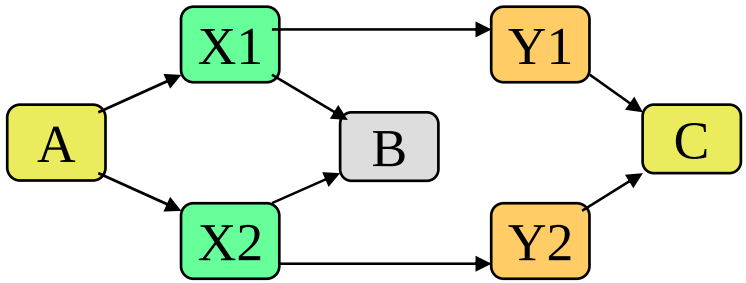
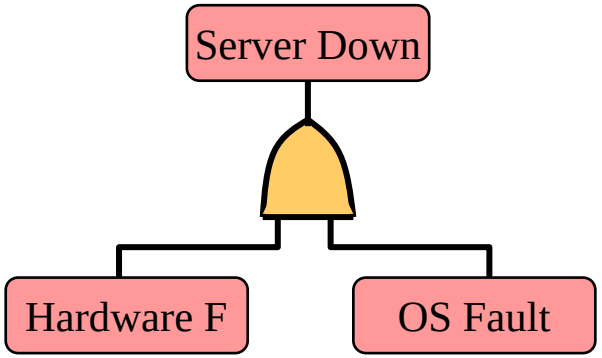
Modellierung und Analyse eingebetteter und verteilter Systeme

—

Thread „Zuverlässigkeit“ *Teil 2*



- Begriffe und Kenngrößen
- Fehler und Ausfälle
- Zuverlässigkeitsblockdiagramm
- Struktur-Funktionsmodell
- Fehlermodell und Ausbreitung
- Redundanz
- Fehlertoleranzverfahren
- Qualitätssicherung, Analyse und Bewertung
- Formale Ansätze



F: Funktionaler Thread – Inhalte

- **Begriffe und Kenngrößen**
- **Fehler und Ausfälle**
- **Zuverlässigkeitsblockdiagramm**
- **Strukturfunktionsmodell**
- **Fehlermodell und Ausbreitung**
- **Redundanz**
- **Fehlertoleranzverfahren**
  - **Fehlerdiagnose**
  - **Fehlerbehandlung**
- **Qualitätssicherung, Analyse und Bewertung**
  - **FMEA**
  - **HAZOP**
  - **Fault Trees**
- **Formale Ansätze**

Literatur

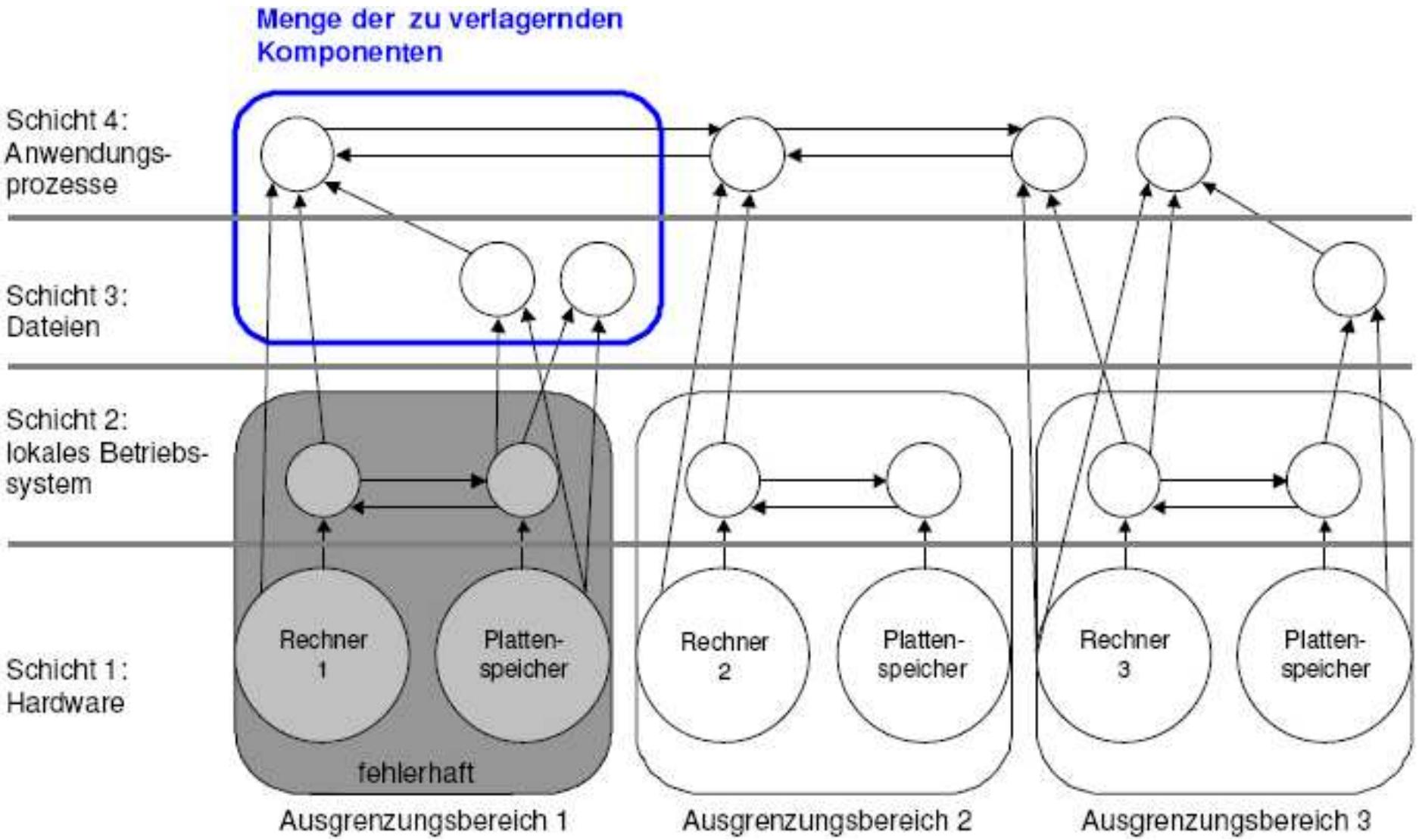
K. Echtle: Fehlertoleranzverfahren, Springer Verlag, 1990.
D. P. Siewiorek, R. S. Swarz: The theory and practice of reliable system design; Digital Press, 1982.
Blockley, David: Engineering Safety, Mcgraw Hill, 1992.
Leveson, Nancy: Safeware: System Safety and Computers, Addison Wesley, 1995.
W. Fredrich: Lehrblattsammlung Zuverlässigkeit und Qualitäts-sicherung, Uni Rostock, Fakultät Informatik und Elektrotechnik, 2003.

Z8: Fehlerbehandlung – Rekonfiguration

- Ausgrenzen fehlerhafter Komponenten
  - Ausgliedern fehlerhafter Komponenten
  - Eingliedern von Ersatzkomponenten
  - Verlagern von Funktionen auf Ersatzkomponenten
    - » Abschalten defekter Rechner eines Mehrrechnersystems,
      Verlagern eines Prozesses auf intakten Rechner
      (reicht u.U. nicht aus, weil Rechner-Fehler sich auf Prozess ausbreitete:
      zusätzliche Fehlerbehandlung für Prozess erforderlich, z.B. Rücksetzen)
    - » Freigabe der Bindung eines fehlerhaften Dienstes,
      Binden eines Ersatzdienstes,
      Anforderung von Operationen des Ersatzdienstes
- Basis: Strukturelle Redundanz
- Fehlerbehandlungsbereich: Ausgrenzungsbereich
  - Menge der Komponenten, die bei einem Rekonfigurationsverfahren stets als Ganzes aus- bzw. einzugliedern ist

Z8: Fehlerbehandlung – Rekonfiguration

- Beispiele zu Ausgrenzungsbereichen und Verlagerungsobjekten
  - Speicherbereiche: Speicherobjekte
  - Peripherieankopplungen: Gerätetreiber
  - Rechner-Verbindungen: Nachrichtenwege
  - Rechner: Dateien und Anwendungsprozesse
  - Rechenprozesse: Aufgaben
  - Server-Prozesse: Aufträge
  - Services: Bindungen und Operationsanforderungen
- Rekonfiguratoren: Verlagerungsziele bestimmen
  - schnelle Reaktion
  - Fehlertoleranz des neuen Systems
  - Günstige Verarbeitungsleistung (graceful Degradation)
  - geringer Kommunikationsaufwand



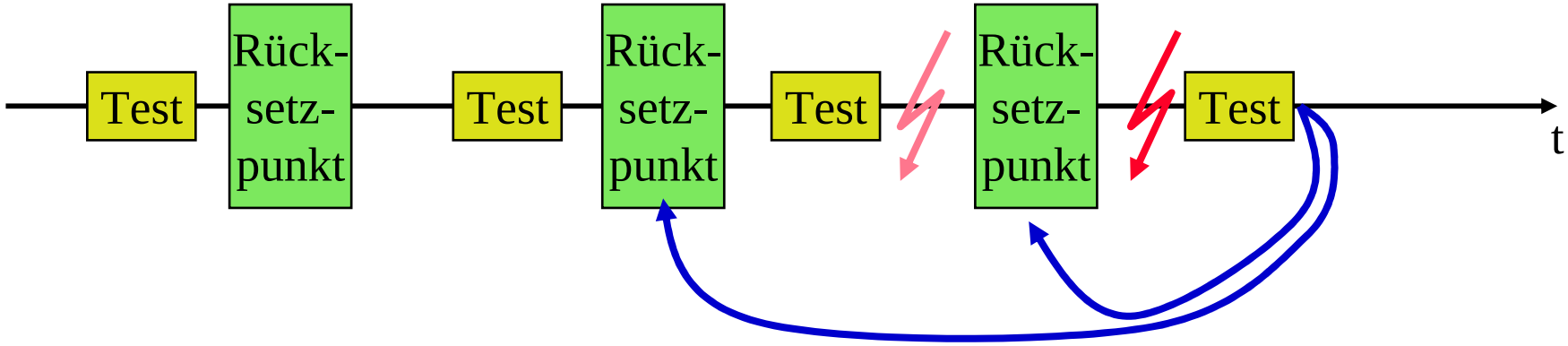
Ausgrenzungsbereiche und Menge der zu verlagernden Komponenten in einem schichtenstrukturierten Mehrrechnersystem (vereinfacht dargestellt). Ausgrenzungsbereich 1 ist Verlagerungsursprung; die beiden anderen Ausgrenzungsbereiche sind mögliche Verlagerungsziele

Beispiel aus K. Echtle: Fehlertoleranzverfahren

Z8: Fehlerbehandlung – Rückwärtsbehebung

- Voraussetzung
  intermittierender Fehler aufgetreten oder *permanenter Fehler* durch *Rekonfigurierung* ausgegrenzt
  (➔ kombinierte Verfahren)
- ***Rückwärtsbehebung***
  Komponenten in einen Zustand versetzen, den sie bereits in der Vergangenheit angenommen hatten oder als konsistenten Zustand hätten annehmen können.
  "Konsistent" bedeutet, dass die lokalen Komponentenzustände und die aktuellen Interaktionen mit anderen Komponenten die Spezifikation nicht verletzen
  - Rücksetzpunkte erstellen
  - Rücksetzen
- Basis: Zeitredundanz
- Fehlerbehandlungsbereich: Rückwärtsbehebungsbereich
  Menge gemeinsam rücksetzbarer Komponenten
  - einzelne Prozesse rücksetzen
  - Mengen interagierender Prozesse rücksetzen

Z8: Fehlerbehandlung – Prozess rücksetzen



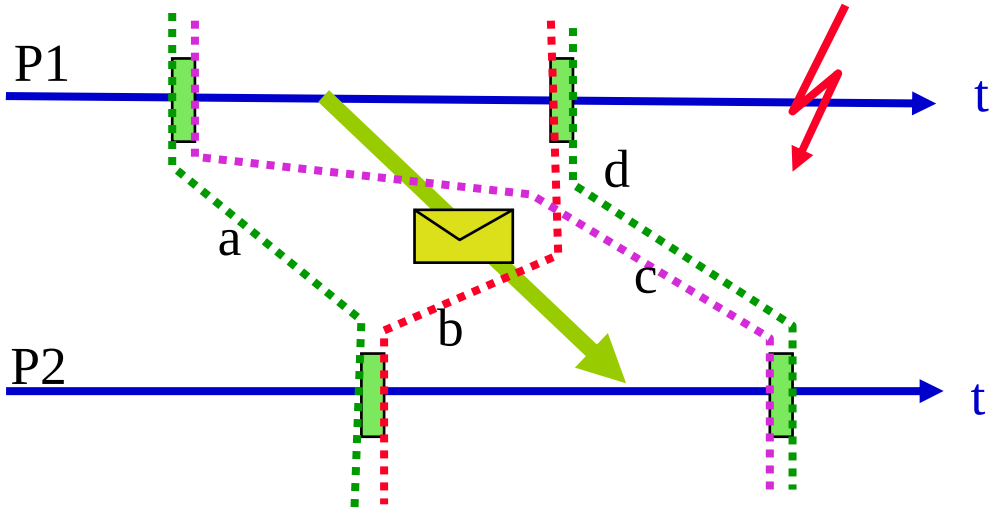
- Einfacher Spezialfall: Rücksetzen in Startzustand – Prozessneustart
- Umfassende Lösung: Transaktionskonzept

- Speicherung
  - als Zustandskopie
  - als Zustandsdifferenz
  - als Änderungshistorie

- Löschen älterer Rücksetzpunkte
- ISO/OSI: Kommunikationssteuerung
  - verletzliche (billige) Nebensynchronisationspunkte und stabile (aufwendige) Hauptsynchronisationspunkte, alles vor dem letzten Hauptsynchronisationspunkt kann gelöscht werden

Z8: Fehlerbehandlung – Prozessmenge rücksetzen

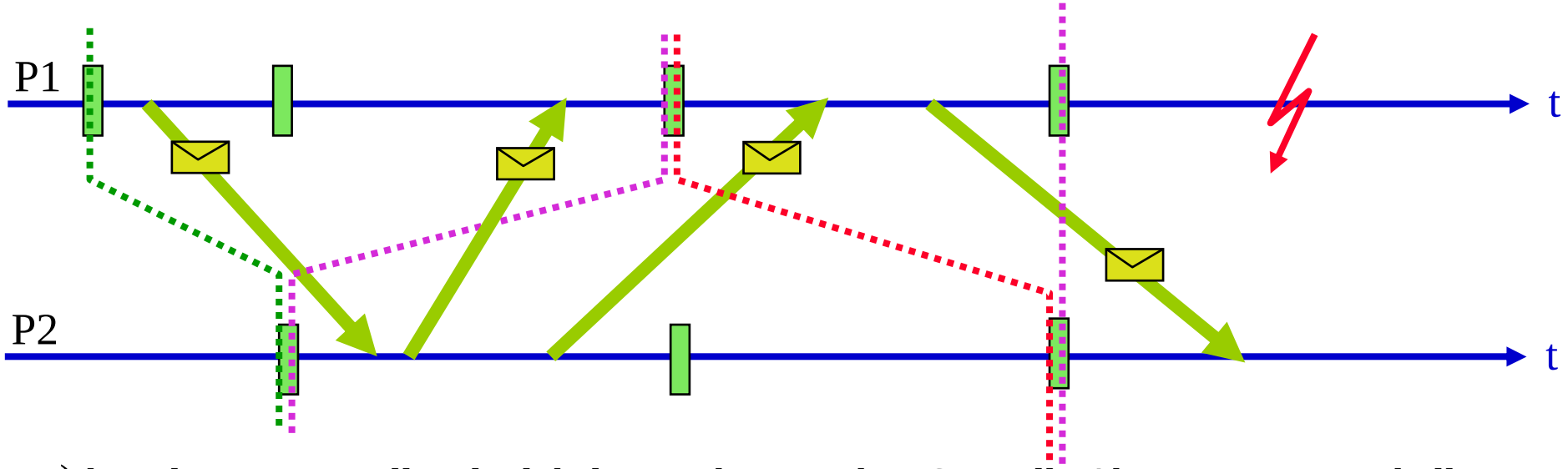
- Rücksetzlinie und Nachrichtenaustausch



a) zulässige Rücksetzlinie, die Nachricht ist weder gesendet noch empfangen
b) unzulässige Linie, die Nachricht ist schon gesendet und noch nicht empfangen, das Rücksetzen führt dazu dass P2 vergeblich auf die Nachricht wartet
c) unzulässige Linie, die Nachricht ist noch nicht gesendet aber schon empfangen, das Rücksetzen führt dazu, dass die Nachricht wieder gesendet wird und deshalb 2-mal bei P2 ankommt
d) zulässige Linie, die Nachricht ist schon gesendet und schon empfangen

Z8: Fehlerbehandlung – Prozessmenge rücksetzen

- Rücksetzlinie und Domino-Effekt



a) koordiniertes Erstellen der lokalen Rücksetzpunkte: Spezielle Abstimmungsprotokolle

b) Speichern der Nachrichtenhistorie, um lokal Rücksetzen zu können

c) Rücksetzpunkt generell bei jeder Kommunikationsaktion erstellen

d) Nachrichten erhalten Sequenzzahlen. Empfänger erkennen und ignorieren Duplikate. Deshalb sind nur noch „Nachrichten aus der Vergangenheit“ problematisch.

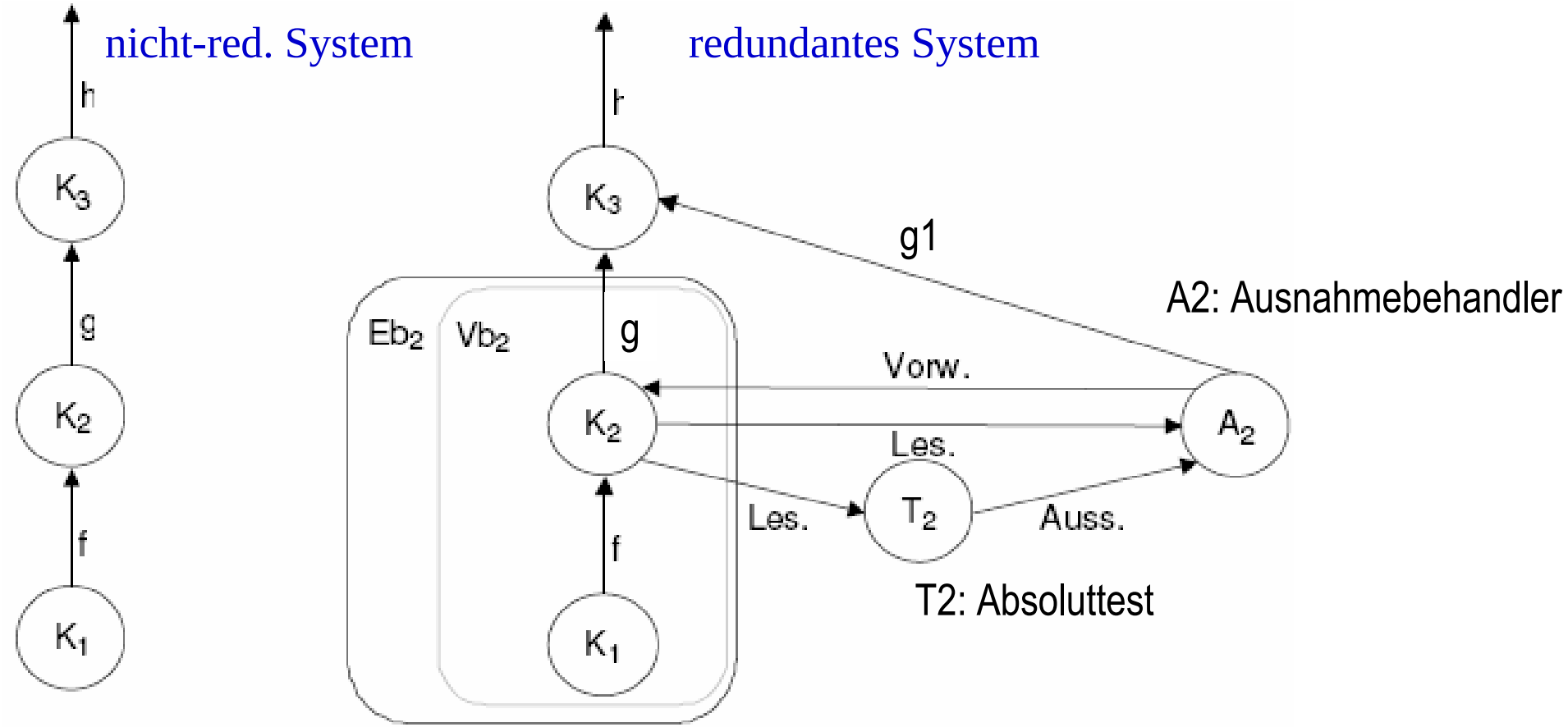
Z8: Fehlerbehandlung – Vorwärtsbehebung

- Begriff bezieht sich nicht auf „Zukunft prophezeien“, sondern auf „keine geretteten alten Zustandsinformationen verwenden“
- ***Vorwärtsbehebung***
  Komponenten aus dem Fehlerzustand heraus in einen konsistenten Zustand versetzen, von dem aus ein Weiterarbeiten möglich ist
  - Besonderheiten erkennen
  - Besonderheiten behandeln

- Anwendungsabhängige Mechanismen, kaum verallgemeinerbare Lösungen
- Oft nötig, wenn keine Zeitredundanz zur Verfügung steht,
  oder wenn Rücksetzen nicht möglich ist, weil auch physikalische Zustände rückgesetzt werden müssten (z.B. Steuerung einer Flugzeuglandung)
  ➔ ***wachsende Bedeutung in eingebetteten Systemen***

- Implementierung durch
  Ausnahmebehandler

Z8: Fehlerbehandlung – Vorwärtsbehebung



K1 und K2: Einzelfehler- und zugleich Vorwärtsbehebungsbereich (Eb2 bzw. Vb2): g wird auch bei Fehler erbracht. T2 und A2 haben lesenden Zugriff (Funktion "Les.") auf K2. Bei Testaussage "fehlerhaft" (Funktion "Auss."), kann A2 die Funktion g in evtl. eingeschränkter Form übernehmen (Funktion g1). Weiterhin versetzt A2 die Komponente K2 noch vor Ablauf der maximalen Bearbeitungsdauer (Zeitredundanz t_R) in einen fehlerfreien Zustand (Funktion "Vorw.")

Beispiel aus K. Echtle: Fehlertoleranzverfahren

Z8: Fehlerbehandlung – Vorwärtsbehebung

Vorteile

- Der Aufwand an ***struktureller Redundanz*** ist **gering**: Nur *Absoluttests* und die erst im *Ausnahmebetrieb* zu aktivierenden *Ausnahmebehandler* sind hinzuzufügen. Replikation der Verarbeitungseinheiten und Rücksetzpunkte erübrigen sich.
- Der **Laufzeitaufwand** im *Normal-Fehlertolerierungsbetrieb* wird nur von den *Absoluttests* verursacht und ist in diesem Sinn **minimal**.

Nachteile

- Vorwärtsbehebung ist **nicht transparent**, sondern anwendungsabhängig.
- Der **Entwurfsaufwand** lässt sich kaum auf mehrere Anwendungen umlegen und ist daher relativ **hoch.**
- Das Gelingen der Vorwärtsbehebung hängt **vom Schwierigkeitsgrad der Anwendung** und **von den Fähigkeiten des Entwerfers** ab.
- Nur durch ***Absoluttest*** erkennbare Fehler sind überhaupt tolerierbar (sofern das primäre Subsystem keine interne statische Redundanz aufweist). Die erzielte *Fehlererfassung* begrenzt die Güte des Verfahrens.

Programmierung von Ausnahmebehandlern

- Wünschenswert sind Programmiersprachen (vgl. Java, Ada)
  - welche die explizite Festlegung von *Vorwärtsbehebungsbereichen* gestatten,
  - die über ein Laufzeitsystem mit *Absoluttests* verfügen, wobei die Testaussagen dem Programm zugänglich sind, und
  - die im Fehlerfall implizit zu dem jeweiligen *Ausnahmebehandler* verzweigen.
- *Ausnahmebehandler* Algorithmus (Entwurf im Einzelfall), Ideen:
  - zerstörte Variableninhalte aus noch fehlerfreien rekonstruieren,
  - die bereits eingetretenen irreversiblen Fehlerauswirkungen durch geeignete **Gegensteuerung** reduzieren
  - wenigstens die wichtigsten Aufgaben erfüllen, d. h. zu einem **Notbetrieb** überzugehen
  - einen **sicheren Zustand** zu erreichen
  - falls Rücksetzpunkte in irgendeiner Form verfügbar sind, auf diese **zurückzusetzen** *(dann ist es keine Vorwärtsbehebung mehr)*

Z8: Fehlerbehandlung – Vorwärtsbehebung

Fortsetzung nach Ausnahmebehandlung

- ***Terminierungs-Fortsetzung***
  mit dem *Ausnahmebehandler werden* auch die Module des zugehörigen *Vorwärtsbehebungsbereichs* beendet (vgl. Java-Konstrukt)
  - wird in der Regel verwendet, weil einfacher zu programmieren
- ***Wiederaufnahme-Fortsetzung***
  nach Beendigung des *Ausnahmebehandlers* erfolgt Rücksprung an diejenige Stelle des zugehörigen *Vorwärtsbehebungsbereichs* zurückzuspringen, von der aus die fehlerbedingte Verzweigung in den Ausnahmebehandler erfolgt ist
  - Der Ausnahmebehandler muss alle nach der Wiederaufnahme benötigten Variableninhalte in einen konsistenten Zustand versetzen
  - Die Wiederaufnahme des primären Programmstücks erfolgt ähnlich dem Rücksprung nach einer Unterbrechung an derjenigen Programmstelle, die der zuletzt ausgeführten unmittelbar folgt.

Ausnahme—Weiterleitung

- Hat der Entwerfer keine Idee zur Fehlerbehandlung im lokalen Kontext, wohl aber in einem umfassenderen Kontext, so empfiehlt sich eine ***Ausnahme-Weiterleitung*** über mehrere *Vorwärtsbehebungsbereiche* hinweg

Z8: Fehlerbehandlung – Vorwärtsbehebung

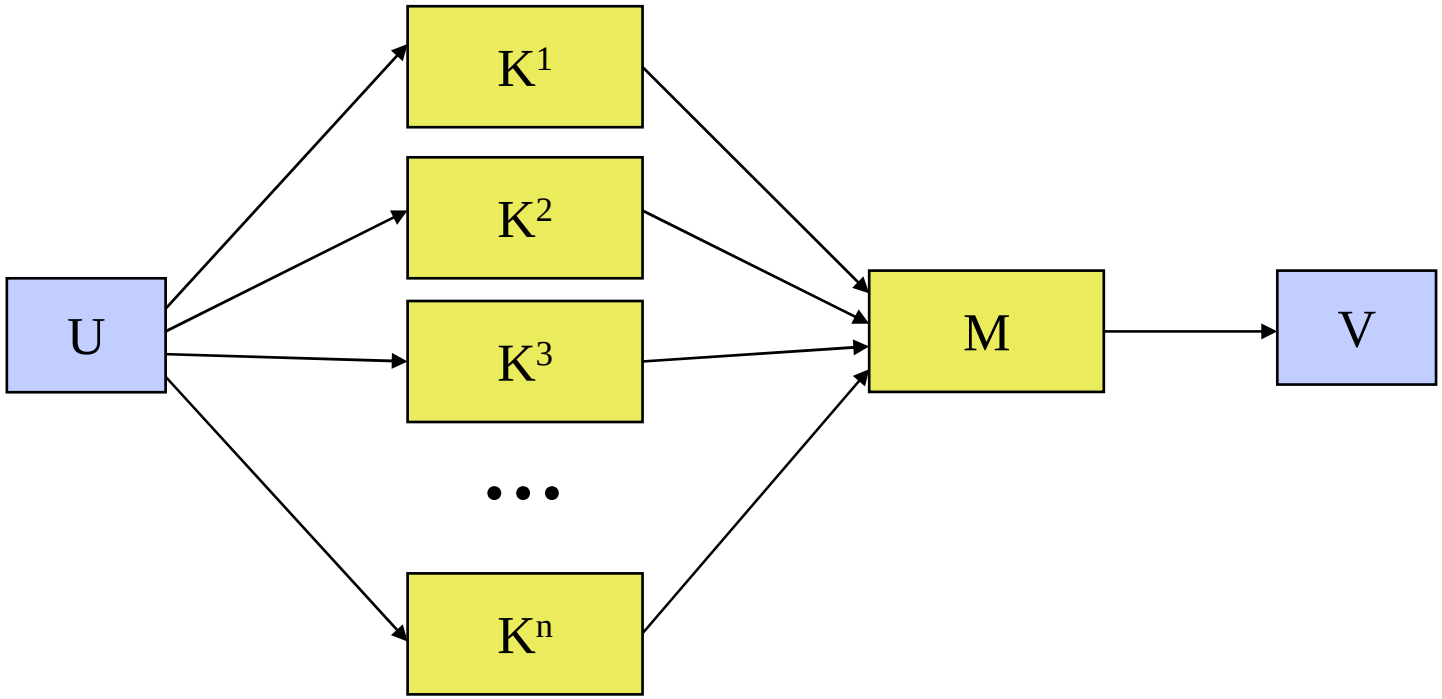
Beispiele

- Bei der Bearbeitung eines Werkstücks tritt ein Geräte-Fehler auf, das Werkstück ist deshalb nicht maßhaltig → Aussondern des Werkstücks in den Ausschuss, Weitermachen mit dem nächsten.
- In einer Waschmaschine wird die Temperatur überwacht, geht ein Messwert verloren, wird ein Mittelwert der zuletzt gemessenen Werte verwendet.
- Zur besseren Kapazitätsauslastung von Maschinen, die Werkstücke individuell bearbeiten, könnten die zugeführten Werkstücke geprüft werden, um eine Vorhersage ihrer Bearbeitungsdauer und eine günstige Verteilung auf die verfügbaren Maschinen zu ermöglichen. Fällt der prüfende Rechner aus, so könnte ein allgemeiner Erfahrungswert die Bearbeitungsdauer schätzen.
- Ein Regelkreis enthält einen Rechner, der Soll- und Ist-Wert einliest, und daraus entsprechend einer Regelfunktion eine Stellgröße berechnet und diese an ein Stellglied ausgibt. Nach einer fehlerhaften Berechnungen wird einfach keine Stellgröße ausgegeben, so dass das Stellglied seine bisherige Einstellung beibehält (Achtung: Permanente Fehler erkennen und anders behandeln).
- Ein Rechner soll zwei Flüssigkeiten A und B genau im Verhältnis x : y mischen, indem er über Ventile den Flüssigkeits-Zustrom einstellt und die Eigenschaften des Gemischs misst. Wurde aufgrund eines Rechnerfehlers die Menge der Flüssigkeit A verdreifacht, was ein Mischungsverhältnis von 3×x : y ergibt, so kann zur Vorwärtsbehebung die vierfache Gemisch-Menge hergestellt werden (Achtung: Behälter-Kapazität).

Z8: Fehlerbehandlung – Fehlermaskierung

- Ergebnisse werden mehrfach berechnet
  - ein fehlerfreies Ergebnis wird ausgewählt
  - fehlerhafte Ergebnisse werden ausgeblendet und ignoriert
- Die auswählende Fehlertoleranz-Instanz enthält quasi eine Maske die nur fehlerfreie Ergebnisse durchlässt
  - Verfahren heißt ***Fehlermaskierung***
  - Fehlertoleranz-Instanz heißt ***Maskierer***
  - Maskierer oft als ***Mehrheitsentscheider*** ausgebildet (Relativtest)
    - » n aus m – Mehrheitsentscheidung
    - » die m Ergebnisse werden in disjunkten Fehlerbereichen berechnet
- Basis: Ergebnisse haben *Informationsredundanz*, sie wird zur *Fehlerkompensierung* benutzt
- Besondere Eigenschaft
  - Die Fehlerdiagnose ist implizit.
  - Man benötigt keine weiteren Techniken zur Fehlerdiagnose und auch keine Fehlerausgrenzung.

Z8: Fehlerbehandlung – Fehlermaskierung



m-aus-n-System

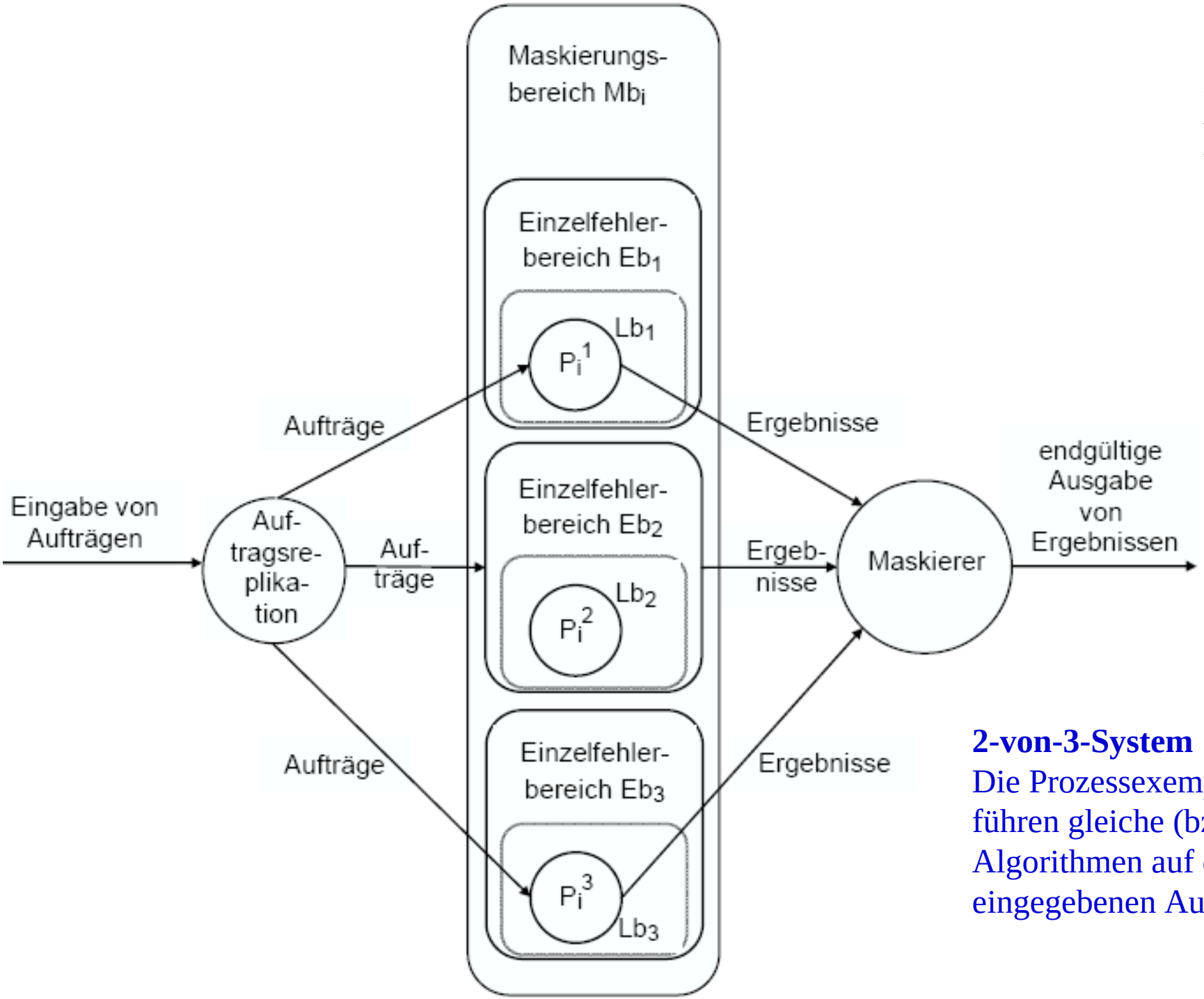
Z8: Fehlerbehandlung – Fehlermaskierung

Vorteile

- Fehlermaskierung (die zugehörigen *Relativtests* eingeschlossen) reicht als **einziges Fehlertoleranz-Verfahren** aus. *Rückwärtsbehebung* würde dagegen bei *permanenten Fehlern* eine zusätzliche *Rekonfigurierung* erfordern.
- *Maskierer* lassen sich vergleichsweise **einfach** implementieren.
- Da *Wiederholungsbetrieb* entfällt, erfolgt die Fehlerbehandlung **schneller** als bei der *Rückwärtsbehebung*.
- Fehlerhafte Subsystemexemplare dürfen **beliebiges fehlerhaftes Verhalten** aufweisen, da nicht Absoluttests die Subsystemexemplare selbst, sondern Relativtests die von ihnen produzierten Ergebnisse prüfen (die *Fehlerbereichs-Annahme* muss aber stets die Anzahl der fehlerhaften Subsystemexemplare begrenzen).
- Fehlermaskierung kann **transparent** implementiert werden.

Nachteile

- Hoher Aufwand, wenn strukturelle Redundanz einzusetzen ist.
- Zeit-unkritische Anwendungen legen deshalb nahe, eher *Rückwärts*- oder *Vorwärtsbehebung* zu bevorzugen.



Beispiel aus K. Echtle: Fehlertoleranz-verfahren

2-von-3-System

Die Prozessexemplare Pi führen gleiche (bzw. diversitäre) Algorithmen auf den gleichen eingegebenen Auftragsdaten aus

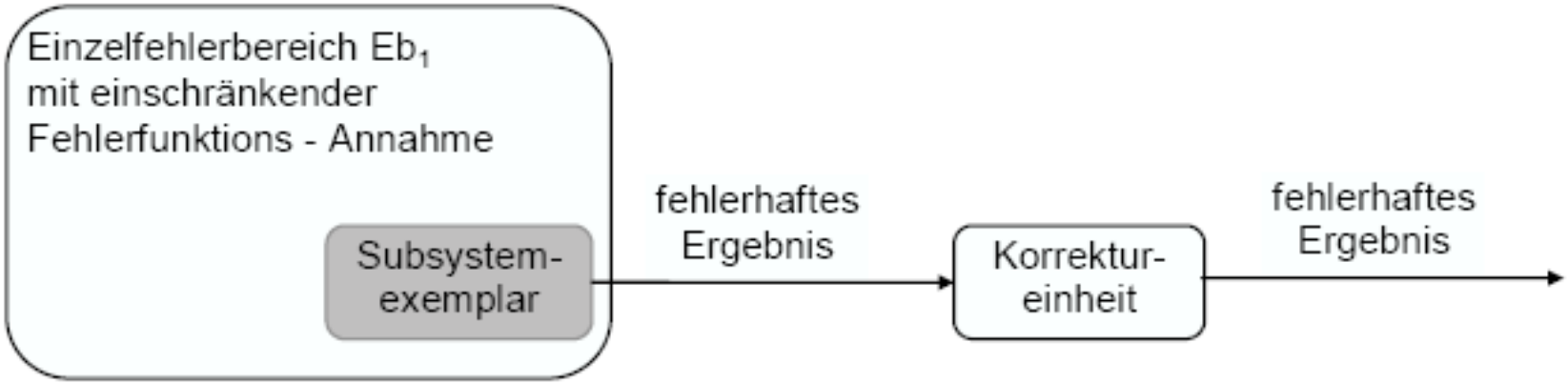
Z8: Fehlerbehandlung – Fehlermaskierung

Maskierungentscheidungen

- Deterministische Prozesse
  - *Mehrheitsentscheidung*
    - » Übliche Lösung
  - *Paarentscheidung*
    - » wenn zufällig gleiche Fehler ausgeschlossen werden können
  - *Meiststimmenentscheidung*
    - » *relative Mehrheit, reduzierte Anforderungen*
  - *Einstimmigkeitsentscheidung*
    - » *keine Fehlerbehandlung, nur Erkennung, oft eingesetzt um Notabschaltung o.ä. zu triggern*
- Nichtdeterministische Prozesse
  Menge der laut Spezifikation korrekten Ergebnisse bildet Intervall
  - *Medianentscheidung*
  - *Abstandsentscheidung*
    - » *Kugelentscheidung*
    - » *Intervallentscheidung*

Z8: Fehlerbehandlung – Fehlerkorrektur

- Fehlerhafte Ergebnisse werden auf fehlerfreie abgebildet
  - Die Fehlerfolgen werden durch die ***Korrektureinheit*** durch kompensiert
  - Aufgetretene Fehler verlassen ihren ***Fehlerbereich*** nicht, so dass das Gesamtsystem ungestört weiterarbeiten kann.
- Basis: Informationsredundanz schon in einem einzigen Ergebnis ermöglicht Fehlerkompensation
- Fehlerbehandlungsbereich
  Die Menge der Komponenten, deren fehlerhafte Ergebnisse korrigierbar sind, heißt ***Korrekturbereich***
  - die Korrektureinheit ist nicht im Korrekturbereich enthalten



Z8: Fehlerbehandlung – Fehlerkorrektur

- Verfahren
  - Fehlerkorrigierende Codes
    - » Hamming-Codes
    - » Kreuzparität
  - Redundanz in Datenstrukturen (z.B. Mehrfach-Verzeigerung)

- Anwendungen
  - Datenübertragung
    - » insbesondere, wenn zu wenig Zeit zur Nachrichtenwiederholung zur Verfügung steht
  - Datenspeicherung
    - » ECC-Halbleiterspeicher
    - » ECC-Datenspeicherung auf Platte
    - » Dateistrukturen auf Platte, Konsistenztest und Wiederherstellung

Z9: Qualitätssicherung und Zuverlässigkeitsanalyse

Verfahren – allgemein

- Die Verfahren sind nicht-formal. Sie richten sich an das menschliche Verständnis. Erfahrene Experten sollen in systematischer Weise „***Brainstorming***“ betreiben.
  - Expertenkreis **sucht** in einem System Risiken, kritische Situationen, Fälle, mögliche Ereignisse und Folgen: **Suchverfahren**
- Ein Verfahren unterstützt dies mit
  - Richtlinien zur Durchführung der Analyse (Arbeitsprozesse, Schritte und Abläufe)
  - Richtlinien zur Informationsdarstellung (Diagramme und Tabellen)
- Weitere Unterstützung kommt von
  - Anwendungsfeld- und Technologie-spezifischen Listen:
    - » Bekannte Gefahren und Fehlermöglichkeiten, bekannte Zusammenhänge
  - Bauteil-Datenbanken
    - » Zuverlässigkeitskenngrößen von Komponenten
    - » mögliche Ausfälle und Fehler
  - Im Bereich der Zertifizierung: Konkrete Vorschriften, z.B. Medizingerätegesetz

Z9: Qualitätssicherung und Zuverlässigkeitsanalyse

Einzelne Verfahren

- **FTA**: Fault Tree Analysis / Fehlerbaumanalyse
  - Sicherheitstechnik (Funktionssicherheit)
- **ETA**: "Event tree analysis / Ereignisbaumanalyse
  - Sicherheitstechnik (Funktionssicherheit)
- **FMEA**: Failure Mode and Effect Analysis
  - Maschinenbau
  - Automobilproduktion
- **PAAG** (engl. **HAZOP**): Prognose, Auffinden der Ursache, Abschätzen der Auswirkungen, Gegenmaßnahmen (engl. Hazard and Operability)
  - Chemische Anlagen
  - Chemietechnik

Verfahren – allgemein

- ***Vorwärts-Suche***
  - Start: Initiales Ereignis
  - davon ausgehend: Folgen ermitteln
  - so genannte „Induktive Suche“

- ***Rückwärts-Suche***
  - Start: Gefährlicher Zustand
  - davon ausgehend: Mögliche Ursachen ermitteln
  - so genannte „Deduktive Suche“

Ein Lüfter fällt aus
→ **Der Prozessor überhitzt**
→ **Der Steuerrechner fällt aus**
→ **Die Temperaturregelung versagt**
→ **Der Kessel überhitzt**
→ **Überdruck und Explosion**

Das Tragseil reißt
← **Die Schmierung fällt aus**
← **Das Ansteuersignal fehlt**
← **Die Zeitauftragssoftware startet den Steuerprozess nicht**
← **Programmierfehler: Jahrtausendwechsel nicht bedacht**

Verfahren – allgemein

- ***Bottom-Up-Suche***
  - Vom Bauteil über Aggregat und Subsytem zum Gesamtsystem
  - Stellventil B1 klemmt
    - » u.U. kann die Brennstoffzufuhr nicht unterbrochen werden
    - » die Heizung kann nicht abgestellt werden
    - » der Kessel A überhitzt
    - » der Kessel A explodiert und zerstört die Gaspipeline P1
    - » Kohlenmonoxid tritt aus
- ***Top-Down-Suche***
  - Vom Gesamtsystem (der technischen Anlage) über Subsysteme und Aggregate zu den Bauteilen
    - » die Gaspipeline P1 wird leck
    - » Kessel A könnte explodieren und P1 leck schlagen
    - » Überdruck baut sich in Kessel A auf
    - » …

Z9: Verfahren – FTA

Fehlerbaumanalyse

- Top-Down-Rückwärtssuche
- Von der Gefahr zu den möglichen Ursachen
- Ursachen-Kombinationen werden erfasst
  - Boolesche Verknüpfung der Ursachen
- Je Ursache
  - Erfassung von deren Ursachen
  - als Boolesche Verknüpfung
  - weiter mit deren Ursachen etc.
- Stop bei elementaren Ereignissen
- Wahrscheinlichkeitsbewertung der elementaren Ereignisse
- Berechnung der Gefahrenwahrscheinlichkeit über die Booleschen Verknüpfugen

Literatur

W. Vesely, F. Goldberg, N. Roberts, D. Haasl: Fault Tree Handbook, Systems and reliability research office of nuclear regulatory research, U.S. Nuclear Regulatory Commission, Washington, 1981.

U. Biegert: Ganzheitliche modellbasierte Sicherheitsanalyse von Prozessautomatisierungssystemen, Dissertation, Fakultät Informatik, Elektrotechnik und Informationstechnik, Universität Stuttgart, 2002.

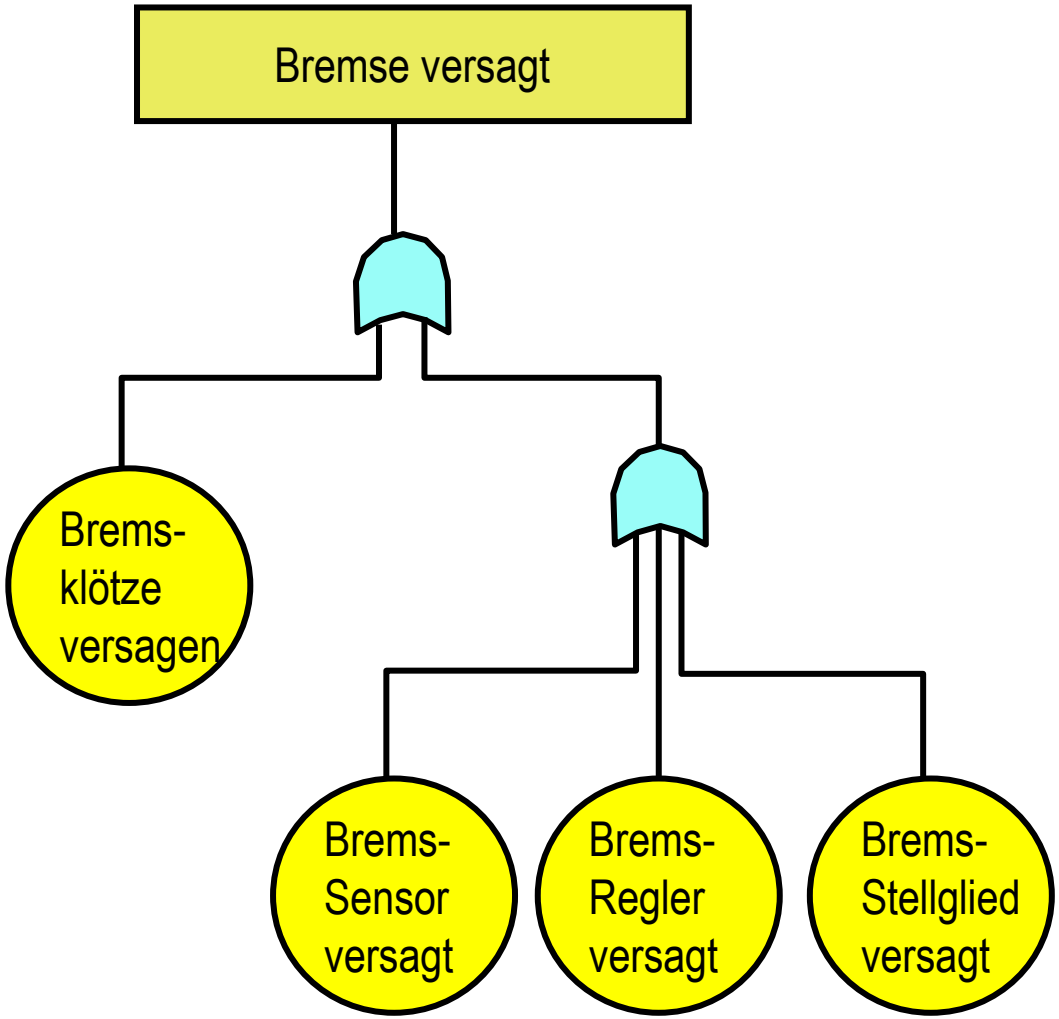
Standard

DIN 25 424

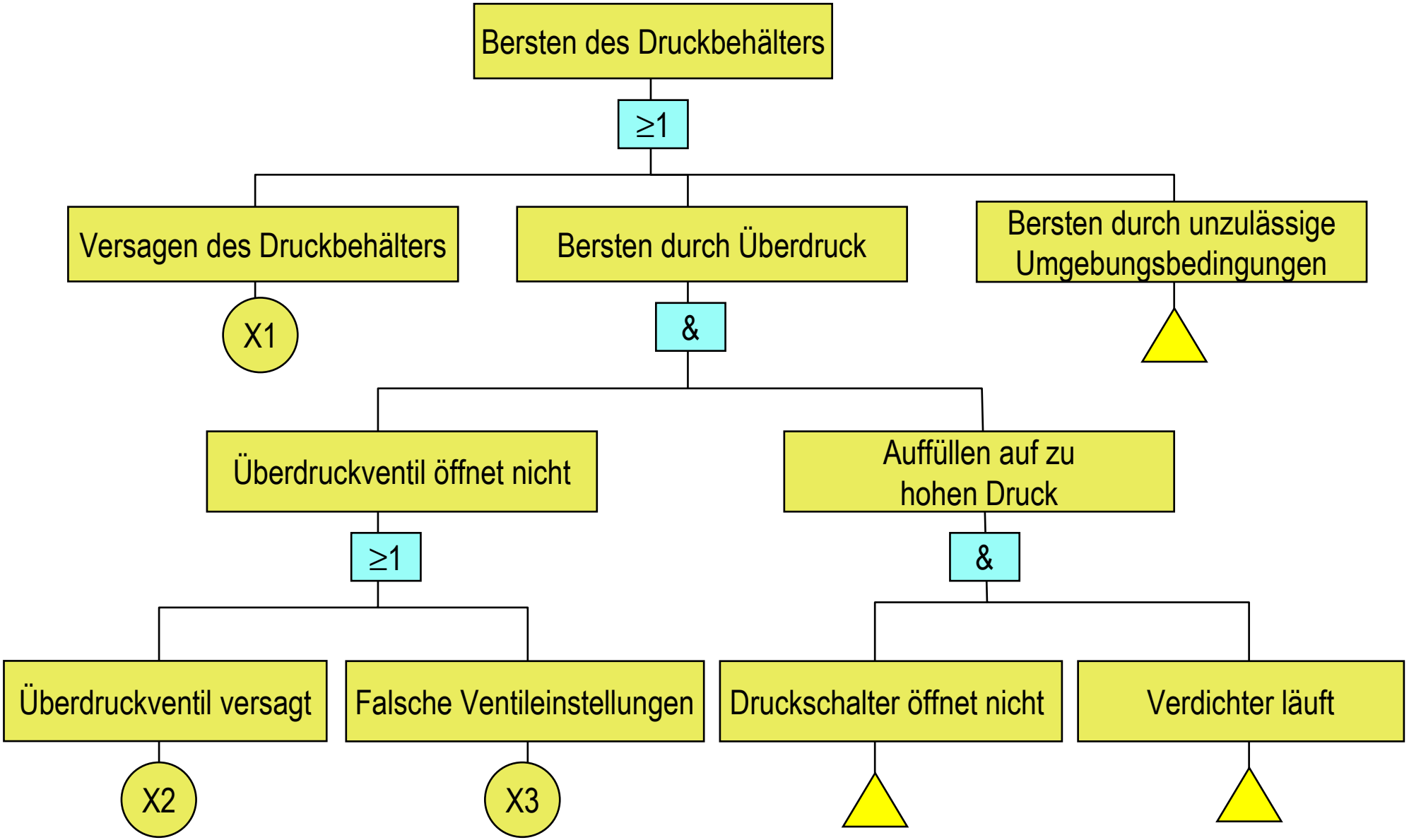
Z9: Verfahren – FTA

Fehlerbaumanalyse

- Knoten: Boolesche Verknüpfungen
  - UND
    - » die verknüpften Ereignisse müssen zusammen auftreten, um Ursache zu sein
  - ODER
    - » es genügt, wenn eines der verknüpften Ereignisse auftritt, um die Gefahr zu verursachen
- Knoten: Ereignisse
  - Elementares Ereignis
  - Hervorgerufenes Ereignis
  - Ungeklärtes Ereignis



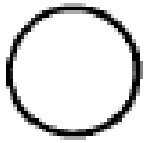
Z9: Verfahren – FTA



nach U. Biegert: Ganzheitliche modellbasierte Sicherheitsanalyse

Z9: Verfahren – FTA

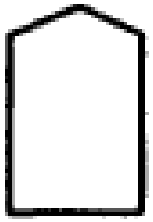
Primäre Ereignisse (werden nicht weiter aufgelöst)



BASIC EVENT – A basic initiating fault requiring no further development

CONDITIONING EVENT – Specific conditions or restrictions that apply to any logic gate (used primarily with PRIORITY AND and INHIBIT gates)

UNDEVELOPED EVENT – An event which is not further developed either because it is of insufficient consequence or because information is unavailable



EXTERNAL EVENT – An event which is normally expected to occur

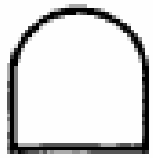
Zwischenereignisse (werden weiter aufgelöst)

INTERMEDIATE EVENT – A fault event that occurs because of one or more antecedent causes acting through logic gates

aus US – Fault Tree Handbook

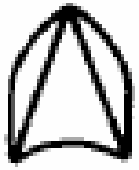
Z9: Verfahren – FTA

Boolesche Verknüpfungen

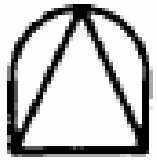


AND – Output fault occurs if all of the input faults occur

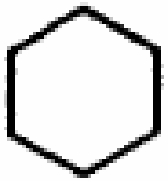
OR – Output fault occurs if at least one of the input faults occurs



EXCLUSIVE OR – Output fault occurs if exactly one of the input faults occurs



PRIORITY AND – Output fault occurs if all of the input faults occur in a specific sequence (the sequence is represented by a CONDITIONING EVENT drawn to the right of the gate)

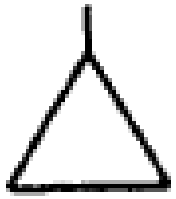


INHIBIT – Output fault occurs if the (single) input fault occurs in the presence of an enabling condition (the enabling condition is represented by a CONDITIONING EVENT drawn to the right of the gate)

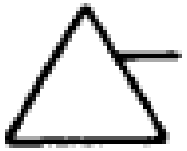
aus US – Fault Tree Handbook

Z9: Verfahren – FTA

Konnektoren



TRANSFER IN — Indicates that the tree is developed further at the occurrence of the corresponding TRANSFER OUT (e.g., on another page)



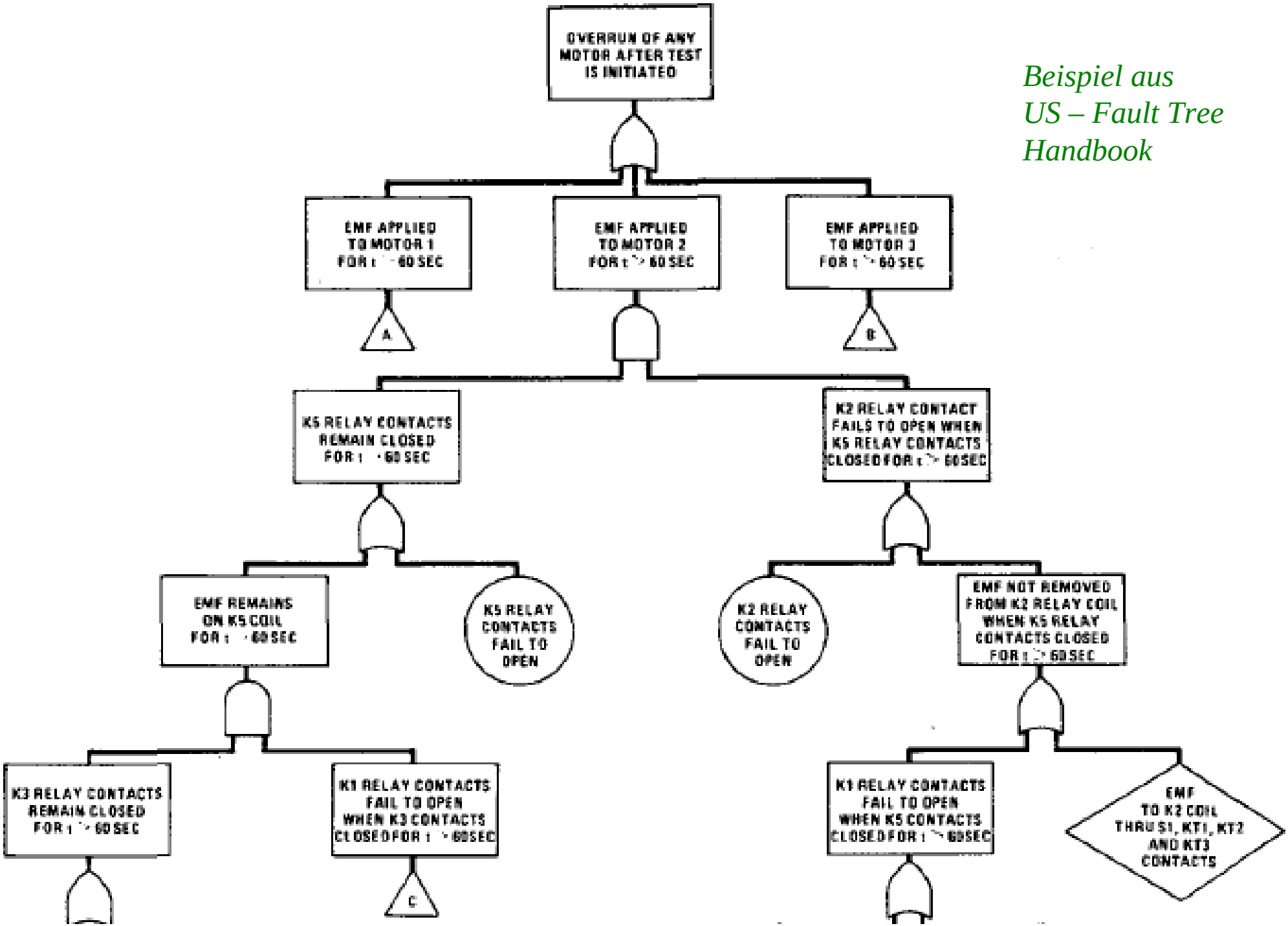
TRANSFER OUT — Indicates that this portion of the tree must be attached at the corresponding TRANSFER IN

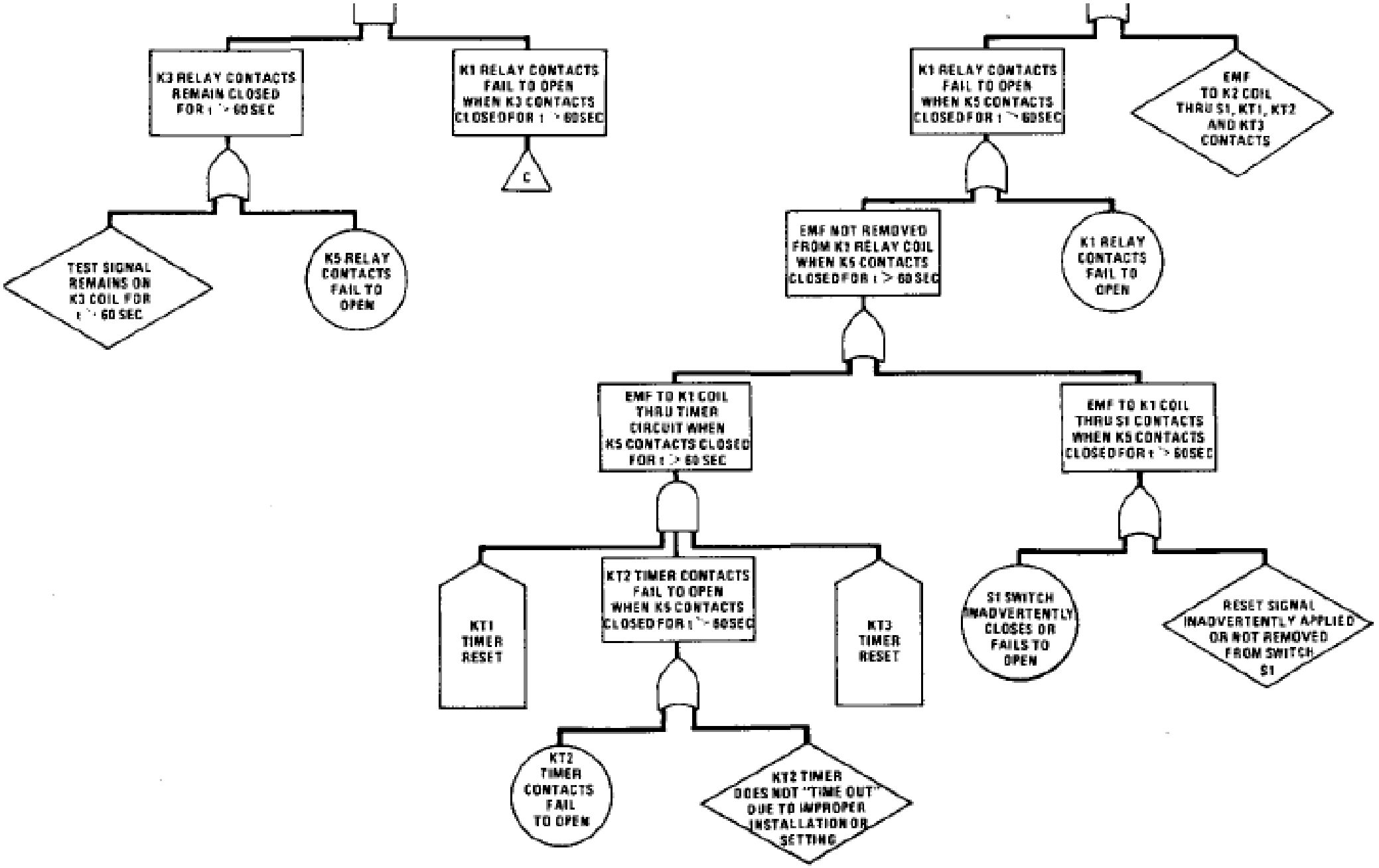
Wahrscheinlichkeitsbewertung

- minimale Schnitte von Primärereignissen als Auslöser für ein Fehlerereignis
- minimale Pfade zu Fehlerereignis
- Unabhängigkeitsannahme
- Berechnung der Ereigniswahrscheinlichkeit entsprechend minimalen Schnitten und minimalen Pfaden
- Genauere Berechnungen über Verteilungsfunktionen und Abhängigkeiten

aus US – Fault Tree Handbook

Beispiel aus US – Fault Tree Handbook





Beispiel aus
US – Fault Tree Handbook

Z9: Verfahren – ETA

Ereignisbaumanalyse

(auch: Ereignisablaufanalyse)

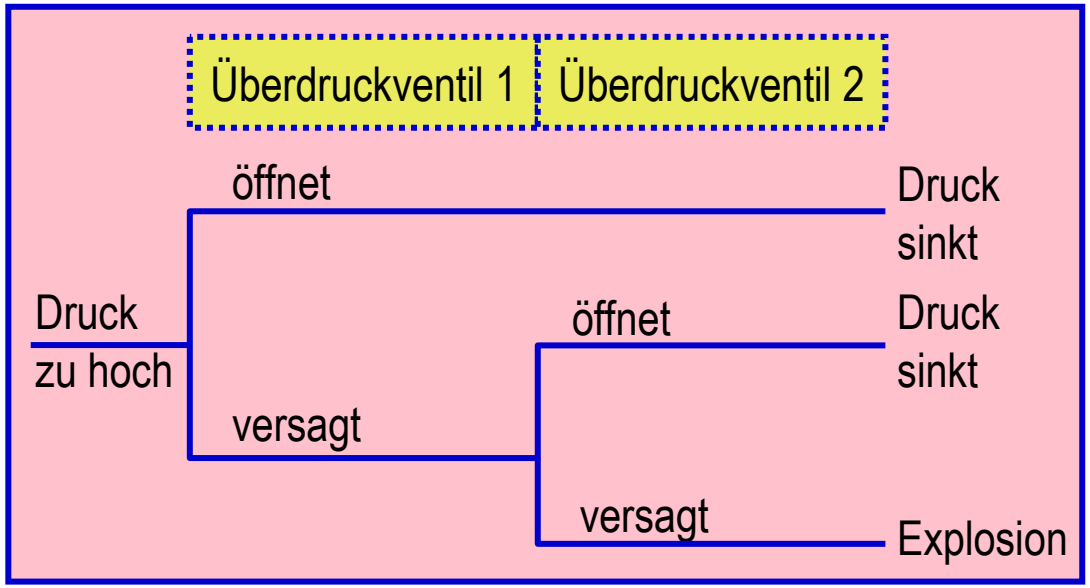
- Bottom-Up-Vorwärtsanalyse
- Von der Ursache zu den möglichen Gefahren
- Darstellung als binärer Baum
  - Wurzel links – Söhne nach rechts
  - oberer Sohn: normales Betriebs-Verhalten
  - unterer Sohn: Ausfall, Fehler

Literatur

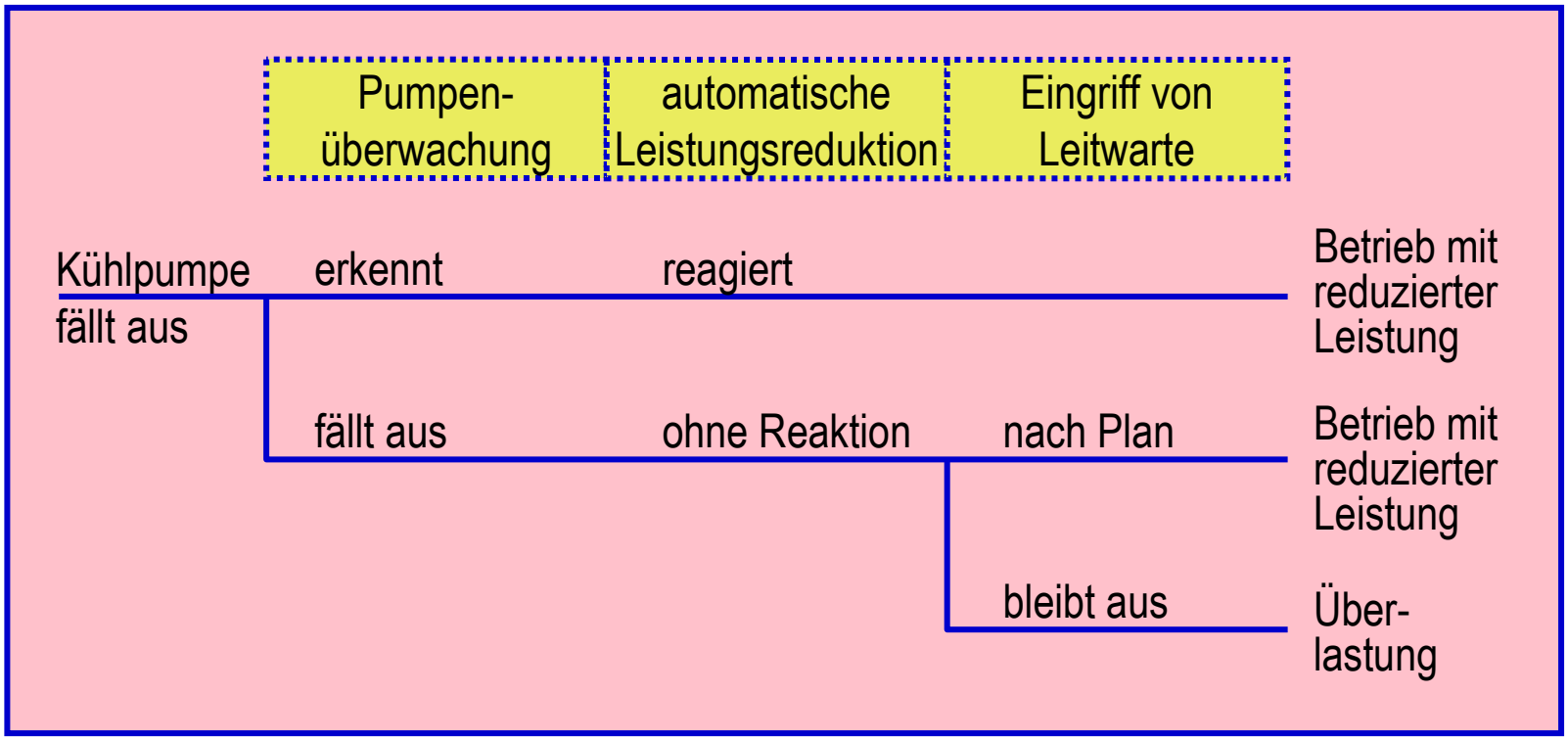
Leveson, Nancy G.: SAFEWARE - System Safety and Computers. Addison-Wesley Publishing Company 1995

Standard

DIN 25 419



Z9: Verfahren – ETA



Z9: Verfahren – FMEA

Failure Mode and Effect Analysis

- Hauptanwendungsfeld: Maschinenbau, Automobilbau
- Qualität durch
  - Entwurf / Konstruktion
  - Fertigung
- 3 Phasen
  - Funktionsbeschreibung
    - » Funktionelle Anforderungen
    - » Use Cases
  - Systembeschreibung
    - » Architektur
      - Funktionsweise
      - Blockdiagramm und Assoziationen
  - Die eigentliche Analyse
    - » Tabelle erzeugen – Funktionen
    - » Tabelle füllen: 3 Schritte

Literatur

B. Beltz, J. Edenhofer, J. Eilers u.a.: Sicherung der Qualität vor Serieneinsatz: System-FMEA, Verband der Automobilindustrie, Band 4, Teil 2, Frankfurt, 1996.

Standards

DIN 25 448

/MIL-STD 1629A, 1980/ Procedures for Performing a Failure Mode, Effects and Critically Analysis

Z9: Verfahren – FMEA

Schritt 1 – Fehlervorkommen, Effekte und Schweregrad

- Bestimme ausgehend von den funktionellen Anforderungen alle Fehlervorkommen und ihre Effekte (z.B. Kurzschluss, Korrosion, Bruch), wie sie vom Nutzer wahrgenommen werden
- Bewerte den Schweregrad mit einer Zahl aus [1 (keine Gefahr),…, 10 (überaus kritisch)]

Schritt 2 – Ursachen und Auftretenshäufigkeit

- Bestimme für jedes Fehlervorkommen die möglichen Ursachen (z.B. Überspannung, Übertemperatur, Feuchtigkeitseinbruch, SW-Entwurfsfehler)
- Bewerte die Ursachen-Häufigkeit mit einer Zahl aus [1 (selten),…, 10 (überaus häufig)]

Z9: Verfahren – FMEA

Schritt 3 – Erkennung

- Bestimme für jede Kombination aus Schritt 1 und 2, wie man den Fall erkennen (und beheben / verhindern) kann, z.B. durch Qualitätstests, Inspektionen, Laufzeittests- und Laufzeitfehlerbehandlungen
- Bewerte die Erkennungs/Behandelbarkeit mit einer Zahl aus [1 (leicht),…, 10 (sehr schwierig)]

Schluss – Risikoprioritäten

- Bestimme für jeden Fall die Risikopriorität durch Multiplikation der drei Maßzahlen
- Risikopriorität
  = Schweregrad * Häufigkeit * Behandelbarkeit
- Alle Fälle über einem Schwellwert sind zu bearbeiten

Z9: Verfahren – FMEA

Example FMEA Worksheet

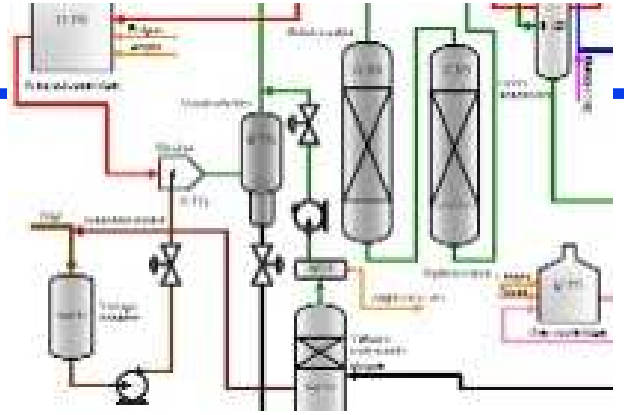
Function	Failure mode	Effects	S (severity rating)	Cause(s)	O (occurrence rating)	Current controls	D (detection rating)	CRIT (critical characteristic	RPN (risk priority number)	Recommended actions	Responsibility and target completion date	Action taken
Fill tub	High level sensor never trips	Liquid spills on customer floor	8	level sensor failed level sensor disconnected	2	Fill timeout based on time to fill to low level sensor	5	N	80	Perform cost analysis of adding additional sensor halfway between low and high level sensors	Jane Doe 10-Oct-2010	

Schweregrad | Häufigkeit | Behandelbarkeit | Risikopriorität

aus Wikipedia (engl.): „FMEA“

FMEA						
Name	Fehler-modus	Ursache	Auswirkung	Wahrschein-lichkeit	Relevanz	Kontroll- u. Verbesserungs-maßnahmen
Schalter ...	fails open ...	„verklebt“ ...	Überlastung Motor ...	$1*10^{-6}$ ...	kritisch ...	regelmäßige Wartung ...

Z9: Verfahren – PAAG/HAZOP



- Verfahren zu chemietechnischen Anlagen
- Basis: Prozess-Flussdiagramm
  - es zeigt die verschiedenen Verarbeitungselemente (Kessel´, Rührer, Reaktor, ..), Röhren, Ventile, Pumpen, etc.
  - es zeigt die Materialflüsse
  - es zeigt u.U. typische Werte und Grenzwerte
- Jedes Element hat nach Plan einen Zweck
  - eine Röhre soll z.B. Schwefelsäure mit 23 kg/sec und bei 20 Grad transportieren
  - ein Wärmetauscher soll z.B. Energie entziehen und die Temperatur reduzieren
- Das Analyse-Expertenteam diskutiert jedes Element, seine Zwecke und die Fehler- und Maßnahmenmöglichkeiten
- Die Analyse wird typischerweise in 3-stündigen Sitzungen durchgeführt
  - mittlere Chemieanlage
    - » 1200 Elemente zu diskutieren
    - » ca. 40 Sitzungen benötigt

Literatur

K. Bareis, H.Hoffmann, L. Rossinelli: PAAG – Verfahren (HAZOP), Risikobegrenzung in der Chemie, Internationale Sektion der IVSS für die Verhütung von Arbeitsunfällen und Berufskrankheiten in der chemischen Industrie, Heidelberg, 1990.

Standards

IEC 61882:2001

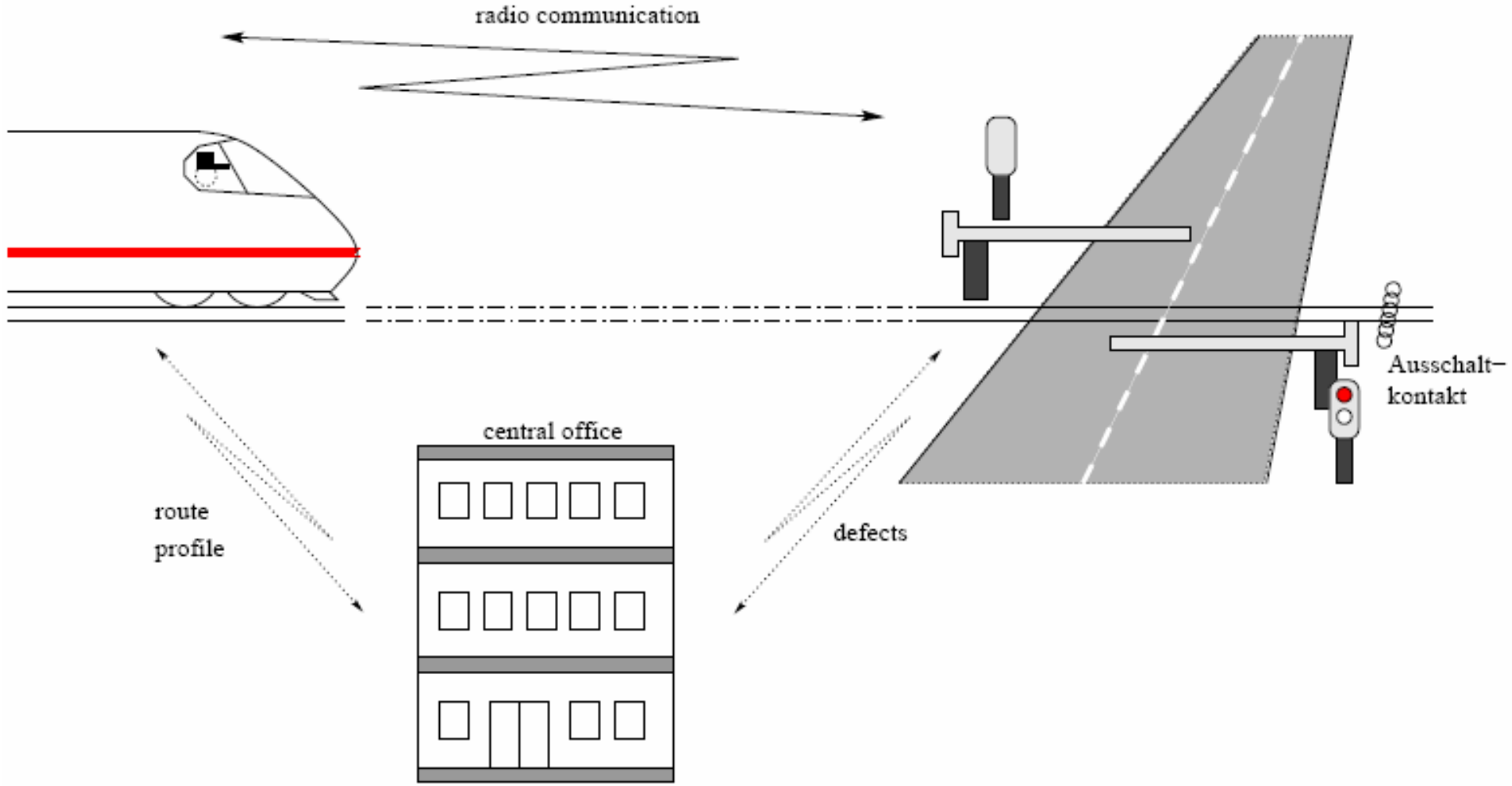
British Standard: Hazard and operability studies (HAZOP studies)- Application Guide

Z9: Verfahren – Beispiel Bahnübergangsteuerung

- FMEA und FTA basieren auf nicht-formalen Systembeschreibungen
  - Mehrdeutigkeiten
  - Übersehen von möglichen Fehlerursachen
- Ziel:
  Formale Fehleranalyse komplexer Systeme entsprechend FMEA und FTA
  - Statecharts
  - algebraische Spezifikationen
  - darauf aufbauende formale Analysen
- Wunsch:
  Möglichkeit, die Korrektheit und Vollständigkeit der Fehleranalyse nachzuweisen

Literatur
W. Reif, G. Schellhorn, A. Thums: Formale Sicherheitsanalyse einer funkbasierten Bahnübergangsteuerung, in Schnieder (ed.): Forms 2000 – Formale Techniken für die Eisenbahnsicherung, Fortschritt-Berichte VDI, Reihe 12, 2000.

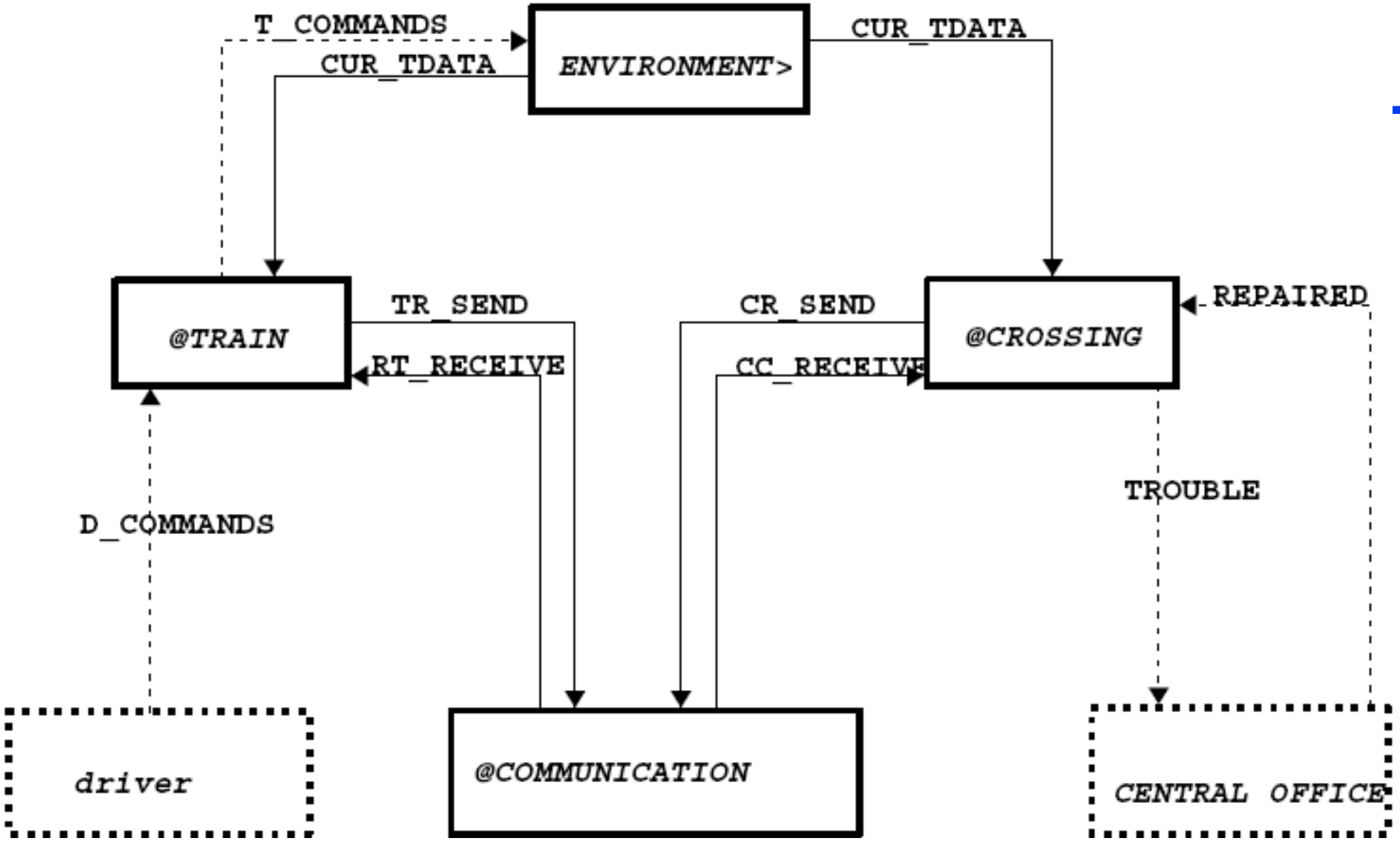
Z9: Verfahren – Beispiel Bahnübergangsteuerung



- *Beispiel: Funkbasierte Bahnübergangsteuerung*

Z9: Verfahren – Beispiel Bahnübergangsteuerung

- DB-Entwicklung: Funkbasierte Bahnübergangsteuerung
  - Statt: Sensoren und Signalen an der Stecke
  - Nun: Funkkommunikation, sowie
    Berechnungen in der Zug- und Bahnübergangsteuerung
  - Z.B.: Schließzeitpunkt der Schranke wird in Abhängigkeit von Zuggeschwindigkeit und Position berechnet, um Wartezeiten zu reduzieren.
- Zug nähert sich, stößt Schließvorgang an.
  - Zug kennt die Position des Bahnübergangs, den sogenannten Gefahrenpunkt, und berechnet aus der Differenz zu seiner eigenen Position und seiner Geschwindigkeit den Einschaltpunkt des Bahnübergangs.
  - Erhält der Bahnübergang ein Schließsignal, schaltet er die Lichtzeichenanlage an, d. h. zuerst das Gelblicht und etwas später das rote Licht. Dann werden die Schranken geschlossen.
  - Wenn Schranken vollständig geschlossen sind, ist der Bahnübergang für eine bestimmte Zeitspanne gesichert. Zug erhält Freigabe und kann Übergang passieren.
  - Bei Defekten bleibt Übergang ungesichert. Zug erfährt dies durch Statusanfrage und leitet das Anhalten ein.

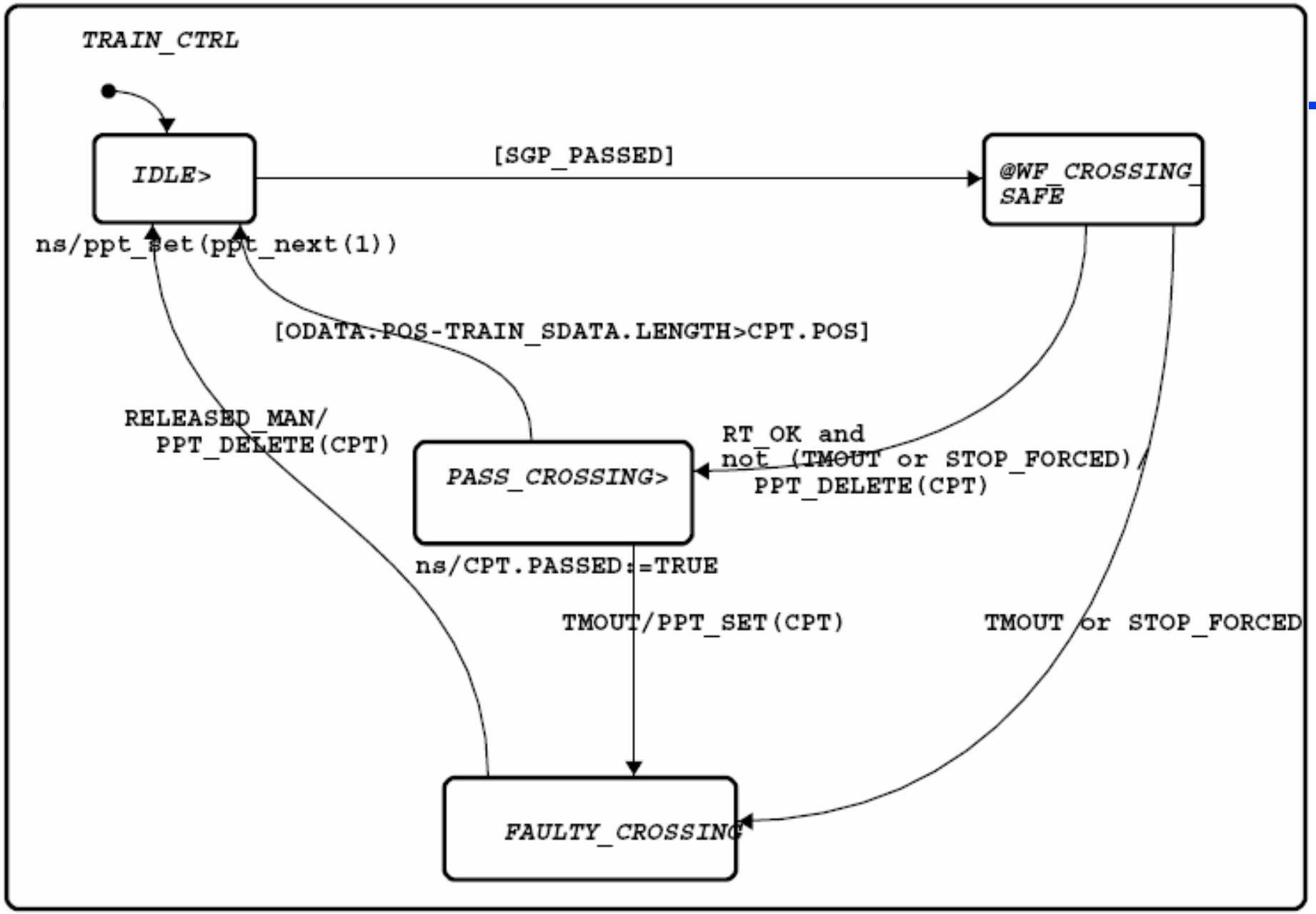


- Activity Chart – Diagramm (mit Statemate erstellt)
  - Struktur des Systems: Systemkomponenten
  - Interaktionen zwischen Komponenten

Z9: Verfahren – Beispiel Bahnübergangsteuerung

Komponenten

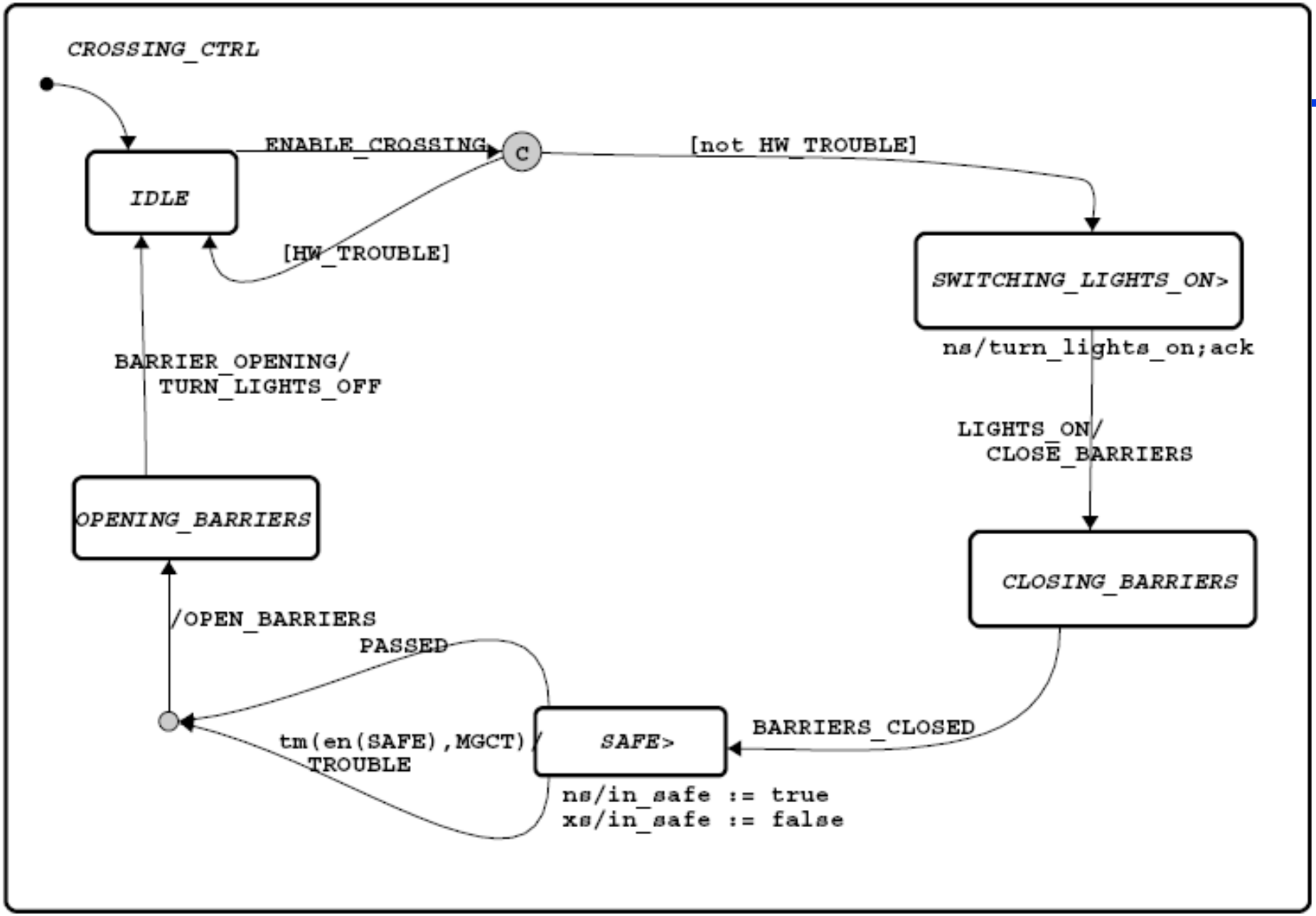
- ***ENVIRONMENT:*** Reales Verhalten des Zuges, wie es von der Umwelt wahrgenommen wird.
  - Ermitteln der Geschwindigkeit und der Position
  - Weitergabe an Zug und Bahnübergang
- ***TRAIN***: Verhalten des Zuges
  - Beschleunigungs- und Bremsdaten
  - Überwachung: Geschwindigkeit und Annäherung an Bahnübergang
  - Bei Annäherung an Bahnübergang sendet Zug Einschaltsignal über Funkkomponente an Bahnübergang
  - Statusanfrage an Übergang, u.U. Nothalt
- ***COMMUNICATION***: Funkkomponente
  - Signalaustausch zwischen Zug und Übergang
- ***CROSSING:*** Übergang
  - Einschalten der Lichtzeichen
  - Schließen der Schranken
  - Gesicherter Zustand, wenn Schranken geschlossen
  - Statusmeldung auf Anfrage an Zug
- *DRIVER*: Lokführer
  - steuert Zuggeschwindigkeit
- *CENTRAL OFFICE*: Zentrale
  - Fehlermeldungen vom Bahnübergang



- Statechart: TRAIN CONTROL

Z9: Verfahren – Beispiel Bahnübergangsteuerung

- Wenn der Zug den Einschaltpunkt des Bahnübergangs passiert (SGP PASSED), sendet er ein Signal an den Bahnübergang und erwartet, dass der Bahnübergang geschlossen wird
- Wenn der Zug die Antwort “Schranke geschlossen” erhält (RT OK), ist der Bahnübergang freigegeben und der Zug kann passieren.
- Ansonsten bremst der Zug und der Bahnübergang wird manuell gesichert (RELEASED MAN).



- Statechart: CROSSING CONTROL

Z9: Verfahren – Beispiel Bahnübergangsteuerung

- Erhält der Bahnübergang das Einschaltsignal (ENABLE CROSSING), schaltet er die Lichtzeichenanlage ein (TURN LIGHTS ON) und schließt nach einigen Sekunden die Schranken (CLOSE BARRIERS).
- Sind die Schranken geschlossen, antwortet der Bahnübergang auf eine Statusabfrage des Zuges mit “gesichert”.
- Hat der Zug den Bahnübergang passiert (PASSED), öffnet der Bahnübergang die Schranken wieder, schaltet die Lichtzeichen aus und geht wieder in den initialen Zustand.
- Wenn der Zug nicht passiert, wartet der Bahnübergang einige Minuten, öffnet die Schranken, schaltet die Lichtzeichen aus und ist wieder ”ungesichert“.

Z9: Verfahren – Beispiel Bahnübergangsteuerung

FMEA

- Softwarekomponente *Velocity Curve*
  - Berechnung der Zug-Geschwindigkeitskurve (f: Ort → Maximalgeschwindigkeit)
- Fehler
  - zu hohe Maximalgeschwindigkeiten (falsche Verzögerungswerte oder Algorithmusfehler): Zug hält nicht mehr rechtzeitig. Zug fährt in ungesicherten Übergang ein, ruft Kollision hervor
- Single-Point-of-Failure
  - Fehler in der Softwarekomponente *Velocity Curve* kann alleine schon zur Kollision führen
- Maßnahmen
  - Plausibilitätstests
  - Software-Verifikation
  - besser auch zusätzlich:
    - » Diagnose per Absolut- und Relativtests und Kompensation mittels Diversität

FMEA Tabelle

(Sub-)System: _____		
name (identification)	function	failure mode
velocity_curve	Diese Funktion berechnet die Geschwindigkeitskurve. Diese dient dazu, jedem Streckenpunkt eine Maximalgeschwindigkeit zuzuordnen. Dazu betrachtet die Funktion das Streckenprofil, die gesetzten Gefahrenpunkte sowie die Verzögerungswerte des Zuges.	1. zu hoch 2. zu niedrig

failure mode	failure cause	local effect
1. zu hoch	• fehlerhafte Berechnungs-vorschrift • fehlerhafte Para-meter – Verzögerung – Streckenatlas – Gefahren-punkte	es wird (zu spät) ge-bremst
2. zu niedrig	siehe oben	es wird zu früh gebremst
3. willkürlich	siehe oben	

FMEA Tabelle

system effect	single-point of failure	failure class	control, recommendation
zu hohe Geschwindigkeit:			Plausibilitätschecks Verifikation der kritischen Berechnungsfunktion
• pot. Entgleisen	y	A	
• pot. Einfahrt in ungesicherten Bahnübergang	y	A	
zu niedrige Geschwindigkeit:			
andauerndes Beschleunigen/Bremsen:			

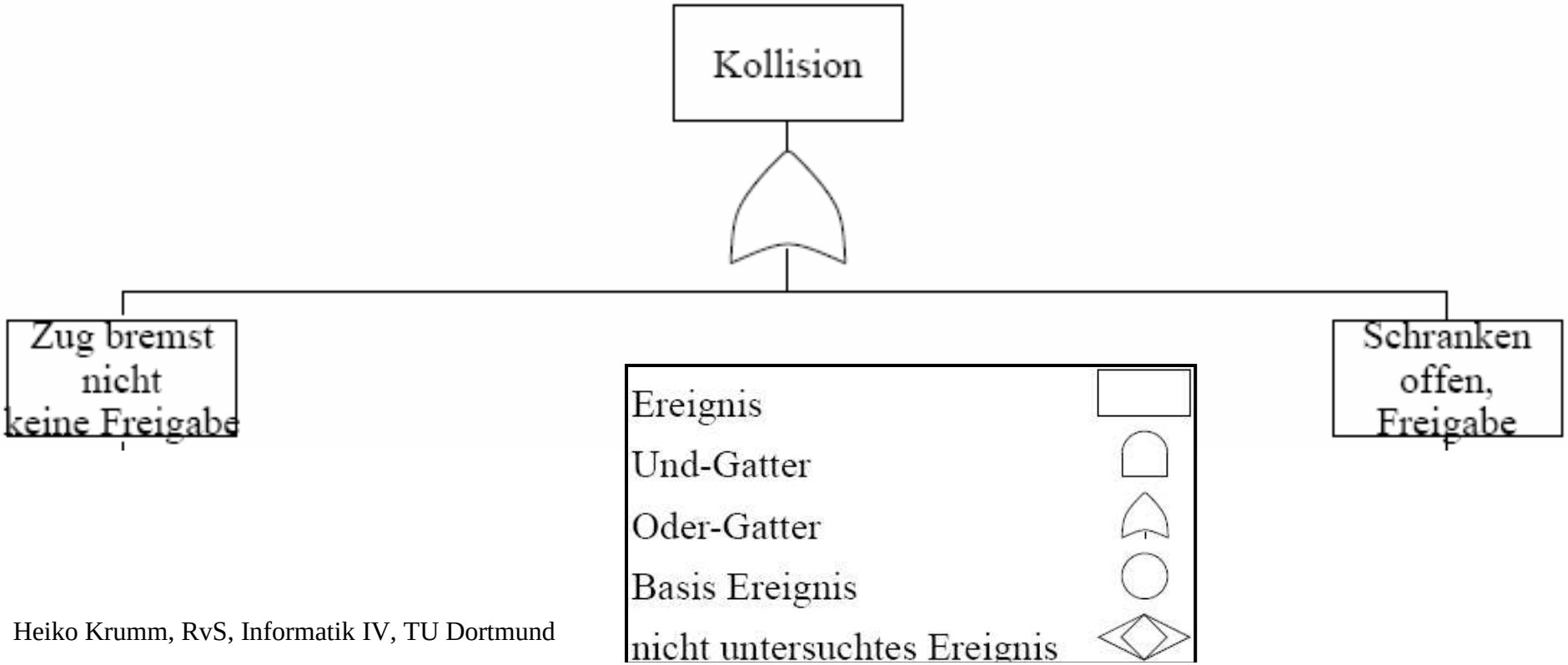
FMEA Tabelle

zu niedrige Geschwindigkeit:			FMEA Tabelle
• Blockierung Autos	y	C	
• Blockierung Zug	y	C	
• Einfahrt in ungesicherten Bahnübergang, wenn sich Schranken nach 240 Sek. wieder öffnen	y	A	
andauerndes Beschleunigen/Bremsen:			
• Verschleiß	y	B	
• Defekte	y	B	

Z9: Verfahren – Beispiel Bahnübergangsteuerung

FTA für *Kollision*

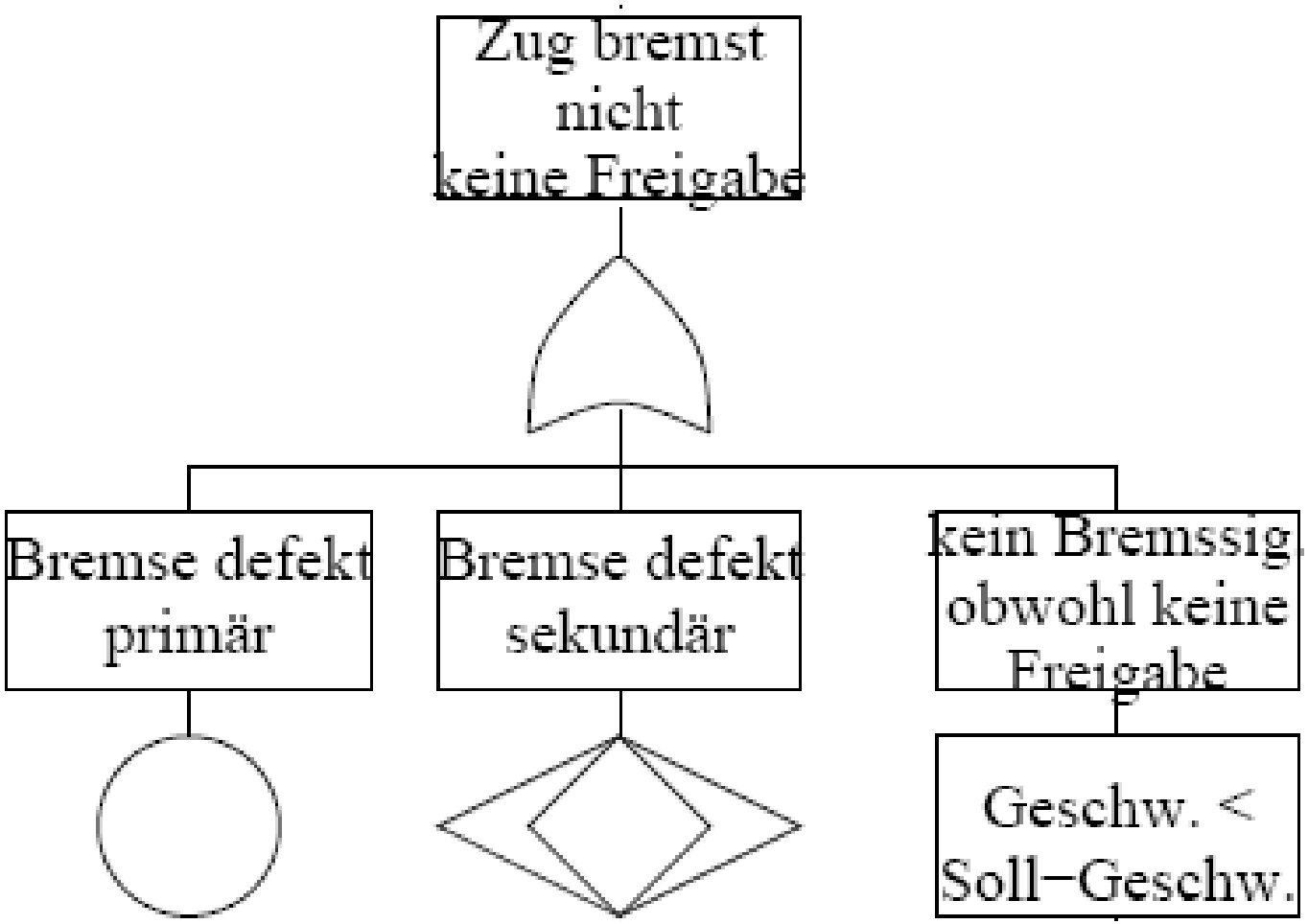
- Zug ist auf Bahnübergang und Schranken nicht geschlossen
- Ursachen
  - Zug bremst nicht, obwohl von Bahnübergang keine Freigabe erhalten
  - Bahnübergang sendet Freigabe, obwohl Schranken nicht geschlossen



Z9: Verfahren – Beispiel Bahnübergangsteuerung

Unterbaum *"Zug bremst nicht"*

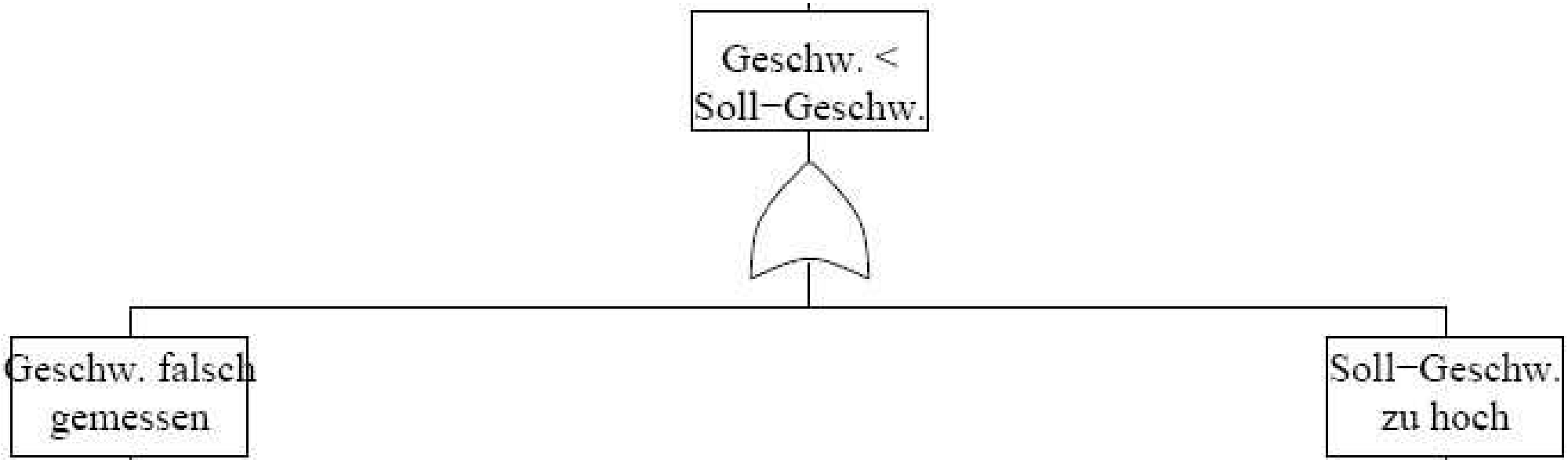
- Bremsendefekte
- Bremssignal fehlt



Z9: Verfahren – Beispiel Bahnübergangsteuerung

Unterbaum *"Geschwindigkeit < Soll"*

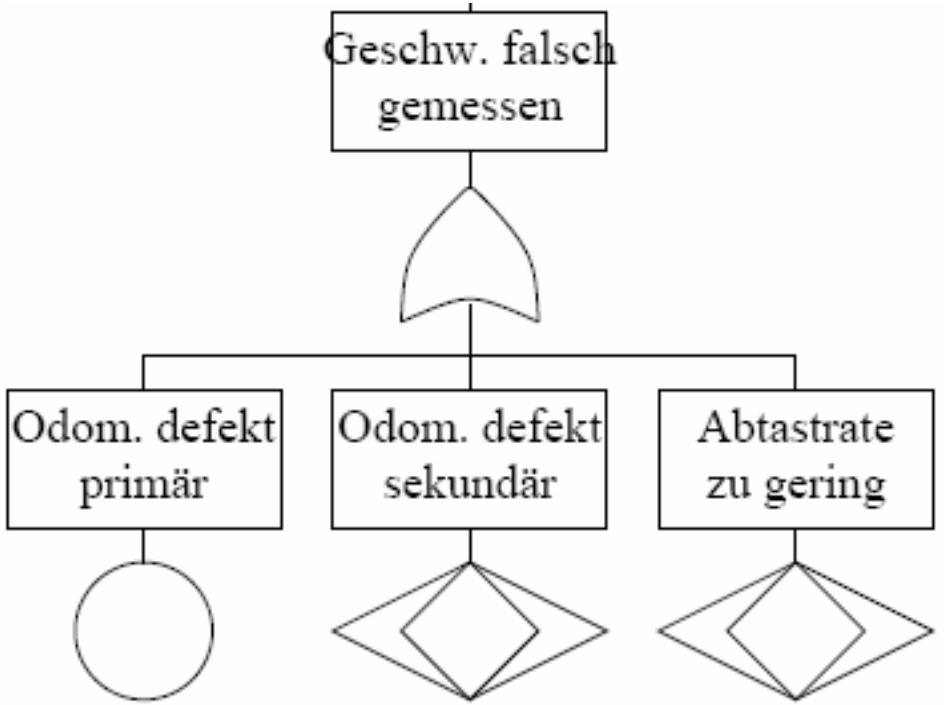
- Tachofehler
- Soll zu hoch



Z9: Verfahren – Beispiel Bahnübergangsteuerung

Unterbaum *"Tachofehler"*

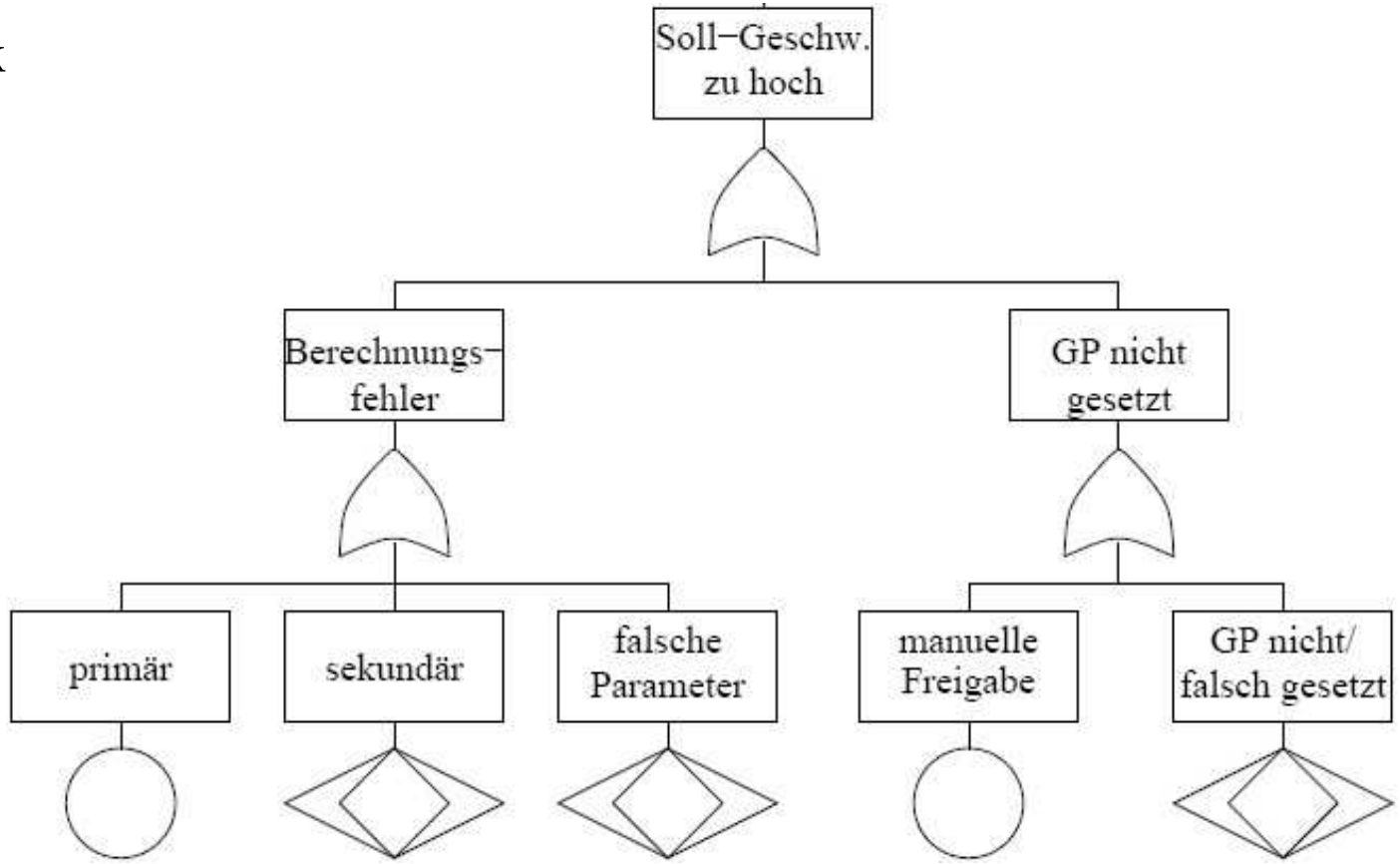
- Tachodefekt
- Abtastrate



Z9: Verfahren – Beispiel Bahnübergangsteuerung

Unterbaum *"Soll zu hoch"*

- Berechnungsfehler
- Gefahrenpunk

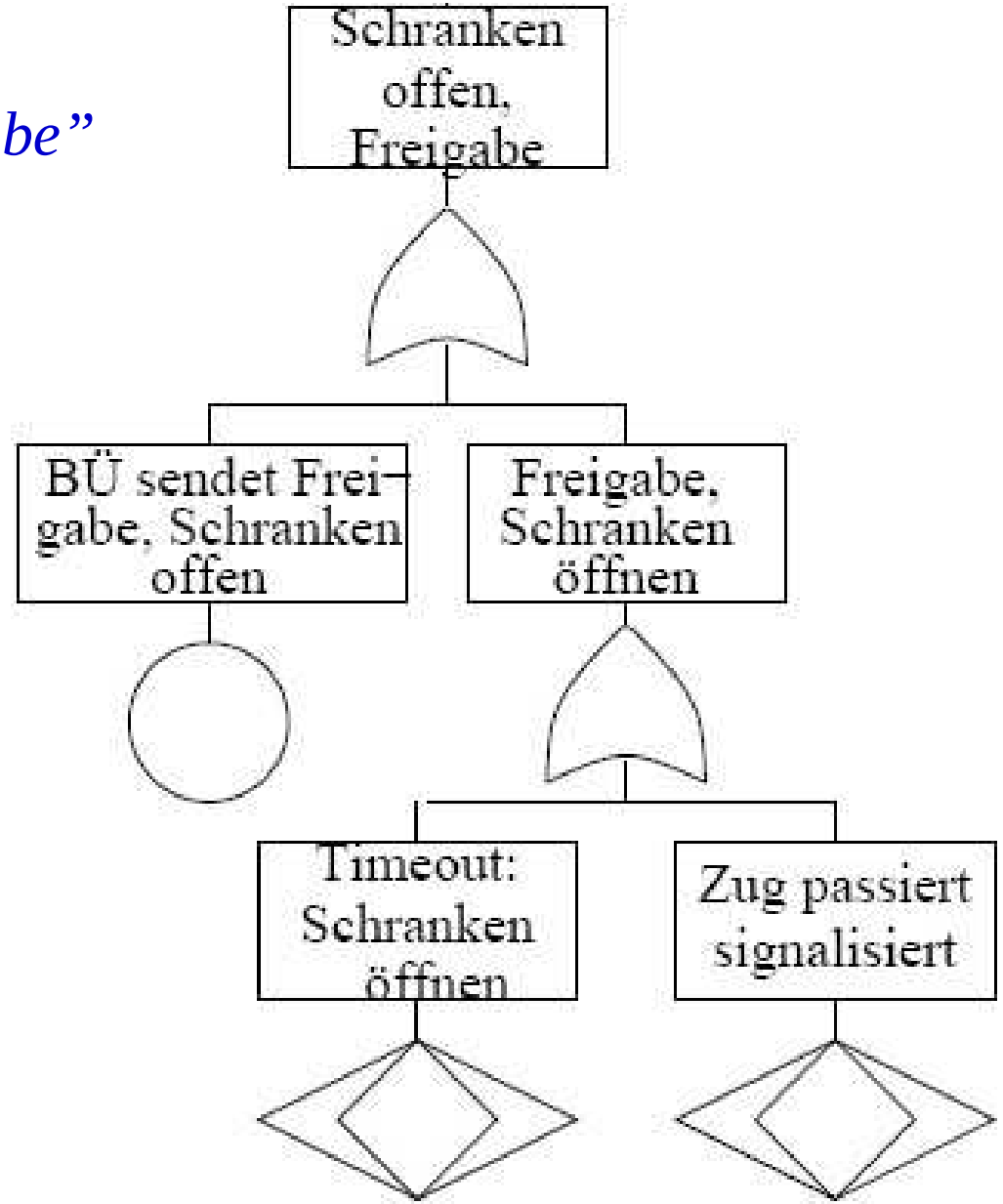


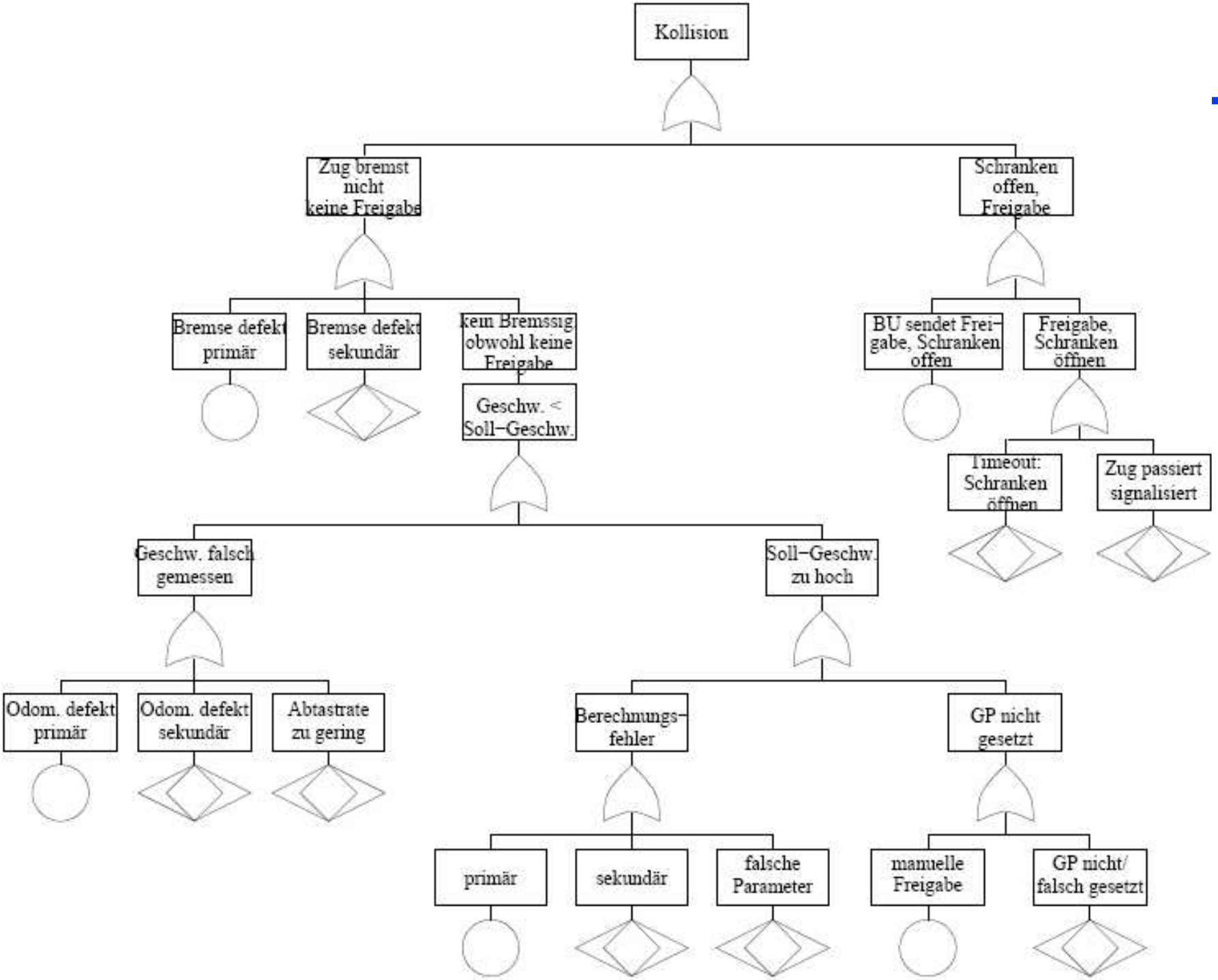
Z9: Verfahren – Beispiel Bahnübergangsteuerung

Unterbaum

“Schranken offen, aber Freigabe”

- Schranken schon offen
- Schranken werden geöffnet




Kollision
Zug bremst nicht keine Freigabe
Schranken offen, Freigabe
Bremse defekt primär
Bremse defekt sekundär
kein Bremssig obwohl keine Freigabe
BU sendet Freigabe, Schranken offen
Freigabe, Schranken öffnen
Geschw. < Soll−Geschw.
Timeout: Schranken öffnen
Zug passiert signalisiert
Geschw. falsch gemessen
Soll−Geschw. zu hoch
Odom. defekt primär
Odom. defekt sekundär
Abtastrate zu gering
Berechnungs-fehler
GP nicht gesetzt
primär
sekundär
falsche Parameter
manuelle Freigabe
GP nicht/ falsch gesetzt

Z9: Verfahren – Beispiel Bahnübergangsteuerung

Entdeckter Fehler

- Bahnübergang öffnet sich nach Timeout, auch wenn Zug nicht passierte, nimmt aber Freigabe nicht zurück
- Solcherweise verspätet an Schranke eintreffender Zug, löschte Gefahrenpunkt bei Empfang der Freigabe, bremst also nicht

Anmerkung 1:
Dies ist ein typisches Ergebnis einer funktionellen Verifikation, das auf der Basis eines adäquaten funktionellen Modells erzielt wird. Die eigentliche Sicherheitsanalyse hat die Aufgabe, das adäquate Modell zu erzeugen. Hier wird (außer den Modelldefini-tions- und Darstellungsmöglichkeiten) kein besonderer Beitrag der formalen Techniken erkennbar.

Anmerkung 2:
Die in dem Papier beschriebene Sicherheitsanalyse ist aus praktischer Sicht gerade in den wesentlichen Aspekten – nämlich der sorgfältigen Suche und dem Finden von Fehler- und Ausfallmöglichkeiten – sehr rudimentär und würde für eine Zertifizierung auf keinen Fall ausreichen.

Z9: Möglichkeiten und Grenzen formaler Safety-Analyse

Möglichkeiten

- Funktionalität – inklusive Fehlermöglichkeiten und Toleranzmechanismen – lässt sich als nichtdeterministisches Verhalten modellieren
- Formale Verifikation: Es kann gezeigt werden, dass Safety-, Liveness- und Realzeit-Eigenschaften unter Anwesenheit der im Modell erfassten Fehler vorhanden sind

Grenzen

- Modellbildung und Fehlermodellierung bleiben die dominierende Hauptaufgabe, welche nicht formalisiert werden kann
  - man könnte zwar formal Modelle erzeugen, welche über definierten Schnittstellen erschöpfend alle Verhaltensweisen zulassen
  - aber solche Modelle sind ohne praktischen Nutzen

Wege

- Erfahrungen mit Komponenten und Architekturmustern formalisieren
  - Bibliotheken mit Modellelementen und Mustern