

CMMI & Software-Qualität

Prof. Dr. Wolfgang Pree

Universität Salzburg und
www.preeTEC.com

Dr. Ernest Wallmüller

Qualität & Informatik www.itq.ch

DI. Johannes Neuhold

AVL List GmbH www.AVL.com

CMMI & Software-Qualität, 10. März 2006



9:15 - 10:00 h Prof. Dr. Wolfgang Pree,
Potenzial von Prozessverbesserungen und
deren Grenzen.

10:00 - 10:30 h Dr. Ernest Wallmüller,
Wesentliche Aspekte von CMMI.

10:30 -10:45 h DI Johannes Neuhold
CMMI-Einführung bei der AVL List GmbH, Graz.

10:45 - 11:00 h



11:00 - 12:00 h Diskussion und weitere Vorgehensweise
kaltes Buffet

Potenzial von Prozessverbesserungen und deren Grenzen

Die Universalmaschine Computer macht Software allgegenwärtig



Flugzeug-/Raketensteuerungen



ca. 70 Prozessoren
im Auto



Qualität ist wesentlich schlechter als bei anderen Produkten

- **Softwarefehler/-mängel** mit drastischen Auswirkungen:
 - | Y2K, €-Umstellung
 - | fehlerhafte Finanztransaktionen
 - | Abstürze (zB Ariane: \$ 800 Mio.)
 - | . . .

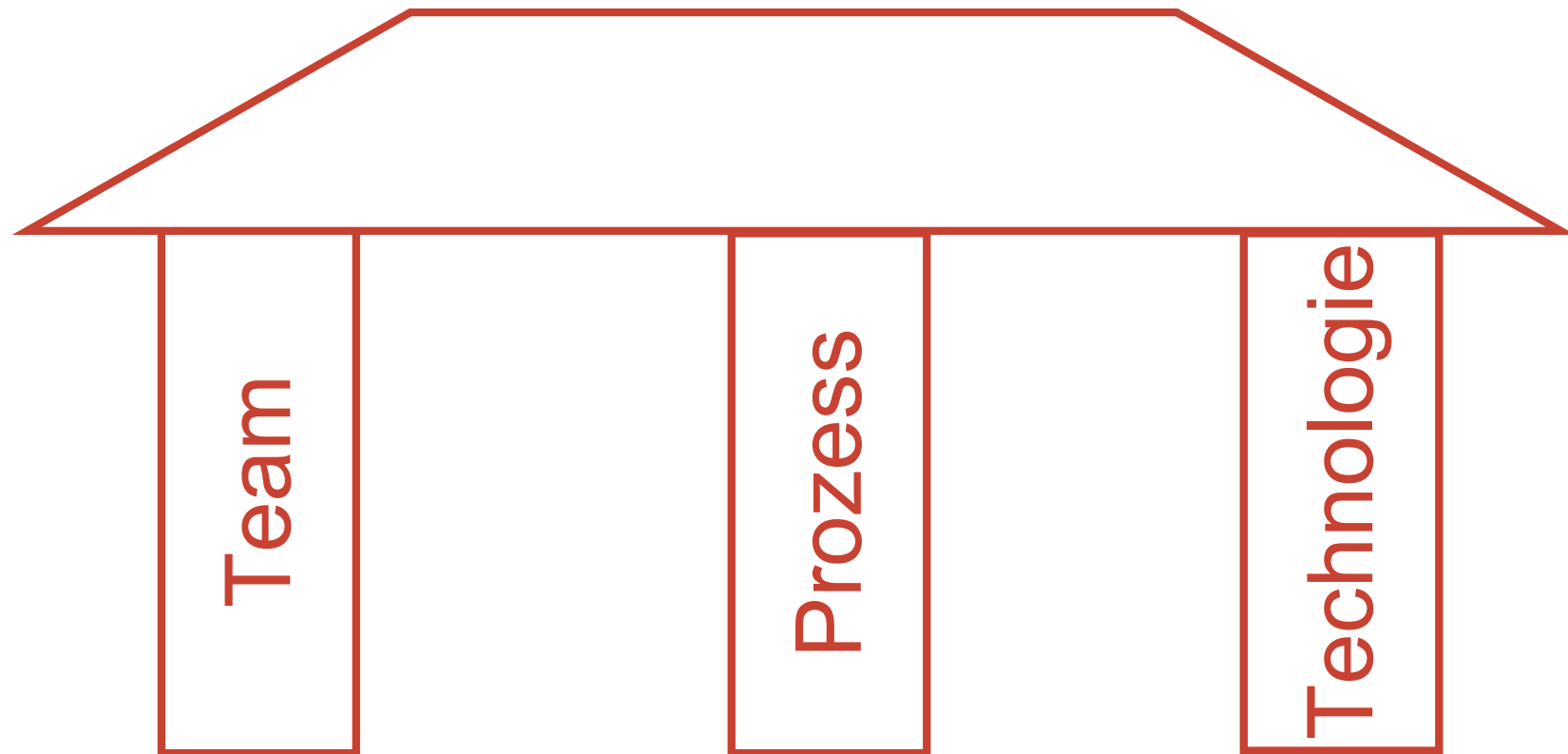
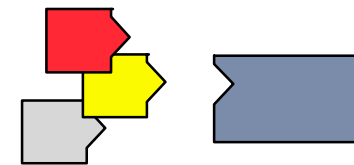


Die Probleme bei der Herstellung von Software resultieren aus der Komplexität der zu realisierenden Produkte

- Spezifikation der Anforderungen ← Prototyping
- Beherrschung der Komplexität ← Programmiermodelle
- Wiederverwendung/PlugIns, Änderbarkeit und Erweiterbarkeit ← Entwurfsmuster Frameworks
- Automatisierung im Herstellungsprozeß
- Portabilität
- Dokumentation
- Produktergonomie (Mensch-Computer-Schnittstelle) ← Psychologie (zB Piaget)
- Projektorganisation u. -kontrolle
- Qualitätssicherung und -bewertung
- Personenunabhängigkeit
- Kostenabschätzung

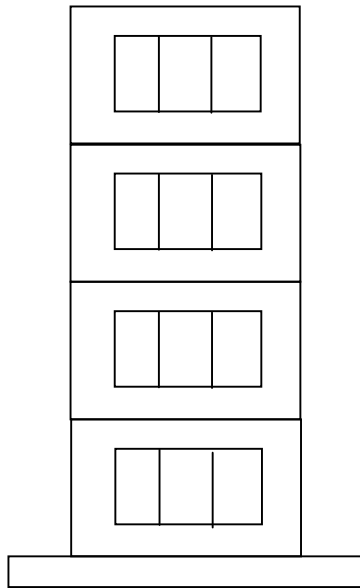


Die drei Säulen qualitativ hochwertiger Software



Die besten Prozesse nützen
kaum, wenn die Software-
Architektur nicht passt

Entspricht die Umsetzung dem (Architektur-)Plan?



Plan



Realität

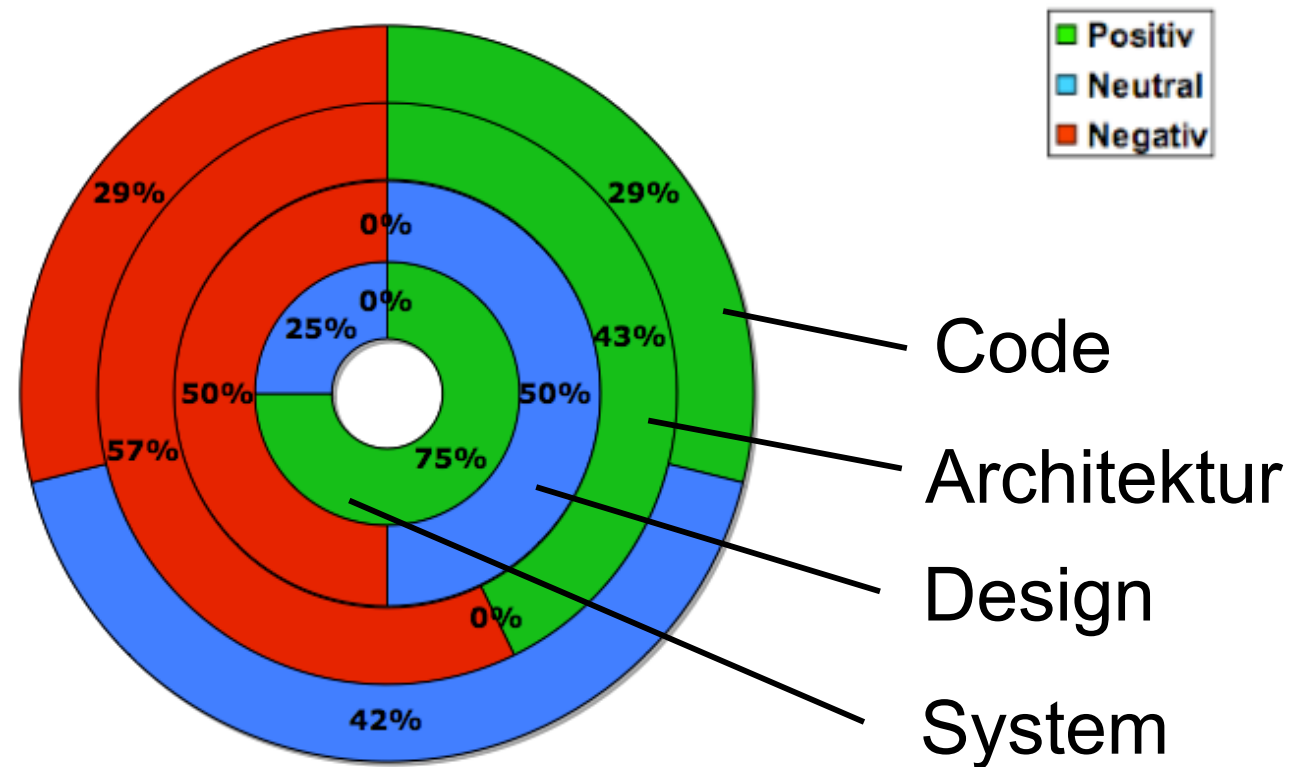
Architektur-Bewertung

- Welche Software-Qualitätsmerkmale sind relevant?
 - Erweiterbarkeit, dynamische Anpassbarkeit (zB durch Scripting), Sicherheit, etc.
- Wie entspricht die umgesetzte Architektur den Erwartungen?
- Ist die Architektur komponenten-orientiert?

Implementierungs-Bewertung (zum Teil automatisch überprüfbar)

- Verständlichkeit und Lesbarkeit
 - Einhalten von Codier-Richtlinien, Kommentare, Strukturierung, Code- und File-Duplikate, Namenskonventionen, Länge von Funktionen bzw. Methoden, nicht erreichbarer Code, etc.
- Robustheit
 - Exception-Handling,
 - Information Hiding,
 - Speicherverwaltung
- Funktionsumfang

Beispiel für die Zusammenfassung einer Bewertung



The race belongs to the swift

