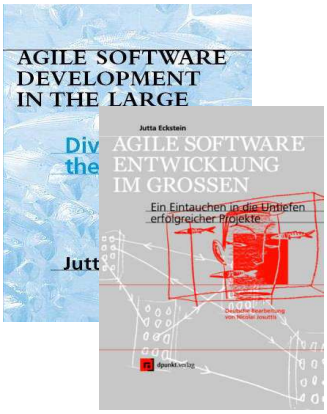
Typical Pitfalls in Agile Software Development



Agiler Tag in Bremen

Jutta Eckstein

What Agile is all about…

Agile Project Management
Agile Testing
Agile Architecture
Agile Offshoring
Agile Investing
Agile PLM
Agile Web Development
Agile Game Development
Agile SOA
Agile AJAX
Agile Web Solutions
Agile MDA / MDD
…

The Agile Value System

Agile development is defined by the value system:

Individuals and interactions
over processes and tools

Working software
over comprehensive documentation

Customer collaboration
over contract negotiation

Responding to change
over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

Source: http://agilemanifesto.org

Agile Principles

- **Value system is based on the following principles:**
  - Early and continuous delivery of valuable software
  - Welcome changing requirements
  - Deliver working software frequently
  - Business people and developers work together
  - Trust motivated individuals
  - Face-to-face conversation
  - Working software is the primary measure of progress
  - Promote sustainable development
  - Technical excellence and good design
  - Simplicity is essential
  - Self-organizing teams
  - Team reflection and adjustment

Pitfall: Practices over Values

- **You can do pair programming in any process**
- **Insisting on specific practices although they don‘t fit**
- **Recommendation:**
  - Use practices that support the team and the team‘s value system
    - Figure out what‘s the goal of the practice
    - Is this also our goal?
    - Does this goal support our value system?
    - How can we achieve this goal?

Pitfall: Treadmill

- **Iterate, but don‘t measure**

Or:

- **„*We have continuous iterations*“**
  - Means we neither plan nor measure
    - How can we know in what shape the project is?
- **Recommendation:**
  - Plan and then measure (and celebrate) your achievements
  - Only time boxes allow to measure

Pitfall: Missing Result-Orientation

- **Plan / estimate but don‘t deliver**
  - Teams wait for one another
  - Tasks are not really done
  - Features are never accomplished
- **Deliver, but don‘t produce value**

- **Recommendation:**
  - Establish cross-functional teams
  - Slice features so they can be delivered in an iteration
    - Including integration and test
  - Plan features that provide a (business) value
    - Testable, measurable

Pitfall: Status On Hold

- **Tasks end in on-hold status**
  - Distraction from current task, because of another one
  - Tasks are planned for persons, who are busy
  - Completion of task depends on other work

- **Recommendation:**
  - Define self-contained tasks
  - Team members commit themselves to tasks
  - Ensure focus on planned tasks in daily synchronization

Pitfall: Unclear Business Value

- **No product owner (customer perspective)**
  - No vision
  - Everyone makes assumptions
  - Nobody provides valuable feedback on achievements
- **Recommendation:**
  - Ensure the defined role of a product owner
    - Provide the communication channel to the customer
    - Steers development
    - Works typically within a team (customer community)

Pitfall: No overall Plan

- **No release plan**
  - Iterations produce value, but this isn‘t fed back in the overall plan
  - Uncertainty about the overall status of the project
- **Recommendation:**
  - Develop an overall release plan early on
    - Combine several iterations to (internal) releases
    - At the end of every iteration check back with the overall release plan

Pitfall: Sanity Check

- **During iteration exists the ritual of a sanity check**
- **Intermediate planning**
  - Fine grained iteration planning
  - Even finer grained mid-iteration planning

- **Recommendation:**
  - Shorten iteration
    - Enables trustful planning
    - Provides confidence in reaching the (realistic) goals

Pitfall: Interruptions are the Norm

- **Continuous changes during iteration**

- **Recommendation:**
  - Define iteration length so you can
    - Deal with the uncertainty of the users need
    - Keep priorities unchanged
  - Think as well about
    - Quiet times
    - Firewall and Gatekeeper
    - Defined slack time box

Pitfall: Same Mistakes

- **Measure / retrospect, but don‘t learn from it**
- **Recommendation:**
  - Come up with an action plan on top 3
    - Figure out who‘s in control
  - Visualize action plan
  - Recapture the changes of top 3 in next retrospective

Pitfall: Importance of Integration is Ignored

- **Build fails too often**
  - Uncoordinated effort is put in build fixes
- **Build time takes too long**
  - People work locally as long as possible
- **Recommendation:**
  - Rule of thumb:
    - Assign 10% of your development resources for integration and build
      - This is a 100% job
  - Ensure that integration has always the highest priority

Pitfall: Daily wasting time

- **Daily Scrums without discipline**
  - Endless daily meetings
  - Management information meetings
  - Status reports to management
- **Recommendation:**
  - Is the team really a team?
  - Recapture rules at the beginning
  - Joint responsibility, but mainly the coach reminds everyone on time
  - As a coach carefully provide feedback

Pitfall: Missing Courage

- **Unable to accept the dependencies of the project variables**
  - Time, scope, quality, resources (people, budget, ...)
- **Recommendation:**
  - Measure your velocity
    - Data always helps to be more realistic and to argue
  - Make progress, estimates and achievements visible
  - Clarify the dependencies and show the options
    - E.g. time delay, elimination of other features, worse quality with maintenance problems
  - Learn to say no!

Pitfall: Agility is for Developers only

- **Try to implement the value system without management support**
- **Recommendation:**
  - Three key roles must support the same approach:
    - Chief architect
    - Project management
    - Process coach
  - Project management has to ensure support higher up the hierarchy

Summary: Common Misunderstandings

Agility is ...

... a specific methodology only

- E.g. XP, Scrum, ...

... defined by practices

- E.g. pair programming, TDD, ...

... an undisciplined approach

- Agile a synonym for unprepared, chaotic, planless

Summary: Lessons Learned

- **Value system**
- **Neither chaos nor dogmatism**
- **Joint responsibility**
- **Culture of change**
- **Continuous learning**

Many Thanks!

Contact information:

Jutta Eckstein

je@it-communication.com
www.it-communication.com