Armin R. Mikler, PhD
University of North Texas

Fehlertoleranz in Computersystemen und Netzwerken

mathematik und informatik

FernUniversität in Hagen

Vorwort

Dieser Kurs besteht aus vier Kurseinheiten, die eine Einführung in das Gebiet der fehlertolerierenden Systeme vermitteln. Die erste Einheit macht den Leser durch eine Reihe von Beispielen zunächst mit dem Problem der Fehlertoleranz bekannt. Darüber hinaus sollen in der ersten Kurseinheit die Unterschiede zwischen Fehlertoleranz und Risikominimierung genauer untersucht werden. Obwohl das Gebiet der fehlertolerierenden Systeme im Allgemeinen im Rahmen der Informatik definiert ist, werden die grundlegenden Prinzipien anhand allgemeiner Beispiele dargestellt. Insbesondere werden hier verschiedene Arten von Fehlern definiert und untersucht, unter welchen Bedingungen ein System trotz Auftretens solcher Fehler weiterarbeiten kann. In der zweiten Kurseinheit werden die theoretischen Grundlagen behandelt, die im Wesentlichen ein Basiswissen der Wahrscheinlichkeitsrechnung und der Statistik umfassen. Hierbei werden die Prinzipien jeweils mittels Beispielen demonstriert, die es ermöglichen die behandelten Methoden auf verschiedene Probleme anzuwenden. In der dritten Kurseinheit werden Ansätze vorgestellt, die zu einer fehlertolerierenden Datenspeicherung und Datenübertragung beitragen. Dabei werden im Speziellen die Unterschiede zwischen Fehlererkennung und Fehlerkorrektur demonstriert. Darüber hinaus wird in der dritten Kurseinheit der Begriff der Redundanz genauer untersucht. Die vierte Kurseinheit fokussiert auf die Entwicklung fehlertoleranter Kommunikationssysteme und verteilte Rechnersysteme mittels protokollbasierter Ansätze. Verschiedene Protokollelemente werden hier auf ihren Beitrag zur Fehlertoleranz hin untersucht. Dabei dient das 7-Schichten OSI Modell dazu die unterschiedlichen Protokollelemente den jeweiligen Systemkomponenten zuzuordenen.

Am Ende jeder Kurseinheit findet der Leser eine Sammlung von Übungsaufgaben, die dazu beitragen einen Bezug zu der oft sehr abstrakten Materie der fehlertoleranten Systeme herzustellen. Zur Lösung dieser Aufgaben muss der Leser die jeweilige Kurseinheit sorgfältig bearbeiten und möglicherweise einen Blick in die verfügbare Literatur werfen.

Die Themen der einzelnen Kurseinheiten basieren auf aktuellen Fachbüchern der Fehlertoleranz, Kommunikationssysteme, Betriebssysteme, verteilte Systeme und der Wahrscheinlichkeitsrechnung und Statistik. Die hier vorgestellte Materie wird im Allgemeinen von allen Fachbüchern dieser Gebiete abgedeckt. Darüber hinaus bietet es sich an, einige spezielle Journalartikel zu konsultieren, die zu einem die historische Entwicklung der fehlertoleranten Systeme darstellen und zum anderen den aktuellen Forschungsaufwand darstellen.

Inhaltsverzeichnis

Kapitel 1

Prinzipien der Fehlertoleranz und Ausfallsicherheit

1.1 Lernziel der Kurseinheit

In dieser Kurseinheit sollen die Begriffe der Fehlertoleranz und der Risikominimierung definiert werden und darüber hinaus die folgenden Konzepte vorgestellt werden:

- Kategorien von Fehlern und Defekten,
- Redundanz,
- Kosten der Fehlertoleranz,
- Ausfallsicherheit und Verfügbarkeit.

Anhand verschiedener Beispiele, die nicht exklusiv aus der Informatik stammen, werden diese Konzepte genauer untersucht. Dabei dienen die am Ende des Kapitels gestellten Übungsaufgaben dazu das Verständnis der vorgestellten Konzepte selbst zu überprüfen. Im Folgenden werden verschiedene Konzepte vorgestellt, die in der zweiten Kurseinheit aufgegriffen und genauer behandelt werden.

1.2 Fehlertoleranz, Risiko und Sicherheit

Täglich treffen wir Entscheidungen, an denen sich unsere persönliche Einstellung zu möglichen Fehlern und unsere Risikobereitschaft widerspiegeln. In diesem Kapitel werden wir versuchen solche Konzepte etwas präziser zu definieren und Entscheidungen oder Verhaltensweisen genauer zu untersuchen. Zwar sind die Begriffe der Fehlertoleranz und der Systemsicherheit meist im Rahmen der Informatik definiert, dennoch ist es nützlich und hilfreich diese Konzepte im Kontext des täglichen Lebens zu untersuchen und zu verstehen. Während die oben aufgeführten Konzepte prinzipiell verschieden sind, ist es oft problematisch sie in expliziten Fällen zu unterscheiden und zu differenzieren. So ist zum Beispiel nicht sofort erkennbar, wie die Entscheidung, sehr früh zum Flughafen aufzubrechen, um das pünktliche Eintreffen zu gewährleisten, einzuordnen ist. Sollte man diese Entscheidung

als fehlertolerierendes oder risikominimierendes Verhalten einstufen? Die Antwort ist ein definitives „Es kommt darauf an...“. Um diese Frage zu beantworten, muss man zunächst genauer definieren, was man unter den Begriffen Fehler und Risiko in dieser Situation versteht. Ist als Fehler das Verpassen des Flugs definiert, so wird zweifelsohne ein frühes Aufbrechen zum Flughafen die Toleranz eines solchen Fehlers nicht erhöhen. Wird allerdings ein unvorhergesehener Stau auf der Autobahn oder gar eine Auto-Panne als möglicher Fehler definiert, so kann man argumentieren, dass die Maximierung der verfügbaren Zeit, den Flughafen rechtzeitig zu erreichen, tatsächlich zu einer Erhöhung der Fehlertoleranz führen kann. Das bedeutet, man kann den definierten Fehler bis zu einem bestimmten Grade tolerieren und trotz einer Verzögerung den Flughafen zeitgerecht erreichen. Wenn man ein System auf seine Fehlertoleranz hin untersuchen möchte, muss man zunächst dessen Ziele und Erwartungswerte genau festlegen.
Während das Konzept der Fehlertoleranz versucht die Effekte eines auftretenden Fehlers zu minimieren, so beschäftigt sich das Konzept der Systemsicherheit damit, die Wahrscheinlichkeit des Auftretens eines Fehlers zu minimieren oder sogar komplett auszuschließen. Die Sicherheit eines Systems wird durch den Designprozess bestimmt, in dem das System so konzipiert wird, dass die Mehrzahl der möglichen Fehlerquellen berücksichtigt werden. Diesbezüglich werden verschiedene Parameter definiert, in deren Schranken das System fehlerfrei operieren muss. Bestimmte Grenzwerte für Temperatur, Feuchtigkeit, Strahlung, Druck, usw. sind Beispiele solcher Parameter. In der Informatik konzentrieren sich solche Parameter allerdings häufig auf Datenbereiche, Datenstrukturen, numerische Grenzwerte und Abweichungen, die für ein Programm oder ein System definiert werden. Die Systemsicherheit wird generell durch verschiedene Tests verifiziert. Die Erstellung dieser Tests, insbesondere im Bereich der Softwareentwicklung, ist nicht trivial, da die Anzahl der Testfälle häufig exponentiell mit der Anzahl der zu berücksichtigenden Parameter ansteigt. Auch in anderen Anwendungsbereichen verlangt die Erstellung adäquater Tests detailliertes Wissen über das System und dessen Anwendungsumgebung.
Als Beispiel eines Systems, von dem extreme Ausfall-Sicherheit gefordert wird, dient der bekannte Flugschreiber (oder *Black Box*), die in jedem Flugzeug installiert ist und verschiedene Systemgrößen der Maschine und Kontrollentscheidungen der Piloten aufzeichnet. Sie dient dazu, im Falle eines Absturzes, die Situation zu rekonstruieren, um die Ursache durch detaillierten Einblick in den Status des Flugzeugs zur Zeit des Absturzes zu erlangen. Natürlich darf diese *Black Box* bei einem Absturz nicht zerstört werden und muss daher so konzipiert werden, dass sie gegen auftretende Kräfte und mögliche Umwelteinflüsse geschützt ist.
Die oben aufgeführten Beispiele sollen helfen, die Konzepte der Fehlertoleranz und der Systemsicherheit zu differenzieren und dem Leser einen ersten Eindruck der Vielfältigkeit dieses Themas zu vermitteln. In der Informatik werden Systeme häufig abstrahiert und durch ein zusammen arbeitendes Netz der verschiedenen Funktionskomponenten dargestellt. Dadurch lässt sich ein solches System einfacher analysieren und auf Fehlerhäufigkeit und Fehlerwahrscheinlichkeit untersuchen.

1.2.1 Fehlertoleranz

Definition 1 (Fehlertoleranz)**.** *Ein System ist* **fehlertolerant***, wenn es trotz des Auftretens unvorhergesehener Fehler weiterhin in der Lage ist seine Funktionen korrekt auszuführen.*

Ein fehlertolerantes System hat somit die Eigenschaft, dass es bei Auftreten von Fehlern graziös degeneriert, d.h. möglicherweise mit verringerter Effizienz oder Genauigkeit weiterarbeitet.

1.2.2 Systemsicherheit

Definition 2 (System Sicherheit)**.** *Ein System gilt als* **sicher***, wenn die Wahrscheinlichkeit auftretender Fehler minimiert ist.*

Das Konzept der Systemsicherheit befasst sich mit der Minimierung des Fehlerrisikos. Natürlich kann man das Auftreten von Fehlern nie völlig ausschließen.

1.2.3 Denkaufgaben

In jedem der folgenden Kontexte sollen je zwei Maßnahmen identifiziert werden, die sowohl die Ausfallsicherheit als auch die Fehlertoleranz erhöhen. Dabei soll genau definiert werden, was als Fehler angesehen werden muss.

1. Fahrzeug (z.B. Motorrad, PKW, usw.)
2. Investition im Aktien- oder Geldmarkt
3. Internet
4. Reiseplanung
5. Industrielle Projektplanung

1.3 Klassifikation möglicher Fehler

In der Umgangssprache verwenden wir Begriffe wie *Fehler, Defekt, Ausfall, und Error.* Während der jeweilige Kontext klar bestimmt was diese Ausdrücke bedeuten, muss man sie für die Fehleranalyse eines Systems genauer definieren. So beschreiben die Begriffe *Fehler, und Defekt* generell das fehlerhafte Verhalten von Systemkomponenten, wobei die Bezeichnung *Error* beschreibt, wie sich ein Fehlverhalten eines Systems manifestiert.
So wird häufig das Verhalten eines logischen Schaltkreises, der unabhängig von den jeweiligen Eingangsgrößen immer den Wert 0 aufweist, als Fehler deklariert. Ein Error tritt auf, wenn dieser Wert dann weiterverarbeitet wird, z.B. in einem Addierer, dessen Ergebnis dann demzufolge fehlerhaft (oder Error-behaftet) ist.
Betrachtet man eine Funktion, bei der dem Programmierer ein *Fehler* bei der Implementierung unterlaufen ist und daher nur positive Funktionswerte berechnet werden. Es präsentiert sich dieser Fehler allerdings nur als *Error*, falls die Funktion mit Parametern aufgerufen wird, die zu einem Ergebnis mit negativen Werten führen.

Generell kann man das Konzept eines Fehlers noch weiter differenzieren:

- **Permanente Fehler** sind Fehler, die bei Auftreten die jeweilige Komponente oder das System permanent außer Betrieb setzen.
- **Transiente Fehler** sind solche Fehler, die dazu führen, dass eine Komponente oder ein System für ein bestimmtes Zeitintervall nicht korrekt arbeitet. Nach diesem Zeitintervall ist der Fehler nicht mehr präsent und das korrekte Systemverhalten ist wiederhergestellt.
- **Sporadische Fehler** sind generell immer präsent, zeigen sich allerdings nur in unregelmäßigen Intervallen als Fehlverhalten des Systems.

Transiente und sporadische Fehler lassen sich häufig sehr schwer diagnostizieren, da ihr Auftreten oft keinem vorhersehbaren Muster folgt. So ist es zum Beispiel schwer einen Programmierfehler zu identifizieren, der nur bei bestimmten Konfigurationen der Eingangswerte oder Parameter auftritt. Hier setzt man spezielle Verfahren der Softwaretechnik ein, die sich mit dem Testen von Software und der Identifikation von Fehlern befassen, die daten- oder konfigurationsabhängig sind.
Viele Fehler können in **unkritische** und **kritische** Fehler klassifiziert werden. So ist beispielsweise ein Fehler, der ein System zu einem kompletten Stillstand bringt als *unkritisch* anzusehen, wenn keine falschen Resultate generiert werden. In diesem Kontext umfassen die *kritischen* Fehler solche, die verfälschte Werte an andere Komponenten weitergeben und somit fehlerhafte Ergebnisse erzeugen. Diese Fehler werden auch häufig als byzantinische Fehler bezeichnet, die wir in einer folgenden Kurseinheit genauer untersuchen werden. Exemplarisch für einen kritischen Fehler betrachtet man ein System, das durch das Austauschen von Messwerten mit anderen Systemen oder Komponenten Entscheidungen koordiniert. Wenn das System allen Komponenten die gleichen (korrekten) Messwerte mitteilt, können alle Teilsysteme unabhängig voneinander Entscheidungen treffen, die mit den bekannten Messwerten korrespondieren. Wenn nun das System aber unterschiedliche (falsche) Messwerte an die Teilsysteme verteilt, so werden diese nicht in der Lage sein, ein konsistentes Ergebnis zu errechnen und somit möglicherweise eine inkonsistente Entscheidung treffen.

1.4 Konzept der Redundanz

Prinzipiell basieren die fehlertolerierenden Eigenschaften eines Systems auf dem Management und Nutzen von **Ressourcen**. Die Verfügbarkeit von mehr Ressourcen als minimal notwendig, um das Funktionieren des Systems zu gewährleisten, wird als **Redundanz** bezeichnet. Dabei umfassen diese Ressourcen nicht nur die notwendigen Hardware- und Softwarekomponenten, sondern auch Zeit und Raum. So kann möglicherweise die räumliche Verteilung oder Ausbreitung von Komponenten zur Fehlertoleranz eines Systems beitragen. Dieser Ansatz wird insbesondere im Gebiet der verteilten Rechnersysteme genutzt, indem ein System so konzipiert wird, dass es keinen zentralen Punkt enthält der durch Auftreten eines Fehlers das gesamte System funktionsunfähig macht.

Die wohl am häufigsten angewandte Form der Redundanz ist die Hardwareredundanz. Dabei werden funktionskritische Elemente eines Systems dupliziert (oder multipliziert), oder es wird eine spezielle Funktion auf unabhängige Komponenten verteilt. Es soll dadurch gewährleistet werden, dass bei Auftreten eines Fehlers die Funktion weiterhin ausgeführt werden kann, obwohl es dabei zu einer Reduzierung der Qualität oder Effizienz kommen kann. Als Beispiel dafür betrachte man eine Zweikreisbremse eines PKW, bei dem die Vorderräder und Hinterräder durch zwei separate Bremskreise gesteuert werden. Sollte in einem Bremskreis ein Fehler auftreten (z.B. durch ein Leck in der Bremsleitung), so ermöglicht der andere Bremskreis noch das Fahrzeug abzubremsen. Die Bremsleistung wird durch das Auftreten eines solchen Fehlers jedoch verringert.

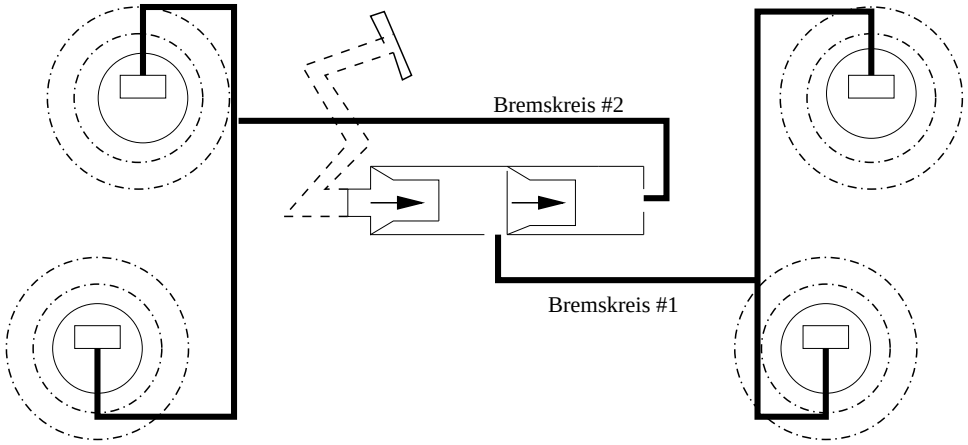

Abbildung 1.1: Zweikreisbremse eines PKW

Weiterhin lässt sich das Konzept der Redundanz in **statische** und **dynamische** Redundanz unterteilen. Während bei statischer Redundanz die redundanten Ressourcen aktiv sind, werden sie bei der dynamischen Redundanz bei Bedarf als *Ersatzkomponenten* aktiviert. Daher wird durch statische Redundanz das Auftreten eines Fehlers maskiert und es ist nicht unmittelbar erkennbar, dass eine Systemkomponente ausgefallen ist. Ein Beispiel für statische Redundanz ist die Verfügbarkeit von Plattenspeichern oder Prozessoren. Das Betriebsystem wird immer versuchen alle verfügbaren Ressourcen zu nutzen, um damit einen Lastausgleich im Rechnersystem zu erreichen. Das Ausfallen dieser Komponenten hat zur Folge, dass diese anfallende zusätzliche Last entweder auf andere Prozessoren verteilt werden muss, oder, im Fall von Speicherfehlern, eine neue Speicherverteilung der Prozesse stattfinden muss.
Ein dynamisches Umschalten auf Ersatzkomponenten erfolgt häufig automatisch, wie zum Beispiel bei einem Netzteil, welches durch einen automatischen Fault-Over im Fehlerfall ein zweites Netzteil aktivieren kann.

1.4.1 Blockdiagramme

Ein System oder ein Prozessablauf wird häufig mittels kanonischer Blockdiagramme abstrahiert. So werden beispielsweise die Komponenten eines Systems und deren Relationen