




Von der Idee zum Erfolg.

# Neue Trends in der Software-Entwicklung

Harry Sneed

© ANECON Software Design und Beratung G.m.b.H. | Alser Str. 4/Hof 1 | A-1090 Wien | Tel.: +43 1 409 58 90 | www.anecon.com | office@anecon.com

## Zur Wirtschaftlichkeit der Qualitätssicherung

- |  |  |
|--|--|
| 1. Fragen zur Wirtschaftlichkeit der QS      | 15. Kosten des Systemstests                  |
| 2. Was ist ein Software-Fehler ?             | 16. Nutzen des Systemstests                  |
| 3. Fehlerhäufigkeit                          | 17. Testkostentreiber                        |
| 4. Das Software Qualitätsdilemma             | 18. Kombinatorische Explosion der Testfälle  |
| 5. Kosten der Software-Fehler in Deutschland | 19. Warum der Test so aufwendig ist          |
| 6. Schaden durch Software-Fehler in den USA  | 20. Kosten des Testens ohne Automation       |
| 7. Kosten eines Fehlers                      | 21. Kosten des Testens mit Automation        |
| 8. Kosten der Fehlerbehebung                 | 22. Kosten/Nutzen Vergleich                  |
| 9. Verhältnis von Entwicklern zu Testern     | 23. Kostenanteile der QS-Maßnahmen           |
| 10. Kostenanteil der Qualitätssicherung      | 24. Nutzenanteile der QS-Maßnahmen           |
| 11. Kosten der Entwurfsbewertung             | 25. Kosten der QS nach COCOMO                |
| 12. Nutzen der Entwurfsbewertung             | 26. Nutzen der QS nach COCOMO                |
| 13. Kosten der Codeprüfung                   | 27. Indirekter Nutzen der Qualitätssicherung |
| 14. Nutzen der Codeprüfung                   | 28. Qualitativer Nutzen des Tests            |

2 | Harry Sneed



## Fragen zur Wirtschaftlichkeit der Qualitätssicherung

---

- Was kostet ein Software Fehler?
- Was kostet die Fehlerbehebung?
- Wie sind Fehler zu verhindern?
- Was kostet ein Review?
- Was nutzt ein Review?
- Was kostet die Codeprüfung?
- Was nutzt die Codeprüfung?
- Was kostet der Systemtest?
- Was nutzt der Systemtest?
- Was bringt die Testautomation?
  
- Wie verhalten sich die Kosten zum Nutzen?

3 | Harry Sneed



## Was ist ein Software-Fehler?

---

- Von einem Fehler kann nur gesprochen werden, wenn ein gegebener Sachverhalt von einer vereinbarten Norm abweicht.
- Bei Software-Systemen ergibt sich die Bezugsnorm aus den Vorstellungen des Benutzers und den aus ihnen abgeleiteten Abnahmekriterien.
- Ein Software-Fehler kommt vor, wenn die eigentliche Leistung der Software von der Leistungsbeschreibung abweicht.
- Ein Software-Fehler ist gegeben, wenn die Software sich anders verhält als in Dokumentationen bzw. Benutzerhandbüchern beschrieben.
- Ein Software-Fehler ist vorhanden, wenn die Software nicht so funktioniert, wie der Benutzer es erwartet.
- Ein Software-Fehler ist alles, was dem Benutzer nicht passt.

4 | Harry Sneed



## Fehlerhäufigkeit

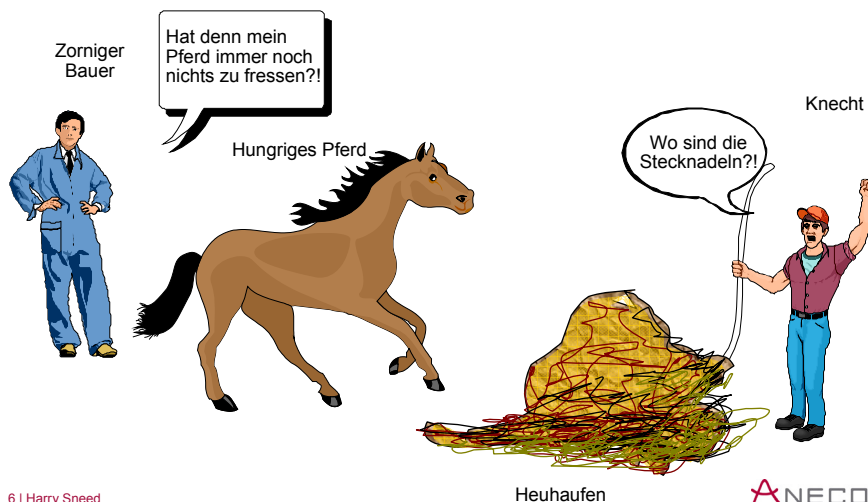
- Der Mensch ist ein fehlerhaftes Wesen, der seine Fehlerhaftigkeit in seinen Werken weitergibt.
- Von 100 Zeilen, die der Mensch schreibt bzw. 100 Strichen, die der Mensch zeichnet, werden **1,5 fehlerhaft** sein.
- Davon wird der Mensch 50% selbst merken. Es bleiben also **0,7 fehlerhafte Zeilen**.
- In einem Programm oder Dokument mit 1000 Zeilen sind das **7 Fehler**.
- In einem Programm oder Dokument mit 10 000 Zeilen sind das **70 Fehler**.
- Je größer und komplexer das Programm desto mehr Fehler
- **Sie sind unvermeidlich!**
- Fehler zu finden ist die Aufgabe des **TESTENS**

5 | Harry Sneed



## Das Software-Qualitätsdilemma

Die Anzahl der zu entfernenden Fehler ist unbekannt, ergo kann man nie wissen wann man sie alle entfernt hat.



6 | Harry Sneed



## Kosten der Software-Fehler in Deutschland

---

Aus einer Studie des deutschen Ministeriums für Forschung und Bildung im Jahre 2001:

- Wert sämtlicher Software Anwendungssysteme =  
**25 Milliarden Euro**
- Größe der Software Anwendungssysteme =  
**1,25 Milliarden Anweisungen**
- Bei einer mittleren Fehlerrate von 3 Fehler pro 1000 Anweisungen =  
**3,5 Millionen Fehler**
- Eine Fehlerbehebung kostet im Durchschnitt 13 Stunden =  
**Euro 650,-**
- Die Behebung 3,5 Millionen Fehler kostet  
**Euro 2,4 Milliarden bzw. 9% des Wertes**

7 | Harry Sneed



## Schaden durch Software-Fehler in den USA

---

- Aus einer Studie des amerikanischen Wirtschaftsministeriums im Jahre 2000 für die Automobil- und Flugzeugbranchen:
- Schaden durch Software-Fehler =  
**1,8 Milliarden US Dollar, bzw. 16% der Software Kosten**
- Für die gesamte amerikanische Wirtschaft macht das insgesamt =  
**59,5 Milliarden US Dollar**
- Durch die rechtzeitige Beseitigung der Fehler wäre es möglich gewesen die Hälfte dieser Kosten zu sparen =  
**30 Milliarden US Dollar**

8 | Harry Sneed



## Erfahrungen aus dem GEOS Projekt

Aus einer Studie der GEOS Anwender in Österreich im Jahre 2001:

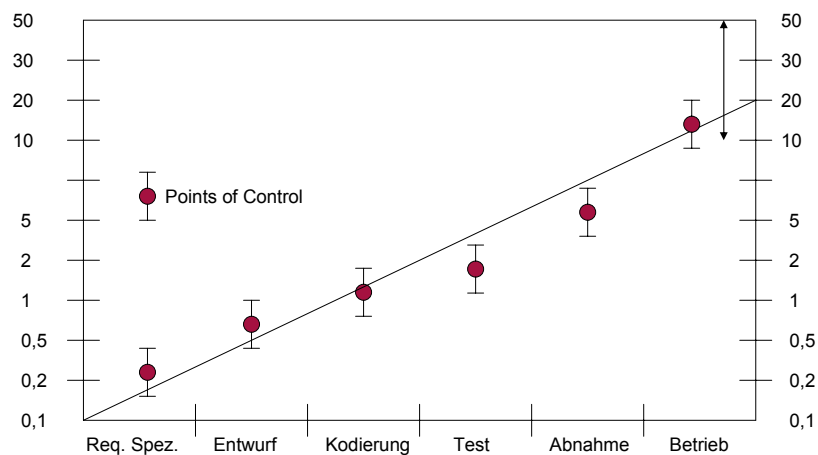
- Mittlerer Schaden durch einen einzigen Software-Fehler in der Produktion einer Großbank =  
**Euro 7,620**
- Kosten einer mittleren Fehlerbehebung = 2,5 PTs =  
**Euro 2,000**
- Gesamte Kosten eines Software-Fehlers =  
**Euro 9,620**
- Bei 795 schwere Fehler pro Jahr macht das  
**Euro 7,3 Millionen**
- Dafür könnten 72 Tester im Jahr beschäftigt werden

9 | Harry Sneed



## Kosten der Fehlerbehebung

Kostenverhältnis, um Fehler zu beheben

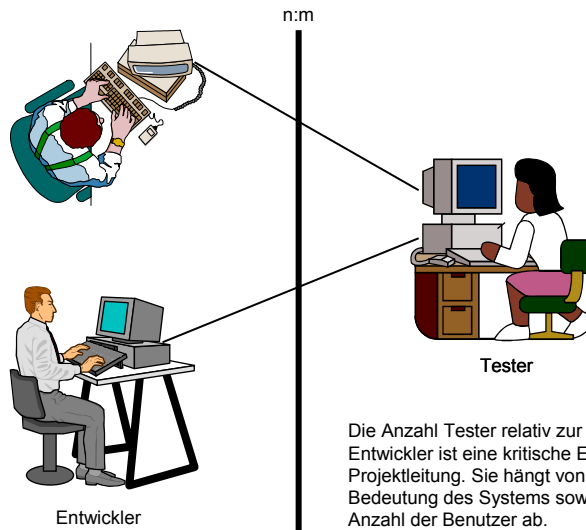


10 | Harry Sneed

Entwicklungsphasen



## Was ist das optimale Verhältnis von Entwicklern zu Testern

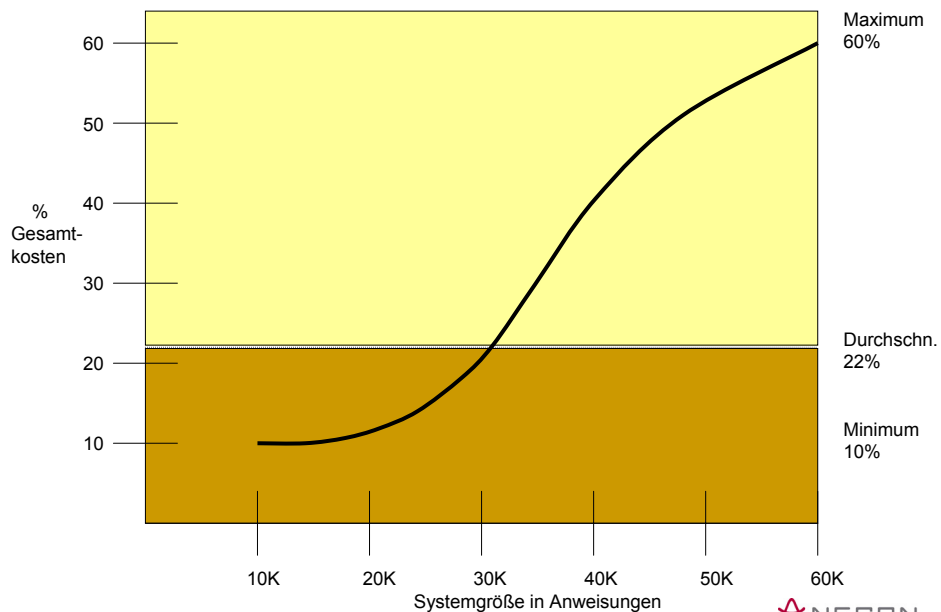


Die Anzahl Tester relativ zur Anzahl Entwickler ist eine kritische Entscheidung der Projektleitung. Sie hängt von der Größe und Bedeutung des Systems sowie von der Anzahl der Benutzer ab.

11 | Harry Sneed



## Kostenanteil der Qualitätssicherung

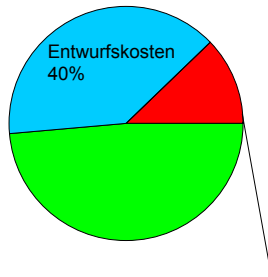


12 | Harry Sneed



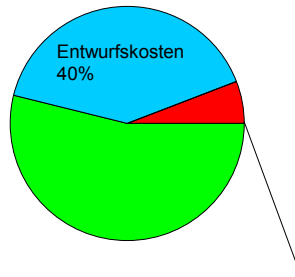
## Kosten der Entwurfsbewertung

Bewertungskosten ohne  
Werkzeugunterstützung



+ 12% für die Entwurfsbewertung

Bewertungskosten mit  
Werkzeugunterstützung

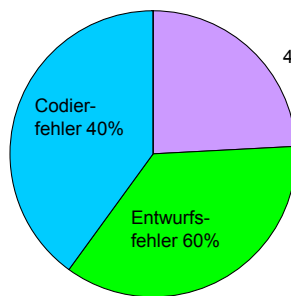


+ 6% für die Entwurfsbewertung

13 | Harry Sneed



## Nutzen der Entwurfsbewertung



40% der Entwurfsfehler  
können durch die  
Entwurfsbewertung  
gefunden werden

Erhöhung der Korrektheit um 24%  
Vollständigkeit  
Konsistenz  
Modularität  
Flexibilität  
Portabilität

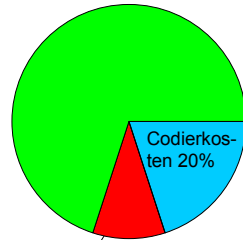
Nach Fagan, IBM: Ersparnis von 94 Programmierstunden per 1000 LOC  
Reduzierung der Folgekosten um ca. 10%

14 | Harry Sneed



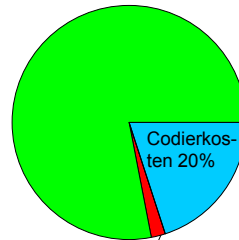
## Kosten der Codeprüfung

Prüfungskosten ohne  
Werkzeugunterstützung



+10% für die Codeprüfung

Prüfungskosten mit  
Werkzeugunterstützung



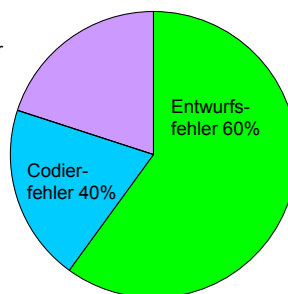
+ 2 % für die Codeprüfung

15 | Harry Sneed



## Nutzen der Codeprüfung

50% der Codierfehler  
können durch die  
Codeprüfung gefun-  
den werden



Erhöhung der Korrektheit um 20%

Integrität  
Sicherheit  
Transparenz  
Wartbarkeit

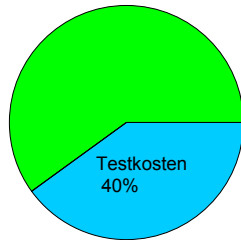
Nach Fagan, IBM: Ersparnis von 51 Programmierstunden per 1000 LOC  
Reduzierung der Folgekosten um ca. 7,5%

16 | Harry Sneed



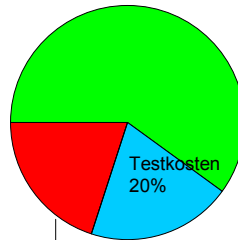
## Kosten des Systemtests

Systemtestkosten ohne  
Werkzeugunterstützung



+ 40% für den Systemtest

Systemtestkosten mit  
Werkzeugunterstützung



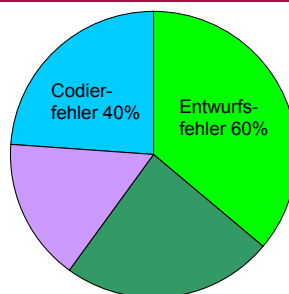
+ 20 % für den Systemtest

17 | Harry Sneed



## Nutzen des Systemtests

40% der Codierfehler können durch systematisches Testen gefunden werden



40% der Entwurfsfehler können durch systematisches Testen gefunden werden

40% aller Fehler können durch systematisches Testen gefunden werden

Erhöhung der Korrektheit um 40%  
Konsistenz  
Integrität  
Sicherheit

Reduzierung der Folgekosten um ca. 12,5%

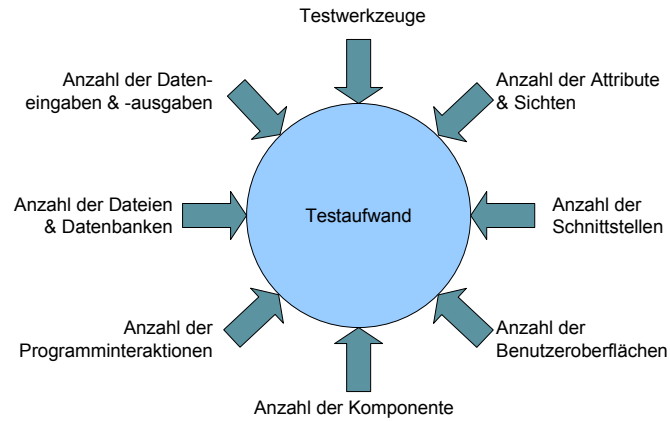
Methode	Fehler gefunden	Fehler
Unit Testing	6	25
Interface Testing	12	25
Functional Testing	17	25

18 | Harry Sneed

Nach Howden in „Empirical Studies of Software Evaluation“



## Testkostentreiber

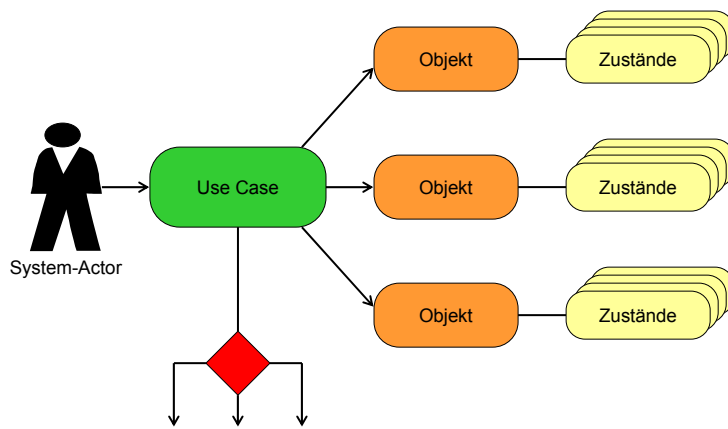


Größe und Komplexität bestimmen die Anzahl der Testfälle.  
Die Anzahl der Testfälle und der Grad der Testautomation bestimmen den Testaufwand.

19 | Harry Sneed



## Kombinatorische Explosion der Testfälle



Testfälle = Use Cases (1)  
\* Ausgänge (3)  
\* Objekte (3)  
\* Zustände (4) = 36

20 | Harry Sneed



## Warum der Test so aufwendig ist

- Zu jedem Use Case, bzw. jeder Transaktion, gibt es n Ausgänge. Z.B. beim Use Case „Geld vom Automaten holen“ gibt es neben dem guten Ausgang mindestens sieben schlechte Ausgänge
  - die Karte ist ungültig,
  - die PIN-Nummer stimmt nicht,
  - der Server ist nicht erreichbar,
  - der Automat hat kein Geld mehr,
  - der gewünschte Betrag ist höher als die Tagesgrenze,
  - das Konto ist gesperrt und
  - das Konto ist überzogen.
- Alle acht Fälle müssen getestet werden. Falls ein Fehler auftritt, müssen die acht Fälle wiederholt werden.
- Alle acht Testfälle müssen spezifiziert, erfasst und verwaltet werden.
- Der Aufwand für die Spezifikation eines Testfalls entspricht dem Aufwand für die Kodierung einer Funktion.
- Der Aufwand für die Spezifikation von acht Testfällen entspricht dem Aufwand für die Kodierung einer kleinen Klasse.
- Normale Use Cases haben 10-20 Ausgänge und betreffen 2-4 Objekttypen. Jeder Objekttyp hat 4-8 Zustände. Um alle Objektzustände mit allen logischen Ausgängen zu verwenden, braucht man 80 bis 640 Testfälle pro Use Case.

21 | Harry Sneed



## Kosten des Testens ohne Testautomation

- **Testobjekt:** Eine Web-Anwendung mit 4000 Anweisungen, 40 Klassen, 4 Datenbanken, 4 Systemschnittstellen und 5 GUI Masken
  1. Der Tester erstellt Programme, um die Datenbanken und Schnittstellen zu generieren. Dazu braucht er 2 Wochen.

Kosten = 5 000,- Euro
  2. Der Tester denkt sich Testdaten aus und gibt sie in die Testdatengenerierungs-Programme ein, um die Datenbanken und Dateien zu generieren. Dazu braucht er eine Woche.

Kosten = 2 500,- Euro
  3. Der Tester füllt Formulare zur Erfassung der Bildschirmdaten aus. Dazu braucht er eine Woche.

Kosten = 2 500,- Euro
  4. Der Tester füllt die Bildschirmmasken aus, um die Anwendung im Dialogbetrieb zu testen. Für jeden Testfall braucht er 1/2 Stunde am Terminal, um die Eingaben ein zu geben und die Ausgaben visuell zu bestätigen. Bei 50 Testfällen braucht er 25 Stunden. Wegen 10 Fehlern muß er den gesamten Test noch einmal wiederholen.

Kosten = 5 000,- Euro
- **Gesamtkosten:** 1 1/2 Mannmonat = 15 000,- Euro oder 300,- Euro pro Testfall oder 1 500,- Euro pro Fehler
- **Testdauer:** 6 Wochen

22 | Harry Sneed



## Kosten des Testens mit Testautomation

- Testobjekt:** Eine Web-Anwendung mit 4 000 Anweisungen, 40 Klassen, 4 Datenbanken, 4 Systemschnittstellen und 5 GUI Masken
  - Der Tester leitet die Testfälle und Testdaten aus dem Fachkonzept ab. Dazu braucht er 2 Wochen.  
**Kosten = 5 000,- Euro**
  - Der Tester verfasst die Testskripte. Dazu braucht er eine Woche.  
**Kosten = 2 500,- Euro**
  - Der Tester führt den Test mit einem Testautomat im Stapelbetrieb aus. Er testet 100 Testfällen. Wegen 14 Fehlern im System muss er den Test zwei mal wiederholen. Dazu braucht er insgesamt 5 Tage  
**Kosten = 2 500,- Euro**
- Gesamtkosten:** 1 Mannmonat = 10 000,- Euro oder  
 100,- Euro pro Testfall oder  
 714,- Euro pro Fehler
- Testdauer:** 4 Wochen

23 | Harry Sneed



## Kosten-/Nutzen Vergleich

	Kosten	Nutzen
Ohne Testsystem	15 000,- Euro	5 000,- Euro Fehlerbehebungskosten eingespart
	300,- Euro pro Testfall	95% Zuverlässigkeit
	7,- Euro pro Anweisung	75% Testdeckung
	1 500,- Euro pro Fehler	10 von 15 Fehlern gefunden
Mit Testsystem	10 000,- Euro	7 000,- Euro Fehlerbehebungskosten eingespart
	100,- Euro pro Testfall	99% Zuverlässigkeit
	5,- Euro pro Anweisung	95% Testdeckung
	714,- Euro pro Fehler	14 von 15 Fehlern gefunden

24 | Harry Sneed



## Kostenanteile der QS-Maßnahmen

Methode	QS-Kosten ohne Werkzeuge	QS-Kosten mit Werkzeugen
Design Review	+ 12%	+ 6%
Code-Inspektion	+ 10%	+ 2%
Systemtest	+ 40%	+ 20%
Insgesamt	+ 62%	+ 28%

Kostenanteil		
Methode	% QS-Kosten ohne Werkzeuge	% QS-Kosten mit Werkzeugen
Design Review	19%	21%
Code-Inspektion	16%	7%
Systemtest	65%	72%

25 | Harry Sneed



## Nutzenanteile der QS-Maßnahmen

Methode	Ersparnis	Fehler
Entwurfsbewertung	10 %	25%
Codeprüfung	7,5%	20%
Funktionstest	12,5%	40%
Insgesamt	30 %	
Restfehler		15%

Nutzenanteil	
Methode	% Nutzen
Entwurfsbewertung	33%
Codeprüfung	25%
Systemtest	42%

26 | Harry Sneed



## Kosten der Qualitätssicherung nach COCOMO

■ Spezifikationskosten	9 - 12,5%
■ Entwurfskosten	7 - 34%
■ Programmierkosten	11 - 50%
■ Testkosten	16 - 32%
■ Gesamtkosten	7 - 50%

Qual = 0,5 : 1,5 Bereich der Qualitätsjustierung

$$E_{MM} = 2,5 (KDS)^{1,05} \times (\text{Prod X Qual})$$

$$E_{MM} = 2,5 (34)^{1,05} \times (1,0 \times 1,4)$$

$$E_{MM} = 100 \times 1,4$$

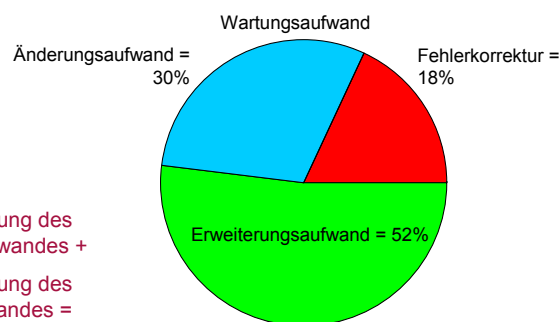
$$E_{MM} = \text{Aufwand} + 40\% = 140 \text{ MM}$$

$$Q_{S_{MM}} = 40 \text{ MM für Qualitätssicherung}$$

27 | Harry Sneed



## Nutzen der Qualitätssicherung nach COCOMO



30% Reduzierung des Änderungsaufwandes +  
18% Reduzierung des Korrekturaufwandes =

**48% Reduzierung der Wartungskosten**

$$W_{MM} = 1,2 (\text{ACT} \times E_{MM}) \times (2,0 - \text{QUAL})$$

$$W_{MM} = 1,2 (0,20 \times 100) \times (2,0 - 1,4)$$

$$W_{MM} = 1,2 (20) \times (0,6)$$

$$W_{MM} = 24 \times 0,6$$

$$W_{MM} = 24 - 40\% = 14,4 \text{ MM jährlich Ersparnis}$$

Nach 3 Jahren hat sich der QS-Aufwand amortisiert.

28 | Harry Sneed



## Indirekter Nutzen der Qualitätssicherung

- Einsparung redundanter Software  
bis zu 85% des Codes nach C. Jones
- Erkennung von Schwachstellen  
nach dem Pareto-Gesetz
- Reduzierung des Wartungsaufwandes  
durch bessere Wartbarkeit, Testbarkeit, Erweiterbarkeit und Übertragbarkeit
- Reduzierung der Fehlerrate  
durch Früherkennung der Probleme und höhere Testdeckung
- Steigerung der Produktivität  
nach dem „Cleanroom“-Ansatz
- Erzwingung der Termintreue  
durch strengere Kontrollen
- Erziehung der Software-Entwickler  
zu mehr Disziplin

29 | Harry Sneed



## Zusammenfassung

- Fehler in der Softwareentwicklung sind unvermeidlich
- Software Fehler verursachen hohe Kosten
- Es komme darauf an, die Fehler möglichst früh zu erkennen
- Der Nutzen der Qualitätssicherung überwiegt die Kosten
- Fehler werden durch diverse QS-Maßnahmen erkannt
- Automation steigert den Nutzen und verringert die Kosten
- Das Endziel ist die automatisierte Qualitätssicherung
- Daher muss mehr in die Entwicklung von Testwerkzeugen investiert werden.

30 | Harry Sneed



ANECON

Danke  
für Ihre  
Aufmerksamkeit

