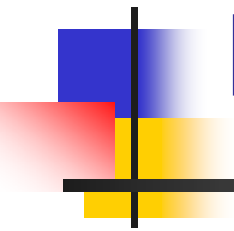
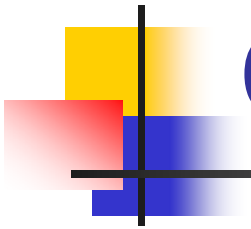


Extreme Programmierung

Seminar: Programmierkonzepte

Julia Mikov, Olga Walker

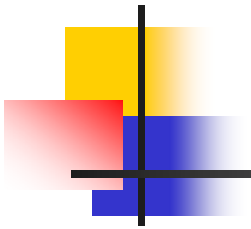
11.01.06



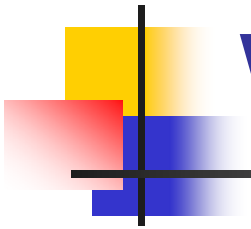
Gliederung

- Einführung
- Werte von XP
- Praktiken von XP
- Zusammenfassung
- Diskussion

Einführung



- XP - ein neuer Stil, Software zu entwickeln
- Anpassung an Veränderungen
- Aufteilung in Geschäfts- und Entwicklungsinteressen
- Ziele: - Kosten gering halten
  - Termine einhalten
  - qualitative Software liefern
- Steuerungsvariablen:
  - Kosten - Zeit
  - Umfang - Qualität



Werte von XP

- Kommunikation
- Einfachheit
- Feedback
- Mut
- Respekt

Praktiken von XP

- Planungsspiel
- Paarprogrammierung
- Einfaches Design
- Gemeinsame Verantwortlichkeit
- Fortlaufende Integration
- Kunde vor Ort
- Kurze Releasezyklen
- 40-Stunden-Woche
- Programmierstandards
- Metapher
- Testen
- Refactoring

- Das Planungsspiel
  - Meeting beider Parteien
  - Storycards
  - Einschätzung der Leistungsmerkmale durch die Entwickler
  - Kunde sortiert die Storycards nach Priorität
  - Gruppierung der Leistungsmerkmale zu Iterationen

Praktiken von XP

- Planungsspiel
- Paarprogrammierung
- Einfaches Design
- Gemeinsame Verantwortlichkeit
- Fortlaufende Integration
- Kunde vor Ort
- Kurze Releasezyklen
- 40-Stunden-Woche
- Programmierstandards
- Metapher
- Testen
- Refactoring

- Programmieren in Paaren
  - ein Bildschirm, eine Tastatur, eine Maus
  - Vorteile: bessere Kommunikation, in jedem Systemteil kennen sich mindestens 2 Programmierer aus, Fehlerkontrolle, Brainstorming
  - Probleme: psychologische Einstellung

Praktiken von XP

- Planungsspiel
- Paarprogrammierung
- Einfaches Design
- Gemeinsame Verantwortlichkeit
- Fortlaufende Integration
- Kunde vor Ort
- Kurze Releasezyklen
- 40-Stunden-Woche
- Programmierstandards
- Metapher
- Testen
- Refactoring

- Einfaches Design
  - Anpassung an Veränderungen
  - einfachste Lösung wählen
  - Verzicht auf unnötige Funktionalitäten
  - Vorteile: übersichtlich, schnellere Einarbeitung des neuen Personals

Praktiken von XP

- Planungsspiel
- Paarprogrammierung
- Einfaches Design
- Gemeinsame Verantwortlichkeit
- Fortlaufende Integration
- Kunde vor Ort
- Kurze Releasezyklen
- 40-Stunden-Woche
- Programmierstandards
- Metapher
- Testen
- Refactoring

- Gemeinsame Verantwortlichkeit
  - Der Code gehört jedem Teammitglied
  - Jeder darf den Code verändern
  - Testen nach Veränderungen
  - Regel: „You break it, you fix it.“

Praktiken von XP

- Planungsspiel
- Paarprogrammierung
- Einfaches Design
- Gemeinsame Verantwortlichkeit
- Fortlaufende Integration
- Kunde vor Ort
- Kurze Releasezyklen
- 40-Stunden-Woche
- Programmierstandards
- Metapher
- Testen
- Refactoring

Fortlaufende Integration

- In den kurzen Abstanden kleine Codestücke integrieren
- Gelegenheit: neu implementierte Module wurden erfolgreich getestet
- Vorteil:

  frühzeitige Konflikterkennung

  → Vermeidung der Kostenexplosion

  → Verringerung des Projektrisikos

Praktiken von XP

- Planungsspiel
- Paarprogrammierung
- Einfaches Design
- Gemeinsame Verantwortlichkeit
- Fortlaufende Integration
- Kunde vor Ort
- Kurze Releasezyklen
- 40-Stunden-Woche
- Programmierstandards
- Metapher
- Testen
- Refactoring

Kunde vor Ort

- Regulärer, am besten kontinuierlicher Zugang zu dem Kunden (Vertreter)
- Vorteile:
  - keine unnötige Verzögerungen
  - keine Kosten für nicht korrekte Funktionalität

Praktiken von XP

- Planungsspiel
- Paarprogrammierung
- Einfaches Design
- Gemeinsame Verantwortlichkeit
- Fortlaufende Integration
- Kunde vor Ort
- Kurze Releasezyklen
- 40-Stunden-Woche
- Programmierstandards
- Metapher
- Testen
- Refactoring

Kurze Releasezyklen

- Häufige Versionsausgabe
- Vorteile:
  - Gewinn für den Auftraggeber
  - frühere Feedbacks
  - Unterstützung der Funktionalitätsimplementierung nach Priorität

Praktiken von XP

- Planungsspiel
- Paarprogrammierung
- Einfaches Design
- Gemeinsame Verantwortlichkeit
- Fortlaufende Integration
- Kunde vor Ort
- Kurze Releasezyklen
- 40-Stunden-Woche
- Programmierstandards
- Metapher
- Testen
- Refactoring

40-Stunden-Woche

- Überstunden sind kein Normalität
- Vorteil:

  bessere Konzentration

  → weniger Fehlern

  → höhere Geschwindigkeit

  → Termineinhaltung, Kostensparung

Praktiken von XP

- Planungsspiel
- Paarprogrammierung
- Einfaches Design
- Gemeinsame Verantwortlichkeit
- Fortlaufende Integration
- Kunde vor Ort
- Kurze Releasezyklen
- 40-Stunden-Woche
- Programmierstandards
- Metapher
- Testen
- Refactoring

Programmierstandards

- Regeln für das Codedesign
- Vorteil:

  bessere Lesbarkeit des Codes

  → leichtere Fehlerbehebung, Refactoring

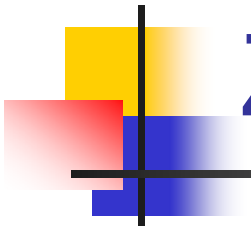
  → Softwarequalitätssteigerung

Praktiken von XP

- Planungsspiel
- Paarprogrammierung
- Einfaches Design
- Gemeinsame Verantwortlichkeit
- Fortlaufende Integration
- Kunde vor Ort
- Kurze Releasezyklen
- 40-Stunden-Woche
- Programmierstandards
- Metapher
- Testen
- Refactoring

Metapher

- Vorstellung von dem System
- Zusammensetzung aller Begriffe
- Vorteile:
  - leichtere Aufgabenauffassung
  - bessere Kommunikation



Zusammenfassung

- Fünf Werte
- Zwölf Praktiken
- Fazit:
  - kein exponentielles Kostenwachstum
  - termingerechte Abgabe des Projekts
  - qualitative Software