Practical Course eXtreme Programming

B-IT/IPEC
Summer School 2008

Organized by
Prof. Dr. Armin B. Cremers, Holger Mügge, Daniel Speicher, Pascal Bihler, Mark Schmatz
International Program of Excellence (IPEC)

„The International Program of Excellence in Computer Science (IPEC) at the B-IT offers, mainly in the time between terms, compact teaching units on the highest level. This results in a speed-up of studying and in a simultaneous increase of quality.“
eXtreme Programming

An Introduction
What is Extreme Programming?

- *XP is a ...*
  - lightweight,
  - efficient,
  - low-risk,
  - flexible,
  - predictable,
  - scientific and
  - fun

  ... way to develop software.

- Kent Beck, eXtreme Programming eXplained, Addison Wesley 1999
How expensive is change?

- Traditional view / Experience
How expensive is change?

- Agile view / Experience

This is the core assumption (and goal) of agile processes.

Growth of change costs can be controlled.
How can this be achieved? Values!

• **Agile Manifesto** ought to 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

  *There is value in the right side items, but we value the left ones more.*

• **Extreme Programming Values**
  - Communication
  - Simplicity
  - Feedback
  - Courage
  - Respect

[http://agilemanifesto.org](http://agilemanifesto.org)
How can this be achieved? XP Practices!
Rapid feedback is the secret of success

XP is like driving.

"Driving is not about getting the car going in the right direction.

Driving is about constantly paying attention, making a little correction this way, a little correction that way."

(Kent Beck)
Remain responsive to change at any time
eXtreme Programming

Experiences from our courses
Things that worked! (1/3)

- Refactoring exercise as acceptance test
  - Provided a realistic picture of the required skills
  - Improved OO knowledge
    - Repair a **broken JUnit test**.
    - Refactoring 1: **Break dependency cycle** by introduction of observer pattern.
    - Refactoring 2: **Eliminate code duplication** and parallel inheritance hierarchy by introduction of state pattern.

- Each participants makes himself an **expert of a special area** before the beginning of the course
  - Necessary because very much knowledge needed
  - Makes the value of team communication obvious

- **Wiki** as a common knowledge base
Things that worked! (2/3)

- **First 3 days**: Development of a trivial application. Meanwhile teach...
  - XP basics, Test First, Pair Programming
- **Four iterations** (each at 5 days)
  - Structure:
    - planning game
    - Implementation
    - presentation + acceptance test
- **Planning poker** (see later)
- **XP-Game** to explain the planning Game
  - Clarifies responsibilities of customers and developers
  - Demonstrates the value of realistic estimations.
  - Sometimes a little bit time consuming. But: Fun!
Things that worked! (3/3)

- **Pair Programming**
- **Daily stand up meeting**
  - Everybody answers questions like:
    - *What did I do yesterday?*
    - *What obstacles do I have?*
    - *What am I going to do today?*
    - *What else should the team know about?*
  - No one gets stuck into problems others could solve.
  - Builds team spirit.

- **Regular Retrospections**
  - THE key to excellence by rapid adaptation
  - Example: Burn-Down Charts (see later)

- **Non-conflicting roles** *(Keep What and How separate)*
  - customer ≠ team leader ≠ expert
Planning Poker (adapted from James W. Grenning, Object Mentor, 2002)

http://www.objectmentor.com/resources/articles/PlanningPoker.zip

- Accelerates story estimation.
- Keeps the whole team involved.
- Mechanics:
  - Customer reads a story.
  - Each programmer selects the card corresponding to his estimate.
  - Cards are turned over simultaneously.
  - In case of agreement: Record the estimate and move on to the next story.

Bricks = Estimated effort for a story

Infeasible story
Risk that the real effort might be much higher.
Chance that the real effort might be much lower.
The XP-Game (adapted from Belgian XP/Agile User Group)

- Story descriptions. Business value printed on each card.
- Clear responsibilities ("We are developers")
- Funny "stories" to "implement"
- Already used planning poker like in the real planning game.

http://www.xp.be/xpgame.html
Storycards
Simple Design, Testing

- **Simple Design**
  - (+) CRC cards nice tool for fast architectural sketches
  - (−) Difficult to get everybody involved
  - (+) Explore protocols by role playing simulations
  - (−) Hard to explain the difference between simple and quick premature design

- **Testing**
  - Recommended but rarely practiced by the students
  - Requires severe discipline
  - GUI testing is really hard
  - Round trip testing even harder (Java ⇒ Compiler ⇒ JTransformer ⇒ Prolog ⇒ Predicate Evaluation)
  - Untested code forms legacy code after the course
Tracking

• Essential to estimate capacity of the next iteration

• Motivation for tracking (estimation + consequent logging of time spent) is hard to find:
  – The XP coach really has to care about this
  – Fast feedback about the quality of the estimates might help (Exercised once)
  – Hard to avoid to pessimistic estimates
  – Burn down charts (see SCRUM) might be a better way because they constantly visualize the remaining effort

• Virtual unit for effort (bricks, see above)
  – facilitates relative estimates
  – are not always taken serious
Burn-Down Charts

Burn Down XPP06b Iteration 3

- Red: Remaining Effort
- Green: Remaining Time
- Grey: Remaining Tasks
Reality Check

• The “ideal world” of our courses
  – Our customer is much more friendly than real customers usually are.
  – Employees have much more interfering responsibilities than our students have for the time of the course.

• Problems that also occur in the “real world”
  – Development teams are seldom self-organizing
    • But those, that are, are the best
  – Building a reliable testbed is really hard
b-it: Excellent working environment
Our practical courses
What we are offering the students...

• Good supervision
  – 2 (+4) research associate for 12 students
  – ... 8 hours a day!

• Ideal working environment
  – Our own office
  – Up-to-date technical equipment (computer, beamer)
  – Wikis, Eclipse, UMPCs, ...

• An interesting and realistic project
  – Developing process is an essential parts of research projects
  – Industry partner ensures quality of product

• A certificate after four and a half weeks
  – ECTS Credit Points 10
The „products“ of the practical courses

Rich Client Applications on Mobile Device
- 2007b: Scotland Yard to go (see next slides)
- 2006b: Context Sensitive Mobile Application (CSI Navigator)
- 2005b: Context Sensitive Mobile Application (CSI PimPro)

Plug-Ins for the Java Development Platform Eclipse
- 2005a: Visual Tool Support for Refactoring to Pattern (Cultivate, PatchWork)
- 2004b: Program Analysis by Logic Meta Programming (JTransformer, Cultivate)
- 2004a2: Tool Support for Pattern Management (PatchWork)
- 2004a1: Synchronized Logic Representation of Java Code (JTransformer)
- 2003b: Improved Editor for Conditional Transformations (ConTraCT) [based on the result of two earlier practical courses]
What have we developed?

- CSI-Customer requested products:

  Fall 2005: An adapting location-aware Personal Information Manager

  Fall 2006: A Context Sensitive Bike-Tour-Navigation
Last year: Scotland Yard to go!

• Use case: Adaptive Mobile Gaming
• Mobile Games:
  – PDA/UMPC based
  – Use context data
  – Augment reality
This year: Mobile gaming platform

• Based on our experiences from last year

• Buzzwords:
  – Modularization
  – Community
  – Game 2.0
  – User centric design
  – Collaboration
  – Cooperation
  – Fun!
This year’s challenge

The chase is on...
Mobile gaming platform

• Based on our experiences with Scotland Yard to go!
• Buzzwords:
  – Modularization
  – Community
  – Game 2.0
  – User centric design
  – Collaboration
  – Cooperation
  – Fun!
Studious and productive collaboration
Satisfied students, associates and prof.