KV Software Engineering
Prof. Dr. Martin Glinz

Kapitel 11

Reviews

Universität Zürich
Institut für Informatik

11.1 Grundlagen

11.2 Review-Formen

11.3 Durchführung eines Review

11.4 Review-Verfahren

11.5 Review-Aufwand

Terminologie

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

- Abgrenzung: keine Reviews im hier betrachteten Sinn sind:
  - informelle Prüfungen, z.B. Durchlesen durch Kollegen
  - Management-Reviews zur Überprüfung von Kosten und Terminen
  - Sonderfall: Selbst-Review, d.h. eine Person überprüft ein eigenes Arbeitsergebnis mit den gleichen Verfahren, wie sie in formellen technischen Reviews zur Anwendung kommen
- Hier: nur formelle technische Reviews (einschließlich Selbst-Review)

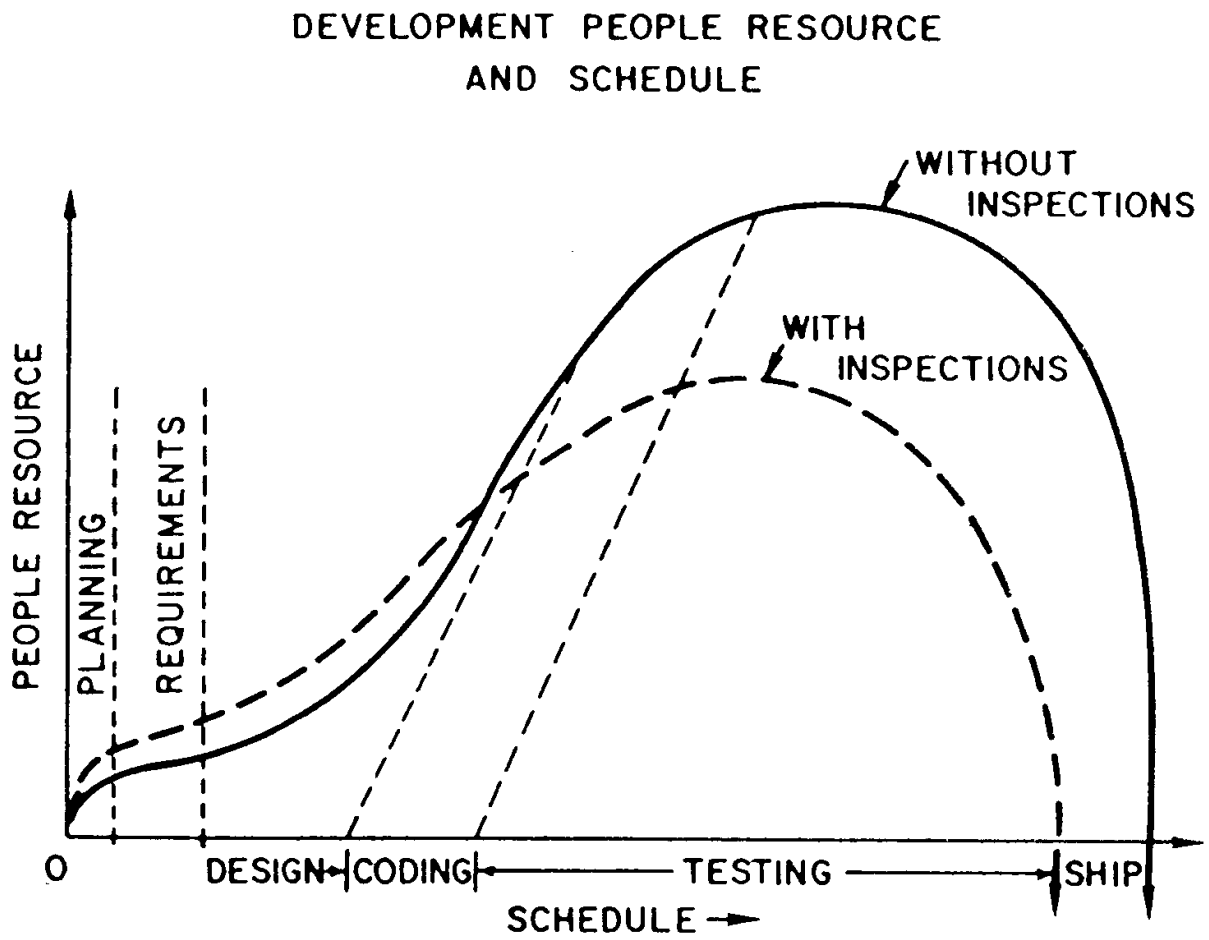
Einordnung

- Statisches Prüfverfahren
- Ziele
  - Aufzeigen der Schwachstellen und Mängel des Prüflings
  - Bestätigen der Stärken des Prüflings
  - Beurteilung der Qualität
- Ergebnis eines Reviews
  - Bericht mit einer Liste von Befunden

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

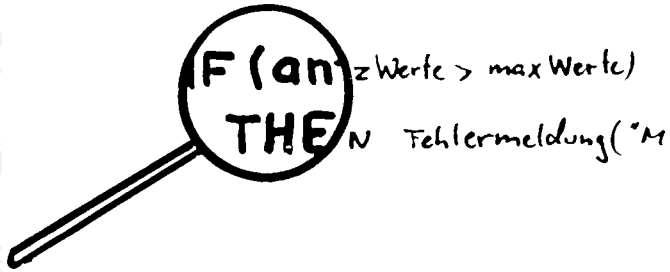
Wirtschaftlichkeit von Reviews



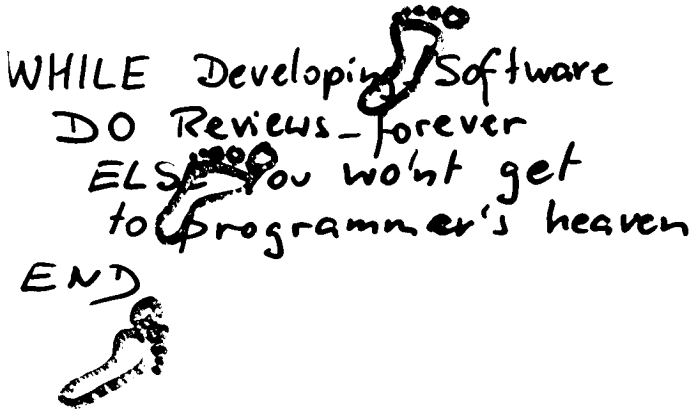
(Fagan, 1986)

Zwei Grundformen

- Inspektion



- Walkthrough



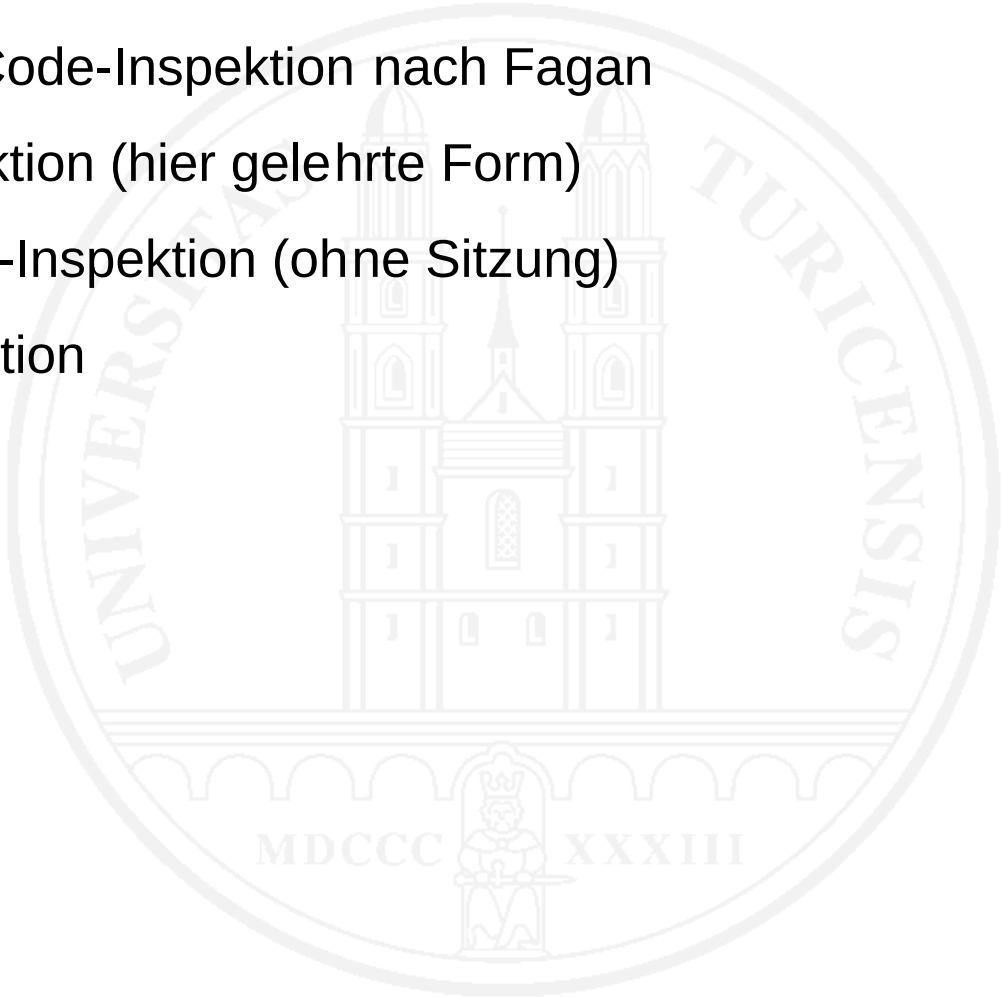
11.2.1 Inspektion

Prinzip: Der Prüfling wird von Gutachtern nach vorgegebenen Prüfkriterien Zeile für Zeile inspiziert

- Gutachter bereiten sich individuell vor
- Gutachter müssen mit den verwendeten Entwicklungsmethoden und -sprachen vertraut sein
- Prüfverfahren: siehe Kapitel 11.4
- Nur sinnvoll, wenn gegen klare Vorgabedokumente geprüft werden kann
- Von Fagan (1976) für Code- und Designüberprüfungen entwickelt
- Wirkungsvollste Reviewtechnik

Arten von Inspektionen

- Klassische Code-Inspektion nach Fagan
- Team-Inspektion (hier gelehrte Form)
- N-Individuen-Inspektion (ohne Sitzung)
- Selbstinspektion



Klassische Code-Inspektion nach Fagan

- Die Urform des technischen Reviews für Software (Fagan 1976)
- Ablauf:
  - Startsitzung: Vorstellen des Prüflings
  - Vorbereitung der Gutachter: Gutachter lesen den Code und müssen ihn verstehen; sie erstellen keine individuellen Befundlisten
  - Sitzung: Rollen wie in 11.3 beschrieben; ferner zusätzliche Rolle: Vorleser
- Der Vorleser liest den Code Zeile für Zeile vor
- Bei jeder Zeile können die Gutachter einhaken und Befunde vorbringen

Team-Inspektion

- Die hier gelehrte Inspektionsform
- Prüfung und Befunderhebung erfolgt individuell durch Gutachter in Vorbereitung, Gutachter erstellen individuelle Befundlisten bringen diese in die Sitzung mit
- Review-Sitzung dient nur zum Zusammentragen und Bewerten der Befunde
- Doubletten und falsche Befunde werden eliminiert
- Gemeinsame Befundliste und Empfehlung repräsentiert den Gruppenkonsens
- Nachteil: Aufwand für die Sitzung

N-Individuen-Inspektion (ohne Sitzung)

- Prüfung und Befunderhebung vollständig individuell durch Gutachter; keine gemeinsame Sitzung
- Review-Befund ist die Summe der Gutachterbefunde
- Wird aus Effizienzgründen teilweise propagiert
- Spart den Aufwand für die Sitzung
- Experimente zeigen, dass die Sitzung kaum neue Befunde erbringt (die kein Gutachter in der Vorbereitung erkannt hat)
- Kritische Durchsicht der Individualbefunde durch die Gruppe fehlt:
  - Befundliste enthält Redundanzen
  - Befunde sind nicht bewertet
  - Experimente zeigen, dass die Anzahl der falschen Befunde ansteigt

Selbstinspektion

- Wie N-Individuen-Inspektion, aber Autor ist sein eigener (und einziger Gutachter)
- Weniger effektiv als Inspektionen durch Fremdgutachter
- Jederzeit möglich (keine Probleme mit der Verfügbarkeit von Fremdgutachtern)
- Effektiver und effizienter als einfaches Durchlesen (da auf rigorosen Prüfverfahren basiert)
- Besser als Laufversuche und ad hoc Test

11.2.2 Walkthrough

Prinzip: Autor geht Prüfling mit Gutachtern durch, die Darstellung wird gemeinsam nachvollzogen

- Vorbereitung nur beschränkt möglich; Gutachter prüfen in der Sitzung durch kritisches Mitvollziehen der Ausführungen des Autors
- Gutachter müssen die verwendeten Entwicklungsmethoden und -sprachen nur soweit kennen, dass sie den Ausführungen des Autors folgen können
- Walkthroughs erfordern nicht zwingend klare Vorgabedokumente. Es kann auch gegen Ideen und Vorstellungen von (beim Walkthrough anwesenden) Anwendern, Fachgebietsexperten, o.ä. geprüft werden
- Weniger wirksam (geringere Fehlerentdeckungsrate) als Inspektion. Gefahr, dass Gutachter sich von geschickt und flüssig vortragendem Autor blenden lassen

11.1 Grundlagen

11.2 Review-Formen

11.3 Durchführung eines Review

11.4 Review-Verfahren

11.5 Review-Aufwand

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

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
- Details: Vorlesung Software Engineering I

Mini-Übung 11.1

Warum ist es wesentlich, dass die Befundliste öffentlich während der Sitzung erstellt wird?

Mini-Übung 11.2

Rekapitulieren Sie die Review-Regeln aus der Vorlesung Software Engineering I.

Begründen Sie die folgenden Regeln:

a) Alle Gutachter kommen vorbereitet

b) Probleme nur nennen, nicht lösen

c) Keine Stilfragen diskutieren

d) Produkt bewerten, nicht Produzenten

11.4.1 Prüfverfahren für Inspektionen

- Paraphrasieren / Erklären
- Checklisten durchgehen ($\rightarrow$11.4.4)
- Prüfprozeduren durcharbeiten ($\rightarrow$11.4.5)
- Szenarien durchspielen ($\rightarrow$11.4.6)
- Symbolisch ausführen
- Verschiedene Perspektiven einnehmen (z.B. Rolle des Benutzers, des Testers, des Pflegeprogrammierers)

11.4.2 Prüfverfahren für Walkthroughs

- Vorlesen
- Ereignisse / Szenarien durchspielen
- Pfade verfolgen
- Rollenspiel

11.4.3 Prüfverfahren für Selbstinspektionen

- Gleiche Verfahren wie bei Inspektion möglich
- Vor allem:
  - Szenarien durchspielen
  - Bei Code: symbolisch ausführen
  - Perspektiven des Testers und des Verwenders des erstellten Produkts einnehmen

11.4.4 Checklisten

- Auflistung von Prüfpunkten
- Müssen auf die Art des Prüflings abgestimmt sein (Anforderungen, Entwürfe, Code, ...)
- Gegebenenfalls abzustimmen auf die Methodik, mit welcher der Prüfling erstellt wurde

Beispiel: Checkliste für Anforderungsspezifikationen (Auszug)

- Allgemeine Fragen
  - Sind die Systemziele definiert?
  - Sind alle Anforderungen adäquat, d.h. drückt jede Anforderung ein Bedürfnis der Benutzer aus?
  - Sind alle Anforderungen klar und eindeutig formuliert?
  - Sind alle Annahmen und Randbedingungen dokumentiert? Sind sie notwendig?
  - ...
- Methodengebundene Fragen
  - Hat jede Klasse einen eindeutigen, treffenden Namen?
  - Ist jede Beziehung in beiden Richtungen benannt, und sind die Kardinalitäten spezifiziert und adäquat?
  - ...

Beispiel: Checkliste für Anforderungsspezifikationen

- Fragen zur Adäquatheit und Vollständigkeit
  - Sind alle genannten Funktionen notwendig zur Erreichung der Systemziele?
  - Ist jedes Systemziel durch Funktionen abgedeckt?
  - Sind die spezifizierten Eingaben ausreichend zur Erzeugung der verlangten Reaktionen?
  - Sind Reaktionen auf unerwartete/unerwünschte Ereignisse spezifiziert?
  - Sind die Leistungsanforderungen für jede Funktion und deren Ein- und Ausgabedaten spezifiziert?
  - ...

11.4.5 Prüfprozeduren

- Handlungsanweisungen zur Prüfung
- So aufgebaut, dass die Prüfpunkte einer Checkliste abgedeckt sind
- Werden in der Literatur (z. B. Porter, Votta und Basili 1995) als Szenarien bezeichnet (was eigentlich ein Missbrauch des Szenarienbegriffs ist)
- Beispiel: Prüfprozeduren zur Inspektion von Anforderungs-spezifikationen (Auszug)

Beispiel: Prüfprozeduren für Anforderungs-spezifikationen (Auszug)

- Bestimme alle Datenobjekte, welche das System mit seiner Umgebung austauscht. Für jedes Datenobjekt
  - Ist der Name adäquat?
  - Ist der Datentyp spezifiziert und adäquat?
  - Gibt es einen voreingestellten Wert und ist dieser adäquat?
  - Wenn das Datenobjekt eine physikalische Größe darstellt, sind Einheit und Genauigkeit spezifiziert?
  - Wenn das Datenobjekt ein Ausgabeobjekt ist, gibt es mindestens eine Funktion, welche dieses Objekt erzeugt?
  - ...

Beispiel: Prüfprozeduren für Anforderungs-spezifikationen (Auszug) – 2

- Bestimme die Ein- und Ausgabedatenobjekte für jede Funktionale Anforderung.
  - Sind alle notwendigen Eingabeobjekte vorhanden?
  - Sind alle vorhandenen Eingabeobjekte notwendig?
  - Werden alle Ausgabeobjekte verwendet?
  - ...
- ...

11.4.6 Szenarien

- Durchspielen problembezogener Benutzungsszenarien
- Beispiele:
  - Prüfung der Anforderungen an ein Bibliothekssystem
    - Ausleihen eines Buchs
    - Zurückgeben eines vorgemerkten Buchs
    - ...
  - Inspektion des Codes für eine verkettete Liste
    - Einfügen in die leere Liste
    - Einfügen in der Mitte einer Liste
    - Suchen des letzten Elements der Liste
    - ...

11.1 Grundlagen

11.2 Review-Formen

11.3 Durchführung eines Review

11.4 Review-Verfahren

11.5 Review-Aufwand

Aufwand-Anteile

Im Wesentlichen nur Personalaufwand:

- Zeitbedarf für Organisation des Review +
- Summe der Vorbereitungszeiten aller Gutachter +
- Dauer der Sitzung x Anzahl der Teilnehmer +
- Aufwand für Nachreviews bei zu vielen Befunden

Aufwandzahlen aus der Literatur

	1	2	3	4	5	6
Vorbereitung	1 p 60 l	3-5 p 150 l	k A 125 l	k A 150 l	10 p 150 l	k A <200 l
Sitzung	2 p 120 l	3-5 p 150 l	90- 125 l	k A 150 l	25 p 300 l	20 p k A

p Seiten eines Dokuments
l Codezeilen ohne Kommentar (NCSS)
k A keine Angabe

1 Gilb und Graham 1993, p. 376-377, 408
2 Strauss und Ebenau 1994, p. 89
3 Fagan 1986
4 Barnard und Price 1994
5 Frühauf, Ludewig und Sandmayr 1991, p. 113 (1 p in [5] entspricht 30 NCSS, pers. Mitteilung, 1994)
6 Weller 1993

Literatur

Barnard, J., A. Price (1994). Managing Code Inspection Information. *IEEE Software* **11**, 2 (Mar 1994). 59-69.

Fagan, M.E. (1976). Design and Code Inspections to Reduce Errors in Program Development. *IBM Systems Journal* **15,** 3. 182-211.

Fagan, M.E. (1986). Advances in Software Inspections. *IEEE Transactions on Software Engineering*, **SE-12**, 7 (Jul 1986). 744-751.

Freedman D.P., G.M. Weinberg (1982). *Handbook of Walkthroughs, Inspections and Technical Reviews*. 3rd edition. Boston, Toronto: Little, Brown and Co.

Frühauf, K., J. Ludewig, H. Sandmayr (1991). *Software-Prüfung. Eine Fibel.* Zürich: vdf und Stuttgart:Teubner.

Gilb, T., S. Graham (1993). *Software Inspection*. Wokingham, etc.: Addison-Wesley.

Porter, A.A., L.G. Votta and V.R. Basili (1995). Comparing Detection Methods for Software Requirements Inspections: A Replicated Experiment. *IEEE Transactions on Software Engineering* **21**, 6 (Jun 1995). 563-575.

Porter, A. and L. Votta (1997). What Makes Inspections Work? *IEEE Software* **14**, 6 (Nov-Dec 1997). 99-102.

Literatur – 2

Russell, G.W. (1991). Experience with Inspection in Ultralarge-Scale Developments. *IEEE Software* **8**, 1 (Jan 1991). 25-31.

Schnurer, K. (1988). Programminspektionen. Erfahrungen und Probleme. *Informatik-Spektrum* **11**, 6 (Dez 1988). 312-322.

Strauss, S.H., R.G. Ebenau (1994). *Software Inspection Process*. New York, etc.: McGraw-Hill.

Weller, E.F. (1993). Lessons from Three Years of Inspection Data. *IEEE Software* **10**, 5 (Sept 1993). 38-45.