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

Three Artefacts
- Product backlog
- Sprint backlog
- Burndown chart

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

Product Backlog → Sprint Backlog → Sprint → Working increment of the software

24 h
2-4 weeks
# The 3 Scrum Roles

<table>
<thead>
<tr>
<th><strong>Scrum Master</strong></th>
<th><strong>Product Owner</strong></th>
<th><strong>Team</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>• Responsible for ensuring that the Scrum team adheres to Scrum values, practices, and rules</td>
<td>• Responsible for managing, prioritizing and maintaining the product backlog</td>
<td>• Turns product backlog into increments of potentially shippable functionality every sprint</td>
</tr>
<tr>
<td>• Helps the Scrum Team understand and use self-organization and cross-functionality</td>
<td>• One person, no comittee</td>
<td>• Teams are self-organized without external interference</td>
</tr>
<tr>
<td></td>
<td>• May integrate backlog entries from other persons</td>
<td>• 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

$t$
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 a **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
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
Where does Scrum *not* fit?

- Support tasks and alike
  - Tasks arriving any asynchronously
  - Reaction before next sprint
  - Priority by urgency rather than business value
  - E.g. Kanban:
    - Limits work in progress instead of sprint length
- Upfront specification required for contract
  - Scrum can then be applied in a more incremental manner for increased transparency
- Small scale fixed price settings
  - Scrum has smaller benefits, if only few changes are expected and concepts are well understood by all stakeholders
Combine with Up-front Planning

**Overall Budgeting**
- Rough estimation („Guesstimate“, „Hausnummern“) as before
- Decide on feasibility (ROI sufficient?) as before

**Optimize ROI**
- Sort requirements according to benefit / cost ratio
- Produce detail estimations of requirements filling release resp. budget (plus 10-30%)

**Adjust**
- Per iteration (e.g. during sprint planning):
  - Estimate newly added and changed requirements
  - Re-Prioritize (update order of requirement)
  - Adjust release scope, if needed
  - Estimate and commit (Scope) of upcoming iteration (sprint)
Fixed Price Contracts

Frequent approach with Scrum:

- Initial Backlog for budgeting
- Client can re-prioritize items given effort estimate is kept
  - (Already a compromise!)
- Requires mutual trust, but not more than otherwise
- "Requirements-Clarification mode" paid by time & material
  - Raises the pressure to state clearly what is required

Compare common practices in large waterfall contracts:

- Product and functional specification produced by time & material (typically 30% of overall budget)
- Call for bids after specifications (with advantage for specifier ...)
- Reserve for change requests (20 – 30% recommended)
- Many projects stopped after concept phase
Early Phase of a New Development

**Trade-offs:**

- Optimizing total effort competes with delivering a potentially shippable product
- Technical foundation and architecture tasks vs. potentially shippable product
- “You ain’t gonna need it ...” can compete with avoiding large refactorings
- Lacking velocity vs. information for decisions
- Instable backlog vs. need for scoping
**Trailing Efforts**

**Should QA tests be done within the sprint or in the sprint after?**

- The larger the team, the more likely the sprint after
- Can be similar for other tasks
- "Potentially releasable" then with pipeline of trailing tasks

---

**Within current sprint:**
- Fixing this sprint’s errors
- According to process

**During next sprint:**
- Limited time for QA tests
- Implementation and developer tests must finish earlier

---

Unit & developer tests as in definition of done always within sprint!
**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
Scrum Role Questions

“The Team consists of developers with all the skills to turn the Product Owner’s requirements into a potentially releasable piece of the product by the end of the Sprint.”

- Good practice to include roles beyond development, such as QA or UI designers, into the team
  - But how to deal with fractions of FTE capacity?

- Scrum Master or Product Owner or both may be part of the team
  - But they should not be the same person (Similar to separation of power / checks and balances in constitutions)
Scaling Scrum

- Scrum recommends team sizes of about 7 people
- Larger projects require more structure
- Common organization is Scrum of Scrums
  - Formed by scrum masters of individual teams
  - Not necessarily daily
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!