Entwicklungsmethoden

Prof. Dr. Josef M. Joller
jjoller@hsr.ch

ÜBERSICHT

Outline

Historische Aspekte

Ökonomische Aspekte

Wartungs-Aspekte

Spezifikations- und Design- Aspekte

Team Programming Aspekte

Das Objekt-orientierte Paradigma

Terminologie

Scope des Software Engineering

Historische Aspekte

- 1968 NATO Conference, Garmisch
- Ziel: Lösung der “Software Crisis”
- Software wird
  - verspätet abgeliefert
  - verteuert, über dem Budget
  - fehlerhaft

Scope des Software Engineering

Warum kann man nicht die Techniken aus anderen Fachgebieten auf die Software übertragen?

- Das Engineering ist nicht perfekt
- Komplexität
- Wartung

Ökonomische Aspekte

Wirtschaftliche Betrachung:

Die Entwicklungsmethode CMneu sei 10% schneller als die aktuelle.
Sollen wir auf die neue Methodik umstellen?

- Common Sense Antwort
  - natürlich!
- Software Engineering Antwort
  - überprüfen, welchen Einfluss die neue Methode CMneu auf die Wartung hat.

Wartungs-Askepte

Software Life Cycle

- Software Entwicklung wird geprägt durch drei fundamentale Grössen / Säulen:
  - das Life-Cycle Modell : Technik
  - die Individuen : Team
  - CASE Tools : Tools

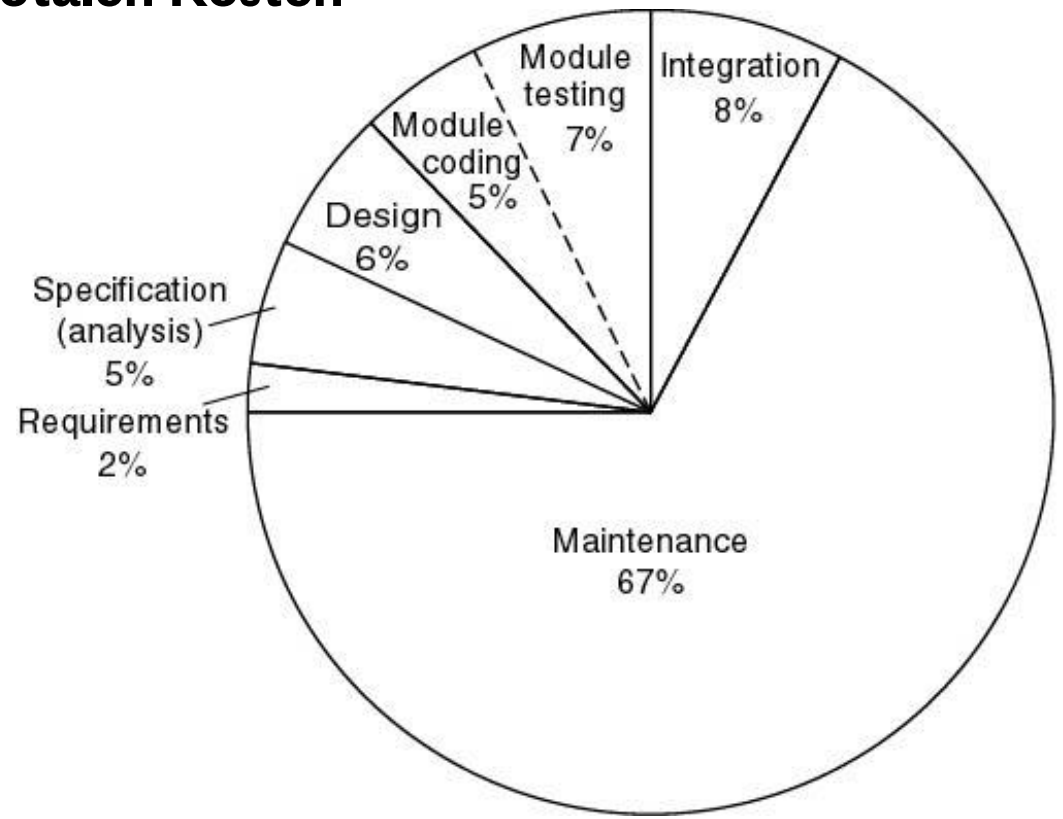
Life-Cycle Modelle

1. Anforderungs Phase
2. Spezifikations Phase
3. Design Phase
4. Implementations Phase
5. Integrations Phase (parallel zu 4)
6. Wartungs Phase
7. Ablösung

Geschätzte Kosten pro Phase

1976–1981 Daten

Wartung : 67% der totalen Kosten



Vergleich der Kosten pro Phase

	Projekte im Zeiraum 1976-1981	132 Projekte bei Hewlett-Paclard
Anforderungs- und Spezifikations / Analyse Phase	21%	18%
Design Phase	18	19
Implementations Phase	36	34
Integrations Phase	24	29

Gute und Schlechte Software

Gute Software wird gewartet - schlechte ersetzt man

Unterschiedliche Bestandteile der Wartung

- korrektive Wartung [ungefähr 20%]
- Erweiterungen
  - verbessernde Wartung [ungefähr 60%]
  - anpassende Wartung [ungefähr 20%]

Welchen Einfluss hat die neue Methode CM neu auf die Wartung?

- Um wieviel Prozent können wr den Wartungsaufwand reduzieren?

Fehler in der Spezifikation und Wartung

60 bis 70 Prozent der Fehler stammen aus der Spezifikations- und Design Phase

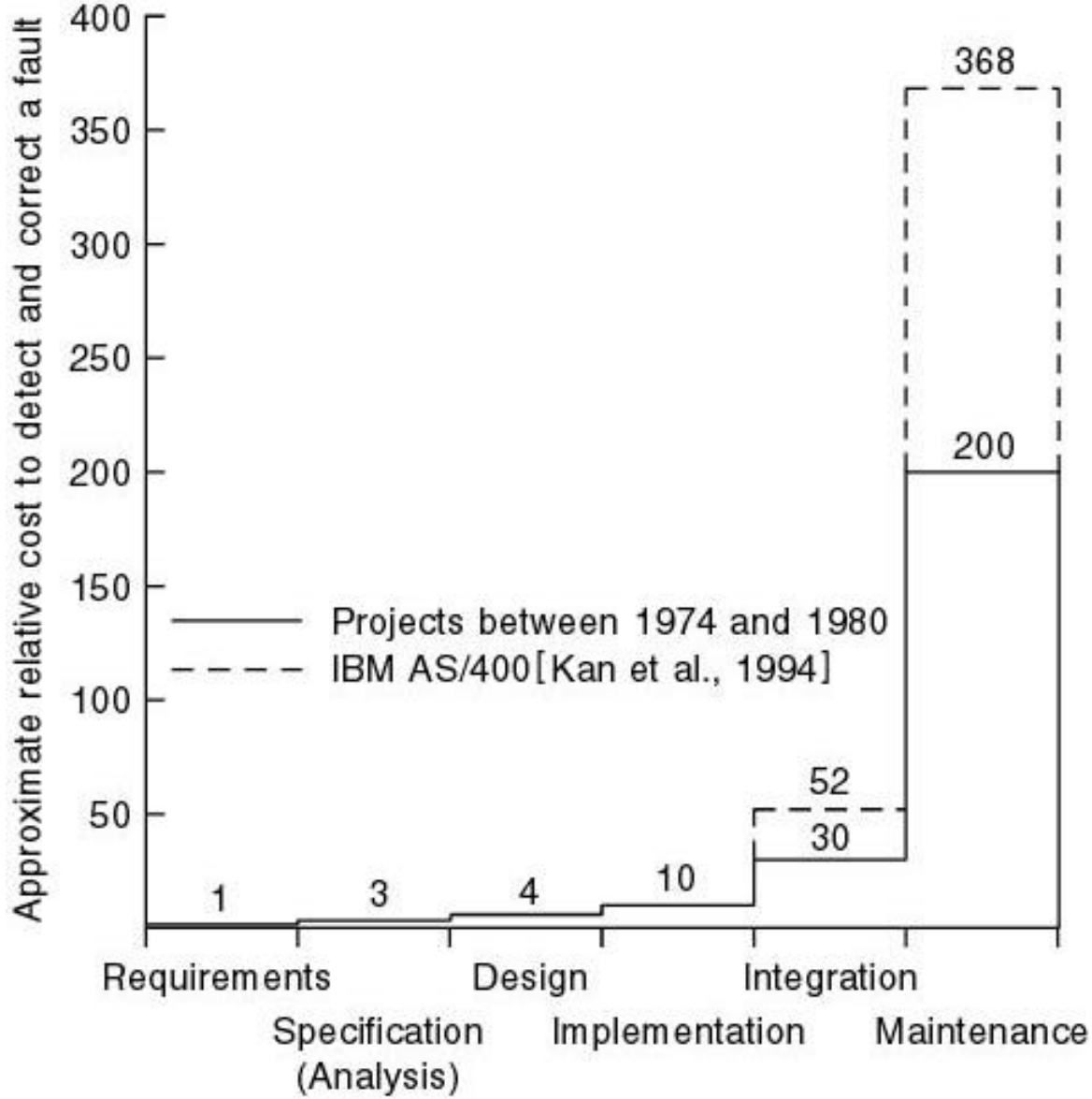
Daten von Kelly, Sherif und Hops [1992]

- 1.9 Fehler pro Spezifikationsseite
- 0.9 Fehler pro Seite Design
- 0.3 Fehler pro Seite Programmcode

Fehler am Ende der Design Phase einer neuen Version des Produkts

- 13% der Fehler stammen aus der Vorgängerversion
- 16% der Fehler stammen aus der neuen Spezifikation
- 71% der Fehler stammen aus dem neuen Design

Kosten für die Behebung nd Korrektur von Fehlern



Team Aspekte

Hardware ist billig

- die Software, die wir schreiben, ist so komplex, dass sie nicht mehr nur durch eine Person alleine erstellt werden kann.

Teams

- Interface Probleme : wer ist für welche Module zuständig
- Meetings
- Qualifikation
- Einsatz neuer Technologien
- schnell veränderliches Geschäftsumfeld

TDas Objekt-Orientierte Paradigma

Das strukturierte Paradigma war am Anfang sehr erfolgreich

- bei grösseren Projekten (> 50,000 LOC) traten Probleme auf

Wartung traditioneller Software ist aufwendig und teuer (heute bis zu 80% des Aufwandes)

Begründung:
strukturierte Methoden sind

- aktionsorientiert (Datenfluss,Prozesse)
- datenorientiert (Entity-Relationship Diagramm, Jackson)
- aber nicht beide Aspekte (Objekte schon)

Das Objekt-orientierte Paradigma

Daten und Aktionen / Prozesse sind gleich wichtig

Objekt:

- Software Komponente, welche Daten und Aktionen (jene, die mit den Daten arbeiten) werden zusammengefasst.

Beispiel:

- Bankkonto
  - Daena: mein Bankkonto (Kontostand)
  - Aktions: enzahlen, abheben, überweisen, ...

Wichtige Aspekte der Objekt-orientierten Lösung

Objekte sind konzeptuell unabhängig voneinander

- Kapselung

Physikalische Unabhängigkeit

- Information Hiding

Einfluss auf die Entwicklung

- Objekte haben physisch entsprechende Objekte

Einfluss auf die Wartung

- Die einzelnen Objekte lassen sich unabhängig voneinander warten

Responsibility-Driven Design

...auch als "Design by Contract" bekannt

Bestellen von Blumen für die Grossmutter

- Gärtnerei oder Blumenladen anrufen
- Auftrag durchgeben
- Gärtnerei / Blumenladen weiss was zu tun ist
- Information Hiding : wie geschieht die Lieferung?

Objekt-orientiertes Paradigma

- "Senden einer Message an eine Methode [Aktion] eines Objekts"

Übergang von der Analyse zum Design

Strukturiertes Paradigma	Objekt-orientiertes Paradigma
1. Anforderungs-Phase	1. Anforderungs-Phase
2. Spezifikations / Analyse Phase	2. Objekt- orientierte Design Phase
3. Design Phase	3. OO Design Phase
4. Implementations Phase	4. OO Programmier Phase
5. Integrations Phase	5. Integrations Phase
6. Wartungs Phase	6. Wartungs Phase
7. Ablösung	7. Ablösung
Analyse : WAS Design : WIE	Objekte bereits möglichst früh

Unterscheidung von Analyse und Design

System-Analyse

- was soll gemacht werden

Design

- wie soll etwas gemacht werden
- Architektur-Design : welche Module
- Detail-Design : Design der Module

Im OO Bereich wird der Unterschied relativiert

OO Analyse

- Was muss ein Objekt können
- Welche Objekte kennt das System

OO Design

- wie agiert das Objekt
- Design der Objekte

Etwas ausführlicher

Strukturiertes Paradigma	OO Paradigma
2. Spezifikations (Analyse) Phase - WAS soll das Produkt können	2'. OO Analyse Phase - was soll das Objekt können - welche Objekte existieren
3. Design Phase - Architektur-Design - Detailldesign	3'. OO Design Phase - detailliertes Design der Objekte
4. Implementationsphase - Programmierung	4'. Implementationsphase - OO Implementierung