Software Process & Agile Process

Dr. Samuel Cho, PhD

NKU

Core ideas of this module

Core Idea 1

- We develop a Web application that stores information on the server and displays the information on the client.
- We use SE rules and SE tools in a team for the design and implementation of the application.

Core Idea 2

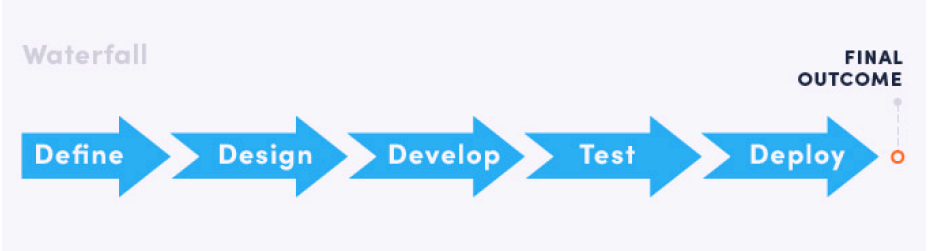
- We use a tool software process with team rules to build and deliver the application on time and within budget.
- We use Agile Software Process to accomplish the goal.

1. Software Process - Stages & SE Box
2. Agile Process & Scrum Framework

Software Process - Stages & SE Box

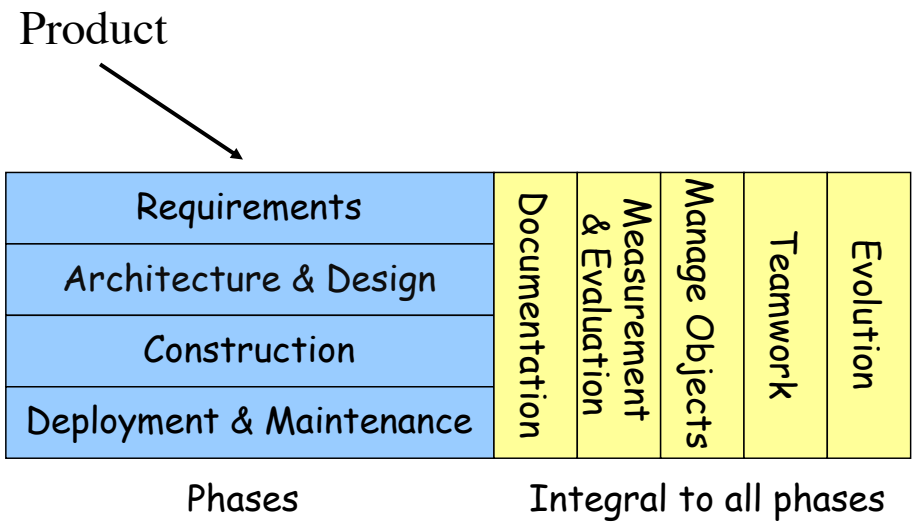
Water Process Model

- We discussed a waterfall process model.
- In this model, we (1) define, (2) design to (3) develop software.
- Then we (4) test the software before we (5) deploy it.



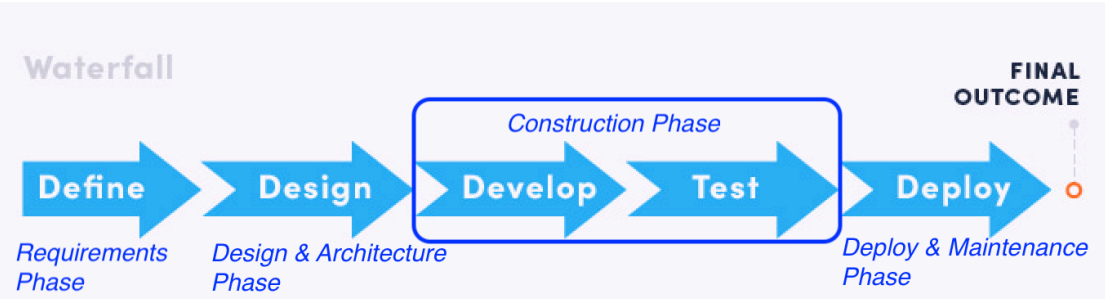
SE Box

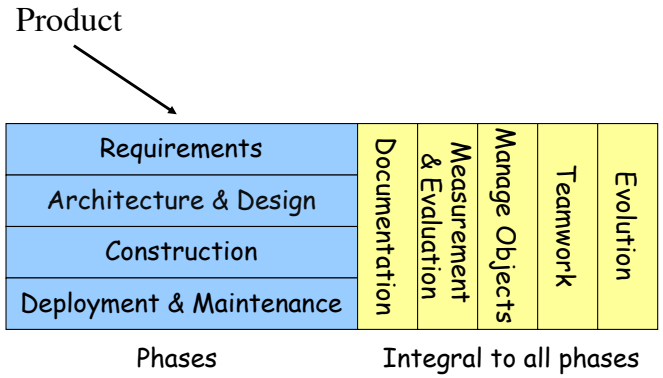
- This is a diagram that interprets the process from a different perspective.
- In this diagram, we need (1) activities in phases and (2) activities integral to all phases to build a 'product'.
- We call this diagram 'SE box'.



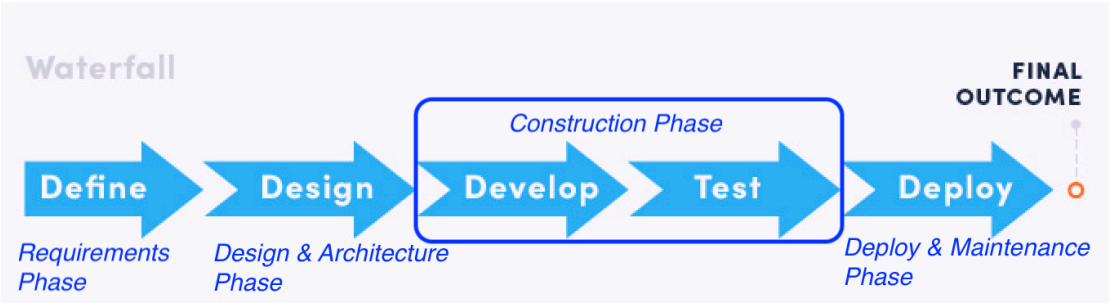
Water Process Model and SE Box

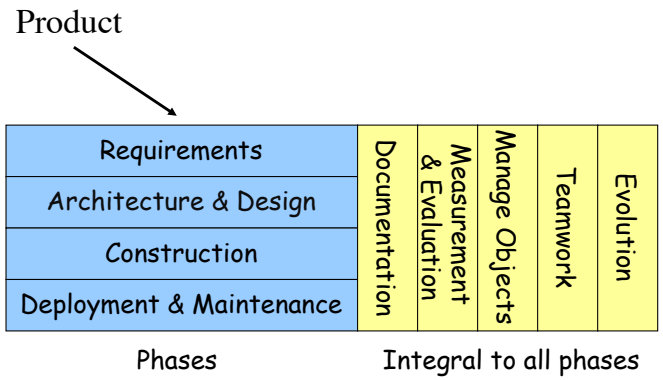
- The waterfall process model focuses on activities, but the SE box focuses on phases that contains activies.
- We can connect the waterfall model to the SE box.



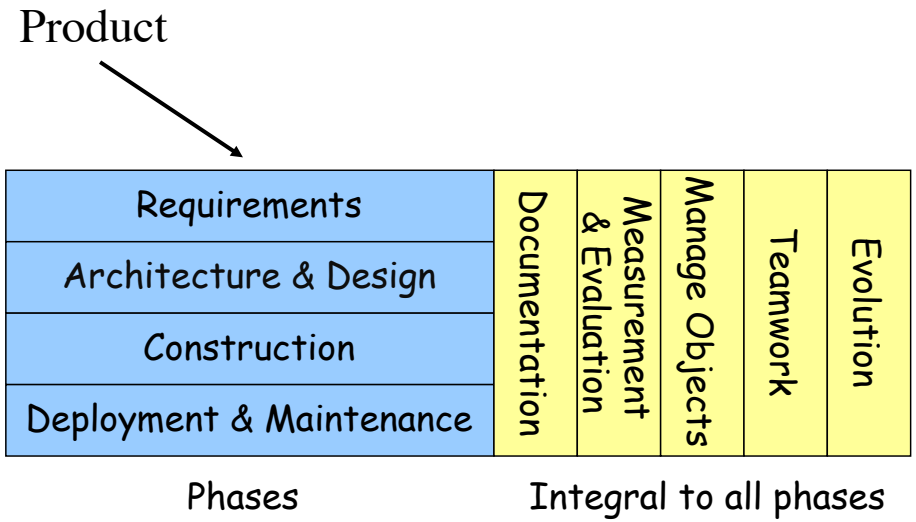


- We need 'requirements' phase to 'define' our goal.
- 'Architecture' is 'high-level design.' In SE Box, we have both 'architecture' and 'design' phases to specify our features.
- We integrate 'develop' and 'test' into a 'Construction' phase.
- We deploy the software we construct in the 'Deployment & Maintenance' phase.

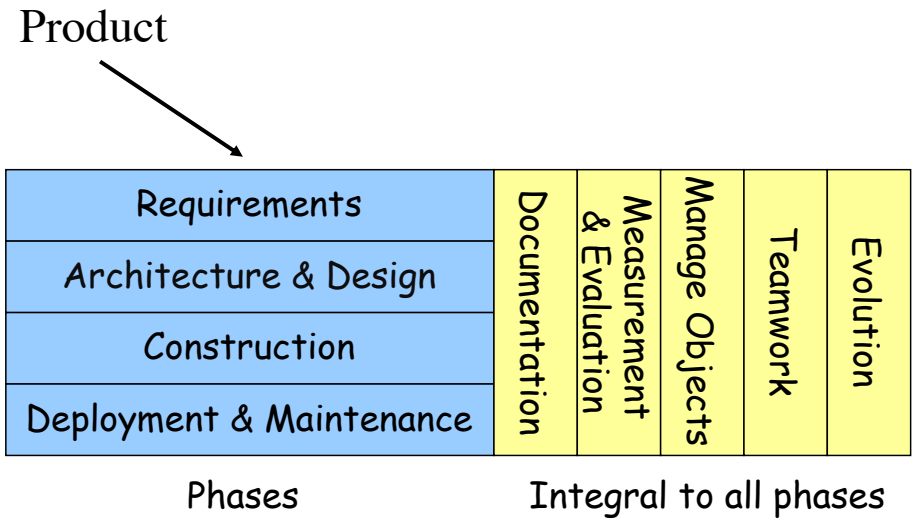




- We have activities that are integral to all phases.
- The software is keep changing, in other words, software evolves.
- We need a team to build and deliver software products. (It is practically impossible nowadays to have a one-man to do the design, develop, marketing, advertising, and sales.)



- We should manage objects (artifacts) such as design and requirement documents, code, test, and schedule files.
- We should measure any objects (artifacts) and measure to assess progress.
- We need to make document for any SE activities.
- We can use the acronym "ET's MOME is Doc" to memorize these activities.



SE Rules Applied

Applied SE Rules #Process

- Divide and Conquer.
- Only Fools Rush In - We don't construct unless we know what to construct: Requirements and Architecture/Design are required to know what to construct.

- What else?

Agile Process & Scrum Framework

Rumsfeld's Law

Rule #Rumsfeld's Law

Initially, we don't know that we don't know (unknown unknowns). Then, we know what we don't know (known unknowns). Finally, we know that we know (known knowns).

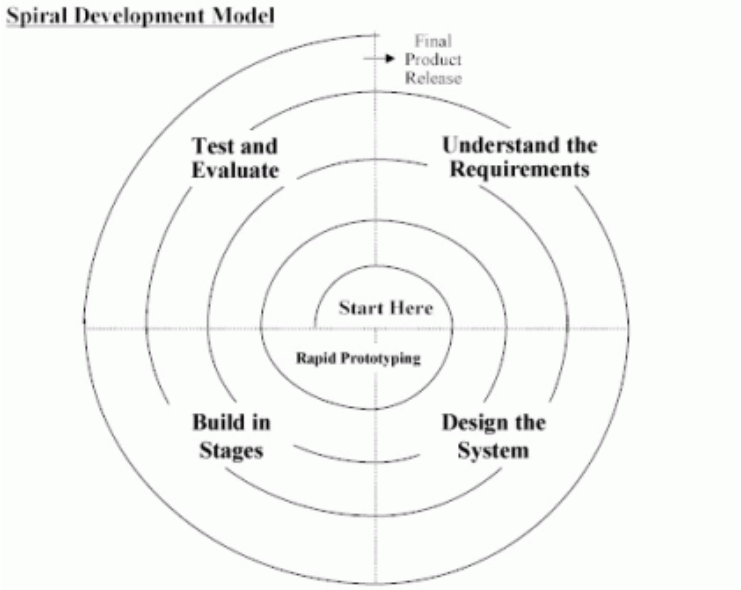
- When we start the project, we don't know what we don't know, but we soon find out the known unknowns as we build the software.
- As a result, we always find something to update (change) when we finish the project.

Changes in SE

- Also, any aspects of software changes include the following.
- Users require more features.
- Programmers have a new version of libraries.
- Users install a new version of operating systems.
- Hardware companies always introduce new hardware features that software should support.
- We cannot avoid the changes in SE.

Spiral Process Model

- We have a Spiral Process Model to incorporate the idea of "changes" in the Software Process Model.
- In this model, we iterate the phases (Requirements, Design, Development/Test, and Deploy) to adapt to the changes.



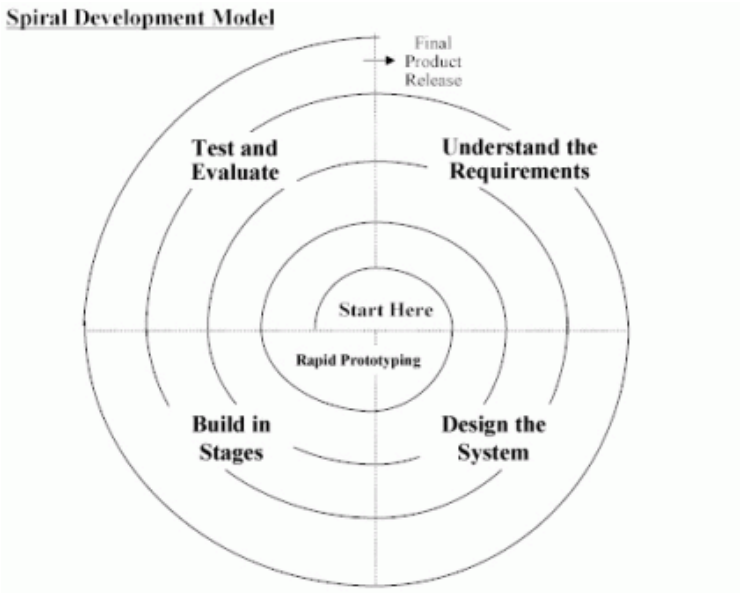
Kaizen/Vertical Slice Rule

Rule #Kaizen/Vertical Slice

Make it work and make it better.

- Make it work with the simplest implementation first and make it better over and over again.
- Instead of making a full version with all the features module by module, we make the simplest working version first. Then, we revise it iteratively.

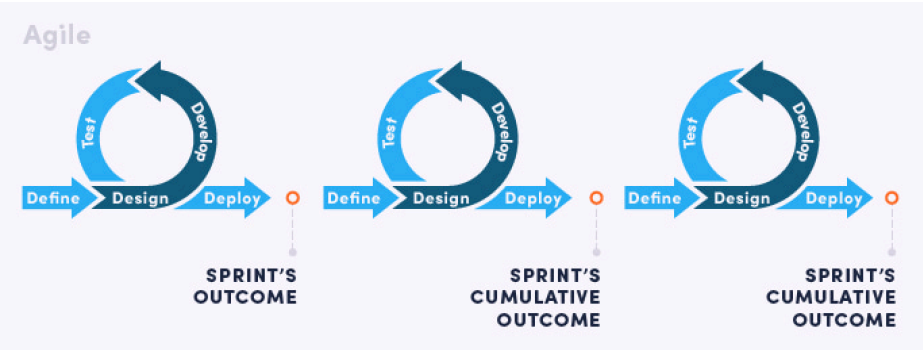
- In this model, we deploy/release a new software version about once per year.
- In some cases, it can take years to release a new version of the software.
- It is too slow in many software development situations wherein requirements and environments are keep changing.



Agile Process Model

- Software Engineers developed a new process model, Agile Process Model, to address this issue.
- In this model, we iterate the phases more frequently in a short amount of time (about two weeks, but it can be two months or more depending on the situation).
- We call the iteration of phases a "Sprint."

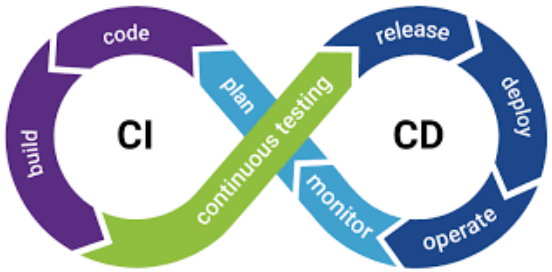
- When we develop software using the Agile Process Model, we can effectively adjust to any changes.
- We can deliver software products more frequently to check users' responses.
- We can add features more frequently from the users' responses.
- We call this concept CI/CD.



CI/CD

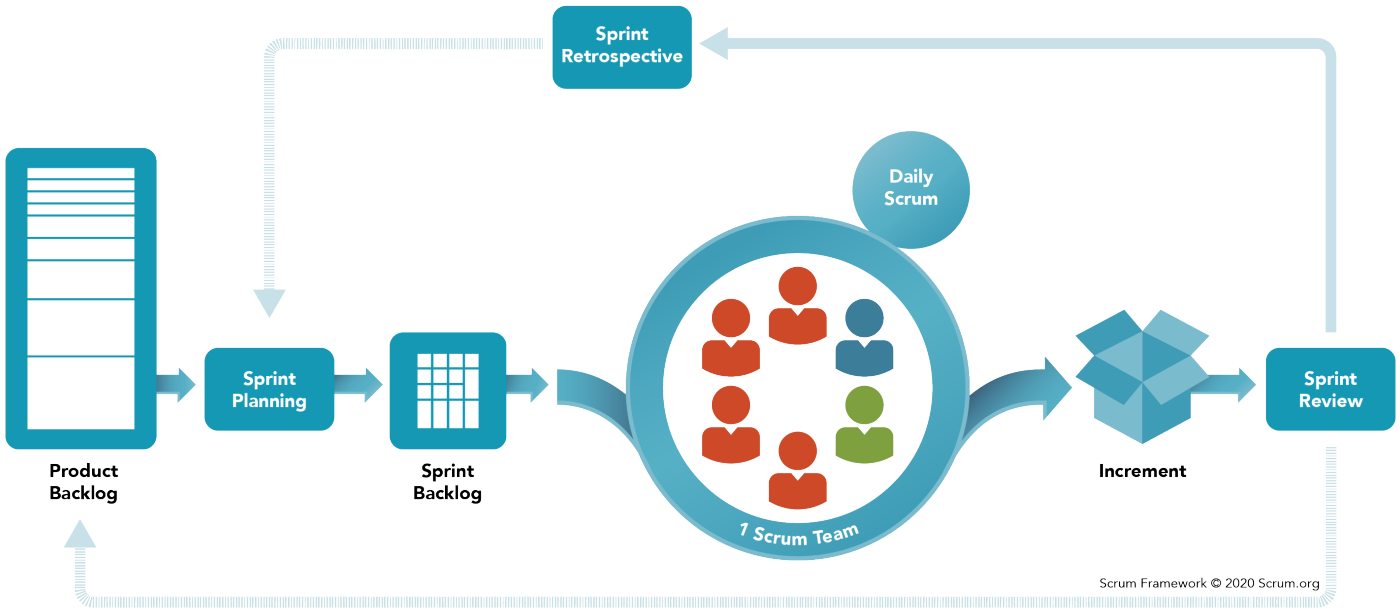
Idea #CI/CD

- CI/CD is the combined practices of continuous integration and either continuous delivery or continuous deployment.
- CI/CD bridges the gaps between development and operation activities and teams by enforcing automation in building, testing and deployment of applications.



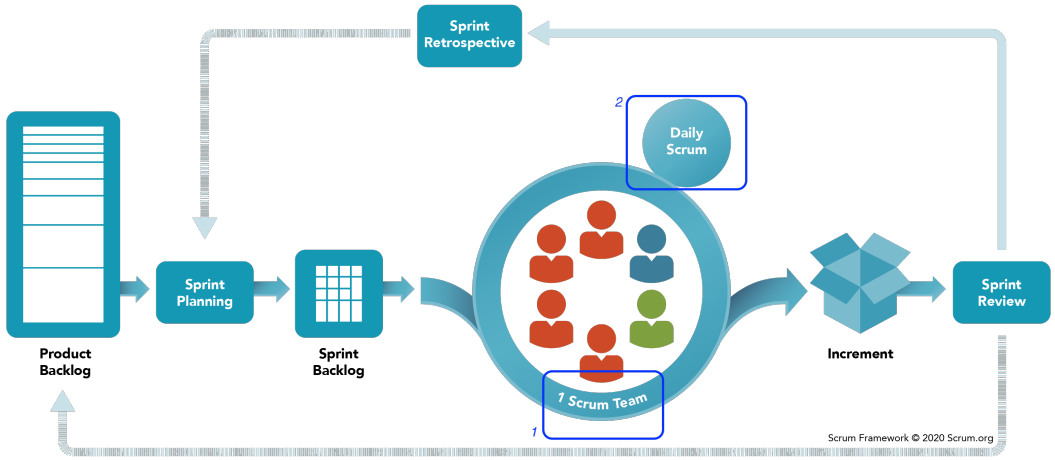
Scrum as Agile Process

- There are many ways to do the Agile Process.
- In this course, we choose to use Scrum.
- Scrum is a framework (a set of tools) for applying Agile Process Model to develop software.

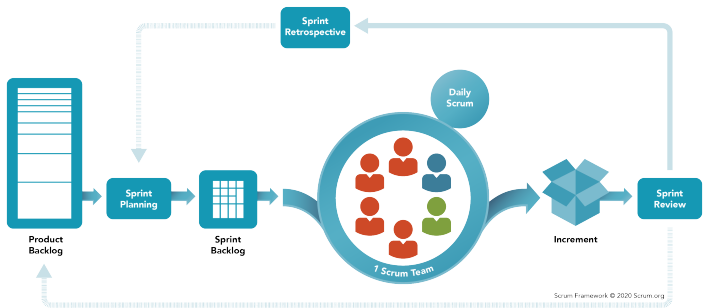


We need to understand new terminology when using Scrum.

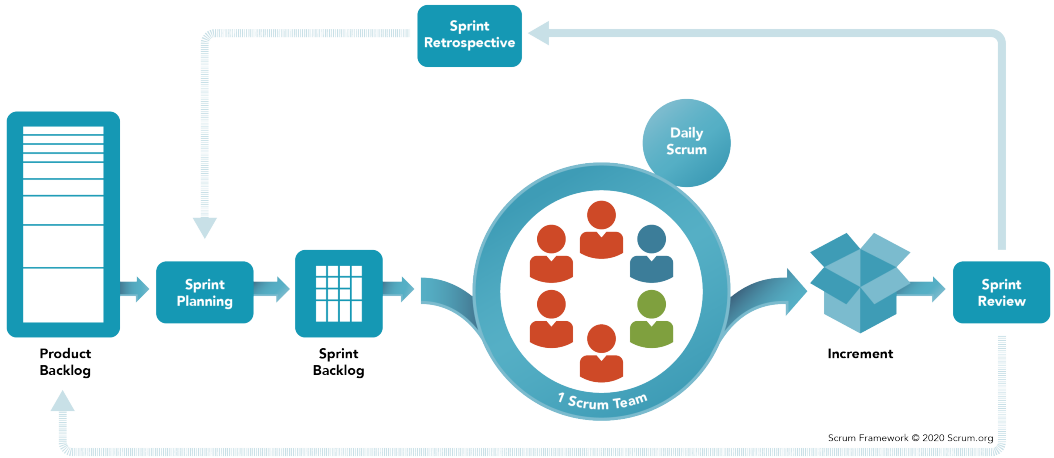
1. Scrum Team is a team that uses the Scrum framework to develop and deliver software.
2. A Scrum Team members do a daily scrum to check (1) what is done, (2) what are issues, and (3) what will be done.



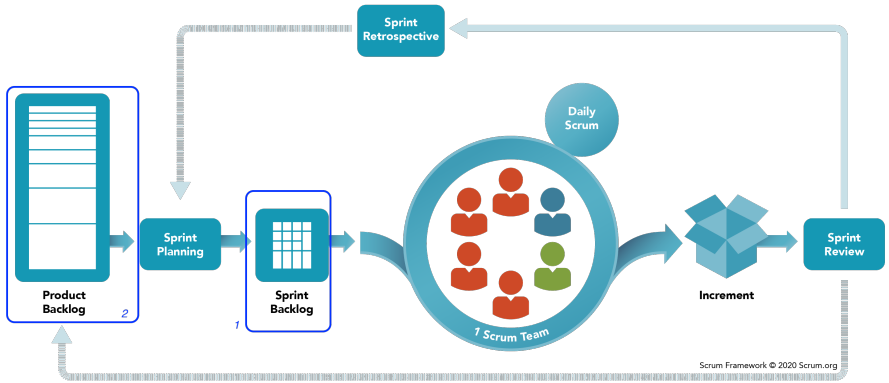
- A Scrum Master is a team leader to manages all the aspects of Scrum.
- Stakeholders are any people—team members, managers, users, or anyone—who are involved in any of the Scrum activities.
- A Product Owner is the lead user of a system. They have a deep understanding of users, the marketplace, competitors, and trends. We can think of them as managers.



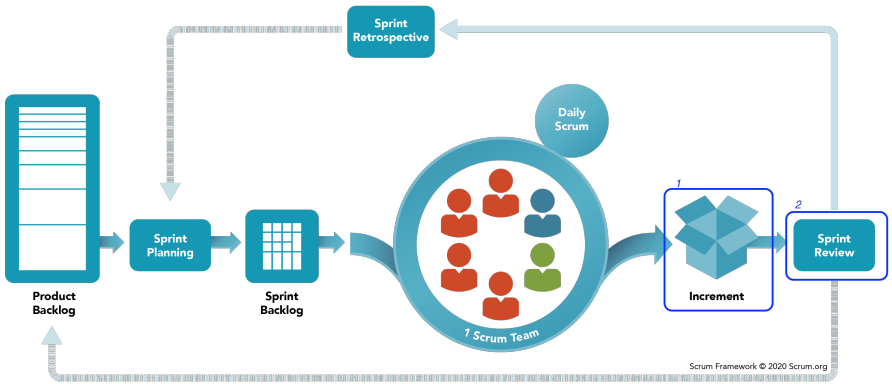
- Value is what Scrum teams are pursue to produce.
- It is normally software products and/or related artifacts, but it can be anything depending on the environment/situation.



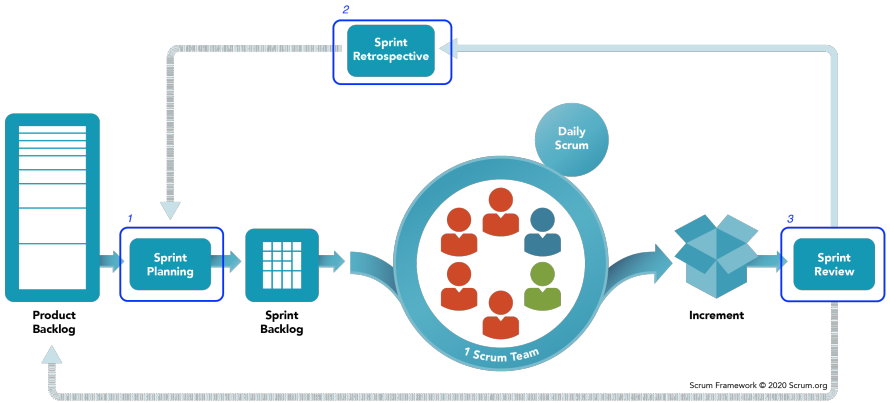
1. Sprint Backlog is a plan by and for the Developers. We can think of it as a "to do list" for a Sprint.
2. Product Backlog is a prioritized list of work for the development team that is derived from the roadmap and its requirements. The most important items are shown at the top of the product backlog, so the team knows what to deliver first.



1. The Increment is the sum of all the Product Backlog items completed during a Sprint and all previous Sprints.
2. A Sprint team can update the Product Backlog from the Increment through Sprint Review.



1. In a Sprint Planning, a Scream team selects what should be done in this Sprint from Sprint Perspective and Product Backlog.
2. The Sprint Retrospect is an opportunity for the Scrum Team to inspect itself and create a plan for improvements to be enacted during the next Sprint.
3. The purpose of the Sprint Review is to inspect the outcome of the Sprint and determine future adaptations.



Definition of Scrum

Let's define Scrum using the new terminology.

Definition #Scrum

Scrum is a lightweight framework that helps people, teams and organizations generate value through adaptive solutions for complex problems.

Scrum Process

Idea #Scrum Process

Scrum requires a Scrum Master to foster an environment where:

1. A Product Owner orders the work for a complex problem into a Product Backlog.
2. The Scrum Team turns a selection of the work into an Increment of value during a Sprint.
3. The Scrum Team and its stakeholders inspect the results and adjust for the next Sprint.
4. Repeat

Software Development (SD) is another form of software

Idea #Software development is software

- It is interesting to observe that the software development itself is software.
- Just like we choose from many programming languages and tools for making software, we can choose from software process models to develop the software.

SE Rules Applied

Applied SE Rules #Agile

- Rumsfeld's law - We don't know what we don't know in the beginning, and we begin to know them iteratively.
- Kaizen (Vertical Slice) - Make it work in the first iteration, and then make it better over and over again.

- What else?