Vorlesung (WS 2014/15)

Softwarekonstruktion

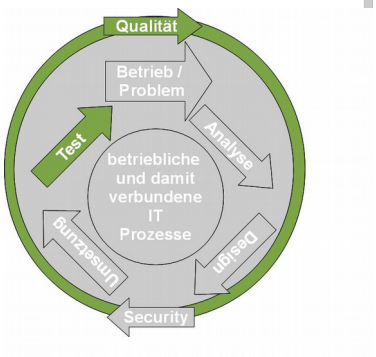
Prof. Dr. Jan Jürjens

TU Dortmund, Fakultät Informatik, Lehrstuhl XIV

2.0: Qualitätsmanagement

v. 08.12.2014

- Modellgetrieben SW-Entwicklung
- Qualitätsmanagement
- Testen



Basierend auf Beiträge von Prof. Summerville (St. Andrews University), Prof. Martin Glinz (Universität Zürich) und dem Foliensatz „Basiswissen Softwaretest - Certified Tester“ des „German Testing Board“.

Literatur (s. Webseite):

- J. Ludewig, H. Lichter: Software Engineering - Grundlagen, Menschen, Prozesse, Techniken. Kapitel 5.

Literatur:

H. Balzert: **Lehrbuch der Software-Technik/2 –** Software-Management, Software-Qualitätssicherung
http://www.ub.tu-dortmund.de/katalog/titel/645541

- Teil III – Kapitel 1
- Teil III – Kapitel 4

- **Letzter Abschnitt:** Technische Grundlagen für UML-Werkzeuge und MDA.
- **Dieses Kapitel 2:** Von Softwareentwicklung zu Softwareverifikation.
  - Dieser Abschnitt 2.0:
    Kurze **Einführung** und **Motivation** für die Inhalte von Kapitel 2: **Qualitätsmanagement**

2.0 Qualitätsmanagement

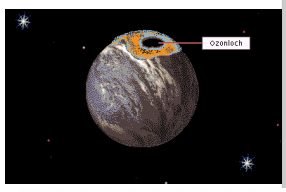
2.0 Qualitäts-management

- Motivation
- Softwarequalität

Auswirkungen von Software-Fehlern

NASA - Erdbeobachtungssatelliten 1979-1985

- **Ozonloch** 7 Jahre (!) lang **nicht erkannt.**
- Ursache: Softwarefehler – Veränderung der Ozonschicht als Sensordrift durch automatische Nullpunktkorrektur »herausgemittelt«.

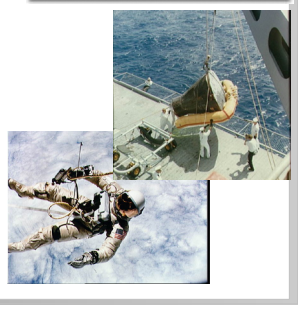


ESA, Kourou, Franz. Guyana, 4. Juni 1996

- **Selbstzerstörung der Ariane** 5 beim Jungfernflug 39 Sekunden nach Start.
- Ursache: Softwarefehler – Lageregelungssoftware aus Ariane 4 ohne Test gegen Start-Trajektorie der Ariane 5 wiederverwendet, dadurch Konvertierungsfehler.

Bemannte **NASA-Raumkapsel** Gemini V

- **Verfehlte ihren Landeplatz** um ca. 160 Kilometer.
- Ursache: Softwarefehler – Rotation der Erde um die Sonne nicht berücksichtigt !



Auswirkungen von Software-Fehlern

21. April 2009: Software-Fehler legt T-Mobile Netz lahm:

- Millionen **T-Mobile-Kunden** waren **nicht erreichbar.**
- Nach wählen der Nummer kam von Frauenstimme:

  „Dieser Anschluss ist aus technischen Gründen vorübergehend nicht erreichbar. Bitte rufen Sie später wieder an!“
- System wurde zurückgesetzt und neu gestartet.
- Ursache: **Softwarefehler** im Home Location Register (HLR).

  → Zuordnung der Telefonnummern einzelnen SIM-Karten.
- 3 Datenbanken mussten verfügbar gemacht werden.

NASA - Mariner 1: Codierfehler

NASA verliert auf Weg zur Venus befindliche **Raumsonde** Mariner 1 am 22.6.1962:
»Because of a launch-vehicle deviation from the planned flight path, Mariner R-1 was destroyed by the range safety officer after approximately 290 seconds of flight.«

- **Korrekt** codiert wäre gewesen:
  ```
  DO 10 I=1,3     (Definition einer Schleife  in FORTRAN IV)
      ...
  10 CONTINUE
  ```
- **Fehlerhaft** codiert wurde
  ```
  DO10I=1.3
      ...
  10 CONTINUE
  ```

Welchen Effekt hat dieser Fehler wohl?

NASA - Mariner 1: Codierfehler

NASA verliert auf Weg zur Venus befindliche **Raumsonde** Mariner 1 am 22.6.1962:
»Because of a launch-vehicle deviation from the planned flight path, Mariner R-1 was destroyed by the range safety officer after approximately 290 seconds of flight.«

- **Korrekt** codiert wäre gewesen:

```
DO 10 I=1,3      (Definition einer Schleife  in FORTRAN IV)
    ...
10 CONTINUE
```

- **Fehlerhaft** codiert wurde

```
DO10I=1.3        (Zuweisung des Werts 1.3
    ...           an die Variable DO10I)
10 CONTINUE
```

NASA - Mariner 1: Codierfehler (2)

Nur FORTRAN-Problem? Nein!

Was ist an den folgenden Ausdrücken das Problem und welchen Effekt hat das ?

C
```
for (i=1; i<=3; i++);
    f(i);
```

Java, C#
```
for (i=1; i<=3; i++){}
    f(i);
```

Perl
```
for ($i=1;$i<=3;$i++){}
   &f(i);
```

Ergänzung von H. Klaeren:
Probleme des Software-Engineering.
Informatik Spektrum (1994) 17: 21-28

NASA - Mariner 1: Codierfehler (2)

Nur FORTRAN-Problem? Nein!

Versehentliche nur 1-malige Ausführung mit i=4

(Annahme: i vorab definiert)

C
```
for (i=1; i<=3; i++);
    f(i);
```

Java, C#
```
for (i=1; i<=3; i++){}
    f(i);
```

Perl
```
for ($i=1;$i<=3;$i++){}
   &f(i);
```

Ergänzung von H. Klaeren:
Probleme des Software-Engineering.
Informatik Spektrum (1994) 17: 21-28

Diskussion: Austesten ?

Programm mit 3 Integer-Eingabewerten testen.
(Annahme: Randbedingungen haben keinen Einfluss auf Testobjekt).

Wieviele Tests für alle möglichen Kombinationen von Eingaben ?

Wie lange dauert das Testen bei **100.000 Tests pro Sekunde** ?

- Mehr als 1 millisec ?
- Mehr als 1 sec ?
- Mehr als 1 min ?
- Mehr als 1 h ?
- ...

Diskussion: Austesten ?

Programm mit 3 Integer-Eingabewerten testen.
(Annahme: Randbedingungen haben keinen Einfluss auf Testobjekt).

Wieviele Tests für alle möglichen Kombinationen von Eingaben ?

Wie lange dauert das Testen bei **100.000 Tests pro Sekunde** ?

- Jeder **Eingabewert** kann bei 16 Bit Integerzahlen **2^{16}** unterschiedliche **Werte** annehmen.
- Bei drei unabhängigen Eingabewerten ergeben sich $2^{16} * 2^{16} * 2^{16} = 2^{48}$ Kombinationen.

=> 281.474.976.710.656 Testfälle.
Dauer: ca. 90 Jahre.

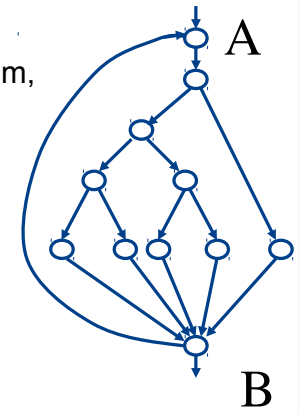
Diskussion: Austesten auf Kontrollflussgraph ?

Idee: Maximale denkbare **Testmenge einschränken.**

- Durch Verwendung interner Informationen über Programm, z.B. Kontrollflussgraph.

Beispiel (s. Abb.): Testen eines Programms, das aus

- **vier Verzweigungen** (IF-Anweisungen) und
- einer umfassenden **Schleife** besteht und
- **fünf mögliche Wege** im Schleifenrumpf enthält.



Annahmen:

- Verzweigungen voneinander **unabhängig**.
- **Max. 20** Schleifendurchläufe.

Wie viele Tests für alle möglichen Pfade im Kontrollflussgraph ?

Wie lange dauert das Testen bei **100.000 Tests pro Sekunde** ?

Mehr als 1 ms ?
Mehr als 1 sec ?
Mehr als 1 min ?
Mehr als 1 h ?
...

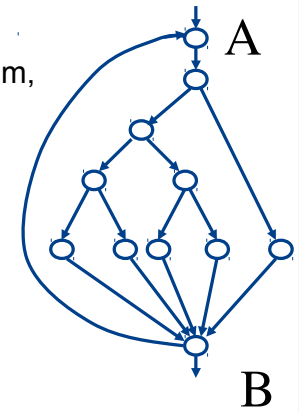
Diskussion: Austesten auf Kontrollflussgraph ?

Idee: Maximale denkbare **Testmenge einschränken.**

- Durch Verwendung interner Informationen über Programm, z.B. Kontrollflussgraph.

Beispiel (s. Abb.): Testen eines Programms, das aus

- **vier Verzweigungen** (IF-Anweisungen) und
- einer umfassenden **Schleife** besteht und
- **fünf mögliche Wege** im Schleifenrumpf enthält.



Annahmen:

- Verzweigungen voneinander **unabhängig**.
- **Max. 20** Schleifendurchläufe

Wie viele Tests für alle möglichen Pfade im Kontrollflussgraph ?

Wie lange dauert das Testen bei **100.000 Tests pro Sekunde** ?

→ **Rechnung:** $5^1 + 5^2 + ... + 5^{18} + 5^{19} + 5^{20}$

- Es sind 119.209.289.550.780 Testfälle. Dauer: ca. 38 Jahre.

Diskussion: Testen auf Fallunterscheidungen ?

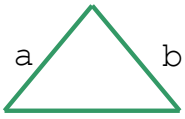
Weitere Idee, um Testaufwand zu begrenzen: **Fallunterscheidungen.**

- Dann z.B. je ein Test pro Fallunterscheidung.

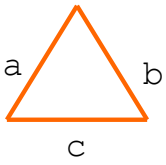
Beispiel: Programm testen, das

- 3 ganzzahlige positive **Werte** einliest,
- sie als **Längen** eines Dreiecks interpretiert und
- ausgibt, ob Dreieck ungleichseitig, gleichschenklig oder gleichseitig ist.

$a \neq b$ und
$b \neq c$ und
$a \neq c$



$a = b \neq c$ oder
$a \neq b = c$ oder
$a = c \neq b$



$a = b = c$

In Anlehnung an Glenford J. Myers: Methodisches Testen von Programmen. 7. Auflage 2001

Wieviele / welche Fallunterscheidungen ergibt dies ?

min. 3 ?
min. 5 ?
min. 8 ?
min. 11 ?
...

Diskussion: Testen auf Fallunterscheidungen ?

Mind. 31 Fallunterscheidungen mit folgenden Testdaten und Soll-Ergebnissen:

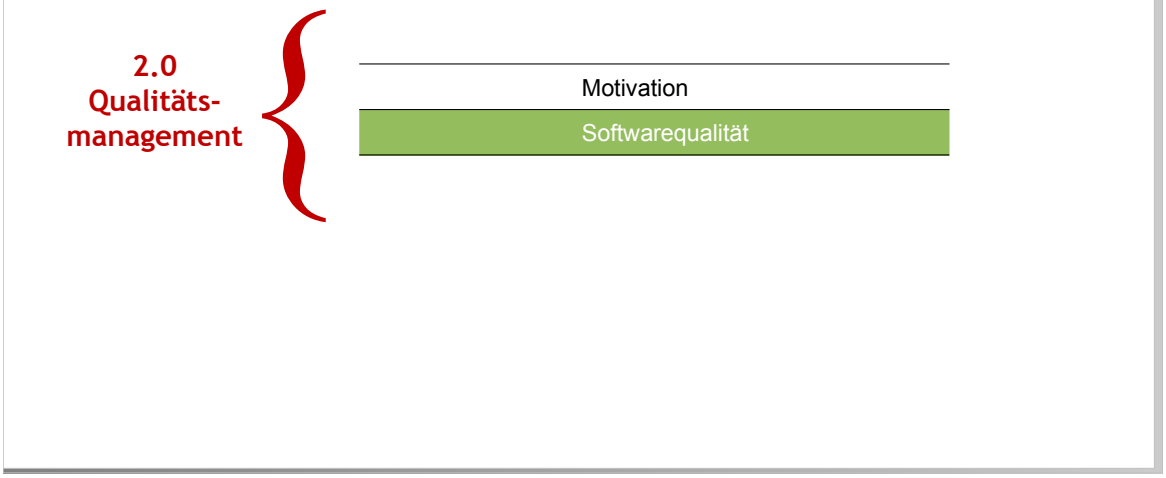
Testnr. Testdaten (die drei Längen des Dreiecks):

1. 2,3,4 - zulässiges ungleichseitiges Dreieck
2. 2,2,2 - zulässiges gleichseitiges Dreieck
3. 2,2,1 - zulässiges gleichschenkliges Dreieck
4./5. 1,2,2 / 2,1,2 - zwei weitere Testfälle mit Permutationen für gleichschenklige Dreiecke
6. 1,0,3 - kein Dreieck, eine Seitenangabe = 0
7./8. 0,1,3 / 1,3,0 - Permutationen
9. 5,-5,9 - kein Dreieck, eine Seitenangabe < 0
10./11. -5,5,9 / 5,9,-5 – Permutationen
12. 1,2,3 - kein Dreieck, Summe der beiden kürzeren Seiten = 3. Seitenlänge
13./14. 2,3,1 / 3,1,2 - Permutationen
15. 1,2,4 - kein Dreieck, Summe der beiden kürzeren Seiten < 3. Seitenlänge
16./17. 2,4,1 / 4,1,2 - Permutationen

Resümee: Einfaches Problem, aber aufwendiger Test

Testnr. Testdaten (die drei Längen des Dreiecks):

18./19. 0,0,0 - kein Dreieck oder Fehlermeldung, alle Seiten = 0, zusätzlich 2 Seiten = 0 - Permutationen?
20.-22. Max_int, Max_int, Max_int – zulässiges gleichseitiges Dreieck, korrekte Dreiecksbestimmung beim Test mit maximalen Werten, zusätzliche Tests mit 2 oder 1 maximalem Wert
23.-25. 1,1,4.4567 - Fehlermeldung »nicht ganzzahlige Werte« Permutationen? - zusätzlich mit 2 oder 3 Werten
26.-28. 1,1,& - Fehlermeldung »Eingabe von Buchstaben oder Sonderzeichen«. Permutationen? - zusätzlich mit 2 oder 3 Werten
29./30. 1,2,3,4 / 2,3 - Fehlermeldung »falsche Anzahl von Werten« (wenn Eingabe möglich)
31. Max_int/2 + 1, Max_int/2 + 1, Max_int/2 + 10 - zulässiges gleichschenkliges Dreieck (Überlauf oder richtige Berechnung? Bei a<=b<=c; Prüfung der Dreiecksbedingung mit a+b>c, führt a+b zum Überlauf, s.a. Testfall 20)



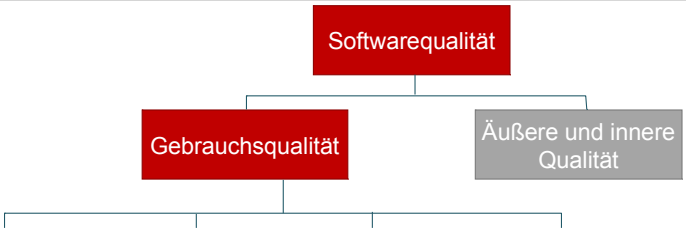
Literatur:

H. Balzert: **Lehrbuch der Software-Technik/2 –** Software-Management, Software-Qualitätssicherung
http://www.ub.tu-dortmund.de/katalog/titel/645541

- Teil III – Kapitel 1
- Teil III – Kapitel 4

Software-Qualität nach ISO/IEC 9126 (1)

Wiederholung SWT



Welche Software-Qualitätseigenschaften kennen Sie ?

(Wiederholung von SWT)

ISO/IEC 9126: Bewerten von Softwareprodukten, Qualitätsmerkmale und Leitfaden zu ihrer Verwendung. (Seit 2005 ersetzt durch ISO/IEC 25000 Software engineering – Software product Quality Requirements and Evaluation (SQuaRE).)

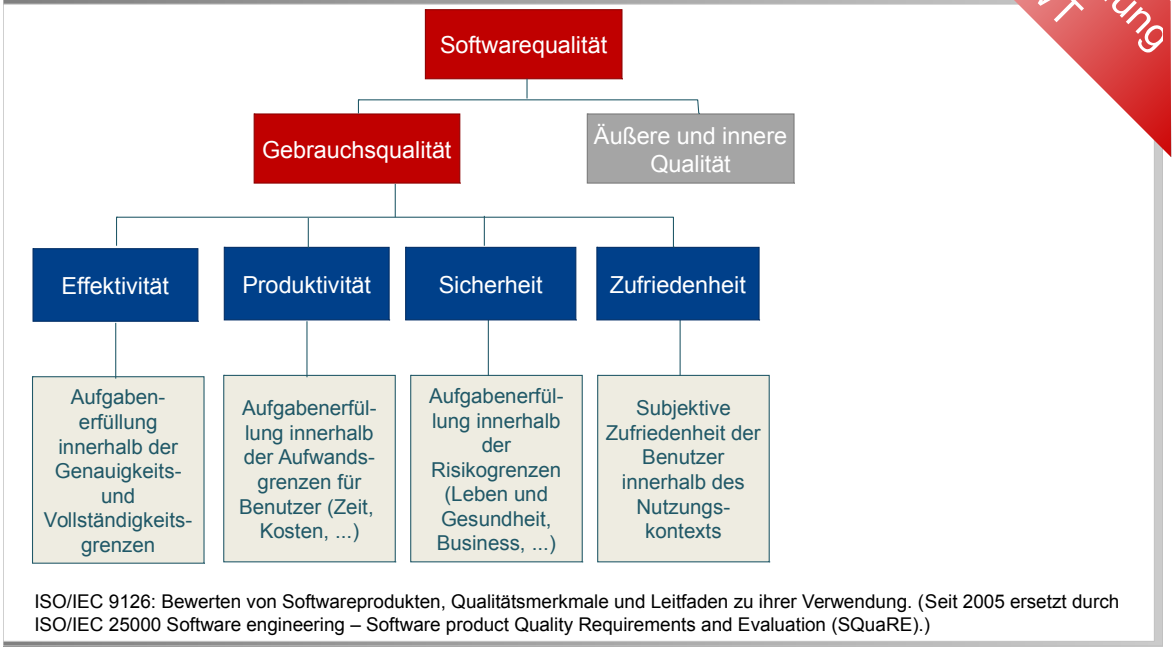
Literatur:

H. Balzert: **Lehrbuch der Software-Technik/2 –** Software-Management, Software-Qualitätssicherung
http://www.ub.tu-dortmund.de/katalog/titel/645541

- Teil III – Abschnitt 1.1 (S.256-257)
- Teil III – Abschnitt 1.2 (S.257-262)

Folien-swt-2014-klein-01: F. 5-8

Software-Qualität nach ISO/IEC 9126 (1)



ISO/IEC 9126: Bewerten von Softwareprodukten, Qualitätsmerkmale und Leitfaden zu ihrer Verwendung. (Seit 2005 ersetzt durch ISO/IEC 25000 Software engineering – Software product Quality Requirements and Evaluation (SQuaRE).)

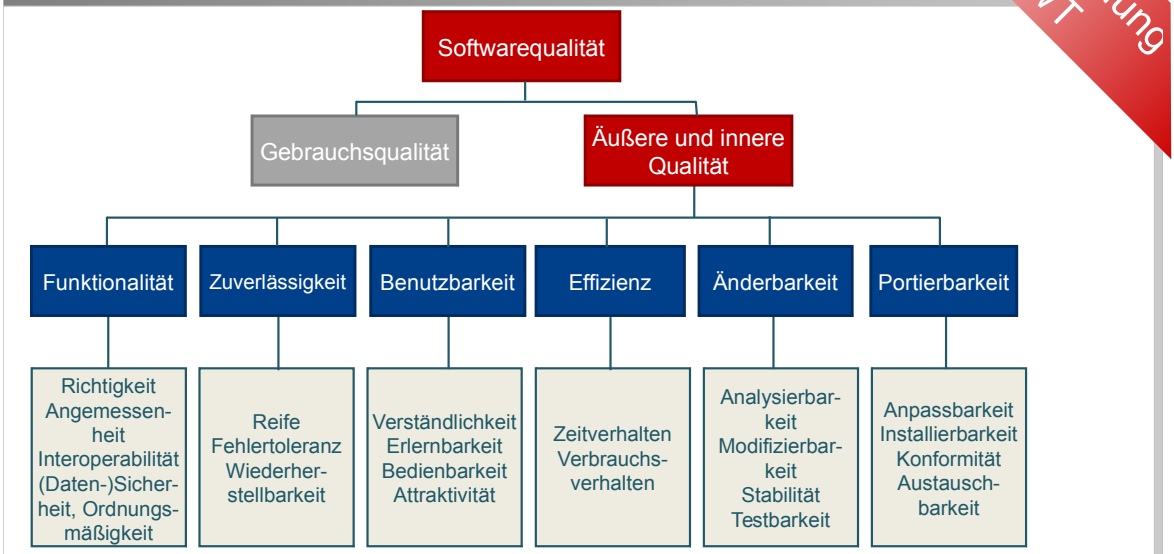
Literatur:

H. Balzert: **Lehrbuch der Software-Technik/2 –** Software-Management, Software-Qualitätssicherung
http://www.ub.tu-dortmund.de/katalog/titel/645541

- Teil III – Abschnitt 1.1 (S.256-257)
- Teil III – Abschnitt 1.2 (S.257-262)

Folien-swt-2014-klein-01: F. 5-8

Software-Qualität nach ISO/IEC 9126 (2)



ISO/IEC 9126: Bewerten von Softwareprodukten, Qualitätsmerkmale und Leitfaden zu ihrer Verwendung. (Seit 2005 ersetzt durch ISO/IEC 25000 Software engineering – Software product Quality Requirements and Evaluation (SQuaRE).)

Literatur:

H. Balzert: **Lehrbuch der Software-Technik/2 –** Software-Management, Software-Qualitätssicherung
http://www.ub.tu-dortmund.de/katalog/titel/645541

- Teil III – Abschnitt 1.1 (S.256-257)
- Teil III – Abschnitt 1.2 (S.257-262)

Folien-swt-2014-klein-01: F. 5-8

Diskussion: Qualitätskonflikte

Welche Qualitätsmerkmale können Ihrer Ansicht nach im Konflikt zueinander stehen ?

Literatur:

H. Balzert: **Lehrbuch der Software-Technik/2 –** Software-Management, Software-Qualitätssicherung
http://www.ub.tu-dortmund.de/katalog/titel/645541

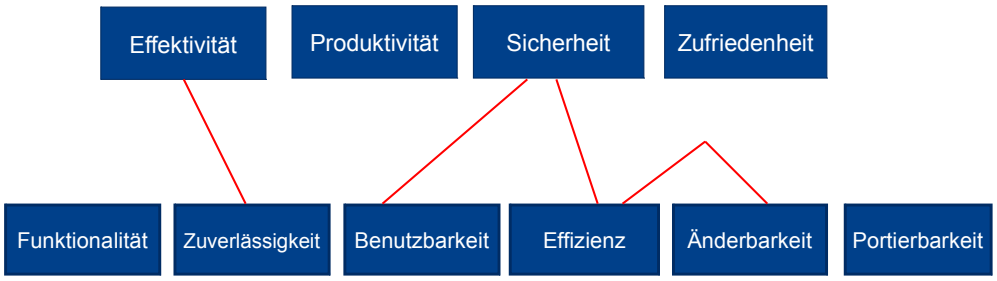
- Teil III – Abschnitt 1.1 (S.256-257)
- Teil III – Abschnitt 1.2 (S.257-262)

Folien-swt-2014-klein-01: F. 5-8

Diskussion: Qualitätskonflikte

Welche Qualitätsmerkmale können Ihrer Ansicht nach im Konflikt zueinander stehen ?

Zum Beispiel:



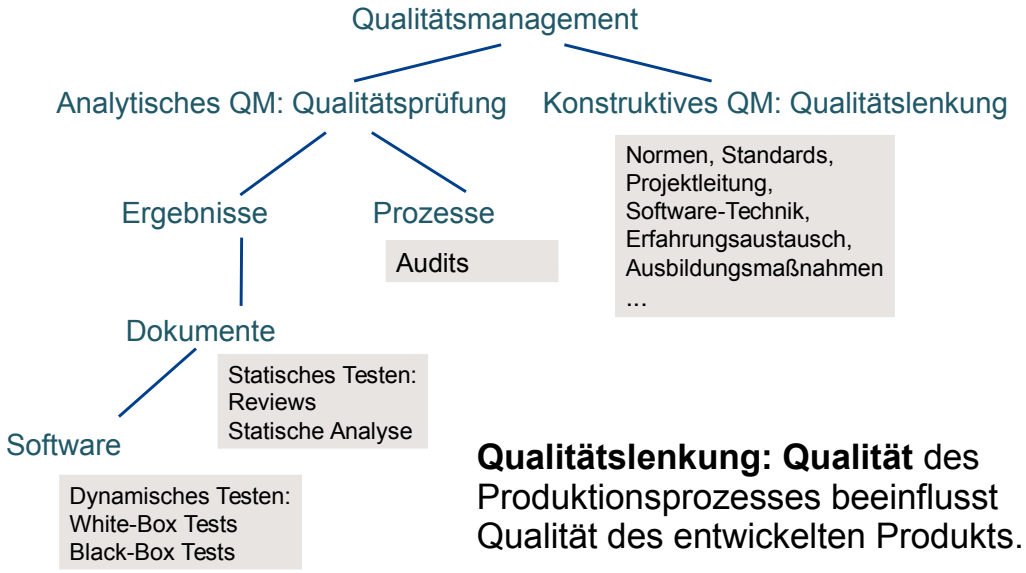
Literatur:

H. Balzert: **Lehrbuch der Software-Technik/2 –** Software-Management, Software-Qualitätssicherung
http://www.ub.tu-dortmund.de/katalog/titel/645541

- Teil III – Abschnitt 1.1 (S.256-257)
- Teil III – Abschnitt 1.2 (S.257-262)

Folien-swt-2014-klein-01: F. 5-8

Qualitätslenkung vs. -prüfung



Qualitätslenkung: Qualität des Produktionsprozesses beeinflusst Qualität des entwickelten Produkts.

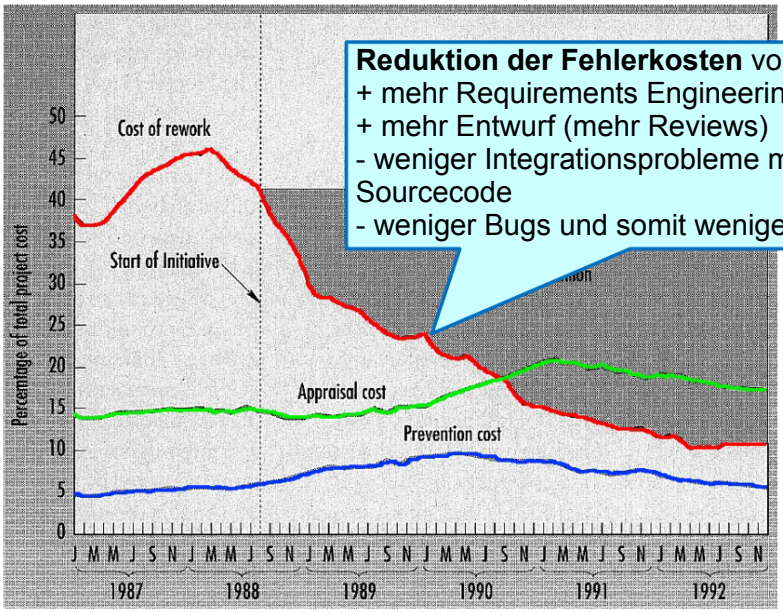
Literatur:

H. Balzert: **Lehrbuch der Software-Technik/2 –** Software-Management, Software-Qualitätssicherung
http://www.ub.tu-dortmund.de/katalog/titel/645541

- Teil III – Abschnitt 2.1 (S.278-284)
- Abbildung 2.1-1 (S.280)
- Abbildung 2.1-2 (S.281)

Was bringt Prozessverbesserung ?

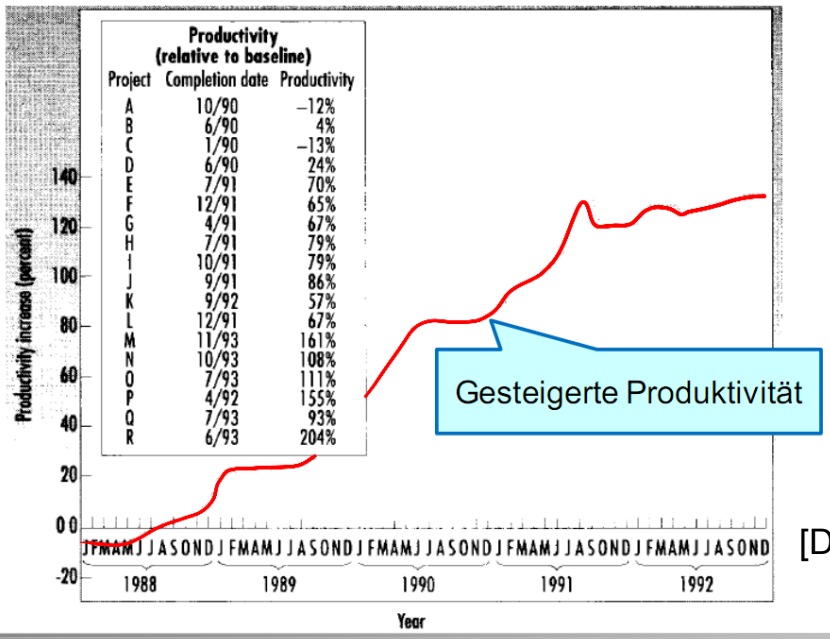
Reduktion der Fehlerkosten



Reduktion der Fehlerkosten von 41% auf 11%:
+ mehr Requirements Engineering
+ mehr Entwurf (mehr Reviews)
- weniger Integrationsprobleme mit dem Sourcecode
- weniger Bugs und somit weniger Nachtesten

[Dion 1993]

Was bringt Prozessverbesserung ?
Gesteigerte Produktivität



[Dion 1993]

Prozessqualität: Normenreihe ISO 900x

- Internationale Sammlung von Standards.
  → Als Basis des Qualitätsmanagements verwendbar.
- 1987 eingeführt.

Grundlage ISO 9000:

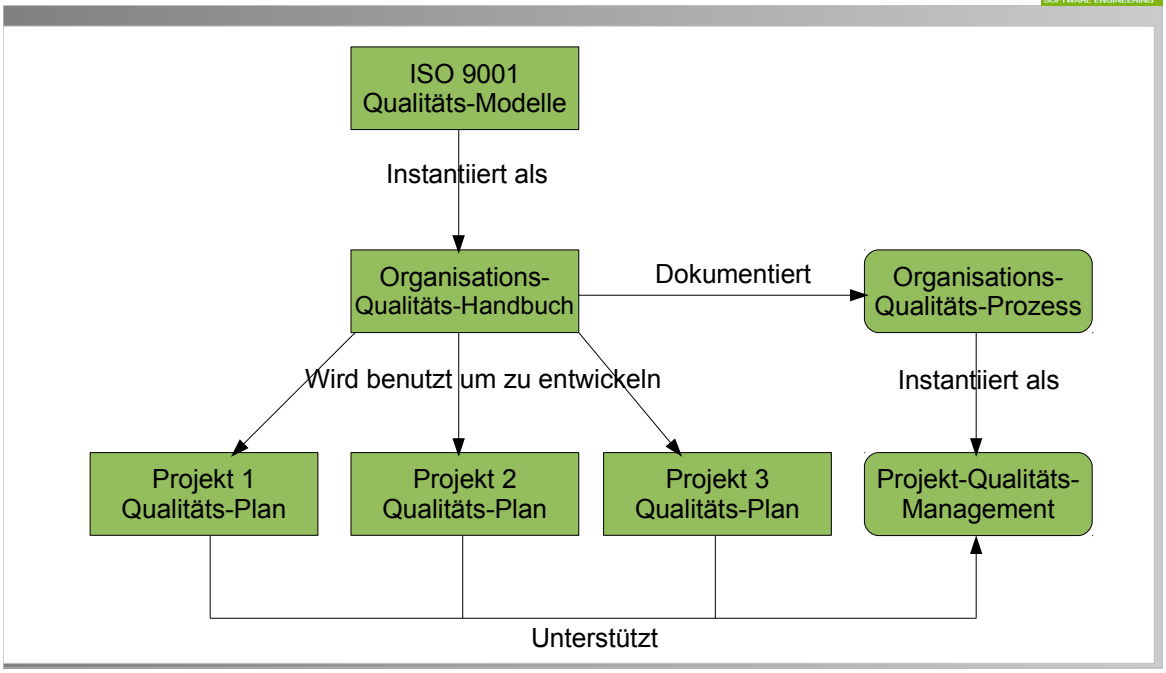
- ISO 9000 **„Qualitätsmanagementsysteme - Grundlagen und Begriffe“**
- Erläutert Grundlagen für Qualitätsmanagementsysteme und in Normenreihe ISO 900x verwendete Begriffe.
- Erklärt **prozessorientierten Ansatz** des Qualitätsmanagements.
- Aktuelle Version von 2005 (ISO 9000:2005).

Literatur:

H. Balzert: **Lehrbuch der Software-Technik/2 –** Software-Management, Software-Qualitätssicherung
http://www.ub.tu-dortmund.de/katalog/titel/645541

- Teil III – Abschnitt 4.1 – ISO 9000-Ansatz (ab S.328)
- Teil III – Abschnitt 4.1.1 (S.330-333)

ISO 9001: Qualitätsmanagement



Literatur:

H. Balzert: **Lehrbuch der Software-Technik/2 –** Software-Management, Software-Qualitätssicherung
http://www.ub.tu-dortmund.de/katalog/titel/645541

- Teil III – Abschnitt 4.1 – ISO 9000-Ansatz (ab S.328)
- Teil III – Abschnitt 4.1.1 (S.330-333)

ISO 9001-Zertifizierung

- **Unternehmens-Qualitätshandbuch** sollte Qualitätsstandards und Abläufe enthalten.
- Externe Stelle muss Konformität des Qualitätshandbuchs mit ISO 9001 bestätigen.
- Manche Kunden verlangen **ISO 9001-Konformität** von Anbietern.
- Weltweit mehr als 1 Millionen Organisationen ISO 9001-zertifiziert.
- Trotzdem als **aufwendig** und nicht für alle Unternehmen gleichermaßen geeignet kritisiert.

Literatur:

H. Balzert: **Lehrbuch der Software-Technik/2 –** Software-Management, Software-Qualitätssicherung
http://www.ub.tu-dortmund.de/katalog/titel/645541

- Teil III – Abschnitt 4.1 – ISO 9000-Ansatz (ab S.328)
- Teil III – Abschnitt 4.1.1 (S.330-333)

"ISO 9001 certifications top one million mark, food safety and information security continue meteoric increase" (Press release). International Organization for Standardization. October 25, 2010. http://www.iso.org/iso/pressrelease.htm?refid=Ref1363.

Diskussionsfrage: ISO 9001 vs. Produktqualität

- Mit ISO 9001 Qualität eines Softwareprodukts zertifizierbar ?
- **Antwort:**

Diskussionsfrage: ISO 9001 vs. Produktqualität

- Mit ISO 9001 Qualität eines Softwareprodukts zertifizierbar ?
- **Antwort:**
  - Nein, ISO 9001-Zertifikat besagt nur, dass Qualitätsmanagement der Firma der ISO 9001 entspricht.

Prozessverbesserung:
Capability Maturity Model Integrated (CMMI)

Wurzeln:

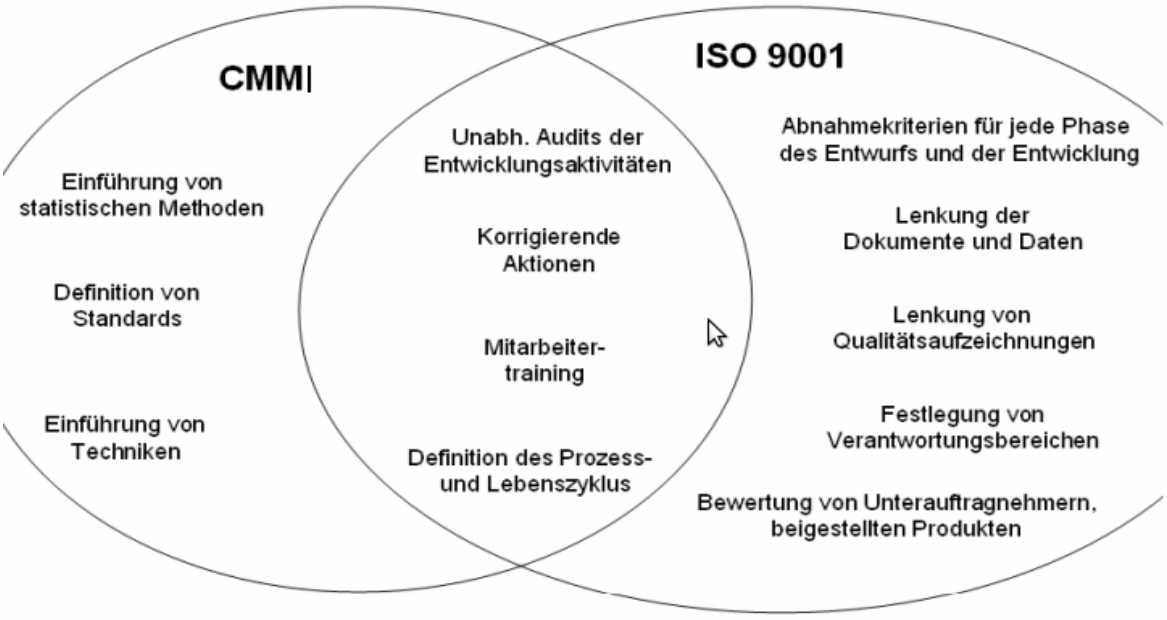
- Systematische **Prozessverbesserung** (Deming 1986).
- Prozessorientierte Software-Entwicklung (Humphrey 1989).
- Beurteilung des **Reifegrads** der Prozesse eines Software-Lieferanten.
- **Capability Maturity Model (CMM)** (Paulk et al. 1993).
- Modelle als Grundlage und Gerüst für Prozessverbesserung:
  - CMM.
  - SPICE – Software Process Improvement and Capability Determination (ISO/IEC 15504).
- CMM spezialisiert für Systeme, Leute, Beschaffung,...
- Entwicklung eines umfassenden, zuschneidbaren Rahmenmodells: **CMMI (Capability Maturity Model Integrated).**

Literatur:

H. Balzert: **Lehrbuch der Software-Technik/2 –** Software-Management, Software-Qualitätssicherung
http://www.ub.tu-dortmund.de/katalog/titel/645541

- Teil III – Abschnitt 4.3 – Der CMM-Ansatz (ab S.362)

CMMI vs. ISO 9001



Ist CMM Stufe 3 erreicht, ist wahrscheinlich noch einiges zu tun, um ISO 9001 zu erreichen, da einige Bereiche durch CMM nicht abgedeckt werden.

Ist ISO 9001 erfüllt, dann gibt CMM zusätzlich Hilfestellung insbesondere auf den Gebieten der Technik, Prozessdefinition und Metriken.

Literatur:

H. Balzert: **Lehrbuch der Software-Technik/2 –** Software-Management, Software-Qualitätssicherung
http://www.ub.tu-dortmund.de/katalog/titel/645541

- Teil III – Abschnitt 4.3 – Der CMM-Ansatz (ab S.362)
- Abschnitt 4.3.5 – CMM vs. ISO 9000 (S.373-376)
- Tabelle 4.3-4 (S.374)

Zusammenfassung

In diesem Abschnitt:

- **Software-Fehler** können Menschen, Maschinen, Unternehmen etc. Schaden zufügen**.**
- **„Austesten“** eines Programms in der Regel **nicht möglich**.
- **Software-Qualitätsmanagement** soll sicherstellen, dass Produkt wenig Fehler hat.
- Beinhaltet, dass
  - für Abläufe und Produkte Standards erstellt werden
  - bei diesen Abläufen auch Standards eingehalten werden.

Im nächsten Abschnitt:

- Grundlagen der **Softwareverifikation**: Fehlerarten.

Anhang

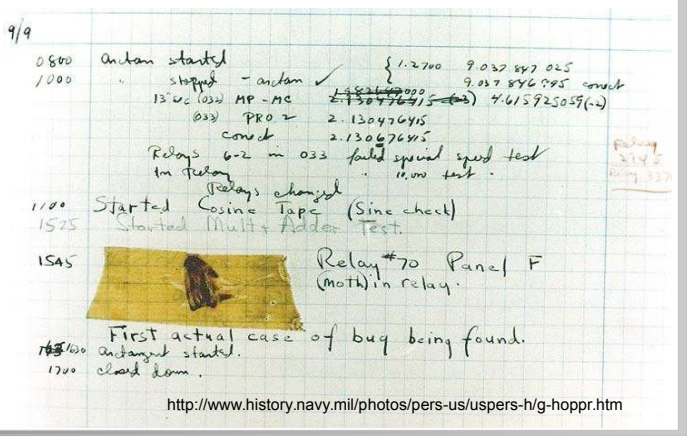
- Mehr Informationen für einige der behandelten Begriffe; zum Nachlesen zu Hause.

Der erste Software-Fehler !

Motte im Rechner Mark II **verursacht Fehler** in Relay Nr. 70, Panel F.

Mrs. Grace Murray Hopper beseitigt Fehler, dokumentiert im Log-Buch:
»First actual case of bug being found.«

- »offen«-sichtlicher Fehler.
- **Beseitigung ist einfach.**

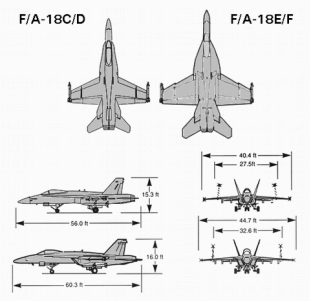


http://www.history.navy.mil/photos/pers-us/uspers-h/g-hoppr.htm

F-18 am Äquator: Dokumentationsfehler

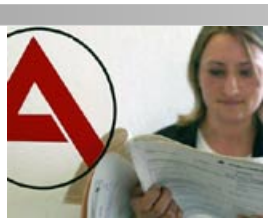
US Air Force, Programm zur Raketensteuerung

- Aus Speicherplatzmangel **keine Neuberechnung** der Flugkoordinaten.
- Nur **Änderung** des Vorzeichens.
- **Programm** unverändert in Autopiloten des Jäger F-18 **übernommen**.
- Beim Überflug des Äquator drehte sich Maschine auf den Kopf wie zuvor!
- **Glück im Unglück:** Man bemerkte dies im Simulator ...



ALG2 LL (Harz IV) Software

September 2004: Drei Wochen Verspätung
Durch anhaltende Probleme mit der Computersoftware können die Arbeitsagenturen erst drei Wochen später als geplant die Antragsdaten für das Arbeitslosengeld II flächendeckend erfassen. Ursprünglich sollte das Programm schon ab 4. Oktober 2004 zur Verfügung stehen.
Mittlerweile haben sich die Bundesagentur für Arbeit (BA) und die Software-Entwickler der Telekom-Tochter T-Systems geeinigt: Das Programm soll stufenweise eingeführt werden. "Wir sind zurzeit in der Testphase, um Fehler zu ermitteln und zu beheben. Am 4. Oktober werden wir planmäßig starten und das System stufenweise hochfahren", sagte ein Sprecher von T-Systems im Gespräch mit **wdr.de** am Freitag (10.09.04). Das Computerprogramm werde Anfang Oktober in einigen Agenturen anlaufen, damit die BA Erfahrungen sammeln kann.



In einigen Agenturen wird das Programm getestet.

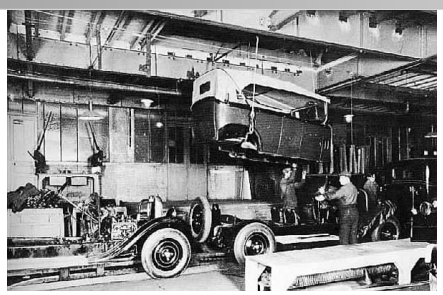
September 2005: Durch einen Fehler in der Software A2LL kommt es derzeit zu falschen Krankenkassen-Meldungen, teilt die Bundes-agentur für Arbeit (BA) mit. In mehreren hunderttausend Fällen seien Meldungen zur Krankenversicherung, also Anmeldungen, Abmeldungen, Veränderungsmitteilungen, von Arbeitslosengeld-II-Empfängern ohne Grund automatisch storniert worden.

http://www.heise.de/newsticker/meldung/62595

Wie testen andere ?

Automobil-Industrie vs. Softwaretest

- Eingangstest der Komponenten (= **Komponententest**)
- Laufende Zwischenkontrollen am Fließband (= **Integrationstest**)
- Realitätsnaher Einsatztest (= **Systemtest**)
- Probefahrt des Kunden (= **Abnahmetest**)
- Renneinsatz (= **Performanztest**, Lasttest) (Stabilität, Zuverlässigkeit, Robustheit)
- Crashtest (= **Stresstest**)



Was ist Softwarequalität?

- Grad, in dem ein System, eine Komponente oder ein Prozess die Kundenerwartungen und -bedürfnisse erfüllt.
  [nach IEEE 610: „Standard Glossary of Software Engineering Terminology“]
- Qualitätsanforderungen geben vor, welche **Qualitätsmerkmale** das Produkt in welcher **Güte** aufweisen soll (Qualitätsniveau).
- Gesamtheit aller Qualitätsmerkmale und deren geforderte Ausprägung.
- Qualitätsmerkmale beziehen sich auf Anforderungen:
  - **Funktionale Anforderungen**
    (Fachlichkeit, Funktionen, Schnittstellen, …).
  - **Nicht-funktionale Anforderungen**
    (Qualitäts- und Realisierungsanforderungen, Projektspezifische Anforderungen, …).

Äußeres Qualitätsmerkmal Funktionalität (1)

Vorhandensein von Funktionen mit festgelegten Eigenschaften. Diese Funktionen erfüllen die festgelegten oder vorausgesetzten Anforderungen.

Angemessenheit

- Merkmale von Software, die sich auf das Vorhandensein und die Eignung einer Menge von Funktionen für spezifizierte Aufgaben beziehen.
- Die Fähigkeit eines Softwareprodukts für spezifizierte Aufgaben und Ziel-setzungen der Benutzer einen geeigneten Satz Funktionen zu liefern.

Richtigkeit

- Merkmale von Software, die sich auf das Liefern der richtigen oder vereinbarten Ergebnisse oder Wirkungen beziehen.
- Die Fähigkeit eines Softwareprodukts, die richtigen oder vereinbarten Ergeb-nisse oder Wirkungen mit dem benötigten Grad an Genauigkeit zu liefern.

Äußeres Qualitätsmerkmal Funktionalität (2)

Interoperabilität

- Merkmale von Software, die sich auf ihre Eignung beziehen, mit vorgegebenen Systemen zusammenzuwirken.
- Die Fähigkeit eines Softwareprodukts, mit einer oder mehreren spezifizierten Komponenten zusammenzuwirken.

Ordnungsmäßigkeit

- Merkmale von Software, die bewirken, dass die Software anwendungsspezifische Normen oder Vereinbarungen oder gesetzliche Bestimmungen und ähnliche Vorschriften erfüllt.

Sicherheit

- Merkmale von Software, die sich auf ihre Eignung beziehen, unberechtigten Zugriff, sowohl versehentlich als auch vorsätzlich, auf Programme und Daten zu verhindern.
- Eigenschaften der Software, die sich auf die Fähigkeit beziehen, nicht autorisierte Zugriffe auf Programme oder Daten zu verhindern, unabhängig davon, ob diese versehentlich oder vorsätzlich erfolgen.

Äußeres Qualitätsmerkmal Zuverlässigkeit

Merkmale, die sich auf die Fähigkeit der Software beziehen, ihr Leistungsniveau unter festgelegten Bedingungen über einen festgelegten Zeitraum zu bewahren.

Reife

- Merkmale von Software, die sich auf die Häufigkeit von Versagen durch Fehlzustände in der Software beziehen.
- Die Fähigkeit eines Softwareprodukts, Fehlerwirkungen aufgrund von Fehlerzuständen in der Software zu vermeiden.

Fehlertoleranz

- Merkmale von Software, die sich auf ihre Eignung beziehen, ein spezifiziertes Leistungsniveau bei Software-Fehlern oder Nicht-Einhaltung ihrer spezifizierten Schnittstelle zu bewahren.
- Die Fähigkeit eines Softwareprodukts, ein spezifiziertes Leistungsniveau auch bei Fehl-funktionen oder trotz Fehleingaben (z.B. falsche Bedienung) aufrecht zu erhalten.

Wiederherstellbarkeit

- Merkmale von Software, die sich beziehen auf die Möglichkeit, bei einem Versagen ihr Leistungsniveau wiederherzustellen und die direkt betroffenen Daten wiederzugewinnen und auf die dafür benötigte Zeit und den benötigten Aufwand.
- Die Fähigkeit eines Softwareprodukts, bei einer Fehlerwirkung das spezifizierte Leistungsniveau des Systems wiederherzustellen und die direkt betroffenen Daten wiederzugewinnen.

Äußeres Qualitätsmerkmal Benutzbarkeit

Merkmale, die sich auf den zur Benutzung erforderlichen Aufwand beziehen, und auf die individuelle Bewertung einer solchen Benutzung durch eine festgelegte oder vorausgesetzte Gruppe von Benutzern.

Verständlichkeit

- Merkmale von Software, die sich auf den Aufwand für den Benutzer beziehen, das Konzept und die Anwendung zu verstehen.
- Die Fähigkeit eines Softwareprodukts, den Benutzer in die Lage zu versetzen zu verstehen, ob die Software geeignet ist, und wie sie für eine bestimmte Aufgabe und Benutzungsbedingungen brauchbar ist.

Erlernbarkeit

- Merkmale von Software, die sich auf den Aufwand für den Benutzer beziehen, die Anwendung zu erlernen.
- Die Fähigkeit eines Softwareprodukts, einem Benutzer das Erlernen der Anwendung leicht zu machen.

Bedienbarkeit

- Merkmale von Software, die sich auf den Aufwand für den Benutzer bei der Bedienung und Ablaufsteuerung beziehen.

Äußeres Qualitätsmerkmal Effizienz

Merkmale, die sich auf das Verhältnis zwischen dem Leistungsniveau der Software und dem Umfang der eingesetzten Betriebsmittel unter festgelegten Bedingungen beziehen.

- **Zeitverhalten**
  Merkmale von Software, die sich beziehen auf die Antwort- und Verarbeitungszeiten und auf den Durchsatz bei der Ausführung ihrer Funktionen.
- **Verbrauchsverhalten**
  Merkmale von Software, die sich darauf beziehen, wie viele Betriebsmittel bei der Erfüllung ihrer Funktionen benötigt werden und wie lange.

Inneres Qualitätsmerkmal Änderbarkeit

Merkmale, die sich auf den Aufwand für Durchführung vorgegebener Änderungen beziehen.

Analysierbarkeit

- Merkmale von Software, die sich auf den Aufwand beziehen, der notwendig ist, um Mängel oder Ursachen von Versagen zu diagnostizieren oder um änderungsbedürftige Teile zu bestimmen.
- Die Fähigkeit eines Softwareprodukts, die Diagnose von Mängeln oder Ursachen von Fehlerwirkungen zu ermöglichen oder änderungsbedürftige Teile zu bestimmen.

Modifizierbarkeit

- Merkmale von Software, die sich auf den Aufwand beziehen, der zur Ausführung von Verbesserungen, zur Fehlerbeseitigung oder zur Anpassung an Umgebungsänderungen notwendig ist.
- Die Fähigkeit eines Softwareprodukts, die Durchführung spezifizierter Änderungen zu ermöglichen.

Stabilität

- Merkmale von Software bezüglich des Risikos unerwarteter Wirkungen von Änderungen.
- Fähigkeit eines Softwareprodukts, unerwartete Auswirkungen von Änderungen zu vermeiden.

Testbarkeit

- Merkmale von Software, die sich auf den Aufwand beziehen, der zum Testen der geänderten Software notwendig ist.
- Die Fähigkeit eines Softwareprodukts für einen Test nach einer Änderung.

Inneres Qualitätsmerkmal Übertragbarkeit (1)

Merkmale, die sich auf die Eignung der Software beziehen, von einer Umgebung in eine andere übertragen zu werden.

Anpassbarkeit

- Merkmale von Software, die sich auf die Möglichkeit beziehen, sie an verschiedene festgelegte Umgebungen anzupassen, wenn nur Schritte unternommen oder Mittel eingesetzt werden, die für diesen Zweck für die betrachtete Software vorgesehen sind.
- Die Fähigkeit eines Softwareprodukts, dass sie auf verschiedene Laufzeitumgebungen angepasst werden kann und dabei nur die Anpassungen vorzunehmen sind, die genau diesem Zweck dienen.

Installierbarkeit

- Merkmale von Software, die sich auf den Aufwand beziehen, der zur Installierung der Software in einer festgelegten Umgebung notwendig ist.
- Die Fähigkeit eines Softwareprodukts, in einer spezifizierten Umgebung installierbar zu sein.

Inneres Qualitätsmerkmal Übertragbarkeit (2)

Konformität

- Merkmale von Software, die bewirken, dass die Software Normen oder Vereinbarungen zur Übertragbarkeit erfüllt.
- Die Fähigkeit eines Softwareprodukts, anwendungsspezifische Normen oder Vereinbarungen oder gesetzliche Bestimmungen und ähnliche Vorschriften zu erfüllen.

Austauschbarkeit

- Merkmale von Software, die sich beziehen auf die Möglichkeit, diese anstelle einer spezifizierten anderen Software in der Umgebung jener Software zu verwenden, und auf den dafür notwendigen Aufwand.
- Die Fähigkeit eines Softwareprodukts an Stelle einer anderen spezifizierten Software zum selben Zweck in der gleichen Umgebung genutzt zu werden.

Qualitätskonflikte

- Nicht alle **Qualitätsmerkmale** lassen sich gleichzeitig gleich gut erfüllen.
- **Qualitätsplan** sollte für zu entwickelnde Software wichtigste Qualitätsmerkmale herausstellen.
- Prioritäten festlegen:
  - In engster Absprache mit Auftraggebern und Anwendern.
  - **Qualitätsanforderungen:** Bestandteil der **nicht-funktionalen Anforderungen** im Pflichtenheft.
- Qualitätsplan sollte vorgeben, wie **Qualitätsbewertung** der Software ablaufen soll.

Qualitätslenkung: Konstruktive Maßnahmen

- **Fehlerverhindernde** / fehlervermeidende Prozesse definieren.
- Prüf- und **Korrekturverfahren** in Prozesse integrieren.
- Prüfergebnisse zur Verbesserung des Prozesses verwenden.
- Keine systematische, ingenieurmäßige Vorgehensweise, welche Erreichung gegebener Qualitätsanforderungen garantiert.
- Konstruktive Maßnahmen, um generelles **Qualitätsniveau** zu heben.
- **Rigorose Qualitätsprüfung** (und Behebung der festgestellten Mängel):
  - Mittel zur Sicherstellung konkreter Qualitätsanforderungen an Software, während aller Phasen der Entwicklung.

Software-Qualitätsprüfung

- Gruppe von **Qualitätsprüfern** prüft teilweise oder ganz Softwaresystem und dazugehörige Dokumentation.
- Code, Entwurf, Spezifikationen, Testentwürfe, Standards, etc. können überprüft werden.
- Um fortzufahren Software oder Dokumente als freigegeben markieren.
- Drei verschiedene Prüfungen mit verschiedenen Zielen:
  - Inspektion zur **Mängelbeseitigung** (Produkt).
  - Prüfung der **Fortschrittsbewertung** (Produkt / Abläufe).
  - Prüfung der **Qualität** (Produkt / Standards).

Prozess- und Produktqualität

- **Qualität** des Produktionsprozesses beeinflusst Qualität des entwickelten Produkts.
- Wichtig in Softwareentwicklung, weil einige **Produktqualitätseigenschaften** schwer zu beurteilen sind.
- **Komplexe Beziehung** zwischen Softwareprozess und Produktqualität.
  - Anwendung von **individuellen Fähigkeiten** und **Erfahrungen:** Wichtig in Softwareentwicklung.
  - **Externe Faktoren** (*Neuartigkeit der Anwendung oder beschleunigte Entwicklungszeitpläne*) können Produktqualität beeinträchtigen.

Literatur:
H. Balzert: **Lehrbuch der Software-Technik/2 –** Software-Management, Software-Qualitätssicherung
http://www.ub.tu-dortmund.de/katalog/titel/645541
- Teil III – Abschnitt 4 Prozessqualität (ab S.328)

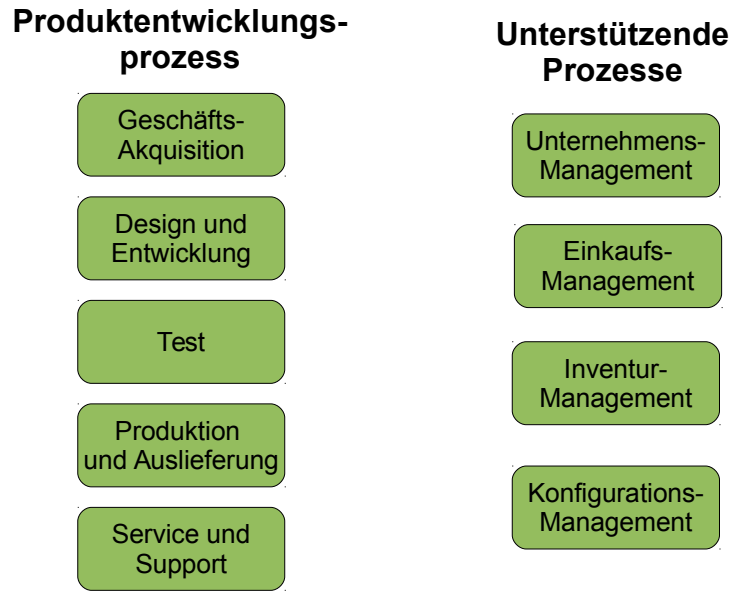
ISO 9001 (1)

- Legt **Mindestanforderungen** an Qualitätsmanagementsystem fest, die Organisation erfüllen muss.
- Um Produkte und Dienstleistungen bereitstellen zu können, die **Kundenerwartungen** und behördliche Anforderungen erfüllen.
- ISO 9001 wird insbesondere herangezogen, um Softwareprodukte entwerfen, entwickeln und pflegen.
  - Beschreibung von allgemeinen **Qualitätsmerkmalen** und Qualität von Abläufen.
  - Darlegung von organisatorischen und prozeduralen Normen, die definiert und in einem **Qualitätshandbuch** niedergeschrieben werden sollten.

ISO 9001 (2)

- **Beschreibt Modelle** zur Darlegung der Qualitätssicherung in Entwicklung, Produktion, Montage und Kundendienst.
- Legt
  - kein **Vorgehensmodell** (Phasenmodell) fest.
  - einmalig oder periodisch im Unternehmen durchzuführende und pro Projekt durchzuführende **Maßnahmen** fest.
- Verlangt
  - **Darlegung** der Phasen, Ergebnisse und der jeweiligen Qualitätssicherungsmaßnahmen (Verifizierung).
  - Maßnahmen zur Dokumentation der Projektabläufe.
- **Erfüllung der Vorgaben** durch Audits unabhängiger Zertifizierungsstelle.
- Regelmäßige **Überwachungs- bzw. Wiederholungsaudits**.
- ISO IEC 90003 (früher ISO 9000-3):
  - Richtlinie für Anwendung von ISO 9001 auf Entwicklung und Wartung von Software.

ISO 9001: Kernprozesse



Qualitätsprüfung: Analytische Maßnahmen

- Prüfung der Produkte:
  - Statische Prüfung:
    - Review
    - Statische Analyse
    - Formale Programmverifikation
    - Model Checking
  - Dynamische Prüfung:
    - Test
    - Simulation
    - Prototypen
- Prüfung der Prozesse:
  - Audits
  - Prozessbeurteilung

Literatur:

H. Balzert: **Lehrbuch der Software-Technik/2 –** Software-Management, Software-Qualitätssicherung
http://www.ub.tu-dortmund.de/katalog/titel/645541

- Teil III – Abschnitt 2.1 (S.278-284)
- Abbildung 2.1-1 (S.280)
- Abbildung 2.1-2 (S.281)