Qualitätssicherung von Software

Prof. Dr. Holger Schlingloff

Humboldt-Universität zu Berlin
und
Fraunhofer FIRST

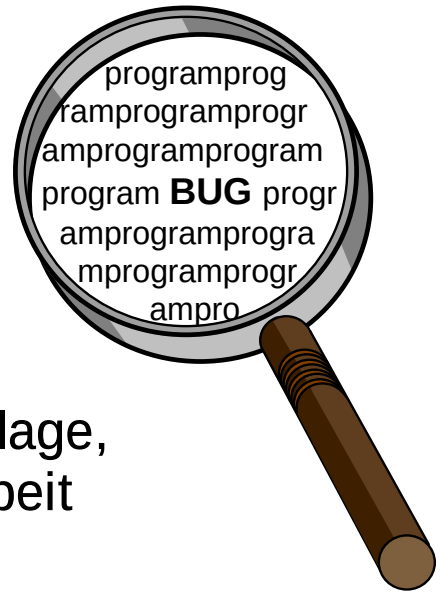
Wo stehen wir?

1. Einleitung, Begriffe, Software-Qualitätskriterien
2. manuelle und automatisierte Testverfahren
3. Verifikation und Validierung, Modellprüfung
4. statische und dynamische Analysetechniken
5. Softwarebewertung, Softwaremetriken
6. Codereview- und andere Inspektionsverfahren
7. Zuverlässigkeitstheorie, Fehlerbaumanalyse
8. Qualitätsstandards, Qualitätsmanagement, organisatorische Maßnahmen

Software-Reviews

Verschiedene Arten

- Formales (Design) Review
  - Statusreport, Hearing
  - Audit, Begehung



Entscheidungsgremium entscheidet nach Aktenlage, Vortrag und Anhörung über Fortsetzung der Arbeit

- Peer Review
  - Walkthrough
  - (Fagan) Inspektion

Gutachter beraten das Entwicklungsteam über Verbesserungsmöglichkeiten

Wozu ist manuelle QS notwendig?

- Abgleich mit den ursprünglichen Zielen
  - z.B. substantielle Notwendigkeit versus Korrektheit von Modulen (statische Analysen)
- Aufzeigen von inhaltlichen (nichtformalen) Fehlern
  - z.B. intuitive Bedeutung versus textuelle Gestalt eines Identifiers (Codierstandards)
- Verbesserung von Lesbarkeit und Verständlichkeit

- externe Beratung
- „Faktor Mensch“
- Kommunikation, Lernen
- oft einzige Möglichkeit

Capability Maturity Model (CMM)

Level	Focus	Key Process Areas
Level 5 Optimizing	Continuous improvement	Process Change Management Technology Change Management Defect Prevention
Level 4 Managed	Product and process quality	Software Quality Management Quantitative Process Management
Level 3 Defined	Engineering process	Organization Process Focus, Org. Process Definition **Peer Reviews,** Training Program Intergroup Coordination, SW Product Engineering Integrated Software Management
Level 2 Repeatable	Project management	Requirements Management, SW Project Planning SW Project Tracking and Oversight SW Subcontract Management, SW Quality Assurance SW Configuration Management
Level 1 Initial	Heroes	No KPAs at this time

Source: www.software.org/quagmire/descriptions/sw-cmm.asp

Ziele eines Peer Reviews

- Entdeckung von Design- und Analysefehlern in den zu untersuchenden Dokumenten
- Aufzeigen von Risiken, die den Projektfortschritt beeinträchtigen könnten
- Lokalisierung von Abweichungen gegenüber externen und internen Vorgaben und Richtlinien
- Bewertung bzw. Verbesserung der Qualität der Artefakte

- Kommunikationsmöglichkeit für die Beteiligten
- Datenbasis von Befunden für künftige Projekte

Artefakte für den Review

- *Jedes* Artefakt, welches als Ergebnis eines Entwicklungszyklusses vorliegt, kann per Review bewertet werden:
  - Anforderungsbeschreibung, Vermarktungsplan
  - Entwicklungsplan, Ressourcenverteilung
  - Entwurfsdokumente (Grob/Feinarchitektur)
  - Algorithmen und Datenstrukturen, **Code**
  - Testpläne, Testergebnisse
  - Manuale, Handbücher,
  - Versions- und Releasedokumente
- **Wichtig:** es muss eine stabile Version des Artefakts vorliegen

Teilnehmer

- Formales Review
  - Entscheidungsträger
    - Schriftführer und Review-Team
    - dürfen nicht mit dem Projekt zu tun haben
  - Autor(en) des Artefakts
    - Sachbearbeiter, Chef des Entwicklungsteams, ...
- Peer Review
  - externe Berater: Designer, Implementierer, Tester, Benutzer, Qualitätsbeauftragter, Produktlinienmanager, ...
    - Schriftführer sollte Erfahrung mit Reviews und Moderation haben
  - Autor(en)
- Optimale Größe: 3-5 Reviewer + Autor(en), optimale Zeitdauer: 2h (max. 2 mal pro Tag)

Durchführung Review

- Präsentation des Dokuments
- Kommentare des Review-Teams
- Diskussion der einzelnen Kritikpunkte
- Ergebnis
  - Formales Review: Beratung mit Entscheidung über Fortführung, bedingte Fortführung oder Ablehnung (mit Begründung)
  - Peer Review: Eintrag der gefundenen „Issues“ in Defektverfolgungssystem, Protokoll der Sitzung für künftige Projekte

Inspektionsprotokoll

Inspection Master Plan Insp ID ____ Objectives: _____

Inspection Leader: _____ Tel: _____

Author(s): _____ Date Inspection requested _____

Product: _____ Total pages _____ Doc Status _____

(exited, reviewed, etc.)

*Status	Entry Criteria	Exit Criteria	*Status

(*date met, waived, not applicable)

Documents	Name / Reference	Tag	Relevant sections/pages
Product Chunk(s)			
Rule Sets			
Generic Checklists			
Source(s)			
Kin (related documents)			

Walkthrough und Inspektion

- **Walkthrough:** „geführtes Vorlesen vor aufmerksamem Publikum“
  - detaillierte Erklärung durch den Autor
  - keine bzw. minimale Vorbereitung der Reviewer
  - gemeinsames Verständnis als Hauptziel
- **Inspektion:** „Frage- und Antwortstunde“
  - Vorbereitung von Fragen durch Reviewer (3:1) (anhand Checklisten, 30-90% individuelle Findungen)
  - Beantwortung durch Autor so weit möglich

Inspektion ist aufwändiger aber effektiver!

Checklisten

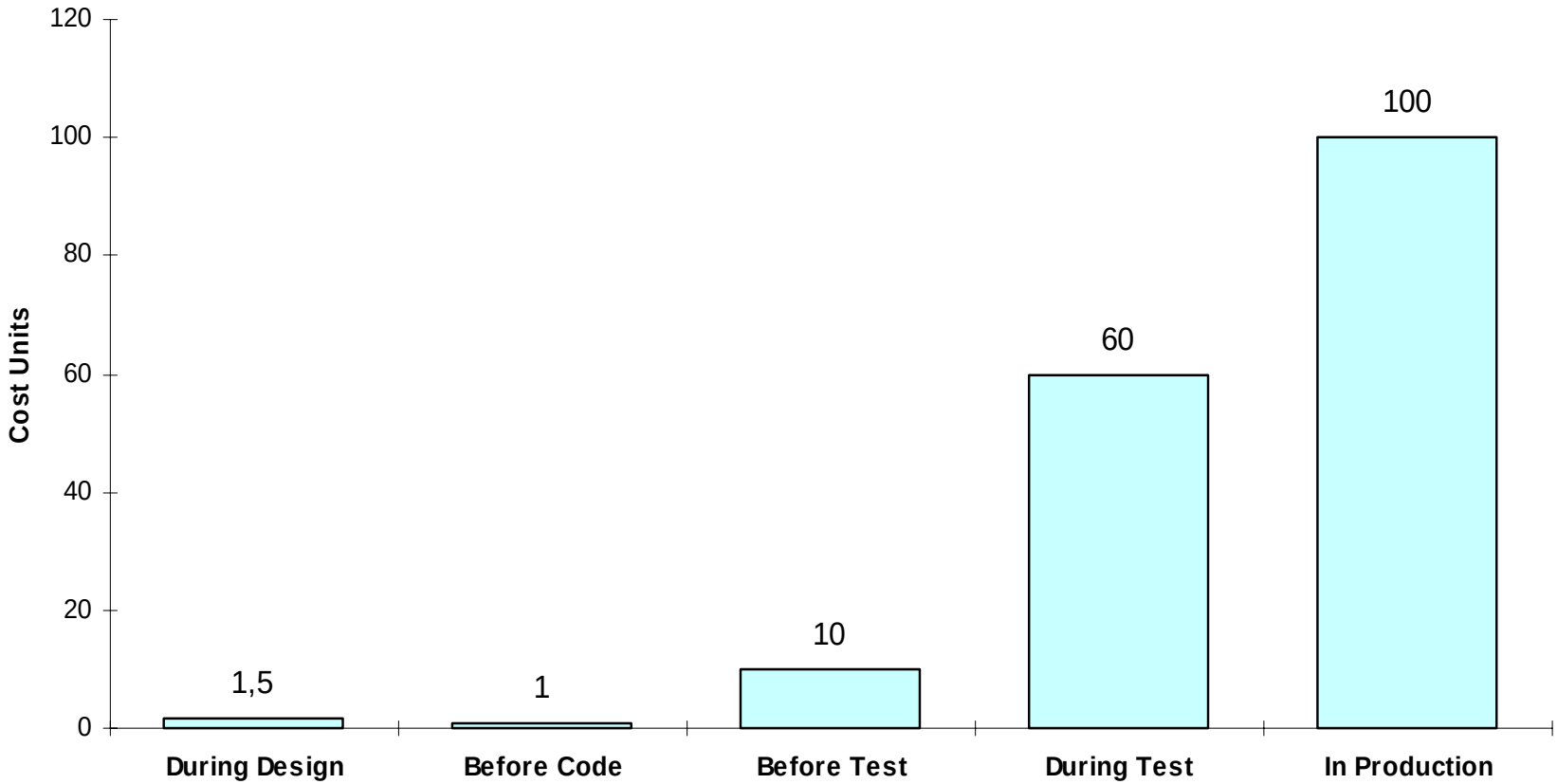
- essenziell für die Vorbereitung des Reviews
- selbe Form, aber deutlich andere Schwerpunktsetzung als Codierrichtlinien
- sind vor Beginn der Entwicklung bekannt, werden den Reviewern bekannt gemacht
- dienen als Richtlinie bei der Durchführung des Reviews

- Kategorisierung der Defekte, Fokus auf Probleme mit hohen ökonomischen Auswirkungen!

Erfahrungen

- richtig angewandt, sind Reviews ein extrem effizientes Mittel der QS
- Angaben aus der Literatur:
  - *An AT&T Bell Laboratory project with 200 professionals instituted several changes, including inspections. Productivity improved by 14% and quality by a factor of ten.*
  - *Aetna Insurance Company: inspections found 82% of errors in a COBOL program, productivity was increased by 25%.*
  - *Another COBOL example (Gilb, Software Metrics): 80% of the development errors were found by inspections, productivity was increased by 30%.*

Relative Fehlerbehebungskosten



Source: Tom Gilb, Software Engineering Management, Daten der Standard Chartered Bank

Inspektionsrate und Ergebnisprotokoll

Inspektionsrate

- für Programme: 100 – 150 NLOC / h (1-2 Seiten / h)
- für Textdokumente
  - Gilb/Graham: ca. 1 Seite / h
  - Strauss/Ebenau: 3 – 5 Seiten / h
  - Zum Vergleich: „Rechtschreibfehler-Leserate“ beträgt ca. 7 – 25 Seiten / h

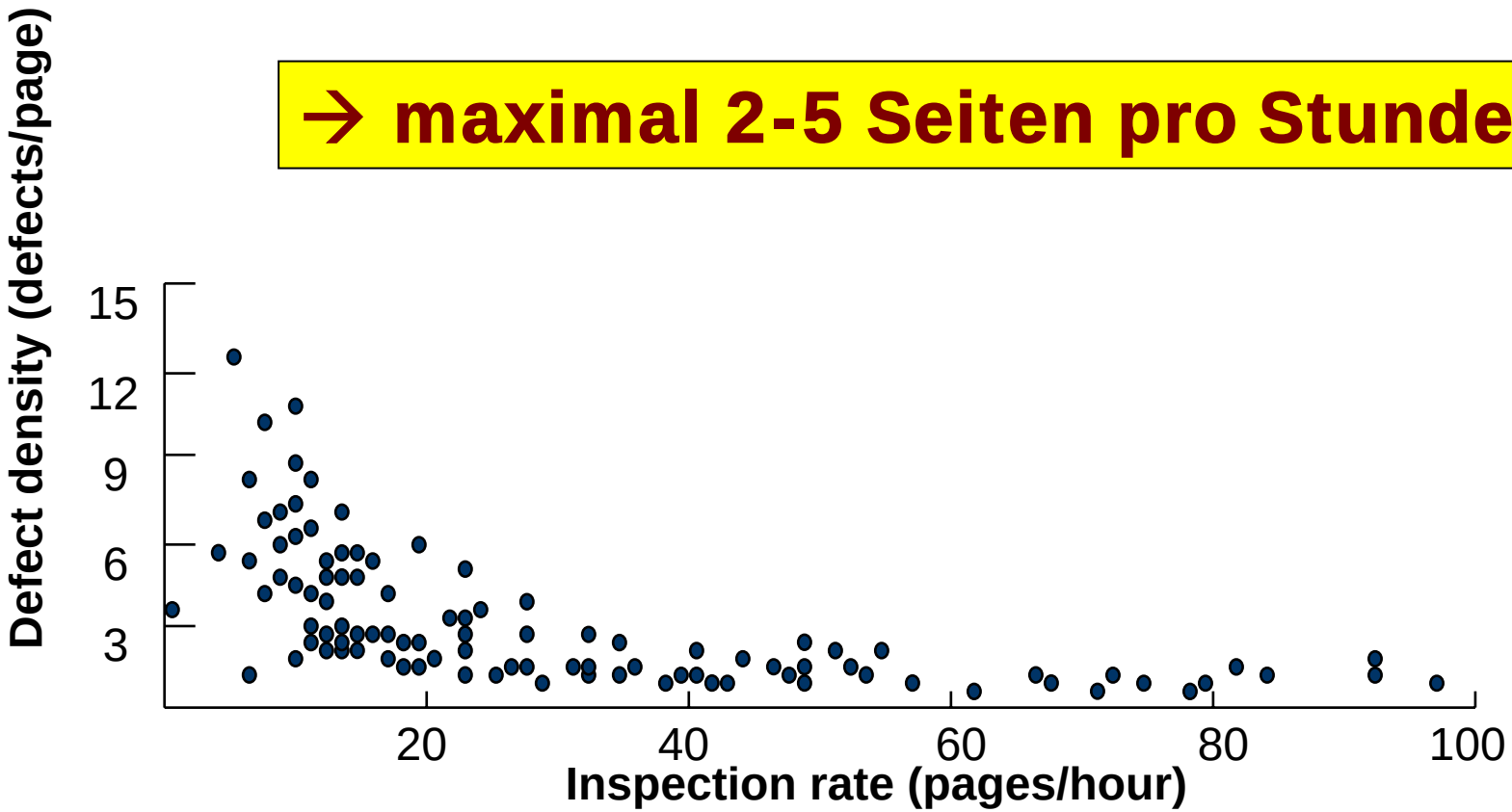
- höhere Rate, falls es sich bloß um eine Überprüfung handelt

Ergebnisprotokoll

- Dokument wird geprüft, nicht der Autor!
- keine Diskussion von Fehlern und Lösungswegen
- hohe Protokollrate (zum Teil mehr als 1 Eintrag pro Minute)

Defektaufdeckungs- und Inspektionsrate

→ maximal 2-5 Seiten pro Stunde!



Source: Tom Gilb, Denise Leigh Software Inspection p 334, 230 inspections of Sema Group (GB)

Fagan's Inspektionsmethode

1. überall im Entwicklungsprozess
2. alle Arten von Fehlern
3. ohne big boss
4. mehrere Einzelschritte
5. Checklistenbasiert
6. max. 2 Stunden
7. Rollen werden zugewiesen
8. trainierter Moderator
9. Statistiken werden geführt
10. Inspektionsrate wird einhalten

Pause!

Checklisten für Codereviews

Beispiel: *Java Code Inspection Checklist* von Christopher Fox

- **Variable and Constant Declaration Defects**
  1. Are descriptive variable and constant names used in accord with naming conventions?
  2. Are there variables with confusingly similar names?
  3. Is every variable properly initialized?
  4. Could any non-local variables be made local?
  5. Are there literal constants that should be named constants?
  6. Are there macros that should be constants?
  7. Are there variables that should be constants?

- **Function Definition Defects (FD)**
  8. Are descriptive function names used in accord with naming conventions?
  9. Is every function parameter value checked before being used?
  10. For every function: Does it return the correct value at every function return point?
- **Class Definition Defects (CD)**
  11. Does each class have an appropriate constructor and destructor?
  12. For each member of every class: Could access to the member be further restricted?
  13. Do any derived classes have common members that should be in the base class?
  14. Can the class inheritance hierarchy be simplified?
- ...

- ...
- **Performance Defects**

54. Can better data structures or more efficient algorithms be used?
55. Are logical tests arranged such that the often successful and inexpensive tests precede the more expensive and less frequently successful tests?
56. Can the cost of recomputing a value be reduced by computing it once and storing the results?
57. Is every result that is computed and stored actually used?
58. Can a computation be moved outside a loop?
59. Are there tests within a loop that do not need to be done?
60. Can a short loop be unrolled?
61. Are there two loops operating on the same data that can be combined into one?

Protokollschema

Inspection Issue Log

Inspection Identification _____. Page _____ of _____.

No	Document Reference Tag	Doc. Page	Line No./ Location	Type of Item (circle)	Checklist/ Rule Tag	Description (if needed) (key offending words)	Number of occur.	Editor Note
1				Major PI New minor ?				
2				Major PI New minor ?				
3				Major PI New minor ?				
4				Major PI New				

Beispiel: Autocode-Review

ACQ1-4	Art: M	Std: MG 72	Effz:	Port:	Reus:	Sich: ++

1.Redundanzen

Ist ausgeschlossen, dass es durch ähnliche Namen zu Verwechslungen kommt?

Modellbeispiel

✘	✓
nb_sf_notbremsen = FALSE nb_sf_notbremsung= FALSE	nb_sf_not??? nb_sf_notbremsung

ACQ2-24	Art: M	Std:	Effz:	Port:	Reus:	Sich:

Sind alle vorhandenen Typecasts notwendig?

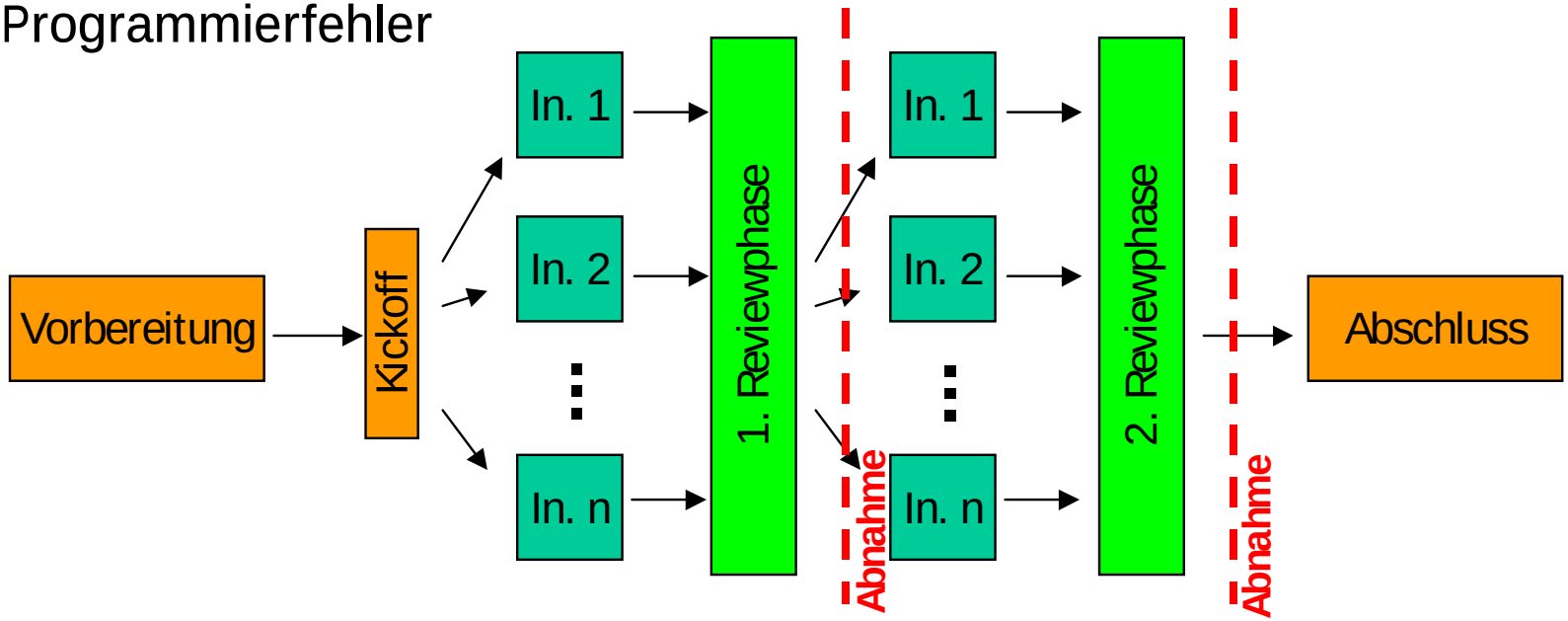
Beispiel:

Modellbeispiel

✘	✓
Sb130_Switch1= (Int32)(((Int32)Sb115_v_eigen) << 7);	

Aufteilung auf zwei Phasen

- ersten Reviewphase: generierter Code als solches
  - verständlich
  - strukturiert
- zweiten Reviewphase: spezifische Fehlerursachen Autocode
  - inkorrekte Skalierungen
  - Programmierfehler

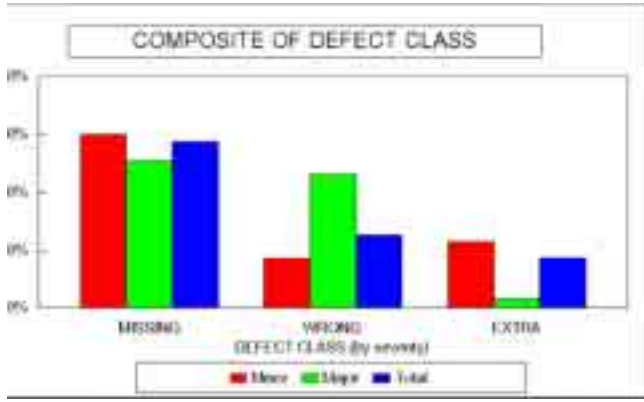


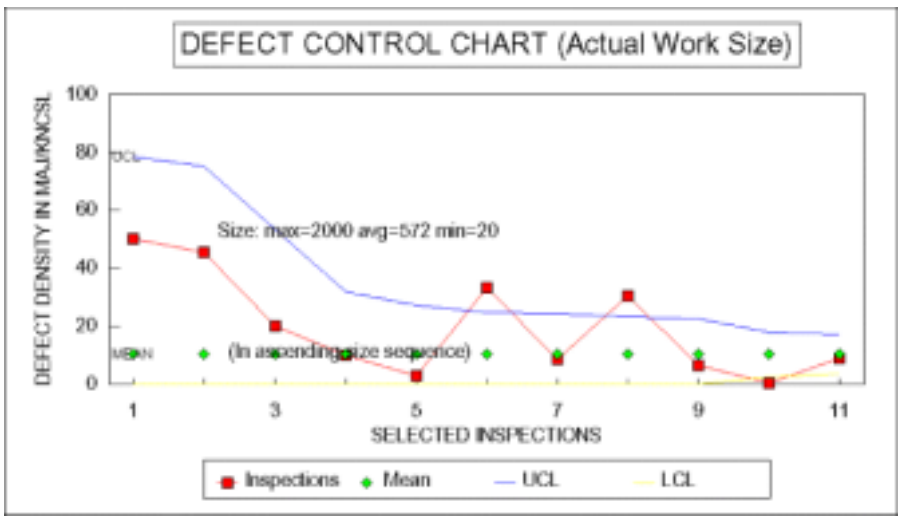
Probleme mit Checklisten

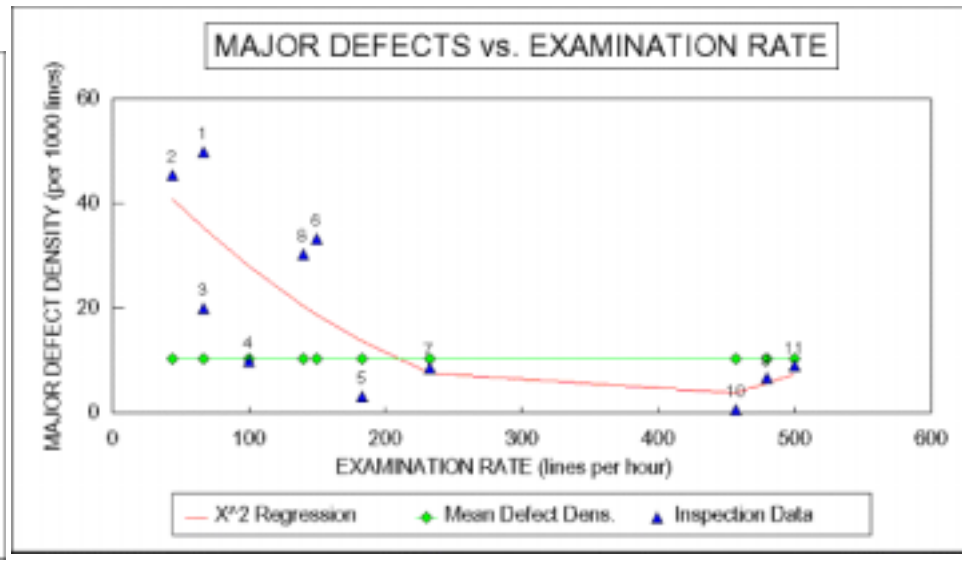
- **Umfangreiche Listen sind schwer im Kopf zu behalten!!!**
  - Aufteilung auf mehrere Phasen
  - Training / Einarbeitung
  - Preprocessing (z.B. Coding rules)
  - Werkzeugunterstützung
- LIDS: Lotus Inspection Data System
  - Datenverwaltung
  - Datenanalyse

LIDS Screenshots

INSP #	PROJ	REL	ACT	DOC	COMPONENT	INSP-DATE
1	43225	NEW	1.75	1.7512	ACCOUNTRECORDIMPL	04/05/94
2	1147	new	1.75	1.7506	dc102/DUT	03/09/94
3	1175	NEW	1.75	1.7505	DIAGS	03/03/94
4	1175	NEW	1.75	1.7504	EXPSTORAGE/FS	02/15/94
5	1108	NEW	1.75	1.7506	ENABLESCSIDRIVES	03/14/94
6	1171	NEW	1.75	1.7511	SECURITY TOOL PHASE 2	04/06/94
7	1107	NEW	1.75	1.7508	FTPFILING	03/20/94
8	1174	NEW	1.75	1.7505	PP	02/04/94
9	1107	NEW	1.75	1.7509	ESSDC306IMPL	03/30/94
10	1175	NEW	1.75	1.7509	RTOSTAR	04/04/94
11	1107	NEW	1.75	1.7510	SCSIMGR	04/05/94





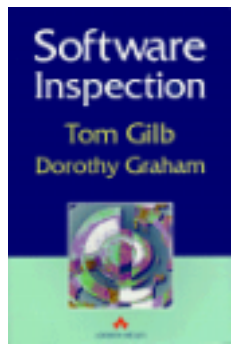


Weitere Informationsquellen

www.reviewtechnik.de (Peter Rösler):

- Kostenlose „Reviewtechnik-Sprechstunde“
- Linksammlung zu Reviewtechnik
- Checklisten

„Software Inspection“ von Tom Gilb und Dorothy Graham, ISBN 0-201-63181-4



„Peer Reviews in Software: A Practical Guide“ von Karl E. Wiegers, ISBN 0-201-73485-0