

# Validierungsmöglichkeiten von Modellen



**Sven Feja**  
**Christian-Albrechts-Universität Kiel**

GEFÖRDERT VOM



Bundesministerium  
für Bildung  
und Forschung

PROJEKTRÄGER



DLR

## 1 Motivation

## 2 Modellprüfer

### 2.1 Kripke-Struktur

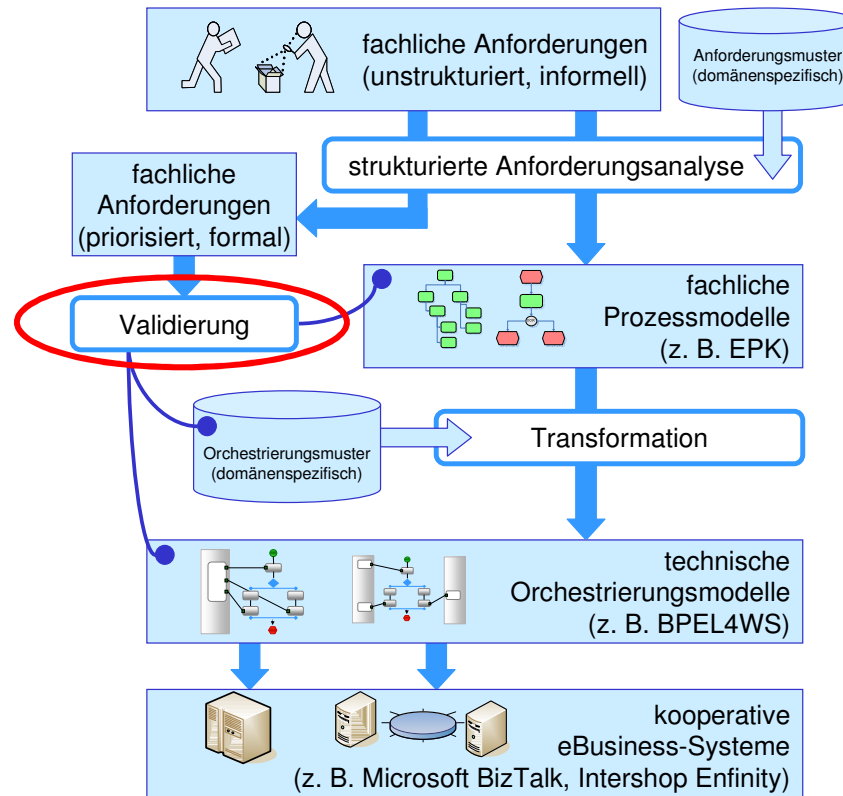
### 2.2 Temporale Logik

### 2.3 Beispiele

## 3 Erweiterungen des Grundprinzips der Modellprüfung

## 4 Fragen

## Einordnung in das Projekt OrViA:

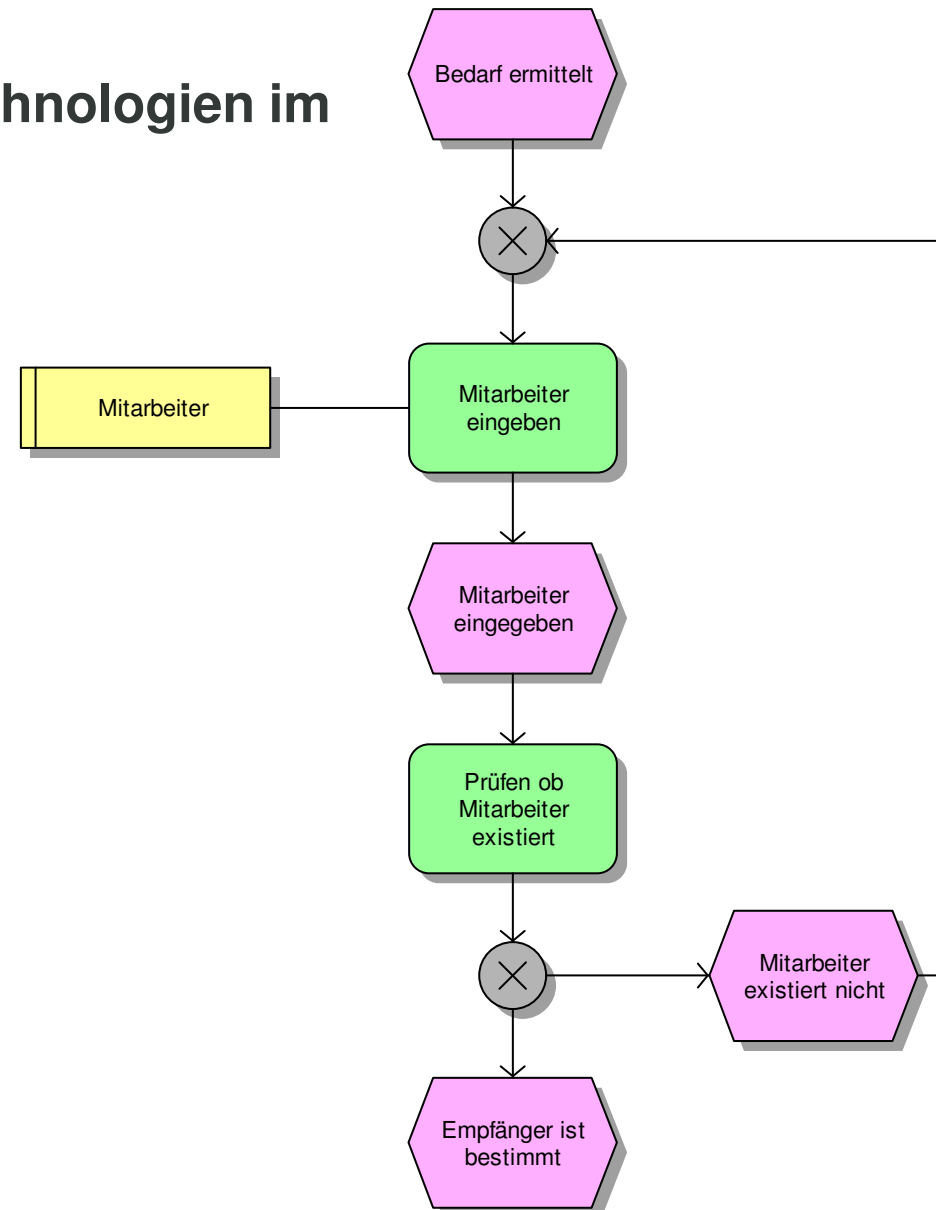


- **Abgleich der Spezifikation mit den erstellten Prozess-Modellen im Rahmen des Softwareentwicklungsprozesses**
  
- **Probleme:**
  - ▶ Spezifikation wird nicht formal aufgenommen und somit nur manueller, subjektiver Abgleich möglich
  - ▶ Erstellung einer formalen Spezifikation weder für Consultant noch Kunden sinnvoll realisierbar, da:
    - ▶▶ **temporale Logiken zu „mathematisch“ für Consultant und Kunden**
  
- **Ziel: (teil-)automatisierter Abgleich einer formalen Spezifikation mit den erstellten Prozess-Modellen**

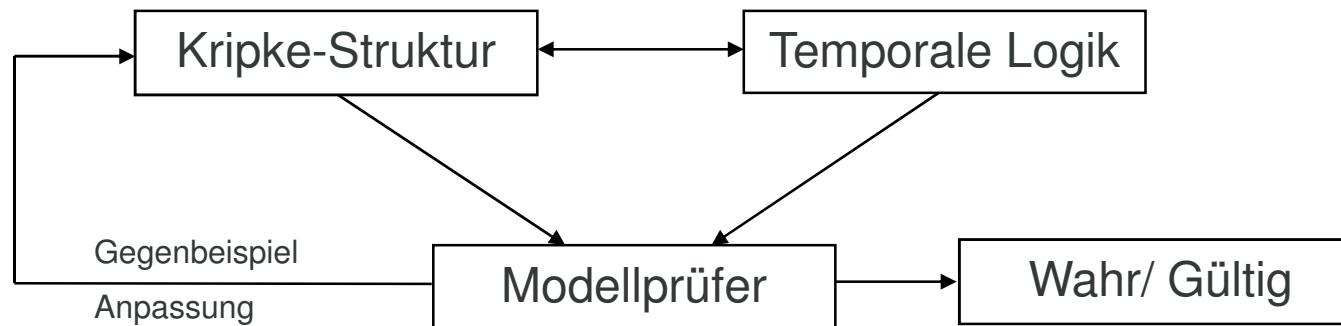
- Einsatz von Modellprüfungstechnologien im Softwareentwicklungsprozess

- Beispiel anhand EPK-Prozess:

- ▶ Überprüfung, ob für einen Bedarf ein Empfänger vorhanden ist

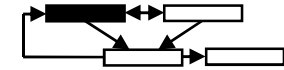


### ■ Grundprinzip der Modellprüfung:



- ▶ Beispiele für Modellprüfer: SMV, SPIN,...
- ▶ Beispiele für temporale Logik: CTL, LTL,...

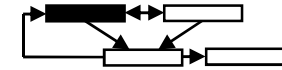
### ■ Formalismus zur Beschreibung von Systemmodellen



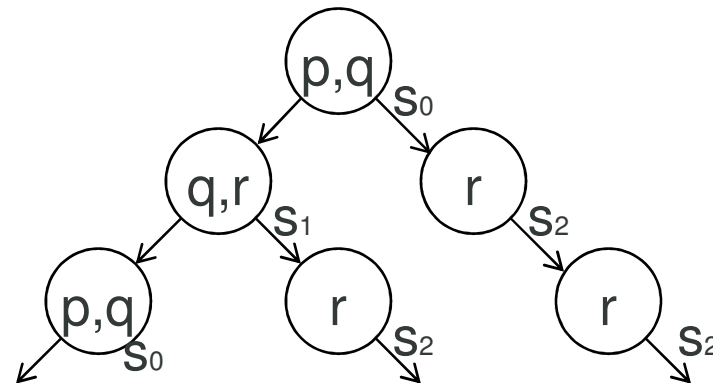
### ■ $M=(S,S_0,R,L)$

- ▶  $S$  – Menge von Zuständen
- ▶  $S_0 \subseteq S$  – Menge von Startzuständen
- ▶  $R \subseteq S \times S$  ist eine totale Transitionsrelation (für jeden Zustand  $s \in S$  existiert ein Folgezustand  $s'$  in  $S$  mit  $R(s,s')$ )
- ▶  $L:S \rightarrow A$  ist eine Beschriftungsfunktion, die jeden Zustand auf eine Menge  $A$  von atomaren Aussagen (=Eigenschaften) abbildet

- Beschreibung eines Zustandsübergangssystems

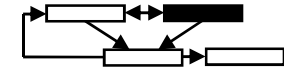


- Daraus ergeben sich die Berechnungspfade im folgenden Baum:



- ▶ Pfade besitzen unendliche Länge

### CTL (Computational Tree Logic)



#### ■ Boolesche Operatoren:

$\Phi ::= \text{false} \mid \text{true} \mid p \mid (\neg\Phi) \mid (\Phi \vee \Psi) \mid (\Phi \wedge \Psi) \mid (\Phi \rightarrow \Psi)$

#### ■ zusätzliche temporale Operatoren:

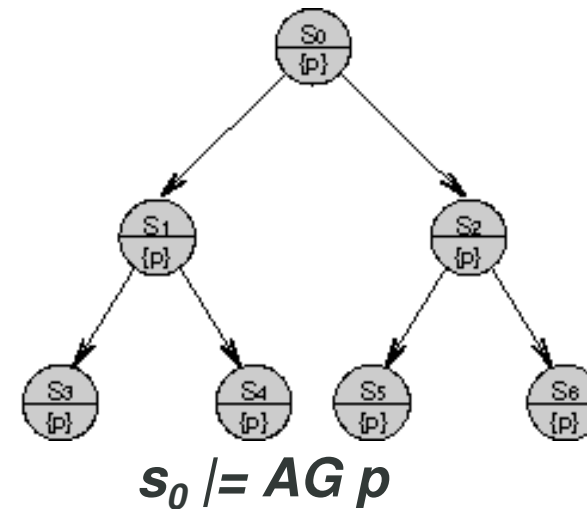
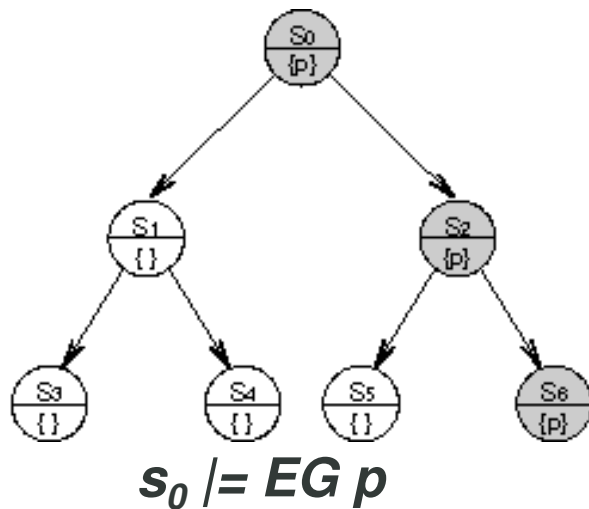
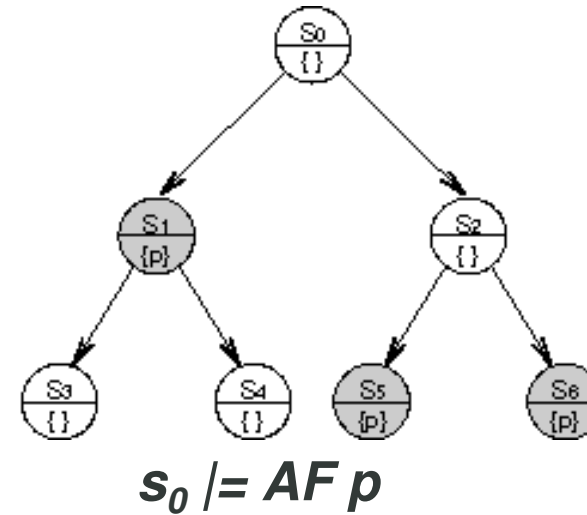
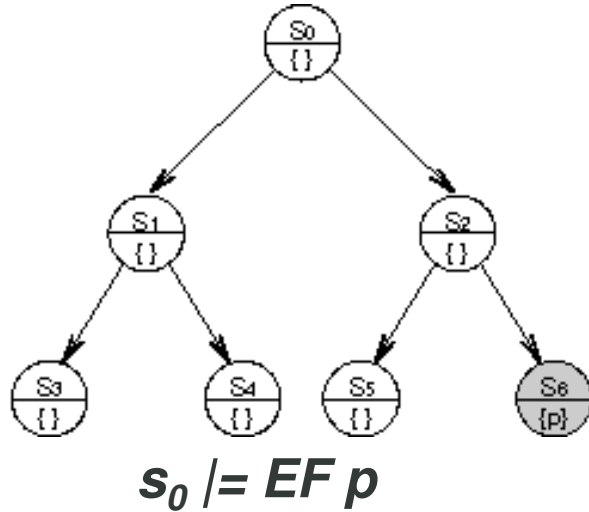
immer paarweise Anordnung *Temporal Connectives*

$\Phi ::= AX\Phi \mid EX\Phi \mid A(\Phi U \Psi) \mid E(\Phi U \Psi) \mid AG\Phi \mid EG\Phi \mid AF\Phi \mid EF\Phi$

**A** : Always , **E** : Exists

**X** : neXt , **U** : Until , **G** : Globally , **F** : Future

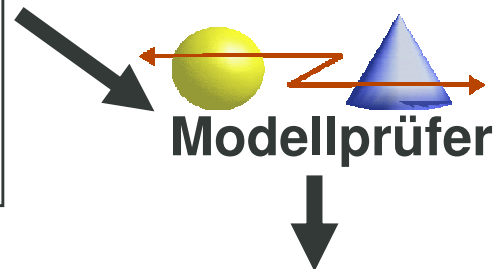
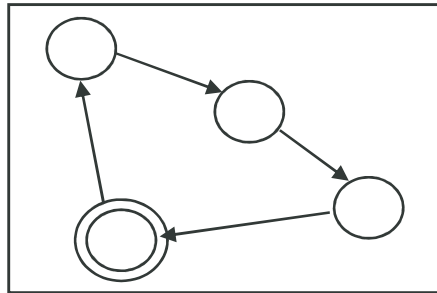
### ■ CTL (Computational Tree Logic)



## 2.3 Beispiele

OrVIA

Geschäftsprozess  
modelliert als Automat



AG (Kunde\_ist\_auf\_Homepage\_E  
-> AF (Katalogauswahl\_F))

zu überprüfende Spezifikation

-- specification AG (Kunde\_ist\_auf\_Homepage\_E -> AF Katal... is true

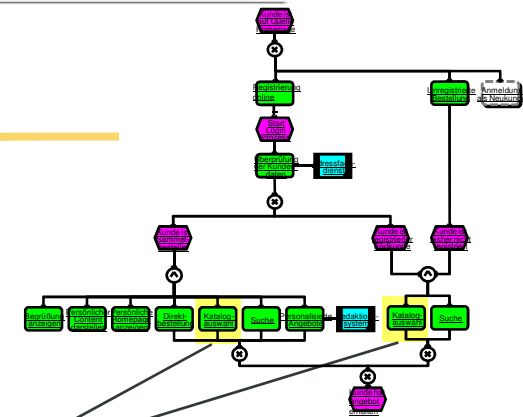
resources used:

user time: 0.06 s, system time: 0.03 s

BDD nodes allocated: 3119

Bytes allocated: 1245184

BDD nodes representing transition relation: 595 + 4

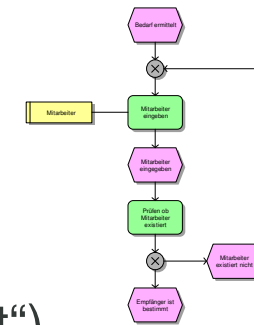


- Für den EPK-Beispiel-Prozess müsste folgende formale Aussage formuliert werden:

- ▶ Ist für den Bedarf ein Empfänger vorhanden?

entspricht:

„Bedarf\_ermittelt“  $\rightarrow$  EF(„Empfänger\_ist\_bestimmt“)

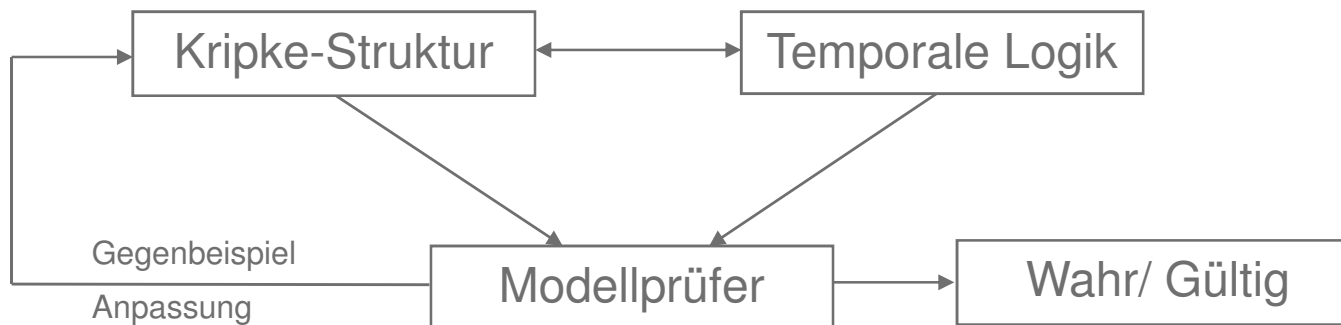
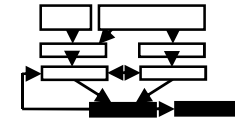


- Dies ist ein einfaches Beispiel, jedoch können diese Formeln schnell komplex werden.

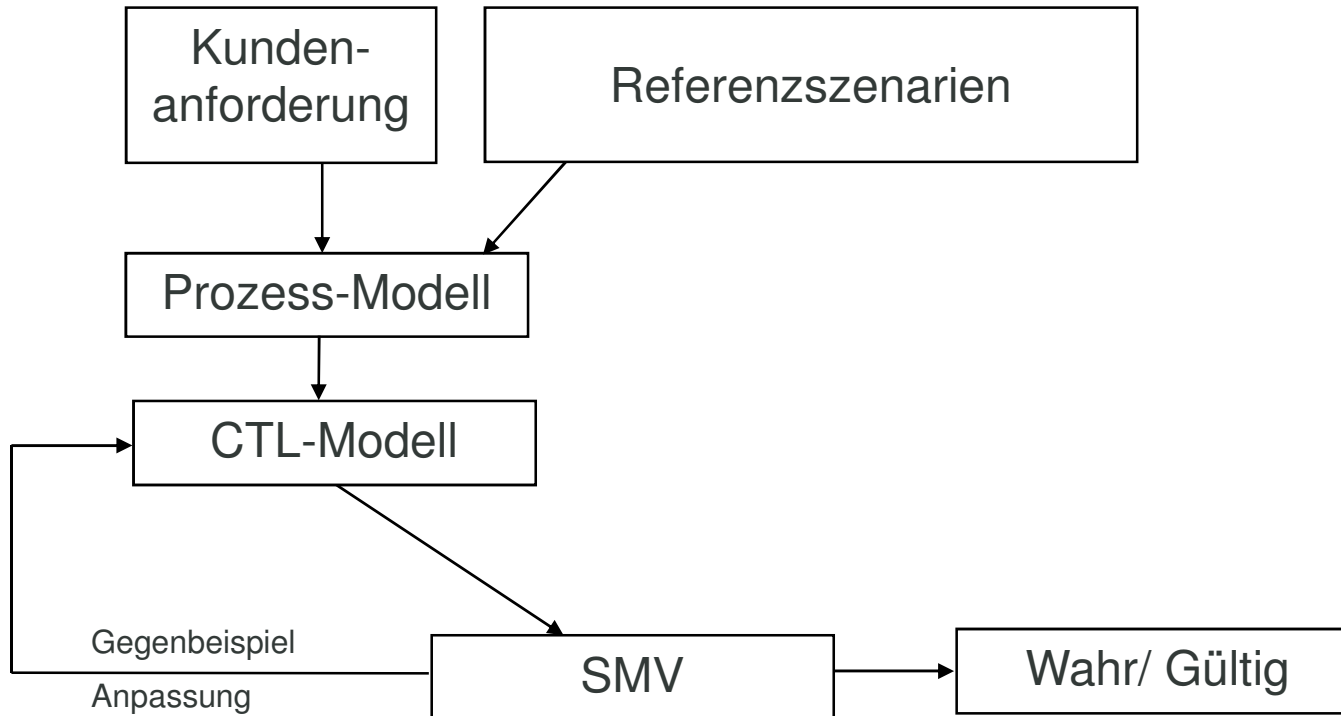
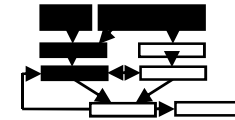
- ▶ Wie kann man CTL-Formeln leichter erstellen?
- ▶ Gibt es allgemeingültige CTL-Formeln?

### 3 Erweiterung des Grundprinzips der Modellprüfung

- ▶ Verwendeter Modellprüfer: SMV
- ▶ Verwendete temporale Logik: CTL

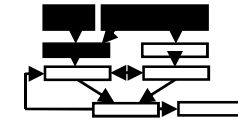


### 3 Erweiterung des Grundprinzips der Modellprüfung

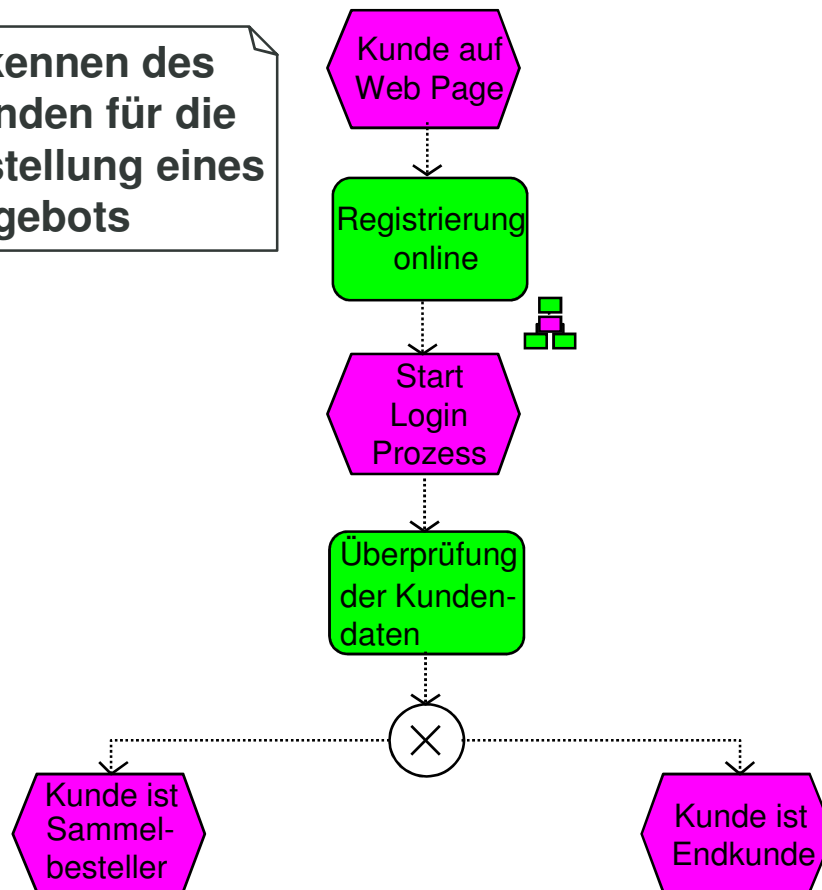




#### ■ Kundenanforderungen und Referenzszenarien:



Erkennen des Kunden für die Erstellung eines Angebots



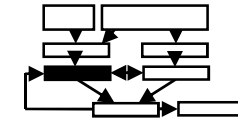
**Beispiel auf abstrakter Ebene:**  
Bestimmung allgemeiner Musterprozesse;  
Anpassung und Erweiterung entsprechend den Anforderungen der Kunden

Problemquellen:

- Einfügen von Funktionen und Ereignissen, die komplette Neuimplementierungen erfordern
- Einfügen von Funktionen und Ereignissen an falschen Stellen
- Veränderung des Ablaufs

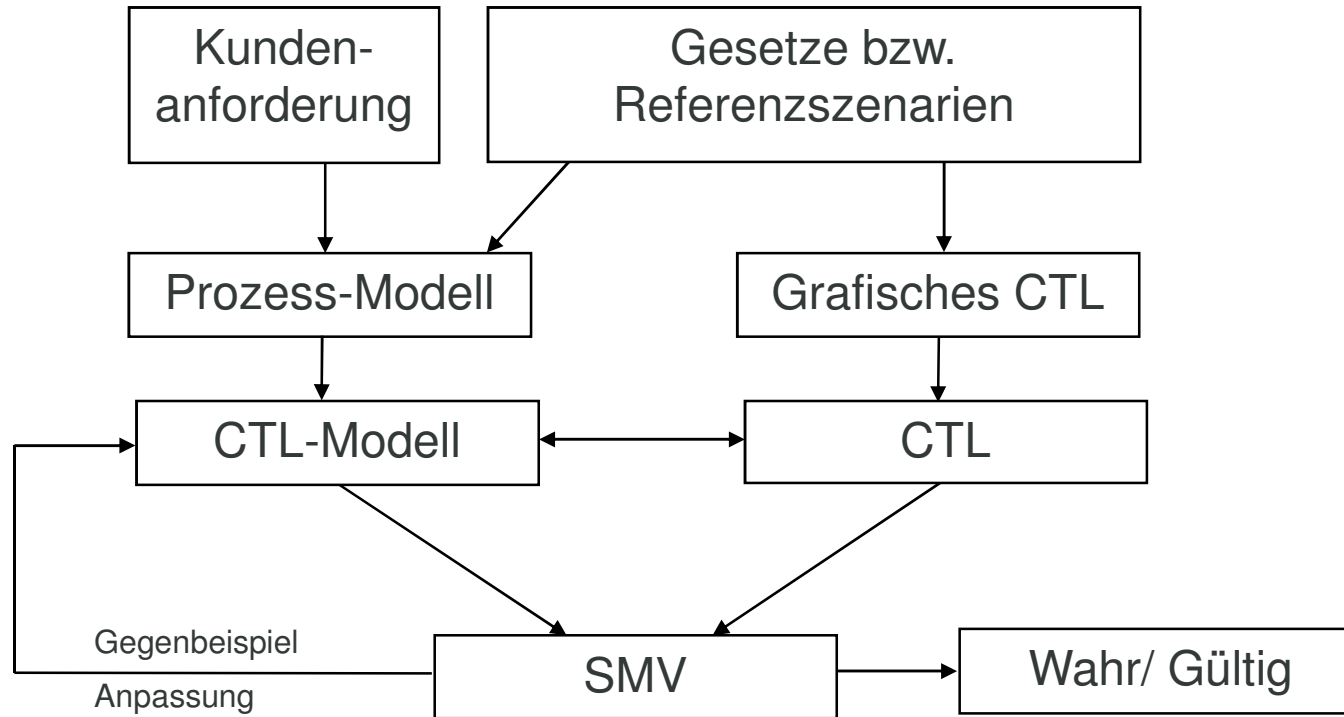
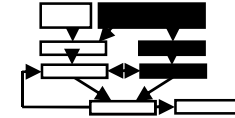
Korrekte Regeln und Erfahrungswerte werden als Referenzen gespeichert (*Best Practices*)

#### ■ CTL-Modell:

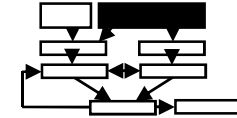


- ▶ CTL-Modell als XML-Darstellung zwischen Prozess-Modell und Kripke-Struktur
- ▶▶ **Austauschbarkeit des Modellprüfers soll dadurch erreicht werden**

### 3 Erweiterung des Grundprinzips der Modellprüfung

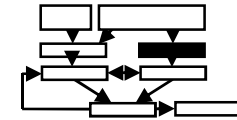


#### ■ Gesetze und Referenzszenarien:



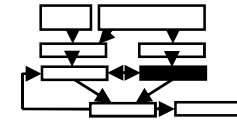
- ▶ Zu den Referenzszenarien existieren vordefinierte Regeln im grafischen CTL
- ▶ Aus Gesetzen sind ebenfalls Regeln im grafischen CTL erstellbar
- ▶ Die Anpassung der grafischen CTL-Regeln ermöglicht die Berücksichtigung von unternehmensspezifisch angepassten Modellen bzw. Prozessen

#### ■ Grafisches CTL:



- ▶ Es wird ein intuitives und „bekanntes“ Prinzip der Erstellung von CTL-Formeln benötigt.
  - ▶ **Für neue Prozesse sollen die Validierungsformeln grafisch erstellt werden können.**
  - ▶ **Durch die grafische Darstellung des CTLs soll eine leichtere Anpassung der Validierungsformeln ermöglicht werden.**

#### ■ CTL:



- ▶ CTL als XML-Darstellung zwischen dem grafischen CTL und der temporalen Logik (CTL im SMV-Format)
  - ▶▶ **Austauschbarkeit des Modellprüfers**

#### ■ **Ablauf der Validierung unter Verwendung von Referenzprozessen und dafür vorhandene Validierungsformeln**

Erstellung des Prozesses aus

##### ▶ **Referenzprozess**

Dazu sind bereits vorgefertigte, (grafische) Validierungsformeln vorhanden. Bei Anpassung des Prozesses werden im Falle der Abweichung Fehlermeldungen angezeigt.

##### ▶ **Anforderung des Kunden**

Die (grafischen) Validierungsformeln müssen selbst erstellt werden.

- Fragen?

- Clarke, E. M. et al.: *Model Checking*. Cambridge, London: MIT Press, 2001
- Virtuelles Software Engineering Kompetenzzentrum: *Kripke Strukturen*. Internet: <http://www.software-kompetenz.de/?4773> [01.11.2006]
- Bubeck, U.: *CTL Model Checking*. 13.02.2003.