
Vorlesung „Methoden des Software Engineering“

Block R „Rahmen“

Aktivitäten der Software-Entwicklung

Martin Wirsing

Einheit R.2, 21.10.2004

Ziel heute

-
- Historische Entwicklung des Software Engineering
 - Software-Entwicklungsaktivitäten

Anfänge des Software Engineering

Sehr häufiges Scheitern von Softwareprojekten führt zum Schlagwort

Softwarekrise

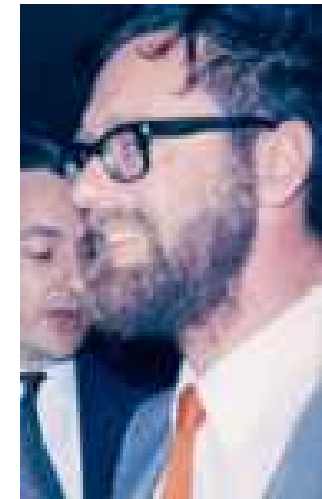
Und zur Konferenz über

Software Engineering

NATO Konferenz,
Garmisch-Partenkirchen
1968



F.L. Bauer



E.W. Dijkstra

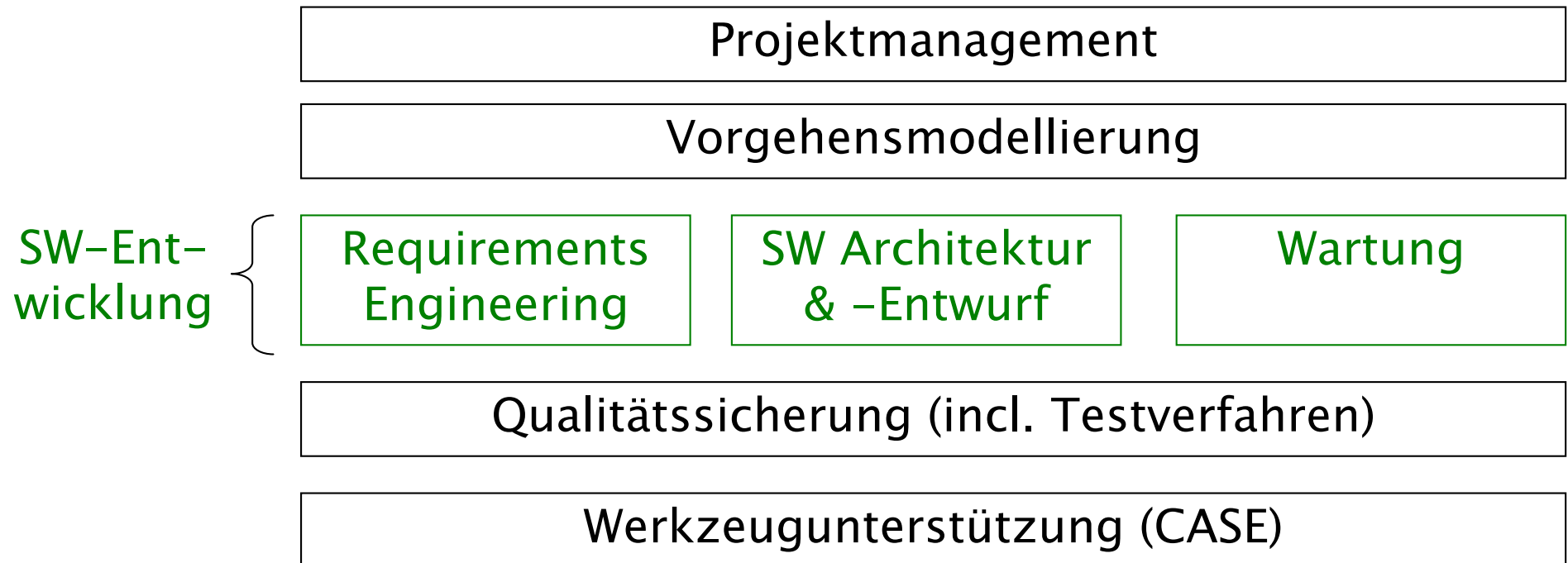
Software Engineering (Wiederholung)

- Software Engineering bedeutet die Anwendung von systematischen, disziplinierten und quantifizierbaren Ansätzen zur Entwicklung, Betrieb und Wartung von Software.

IEEE Std. 610.12 (1990)

Gebiete des Software Engineering

Allgemeine Einteilung:



Spezielle Methoden hängen ab von

- Systemart: z.B. reaktiv, parallel, eingebettet, web-basiert, ...
- Anwendungsgebiet

Jede Software-Entwicklungsmethode

- umfasst

Werkzeugunterstützung

Vorgehensmodellierung

legt fest Schritte, Reihenfolge, erwartete Ergebnisse

Verfahren

zur Konstruktion, Analyse, Transformation

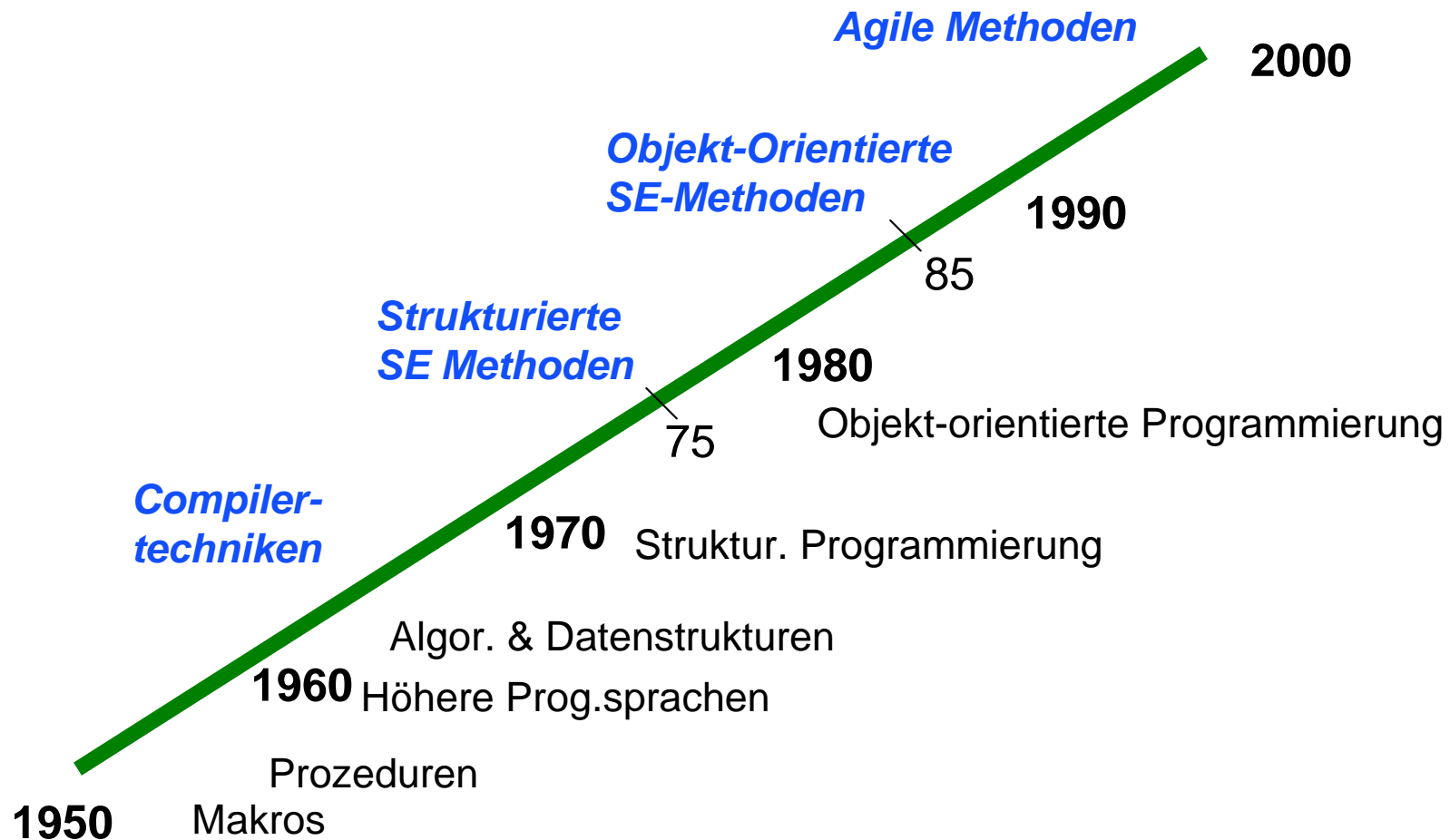
Notation

Syntax von Dokumenten, Diagrammtypen, (Semantik)

- basiert auf

Grundprinzipien

Historische Entwicklung: Programmierkonstrukte und SE-Methoden



Historische Entwicklung: Programmierung und SW Engineering Fragestellungen

	<i>1960 + 5 Program- mieren irgendwie</i>	<i>1970 + 5 Program- mieren im Kleinen</i>	<i>1980 + 5 Program- mieren im Großen</i>	<i>1990 + 5 Program- mieren Verteilter Systeme</i>	<i>2000 + 5 Program- mieren in der Welt</i>
Spezifikation	Präziser Ge- brauch natür- licher Sprache	Einfache Ein/Ausgabe Spez.	Systeme mit Komplexen Spez.	Spez. Verteil- ter Systeme	Spez. globaler, eingebetter Systeme
Entwurf	Kleine Pro- gramme	Algorithmen	Systemstruk- turierung	Subsystem- interaktionen	Subsystem- interaktionen
Daten	Symbolische Daten	Datenstruk- turen & Typen	Lang-lebende Datenbanken	Kompo- nenten	Agenten
Kontrolle	Einfache Kon- trolle (Goto)	Progr. 1-mal ausführen & terminieren	Nichttermin. Progr.	Kommu- nizierende Programme	Mobile, pervasive Programme

Software-Entwicklungsaktivitäten

Anforderungserwerb	Sammlung der Nutzeranforderungen
Anforderungsanalyse	Modellierung und Spezifikation der Systemanforderungen („Was“)
Architektur & Entwurf	Modellierung und Spezifikation einer Lösung („Wie“)
Implementierung	Konstruktion einer Software-Lösung
Test	Validierung der Lösung bzgl. der Anforderungen
Wartung	Reparatur von Fehlern und Anpassung an neue Anforderungen

Bemerkung: Dies sind keine sequenziellen Phasen!

Anforderungserwerb

- Benutzeranforderungen werden häufig informell beschrieben durch
 - Features und
 - Nutzungsszenarios
- Obwohl Anforderungen schriftlich dokumentiert sind, können sie sein:
unvollständig, mehrdeutig oder sogar **inkorrekt**.
- Anforderungen ändern sich! Weil
 - sie zunächst nicht angemessen ermittelt oder ausgedrückt wurden
 - Nutzer- und Geschäftsbedürfnisse sich während des Projekts ändern können
- **Validierung** wird überall im SW-Lebenszyklus benötigt, nicht nur wenn das endgültige System ausgeliefert wird
 - Wichtig: Feedback und mögliche Änderungen im Projektplan vorsehen
 - Frühe Prototyperstellung (z.B. eines UI) kann zur Klärung der Anforderungen beitragen

Anforderungsanalyse

- Mit **Analyse** bezeichnet man den Prozess des Spezifizierens, **was** das geplante System erfüllen soll
Das Resultat der Analyse ist das (Anforderungs-) **Spezifikationsdokument**.
- Erfüllt die **Anforderungsspezifikation** die aktuellen Wünsche des Benutzers?
- Objekt-Orientierte Analyse
- Zielorientierte Analyse

Entwurf

- **Entwurf** ist der Prozess des **Spezifizierens, wie** das geforderte Systemverhalten (durch Softwarekomponenten) realisiert wird.
- Resultat sind Architektur- und detaillierte Entwurfskomponenten.
- **Objekt-orientierter Entwurf** konstruiert Lösungen, die beschreiben:
 - wie Systemoperationen durch interagierende Objekte implementiert werden
 - welche Beziehungen und Vererbungsstrukturen Klassen untereinander besitzen
 - welche Attribute und Operationen zu den Klassen gehören
- Entwurf ist ein iterativer Prozess

Conway's Law

“Organizations that design systems are constrained to produce designs that are copies of the communication structures of these organizations”

Implementierung und Test

- **Implementierung** ist die Aktivität der Konstruktion einer Software-Lösung für die Benutzeranforderungen
- **Testen** ist der Prozess der Validierung der Lösung gegenüber den Anforderungen.

Das Resultat von Implementierung und Test ist eine vollständig dokumentierte und validierte Lösung

- **Entwurf, Implementierung and Test** sind iterative Aktivitäten
- Testen und Implementierung gehen Hand in Hand:
Idealerweise werden die Testfälle **vor** Entwurf und Implementierung geschrieben (Test-first programming, XP)

Wartung

- Mit **Wartung** bezeichnet man den Prozess der Änderung des Systems, nachdem es ausgeliefert wurde.
 - **Korrigierende Wartung**: Fehler identifizieren und beheben
 - **Adaptive Wartung**: Die existierende Lösung an neue Plattformen anpassen
 - **Perfektionierende Wartung**: Optimierung, Implementierung neuer Anforderungen
- “Wartung” erfordert:
- **Konfigurations- und Versionsmanagement**
- **Re-engineering**: Redesign und Restrukturierung (Refactoring)
- **Anpassung** aller Analyse-, Entwurfs- und Benutzerdokumentation
- **Wiederholbare, automatisierte Test ermöglichen Evolution und Restrukturierung**

Zusammenfassung

- Software Engineering umfasst folgende **Gebiete**:
 - Projektmanagement
 - Vorgehensmodelle
 - Software-Entwicklung
 - Software-Qualitätsmanagement
 - Werkzeugunterstützung (CASE)

- Software-Entwicklung umfasst folgende **Aktivitäten**:
 - Anforderungserwerb
 - Anforderungsanalyse
 - Architektur & Entwurf
 - Implementierung
 - Test
 - Wartung