

Summary: Dependable Systems

An dieser Zusammenfassung und der zugehörigen Formelsammlung kann gerne auf [Github](#) mitgewirkt werden!

Inhaltsverzeichnis

Basics	1
Allgemeine Begriffe und Definitionen	1
Means to attain dependability (and security)	3
Mathematical Models	3
(Arbitrary) Block Diagram	3
Maintainability	4
Markov Models	4
Generalized Stochastic Petri Nets (GSPN)	4
Reliability Growth Models	5

Basics

Allgemeine Begriffe und Definitionen

- Definition von Dependability¹:
 - original: Dependability ist die Fähigkeit, Dienste anzubieten, auf die gerechtfertigterweise vertraut werden kann.
 - alternativ: Dependability eines Systems beschreibt die Fähigkeit, Ausfälle zu verhindern, die frequenter auftreten oder gravierendere Folgen haben, als zulässig.

¹IEEE: “Basic concepts and taxonomy of dependable and secure computing”, ([Link](#))

- System: Eine Einheit, die mit anderen Einheiten (i.e. Systemen) interagiert.
- Umgebung/Umfeld: Alle Systeme, mit denen ein bestimmtes System interagiert.
- Systemgrenze: Die Schnittstelle zwischen einem System und dessen Umgebung.
- Funktion eines Systems: gibt an, was ein System tun soll, d.h. seine Intention.
- Verhalten eines Systems: gibt an, was ein System unternimmt, um seine Funktion zu erfüllen. Wird als Folge von Zuständen angegeben.
- Struktur eines Systems: gibt an, was einem System sein Verhalten ermöglicht -> Komponenten.
- Service Interface: der Teil der Systemgrenze zu einem von möglicherweise mehreren Benutzern.
- externe Zustände: die Teilmenge an Zuständen eines Systems, die über das Service Interface wahrnehmbar sind.
- interne Zustände: alle Zustände eines Systems, die nicht extern sind.

Attribute von Dependability:

- Zuverlässigkeit (reliability): die Wahrscheinlichkeit, dass ein System über eine gewisse Periode hinweg seine Funktion erfüllt
- Verfügbarkeit (availability): prozentueller Anteil der Zeit, in der das System seine Funktion erfüllt
- Wartbarkeit (maintainability)
- Sicherheit (safety): Wahrscheinlichkeit, dass ein System während einer gewissen Zeitperiode kein spezifiziertes, unerwünschtes Verhalten zeigt.
 - oft wird Security zu Safety dazu gezählt. Wichtigste Schutzziele: CIA-Triade
- Integrität

Lebensphasen eines Systems:

- Entwicklungsphase
 - Konzeption
 - Design, Entwicklung, Validation, Verifikation
- Benutzungsphase
 - service delivery
 - service outage
 - service shutdown
 - maintainance

Klassifikation von Faults durch:

- Phase of creation or occurrence (development vs. use phase)
- System boundaries (internal vs. external)
- Phenomenological cause (natural vs. human-made)
- Dimension (hardware vs. software)
- Objective (malicious vs. non-malicious)
- Intent (deliberate vs. non-deliberate)
- Capability (accident vs. incompetence)
- Persistence (permanent vs. transient)

Means to attain dependability (and security)

- Fault Prevention: takes place in both hardware and software
- Fault Tolerance: error detection, damage confinement, recovery, fault treatment
- Fault Removal: verification, diagnosis, correction
- Fault Forecasting

Mathematical Models

Reliability and Failure Probability:

- $Q(t)$... Failure probability: the probability that a system will *not* conform to its specification throughout $[0, t]$
- $R(t)$... Reliability: the probability that a system *will* conform to its specification throughout $[0, t]$

$$R(0) = 1, R(\infty) = 0, R(t) = 1 - Q(t)$$

Failure Probability Density Function:

$$f(t) = \frac{dQ(t)}{dt} = -\frac{dR(t)}{dt}$$

Failure Rate:

$$\lambda(t) = \frac{f(t)}{R(t)} = -\frac{dR(t)}{dt} \cdot \frac{1}{R(t)} \quad \text{Unit: FIT (failures in time) := 1 failure in } 10^9 \text{ hours}$$

Constant Failure Rate:

- $\lambda(t) = \lambda$ (const.)
- $\Rightarrow f(t) = \lambda e^{-\lambda \cdot t}$
- $\Rightarrow R(t) = e^{-\lambda \cdot t}$

Weibull distributet failure rate:

- $\lambda(t) = \alpha \lambda (\lambda t)^{\alpha-1}$
- $\Rightarrow f(t) = \alpha \lambda (\lambda t)^{\alpha-1} \cdot e^{-(\lambda t)^\alpha}$
- $\Rightarrow R(t) = e^{-(\lambda t)^\alpha}$

(Arbitrary) Block Diagram

Serienschaltung:

$$R_{ser.}(t) = \prod_{i=0}^{n-1} R_i(t) = 1 - \prod_{i=0}^{n-1} (1 - Q_i(t))$$

Parallelschaltung:

$$Q_{par.}(t) = \prod_{i=0}^{n-1} Q_i(t) = 1 - \prod_{i=0}^{n-1} (1 - R_i(t))$$

For constant Failure Rate:

$$\bullet \quad \lambda_{ser.} = \sum_{i=0}^{n-1} \lambda_i$$

- For serial blocks, failure rate remains constant. For parallel blocks, however, failure rate is generally not constant and formula is not as neat.

Extension to arbitrary block diagrams:

- arbitrary connections between blocks
- switches for passive stand-by redundancy
- voters

still not taken into account: maintainability

Maintainability

Mean Time To Failure (MTTF): $MTTF = \int_0^{\infty} t \cdot f(t) dt$

Applied to constant failure rate:

- $MTTF = \int_0^{\infty} t \cdot \lambda e^{-\lambda t} dt = \lambda^{-1}$
- $MTTF_{ser.} = \frac{1}{\lambda_1 + \lambda_2 + \dots + \lambda_n}$

Meant Time To Repair (MTTR): $MTTR = \int_0^{\infty} t \cdot f_r(t) dt$

Repair Rate (analogous to failure rate): μ

For constant repair rate: $MTTR = \mu^{-1}$

Steady State Availability (A): $A = \frac{MTTF}{MTTF + MTTR}$

Mission time t_m : during mission, no repair is possible

Mission Reliability: $R(t_m)$

Markov Models

General properties:

- Suitable for modeling of:
 - arbitrary structures (active, passive and voting redundancy)
 - systems with complex dependencies (assumption of independent failures is no longer necessary)
 - coverage effects
- Markov property: The system behavior at any time instant is independent of history (except for the last state)
- Restriction to constant failure rates

Generalized Stochastic Petri Nets (GSPN)

more complex mechanisms $\Rightarrow$ less complicated models

Reliability Growth Models

System is treated as blackbox $\Rightarrow$ no need to identify separat components

Diese Zusammenfassung ist noch sehr unvollständig. Bei Interesse kann sie auf [Github](#) erweitert werden. Mitwirken ist ausdrücklich erwünscht!
