

# SCRUM PROCESS

# Agenda

## Part I: Agile

1. Waterfall
2. What is Agile?
3. Agile manifesto
4. 12 Agile principles
5. Agile vs Waterfall
6. Agile methods

## Part II: Scrum

1. What is Scrum?
2. Scrum pillars
3. Scrum Values
4. Scrum Roles
5. Scrum Artifacts
6. Scrum Events
7. User Story and Story point
8. Burndown/up Chart
9. Definition of Ready and Done
10. Velocity and Capacity

# Part I: Agile

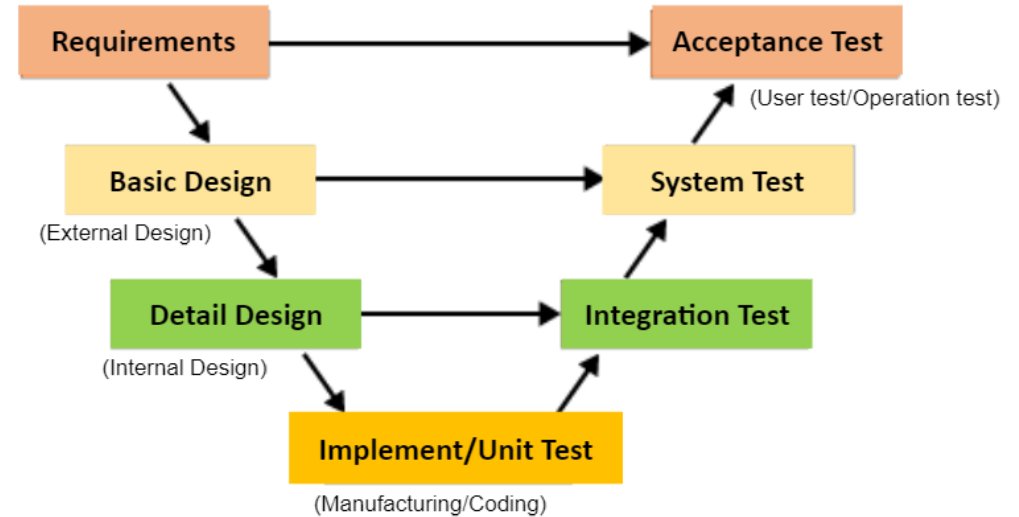


**BOSCH**

Parkhaus

# Agile Waterfall

- Divide the development project into working processes with the flow of time  
ex. Requirement analysis / Basic Design / Detail Design / Implement / Integration Test / System Test
- Use WBS or Gantt chart to make a plan to end these processes at one time. This is the plan-focused development process.
- If previous process doesn't finish, we can't proceed to next process in principle.



# Agile

## Disadvantages of Waterfall

### ~ ENDLESS DOCUMENTS ~

- ▶ Development cannot progress without all the documents, such as Requirement, all the Designs, all the Test Cases, etc. There is a tendency to increase the number of documents in each phase.
- ▶ Developers should spend their efforts on coding, but they have to make a lot of documents, so productivity is lower. They can't check the working software in the middle of development, so development schedule is prolonged.

### ~ NEED TO COMPLETE SPEC IN ADVANCE ~

- ▶ In Waterfall model, we have to receive the order of the project after making clear requirement specifications in advance. To make clear requirements, need to see specification from various perspectives, that requires rich experience and skills.
- ▶ Ideally, it's best to have clear specifications in advance, but in reality, specifications can't cover all the info. One way or another, the requirement is not enough, so in many cases, the products don't come out as expected.

# Agile

## Disadvantages of Waterfall

### ~ FREQUENT REWORK ~

- ▶ In later phases, if we find out some missing info in spec, we must rework the previous phases. If we're testing in the final part and find out some errors related to design, we have to rework from the design phase or add only the design related to the info that was missing.
- ▶ If we don't proceed with the project completely, we have to rework previous phases many times, and cannot keep the project on schedule.

### ~ CANNOT HANDLE REQUIREMENT CHANGE ~

- ▶ Because all phases from Design onward are based on the fixed requirement, after that, we cannot add the requirement for cases like “want to add the features more”, “requirement was not enough”... Basically, we will face the dilemma of not knowing whether to leave what requesters want to do until later or to push back the release schedule.
- ▶ Cannot change/add requirements easily, cannot be flexible.

# Agile

## Disadvantages of Waterfall

### ~ HUGE EFFORT ~

- ▶ The more requirements we have, the more development resource (development cost) we cost, and the more management cost occurs as well.
- ▶ Waterfall model makes the estimation cost larger. With all these reasons, we cannot realize what requesters want at a low cost.

### ~ MANAGEMENT DIFFICULTY ~

- ▶ The larger the project scale, the more difficult the project becomes, we need a PM with high-skilled management.

# Agile

## Disadvantages of Waterfall

### ~ EMPHASIS ON PROCESS, PROCEDURE ~

- ▶ According to all the rules of orders, planners spend their effort making spec for requirements, and the development team also delivers the product based on the spec. When providing interaction thoroughly like this, all members tend to follow the process and steps completely, so contracts and negotiation will increase.
- ▶ If we focus on all procedures, the corroboration between both sides won't happen. In case the project fails, both sides tend to blame it on each other.
- ▶ Developers also should consider user perspective, but when contracts/negotiation increase, they will work for planners, their service mindset will become low.



# Agile

## Waterfall vs Agile – Cases to apply

Below are cases of the plan-focused development process, like Waterfall.

- ▶ **Mission Critical System**

(Systems that may have serious obstacles to customer's work, systems related to human life, projects that customers can never allow developers' failure, etc.)

- ▶ **Many inexperienced developers**

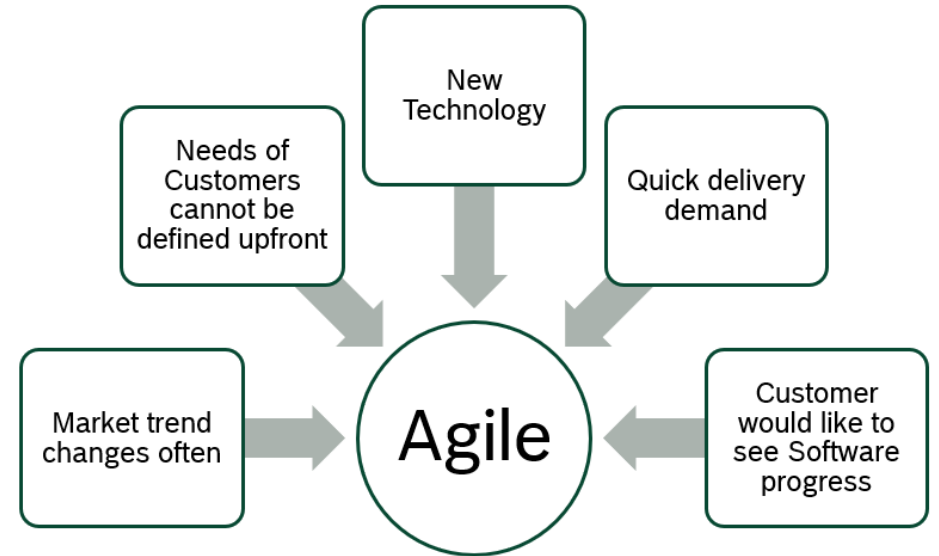
- ▶ **No requirement change during development term**

(especially back-office systems)

- ▶ **Organizational culture focusing on orders, structure**

(especially hierarchical organization)

Below are cases of Agile



# Agile

## What is Agile?

▶ **Agile** = (adj.) "alert", "quick"

The Agile is a software development process to quickly and flexibly handle requirement specification change. In the Agile, the specified ways are not defined, and "Agile" is called as a general term from many advanced methods.

Most of the Agile software development have features below:

- 1. Adopt a short time-boxed unit called "Iteration" and minimize risks. By continuing iterative cycle, you can develop functions one by one. (Iteration = Sprint)**
- 2. Focus on thinking about customer actively and making communication closely**

# Agile







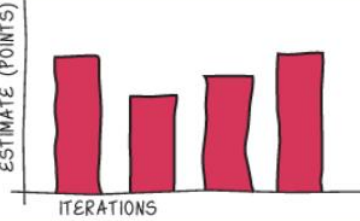
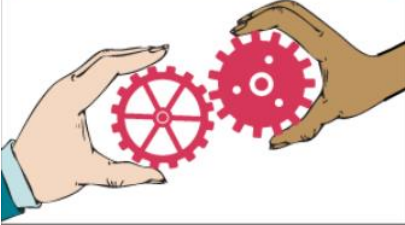



## Agile Manifesto

**Individuals and interactions** over **processes and tools**  
**Working software** over **comprehensive documentation**  
**Customer collaboration** over **contract negotiation**  
**Responding to change** over **following a plan**

While there is value in the items **on the right**, we  
value the items **on the left** more

# Agile

## 12 Principles behind Agile Manifesto

<p><b>1</b> Satisfy the <b>customer</b></p> 	<p><b>2</b> Welcome <b>change</b></p> 	<p><b>3</b> Deliver <b>frequently</b></p> <table border="1" data-bbox="1143 319 1513 529"> <thead> <tr> <th>Sprint 1</th> <th>Sprint 2</th> <th>Sprint 3</th> <th>Sprint 4</th> <th>Sprint 5</th> </tr> </thead> <tbody> <tr> <td>story</td> <td>story</td> <td>story</td> <td>story</td> <td>story</td> </tr> <tr> <td>story</td> <td>story</td> <td>story</td> <td>story</td> <td>story</td> </tr> <tr> <td>story</td> <td>story</td> <td>story</td> <td>story</td> <td>story</td> </tr> </tbody> </table>	Sprint 1	Sprint 2	Sprint 3	Sprint 4	Sprint 5	story	story	story	story	story	story	story	story	story	story	story	story	story	story	story	<p><b>4</b> Work <b>together</b></p> 
Sprint 1	Sprint 2	Sprint 3	Sprint 4	Sprint 5																			
story	story	story	story	story																			
story	story	story	story	story																			
story	story	story	story	story																			
<p><b>5</b> Trust and <b>support</b></p> 	<p><b>6</b> Face-to-face <b>conversation</b></p> 	<p><b>7</b> Working <b>software</b></p> 	<p><b>8</b> Sustainable <b>development</b></p> 																				
<p><b>9</b> Continuous <b>attention</b></p> 	<p><b>10</b> Maintain <b>simplicity</b></p> 	<p><b>11</b> Self-organizing <b>teams</b></p> 	<p><b>12</b> Reflect and <b>adjust</b></p> 																				

# Agile

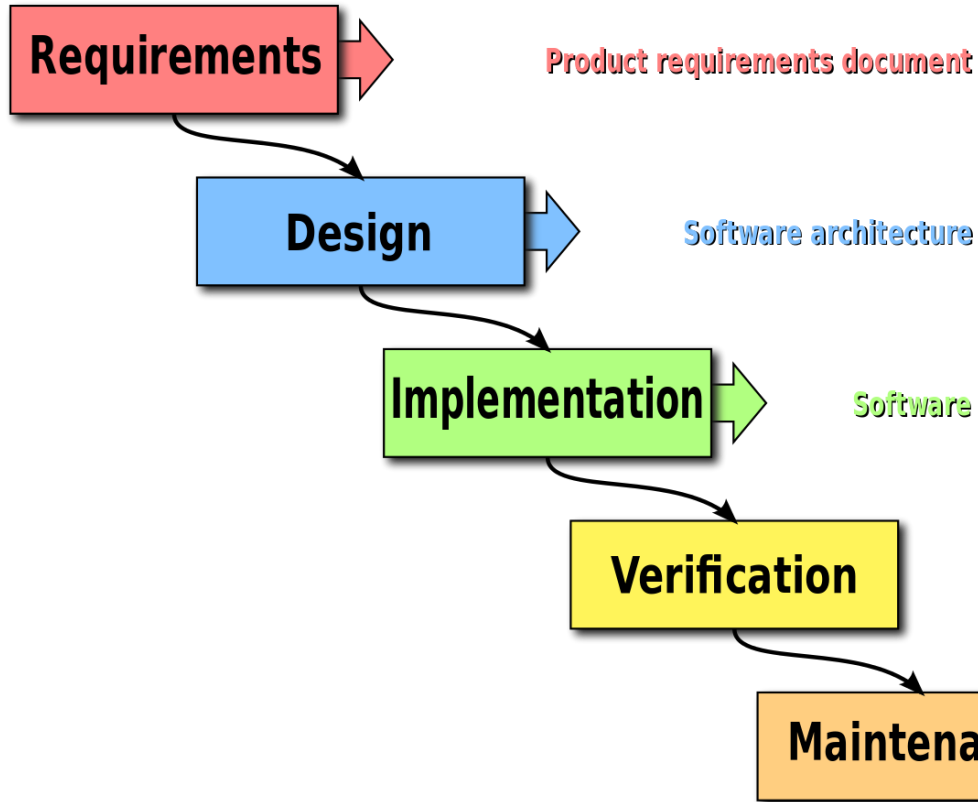
## 12 Principles behind Agile Manifesto

No	Principles	Explanation
1	<b>Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.</b>	We think about how to achieve business goal, not target of QCD.
2	<b>Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.</b>	Recently, as there is no definite answer for requirements, it's not surprising that requirement changes. If it's worth, changing should be accepted positively at any time.
3	<b>Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.</b>	By delivering in short timescale, we can get the feedback early from customers and verify hypothesis
4	<b>Business side and developers must work together daily throughout the project.</b>	All people concerned always confirm the direction and goal. Working in the same place will make cycle of continuous release and feedback run smoothly.
5	<b>Build projects around motivated individuals. Give them the environment and support they need and trust them to get the job done.</b>	Build a developers with motivated and positive members. Do trust developers. Do not give detailed instructions as they will block the autonomous work of developers.
6	<b>The most efficient and effective method of conveying information to and within a developers is face-to-face conversation.</b>	Try to interactively discuss face-to-face in all people concerned. We should know that dialogue with words only is inefficient.

No	Principles	Explanation
7	<b>Working software is the primary measure of progress.</b>	Confirm both progress and quality by looking to articles(software). Make plan to create things in small units so that we can let the requesters see the working software soon.
8	<b>Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.</b>	People concerned care about the repeating "rhythm" regularly. Keep mind and body in healthy condition.
9	<b>Continuous attention to technical excellence and good design enhances agility.</b>	Developers always learn new techniques, apply effective ones to product. They try to control technical debt and find problems early in auto testing.
10	<b>Simplicity - the art of maximizing the amount of work not done - is essential.</b>	All people concerned think about values customers need, they should regard work or function not linking to those values as a waste. Developers should not take requests from customers at face value without questioning "Is that function really necessary?".
11	<b>The best architectures, requirements, and designs emerge from self-organizing teams.</b>	Self-organizing team (autonomous team) is a team that each developer is responsible for own action and work and is also a team that can get maximum effect by the cooperation between developers. The developers decides disciplines/rules by himself, and developers take the initiative in improving their work or solving problems by themselves
12	<b>At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.</b>	Hold the Retrospective meeting regularly and keep adjusting our working style. Do not blame individuals. Allow failures, learn from them, and lead to improvement.

# Agile

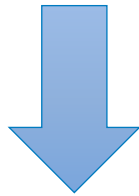
## Agile Development Process Flow



# Agile

## Solve the Problems of Waterfall Model

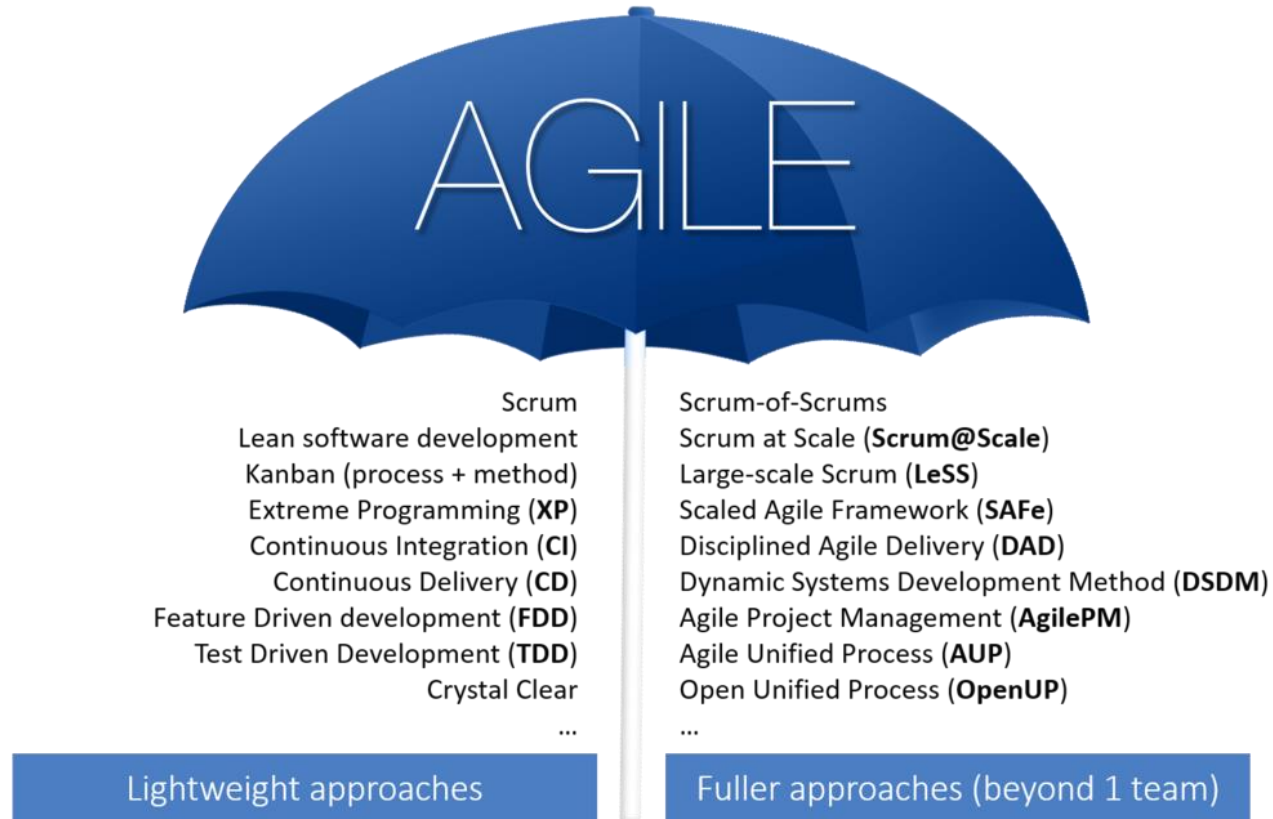
- ▶ Spend more time on development rather than creating documents, productivity will be improved
- ▶ Shorten the schedule (less rework as well)
- ▶ Reduce the load on requirement definition
- ▶ Accept the specification change flexibly
- ▶ Reduce the total development cost
- ▶ Reduce the risks
- ▶ More collaborative because communication is important



**As a result, QCD problems are resolved, the project becomes more successful.**

# Agile

## Representative Methods of Agile





# Part II: Scrum



**BOSCH**

Parkhaus

# Scrum

## What is Scrum?

- ▶ Scrum is one of Agile development methods which focuses on **“how to proceed the work as a team”**, and a framework within which people can address complex adaptive problems. It can productively and creatively deliver the most valuable products as much as possible. It’s important to work together as a team in order to achieve common goals.
- ▶ Team is required to work as an autonomous organization.

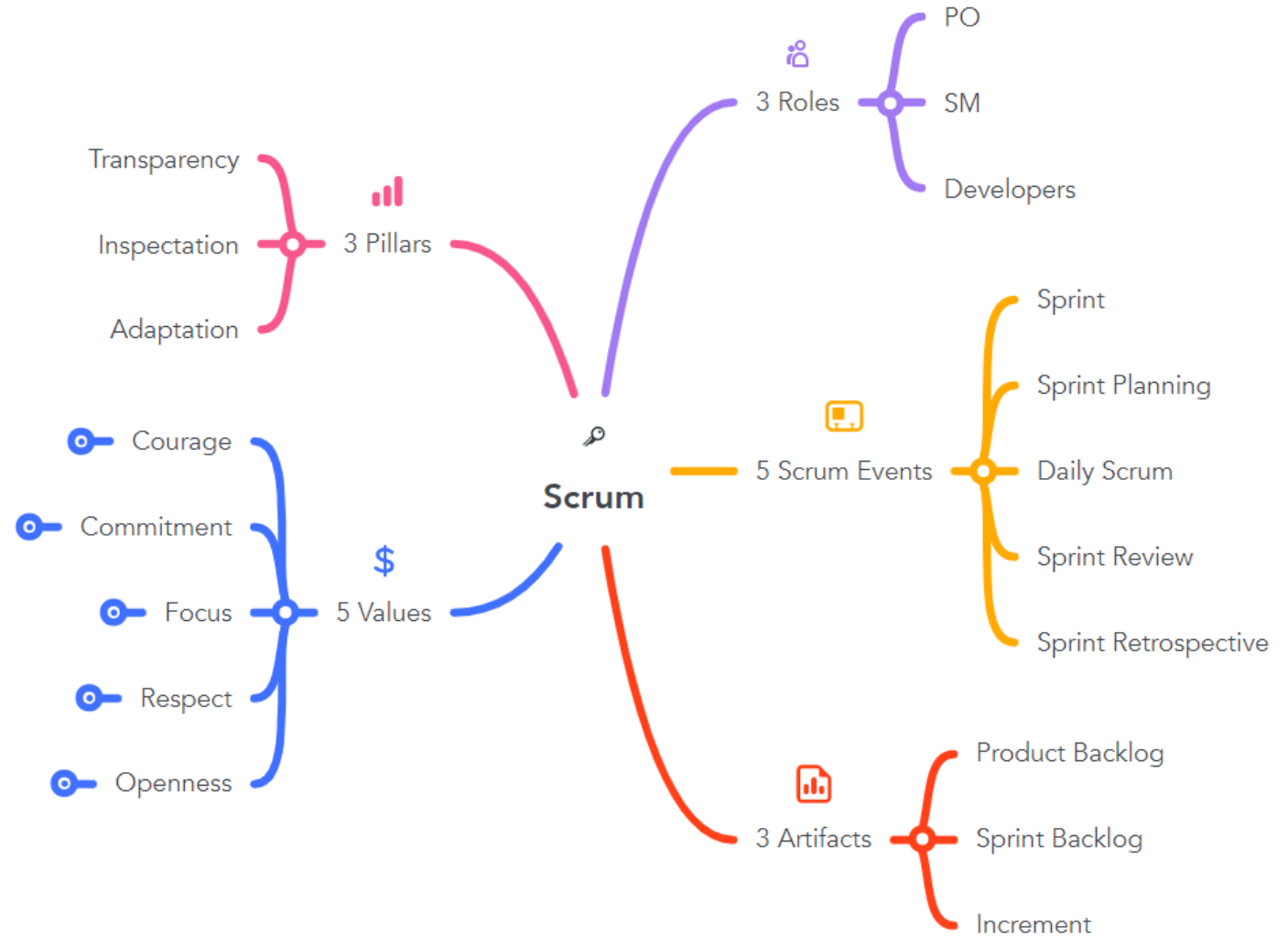


# Scrum

## What is Scrum?

Scrum is an Agile Development framework that:

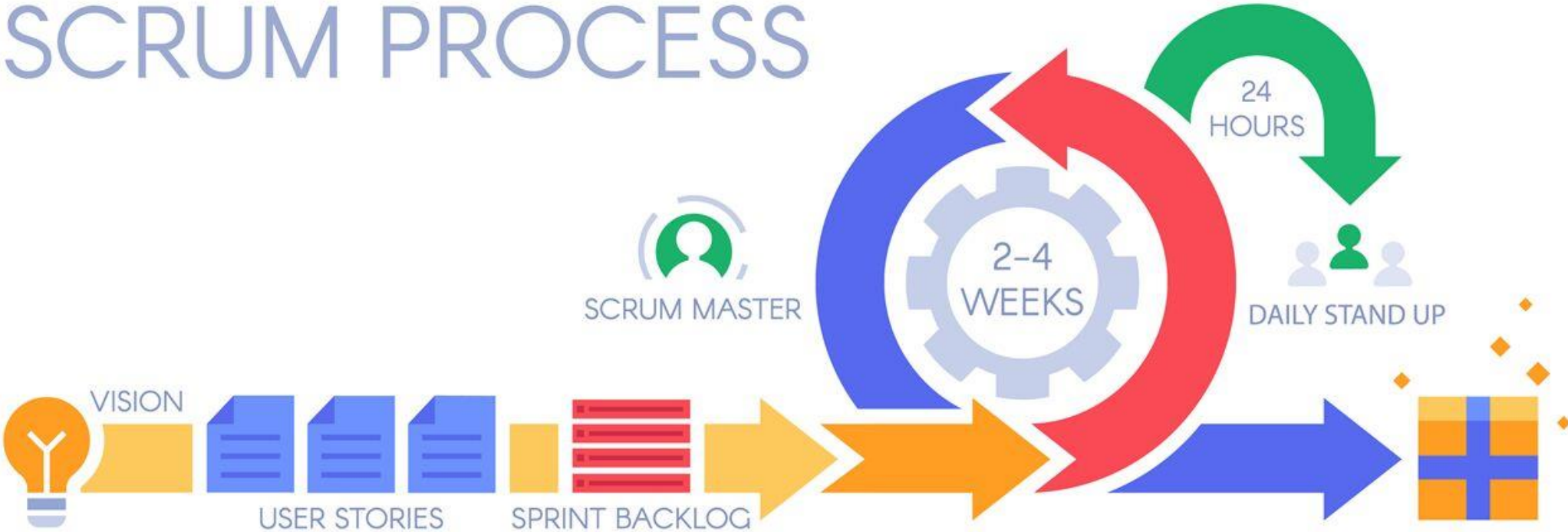
- ✓ Focuses on delivering the highest priority **business value** to the customer
- ✓ Team is **self-managing** and **cross-functional**
- ✓ Shortens **feedback** loop between customer and developer
- ✓ Tests early and often to see if the system being developed will **deliver value**



# Scrum

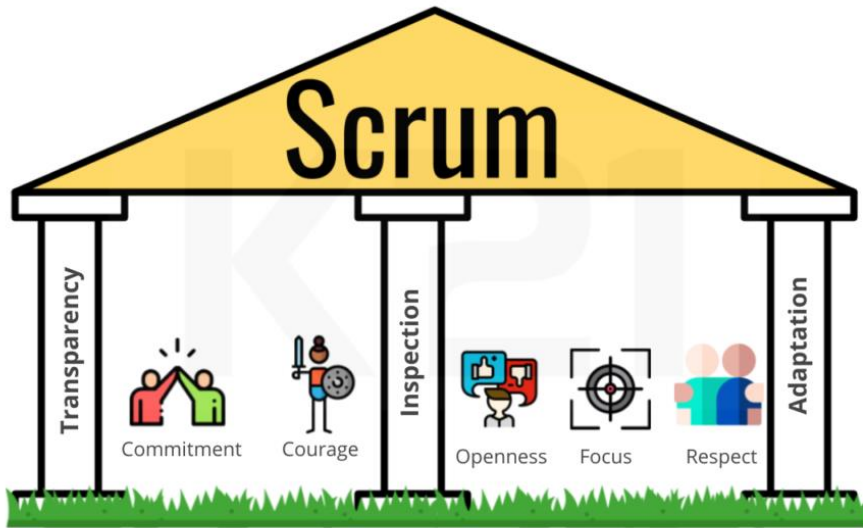
## What is Scrum?

# SCRUM PROCESS



# Scrum

## Scrum Pillars



### Transparency

- ▶ The process used by the Scrum Team, whose activities and expectations must be visible to all those interested in the Team's results. Nothing should be swept under the carpet.
- ▶ Normally this transparency is implemented via the Task Board which the team uses to give visibility to the workflow and items of value being developed.
- ▶ More mature teams are more transparent. The relationship of trust between them and the other stakeholders has a strong influence on transparency.
- ▶ If there is a relationship of mistrust or immaturity, there will be plenty of "hidden" work with little visibility.
- ▶ Furthermore, transparency is related to the roles people perform during the development of the product, results achieved, expectations, collaboration, etc.

### Inspection

- ▶ Work should be inspected frequently. The Scrum Team asks itself whether it is progressing towards the goals and business results. It should also ask itself how to perfect its working method, the team atmosphere, and the quality of the product being delivered. Using Scrum is to constantly question: how can we improve?

### Adaptation

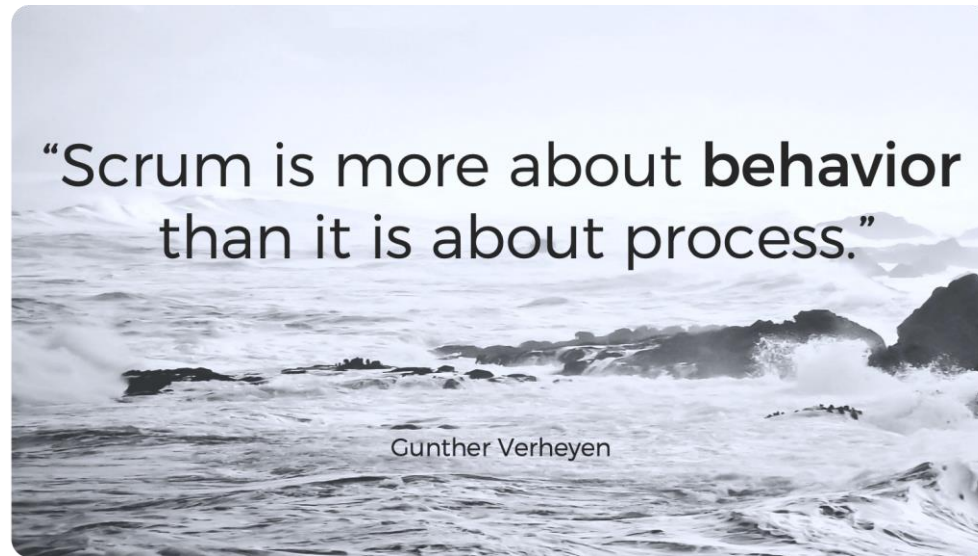
- ▶ If the inspection determines that we're not progressing to our goals, it's time to adapt. This adaptation can be motivated by different factors: changes in the market, competition, client needs, business needs, etc

*"Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage." (The 12 Principles of the Agile Manifesto, 2001)*

# Scrum

## Scrum Values

- ▶ Values drive behavior.
- ▶ Behavior reflects values.



- ▶ Scrum is a framework of rules, principles and...**values**.

# Scrum

## Scrum Values



# Scrum Roles



Owens "what" is desired and "why" it is desired



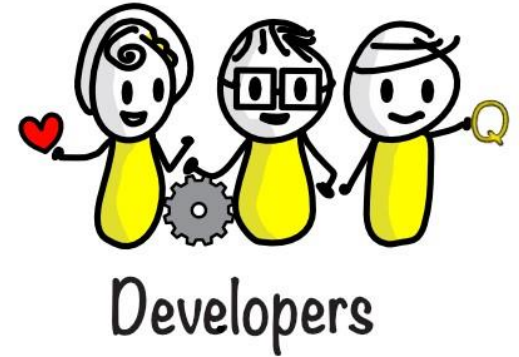
Direct communication encouraged

Scrum Teams are **cross-functional**, meaning the members have **all the skills necessary** to create value each Sprint.

They are also **self-managing**, meaning they internally **decide who does what, when, and how**



Keeper of Scrum process, facilitator

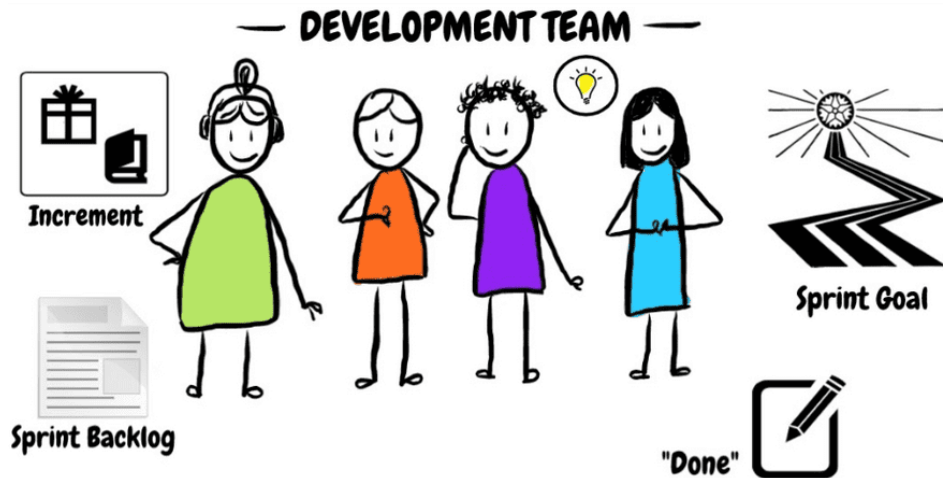


Owens "how" and "how quickly" is delivered



# Scrum

## Roles - Developers



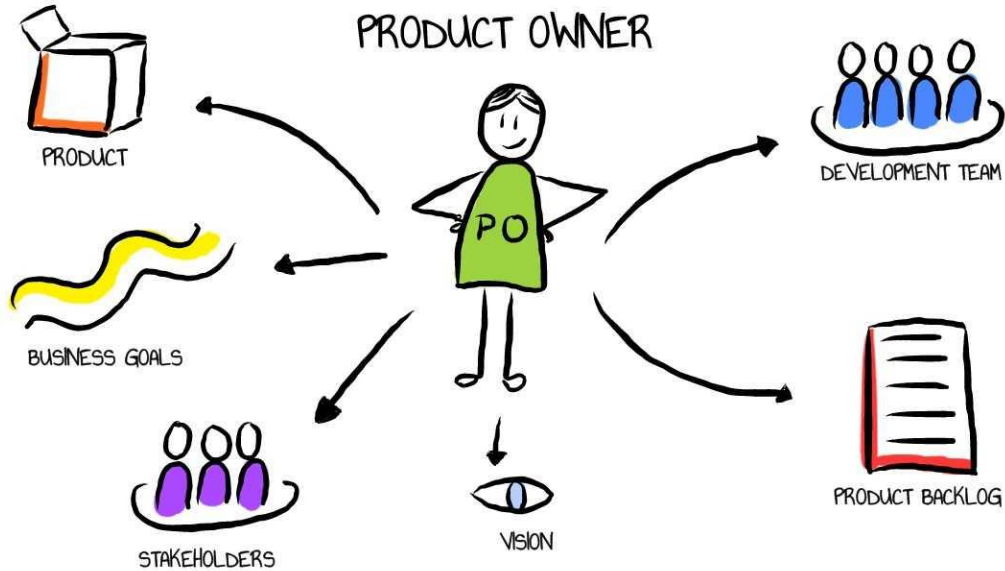
Developers are the people in the Scrum Team that are committed to creating any aspect of [a usable Increment for each Sprint](#).

Developers are always **accountable** for:

- Creating a plan for the Sprint, the Sprint Backlog;
- Instilling quality by adhering to a Definition of Done;
- Adapting their plan each day toward the Sprint Goal;
- Holding each other accountable as professionals.

# Scrum

## Roles – Product Owner



R.B.

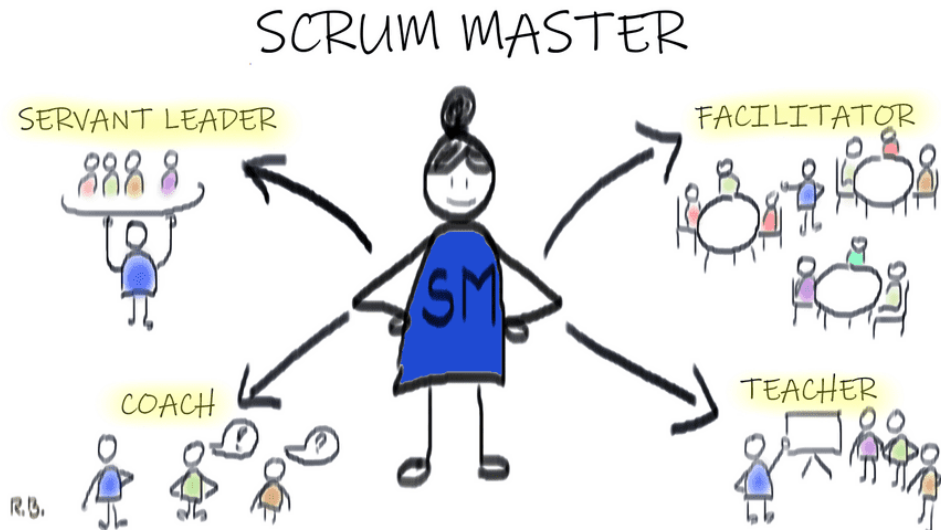
The Product Owner is accountable [for maximizing the value of the product](#) resulting from the work of the Scrum Team.

The Product Owner is also **accountable** for [effective Product Backlog management](#), which includes:

- Developing and explicitly communicating the Product Goal;
- Creating and clearly communicating Product Backlog items;
- Ordering Product Backlog items;
- Ensuring that the Product Backlog is transparent, visible, and understood.

# Scrum

## Roles – Scrum Master



Scrum Master is **accountable** for establishing Scrum as defined in the Scrum Guide. Helping everyone understand Scrum theory and practice, both within the Scrum Team and the organization.

The Scrum Master is **accountable** for the Scrum Team's effectiveness. Enabling the Scrum Team to improve its practices, within the Scrum framework. Scrum Masters are true leaders who serve the Scrum Team and the larger organization.

The Scrum Master [serves the Scrum Team](#) in several ways, including:

- Coaching the team members in self-management and cross-functionality;
- Helping the Scrum Team focus on creating high-value Increments that meet the Definition of Done;
- Causing the removal of impediments to the Scrum Team's progress;
- Ensuring that all Scrum events take place and are positive, productive, and kept within the timebox.

The Scrum [Master serves the Product Owner](#) in several ways, including:

- Helping find techniques for effective Product Goal definition and Product Backlog management;
- Helping the Scrum Team understand the need for clear and concise Product Backlog items;
- Helping establish empirical product planning for a complex environment;
- Facilitating stakeholder collaboration as requested or needed.

The Scrum Master [serves the organization](#) in several ways, including:

- Leading, training, and coaching the organization in its Scrum adoption;
- Planning and advising Scrum implementations within the organization;
- Helping employees and stakeholders understand and enact an empirical approach for complex work;
- Removing barriers between stakeholders and Scrum Teams.

# Scrum

## Anti patterns of PO

### ► Originally...

- Too busy Product owner
- Absent Product owner
- Product owner not participating in Scrum events
- Product owner is also assigned just because he/she is a Manager or Leader
- Originally the Product owners with many people whose decision-making authority is shared
- Concurrently serve as Product Owner and Scrum master
- Do not talk with Scrum Master and get feedback from Scrum Master

### ► How to interact with the Developers and Scrum Master

- PO regards the relation between PO and developers as the confrontational partner, so PO doesn't recognize PO is also a member of Scrum Team
- Interfere and give instruction to how developers works
- Do not trust developers' estimation even if it explains
- Do not allow developers to reduce technical debt
- Do not allow developers to communicate directly with stakeholders
- Rely too much on e-mails and digital tools to communicate with developers, face-to-face communication is not enough
- Don't want to answer questions from developers or don't answer
- Not listen to developers' conversation
- Instruct developers what to do (=instruct detail tasks), and act as a Manager
- Monitor developers' behaviors, progress, and task status
- Project the order schema to the relationship between developers and PO

# Scrum

## Anti patterns of PO

### ▶ Relation with Stakeholders

- Do not say “No” to Stakeholders, product backlog gets bigger and bigger
- Change the order of product backlogs frequently as misled by stakeholders
- Do not invite Stakeholders to Sprint Review
- Not listen to the needs or opinions of Stakeholders and customers
- Have not committed to business KGI or KPI

### ▶ Product

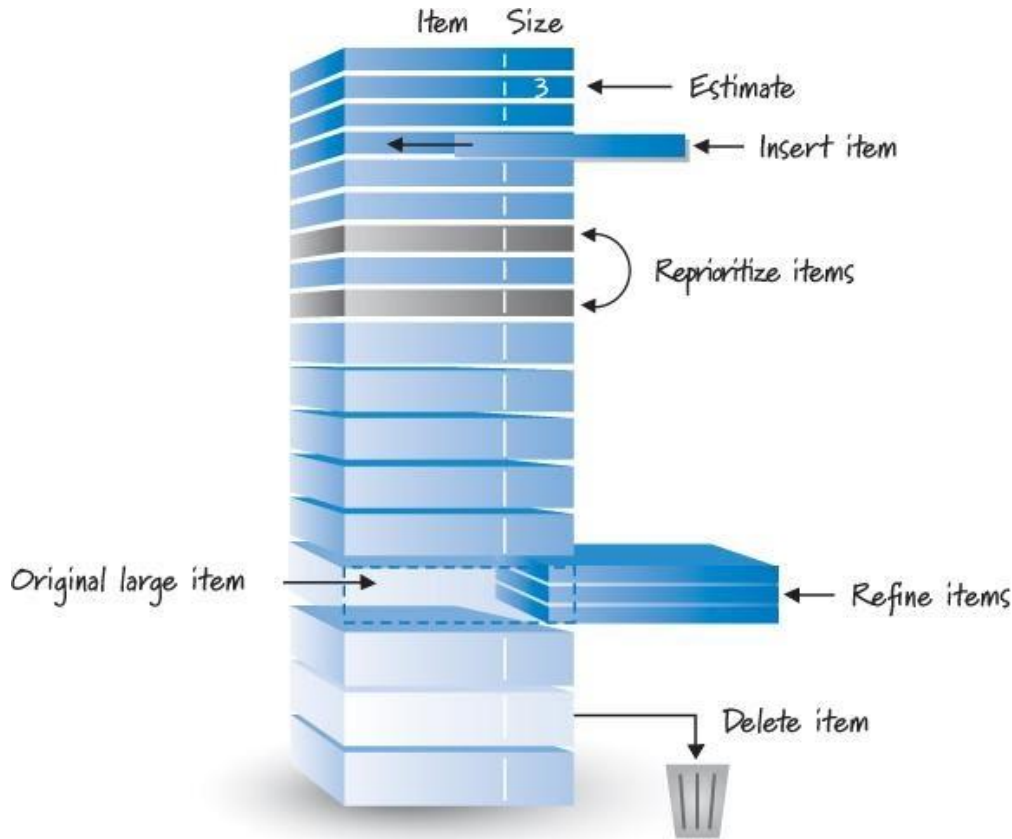
- Not verify ideas before implementing ideas
- Lack of product vision under development
- PO thinks it's valuable to provide more features than useful features
- Do not place importance on quality

### ▶ Product Backlog

- Do not clarify why that User Story is important
- Not dividing too big User Story
- Prepare too detailed User Story (no room for negotiation)

# Scrum

## Artifacts – Product Backlog



- A list of User Stories.
- The owner of PBL is PO.
- When planning sprint, PO sees the Product Backlog, and discusses who wants to do it and reasons for doing it, then agrees to User Stories developing on next sprint.
- The higher the priority is, the more accurate the Feature, User Benefit, Story Point, Ready (Start Condition), Done (Completion Condition), Acceptance Criteria are.
- The Developers who will be doing the work are responsible for the sizing. The Product Owner may influence the Developers by helping them understand and select trade-offs.

### Commitment: Product Goal

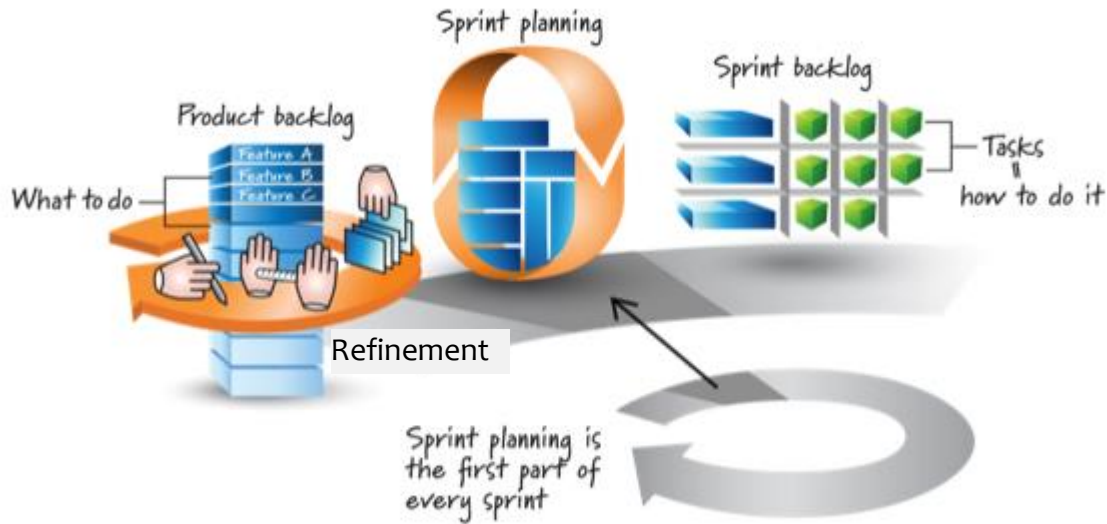
The Product Goal describes a future state of the product which can serve as a target for the Scrum Team to plan against. The Product Goal is in the Product Backlog. The rest of the Product Backlog emerges to define “what” will fulfill the Product Goal.

*A product is a vehicle to deliver value. It has a clear boundary, known stakeholders, well-defined users or customers. A product could be a service, a physical product, or something more abstract.*

The Product Goal is the long-term objective for the Scrum Team. They must fulfill (or abandon) one objective before taking on the next.

# Scrum

## Artifacts - Sprint Backlog



Copyright © 2012, Kenneth S. Rubin and Innolution, LLC. All Rights Reserved.

- A list of tasks that divided User Stories for 1 sprint selected from the Product Backlog into.
- The owner of SBL is Developers.
- Each task listed in Sprint Backlog needs to be completed within one day at most.

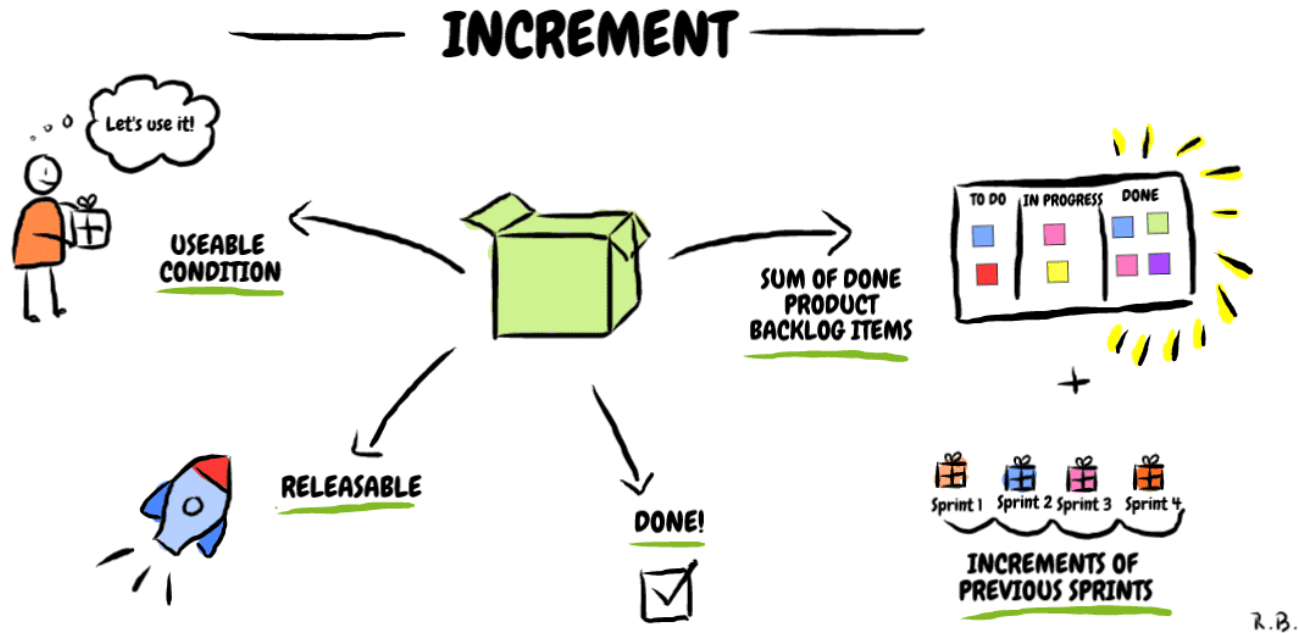
### Commitment: Sprint Goal

The Sprint Goal is the single objective for the Sprint. Although the Sprint Goal is a commitment by the Developers, it provides flexibility in terms of the exact work needed to achieve it. The Sprint Goal also creates coherence and focus, encouraging the Scrum Team to work together rather than on separate initiatives.

The Sprint Goal is created during the Sprint Planning event and then added to the Sprint Backlog. As the Developers work during the Sprint, they keep the Sprint Goal in mind. If the work turns out to be different than they expected, they collaborate with the Product Owner to negotiate the scope of the Sprint Backlog within the Sprint without affecting the Sprint Goal.

# Scrum

## Artifact - Increment



- ✓ Sum of all product backlog items completed to date
- ✓ Must always be in a readily releasable ('done') state



# Scrum

## User Story

- ▶ **User Story is a simple description of what wants to realize in natural language**
- ▶ You should write a user story **from the user perspective**. It is expected for PO to write, but developers may do.

As XX user  
I want/can XX (function/ non-function)  
So that XX (merit)



- Who is a story for?
- What you want to do?
- Why you want to do that?

- ▶ **Example:**  
As [secretary of drinking party]  
I can [remind participants right before drinking party]  
So that [prevent the cancellation of participants]

# Scrum

## User Story - Exception

- ▶ Although it is basically required to write a story from a user viewpoint, there are exceptions as follows.
- It also describes a technical story (technical work necessary for system view, an investigation that couldn't finish in Spike, refactoring, non-functional requirement, etc.)
- Sometimes necessary work occurs in the middle to solve the risk. It is better to put risk solutions in Product Backlog.
- When it comes to release sprint, list up stories such as release preparation, important tasks brought back after release judgment meeting, and release work, etc.

✘ Because the product backlog is also a worklist for developers, **all works that are done in sprints must be described.**

# Scrum

## User Story – INVEST

► 6 elements of a well written User Story (= **INVEST**)

### Independent

It must be independent of each other. Eliminate dependency/context as much as possible.

### Negotiable

Can adjust the plan of how to make, and there is room for negotiation.

### Valuable

Value is created by implementing the story.

### Estimable

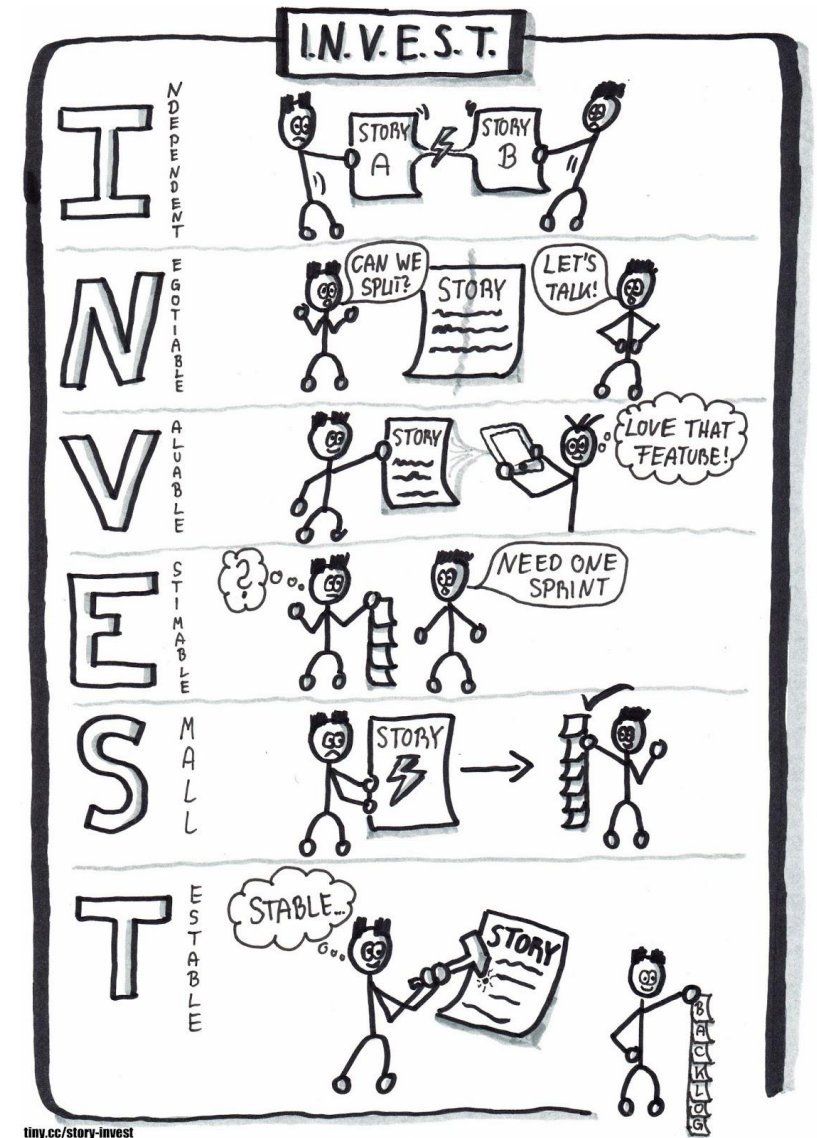
The story has a granularity that can estimate.

### Sized Right (Small)

It has a proper size. The story is divided sufficiently.

### Testable

Acceptance criteria are clear. You can write the acceptance test.



[tiny.cc/story-invest](http://tiny.cc/story-invest)

# Scrum

## User Story – Acceptance criteria

In a project, there are requirements, also called **User Stories** in Scrum. They're written on index cards or post-it notes.

On the front of the card, write the user story/requirement, and at the back, write the acceptance criteria.

By definition, **Acceptance criteria** is the criterion that a **product requirement must meet** to be considered complete.

### Customer Value:

As a user, I want to change the display brightness so that I can save energy or I can choose the perfect brightness for the light management of the whole kitchen.

### Acceptance Criteria:

- n different levels of display brightness
- Level 1 is dark
- Level n is bright
- the usersetting shall be saved directly each change
- The configuration defines the default value
- For the first implementation n shall be 5
- The change of the brightness shall directly visible.
- It must be possible to slide from 1 to n
- Back button must lead back to the previous menu for testing

# Scrum

## Acceptance Criteria vs Definition of Done



Acceptance criteria are unique criteria that apply to a specific user story (it completes that user story).  
In contrast, Definition of Done is the common criterion that applies to all user stories in a sprint or project

# Scrum

## Definition of “Ready” and “Done”

When the Agile development is proceeded within the allotted time as agreed. If the definition of [*Ready*] and [*Done*] are vague then the problems such as “unable to complete within the Sprint” and “The team considers that the User Story/Task completed, however that is not the perception of the PO” are likely to occur. In order to prevent this, the results of the following two definitions are agreed between the PO and the team to assure that everyone is one the same page.

- ▶ **Definition of “Ready”** - Having a Definition of Ready means that stories must be immediately actionable. The Team must be able to determine what needs to be done and the amount of work required to complete the User Story or PBI.
- ▶ **Definition of “Done”** - In order to be able to decide when an activity from the Sprint Backlog is completed, the Definition of Done (DoD) is used. It is a comprehensive checklist of necessary activities that ensure that only truly done features are delivered, not only in terms of functionality but in terms of quality as well. The DoD may vary from one Scrum Team to another but must be consistent within one team.

# Scrum

## Definition of “Ready” and “Done” - Example

### ► DOD

#### Documentation

##### Inside code:

- Interfaces have to be documented 100%

##### Outside code:

- If necessary a deeper documentation is done in the wiki
- We integrate the Jira links into the wiki documentation.
- We use the [EOX6021-Coffee Project Diary area](#) for every kind of documentation.

#### Test

- Everything has to be tested!
- Domain: Unit & Integration tests are necessary
- UI: Squish should be used whenever possible in a rational way

#### Review

- We don't start with a new ticket, before an open review is done
- Every developer decides for himself if one or two reviewers are needed
- We name our preferred reviewer directly
- We do a review for our own code (according to our checklist) before doing a pull-request
- No pull requests > 500 lines of code!

#### Code Conventions

- We develop compliant to the [GED-SH Code Conventions](#)
- We develop compliant to clang-tidy & clang-format

#### Build System

- Jenkins is running green “honestly”

### ► DOR

- Acceptance criteria was created and updated to latest by PO, there is no “waiting to confirm”
- Dummy data was provided by customer
- UI design was provided by designers
- Source code of previous Sprints was merged into development branch

# Scrum

## Burndown/up Chart

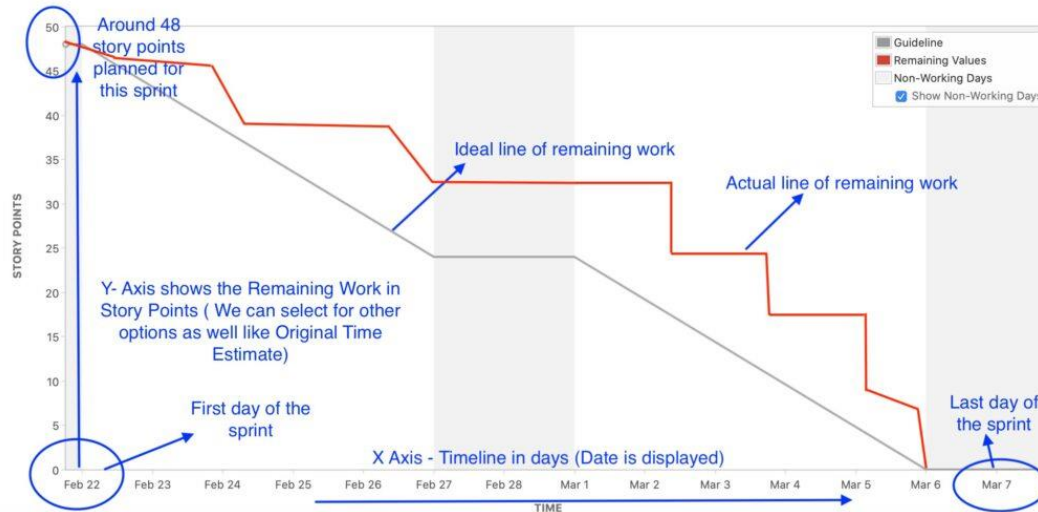
A line graph that you can confirm the progress at a glance.

### ► Release Burndown Chart

You can check the progress until the product is released. PO manages it. Ideal term is a quarter (2-3 months). The vertical axis shows total story points, and the horizontal axis shows the number of sprints. (Because the scale on the horizontal axis is for each sprint, it is better to write it as the number like the XX sprint or set the start date of each sprint.) Each time you do a sprint, it reflects the remaining cumulative story points.

### ► Sprint Burndown Chart

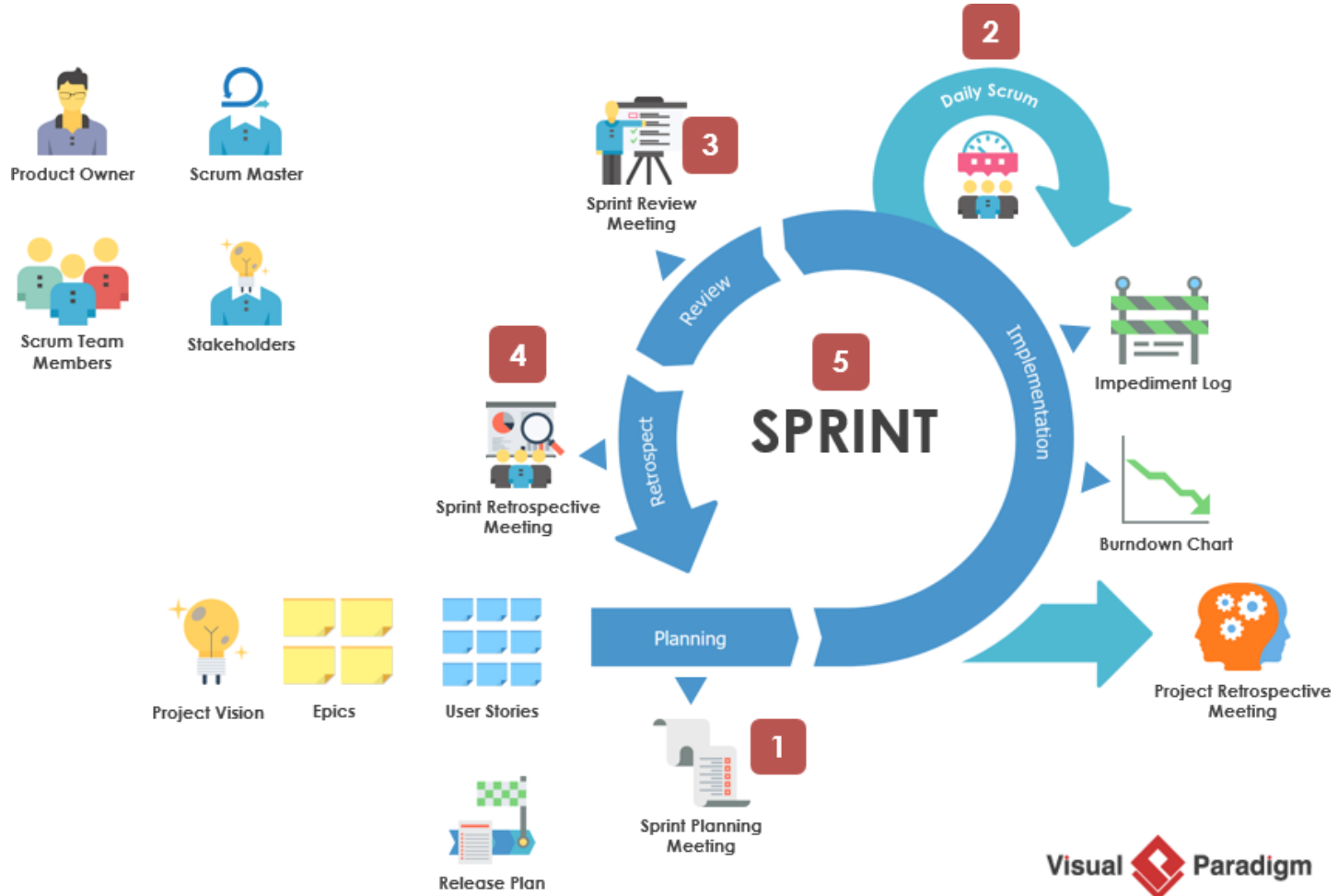
You can check the work progress in 1 sprint. Team manages it. The vertical axis shows the total efforts (hours) of the task, the horizontal axis shows 'day XX', or the date. It reflects the remaining cumulative efforts (hours) every day.





# Scrum

## Scrum Events



# Scrum

## Scrum Events

Event	Participant	Timebox	Input	Output	Process to conduct
Sprint	Scrum Team	1-4 weeks => Customer would like to check progress at RBVH each 3 weeks	Previous Sprint conditions	Working Increment product	
Sprint Planning	Scrum Team	6 hours/3 weeks	Product Backlog Velocity/Capacity Definition of Done Retrospective Actions	Sprint Goal Forecast Sprint Backlog	PO presents business objective Scrum Team come with Sprint Goal Developers forecast to meet Sprint Goal (base on velocity) Developers come with plan to meet Sprint Goal (break User Story into small tasks and estimate hours for each task)
Daily Scrum	Developers	15 minutes	Sprint Goal Sprint Backlog Sprint Burndown/up Impediment list	Updated Sprint Backlog New impediments	Developers review last 24 hours work Plan work for next 24 hours Update Remaining hour for task Share impediments if there any
Sprint Review	Scrum Team + Stakeholders	3 hours/3 weeks	Increment Sprint Results Product Backlog Stakeholder Feedbacks	Updated Product Backlog	PO share goal to everyone Developers demonstrate working increment Stakeholders provide feedbacks Everyone discuss about progress and what next
Sprint Retrospective	Scrum Team	2,5 hours/3 weeks	Processes and Tools used in Sprint People and Relationship Definition of Done Previous Retrospective Actions	Retrospective Actions Plan Updated Definition of Done	Everyone discuss and collect data about how sprint went Brainstorm ideas how to improve next sprint => Retrospective Methods Agree to retrospective actions plan

# Scrum

## Backlog Refinement

- ▶ This is the refinement of product backlog.
- ▶ Review priority change, start condition (Ready), completion condition (Done), acceptance criteria, story point, etc.
- ▶ Prepare User Stories to undertake in the next sprint or after the next sprint.
- ▶ It is a crucial task for any Agile developers recurring every week to ensure there is a sufficient number of products in the backlog list. It occurs between two successive Sprint Planning meetings to keep the backlog healthy and organized.

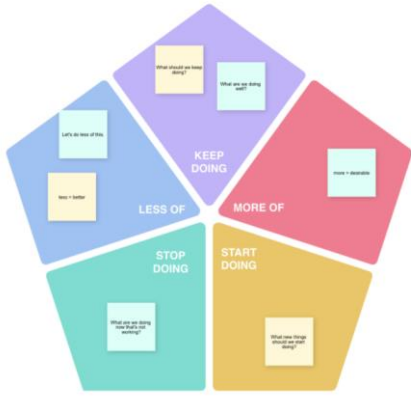
Event	Participant	Timebox	Input	Output	Process to conduct
Backlog Refinement	Scrum Team	10% length of Sprint	Product Backlog Velocity/Capacity	User Stories are prioritized, detailed and estimated by story point	Removes user stories that are no longer relevant. Creating and reassessing the relative priority in the backlog list as per new needs. Confirming that the Scrum Team is on the same page by keeping a close check on priority updates. Assigning and correcting estimates to stories as per the new information received. Breakdown of user stories into smaller tasks.

# Scrum

## Backlog Refinement vs Sprint Planning

Product Backlog Refinement	Sprint Planning
<ul style="list-style-type: none"> <li>•It is an unofficial meeting.</li> </ul>	<ul style="list-style-type: none"> <li>•It's an official meeting.</li> </ul>
<ul style="list-style-type: none"> <li>•The Product Owner does the work of Backlog Refinement and is responsible for maintaining the backlog.</li> </ul>	<ul style="list-style-type: none"> <li>•The Product Owner meets the developers and ScrumMaster. The ScrumMaster is responsible for the events of a Sprint Planning meeting.</li> </ul>
<ul style="list-style-type: none"> <li>•It is conducted between two Sprint Planning Meetings.</li> </ul>	<ul style="list-style-type: none"> <li>•Sprint Planning is conducted once every week or so.</li> </ul>
<ul style="list-style-type: none"> <li>•Only members involved in prioritizing backlog product list attend the Backlog Refinement meeting.</li> </ul>	<ul style="list-style-type: none"> <li>•Sprint Planning meeting involves the Scrum Team, Scrum Master, Product Owner and Stakeholders as well.</li> </ul>
<ul style="list-style-type: none"> <li>•In the Backlog Refinement meeting, backlog products are estimated and discussed to prepare an approachable backlog list.</li> </ul>	<ul style="list-style-type: none"> <li>•The prepared backlog list is discussed and estimated for the demo purpose in the Sprint Planning meeting.</li> </ul>
<ul style="list-style-type: none"> <li>•The products are prioritized based on a prepared backlog list to discuss in Sprint meetings.</li> </ul>	<ul style="list-style-type: none"> <li>•The prioritized products are set for a demo to collect feedback and seek scope for improvement.</li> </ul>
<ul style="list-style-type: none"> <li>•Backlog Refinement makes it easier to conduct a Sprint meeting and increase work efficiency.</li> </ul>	<ul style="list-style-type: none"> <li>•Sprint Planning keeps the backlog clean and healthy without any mess so that the products are delivered on time and with full customer satisfaction.</li> </ul>

# Scrum Retrospective Methods

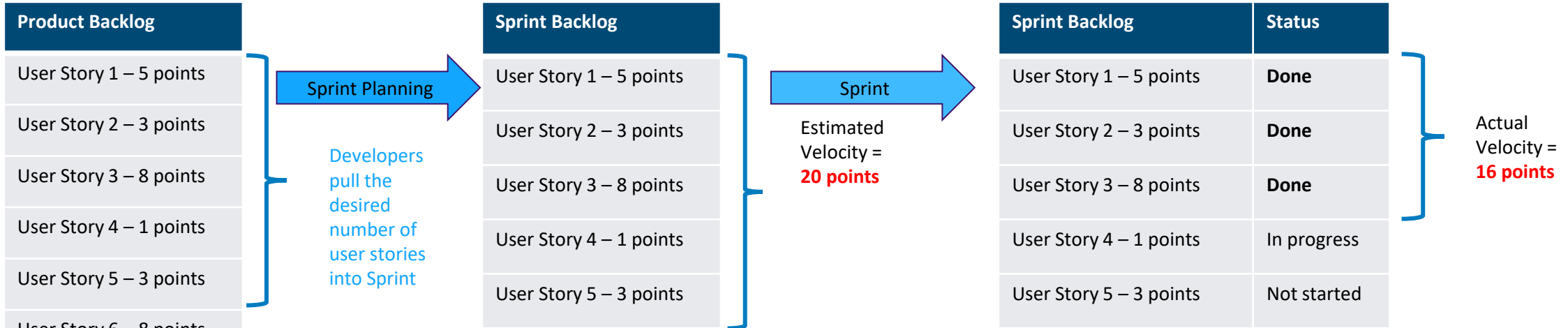


- There are a lot of techniques we can use in Retrospective such as KPT, 4Ls, Start/Stop/Continue, Mad/Sad/Glad
- Tools: whiteboard, sticky note, pen, online tools
- The most important condition is **“everybody voice”**

# Scrum

## Velocity and Capacity

► Velocity is the measure of the average amount of the work that Scrum Team can complete during a Sprint



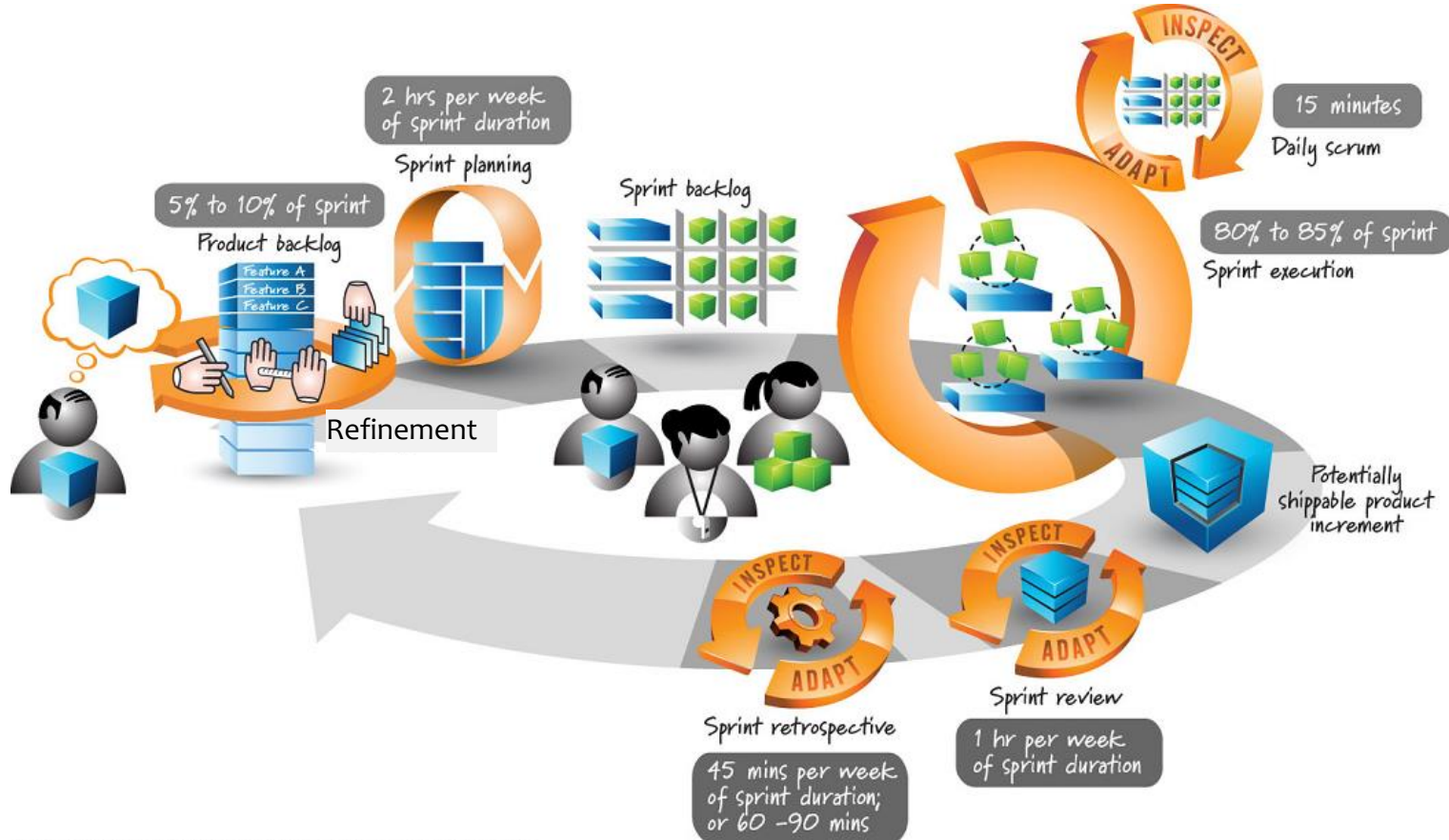
Each user story has an estimated number of points as a measure of required effort to complete

$$\text{Velocity} = \frac{\text{Sum of the velocity of } x \text{ Sprints}}{\text{Number of Sprints } (x)}$$

$$\text{Capacity} = \text{Number of Developers} * 6-8 \text{ hours} * \text{days of Sprint} - \text{day off} * (\text{Developers})$$

# Scrum

## Scrum Events and Pillars



Copyright © 2012-2018, Kenneth S. Rubin and Innolution, LLC. All Rights Reserved.

# Part III: Development Flow



**BOSCH**

Parkhaus



# NEFF's Agile Basic Process Overview



### Set up/Training

- Agile/Scrum
- Technology
- Testing
- Situation Board
- Kanban Board

Account Preparation

- Create Product Backlog
- GUI Framework
- APP Framework

### Sprint 0

- Development Environment
- Set up Scrum Events
- Product Backlog Estimation
- Release Plan

- Rules of Team
- Spike
- Product Roadmap
- 1<sup>st</sup> Backlog Refinement

Team Building

### Sprint 1-N

- Sprint Planning 1
- Backlog Refinement 1
- Sprint Retrospective

DSM

- Sprint Planning 2
- Backlog Refinement 2
- Sprint Review
- Internal Retrospective

### Release

- UAT
- Regression
- Automation
- Release Document
- Release

# *Scrum Events*

# Scrum Events

## Sprint Planning

Sprint Planning - Part 1		
<b>Goal</b>	Determine the items of the Product Backlog targeted in the current Sprint development.	
<b>Time</b>	2 hours (In case of a three-week Sprint) * To be conducted at the beginning of a new Sprint development	
<b>What to prepare</b>	<i>PO</i>	<ul style="list-style-type: none"> <li>Update and bring the Product Backlog (item content, order, etc.)</li> <li>Bring a Release Burn Down/Up Chart (visualizing transitions in velocity)</li> </ul>
	<i>Team</i>	<ul style="list-style-type: none"> <li>Assure that the upper-level items meet the definition of Ready</li> </ul>
	<i>SM</i>	Facilitate
<b>How to proceed</b>	<ul style="list-style-type: none"> <li>PO speaks about the updated parts since the previous Product Backlog Refinement</li> <li>PO and the team agree on the scope of the Sprint. (Facilitated by the SM)</li> <li>PO agrees on the effort rate (80% capacity of Sprint)</li> </ul>	
<b>Deliverables</b>	Product Backlog which the scope of development of this Sprint is clarified	

# Scrum Events

## Sprint Planning

### Precautions

- Items that do not meet the requirements of what has been defined in Ready shall not be considered as development targets
- Assure statements such as “it can be done up to this point “ by the PO or SM do not interfere with the team’s judgment
- Discuss while recognizing that the accuracy of the estimation from 1<sup>st</sup>, 2<sup>nd</sup>, and 3<sup>rd</sup> Sprints are low

# Scrum Events

## Sprint Planning

Sprint Planning - Part 2			
<b>Goal</b>	Assure mutual understanding as well as creating estimates regarding the items that were included within the scope of development in part 1		
<b>Time</b>	4 hours (In case of a three-week Sprint) *To be conducted as soon as possible after completing the Part 1		
<b>What to prepare</b>	<table border="1"> <tr> <td><i>Team</i></td> <td> <ul style="list-style-type: none"> <li>Bring the planning poker card and the Product Backlog which reflects the results of part 1</li> <li>Check public holidays and scheduled vacations</li> </ul> </td> </tr> </table>	<i>Team</i>	<ul style="list-style-type: none"> <li>Bring the planning poker card and the Product Backlog which reflects the results of part 1</li> <li>Check public holidays and scheduled vacations</li> </ul>
<i>Team</i>	<ul style="list-style-type: none"> <li>Bring the planning poker card and the Product Backlog which reflects the results of part 1</li> <li>Check public holidays and scheduled vacations</li> </ul>		
<b>How to proceed</b>	<ul style="list-style-type: none"> <li>✓ Divide Product Backlog items selected from part 1 into tasks.</li> <li>✓ While sharing a rough implementation method for each task, estimate time</li> <li>✓ Consult with the PO when the estimated workload of the divided tasks result in excess or deficiency</li> </ul>		
<b>Deliverables</b>	Group of tasks with the time estimate		
<b>Precautions</b>	<ul style="list-style-type: none"> <li>The PO does not have to participate in the meeting, however, he/she should be available for consultation as well as answering questions from the team</li> <li>Notes and other materials created at the time of the design study shall be kept as photographs or made into documents</li> <li>Each task should be divided into less than 8 hours</li> <li>Clarify the state in which each task can be considered as complete</li> </ul>		

# Scrum Events

## Daily Scrum

<b>Daily Scrum</b>			
<b>Goal</b>	Team to synchronize activities and create a plan for the next 24 hours.		
<b>Time</b>	The Daily Scrum is a 15-minute time-boxed event for the developers		
<b>What to prepare</b>	<table border="1"> <tr> <td><i>Team</i></td> <td> <ul style="list-style-type: none"> <li>Organize the content of the presentation in order to avoid over-thinking when making the actual presentation</li> </ul> </td> </tr> </table>	<i>Team</i>	<ul style="list-style-type: none"> <li>Organize the content of the presentation in order to avoid over-thinking when making the actual presentation</li> </ul>
<i>Team</i>	<ul style="list-style-type: none"> <li>Organize the content of the presentation in order to avoid over-thinking when making the actual presentation</li> </ul>		
<b>How to proceed</b>	<ul style="list-style-type: none"> <li>✓ Each developer member answer 3 questions mentioned below <ul style="list-style-type: none"> <li>❖ ① What has been done since the last daily scrum,</li> <li>❖ ② What should be done until the next daily scrum,</li> <li>❖ ③ Issues/impediment</li> </ul> </li> <li>✓ Update the Sprint Burn Down Chart</li> </ul>		
<b>Deliverables</b>	Latest Kanban (including the Sprint Burn Down Chart)		
<b>Precautions</b>	<ul style="list-style-type: none"> <li>It shall be a 15-minute time box held every day at the same time and location</li> <li>PO/SM may participate however assure that it won't become a progress report to the PO/SM</li> <li>Studies regarding problem solving shall be carried out separately from the daily scrum</li> </ul>		

# Scrum Events

## Backlog Refinement

Product Backlog Refinement							
<b>Goal</b>	Update the information of the top 2-3 Sprints of the Product Backlog items and prepare for the next Sprint						
<b>Time</b>	2-4 hours(In case of 3 weeks Sprint)						
<b>What to prepare</b>	<table border="1"> <tr> <td><i>PO</i></td> <td> <ul style="list-style-type: none"> <li>Bring the updated (item content, order, etc.) Product Backlog</li> </ul> </td> </tr> <tr> <td><i>Team</i></td> <td> <ul style="list-style-type: none"> <li>Look over the Product Backlog and summarize suggestions and points to be confirmed</li> <li>UI designers bring materials (sketches, etc.) that will help the understanding of the participants</li> </ul> </td> </tr> <tr> <td><i>SM</i></td> <td>Facilitate</td> </tr> </table>	<i>PO</i>	<ul style="list-style-type: none"> <li>Bring the updated (item content, order, etc.) Product Backlog</li> </ul>	<i>Team</i>	<ul style="list-style-type: none"> <li>Look over the Product Backlog and summarize suggestions and points to be confirmed</li> <li>UI designers bring materials (sketches, etc.) that will help the understanding of the participants</li> </ul>	<i>SM</i>	Facilitate
	<i>PO</i>	<ul style="list-style-type: none"> <li>Bring the updated (item content, order, etc.) Product Backlog</li> </ul>					
	<i>Team</i>	<ul style="list-style-type: none"> <li>Look over the Product Backlog and summarize suggestions and points to be confirmed</li> <li>UI designers bring materials (sketches, etc.) that will help the understanding of the participants</li> </ul>					
<i>SM</i>	Facilitate						
<b>How to proceed</b>	<ul style="list-style-type: none"> <li>✓ PO talks about the updated sections of the Product Backlog</li> <li>✓ Ask questions, make proposals, split items and confirm risks regarding the top items to fulfill the definition of Ready</li> <li>✓ If top items do not meet the definition of Ready, decide the deadline and person in charge for the problem to be solved</li> </ul>						
<b>Deliverables</b>	<ul style="list-style-type: none"> <li>A Product Backlog with the top items meeting the definition of Ready</li> </ul>						
<b>Precautions</b>	<ul style="list-style-type: none"> <li>Consider whether it is possible to start some parts of the task which you don't have past experience</li> <li>Make a re-estimation if you realize there is a problem with the initial estimation</li> </ul>						

# Scrum Events

## Sprint Review

Sprint Review			
<b>Goal</b>	Judge the acceptability of the Product Backlog items which were decided during the Sprint planning meeting, Confirm release plan		
<b>Time</b>	3 hours (In case of 3 weeks Sprint)		
<b>What to prepare</b>	<table border="1"> <tr> <td><i>Team</i></td> <td> <ul style="list-style-type: none"> <li>Prepare demonstration (Bring Product Backlog and Release Burn Down Chart)</li> </ul> </td> </tr> </table>	<i>Team</i>	<ul style="list-style-type: none"> <li>Prepare demonstration (Bring Product Backlog and Release Burn Down Chart)</li> </ul>
<i>Team</i>	<ul style="list-style-type: none"> <li>Prepare demonstration (Bring Product Backlog and Release Burn Down Chart)</li> </ul>		
<b>How to proceed</b>	<ul style="list-style-type: none"> <li>✓ Present a demonstration of the completed items developed in this Sprint, thereafter to be determined OK or NOT by the PO</li> <li>✓ Share feedback from PO, team, and stakeholders</li> </ul>		
<b>Deliverables</b>	<ul style="list-style-type: none"> <li>Product Increment, Product Backlog with the result of the acceptance judgment and updated Release Burn Down Chart</li> </ul>		
<b>Precautions</b>	<ul style="list-style-type: none"> <li>Prepare data as well as double-check the order of the demonstration in advance, to assure the demonstration goes smoothly</li> <li>Do not include items which do not meet the requirements defined as Done</li> <li>Make sure to save the acceptance test cases of the PO due to them being necessary material for the release decision meeting</li> </ul>		



# Scrum Events

## Sprint Retrospective

Sprint Retrospective			
<b>Goal</b>	Create an improvement plan based on confirmed problems of the scrum team		
<b>Time</b>	2.5 hours (in case of 3 weeks Sprint) <i>*Is conducted after the Sprint review meeting and before the next Sprint planning meeting</i>		
<b>What to prepare</b>	<table border="1"> <tr> <td><i>Team</i></td> <td> <ul style="list-style-type: none"> <li>Bring the previous improvement plan, Sprint Burn Down Chart as well as the definition of Ready and Done</li> </ul> </td> </tr> </table>	<i>Team</i>	<ul style="list-style-type: none"> <li>Bring the previous improvement plan, Sprint Burn Down Chart as well as the definition of Ready and Done</li> </ul>
<i>Team</i>	<ul style="list-style-type: none"> <li>Bring the previous improvement plan, Sprint Burn Down Chart as well as the definition of Ready and Done</li> </ul>		
<b>How to proceed</b>	<ul style="list-style-type: none"> <li>✓ Confirm if previous action plans for improvement have been achieved</li> <li>✓ Present the good points (Keep), issues (Problem) and improvement proposals (Try) respectively of the current Sprint</li> <li>✓ Participants consider and propose improvement plans in regard to current issues</li> <li>✓ Decide which action should be taken immediately and which should be taken in the next Sprint, then break them down into tasks</li> <li>✓ Update the definition of Ready and Done if improvements can be made</li> </ul>		
<b>Deliverables</b>	<ul style="list-style-type: none"> <li>The Improvement action tasks (at least 1 action related to process) and the updated definition of Ready and Done</li> </ul>		
<b>Precautions</b>	<ul style="list-style-type: none"> <li>Make sure that the improvement actions are achievable</li> <li>Store the results of the Sprint retrospective meeting as documents and photographs</li> </ul>		

# THANK YOU

LET LEARNING  
LET WORKING  
LET SCRUMMING