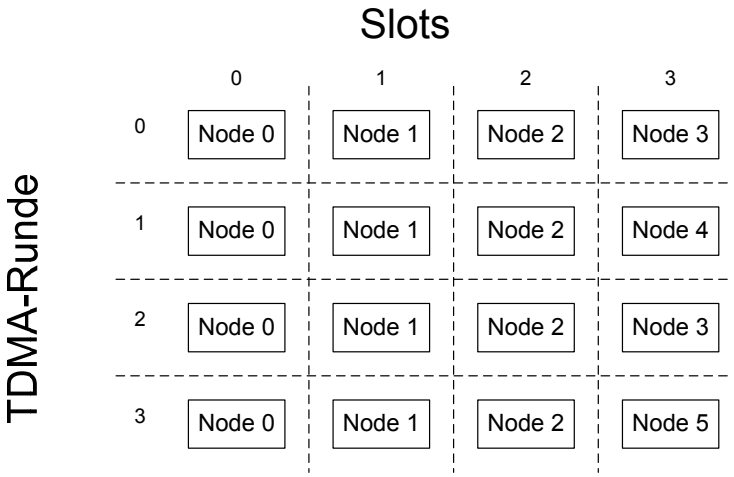
TTP: Arbeitsprinzip

- Die Controller arbeiten autonom vom Hostcomputer (notwendige Daten sind in MEDL enthalten)
  - für jede zu empfangende und sendende Nachricht: Zeitpunkt und Speicherort in der CNI
  - zusätzliche Informationen zur Ausführung des Protokolls
- In jeder TDMA-Runde sendet ein Knoten genau einmal
  - Unterscheidung zwischen
    - reellen Knoten: Knoten mit eigenem Sendeschlitz
    - virtuelle Knoten: mehrere Knoten teilen sich einen Sendeschlitz
    - Die Länge der Sendeschlitze kann sich dabei unterscheiden, für einen Knoten ist die Länge immer gleich
      ⇒ TDMA-Runde dauert immer gleich lang



Protokolldienste

- Das Protokoll bietet:
  - Vorhersagbare und kleine, nach oben begrenzte Verzögerungen aller Nachrichten
  - Zeitliche Kapselung der Subsysteme
  - Schnelle Fehlerentdeckung beim Senden und Empfangen
  - Implizite Nachrichtenbestätigung durch Gruppenkommunikation
  - Unterstützung von Redundanz (Knoten, Kanäle) für fehlertolerante Systeme
  - Unterstützung von Clustermoduswechseln
  - Fehlertoleranter, verteilter Uhrensynchronisationsalgorithmus ohne zusätzliche Kosten
  - Hohe Effizienz wegen kleinem Protokollaufwand
  - Passive Knoten können mithören, aber keine Daten versenden.
  - Schattenknoten sind passive redundante Knoten, die im Fehlerfall eine fehlerhafte Komponente ersetzen können.

Fehlerhypothese

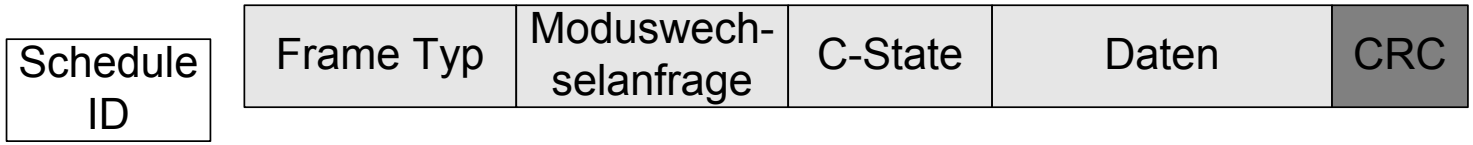
- Interne physikalische Fehler:
  - Erkennung einerseits durch das Protokoll, sowie Verhinderung eine babbling idiots durch Guards.
- Externe physikalische Fehler:
  - Durch redundante Kanäle können diese Fehler toleriert werden.
- Designfehler des TTP/C Kontrollers:
  - Es wird von einem fehlerfreien Design ausgegangen.
- Designfehler Hostcomputer:
  - Protokollablauf kann nicht beeinflusst werden, allerdings können inkorrekte Daten erzeugt werden.
- Permanente Slightly-Off-Specification-Fehler:
  - können durch erweiterte Guards toleriert werden.
- Regionale Fehler (Zerstören der Netzwerkverbindungen eines Knotens):
  - Folgen können durch Ring- und Sternarchitektur minimiert werden.

Zustandsüberwachung

- Das Protokoll bietet Möglichkeiten, dass Netzwerk zu analysieren und fehlerbehaftete Knoten zu erkennen.
- Der Zustand des Netzwerkes wird dabei im Kontrollerzustand (C-State) gespeichert.
- Der C-State enhält:
  - die globale Zeit der nächsten Übertragung
  - das aktuelle Fenster im Clusterzyklus
  - den aktuellen, aktiven Clustermodus
  - einen eventuell ausstehenden Moduswechsel
  - den Status aller Knoten im Cluster
- Das Protokoll bietet einen Votierungsalgorithmus zur Überprüfung des eigenen Zustands an.
- Ein Knoten ist korrekt, wenn er in seinem Fenster eine korrekte Nachricht versendet hat.
- Knoten können sich durch die Übernahme der Zeit und der Schedulingposition integrieren, sobald ein integrierender Rechner eine korrekte Nachricht sendet, erkennen in die anderen Knoten an.

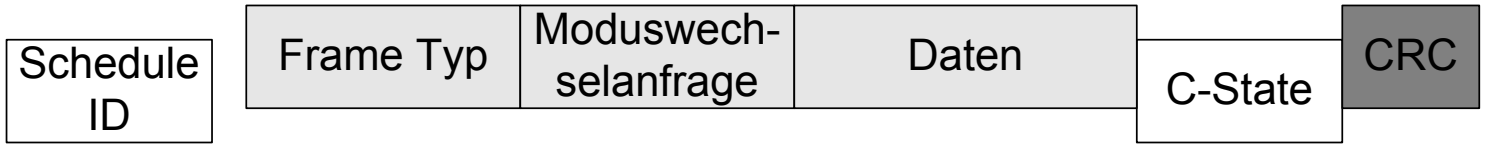
Datenpakete in TTP

- Paket mit explizitem C-State



- Kaltstartpaket

- Paket mit implizitem C-State



TTP: Clusterstart

- Der Start erfolgt in drei Schritten:
  1. Initialisierung des Hostcomputers und des Controllers
  2. Suche nach Frame mit expliziten C-State und Integration
  3. a) Falls kein Frame empfangen wird, werden die Bedingungen für einen Kaltstart geprüft:
     - Host hat sein Lebenszeichen aktualisiert
     - Das Kaltstart Flag in der MEDL ist gesetzt
     - die maximale Anzahl der erlaubten Kaltstarts wurde noch nicht erreicht

     Sind die Bedingungen erfüllt, sendet der Knoten ein Kaltstartframe.
  3. b) Falls Frame empfangen wird: Versuch zur Integration

TTP: Sicherheitsdienste / Synchronisation

- Sicherheitdienste:
  - Korrektheit: Alle Knoten werden über die Korrektheit der anderen Knoten mit einer Verzögerung von etwa einer Runde informiert.
  - Cliquenentdeckung: Es werden die Anzahl der übereinstimmenden und entgegen gesetzten Knoten gezählt. Falls mehr entgegen gesetzte Knoten gezählt werden, so wird ein Cliquenfehler angenommen.
  - Host/Kontroller Lebenszeichen: der Hostcomputer muss seine Lebendigkeit dem Kontroller regelmäßig zeigen. Sonst wechselt der Kontroller in den passiven Zustand.
- Synchronisation:
  - In regelmäßigen Abständen wird die Uhrensynchronisation durchgeführt.
  - Es werden die Unterschiede der lokalen Uhr zu ausgewählten (stabilen) Uhren (mind.4) anderer Rechner anhand den Sendezeiten gemessen.
  - Die beiden extremen Werte werden gestrichen und vom Rest der Mittelwert gebildet.
  - Die Rechner einigen sich auf einen Zeitpunkt für die Uhrenkorrektur.

Echtzeitfähige Kommunikation

Zusammenfassung

Zusammenfassung

- Die Eignung eines Kommunikationsmediums für die Anwendung in Echtzeitsystemen ist vor allem durch das Medienzugriffsverfahren bestimmt.
- Die maximale Wartezeit ist bei
  - CSMA/CD: unbegrenzt und nicht deterministisch ($\Rightarrow$ keine Eignung für Echtzeitsysteme)
  - CSMA/CA,tokenbasierten Verfahren: begrenzt, aber nicht deterministisch (abhängig von anderen Nachrichten)
  - zeitgesteuerten Verfahren: begrenzt und deterministisch.
- Die Priorisierung der Nachrichten wird von CSMA/CA und tokenbasierten Verfahren unterstützt.
- Nachteil der zeitgesteuerten Verfahren ist die mangelnde Flexibilität (keine dynamischen Nachrichten möglich).
- Trotz diverser Nachteile geht der Trend hin zum Ethernet.

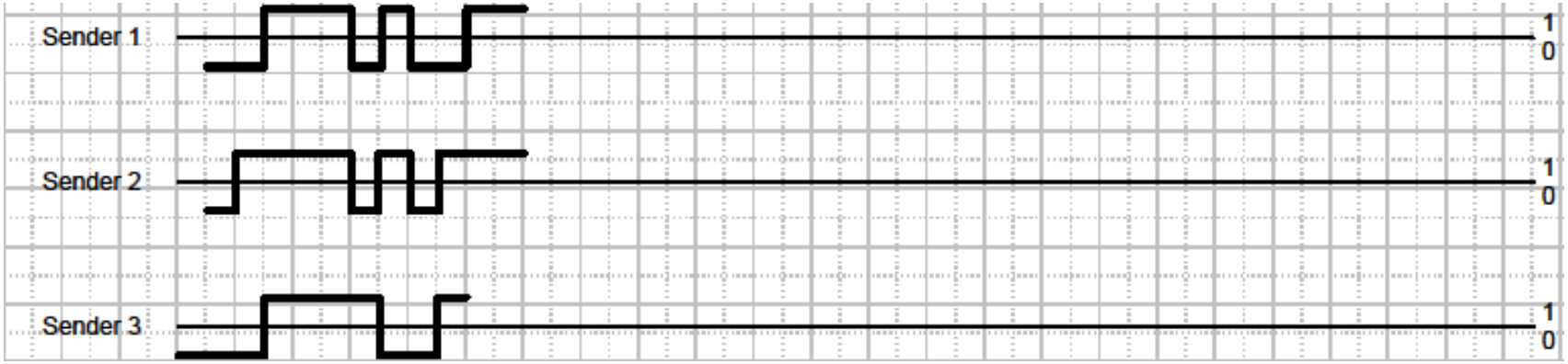
Trends: Real-Time Ethernet

- Es existieren verschiedene Ansätze
  - Beispiel: Ethercat von Beckhoff
    - Die Nachrichten entsprechen dem Standardnachrichtenformat von Ethernet
    - Pakete werden von einem Master initiiert und werden von den Teilnehmern jeweils weitergeleitet.
    - Jeder Knoten entnimmt die für ihn bestimmten Daten und kann eigene Daten anfügen.
    - Die Bearbeitung erfolgt on-the-fly, dadurch kann die Verzögerung minimiert werden.
  - Beispiel: Profinet von Siemens
    - Drei verschiedene Protokollstufen (TCP/IP – Reaktionszeit im Bereich von 100ms, Real-time Protocol - bis 10ms, Isochronous Real-Time - unter 1ms)
    - Profinet IRT benutzt vorher bekannte, reservierte Zeitschlitze zur Übertragung von echtzeitkritischen Daten, in der übrigen Zeit wird das Standard-Ethernet Protokoll ausgeführt

Klausurfragen

- Klausur Wintersemester 07/08
  - Erläutern Sie kurz die wesentlichen Unterschiede zwischen TokenRing, TokenBus und Ethercat in Bezug auf Topologie und Mediumszugriffverfahren.
- Wiederholungsfragen:
  1. Was ist der Unterschied zwischen dominanten und rezessiven Bits.
  2. Nennen Sie zwei Mechanismen zur Bitsynchronisierung und erklären Sie diese.
  3. Was ist der Unterschied zwischen CSMA/CD und CSMA/CA?
  4. Erläutern Sie zwei verschiedene Ansätze um Ethernet echtzeitfähig zu machen.
  5. Beurteilen Sie die Kommunikationsprotokolle Ethernet, CAN, TTP nach Ihrer Echtzeitfähigkeit und gehen Sie vor allem auf die Möglichkeit zur Vorhersage der maximalen Nachrichtenlatenz ein.

Klausur Wintersemester 07/08



In der Abbildung sehen Sie drei Knoten und Ihre jeweilige Nachricht für den Fall, dass der jeweilige Knoten als einziger senden würde. Dabei entspricht die Länge eines Bits einem Kästchen.

Gehen Sie davon aus, dass für die Lösung der Aufgabe alle Daten bitsynchron übertragen werden. Das JAM-Signal soll aus einer Folge von 5 0-Bits bestehen. Das 0-Bit ist dominant. Zwischen zwei Nachrichten gibt es eine Pause (interframe gap) von mindestens 3 Bits.

- Zeigen Sie für die angegebenen Nachrichten einen möglichen Ablaufplan in CSMA-CD.
- Geben Sie den entsprechenden Plan in CSMA-CA an.
- Für ein konkretes Netzwerk ist die maximale Signallaufzeit mit einer Zeiteinheit angegeben. Welche der angegebenen Bitübertragungsdauern würden Sie für CSMA/CA auswählen. Geben Sie eine knappe Begründung für Ihre Antwort.
  - 0,5 Zeiteinheiten
  - 1 Zeiteinheit
  - 4 Zeiteinheiten
  - 10 Zeiteinheiten

Kapitel 9

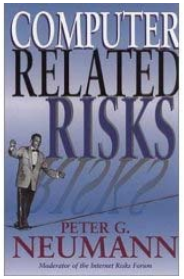
Fehlertoleranz

Inhalt

- Einleitung
- Grundlagen
- Fehlertoleranzmechanismen
- Quantitative Bewertung fehlertoleranter Systeme

Literatur

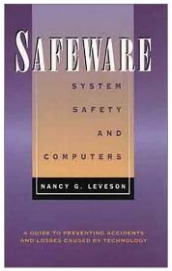
Dhiraj K. Pradhan: Fault-Tolerant Computer System Design, Prentice Hall 1996



Peter G.Neumann: Computer Related Risks, ACM Press 1995

W.A.Halang, R.Konakovsky: Sicherheitsgerichtete Echtzeit-systeme, Oldenburg 1999

Nancy G.Leveson: Safeware, Addison-Wesley 1995



Klaus Echtle: Fehlertoleranzverfahren, Springer-Verlag 1990 (elektronisch unter http://dc.informatik.uni-essen.de/Echtle/all/buch ftv/)

http://www.system-safety.org/

Fehlertoleranz

Negativbeispiele (Motivation)

Sicherheit fängt schon im Kleinen an

- Lexikalische Konventionen können Fehler verhindern.
- Negatives Beispiel: FORTRAN
  - In FORTRAN werden Leerzeichen bei Namen ignoriert.
  - Variablen müssen in FORTRAN nicht explizit definiert werden
- Problem in Mariner 1:
  Aus einer Schleife

```
DO 5 K = 1,3
```

  wird durch versehentliche Verwendung eines Punktes

```
DO5K=1.3
```

  eine Zuweisung an eine nicht deklarierte Variable.
  ⇒ Zerstörung der Rakete, Schaden 18,5 Millionen $

Ariane 5 (1996)

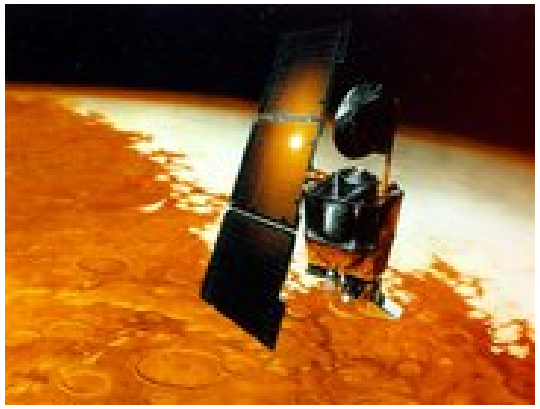
- Selbstzerstörung bei Jungfernflug:
- Design:
  - 2 redundante Meßsysteme (identische Hardware und Software) bestimmen die Lage der Rakete (hot-standby)
  - 3-fach redundante On-Board Computer (OBC) überwachen Meßsysteme
- Ablauf:
  - Beide Meßsysteme schalten aufgrund eines identischen Fehlers ab
  - OBC leitet Selbstzerstörung ein
- Ursache:
  - Wiederverwendung von nicht-kompatiblen Komponenten der Ariane 4 (Speicherüberlauf, weil Ariane 5 stärker beschleunigt)

Weitere Informationen unter
http://sunnyday.mit.edu/accidents/Ariane5accidentreport.html

Therac-25 (1985-1987)

- Computergesteuerter Elektronenbeschleuniger zur Strahlentherapie
- Das System beinhaltete 3 schwere Mängel:
  - Sicherheitsprüfungen im Programm wurden durch einen Softwarefehler bei jeder 64. Benutzung ausgelassen (wenn ein 6-bit Zähler Null wurde).
  - Behandlungsanweisungen konnten mittels Editieren am Bildschirm so abgeändert werden, dass die Maschine für die nächste Behandlung nicht den gewünschten Zustand einnahm (nämlich Niederintensität).
  - Mehrere Sicherheitsverriegelungen, die beim Vorgängermodell Therac-20 in Hardware realisiert waren, wurden nicht übernommen, sondern durch Software ersetzt.
- Folgen:
  - Mehrere Patienten erhielten anstatt der vorgesehenen Dosis von 80-200 rad Strahlungsdosen von bis zu 25000 rad (mehrere Tote und Schwerverletzte).
- Weitere Informationen unter http://sunnyday.mit.edu/papers/therac.pdf

Mars Climate Orbiter (1998)



- Verglühen beim Eintritt in die Atmosphäre
- Ursache:
  - Verwendung von unterschiedlichen Maßeinheiten (Zoll, cm) bei der Implementierung der einzelnen Komponenten.
  - Mangelnde Erfahrung, Überlastung und schlechte Zusammenarbeit der Bodenmannschaften

Weitere Informationen unter http://mars.jpl.nasa.gov/msp98/orbiter/

Explosion einer Chemiefabrik (1992)

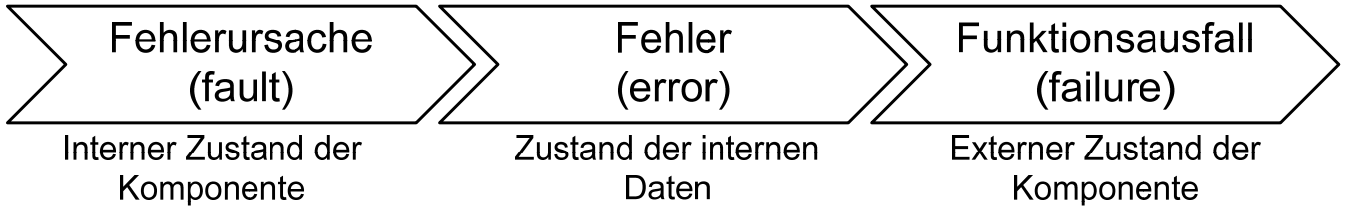
- Explosion einer holländischen Chemiefabrik aufgrund eines Bedienfehlers
- Ablauf:
  - Computergesteuertes Mischen von Chemikalien.
  - Operateur (in Ausbildung) verwechselt beim Eintippen eines Rezeptes 632 (Harz) mit 634 (Dicyclopentadien).
- Folgen:
  - Explosion fordert 3 Menschenleben, Explosionsteile finden sich noch im Umkreis von 1 km.

Fehlertoleranz

Definitionen

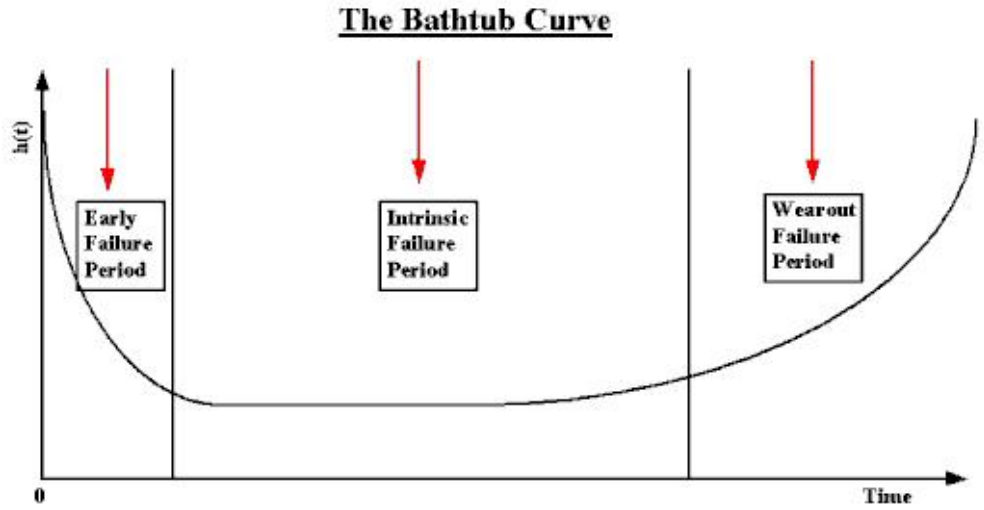
Begriffe:

- **Fehlerursache** (**fault**)**:** physikalischer Fehler oder Störstelle in einer Hardware- oder Softwarekomponente
- **Fehler** (**error**): Erscheinungsform eines Fehlzustands, z.B. durch das Abweichen eines Wertes vom erwarteten Wert in den internen Daten
- **Funktionsausfall** (**failure**): Ausfall oder fehlerhafte Durchführung von Funktionen eines Systems, Auftritt an der Benutzerschnittstelle



Fehlerrate

- Die Fehlerrate gibt die erwartete Anzahl an Fehler eines Gerätes oder eines Systems für eine gegebene Zeitperiode an.
- Typischerweise wird die Fehlerrate als konstant angenommen (siehe Badewannenkurve – gültig für Hardwarefehler) und mit λ bezeichnet. Typische Einheit der Fehlerrate ist Fehler pro Stunde.



Aspekte des Begriffs Fehlertoleranz

- Systeme zum Einsatz in sicherheitskritischen Anwendungen erfordern ein hohes Maß an Systemstabilität (**dependability**).
- Dieser Begriff umfasst:
  - Zuverlässigkeit
  - Sicherheit
  - Verfügbarkeit
  - Leistungsfähigkeit
  - Robustheit
  - Wartbarkeit
  - Testbarkeit

Zuverlässigkeit

- Definition: Die Zuverlässigkeit (**reliability**) eines Systems ist eine Funktion $0 \leq R(t) \leq 1$, definiert als die bedingte Wahrscheinlichkeit, dass das System korrekt während des Intervals $[t_0, t]$ funktioniert unter der Annahme, dass das System zum Zeitpunkt t_0 korrekt arbeitete.
- Wird eine konstante Fehlerrate angenommen, so kann die Zuverlässigkeit durch folgende Gleichung angegeben werden:

$$R(t) = e^{-(\lambda^*(t-t_0))}$$

Sicherheit

- Sicherheit (**safety**) ist die Wahrscheinlichkeit $0 \leq S(t) \leq 1$, dass ein System zum Zeitpunkt t entweder korrekt arbeitet oder seine Funktion auf eine Art und Weise beendet, so dass es nicht die Funktionsweise anderer Systeme gestört oder Menschen gefährdet werden.
- Sicherheit ist damit ein Maßstab für die Fähigkeit eines Systems auf eine sichere Art und Weise auszufallen.

Verfügbarkeit

- Verfügbarkeit (**availability**) wird als eine Funktion $0 \leq A(t) \leq 1$ über die Zeit ausgedrückt, die die Wahrscheinlichkeit angibt, dass ein System zum Zeitpunkt *t* korrekt arbeitet. Im Gegensatz zur Zuverlässigkeit wird bei der Verfügbarkeit neben der Häufigkeit der Dienstausfälle auch die Dauer der Reparaturen und Wartungsarbeiten berücksichtigt.
- Während bei der Zuverlässigkeit die Korrektheit des Systems zu allen Zeitpunkten eines gegebenen Intervalls gefordert wird, gibt die Verfügbarkeit die momentane Wahrscheinlichkeit der korrekten Ausführung des Systems an.
- Eine hohe Verfügbarkeit ist beispielsweise bei transaktionsbasierten Systemen, z.B. ein Fluglinienreservierungssystem, nötig. Wartungsarbeiten und Reparaturen sollten schnell durchgeführt werden, eine andauernde korrekte Funktion im Sinne der Zuverlässigkeit wird hingegen nicht gefordert.

Leistungsfähigkeit

- In vielen Fällen ist es möglich und sinnvoll Systeme zu konstruieren, die nach Auftreten von Hardware oder Softwarefehler in einzelnen Komponenten (siehe spätere Einführung von Fehlerbereichen) in einem degradierten Modus weiterarbeiten.
- Unter **Leistungsfähigkeit** (**performability**) wird eine Funktion $0 \leq P(L,t) \leq 1$ über der Zeit verstanden, die eine Wahrscheinlichkeit angibt, dass die Funktionalität des Systems zum Zeitpunkt t mindestens das Niveau L erreicht. Im Gegensatz zur Zuverlässigkeit, bei der immer nur die Wahrscheinlichkeit angegeben wird, dass alle Funktionen korrekt funktionieren, können nun auch Teilmengen betrachtet werden.

Robustheit, Wartbarkeit, Testbarkeit

- Unter **Robustheit** (**robustness**) eines Systems wird die Fähigkeit verstanden auch unter erschwerten Betriebsbedingungen (z.B. Fehleingaben (siehe Chemiefabrik) oder widersprüchlichen Meßwerten) die korrekte Funktionalität zu wahren.
- **Wartbarkeit** (**maintainability**) ist ein Maßstab für die Reparaturfreundlichkeit eines Systems. Quantitativ kann die Wartbarkeit als die Wahrscheinlichkeit M(t) ausgedrückt werden, dass das fehlerhafte System innerhalb einer Zeitdauer t repariert werden kann.
- **Testbarkeit** (**testability**) ist ein Maßstab für die Möglichkeit bestimmte Eigenschaften eines Systems zu testen. So kann es möglich sein, bestimmte Tests zu automatisieren und als Mechanismen in das System zu integrieren.
- Die Testbarkeit eines Systems ist durch die hohe Bedeutung der schnellen Fehleranalyse direkt mit der Wartbarkeit eines Systems verbunden.

Konzepte zur Erhöhung der Systemstabilität

- Systemstabilität kann durch Anwendung der folgenden Konzepte erreicht werden:
  - Fehlervermeidung
    - Designmethoden + Werkzeugunterstützung
    - Modellierung mit Anwendung von Verifikations- und Validierungsmethoden
  - Fehlerentfernung
    - Einheitentests
    - Integrationstests
    - Back-To-Back Testing (Vergleich von Resultaten unterschiedlicher Versionen bei N-Versionsprogrammierung)
  - Fehlertoleranz