# Modell-basierte Sicherheitsanalyse


**Dr. Frank Ortmeier**
Lehrstuhl Softwaretechnik und Programmiersprachen
Universität Augsburg

# Warum?

- Trends bei sicherheitskritischen Systemen
  - Steigende Komplexität
  - Zunehmende Kritikalität der Konsequenzen
  - Immer mehr Funktionalitäten werden von Software übernommen
- Konsequenzen
  - Sicherheitsanalyse wird zunehmend schwieriger
  - Anforderungen an die Qualität der Resultate steigt


Ariane 5


Tschernobyl

Boeing 737 - Chicago

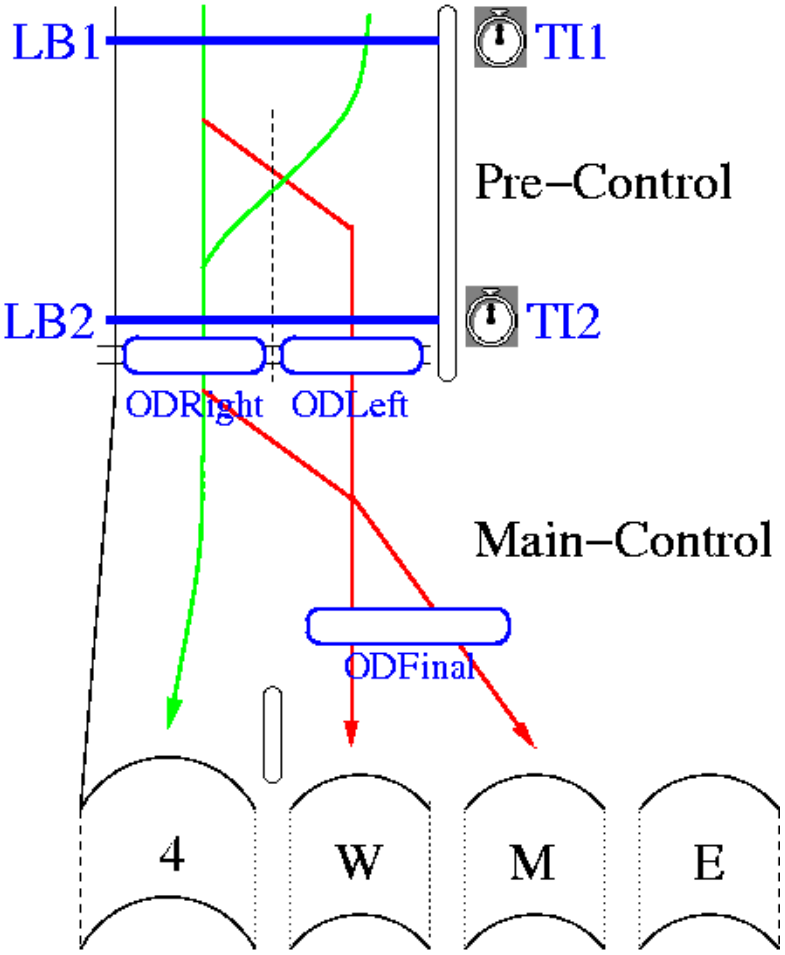

Airbus – Puerto Plata

# Ein Beispielsystem

- Beitrag zu einer Industriekooperation mit (Siemens)
- Erweiterung des neuen Hamburger Elbtunnels um eine 4. Röhre
- Inbetriebnahme 2004

# Die Höhenkontrolle

- Komponenten:
  - Lichtschranken (LB)
  - Überkopfdetektoren (OD)
  - Timers (TI)
  - Steuersoftware
- Umgebung:
  - Normale Autos
  - Hohe Fahrzeuge (HV)
  - Überhohe Fahrzeuge (OHV)



# Sicherheitsziele

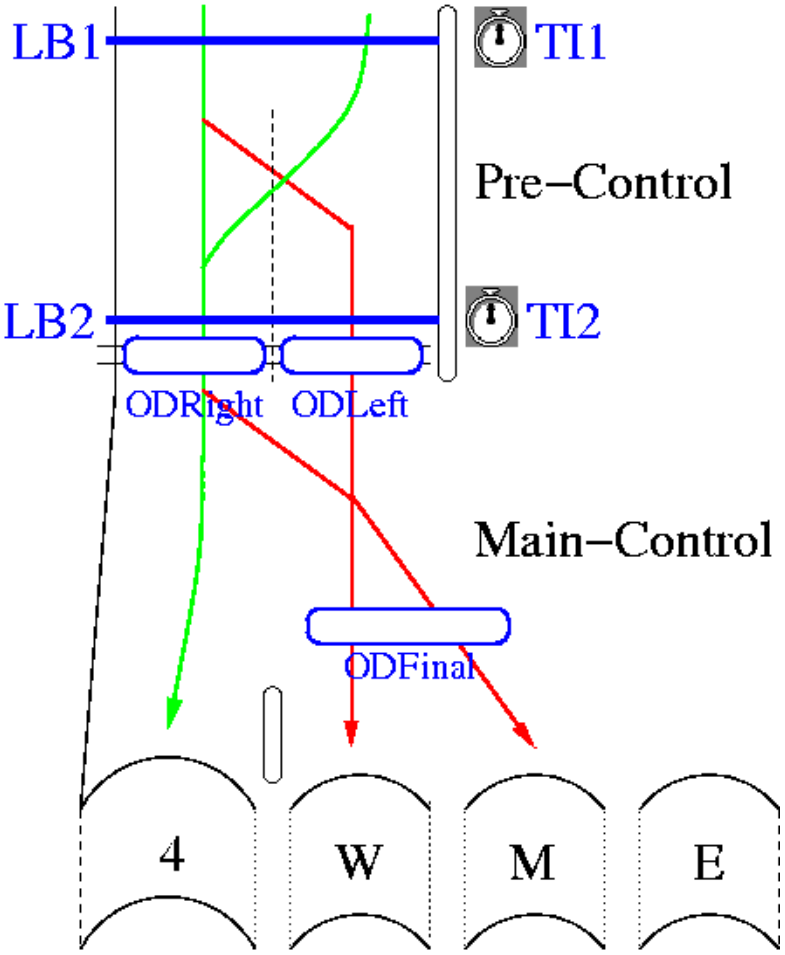
- Antagonistische Sicherheitsziele:
  - „Keine Kollisionen“

    *falls ein überhohes Fahrzeug auf eine falsche Röhre zu steuert, muss ein Nothalt signalisiert werden.*

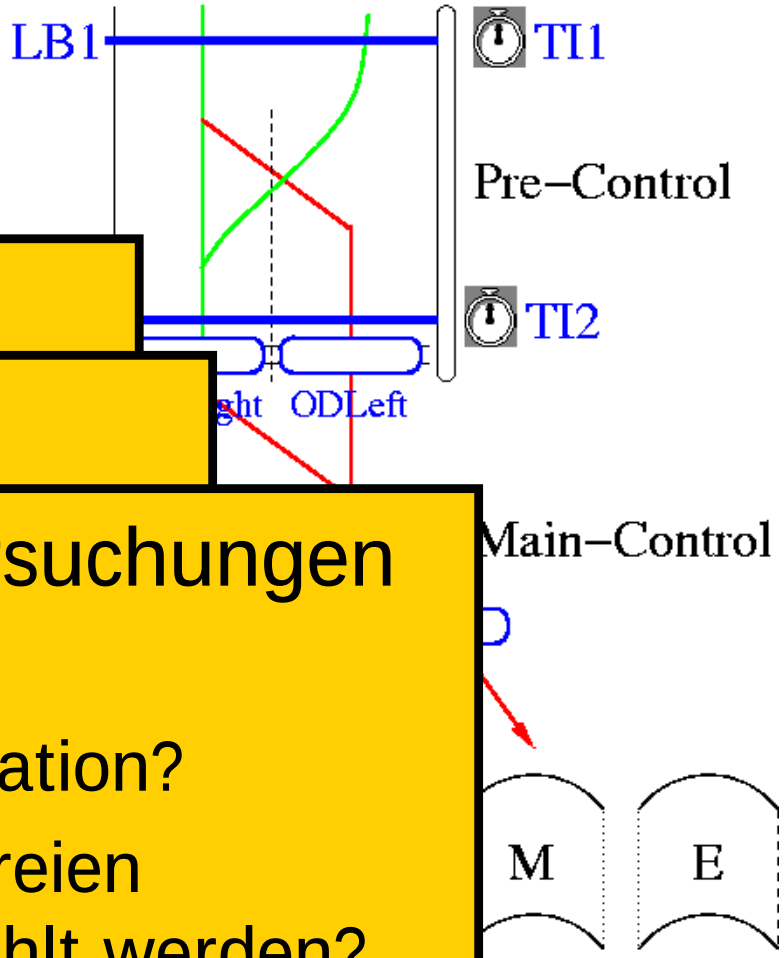
    *(hazard Kollision)*
  - „Keine Fehlalarme“

    ein Alarm darf nicht ausgelöst werden, falls kein OHV auf einer falschen Route fährt.

    *(hazard Fehlalarm)*



# Sicherheitsrelevante Fragen




Frage 1: Funktionale Korrektheit

- 
- 
- 
- 

Frage 2: Fehlertoleranz

- 
- 

Frage 3: Quantitative Untersuchungen

- Wie sicher ist das System?
- Was ist die optimale Konfiguration?
- Welche Werte sollen für die freien Parameter des Systems gewählt werden?

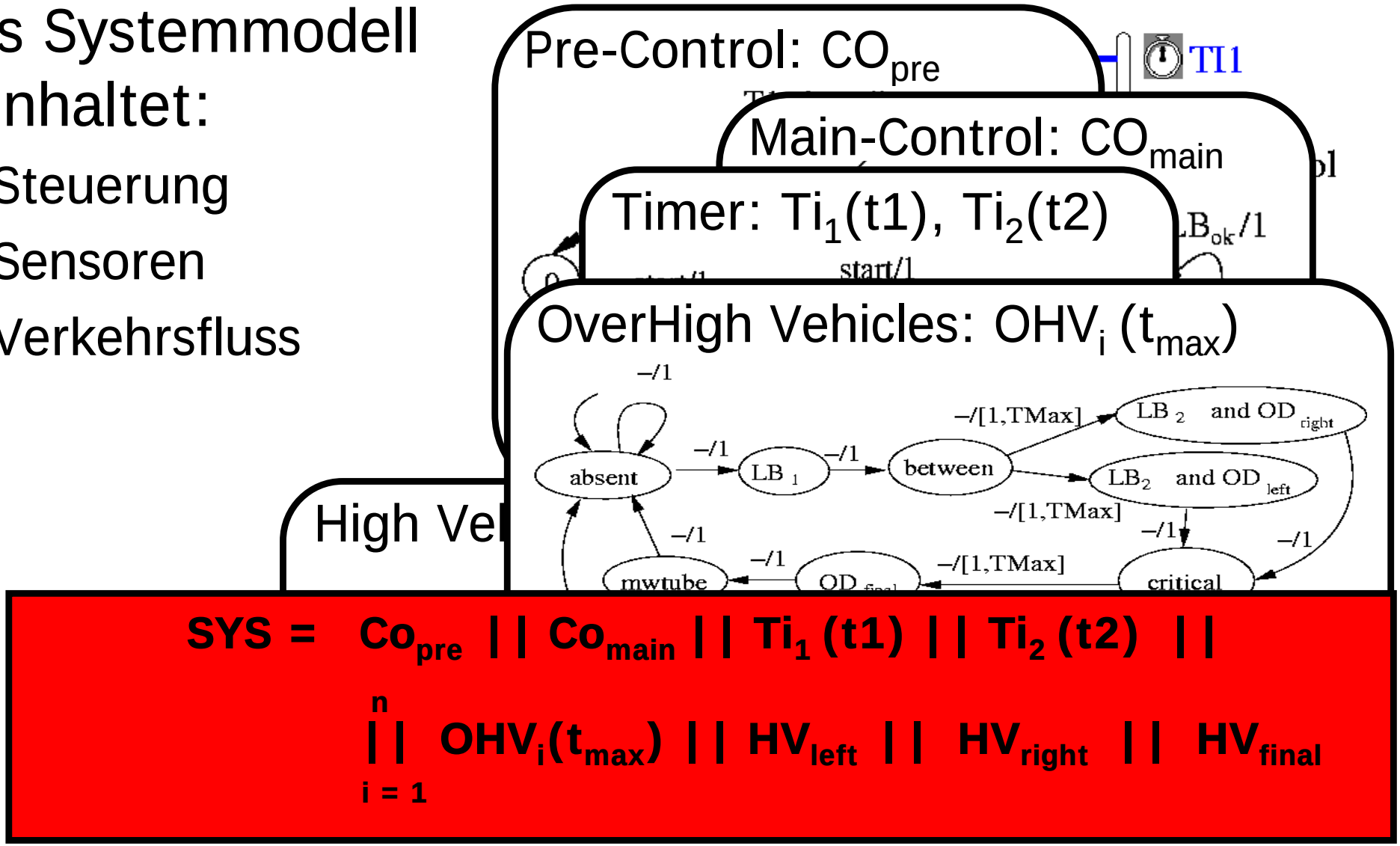
# Modell-basierte Sicherheitsanalyse

- Modell-basierte Sicherheitsanalyse beruht auf der Untersuchung eines Modells des Gesamtsystems

- Dazu gehören Modelle der
  - Steuersoftware
  - Hardwarekomponenten
  - Umgebung
  - Fehler-/Ausfallmodi

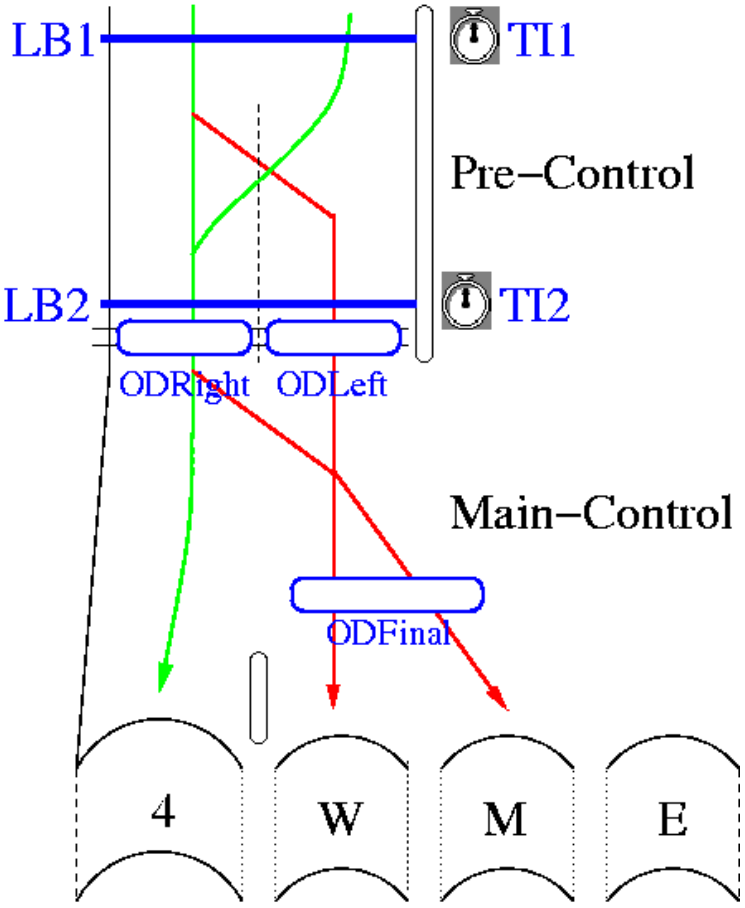
# Modellbildung am Beispiel Elbtunnel

- Das Systemmodell beinhaltet:
  - Steuerung
  - Sensoren
  - Verkehrsfluss



# Beispiel: Elbtunnel

- Formales Modell beinhaltet außerdem **Fehler- und Asufallmodi**:
  - Ausfälle der Überkopfdetektoren
    - Misdetektion ($MD_{Left}$, $MD_{Right}$, $MD_{Final}$)
    - Fehldetektion ($FD_{Left}$, $FD_{Right}$, $FD_{Final}$)
    - -> insgesamt: 6 Fehlermodi
  - Ausfälle der Überkopfdetektoren
    - Fehldetektion ($FD_{LB1}$, $FD_{LB2}$)
    - -> insgesamt: 2 Fehlermodi
  - „Fehlermodi“ der Fahrzeugführer
    - LKW-Fahrer ignoriert StVO und fährt auf der linken Spur ($HV_{Left}$, $HV_{Final}$)
    - -> insgesamt: 2 Fehlermodi
  - Überhohe Fahrzeuge geraten in einen Stau ($OT_1$, $OT_2$)
    - Insgesamt -> 2 Fehlermodi




➡ Formale Fehlermodellierung (nicht in diesem Vortrag)

**[Ortmeier, Schellhorn 2007]**

# Frage 1: Funktionale Korrektheit

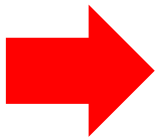
- Zentrale Frage: „Erfüllt das System seine Sicherheitseigenschaft?“
- Traditionelle Antworten:
  - Beachten von Design- und Konstruktionsrichtlinien
  - Orientierung an Erfahrungswerten
  - Expertenbefragungen/-reviews
  - Tests

# Beobachtung

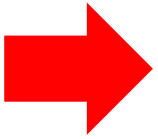
- Funktionale Korrektheit (Frage 1) ist ein Spezialfall von Fehlertoleranz (Frage 2)

- Funktionale Korrektheit: „Erfüllt das System seine Sicherheitseigenschaft (im ungestörten Betrieb)?“

- Fehlertoleranz: „Erfüllt das System seine Sicherheitseigenschaft, obwohl n-Ausfälle auftreten?“



Funktionale Korrektheit und Fehlertoleranz können gemeinsam betrachtet werden.



Deduktive-Ursache-Wirkungsanalyse (DCCA)

# DCCA - Definitionen

Informell:

Sei $\Delta$ die Menge aller betrachteten Fehlermodi, eine Teilmenge $\Gamma$ $\mu$ $\Delta$ heißt kritisch für einen hazard H, gdw. es einen Ablauf gibt

a) auf dem der hazard H auftritt und

b) auf dem keine Ausfälle aus $\Delta$ n $\Gamma$ zuvor aufgetreten sind.

Formal:

$$\text{Critical}(\Gamma) \Leftrightarrow \text{SYS} \models \text{E}\,(\text{only}_{\Delta}(\Gamma)\ \text{until H})$$

wobei $\text{only}_{\Delta}(\Gamma) \Leftrightarrow \bigwedge_{F \in \Delta \setminus \Gamma} \neg\, \text{F}$

A Menge heißt minimal kritisch, gdw. keine echte Teilmenge von ihr kritisch ist.

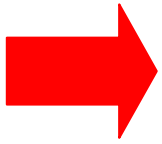
# DCCA – Theorem

Definition:

DCCA ist die Bestimmung minimal kritischer Mengen. Eine vollständige DCCA ist die Bestimmung aller minimal kritischen Mengen.

Theorem:

Wenn ein Element jeder minimal kritischen Menge verhindert wird, so wird auch der hazard sicher verhindert.



- Keine Ursachen wurden vergessen
- Minimal kritische Menge beschreiben das intuitive Verständnis einer Ursache-Wirkungsrelation
  - Ausfälle führen – in zumindest einem denkbaren Szenario – zum Hazard
  - Hazard kann nicht auftreten ohne, dass zuvor die Ausfälle aufgetreten sind

# Frage 1: Funktionale Korrektheit

- Ist die leere Menge von Fehlermodi kritisch?
- Beweisverpflichtung:

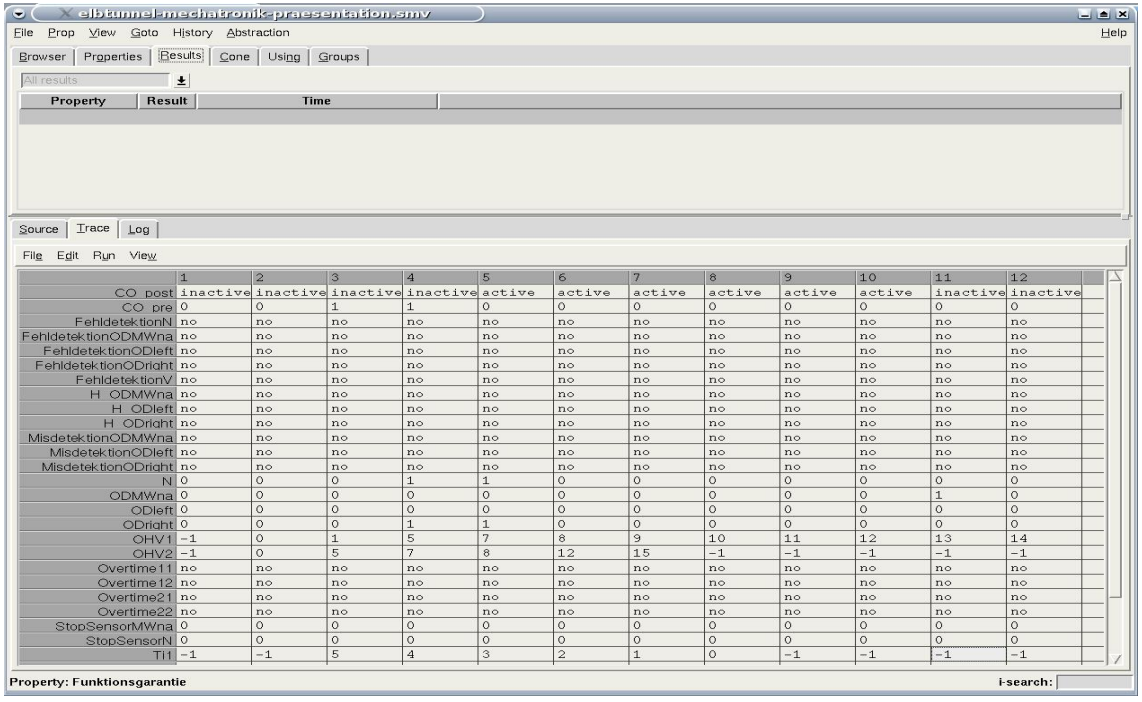
$$SYS \vdash E\ (only_{\Delta}(;)\ until\ H)$$

- In Umgangssprache:

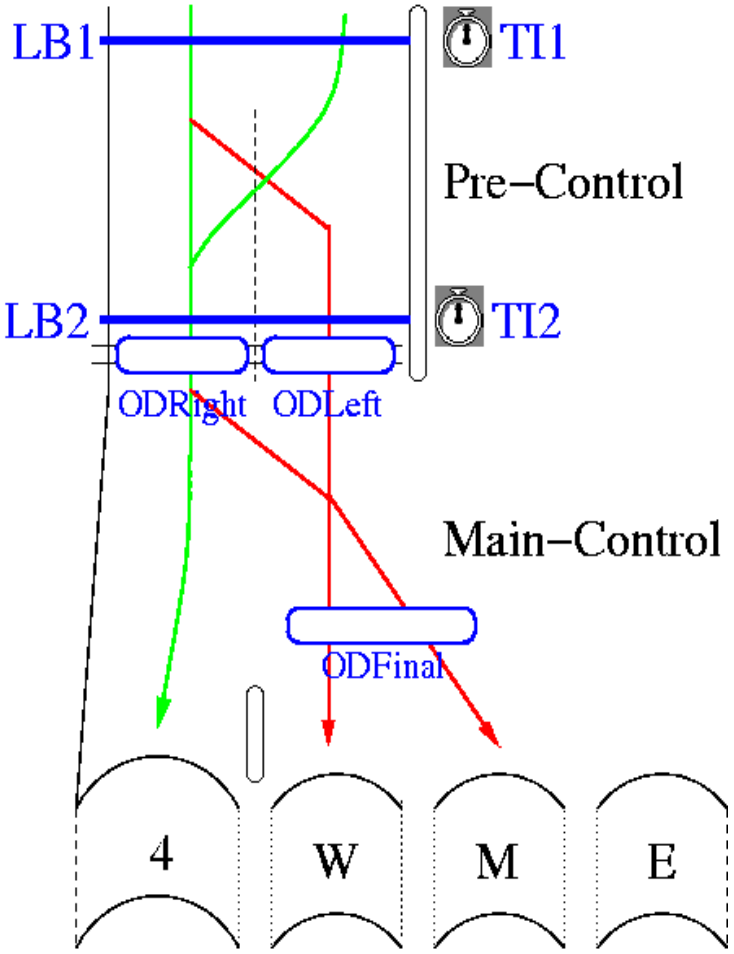
  „Gibt es einen Ablauf, auf dem der hazard H auftritt und kein Komponentenausfall zuvor aufgetreten ist?“ (= Funktionale Inkorrektheit)

# Antwort auf Frage 1 am Beispiel

- Leider nein!
- Gegenbeispiel wird generiert

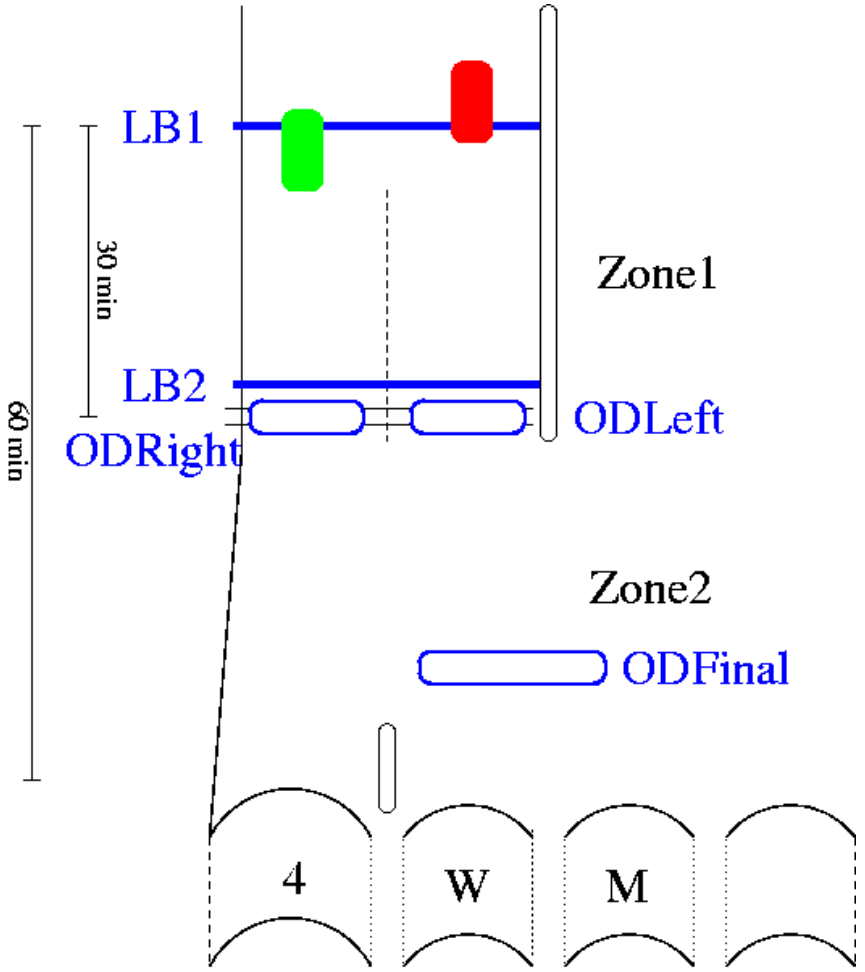



| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CO_post | inactive | inactive | inactive | inactive | active | active | active | active | active | active | inactive | inactive |
| CO_pre | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| FehldetektionN | no | no | no | no | no | no | no | no | no | no | no | no |
| FehldetektionODMWna | no | no | no | no | no | no | no | no | no | no | no | no |
| FehldetektionODleft | no | no | no | no | no | no | no | no | no | no | no | no |
| FehldetektionODright | no | no | no | no | no | no | no | no | no | no | no | no |
| FehldetektionV | no | no | no | no | no | no | no | no | no | no | no | no |
| H_ODMWna | no | no | no | no | no | no | no | no | no | no | no | no |
| H_ODleft | no | no | no | no | no | no | no | no | no | no | no | no |
| H_ODright | no | no | no | no | no | no | no | no | no | no | no | no |
| MisdetektionODMWna | no | no | no | no | no | no | no | no | no | no | no | no |
| MisdetektionODleft | no | no | no | no | no | no | no | no | no | no | no | no |
| MisdetektionODright | no | no | no | no | no | no | no | no | no | no | no | no |
| N | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| ODMWna | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| ODleft | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| ODright | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| OHV1 | -1 | 0 | 1 | 5 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| OHV2 | -1 | 0 | 5 | 7 | 8 | 12 | 15 | -1 | -1 | -1 | -1 | -1 |
| Overtime11 | no | no | no | no | no | no | no | no | no | no | no | no |
| Overtime12 | no | no | no | no | no | no | no | no | no | no | no | no |
| Overtime21 | no | no | no | no | no | no | no | no | no | no | no | no |
| Overtime22 | no | no | no | no | no | no | no | no | no | no | no | no |
| StopSensorMWna | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| StopSensorN | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| TI1 | -1 | -1 | 5 | 4 | 3 | 2 | 1 | 0 | -1 | -1 | -1 | -1 |



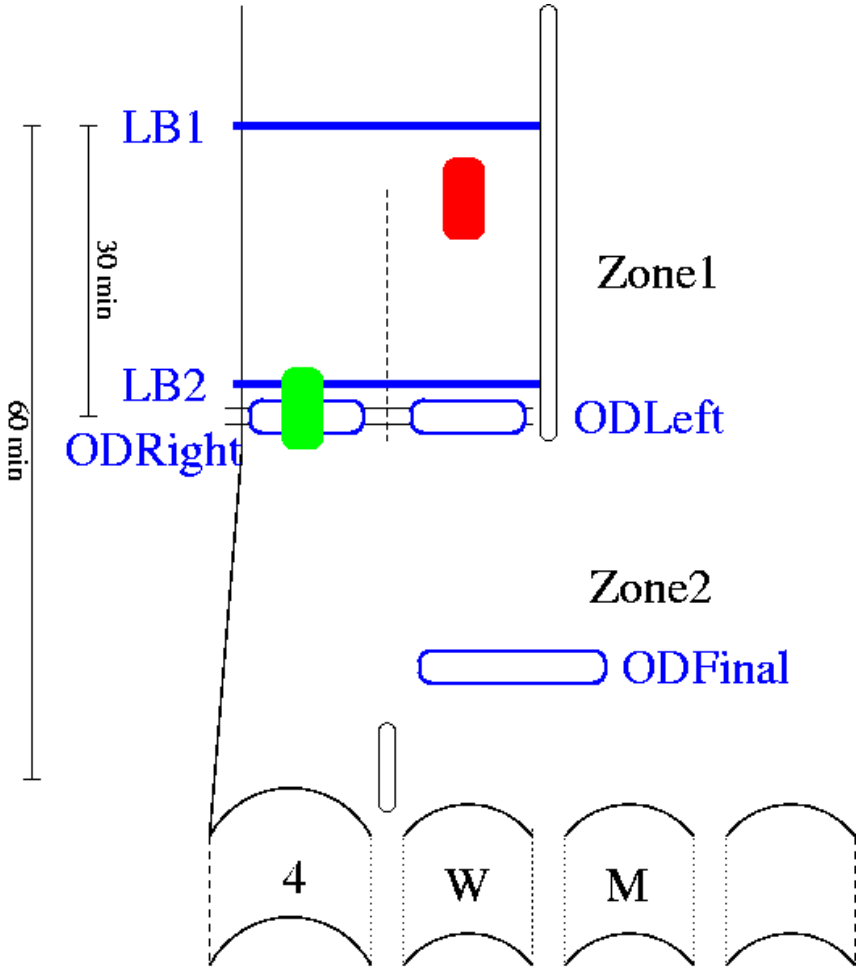
# Wo liegt das Problem?

- T=0min
  - zwei OHVs fahren gleichzeitig durch LB1
  - OHV-Zähler wird um 1 erhöht (ZV=1)
  - ABER: das System hat ein OHV „vergessen“



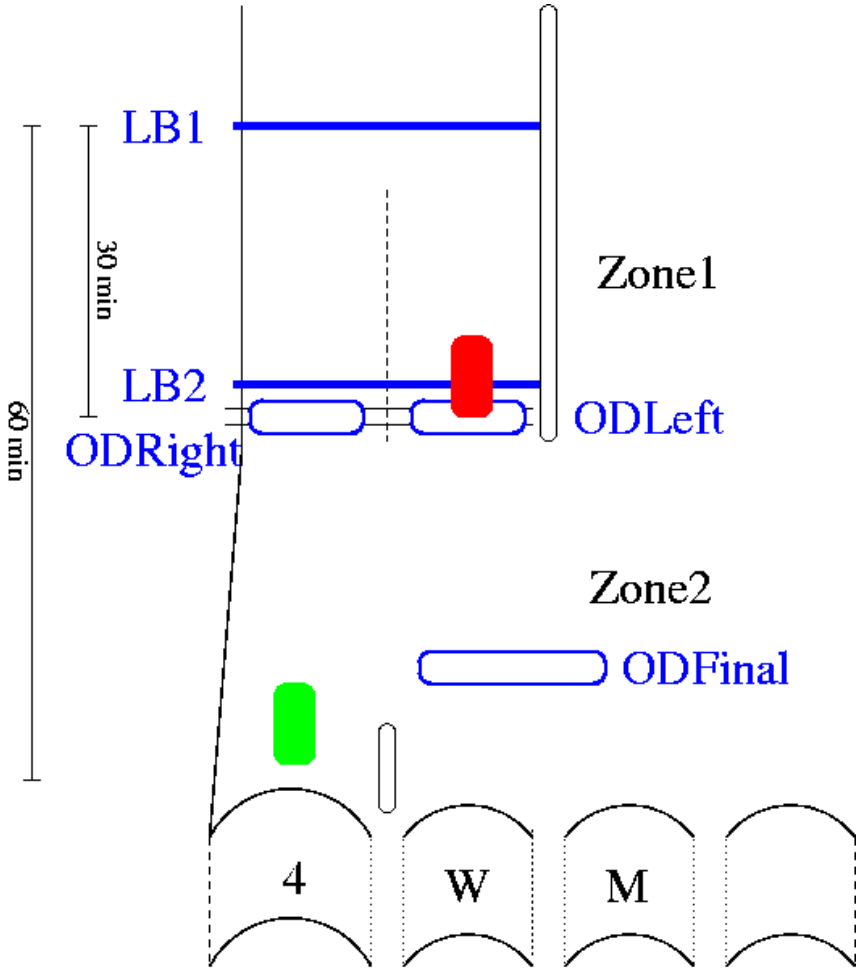
# Wo liegt das Problem?

- T=5min
  - grünes OHV passiert LB2
  - LB2 wird deaktiviert (OHV-Zähler ZV=0)
  - beide OHVs bewegen sich mit stark unterschiedlicher Geschwindigkeit
  - ODFinal wird aktiviert



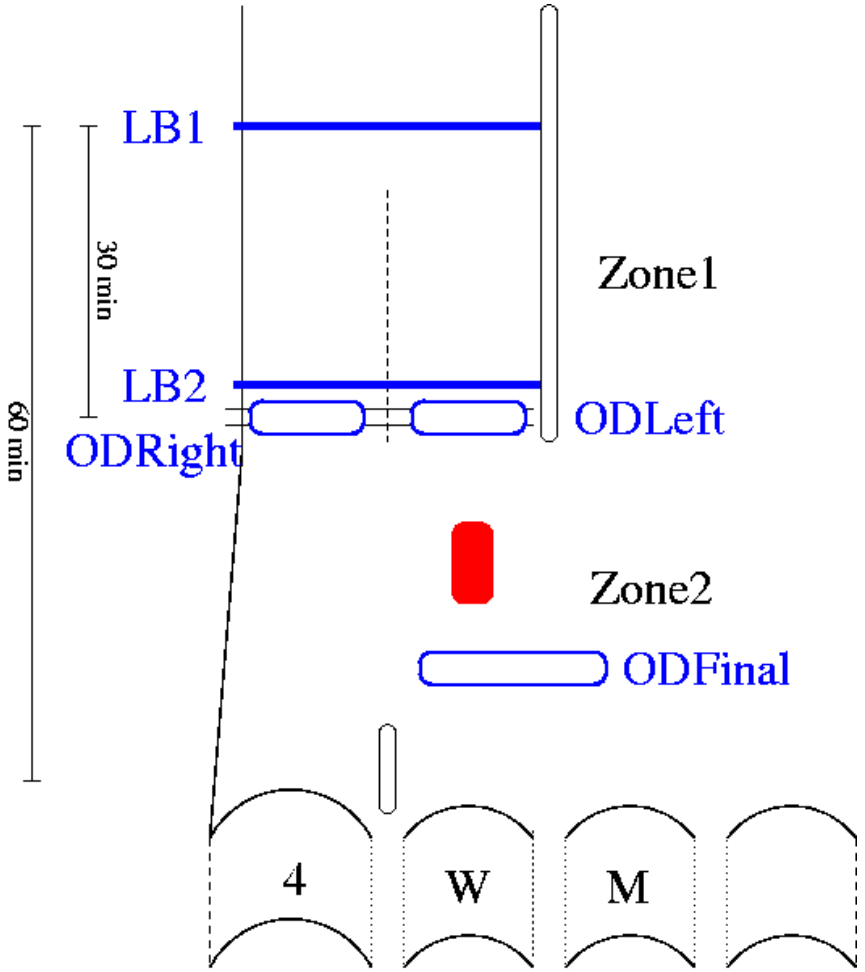
# Wo liegt das Problem?

- T=20min
  - rotes OHV passiert LB2
  - aber LB2 ist deaktiviert
  - grünes OHV passiert den Tunnel korrekt



# Wo liegt das Problem?

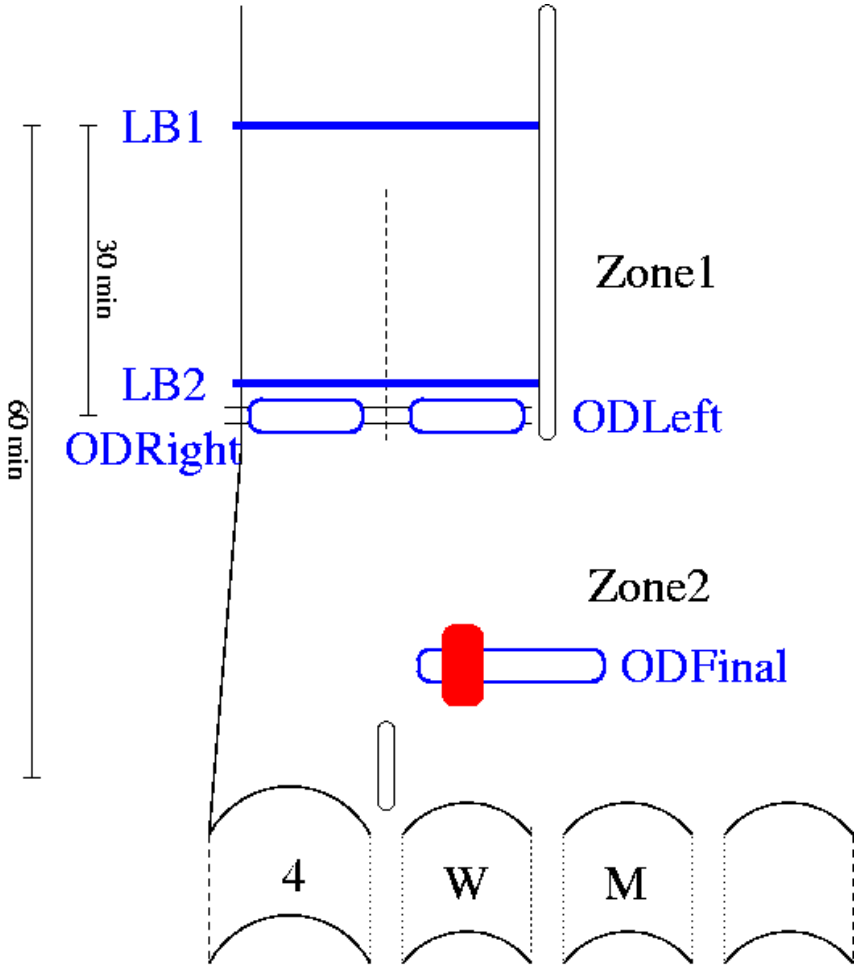
- T=35min
  - ODFinal wird deaktiviert, wegen Timeout von Timer2 (da LB2 von grünem OHV bei T=5min passiert wurde)



# Wo liegt das Problem?

- T=40min
  - rotes OHV passiert ODFinal,
  - aber der Sensor ist deaktiviert

**KOLLISION**



# Elbtunnel - Sicherheitslücke

- Sicherheitslücke war bislang unentdeckt!
- Problem wurde den Entscheidungsträgern vorgelegt

- Lösung (durch die Träger vorgegeben):
  - Durchfahren von LB1 auf der linken Spur wird per StVO verboten

- Konsequenz (für die Analyse):
  - Füge diese Situation zur Menge der Fehlermodi hinzu:

$$\Delta' = \Delta \cup \{OHV_{Sim}\}$$

# Antwort auf Frage 1 am Beispiel

- Ist die leere Menge von Fehlermodi kritisch?
- Beweisverpflichtung:

$$SYS \vdash E\ (\ only_{\Delta'}(;)\ until\ H)$$

- In Umgangssprache:

  „Gibt es einen Ablauf, auf dem der hazard H auftritt und kein Komponentenausfall zuvor aufgetreten ist?“ (= Funktional Inkorrektheit)

- Ergebnis: NEIN!
- Folgerung:
  - Funktionsgarantie (=Antwort auf Frage 1)

# Frage 2: Fehlertoleranz

- Welche n-elementigen Menge von Fehlermodi sind kritisch?
- Beweisverpflichtung:

$$SYS \vdash E\ (\ only_{\Delta'}(\{F_i\})\ until\ H)$$

- In Umgangssprache:

  „Welche einzelne Ausfälle können zum Systemversagen führen?“ (vgl. FMEA)

  „Welche Kombinationen von Ausfällen können zum Systemversagen führen?“ (vgl. FTA, ETA, ...)

- Ergebnis: critical sets

# Vollständige DCCA

- Kombinationen von 2,3, ... Fehlermodi werden untersucht
- Exponentielle Zahl an Beweisverpflichtungen

- Im Beispiel:
  - 13 Fehlermodi werden untersucht
  - Worst case $2^{13}$ = 8192 Beweise notwendig!!
- Aber: ...

# Aufwand DCCA

- Kritikalität ist monoton
  - Falls $\Gamma$ kritisch ist, dann ist jede Obermenge davon kritisch.

$$\Gamma_1 \mu \Gamma_2 ) (\text{critical}(\Gamma_1) ) \text{critical}(\Gamma_2))$$

- Im Beispiel:
  - Für den Elbtunnel sind nur 18 von 8192 Beweisen notwendig um alle minimal kritischen Mengen zu finden
  - Zusammen: weniger als 1 Minute Rechenzeit mit SMV

# Antwort auf Frage 2 am Beispiel

- Minimal kritische Mengen für <u>Fehlalarme</u>
  - $\{MD_{Right}\}$
  - $\{FD_{Left}\}$
  - $\{HV_{Left}\}$
  - $\{HV_{Final}\}$
  - $\{FD_{Final}\}$
  - $\{FD_{LB2}\}$

- Minimal kritische Mengen für <u>Kollision</u>
  - $\{OHV_{Sim}\}$
  - $\{MD_{Final}\}$
  - $\{OT_1\}$
  - $\{OT_2\}$
  - $\{FD_{LB2}\}$

# Frage 3: Ausfallwahrscheinlichkeiten

- Typisches Vorgehen:
  - Wahrscheinlichkeiten für Komponentenausfälle werden vorgegeben
  - Systemausfallwahrscheinlichkeit wird aus diesen Werten und den cut sets approximiert

$$P(hazard) = \sum_{all\ min.CS} P(CS)$$

$$P(CS) = \prod_{FM \in CS} P(FM)$$

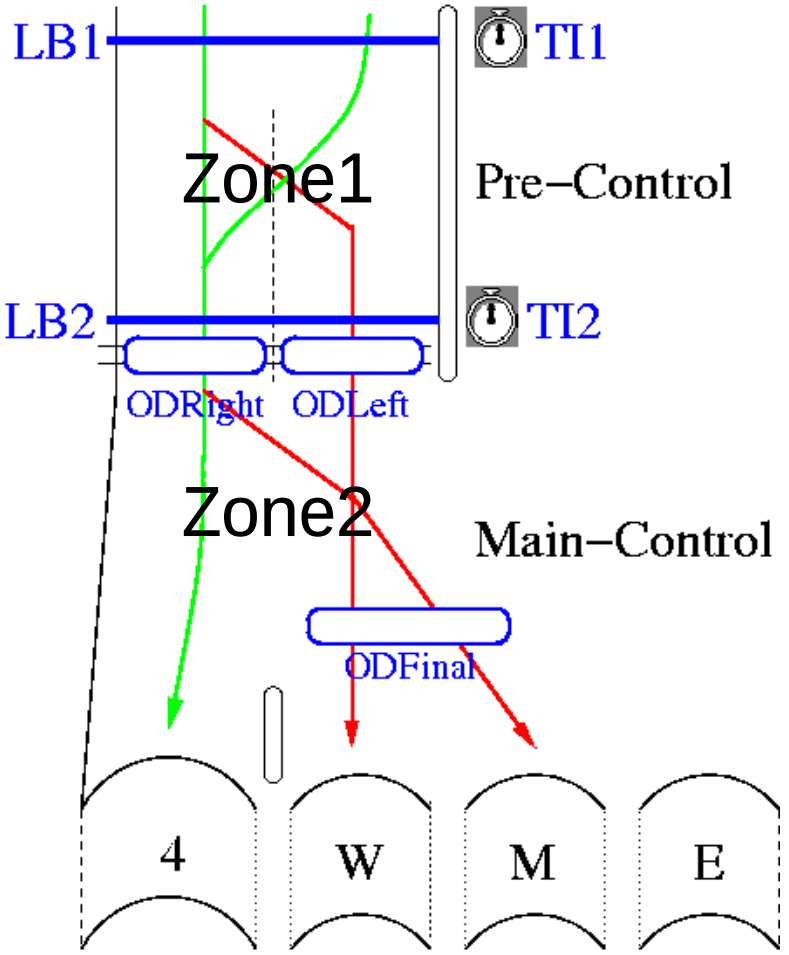
[Vesely, Leveson, ...]

  - Ergebnisse für das Beispiel:
    - P(Fehlalarm) ~ $3*10^{-4}$ /min
    - P(Kollision) ~ $3*10^{-8}$ /min

ABER: Wieso sind das feste Werte????

# Beispiel Elbtunnel

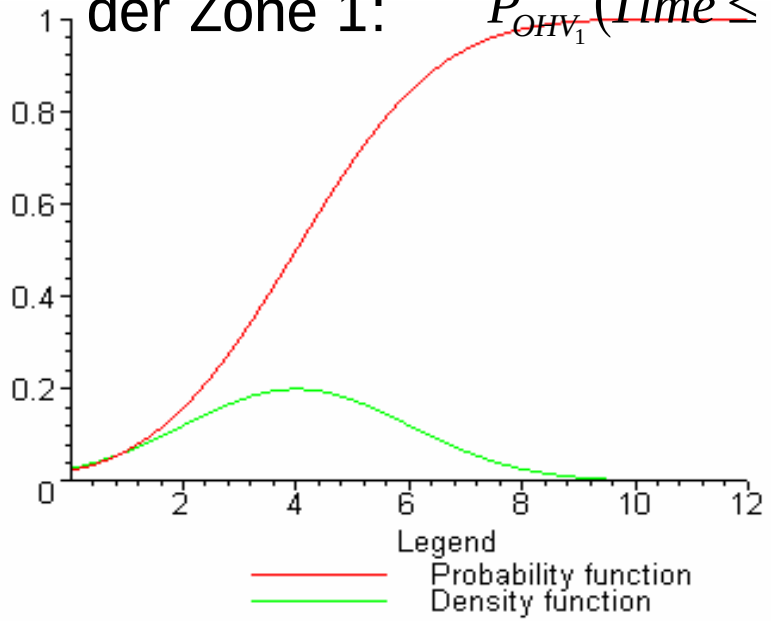
- Ein Single-point-of-failure:
  - „OHV bleibt in einem Stau in Zone 1 länger als die Laufzeit von T1“ ( $\{OT_1\}$ )
  - $P(\{OT_1\}) = ??$



# Alternative

- Stat. Modellierung der Fahrzeiten der OHVs
  - Z.B. Normalverteilung (Erwartungswert 4 min., Standard-abweichung 2 min.)

Fahrzeit zum Durchqueren der Zone 1: $P_{OHV_1}(Time \leq t)$




- Parameterisierte Wahrscheinlichkeit

$$P(OT_1)(runtime1) = 1 - P_{OHV}(Time \leq runtime1)$$

# Ergebnis

- Setze diese parametrisierten Wahrscheinlichkeiten zusammen

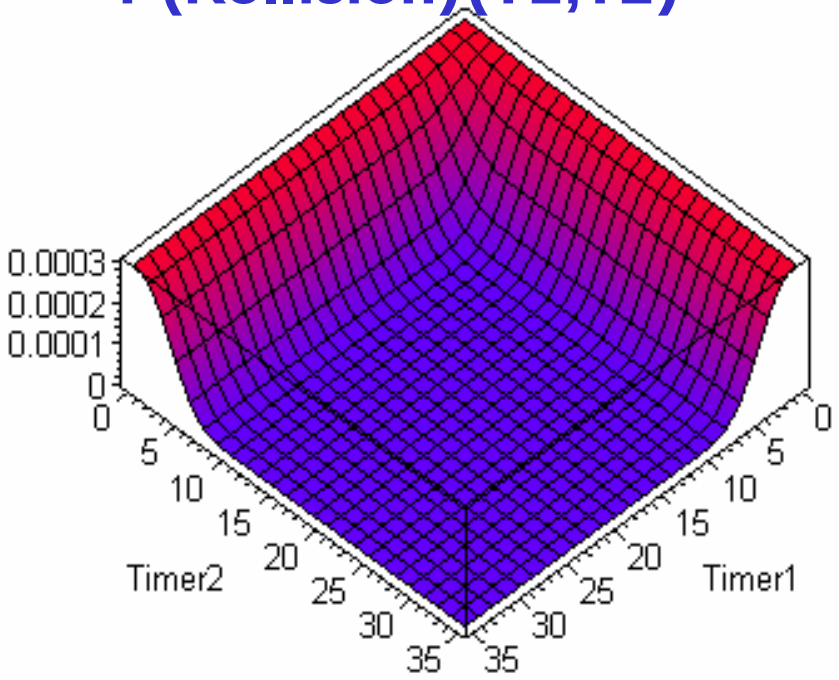
$$P(hazard)(x_1,...,x_n) = \sum_{all\ min.CS} P(CS)(x_1,...,x_n)$$

-> Parametrisierte Wahrscheinlichkeit für den Ausfall des Gesamtsystems

# Beispiel: Elbtunnel

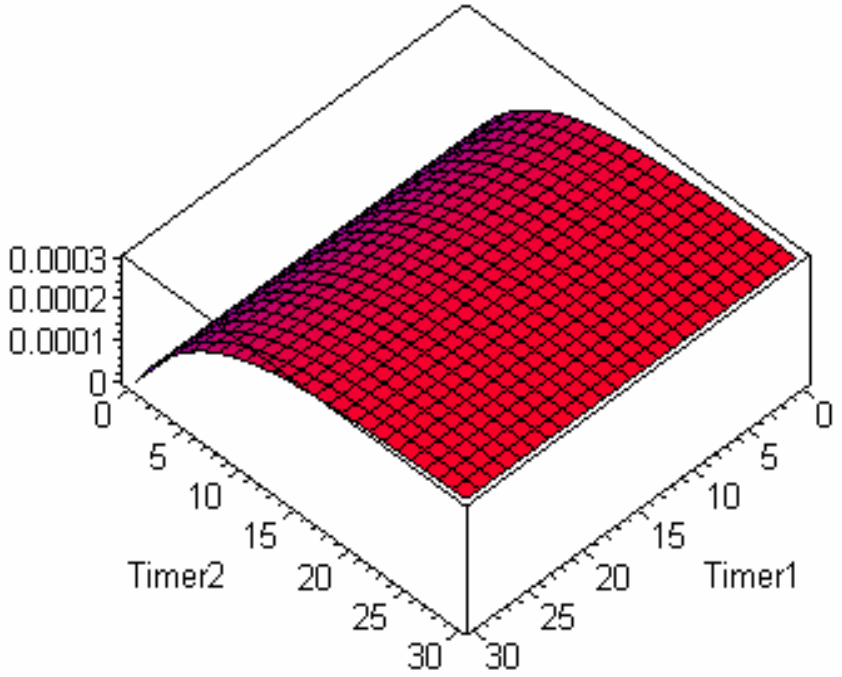
## Kollision

P(Kollision)(T1,T2)



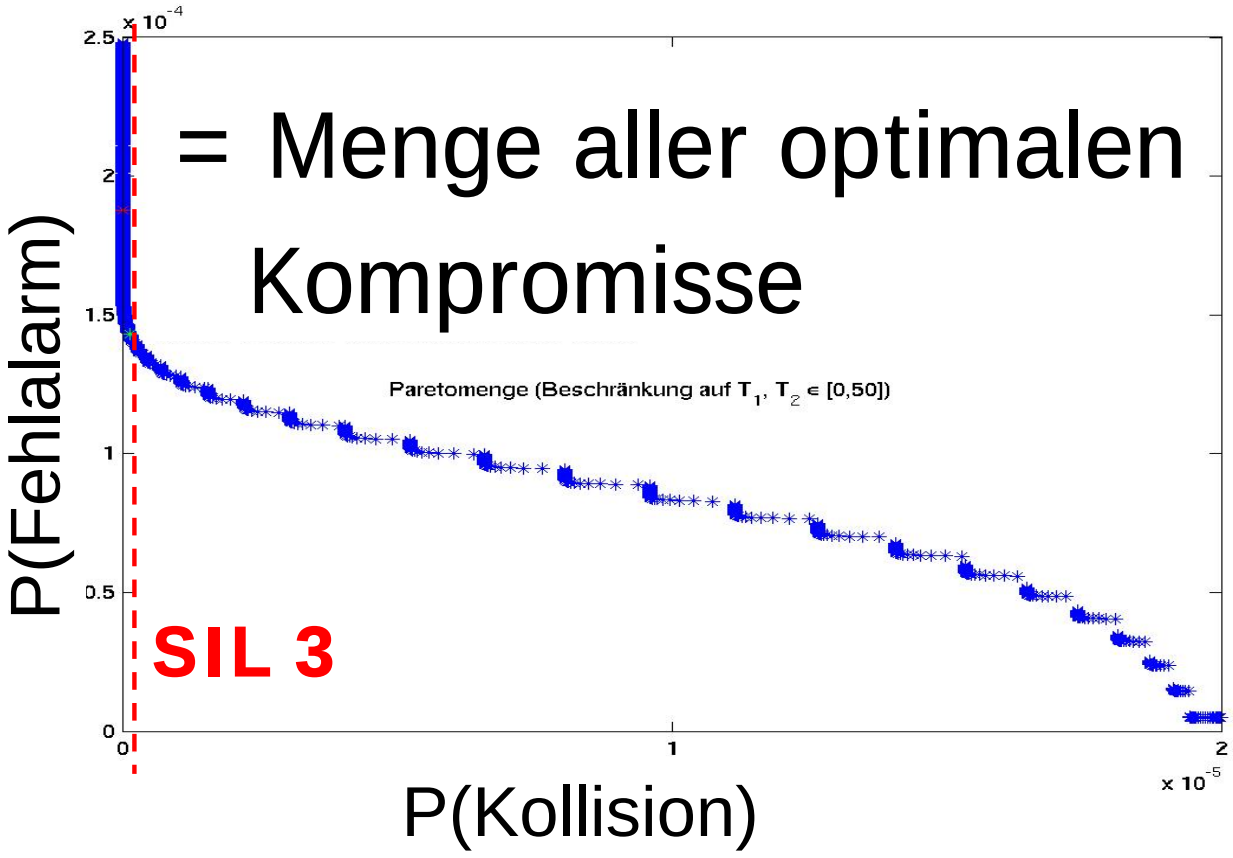

## Fehlalarm

P(Fehlalarm)(T1,T2)




**Antagonistische Ziele**
**-> optimaler Kompromiss??**

# Pareto-Optimierung




= Menge aller optimalen Kompromisse

Fehlalarme sinken um 10%, bei Einhaltung von SIL 3

**Optimale Timerlaufzeiten:**

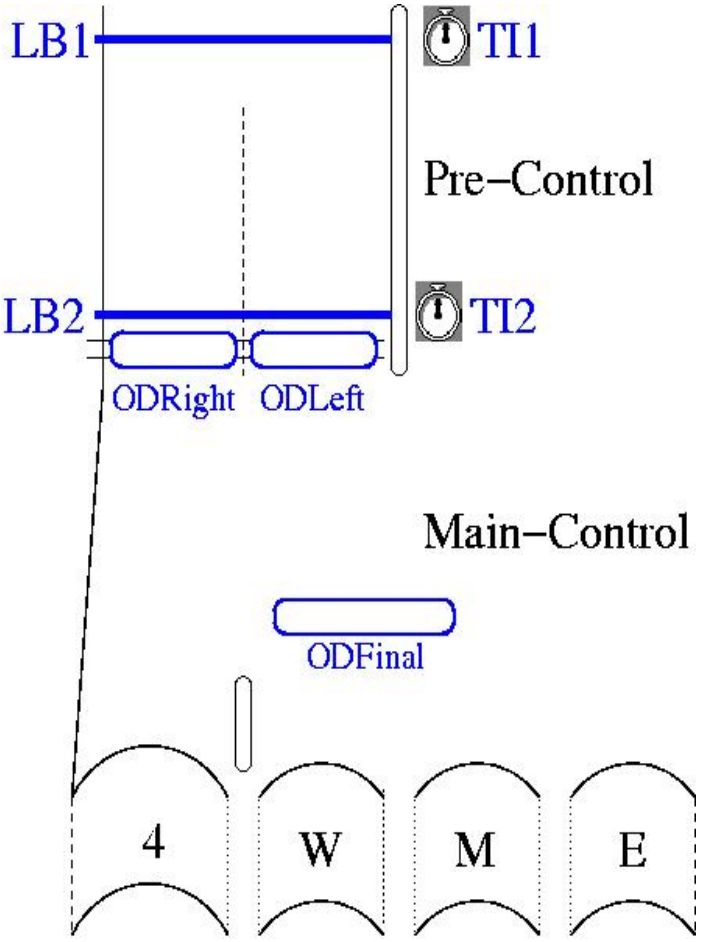
Timer1 ca. 19 Minuten
Timer2 ca. 15.6 Minuten
(statt jeweils 30 Min)

Wieso trotzdem nur 10% Reduktion bei Fehlalarmen?

# Wieso nur 10 % Verbesserung?

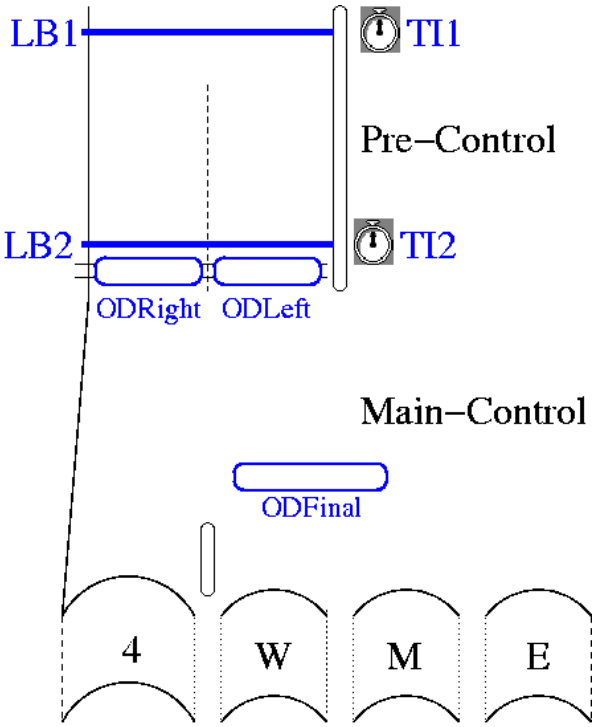
- Über 80% der Fehlalarme werden durch LKWs bei $OD_{final}$ ausgelöst, wenn ein OHV den Tunnel korrekt passiert.
- Wenn ein OHV den Tunnel korrekt durchfährt, wird fast sicher ein Alarm ausgelöst.
- Die Zahl solcher Fehlalarme wächst linear mit Anzahl der OHVs.

**Nicht akzeptabler Designfehler!!**

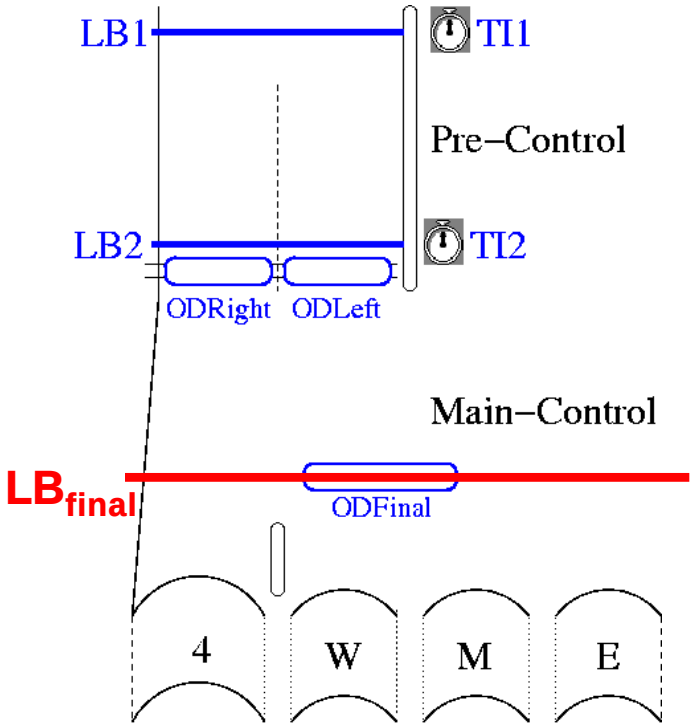


# Variantenexploration

## Geplantes System




## Designvariante




Vorteile wachsen mit der Zahl der OHVs

# Zusammenfassung

- Komplexität der sicherheitskritischen Systeme wächst zunehmend
- Steigende Kritikalität schlägt sich in steigenden Sicherheitsanforderungen nieder
- Modell-basierte Ansätze unterstützen den Ingenieur und können in verschiedenen Entwurfsstadien eingesetzt werden

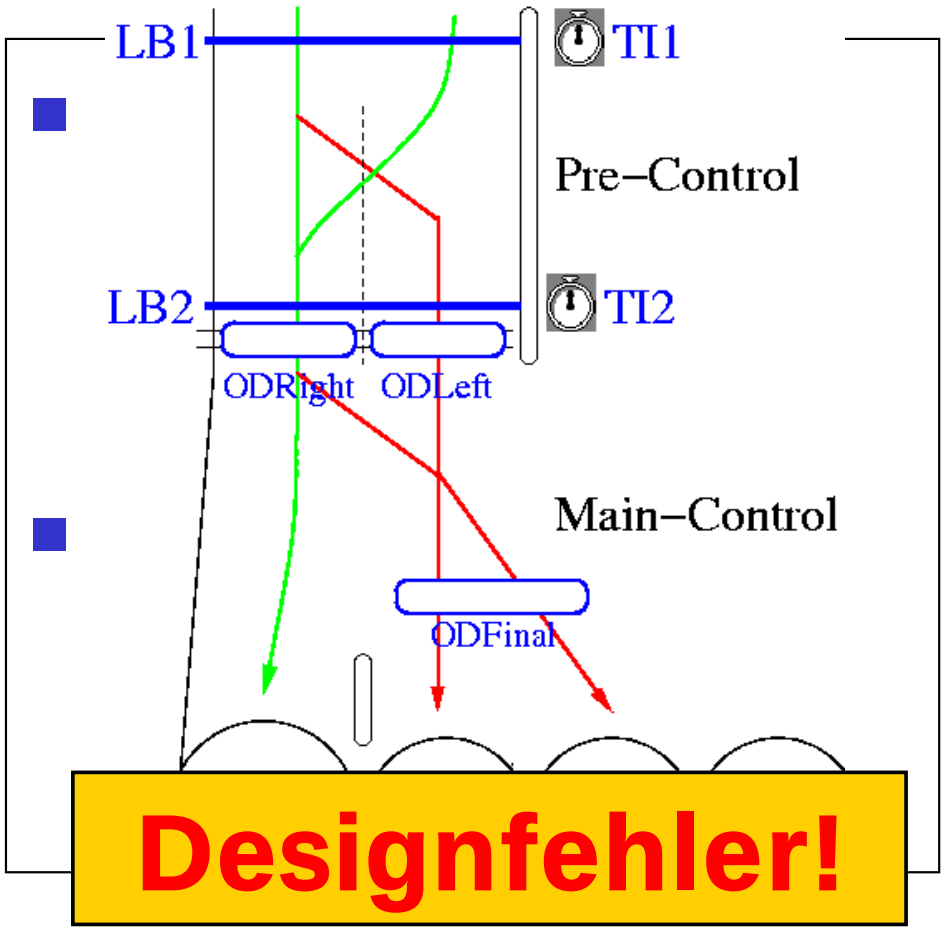
- Toolintegration möglich

Vielen Dank...

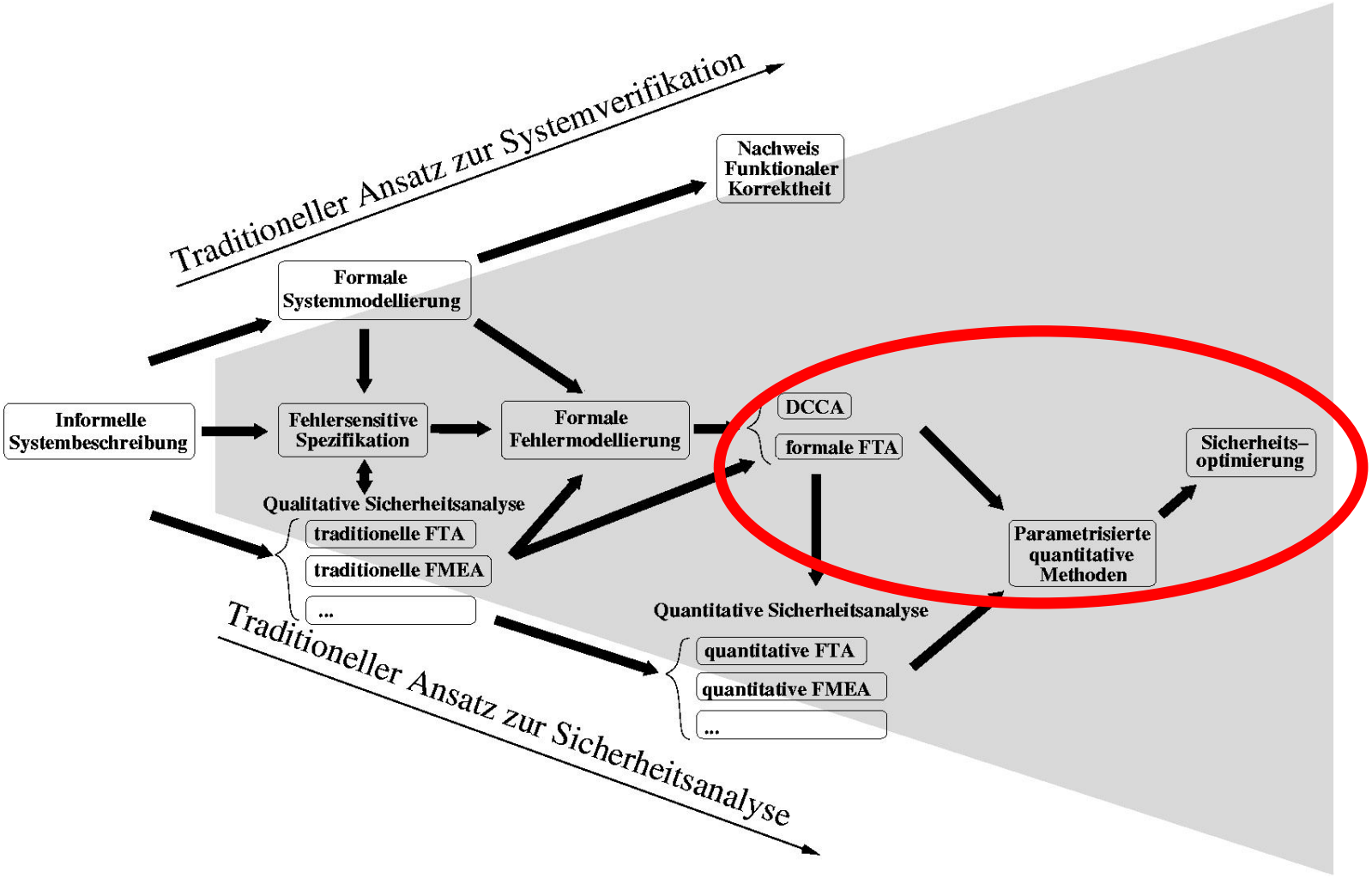

**Dr. Frank Ortmeier**
Lehrstuhl Softwaretechnik und Programmiersprachen
Universität Augsburg

# Ergebnisse für das Beispiel

- Minimal kritische Mengen für Fehlalarme
  - $\{MD_{Right}\}$
  - $\{FD_{Left}\}$
  - $\{HV_{Left}\}$
  - $\{HV_{Final}\}$
  - $\{FD_{Final}\}$
  - $\{FD_{LB2}\}$



# Der ForMoSA Ansatz




Formal Models Safety Analysis (**www.informatik.uni-augsburg.de/swt**)

[Ortmeier et al 04]