

## Teil II – DO

# 20. Qualitätssicherung und -management

Prof. Dr. rer. nat. Uwe Aßmann  
Lehrstuhl Softwaretechnologie  
Fakultät Informatik  
Technische Universität Dresden  
[http://st.inf.tu-  
dresden.de/teaching/swm](http://st.inf.tu-dresden.de/teaching/swm)  
2016-0.3, 04/06/16

- 1) Warum man QS braucht
  - 1) Qualitätsbegriff
- 2) Konstruktive Qualitätssicherung
  - 1) Zertifizierung
- 3) Analytische QS
  - 1) Analyseverfahren
  - 2) Testverfahren
- 4) Aufgabenmanagement



# Referenzierte Literatur

- ▶ [Wallmüller] Wallmüller, E.: Software-Qualitätssicherung in der Praxis; Hanser Verlag 1990 sowie 2. Auflage erschienen 2001
- ▶ [Trauboth] Trauboth; H.: SW-Qualitätssicherung; Oldenbourg Verlag 1996
- ▶ [BalzertSM] Balzert, H.: Lehrbuch der Softwaretechnik – Softwaremanagement. Spektrum Verlag 2008
- ▶ [LLL08] Rüdiger Lincke, Jonas Lundberg, and Welf Löwe. Comparing software metrics tools. Int. symposium on Software testing and analysis, ISSTA '08, pages 131-142, New York, NY, USA, 2008. ACM.
  - <http://dx.doi.org/10.1145/1390630.1390648>
- ▶ <http://www.qz-online.de/>
- ▶ American Society for Quality <http://www.asq.org/>
- ▶ M. Gharbi, A. Koschel, A. Rausch, G. Starke: Basiswissen für Softwarearchitekten. dpunkt-Verlag. 2013. Ausbildungsmaterial zum iSAQB-Standard für Zertifizierung von Softwarearchitekten
- ▶ codeBeamer Werkzeug als Beispiel für Requirements- und Qualitätsmanagement
  - <http://intland.com/documents/CodeBeamer-Requirement-Management-2013-Low.pdf>

# Standards zur QS

Norm	Erläuterung
<b>DIN 55350-11</b>	Definition der Qualitätseigenschaften von Softwareprodukten
<b>ISO/IEC 14598-1</b>	Modell für Erkennen der Qualität, Bewertung
<b>ISO/IEC 9126-1</b>	6 Hauptkategorien für Softwarequalität: u. a. Usability
<b>DIN 66271</b>	Softwarefehler und ihre Beurteilung durch Kunden und Lieferanten
<b>ANSI/IEEE 829</b>	Standard for Software-Test-Dokumentation
<b>DIN 66270</b>	Bewerten von Softwaredokumenten, Qualitätsmerkmale
<b>ANSI/IEE 1008</b>	Standard for Unit Testing, Modultest
<b>BS/ISO/IEC 25000</b>	Anforderungen an Software-Produkt-Qualitätsanforderungen und Evaluation
<b>EN ISO/IEC 17024</b>	Konformitätsbewertung – Allgemeine Anforderungen an Stellen, die Personen zertifizieren

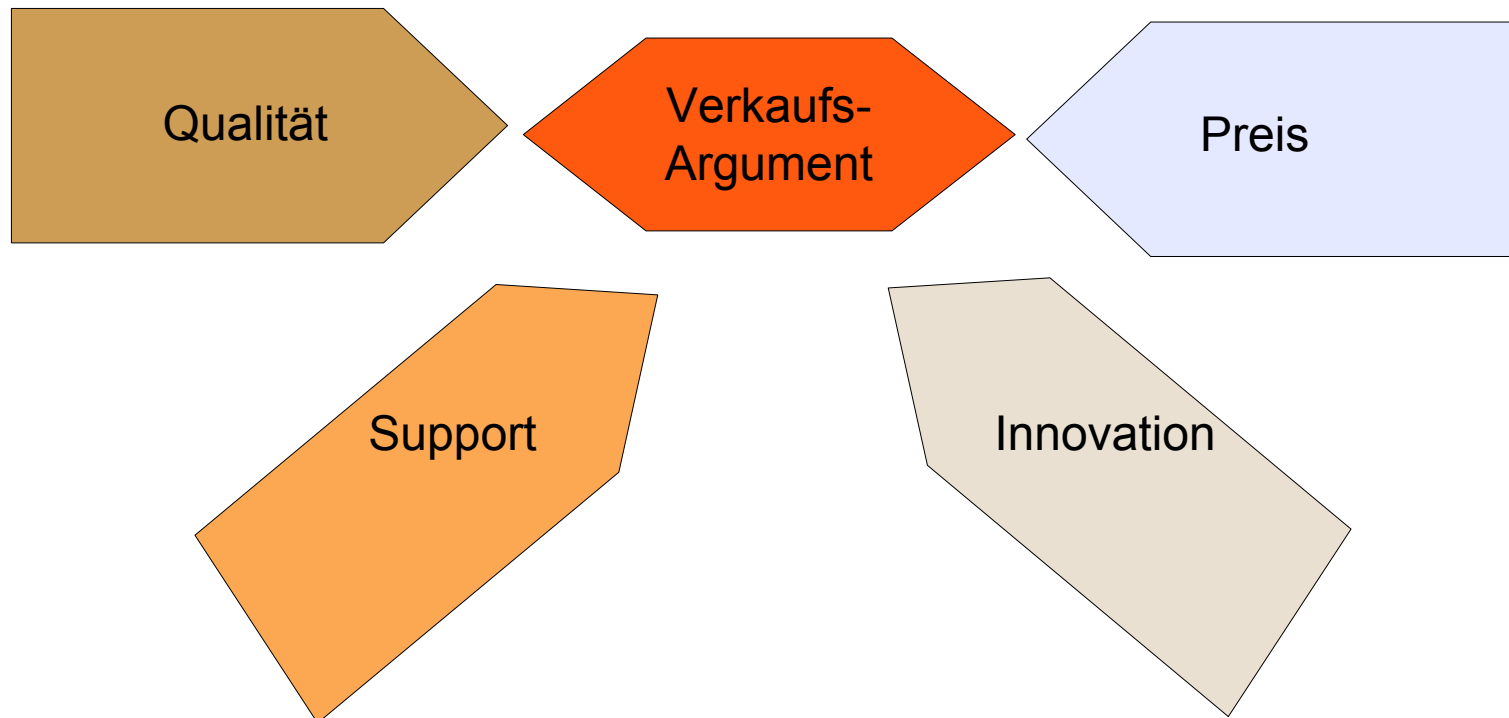


# 20.1 Warum man Qualitätssicherung und -management braucht



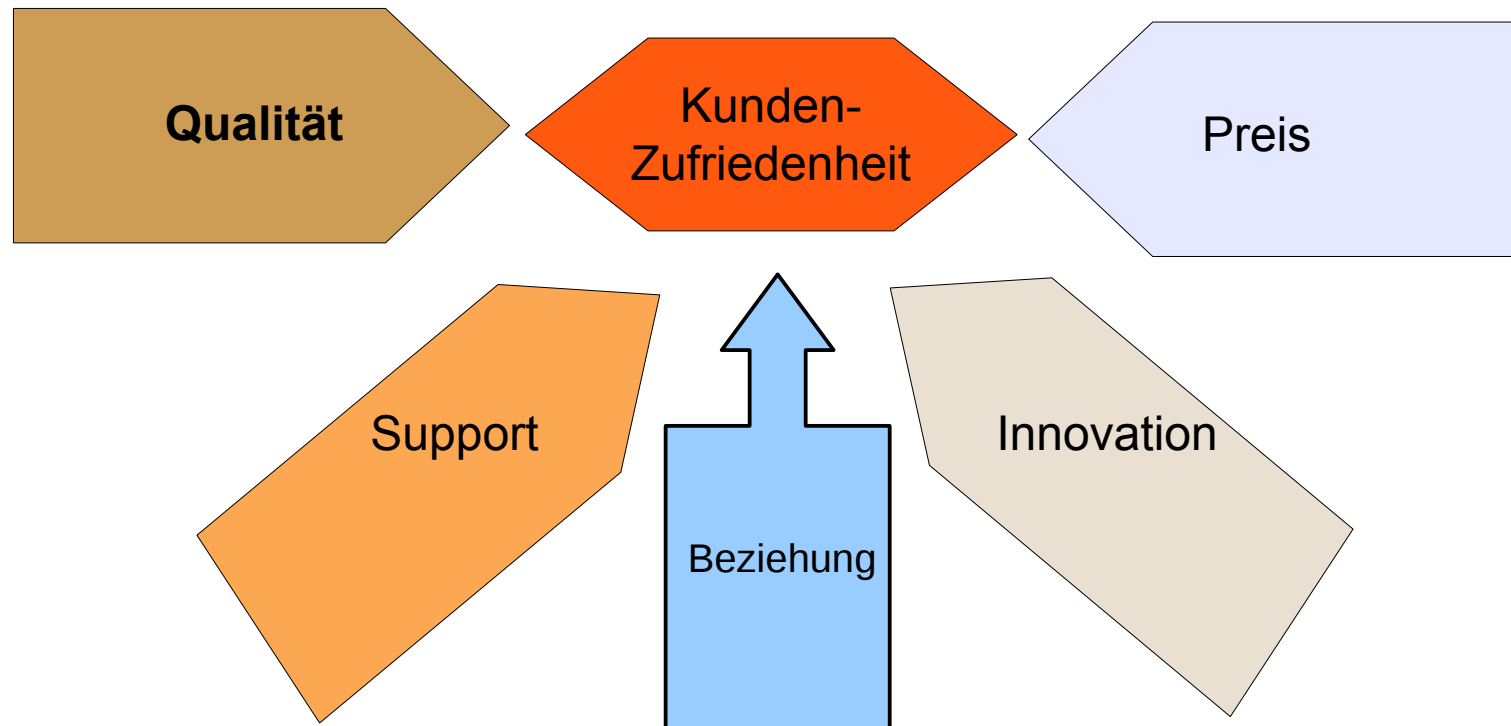
# Über welche Eigenschaft verkauft man Software?

95% aller Dresdner Softwarefirmen verkaufen über die Qualität.  
[KMU-Studie]



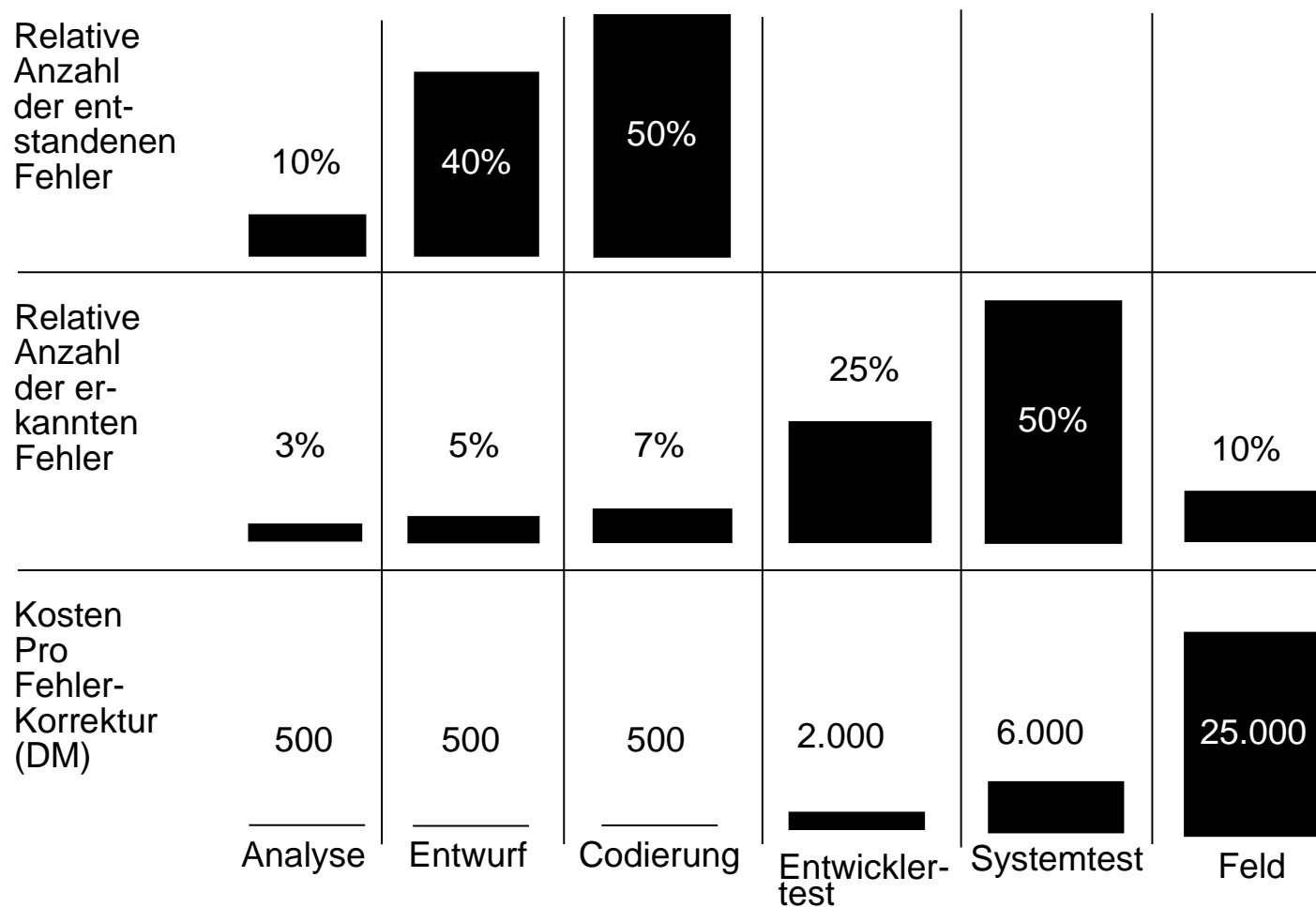
# Stammkundengeschäft

Der wesentliche Erfolgsfaktor, einen Stammkunden zu erwerben und zu halten, ist eine hohe Qualität der ausgelieferten Software und Dienstleistung.



# Der Gegner: Fehleranzahl und -kosten

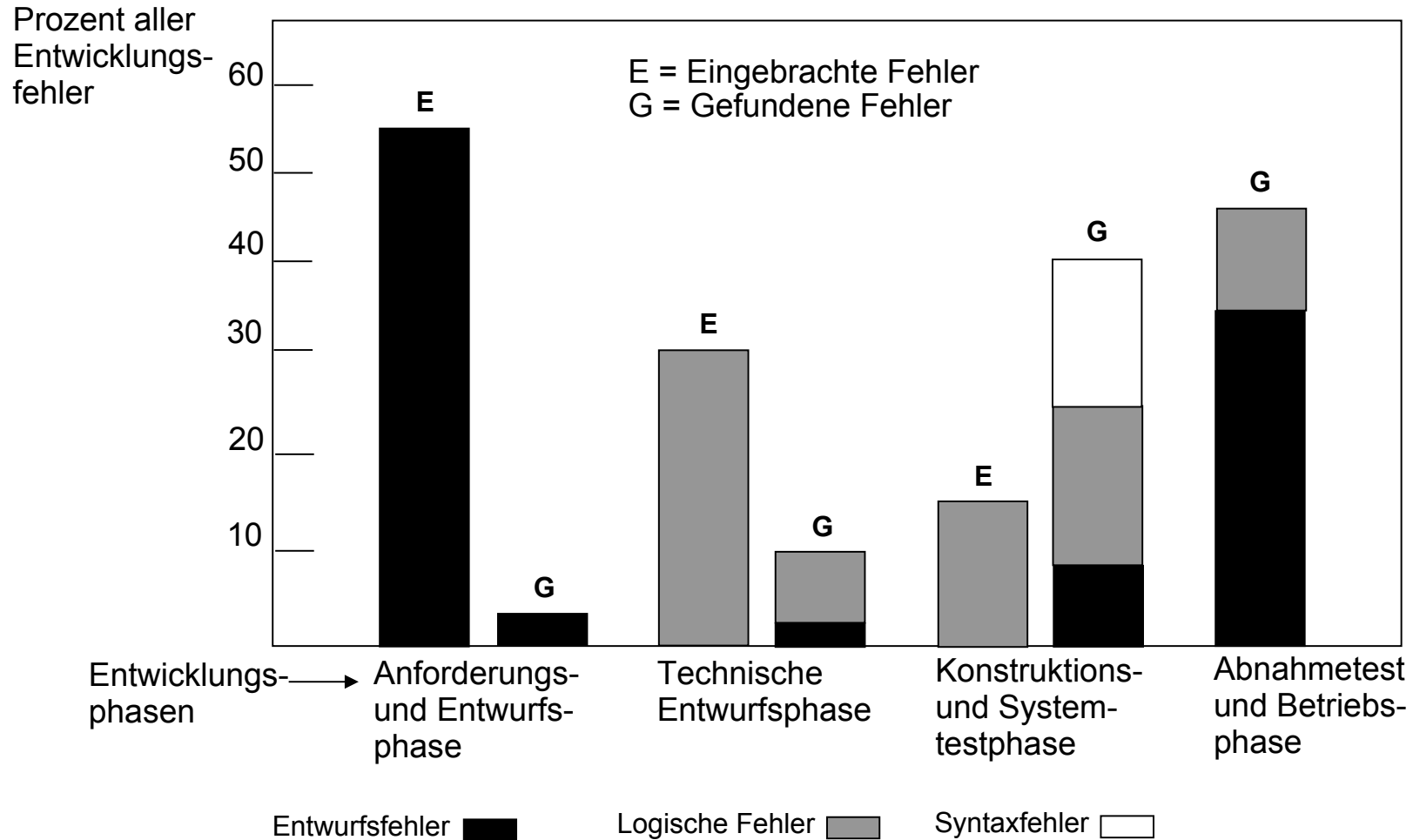
## (Empirische Daten)



**Quelle:** Liggesmeyer u. a.: Qualitätssicherung software-basierter technischer Systeme; Informatik-Spektrum 21(1998) S. 249 - 258



# Fehlerbeseitigungskosten

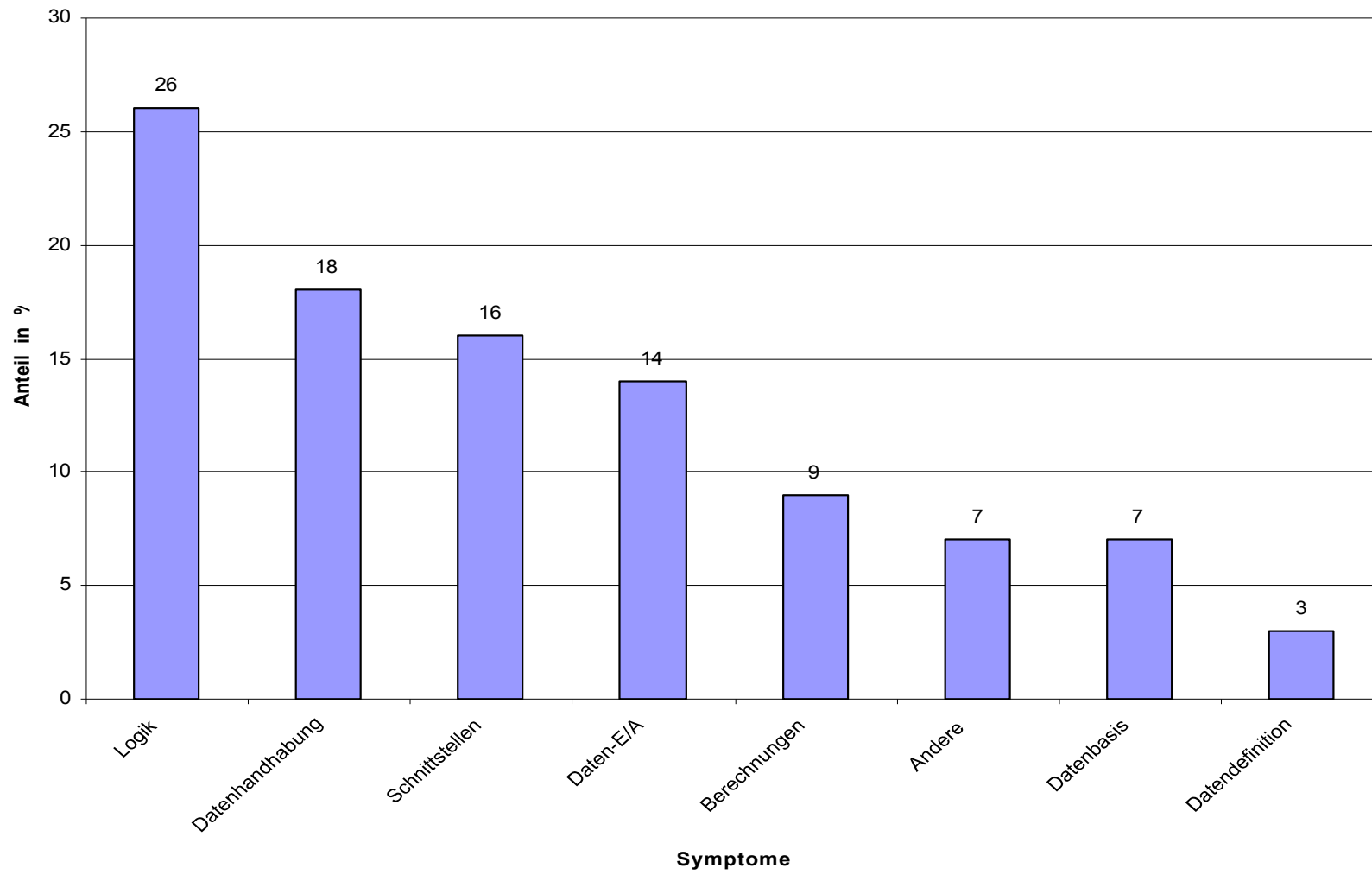


**55% aller Fehler entstehen in der Anforderungs- und Entwurfsphase**

Quelle: [ BalzertSM ]

# Fehlerverteilung

Fehlerverteilung in %



Quelle: [Trauboth]

## **1. Kundenorientierung**

*(Bedürfnisse erfüllen, übertreffen, vorwegnehmen)*

## **2. Führung**

*(Leiten durch Vorbild, Beachtung von Interessengruppen, Entwickeln einer Vision)*

## **3. Einbeziehung der Menschen**

*(Problemlösungskompetenz entwickeln, Initiative zu Verbesserungen)*

## **4. Prozessorientierter Ansatz**

*(Tätigkeiten und Ressourcen als Prozess darstellen, effiziente Prozesse)*

## **5. Systemorientierter Managementansatz**

*(Wechselwirkungen zwischen Einzelprozessen, Koordination von Zuständigkeiten)*

## **6. Ständige Verbesserung**

*(„Wer aufhört besser zu werden, hat aufgehört gut zu sein“)*

## **7. Sachlicher Ansatz zur Entscheidungsfindung**

*(Analysen, Mitarbeiter-Umfragen, Vorschläge)*

## **8. Lieferantenbeziehungen zum gegenseitigen Nutzen**

*(transparente Kommunikation, Verständigung über gemeinsame Ziele)*

# Total Quality Management (TQM)

## (Ständige Qualitätsverbesserung)

- ▶ zuerst eingeführt in Japan, abgeleitet vom PDCA (Deming)
  - **horizontal:** über alle Abteilungen hinweg und
  - **vertikal:** über alle Leitungsebenen
- ▶ Ziel: Kundenzufriedenheit in CoTiQQ (cost, time quality, quantity)

Erfahrungswerte	Konsequenzen <sup>1)</sup>
<ul style="list-style-type: none"><li>• <b>Zufriedener Kunde:</b> erzählt positives Erlebnis 4 - 8 mal weiter</li><li>• <b>Unzufriedener Kunde:</b> erzählt „Geschichte“ 9 - 16 mal weiter ==&gt; unkontrollierter negativer Multiplikator</li><li>• <b>Neukunden</b> zu gewinnen ist schwieriger und aufwendiger, nämlich 6 mal teurer als Bestandskunden gut zu betreuen</li></ul>	<ul style="list-style-type: none"><li>• Jeder <b>unzufriedene Kunde</b> besitzt keine Kundenbindung und wird kein Stammkunde werden</li><li>• <b>Beschwerde-(Issue-)Management</b> heißt, aus einem unzufriedenen Kunden einen zufriedenen Kunden zu machen und Kundenbindung zu erzeugen</li></ul>

⇒ **Nur ein zufriedener Kunde bleibt auch ein Kunde**

<sup>1)</sup> **Quelle:** Knöll u. a.: Entwicklung und Qualitätssicherung von Anwendungssoftware; Spektrum Verlag 1996

## **14 Führungspflichten** zur Verbesserung der Qualität und Produktivität (nach Deming):

- ▶ Für Unternehmen
  - Unternehmenspolitik muss Willen zur Verbesserung klar aufzeigen.
  - Prüfung primär, um die Prozesse zu verbessern und die Kosten zu reduzieren.
  - Der Preis eines Produktes allein zählt nicht, sondern das unablässiges Verbessern von Prozessen und Produkten.
  - Institutionalisierung des Trainings, insbesondere Training on the job.
  - Leiterschaft ist zu institutionalisieren.
  - Barrieren zwischen Organisationseinheiten abreißen.
  - Das Qualitätsmanagementsystem kontinuierlich verbessern.
  - Leistungsquoten beseitigen.
- ▶ Für Belegschaft
  - Adäquate Einstellung zur Qualität.
  - Angst abbauen, Vertrauen und Klima für Innovation schaffen.
  - Förderung von Ausbildung und Selbstverbesserung für jeden im Unternehmen.
  - Andere am Erfolg teilhaben lassen.

# Experiment von Weinberg

5 Gruppen entwickeln ein Programm mit identischen funktionalen Anforderungen u. einer zusätzlichen, für alle Gruppen unterschiedlichen nicht-funktionalen Anforderung. Die erzielte Qualität wird auf einer Skala von 1 (sehr gut) bis 5 (schlecht) gemessen.

**Ziel:**

**Optimiere ...**  
Erstellungsaufwand  
Ausgaben

**Qualität der Ergebnisse**

	Erstellungsaufwand	Anzahl Anweisungen	Speicherbedarf	Klarheit des Programms	Klarheit der Ausgaben
Erstellungsaufwand	1	4	4	5	3
Anzahl Anweisungen	2-3	1	2	3	5
Speicherbedarf	5	2	1	4	4
Klarheit d. Programms	4	3	3	2	2
Klarheit d. Ausgaben	2-3	5	5	1	1

- Qualität muss quantifizierbar sein, damit Anforderungserfüllung gemessen werden kann
- auch nicht funktionale Anforderungen sind erreichbar





## 20.1.2 Qualitätsbegriff



# Was ist Qualität?

**Qualität** ist die Gesamtheit von Eigenschaften und Merkmale eines Produkts oder einer Tätigkeit, die sich auf deren *Eignung* zur Erfüllung gegebener Erfordernisse bezieht. [DIN 55350, Teil 11, Norm für Qualitätsmanagement]

**Quality** is fitness for use.

- ▶ **Qualität**
  - **des Produkts (Produktqualität)**
  - **des Entwicklungsprozesses (Prozessqualität)**
  - **der Beteiligten (Personenqualität)**
- ▶ **Qualitätsmerkmale:**
  - **Teilmerkmale (Kriterien)**
  - **Elementarmerkmale**
  - **Indikatoren (Metriken)**







# Unterteilung der Qualitätsmerkmale nach ISO/IEC 25000, früher DIN ISO 9126

## Merkmal

- **Funktionalität**
- **Zuverlässigkeit**
- **Benutzbarkeit**
- **Effizienz**
- **Änderbarkeit**
- **Übertragbarkeit**

## Teilmerkmale

Richtigkeit (Korrektheit)  
Angemessenheit  
Interoperabilität  
Ordnungsmäßigkeit (governance)  
Sicherheit

Reife  
Fehlertoleranz  
Wiederherstellbarkeit

Verständlichkeit  
Erlernbarkeit  
Bedienbarkeit

Zeitverhalten  
Verbrauchsverhalten

Analysierbarkeit  
Modifizierbarkeit  
Prüfbarkeit  
Stabilität

Anpassbarkeit  
Installierbarkeit  
Austauschbarkeit,  
Konformität (gegenüber Normen)

## Merkmalsbeschreibung

Fähigkeit des Systems die geforderten Anforderungen zu erfüllen

Einhaltung eines Leistungsniveaus unter festgelegten Bedingungen über einen definierten Zeitraum

Aufwand zur Benutzung der Software durch unterschiedliche Benutzergruppen

Benötigte Zeit und Verbrauch Betriebsmitteln für Aufgabe

Maß für Möglichkeit der Modifizierung von Software auf Basis interner und externer Einflüsse

Maß für Offenheit und Portabilität von Software zur Lauffähigkeit auf anderen Soft- und Hardwaresystemen

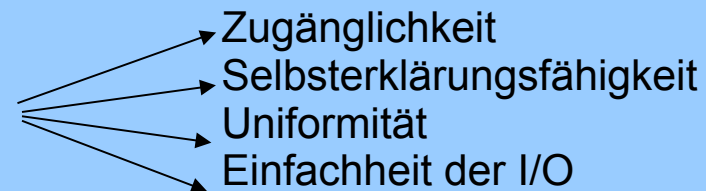
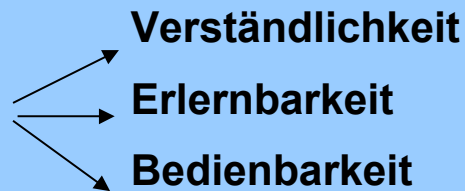
# Qualitätsmerkmale ISO/IEC 25000

## Merkmale

## Teilmerkmal e

## Elementar- merkmale

**Bsp.: Benutzbarkeit**



***jeweils***  
**Zeit** und  
**Zufrieden-**  
**heit**

# FURPS Classification of Requirements

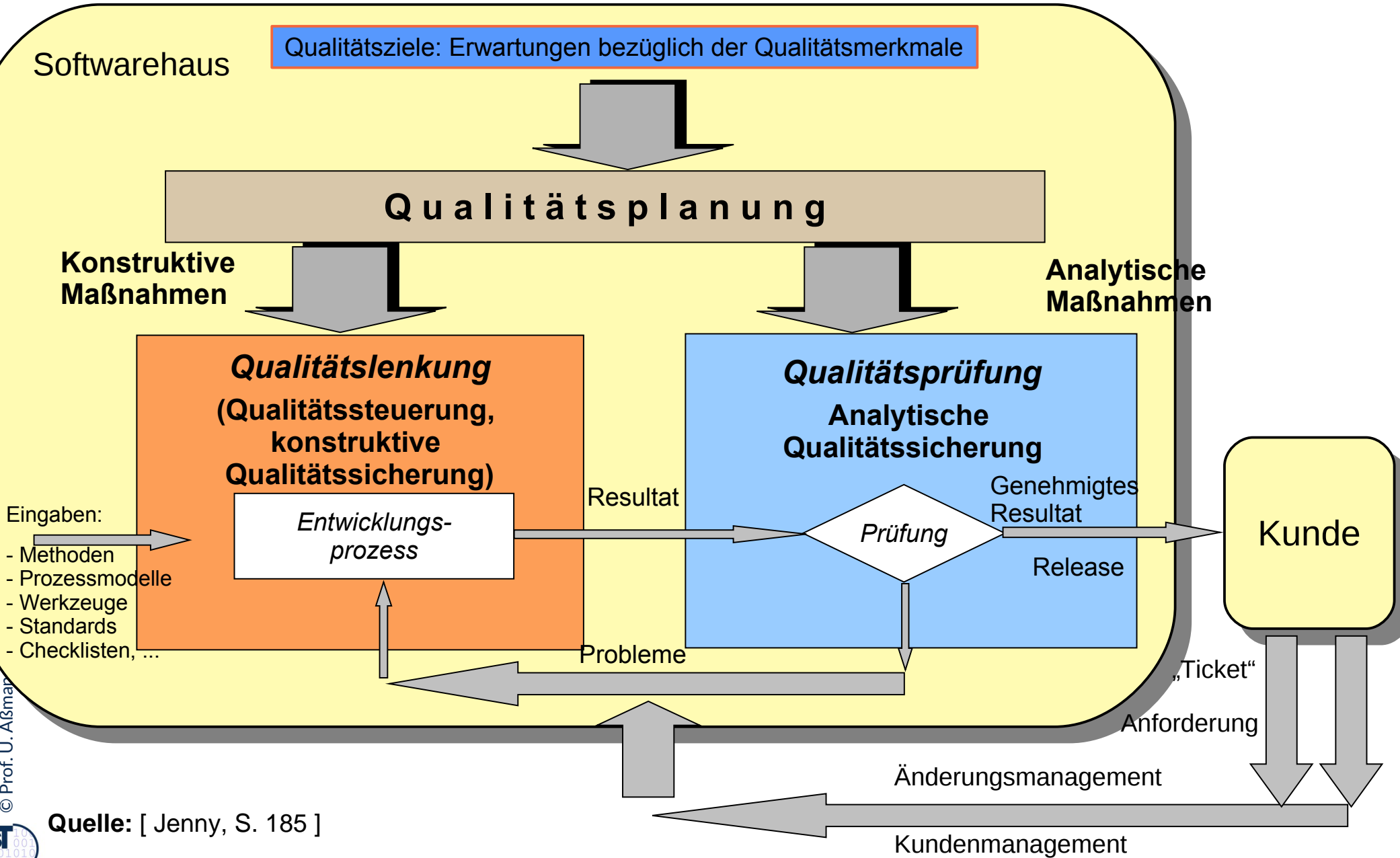
FURPS definiert Anforderungstypen, vier davon mit Qualität:

[Wikipedia] [Grady/Caswell] in Hewlett-Packard

- ▶ **Functionality** - Feature set, Capabilities, Generality, Security
- ▶ **Usability** - Human factors, Aesthetics, Consistency, Documentation
- ▶ **Reliability** - Frequency/severity of failure, Recoverability, Predictability, Accuracy, Mean time to failure
- ▶ **Performance** - Speed, Efficiency, Resource consumption, Throughput, Response time
- ▶ **Supportability** - Testability, Extensibility, Adaptability, Maintainability, Compatibility, Configurability, Serviceability, Installability, Localizability, Portability

- ▶ [IBM]
- ▶ Design, Implementation, Interface, and Physical Requirements
- ▶ The "+" in the FURPS+ acronym is used to identify additional categories that generally represent constraints.
- ▶ A **design requirement**, often called a design constraint, specifies or constrains the options for designing a system. For example, if you specify that a relational database is required, that's a design constraint.
- ▶ An **implementation requirement** specifies or constrains the coding or construction of a system. Examples might include required standards, implementation languages, and resource limits.
- ▶ An **interface requirement** specifies an external item with which a system must interact, or constraints on formats or other factors used within such an interaction.
- ▶ A physical requirement specifies a physical constraint imposed on the hardware used to house the system – shape, size, or weight, for example.

# Qualitätsmanagement-System (im Application Lifecycle Management ALM)



Das QM läuft in einem PDCA-Regelkreis, da es geplant, durchgeführt, überprüft und verbessert werden muss:

- ▶ **Qualitätsplanung (PLAN)**
  - Ist-Zustand ermitteln, Ziele und Rahmenbedingungen für das QM festlegen
  - Konzepte und Abläufe erarbeiten
- ▶ **Qualitätslenkung/-steuerung (DO)** - die in der Planphase gewonnenen Ergebnisse werden umgesetzt
- ▶ **Qualitätsprüfung (-sicherung) (CHECK)** - Auswerten qualitativer und quantitativer Qualitätsinformationen
  - Entgegennahme von Fehlern, die beim Kunden gefunden werden ("tickets")
  - Entgegennahme von neuen Anforderungen für Funktionen ("feature request")
  - Kosten-Nutzen-Betrachtungen (Wirtschaftlichkeitsbetrachtungen): Was kostet welche Fehlerkorrektur, welches neue Feature/Funktion?
  - Releaseplanung: Wann kommt eine neue Funktionalität ins Produkt?
  - Überprüfen von gemachten Annahmen
- ▶ **Qualitätsverbesserung (ACT)** -
  - Maßnahmen umsetzen zur Steigerung der Produktqualität und Prozessoptimierung.
  - Erfolge und Ergebnisse werden kommuniziert.
  - Fehlerkorrektur
  - Einführung von neuen Funktionen

[nach Wikipedia]

- ▶ **Qualitätsplanung (PLAN)**
  - Festlegung aller Anforderungen und Ziele an das System und den Projektabwicklungsprozess
  - Bestimmen, Klassifizieren und Wichten aller Qualitätsmerkmale
  - Zugrundelegung von Normen für die Qualitätsplanung
- ▶ **Qualitätslenkung (-steuerung) durch konstruktive Maßnahmen (DO)**
  - **konstruktive** Maßnahmen bis hin zum Einsatz von SE-Methoden, Werkzeugen
  - **organisatorische** Maßnahmen wie Einsatz von Vorgehensmodellen, Richtlinien, Standards, Checklisten und Dokumentationsvorschriften
- ▶ **Qualitätsprüfung durch analytische Maßnahmen (CHECK)**
  - **Prozessgestützte Qualitätsprüfung:**
    - Analyse und Auswertung des Entwicklungsprozesses nach den häufigsten und gravierendsten Qualitätsmängeln
  - **Produktgestützte Qualitätsprüfung:**
    - **statische** Prüfungen (Prüfung der Entwurfsdokumente)
    - **dynamische** Prüfungen (Ausführung des Prüfobjekts, Testen)



# Festlegung von Qualitätszielen für Projekte in der Qualitätsplanung

- ▶ Zunächst sollten Qualitätsziele festgelegt werden, separat für die Stakeholdergruppen: Nutzer, Entwickler, Management
- ▶ Qualitätszielbestimmung für das Projekt in Form von einfachen ordinalen Güteklassen.  
Beispiel:

Quelle: [BalzertSM]

Produktqualität		sehr gut	gut	normal	nicht relevant
Nutzerqualitäten	Funktionalität	x			
	Zuverlässigkeit		x		
	Benutzbarkeit	x			
	Sicherheit			x	
	Effizienz				x
Entwicklerqualitäten	Änderbarkeit		x		
	Übertragbarkeit	x			
Managerqualitäten	Return of Investment	x			
	Marktattraktivität		x		

# Qualitätsziele und abgeleitete Maßnahmen für Zuverlässigkeit (Bsp.)

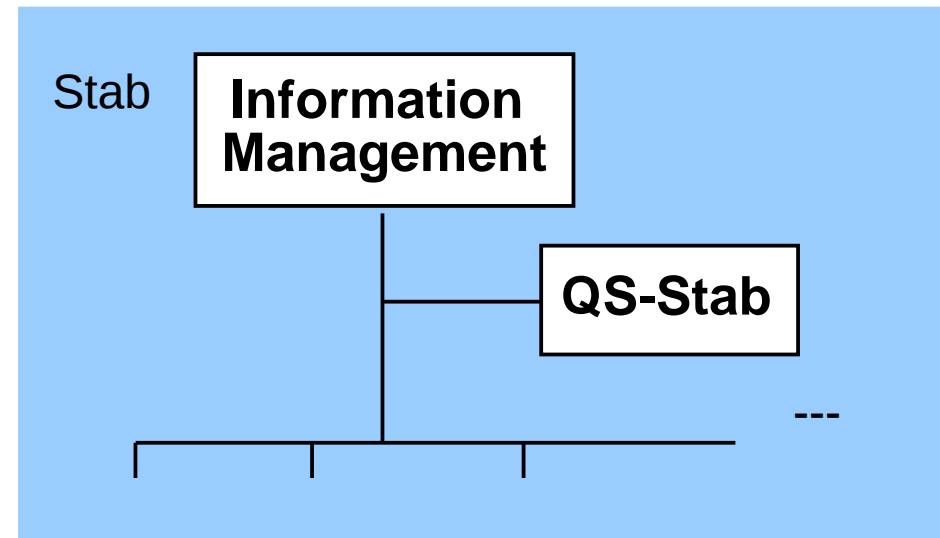
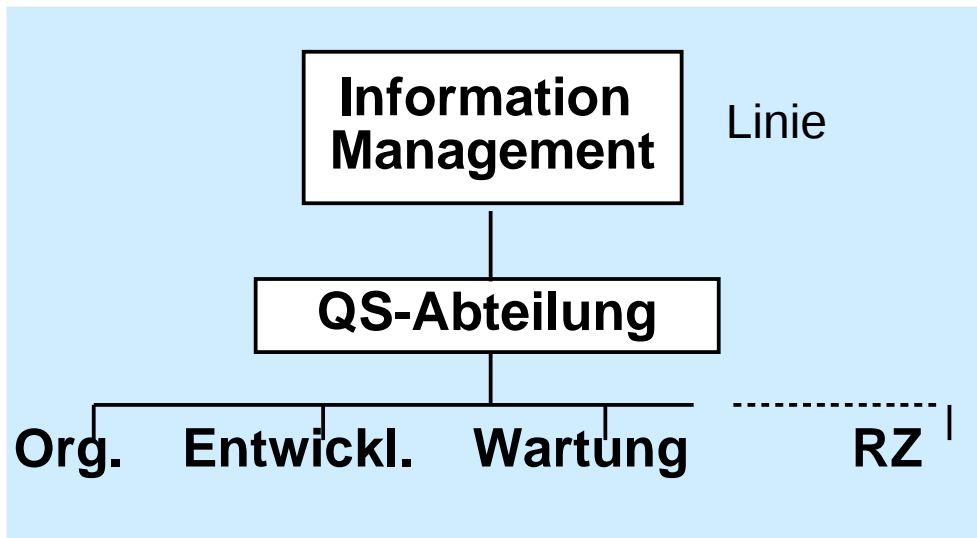
- ▶ a) Verbesserung der Fehlerverhütung
  - Ziele des Projektes festlegen (Req.-Katalog, PH)
  - Projektmanagement: Die Qualität eines Produkts entsteht aus Qualität der Phasenergebnisse ==> Summationseffekt
  - Qualifikation der Mitarbeiter
- ▶ b) Fehlerentdeckung/Beseitigung (mittels analysierender Verfahren)
  - Audits, Reviews, Code-Inspektionen, Walkthroughs
  - statische Programmanalyse
  - Verifikation
- ▶ c) Verbesserung der Systemstruktur
  - Metriken, Refactoring
- ▶ d) Verbesserung der Entwicklung (Entwicklungshilfen, Werkzeuge)
  - Integrationshilfen
  - Testfallbibliotheken
  - Fehlersuchhilfen (Debugging)

# Skalen, um Qualitätsziele zu messen

- ▶ Quantifizierbare Skalen
  - Cardinal scale (Kardinalskala, metric): ex.
  - Interval scale (Intervallskala, metric, but in pairs)
  - Ratio scale (Verhältnisskala): Interval scale with origin point (Nullpunkt).  
Verhältnisse dürfen gebildet werden.
- ▶ Absolute scale (Absolutskala): Absolute values
- ▶ Kategorielle Skalen
  - Nominale Skala (non-metric, Notenskala mit Kategorien, Güteklassen, Noten, Enumeration)
  - Ordinal scale (Ordinalskala, Rangordnung), for prioritizations

## 20.1.3 Aufbauorganisation der QS

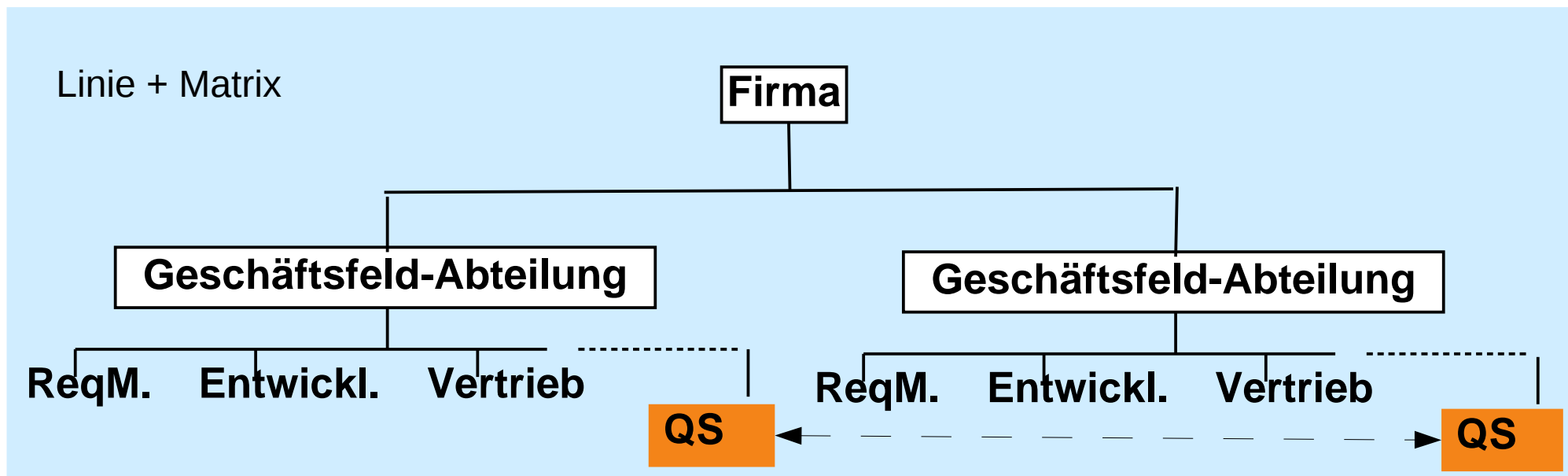
- ▶ a) **QS durch externe Unternehmen (Berater, Auditoren)**
  - Akzeptanzproblem der QS-Mitarbeiter („unproduktiv“)
  - Einsatz externer Subunternehmer (Test- und QS-Dienstleister)
  - Wichtig, wenn man intern nicht weiterkommt
- ▶ b) **QS durch eigenständige Abt. im Org./DV-Bereich**
  - QS unterhalb des Information Management (IM), entweder als Linienstelle (Abteilung)
- ▶ c) **Projektmanagement-Büro oder Stabsstelle**
  - Mit Durchgriffsrecht und Finanzbudget



# Forts.: Aufbauorganisation der QS

## ▶ d) QS durch Kompetenzteams

- als kleine Teams oder auch Rollen in einem Produkt- oder Geschäftsfeldabteilung
- Die QS-Teams aller Abteilungen bildet "Projekt" in einer Matrix

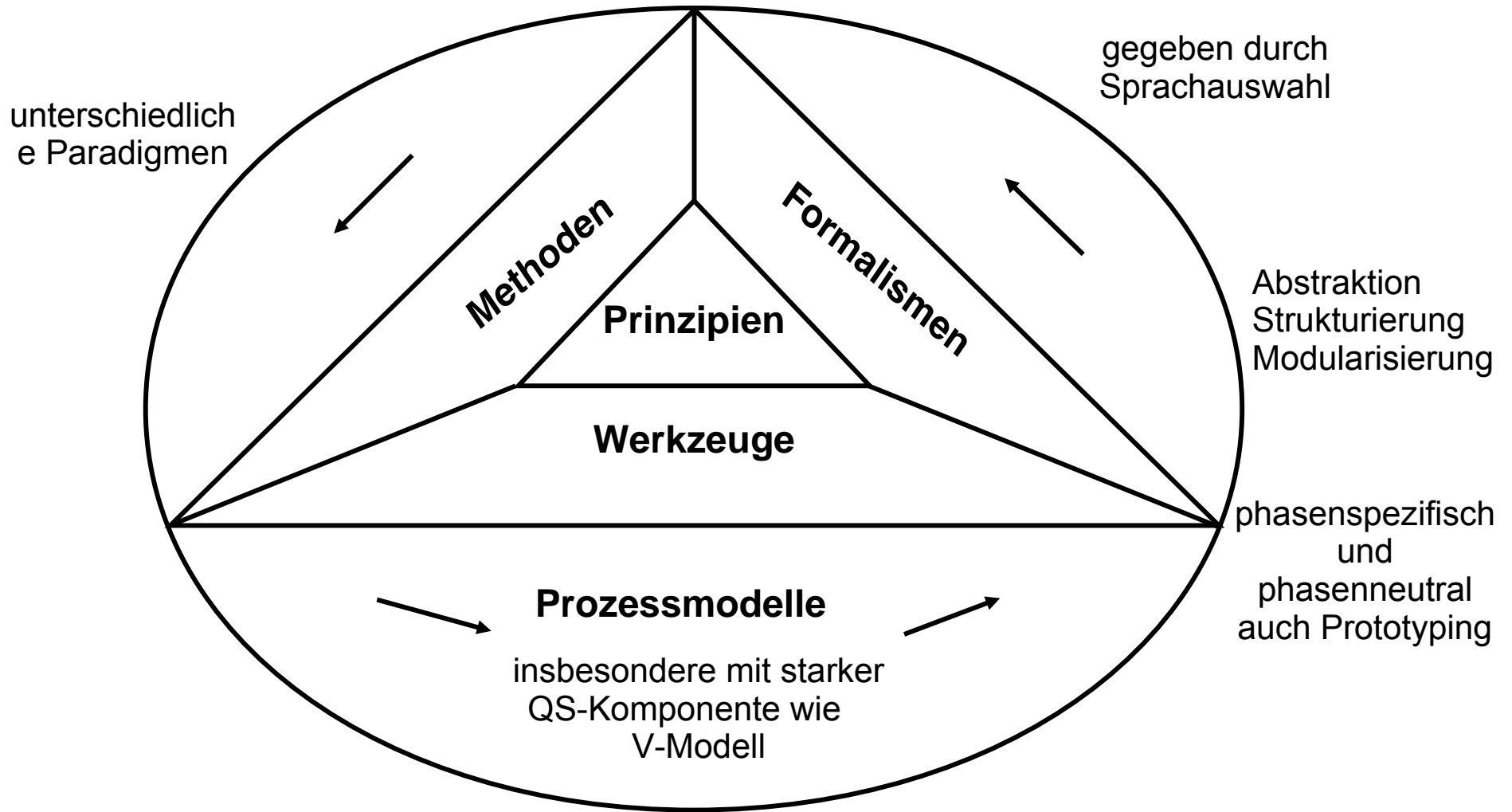


## 20.2 Konstruktive Qualitätssicherung im Prozess (Qualitätslenkung)

- ▶ Konstruktive QS verbessert die Konstruktionsprozess des Produkts durch Qualitätslenkung



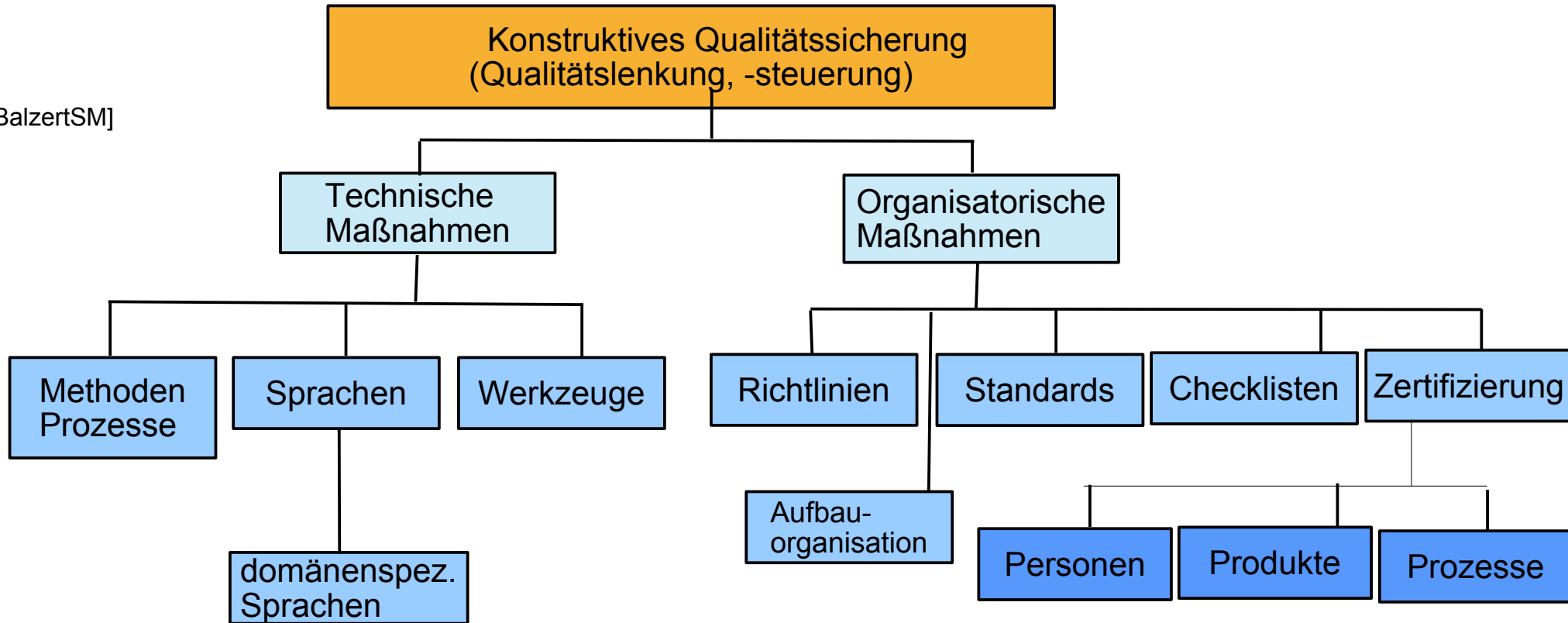
# Konstruktive Elemente der Softwaretechnik



Quelle: [Wallmüller]

# Maßnahmen zum konstruktiven QS

[BalzertSM]





# Beispiele für technische Maßnahmen zur konstruktiven QS

## ▶ Methoden:

- Einsatz einer **Schätzmethode**, wie Delphi, Function Point oder COCOMO.
- Nutzung des **Requirementmanagements**, um Anforderungsstufenkonzepte aufbauen zu können.
- Förderung der **Persönlichkeitsbildung**, wie fachliche Fortbildung oder psychologisch-orientierte Maßnahmen.
- Frühzeitige Prüfung der Entwurfs- und Implementierungsanforderungen durch den Aufbau von **Prototypen**

## ▶ Sprachen

- Einsatz von **Modellgetriebener Entwicklung**, um Verfolgbarkeit von Anforderungen zum Code zu realisieren
- Einsatz von **domänenspezifischen Sprachen**, um Fehler zu vermeiden und domänenspezifische Constraints durchzusetzen
- **Programmiersprache** mit strengem Typkonzept, um auch zur Laufzeit Typprüfungen vornehmen zu können.

## ▶ Werkzeuge

- **Ticketingsysteme**, um Fehler zu erfassen und zu korregieren, um Anforderungen zu sammeln
- **Metrikwerkzeuge**, um Architektur und Code nach Güte zu beurteilen

# Beispiele für organisatorische Maßnahmen zur konstruktiven QS

## ▶ Richtlinien:

- Definition von Entwicklungsprozessen in einer **Prozesslandkarte (process map)**
- Projektbegleitende **Dokumentationsfortschreibung** möglichst nach einem Standard und werkzeuggestützt
- Dokumentenmuster für **Pflichtenheft**, dass eine sichere Erfassung aller Anforderungen gewährleistet.
- Software-**Konfigurationsmanagement** für eine saubere Verwaltung aller bei der Entwicklung entstehender Produkte

## ▶ Zertifizierung

- der Entwicklungsprozesse durch externe Organisationen, z.B. TÜV (ISO 9000, SPICE, PRINCE)
- der Mitarbeiter (ISTQB, iSAQB)
- der Produkte
- der Dienstleistungen

## 20.2.1 Technische Maßnahmen zur konstruktiven Qualitätssicherung

- ▶ Die Konstruktion von domänenspezifischen Sprachen, sowie viele andere Werkzeuge werden in der Vorlesung “Softwarewerkzeuge” im WS behandelt
  - WS 15/16: “Model-Driven Software Development in Technical Spaces”
  - WS 16/17: “Software Factories”



## 20.2.2 Organisatorische Maßnahmen zur konstruktiven Qualitätssicherung

- ▶ Konstruktive QS verbessert die Konstruktionsprozess des Produkts durch Qualitätslenkung



# Bsp: Checkliste für Qualität von Anforderungsspezifikationen

- ▶ Sind Anforderungen vollständig und widerspruchsfrei? (CCC)
  - Wurden alle Funktionen spezifiziert?
  - Sind alle Algorithmen für Funktionen spezifiziert?
  - Wurden die Datenströme im Kontextmodell in Form von Menge pro Zeit bzw. in Form einer statistischen Verteilung spezifiziert?
  - Sind alle Hardware-Ressourcen spezifiziert?
  - Sind alle Schnittstellen beschrieben?
  - Ist der Initialzustand des Systems spezifiziert?
- ▶ Sind die spezifischen Antwortzeiten realisierbar? (SMART)
  - Wurden für Software-Qualitätsanforderungen Genauigkeitsangaben (Messbarkeitsskala, Schwellwerte) spezifiziert?
  - Gibt es zu jeder Funktion Abnahmekriterien?
  - Gibt es Gültigkeitsprüfungen für Daten?
- ▶ Sind die Anforderungen verständlich für die Entwerfer?
- ▶ Ist an spätere Erweiterungen gedacht?
- ▶ Wurde an die Ausbildung des Bedienpersonals gedacht?

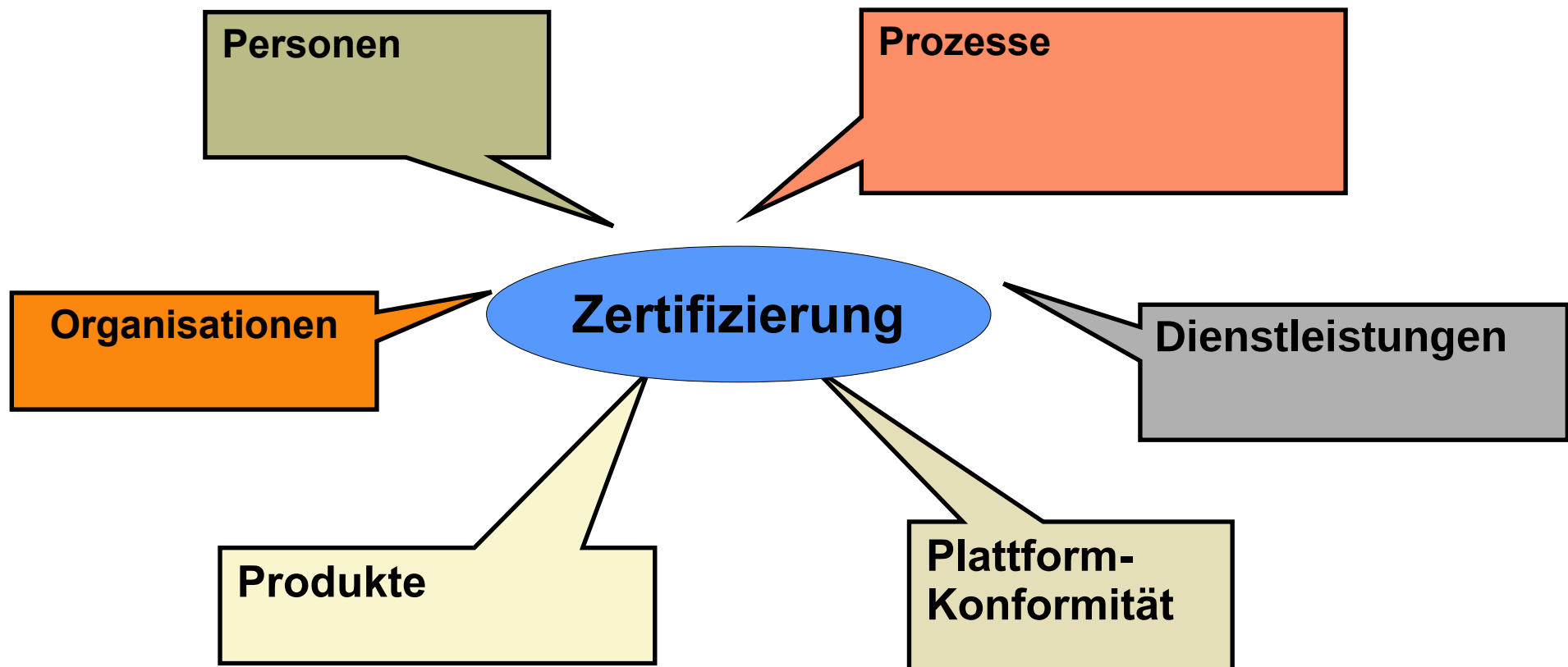
Ein **Prozesslandkarte** ist eine Sammlung von im Unternehmen vorgeschriebenen Prozessen zur Entwicklung von Software.

- ▶ Prozesslandkarten im Büro, für alle Mitarbeiter sichtbar, aufhängen
  - Als Checklisten im Wiki spezifizieren
  - Als Workflows spezifizieren (BPMN)
- ▶ Die Prozesslandkarten unterstehen dem Projektmanagement-Office (PMO)
  - In der Erfahrungsdatenbank speichern
- ▶ Erstellung einer Prozesslandkarte ist eines der einfachsten Methoden, die Prozessqualität und, indirekt, die Produktqualität zu steigern

- ▶ <http://de.wikipedia.org/wiki/Programmierstil>
- ▶ Für jede Programmiersprache finden sich de-facto-Standards für Programmierrichtlinien:
  - <http://www.gnu.org/prep/standards/> (C, C++)
  - <http://www.linuxfromscratch.org/alfs/view/hacker/part2/hacker/coding-style.html> (Linux)
  - <http://csis.pace.edu/~bergin/patterns/codingpatterns.html> (Java)
  - [http://help.sap.com/saphelp\\_46c/helpdata/de/a5/3ec9b64ac011d1894e0000e829fbbd/content.htm](http://help.sap.com/saphelp_46c/helpdata/de/a5/3ec9b64ac011d1894e0000e829fbbd/content.htm) (SAP ABAP)
- ▶ Viele Firmen haben eigene Richtlinien
- ▶ Ziel: Gute Dokumentation, einfache Lesbarkeit, Wartbarkeit, schnelle Erlernbarkeit
- ▶ [http://de.wikipedia.org/wiki/Style\\_Checker](http://de.wikipedia.org/wiki/Style_Checker) Stilprüfer können die Richtlinie automatisch überprüfen

## 20.2.2.2 Zertifizierung

Als **Zertifizierung** (von lat. „certe“ = bestimmt, gewiss, sicher und „facere“ = machen, schaffen, verfertigen) bezeichnet man ein Verfahren, mit dessen Hilfe die Einhaltung bestimmter Anforderungen nachgewiesen wird.  
[Wikipedia]





# Was kann alles zertifiziert werden?

- ▶ Zertifizierung von Prozessen
  - Zertifizierung eines Erstellungsprozesses oder Managementsystems (zum Beispiel nach ISO 9001, ISO 14001).
  - Bis Ende 2009 über 1 Mio. Zertifikate ISO 9001; über 200k Zertifikate ISO 14001
- ▶ Zertifizierung von Organisationen
  - Zertifizierung der Informationssicherheit nach BS 7799 oder ISO/IEC 27001.
  - Zertifizierung der IT-Umgebung nach IT-Grundschutz mit Grundschutz-Auditoren und das BSI
- ▶ Zertifizierung von Produkten
  - Für Zertifizierungsstellen, die Zertifizierungssysteme für Produkte oder Dienstleistungen betreiben, besteht die EN 45011 bzw. der ISO/IEC Guide 65.
  - Zertifizierung von Softwareprodukten in Hinblick auf Funktionalität und Qualität.
    - TCSEC (USA), ITSEC (EU), Common Criteria (CC).
    - In Deutschland erfolgt die Zertifizierung durch das BSI.
- ▶ Zertifizierung von Personen
  - Entwickler
  - Tester
  - Softwarearchitekten

# Was kann alles zertifiziert werden? (ctd.)

- ▶ Zertifizierung von Dienstleistungen
  - ITIL ist eine Richtlinie für IT-Dienstleistungen. Es definiert drei Zertifizierungsniveaus: Foundation, Practitioner und Service Manager.
  - ITIL Foundation
  - ITIL Service Manager
  - ITIL Practitioner Configuration Management, Incident Management, Problem Management, Release Management, Change Management, Service Level Management, Financial Management, Capacity Management, Availability Management

# 20.2.2.a Zertifizierung von Software-Produkten als Wettbewerbsvorteil

- ▶ **Grundidee:** Prüfung von **Software** auf bestimmte Eigenschaften (DIN, ISO) durch unabhängige Stellen, danach **Zertifizierung**
  - Wichtig für sicherheitskritische Software in eingebetteten Systemen
    - Auto (Drive-by-Wire)
    - Flugzeug (Fly-by-Wire: Airbus, Boeing)
    - cyber-physikalische Systeme (Life-by-Wire)
  - Überprüfung mittels Checklisten, Verifikation und Validation
- ▶ **Deutschland:** Mutter aller Akkreditierungsstellen **DAKKS**
  - Deutschen Akkreditierungsstelle GmbH (Dakks) <http://www.dakks.de/>
    - [http://www.dakks.de/doc\\_ze-geraete](http://www.dakks.de/doc_ze-geraete)
- ▶ **Europa: ECITC** („European Committee for Information Technology Certification“)

## **Aufgabe der DAKKS:**

- ▶ Neuzulassung von Zertifizierungsstellen für Produkt-, Personen- und Prozessakkreditierung
- ▶ Koordinierung der nationalen IT&T-Prüf- und Zertifizierungsaktivitäten
- ▶ Begutachtung und Akkreditierung von Prüflaboratorien und Zertifizierungsstellen
- ▶ Bestellung von Gutachtern

## **Ablauf von Prüfung und Zertifizierung:**

- ▶ Antrag auf Konformitätsprüfung eines Softwarepaketes an ein Prüflaboratorium
  - auf Wunsch wird Prüfmethode, Ort und Zeit bekanntgegeben wegen eigener Vorprüfung und Verbesserung
  - Prüfbericht ist Eigentum des Antragstellers
  - wenn überhaupt, dann Veröffentlichung in vollständiger Form
- ▶ Erteilung des Zertifikates mit Angabe der Prüfbedingungen

Ein **Software-Ökosystem** besteht aus einer Welt von Zulieferern, die auf einer Software-Plattform eines Herstellers oder Konsortiums Produkte durch den Bau von **Komplementen (plugins, add-ons)** zur Plattform erzeugen (*gemischte Wertschöpfung*)

- ▶ Beispiele:
  - Microsoft-Ökosystem
  - Java-Ökosystem
  - AutoSAR: Steuerungssoftware im Auto
  - GenIVI.com: in-vehicle infotainment im Auto
- ▶ Zertifizierung von Plugins und Produkten für Plattformkonformität:
  - Im Bereich Linux und freie Software ist ein wichtiges zertifizierendes Institut das kanadische LPI.
- ▶ [http://de.wikipedia.org/wiki/Liste\\_der\\_IT-Zertifikate](http://de.wikipedia.org/wiki/Liste_der_IT-Zertifikate)

## 20.2.2.b. Zertifizierung von Personen

- ▶ Prozess-Zertifizierung, z.B. nach ISO 9000, wird im Kap. “Prozessverbesserung” behandelt
- ▶ CERT-IT <http://www.cert-it.com/>
- ▶ Zertifizierungsstelle für die deutsche IT- und Bildungs-Branche, anerkannt von der Deutschen Akkreditierungsstelle (DAkkS)
- ▶ Gegründet von BITKOM, ZVEI, IG Metall, ver.di, Gesellschaft für Informatik, Fraunhofer Gesellschaft.
- ▶ <http://www.cert-it.com/personen-zertifizierung/certified-tester/literatur/>



- ▶ Nachweis von Ausbildungsstandards oder besonders ausgearbeiteten Fachnormen bei Personenzertifizierungen.
- ▶ Zertifizierung der Mitarbeiter zur Dokumentation von Fähigkeiten, Qualifikation und Kompetenz. Siehe dazu Liste der IT-Zertifikate.
  - PMP (Project Management Professional) durch das PMI (Project Management Institute)
  - IPMA-Zertifikate Level D-A für Projektmanager
  - ISQF Tester-Zertifizierungen: Foundation level – Advanced level
  - iSAQB-Zertifizierungen (International Software Architecture Qualifications Board)
    - Zertifizierung für Softwarearchitekten in verschiedenen Abstufungen und Spezialisierungsrichtungen
- ▶ Norm für Zertifizierungsstellen: EN ISO/IEC 17024 („Konformitätsbewertung – Allgemeine Anforderungen an Stellen, die Personen zertifizieren“)

# Zertifizierungen von Testern mit den ISQF Levels

- ▶ Der ISQF ist eine internationale Vereinigung zum Ausbilden und Zertifizieren von Testern.
- ▶ Foundation Level (CTFL)
  - In etwa das, was Sie hier im Master lernen (Vorlesungen Softwaretechnologie-II, Softwarewerkzeuge)
- ▶ Advanced Level – Test Manager
- ▶ Advanced Level – Test Analyst (vormals Functional Tester)
- ▶ Advanced Level – Technical Test Analyst (vormals Technical Tester)
- ▶ Advanced Level (CTAL) – Full Advanced Level (nach Bestehen der o. a. Teilprüfungen Advanced Level)
- ▶ Expert Level – in Vorbereitung



# Zertifizierungen von Software-Architekten mit den iSAQB Levels

- ▶ Der **International Software Architecture Qualification Board (iSAQB)** ist eine internationale Vereinigung zum Ausbilden und Zertifizieren von Softwarearchitekten
  - <http://www.isaqb.org/>
- ▶ Foundation Level (Certified Professional for Software Architecture, CPSA-F)
  - <http://www.isaqb.org/downloads/pdf/isaqb-Lehrplan-foundation.pdf>
  - In etwa das, was Sie hier im Master lernen (Vorlesungen Design-Patterns and Frameworks, Component-Based Software Engineering, Softwaretechnologie-II)
- ▶ Advanced Level (CPSA-F) [Quelle <http://www.isaqb.org/>]
  - Methodische Kompetenz: Systematisches Vorgehen bei Architekturaufgaben, unabhängig von Technologien
  - Technische Kompetenz: Kenntnis und Anwendung von Technologien zur Lösung von Entwurfsaufgaben
  - Kommunikative Kompetenz: Fähigkeiten zur produktiven Zusammenarbeit mit unterschiedlichen Stakeholdern, Kommunikation, Präsentation, Argumentation, Moderation
- ▶ Expert Level (CPSA-E): in Vorbereitung

# Personenzertifizierung für Dienstleistungen nach ITIL

- ▶ <http://www.cert-it.com/personen-zertifizierung/itilr/>

# IT Service Management (ITSM)

- ▶ ISO/IEC 20000
- ▶ <http://www.cert-it.com/it/iso-20000/>

## 20.3 Analytische Qualitätssicherung des Produkts (Qualitätsprüfung)

Qualitätsprüfung analysiert die Qualität von Produkt und Prozess und versucht, Verbesserungen vorzuschlagen

- ▶ Analyse
- ▶ Test



# Analytisches Qualitätssicherung (Qualitätsprüfung)

Vorlesung SEW (WS)

Vorlesung DPF

Vorlesung ST-2

Vorlesung SWM

## 20.3 Analytische Qualitätssicherung (Qualitätsprüfung)

### 20.3.1 Analysierende Verfahren

Werkzeug-gestützt

Statische Programanalyse

Animation Visualisierung

Programm-verifikation

Metriken

Refactoring

Prozess-gestützt

Inspektion

Walkthrough

Review

Audit

### 20.3.2 Testende Verfahren

Werkzeug-gestützt

Dynamischer Test

Symbolischer Test

Simulation

Prozess-gestützt

Schreibtischtest

Ein Analysewerkzeug nutzt eine **Softwaremetrik** zur statischen Auswertung von Komplexitätsattributen von Code und Modellen.

## Beispiele [LLT08]

- ▶ Coupling Between Object Classes (CBO): number of neighbored, i.e., coupled classes
- ▶ Depth of inheritance hierarchy (DIT)
- ▶ Breadth of inheritance hierarchy (BIT) or Number of Children (NOC)
- ▶ Number of Methods (NOM)
- ▶ Lack of Cohesion of Methods (LCOM)
- ▶ Lines Of Code (LOC) in a class
- ▶ Response for a Class (RFC)
- ▶ Weighted Methods per Class (WMC)

## 20.3.1. Prozessgestützte Analytische QS-Verfahren

- ▶ 20.3.1.1 Prozessgestützte QS-Prüfung



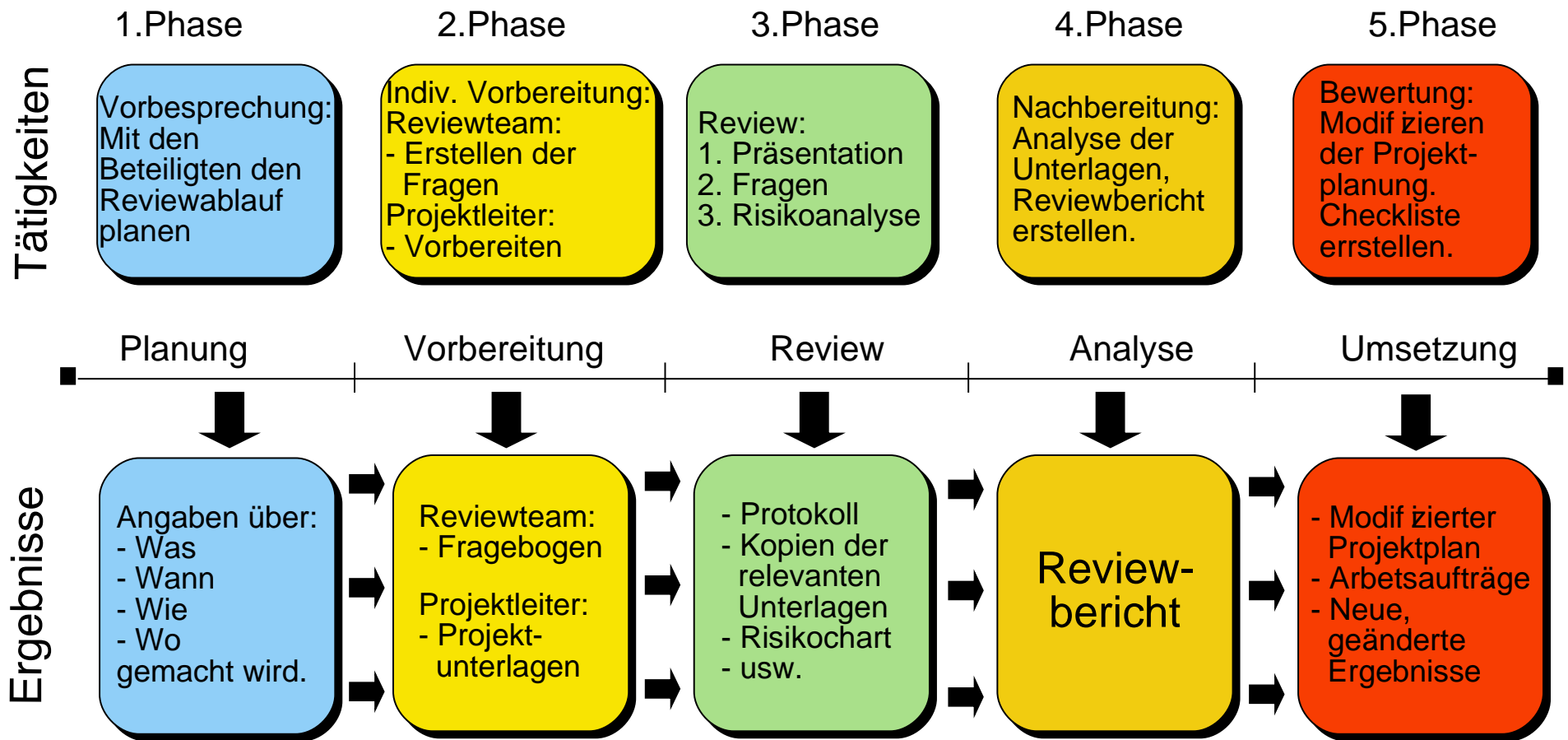
Ein **Review** ist eine manuelle Prüfmethode mit festgelegtem Ablauf, mit der ein bestehender Zustand (z.B. Projektergebnisse) oder die Wirksamkeit eingeführter Maßnahmen einem *Team von Gutachtern* vorgelegt und von diesen kommentiert oder genehmigt werden (Projektplan-Review, Anforderungs-Review, Entwurfs-Review, Code-Review u.a.)

- ▶ Reviews fokussieren sich auf Produktqualität
  - Endprodukt (PBS)
  - Zwischenprodukte (Modelle, Dokumentation, Testsuiten)

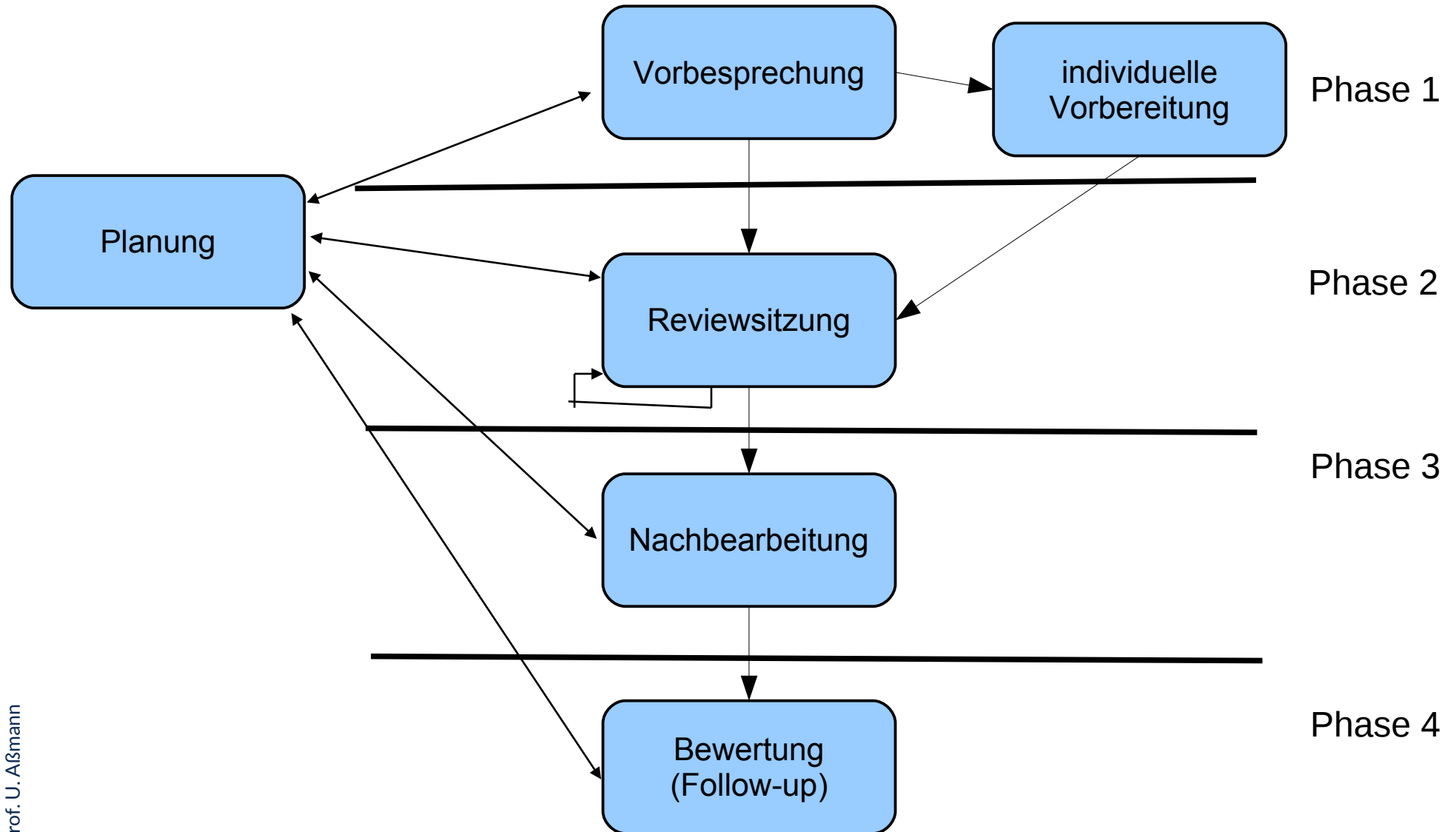


# Projektreview-Ablauf

- Der Reviewablauf besteht aus fünf Phasen:



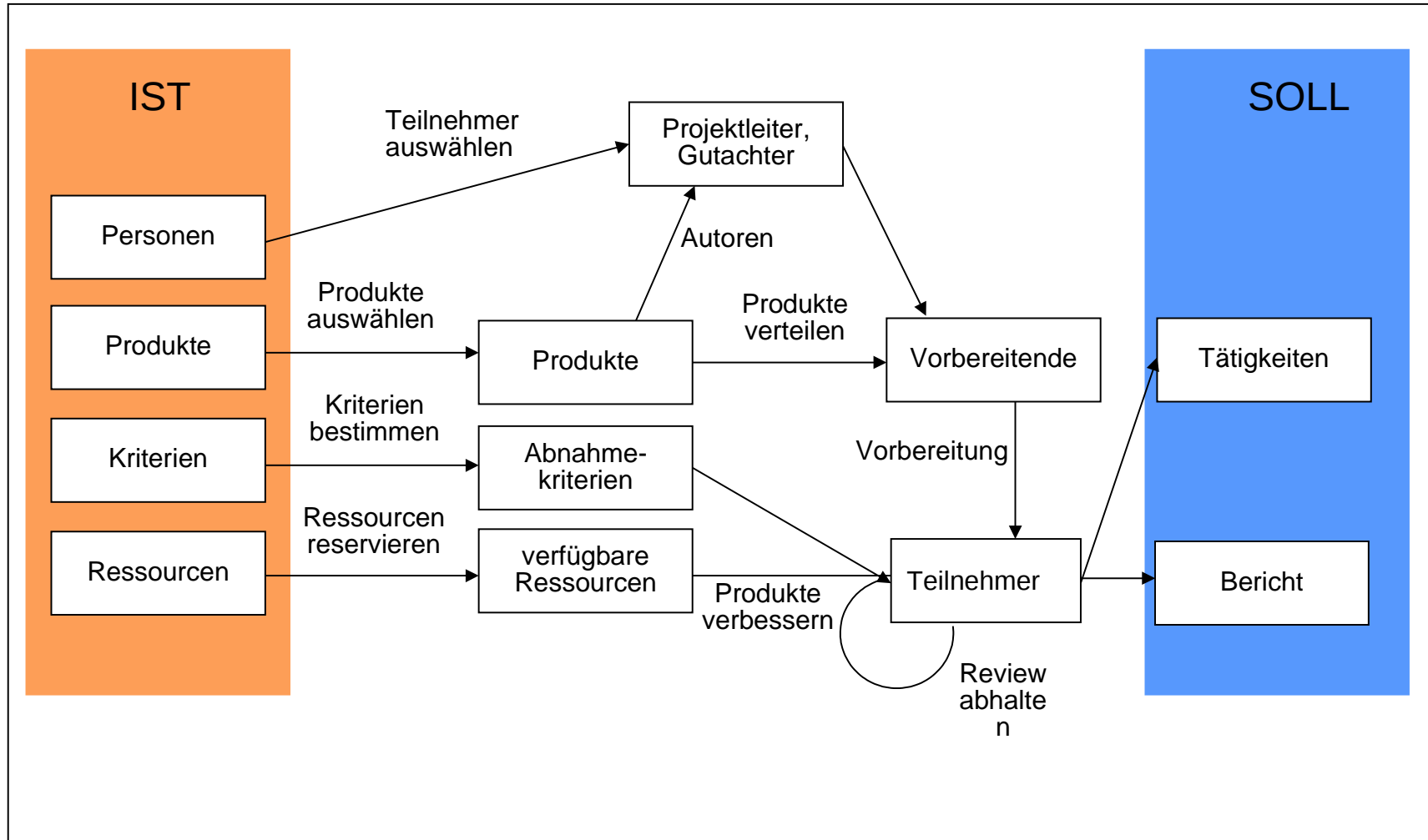
# Wdh.: Ablauf eines Projekt-Reviews als Aktivitätendiagramm (Vorgangsknotennetz)



# Wdh.: Abhängigkeitsdiagramm(-graph) eines Review als Vorgangspfeil-Netz über Daten

59

Softwaremanagement (SWM)



Quelle: [ Zuser, W. S.  
122 ]

# Arten von Reviews (1)

- ▶ **Inspektionen: (white-box review)** Die Dokumentation des Prüfgegenstandes wird von den Inspektoren *Zeile für Zeile gelesen und geprüft*
  - Teilnehmer (Moderator, Autor, Gutachter, Protokollführer)
  - Inspektionen sind in jeder Phase möglich (Bsp. Code-, Design-, Pflichtenheft-Inspektion)
  - Vorbereitung nötig (Einladung, Richtlinien, Rollen, Prüflinge)
  - Prüfung anhand von Checklisten
  - Inspektoren tragen Fehler vor
  - Inspektionsprotokoll durch Protokollführer
  - ggf. Freigabe durch Moderator
- ▶ **Walkthroughs: (black-box review)**
  - Die **Funktionalität** des Prüfgegenstandes wird anhand von vorbereiteten Beispielen und Testfällen **durchgespielt**.
  - ohne Moderator, evtl. ohne individuelle Vorbereitung, Autor stellt sein Prüfobjekt vor.
- **Tools:**
  - [http://wiki.eclipse.org/Reviews/R4E/User\\_Guide](http://wiki.eclipse.org/Reviews/R4E/User_Guide)
  - <http://www.agilereview.org/wp-content/uploads/2012/04/quickstart.pdf>

# Arten von Reviews (2)

## ▶ Round-Robin-Review:

- Die Gutachter sollen in der Vorbereitung nach Argumenten suchen, warum die Qualität des Prüflings hoch ist.
- In der Sitzung trägt jeder sein Plädoyer vor, die anderen Gutachter intervenieren
- Argumente für und gegen den Prüfling werden notiert.

## ▶ Peer Review: („Späher“)

- Gutachter werden „eingeschlossen“, untersuchen die Prüflinge und erstellen Gutachten.
- Ein Moderator leitet das Team.
- Das Team wird entweder ad hoc zusammengestellt oder existiert als permanente Einrichtung („professionelle Peers“)

# Bsp.: Checkliste für Grobentwurfs-Reviews

## Performance

⇒ Gibt es Hinweise auf die Nichterfüllung von Performance-Anforderungen?

## Benutzungsschnittstelle

⇒ Sind die Layouts der Benutzungsschnittstelle einheitlich?

⇒ Sind die Bildschirmmasken mit Informationen nicht überladen?

⇒ Sind die Bildschirmausgaben übersichtlich?

⇒ Ist die Benutzerführung ausreichend?

⇒ Sind die Benutzereingaben auf ein Minimum beschränkt?

## Daten

⇒ Wurde das Datenmodell geprüft?

⇒ Gibt es fehlende oder nicht benutzte Variablen in einem I.-, O.- oder Update-Modul?

⇒ Gibt es falsche oder fehlende Datentypen in einem Input-, Output- oder U.-Modul?

## Funktionalität

⇒ Ist in einem Verarbeitungsmodul ein Teil nicht vorhanden, überflüssig oder falsch?

⇒ Sind in einem V.-modul logische Bed. nicht vorhanden, überflüssig oder falsch?

## Außerdem:

**Schnittstellen, Dokumentation, Standards, Syntax der Entwurfsbeschr., . . .**

# Audits (1) - Formale Reviews von Außen

Ein **Audit** ist eine *systematische* und *unabhängige* Untersuchung, bei der sowohl die Übereinstimmung mit Spezifikationen, Standards, vertraglichen Vereinbarungen oder anderer Kriterien (Angemessenheit, Einhaltung vorgegebener Vorgehensweisen und Anweisungen), als auch deren Wirksamkeit und Sinnhaftigkeit überprüft werden.

- ▶ **Audit der Produktqualität:** quantitative Bewertung der Konformität des Produktes mit den geforderten Produktmerkmalen lt. Pflichtenheft
  - ▶ **Audit der Prozessqualität:** Überprüfung der Elemente eines Prozesses auf Vollständigkeit und Wirksamkeit z. B. im Vergleich zu einem Vorgehens- oder Prozessmodell
  - ▶ **Audit des QS-Systems:** Prüfung, ob vorhandene Elemente des QS-Systems entsprechend den Anforderungen vollständig, dokumentiert und wirksam sind.
  - ▶ Audit des Finanzmanagements
  - ▶ Audit des Entwicklungs- und Managementprozesses:
    - z. B.: - Produktivität des Projektteams , Einhaltung vorgegebener Standards
- ⇒ Während eines größeren Projekts sollten mehrere Audits durchgeführt werden

# Audits (2)

- ▶ Systematische und unabhängige Untersuchung mit formalem Charakter
- ▶ Validation der Systeme, Prozesse, Produkte mit den Vorgaben (Spezifikationen) *durch Dritte, meist spezialisierte Audit-Firmen*
- ▶ Audits werden durch ausgebildete Auditoren nach einem definierten Ablauf durchgeführt:
  - Vorbereitung: Die Auditoren fordern von extern ausgewählte Untersuchungs- und/ oder Prüfdokumente an (z.B. Projektplan, Vorgehensmodell, zugrunde liegende Vorgaben, Metriken u.a.).
  - Durchführung: erfolgt in der Firma durch Interviews mit Prozessverantwortlichem und Dokumentensichtung
  - Abschluss: zum Abschluss des Audits erfolgt ein vorläufiges Feedback an alle Beteiligten
  - Ergebnisse werden in einem ausführlichen Audit-Bericht dokumentiert
- ▶ Audits müssen oft durchgeführt werden, um Berichtspflichten nach außen zu erfüllen
  - SOX Sarbanes-Oxley-Act
  - Basel I-III Kriterien
  - Prüfung auf Gemeinnützigkeit



- ▶ Schritte für den Ablauf eines Audits sind:
  - Ziele der Überprüfung definieren
  - Umfang und Anwendungsbereich definieren
  - Initiierung
  - Überblick gewinnen und Daten sammeln
  - Analyse und Auswertung der gesammelten Daten
  - Lösungs- und Verbesserungsvorschlag erarbeiten
  - Erstellen und Präsentieren des Ergebnisberichts
  
- ▶ Der angetroffene Ist-Zustand wird mit den Zielen und dem Erreichungsgrad verglichen.
  
- ▶ Im Ergebnisbericht sind alle Maßnahmen aufzuführen, die zur Einhaltung des angestrebten Projektverlaufs eingeleitet werden müssen.

**Quelle:** ANSI-Norm N45.2.10-1973



## 20.3.2 Werkzeuggestützte Analytische QS-Verfahren

- ▶ zur Qualitätsprüfung eines Produktes



- ▶ mit der Hilfe von Werkzeugen ==> Vorlesung “Software Factories”
- ▶ **Lexikalische Analyse** ermittelt lexikalische Informationen
  - z. B. Länge und Häufigkeit von Programmelementen, unerreichbarer Code, falsche bzw. nicht referenzierte Sprungmarken
- ▶ **Syntaktische Analyse** und **Metriken** ermitteln syntaktische Informationen
  - z. B. Komplexitätsgrade, Aufrufgraphen, Strukturbäume, Architekturprinzipien, Endlosschleifen, Aufrufe nicht existierender Prozeduren, unerlaubte Verschachtelung von Schleifen und Verzweigungen
  - **Layout-Prüfung und Verbesserung:** Pretty-printing
- ▶ **Statische semantische Analyse** ermittelt **semantische Informationen**
  - **Abstrakte Interpretation** interpretiert das Programm statisch mit abstrakten Werten, die Fehlerwerte entdecken lassen
  - Typprüfungen (wie Typkonflikte, falsche Parameterübergaben)
  - Steuerflussanomalien wie Sicherheitsprüfungen (z.B. Buffer overflow analysis, driver protocol analysis)
  - Datenflussunverträglichkeiten wie deklarierte aber nicht verwendete Variable, nicht initialisierte Variable, falsche Verwendung globaler und lokaler Variablen

# Statische Programmanalyse (Forts.)

- ▶ **Statische Vertragsprüfung** mit Werkzeugen
  - Theorembeweiser, gute Übersetzer für Programmiersprachen mit Verträgen wie Eiffel
  - Prüfung von Qualitätsverträgen z.B. mit Qualitung ([www.qualitune.org](http://www.qualitune.org))
- ▶ **Model checking** prüft die Gültigkeit von Prädikaten in einem Zustandssystem

## 20.3.3. Testende QS-Verfahren

- ▶ (Wdh aus Softwaretechnologie-II, zum Selbststudium)



# Formelle und informelle Verifikation

- ▶ Verifikation und Validation (V & V) (vgl. ISO 9000:2005)
- ▶ **Verifikation:** „Bestätigung durch einen objektiven Nachweis, dass die Anforderungen für eine bestimmte Anwendung oder einen bestimmten Gebrauch erfüllt sind.“
  - formale Verifikation
  - systematische, informelle Verifikation durch Testen (ist leider unvollständig)
- ▶ **Validation:** „Bestätigung, dass ein Produkt ein vom Kunden erstelltes Lastenheft und damit die Anforderungen an den Gebrauch durch den Kunden erfüllt“.
  - Tests werden auch zur Validation eingesetzt
- ▶ „**Verifikation** heißt, die Arbeit richtig durchzuführen, und **Validation** heißt, die richtige Arbeit durchzuführen“ [B. Boehm]

## ► Dynamische Prüfungen:

- Datenbezogenes Testen mit Testdaten: Datenstrukturen, Referenz- oder Betriebsdaten (bei großen Programmen lassen sich kaum alle Datenkombinationen erproben)
- Funktionsbezogenes Testen: abschnittsweiser Vergleich des Codes incl. E/A-Verhalten mit der Spezifikation
- Ablaufbezogenes Testen: werden alle Schleifen, Verzweigungen durchlaufen?
  - (Kontrollflussorientierter Test - „Durchspielen“ aller Fälle; ==> aufwendig)
  - vergessene Funktionen werden nicht gefunden !!

**Quelle:** [nach Zehnder, C.,A.. Informatik-Projektentwicklung; Teubner Verlag 1991]

# SW-Testmethoden – Datenbezogener Test

- ▶ **Regressionstest:** Vergleich zweier Versionen des gleichen Programms

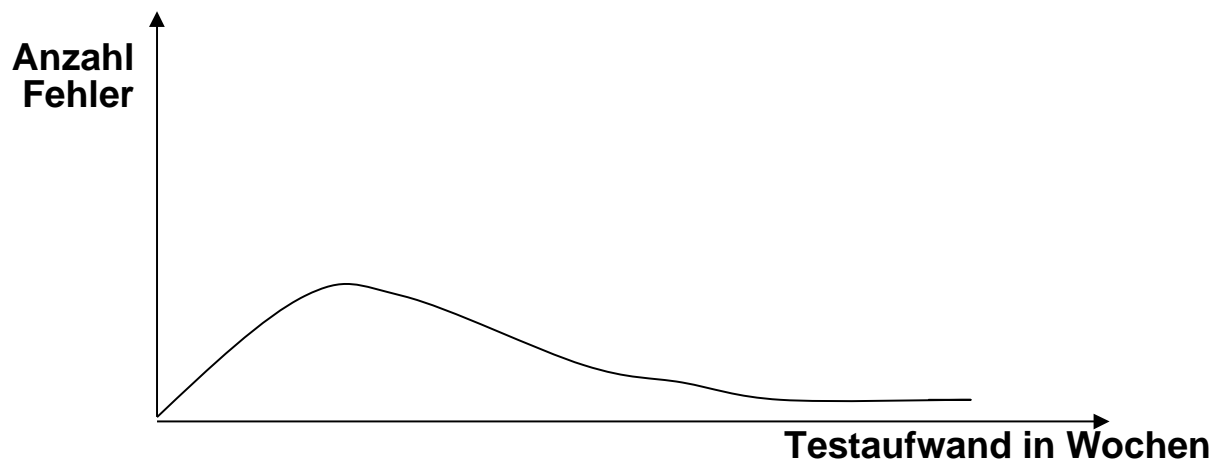
Siehe Vorlesung Softwaretechnologie-II

- ▶ Test-Endekriterien

a) aus Erfahrung: in 5% aller Module stecken 95% der Fehler

==> Stichproben; Ende, wenn 5% fehlerhafte Moduln gefunden

b) nach Fehlerrate:





# SW-Testmethoden (Black Box-Test)

Ziel: Feststellung von Abweichungen gegenüber Anforderungen bzw. Spezifikation (innere Struktur ist nicht von Interesse)

- ▶ Ermittlung der Testfälle für SW-Module, -Komponenten, Pakete, Subsysteme

Methoden:

- ▶ **Äquivalenzklassenbildung**

- Einteilung der E/A-Daten in Äquivalenzklassen (gültige und ungültige)

- ▶ **Grenzwertanalyse**

- Testfälle an den Grenzen der Wertebereiche

- ▶ **Intuitive Testfallermittlung** (unsystematisch)

- zusätzliche Testfälle durch Intuition (Liste möglicher Fehler aus Erfahrung, Standardfehler)

- ▶ **Funktionsabdeckung**

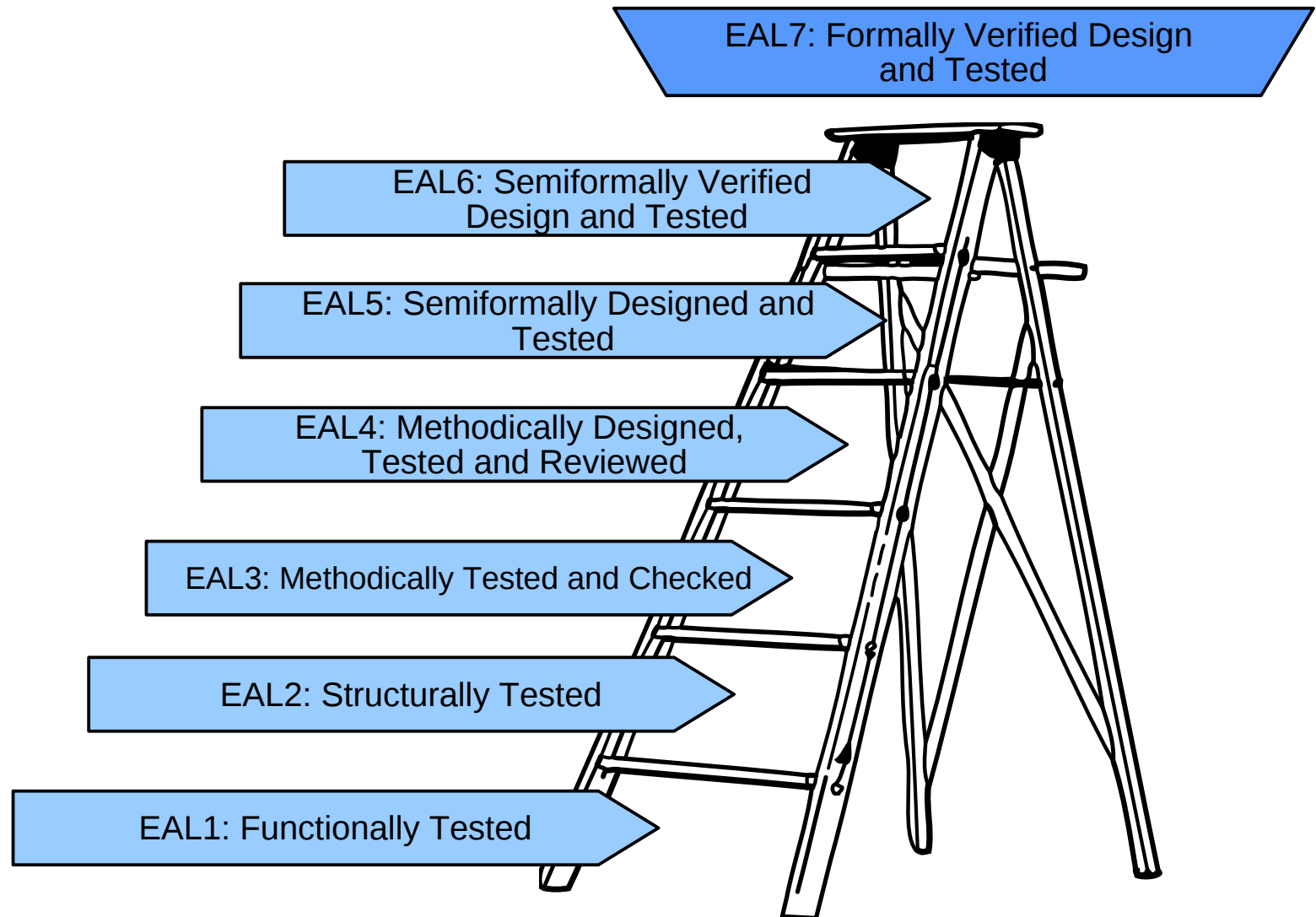
- Testfälle für Normal- und Ausnahmeverhalten
- Vermeidung von Redundanz durch Testfallmatrix

# Überdeckungs-Testmethoden (White Box-Test)

- ▶ Ziel: Entdeckung von Fehlern durch ablauforientierte Testfälle
  - interne Struktur / Quelltext muss bekannt sein
  - Es wird eine Teilmenge aller möglichen Pfade durch Testfälle abgedeckt (Pfadabdeckung)
- ▶ Methoden:
  - **Pfadabdeckung** (wenigstens eine Mindestzahl von Pfaden prüfen)
  - **Anweisungsabdeckung** (entsprechend Spezifikation, alle oder Auswahl)
  - **Bedingungsabdeckung**: Abdeckung aller If- und Case-Bedingungen
  - **Zweig-/Bedingungsabdeckung**: Abdeckung aller einzelnen alternativen Pfade
  - Abdeckung aller **Kombinationen** von verschachtelten Mehrfachbedingungen
  - **n-Schleifenabdeckung**: Abdeckung der ersten n Schleifendurchläufe von Schleifen
    - (nur n=1 oder n=2 praktikabel)

# Evaluation Assurance Levels (EAL)

- ▶ From Common Criteria Standard



# QS: Anforderungsdefinition und Abnahmekriterien

- ▶ Abnahmekriterien bereits während der Anforderungsdefinition aus den Qualitätszielen und den Anforderungen ableiten (im Pflichtenheft)
  - Aufdeckung von Lücken, Überschneidungen, Widersprüchen
    - hat oft die Überarbeitung von Anforderungen zur Folge
  - Grundlage für den Nachweis des Erfüllungsgrades
  - Spezifikation eines oder mehrerer Abnahmekriterien zu einer Anforderung
    - Den Anwender interessieren vorrangig ergebnisorientierte (Black-Box-) Abnahmekriterien
  - EAL-Level festlegen
- ▶ Funktionsabdeckung mit Testfällen
  - Äquivalenzklassenbildung
  - Grenzwertanalyse
  - Intuitive Testfallermittlung (Ergänzung der o. g. aus Erfahrung)
- ▶ ablaforientierte (White-Box-) Abnahmekriterien: welches Überdeckungskriterium?

# 20.5 Aufgabenmanagement im Qualitätsmanagement (Issue Management)



- ▶ **Aufgabenmanagement** beinhaltet die Erfassung, Registratur, Speicherung und Verfolgung von
  - Stand der Bearbeitung und Arbeitspaketen und Aktivitäten eines Projektplans
  - Qualitätssicherungsaufgaben
  - Änderungsmanagement (sieht der Kunde):
    - Problem- und Fehlermeldungen (“tickets”) in der korrektiven Wartung
    - Änderungsanforderungen (“change requests”) in der Pflege
    - Anforderungen für neue Funktionalität (“feature requests”)
    - beinhaltet also das kontinuierliche Requirements-Management
- ▶ und nutzt dazu
  - verteilte Eingaben über das Web
  - die Speicherung aller “tickets” und “requests” in einer zentralen Repository/Datenbank

Das Aufgabenmanagement ist heute von zentraler Bedeutung für die Steuerung eines Softwarehauses, denn es hilft, die Qualität der Produkte und Dienstleistungen zu verbessern und ein Stammkundengeschäft aufzubauen

# Die Bedeutung des Aufgabenmanagements

- ▶ Das Aufgabenmanagement dient als zentrales Element des Qualitätsmanagements und steuert alle dazu notwendigen Vorgänge:
  - **Bewertung der Änderung:** Nach der Notwendigkeit der Auswirkungen, wobei jede Änderung einer Version zu einer neuen Version führt
  - **Planung und Entscheidung des Änderungsvorgehens:** Vorgehen nach einer definierten Änderungsprozedur mit Durchlauf geforderter Zustände (V-Modell).
  - **Einleitung und Überwachung der Änderungsdurchführung:** Darstellung des Änderungsgeschehens in einem Repository. Daraus sollen Änderungsstatistiken auf Abruf generierbar sein.
  - **Abschluß und Auswertung der Änderung:**
    - Alle Änderungen sollen nachvollziehbar und rekonstruierbar sein.
    - Es ist eine Historie zu führen, die alle Änderungsdaten einschließlich eines ausführlichen Kommentars enthält.
  - **Release-Planung und Management:** Aufgabenmanagement erlaubt Entscheidungen, wann neue Funktionalität ins Produkt einzieht

# Aufgabenmanagement und Strategisches Management

- ▶ Das Aufgabenmanagement dient ebenfalls als strategisches Instrument:
  - **Analysen auf dem Repository:** Statistiken über Häufigkeit von Tickets
  - **Schlussfolgerungen über die Effizienz** des eigenen Entwickelns und der eigenen Qualitätssicherung
  - **Ständige Verbesserung des QM**
- ▶ Als Vorgehensbaustein „Problem- und Änderungsmanagement“ im V-Modell XT realisiert.



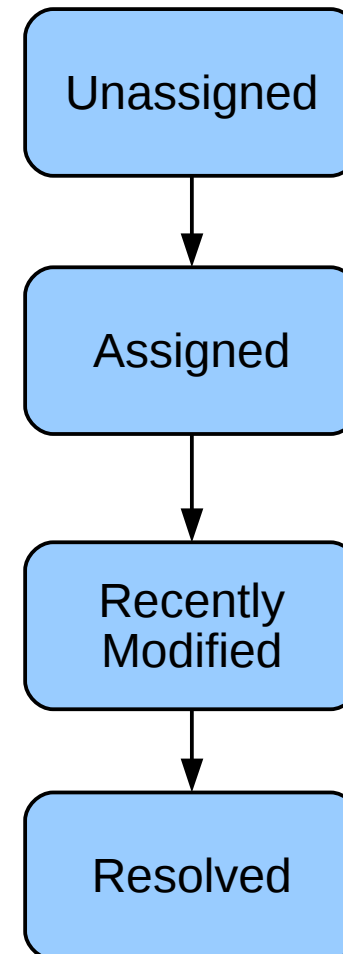
# Aufgabenmanagement-Werkzeuge

- ▶ **Issue Management:** betont, dass jedes Ticket für das Softwarehaus eine Aufgabe darstellt (“issue”)
- ▶ **Ticket Management:** betont, dass der Kunde “tickets” schreibt

Werkzeug	Status	Webadresse
Bugzilla	Mozilla (OSS)	<a href="http://www.bugzilla.org">www.bugzilla.org</a>
Mantis	OSS	<a href="http://www.mantisbt.org/">http://www.mantisbt.org/</a>
JIRA	Atlassian	<a href="http://atlassian.com/software/jira">http://atlassian.com/software/jira</a>
codeBeamer	IntLand Software	<a href="http://intland.com/products/codebeamer/overview/">http://intland.com/products/codebeamer/overview/</a>
RedMine	OSS	<a href="http://en.wikipedia.org/wiki/Redmine">http://en.wikipedia.org/wiki/Redmine</a>
Team Foundation Server	Microsoft	<a href="http://en.wikipedia.org/wiki/Team_Foundation_Server">http://en.wikipedia.org/wiki/Team_Foundation_Server</a>

# Aufgabenmanagement (Fehler und Änderungen) mit Mantis

- ▶ Mantis ist ein webbasiertes Aufgabenmanagement-System (issue tracking system)
  - Zustandsmaschine für Fehler und Änderungswünsche
  - Generierung von emails über Statusänderungen
  - Visualisierung von Zuständen und Bearbeitern



# Überblick über Aufgaben [www.mantisbt.org]

View Issues - MantisBT DEMO site

Anonymous | [Login](#) | [Signup for a new account](#) 2009-01-28 14:23 EST Project: Demo [Switch] [RES]

[Main](#) | [My View](#) | [View Issues](#) | [Change Log](#) | [Roadmap](#) | [Docs](#) | [Wiki](#) | [Billing](#) Issue #

Search:   [ [Advanced Filters](#) ] [ [Create Permalink](#) ] [Reset Filter]

Viewing Issues (1 - 50 / 1520) [ [Print Reports](#) ] [ [CSV Export](#) ] [ First Prev 1 2 3 4 5 6 7 8 9 10 11 ... Next Last ]

P	ID	US\$	#	Category	Severity	Status	Updated	Summary
	<a href="#">0005069</a>		3	GUI	minor	resolved (MacGyver)	2009-01-28	Invalid Password
	<a href="#">0005095</a>			Other	minor	assigned (mow)	2009-01-28	error prueba
	<a href="#">0005091</a>			Other	trivial	confirmed (ossgevalt)	2009-01-28	foo bar baz
	<a href="#">0004707</a>			Other	minor	resolved (000willberty)	2009-01-28	relationships I
	<a href="#">0005093</a>			GUI	minor	assigned (enzyme)	2009-01-28	Test fichier attache
	<a href="#">0005082</a>			Other	minor	assigned (121212)	2009-01-28	test
	<a href="#">0005094</a>			GUI	minor	new	2009-01-28	Test fichier attache
	<a href="#">0005092</a>			GUI	minor	new	2009-01-28	Test fichier attache
	<a href="#">0005090</a>			GUI	minor	assigned (flandersen)	2009-01-28	Needs attention
	<a href="#">0005089</a>		3	Website	major	resolved (Russell)	2009-01-28	Problems loading JavaScript on Main.html
	<a href="#">0005076</a>		1	GUI	feature	assigned (darksaboteur)	2009-01-28	I can't poo...
	<a href="#">0005088</a>			GUI	tweak	assigned (abarbosa)	2009-01-27	sadasdasda
	<a href="#">0005087</a>			GUI	trivial	assigned (ramyap)	2009-01-27	asdasddsad
	<a href="#">0005086</a>		9	GUI	trivial	assigned (Element)	2009-01-27	Detailbereich wird nicht angezeigt
	<a href="#">0005083</a>		1	Other	minor	acknowledged (aaloc)	2009-01-27	test
	<a href="#">0005085</a>			GUI	major	assigned (celso)	2009-01-27	teste com erro e associação com arquivo de repositório
	<a href="#">0005081</a>			GUI	minor	resolved (deepak84)	2009-01-27	Sql Error
	<a href="#">0005080</a>			GUI	feature	assigned (mmiot)	2009-01-27	????
	<a href="#">0005079</a>			GUI	minor	assigned (kedar)	2009-01-26	test
	<a href="#">0005078</a>		4	GUI	text	assigned (jodji)	2009-01-26	test
	<a href="#">0005077</a>			GUI	minor	new	2009-01-26	cant abc
	<a href="#">0005065</a>		3	GUI	tweak	new	2009-01-26	hhhh
	<a href="#">0005074</a>		2	Other	minor	assigned (maxadmin)	2009-01-26	VIZ funktioniert nicht
	<a href="#">0005075</a>			Website	major	assigned (patriciagomes)	2009-01-26	Erro no site
	<a href="#">0005071</a>		2	Website	major	assigned (deepak84)	2009-01-26	Website Issue
	<a href="#">0005072</a>			GUI	minor	assigned (trm)	2009-01-26	Alles Schiefe

# Überblick über Zustände der Aufgaben

The screenshot displays the MantisBT web interface. At the top, there is a navigation bar with the Mantis logo and the text "MANTIS". Below the logo, there are links for "Anonymous", "Login", and "Signup for a new account". The date and time "2009-01-28 14:21 EST" are shown. On the right, there is a "Project:" dropdown menu set to "Demo" and a "Switch" button. Below the navigation bar, there are links for "Main", "My View", "View Issues", "Change Log", "Roadmap", "Docs", "Wiki", and "Billing".

The main content area is divided into three sections:

- Unassigned [^] (1 - 10 / 271)**: A list of 10 unassigned tasks, each with a unique ID, description, and category. The tasks are: 0005094 (Test fichier attache, GUI, 2009-01-28 09:01), 0005092 (Test fichier attache, GUI, 2009-01-28 09:00), 0005077 (cant abc, GUI, 2009-01-26 10:38), 0005065 (hhhh, GUI, 2009-01-26 10:07), 0004953 (dede, GUI, 2009-01-16 00:04), 0004921 (test da Bologna, GUI, 2009-01-16 00:03), 0004851 (dede, GUI, 2009-01-16 00:03), 0004845 (Enhance CP/M to support 64 bit processors, Other, 2009-01-16 00:03), 0004835 (sddd, GUI, 2009-01-16 00:03), and 0004832 (????, GUI, 2009-01-16 00:03).
- Resolved [^] (1 - 10 / 154)**: A list of 10 resolved tasks, each with a unique ID, description, and category. The tasks are: 0004707 (relationships I, Other, 2009-01-28 13:01), 0005089 (Problems loading JavaScript on Main.html, Website, 2009-01-28 04:27), 0005081 (Sql Error, GUI, 2009-01-27 09:53), 0004728 (test test, GUI, 2009-01-22 17:32), 0004934 (Hello Puur, Website, 2009-01-16 00:04), 0004915 (Hello Puur, Website, 2009-01-16 00:03), 0004839 (Update error, Other, 2009-01-16 00:03), 0004834 (sum, GUI, 2009-01-16 00:03), 0004556 (Testing, checking, Knowing.., Other, 2009-01-16 00:03), and 0004766 (urgent FAIL, GUI, 2008-12-23 10:10).
- Recently Modified [^] (1 - 10 / 2234)**: A list of 10 recently modified tasks, each with a unique ID, description, and category. The tasks are: 0005069 (Invalid Password, GUI, 2009-01-28 14:20), 0005095 (errorr prueba, Other, 2009-01-28 13:14), 0005091 (foo bar baz, Other, 2009-01-28 13:04), 0004707 (relationships I, Other, 2009-01-28 13:01), 0005096 (errorr prueba, Website, 2009-01-28 12:57), 0005093 (Test fichier attache, GUI, 2009-01-28 10:22), 0005082 (test, Other, 2009-01-28 09:43), 0005094 (Test fichier attache, GUI, 2009-01-28 09:01), 0005092 (Test fichier attache, GUI, 2009-01-28 09:00), and 0005090 (Needs attention).



# OTRS

- ▶ <http://www.otrs.com/software/?lang=de>

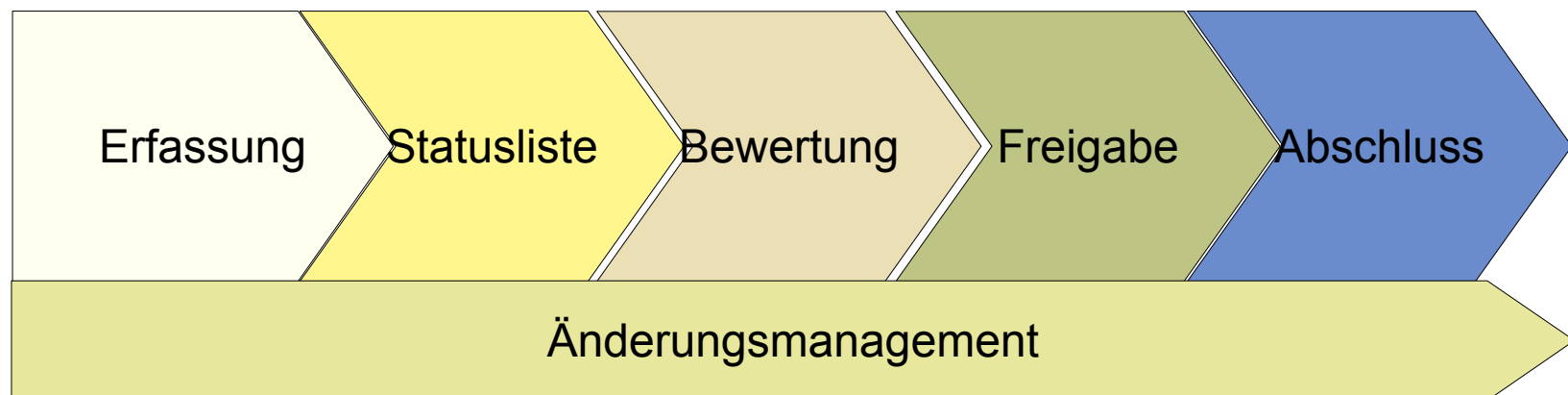
## 20.5.2 Änderungsmanagement

- ▶ Änderungsmanagement ist der Teil vom Aufgabenmanagement, den der Kunde sieht
- ▶ Also ein spezielles Aufgabenmanagement in der Anforderungsmanagement, Wartung, Pflege und Releasemanagement

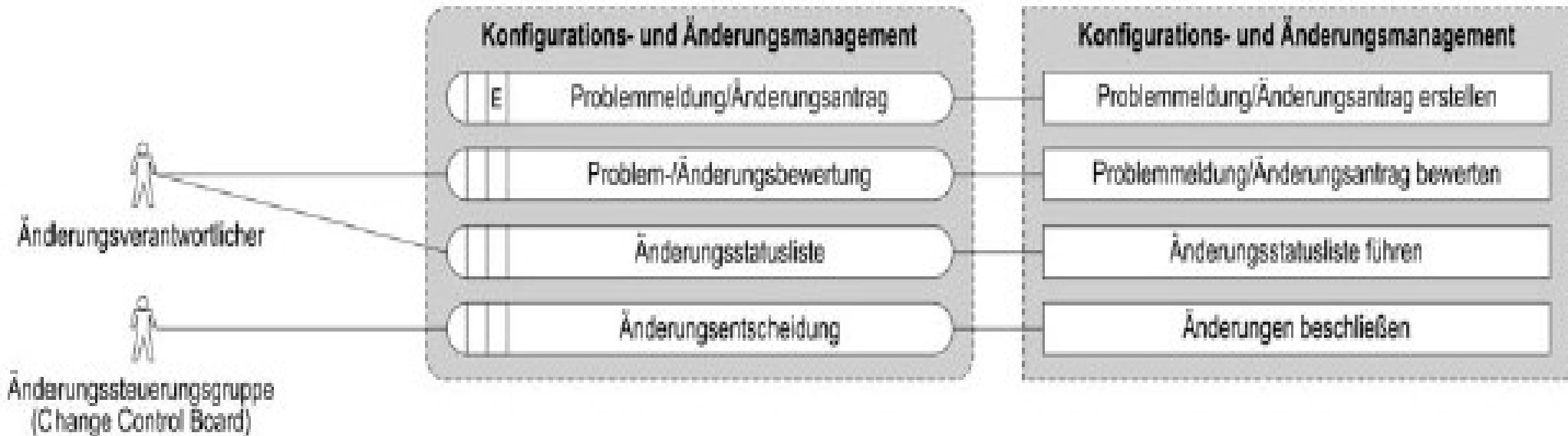


# Aufgaben der Aktivitäten des Problem- und Änderungsmanagements laut V-Modell XT

- ▶ **Zustandserfassung** von Problemmeldungen/Änderungsanträgen (korrektive Wartung, Pflege)
- ▶ Dokumentieren und Verwaltung aller Problemmeldungen und Änderungsanträge über eine **Statusliste**
- ▶ Änderungen **bewerten** (Ursachen, Auswirkungen, ...)
- ▶ Entscheidung, **Freigabe** und Veranlassung der Bearbeitung
- ▶ **Abschluss** der Änderung, Information der Betroffenen
- ▶ Erfassung von Problemmeldungen, Fehlermeldungen, Verbesserungsvorschlägen und Änderungswünschen



# Vorgehensbaustein Problem- und Änderungsmanagement



Notwendige Artefakte (Produkte, Belege) sind:

- Problemmeldung und Änderungsantrag
- Problem- und Änderungsbewertung
- Änderungsentscheidung, -mitteilung
- Änderungsstatusliste

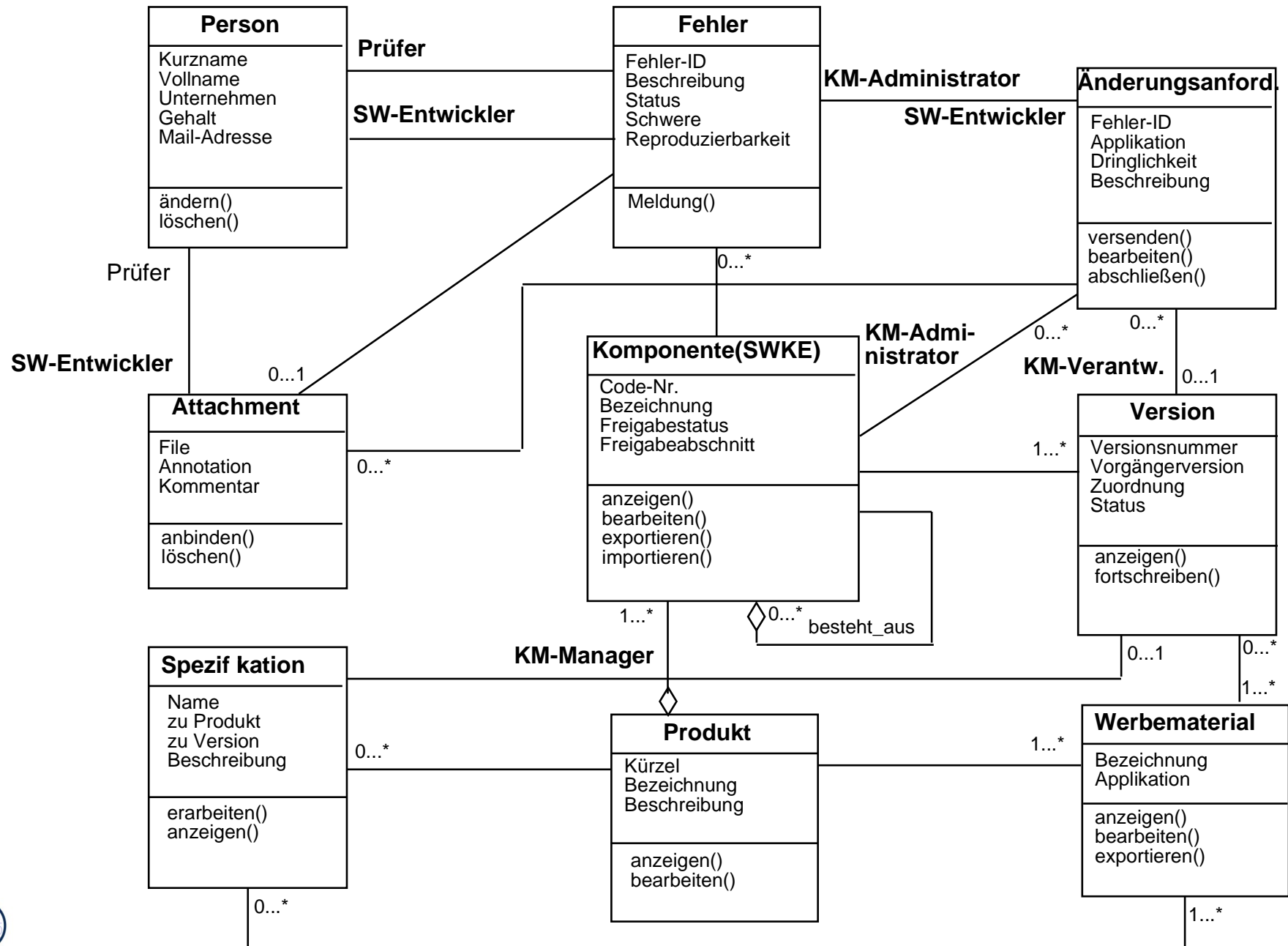
Sie werden in den zugehörigen Aktivitäten des V-Modells XT bearbeitet.



# Änderungsmanagement

- ▶ ist nötig für Firmen, die sukzessive neue Versionen ihrer Produkte erzeugen
  - im Produktgeschäft tätig sind
  - im Produktlinien-Geschäft tätig sind
- ▶ weniger nötig für eine Anwendungslandschaft in einer Firma

# Metamodell des Änderungsmanagements



# Aktivität Problemmeldung/ Änderungsantrag erstellen

- ▶ Jede Rolle kann aus den verschiedensten Gründen eine Problemmeldung/ **Änderungsantrag** auslösen, der grundsätzlich folgende Informationen enthält:
  - Beschreibung des Problems bzw. der gewünschten Änderung
  - Identifikation Antragsteller, Projekt, betroffenen Konfiguration
  - Begründung des Antrages bzgl. Nutzen bzw. Schaden bei Nichtdurchführung
  - Lösungsvorschlag aus Sicht des Antragstellers
  - Nummer Änderungsantrag/Problemmeldung
  - Vergabe einer Registriernummer pro Problemmeldung/Änderungsantrag
- ▶ Gründe für Änderungen können sein:
  - neue Entwicklungserfordernisse
  - Probleme im Controlling: Zeitprobleme, Kosteneinhaltung
  - Compliance: Änderungen gesetzlicher Vorschriften
    - Sarbanes-Oxley Act, Basel-II
  - Verbesserung von Marktchancen
    - Erscheinen neuer Plattformen
  - Änderungen in den Anforderungen: Nutzerwünsche

# Aktivität Problemmeldung/ Änderungsantrag bewerten

- ▶ Problemmeldung/Änderungsantrag analysieren wie dringend Lösung des Problems bzw. der beantragten Änderung ist
- ▶ Lösungsvorschläge erarbeiten mit vollständiger bzw. auch erst nur teilweiser Lösung. Folgende Informationen sollte er enthalten:
  - Teile des Projektes, die von der Änderung betroffen sind
  - Phase des Entwicklungsprozesses, in der Änderung anfällt
  - Lösungsbeschreibung und -vorgehen
  - erforderliche Aufwendungen
  - Auswirkungen der Änderung auf das Projekt
- ▶ Empfehlung aussprechen:
  - auf Basis der erarbeiteten alternativen Lösungsvorschläge
  - alle Lösungsvorschläge sind anhand ihrer Auswirkungen auf das Projekt zu bewerten
  - aus dieser Basis ist eine Entscheidung zu fällen und zu begründen
- ▶ Alle Bewertungsaktivitäten werden im Produkt Problem-/Änderungsbewertung niedergelegt

# Aktivität

## Änderungen beschließen

- ▶ Vorbereitung des Entscheidungsmeetings durch Sammeln alle Anträge und Bewertungen, **Erstellen der Agenda** für das Meeting
- ▶ Einladungen an beteiligte Rollen oder Stakeholder **verschicken**
- ▶ Anträge vorstellen und **präsentieren** mit:
  - entstehenden Kosten
  - Verfügbarkeit von Mitteln und Personal
  - zeitliche Projektverzögerung
  - technische Eignung der vorgeschlagenen
- ▶ Änderungsentscheidung **beschließen** und Dringlichkeit der Umsetzung festlegen
  - Festlegung der Kategorie (Fehler [in Spezifikation, Entwurf, Codierung, im Verfahren], Problem, Modifikation, Erweiterung, Verbesserung, usw.)
  - gewünschter Fertigstellungszeitpunkt
- ▶ Auswirkungen der Änderung ermitteln
- ▶ Änderungsentscheidung – im Änderungsbescheid - **protokollieren**
- ▶ Änderungsentscheidung **verteilen** bzw. kommunizieren
- ▶ Alle beschlossenen Änderungen werden im Produkt Änderungsentscheidung/-mitteilung niedergelegt

# Aktivität

## Änderungsstatusliste führen

- ▶ Änderungsstatusliste dient dem Ziel, alle wichtigen Informationen zum Projekt hinsichtlich Änderungsanforderungen und -auswirkungen zu aktualisieren und dokumentieren
- ▶ Ablauf und Dokumentation ist für jede Änderungsanforderung gleich:
  - **Änderungsanforderungen registrieren** mit Prüfung der benötigten Daten auf Vollständigkeit
  - **Änderungsanforderungen prüfen** auf Realisierbarkeit und Festlegung der erforderlichen Mittel, Termine und Verantwortlichkeiten
  - **Änderungsstatusliste aktualisieren** nach bereits bestehenden Änderungsanforderungen bzw. durch Hinzufügen neuer Anforderungen
- ▶ Änderungsstati sind z. B.: ‚beantragt‘, ‚beabsichtigt‘, ‚abgelehnt‘, ‚genehmigt‘, ‚zurückgestellt‘, ‚beauftragt‘, ‚erledigt‘
- ▶ Bemerkungen bei Beziehungen zu bereits gestellten Änderungsanträgen
- ▶ Referenzen auf die Änderungsbewertung oder die Änderungsentscheidung sind in der Änderungsstatusliste ebenfalls festzuhalten
- ▶ Wird oft in einer Datenbank geführt
  - z.B. MANTIS-System



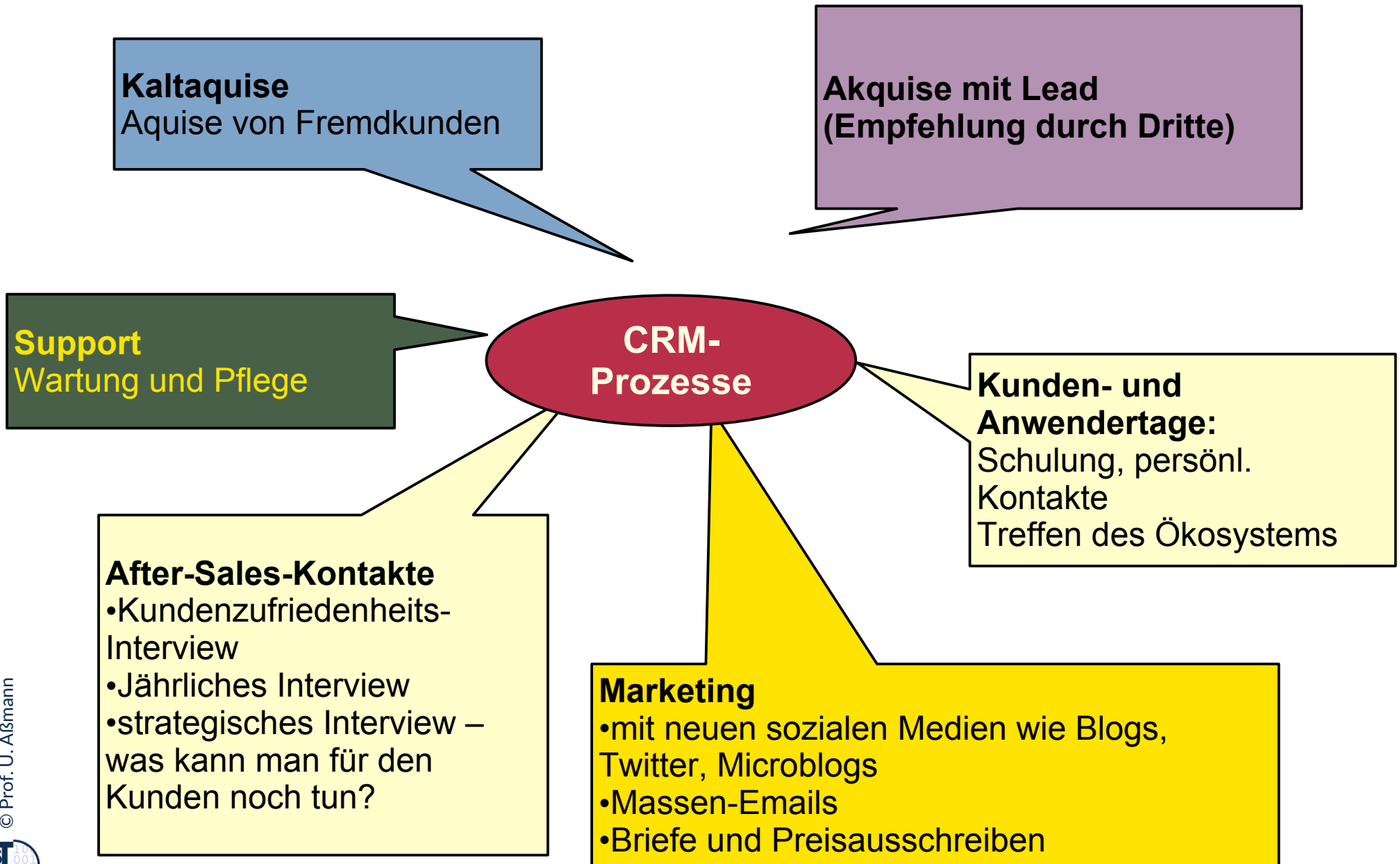
## 20.6 Kundenmanagement



Das **Kundenmanagement (Customer relationship management, CRM)** verwaltet die Kontaktdaten und Geschichte der Interaktionen mit Kunden (Dokumente, Verträge, emails, Telefonnotizen, Kontaktnotizen). Ziel ist es, kundenzentriert alle Daten auf einen Blick, an einem Platz zu sammeln und für die ganze Firma zugreifbar zu halten.

- ▶ Besonders wichtig bei
  - **Stammkunden**, denn man darf keine Fehler mit ihnen machen
  - einem **Massenmarkt** mit Tausenden von Kunden
- ▶ Ein gutes CRM ist eine der wichtigsten Prozesse einer Softwarefirma, weil das Feedback des Kunden die wichtigste Voraussetzung zum Halten von Stammkunden ist





- ▶ Ein CRM-System ist ein Informationssystem, das folgende Aufgaben erfüllt:
  - Kontakt-Datenbank mit Geschichte der Kontakte und Interaktionen
  - Unterstützung aller CRM-Prozesse (s.o.)
  - Schnittstellen zu anderen Systemen, wie dem Dokumentenmanagementsystem, dem ERP-System, dem Aufgabenmanagement
- ▶ Die Auswahl von passenden CRM-Systemen wird als schwierig eingeschätzt, ist aber unerlässlich für den Erfolg der Firma

Beispiele			
vtiger	OSS	<a href="https://www.vtiger.com">https://www.vtiger.com</a>	
Dynamics	Microsoft	<a href="http://www.microsoft.com/dynamics/">http://www.microsoft.com/dynamics/</a>	
salesforce	Salesforce	<a href="http://www.salesforce.com/">http://www.salesforce.com/</a>	Cloud-basiert
SugarCRM	dual licensing	<a href="http://www.sugarcrm.com/">http://www.sugarcrm.com/</a>	

- ▶ [http://en.wikipedia.org/wiki/CRM\\_software](http://en.wikipedia.org/wiki/CRM_software)
- ▶ [http://en.wikipedia.org/wiki/Comparison\\_of\\_CRM\\_systems](http://en.wikipedia.org/wiki/Comparison_of_CRM_systems)

# The End

- ▶ Erklären Sie die Rolle von Metriken im QS
- ▶ Erklären Sie die Rolle von Software-Analyse-Werkzeugen im QS
- ▶ Erklären Sie die EAL-Level.
- ▶ Warum ist es für eine Firma oft leichter, Prozesse für QS zu etablieren als die Produktqualität analytisch zu prüfen?

# Future Topics

100 Softwaremanagement (SWM)

---

- ▶ Six Sigma
- ▶ Metriktools von Welf