LE 10
1

Software-Qualitätssicherung

2 Qualitätssicherung
[ohne 2.3]

Prof. Dr. Helmut Balzert
Lehrstuhl für Software-Technik
Ruhr-Universität Bochum

© Helmut Balzert 1998

LE 10
2

Lernziele

- Die Prinzipien der Software-Qualitätssicherung aufzählen, erklären und begründen können.

LE 10
3

Inhalt

2.1 Qualitätsmanagement und Qualitätssicherung
2.2 Prinzipien der Software-Qualitätssicherung.

LE 10
4

2.1 Qualitätsmanagement & Qualitätssicherung

- **Qualitätsmanagement** (QM)
  - Umfaßt »alle Tätigkeiten der Gesamtführungsaufgabe, welche die Qualitätspolitik, Ziele und Verantwortungen festlegen sowie diese durch Mittel wie Qualitätsplanung, Qualitätslenkung, Qualitätssicherung und Qualitätsverbesserung im Rahmen des Qualitätsmanagementsystems verwirklichen«
    (DIN EN ISO 8402).

LE 10 5

2.1 Qualitätsmanagement & Qualitätssicherung

- **Qualitätssicherung (QS)**
  - »Alle geplanten und systematischen Tätigkeiten, die innerhalb des Qualitätsmanagementsystems verwirklicht sind, und die wie erforderlich dargelegt werden, um angemessenes Vertrauen zu schaffen, daß eine Einheit die Qualitätsforderung erfüllen wird« (DIN EN ISO 8402).
  - Analytische Maßnahmen
- **Qualitätslenkung**
  - Die »Arbeitstechniken und Tätigkeiten, die zur Erfüllung der Qualitätsforderungen angewendet werden« (DIN EN ISO 8402)
  - Konstruktive Maßnahmen.

LE 10 6

2.1 Qualitätsmanagement & Qualitätssicherung

- **Produktorientiertes Qualitätsmanagement**
  - Überprüfung von Software-Produkten und Zwischenergebnissen auf vorher festgelegte Qualitätsmerkmale
  - Bei Anwendungs-Software zusätzlich auf Gütebedingungen und Prüfbestimmungen
    - Diese können Gegenstand einer Zertifizierung sein
    - Zertifizierung
      - Bestätigung durch anerkannte (akkreditierte) Stellen, daß bestimmte Normen eingehalten wurden.

LE 10 7

2.1 Qualitätsmanagement & Qualitätssicherung

- **Prozeßorientiertes Qualitätsmanagement**
  - Erstellungsprozeß der Software
    - Methoden, Werkzeuge, Richtlinien und Standards
  - Mitarbeiter müssen ein Qualitätsbewußtsein entwickeln, das den Produktionsprozeß hin zu einer optimalen Qualität verändert
  - Das Qualitätsoptimum für alle Zeiten wird es nicht geben
  - Geht man von einem dynamischen Qualitätsoptimum aus, dann dient der Qualitätsprozeß der permanenten Adaptierung.

LE 10 8

2.1 Qualitätsmanagement & Qualitätssicherung

- **Qualitätsmanagement**
  - Kann durch konstruktive und analytische Maßnahmen erreicht werden
- **Konstruktive Qualitätsmanagementmaßnahmen**
  - Methoden, Sprachen, Werkzeuge, Richtlinien, Standards und Checklisten
  - Sollen dafür sorgen, daß das entstehende Produkt bzw. der Erstellungsprozeß à priori bestimmte Eigenschaften besitzt.

LE 10
9

2.1 Qualitätsmanagement & Qualitätssicherung

- **Produktorientierte Maßnahmen: Beispiele**
  - Ein fest vorgegebenes Gliederungsschema für ein Pflichtenheft sorgt dafür, daß alle Punkte behandelt und beschrieben werden (Richtlinie)
  - Der Einsatz der Strukturierten Analyse SA führt automatisch zu guter Lokalität
    - Alle relevanten Informationen befinden sich auf einem DIN-A4-Blatt (Methode)
  - Beim Einsatz einer Programmiersprache mit statischer Typprüfung können Typfehler während der Laufzeit nicht auftreten (Sprache)
  - Programmiersprachen ohne *goto*-Konstrukte, erzwingen automatisch lineare Programme (Sprache).

LE 10
10

2.1 Qualitätsmanagement & Qualitätssicherung

- Ein Programmierer, der alle importierten Größen von seinem Programm auf Richtigkeit überprüfen läßt, stellt sicher, daß Programmfehler wegen falscher Eingabedaten vermieden werden (defensives Programmieren) (Richtlinie)
- Die objektorientierte Software-Entwicklung stellt sicher, daß es keinen »Strukturbruch« zwischen der Definition, dem Entwurf und der Implementierung gibt (Methode)
- Die objektorientierte Software-Entwicklung unterstützt durch die Polymorphie die Wiederverwendbarkeit und durch die Vererbung die Wiederverwendung (Methode).

LE 10
11

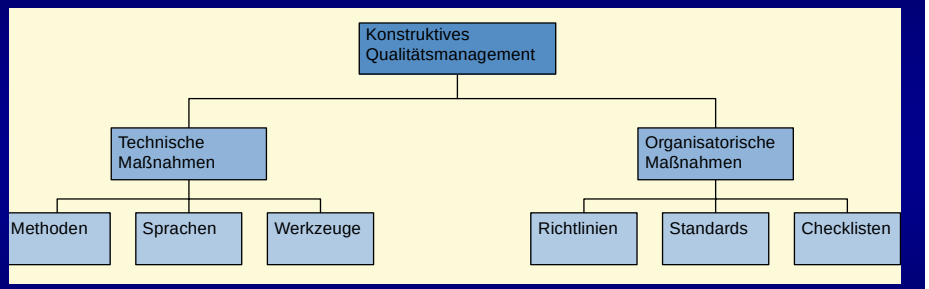
2.1 Qualitätsmanagement & Qualitätssicherung

- **Prozeßorientierte Maßnahmen: Beispiele**
  - Eine Festlegung, welche Teilprodukte mit welchem Inhalt und welchem Layout wann und von wem erstellt werden müssen, standardisiert den Entwicklungsprozeß (Richtlinie)
  - Ein Konfigurationsmanagementsystems erlaubt es, die Software-Elemente, die zu einer Konfiguration gehören, zu identifizieren (Richtlinie, Werkzeug)
  - Die Festlegung von zulässigen Zustands-übergängen) gibt einen Überblick über den Fertigungsstand einer Entwicklung (Richtlinie, Werkzeug)
  - Die Festlegung eines Vorgehensmodells für die Software-Entwicklung sorgt für Transparenz und Planbarkeit (Richtlinie, Werkzeug).

LE 10
12

2.1 Qualitätsmanagement & Qualitätssicherung

- **Maßnahmen zum konstruktiven Qualitätsmanagement**



LE 10
13

2.1 Qualitätsmanagement & Qualitätssicherung

- **Analytische Qualitätsmanagementmaßnahmen**
  - Diagnostische Maßnahmen
  - Sie bringen in das Produkt oder den Entwicklungsprozeß keine Qualität per se
  - Das existierende Qualitätsniveau wird gemessen
  - Ausmaß und Ort der Defekte können identifiziert werden
  - Das Ziel ist die Prüfung und Bewertung der Qualität der Prüfobjekte.

LE 10
14

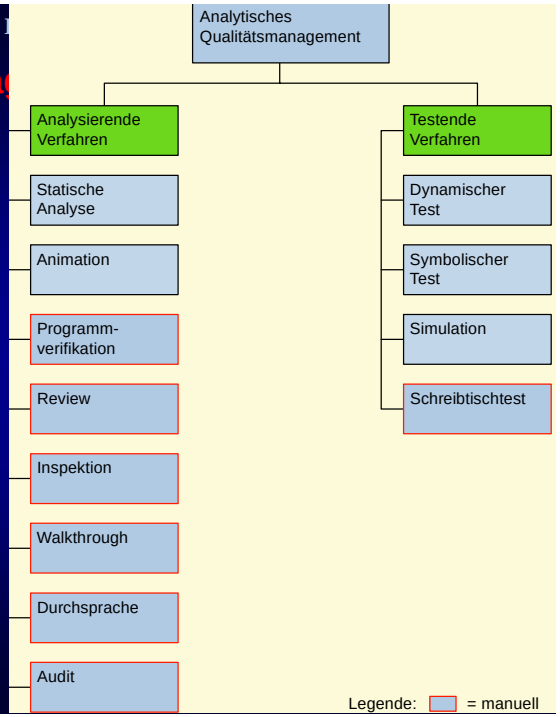
2.1 Qualitätsmanagement & Qualitätssicherung

- **Gliederung analytischer Maßnahmen**
  - Bezug der Prüfung (Produkt- oder Prozeßprüfung)
  - Automatisierungsgrad der Prüfung (manuell oder/und Einsatz von Software-Werkzeugen)
  - Nachvollziehbarkeit der Prüfung (Selbstprüfung oder Nachweisführung)
  - Einsatzbereich der Prüfung
    - Definitions-Phase
    - Entwurfs-Phase
    - Implementierungs-Phase
    - Abnahme-Phase
    - Wartungs- & Pflege-Phase.

LE 10
15

2.1 Qualitätsmanag

- **Maßnahmen zum analytischen Qualitäts-management**



LE 10
16

2.2 Prinzipien der Software-Qualitätssicherung

- **Grundsätze**
  - Prinzip der produkt- und prozeßabhängigen Qualitätszielbestimmung
  - Prinzip der quantitativen Qualitätssicherung
  - Prinzip der maximalen konstruktiven Qualitätssicherung
  - Prinzip der frühzeitigen Fehlerentdeckung und -behebung
  - Prinzip der entwicklungsbegleitenden, integrierten Qualitätssicherung
  - Prinzip der unabhängigen Qualitätssicherung.

LE 10
17

2.2 Prinzipien der Software-Qualitätssicherung

- **Prinzip der produkt- und prozeßabhängigen Qualitätszielbestimmung**
  - Praxis
    - Oft wird die Qualität weder explizit noch implizit festgelegt
  - Die gewünschte Qualität beeinflußt...
    - Kosten und Termine
    - konstruktive und analytische QM-Maßnahmen
  - Explizite und transparente Qualitätszielbestimmung vor Entwicklungsbeginn
    - Wichtig für den Auftraggeber
      - Klar, welche Qualität der Auftraggeber benötigt
    - Wichtig für den Lieferanten
      - Maßnahmen für den Entwicklungs- und Prüfprozeß.

LE 10
18

2.2 Prinzipien der Software-Qualitätssicherung

- **Prinzip der quantitativen Qualitätssicherung**
  - »Ingenieurmäßige Qualitätssicherung ist undenkbar ohne die Quantifizierung von Soll- und Istwerten«
  - Schwierigkeiten
    - Metriken sind ziel- und kontextabhängig
    - Anzahl der Variationsparameter ist um ein Vielfaches höher als bei traditionellen Produktionsprozessen
    - Kreativer Charakter vieler Aspekte der Software-Entwicklung
    - Unkontrollierte Variabilität von Entwicklungsprozessen.

LE 10
19

2.2 Prinzipien der Software-Qualitätssicherung

- Heutiger Erkenntnisstand
  - Messen ist geeignet ...
    - zum besseren Verständnis unterschiedlicher Qualitätsmerkmale
    - zur besseren Planung und Sicherung von Qualitätsmerkmalen
    - zur Verbesserung von Entwicklungsansätzen durch die Erprobung alternativer Methoden
  - Praktisch einsetzbare Methoden und Werkzeuge zur Planung und Durchführung der Datenerfassung sowie zur statistischen Auswertung und Präsentation von Meßdaten sind vorhanden
  - Es gibt keinen allgemeingültigen Satz von Metriken unabhängig von Ziel und Kontext.

LE 10
20

2.2 Prinzipien der Software-Qualitätssicherung

- **Prinzip der maximalen konstruktiven Qualitätssicherung**
  - »Vorbeugen ist besser als heilen«
  - »Fehler, die nicht gemacht werden können, brauchen auch nicht behoben werden«
  - Durch vorausblickende konstruktive Maßnahmen sollen analytische Qualitätsmanagement-maßnahmen reduziert werden
  - In vielen Fällen werden durch konstruktive Maßnahmen analytische Maßnahmen erst möglich.

LE 10
21

2.2 Prinzipien der Software-Qualitätssicherung

- Beispiele
  - Vollständige Zweigüberdeckung
    - ist ökonomisch nur dann sinnvoll zu erreichen, wenn der Codeumfang klein und die Verzweigungslogik nicht zu komplex ist
  - Flugbahnrechner der Venussonde Mariner 1
    - Das FORTRAN-Programm

```
DO 3 I = 1.3
...
3 CONTINUE
```

    - enthielt einen der teuersten Tippfehler
    - Das Verwechseln von ‘,’ mit ‘.’ führte zu einer Wertzuweisung an die Variable DO3I (Leerzeichen überliest der Compiler).

LE 10
22

2.2 Prinzipien der Software-Qualitätssicherung

- Am 22. Juli 1962 mußte die Raumsonde nach 290 Sekunden gesprengt werden, da die Flugbahn falsch berechnet worden war
- Dieser Fehler verursachte Kosten in Höhe von 18,5 Millionen Dollar

- Explizite Vereinbarung aller Variablen in einem Programm mit zugehöriger Typfestlegung
  - Ermöglicht erst folgende analytische Überprüfungen:
    - Sind alle verwendeten Variablen vereinbart?
    - Werden alle Variablen typgerecht angewandt?

LE 10
23

2.2 Prinzipien der Software-Qualitätssicherung

- Vorteile dieses Prinzips
  - + Direkte Verbesserung der Produktqualität
  - + Reduktion analytischer Maßnahmen
  - + Ermöglicht analytische Maßnahmen
  - + Vermeidung von Fehlern.

LE 10
24

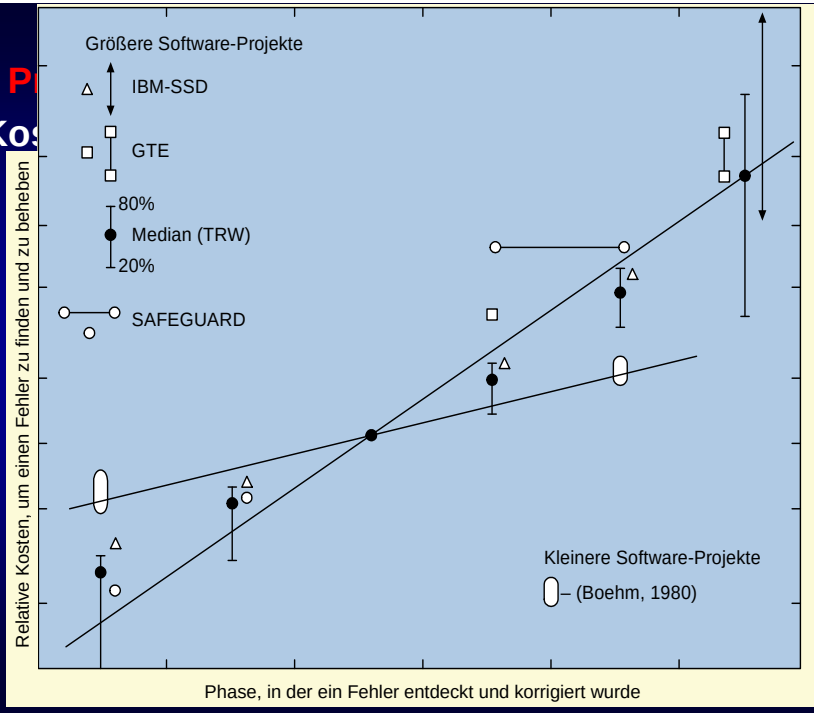
2.2 Prinzipien der Software-Qualitätssicherung

- Prinzip der frühzeitigen Fehlerentdeckung und -behebung
  - Bei Abweichungen von den Anforderungen nach der Programmierung, muß nicht nur das Programm, sondern auch der Entwurf geändert werden
  - Noch aufwendiger werden Modifikationen, wenn Fehler erst im Betrieb des fertiggestellten und freigegebenen Produktes festgestellt wurden
  - Ein Fehler ist...
    - jede Abweichung von den Anforderungen des Auftraggebers
    - jede Inkonsistenz in den Anforderungen.

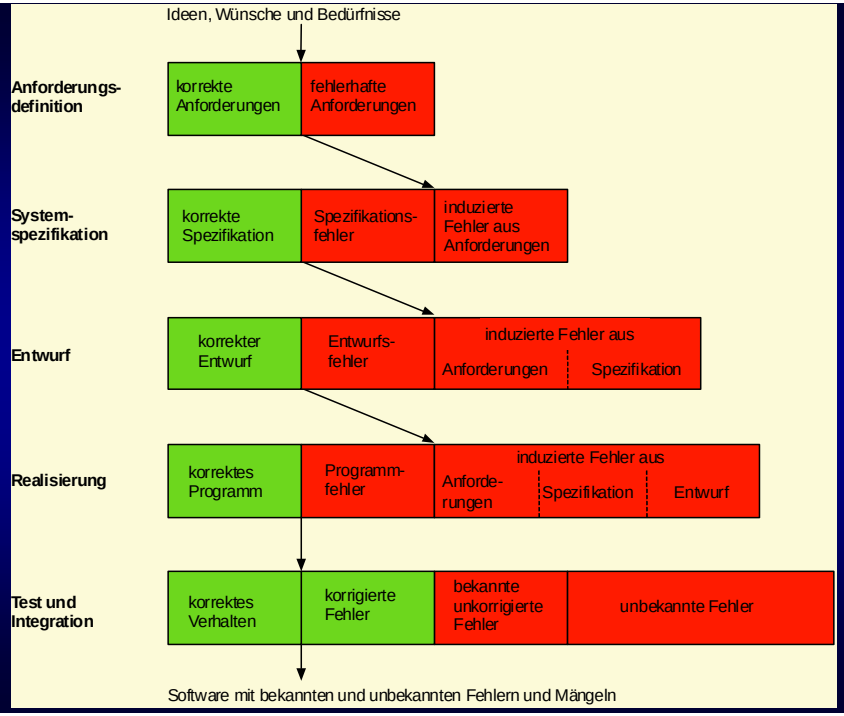
LE 10
25

2.2 P

- Ko

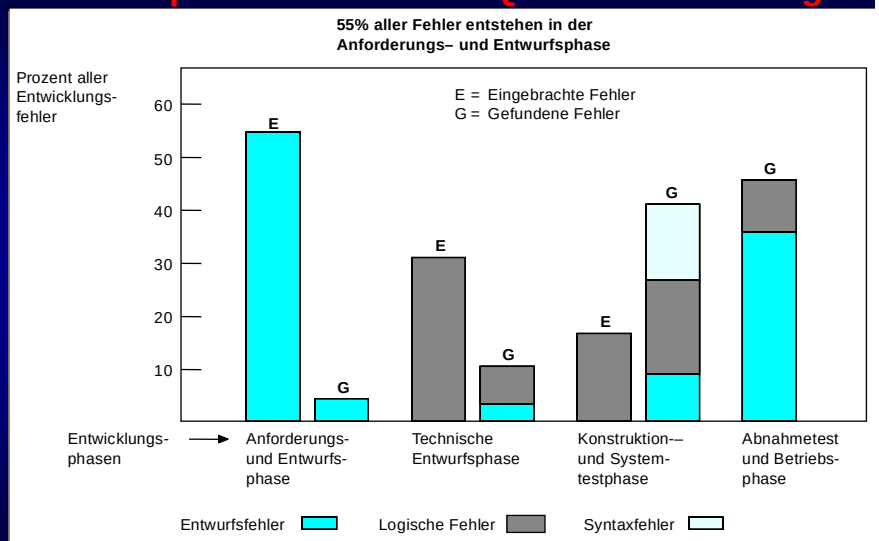


LE 10
26



LE 10
27

2.2 Prinzipien der Software-Qualitätssicherung



Fehlerbeseitigungskosten (abgeleitet von Alberts 1985)
(Zitiert nach: IEEE Software, Jan. 1985, S. 83)

LE 10
28

2.2 Prinzipien der Software-Qualitätssicherung

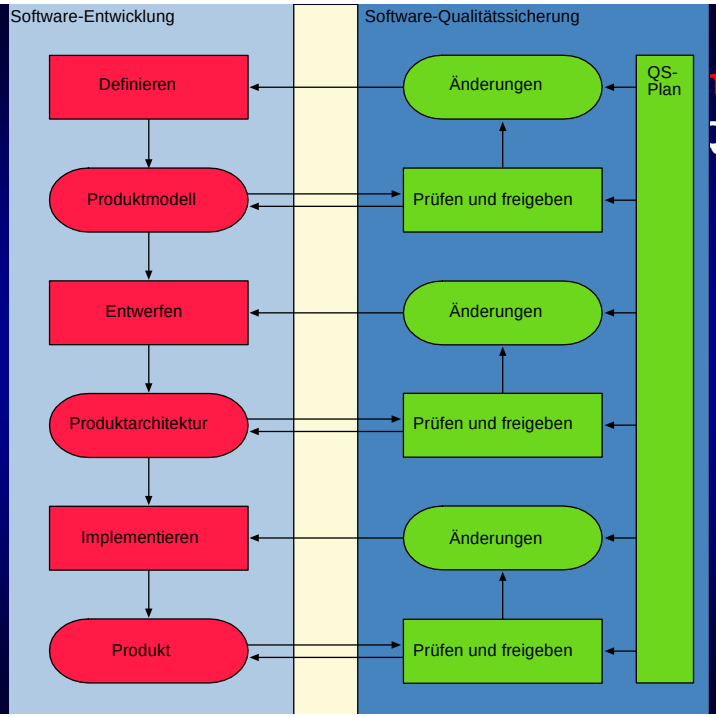
- **Vorteile dieses Prinzips**
  - + **Fehler in späteren Phasen werden vermieden**
  - + **Kosten werden reduziert**
  - + **Mit höherer Wahrscheinlichkeit werden Fehler richtig korrigiert**
  - + **Die Fehlerfortpflanzung wird reduziert.**

LE 10
29

2.2 Prinzipien der Software-Qualitätssicherung

- **Prinzip der entwicklungsbegleitenden, integrierten Qualitätssicherung**
  - Konventionelle Vorgehensmodelle
    - Analytisches Qualitätsmanagement setzt erst am Ende des Entwicklungsprozesses ein
    - Modul-, Integrations- und Systemtest
  - Prinzip der frühzeitigen Fehlerentdeckung und systematisches, vorgeplantes Qualitätsmanagement
    - Qualitätssicherung erforderlich, die die Software-Entwicklung **begleitet** und in den Entwicklungsprozeß **integriert** ist.

LE 10
30



LE 10
31

2.2 Prinzipien der Software-Qualitätssicherung

- **Vorteile dieses Prinzips**
  - \+ Einbettung der QS in das organisatorische Ablaufmodell der Software-Entwicklung
  - \+ Die QS findet jeweils zu dem Zeitpunkt statt, zu dem sie im Entwicklungsprozeß angebracht ist
  - \+ QS wird – wie in industriellen Fertigungs-organisationen – nicht als Fremdkörper empfunden, sondern gehört per se zur Software-Erstellung dazu
  - \+ Ein Teilprodukt steht der nächsten Phase erst dann zur Verfügung, wenn eine bestimmte Qualität sichergestellt ist
  - \+ Das Qualitätsniveau ist zu jedem Zeitpunkt sichtbar
  - \+ Realistische Beurteilung des Entwicklungsfortschritts ist möglich.

LE 10
32

2.2 Prinzipien der Software-Qualitätssicherung

- **Prinzip der unabhängigen Qualitätssicherung**
  - *»...testing is a **destructive** process, even a sadistic process,...«*
  - Derjenige, der ein Produkt definiert, entwirft und implementiert, ist **am schlechtesten** dazu geeignet, die Ergebnisse seiner Tätigkeit destruktiv zu betrachten
  - Qualitätsmanagement für das Teilprodukt X sollte **nicht** vom Entwickler des Teilprodukts X selbst vorgenommen werden
  - Entwickler soll aber andererseits eigene Aufgaben **nicht** an die Qualitätssicherung abschieben.

LE 10
33

2.2 Prinzipien der Software-Qualitätssicherung

- 3 Möglichkeiten für die Personalausstattung der QS
  - Personal wird für die QS eingestellt und arbeitet nur in der QS
  - Jeder Mitarbeiter rotiert in festgelegten Abständen zwischen QS und Entwicklung
  - Jeder Mitarbeiter arbeitet sowohl an einer Entwicklung als auch an der QS einer oder mehrerer anderer Entwicklungen mit.

LE 10
34

2.2 Prinzipien der Software-Qualitätssicherung

- In der Praxis häufig: sowohl als auch
  - Beispielsweise arbeitet ein Mitarbeiter zu 80% an einer Software-Entwicklung mit
  - 20% seiner Zeit verwendet er dazu, die Qualität anderer Entwicklungen zu überprüfen

  + Flexibler Personaleinsatz möglich
  + Kein Overhead für die QS

  – Die Vermischung von Entwicklungsarbeit und QS kann dazu führen, daß keine Arbeit mehr richtig gemacht wird
  – Es ist ein dauernder Wechsel zwischen unterschiedlichen Tätigkeiten nötig.

LE 10
35

2.2 Prinzipien der Software-Qualitätssicherung

- Verantwortung der Entwicklung vs. Verantwortung der QS
  - 1. Alternative: Die Entwicklung erstellt die Produkte
    - QS ist für die Überprüfung zuständig
    - Fehler werden der Entwicklung gemeldet und von ihr behoben
    - Beispiel
      - Ein Programmierer bringt sein Programm durch den Compilerlauf
      - Das Finden von Fehlern ist Aufgabe der QS
    + Der Entwickler kann sich auf die konstruktiven Aspekte konzentrieren
    – Allerdings sinkt die Sorgfalt der Entwickler
      - Motto: Laß die anderen meine Fehler finden.

LE 10
36

2.2 Prinzipien der Software-Qualitätssicherung

  - 2. Alternative: Entwicklung ist für einen definierten Qualitätszustand ihrer Produkte selbst zuständig
    - Erst wenn dieser Grad erreicht ist, übernimmt die QS die weitere Überprüfung
    - Beispiel:
      - Der Programmierer muß eine Zweigüberdeckung von 80% nachweisen
    + Klar definierte, transparente Verantwortlichkeiten
    + Der Entwickler muß sein eigenes Produkt überprüfen
    + Die Eigenverantwortlichkeit der Entwickler wird gefördert
    – Es müssen meßbare Qualitätsstufen definiert sein.

LE 10
37

2.2 Prinzipien der Software-Qualitätssicherung

- Quantitative QS
  - Mitarbeiter der QS können mehr und mehr von Aktivitäten der Qualitätsprüfung entlastet werden, da Maße die Überprüfung der ordnungsgemäßen Durchführung der QS-Maßnahmen erleichtern
  - Erfahrungen im Umgang mit Maßen und Gefühl für Interpretationsmöglichkeiten von Meßergebnissen sind erforderlich
  - Zusätzlich folgende Aktivitäten erforderlich:
    - Sammlung von Daten (Maßen)
    - Validierung dieser Daten
    - Einrichtung einer quantitativ orientierten Entwicklungsdatenbank.

LE 10
38

2.2 Prinzipien der Software-Qualitätssicherung

- Ziel
  - Unabhängige, eigenständige organisatorische Einheit »Qualitätssicherung«
  - Kritische Mitarbeiteranzahl muß erreicht werden
    - Personalanteil der QS muß größer als 15% der Entwicklungskapazität sein
- Vorteile einer unabhängigen QS
  - + Objektive, unabhängige Qualitätssicherung
  - + Das Wissen, daß das eigene Produkt noch von der QS überprüft wird, hat eine heilsame Wirkung auf die Entwicklung und verbessert die Qualität
  - + Qualitätsvergleiche über mehrere Produkte hinweg sind möglich.

LE 10
39

- Danke!
- Aufgaben
- Diese Präsentation bzw. Teile dieser Präsentation enthalten Inhalte und Grafiken des Lehrbuchs der Software-Technik (Band 2) von Helmut Balzert, Spektrum Akademischer Verlag, Heidelberg 1998