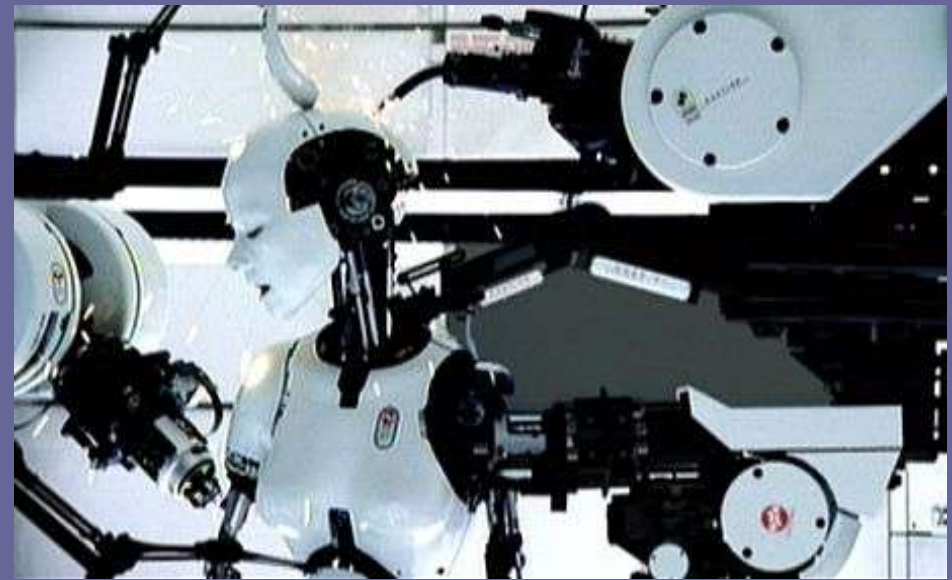


TAV 21 - Berlin
18. Juni 2004

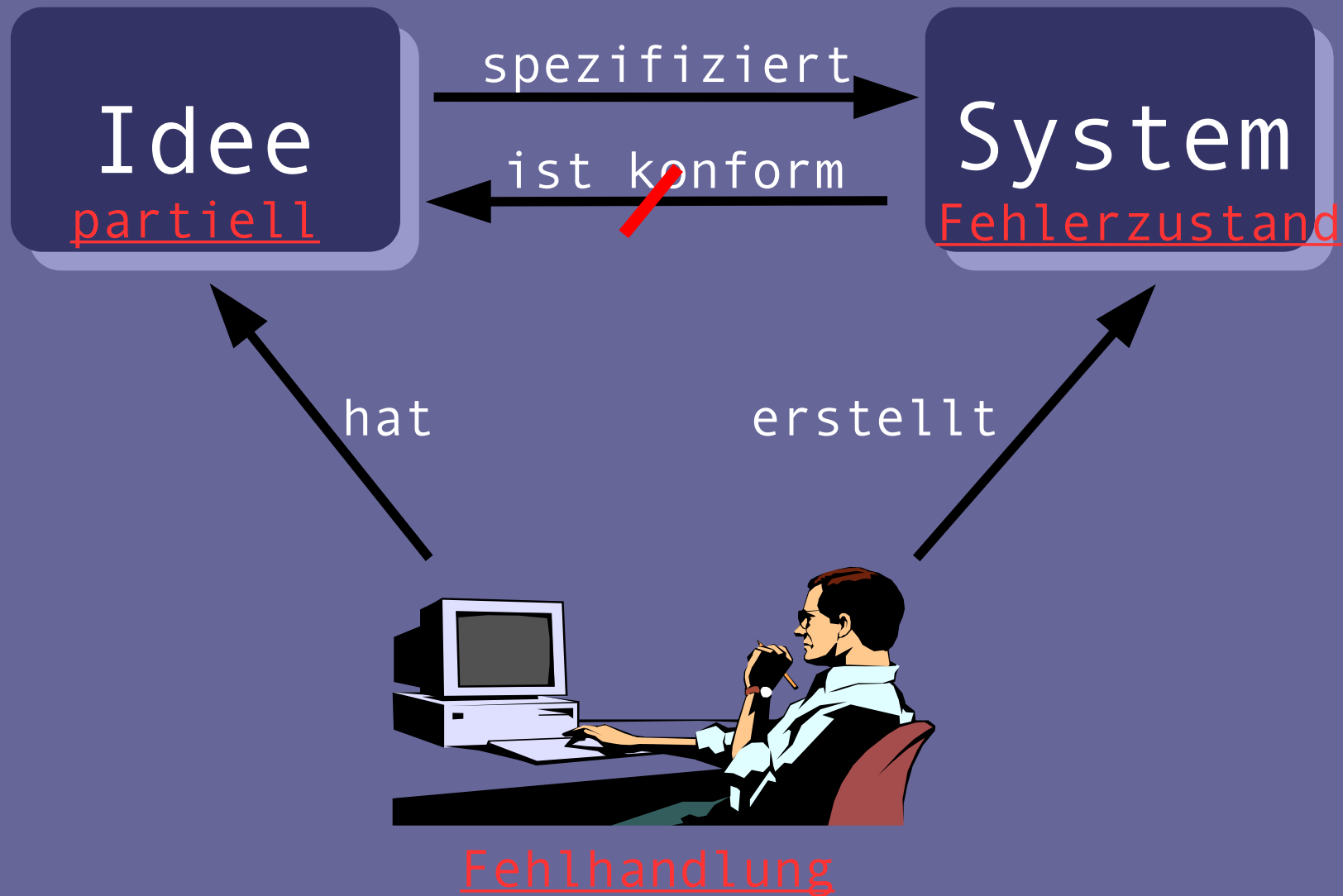


Modellbasiertes Testen - Der ioco Ansatz

Lars Frantzen
Jan Tretmans
{lf,tretmans}@cs.kun.nl
Radboud University Nijmegen



Ideen, Systeme, uns der Human Factor



● Qualitätsaspekte

ISO 9126:

- Funktionalität
- Zuverlässigkeit
- Benutzbarkeit
- Effizienz
- Änderbarkeit
- Übertragbarkeit

Qualität beschreibt die Konformität mit Aspekten einer Systemidee.

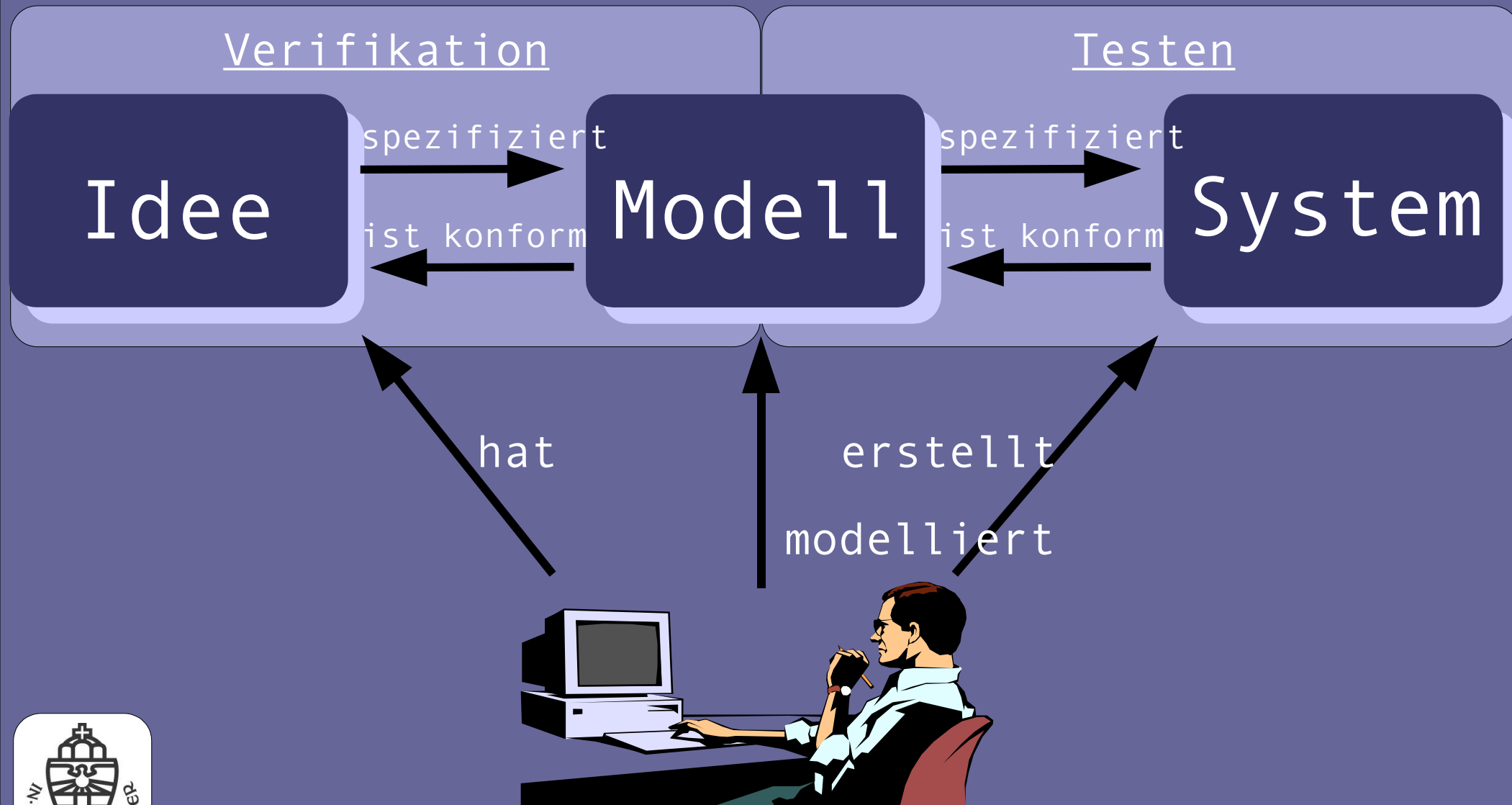
Um Qualität zu messen, müssen diese Aspekte formalisiert, d.h. modelliert werden.

Aspekte können i.A. nicht vollständig modelliert werden.

→ Abstraktion



● Modellbasierte Entwicklung



● Testarten

● Statischer Test:

- Strukturierte Gruppenprüfung
 - Reviews
- Statische Analyse
 - Compiler
 - Standards
 - Datenflussanalyse
 - Kontrollflussanalyse
 - Metriken



● Dynamischer Test:

- Blackbox-Verfahren
 - Äquivalenzklassenbildung
 - Zustandsbezogener Test
- Whitebox-Verfahren
 - Anweisungsüberdeckung
 - Zweigüberdeckung
 - Test der Bedingungen
 - Zweigüberdeckung

Intuitive Testfallermittlung!



● Grenzen des Testens



Testen kann niemals die Fehlerfreiheit eines Systems zeigen, sondern nur dessen Fehlerbehaftung.

Edsger W. Dijkstra

Systeme sind:

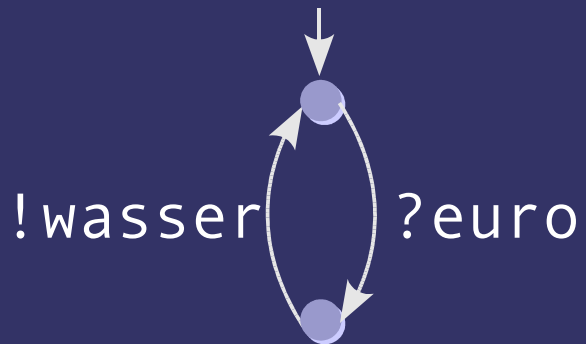
- Immer eine Black Box
 - ➔ Experimentieren ist die einzige Methode, um Wissen zu erlangen.
- Komplex und nicht terminierend
 - ➔ Sie entziehen sich vollständigem Testen.

➔ Testfall Selektion!



● Reaktive Systeme

Modell



System

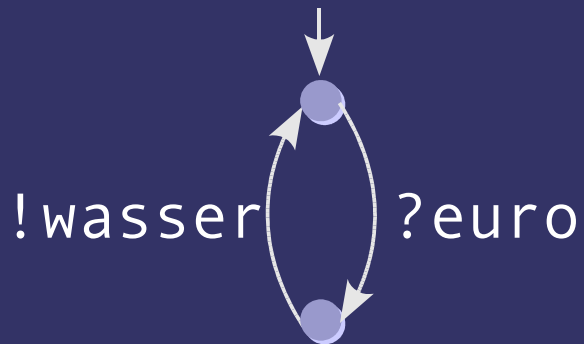


● Test Hypothese

Modell



System

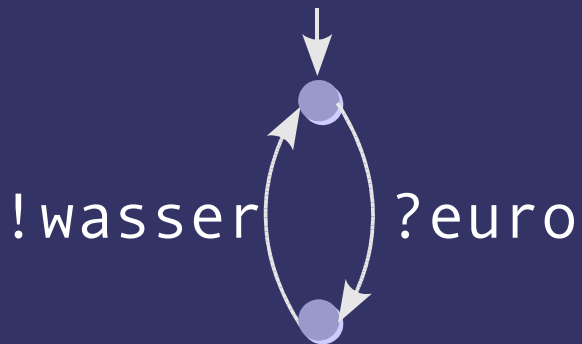


● Modellbasiertes Testen

Modell



System



spezifiziert

ist konform

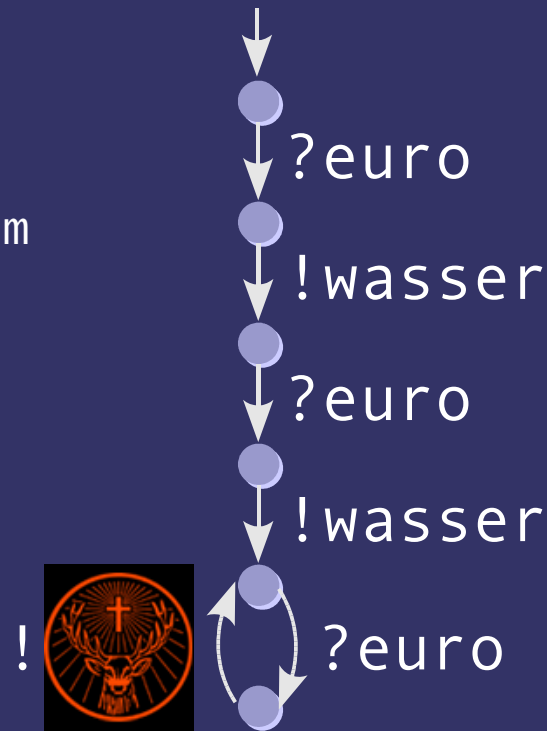
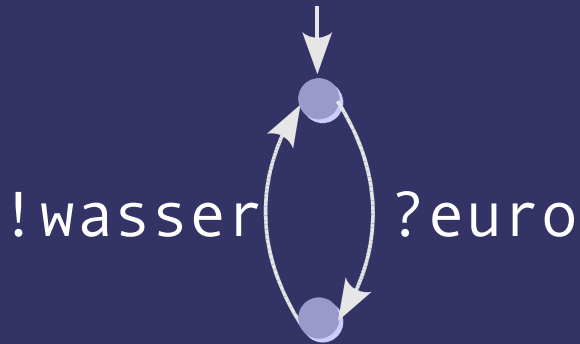


● Modellbasiertes Testen

Modell



System



● Was ist Konformität?

Erster Ansatz: Äquivalenz!

Mealy Machine

\cong

System

$M = \langle I, O, S, \delta, \lambda \rangle$ where

- I , O , and S are finite, nonempty sets of *input symbols*, *output symbols*, and *states* respectively,
- $\delta : S \times I \rightarrow S$ is the *state transition function*,
- $\lambda : S \times I \rightarrow O$ is the *output function*.



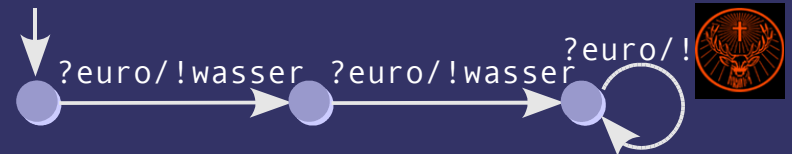
● Was ist Konformität?

Erster Ansatz: Äquivalenz!

Mealy Machine



Mealy Machine



Zwei Maschinen sind äquivalent, wenn sie gleiche Ausgaben für jede Eingabesequenz erzeugen.

● Was ist Konformität?

Erster Ansatz: Äquivalenz!

Problem: Unendliche Eingabesequenzen!

?euroⁿ → !wasserⁿ

Dijkstra revisited:

Systeme erlauben keinen vollständigen Test!

“Lösung” 1: Annahme:

Das System hat nicht mehr Zustände als das Modell

“Lösung” 2: Äquivalenz aufgeben



● Was ist Konformität?

Zweiter Ansatz: Flexiblere Relationen

LTS

conf

System

$M = \langle S, L, T, s_0 \rangle$ where

- S is a countable, non-empty set of *states*,
- $L = L_I + L_U$ is a countable set of *labels*,
- $T \subseteq S \times (L \cup \{\tau\}) \times S$ is the *transition relation*,
- $s_0 \in S$ is the *initial state*.



● Was ist Konformität?

Zweiter Ansatz: Flexiblere Relationen

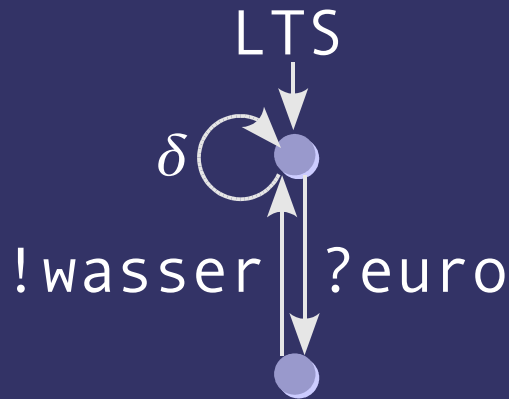
Labelled Transition Systems – Eigenschaften

- Potentiell unendlicher Zustandsraum
- Potentiell nichtdeterministisch
- Kompositional
- Asynchrones I/O Verhalten
- Explizites Testen von Ruhezuständen des SUT

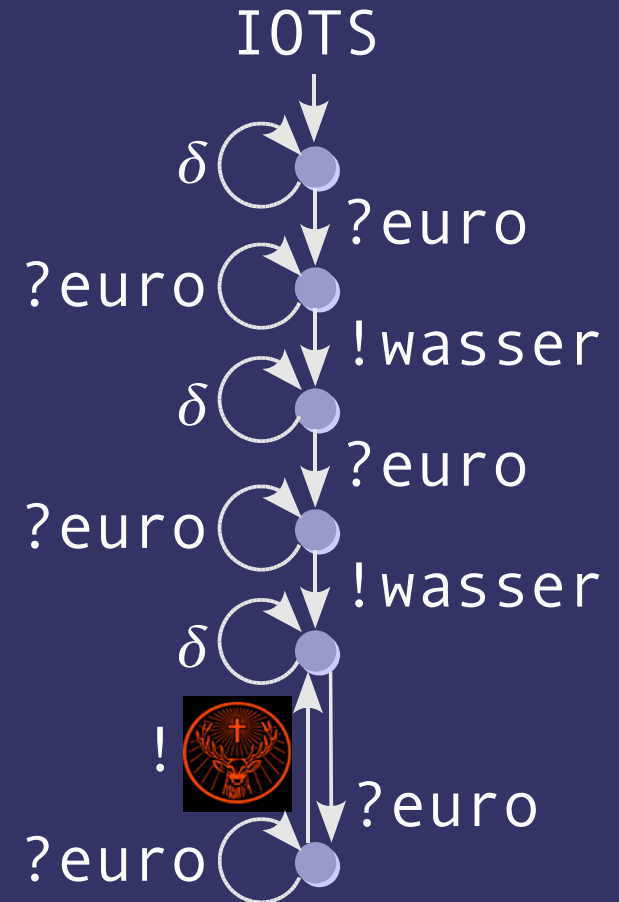


Was ist Konformität?

Die ioco Relation



~~ioco~~



$i \text{ ioco } s \Leftrightarrow_{\text{def}}$

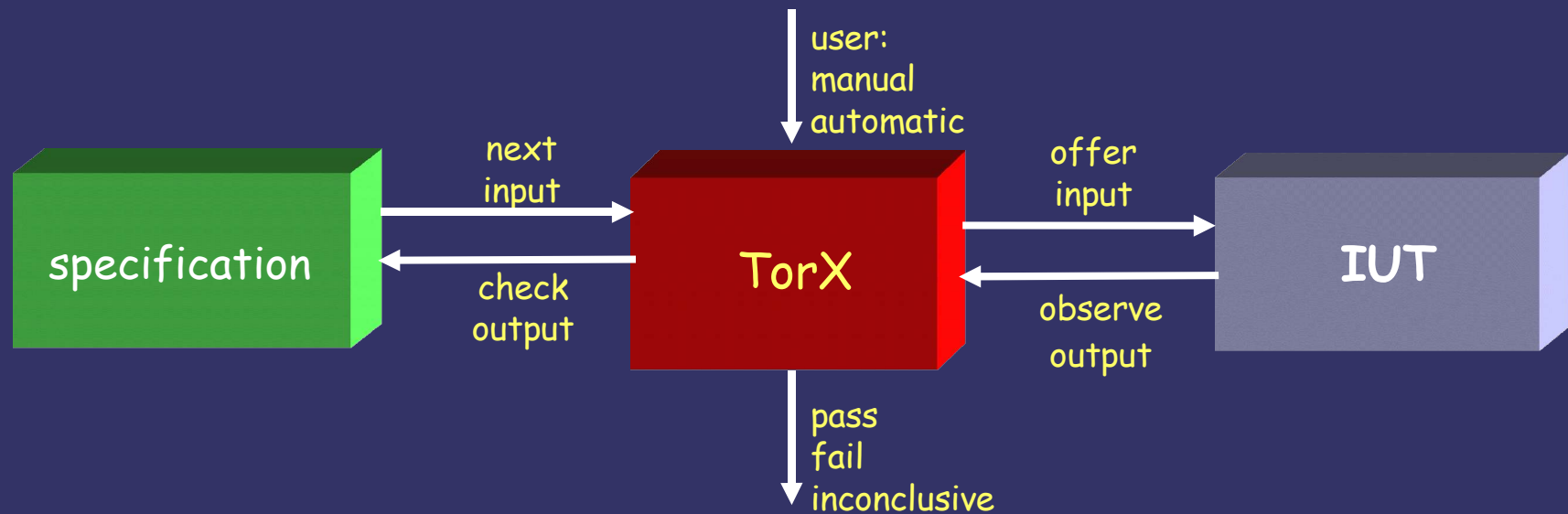
$\forall \sigma \in \text{Straces}(s) : \text{out}(i \text{ after } \sigma) \subseteq \text{out}(s \text{ after } \sigma)$



● Automatisches Testen mit dem TorX Tool

Die ioco Theorie wird von dem Tool Torx implementiert:

<http://www.purl.org/net/torx/>




Das TorX Tool

The image displays two windows from the TorX 1.2.0 tool. The left window, titled "TorX 1.2.0: Config: conf.jan.prom", shows the configuration interface with buttons for "(Re)Start", "Stop", and "Kill", and a "Mode" dropdown set to "Manual". Below these are sections for "Path", "Current state offers:", "Inputs:", and "Verdict:". The "Inputs:" section lists various messages like "from_upper ! LEAVE ! var_byte ! var_byte" and "from_lower ! PDU_JOIN ! var_byte ! var_byte ! var_byte". The "Verdict:" section contains a log of system messages, including "IUT Stderr: Debug: cf_rt.c: Joining sender is not a partner!" and "IUT Stderr: Debug: mc_st.c: Sending ANSWER-pdu (21 bytes) to user 3".

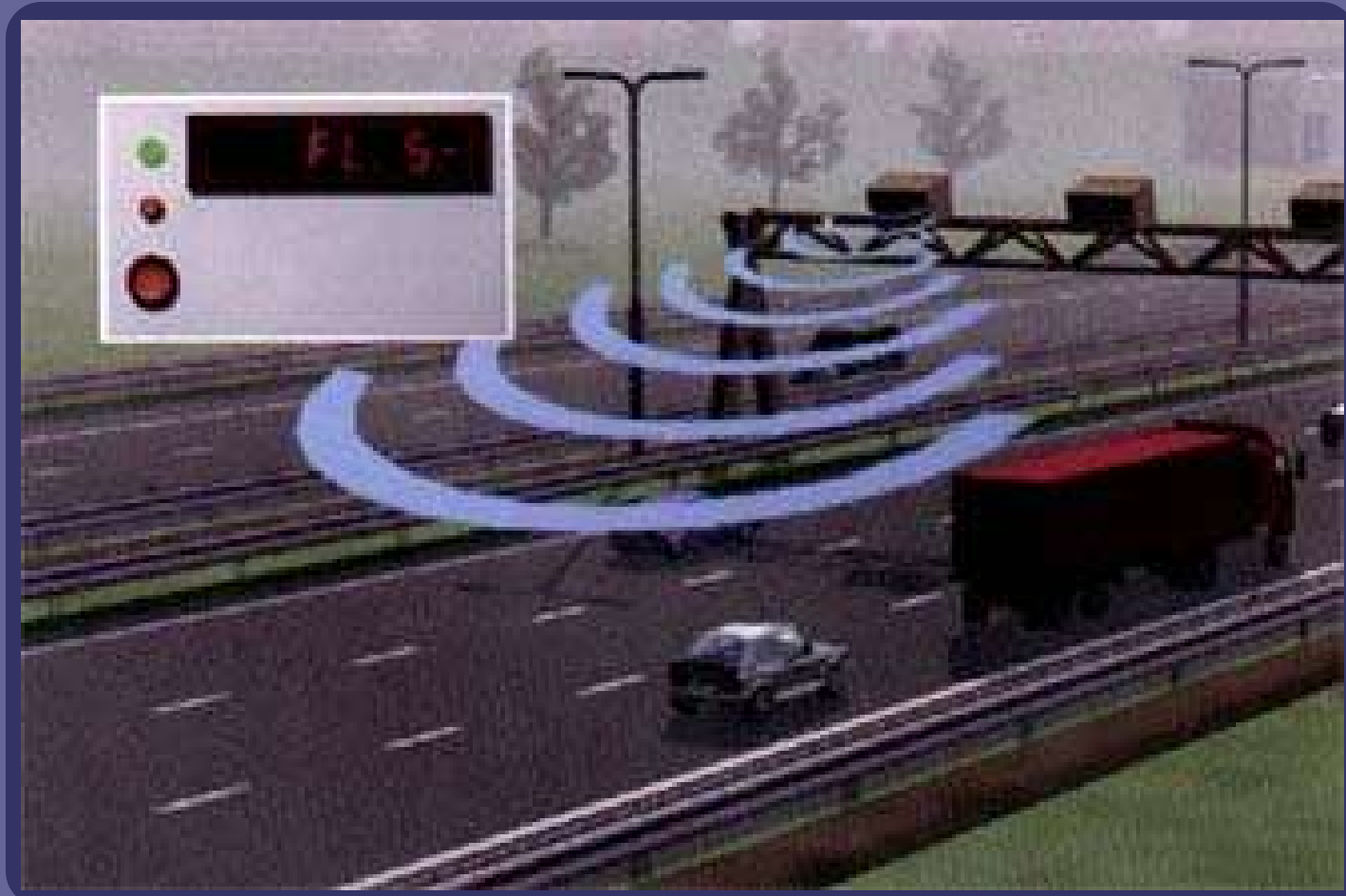
The right window, titled "Message Sequence Chart: conf.jan.prom", displays a sequence diagram with five lifelines: iut, udp2, udp0, and cf1. The diagram shows a series of messages and quiescence periods. Key messages include "from_lower ! PDU_JOIN ! 103 ! 51 ! 2 ! 1", "from_lower ! PDU_LEAVE ! 102 ! 52 ! 0 ! 1", "from_upper ! JOIN ! 102 ! 52", "from_lower ! PDU_DATA ! 21 ! 32 ! 2 ! 1", "to_lower ! PDU_JOIN ! 102 ! 52 ! 1 ! 2", "to_lower ! PDU_JOIN ! 102 ! 52 ! 1 ! 0", "from_lower ! PDU_DATA ! 21 ! 34 ! 0 ! 1", "to_lower ! PDU_JOIN ! 102 ! 52 ! 1 ! 2", "to_lower ! PDU_JOIN ! 102 ! 52 ! 1 ! 0", "from_upper ! DREQ ! 21 ! 31", "from_lower ! PDU_JOIN ! 103 ! 52 ! 2 ! 1", and "to_lower ! PDU_ANSWER ! 102 ! 52 ! 1 ! 2".



● Das TorX Tool – Fallstudien

- Conference Protocol academic
- EasyLink TV-VCR protocol Philips
- Cell Broadcast Centre component CMG
- “Rekeningrijden” Payment Box protocol 
- V5.1 Access Network protocol Lucent
- Easy Mail Melder CMG
- FTP Client academic
- “Oosterschelde” storm surge barrier-control CMG

● Interpay Mautsystem



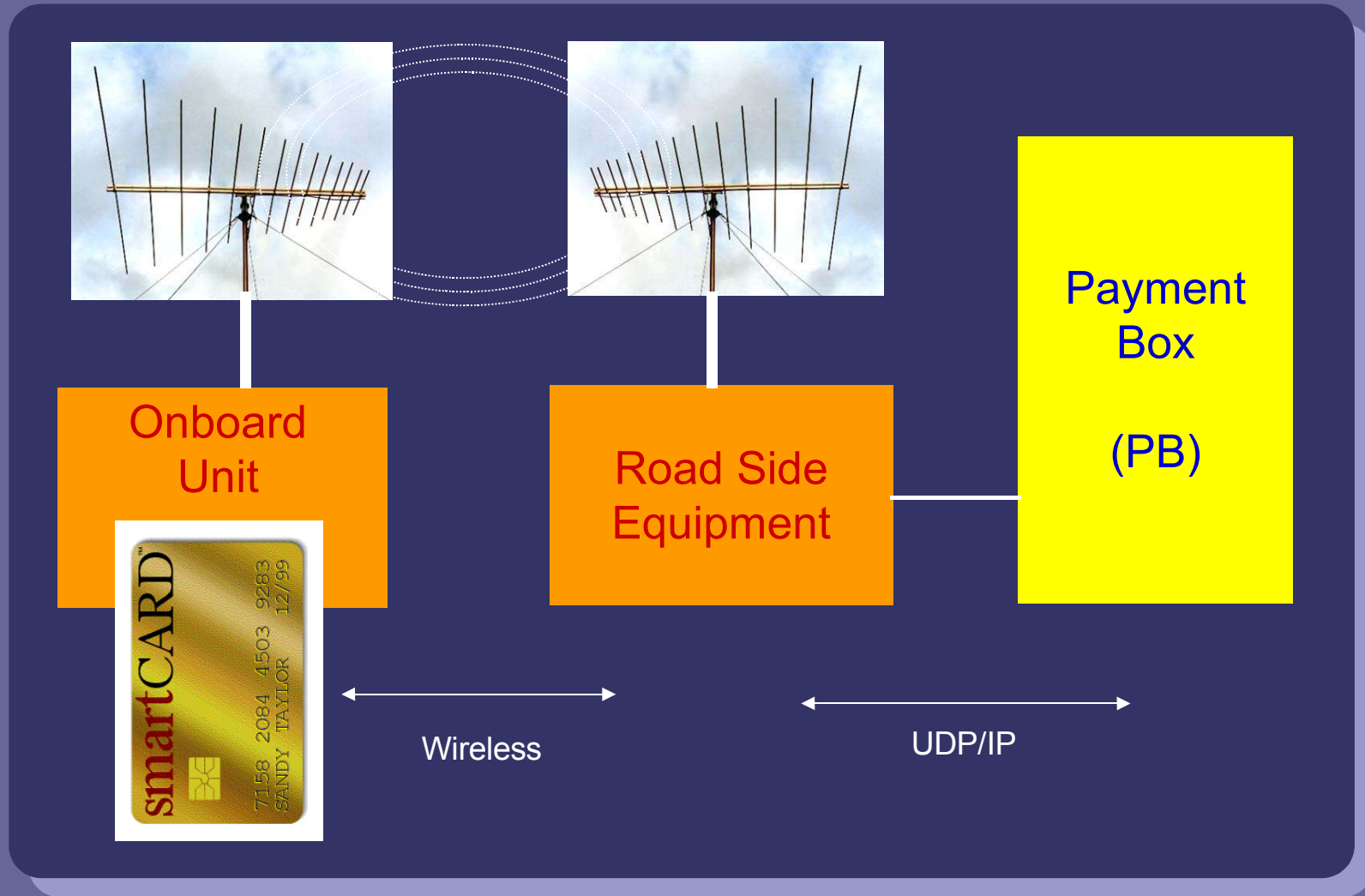
● Interpay Mautsystem

Merkmale:

- Einfaches Protokoll
- Parallelität: viele KFZe gleichzeitig
- Verschlüsselung
- Echtzeitaspekte
- System hatte klassische Tests erfolgreich passiert



● Interpay Mautsystem



● Interpay Mautsystem

Herausforderungen:

- Parallelismus:
 - kein Problem
- Verschlüsselung:
 - konnte nicht komplett umgesetzt werden
- Echtzeitaspekte:
 - konnten nicht umgesetzt werden
 - effiziente on-the-fly Implementation unklar
 - Theorie nicht vorhanden – Ruhezustand vs. time-out



● Interpay Mautsystem

Resultate:

- 1 Fehler gefunden während Verifikation (Designfehler)
- 1 Fehler gefunden durch Testen (Fehlerzustand)
- Automatisches Testen mit TorX:
 - von enormem Nutzen
 - hohe Testdichte und Zuverlässigkeit
 - viele, lange Tests generiert (>50.000 test events)
 - sehr flexibel adaptier- und konfigurierbar



● Ausblick

Aktuell wird die Theorie erweitert bezüglich:

- Symbolisches Testen
- Echtzeitbasiertes Testen
- Neue Fallstudien: ASML Wafer Stepper, Smart Cards



● Literatur

- Broy, Jonsson, Katoen, Leucker, Pretschner (Eds.):
Model-based Testing of Reactive Systems -
A seminar volume
LNCS, to appear in 2004
- Brinksma, Tretmans:
Testing Transition Systems:
An Annotated Bibliography
LNCS 2067:187-195, 2001
- Lee, Yannakakis:
Principles and Methods of Testing
Finite State Machines - a Survey
Proc. IEEE, 48(8):1090-1126, 1996



● Vielen Dank!

