

# Scrum

---



**SoftEng**  
<http://softeng.polito.it>

Version 1.3.0

© Marco Torchiano, Antonio Vetrò – September 28, 2021



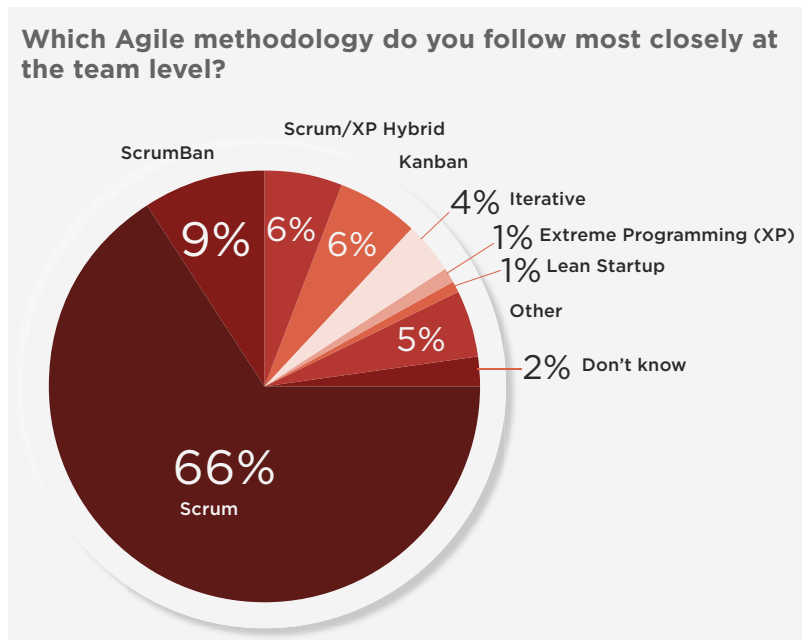
## Definition

---

- Scrum:
  - An ordered formation of players, used to restart play in rugby
  - A team-based framework to develop complex systems and products.

# Diffusion

---



Source: 15<sup>th</sup> State of Agile Report, available at: <https://explore.digital.ai/state-of-agile/15th-state-of-agile-report>

## Why «scrum»

---



YEAR: 1984

137

---

The new new  
product development  
game

*Stop running  
the relay race and  
take up rugby*

---

*Hiroataka Takeuchi and  
Ikujiro Nonaka*

---

<https://hbsp.harvard.edu/product/86116-PDF-ENG?E=48834&R=86116-PDF-ENG>

# The new new product development game

---

## Abstract (1 / 2)

*"In today's fast-paced, fiercely competitive world of commercial new product development, speed and flexibility are essential. Companies are increasingly realizing that **the old, sequential approach to developing new products simply won't get the job done**. Instead, companies in Japan and the United States are using a **holistic method—as in rugby, the ball gets passed within the team as it moves as a unit up the field.** "*

# The new new product development game

---

## Abstract (2 / 2)

*"This holistic approach has six characteristics: **built-in instability, self-organizing project teams, overlapping development phases, "multilearning", subtle control, and organizational transfer of learning**. The six pieces fit together like a jigsaw puzzle, forming a fast and flexible process for new product development. Just as important, the new approach can act as a change agent: it is a vehicle for introducing creative, market-driven ideas and processes into an old, rigid organization"*

# The new new product development game

---

*"[...] a holistic or "rugby" approach – where a team tries to go the distance as a unit, passing the ball back and forth – may better serve today's competitive requirements. "*



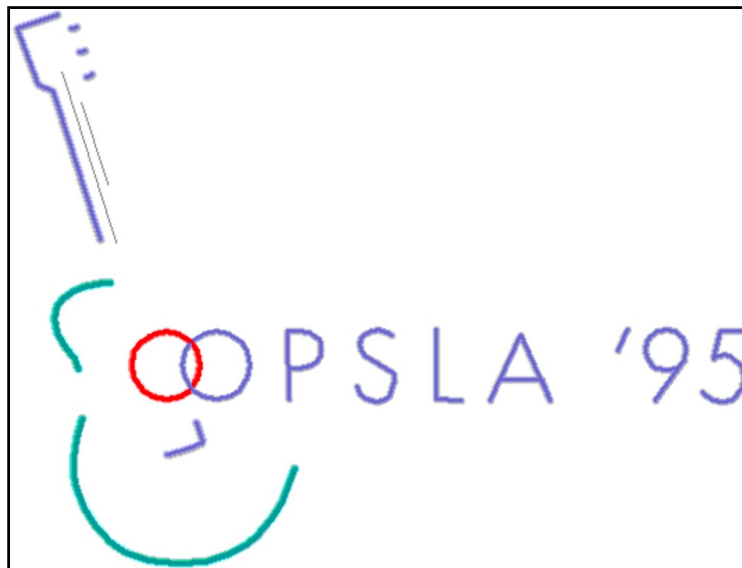
*Image source: Wikipedia*

---

**SoftEng**  
http://softeng.polito.it

## Scrum in sw development

---



***Workshop Report: Business Object Design and Implementation***

<http://jeffsutherland.com/oopsla/oo95summary.html>

---

**SoftEng**  
http://softeng.polito.it

# Scrum in sw development

---

## SCRUM Development Process

**Ken Schwaber**

*Advanced Development Methods*  
131 Middlesex Turnpike Burlington, MA 01803  
email virman@aol.com Fax: (617) 272-0555

---

**ABSTRACT.** *The stated, accepted philosophy for systems development is that the development process is a well understood approach that can be planned, estimated, and successfully completed. This has proven incorrect in practice. SCRUM assumes that the systems development process is an unpredictable, complicated process that can only be roughly described as an overall progression. SCRUM defines the systems development process as a loose set of activities that combines known, workable tools and techniques with the best that a development team can devise to build systems. Since these activities are loose, controls to manage the process and inherent risk are used. SCRUM is an enhancement of the commonly used iterativ/incremental object-oriented development cycle.*

---

**SoftEng**  
http://softeng.polito.it

Schwaber, K. (1997). Scrum development process. In *Business object design and implementation* (pp. 117–134). Springer, London.

## Scrum creators

---



*Jeff Sutherland and Ken Schwaber*

Picture from their article: «The Scrum Guide – The Definitive Guide to Scrum: The Rules of the Game» Nov. 2017

---

**SoftEng**  
http://softeng.polito.it

# Outline

---

- Principles
- Activities
- Roles
- Artifacts

---

## Principles

# Agile Development Principles

---

- Test as you go
- Deliver product early and often
  - Feedback
- Document as you go, only as required
- Build cross-functional teams

## Theoretical foundations

---

- Empirical process control theory
- Iterative and incremental approach
  - to optimize predictability
  - to control risk

# Scrum pillars

---

- Transparency
  - Process visible to whom is responsible
- Inspection
  - On artifacts and goal to detect variances
- Adaptation
  - To meet the goals

# Adherence to pillars

---

- Four formal events:
  - Sprint planning
  - Daily scrum
  - Sprint review
  - Sprint retrospective



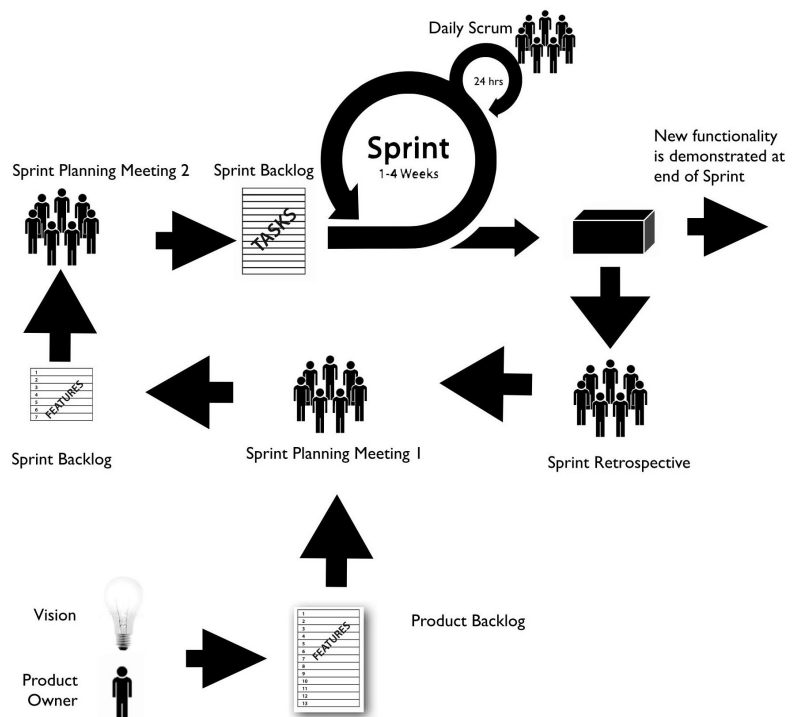
# Scrum values

---

- Commitment
- Courage
- Focus
- Openness
- Respect

# Sprint planning I and II

---



Sutherland,  
Scrum Handbook, 2010

---

# Roles

## Scrum Roles

---

- Product Owner
- Scrum Master
- Team member

# Product Owner

---

- Controls the priority order of items in the team's backlog
  - Directs team toward most valuable work
  - Work closely with the stakeholders in order to deliver the maximum business value
    - What needs to be built and when

# Product Owner

---

- Makes sure that the needs of the customers and end-users are understood by the team
  - Requirements
  - Collaboration
    - Available to the team
- Keeper of the product vision
  - Who the product is built for
  - Why they need it
  - How they will use it

# Product Owner

---

- Holds the vision for the product
- Represents the business
- Represents the customers
- Owns the product backlog
- Prioritizes stories
- Creates acceptance criteria for stories
- Is available to answer team members' questions

# Scrum Master

---

- Acts as a coach
  - His goal is to produce a self-organizing team
  - Is a facilitator, NOT a boss
    - Good shepherd
    - Champion
    - Guardian
    - Scrum expert

# Scrum Master

---

- Adjust to the team experience
  - As the team becomes self-managing the master steps back
- Remove impediments for the team
  - External
    - Escalates issues
    - Help solving resource issues (e.g. hw)
  - Internal
    - Helps the team see the problem
    - Encourages the team find a solution

# Scrum Master

---

- Scrum expert and advisor
- Coach
- Impediment *bulldozer*
- Facilitator

# Team Member

---

- The team is self-organizing
  - Tools
  - Techniques
  - Task assignment
- Team member estimate the effort for the features
- Team size: typically 5 to 9
  - Enough skill variety
  - Limited communication overhead

# Team Member

---

- Cooperate to achieve the goal
- Willing to work outside his comfort zone when the team needs it
- The Scrum Team
  - P.Owner + S.Master + Team Members
- The Team
  - Scrum team except P.Owner and S.Master

# Pigs and Chickens

---

- A pig and a chicken are walking down a road.
- The chicken looks at the pig and says: “Hey, why don’t we open a restaurant?”
- The pig looks back at the chicken and says: “Good idea, what do you want to call it?”
- The chicken thinks about it and says: “Why don’t we call it ‘Ham and Eggs’?”
- “I don’t think so,” says the pig, “I’d be committed, but you’d only be involved.”

# Pigs and Chickens

---

- A pig and a chicken are walking down a road.
  - The chicken looks at the pig and says: “Hey, why don’t we open a restaurant?”
  - The pig looks back at the chicken and says: “Good idea, what do you want to call it?”
  - The chicken thinks about it and says: “Why don’t we call it ‘Ham and Eggs’?”
  - “I don’t think so,” says the pig, “I’d be committed, but you’d only be involved.”
- Product Owner, Scrum Master, and Team Members are pigs; all other interested parties are chickens

---

# Events

# Sprint

---

- The basic iteration in the Scrum approach
- Produces a piece of working software to be demonstrated and reviewed at the end of the sprint
- Length
  - From 1 to 4 weeks long



# Sprint organization (1 week)

Monday	Tuesday	Wednesday	Thursday	Friday
Sprint Planning (2h)	Scrum (15m)	Scrum (15m)	Scrum (15m)	Scrum (15m)
		Story Time (1h)		Sprint Review (1/2 h)
				Retrospective (1h)

## Sprint Planning

Monday	Tuesday	Wednesday	Thursday	Friday

- 1. What will we do?
  - Set of committed stories
  - Product owner proposes story
  - Team members decide whether commit
    - Team’s velocity / Yesterday’s weather
- 2. How will we do it?
  - Decompose stories into tasks
    - May trigger renegotiation of stories
  - Max effort per task: half-day

# Sprint Backlog

---

Monday	Tuesday	Wednesday	Thursday	Friday
■		■		■ ■

- List of stories
- Related tasks
  
- Estimation:
  - Task hours
  - Task points
  - Task count

## Effort Estimation

---

- Essential to define a predictable schedule
- Relative vs. Absolute estimates
  - Former are much easier
  - Stories are sized in term of “points”
    - Using Fibonacci series
  - After a sprint is complete an estimate of stories per sprint i.e. “velocity” is done
    - Velocity is NOT a performance measure


# Effort Estimation

---

What if planning for a piece of software was like shopping?

## Analogy to shopping

---

- The items
  - The prices
  - The budget
  - The constraints
- 
- Stories
  - Estimates
  - Progress
  - Business and/or technology constraints

# Team Estimation Game

---

- Part 1 Order stories
  - In turn, every team member
    - Place a new story card on board
    - Move a story card (keeping other sorted)

← Smaller

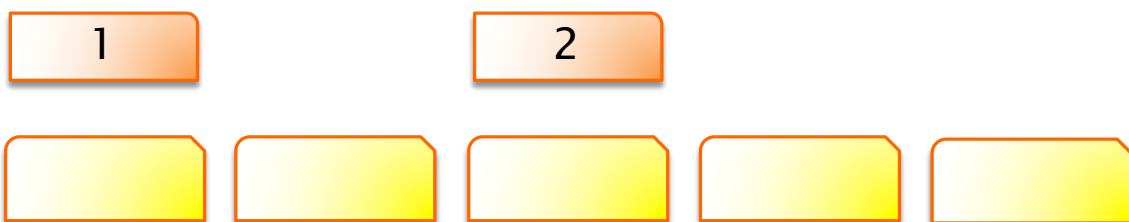
Larger →



# Team Estimation Game

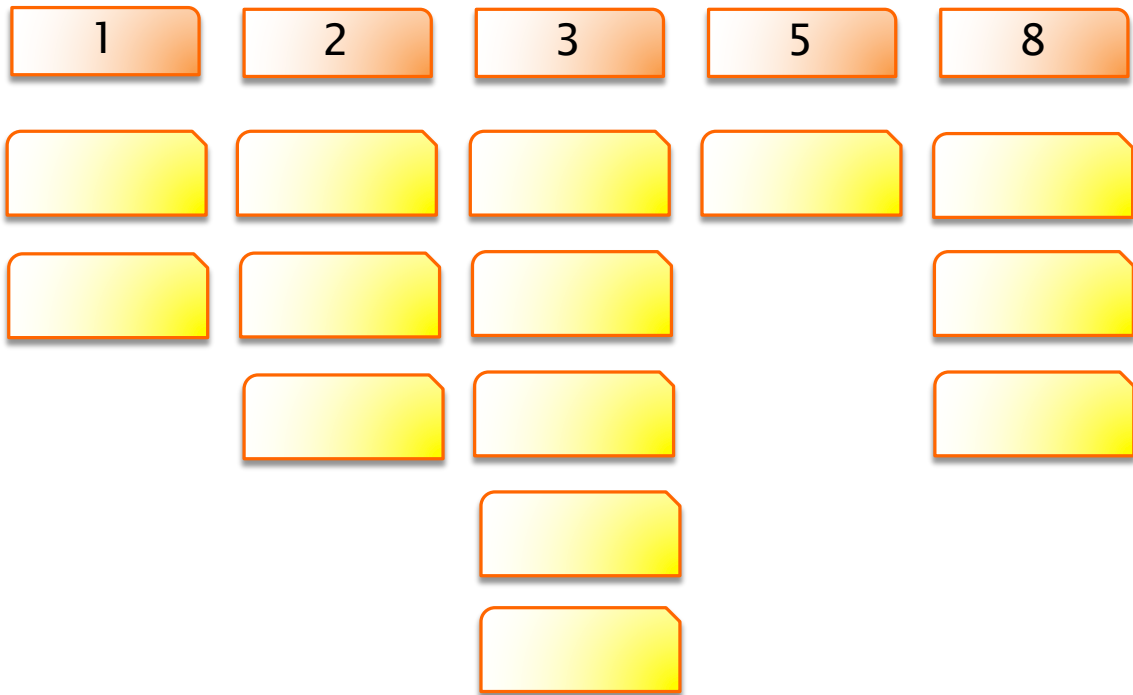
---

- Part 2: Gauge
  - In turn, every team member
    - Place a Fibonacci card above a story card
    - Move Fibonacci card
    - Move a story card



# Team Estimation Game

---



## Estimation Poker

---

- Every team member pick a card and show it at the same time as the others
- High and low discuss the rationale
- Consensus is achieved with feedback from the product owner



# Estimation Poker

- Every team member pick a card and show it at the same time as the others
- Then each one explain the rationale
- Consensus is achieved with feedback from the product owner



## «Estimation wall»



Rupert Dürre,  
op. cit..

# It's effort, not complexity

---

*Suppose a team consists of a little kid and a brain surgeon. Their product backlog includes two items: lick 1,000 stamps and perform a simple brain surgery -- snip and done.*

*These items are chosen to presumably take the same amount of time. If you disagree, simply adjust the number of stamps in the example.*

*Despite their vastly different complexities, the two items should be given the same number of story points because each is expected to take the same amount of time. In this carefully chosen example, the volume of work (licking 1,000 stamps and one snip to some part of the brain) and the complexity of that work combine such that each will take the same amount of time*

(Mike Cohn)

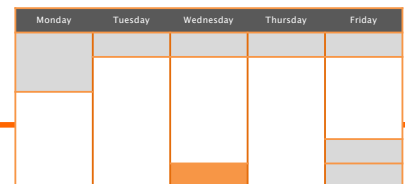
<https://www.mountaingoatssoftware.com/blog/its-effort-not-complexity>

---

**SoftEng**  
http://softeng.polito.it

## Story time

---



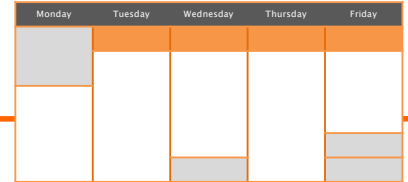
- Backlog refinement / grooming
  - Review upcoming stories
  - Sizing (estimating) future stories
  - Clarification of requirements
  - Splitting stories
- Goal:
  - start next sprint with a set of small, well-understood, sized stories (PBIs)

---

**SoftEng**  
http://softeng.polito.it

# Daily Scrum

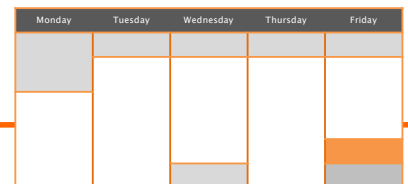
---



- Daily
  - Any time suitable to the team
- Small
  - Only members of the dev team
- Brief
  - Keep updated, standing, 15 min max
- Pointed
  - What has been done
  - What will be done
  - Current obstacles

# Sprint review

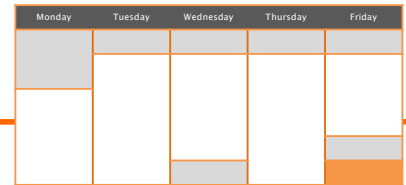
---



- Goal:
  - Show off some piece of working software to stakeholders
  - Report on incomplete stories
  - Transparency
  - Record reactions of stakeholders
    - Basis for Product Owner future decisions
    - No planning at this time



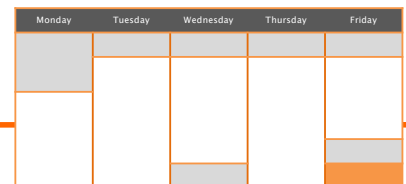
# Retrospective



- Set the stage
- Gather data
  - What happened?
  - Time-line and artifacts
- Generate insights
  - Understand causes, no finger pointing
- Decide what to do
  - One team-improvement task for next sprint, under the team's control
- Close
  - Appreciation exercise

Esther Derby and Diana Larsen,  
"Agile Retrospectives: Making Good Teams Great"

# Retrospective



- Focus on lessons learned
  - How they can be applied to next sprint
- Goal:
  - Identify one/two things to improve
  - Define the relative action plan

*"At the end of a project everyone knows so much more. Naturally we will discover decisions and actions we wish we could do over. This is wisdom to be celebrated..."*

Norm Kerth,  
"Project Retrospectives: A Handbook for Team Reviews"


# Abnormal Sprint Termination

---

- Product Owner may call for AST
  - It is a business decision
  - Typically something happened outside the team
  - Rarely the team may ask PO to call AST
- Demo (if possible) and Retrospective are held as in normal sprint

## Release Planning

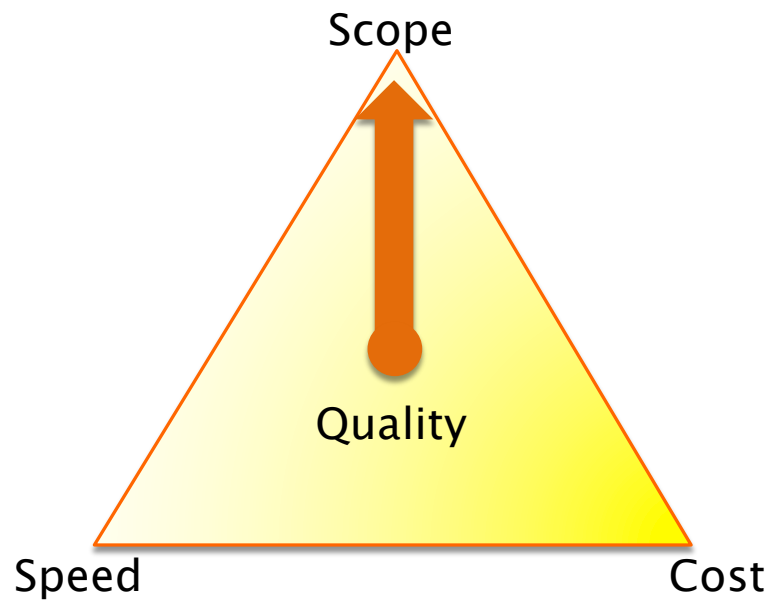
---

- Fixed Scope
- Fixed Date
- Fixed Scope & Fixed Date
  - Scrum is not a 

*Adding manpower to a late software project makes it later*

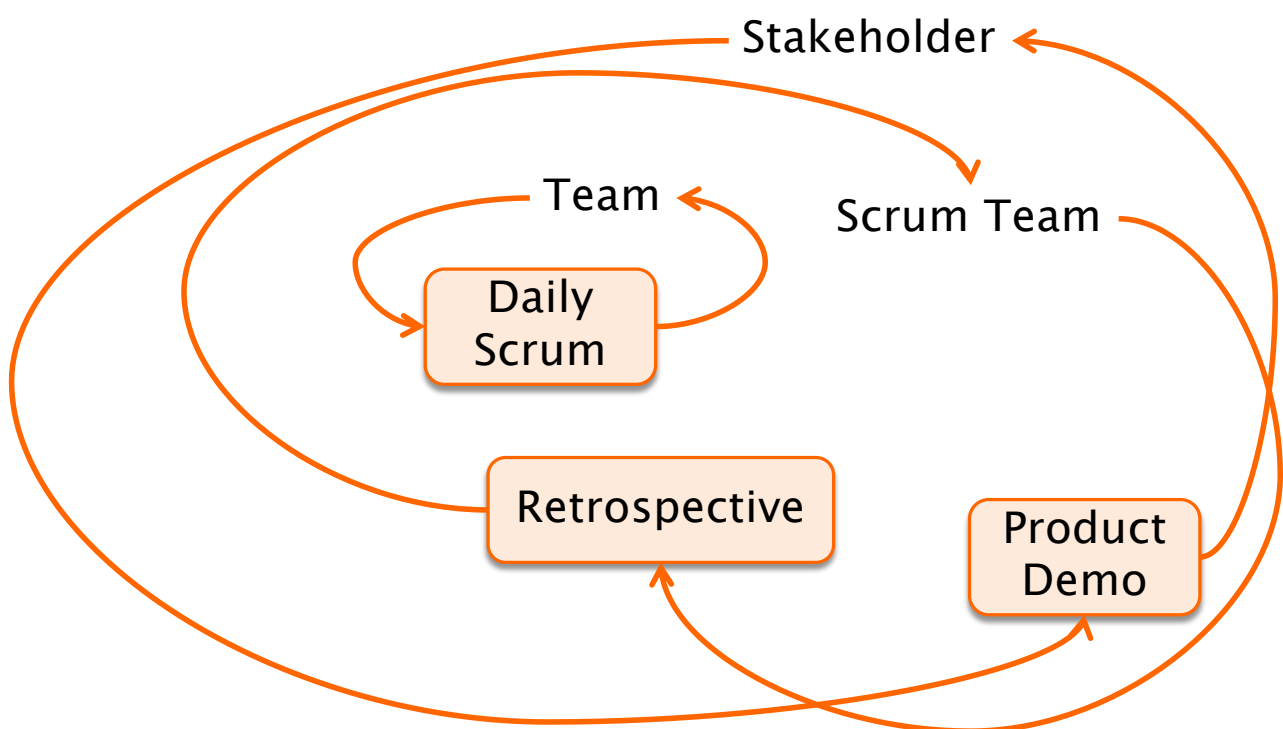
# Iron Triangle

---



# Inspect and Adapt

---



---

# Artifacts

## Product Backlog

---

- Anything that *will* consume team resources
- Ever changing
- Strict ordering of the **stories**
- Property of the Product Owner
- Can be a spreadsheet or a wall

# Example

Story	Story points
As an <i>unregistered user</i> I want to <i>create a new account</i> So that <i>I can buy items</i>	3
As a <i>customer</i> I want to <i>use my Card</i> So that <i>I can buy the items</i>	8
As a <i>user</i> I want to <i>add items</i> to my <i>wish list</i>	5

TOT 16

Forecasted velocity: 12 pt per sprint  
Sprint length: 2 weeks

## Product Backlog

- **P**roduct **B**acklog **I**tem / Story
  - Independent
  - Negotiable
  - Valuable
  - Estimable
  - Small
  - Testable

“INVEST”

# Properties of a good Product Backlog

---

- “DEEP”
  - Detailed appropriately
  - Estimated
  - Emergent
  - Prioritized

## Story

---

- Essential description of a desired functionality:
  - As a *<actor type>*
  - I want *<to do something>*
  - So that *<some value is created>*
- Acceptance criteria
  - Written by PO
  - Easy to turn into (automated) tests

# Story – example

---

*As an unregistered user  
I want to create a new account  
So that I can buy items*

# Story – example

---

*As a customer  
I want to use my Card  
So that I can buy the items*

# Story – example

---

As a *user*  
I want *to add items*  
~~So that~~ *to my wish list*

# Story – example

---

As a *user*  
I want *to delete items*  
~~So that~~ *from my wish list*



# Story – example

---

*As an administrator*  
*I want to list history of purchases*  
*So that I can compute statistics*

# Story – example

---

*As an administrator*  
*I want to insert items in the website*  
*So that I can sell them*

# Story – example

---

*As an administrator*  
*I want to modify the price of existing items*  
**So that**

# Story – example

---

*As an administrator*  
*I want to receive an alert of low quantity of items in the inventory*  
*So that I can decide whether to send a new order*

# Story – example

---

*As an inventory user  
I want to modify the quantity of items  
So that I can update websites with new  
orders*

## Aspects of a user story 1 / 3

---

- Card
  - Essential description
  - Extended with notes for estimation
  - “Promise for a conversation”

# Aspects of a user story 2 / 3

---

- Conversation
  - Involves
    - the development team
    - the product owner
    - the customer
    - other stakeholders
  - Several times
    - During estimation
    - When selected for implementation
    - When divided in subtasks
    - During implementation
    - In retrospectives

# Aspects of a user story 3 / 3

---

- Confirmation
  - Conditions of satisfaction
  - Defined through conversation
  - Recorded as acceptance tests
  - Complementary to “definition of done”

# Stories and Epics

---

- Story
  - Manageable piece of requirement
    - It is INVEST
- Epic
  - Too large a story
  - Needs to be split into stories

## Epic – example

---

*As a customer*

*I want to be able to view and modify  
wish-list*

*So that I can decide later which items to  
buy*

# Sprint Backlog

---

- Committed stories
  - Relative tasks (may change during sprint)
- Additional tasks, for example:
  - Team improvement
  - Research work
  - Performance and security requirements
  - Bug fixing
- Frozen at the end of sprint planning
- Property of the team

## Additional tasks – examples

---

Diagnose and fix the order processing script error (issue#8367291)

# Additional tasks – examples

---

Investigate solutions for speeding up updating the inventory system after a purchase

# Additional tasks – examples

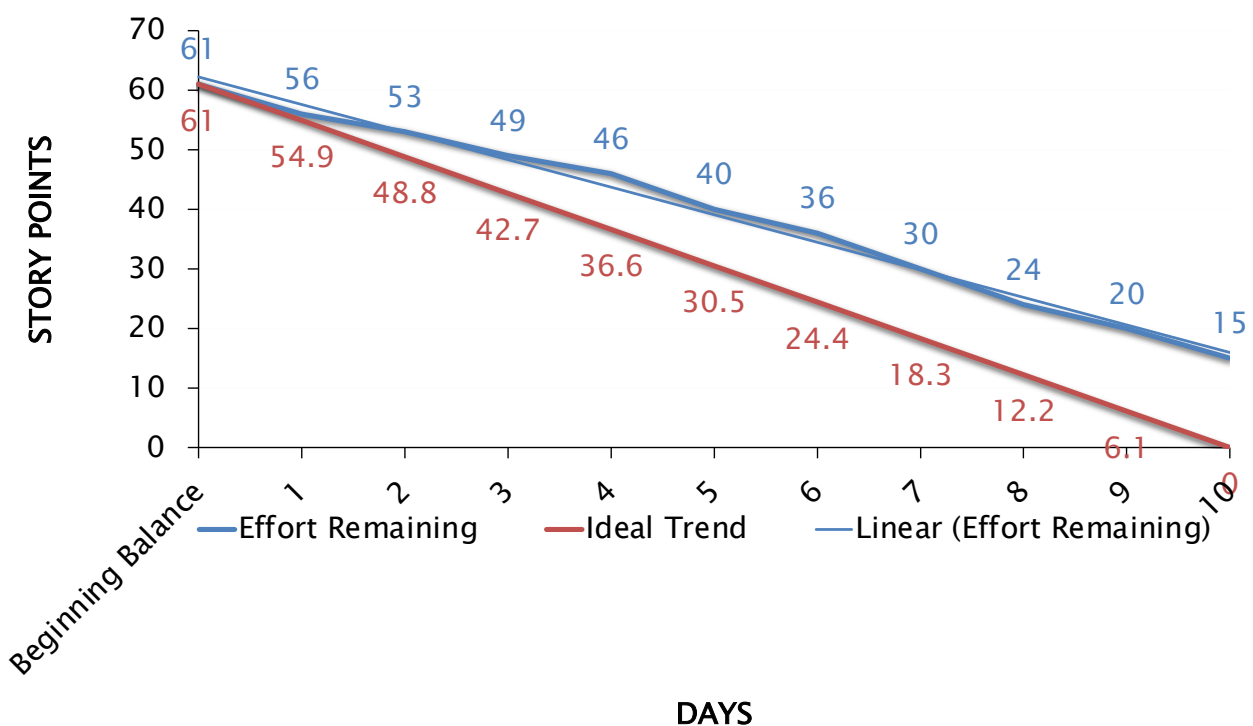
---

Reduce complexity of Class  
OrderDiscountManager

# Information Radiators

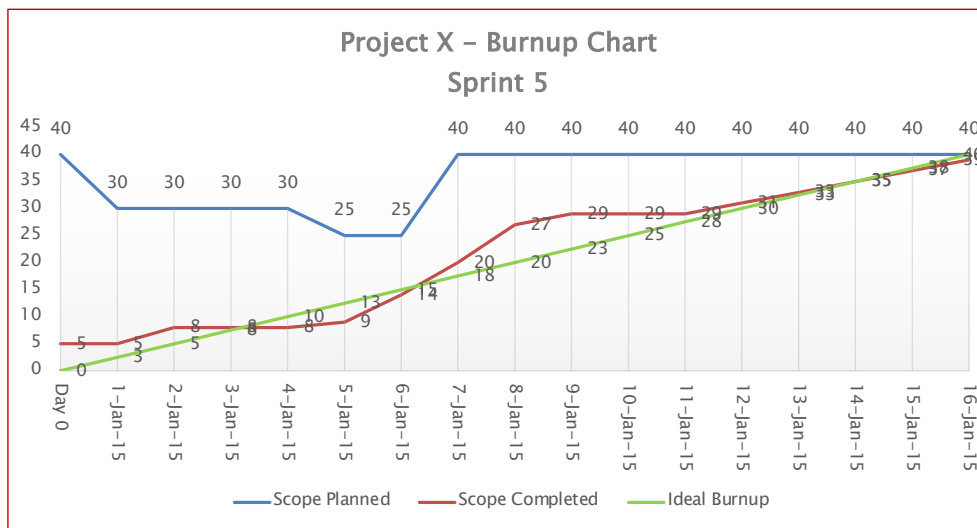
- Burn down chart
  - Residual work units
  - Sprint (team)
  - Release (PO)
- Burn up chart
  - Completed work units
- Task board
  - To do, In progress, Done

## Burn down chart – example

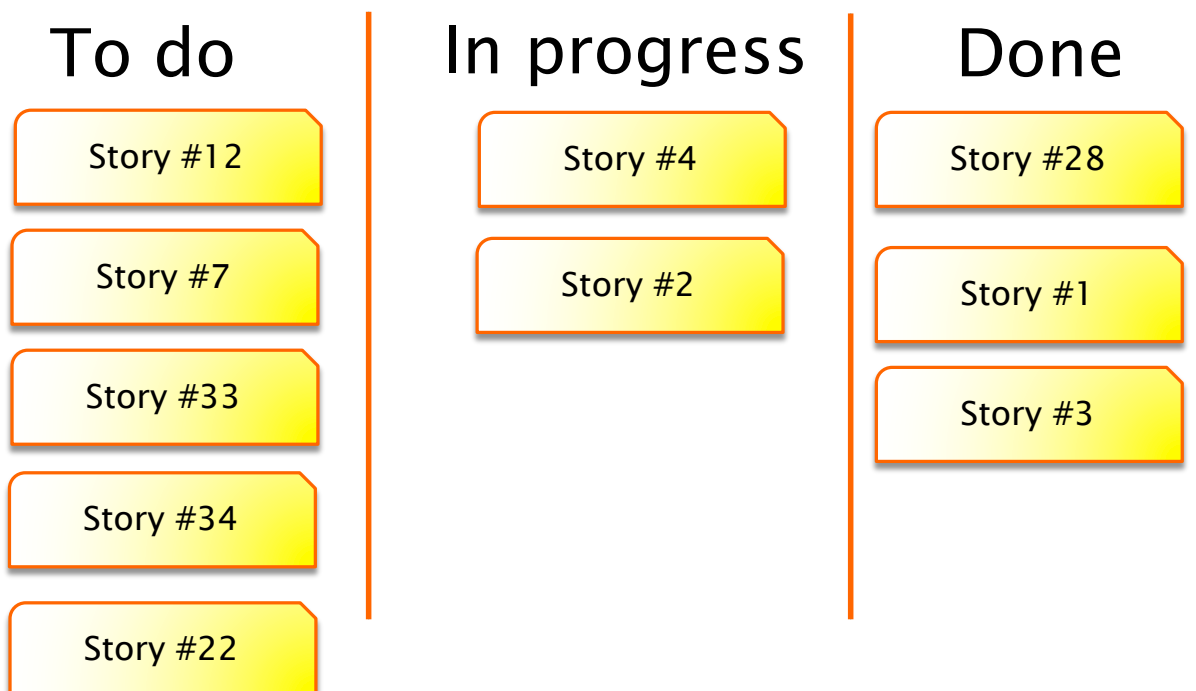




# Burn up chart – example



# Task board – example



# Definition of Done

- Team shared definition
- Typically: shippable
  - Code review
  - Design review
  - Refactoring
  - Performance testing
  - Unit tests passed

## Overview of main artifacts, roles and activities

		Participants				Artifacts & Activities	
		Product owner	Scrum Master	Devel. Team	Stakeholder	Used Artifacts	Activities
Events	Sprint Planning	X	X	X	(opt)	Product backlog Sprint backlog	Definition of a sprint goal Selection of user stories Definition of tasks
	Daily Scrum	(opt)	(opt)	X	(opt)	Sprint backlog Burn-Down/up chart	Update each other
	Story time	(opt)	X	X	(opt)	Product backlog	Backlog refinement Estimation of future stories
	Sprint review	X	X	X	X	Software Sprint backlog	Show piece of working software Collect feedback
	Sprint Retrospective	(opt)	X	X	(opt)	What is deemed useful/necessary	Examination of the last sprint in order to identify possible improvements

# References

---

- K. Schwaber and J. Sutherland, The Scrum Guide, 2017 <http://www.scrumguides.org>
- K.Schwaber, M.Beedle. Agile Software Development with Scrum, Prentice–Hall, 2001
- C.Sims, H.L.Johnson. “The Elements of Scrum” DYNAMICON, 2011.
- Jeff Sutherland's [Scrum Handbook](#)
- H. Takeuchi and I. Nonaka, "The New New Product Development Game," Harvard Business Review, 1986.

