# Test und Verlässlichkeit (F5)
# Foliensatz 5: Debuggen, Reifeprozesse, Fehlervermeidung, Verlässlichkeitsbewertung

Prof. G. Kemnitz

Institut für Informatik, Technische Universität Clausthal

28. April 2015

# Inhalt F5: Debuggen, Reifeprozesse, ...

# Debuggen

# Fehlerbeseitigung

> Die Reparatur alter Fehler kostet oft mehr als die Anschaffung neuer. (Wieslaw Brudzinski, 1920*)

---

Vorabtests vor der Fehlersuche:

- Fehler oder Störung? ⇒ Fehlverhalten reproduzierbar? Lassen sich Tests finden, die bei Wiederholung dasselbe Fehlverhalten anregen? Fehlerwirkung beseitigbar?
- Entwurfs- oder Fertigungsfehler? ⇒ Unterschiedliches Verhalten baugleicher Systeme?

Möglichkeiten der Fehlerbeseitigung:

- Ersatz

Reparatur (oft Ersatz von Teilsystemen)

# Ersatz

# Ersatz, Fehleranzahl und Fehleranteil

Eine Einheit wird getauscht, wenn sie mindestens einen nachweisbaren Fehler enthält. Zu erwartende Fehleranzahl ungetesteter Systeme nach F2, Abschn.3.2:

$$E(\varphi) = \sum_{i=1}^{N_\varphi} h_i = N_\varphi \cdot \bar{h}$$

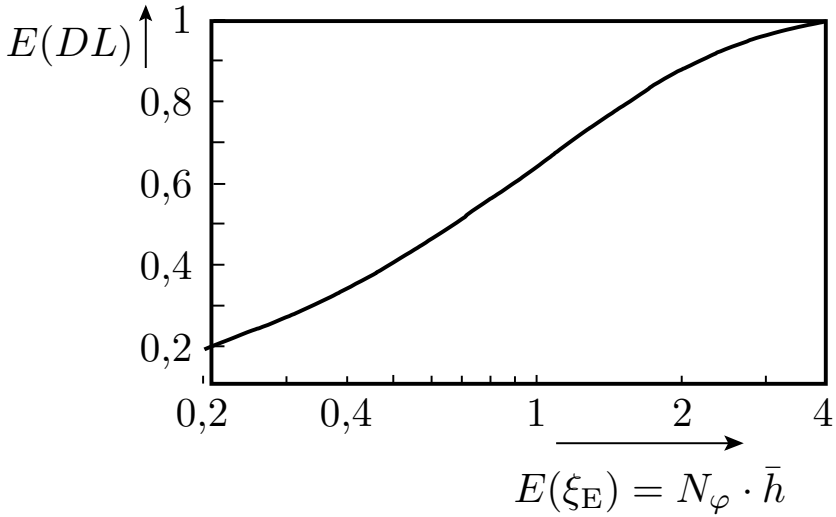
($N_\varphi$ – Anzahl der potentiellen Fehler; $h_i$ – Auftrittshäufigkeiten der einzelnen potenziellen Fehler; $\bar{h}$ – mittlere Fehlerauftrittshäufigkeit). Der zu erwartende Fehleranteil ist die Wahrscheinlichkeit, dass das System mindestens einen Fehler enthält:

$$E(DL) = 1 - \prod_{i=1}^{N_\varphi} (1 - h_i) = 1 - e^{-N_\varphi \cdot \bar{h}}$$

Ersatz ist nur bei Verfügbarkeit fehlerarmer Ersatzteile oder beim Fertigungstest für hinreichend kleine Bausteine mit im Mittel nicht viel mehr als einem Fehler sinnvoll.

$$E\left(DL\right) = 1 - e^{-N_\varphi \cdot \bar{h}}$$

Für den Fertigungstest steht die Anzahl der potenziellen Fehler $N_\varphi$ für die Systemgröße und $\bar{h}$ für die Güte des Entstehungsprozesses. Derzu erwartende Fehleranteil ist gleichzeitig der zu erwartende Ausschuss.




Durch Aussortieren werden alle erkennbaren Fehler beseitigt. Übrig bleiben nur nicht erkannte Fehler:

$$E\left(\varphi_\mathrm{T}\right) = \left(1 - E\left(FC\right)\right) \cdot E\left(\varphi\right)$$

($FC$ – Fehlerüberdeckung). Fehleranteil nach Ersatz erkannter fehlerhafter Systeme:

$$E\left(DL_\mathrm{T}\right) = 1 - e^{-(1-E(FC)) \cdot N_\varphi \cdot \bar{h}}$$

Fehleranteil nach Aussortieren aller fehlerhaften Objekte in dpu (defects per unit) bzw. dpm (defects per million):

| vor dem Test | $E(\varphi) = 2$ | $E(\varphi) = 1$ | $E(\varphi) = 0{,}5$ | $E(\varphi) = 0{,}2$ |
|---|---|---|---|---|
| ohne Test | 0,865 dpu | 0,632 dpu | 0,393 dpu | 0,181 dpu |
| $FC = 70\%$ | 0,45 dpu | 0,26 dpu | 0,14 dpu | 0,058 dpu |
| $FC = 90\%$ | 0,18 dpu | 0,095 dpu | 0,049 dpu | 0,020 dpu |
| $FC = 99{,}7\%$ | 0,058 dpu | 0,030 dpu | 0,015 dpu | 6000 dpm |
| $FC = 99\%$ | 0,020 dpu | 0,010 dpu | 5000 dpm | 2000 dpm |
| $FC = 99{,}7\%$ | 5000 dpm | 3000 dpm | 1500 dpm | 600 dpm |
| $FC = 99{,}9\%$ | 2000 dpm | 1000 dpm | 500 dpm | 200 dpm |

# Fehleranteil und -überdeckung für Schaltkreise

| vor dem Test | $E(\varphi) = 2$ | $E(\varphi) = 1$ | $E(\varphi) = 0{,}5$ | $E(\varphi) = 0{,}2$ |
|---|---|---|---|---|
| Ausbeute | 13,5% | 36,8% | 60,1% | 81,9% |
| $FC = 90\%$ | 0,18 dpu | 0,095 dpu | 0,049 dpu | 0,020 dpu |
| $FC = 99{,}7\%$ | 0,058 dpu | 0,030 dpu | 0,015 dpu | 6000 dpm |
| $FC = 99\%$ | 0,020 dpu | 0,010 dpu | 5000 dpm | 2000 dpm |
| $FC = 99{,}7\%$ | 5000 dpm | 3000 dpm | 1500 dpm | 600 dpm |
| $FC = 99{,}9\%$ | 2000 dpm | 1000 dpm | 500 dpm | 200 dpm |

- Gründlich getestete Schaltkreise haben einen Fehleranteil von 100...1000 dpm (grün gekennzeichnet).
- Verlangt etwa eine Ausbeute $Y \approx 1 - DL = e^{-E(\varphi)}$ von 50% und eine Fehlerüberdeckung von $FC \approx 99{,}9\%$.

| vor dem Test | $E(\varphi) = 2$ | $E(\varphi) = 1$ | $E(\varphi) = 0{,}5$ | $E(\varphi) = 0{,}2$ |
|---|---|---|---|---|
| Ausbeute | 13,5% | 36,8% | 60,1% | 81,9% |
| $FC = 99{,}7\%$ | 5000 dpm | 3000 dpm | 1500 dpm | 600 dpm |
| $FC = 99{,}9\%$ | 2000 dpm | 1000 dpm | 500 dpm | 200 dpm |

Die angestebten Haftfehlerüberdeckungen sind typisch nur $FC_{\text{sa}} \approx 98\%$ und die erzielten Ausbeuten nicht viel größer als 50%. Die Zahlen passen nicht zusammen!

Ansätze für die Forschung:

- Sind die Abschätzungen für den Fehleranteil in der Literatur zu optimistisch?
- Ist der Anteil der nicht nachweisbaren tatsächlichen Fehler eine Zehnerpotenz kleiner als der der Haftfehler und wenn ja, warum?

# Fehleranteil von Bauteilen und Baugruppen

- Gründlich getestete Schaltkreise: 100...1000 dpm = $10^{-4} \ldots 10^{-3}$ dpu
  (dpm – defects per million; dpu – defects per unit)
- Rechner aus $10^2$ Schaltkreisen mit $DL_{\mathrm{IC}} = 10^{-4}$ dpu:
- Wie viele Rechner enthalten (mindestens) einen defekten Schaltkreis?

$$DL = 1 - \left(1 - 10^{-4}\right)^2 \approx 10^{-2}$$

Jeder hundertste Rechner.

---

- Vom Herstellertest übersehene Schaltkreisfehler beeinträchtigen die Funktion fast nicht / kaum zu erkennen.
- Ein HW-Fehler im Einsatz auf 100 Rechner ist eine glaubhafte Größenordnung.

# Experimentelle Reparatur

# Reparatur

Die Reparaturmöglichkeiten legt der Entwurf fest. Einige Prinzipien des reparaturgerechten Entwurfs:

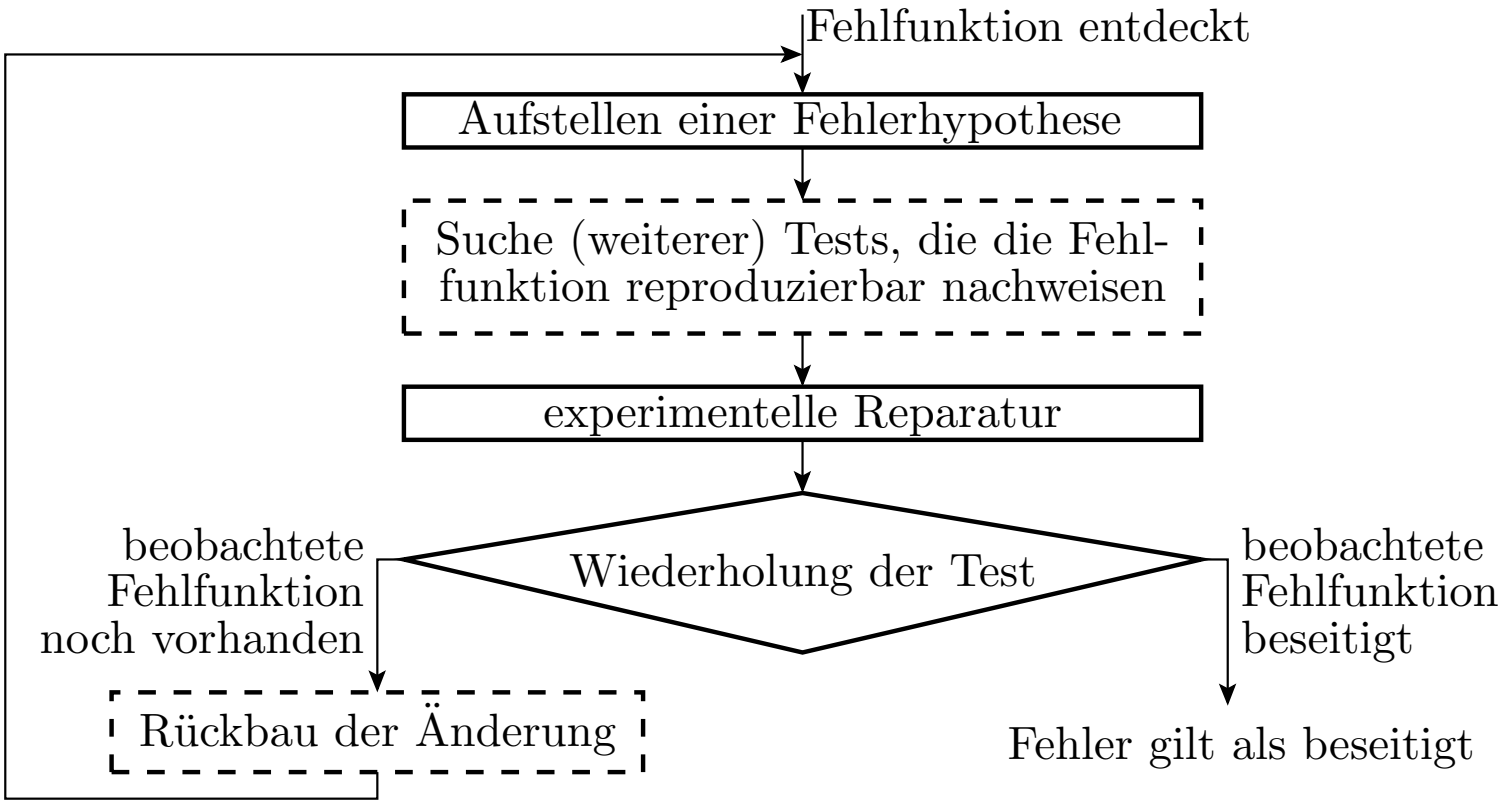
- Zusammensetzten aus austauschbaren Einheiten:
- Software aus Bibliothekselementen, Funktionen, Anweisungen, ...
- Geräte aus Baugruppen, Verbindungen, ...
- Baugruppen aus Bauteilen, ...
- hochintegrierte Schaltkreise: über Programmierelemente abschaltbare Einheiten und zuschaltbare Reserveeinheiten

Unterstützung der Fehlerlokalisierung:

- auftrennbare Steck- und Lötverbindungen, ... (Baugruppen),
- Assertanweisungen, Fehlercodes, ... (Software),
- Einschalttest (Rechner),
- Diagnosebus, lesbarer Fehlerspeicher (Fahrzeugsteuergeräte).

# Experimentelle Reparatur

Intuitives iteratives Vorgehen:




Der Reparaturversuch ist die Kontrolle, ob die aufgestellte Hypothese über die Ursache der Fehlfunktion richtig war.

- Jede Fehlerbeseitigungsiteration startet mit einer beobachteten Fehlfunktion.
- Für die Fehlfunktion werden Testbeispiele festgelegt, die sie nachweisen[1].
- Wiederhole, bis die Fehlfunktion nicht mehr auftritt:
  - Ersatz von Einheiten, Reparatur von Verbindungen, ...[2]
  - Erfolgskontrolle mit Testbeispielen.
    - kein Erfolg: (möglichst) Rückbau vorheriger Zustand[3]
    - Erfolg: Iterationsabbruch.

---

[1] Für beim Test bemerkte Fehlfunktionen ist das der Testsatz. Für im Einsatz erkannte Fehlfunktionen müssen diese u.U. erst konstruiert werden.

[2] Aufwandsminimierung: Zusätzliche Tests zur Einschränkung der Fehlermöglichkeiten und damit der zu erwartenden Iterationsanzahl, z.B. Ausschluss von Unterbrechungen oder Kurzschlüssen durch Widerstandsmessungen. Beginn mit einfachen Reparaturmöglichkeiten. ...

[3] Z.B. Ersatz einer geänderten Programmdatei durch ein Backup, wenn die Programmänderung des Fehlersymptom nicht beseitigt hat. Ohne gewissenhaften Rückbau kann sich durch die Reparaturiterationen die Fehleranzahl vergrößern statt abnehmen.

**Fakt 1**

Eine Fehlerbeseitigung auf Verdacht beseitigt alle erkennbaren Fehler, auch wenn sich der Verdacht, welche Einheit oder Verbindung defekt ist, mehrmals nicht bestätigt.

Für die Fehlerlokalisierung

- genügt eine Erfolgswahrscheinlichkeit weit unter 100%
- wesentlich weniger qualifiziertes Personal als für die Testauswahl.[4]

---

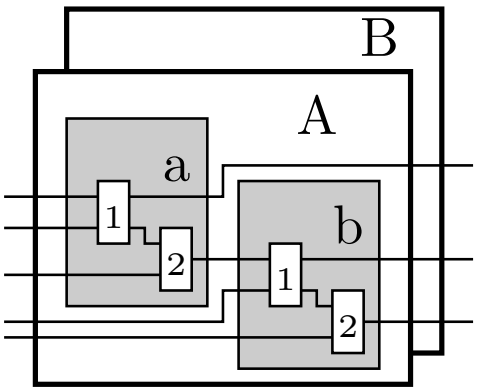
Praktische Lokalisierungstechniken:

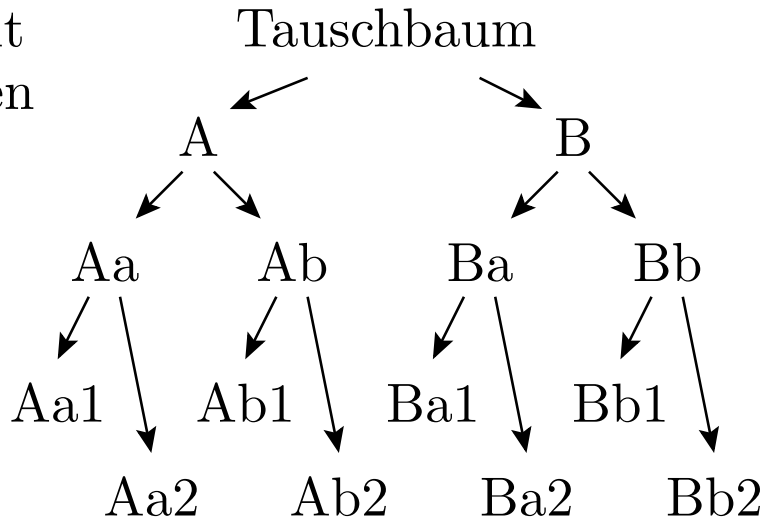
- Systematisches Tauschen.
- Erfahrungsbasierte Reparaturentscheidung.
- Rückverfolgung.

[4] Braucht im Studium deshalb auch nicht unterrichte zu werden.

# Systematisches Tauschen

hierarchisches System mit tauschbaren Komponenten



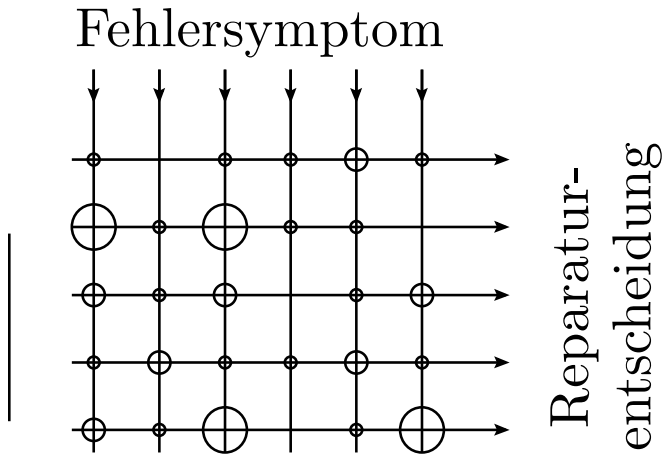




- Nach jedem Tausch, Erfolgskontrolle durch Testwiederholung.
- Ideal: binärer Suchbaum, Tausch der Hälfte, eines Viertel, ... der Komponenten.

# Erfahrungsbasierte Reparaturentscheidung

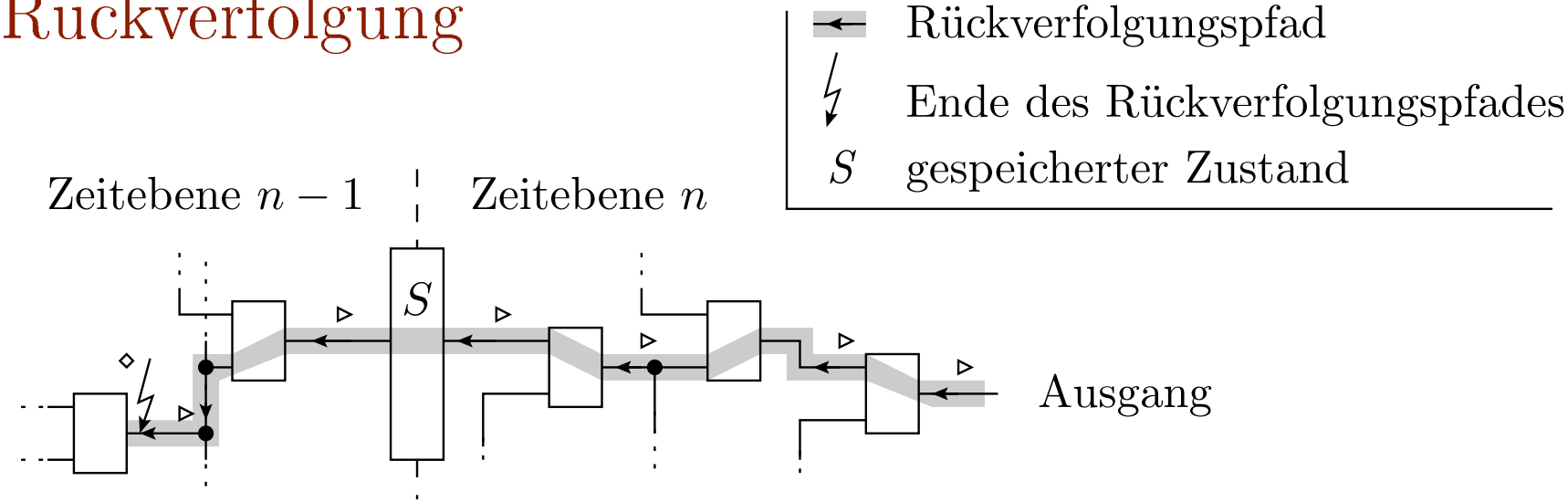
- Pareto-Prinzip: Produkte haben Schwachstellen. Richtwert: 80% der Probleme geht auf 20% der Ursachen zurück.
- Zählen der erfolgreichen und erfolglosen Reparaturversuche.
- Bei Alternativen, Beginn mit der erfolgsversprechenden Reparaturmöglichkeit.

◎ bisherige Häufigkeit, mit der die Reparaturentscheidung für das Symptom richtig war




- Nach erfolglosen Reparaturversuchen Vorzustand wieder herstellen.

# Rückverfolgung




- Aufzeichnung der Zeitverläufe potenziell verfälscher Signalverläufe oder Variablenwerte (Simulation, Logikanalyse, Programm-Trace)
- Ausgehend von einer erkannten falschen Ausgabe Rückwärtsuche nach dem Entstehungsort.
- Entstehungsort: Funktionsbaustein, der aus richtigen Eingaben falsche Ausgaben erzeugt.

Am Fehlerort erfahrungsbasierte Reparaturentscheidung:

- Nicht unbedingt der Funktionsbaustein mit richtigen Eingaben und falschen Ausgaben verursacht den Fehler.
- Weitere potenzielle Ursachen:
  - Kurzschluss zu einem anderen Signal, Unterbrechung, defekter Schaltkreiseingang, ... (Baugruppen),
  - Schreiboperation auf eine falsche Variable, ... (Programme).
- Spezielle Fehlerausschlusstests z.B.
  - Widerstandsmessungen zwischen und entlang von Verbindungen.
  - Suche nach falscher Schreiboperation auf die Variable.

# Reparatur der Hardware eines PCs

Typisches Mechaniker-Vorgehen:

- Grobabschätzung, welcher Rechnerteil defekt sein könnte aus den Fehlersymptomen.
- Kontrolle der Steckverbinder auf Kontaktprobleme durch Abziehen, Reinigen, Zusammenstecken, Ausprobieren.[5]
- Tausch möglicherweise defekter Baugruppen gegen Ersatzbaugruppen, Ausprobieren, ...

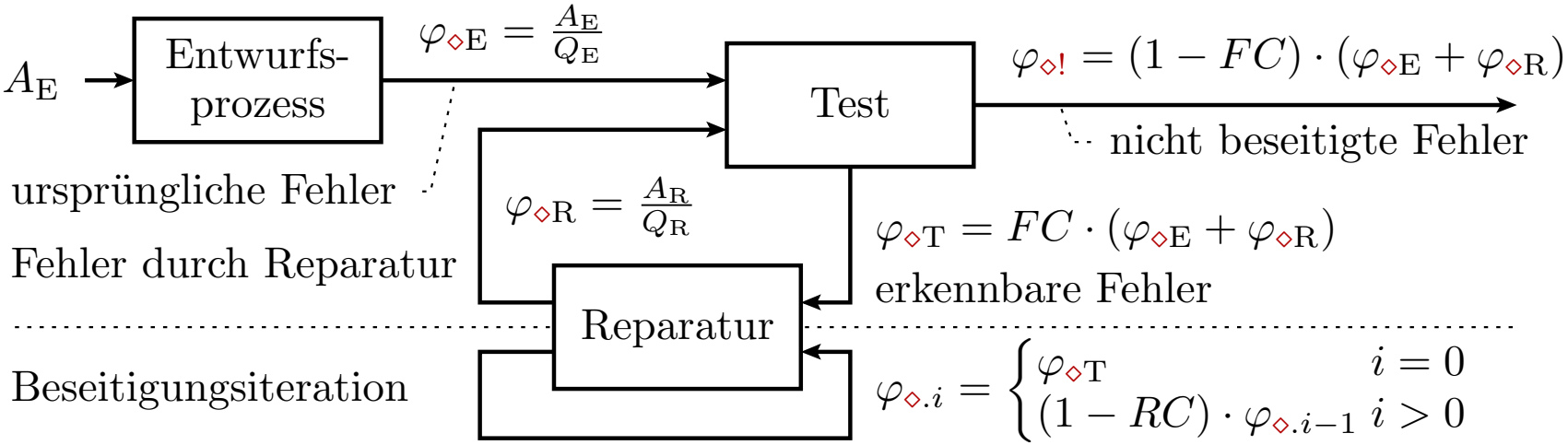
---

Voraussetzungen:

- Wiederholbare Tests, die den Fehler nachweisen.
- Ausreichend Ersatzteile.
- Verlangt nur allgemeine Mechnikerkenntnisse, aber keine Kenntnis der Funktionsweise des zu reparierenden Systems.

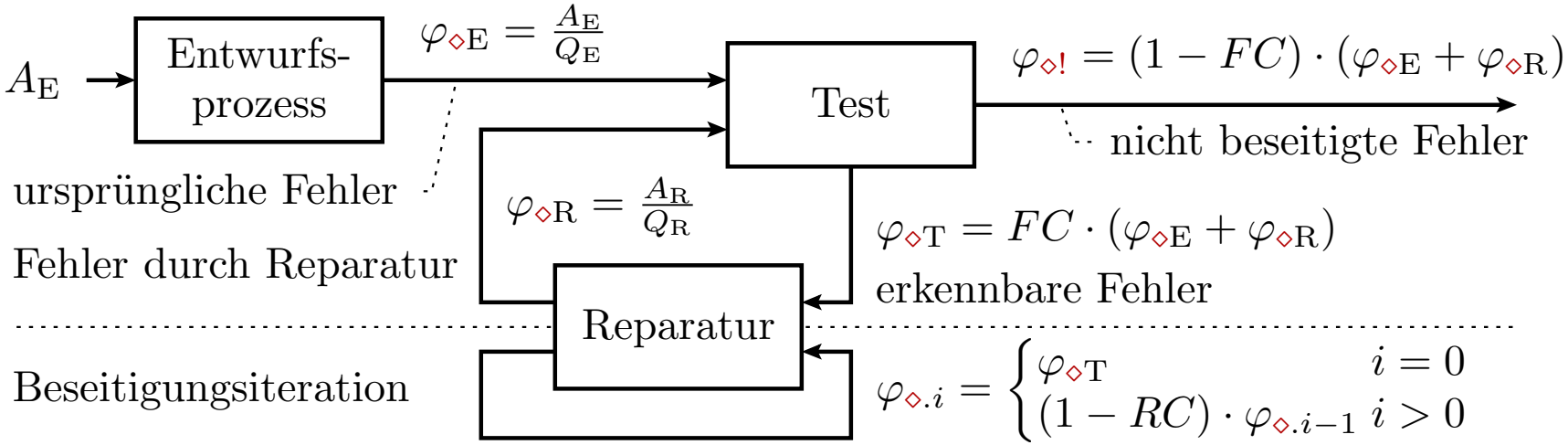
[5] Ein cleverer Mechaniker bauen getauschte vermutlich ganze Teile statt in denselben, in einen anderen Rechner ein. Warum sollte das unterbleiben?

# Abschätzung der Anzahl nicht beseitigten Fehler




| | | | |
|---|---|---|---|
| $A_E$ | Entwurfsaufwand | $Q_R$ | Güte des Reparaturprozesses |
| $A_R$ | Reparaturaufwand | $FC$ | Fehlerüberdeckung des Tests |
| $Q_E$ | Güte des Entwurfsprozesses | $RC$ | Erfolgsrate der Reparatur |

- Die Fehleranzahl sei proportional zum Aufwand und umgekehrt proportional zur Prozessgüte.
- Fehler, die der Test durchlässt, erreichen nie die Reparatur
- Nachweisbare Fehler werden iterativ beseitigt.
- Je mehr Beseitigungsiterationen, desto mehr neue Fehler.




- Fehleranzahl je Beseitigungsschritt $\Rightarrow$ geometrische Reihe:

$$\varphi_{\diamond .i} = (1 - RC)^i \cdot \varphi_{\diamond T}$$

- Anzahl der Reparaturen gleich Summe aller $\varphi_{\diamond .i}$:

$$A_R = \sum_{i=0}^{\infty} \varphi_{\diamond .i} = \varphi_{\diamond T} \cdot \sum_{i=0}^{\infty} (1 - RC)^i = \frac{\varphi_{\diamond T}}{RC}$$

- Anzahl der durch Reparaturen verursachten Fehler

$$\varphi_{\diamond R} = \frac{\varphi_{\diamond T}}{Q_R \cdot RC} = \frac{FC \cdot (\varphi_{\diamond E} + \varphi_{\diamond R})}{Q_R \cdot RC}$$

$$\varphi_{\diamond \mathrm{R}} = \frac{\varphi_{\diamond \mathrm{T}}}{Q_\mathrm{R} \cdot RC} = \frac{FC \cdot (\varphi_{\diamond \mathrm{E}} + \varphi_{\diamond \mathrm{R}})}{Q_\mathrm{R} \cdot RC}$$

$$Q_\mathrm{R} \cdot RC = \frac{\text{Anz. Reparaturen}}{\text{neue Fehler}} \cdot \frac{\text{beseitigte Fehler}}{\text{Anz.Reparaturen}} = \frac{\text{beseitigteFehler}}{\text{neue Fehler}}$$

---

- Wenn die Anzahl der nachweisbaren Fehler abnimmt:

$$Q_\mathrm{R} \cdot RC > FC$$

- Gesamtanzahl der durch Reparatur verursachen Fehler:

$$\varphi_{\diamond \mathrm{R}} = \varphi_{\diamond \mathrm{E}} \cdot \frac{FC}{Q_\mathrm{R} \cdot RC - FC}$$

- Ursprüngliche + reparaturbedingte Fehler

$$\varphi_{\diamond \mathrm{E}} + \varphi_{\diamond \mathrm{R}} = \varphi_{\diamond \mathrm{E}} \cdot \left(1 + \frac{FC}{Q_\mathrm{R} \cdot RC - FC}\right) = \varphi_{\diamond \mathrm{E}} \cdot \frac{Q_\mathrm{R} \cdot RC}{Q_\mathrm{R} \cdot RC - FC}$$

- Gesamtanzahl der nicht beseitigten Fehler $(\ldots \cdot (1 - FC))$

$$\boxed{\varphi_{\diamond !} = \varphi_{\diamond \mathrm{E}} \cdot \frac{(1-FC) \cdot Q_\mathrm{R} \cdot RC}{Q_\mathrm{R} \cdot RC - FC}}$$

# Idealer Reparaturprozess

- Es werden viel mehr Fehler beseitigt als neu eingebaut:

$$Q_\mathrm{R} \cdot RC = \frac{\text{beseitigteFehler}}{\text{neue Fehler}} \gg 1$$

Fehlerbeseitigung ist so gut wie der Test

$$\varphi_{\diamond !} = \varphi_{\diamond \mathrm{E}} \cdot \frac{(1-FC) \cdot Q_\mathrm{R} \cdot RC}{Q_\mathrm{R} \cdot RC - FC} \approx (1-FC) \cdot \varphi_{\diamond \mathrm{E}}$$

---

Wenn im Mittel für jeden beseitigten Fehler ein neuer eingebaut wird: $Q_\mathrm{R} \cdot RC = 1$

$$\varphi_{\diamond !} = \varphi_{\diamond \mathrm{E}} \cdot \frac{(1-FC) \cdot Q_\mathrm{R} \cdot RC}{Q_\mathrm{R} \cdot RC - FC} = \varphi_{\diamond \mathrm{E}} \cdot \frac{(1-FC) \cdot 1}{1-FC} = \varphi_{\diamond \mathrm{E}}$$

- Werden alle nachweisbaren Fehler beseitigt und
- entstehen etwas genauso viele nicht nachweisbare Fehler.

# Typische studentische Programmierarbeiten

$$\boxed{\varphi_{\diamond !} = \varphi_{\diamond \mathrm{E}} \cdot \frac{(1-FC)\cdot Q_\mathrm{R}\cdot RC}{Q_\mathrm{R}\cdot RC - FC}}$$

---

Fall A: wenige Testbeispiele, brauchbarer Reparaturprozess

- Beispiel: $FC = 30\%$ erkennbare Fehler, $Q_\mathrm{R} \cdot RC = 2$ beseitigte je neuer Fehler

$$\varphi_{\diamond !} = \varphi_{\diamond \mathrm{E}} \cdot \frac{(1-0.,3)\cdot 2}{2-0.,3} \approx 82\% \cdot \varphi_{\diamond \mathrm{E}}$$

- Reduktion der Fehleranzahl auf 82%. Davon sind 70% nicht erkannte ursprüngliche und $12\% \cdot \varphi_{\diamond \mathrm{E}}$ bei der Reparatur entstandene Fehler.
- Erkannt und beseitigt werden die am meisten störenden Fehler (siehe Zufallstest). Es bestehen Chancen, dass das System einen Abnahmetest mit 1 bis 2 neuen zufälligen Testbeispielen erfolgreich passiert.

$$\boxed{\varphi_{\diamond !} = \varphi_{\diamond \mathrm{E}} \cdot \frac{(1-FC)\cdot Q_\mathrm{R}\cdot RC}{Q_\mathrm{R}\cdot RC - FC}}$$

---

Fall B: Entwurf wird beherrscht, aber Test und Reparaturtechniken nicht

- Beispiel: $FC = 25\%$ erkennbare Fehler, $Q_\mathrm{R} \cdot RC = 0{,}5$ beseitigte je neuer Fehler

$$\varphi_{\diamond !} = \varphi_{\diamond \mathrm{E}} \cdot \frac{(1-0,25)\cdot 0,5}{0,5-0,25} = 2{,}5 \cdot \varphi_{\diamond \mathrm{E}}$$

- System enthält nach Test und Fehlerbeseitigung viel mehr Fehler als zuvor.
- Reparatur versteckt die Fehler (ersetzt alle erkennbaren durch nicht erkennbare Fehler); Testbeispiele entwertet
- Begleitsymtom: übermäßig lange Test- und Reparaturphase.
- Abnahmetest mit 1 bis 2 neuen zufälligen Testbeispielen findet in der Regel Fehler.

$$\boxed{\varphi_{\diamond !} = \varphi_{\diamond \mathrm{E}} \cdot \frac{(1-FC)\cdot Q_\mathrm{R}\cdot RC}{Q_\mathrm{R}\cdot RC - FC}}$$

---

Fall C: Studierender ist mit seiner Aufgabe überfordert

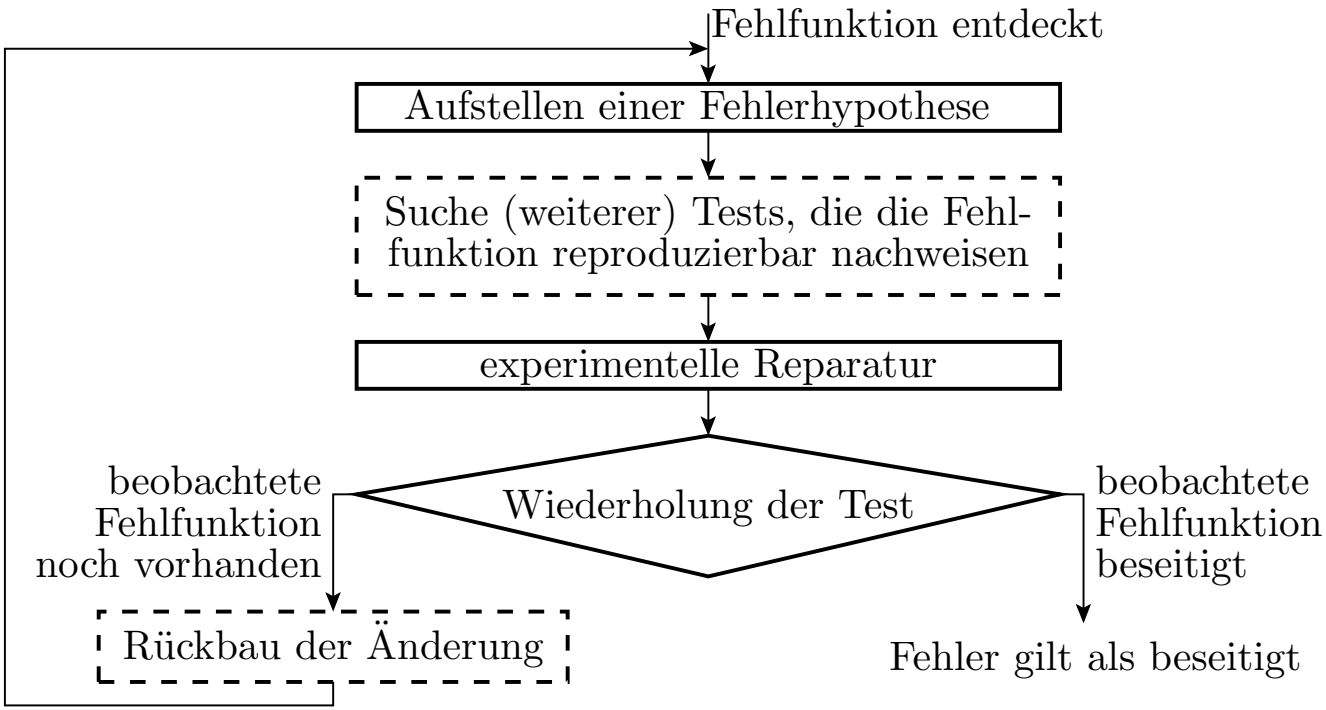
$$FC > Q_\mathrm{R} \cdot RC$$

- Zunahme der Anzahl der nachweisbaren Fehler mit Fortschreiten der Beseitigungsversuche.
- Projekt wird nie fertig.

---

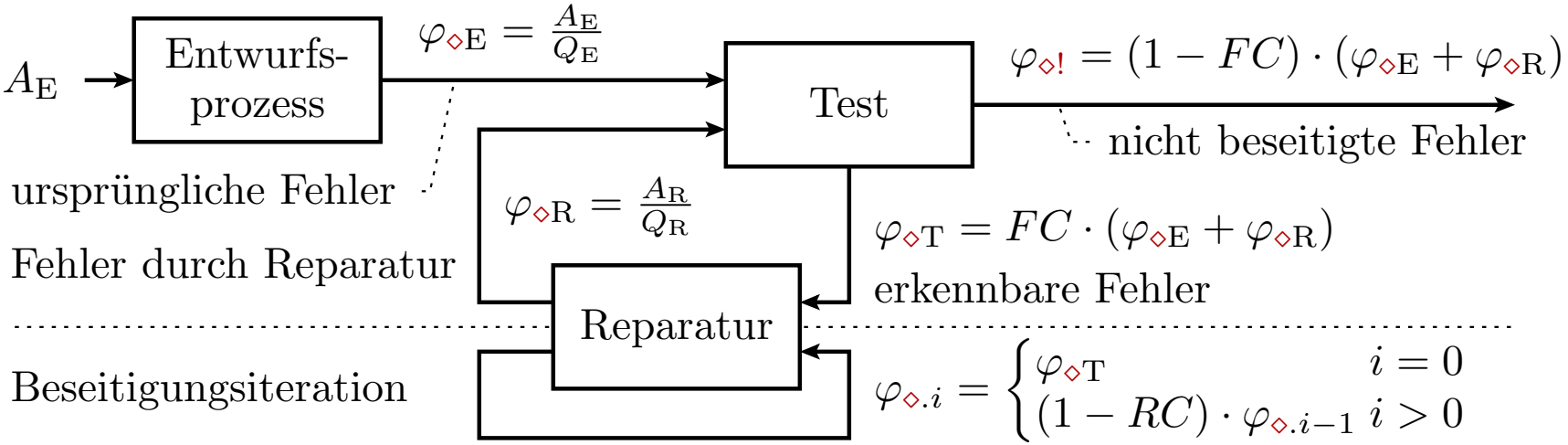
Vor der Übertragung einer Entwicklungsaufgabe sollte der personengebundenen Parameter $Q_\mathrm{R} \cdot RC$ (beseitigte Fehler je neuer Fehler) kontrolliert und, wenn schlecht, Einarbeitungszeit verlängert werden.

# Reparaturempfehlungen




Zur Minimierung der Iterationszahl:

- Ausschluss von Hypothesen durch zusätzliche Tests.
- Reparaturentscheidungen so wählen, dass bei Nicht-Erfolg weitere Hypothesen ausgeschlossen werden. ...




Fehlervermeidung bei der Reparatur:

- Sorgfältig reparieren (hohe Güte $Q_R$ und hohe Erfolgsrate $RC$ des Reparaturprozesses).
- Sorgfältiger Rückbau. (Beseitigt auch entstandene Fehler, die der Test nicht erkennt.)
- Zahl der erfolglosen Reparaturversuche je Fehler begrenzen, z.B. auf drei, dann Ersatz.
- Von Systemen mit sehr vielen Änderungen nur die Zielfunktion und die Testbeispiele übernehmen. System selbst neu entwerfen.

Regel für den Einkauf von IT-Systemen:

- Zur Kontrolle, ob der Hersteller eine akzeptable Prüf- und Reparaturtechnologien hat, neu angeschaffte IT-System mit einer Stichprobe zufälliger Eingaben testen.
- Wenn ein System diesen Test nicht besteht, zurückgeben und von diesem Hersteller nie wieder etwas kaufen. Denn das ist ein sicherer Hinweis darauf, dass dieser Hersteller seine Entwurfs- und Reparaturprozesse nicht beherrscht.

Akzeptiere nie ein IT-System, ohne vorher selbst gewählte, dem Entwerfer unbekannte Tesetbeispiele auszuprobieren.

# post mortem

# in the large

# Aufgaben

# Aufgabe 1.1: Fehleranteil

1. Die zu erwartende Fehleranteil eines Systemtyps sei vor dem Test 0,7 dpu und nach dem Test 0,01 dpu. Welche Fehlerüberdeckung hat der Testsatz unter der Annahme, dass die Fehleranzahl vor und nach dem Test poison-verteilt ist?
2. Ein Schaltkreistest hat 60% aller gefertigten Schaltkreise als fehlerhaft aussortiert. Aus dem Garantierückläufen seitens der Anwender wurde für die getesteten Schaltkreise ein Fehleranteil von 200 dpm abgeschätzt. Auf welche Ausbeute und auf welche Fehlerüberdeckung lassen diese Zahlen schließen, wenn eine poisson-verteile Fehleranzahl unterstellt wird?

# Aufgabe 1.2: Input Workaround

1. Beschreiben Sie in der Programmiersprache C einen Work-Around zur Ausführung der Anweisung

```
int a, b, c;
...
a = b * c;
```

für einen Rechner, der bei einer Multiplikation mit null aufgrund eines Fehlers eins berechnet so, dass das Programm auch funktioniert, wenn der Fehler in späteren Rechnergenerationen nicht mehr vorhanden ist.

2. Welche Dokumente eines Software-Prototypentwurfs, an denen sehr viel geändert und in dem bereits sehr viele Fehler beseitigt wurden, sollten weiterverwendet und welche neu geschrieben werden?

# Aufgabe 1.3: Fehlerlokalisierung

Für die nachfolgende Schaltung wurden die Signalwerte in der Tabelle während des Tests aufgezeichnet.

1. Welche Bauteile oder Verbindungen könnten die Ursache der Fehlfunktion sein?
2. Wie ließen sich die Fehlermöglichkeiten vor dem ersten Reparaturversuch weiter einschränken?

# Aufgabe 1.4: Reparaturprozess

Ein Student macht angenommen beim Programmieren im Mittel 5 Fehler auf Hundert Programmzeilen, die weder vom Syntaxtest noch von den gewählten Testbeispielen erkannt werden. Auf jeden nicht erkannten Fehler kommt im Mittel ein Syntaxfehler und ein erkannter semantischer Fehler. Die Beseitigung eines erkannten Fehlers erfordert im Mittel drei Versuche. Bei jedem zweiten Versuch entsteht ein neuer Fehler. Von den neuen Fehlern wird ein Drittel vom Syntaxtest, ein Drittel von den Testbeispielen und ein Drittel nicht erkannt. Wie hoch ist die zu erwartende Fehleranzahl nach Beseitigung aller erkennbaren Fehler eines 200 Codezeilen großen Programms?

# Aufgabe 1.5: Fehlerbeseitigungsiteration

Wie groß ist die zu erwartende Fehleranzahl nach einer Iteration aus Test und Fehlerbeseitigung?

- Fehleranzahl vor der Fehlerbeseitigungsiteration 20;
- Fehlerüberdeckung $FC = 60\%$ und
- im Mittel einem neu entstehenden Fehler je fünf Beseitigungsversuche.

Unter welcher Bedingung nimmt die Anzahl der nachweisbaren Fehler in einer Fehlerbeseitigungsiteration denoch ab, wenn bei jedem erfolgreichen Beseitigungsversuch im Mittel 1,2 neue Fehler in das System eingebaut werden?

# Aufgabe 1.6: Fehleranteil Baugruppen

1. Eine Leiterplatte wird aus folgenden Bauteilen mit bekanntem Fehleranteil zusammengesetzt. Wie hoch ist der Fehleranteil der Baugruppe nach Beseitigung aller Verbindungsfehler?

| Typ | Anzahl | Fehleranteil |
|---|---|---|
| Widerstände, Kondensatoren, ... | | |
| Schaltkreise | | |
| Steckkontakte | | |
| Leiterplatte | | |

2. Begründen Sie aus Reparatursicht, warum auf Baugruppen auch manchmal Teilbaugruppen als vorgefertigte, seperat testbare Module gesteckt werden.

# Aufgabe 1.7: Fehleranteil Rechner

Ein Rechner besteht aus Leiterplatten, Schaltkreisen, diskreten Bauteilen (Widerstände, Kondensatoren, ...) und Lötstellen.

| Typ | Anzahl | $E(DL_{\mathrm{BT}})$ |
|---|---|---|
| Leiterplatten | 10 | 10 dpm |
| Schaltkreise | 100 | 200 dpm |
| diskrete Bauteile | 200 | 10 dpm |
| Lötstellen | 10000 | 1 dpm |

Fehleranteil des gesamten Rechners [6]?

[6] Für die Bauteile und den kompletten Rechner sei unterstellt, dass die Fehleranzahl poisson-verteilt ist.

# Aufgabe 1.8: Objektüberdeckung, Ausbeute, ...

- Ist die Objektfehlerüberdeckung größer oder kleiner als die Fehlerüberdeckung?
- Die Objektfehlerüberdeckung sei 50% und die Ausbeute 80%. Wie hoch ist der Fehleranteil, wenn die Fehleranzahl Poisson-verteilt ist?
- Für einen bestimmten Schaltkreistyp, z.B. Speicherschaltkreise sei die Ausbeute 80%. Um welchen Faktor kann die Chipfläche vergrößert werden, damit die Ausbeute nicht unter 20% absinkt? Annahmen: Die Fehleranzahl sei Poisson-verteilt und verhalte proportional zu Chip-Fläche.

# Reifeprozesse

# Zuverlässigkeitswachstumsprozess

- Zufallstest, beim dem sich die Nutzungsdauern bei allen Anwendern, bei denen die bobachteten Fehlfunktionen erfasst werden, als Testdauer akkumulieren.
- Erlaubt um Zehnerpotenzen längere Testdauern.
- Erfordert Kontrollfunktionen im System und organisatorische Maßnahmen, die die aufgetretenden Fehlfunktionen erfassen und an den Hersteller zur Erstellung von Tests für ihren Nachweis weiterleiten.
- Die Fehlerbeseitigung ist unsicherer, als wenn der Hersteller selbst testet. Beseitigungswahrscheinlichkeit für nachweisbare Fehler $p_{\mathrm{B}} \ll 1$ (z.B. 10%).

Die Zuverlässigkeit in Service-Anforderungen je Fehlfunktion

$$Z = N_{\xi} / E\left(\xi_{\mathrm{F}}\right)$$

nimmt trotzdem zu.

Die zu erwartende Anzahl der Fehlfunktionen je $N_\xi$ Servic-Anforderung ist die Summe der Auftrittshäufigkeiten mal der Nachweiswahrscheinlichkeit je Service-Anforderung aller potenzieller Fehler $N_\varphi$:

$$\frac{E\left(\xi_\mathrm{F}\right)}{N_\xi} = E\left(\varphi\right) \cdot \sum_{i=1}^{N_\varphi} h_i \cdot p_i$$

und beträgt ausgedrückt durch die Fehlernachweisdichte

$$\frac{E\left(\xi_\mathrm{F}\right)}{N_\xi}\left(\xi_\mathrm{F}\right) = E\left(\varphi\right) \cdot \int_0^1 h\left(p\right) \cdot p \cdot dp \tag{1}$$

Die Wahrscheinlichkeit, dass ein Fehler nach einem Service-Aufruf beseitigt wird, ist die Wahrscheinlichkeit, dass er nachgewiesen, und wenn nachgewiesen, auch beseitigt wird:

$$p \cdot p_\mathrm{B}$$

Die Beseitigungswahrscheinlichkeit bei Abarbeitung von $n$ Service-Leistungen ist:

$$1 - e^{-n \cdot p \cdot p_\mathrm{B}}$$

Änderung der Fehlernachweisdichte mit $n$:

$$h\left(p,n\right)=\frac{h\left(p\right)\cdot e^{-n\cdot p\cdot p_{\mathrm{B}}}}{\int_0^1 h\left(p\right)\cdot e^{-n\cdot p\cdot p_{\mathrm{B}}}\cdot dp}$$

($h\left(p\right)$ – Fehlernachweisdichte des ungetesteten Systems). Gegenüber F2, Abschn. 3.3 ist, wenn nur ein Anteil $p_{\mathrm{B}}$ der erkennbaren Fehler beseitigt wird, $n$ durch $n\cdot p_{\mathrm{B}}$ zu ersetzen:

$$E\left(\xi_{\mathrm{F}}\right)=N_{\xi}\cdot E\left(\varphi_{\mathrm{E}}\right)\cdot\int_0^1 h\left(p\right)\cdot p\cdot e^{-n\cdot p_{\mathrm{B}}\cdot p}\cdot dp$$

Mit der Potenzfunktion

$$h\left(p\right)=k\cdot p^{k-1}$$

ergibt sich für die zu erwartende Anzahl der durch Fehler verursachten Fehlfunktionen:

$$\begin{aligned}E\left(\xi_{\mathrm{F}}\right) &= N_{\xi}\cdot E\left(\varphi_{\mathrm{E}}\right)\cdot\int_0^1 k\cdot p^{k-1}\cdot p\cdot e^{-n\cdot p_{\mathrm{B}}\cdot p}\cdot dp\\ &= N_{\xi}\cdot E\left(\varphi_{\mathrm{E}}\right)\cdot\int_0^1 k\cdot p^{k}\cdot e^{-n\cdot p_{\mathrm{B}}\cdot p}\cdot dp\end{aligned}$$

Substitution $p = \frac{x}{p_B \cdot n}$ und $dp = \frac{dx}{p_B \cdot n}$

$$E(\xi_F) = \frac{k \cdot N_\xi \cdot E(\varphi_E)}{(p_B \cdot n)^{k+1}} \cdot \underbrace{\int_0^n x^k \cdot e^{-x} \cdot dx}_{\approx \Gamma(k+1) \approx 1 \text{ für } 0 < k \leq 1}$$

Die Zuverlässigkeit in Service-Leistungen pro Fehlfunktion wächst überproportional mit dem Produkt $p_B \cdot n$:

$$Z(n) = \frac{N_\xi}{E(\xi_F)} = \frac{(p_B \cdot n)^{k+1}}{k \cdot E(\varphi_E)}$$

Mit der Zuverlässigkeit einer Bezugsanzahl von Service-Aufrufen $n_0$:

$$Z(n) = Z(n_0) \cdot \frac{(p_B \cdot n)^{k+1}}{(p_B \cdot n_0)^{k+1}} = Z(n_0) \cdot \left(\frac{n}{n_0}\right)^{k+1}$$

Die Zuverlässigkeit nimmt überproportional mit der Anzahl der Service-Aufrufe, bei denen erkannte Fehlfunktionen Fehlerbeseitigungsversuche auslösen, zu.

# Nutzungsdauer und Zuverlässigkeit

Unter der Annahme, dass sich die Anzahl der Service-Aufrufe $n$ proportional zur Reifezeit[7] $t$ verhält:

$$Z(t) = Z(t_0) \cdot \left(\frac{t}{t_0}\right)^{k+1}$$

$t_0$ und $k$ – Modellparameter, vorzugsweise abzuschätzen aus der zu beobachtenden Zunahme der Zeit zwischen den beobachteten Fehlfunktionen.

[7] Große IT-Systeme müssen nach ihrer Entstehung eine Weile reifen, bevor sie ausreichend zuverlässig für den Einsatz sind. Eine Strategie hierfür ist die kostenlose Freigabe als Beta-Software.

# Abschätzung der noch erforderlichen Reifezeit

Die Zuverlässigkeit eines Systems sei nach eine Reifedauer von 1 Jahr 10 Stunden pro Fehlfunktion. Wie lange muss das System noch reifen, um die Zuverlässigkeit auf 50 Stunden je Fehlfunktion zu erhöhen? Schätzwert für $k \approx 0{,}5$.
Lösung:

$$t = 1\,\text{Jahr} \cdot \left(\frac{50\,\text{h}}{10\,\text{h}}\right)^{\frac{1}{1,5}} = 2,92\,\text{Jahre}$$

Das System müsste noch zwei Jahre weiter reifen.

# Überwachung eines Reifeprozesses

Gegeben: Für $i = 1$ bis 4 Versionen eines Softwareprodukts Erscheinungsdatum, die Zeitdifferenz $t_i$ zur Herausgabe der ersten Version und die im Mittel pro Woche gezählten Fehlfunktionen $\zeta_i$ für zehn eingesetzte Systeme:

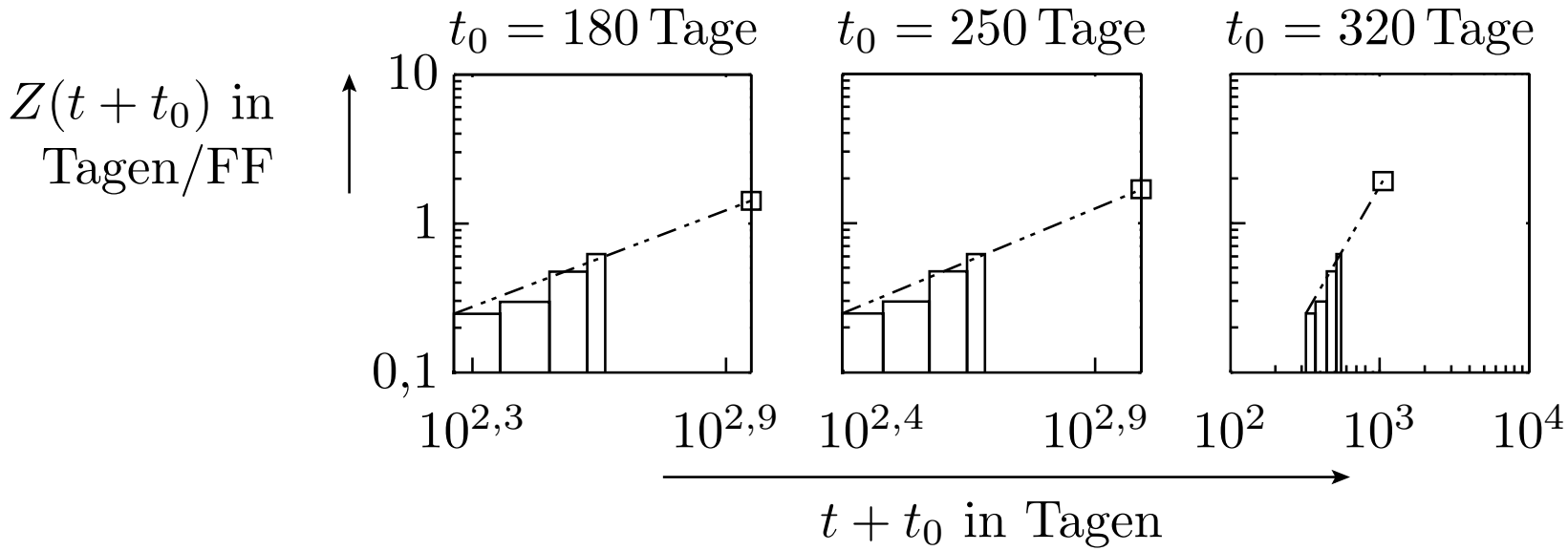
| $i$ | 1 | 2 | 3 | 4 | Ende |
|---|---|---|---|---|---|
| Datum | 01.07.11 | 22.08.11 | 01.11.11 | 10.01.12 | 30.01.12 |
| $\zeta_i$ | 290,5 | 238,4 | 147,0 | 32,3 | |

Gesucht: $t_0$ und $k$ zur Modellierung des Reifeprozesses.

Umrechnung die Daten in Reifezeiten und Zuverlässigkeiten:

| $i$ | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| $t_i$ in Tagen | $t_0$ | $t_0 + 52$ | $t_0 + 123$ | $t_0 + 193$ |
| $Z(t_i)$ in Tagen | 0,248 | 0,298 | 0,476 | 0,619 |

Vorgabe von $t_0$ und Abschätzung von $k$ aus dem Anstieg in der doppellogarithmischen Darstellung der Zuverlässigkeit als Funktion der Testdauer:




| Approx. | $t_0$ | $k$ | $Z\,(t_0 + 2\,\text{Jahre})$ |
|---|---|---|---|
| 1 | 180 Tage | 0,08 | 1,42 Tage |
| 2 | 250 Tage | 0,68 | 1,68 Tage |
| 3 | 320 Tage | 0,92 | 1,92 Tage |

Mit den Schätzwerten kann die Zuverlässigkeit zu einem späteren Zeitpunkt geschätzt werden, im Bsp. für eine Version, die 2 Jahre nach der ersten erscheint. Erstaunlicherweise sind die Ergebnisse für alle drei Paramtersätze sehr ähnlich.

| Approx. | $t_0$ | $k$ | $Z\,(t_0 + 2\,\text{Jahre})$ |
|---|---|---|---|
| 1 | 180 Tage | 0,08 | 1,42 Tage |
| 2 | 250 Tage | 0,68 | 1,68 Tage |
| 3 | 320 Tage | 0,92 | 1,92 Tage |

**Fakt 2**

Es ist möglich, mit Modellrechungen auf ein Zuverlässigkeitswachstum zu schließen. Das erfordert Annahmen über die Nachweiseigenschaften der nicht gefundenen Fehler. Der Ansatz, eine Potenzfunktion als Fehlernachweisdichte zu nehmen, erscheint vielversprechend.

# Modell von Musa bzw. Goel-Okumoto [2]

Am häufigsten zitiertes Zuverlässigkeitswachstumsmodell.
Unterstellter Zusammenhang für die Anzahl der nachweisbaren Fehler in Abhängigkeit von der Test- oder Reifezeit $t$:

$$\varphi(t) = a\left(1 - e^{-bt}\right)$$

($a$, $b$ – experimentell zu bestimmende Parameter). Was für eine Fehlernachweisdichte müsste das IT-System haben?

$$\varphi(t) = a\left(1 - e^{-b \cdot t_{\mathrm{T}}}\right) \Rightarrow E\left(\varphi_{\mathrm{N}}(n)\right) = E\left(\varphi_{\mathrm{E}}\right) \cdot \int_0^1 h(p) \cdot \left(1 - e^{-n \cdot p}\right) \cdot dp$$

Das Modell unterstellt offenbar, dass alle Fehler mit gleicher Wahrscheinlichkeit nachweisbar sind:

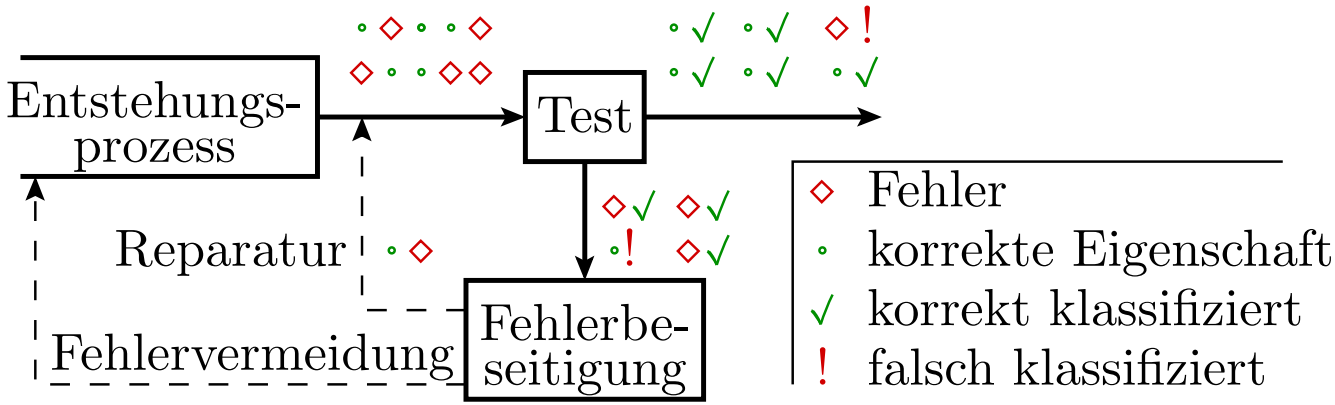
$$h(p) = \begin{cases} 1 & \text{für } p = b \\ 0 & \text{sonst} \end{cases}$$

Untypisch für IT-Systeme!

# Fehlervermeidung

# Fehlervermeidung

Die Fehler in einem IT-System entstehen mit dem System.




In (annähernd) reproduzierbar ablaufenden Entstehungsprozessen lassen sich erkannte Ursachen für die Fehlerentstehung abstellen. Die Fehlervermeidungsiteration umfasst

- Kontrollen der Prozessschritte und Produkte,
- Verbesserung der Reproduzierbarkeit,
- Lokalisierung von Fehlerentstehungsursachen und
- Beseitigung erkannter Schwachstellen und Prozessfehler.

**Fakt 3**

Fehlervermeidung ist eine Reifeprozess für Entstehungsprozesse.

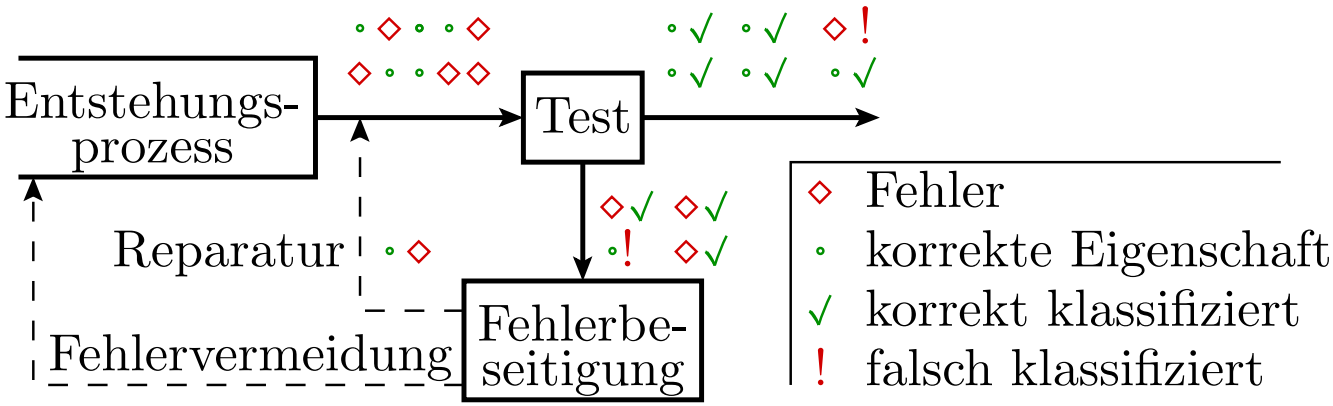
## Das besondere an Entwurfsprozessen

Entwurfsprozesse sind projektorientiert und enthalten einen hohen Anteil kreativer Handarbeit.

- Projekt: Einmaliges Vorhaben ... zur Erreichung eines Ziels. Steht im Widerspruch zu den Voraussetzungen für Reifeprozesse »Wiederholt gleiches Vorgehen ...«.
- Kreativität steht auch im Widerspruch zu einem anzustebenden reproduzierbaren Ablauf.
- Bei Handarbeit entstehen mehr und vielfältigere Fehler als bei automatisierten Abläufen.

Fehlervermeidung für Entwürfe bewegt sich deshalb oft auf der Vorstufe, Verbesserung der Verhersagbarkeit des Zeitaufwands, der Kosten, der Systemgröße, der

# Fehlervermeidung




Fehlervermeidung umfasst

- die Kontrollen der Prozessschritte und Produkte,
- Verbesserung der Reproduzierbarkeit,
- Lokalisierung von Fehlerentstehungsursachen und
- Beseitigung erkannter Schwachstellen und Prozessfehler.

Fehlervermeidung ist eine Reifeprozess, der viele Entstehungsprozesse (Fertigungsprozesse, Entwurfsprozesse) wärend ihrer gesamten Existenz begleitet.

# Deterministische Prozesse

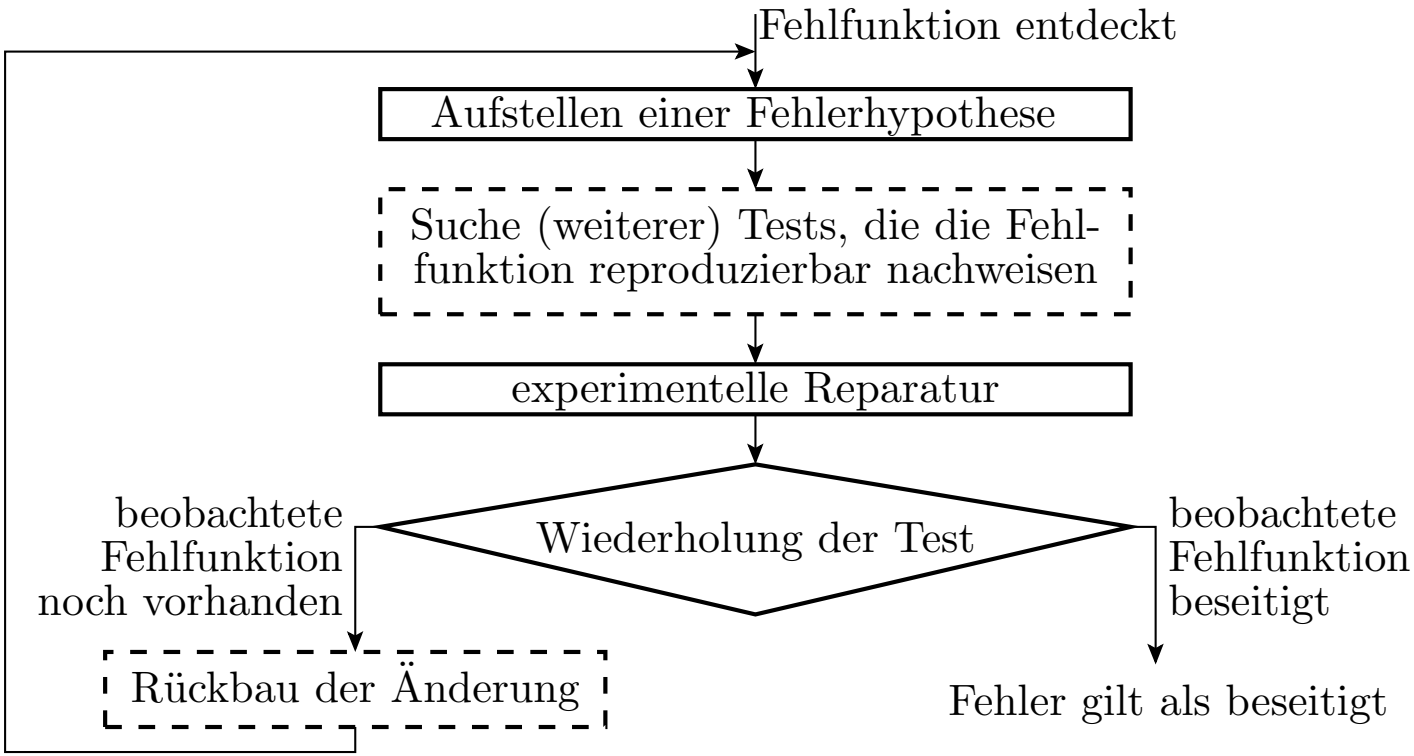
# IT-Systeme als Entstehungsprozesse

Ein Entstehungsprozess lässt sich wie ein IT-System als Service-Leister beschreiben:

- Eingaben sind dann die Entwurfs- oder Fertigungsvorgaben,
- Ergebnisse die Entwürfe oder Produkte bzw. deren Beschreibungen und (messbare) Eigenschaften.
- Die Abbildung erfolgt in Schritten und auch hierarchisch unter Nutzung von Teil-Service-Leistungen.
- Erkennbare Fehlfunktionen sind »kein Ergebnis«, Soll-Ist-Abweichungen kontrollierter Eigenschaften, auch von Zwischenschritten, und Fehler in den entstehenden Produkten.

Es gibt sogar Entstehungs-Service-Leistungen, die direkt von IT-Systemen ausgeführt (z.B. Programmübersetzung mit Compiler) oder von IT-Systemen gesteuert werden (z.B. menschenfreie Fertigung).

Ein IT-System als Entstehungsprozess arbeitet deterministisch. Die Fehlerbeseitigung erfolgt nach dem Prinzip der experimentellen Reparatur:




Es gibt Tests zu Erfolgskontrolle der Reparaturversuche. Die Änderungen durch erfolglose Reparaturversuche werden (idealerweise) rückgängig gemacht.

Eine experimentelle Reparatur erlaubt, dass alle erkannten Fehler mit hoher Wahrscheinlichkeit beseitigt werden, ohne dass dabei zu viele neue Fehler entstehen.

Die überwachte Abarbeitung von Entstehungs-Service-Leistungen entspricht dem Betrieb eines IT-Systems mit Daten aus der Anwendungsumgebung mit internen Kontrollen, die die Informationen über bemerkte Fehlfunktionen an den Hersteller zur Beseitigung senden. Derselbe Typ von Reifeprozess.

Für grobe Abschätzungen sei für die Fehlernachweisdichte der Prozessfehler wieder die bisher verwendete Potenzfunktion unterstellt, mit der die mittlere Zeit zwischen Fehlfunktionen wie folgt abnimmt:

$$Z(t) = Z(t_0) \cdot \left(\frac{t}{t_0}\right)^{k+1}$$

($t_0$ – Bezugszeit; $0 < k < 1$ – Modellparameter; $Z(t_0)$ – mittlere Zeit zwischen Fehlfunktionen zur Bezugszeit).

Die Fehlerentstehungshäufigkeit nimmt mit dem Kehrwert der mittleren Zeit zwischen der Fehlerentstehung ab, d.h. überproportion mit dem Kehrwert der Prozessnutzungsdauer:

$$h(t) = h(t_0) \cdot \left(\frac{t_0}{t}\right)^{k+1}$$

Dieses sehr günstige Reifeverhalten ist nur mit perfekt deterministischen Abläufen erzielbar, d.h. IT-Systemen, bei denen Fehlfunktionen nur durch deterministisch wirkende Fehler entstehen. Für nahezu deterministisch wirkende Systeme kommt noch eine Fehlerentstehungshäufigkeit durch zufällige Einflüsse (Störungen) hinzu, die nicht mit $t$ abnimmt:

$$h(t) = h(t_0) \cdot \left(\frac{t_0}{t}\right)^{k+1} + h_S$$

Zu den nahezu deterministisch wirkende Systemen gehören mit graduellen Abstufungen auch:

- rechnergestützte Fertigung,
- Fließbandfertigung, ...

# Zufällige Einflüsse

# Nicht deterministische Prozesse

In nicht deterministischen Prozessen sind auch die Fehlerwirkungen nicht deterministisch, d.h. bei Wiederholung ändert sich das Verhalten, ohne das sich daraus auf einen Fehler schließen lässt. Es lassen sich keine Tests zur Kontrolle des Reparaturerfolgs aufstellen. Die Iteration der experimentellen Reparatur aus Reparaturversuch und Erfolgskontrolle funktioniert nur eingeschränkt:

- Suche nach gehäuftem Auftreten gleicher Fehlfunktionen.
- Lokalisierung möglicher Ursachen.
- Schrittweise Beseitigungsversuche wahrscheinlicher Ursachen.
- Erfolgskontrolle anhand der Veränderung der Auftrittshäufigkeiten zu beobachtender Fehlerbilder.

Geringere Beseitigungswahrscheinlichkeit und höhere Entstehungswahrscheinlichkeit für neue Fehler als bei deterministischen Prozessen. Anderes Reifeverhalten.

# Der Technologiegedanke

Technologie: Lehre von reproduzierbaren Abläufen zur Erzeugung von Produkten (heute auch Fertigungstechnik)[8]

- Ein technologischer Prozess ist so zu beschreiben, dass, wenn er unter gleichen Bedingungen wiederholt wird, gleiche Produkte mit (nahezu) gleichen Eigenschaften entstehen.
- Dieser Technologiegedanke ist die Voraussetzung für eine Fehlervermeidung bzw. einen Reifeprozess nach dem Schema:
  - Notiere alle messbaren Ergebnisse des Entstehungsprozesses (Produkteigenschaften, beobachtete Prozessprobleme, ...).
  - Analysiere statistische Eigenschaften.
  - Variiere im Prozess so, das bestimmte Fehler nicht mehr oder seltener entstehen.

[8] Der Begriff »Technologie« wurde in diesem Sinne erstmalig von Johann Beckmann (1739-1811) in seinem Lehrbuch »Grundsätze der teutschen Landwirthschaft« verwendet. Heute interdisziplinäres Gebiet.

# Prozesszentrierung und Prozessverbesserung

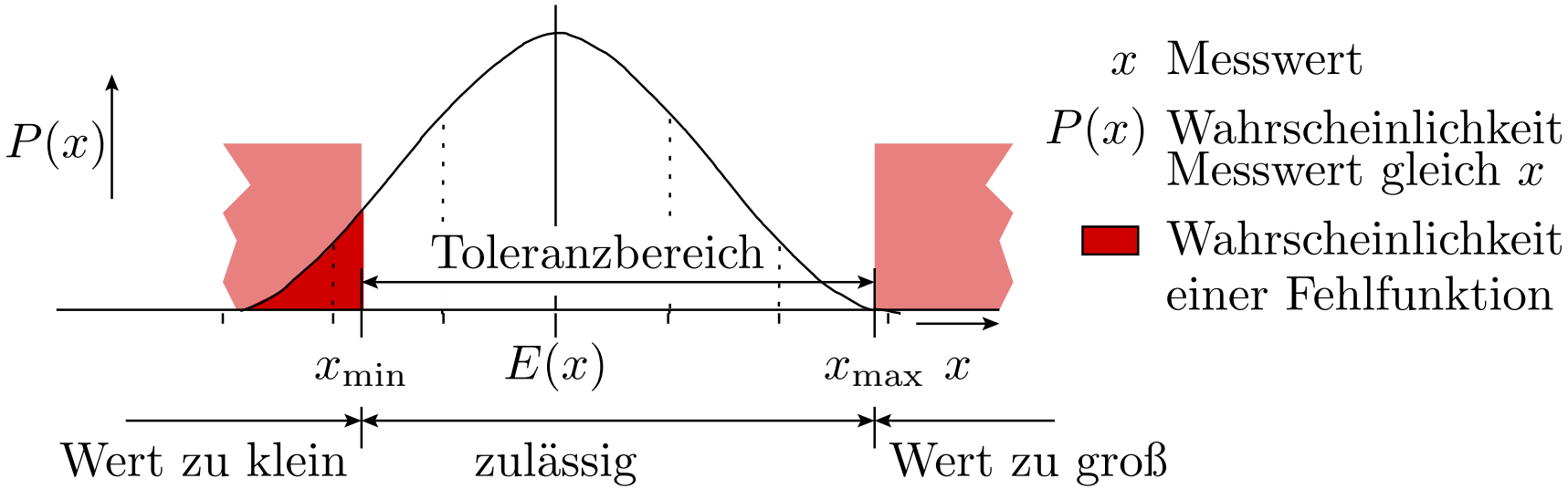
Nicht deterministische Entstehungsprozesse für einfache Produkte mit wenigen messbaren Produktmerkmalen, z.B. elektronische Bauteile, haben einen zweiphasigen Reifeprozess aus:

- Prozessverbesserung (Modernisierung der Maschinen, Abläufe, ...) und
- Prozesszentrierung (Fine-Tuning der Prozesssteuerparameter).

Das führt, wie im weiteren gezeigt, zu einem sägezahnförmigen Verlauf der Fehlerentstehungswahrscheinlichkeit in Abhängigkeit von der Prozessnutzungsdauer.

# Prozesszentrierung

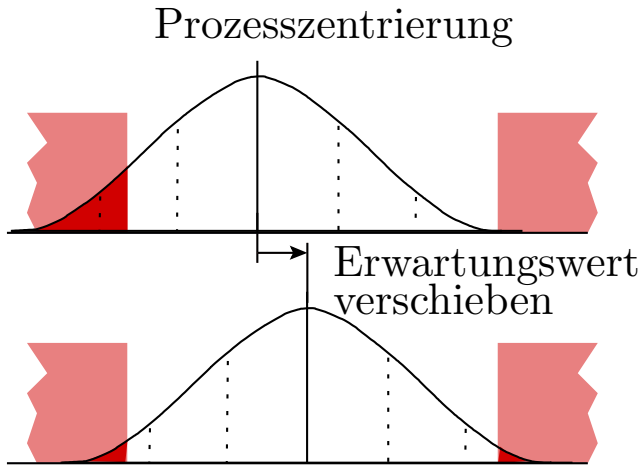
In funktionierenden Technologien sind die messbaren Produkteigenschaften meist normalverteilt:




Prozesszentrierung bedeutet, den Erartungswert der messbaren Parameter in die Mitte der Gaus-Glocke zu schieben. Dazu werden Prozesssteuerparameter (z.B. Temperatur, Druck, Matrialzusammensetzung etc.) in kleinen Schritten geändert.

Beispiel sei ein Prozess zur Herstellung von Widerständen durch Bedampfung eines Keramikträgers mit leitfähigem Material.

- Messbare Eigenschaften: Widerstandswert, Schichtdicke, Schichteigenschaften, ...
- Variierbare Parameter zur Prozesszentrierung:
  - Bedampfungsdauer,
  - Temperatur,
  - Materialzusammensetzung.

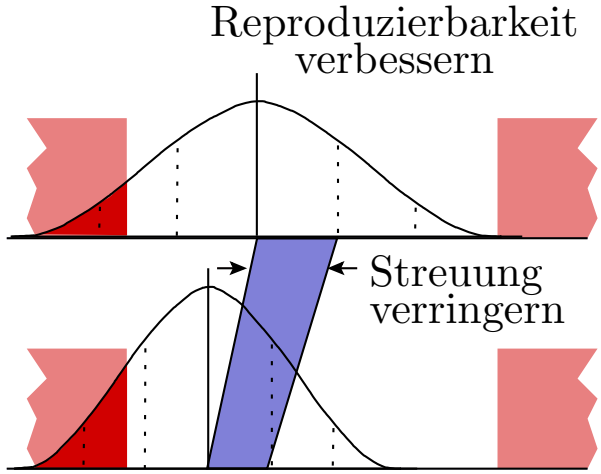



Iteratives Vorgehen:

- Ändern einer Eigenschaft,
- Bestimmen des Einfusses auf den Erwartungswert.
- Wenn gut, beibehalten, sonst zurücksetzen.

Bei einem zentrierten Erwartungswert ist bei gleicher Varianz und Toleranz die Wahrscheinlichkeit, dass der Wert außerhalb liegt (Fehlerentstehungswahrscheinlichkeit) am geringsten.

# Prozessverbesserung

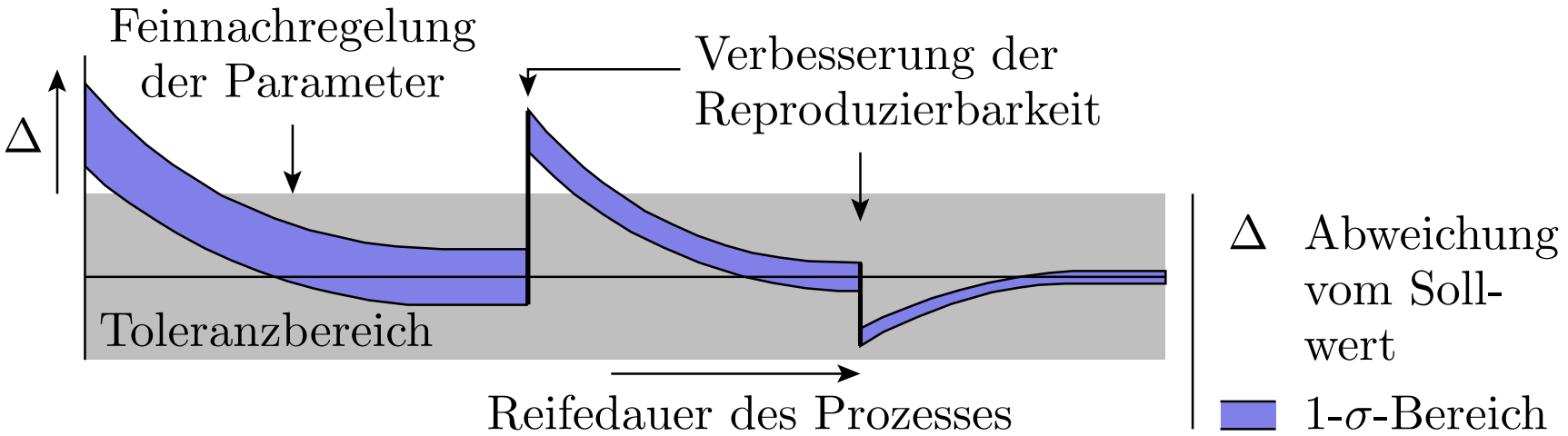



Bei einem zentrierten Prozess lässt sich die Fehlerentstehungswahrscheinlichkeit nur noch durch eine Verringerung der Varianz verringern. Das erfordert einen wesentlich größeren Aufwand und größere Eingriffe in den Prozess:

- neue Geräte, Anlagen, Materialien, Verfahren,
- neue Management-Strategien, ...

Bei größeren Prozesseingriffen geht in der Regel die Zentrierung verloren. Die Fehlerentstehungswahrscheinlichkeit nimmt sprunghaft zu. Danach folgt wieder eine Prozesszentrierung. Erst nach der Zentrierung bewirkt die Prozessverbesserung eine geringere Fehlerentstehungswahrscheinlichkeit.

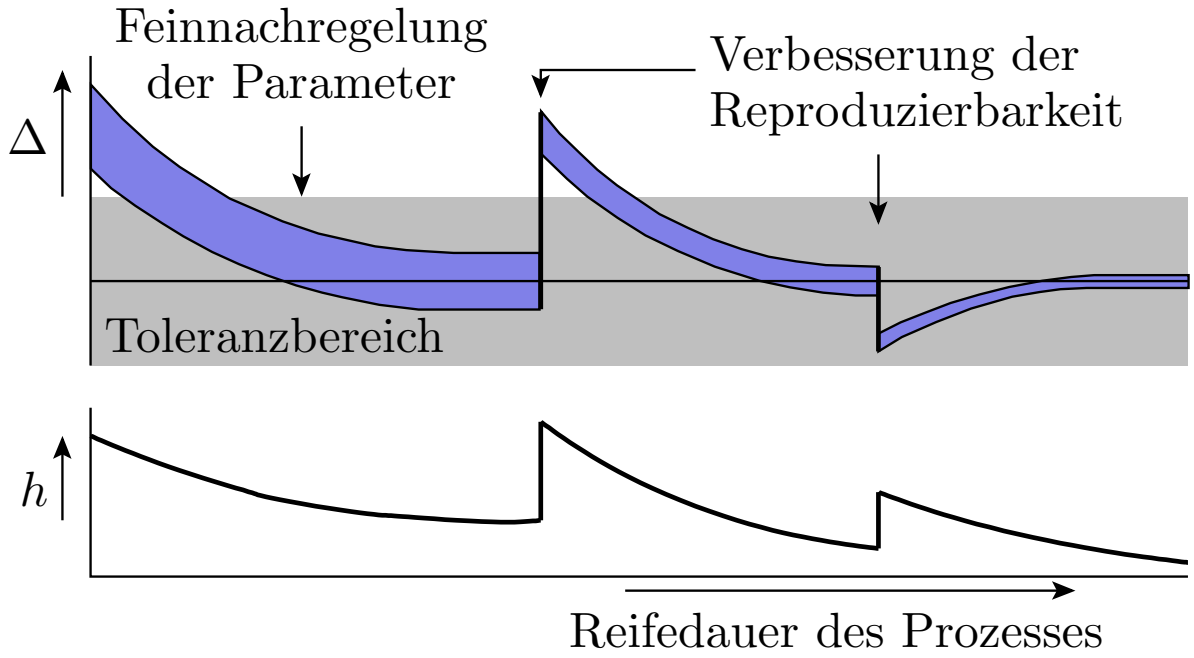
# Reifen nicht deterministischer Entstehungsprozesse




Technologien entwickeln sich ständig in den Phasen weiter:

- Prozessverbesserung (aller Jahre) und
- Prozesszentrierung (kontinuierlich).

Für alle Produktparameter gilt tendenziell, dass sie nach jeder Prozessverbesserung weniger streuen, aber die Mitte ihrer Toleranzbereiche verlassen, beobachtbar an einer sprunghaften Zunahme der Fehleranzahl.




Δ Abweichung vom Sollwert

1-$\sigma$-Bereich

$h$ Häufigkeit eines Parameterfehlers

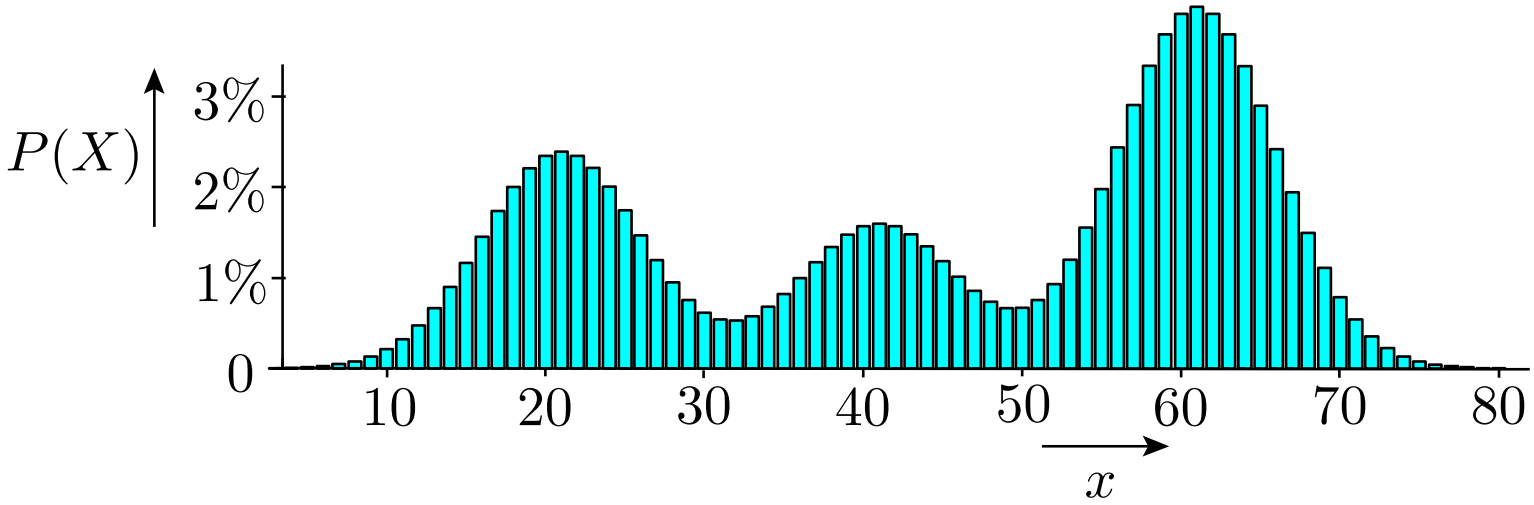
In der Zentrierungsphase nimmt die Fehleranzahl ab, und zwar weiter als vor der Prozessverbesserung.

**Fakt 4**

Innovationen sind zuerst schädlich, bevor sie sich rechnen.

# Multimodale Verteilung

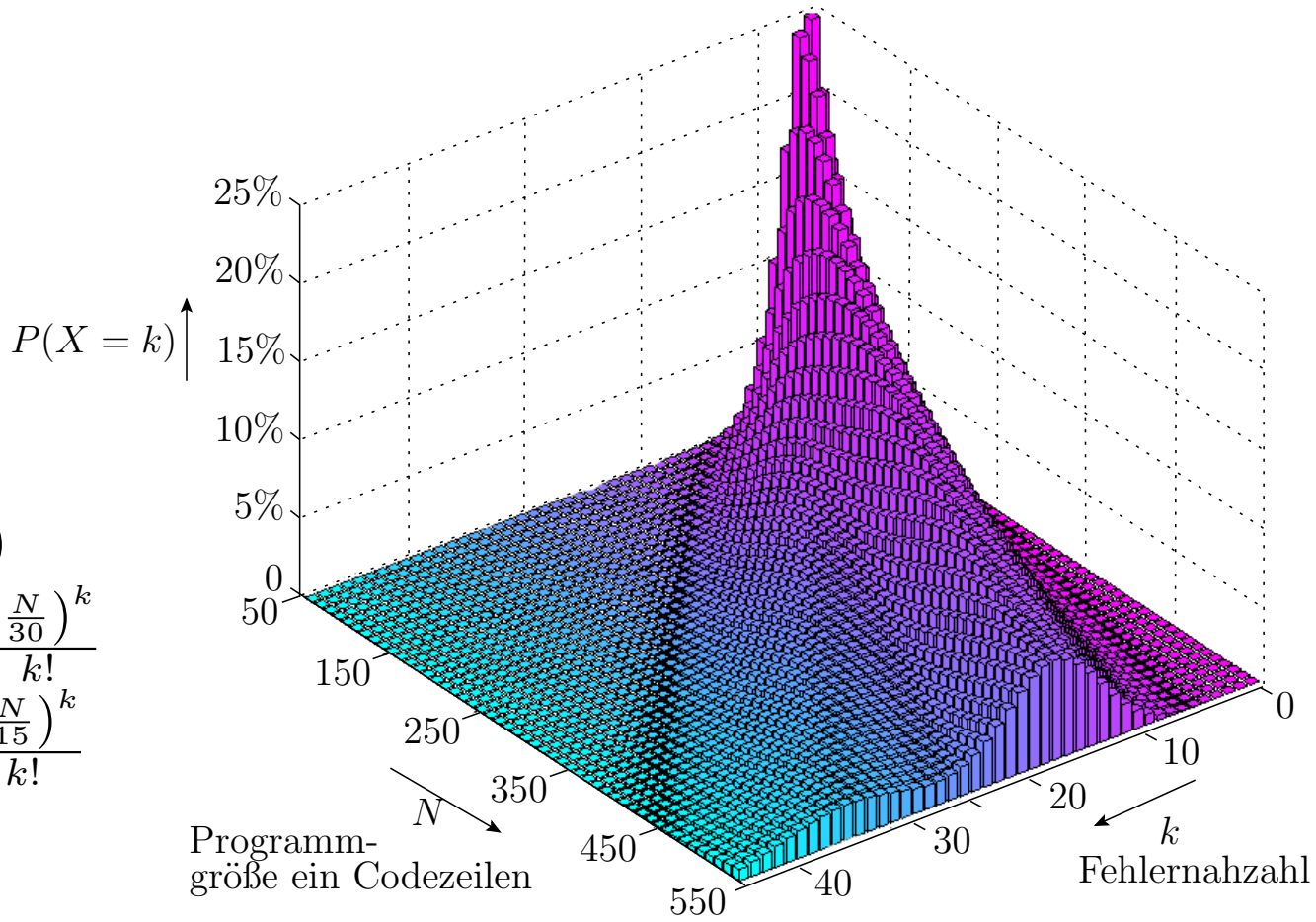
# Multimodale (mehrgipflige) Verteilung



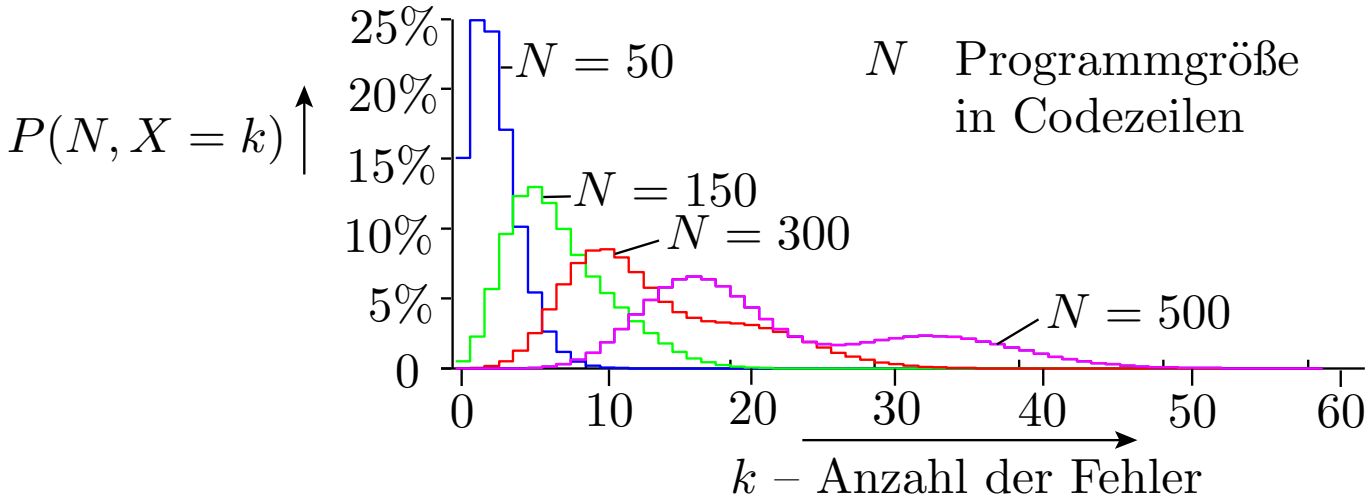

Eine Verteilung eines Messwertes mit mehreren Maxima deutet auf eine Mischung von Objekten aus besseren und schlechteren Entstehungsprozessen. Naheliegender Ansatz ist, die schlechteren Entstehungsprozesse durch den besten zu ersetzen. Angestebtes Ergebnis ist die günstigste Normalverteilung, d.h. die mit dem günstigsten Erwartungswert oder der kleinsten Streuung.

Beispiel war der Software-Entstehungsprozess auf F2, in dem ein Anfänger und ein Profi Software-Bausteine aus $N$ Code-Zeilen entwickeln, der Profi 66% der Bausteine mit ca. einem Fehler je 30 Codezeilen und der Anfänger 33% der Bausteine mit einem Fehler je 15 Codezeilen.

$$P(N, X = k) = \frac{2}{3} \cdot e^{-\frac{N}{30}} \cdot \frac{\left(\frac{N}{30}\right)^k}{k!} + \frac{1}{3} \cdot e^{-\frac{N}{15}} \cdot \frac{\left(\frac{N}{15}\right)^k}{k!}$$



Die Wahrscheinlichkeit, dass ein Modul genau $k$ Fehler enthält, ist 2/3 mal der Wahrscheinlichkeit, das es $k$ Fehler enthält und vom Profi stammt plus 1/3 mal der Wahrscheinlichkeit, dass es vom Anfänger stammt:




Die Polarisierung nimmt mit der Größe der Software-Bausteine, die vom Profi und vom Anfänger getrennt entwickelt werden, zu.

Naheliegende Fehlerbeseitiungsmaßnahme:

- Anfänger besser anlernen und dessen Ergebnisse dazu
- vom Experten kontrollieren lassen, ...

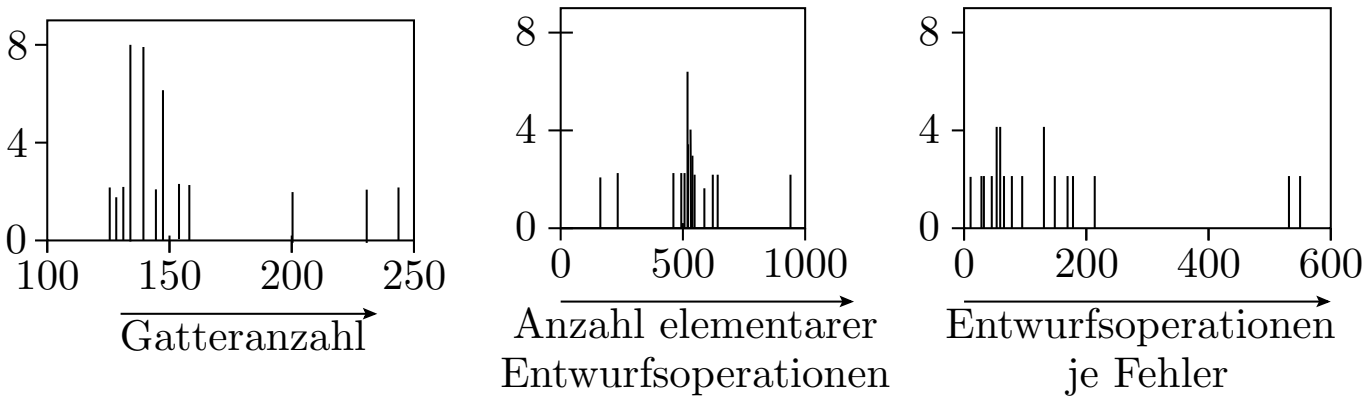
# Entwurfsprozesse

# Menschen im Entstehungsprozess

- In Entwurfsprozessen werden die meisten Fehler durch Menschen verursacht.
- Der Mensch erlernt in seinem Leben viele Vorgehensmodelle, die er der jeweiligen Aufgabe oder Situation anpasst.
- Je mehr er sich von Bekanntem entfernt, desto unvorhersehbarer ist das Ergebnis,
  - desto mehr Fehler entstehen und
  - desto größer die Varianz messbarer Parameter.
- Bei wiederholtem ähnlichen Vorgehen
  - stabilisiert sich das Vorgehen,
  - nimmt der Zeitaufwand ab,
  - nimmt die Varianz messbarer Arbeitsergebnisse z.B. die des Zeitaufwands ab,
  - nimmt der Fehleranzahl pro bewältiger Arbeit ab und
  - nimmt allgemein die Vorhersagbarkeit zu.

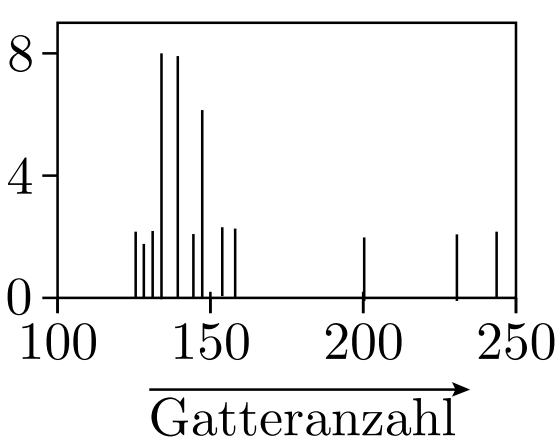
# Ein Experiment [1]

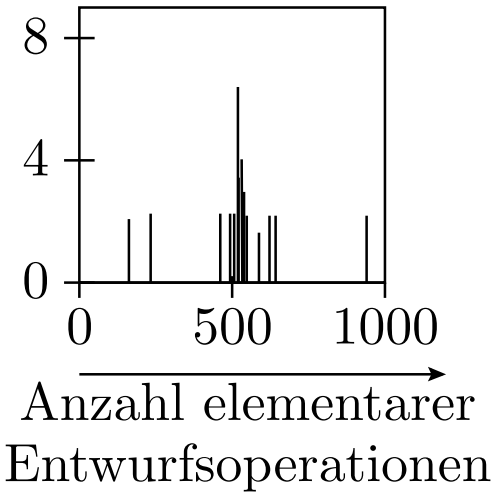
Eine Gruppe von 72 Studenten hatte die Aufgabe, aus der Beschreibung eines PLAs[9] eine Gatterschaltung zu entwickeln und diese über die grafische Benutzeroberfläche eines CAD-Systems in den Rechner einzugeben. Für jeden Entwurf wurden die elementaren Entwurfsoperationen[10], die Gatteranzahl und die Entwurfsfehler gezählt.

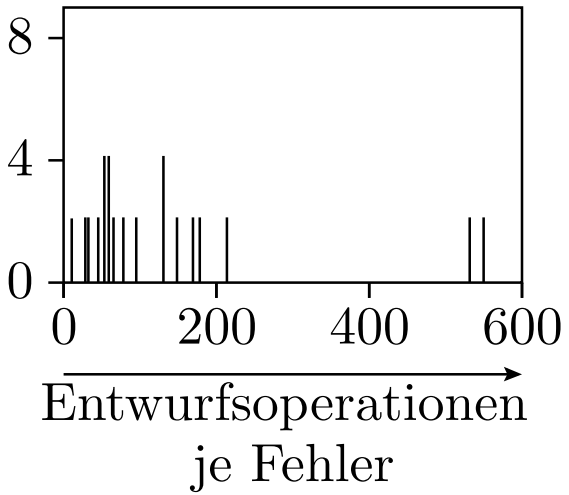



[9] PLA: programmable logic array

[10] Als elementare Entwurfsoperationen galten das Anordnen eines Gatters auf dem Bildschirm und das Zeichnen einer Verbindung.




Die Gatteranzahl der Entwürfe bewegte sich in einem Bereich von 131 bis 245, der gemessene Entwurfsaufwand zwischen 160 und 940 elementaren Entwurfsoperationen je Fehler in einem Bereich von 29 bis 550. Ableitbare Aussagen:

- Erhebliche Streuung, schlechte Vorhersagbarkeit.
- Die Prozessgüte schwankt in Abhängigkeit vom Studierenden zwischen 10 und 500 Operationen je Fehler und
- hat eine Mischverteilung mit Maximal bei 100 und 500.

Würde man diesen 72 Studierenden dieselbe Aufgabe nach einem Jahr noch einmal geben, wäre folgendes zu erwarten:

- Abnahme der mittleren Anzahl der Gatter und Operationen je Entwurf,
- Zunahme der Prozessgüte $Q$ (Entwurfsoperationen je Fehler),
- Abnahme der Streuung und
- Annäherung an eine Normalverteilung.

---

Studenten, Promoventen und andere Personen in einem Lernprozess sind im Grunde nicht fähig zu qualitativ hochwertigen Entwurfsarbeiten. Die dafür notwändige Routine fehlt. Die Ergebnisse sind noch zu wenig vorhersagbar. Dem Entstehungsprozess mit ihnen fehlt die Reifezeit. Grundlegendes Problem der Drittmittelforschung an Hochschulen[11].

[11] Die Industrie kann Entwurfsergebnisse aus Drittmittelforschung in der Regel nur über einen Know-How-Transfer (z.B. durch spätere Übernahme der Bearbeiter), aber kaum in einzusetzenden Produkten nutzen.

# Das Fähigkeitsmodell

Software-Entwicklungen sind traditionell hoch-kreative Prozesse mit kaum vorhersagbaren Ergebnissen:

- schwer vorhersagbare Projektdauer,
- schwer vorhersagbare Kosten,
- schwer vorhersagbare Benutzbarkeit, ...

Viele Projekte enden erfolglos. Aus diesem sehr unbefriedigenden Zustand hat sich die Software-Technik als Teilgebiet der Informatik herausgebildet, mit dem Ziel, auch in diesen Entstehungsprozessen den Technologiegedanken zu verankern.
Das Fähigkeitsmodell ist eine qualitative Klassifikation, wie weit ein Software-Entstehungsprozess Elemente einer Technologie enthält.

# CMU Capability Maturity Model (Reifegradmodell)

Im CMM wird ein Prozess mit einer von fünf Stufen bewertet:

**Initial:** Grundzustand, ohne einen definierten Prozess für die Softwareentwicklung. Kosten und Qualität unterliegen starken Schwankungen.

**Repeatable:** Die Planung neuer Projekte erfolgt anhand von Erfahrungen mit vergangenen Projekten. Zeiten sind einigermaßen kontrollierbar. Kosten und Qualität schwanken stark.

**Defined:** Es sind Software-Entwicklungs- und -wartungsprozess eingeführt und dokumentiert und die Verantwortlichkeiten für die Umsetzung geklärt. Kosten und Zeiten werden einigermaßen bewertbar. Qualität schwankt noch stark.

Managed (gesteuert): Sowohl für das Produkt als auch für den Prozess werden quantitative Ziele vorgegeben und ihre Einhaltung gemessen / überwacht. Einbeziehung der Qualität in die bewertbaren /vorhersagbaren Größen.

Optimizing: Die gesamte Organisation konzentriert sich auf das Finden von Schwächen und die weitere Verbesserung des Prozesses. Reifen des Entstehungsprozesses.

Erst in der letzten Stufe ist das erklärte Ziel Fehlervermeidung. Aber bereits ab »Repeatable,« wo das erklärte Ziel nur Reproduzierbarkeit und in den höheren Stufen Kontrolle ist, führt die Zielstellung, weil Reproduzierbarkeit und Kontrolle Fehlervermeidungstechniken sind, zur tendenziellen Absenkung der Fehleranzahl in den entstehenden Systemen.

# Vorgehensmodelle

# Ein Abstecher zu Lernrozessen

In der Schule und beim Erlernen praktischer Tätigkeiten werden zum erheblichen Teil Vorgehensmodelle vermittelt und trainiert:

- Rechnen, Schreiben, Handwerkern, Programmieren, ...
- Bewertung in Arbeitsmenge pro Fehlern.

Lernphasen:

1. Wissenvermittlung: anlesen, erklärt bekommen, ...
2. Training, bis Ergebnisse vorhersagbar
3. Professionalisierung: Prozessüberwachung; Beseitigung von Vorgehensfehlern und -schwachstellen.

An Universitäten:

- Phase 1: Vorlesung, Seminare, Selbststudium, ...
- Phase 2: Übung, Klausurvorbereitung[12], Praktika
- Phase 3: Aus Zeitgründen erst in der Berufspraxis für den eigenen eingeschränkten Tätigkeitsbereich.

[12] Auch Bewertung in Arbeitsmenge pro Fehler

# Vorgehensmodelle für Entstehungsprozesse

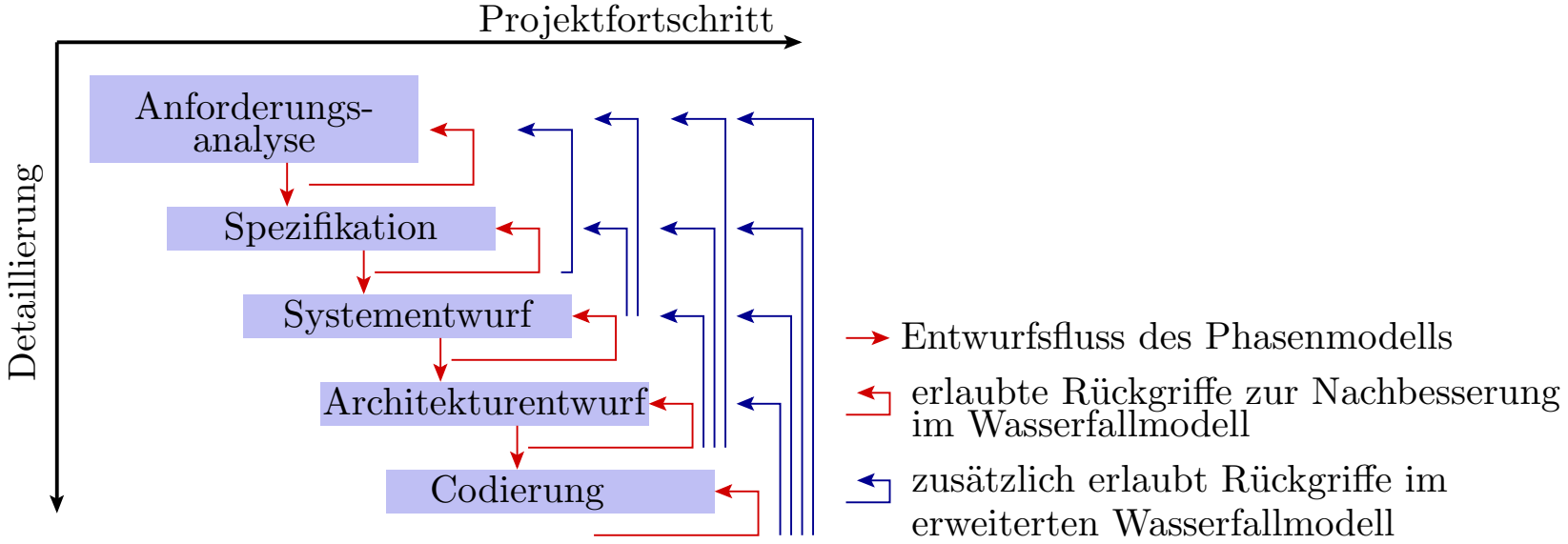
Wenn sich ein technologischer Ablauf nicht durch einen Algorithmus (schrittweise Abarbeitungsvorschrift) beschreiben lässt (wie für Projekte, Entwurfs- und Management-Prozesse), ist ein Vorgehensmodell die nächstbeste Alternative, um einen maximalen Grad an Reproduzierbarkeit zu erhalten. Typische Elemente von Vorgehensmodellen sind:

- Referenzabläufe,
- Unterteilung in Schritte und Phasenen und
- und die Definition von Zwischen- und Endkontrollen.

Das klassische Vorgehensmodell für die Software-Entwicklung ist das Stufenmodell. Grundphasen eines Software-Projekts:

- Anforderungsanalyse,
- Spezifikation der Ziele,
- Architekturentwurf, Codierung, Test, ...

# Varianten des Stufenmodells




- Wasserfallmodell: Fehlersuche und Beseitigung nach jeder Phase. Keine iterative Nachbesserung der vorheriger Phasen, sondern dann Neustart ab Fehler.
- Erweitertes Wasserfallmodell: Erlaubt auch iterative Nachbesserungen der Ergebnisse vorheriger Phasen, z.B. der Spezifikation nach der Codierung.

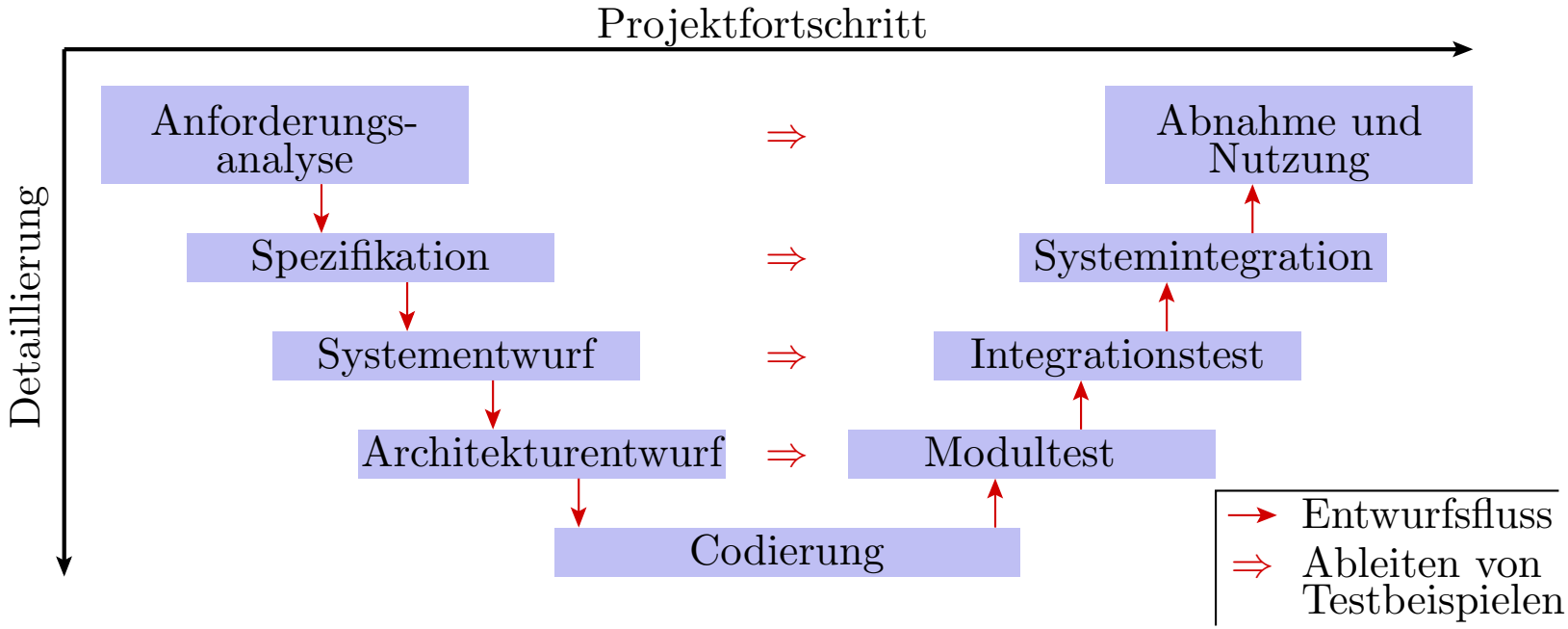
Das Wasserfallmodell hat den Nachteil, dass es Änderungen in vorherigen Phasen, z.B. der Architektur oder der Spezifikation in der Codierungsphase verbietet. Dadurch wird viel Verbesserungspotential verschenkt. Bei schwerwiegenden Fehlern in vorherigen Phasen werden die Ergebnisse späterer Phasen verworfen.

Das erweiterte Wasserfallmodell erlaubt freizügigere Änderungen in den vorherigen Phasen. Der Preis ist ein schlechter vorhersagbarer Entwurfsablauf und die zusätzliche erhebliche Fehlerquelle »nachträgliche Änderungen«.

Entscheident für die Praxistauglichkeit beider Modelle ist die Gründlichkeit der Tests zwischen den Phasen, die das Risiko für einen notwändigen Neubeginn oder nachträgliche Änderungen bestimmen.

Wedes das absolute Verbot noch die absolute Freiheit für Rückgriffe ist die optimale Lösung. Für Prozesse ab »Repeatable« erfolgt hier ein individuelles Fine-Tuning.

# Das V-Modell




Das V-Modell ist ein schwergewichtiges Stufenmodell. Die Gewichtigkeit steht für die zu produzierende und zu kontrollierende Menge von Dokumenten in den Phasen. Der zweite Ast für das »V« ist ein Stufenmodell für den Test mit Vorgaben für das Vorgehen für die Testauswahl.

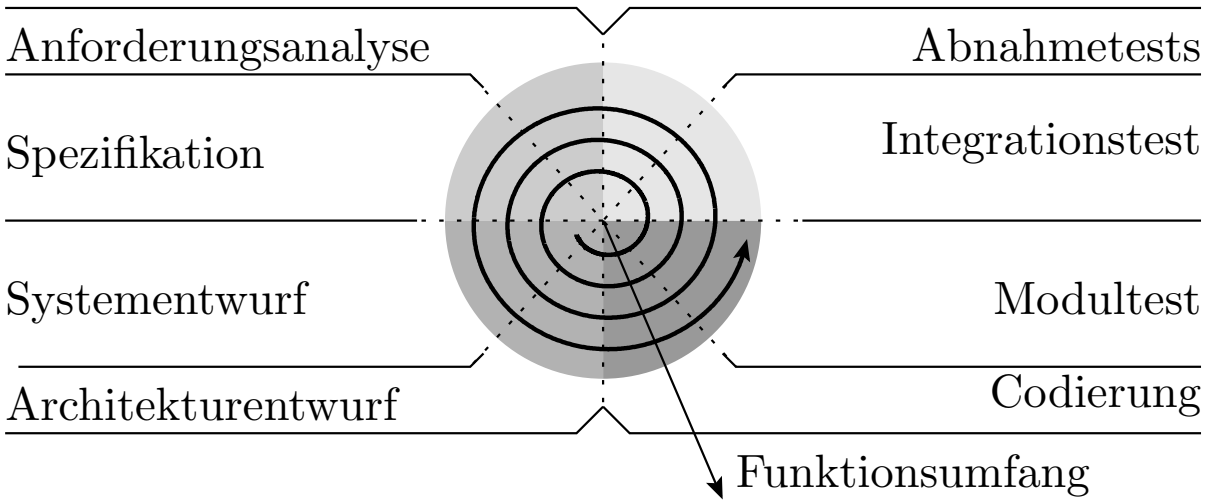
Schwergewichtige Prozesse bezahlen für die Reproduzierbarkeit mit einem hohen Aufwand an zusätzlichen Arbeitsschritten (Team-Besprechungen, Buchführung über jeden erkannten Fehler, ...) und wenig Flexibilität für sich ändernde Anforderungen.

Reproduzierbarkeit ist ein Mittel der Fehlervermeidung. Auf der anderen Seite wächst die Fehleranzahl in einem System mit dem Arbeitsaufwand und auch mit der Menge der zu verwaltenden und zu bearbeitenden Dokumentationen (Schwergewichtigkeit als Fehlerquelle).

Wichtig ist der richtige Kompromiss, der für unterschiedliche Aufgaben und Organisationen durchaus unterschiedlich ausfallen kann.

# Evolutionäre Modelle

Für innovative Produkte ist es oft besser, mit elementaren Grundfunktionen zu beginnen und diese bis zum funktionierenden System zu führen, dann die Zielfunktionen in mehreren Iterationen von der Anforderungsanalyse bis zum Abnahmetest zu erweitern. Bei evolutionären Vorgehensmodellen wird der Phasenzyklus mit einer zunehmend komplexeren Menge von Anforderungen mehrfach durchlaufen:



Evolutionäre Modelle haben gegenüber starren Modellen einen prinzipiellen Nachteil. Der Entstehungsprozess, in dem Fehler entstehen, verlängert sich durch die mehrfache Erweiterung und die damit verbundenen Änderungen am bisherigen. Es werden mehr Fehler entstehen. In einem evolutionären Prozess entstandene Systeme tendieren dazu, weniger verlässlich zu sein.

Evolutionären Prozesse eignen sich vor allem für die Entwicklung von Demonstrations- und Untersuchunsobjekten (prove of concept), für die Flexibilität wichtiger ist als Verlässlichkeit.

# Qualität und Kreativität

In Entstehungsprozessen sind Qualität und Kreativität zwei entgegengesetzte Zielstellungen. Qualität verlangt

- eine hohe Wiederholrate gleicher oder ähnlicher Tätigkeiten,
- strenge Kontrollen, dass vorgeschriebene Arbeitsabläufe pedantisch eingehalten werden,
- schmalbandig spezialisiertes Personal und
- die Beschränkung der Kreativität auf Details wie das Ausfüllen von Formblättern und die Protokollierung von Abspracheergebnissen.

Für Kreativität im Sinne des Einbringens neuer Konzepte, Ausprobieren neuer Lösungswege, ... ist in Entwurfs- und Fertigungsprozessen von Systemen für den Einsatz wenig Raum.

Kreativität und innovative Ideen gehören in den Prototypentwurf. Prototypen sind nur bedingt für den Einsatz geeignet.

# Inspektionstechnologien

# Inspektion (Review)

Kontrolltätigkeit, Sichtprüfung (von lat. inspicere = besichtigen, betrachten); Anwendbar auf:

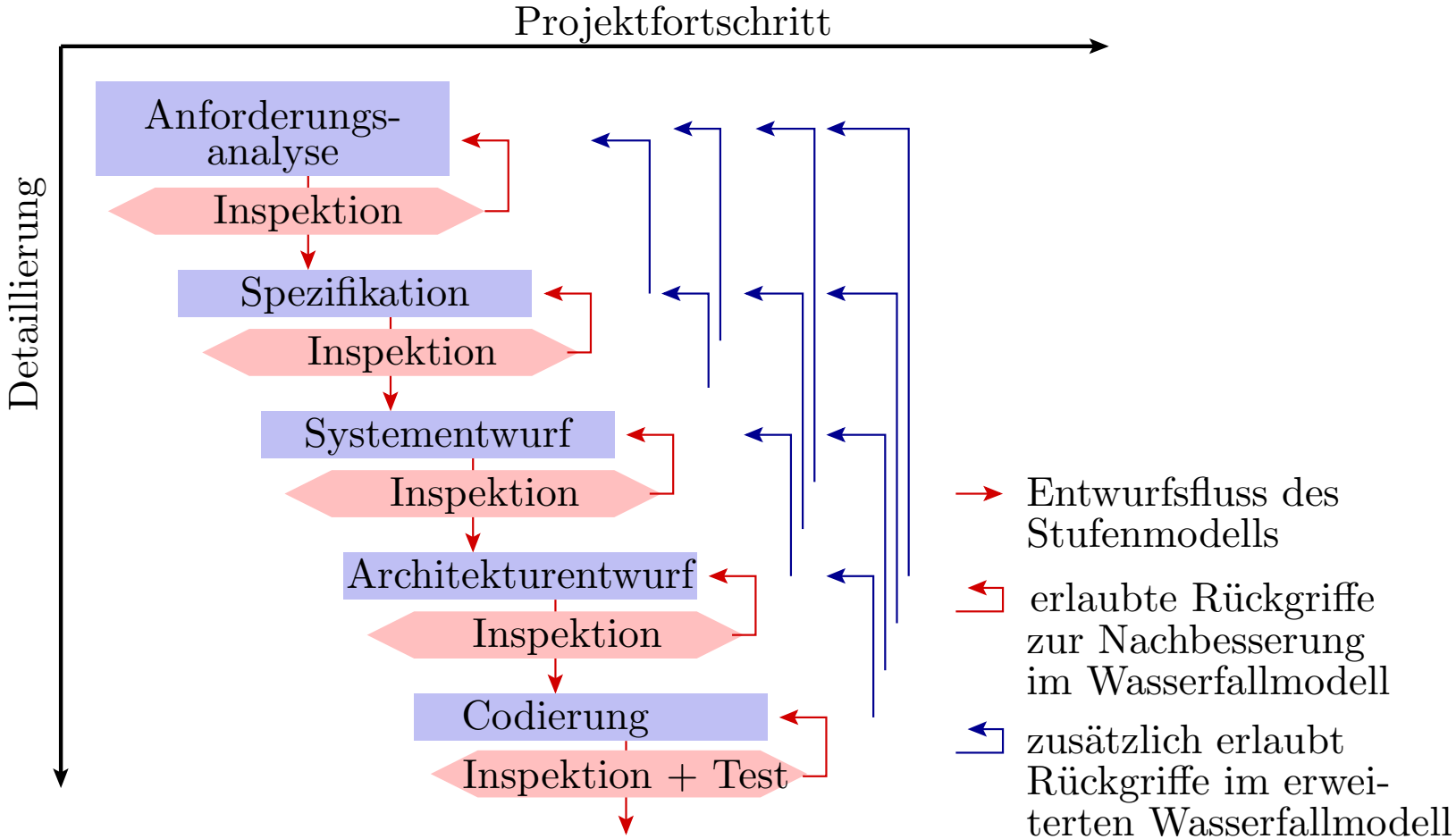
- Dokumente (Spezifikation, Nutzerdokumentationen, ...),
- Programmcode, Testausgaben,
- Schaltungsbeschreibungen,
- gefertigte Schaltungen (Sichtprüfung).

Wichtigstes Kontrollverfahren für Entwurfsprozesse.

Genau wie bei Entwurfsprozessen sind hier hier die Schwachpunkte:

- Projektorientierung,
- Kreativität und
- Handarbeit (im Sinne von nicht automatisiert).

Den subjektiven Einflüssen wird durch starke Formalisierung der Arbeitsabläufe entgegengewirkt. Gilt auch für Entwerfen.




In einem Entwurfsprozess werden idealerweise nach jeder Entwurfsphase die Ergebnisse durch eine Inspektion kontrolliert. Selbstverständlich ist, dass der Autor die Dokumente selbst auf Fehler durchsieht. Besser ist die Hinzunahme weiterer Personen.

Denn der Autor ist nach einer gewissen Inspektionsdauer blind für die noch vorhandenen Fehler.

Inspektionstechnologien sind ähnlich bei Entwürfen Vorgehensmodelle, um den Ablauf und das Ergebnis kontrollierbar reproduzierbar zu machen. Dazu gehört die Definition messbare Kennwerte:

- Effizienz: gefundene Abweichungen pro Mitarbeiterstunde
- Effektivität: gefundene Abweichungen je 1000 NLOC

(NLOC – netto lines of code, Programmzeilen kommentarbereinigt. Weitere Elemente sind Rollenverteilungen, Ablaufdefinitionen, ... Günstig für ein gutes Inspektionsergebnis sind:

- eine gleichbleibende Geschwindigkeit (es gibt Richtwerte für die optimale Anzahl der zu inspizierenden Code-Zeilen pro Stunde),
- Klare Reglungen für den Informationsfluss zwischen Autor und Inspektor oder mehreren Inspektoren, ...

# Einteilung der Inspektionstechniken

- Review in Kommentartechnik: Korrekturlesen und Dokument mit Anmerkungen versehen. Keine Ablaufkontrolle. Starke Schwankungen der Effizienz, Effektivität und Fehlerüberdeckung.
- informales Review in Sitzungstechnik: Lösungsbesprechung in der Gruppe, Vier-Augen-Prinzip. Nimmt die Monotonie, steigert die Aufmerksamkeit, fördert den Wissensaustausch. Für eine Mindesteffizienz und Effektivität Die Teilnehmer sollten mit kommentierten Reviews erscheinen.
- formales Review in Sitzungstechnik: fester Rollenteilung (Leser, Moderator, Autor, Inspekteure). Festgeschriebenen Organisationsablauf: Vorlesen, besprechen, Ergebnisse protokollieren, ... max. eine Stunde am Stück. Inspekteuer fragen, Autor antwortet, ... starke Anlehnung an den Technologiegedanken.

| Quelle [5] | NLOC | OwA | Mitarbeiter-stunden | Effizienz | Effek-tivität |
|---|---|---|---|---|---|
| formal, Sitzung | 11909 | 87 | 501 | 0,17 | 7,3 |
| informal, Sitzung | 176391 | 226 | 2680 | 0,05 | 1,3 |
| Kommentartechn. | 188300 | 334 | 6112 | 0,08 | 1,8 |

- NLOC (netto lines of code): Programmzeilen kommentarbereinigt
- OwA: gefundene operational wirksame Abweichungen
- Effizienz: gefundene Abweichungen pro Mitarbeiterstunde
- Effektivität: gefundene Abweichungen je 1000 NLOC

---

- formale Inspektionen sind sehr aufwändig, haben aber die größte Effizienz und Effektivität
- informale Techniken sind aufwandsärmere Alternativen.

# Aufgaben

- Geg. Toleranzbereich, Standardabweichung, Dezentrierung. ges. Wahrsch. Fehlerentstehung.
- Was ist der Nachteil eines kreativen Arbeitsstils bei der Software-Entwicklung?

# Verlässlichkeitsbewertung

In einer neuene Forschungsrichtung dauert es Jahrzehnte, bis das Begriffsgefüge schlüssig und verständlich ist. Einige Definitionen für den Zuverlässigkeitsbegriff:

- DIN EN ISO 8402 bezeichnet Zuverlässigkeit als Sammelbegriff bezüglich der Eigenschaften, richtig zu funktionieren, das eine Wartung möglich ist etc. Entspricht etwa dem, was in der Vorlesung als Verlässlichkeit bezeichnet wird.
- DIN 40041 definiert Zuverlässigkeit als Teil der Qualität im Hinblick auf das Verhalten während oder nach einer vorgegebene Zeitspanne bei vorgegebenen Anwendungsbedingungen. Das entspricht etwa der Überlebenswahrscheinlichkeit in der Reparatur und Erneuerungstheorie.

- DIN ISO 9000 Teil 4 definiert Zuverlässigkeit als Beschaffenheit einer Einheit bezüglich ihrer Eignung, während oder nach einer Zeitspannen bei vorgegebenen Zuverlässigkeitsanforderungen die Zuverlässigkeitsanforderungen zu erfüllen. Konsistent zu DIN 40041.

Es gibt weitere Begriffsbeschreibungen z.B.

- »Fähigkeit, alle Anforderungen zu erfüllen«. Größere IT-Systeme enthalten mit an Sicherheit grenzender Wahrscheinlichkeit Fehler und wären nach dieser Definition unzuverlässig.
- »Wahrscheinlichkeit, innerhalb einer Nutzungsdauer alle Anforderungen zu erfüllen.« Dann wäre Zuverlässigkeit eine von der Nutzungsdauer abhängige Größe, was der allgemeinen Vorstellung von Zuverlässigkeit widerspricht.

Alles noch unbefriedigend. Die Vorlesung versucht deshalb, die die verlässlichkeitsrelvanten Begriffe so zu definieren, dass ihnen experimentell bestimmbare Werte zugordenbar sind, mit denen gerechnet werden kann.

# Quantifizierung der Verlässlichkeit

Verlässlichkeit charakterisiert die Seltenheit von Problemen während des Betriebs. Die Probleme können dabei unterschiedlicher Natur sein:

- Abstürze, Ausfälle, falsche Ergebnisse,
- Ergebnis nicht rechtzeitig verfügbar,
- Gefährdungen, Datenverlust, ...

Wenn nur ein Teil der möglichen Probleme betrachtet wird, wird das durch einen Unterbegriff der Verlässlichkeit beschrieben:

- Verfügbarkeit: Seltenheit, dass das System nicht verfügbar ist.
- Zuverlässigkeit: Seltenheit, dass das System falsch Ergebnisse liefert.
- Betriebsicherheit: Seltenheit, dass von dem System Gefahren ausgehen. ...

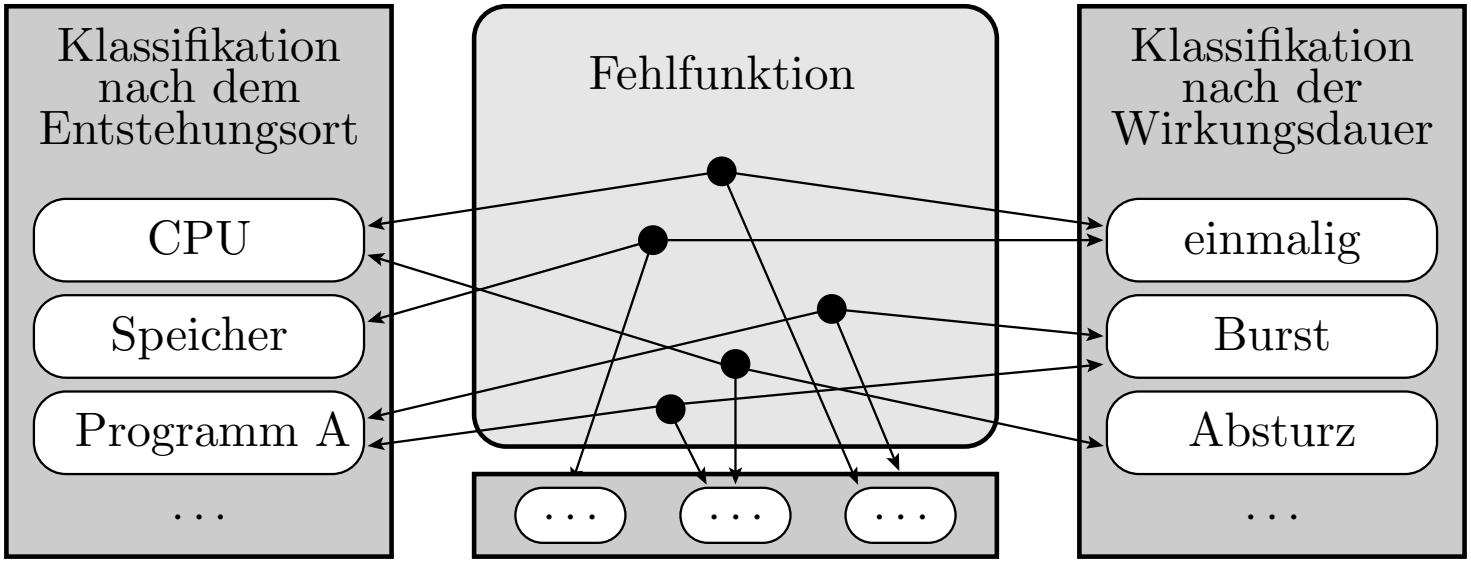
# Problemraten: Probleme pro Zeit (und System)

Die quantitativ abschätzbaren Größen zur Bewertung der Verlässlichkeit sind die Häufigkeiten der Problemen, sowohl insgesamt als auch für spezielle Teilmengen. Experiment zur Bestimmung: Zählen der beobachtbaren Probleme und Aufsummieren der Zeiten, in denen gezählt wird, für viele gleichartige Systeme über eine lange Zeit.

Beispiele für Problemraten:

- Fehlfunktionen pro Zeit. Häufigkeit $\gamma$ der Fehlfunktionen je Service-Aufruf multipliziert mit der mittleren Anzahl von Service-Aufrufen pro Zeit.
- Ausfallrate siehe Abschn. #,
- Absturzrate (Abstürze pro Zeit und System): die für IT-Nutzer am einfachsten zu beobachtende Problemrate. ...

# Gesamt- und Teilproblemraten




Bei einer eindeutigen Zuordnung von Problemen zu Problemklassen

$$\lambda = \sum_{i=1}^{N_{\mathrm{PK}}} \lambda_i \tag{2}$$

ist die Summe alle Problemraten die Summe der Problemraten aller Klassen.

Umgekehrt entfällt auf jede Teilproblemrate $i$ ein Anteil $a_i$ der Gesmtproblemrate:

$$\lambda_i = a_i \cdot \lambda \texttt{ mit } \sum_{i=1}^{N_{\mathrm{PK}}} a_i = 1$$

Wenn durch eine verlässlichkeitssichernde Maßnahme die Gesamtproblemrate verringert wird, nehmen in erster Näherung auch die Teilproblemraten proportional ab.

Für die Beschreibung von Verlässlichkeit, Zuverlässigkeit etc. ist die Problemrate ungünstig, weil sie bei größerer Zuverlässigkeit kleiner und ungkehrt ist. Besser ist der Kehrwert, die mittlere problemfreie Zeit MTBFx – Mean Time between Failtures (x – Art der Probleme). Bei einer zeitinvarianter Problemrate ($\lambda \neq f(t)$ ):

$$\mathrm{MTBFx} = \frac{1}{\lambda_{\mathrm{x}}}$$

Maßeinheit einer Zeit.

# Zuverlässigkeit

# Zuverlässigkeit

## Definition 5

Die Zuverlässigkeit eines IT-Systems sei die mittlere Betriebsdauer ohne Fehlfunktion:

$$Z = \frac{t_\mathrm{B}}{\xi} \tag{3}$$

($t_\mathrm{B}$ – Betriebsdauer; $\xi$ – Anzahl der beobachteten Fehlfunktion).

Gezählt werden alle Fehlfunktionen und addiert alle Betriebszeiten, in denen Fehlfunktionen gezählt werden.

Zuverlässigkeit unterschiedlicher Windows-Versionen nach [4][13]:

- Windows 98: $Z = 216\,\mathrm{h}$ (ca. 1 Woche)
- NT 4.0: $Z = 919\,\mathrm{h}$ (ca. 5,5 Wochen)
- Windows 2000 Professional: $Z = 2893\,\mathrm{h}$ (ca. 4 Monate).

[13] Die Quelle sagt nicht genau, was gezählt wurde.

# Systeme aus mehreren Komponenten

In einem System aus mehreren Komponenten:

- Rechner, Betriebssystem,
- Anwendungssoftware, ...

addieren sich die Problemraten der gleichzeitig genutzten Komponenten und damit die Kehrwerte ihrer Zuverlässigkeiten.

Beispielsystem mit vier gleichzeitig genutzten Komponenten:

- Rechnerhardware: $Z \approx 10^4\text{h}$
- Internetzugang: $Z \approx 5 \cdot 10^2\text{h}$
- Betriebssystem (NT 4.0): $Z \approx 10^3\text{h}$
- Web-Browser: $Z \approx 3 \cdot 10^2\text{h}$

Gesamtzuverlässigkeit:

$$Z_{\text{ges}} = \frac{1\,\text{h}}{\frac{1}{10^4} + \frac{1}{5 \cdot 10^2} + \frac{1}{10^3} + \frac{1}{3 \cdot 10^2}} = 155\,\text{h}$$

Gesamtzuverlässigkeit kleiner kleinste Teilzuverlässigkeit.

# Einflussfaktoren auf die Zuverlässigkeit

Zu erwartende Anzahl der entstehenden Fehler:

$$E(\varphi_{\mathrm{E}}) \approx \frac{N}{Q}$$

($N$ – Entstehungsaufwand in Entwurfsoperationen; $Q$ – Prozessgüte in Entwurfsoperationen je entstehender Fehler). Anteil der Fehler davon, die noch zu Beginn des Einsatzes im System sind Fehler[14]:

$$E(\varphi) \approx (1 - FC) \cdot \frac{N}{Q}$$

($FC$ – Fehlerüberdeckung aller Tests zusammen). Mit einer mittleren Rate von Fehlfunktionen je Fehler $\bar{h}$ und eine nicht durch Fehler verursachten Fehlfunktionsrate $\lambda_{\mathrm{S}}$ ist die Rate der Fehlfunktionen insgesamt:

$$\lambda_{\mathrm{FF}} \approx \bar{h} \cdot (1 - FC) \cdot \frac{N}{Q} + \lambda_{\mathrm{S}}$$

[14] Unter der Annahme, dass alle gefundenen Fehler beseitigt werden.

Die Zuverlässigkeit beträgt:

$$Z \approx \frac{1}{\bar{h} \cdot (1 - FC) \cdot \frac{N}{Q} + \lambda_S}$$

Unter Vernachlässigung von $\lambda_S$ verhält sich die Zuverlässigkeit eines Systems proportional zu Güte seines Entstehungsprozesses $Q$ , umgekehrt proportional zum Entstehungsaufwand $N$ und umgekehrt proportional zum Anteil der nicht nachweisbaren Fehler $(1 - FC)$:

$$Z \approx \frac{Q}{\bar{h} \cdot (1 - FC) \cdot N}$$

Die mittlere Häufigkeit der Fehlfunktionen je Fehler $\bar{h}$ hängt von den Tests ab. Für einen Zufalls- oder einen Test in der Anwendungsumgebung nimmt sie umgekehrt proportional mit der Testdauer $t_T$ ab:

$$Z \approx \text{konst} \cdot \frac{Q \cdot t_T}{(1 - FC) \cdot N}$$

(konst. – Proportionaltätsfaktor).

Eingebaute Kontroll- und Fehlerbehandlungsfunktionen erhöhen die Zuverlässigkeit ungekehrt proportional zur mittlern Maskierungswahrscheinlichkeit $p_{\mathrm{R}}$:

$$Z \sim \frac{Q \cdot t_{\mathrm{T}} \cdot t_{\mathrm{R}}^{k+1}}{p_{\mathrm{R}} \cdot (1 - FC) \cdot N}$$

In einem Reifeprozess während des Einsatzes nimmt die Häufigkeit der durch Fehler verursachten Fehlfunktion mit der Reifedauer $t_{\mathrm{R}}$ mit dem dem Exponent $1 + k$ ($0 < k < 1 -$ Exponent der Fehlernachweisdichte) ab und die Zuverlässigkeit zu:

$$Z \sim \frac{Q \cdot t_{\mathrm{T}} \cdot t_{\mathrm{R}}^{k+1}}{p_{\mathrm{R}} \cdot (1 - FC) \cdot N}$$

- Fehlervermeidung, Test und Fehlerbeseitigung,
- eingebaute Kontrollfunktionen und Fehlerbehandlung und
- Reifprozesse im Einsatz.

haben einen vergleichbaren Einfluss auf die Zuverlässigkeit eines Systems im Einsatz. Weiterhin wichtig ist ein deterministisches Verhalten (vernachlässigbares $\lambda_{\mathrm{S}}$).

## Interessante Fragestellungen

- Wie groß darf die Maskierungswahrscheinlichkeit eingebauter Kontrollfunktionen maximal sein, damit sich die Zuverlässigkeit signifikat verbessert?

$$p_{\mathrm{R}} \ll \cdot \frac{N}{N_{\mathrm{K}} + N}$$

($N_{\mathrm{K}}$ – Entstehungsaufwand der Kontrollfunktionen). Sie muss so klein sein, dass die Fehlfunktionsrate durch Fehler in den Kontrollfunktionen mehr als ausgeglichen wird.

- Ein diversitäres 3-Versionssystem habe eine Maskierungswahrscheinlicht von $p_{\mathrm{R}} \approx 1\%$. Welche Zuverlässigkeitsverbesserung ist zu erwarten?

$$\frac{Z_{3\mathrm{Vers}}}{Z_{3\mathrm{Vers}}} \approx \frac{\frac{Q \cdot t_{\mathrm{T}} \cdot t_{\mathrm{R}}^{k+1}}{1\% \cdot (1-FC) \cdot (3 \cdot N - N_{\mathrm{Vot}})}}{\frac{Q \cdot t_{\mathrm{T}} \cdot t_{\mathrm{R}}^{k+1}}{(1-FC) \cdot N}} \approx 33$$

($N_{\mathrm{Vot}}$ – Zusatzaufwand für den Entwurf des Voter und das Zusammenfassen der drei Versionen zu einem System).

- Welche Reifedauererhöhung ist zur Kompensation einer Verringerung der Güte $Q$ des Entstehungsprozesses auf ein Viertel z.B. durch die Beschäftigung eines Studenten als Entwerfer erforderlich?

$$Q \cdot t_{\mathrm{R}}^{k+1} \geq 1$$

Mindestens eine Verdopplung $k < 1$ bis max. eine Vervierfachung für $k > 0$.

# Betriebssicherheit

# Teilaspekte der Verlässigkeiten

In Abhängigkeit von der Funktion und vom Einsatz gibt es oft einige besonders gefürchtete Problemsituationen, die spezielle Schutzmaßnahmen verlangen und durch Unterbegriffe der Verlässlichkeit beschreiben werden:

- Betriebssicherheit: Großer Material- und Personenschaden. Robotik, Anlagen-, Maschinen- und Fahrzeugsteuerungen. Schutzmaßnahmen: Fehlerbehandlungsfunktionen zur Herstellung gefahrenfreier Zustände. Notfallpläne.
- Datensicherheit vor Verlust: Verlust aufwändig wiederzubschaffender Daten. Arbeitsplatzrechner, Datenbanken, Server. Schutzmaßnahmen: Redundanten Datenspeicherung und Back-Ups.

- Datensicherheit vor unbefugtem Zugriff. Finanzdaten, Gesundheitsdaten, ... Schutzmaßnahmen: Kryptographische Verschlüsselung, Passwortgeschützter Zugang.
- Absturzsicherheit. Schutzmaßnahmen: Fehlerisolation, Watchdog.
- Verfügbarkeit. Kommunikationssysteme, Produktionsanlagen, ... Schutzmaßnahmen: Redundanz, regelmäßige Wartung und organisatorische Vorbereitungen für eine schnelle Problembehebung, Notfallplan.

Dieser Abschnitt beschränkt sich auf den Teilaspekt Betriebssicherheit. Die Besonderheit der Modellierung der Betriebssicherheit ist, dass die zu betrachtenden Problemsituationen

- Explosionen von Anlagen,
- Autounfälle, Flugzeugabstürze,
- Fehldiagnosen und Fehlversorgung in der Medizin, ...

sicher auszuschließen sind.

# Sicherheitsanalyse

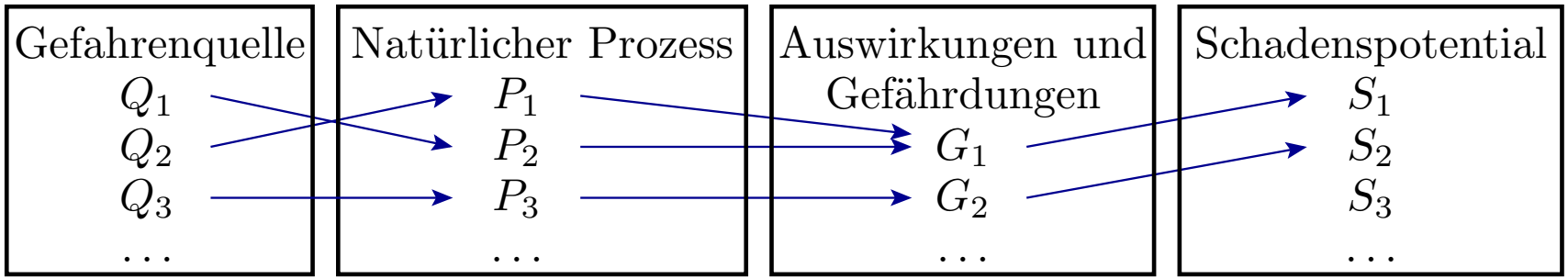
Die Sicherheitsanalyse hat zum Ziel, Nebeneffekte und Gefährdungen durch Systeme und Technologien aufzuzeigen, Schadensgrößen und Eintrittsrisiken einzuschätzen und über die Einsatzzulassung zu entscheiden. Vierstufiges Vorgehen nach [15]:

1. Identifizieren der Gefahrenquellen, auch unwahrscheinlicher.
2. Schadenspotential aus der zu erwartenden Schadensgröße und der Eintrittshäufigkeit abschätzen.
3. Szenarienbildung: Zusammenstellung kausaler Ketten aus Gefahrenquelle, natürlichem Prozess, potentieller Gefährdung und Schadenspotential.
4. Risiken-Nutzen-Analyse: Vergleichende Beurteilung von Nutzen und Gefährdung. Bei unakzeptabler Gefährdung keine Einsatzzulassung bzw. Erteilung von Verbesserungsauflagen für die sicherheitsrelevanten Eigenschaften.

[15] http://www.bats.ch/bats/biosicherheit/methodik/vorgehen.php

Konstruktion der kausalen Ketten:

- Zusammenstellen Gefahrenquelle $Q_i$.
- Konstruktion von Verbindungen über natürliche Prozesse (Pi) zu den möglichen Gefährdungen $G_i$.
- Zuordnung der Schadenspotenzials $S_i$.




Naheliegendes weitere Vorgehen wäre die Zuordnung von Wahrscheinlichkeiten zu den kausalen Beziehungen und die Abschätzung von Eintrittswahrscheinlichkeiten der Gefährdungen, ... Stand der Technik sind vereinfachte standardisierte Vorgehen, die das Haftungsrisiko für die Organisationen, die die Sicherheitsanalyse durchführen, die Systeme herstellen oder die Systeme einsetzen, beschränken.

# FMCA [DIN 25448 90], [3, S. 433]

Das bis hier beschriebene Vorgehen zur Sicherheitsanalyse dient zur Bewertung von Biotechnologien, von denen genau wie von IT-Systemen große Sicherheitsrisiken ausgehen können[16]. FMCA (Failure Mode, Effect and Criticality Analysis) beschreibt ein korrespondierendes Vorgehen für IT-Systeme:

- Zusammenstellung möglicher sicherheitskritischer Fehlfunktionen einschließlich der verfügbaren Informationen zu Art, Ursache und Folgen.
- Risikobewertung durch eine manuelle Zuordnung von Risikoprioritätszahlen durch Expertenbefragung.
- Erarbeitung von Maßnahmenvorschlägen nach absteigender Risikopriorität.

[16] Prinzipiell auch auf IT-Systeme anwendbar.

Berechnungsvorschrift für die Risikoprioritätszahl:

$$RPZ = XE \cdot XF \cdot XN$$

Die Faktoren $XE$, $XF$ und $XN$ sind über Expertenbefragungen zu erfassende Kennziffer zwischen 1 und 10

- $XE$ für die Eintrittswahrscheinlichkeit,
- $XF$ für die Folgekosten und
- $XN$ für das Risiko, das Fehler mit dieser Wirkung unentdeckt bleiben.

Die Modellierung der kausalen Ketten für die Schadensentstehung und die Gegenüberstellung von Nutzen und Gefährdung fehlen.

# Betriebssicherheit als Teilzuverlässigkeit

## Definition 6

Die Betreibssicherheit eines IT-Systems sei die mittlere Betriebsdauer ohne sicherheitskritische Fehlfunktion:

$$S = \frac{t_{\mathrm{B}}}{\xi_{\mathrm{S}}}$$

($t_{\mathrm{B}}$ – Betriebsdauer; $\xi_{\mathrm{S}}$ – Anzahl der beobachteten Fehlfunktion).

Gezählt werden alle sicherheitkritischen Fehlfunktionen und addiert alle Betriebszeiten, in denen gezählt wird. Problem: Die zu zählenden Fehlfunktionen sind sehr selten:

- Havarien von Kernkraftwerken: weltweit ca. 1/Jahr [17].

[17] 1952 (Ottawa, Kanada), 1955 (Idaho, USA), 1957 (Kyschtym, Russland; Windscale, GB), ...; http://de.wikipedia.org/wiki/Liste_von _Unfällen_ in_ kerntechnischen_ Anlagen#1940.E2.80.931949

- Autounfälle: in Deutschland $\approx 2 \cdot 10^6$ pro Jahr, ca. 75% durch menschliches Versagen, Technisches-Versagen angenommen 10%. Bei $4 \cdot 10^7$ in Deutschland zugelassenen Autos ergibts sich ca. ein Unfall durch technisches Versagen pro 5 Jahre und Auto. Technische Sicerheit ca. 5 Jahre.

In der Praxis wäre eine weitere Unterteilung der Sicherheitsangaben nach Schadensklassen notwendig, z.B.

- SK1: mittlere Nutzungsdauer ohne große Folgeschäden,
- SK2: mittlere Nutzungsdauer ohne nennenswerte Folgeschäden,
- SK3: mittlere Nutzungsdauer ohne Situationen in denen Folgeschäden aufgetreten sind oder hätten auftreten können.

Auf $\approx 2 \cdot 10^6$ pro Jahr entfallen ca. $4 \cdot 10^4$ tötliche Opfer, d.h. ca. 2 auf 100 Unfälle. Abschätzungsweise sind auch die Todesfälle durch technisches Versagen um den Faktor 50 geringer, d.h., die Betriebssicherheit, wenn statt Unfällen nur die Todesfälle gezählt werden, ist $50\times$ so groß, d.h. etwa 250 Jahre.

# Sicherheitsbewertung für den IT-Einsatz

Eine Neudefinition der Betriebssicherheit im vorgeschlagenen Sinne hätte erhebliche Vorteile für die Sicherheitsbewertung für den IT-Einsatz.

- IT-Einsatz kann einen Sicherheitsgewinn bewirken, aber das System selbst hat nur eine endliche Sicherheit, die den Sicherheitsgewinn zum Teil kompensiert oder sogar negiert.

Beispielabschätzung der Sicherheitsverbesserung eines fiktives IT-Systems zur Unterbindung des Fahrens mit überhöhter Geschwindigkeit.

- Sicherheit ohne Verbesserungsmaßnahmen: 5 Jahre (in Deutschland und beim Zählen aller Unfälle).
- Verbesserungspotenzial: Annahme 25% der Unfälle seien auf überhöhte Geschwindigkeit rückführbar. Sicherheit, wenn überhöhte Geschwindigkeit unterbunden wird:

$$S_{\mathrm{GG.Pot}} \approx \frac{5\,\mathrm{Jahre}}{1-25\%} \approx 6{,}67\,\mathrm{Jahre}$$

- Das zusätzliche IT-System habe eine noch umbekannte Sicherheit von $S_{\mathrm{ZS}}$ als mittlere Zeit, die das System selbst keine Unfälle verursacht.
- Die Gesamtsicherheit ergibt sich durch Addition der Problemraten, d.h. der Kehrwerte der Teilsicherheiten:

$$S_{\mathrm{GG}} \approx \frac{1}{\frac{1}{S_{\mathrm{GG.Pot}}} + \frac{1}{S_{\mathrm{ZS}}}}$$

- Die Gesamtsicherheit soll sich, damit sich das Zusatzsystem lohnt, um mindetens 20% auf 6 Jahre vergrößern. Welche Sicherheit $S_{\mathrm{ZS}}$ ist dafür von dem Zusatzsystem zu fordern?

$$S_{\mathrm{ZS}} \approx \frac{1}{\frac{1}{S_{\mathrm{GG}}} - \frac{1}{S_{\mathrm{GG.Pot}}}} \approx \frac{1}{\frac{1}{6\,\mathrm{Jahre}} - \frac{1}{6{,}67\,\mathrm{Jahre}}} \approx 60\,\mathrm{Jahre}$$

Die Sicherheit von 60 Jahren lässt sich wiederum auf Kosten für Fehlervermeidung, Test, Fehlerbeseitigung, ... zurückrechnen. ...

# Fehlervermeidung, Test, ... als Sicherheitseinflüsse

Für die Zuverlässigkeit wurde Folie 114 für ein ideal determinitischen System folgender Zusammenhang abgeschätzt:

$$Z \sim \frac{Q \cdot t_\mathrm{T} \cdot t_\mathrm{R}^{k+1}}{p_\mathrm{R} \cdot (1 - FC) \cdot N}$$

($N$ – Entstehungsaufwand in Entwurfsoperationen; $Q$ – Prozessgüte in Entwurfsoperationen je entstehender Fehler; $FC$ – Fehlerüberdeckung aller Tests zusammen; $t_\mathrm{T}$ – Testdauer bei Zufallstest; $t_\mathrm{R}$ – Reifedauer; $0 < k < 1$ – Exponent der Fehlernachweisdichte; $p_\mathrm{R}$ – Maskierungswahrscheinlichkeit eingebauter Kontrollfunktionen). Die Sicherheit als Teilzuverlässigkeit ist wesentlich größer und verhält sich in erster Näherung proportional zu Zuverlässigkigkeit:

$$S \sim \frac{Q \cdot t_\mathrm{T} \cdot t_\mathrm{R}^{k+1}}{p_\mathrm{R} \cdot (1 - FC) \cdot N}$$

Alle Maßnahmen zur Verbesserung der Zuverlässigkeit wirken sich in ähnlicher Weise auf die Betriebssicherheit aus. Darüber hinaus gibt es spezielle Maßnahmen, die nur auf die Minderung sicherheitkritischer Probleme, statt auf alle potentiellen Probleme abzielen, d.h. die gezielt die Betriebssicherheit verbessern:

- Sorgfältigerer Entwurf, gründlichere Tests und Inspektionen, ... der sicherheitskritischen Teile.
- Ruhestromprinzip (vergl. F3, Abschn. 5.1),
- ...

# Aufgaben

# Aufgabe 4.1: Zuverlässigkeit

1. Welche Zuverlässigkeit hat ein System im Dauerbetrieb, bei dem im Mittel pro Jahr 100 Fehlfunktionen durch Störungen, 200 Fehlfunktionen durch Bedienfehler und 500 Fehlfunktionen durch nicht erkannte Fehler auftreten?
2. Ein IT-System habe eine Zuverlässigkeit von 10h. Nach Erkennung und Beseitigung eines Fehlers erhöht sich die Zuverlässigkeit um 10%. Mit welcher Häufigkeit hatte dieser Fehler Fehlfunktionen verursacht?
3. Die Ausgabekontrolle eines Systems erkennt 99% aller Fehlfunktionen. Wie hoch muss die Korrekturwahrscheinlichkeit sein, damit sich die Zuverlässigkeit verzwanzigfacht?

# Aufgabe 4.2: Sicherheit

1. Um welchen Faktor erhöht sich die Sicherheit eines Systems, wenn die Fehlerüberdeckung des Tests von 80% auf 90% erhöht, und sich durch Beseitigung der Fehler, die am häufigsten Fehlfunktionen verursachen, die mittlere Anzahl der Fehlfunktionen je Fehler halbiert? (Die Häufigkeit der Fehlfunktionen durch Störungen und Fehlbedienungen sei vernachlässigbar und alle erkannten Fehler werden beseitigt.)

# Literatur

# 5. Literatur


[1] J. E. Aas and I. Sundsbo.
Harnessing the human factor for design quality.
IEEE Circuits and Devices Magazine, 11(3):24–28, 1995.

[2] Benedikte Elbel.
Zuverlässigkeitsorientiertes Testmanagement.
www.systematic-testing.com/.../Zuverlaessigkeitsorientiertes_Testmanagement.pdf, 2003.

[3] Peter Liggesmeyer.
Software-Qualität: Testen, Analysieren und Verifizieren von Software.
Spectrum, 2002.

[4] NSTL Test Report.
Microsoft Windows 2000 Professional – Comparison of the Reliability of Desktop Operating Systems.
http://??

[5] M. Thaler and M. Utesch.
Effektivität und Effizienz von Softwareinspektionen.
In Müllerburg et. al. (Hrdg.) Test, Analyse und Verifikation von Software, GMD-Bericht Nr. 260, pages 183–196. Oldenbourg, 1996.