Scrum and Agile Processes: Outline

- Classical processes and their limitations
- Agile processes
- Scrum
  - Overview
  - History
  - Process
- Budgeting and planning agile projects
- Where does Scrum not fit?
- Advanced questions
- Summary
Classical Process Models

- Waterfall: adapted from hardware development by DoD 1960s/70s
- Phases separated by activity:
  0. Planning
  1. Requirements analysis
  2. Design
  3. Implementation
  4. Test
  5. Maintenance
- Many refinements, e.g.:
  - V-Model
  - Boehms spiral model
    (already predecessor to iterative methods)
- PS: „Maintenance“ often 80% of overall effort ...
Where’s the problem?

Problems:

- Errors are made in every phase, including requirements specification
- Specification errors are often not detected before system is running
- Late requirement changes are nowadays more norm than exception
- Processes depending on absence of errors are doomed to fail

Effect:

⇒ Changes to requirements cause all phases to be re-done
⇒ Cost of a change is multiplied by number of phases and/or affected documents!
⇒ Approaches to fix this by increased perfection lead to again more steps and then even higher costs
“Proof”

- Research found between 40% and 70% of all SW projects failing
- Governmental SW projects require very formal, typically pre-descriptive SW processes (e.g. V-model in Germany)
- Studies report 90% of large governmental projects to fail ...
  - (There are of course other reasons as well, to be honest: E.g. large governmental projects tend to be a) complex and b) simply too ambitious.)
Non-Trivial Systems Cannot be Fully Specified both in Detail and in Advance

3-Requirements Example:

- A robot may not injure a human being or, through inaction, allow a human being to come to harm.
- A robot must obey any orders given to it by human beings, except where such orders would conflict with the First Law.
- A robot must protect its own existence as long as such protection does not conflict with the First or Second Law.

(Isaac Asimov, "I, Robot", 1942)
Agile Processes

In a complex environment, following a plan produces the product you intended, just not the product you need. (Jim Highsmith)

Agile Manifesto:

Value

- **individuals and interactions** over processes and tools
- **working software** over comprehensive documentation
- **customer collaboration** over contract negotiation
- **responding to change** over following a plan

Numerous variants, e.g.:

- Adaptive Systems Development (ASD)
- Crystal
- Scrum
- Dynamic Systems Development Method (DSDM)
- eXtreme Programming (XP)
- Feature Driven Development (FDD)
Scrum: Overview

- Agile methods are meanwhile the norm in software industry
- Scrum is by far the most widespread agile method
- Advantages compared to other agile methods (e.g. RUP):
  - More a library than a framework with less need for company specific adaptation
  - Highly standardized and thus easier to apply and to onboard new team members


- Sure enough, there is no such thing as a silver bullet!
Some Terms

- **Incremental**: iterate development and test
- **Iterative**: iterate requirements, development and test
- **Agile**: according to agile manifesto
- **Scrum**: one out of many, but recently the most successful, iterative software process, that can be practiced in an agile manner
History & References

- **1986**: Takeuchi and Nonaka describe a iterative production as a rugby approach, compared to a classical relay race approach ("The New New Product Development Game" in Harvard Business Review, Jan/Feb 86)

- **1990/91**: Ken Schwaber and Jeff Sutherland with others used such an approach at their companies and referred to it as “Scrum”

- **1995**: Sutherland and Schwaber present the "Scrum Development Process" (OOPSLA’95 Business Object Design and Implementation Workshop in Austin, Texas)

- **2001**: Schwaber and Sutherland are among 17 first signees of the Agile Manifesto
Scrum Process

Three Roles
- Scrum Master
- Product Owner
- Team

Four Ceremonies
- Sprint planning
- Daily scrum meeting
- Sprint demo
- Sprint retrospective

Three Artefacts
- Product backlog
- Sprint backlog
- Burndown chart
# The 3 Scrum Roles

<table>
<thead>
<tr>
<th>Scrum Master</th>
<th>Product Owner</th>
<th>Team</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Responsible for ensuring that the Scrum team adheres to Scrum values, practices, and rules&lt;br&gt;• Helps the Scrum Team understand and use self-organization and cross-functionality</td>
<td>• Responsible for managing, prioritizing and maintaining the product backlog&lt;br&gt;• One person, no comitee&lt;br&gt;• May integrate backlog entries from other persons</td>
<td>• Turns product backlog into increments of potentially shippable functionality every sprint&lt;br&gt;• Teams are self-organized without external interference&lt;br&gt;• Optimal size is seven people, plus or minus two</td>
</tr>
</tbody>
</table>

Scrum Team = SM + PO + Team
Optimize ROI along Business Value

Accumulated business value

Business value of stories

Optimal time of market entry

Story

$t \rightarrow$
Some Principles

- **Pull instead of push:** Tasks are not assigned to individuals, but are taken on by individuals.
- PO is always **available** for clarifying requirements, but may introduce new ideas only in form of prioritized requirements for the next sprint planning.
- The team is **self-organizing** without external interference.
- **Time boxing** both sprints and meetings: time determines scope, not the other way round.
Technical Analogy

- A pre-descriptive process corresponds to an **open-loop controller** (German “Steuerung”)
- An iterative process corresponds to a **closed-loop controller** (German “Regelung”)

Non-trivial processes require the feedback of a closed-loop control
Planning Poker

- Origins in Scrum
- Accelerates Delphi-method
- "Poker"-cards avoid undesired mutual inducement
- (Roughly) Fibonacci numbers to model progression
- Many web applications available, e.g. for distributed teams
Sprint Burndown Chart

Sample Burndown Chart

- Completed tasks
- Remaining effort
- Ideal burndown
- Remaining tasks

Remaining effort (hours)

Remaining and completed tasks

Day 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
Advanced Questions

- Where does Scrum not fit?
- Up-front planning
- Fixed price contracts
- Early phase of a new development
- Trailing efforts
- Cards & walls or tools?
- Scrum role questions
- Scaling scrum
- The tea leaves effect
Cards or Tools?

In most teams:

- Product Backlog with tool
  - Spreadsheet or (preferably) task tracker
  - Nowadays Scrum-plugins for many trackers

- Sprint Backlog:
  - Cards on the wall for co-located teams
  - Tool for distributed teams
    - Tracker, since spreadsheet does not scale
    - Must support hierarchical decomposition

- Some trackers can visually simulate a task board
The Tea-Leaves-Effect

- Tea leaves swim at the very top in the beginning
- Dwindle down inside the cup until they reach the bottom a bit later
- Similar with product backlog items
  - Real business value is re-considered
- This occurs in many if not most projects
  - How would a pre-specifying project deal with it?
Scrum: Summary of Advantages

- Meanwhile the one standard among software processes
- Easier to introduce than many other processes
- Many open questions, but most issues are shared by other approaches

And now:
Your questions please!