

# PRAKTIKUM BERUFSAKADEMIE KARLSRUHE eXtreme Programming (XP)

Oliver Kühn

## Agenda

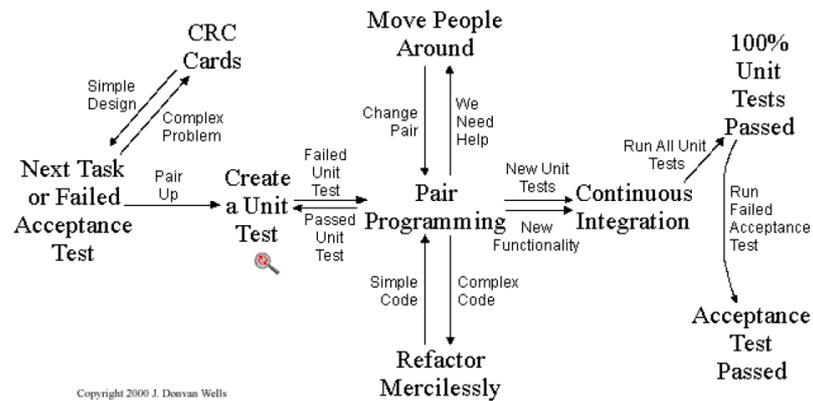
2

- **Was ist eXtreme Programming (XP)?**
- **Elemente von XP**
- **Grundwerte und Rahmenbedingungen von XP**
- **Einsatzbereiche und Probleme bei XP**
- **Literatur zu XP**

## Was ist eXtreme Programming (XP)?

3

- “XP is a lightweight, efficient, low-risk, flexible, predictable, scientific, and fun way to develop software” Kent Beck



Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## XP ist wie Autofahren

4

- 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.
- XP erfordert nicht weniger Disziplin als andere Methodiken

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## Agenda

5

- Was ist eXtreme Programming (XP)?
- **Elemente von XP**
- Grundwerte und Rahmenbedingungen von XP
- Einsatzbereiche und Probleme bei XP
- Literatur zu XP

---

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## Elemente von XP

6

- Automatic Testing
- Test First/ Test Driven Development (TDD)
- Collective Ownership
- Simple Design – KISS (Keep It Simple, Stupid)
- Refactoring
- Pairprogramming
- Continius Integration
- Coding Standards

---

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## Was ist eXtreme Programming (XP)?

7



Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## Grundidee von XP

8

- Die Umwelt wird immer turbulenter, dadurch kommt es immer öfter zu Änderungen an der Anforderungsspezifikation einer Software.
- Die Frage ist daher nicht, wie man dies verändern oder vorhersehen kann, sondern wie man dies bestmöglich behandelt.

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## Grundidee von XP

9

- Programmierung
  - Konzentration auf das Wesentliche (nur Tasks mit hoher Priorität)
  - Nicht allein sein in schwierigen Situationen
  - Das zu tun, was Mensch am besten kann
  - Programmierer übernehmen mehr Verantwortung
  - Team steht im Vordergrund
- Kunden und Auftraggeber (Management)
  - Konkrete Ergebnisse in kurzen Intervallen
  - Kurze Reaktionszeiten bei Änderungen
  - Vertretbarer Aufwand für Änderungen

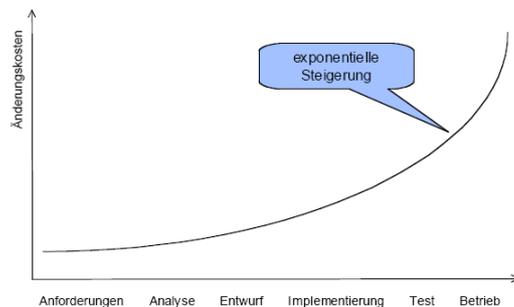
Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## Wie teuer sind Änderungen?

10

### Wie teuer sind Änderungen?

- Traditionelle Sichtweise / Erfahrung



Dr. Günter Kriesel, Institut für Informatik III, Universität Bonn



Vorlesung Softwaretechnologie I, Wintersemester 2000 2a-23

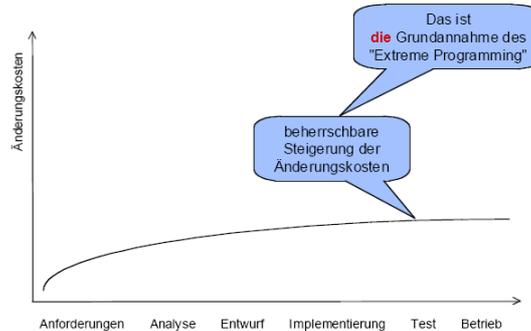
Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## Wie teuer sind Änderungen?

11

### Wie teuer sind Änderungen?

- XP-Sichtweise / Erfahrung(?)



Dr. Günter Kriesel, Institut für Informatik III, Universität Bonn



Vorlesung Softwaretechnologie I, Wintersemester 2000 24-24

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## XP – Automatic Testing

12

### □ Automatic Testing

- Ist eine der wichtigsten Tätigkeiten in XP.
- Parallele Entwicklung von Code und Tests garantiert „Testüberdeckung“.
- Tests sichern (weitestgehend) die Korrektheit vom Refactoring und machen Refactoring bezahlbar.
- Schreibe für jede Klasse eine Testklasse.
- Die Tests müssen automatisiert werden, um Wiederholbarkeit zu sichern.
- Häufig testen – mindestens am Ende des Tages müssen alle Testfälle durchlaufen.
- Im Idealfall erst die Testklasse schreiben und dann die zu testende Klasse – Test First.
- Framework JUnit unterstützt die Testdefinition.

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## XP – Test Driven Development

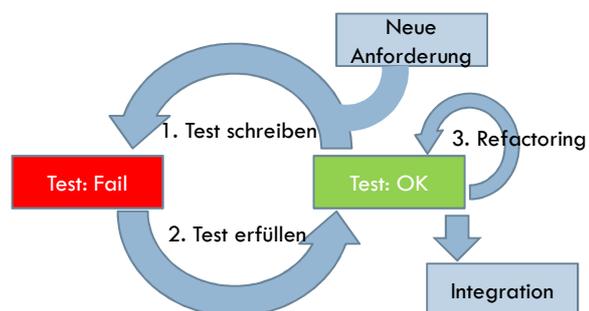
13

- Test First/ Test Driven Development (TDD)
  - Grundidee der testgetriebenen Software-Entwicklung:
    - Wenn Testen so wichtig ist, warum (automatisierte) Tests nicht an den Anfang statt ans Ende der Entwicklungsaktivitäten setzen?
    - Systematisches Unit-Testing statt „zufälliger“ Entwicklertests
    - Immer, wenn ein Fehler gefunden wurde, erst die Testklasse so erweitern, dass sie den Fehler gefunden hätte. Dann die Korrektur durchführen.

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## XP – Test Driven Development

14



Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## XP – Collective Ownership

15

- Collective Ownership
  - Jeder darf jederzeit Code im System ändern – keine Wissensmonopole
  - Der Code (und alle anderen Dokumente im Entwicklungsprozess) „gehören“ dem Team gemeinsam
  - Wenn etwas nicht funktioniert, ist immer das gesamte Team in der Verantwortung

---

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## XP – Simple Design

16

- Simple Design – KISS (Keep It Simple, Stupid)
  - Einfachheit ist gut, daher alles so einfach wie möglich bzw. so komplex wie nötig.
  - Baue das einfachste, was gerade noch funktionieren kann.
  - Einfache Entwürfe haben viele Vorteile gegenüber komplexen Entwürfen:
    - einfacher zu verstehen
    - schneller zu ändern
    - weniger Aufwand zur Erstellung
    - Testklassen sind leichter zu schreiben
  - Keine Technologie auf Vorrat bauen.
    - Technologie, die erstmal nicht benötigt wird, bindet Entwicklungsressourcen zur falschen Zeit.
    - Nicht verwendete Technologie wird bei den Praxistests nicht mit überprüft.
    - Da nicht verwendete Technologie für die Anwendung nicht benötigt wird, altert diese deutlich schneller als der Rest der Anwendung.

---

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## XP - Refactoring

17

- Refactoring
  - Design ist wichtig, daher ständiges (re)design notwendig.
  - Es ist keine Schande schlechten Code zu schreiben. Aber es ist eine Schande, diesen nicht zu ändern.
  - Die alte Leitlinie „Never change a running system“ führt zu großen Problemen und sollte daher ad Akta gelegt werden.
  - Das Restrukturieren von Software unter Beibehaltung des Verhaltens, nennt man „Refactoring“.
  - Ob das Verhalten beibehalten wurde, kann mit Regressionstests festgestellt werden.
  - Umstrukturierungen sollen so zerlegt werden, dass sie sehr schnell (binnen weniger Stunden) umgesetzt werden können.

---

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## XP - Refactoring

18

- Metriken zur Messung der Qualität und als Grundlage für Refactoring
  - Kenngrößen
    - Anzahl Lines of Code
    - Anzahl Packages
    - Anzahl Klassen
    - Anzahl Methoden
  - Äußere Qualität
    - Anzahl Bugs
    - Testabdeckung gesamt in %
  - Innere Qualität
    - Anzahl Compiler Warnings
    - Anzahl Klassen mit mehr als 20 Methoden
    - Anzahl Methoden mit mehr als 15 Statements

---

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## XP – Pair Programming

19



Copyright © 2003 United Feature Syndicate, Inc.

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## XP – Pair Programming

20

- „Code reviews“ sind gut, daher ständig „Code reviews“ mittels Pair Programming
- Es sitzen immer zwei Entwickler gleichzeitig vor demselben Rechner.
- Pair Programming: 4 Augen sehen mehr als 2
  - ▣ Hat man oft schon erlebt: man findet lange Zeit einen Fehler nicht, bis der Kollege vorbei kommt und den Fehler anzeigt
- Pair Programming führt zu einer deutlichen Erhöhung der Programmqualität:
  - ▣ weniger Fehler
  - ▣ bessere Klassenschnittstellen
  - ▣ weniger Redundanzen

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## XP – Pair Programming

21

- Bei gleich hoch qualifizierten Entwicklern kann Pair-Programming zu deutlicher Produktivitätssteigerung führen.
- Pair-Programming mit wechselnden Paaren verbreitet die Kenntnisse über das System sehr schnell im Projekt.
- Tastatur wird häufig gewechselt.
- Wenn es dem „Zuschauer“ zu bunt wird, nimmt er sich einfach die Tastatur (Konfliktfähigkeit).
- Der „Besitzer“ der Tastatur erläutert jeweils was er tut. Der andere betrachtet die Entwicklung mit etwas Abstand und gibt aus dieser Perspektive Anregungen und Tipps (Kommunikationsfähigkeit).
- Paare werden nicht fest zusammengestellt, sondern bilden sich situationsabhängig neu (während der Planung).
- Büro-Ausstattung darauf ausrichten.
- Aber: Pair Programming nicht immer möglich, auch aus Budgetgründen des Kunden.

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## XP – Pair Programming

22

- **Kritisch:**
  - Pair Programming muss zur Persönlichkeit der Beteiligten passen
  - Erste quantitative Versuche mit Studenten: (Paar vs. single)
    - Mehraufwand von ca. 30%
    - Time-to-finish: um ca. 15 % reduziert
    - Qualität (korrekte Testläufe): 97% pair, vs. 83% single

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## XP – Continius Integration

23

- Integrationstests sind wichtig, daher mehrmals täglich.
- Sobald eine Aufgabe erledigt ist wieder integrieren und testen angesagt.
- Am Abend läuft die Version auf dem Integrationsrechner.
- Jeder im Team kann sofort von der Neuentwicklung profitieren.
- Neuer Code kann gleich von den anderen ausprobiert und begutachtet werden.
- Risiko der Doppelentwicklung wird reduziert.
- Wenn sich etwas nicht am selben Tag integrieren lässt, kann es sinnvoll sein, den neuen Code zu löschen und am nächsten Tag von neuem zu beginnen.

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## XP – Die häufigsten Fehler

24

- Eine Teilmenge der Kernpraktiken wird umgesetzt:
  - Die fünf Praktiken (Test-First-Programmierung, Refactoring, Akzeptanztests, Pair Programming und fortlaufende Integration, Collective Code Ownership) sind synergetisch auf einander abgestimmt: Ohne zuverlässige Tests wird häufiges Refactoring schlicht unmöglich. Ohne häufiges Refactoring und ohne Testabdeckung wird Collective Code Ownership scheitern. Und schließlich: TDD verlangt viel Aufmerksamkeit und Disziplin – gesichert über Pair Programming.
  - Insbesondere Pair Programming ist ungewohnt und erscheint unbequem. Argumente wie „unökonomisch“ oder „nicht praktikabel, weil ...“ sind vorgeschoben: Es geht um die persönliche Arbeitsweise. Wir empfehlen dringend, Pair Programming ernsthaft auszuprobieren.
- Pair Programming ohne Erfahrung:
  - Pair Programming ist schwierig. Einer der Partner sollte Erfahrung im Pair Programming haben.

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## XP – Die häufigsten Fehler

25

- Fehler und Probleme sind nicht Team-Sache
  - TDD setzt auf Teamwork. Ohne Teamwork wird das Zusammenspiel der Praktiken nicht klappen. Teamwork bringt Transparenz in die Arbeitsleistungen der Einzelnen. Diese Transparenz setzt ein solidarisches und offenes Umgehen mit Problemen, Konflikten und Fehlern voraus. Das Team und insbesondere das Management müssen eine entsprechende Atmosphäre befördern.
- Tests werden nicht vor dem Anwendungscode geschrieben: Testen hat einen offensichtlichen Vorteil: Qualität wird durch Testabsicherung erreicht.
  - Mit Test First sind weitere, nicht auf den ersten Blick sichtbare Gewinne verbunden: Die Beschäftigung mit dem Test vor der Beschäftigung mit dem Programmcode führt zu einer intellektuellen Trennung der Beschäftigung mit dem Interface einer Klasse und dessen Implementierung. Die Trennung von Implementierung und Interface ist der Kern guten Designs. Mit anderen Worten: Test First induziert eine Denkweise und Arbeitsdisziplin, die gutes Design fördert.
  - Test First führt zur Disziplin der Fertigstellung einer Programmieraufgabe in abgrenzbaren, definierten Schritten.
  - Test First gibt Sicherheit für Refactorings und sorgt somit für schlanken Code.

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## XP – Die häufigsten Fehler

26

- Keine Zeit für Refactoring:
  - Agile Entwicklung legt Wert auf Geschwindigkeit. Moderne Entwicklungswerke erlauben eine enorm rasante Entwicklung. Damit Software nachhaltig entwickelbar und erst recht wartbar bleibt, muss ein klares Design erhalten bleiben, müssen Redundanzen eliminiert werden, müssen Coding- und Qualitätsstandards erhalten bleiben. Dafür sorgt Refactoring. Die Vernachlässigung von Refactoring ist genauso teuer wie das Unterlassen der Ölwechsel bei Ihrem Auto.
- Unit-Tests sind nicht mehr zu 100% grün:
  - Unit Tests bilden den Bezugsrahmen. Sie geben Sicherheit, sorgen für Qualität und Geschwindigkeit. Diese Funktion können sie nur wahrnehmen, wenn das Team sich zu 100% auf die Tests verlassen kann. Dies geht nur, wenn die Testabdeckung angemessen hoch ist und die Tests zu 100% grün sind.

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## XP – Die häufigsten Fehler

27

- Metriken werden schlechter:
  - Ein deutliches Indiz, dass Refactoring zu wenig Bedeutung beigemessen wird (s.o.)
- Test-, Dokumentations- und Refactoringphase nach eigentlicher Entwicklung:
  - Agiles Programmieren heißt, „make it run, make it good, make it better“. Agile Entwicklung heißt, dass definierte Teilaufgaben nacheinander zu 100 % fertig gestellt werden – inklusive Tests und Dokumentation.
- Akzeptanz-Tests keine dezidierte Aufgabe
  - Die gemeinsam mit dem Kunden vorzunehmende Umsetzung der Kundenanforderungen in Akzeptanzkriterien und in automatisierte Akzeptanztests ist eine eigenständige Aufgabe. Ebenso die beständige Pflege der Akzeptanztests

---

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## Agenda

28

- Was ist eXtreme Programming (XP)?
- Elemente von XP
- **Grundwerte und Rahmenbedingungen von XP**
- Einsatzbereiche und Probleme bei XP
- Literatur zu XP

---

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## Grundwerte von XP

29

- Mut neue Wege zu gehen
- Mut zu Änderungen
- Mut Code wegzuwerfen
- Gegenseitige Verstärkung und Unterstützung
- Gegenseitigen Respekt
  - ▣ Andere Meinungen ernst nehmen
  - ▣ Gegenseitiges persönliches Interesse
- Verantwortung zu übernehmen

---

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## Rahmenbedingungen von XP

30

- Maximal 5-10 Programmierer sind an einem Projekt beteiligt.
- Kunde steht zur intensiven Einbindung in das Projekt zur Verfügung.
- Räumlichkeiten erlauben intensive Kommunikation und Kontakt zwischen den Entwicklern.
- Kunde verzichtet auf ausführliche Dokumentation von Analyse und Entwurf.
- Entwickler-Team ist motiviert, eigenverantwortlich zu handeln.

---

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## Agenda

31

- Was ist eXtreme Programming (XP)?
- Elemente von XP
- Grundwerte und Rahmenbedingungen von XP
- **Einsatzbereiche und Probleme bei XP**
- Literatur zu XP

---

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## Einsatzbereiche für XP

32

- Gut geeignet für:
  - kritische Projekte
  - Projekte mit Zeitdruck
  - Projekte mit viel Neuland
  - Projekte mit qualifizierten Entwicklern mit kommunikativen Fähigkeiten

---

Praktikum Berufsakademie Karlsruhe ■ eXtreme Programming (XP) ■ Oliver Kühn ■ 11.01.2008

## Probleme bei XP

33

- Weniger geeignet für:
  - Projekte, die keine qualifizierten Entwickler haben oder „Eigenbrötler“.
  - Manchmal Akzeptanzprobleme bei konventionellen Entwicklern
  - Skaliert nur bis max. 10 Personen
  - Manche Projekte schreiben Dokumentationsleistung vor

---

Praktikum Berufsakademie Karlsruhe ▪ eXtreme Programming (XP) ▪ Oliver Kühn ▪ 11.01.2008

## Agenda

34

- Was ist eXtreme Programming (XP)?
- Elemente von XP
- Grundwerte und Rahmenbedingungen von XP
- Einsatzbereiche und Probleme bei XP
- **Literatur zu XP**

---

Praktikum Berufsakademie Karlsruhe ▪ eXtreme Programming (XP) ▪ Oliver Kühn ▪ 11.01.2008

## Literatur und Quellenangaben

35

- „Extreme Programming“, Kent Beck, Addison Wesley
- „Extreme Programming“, Martin Fowler und Kent Beck, Addison Wesley
- andrena Objects, Karlsruher Softwareunternehmen, Veranstalter Objekt Forum und XP-days, <http://www.andrena.de/>

36



## Highlights XP Techniken Elemente von XP

37

- Automatic Testing
- Test First/ Test Driven Development (TDD)
- Collective Ownership
- Simple Design – KISS (Keep It Simple, Stupid)
- Refactoring
- Pairprogramming
- Continius Integration
- Coding Standards