Software Engineering I

Prof. Dr. Martin Glinz

Kapitel 9

Qualitätsmanagement

Universität Zürich
Institut für Informatik

Grundlagen und Definitionen – 1

„... liefern wir Ihnen termingerecht 50000 Programmzeilen Schrott.“

➪ Qualität ist wichtig.

Was ist Qualität?

- **Qualität** (quality) – der Grad, in dem ein Satz inhärenter Merkmale Anforderungen erfüllt. (ISO 9000:2000)
- **Anforderung** (requirement) – ein Erfordernis oder eine Erwartung, das oder die festgelegt, üblicherweise vorausgesetzt oder verpflichtend ist.
- **Inhärentes Merkmal** (inherent characteristic) – eine kennzeichnende Eigenschaft einer Einheit (Produkt, Dienstleistung, Prozess, System, Person, Organisation, etc.), welche diese aus sich selbst heraus hat und die ihr nicht explizit zugeordnet ist.

Grundlagen und Definitionen – 2

- Qualität ist Zielerfüllung
- Ziele (Anforderungen) explizit festgelegt oder implizit durch gemeinsame Vorstellungen der Beteiligten gegeben
- Kein absolutes Maß für die Güte einer Einheit
- Eine Auffassung von Qualität als reine Zweckeignung oder Kundenzufriedenheit greift zu kurz

Grundlagen und Definitionen – 3

Qualitätsmanagement (quality management) – aufeinander abgestimmte Tätigkeiten zum Leiten und Lenken einer Organisation bezüglich Qualität.

Leiten und Lenken bezüglich Qualität umfassen üblicherweise das Festlegen der Qualitätspolitik und der Qualitätsziele, die Qualitätsplanung, die Qualitätslenkung, die Qualitätssicherung und die Qualitätsverbesserung. (ISO 9000:2000)

Qualitätsmanagementsystem, QM-System (quality management system) – Managementsystem zum Leiten und Lenken einer Organisation bezüglich der Qualität. (ISO 9000:2000)

Anmerkungen zur Terminologie

- Historischer Name für Qualitätsmanagement: «Qualitätssicherung» (quality assurance) mit zwei Bedeutungen (bis ca. 1995):
  - umfassend: Management von Qualität
  - eng: Sicherstellung und Darlegung von Qualität
- Terminologie heute:
  - Umfassender Begriff: Qualitätsmanagement
  - Bedeutung von «Qualitätssicherung» beschränkt auf Maßnahmen zur Schaffung von Vertrauen, dass die Qualitätsanforderungen erfüllt werden
- Aber: «Qualitätssicherung» wird häufig noch im alten, umfassenden Sinn gebraucht

Warum Software-Qualitätsmanagement – 1

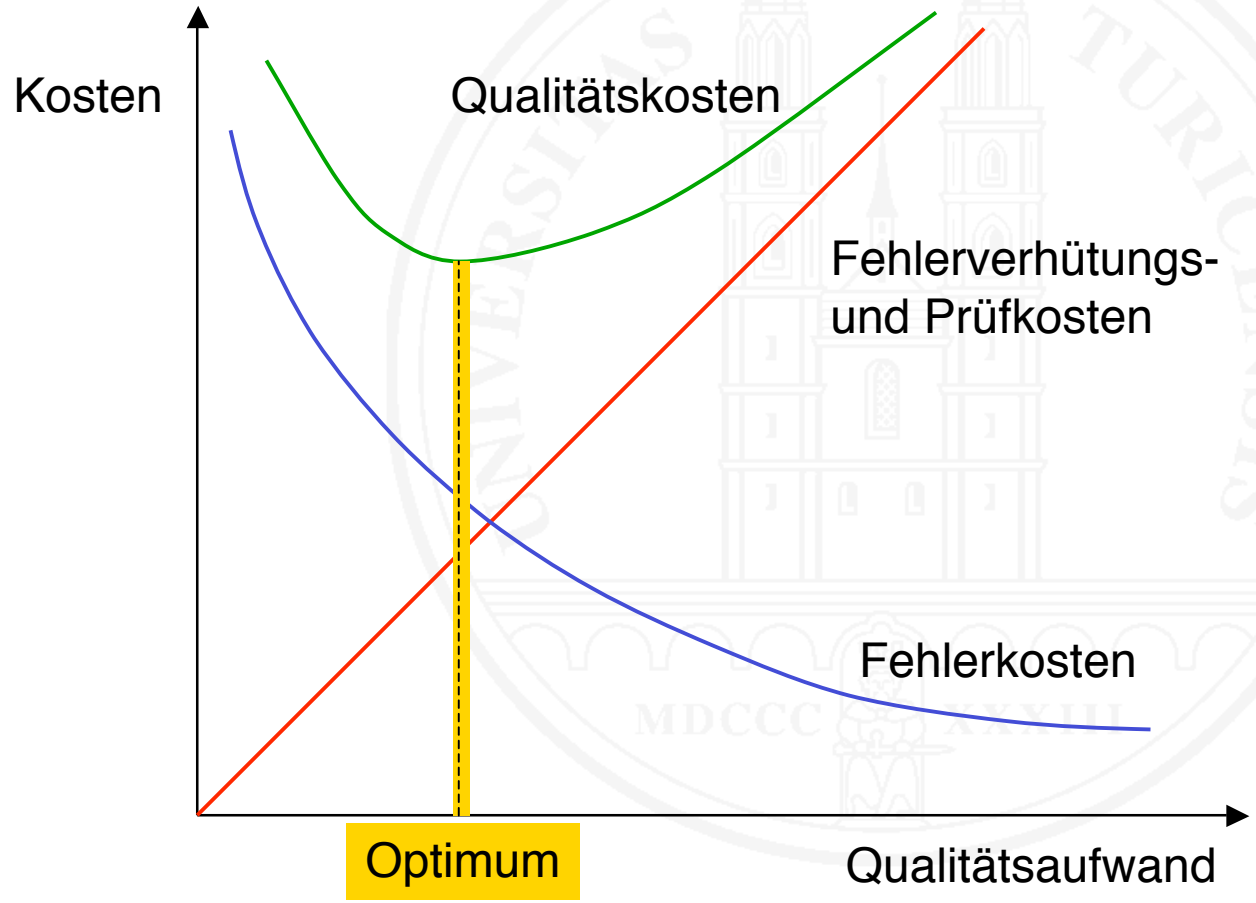
- Handwerk, historisch
  - Qualitätsbewusstsein, tradierte Standards, direkte Rückkopplung
  - ⇨ Kein explizites Qualitätsmanagement erforderlich
- Massenproduktion
  - Arbeitsteilige Produktion, kaum Rückkopplung, große Stückzahlen
  - ⇨ Explizite Maßnahmen zur Vermeidung von Pfusch und Ausschuss notwendig
  - Mittel: Messung und Lenkung von Qualität mit statistischen Verfahren
  - Verfahren: Statistische Qualitätsanforderungen an die Produkte ⇒ Anforderungen an den Produktionsprozess ⇒ Messen und statistische Auswertung ⇒ Rückkopplung der Prüfergebnisse auf den Produktionsprozess ⇒ Produktqualität

Warum Software-Qualitätsmanagement – 2

- Entwicklung von Produkten
  - Erst bei großen, komplexen Entwicklungsvorhaben notwendig
  - Problem: keine großen Stückzahlen: was messen? – wie messen? statistische Verfahren?
  - ⇨ vorwiegend Individualprüfung, Rückkopplung auf Prozess schwierig
- Software-Qualitätsmanagement: Besonderheiten
  - nur Entwicklung, keine Produktion
  - keine tradierten Standards
  - immateriell: schwierig zu messen und zu prüfen
  - Rückkopplung wird nur ansatzweise beherrscht

Qualitätskosten

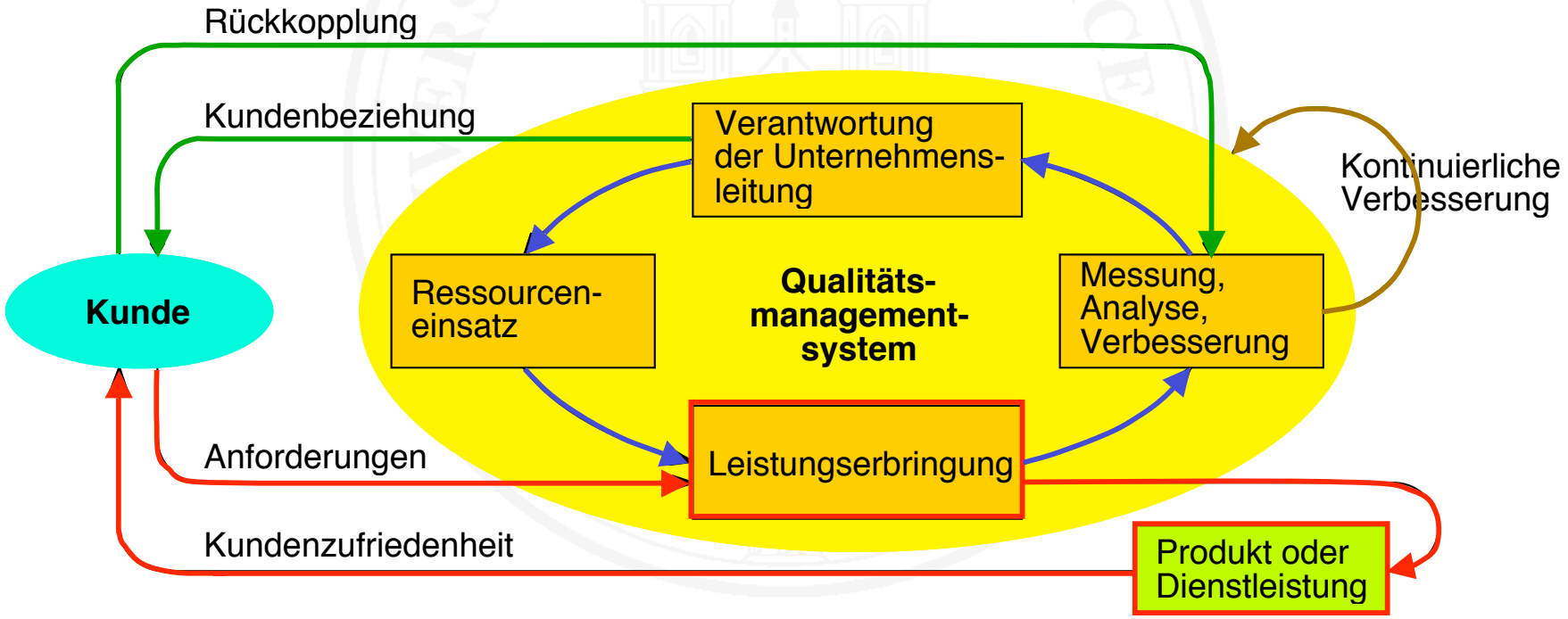
Qualität ist wirtschaftlich



nach Frühauf, Ludewig, Sandmayr (1988/2000)

Das Qualitätsmanagementsystem – Prinzipien

- Orientiert an
  - Selbstverantwortung aller Beteiligten
  - Kundenzufriedenheit
- Prozessorientiert, systemischer Ansatz zur Realisierung

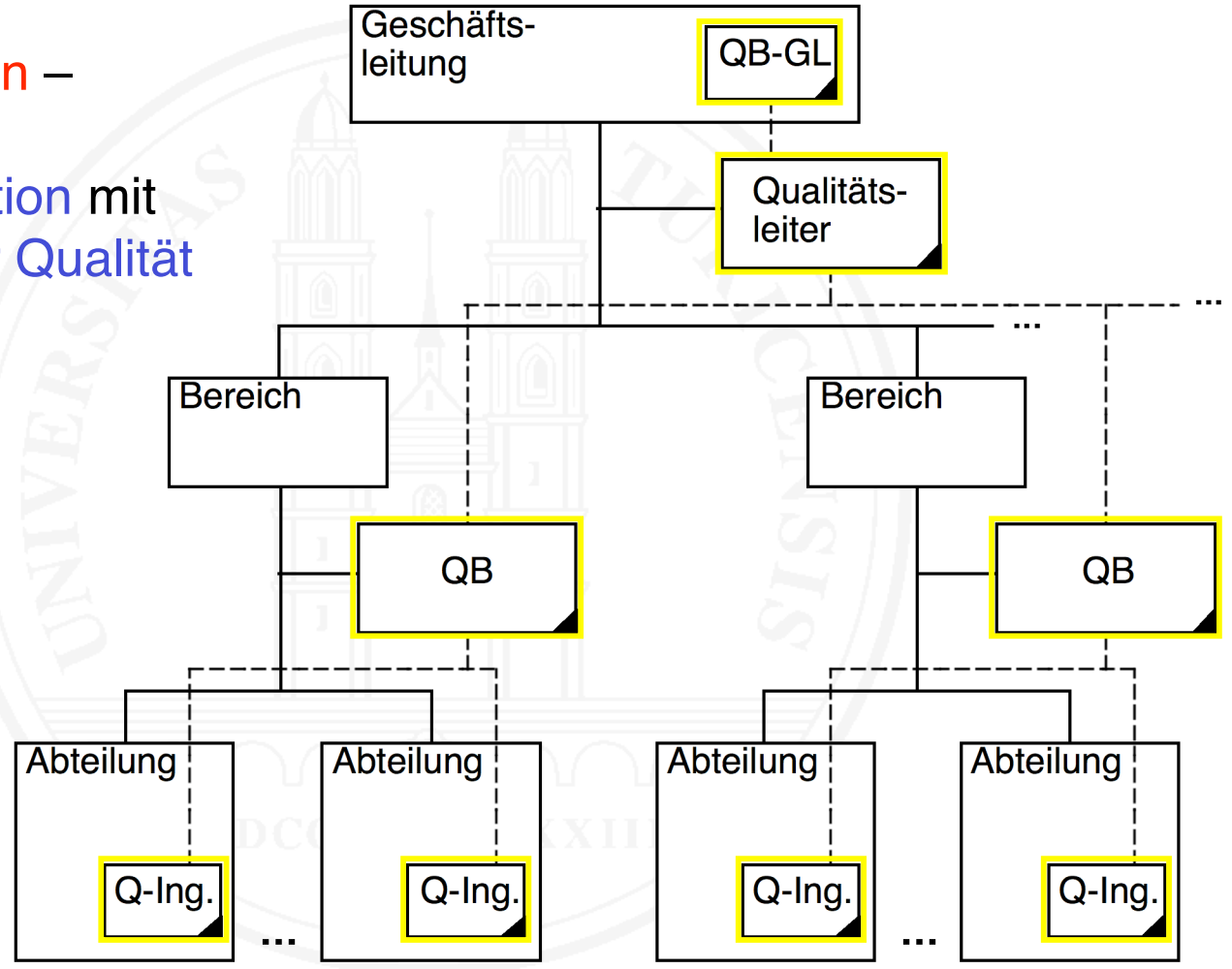


Sechs Grundsätze des Qualitätsmanagements

1. Qualität muss erzeugt werden, sie kann nicht erprüft werden
2. Qualität bezieht sich immer auf Produkte und auf Prozesse
3. Qualitätsverantwortung ist untrennbar verbunden mit Sach-, Termin- und Kostenverantwortung
4. Das Qualitätswesen erbringt Dienstleistungen und ist verantwortlich für die Ermittlung (Messung) der Qualität
5. Das Qualitätswesen muss einen unabhängigen Berichterstattungspfad haben, der bis zur Geschäftsleitung geht
6. Die Entwickler müssen über die Qualität ihrer Arbeit orientiert werden

Die Aufbauorganisation

Das Qualitätswesen –
eine
Sekundärorganisation mit
den Fachleuten für Qualität



Q-Ing.: Qualitätsingenieur QB: Qualitätsbeauftragter

Die Ablauforganisation

- Das Qualitätsmanagementsystem regelt alle qualitätsrelevanten
  - Kompetenzen
  - Verantwortlichkeiten
  - Beziehungen
- Qualitätsaufgaben in die Unternehmensprozesse integriert
- Möglichst wenig Qualitätsaufgaben separat geregelt

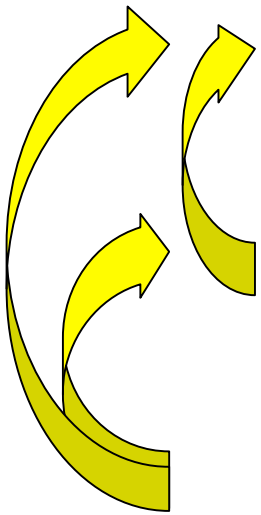
Mini-Übung 9.1 (vgl. Aufgabe 9.1 im Skript)

Sie arbeiten an der Entwicklung eines Softwareprodukts mit. Wer ist für die Qualität dieses Produkts verantwortlich? Kreuzen Sie an. Höchstens eine Antwort ist richtig.

- ☐ ausschließlich die beteiligten Entwicklerinnen und Entwickler
- ☐ sicher nicht ich
- ☐ das Management
- ☐ die Projektleiterin
- ☐ der Qualitätsingenieur der Abteilung
- ☐ die Qualitätsbeauftragte der Geschäftsleitung
- ☐ derjenige, der die Spezifikation unterschrieben hat

- ☒ ich

Qualitätsmanagement-Verfahren – 1



Qualitätsplanung

Definition der Qualitätsziele: Das wollen wir erreichen!

Qualitätslenkung

Konstruktive Maßnahmen: So müssen wir arbeiten!

Qualitätsprüfung

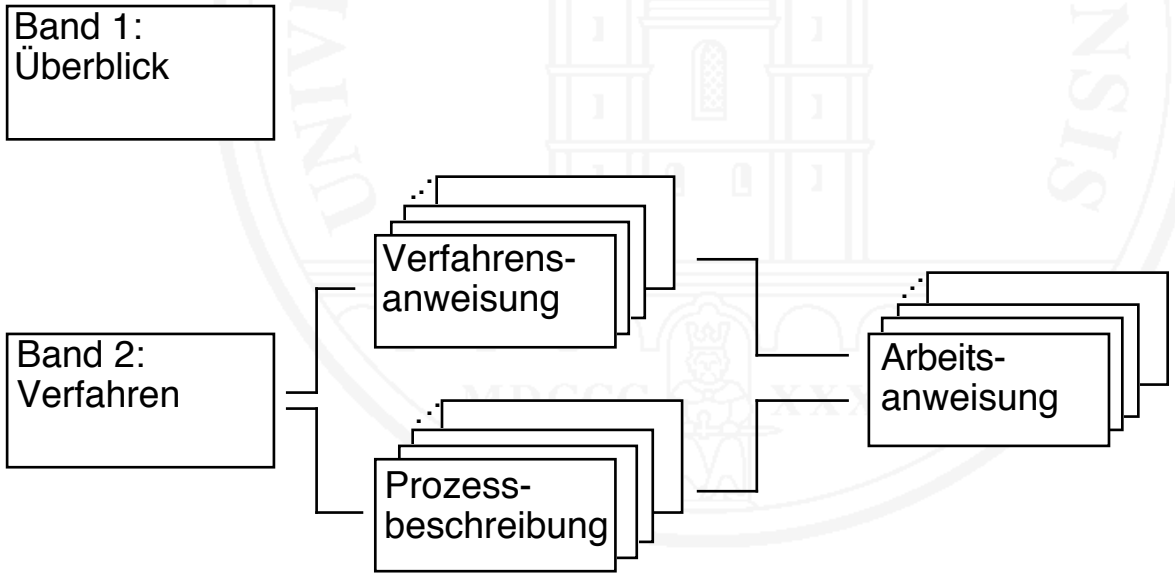
Analytische Maßnahmen: Haben wir die Ziele erreicht?
Haben wir richtig gearbeitet?

Qualitätsmanagement-Verfahren – 2

- Qualitätsplanung
  - Im Allgemeinen: Aufbau und Dokumentation des QM-Systems, allgemeine Qualitätsziele
  - Im Speziellen: Festlegung der Qualitätsziele für individuelle Projekte
- Qualitätslenkung
  - Im Allgemeinen:
    - Methoden, Sprachen, Werkzeuge
    - Ausbildung
    - Vereinheitlichung der Arbeitsweise
  - Im Speziellen: Maßnahmen der Projektführung zur Erreichung der geplanten Qualität
- Qualitätsprüfung: folgt später

Dokumentation des Qualitätsmanagements – 1

- Qualitäts-Handbuch: beschreibt das Qualitätsmanagementsystem
  - Band 1: Qualitätsmanagement-Organisation, Überblick über die Qualitätsmanagement-Maßnahmen.
  - Band 2 (vertraulich): Handbuch der Qualitätsmanagement-Verfahren



Dokumentation des Qualitätsmanagements – 2

- Software-Qualitätsplan: Vorgaben betreffend Qualität für die Entwicklung von Produkten
  - In der Regel existiert ein allgemeiner Rahmenplan
  - Rahmenplan wird bei Bedarf projektspezifisch ergänzt
  - Manchmal auch vom Kunden vorgegeben

- Software-Entwicklungshandbuch: detaillierte Entwicklungsrichtlinien und Ausführungsbestimmungen
  - Software-spezifisch
  - Entlastet das Qualitätshandbuch von Software-Details

Qualitätssicherung

Die bloße Existenz eines Qualitätsmanagementsystems genügt nicht.

Qualitätssicherung (quality assurance) – Darlegung des Qualitäts-managements, d.h. alle Tätigkeiten zur Schaffung von Vertrauen, dass die Qualitätsanforderungen erfüllt werden

Überprüfung durch Audits

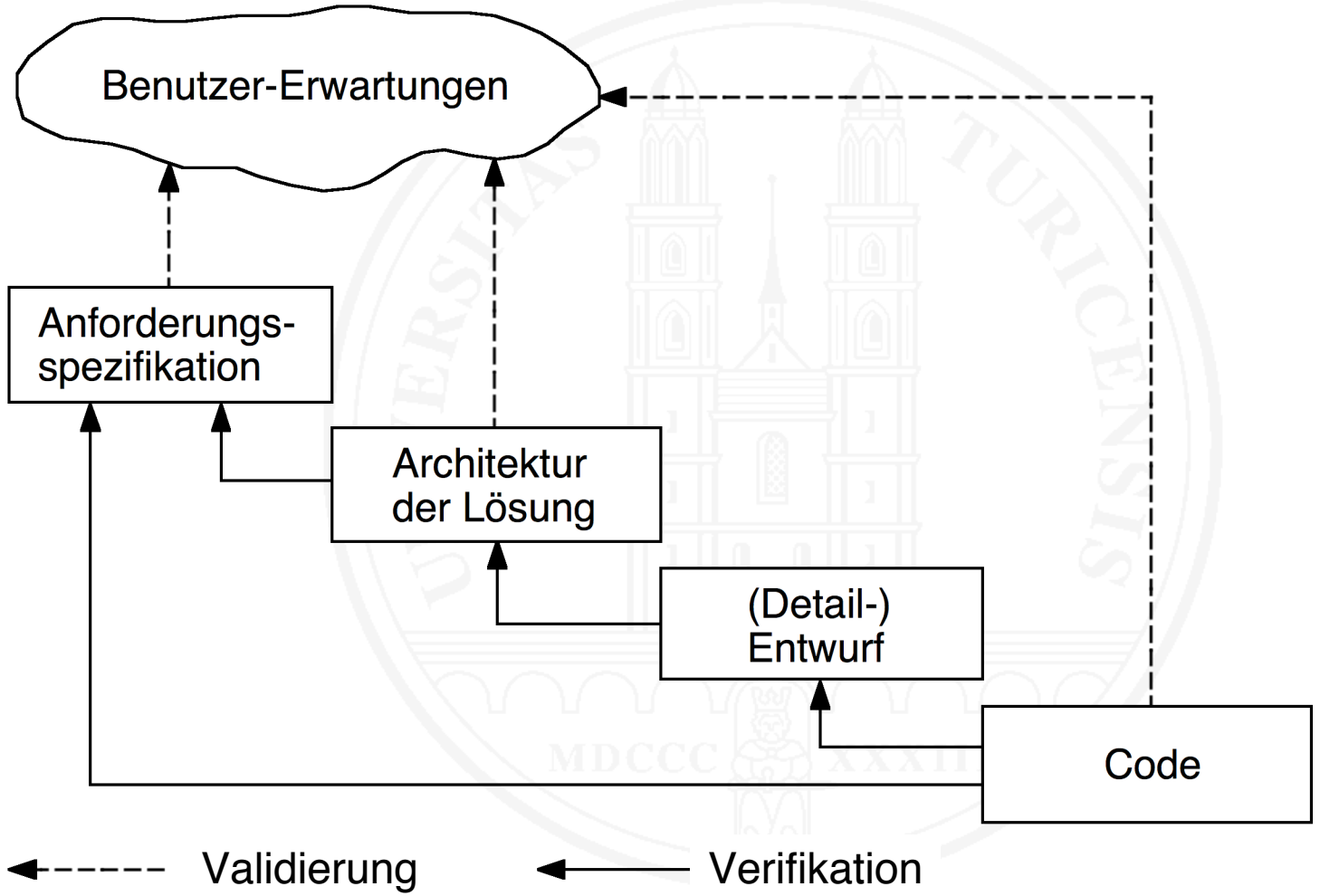
- Interner Audit: durch das Qualitätswesen
- Externer Audit: Überprüfung durch eine externe, auf Qualitätsmanagement spezialisierte Organisation
  - auf Veranlassung eines Kunden
  - zur Zertifizierung (nach ISO 9000)

Qualitätsprüfung

- Entsprechen die Arbeitsergebnisse den Vorgaben und Erwartungen?
- Prüfung am Schluss kann zu spät (weil zu teuer) sein
- ➪ Qualitätsprüfung ist eine permanente, die Entwicklung begleitende Aufgabe

- Zwei Arten der Überprüfung:
  - Erwartungen erfüllt? Das Richtige getan? ➪ Validierung
  - Vorgaben eingehalten? Richtig gearbeitet? ➪ Verifikation

Validierung und Verifikation – 1



Validierung und Verifikation – 2

Validierung (validation) – Der Prozess der Beurteilung eines Systems oder einer Komponente während oder am Ende des Entwicklungsprozesses, mit dem Ziel, festzustellen, ob die spezifizierten Anforderungen erfüllt sind.

Verifikation (verification) – 1. Der Prozess der Beurteilung eines Systems oder einer Komponente mit dem Ziel, festzustellen, ob die Resultate einer gegebenen Entwicklungsphase den Vorgaben für diese Phase entsprechen. 2. der formale Beweis der Korrektheit eines Programms.

Prüfverfahren

- Review – Eine formell organisierte Überprüfung eines Arbeitsergebnisses durch eine Gruppe von Gutachtern
- Test – Die Überprüfung eines Programms, ob dieses bei gegebenen Eingaben die erwarteten Resultate liefert
- Prototyp – Eine Vorab-Realisierung eines kritischen Systemteils

Seltener:

- Simulation – Modellierung eines bestimmter Aspekt eines Systems zur Überprüfung seines Verhaltens

In Spezialfällen:

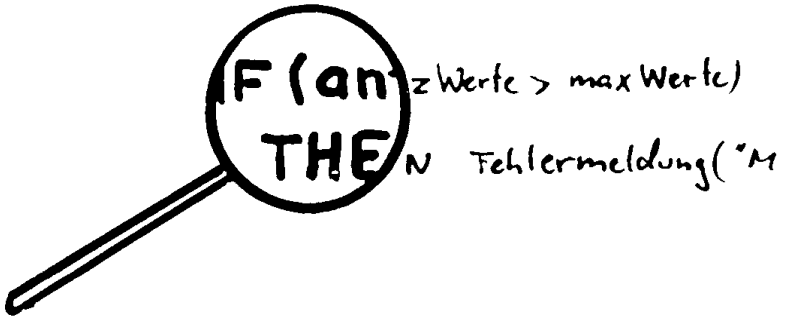
- Formale Verifikation – Mathematischer Korrektheitsbeweis
- Model checking – Überprüfung der Gültigkeit wichtiger Eigenschaften durch systematisches, automatisiertes Erproben

Reviews

Review – Eine formell organisierte Zusammenkunft von Personen zur inhaltlichen oder formellen Überprüfung eines Produktteils (Dokument, Programmstück, etc.) nach vorgegebenen Prüfkriterien und -listen.

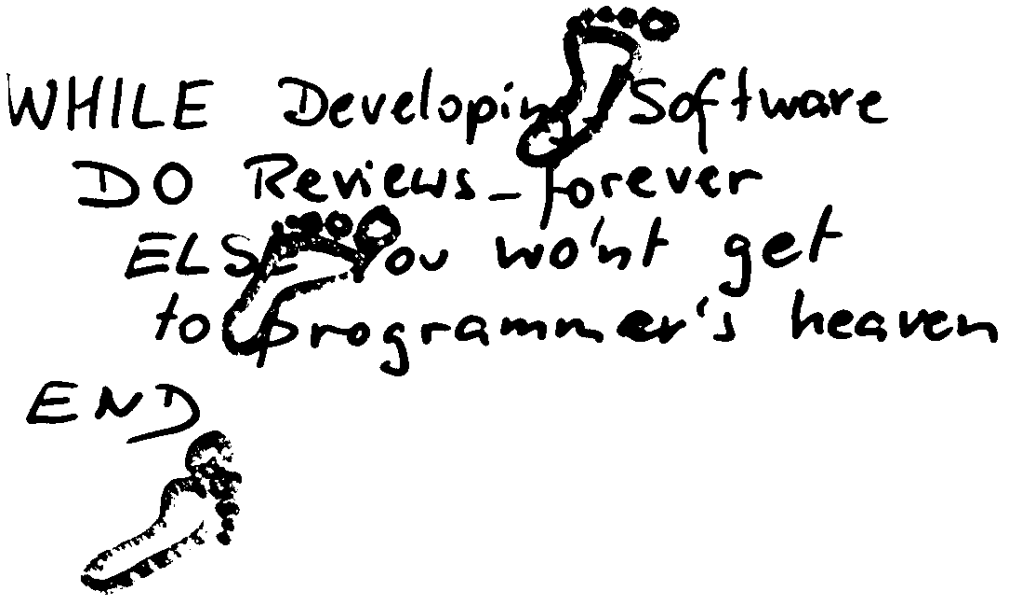
- Ziele eines Reviews:
  - Aufzeigen der Schwachstellen und Mängel des Prüflings
  - Bestätigen der Stärken des Prüflings
  - Beurteilung der Qualität
- Abgrenzung: keine Reviews im hier betrachteten Sinn sind:
  - informelle Prüfungen, z.B. Durchlesen durch Kollegen
  - Management-Reviews zur Überprüfung von Kosten und Terminen

Review-Formen – 1: Inspektion



- Prüfling wird von Gutachtern nach vorgegebenen Prüfkriterien systematisch Zeile für Zeile inspiziert
- Setzt voraus, dass Gutachter das zu prüfende Material ohne Hilfe lesen und verstehen können
- Wirksamste Reviewtechnik
- ➪ Wo immer möglich: Reviews als Inspektionen durchführen
- Nachstehende Ausführungen gelten primär für Inspektionen

Review-Formen – 2: Walkthrough



- Autor geht Prüfling mit Gutachtern durch, die Darstellung wird gemeinsam nachvollzogen
- Gutachter haken bei allen Mängeln und Problemen ein
- Auch einsetzbar, wenn Gutachter das zu prüfende Produkt nicht ohne Hilfe lesen können

Review-Material

- Prüfunterlagen
  - Der zu prüfende Gegenstand in lesbarer Form
  - ➪ Jedes von Menschen lesbare Artefakt ist reviewbar

- Referenzunterlagen
  - sachlich-inhaltliche Vorgaben
  - vorgeschriebene Standards
  - Prüflisten (Checklisten)

Warum Reviews?

- Fehler finden
  - Inspektionen finden die meisten Fehler
  - Reviewen ist billiger als testen:
    - weniger Vorbereitungs- und Durchführungsaufwand
    - Findet Fehlerursachen, nicht nur Symptome
    - Nicht jede Software-Einheit ist testbar, aber jede ist reviewbar
- Besser werden
  - Richtiges bestätigen / beibehalten / verstärken
  - Arbeitsweise vereinheitlichen
  - Ergebnisse auswerten ➪ Prozesse / Qualität lenken
  - Aus Schwach- und Starkstellen lernen: Wissenstransfer von den guten zu den schlechten Software-Entwicklern
- Reviews sind wirtschaftlich

Durchführung eines Reviews (Inspektion)

- Planung
  Termine einplanen – Aufwand budgetieren – Teilnehmer einladen – Material verteilen
- Vorbereitung
  Teilnehmer bereiten sich individuell vor – inspizieren den Prüfling – notieren Befunde
- Sitzung
  Moderiertes Zusammentragen und Bewerten der Befunde – Erstellen des Review-Berichts
- Überarbeitung und Nachkontrolle (nach dem Review)
  Entscheidung über Änderungen – Durchführen der Änderungen – Nachkontrolle durch Projektverantwortlichen oder neues Review

Rollen der Beteiligten (Inspektion)

- Moderator
  organisiert und leitet
- Gutachter
  prüft, nennt Befunde, bewertet, verhält sich positiv und kooperativ
- Schreiber
  protokolliert für alle Beteiligten sichtbar
- Autor
  hört zu, verhält sich passiv

Die Qualitätsverantwortung bei Reviews

- Qualität des Reviews → Review-Teilnehmer
- Qualität des Produkts → Hersteller und deren Chefs

⇨ Qualitätsverantwortung und Sach-/Kosten-/Terminverantwortung gehören immer zusammen!

Der Review-Bericht

- Formblätter verwenden
- Liste der Befunde entsteht öffentlich während der Sitzung (Datenprojektion, Folien auf Hellraumprojektor, kopierbare Tafel)
- Bericht am Ende des Reviews sofort fertigstellen
- Alle unterschreiben

Review-Bericht: Deckblatt – 1

Review-Bericht	Review-Nr.: Seite _____ von _____ Datum: Zeit von: bis:
Zu prüfendes Produkt	
Nummer / Version	Titel
Referenzunterlagen	
Nummer / Version	Titel
Bewertung	❑ akzeptiert (kein neues Review erforderlich) ❑ nicht akzeptiert (neues Review erforderlich) ❑ Review nicht beendet ❑ wie es ist ❑ kleine Änderungen ❑ große Änderungen ❑ Überarbeitung
Review-Team	

Review-Bericht: Deckblatt – 2

	Review erforderlich) ❑ Review nicht beendet		❑ Überarbeitung
Review-Team			
Name	Funktion	Vorbereitungszeit	Unterschrift
Freigabe (nach Nachkontrolle)			
Name	Datum		Unterschrift

Review-Bericht: Befundliste

Review Nr.: Review-Bericht Seite _____ von _____

Liste der Befunde

❑ im zu prüfenden Produkt
❑ in den Referenzunterlagen

Nr.	Ort	Befund	Gewicht

Review-Regeln – 1

- Material rechtzeitig vor Sitzung verteilt
- Alle Teilnehmer rechtzeitig eingeladen
- Alle Gutachter kommen vorbereitet (unverzichtbar bei Inspektionen!)
- 3 bis 7 Beteiligte
- Sitzungsdauer maximal 2 h
- Nicht mehr Material verteilen, als in Sitzung zu bewältigen ist
- Probleme nur nennen, nicht lösen
- «Dritte Stunde» nach Review-Sitzung zur Diskussion von Problemlösungen

Review-Regeln – 2

- Positives und Negatives nennen
- Keine Stilfragen diskutieren
- Produkt bewerten, nicht Produzenten
- Review-Berichte niemals zur schematischen Bewertung von Mitarbeitern verwenden
- Anhand von Standards und Prüflisten bewerten
- Fehler in Referenzunterlagen ebenfalls erheben und in separater Befundliste notieren

Mini-Übung 9.2 (Aufgabe 9.2 im Skript) – 1

Sie sind Projektleiter in einem Software-Entwicklungsprojekt. Soeben haben Sie den Review-Bericht über das Lösungskonzept der zu entwickelnden Software erhalten. Der Bericht weist neben einer Reihe leichter Fehler drei schwere Fehler und eine als kritisch eingestufte Auslassung aus. Sie sind gegenüber dem Zeitplan drei Wochen in Verzug und möchten daher so rasch wie möglich mit dem Detailentwurf und der Codierung beginnen lassen.

Wie verhalten Sie sich?

Was halten Sie von folgenden Maßnahmen?

Mini-Übung 9.2 (Aufgabe 9.2 im Skript) – 2

a) Sie lassen alle festgestellten Mängel beheben und beginnen nicht mit dem Detailentwurf, bis das Dokument das Nachreview bestanden hat und freigegeben ist.

b) Sie lassen sofort mit Detailentwurf und Codierung beginnen und beschließen, die Fehler und Lücken erst zu beseitigen, wenn man beim Entwerfen bzw. Codieren an diese Stellen kommt.

c) Sie gehen den Review-Bericht durch, markieren die Punkte, die Ihnen schwerwiegend erscheinen und lassen nur diese beheben.

d) Sie veranlassen die Behebung der festgestellten Fehler, lassen aber gleichzeitig mit dem Detailentwurf eines Teilsystems beginnen, in dessen Konzept nur zwei leichte Mängel gefunden wurden.

Testen

Test – Ausführung eines Programms oder eines Software-Systems zwecks Überprüfung, ob dieses bei gegebenen Eingaben die erwarteten Resultate liefert

- Zwei mögliche Testziele:
  - Möglichst viele Fehler finden
    Myers (1979): Testen ist der Prozess, ein Programm mit der Absicht auszuführen, Fehler zu finden
  - Aus fehlerfreier Ausführung mit Testdaten statistisch auf fehlerfreie Ausführung mit realen Daten im Einsatz schließen

Testen – Grundlagen

- Jeder Test ist eine Stichprobe
- Korrektheit kann durch Testen nicht bewiesen werden
  - Beispiele: Addition von zwei 32-Bit-Zahlen: 2^{64} mögliche Testfälle
    Bearbeitung einer Zeichenkette mit 32 Zeichen: 2^{256} mögliche Testfälle
- Erwartete Ergebnisse müssen im Voraus bekannt sein
  - Testen gegen die Spezifikation
  - Testen gegen vorhandene Ergebnisse (Regressionstest)
- Unvorbereitete und undokumentierte Tests sind Zeitverschwendung
- Testen findet Fehlersymptome, keine Fehlerursachen
- Nach dem Test: Fehlerursachen finden und beheben (Debugging)

Aufwand für vollständigen Test

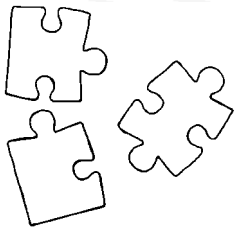
Kleine Aufwandrechnung: $2^{64} \approx 1{,}8 \cdot 10^{19}$

Annahme 1: Manueller Test mit 1 Testfall/s
Vollständiger Test dauert ca. $1{,}8 \cdot 10^{19}$ s ≈ 58,5 Milliarden Jahre

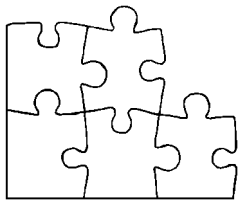
Annahme 2: Automatisierter Test auf 1000 Rechnern parallel,
1 Testfall / µs → 10^9 Testfälle/s
Vollständiger Test dauert ca. $1{,}8 \cdot 10^{10}$ s ≈ 58,5 Jahre

Testgegenstände

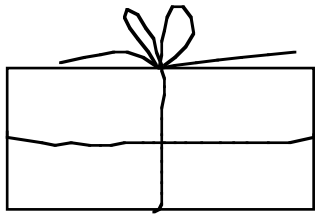
- Einzelkomponenten (Modultest)



- Baugruppen (Integrationstest)



- Systeme (Systemtest, Abnahmetest)



Testablauf – 1

- Planung
  - Teststrategie: was - wann - wie - wie lange
  - Einbettung des Testens in die Entwicklungsplanung
- Vorbereitung
  - Auswahl der Testfälle
  - Bereitstellen der Testumgebung
  - Erstellung der Testvorschrift
- Durchführung
  - Testumgebung einrichten
  - Testfälle nach Testvorschrift ausführen
  - Ergebnisse notieren
  - Prüfling während des Tests nicht verändern

Testablauf – 2

- Auswertung
  - Testbefunde zusammenstellen

[- Fehlerbehebung (ist nicht Bestandteil des Tests!)
  - gefundene Fehler(symptome) analysieren
  - Fehlerursachen bestimmen (Debugging)
  - Fehler beheben]

Testverfahren

- Funktionsorientierter Test (Black-Box-Test)
  - Auswahl der Testfälle aufgrund der Spezifikation
  - Programmstruktur braucht nicht bekannt zu sein
  - Geforderte Eigenschaften des Prüflings möglichst vollständig durch Testfälle abdecken
  - Fehlerträchtige Situationen gezielt provozieren

- Strukturorientierter Test (White-Box-Test, Glass-Box-Test)
  - Auswahl der Testfälle aufgrund der Programmstruktur
  - Spezifikation muss bekannt sein (für erwartete Ergebnisse)
  - Programmstruktur möglichst vollständig durch Testfälle abdecken

Mehr zum Thema «Test»

- In dieser Vorlesung nur summarisch behandelt
- Mehr zum Thema:
  - Literatur
  - Kernvorlesung Software Engineering