Roadmap

- SE process management
  - Waterfall model
  - Incremental methods
  - Agile/XP methods, SCRUM
  - Iterative / spiral methods (eg, RUP)
  - Evolutionary methods
  - V -Model
- CMMI

© Scott Adams, Inc./Dist. by UFS, Inc.

The Manifesto for Agile Software Development

JACOBS UNIVERSITY

- *"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 et al

Principles of Agile Methods: CIPCS

- Customer involvement
  - customer closely involved
  - ...to provide & prioritise new system requirements + to evaluate iterations
- Incremental delivery
  - software developed in increments
  - customer specifying requirements to be included per increment
- People, not process
  - Recognize + exploit team skills
  - Leave team to develop own ways of working
- Embrace change
  - Expect system requirements to change
  - design system to accommodate these changes
- Maintain simplicity
  - Focus on simplicity in both software and development process
  - Wherever possible, actively work to eliminate complexity

Extreme Programming

- An 'extreme' variation of iterative development based on very small increments
  - New versions may be built several times per day;
  - Increments are delivered to customers ~every 2 weeks;
  - All tests must be run for every build; build only accepted if all tests run successfully
- Relies on
  - constant code improvement
  - user involvement in the development team
  - pairwise programming
- Perhaps best-known & most widely used agile method
  - originally proposed by Kent Beck

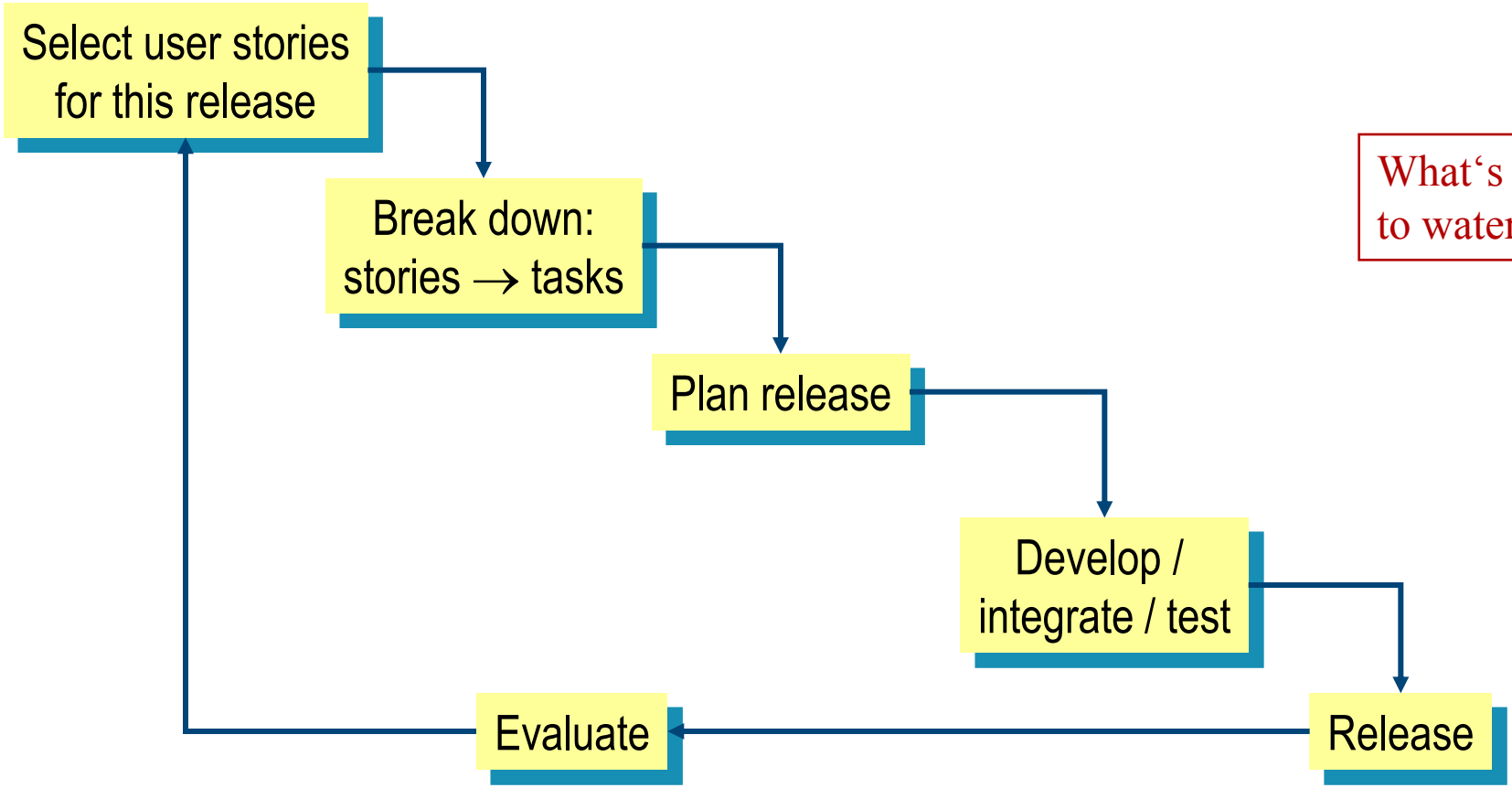
Pair Programming

- programmers work in pairs, sitting together to develop code
  - helps develop common ownership of code
  - spreads knowledge across the team
  - Cross checking of all code
- informal review process
  - each line of code looked at by more than 1 person
- encourages refactoring
  - whole team can benefit
- *Measurements suggest that development productivity with pair programming is similar to that of two people working independently.*

XP and Change

- Conventional wisdom: design for change
  - worth spending time & effort anticipating changes
  - reduces costs later in the life cycle
- XP, however, maintains that this is not worthwhile
  - cannot be reliably anticipated
- Rather, it proposes constant code improvement (refactoring)
  - make changes easier when they have to be implemented

The XP Release Cycle



What's different to waterfall?

Consequences of Extreme Programming

- Incremental planning
  - Stories determined by time available + relative priority
- Small Releases
  - minimal useful set of functionality that provides business value is developed first
- Collective Ownership
  - pairs of developers work on all areas of system
  - no islands of expertise, all developers own all code
  - Anyone can change anything
- Simple Design: Enough design to meet current requirements and no more
- Simple code: Refactoring
- Sustainable pace
  - No large amounts of over-time – net effect often reduced code quality, medium term productivity
- On-site Customer
  - End-user representative available full time
  - Customer member of development team, responsible for bringing system requirements to the team

Agile methods: Appraisal

- Team members may be unsuited to the intense involvement of agile methods
- Developers need to be experienced, not too different in expertise
- can be difficult to keep interest of customers involved in process

Agile methods: Appraisal

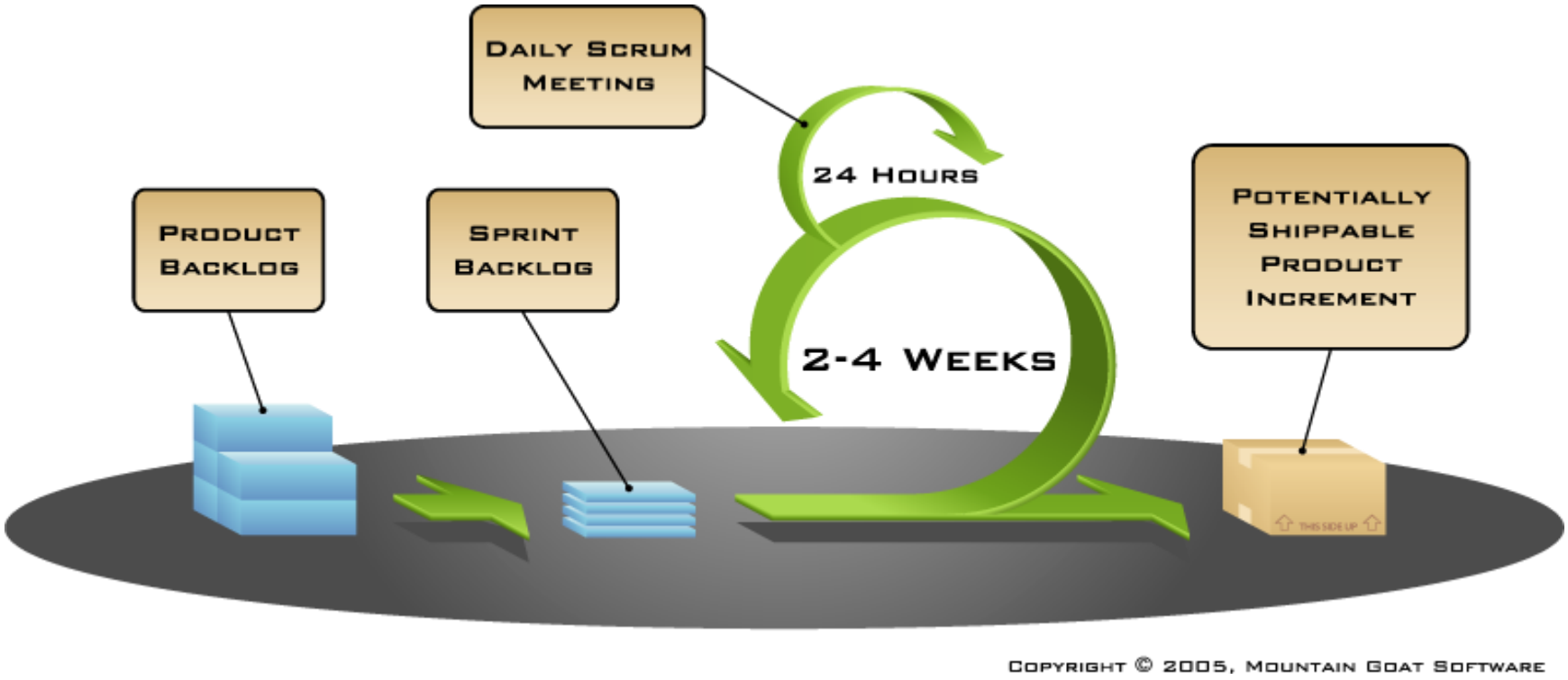
- Maintaining simplicity requires extra work
- Contracts may be a problem
  - Prioritising changes can be difficult when there are multiple stakeholders
  - …as with other approaches to iterative development
- *Agile methods probably best suited to small/medium-sized business systems or PC products = short-term, highly flexible projects*
  - „in future we want to proceed in an agile manner with all such projects. With ROABSO it turned out that after such a long project runtime this was not possible any more. At project start in 2010 the right course could not be set yet.“ [heise.de 2017 (German, translation: me)]

JACOBS UNIVERSITY

Scrum

[PierreSelim / Wikipedia]

Overview of Scrum



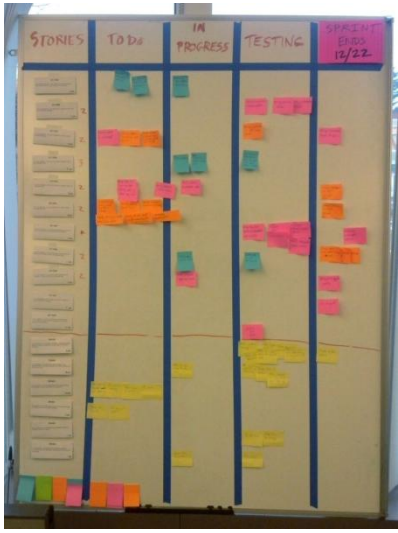
[http://www.mountaingoatsoftware.com/scrum-figures]

Components of Scrum

- Scrum Roles
  - Scrum Master, Scrum Team, Product Owner
- Scrum Process
  - Sprint Planning Meeting
  - Kickoff Meeting
  - Sprint (~Iteration in a Unified Process)
  - Daily Scrum Meeting
  - Sprint Review Meeting
- Scrum Artifacts
  - Product Backlog, Sprint Backlog, Burndown Charts

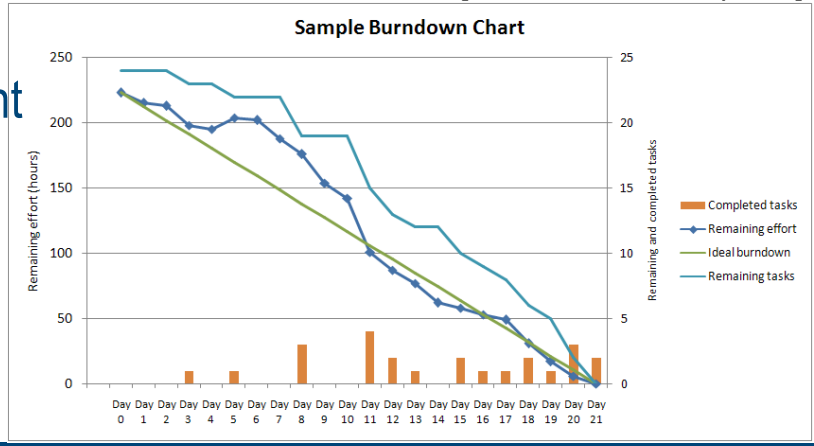
Scrum Artifacts

[Logan Ingalls/ Wikipedia]



- Product Backlog
  - list of requirements: features, bug fixes, non-functional reqs, …
  - prioritized by Product Owner → focus on customer value
  - Feature sequence = delivery sequence
  - product backlog & business value items is responsibility of Product Owner, item size (estimated complexity / effort) determined by development team
- Sprint Backlog
  - items development team must deliver during next sprint
- Burndown Charts
  - public displayed chart showing remaining work in sprint backlog

[Pablo Straub/ Wikipedia]



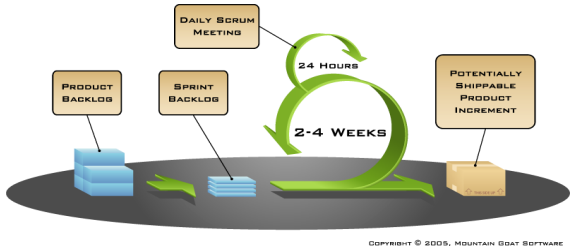
Scrum Roles

- Product Owner
  - Knows what needs to be built & in what sequence this should be done
  - Typically: a product manager
- Scrum Master
  - Represents management to the project
  - Typically: project manager / team leader
  - Responsible for enacting scrum values & practices
  - Main job: remove impediments

- Scrum Team
  - Typically 5-6 people
  - Cross-functional (QA, Programmers, UI Designers, etc.)
  - Members should be full-time
  - Self-organizing
  - Membership can change only between sprints

Scrum Process

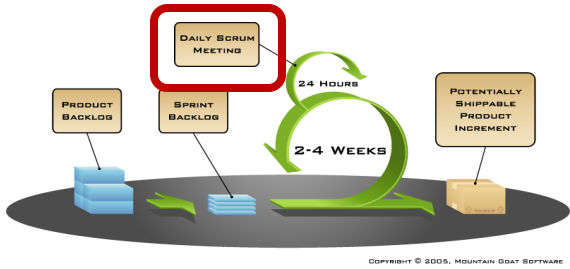
- Project-Kickoff Meeting
  - Collaborative meeting @ beginning of project
  - Participants: Product Owner, Scrum Master
  - Takes 8 hours, consists of 2 parts (“before lunch and after lunch”)
  - Goal: Create Product Backlog
- Sprint Planning Meeting
  - Collaborative meeting @ beginning of each Sprint
  - Participants: Product Owner, Scrum Master, Scrum Team
  - Takes 8 hours, consists of 2 parts (“before lunch and after lunch”)
  - Goal: Create Sprint Backlog



- Sprint
  - = month-long iteration during which product functionality is incremented
  - No outside influence on Scrum team during Sprint
- Daily Scrum Meeting
  - see next

Daily Scrum Meeting

= short (15 min) meeting,
held every day before Team starts working



- Participants: Scrum Master (chairman), Scrum Team
- Every Team member to answer on 3 questions

1. Status: What did I do since the last Scrum meeting?
2. Issues: What is stopping me getting on with the work?
3. Action items: What am I doing until the next Scrum meeting?

Summary

- XP and Scrum are agile methodologies with focus on
  - Empirical process control model
  - Changing requirements are the norm
  - Controlling conflicting interests and needs
- Very simple processes with clearly defined rules
- Self-organizing teams
  - each team member carries lot of responsibility
- No extensive documentation

Scrum: Appraisal

- Advantages
  - Completely developed & tested features in short iterations
  - Simplicity of process
  - Clearly defined rules
  - Increasing productivity
  - Self-organizing
  - team member carry responsibility
  - Improved communication
  - Combination with XP
- Drawbacks
  - “Undisciplined hacking” (no written documentation)
  - Violation of responsibility