Topics

• Abbreviated history of our collaboration
• 10 years of the Agile manifesto
• We witnessed a good adoption of Agile here at NUST in Summer 2010
• New and noteworthy topics in the Agile community and what it means to us
• Recent and next Agile lab in Bonn
ABBREVIATED HISTORY OF OUR COLLABORATION
Bonn, 10/2003 – 10/2004: Research stay of Prof. Dr. YAN Han
Nanjing, 04/2006: Research visit of Prof. Dr. Armin B. Cremers and Dr. Guenter Kniesel in April 2006
Nanjing, 23.07. – 25.08.2007: Extreme Programming Summer School (1/2)
Nanjing, 23.07. – 25.08.2007: Extreme Programming Summer School (2/2)
Nanjing, 10/2007: Visit of a delegation from the University of Bonn, led by Rector Prof. Dr. Matthias Winiger
Bonn, 16.08.-24.09.2008: Extreme Programming Summer School with Dr. ZHAO Xuelong and Mr. YU Ligong (1/2)
Bonn, 16.08.-24.09.2008: Extreme Programming Summer School with Dr. ZHAO Xuelong and Mr. YU Ligong (2/2)
Bonn, 11/2008: Visit of a delegation from NUST, led by President Prof. Dr. WANG Xiaofeng. (Memorandum of Understanding)
Labs in Nanjing in 2009 and 2010

- Nanjing, Summer 2009: Extreme Programming Lab (Without anyone from Bonn)
- Nanjing, Summer 2010: Extreme Programming Lab, witnessed by us (Daniel Speicher, Mark von Zeschau)
10 YEARS OF THE AGILE MANIFESTO
10 years of the Agile manifesto

“[…] the dominant strain [of the Agile virus] was Extreme Programming”

<table>
<thead>
<tr>
<th>Extreme Programming</th>
<th>Agile Manifesto</th>
</tr>
</thead>
<tbody>
<tr>
<td>Planning Game + Whole Team + Small Releases + Customer Tests</td>
<td><strong>Collaboration</strong> over contract negotiation</td>
</tr>
<tr>
<td>Collective Ownership + Continuous Integration + Coding Standard + Metaphor + Sustainable Pace</td>
<td><strong>Responding to change</strong> over following a plan</td>
</tr>
<tr>
<td>Pair Programming + Simple Design + Test-Driven Development + Refactoring</td>
<td><strong>Working software</strong> over comprehensive documentation</td>
</tr>
<tr>
<td></td>
<td><strong>Individuals and interactions</strong> over processes and tools</td>
</tr>
</tbody>
</table>

Prof. Dr. Armin B. Cremers, Daniel Speicher: “Remarks on Agile labs in Bonn und Nanjing”, May 2011, 14 / 42
10 years of the Agile manifesto

Warning: The divide between software developers and customers may return.

“Now the dominant agile strains are Scrum and Lean, which don't care very much about programming”

“[As] Scrum and Lean exacerbate [the divide between software developers and their customers] by neglecting the technical skills, I fear craftsmanship in turn may make the [divide] worse by neglecting the relationship skills.”  “[Kent Beck’s] primary aim with Extreme Programming was to heal the divide [...].”

All citations from [Fowler 2011-01]
10 years of the Agile manifesto
The essence of the Extreme Programming solution to software project failures

Problem: Projects late + over budget + miss customer needs

- After a long time of development (sometimes only after deployment) quality problems become apparent and the product misses customer needs.
- Developers forced to follow a process had hidden the truth about the progress, focused on producing prescribed intermediate results and got demotivated.
- Too much of the documented design and the code does not provide value but becomes an obstacle to change.

Solution: Rapid reliable feedback + Energized work + Changeability

- Shorten the time to delivery as much as possible to get reliable feedback about technical challenges and real customer expectations and needs.
- Empower self organizing teams. People are motivated! They do want to master their profession. Make sure that they can tell the truth.
- Keep implementation simple and of perfect quality, so that it always can be changed and extended as easy as possible, and “done” means “done”.

Required to make the feedback about the progress match the truth.
Chapter 17

When the Master governs, the people are hardly aware that he exists.

Next best is a leader who is loved.
Next, one who is feared.
The worst is one who is despised.
If you don't trust the people, you make them untrustworthy.
The Master doesn't talk, he acts.
When his work is done, the people say, "Amazing: we did it, all by ourselves!"
WE WITNESSED A GOOD ADOPTION OF AGILE HERE AT NUST IN SUMMER 2010
Agile here at NUST in Summer 2010
Teacher and students who contributed

• Leader: Dr. Zhao Xuelong
• Customer: Yu Ligong
• Consultant: Prof. Yan Han
• Technical expert: Xu Jian

• Consultants: Mark von Zeschau, Daniel Speicher
• Team: Xiao Huan, Gu Chen, Yang Fan, Li Chen, Li Jie, Wang Heng, Lin Hui Zhong, Niu Xin Tao
Agile here at NUST in Summer 2010
Open workspace allowed for communication

- Thorough discussion of the user stories with the customer.
- Surprise to us: Verbal instructions (without any visualization) work. Students are more concentrated.
- Stand up meeting in the morning to solve problems and share knowledge.
Agile here at NUST in Summer 2010
Workspace allowed for focused programming

We had the impression that the team members did know their technology (Web Application, Based on Java technologies) much better than in 2007. As a result they could concentrate on learning the XP process.
Agile here at NUST in Summer 2010
The process works: From Planning to Presentation

• Process is taught and sustained by Prof. Yan Han, teacher Yu Ligong und Dr. Zhao Xuelong.
• Customer Collaboration via Planning Game
  – Planning Poker for estimation
  – Story and Task breakdown
• Team learned to plan:
  – Finds the gaps in the Story and Task definitions
  – Favors simple solutions that can be implemented
• Progress is tracked with the planning board.
• Zhao Xuelong leads like a Scrum Master: Solves issues that impede the team and sustains the process.
• Results are presented at the end of the iteration.
Agile here at NUST in Summer 2010
The process works: From Planning to Presentation
Agile here at NUST in Summer 2010
The process works: From Planning to Presentation
Agile here at NUST in Summer 2010
Team knows benefits and challenges (Retrospectives work!)

- The students enjoy:
  - Pair Programming as a chance to learn from each other
  - Visibility of the progress because of story/task structure
  - Energetic working atmosphere
  - Principle of simplicity allows focused work

- The students are aware of challenges:
  - Difficulties of test creation (We heard that in limited cases, testing was possible and gave trust in the code.)
  - Complexity of tasks that depend on each other
  - Complexity of the code base
Agile here at NUST in Summer 2010

Summary of our impressions and suggestions

• What could be improved?
  – The programming practices are always challenging: Testing, Refactoring, sometimes Pair Programming. (Teaching these is still difficult for us as well. We are always looking for new ways.) Writing tests for the used technologies is really difficult. (Mocking required.)
  – We heard that the final prototype had severe performance problems. It would be challenging but worthwhile to find the reasons: Brevity of the lab? Too little focus on quality? Technology too complex? (We don’t know the answer! Do you? How to improve?)

• What could be worth keeping?
  – All the other things we presented. Good preparation (organization + technology), Nice workplace, Communication, Stand-up meetings, Process sustained by teachers, “Servant leadership”, Regular retrospectives, End of iteration presentations, ...

• Overall, we witnessed a good adoption of Agile here at Nanjing University of Science and Technology in Summer 2010. 😊
We witnessed a good adoption of Agile here at NUST in Summer 2010
NEW AND NOTEWORTHY TOPICS IN THE AGILE COMMUNITY AND WHAT IT MEANS TO US
News in Agile and what it means to us
“Agile” becomes main stream, but ...

- Search results for “Agile Conference 2005” ... “Agile Conference 2011”:
- Project Management Institute
  - Plan to incorporate agility into its project management program.
  - But Ken Schwaber (Co-creator of Scrum) warns: Essence might get blurred! [Schwaber 2011-04]
- “[...] the whole philosophy of agile isn't about interpreting some dusty document, but in making your own journey of discovery. The manifesto is a useful part of that journey, but in the end you have to think for yourself.” [Fowler 2011-05]
- *For us: Favor “really works for us” over “following agile”.*
News in Agile and what it means to us

New flavor of Agile: “Kanban” (看板)

• Goal: Minimize cycle time!
  (Time from idea to deployment)
• Tool: Limits on the “Work in Progress”
  (Stories and tasks in progress)
• For us: Students tend to ignore tasks and stories that are not appealing to them.
  => Limits on Work in Progress and tracking cycle time hopefully ensure that these are processed as well.
News in Agile and what it means to us

Craftsmanship might get forgotten and is difficult to teach.

- There is a tension between simplicity and “anticipation of change” or genericity. [Tilkov 2009-11], [North 2010-11]
- Surprising result of (unpublished) questionnaire: “Anticipation of change” most useful, when requirements are stable.
- For us: More effort to understand deeper and teach Simplicity, Test-First, Quality! Approaches still under development.
News in Agile and what it means to us

Some voices: “Maybe we do need a little more design”

- Discussion not that new! E.g. Fowler 2004: “Is design dead?”: “In fact XP involves a lot of design, but does it in a different way than established software processes. XP has rejuvenated the notion of evolutionary design with practices that allow evolution to become a viable design strategy.” [Fowler 2004]
- Eric Evans: “Folding Design into an Agile Process” [Evans 2010-05]
- For us: There is a gap between what we teach in Software Engineering and what we do in Agile labs. Some of our Software Engineering knowledge might be useful in the lab. Some of our Software Engineering knowledge should be adapted to what we practice in our labs.
News in Agile and what it means to us
“First” “research” about the reasons why Agile works

• Marry Poppendieck: “Lean Software Development”
  (Smaller chunks lead to lower cycle time)
• Mark Kennaly: “SDLC 3.0: Beyond a Tacit Understanding of Agile”
  (Systems theory applied. But still very limited.)
• Game theory (Only first thought experiments)
• Dan Pink: “Drive”
  We are motivated by Autonomy, Mastery and Purpose.
  (Hot topic because of financial crisis. Bonuses don’t make smart.)
• Laurie Williams: “The Collaborative Software Process” Ph.D. 2000,
  18 paper about Agile, 15 paper about pair programming.
• For us: These topics become part of our lab’s seminar
News in Agile and what it means to us

“Learning” becomes an explicit topic in the Agile community

• Deliberate practice
  (But for us: Time too short for Mastery)
• For us: Agile worked for us because its built-in feedback (=learning) cycle.
• For us: As students still need to learn some techniques, they might need more guidance than a team of professionals.
• Professionals tell us, that real world development requires learning as well.
RECENT AND NEXT AGILE LAB IN BONN
Recent and next Agile lab in Bonn
Summer 2010 and Summer 2011

• Seminar and Lab Agile Software Development 2010:
  – Product: „Automatic Adaptation of Web Sites“
  – Customer: Pascal Bihler
  – Team leaders: Tobias Rho, Daniel Speicher

• Seminar and Lab Agile Software Development 2011:
  – Seminar: 30.08. - 02.09.2011, Lab: 05. – 30.09.2011
  – Product: „Cultivate goes Android“
  – Customer: Daniel Speicher
  – Team leaders: Jan Nonnen, Paul Imhoff
    (Former participants of our labs)
Recent and next Agile lab in Bonn

Summer 2010: Automatic Adaptation of Web Sites

- Shadow Proxy containing Request + Response Filters
- Original Web Pages (HTML)
- Adapted Web Pages (HTML) (based on Annotations)
- SemS: Semantical Annotations (Database)
Recent and next Agile lab in Bonn

Summer 2011: Cultivate goes Android - Android

- Android
  - Java based Smartphone OS
  - Development inside Eclipse
  - Low performance Hardware

- Suggested performance tweaks on source code side
  - Avoid Internal Getters/Setters
  - Prefer Static Over Virtual
  - Use Floating-Point Judiciously

Example on next slide

Prof. Dr. Armin B. Cremers, Daniel Speicher: “Remarks on Agile labs in Bonn und Nanjing”, May 2011, 38 / 42
Recent and next Agile lab in Bonn

Summer 2011: Cultivate goes Android - Cultivate

The code we wrote last month could be seen as the soil in which new code and new functionalities grow. If this growth is slowed down, this tells us something about the quality of this soil. Then it is time to focus for a while on improving the quality of the soil without directly contributing to the growth of the functionality. This is **Refactoring**.

Quality criteria, concepts, conventions and practices for software depend on the context, from the targeted hardware to the developers. They evolve over time like real cultures do. Cultivate wants to offer developers the tools to **codify and evolve** as much of their **coding culture** as is useful. In this lab we want to develop an "Android Coding Culture".

```prolog
android_smell_internal_getter_call(Call) :-
class_contains_getter_method(Class, Method),
method_is_called_by(Method, Call),
call_is_in_class(Call, Class).
```
References


Image References

Pictures used are licensed under Creative Commons License.

- http://commons.wikimedia.org/wiki/File:Crystal_Clear_app_konsole.png
- http://commons.wikimedia.org/wiki/File:Crystal_Clear_kdm_user_female.png
- http://commons.wikimedia.org/wiki/File:Crystal_Clear_kdm_user_male.png
- http://commons.wikimedia.org/wiki/File:Crystal_Clear_app_Login_Manager.png
Thank you very much for your attention.