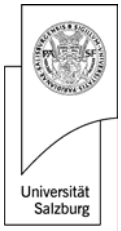
Software Produktentwicklung



Wintersemester 2004 / 2005

Guido Menkhaus and Sebastian Fischmeister
University of Salzburg

SOFTWARE RESEARCH LAB

Building and Software Architecture

Simple

One Person
Minimal modeling
Simple process
Simple tools

Average

Built by a team
Modeling
Process
Tools

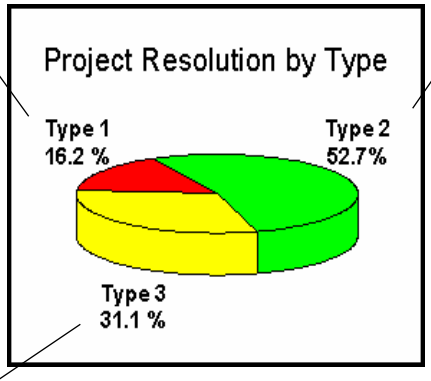
Complex

Built by different teams
Complex Modeling
Well-defined process
Power tools

UNIVERSITÄT SALZBURG

Software project failure statistic

Type 1: Project successful



Type 2:
Project challenged:
The project is completed and operational but over-budget, over the time estimate, and offers fewer features and functions than originally specified.

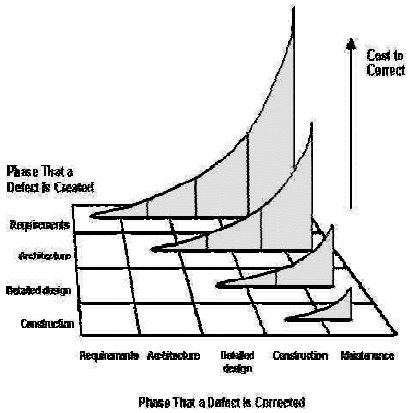
Type 3: project canceled

The CHAOS Report (1994),
Standish Group
http://www.standishgroup.com/sample_research/chaos_1994_1.php

UNIVERSITÄT SALZBURG

Software costs

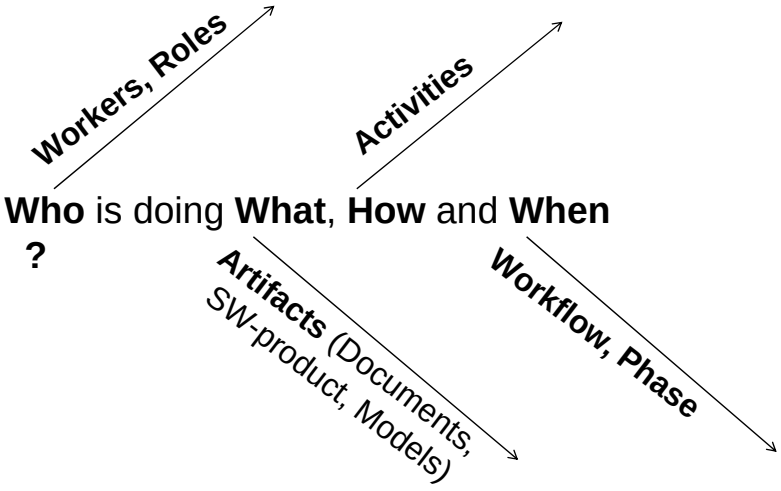
- Software costs often dominate system costs. The costs of software on a PC are often greater than the hardware cost
- Software costs more to maintain than it does to develop. For systems with a long life, maintenance costs may be several times development costs
- Software engineering is concerned with cost-effective software development



Steve McConnell, Software Quality at Top Speed, 1996,
http://www.stevemcconnell.com/articles/art04.htm

UNIVERSITÄT SALZBURG

Software Process



UNIVERSITÄT SALZBURG

What are software engineering methods?

- Structured approaches to software development which include system models, notations, rules, design advice and process guidance
- Model descriptions
  - Descriptions of graphical models which should be produced
- Rules
  - Constraints applied to system models
- Recommendations
  - Advice on good design practice (**best practices**)
- Process guidance
  - What activities to follow

UNIVERSITÄT SALZBURG

XP

- XP ist agile Software-Entwicklung
  - Evolutionäre Entwicklung der Software
  - Lieferung der Software über viele Releases hinweg
- Aufeinanderfolgende Releases streben einer immer kompletteren Implementierung der Produktbeschreibung zu.
- Die Idee ist, im Entwicklungsprozess so früh wie möglich zu ausführbarer Software zu gelangen, welche dann in regelmäßigen kurzen Abständen dem Kunden zur gemeinsamen Abstimmung vorgelegt werden kann.
- Inkrementell ←→Iterativ ←→Evolutionär

Techniken

- Techniken für das XP Team
  - Trainer: bespricht mit dem Team die diszipliniert einzuhaltenden Techniken und erinnert das Team, wenn es die selbstgewählten Regeln verletzt.
  - Verfolger: nimmt regelmässig den aktuellen Status und die geleisteten Programmieraufwände auf, um so zuverlässige Geschichtsdaten über das Projekt zu erhalten
- Techniken für den Kunden
- Techniken für die Entwicklung
- Techniken für das Management

Frank Westphal, Extreme Programming

Techniken für das Team

- **Offene Arbeitsumgebung**
  - Das Team arbeitet zusammen in einem größeren Raum oder eng aneinander grenzenden Räumen.
- **Kurze Iterationen**
  - Die Entwicklung erfolgt in Perioden von ein bis drei Wochen. Am Ende jeder Iteration steht ein funktionsfähiges, getestetes System mit neuer, für den Kunden wertvoller Funktionalität.
- **Gemeinsame Sprache**
  - Das Team entwickelt in seiner Arbeit ein gemeinsames Vokabular, um über die Arbeitsweisen und das zu erstellende System diskutieren zu können. Die Kommunikation im Team erfolgt stets offen und ehrlich.
- **Retrospektiven**
  - Jede Iteration endet damit, in einem Rückblick über die eigenen Arbeitsweisen kritisch zu reflektieren und im Team zu diskutieren, was gut lief und was in Zukunft anders angegangen werden muß.
- **Tägliches Standup-Meeting**
  - Der Tag beginnt mit einem Meeting, das im Stehen gehalten wird, damit es kurz und lebendig bleibt. Jedes Teammitglied berichtet reihum, an welcher Aufgabe er gestern gearbeitet hat und was er heute machen wird. Probleme werden genannt aber nicht gelöst.

Techniken für den Kunden

- **Benutzergeschichten**
  - Die Kunden halten ihre Anforderungen in Form einfacher Geschichten auf gewöhnlichen Karteikarten fest.
- **Iterationsplanung**
  - Jede Iteration beginnt mit einem Planungsmeeting, in dem das Kundenteam seine Geschichten erzählt und mit dem Programmierteam diskutiert. Die Programmierer zerlegen die geplanten Geschichten am Flipchart in technische Aufgaben, übernehmen Verantwortung für einzelne Aufgaben und schätzen deren Aufwände vergleichend zu früher erledigten Aufgaben.
- **Akzeptanztests**
  - Die Kunden spezifizieren während der Iteration funktionale Abnahmekriterien.
- **Kurze Releasezyklen**
  - Nach ein bis drei Monaten wird das System an die wirklichen Endanwender ausgeliefert, damit das Kundenteam wichtiges Feedback für die Weiterentwicklung erhält.

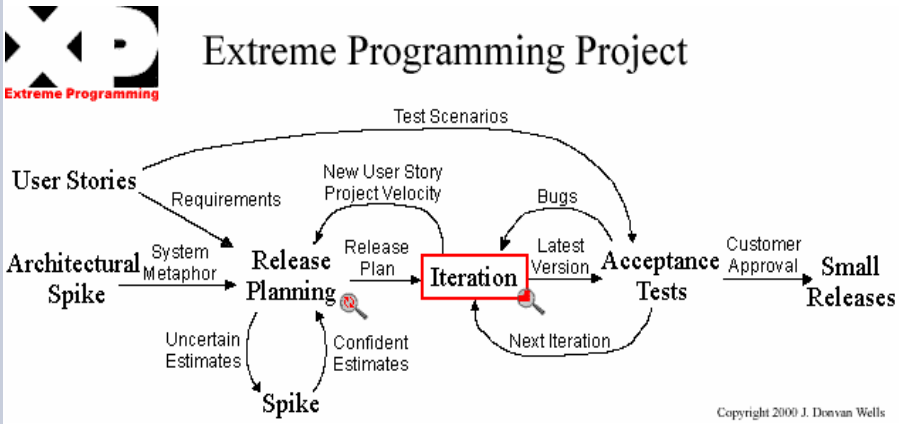
Techniken für die Entwicklung

- **Programmieren in Paaren**
  - Die Programmierer arbeiten stets zu zweit am Code und diskutieren während der Entwicklung intensiv über Entwurfsalternativen
- **Gemeinsame Verantwortlichkeit**
  - Der gesamte Code gehört dem Team. Jedes Paar soll jede Möglichkeit zur Codeverbesserung jederzeit wahrnehmen. Das ist kein Recht sondern eine Pflicht.
- **Erst Testen**
  - Gewöhnlich wird jede Zeile Code durch einen Testfall motiviert, der zunächst fehlschlägt. Die Unit Tests werden gesammelt, gepflegt und nach jedem Kompilieren ausgeführt.
- **Design für heute**
  - Jeder Testfall wird auf die einfachst denkbare Weise erfüllt. Es wird keine unnötig komplexe Funktionalität programmiert, die momentan nicht gefordert ist.
- **Refactoring**
  - Das Design des Systems wird fortlaufend in kleinen, funktionserhaltenden Schritten verbessert.
- **Fortlaufende Integration**
  - Das System wird mehrmals täglich durch einen automatisierten Build-Prozess neu gebaut. Der entwickelte Code wird in kleinen Inkrementen und spätestens am Ende des Tages in die Versionsverwaltung eingecheckt und ins bestehende System integriert. Die Unit Tests müssen zur erfolgreichen Integration zu 100% laufen.

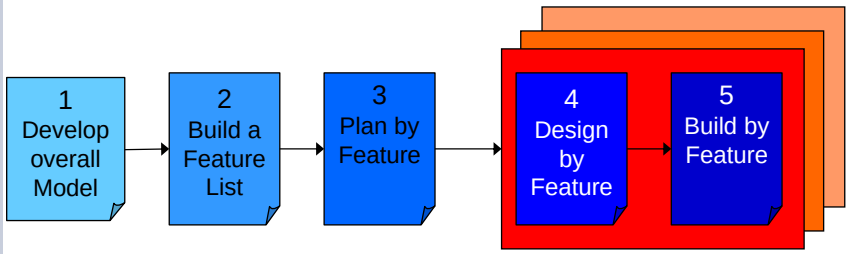
Techniken für das Management

- **Akzeptierte Verantwortung**
  - Das Management schreibt einem XP-Team niemals vor, was es zu tun hat. Stattdessen zeigt der Manager lediglich Probleme auf und läßt die Kunden und Programmierer selbst entscheiden, was zu tun gilt. Dies ist eine große, neue Herausforderung für das Management.
- **Information durch Metriken**
  - Eine der Hauptaufgaben des Managements ist es, dem Team den Spiegel vorzuhalten und zu zeigen, wo es steht. Dazu gehört unter anderem das Erstellen einfacher Metriken, die den Fortschritt des Teams oder zu lösende Probleme aufzeigen. Es gehört auch dazu, den Teammitgliedern regelmässig in die Augen zu schauen und herauszufinden, wo Hilfe von Nöten ist.
- **Ausdauerndes Tempo**
  - Softwareprojekte gleichen mehr einem Marathon als einem Sprint. Viele Teams werden immer langsamer bei dem Versuch, schneller zu entwickeln.

XP Project



Feature Driven Development



Projektplan

- User Stories / Features
- Meilensteine mit Datum
- User Stories / Features pro Meilenstein

UNIVERSITÄT SALZBURG