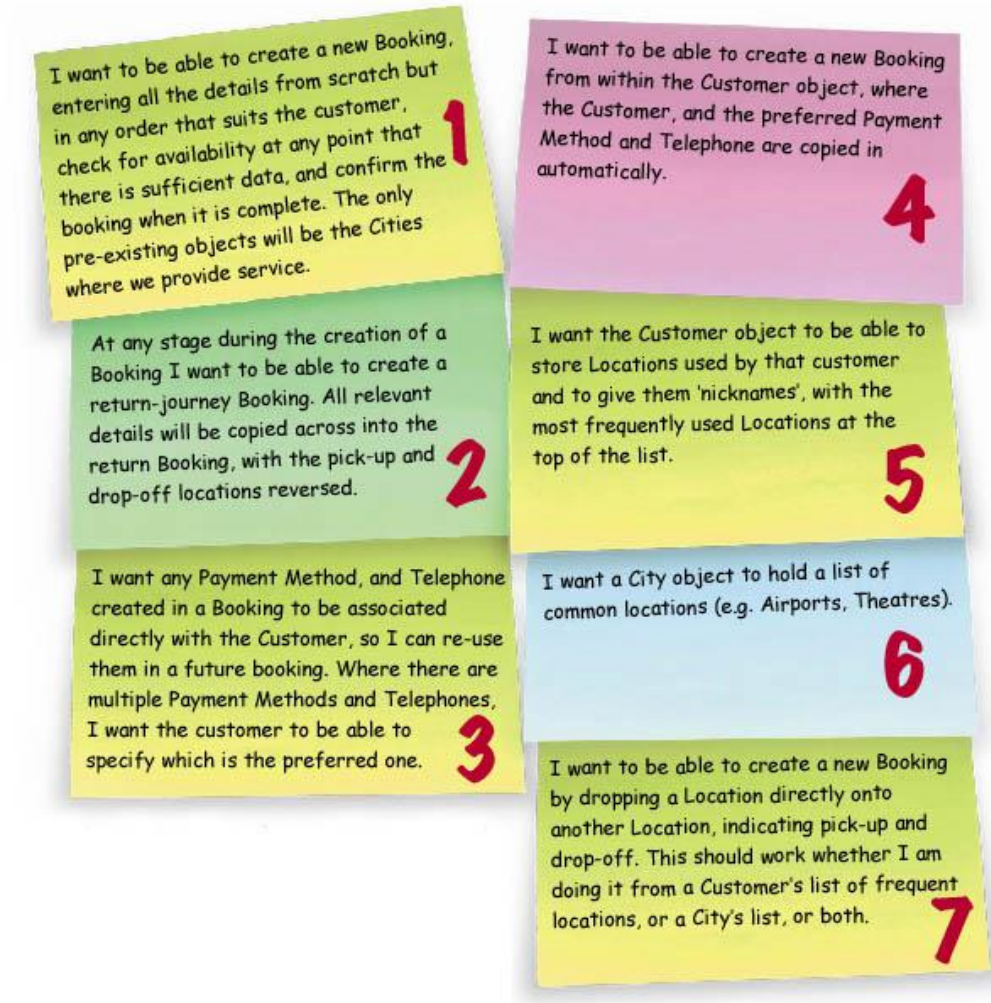


The process depends on “just-in-time" planning and implementation to deal with changing requirements,

minimising work that might be wasted by a new requirement that changes what was assumed to be required before.

User stories

- User stories are short,
- plain text descriptions,
- written by the clients (NOT the program developers), of things that the system should do.
- Each user story is written on an 3x5inch index card



Collection of user stories

- NOT sufficient to serve as a functional requirements document.
- Form the basis for planning,
- Can be used to estimate how long it will take to implement that piece of functionality.
- Annotated with some indication of the importance of that piece of functionality (how urgent it is to implement it)

Pro/Con

- Advantages
  - Completely developed and tested features in short iterations
  - Simplicity of the process
  - Clearly defined rules
  - Increasing productivity
  - Self-organizing
  - each team member carries a lot of responsibility
  - Improved communication
  - Combination with Extreme Programming
- Drawbacks
  - “Undisciplined hacking” (no written documentation)
  - Violation of responsibility
  - Current mainly carried by the inventors
  - Stressfull

References

- A number of slides in this talk is based on:
  - Alan P. Sexton hand-outs (Introduction to Software Engineering. The University of Birmingham. Spring Semester 2014)

Thank YOU ☺