

Scrum 101

Agile Foundations

- ▶ In this module, we will be discussing the following topics:
 - ▶ For complex problems, where the process cannot be completely defined (e.g. most software development), we need empirical process control.
 - ▶ Agile software development is not just a set of practices. It is based on the the agile manifesto, values and principles. Understanding these is key to success with agile.
 - ▶ Introducing agile software development often requires a change in mindset and to achieve this a cultural change is often necessary.

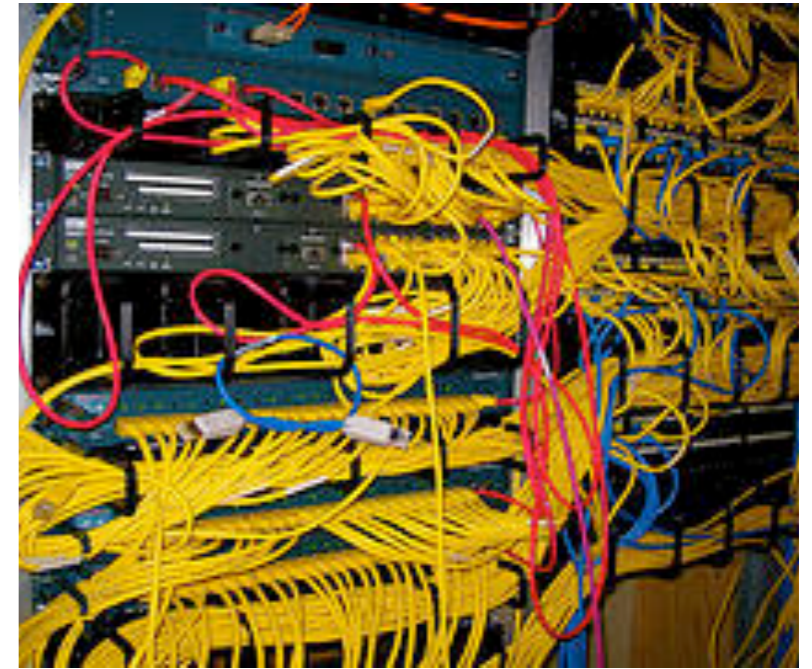
"The **wiring on an aircraft** is complicated. To figure out where everything goes would take a long time. But if you studied it for long enough, you could know with (near) certainty what each electrical circuit does and how to control it. The system is ultimately knowable. If understanding it is important, the effort to study it and make a detailed diagram of it would be worthwhile.

complicated = not simple, but ultimately knowable.

Now, put a **crew and passengers in that aircraft** and try to figure out what will happen on the flight. Suddenly we go from complicated to complex. You could study the lives of all these people for years, but you could never know all there is to know about how they will interact. You could make some guesses, but you can never know for sure. And the effort to study all the elements in more and more detail will never give you that certainty.

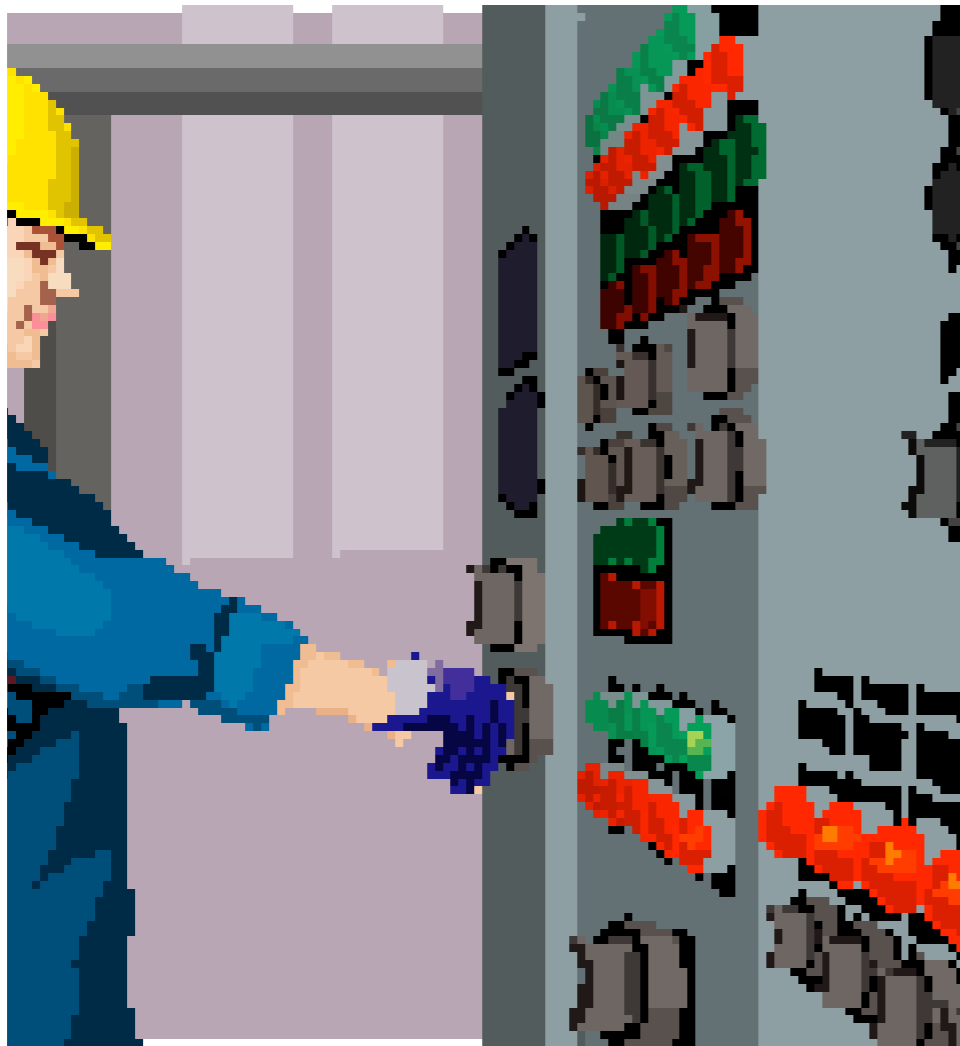
complex = not simple and never fully knowable.
Just too many variables interact.

source: Robert Paternsons Weblog
http://smartpei.typepad.com/robert_paternsons_weblog/2006/11/more_on_complex.html



Complicated
and
Complex

Different Process Types



Defined Process: may be complicated
Follow rules, avoid deviations



Empirical Process: complex
inspect and adapt constantly

Process Type Characteristics

- ▶ Defined processes
 - ▶ We know all premises
 - ▶ We can give exact instructions for each action
 - ▶ May be complicated, but ultimately knowable
- ▶ Empirical processes
 - ▶ Environment and prerequisites are not defined completely
 - ▶ Requirements change over time
 - ▶ The knowledge about the best approach is incomplete
 - ▶ The system is complex, i.e. not simple and never fully knowable

Software Development

- ▶ Software development is (most of the time) an empirical process:
 - ▶ The environment and prerequisites are incompletely defined
 - ▶ Requirements change over time
 - ▶ The knowledge about the best approach is incomplete
 - ▶ The system is complex
- ▶ As a result, success depends on:
 - ▶ Getting early and frequent feedback for the system under development, in terms of requirements and approach - e.g. by demonstrating and evaluating running increments early and often.
 - ▶ Making it easy for the participants to react to this feedback and adapt the requirements and approach - e.g. by making it easier for the participants to steer development.
 - ▶ Effective communication between all of those involved in developing the system so that the business and technical knowledge is available to those who need it. The most effective means of communication is often the spoken word.
- ▶ The agile principles are rooted in these insights.

Environment for Empirical Process Control

- ▶ Solving complex problems needs constant adaption and requires:
 - ▶ Creativity
 - ▶ Initiative
 - ▶ Individuals and teams that learn
- ▶ So that this can flourish, a culture is needed which values:
 - ▶ Trust: not trying to place blame when errors occur and appreciating the learning opportunities
 - ▶ Respect: people are not resources
- ▶ These are codified in the Agile Manifesto and the accompanying principles:
 - ▶ Written in 2001 by 17 prominent figures in the field of software development

Values, Principles, Activities and Practices

The Agile Manifesto

Activities

Practices

Principles

Values

The Agile Manifesto

Individuals and interactions

over

Processes and tools

Running software

over

Comprehensive
documentation

Customer collaboration

over

Contract negotiation

Responding to change

over

Following a plan

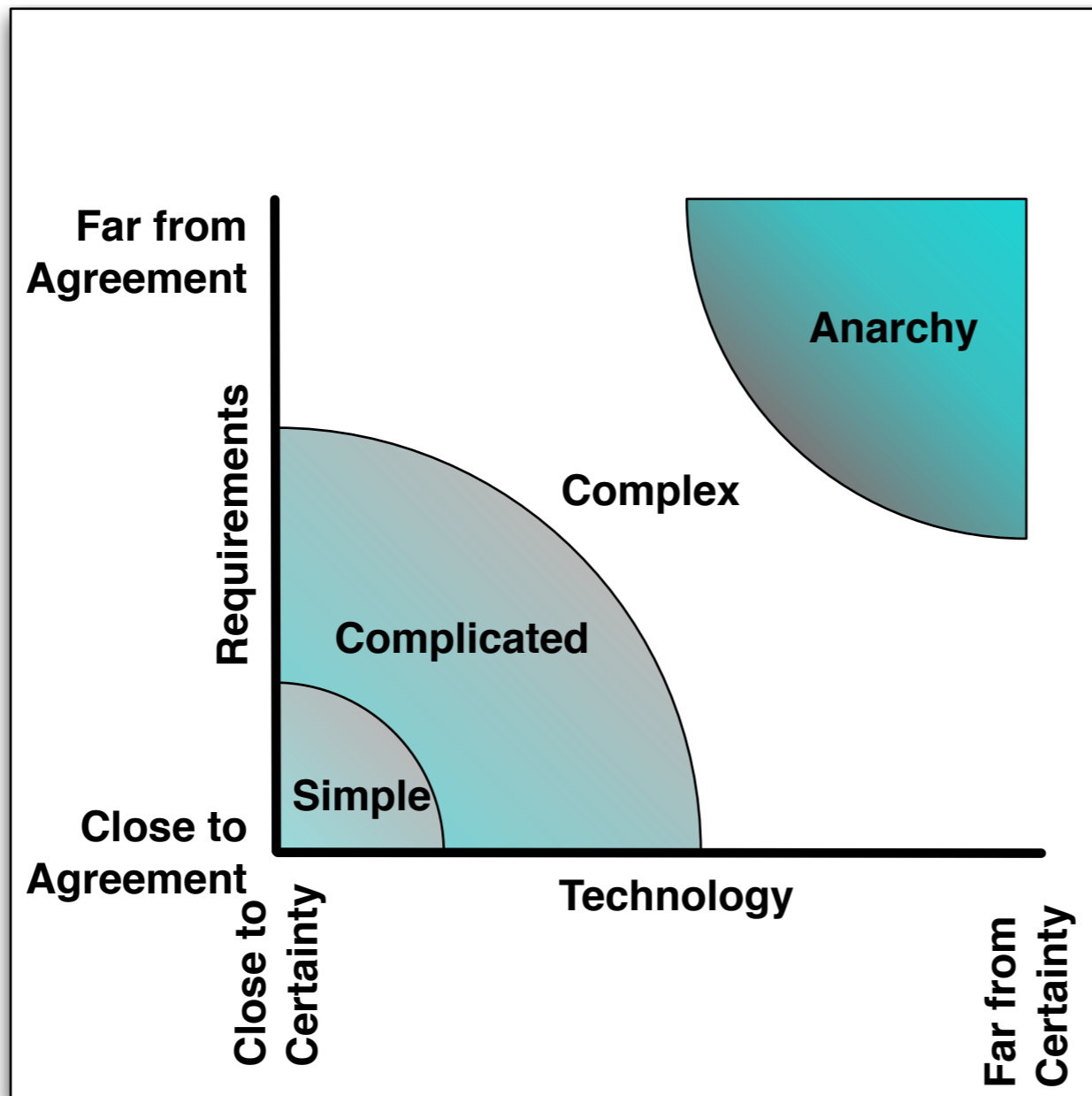
That is, while there is value in the items on the right, we value the items on the left more

Scrum Values

- ▶ Commitment
- ▶ Focus
- ▶ Openness
- ▶ Respect
- ▶ Courage

The Agile Manifesto: Agile Principles

- ▶ Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- ▶ Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
- ▶ Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- ▶ Business people and developers must work together daily throughout the project.
- ▶ Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
- ▶ The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
- ▶ Working software is the primary measure of progress.
- ▶ Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
- ▶ Continuous attention to technical excellence and good design enhances agility.
- ▶ Simplicity--the art of maximizing the amount of work not done--is essential.
- ▶ The best architectures, requirements, and designs emerge from self-organizing teams.
- ▶ At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.



Source: *Strategic Management and Organizational Dynamics* by Ralph Stacey in *Agile Software Development with Scrum* by Ken Schwaber and Mike Beedle.

Agile Development is for Complex Problems

Scrum Roots

▶ History

- ▶ Lean production - values and organization
- ▶ Concurrent engineering - self organizing Teams
- ▶ Developed in the early 1990s, first published 1995 by Ken Schwaber
- ▶ Starting point for other agile methods, e.g. eXtreme Programming (XP)
- ▶ Left its mark in the Agile Manifesto

Scrum Flow

The Essentials of Scrum

Scrum in 5 Minutes

- ▶ The **Product Owner** is responsible for creating and prioritizing a list of open features. This is the **Product Backlog**, a complete but dynamic to-do list for the project.
- ▶ Before each **Sprint**, the team decides in a **Sprint Planning** meeting, how many of the highest prioritized features they can deliver during the Sprint. The team decides what tasks are necessary and enters them into the **Sprint Backlog**.
- ▶ During the Sprint, the team synchronizes in a **Daily Scrum** meeting and follow progress in the **burn down chart**.
- ▶ The **ScrumMaster** coaches the team, removes impediments and ensures that the team is able to work effectively.
- ▶ During the Sprint, valuable functionality is developed and a **potentially shippable product increment** created, which is demonstrated by the team during a **Sprint Review** meeting.



By Clark & Vizdos

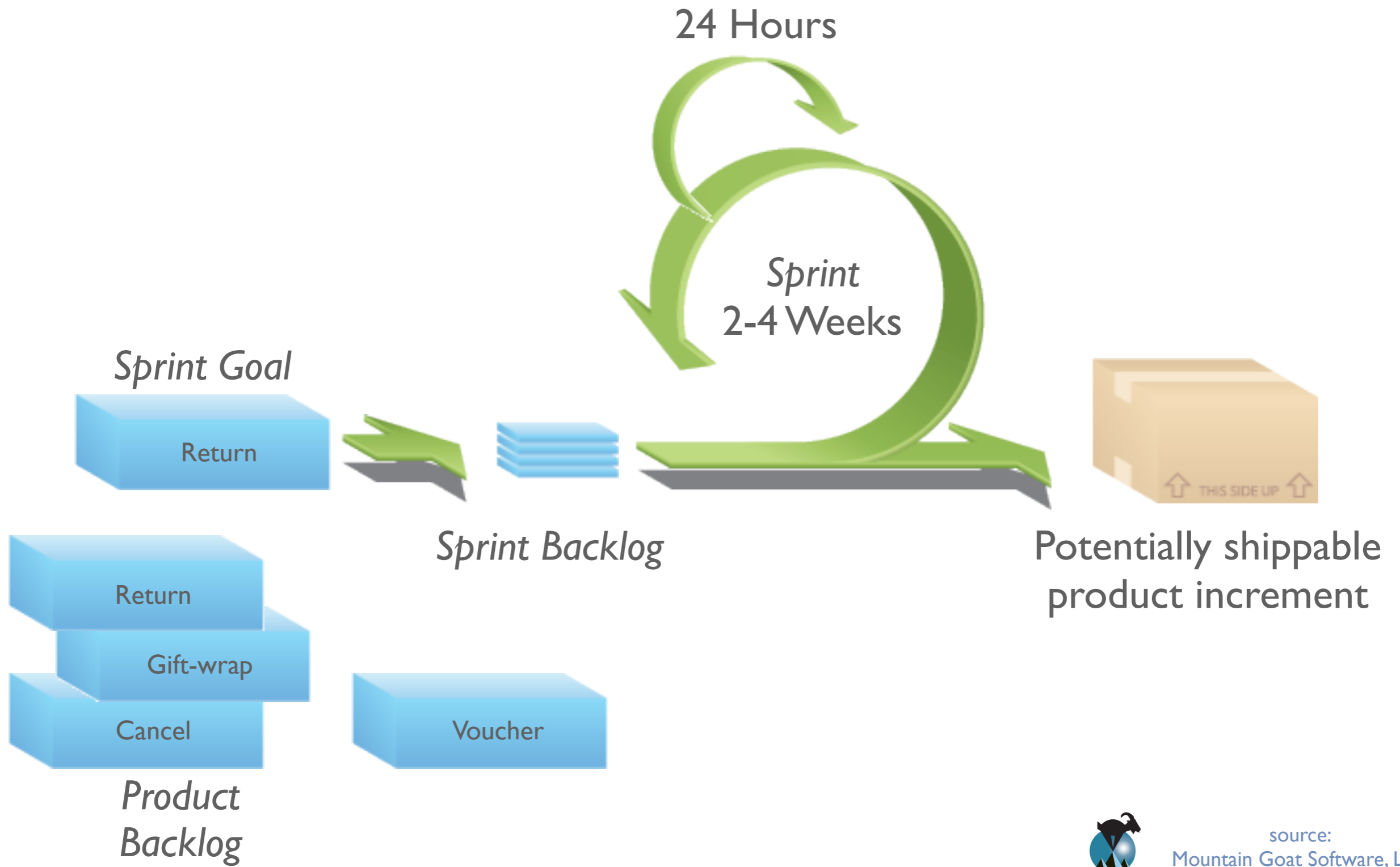
© 2006 implementingscrum.com

Pigs and Chickens

The Essentials

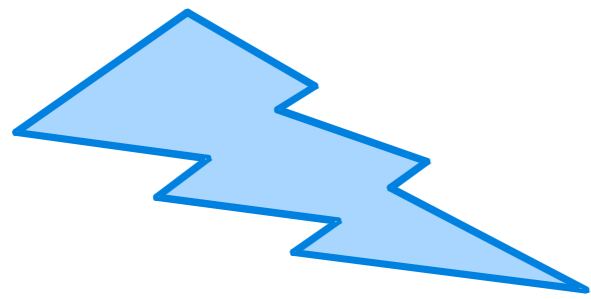
- ▶ Sprint
 - ▶ An iteration with a fixed duration
- ▶ 3 Meetings
 - ▶ Sprint Planning
 - ▶ Sprint Review
 - ▶ Daily Scrum
- ▶ 3 Documents
 - ▶ Product Backlog
 - ▶ Sprint Backlog
 - ▶ Sprint Burn Down Chart
- ▶ 3 Roles
 - ▶ Product Owner
 - ▶ ScrumMaster
 - ▶ Team

Scrum Flow

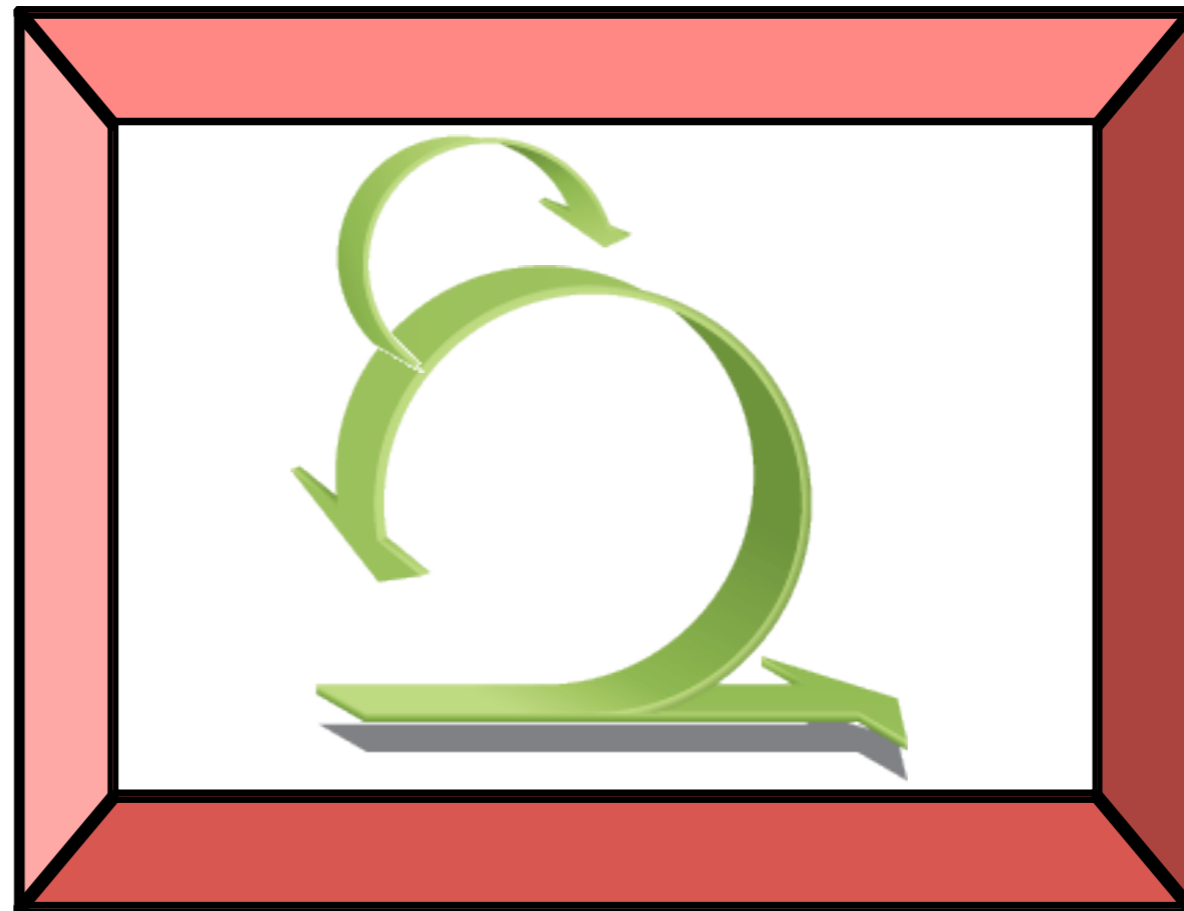


source:
Mountain Goat Software, LLC

No Changes during a Sprint



New Requirements

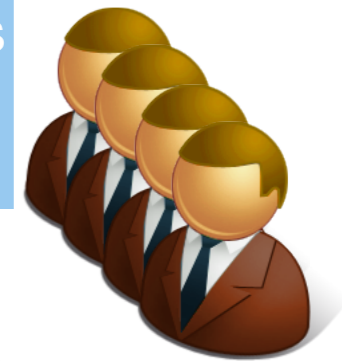


Scrum Roles

Team
Self-managing
responsible for realizing the
product increment



Other stake-holders
coordinated by the
Product Owner



ScrumMaster
removes impediments
responsible for process



Product Owner
defines the product
responsible for the results



The Product Owner

- ▶ Defines the product features, answers questions on required functionality
- ▶ Prioritizes features depending on their business value, consolidates requirements from other stake-holders
- ▶ Decides release dates and decides the content of the product
- ▶ Updates features and their priorities before every Sprint if required
- ▶ Accepts or rejects functionality
- ▶ Responsible for the financial result of the project (ROI)
- ▶ Has a more active role than a traditional project manager

The Product Owner decides **what** will be delivered

The Team

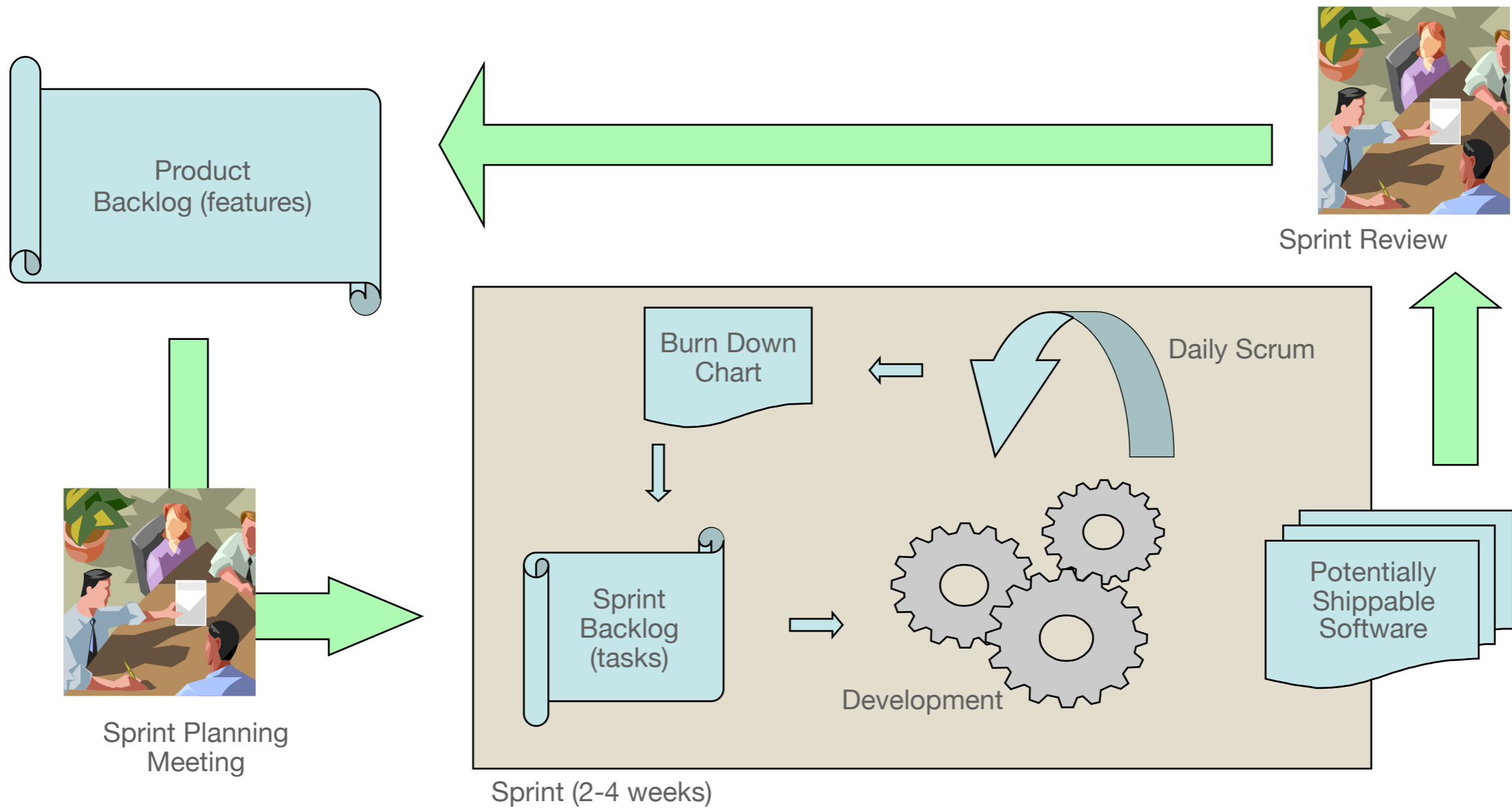
- ▶ Commits to the delivery of the highest-priority features as defined by the Product Owner
- ▶ Estimates how much they can deliver
- ▶ Organize themselves: Ideally multifunctional and no fixed roles (sometimes not avoidable)
- ▶ Typically 5-9 people
- ▶ All necessary skills should be represented in the team: developers, UI designers, testers etc.
- ▶ Members should be full time
 - ▶ Some exceptions (e.g. system administrators)
 - ▶ Membership should only change between Sprints

The team decides **how much** can be delivered

The ScrumMaster

- ▶ Ensures the observance of the Scrum values and practices
- ▶ Removes impediments
- ▶ Ensures that the team is able to work productively
- ▶ Supports collaboration between all roles and team members
- ▶ Protects the team from outside disturbances

Scrum Meetings



Sprint Planning

- ▶ Before the meeting
 - ▶ The team, with the support of the product owner, estimates the relative size of the product backlog items
 - ▶ The product owner prioritizes the product backlog
- ▶ During the meeting
 - ▶ The team decides on its available capacity
 - ▶ The product owner and team talk the product backlog items through
 - ▶ The team takes the highest priority backlog items, breaks it down into engineering tasks and estimates the effort required for each task
 - ▶ This is repeated until the available capacity is used up
- ▶ After the meeting
 - ▶ The ScrumMaster creates the sprint backlog

Daily Scrum

- ▶ Goal of the Daily Scrum
 - ▶ Each team member should know what the other team members are doing
 - ▶ To achieve this, each team member must report about his/her work in enough detail to make it transparent to the others
- ▶ How is this achieved
 - ▶ Each team member answers these three questions:
 - ▶ What did I do since the last meeting?
 - ▶ What do I plan to do until the next meeting?
 - ▶ What impediments or problems do I have?
- ▶ After the Daily Scrum
 - ▶ The ScrumMaster works on removing the impediments
 - ▶ Often there is a follow-up meeting after the Daily Scrum (only for those who need or want to attend) to discuss open functionality questions, technical problems etc.

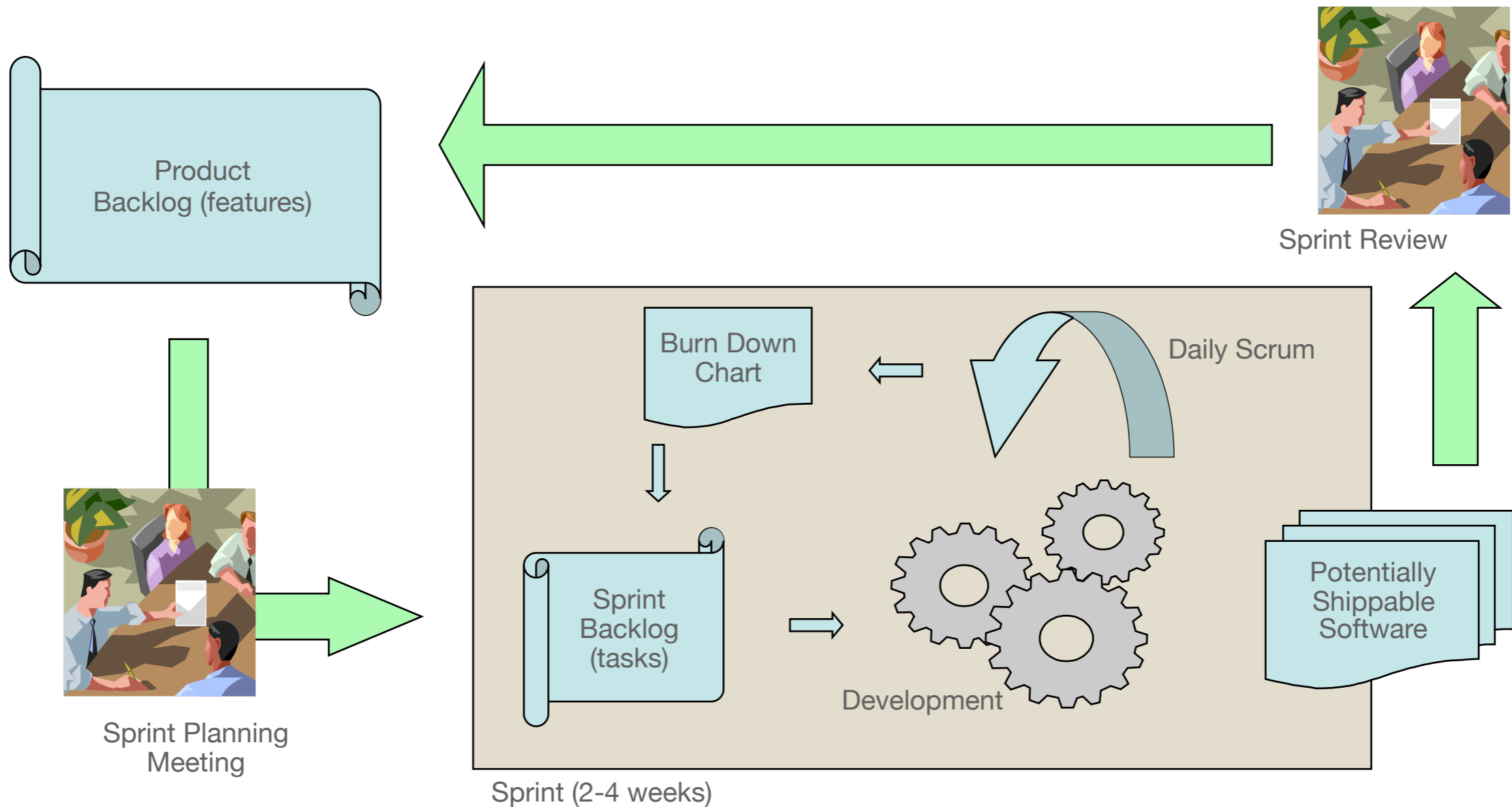
Sprint Review

- ▶ Before the meeting
 - ▶ The ScrumMaster notes which backlog items have been completely finished and calculates the Velocity from these stories only
 - ▶ The team prepares a demonstration of the new product increment. Note only completed functionality may be demonstrated
- ▶ During the meeting
 - ▶ The team demonstrates the completed and tested features
 - ▶ The Product Owner accepts or rejects the results
- ▶ After the meeting
 - ▶ The Product Owner and ScrumMaster update the Product Backlog and prepare for the next sprint planning

Retrospective

- ▶ Goal of a retrospective
 - ▶ For the team to discover ways to improve its work
- ▶ Who takes part in a retrospective
 - ▶ The development team, ScrumMaster and product owner
 - ▶ Others only if explicitly invited by the team
- ▶ What a retrospective is not for:
 - ▶ performance evaluation
 - ▶ criticism of individuals or for assigning blame

Scrum Documents



Product Backlog

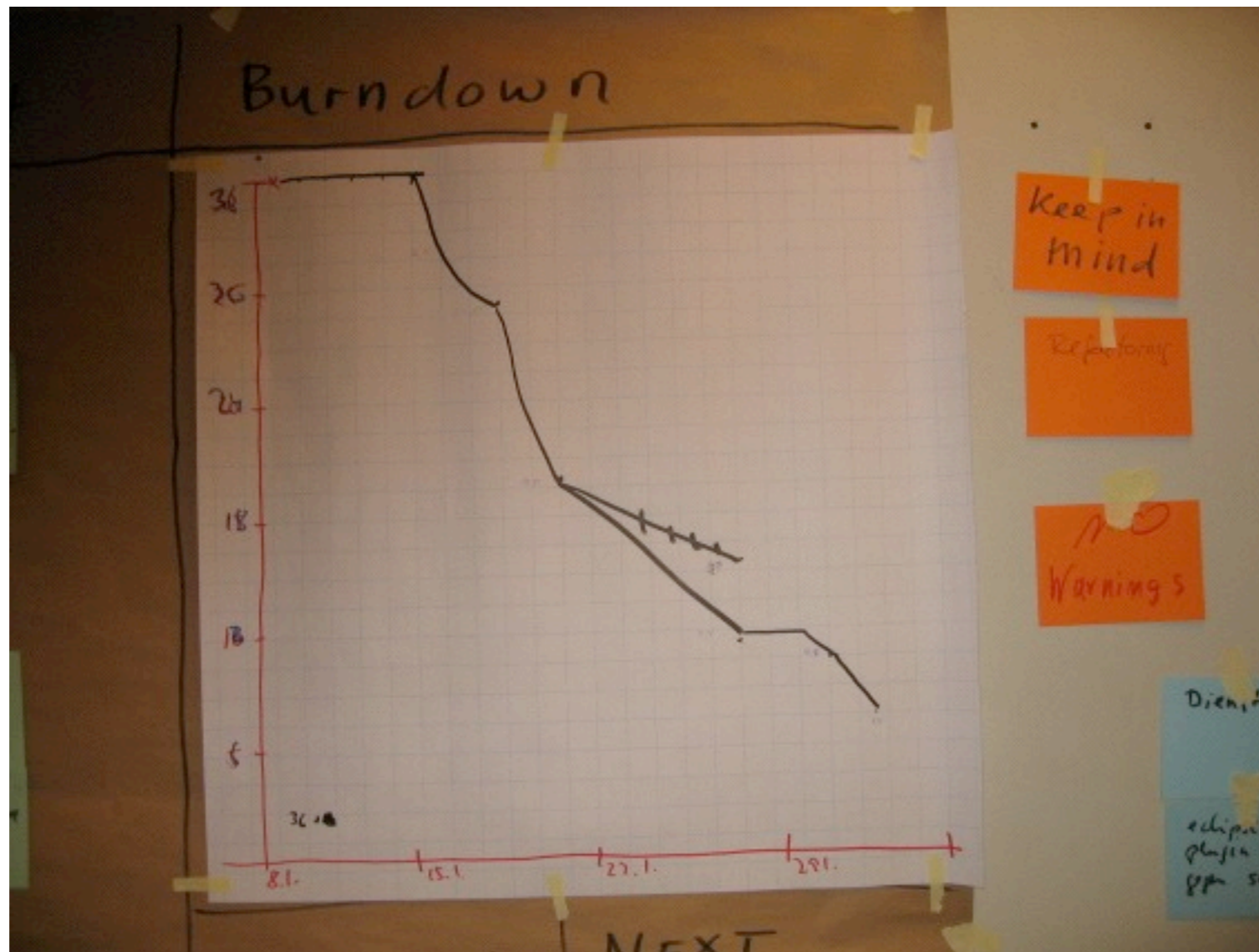
- ▶ Contains requirements in the form of product backlog items
- ▶ Backlog items are often in the form of User Stories
- ▶ Owned and maintained by the product owner
- ▶ The development team estimates the relative size of backlog items

Sprint Backlog

- ▶ Owned by the development team
- ▶ Maintained by the development team or the ScrumMaster on their behalf
- ▶ Contains tasks and the estimated initial and remaining effort for the product backlog items that have been committed to for the sprint
- ▶ Updated by the development team daily so it is represents the remaining effort required to realize the committed product backlog items

Sprint Burn Down Chart

- ▶ Visualizes the progress in the sprint



Task Board with Burn Down Chart



Vielen Dank



scrumcenter.com

christoph.mathis@scrumcenter.com

simon.roberts@scrumcenter.com