**POM 05 – SCRUM I**

*Controlling software development with a process:*

How do we control software development? Two opinions:

1. Through **organizational maturity** (Humphrey 1989)

Repeatable process, Capability Maturity Model Integration (CMMI)

1. Through **agility** (Schwaber 2001):

Large parts of software development is empirical in nature; they cannot be modeled with a defined process

*Defined vs. empirical process:*

* **Defined process:**
  + Planned, follows strict rules
* **Empirical process:**
  + Not entirely planned inspect and adapt

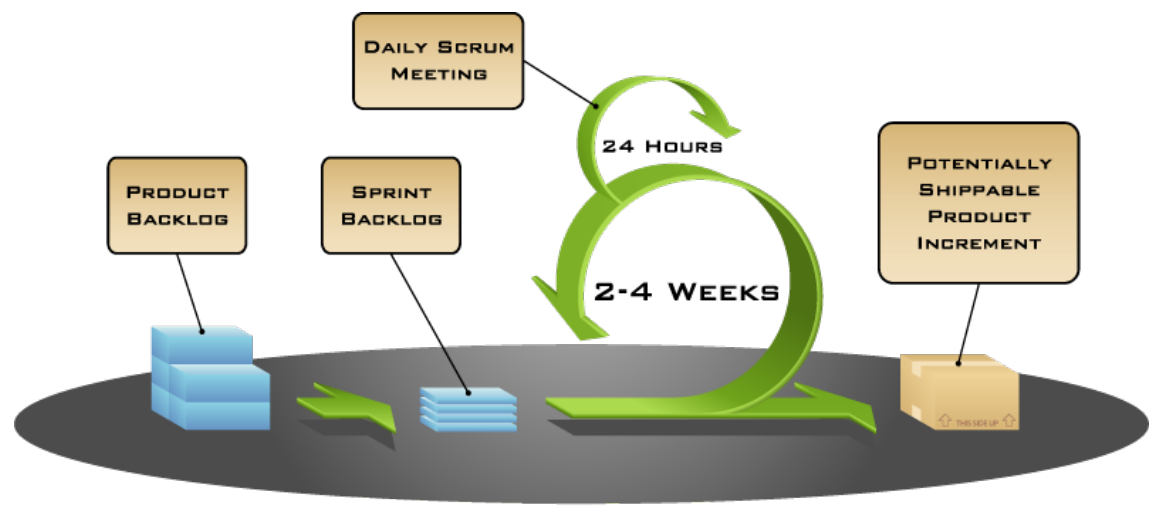
*Defined process control model:*

* Requires that every piece of work is completely understood
  + **Deviations are seen as errors that need to be corrected**
* Given a well-defined set of inputs, the same outputs are generated every time
  + **Precondition** to apply this model: all the activities and tasks are well defined to provide their repeatability and their predictability
  + If the preconditions are **not satisfied**: lots of surprises, loss of control, incomplete or wrong work products

*Empirical process control model:*

* Imperfectly defined process, not all pieces of work are completely understood
  + **Deviations, errors and failures are seen as opportunities that need to be investigated**
* **Expects the unexpected**: control and risk management is exercised through frequent inspection
* When to apply this model:
  + Change is frequent and cannot be ignored
  + Change of requirements, change of technology
  + Change in the organization, change of people

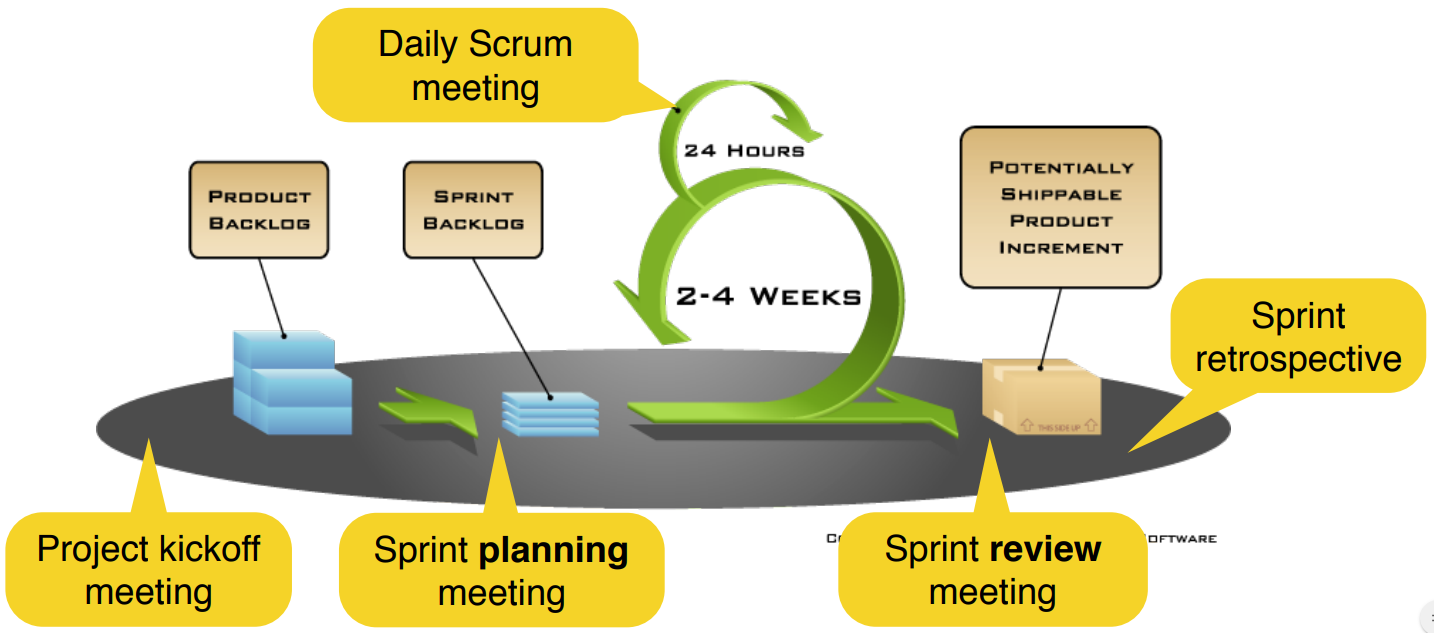
*Example of an empirical process: Scrum:*



*3 Scrum artifacts:*

1. **Product backlog**: List of requirements for the whole product
2. **Sprint backlog**: List of requirements and tasks for one iteration (“sprint”)
3. **Potentially shippable product increment**: Release to the product owner that contains all results of the current sprint

*Scrum meetings:*



*5 Scrum meetings:*

1. **Project kickoff meeting** (start of the project): Create and prioritize the product backlog
2. **Sprint planning meeting** (start of each sprint): Create the sprint backlog
3. **Daily scrum meeting** (every day, 15min): Share status, impediments and promises (in a standup meeting)
4. **Sprint review meeting** (end of each sprint): Demonstrate the realized backlog items to the product owner (and other stakeholders)
5. **Sprint retrospective** (after the sprint): Inspect the previous sprint and create a plan for improvements to be enacted during the next sprint (sometimes combined with the sprint review meeting)

*Scrum team with 3 roles:*

1. Product owner (defines the product, responsible for results)
2. Scrum master (resolves impediments, responsible for the process)
3. Developer (Development team – Self-organizing and cross-functional, realizes the product increment)

*Sprint:*

* Time-box during which the team creates a potentially shippable product increment
* Sprints have consistent durations throughout a project
* A new sprint starts immediately after the conclusion of the previous Sprint
* Typically 2-4 weeks long
* Starts with **sprint planning meeting**
  + Create the Sprint Backlog: Selection of items to be implemented in the Sprint
  + Important: Development Team and Product Owner select the items together
* Ends with **sprint review meeting**
  + Release and deliver the application (product increment)
  + Demo of the application during the meeting
  + Important: Product Owner gives feedback
* Sprint review meeting and sprint planning meetings are typically combined into a single meeting with the product owner
* The Scrum team can additionally perform a sprint retrospective meeting
* The development team
  + Realizes the items in the sprint backlog
  + Uses e.g. a task board to visualize the status of these items
* The Scrum master visualizes the progress, e.g. in a burn down chart

*Product backlog:*

* Collection of items (typically requirements, e.g. user stories, scenarios, etc.) prioritized by the product owner
* The product backlog can always be changed and reprioritized during the projects
* Created on the basis of the problem statement during the project kickoff meeting or in the phase before the actual project starts

*Priority:*

* priority describes the importance of the requirement for the software
* Examples for a priority scheme (that we also use in JIRA)
  + **Prio 1 = Critical** (Candidates for the first development sprints, must be part of the first product increment)
  + **Prio 2 = Major** (Must be realized within the project, can be realized in one of the following product increments)
  + **Prio 3 = Minor** (Desirable, if there is enough time)
  + **Prio 4 = Not Important** (Might not be realized at all)
* Priorities of requirements can change during the project in Scrum
* **Prioritization is done by the product owner**

*Estimation with level of difficulty:*

* Estimation is not easy for developers —> in particular beginners struggle with it
* **Simplified estimation**: T-Shirt sizes represent the rough difficulty of the realization of the backlog item
  + Small (**S**)
  + Medium (**M**)
  + Large (**L**)
  + Extra Large (**XL**) —> too big, split it into smaller issues!
* **More sophisticated estimation**: In the sprint planning meeting, the team decides on the difficulty of the backlog items by estimating story points (1/2, 1, 2, 3, 5, 8, 13, 21, 100) using e.g.:
  + Poker planning
  + Team estimation game
  + Both are agile estimation techniques (later more)