***Lecture from 25.10.2018***

***Software Engineering 2***

***Agile approaches***

*Be agile, don’t do Agile*

We should be able to change with the client needs.

Agile technologies came from early times. SCRUM is one example. Agile approaches created with the idea in mind that the client will change his requirements through time.

Agile manifesto – 12 rules for agile software engineering.

**What people said about Agile**

1. Reduced time-to-market
2. Increased quality
3. Reduced waste
4. Better predictability
5. Better morale

The client has the opportunity to take part in the development of the program. That way the possibility for miscommunications lesser.

**Agile methods**

1. Scrum 52 %
2. Dynamic Systems Development Method 14 %
3. Custom Hybrid 9 %
4. Don’t know 8 %
5. Others the

**Agility definition**

*Agility is the ability to both create and respond to change in order to profit in the ever changing business area.*

**Agility Characteristics**

* **Characteristics**
* Iterative and incremental
* Small release
* Collocation
* Release plan/feature backlog (a list with functional characteristics)
* Iteration plan/task backlog

**Waterfall vs Agile**

In Scrum iterations are called iterations. Iterations usually are 4, 6 or 8 weeks. Usually they are with the same time length.

Waterfall approach – design, spec, code, UAT, launch in a single iteration

Agile approach – user stories. The user gives you a feedback from a scenario he went through or as it is called his client history. This is something we do in every single iteration.

**The agile manifesto (2001) - A statement of values**

Agile vs Classical methods

1. Agile practice favors: Individuals and interactions over Process and tools (classical way)
2. Working product – in every iteration we have working product (in Agile) over Comprehensive documentation (classical way)
3. Customer collaboration – in every single moment we have contact with the client on every small piece of code which is written over Contract negotiation.
4. Responding to change in the business, country or client requirements over Following a plan

**What makes Agile works?**

* Better collaboration with business
* More adapted change/learning
* Communication – between clients, team
* Motivation
* Collective ownership - Every single developer is part of the product ownership
* Time boxes/Time frames for every iteration
* Inspect and adapt
* Collaction
* Information radiators
* Team autonomy for
* Focus on real thing
* The decisions are taken from the whole team.

**Principles of Agile Manifesto**

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

In the traditional analysis, design, code, test in agile methods in every iteration we have the four pieces together. These are not big four periods but, in every iteration, we go through them.

**Agile practices**

* Whole team
  + Communicate in a face-to-face fashion as much as possible
  + Collaborative workspace
  + Hear customers
  + Active in planning process
  + Wall -> commination workspace
  + Daily standup meetings (report on what to do today)
  + All team participate in product presentation to customer
  + Previously known as “On Site Customer” (2004)
* Short releases
  + About 2 months – several short iterations of 1-2 weeks
  + “Business day” – at the begging of the iteration – all stakeholders participate - presentation of “Done”, “measure”, “lesson learned”, “plan ahead”
  + At the end of iteration -> presentation to the customer
* Measures
  + Track and monitor quality
  + Light weight
  + Burndown chart
* Test-Driven Development
  + Encourage automatic test and acceptance test
  + Unit tests are written prior to code writing
  + Acceptance test – defined by customer, outline how each functionality should be tested
  + No shifting to testing department – testers are integrated

**Extreme Programming**

* Simplicity
* Communication
* Feedback
* Courage

12 supporting practices

1. Planning Game
2. Small realeases
3. Custommer acceptance tests
4. Simple Design
5. Pair Programming
6. Test-Driven development
7. Refactoring
8. Continuous integration
9. Collective Code Ownership
10. Coding Standards

**Scrum Roles**

1. Product owner
   * Product vision
   * Prioritize
   * Accept Product
2. Scrum Master
   * Facilities process
   * Supports team
   * Removes roadblocks
   * Enables collaboration
3. Team
   * Cross Functional
   * Self-organizing
   * Focus on commitments
   * Explore How

**Scrum Master**

Scrum Master watches over the scrum processes. He leads the change .

Scrum Master characteristics: Knowledgeable, questioning, Patient, Collaborative, Protective, Transparent.