Model Driven Testing

Prof. Dr. Holger Schlingloff

Fraunhofer FIRST

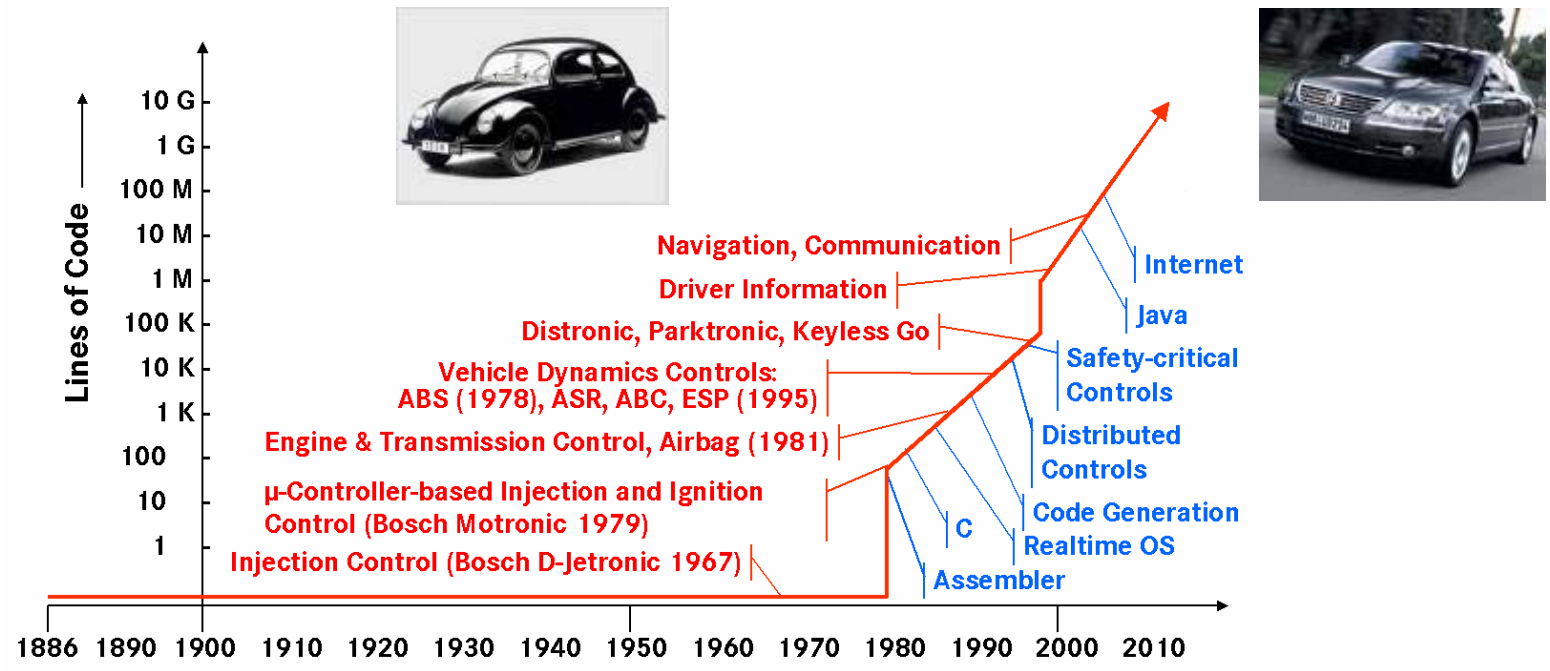
und

Humboldt-Universität zu Berlin

http://www.informatik.hu-berlin.de/~hs

Ausgangssituation

- Automobilindustrie ist wesentliche deutsche Innovationsbranche
- Realisierung durch Software im Fahrzeug
  - Optimierung und Ressourcenmanagement
  - Fahrerassistenz- und Insassenkomfortsysteme



Quelle: DaimlerChrysler AG

FIRST
Fraunhofer Institut
Rechnerarchitektur
und Softwaretechnik

Software im Fahrzeug

- Vorteile
  - Sicherheit und Zuverlässigkeit
  - erhöhte Flexibilität
  - Kostenreduktion
  - Endkunden-Mehrwert
- Stand der Technik
  - 40-80 Steuergeräte
  - 500K-2M Programmzeilen
  - 3-5 verschiedene Bussysteme
- Besonderes Verhältnis Hersteller und Zulieferindustrie
  - Spezifikation und Integration bei Herstellern
  - Entwicklung bei Zulieferern

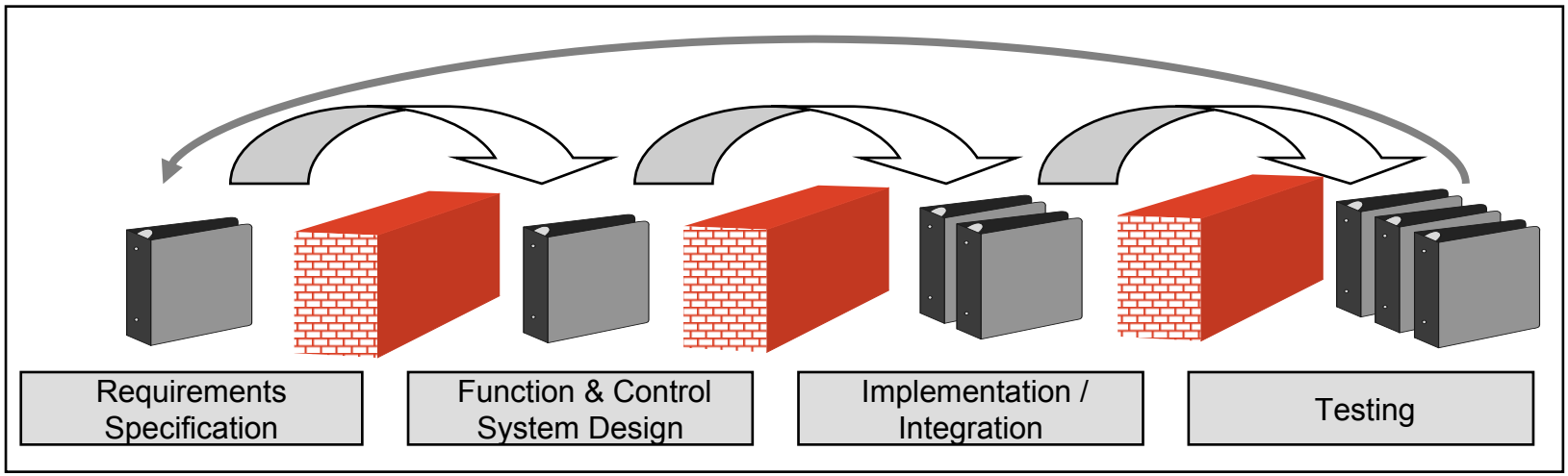
Quelle: Volkswagen AG

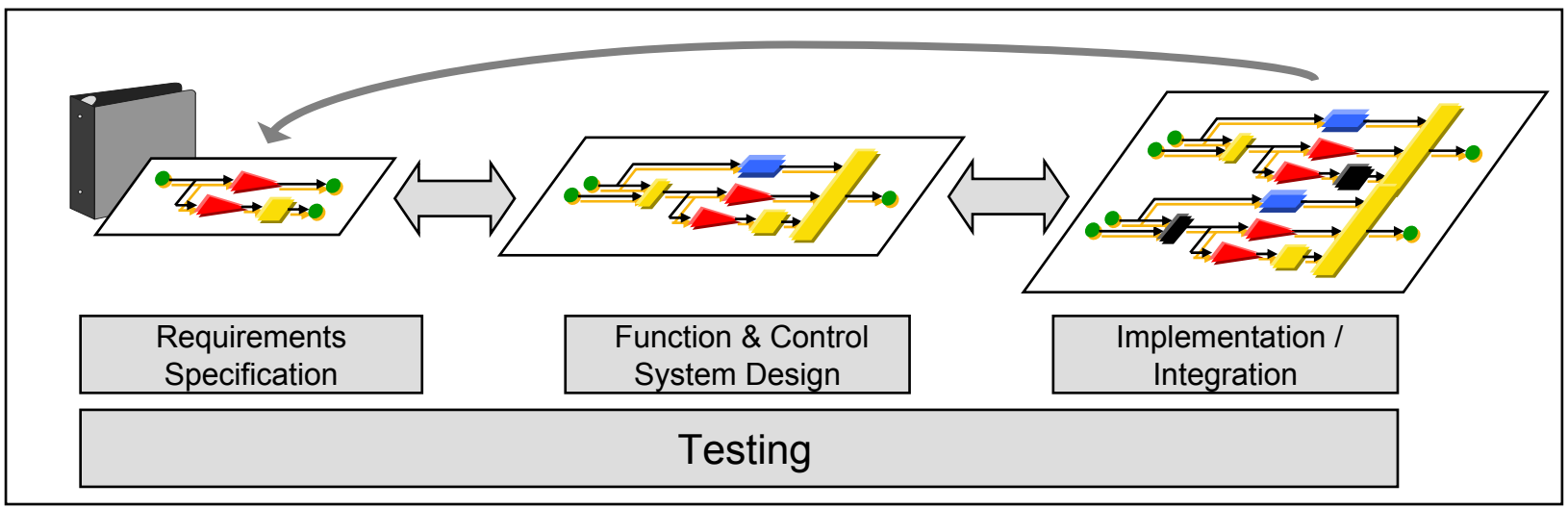
Probleme

- Größen- und Komplexitätszuwachs der Software
  - nicht Hardware, sondern Software und Qualitätssicherung bestimmen die Grenze des Machbaren
- schnelle Zyklenfolge, kurze Entwicklungszeiten
  - von 50 Monaten (1980) auf 18-20 Monate (heute)
- Variantenvielfalt
  - Wiederverwendung, Konsistenz, Interoperabilität
- Spezifikation, Kontrakt Hersteller / Zulieferer
  - keine einheitlichen Beschreibungsmittel
  - Qualitäts- und Verbindlichkeitsproblematik

FIRST
Fraunhofer Institut
Rechnerarchitektur
und Softwaretechnik

Paradigmenwechsel in der Softwareentwicklung

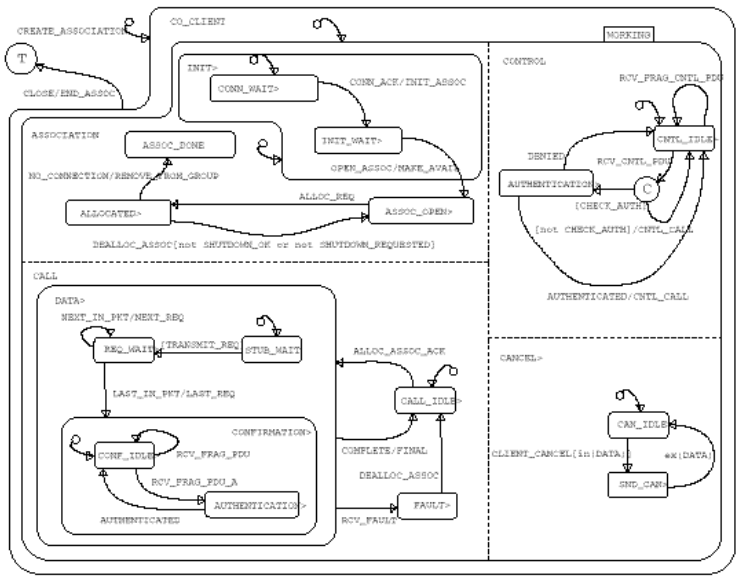




FIRST
Fraunhofer Institut Rechnerarchitektur und Softwaretechnik

modellbasierte Entwicklung

- frühzeitige Erstellung eines ausführbaren Modells (Systemmodell) auf Grundlage der Anforderungen
- Simulation und Erprobung auf Modellebene
- Verfeinerung zum Implementierungs-modell
- automatische Code-generierung für Host und Target

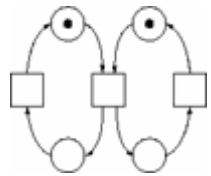


- Effizienz- und Qualitätsgewinn!

FIRST
Fraunhofer Institut
Rechnerarchitektur
und Softwaretechnik

Modellierungsformalismen

- **herkömmliche Formalismen**: Zustandsmaschinen, Diagramme, Petrinetze, StateCharts, Message Sequence Charts, ...
  - gut etabliert, Toolunterstützung vorhanden
  - Wildwuchs, Varianten, mangelnde Konstanz
  - Prognose: werden durch UML2.0 abgelöst werden



- **UML 2.0**
  - vereinigt verschiedene Arten von Diagrammen
  - extrem mächtig, sehr umfangreich zu lernen
  - fehlende Semantik, verschiedene Ausbaustufen

UNIFIED MODELING LANGUAGE™

- **Matlab / Simulink / Stateflow**
  - teilweise gut eingeführt und bekannt
  - limitierte Ausdrucksfähigkeit
  - nur eine Quelle

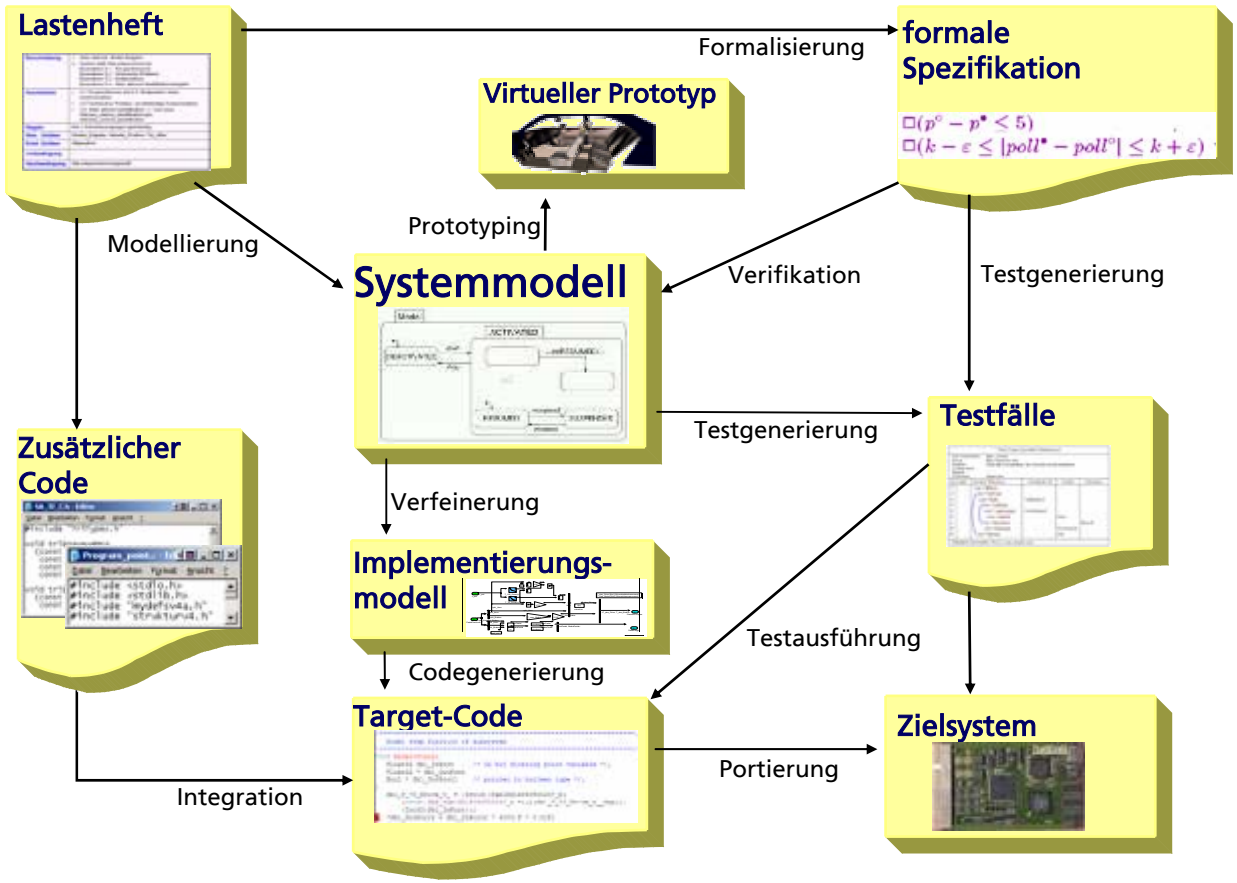
The MathWorks

- **logische und algebraische Modelle**, z.B. TLA, ASM/ASML, CSP
  - nahe an natürlichsprachlichen Formulierungen
  - Forschungsbedarf

SVT

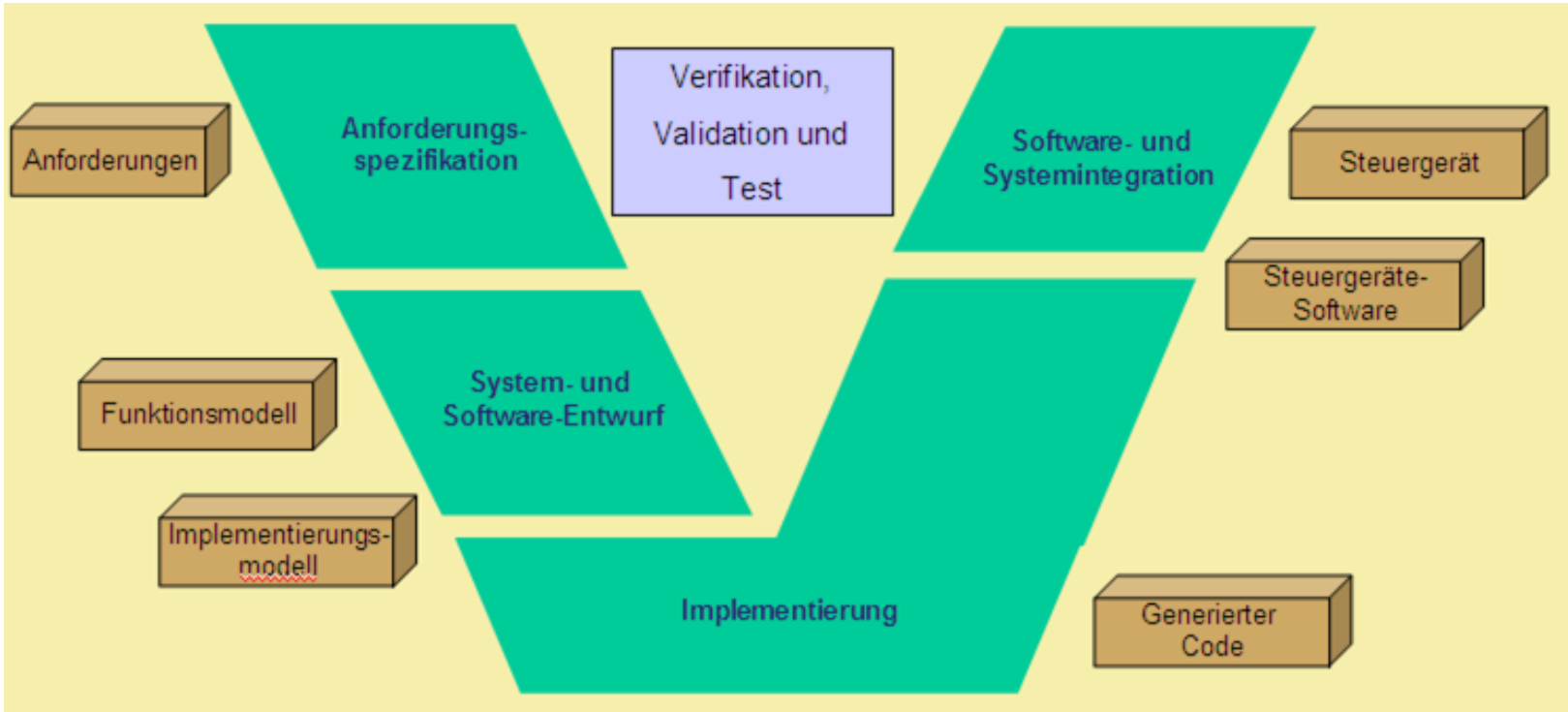
FIRST
Fraunhofer Institut Rechnerarchitektur und Softwaretechnik

Aktivitäten und Artefakte



Fraunhofer Institut Rechnerarchitektur und Softwaretechnik

modellbasierter Entwurfsprozess

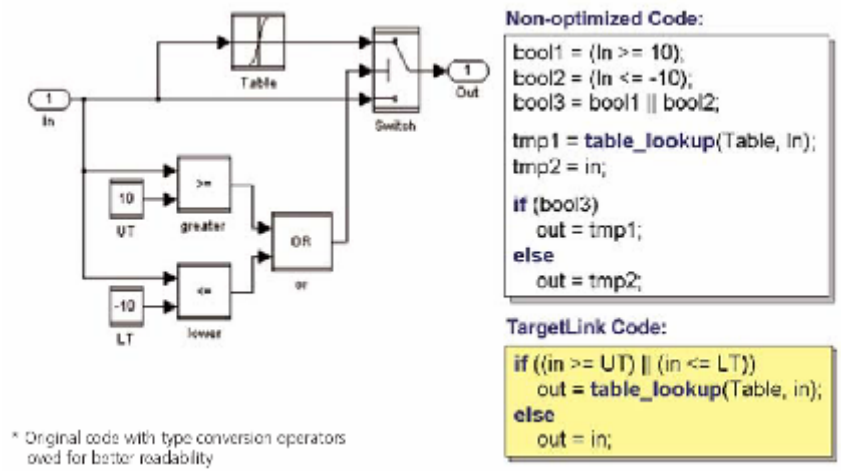


- neue Herangehensweisen erforderlich für
  a) die Qualitätssicherung automatisch erzeugten Codes
  b) die modellgetriebene Testgenerierung
  c) kohärente Testspezifikationssprachen

FIRST
Fraunhofer Institut Rechnerarchitektur und Softwaretechnik

(a) automatische Codegenerierung

- Codegenerator ist „Compiler für Modelle“
  - Wiederverwendung von Modellen
  - schnelle Prototyp- und Produkterstellung
  - erhöhte Zuverlässigkeit gegen Programmierfehler
  - automatische Optimierung des generierten Codes



Quelle: dSPACE GmbH

- gängige Codegeneratoren
  - MathWorks® Real-Time Workshop, dSPACE® TargetLink

Absicherung automatisch generierten Codes?

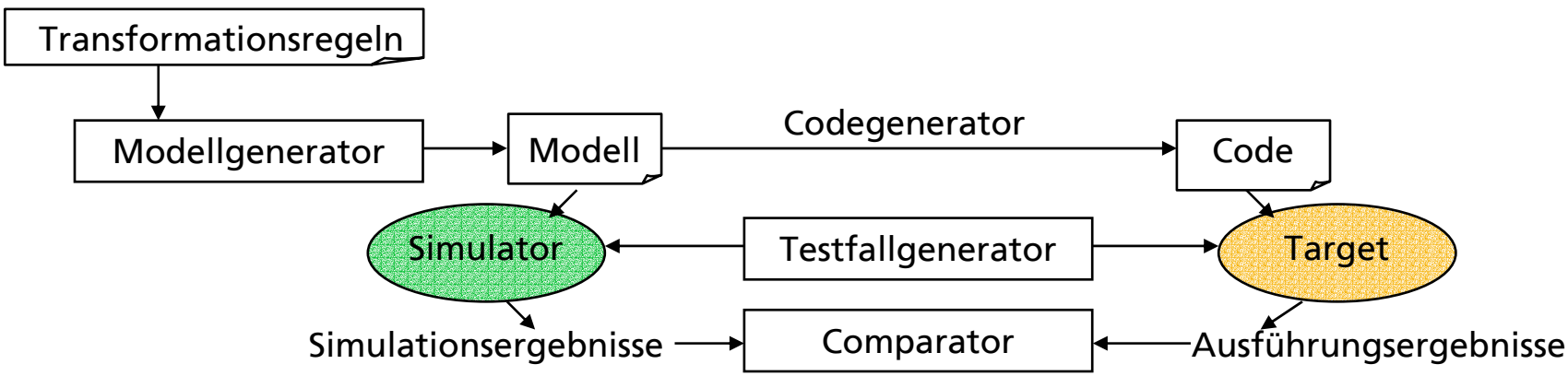
- erübrigt sich unter folgenden Annahmen
  - erfolgreiche Modellprüfung
  - vollautomatische Codegenerierung
  - betriebsbewährter Codegenerator
  - zuverlässige Hard- und Middleware
- leider trifft heute keine dieser Annahmen zu!

- Lösungsansätze
  - Modelchecking von formalisierten Requirements
  - Richtlinien für Design und Review generierten Codes
  - Validation von Codegeneratoren
  - automatische Hardware-in-the-Loop-Tests

FIRST
Fraunhofer Institut
Rechnerarchitektur
und Softwaretechnik

Absicherung von Codegeneratoren

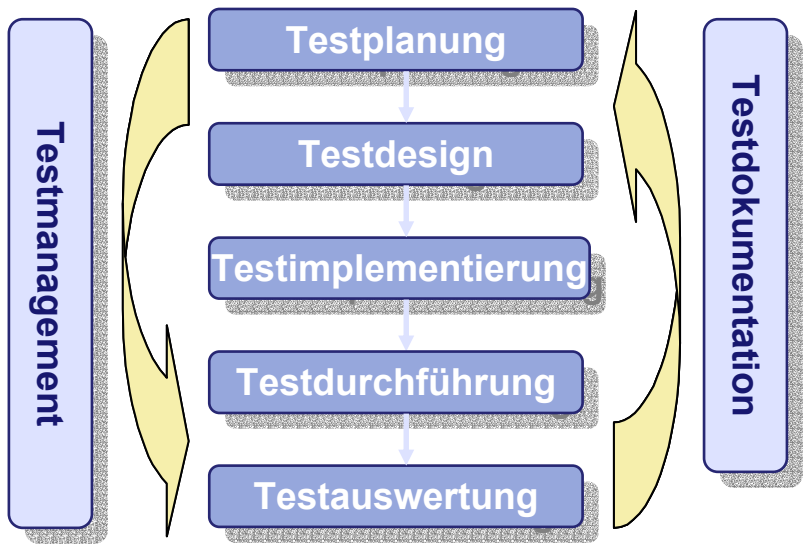
- Probleme
  - häufige Generationenfolge von Prozessoren und Codegeneratoren
  - Notwendigkeit zusätzlicher Absicherung
- Forschungsansatz
  - Validationssuite für Codegeneratoren
  - Graph-Transformationen als logische Basis



FIRST
Fraunhofer Institut
Rechnerarchitektur
und Softwaretechnik

(b) modellgetriebener Testprozess

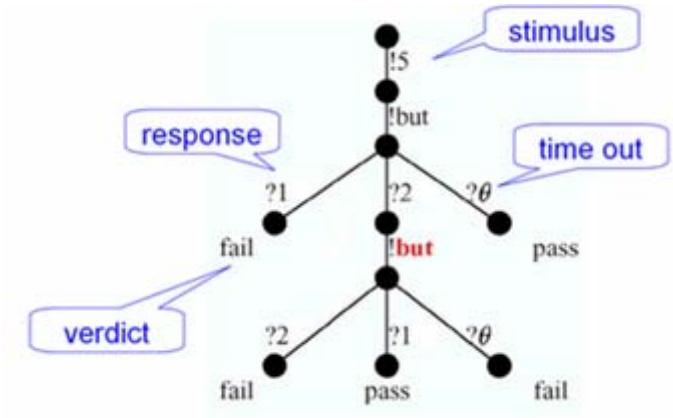
- verschiedene, gut etablierte Testmethoden
  - Überdeckungsmaße
  - Reaktive Black-box Tests
  - Klassifikationsbaum-Methode
  - evolutionäre Testentwicklung
- (auch) auf Modelle anwendbar!
- automatische Testfallerzeugung
  - z.B. ILogix® ATG

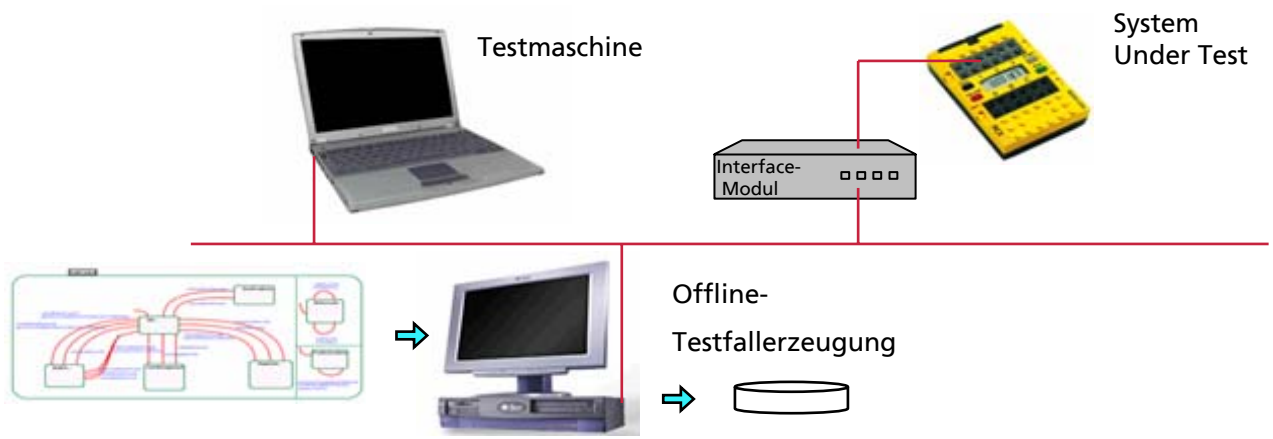


Fraunhofer FIRST Institut
Rechnerarchitektur
und Softwaretechnik

Testgenerierung aus Modellen

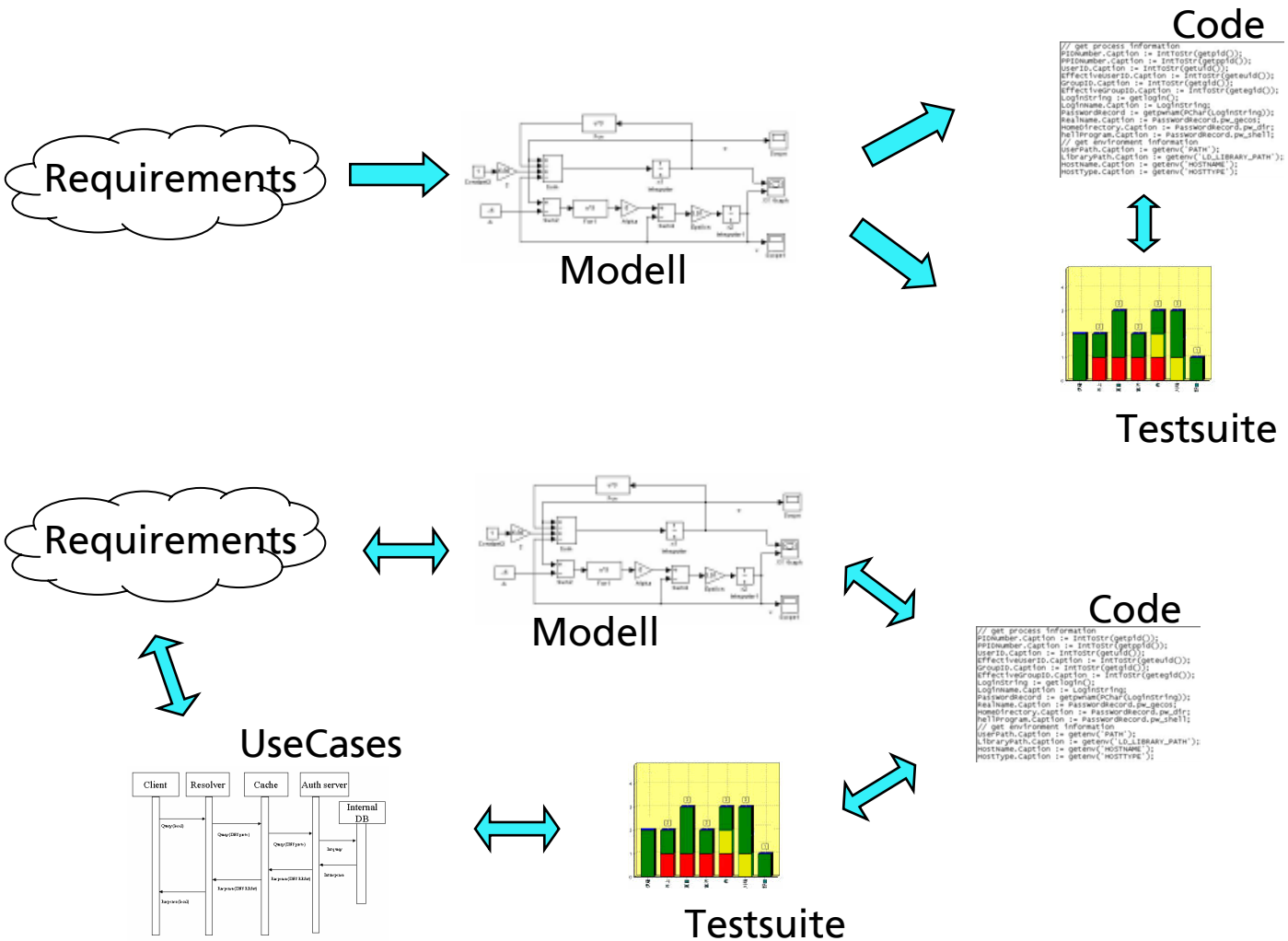
- Modellierung in UML-StateCharts oder StateFlow-Diagrammen
- Kompakte Repräsentation des Zustandsraumes, Generierung einer „transition tour“
- Testausführungs-Engine für HIL-Target





Fraunhofer Institut
Rechnerarchitektur
und Softwaretechnik

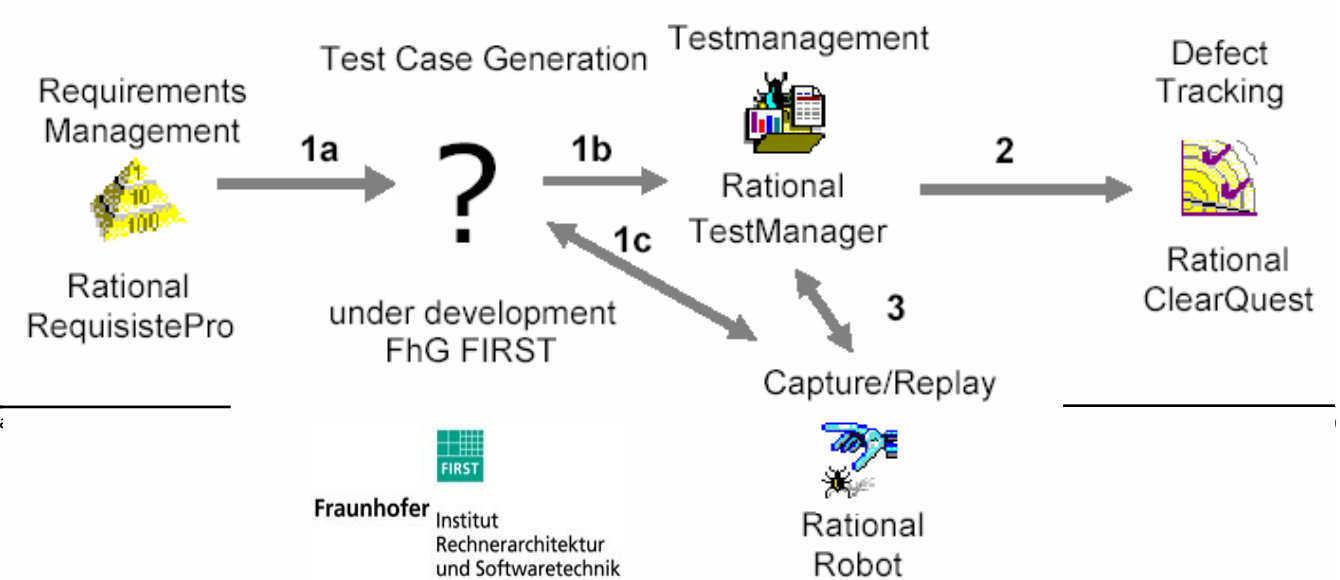
Szenarien für Testautomatisierung



Fraunhofer Institut Rechnerarchitektur und Softwaretechnik

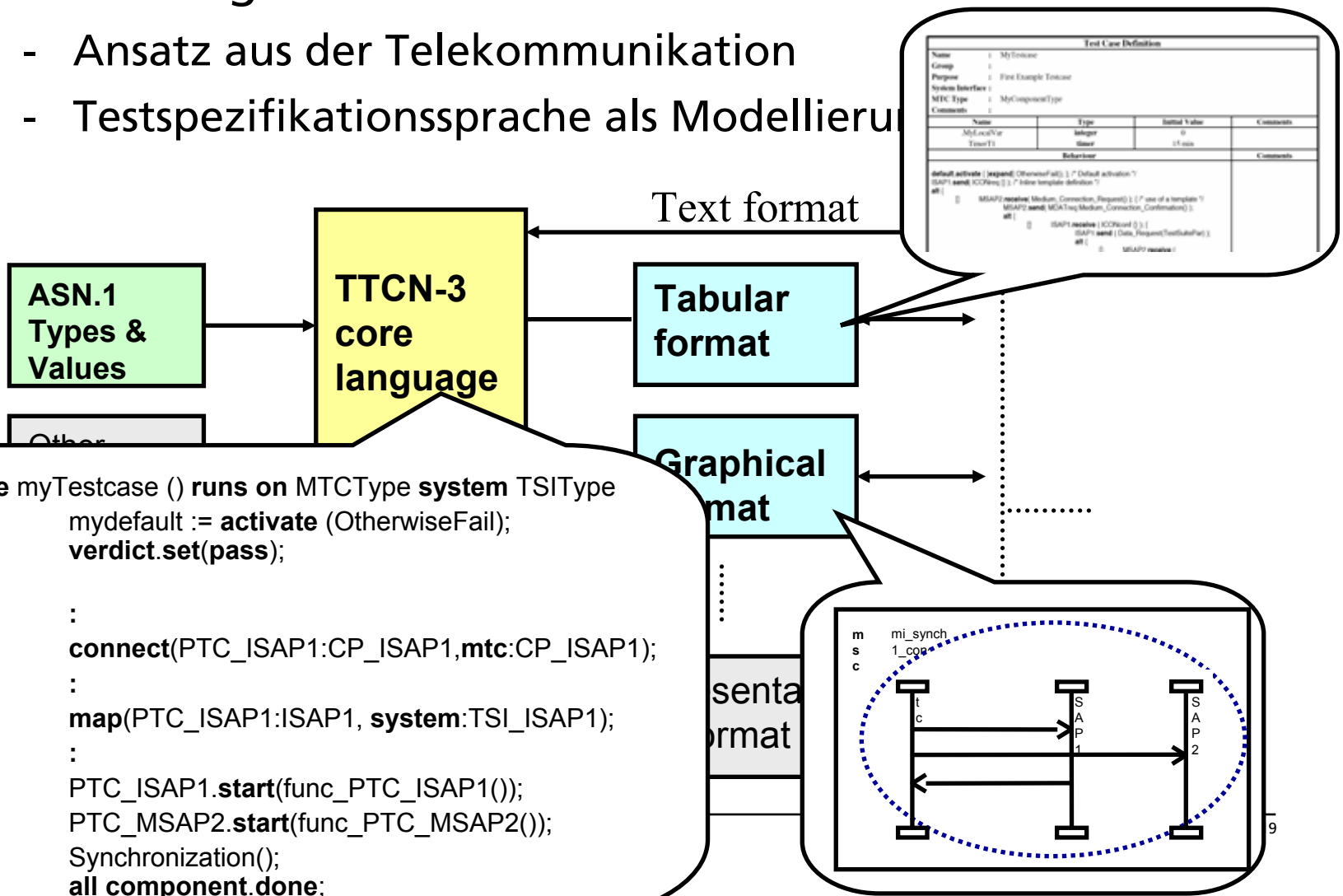
(c) Kohärenz von modellbasierten Tests

- Probleme im modellbasierten Ansatz
  - unterschiedliche Formen der Beschreibungen von Testfällen, Kompatibiltäts- und Konsistenzprobleme
  - Mischsysteme aus Modellen und manuellem Code
- Lösungsansätze
  - syntaktische und semantische Überführung unterschiedlicher Formate, einheitliche Notation
  - kohärente Testumgebung, Toolintegration (Pro2Lab)



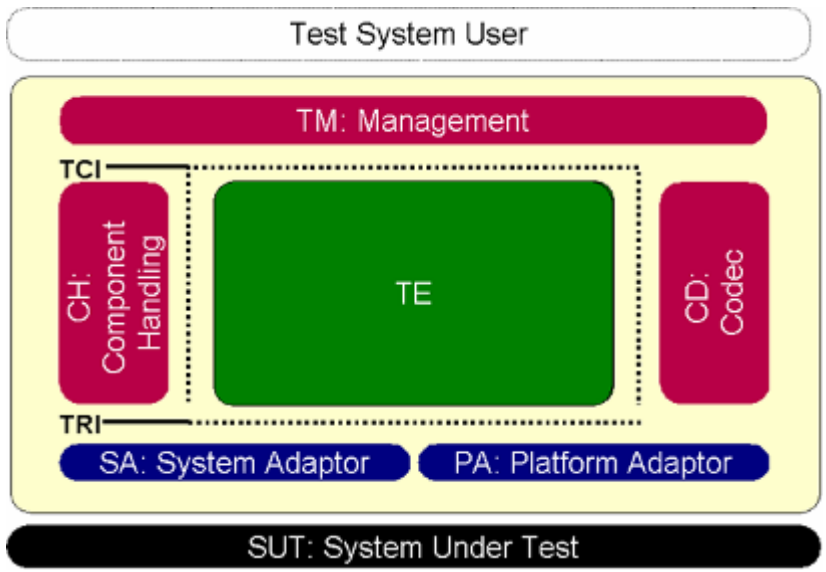
Testspezifikationssprachen am Beispiel TTCN-3

- Notwendigkeit einer einheitlichen Notation
  - Ansatz aus der Telekommunikation
  - Testspezifikationssprache als Modellieru



Testausführungsumgebung

- TTCN-3 beginnt, sich auch außerhalb der Telekommunikation als Test-Spezifikationssprache durchzusetzen:
  - Telelogic Tau, Danet, Testing Tech, daVinci Comm, openttcn, …
  - TTCN-3 für embedded, TTCN-3.Net (FIRST)
- UML Testing Profile
  - Metamodell, TTCN-3 als Instanz



FIRST
Fraunhofer Institut Rechnerarchitektur und Softwaretechnik

Zusammenfassung

- modellbasierte Entwicklung hat hohes Potential zur Produktivitätssteigerung in der Softwaretechnik

- Modell als zentraler Dreh- und Angelpunkt für Systementwicklung
  a) automatische Codegenerierung bedingt neue Absicherungsverfahren für Code und Generator
  b) Erzeugung von Tests aus Modellen und Use-Case-Anwendungsfällen bereits heute möglich und sinnvoll
  c) standardisierte Testspezifikationssprachen entstehen

- Vielen Dank für Ihre Aufmerksamkeit!

FIRST
Fraunhofer Institut
Rechnerarchitektur
und Softwaretechnik