Modellbildung in der Entwicklung mit Schwerpunkt Architekturen

Integration von Modellsichten am Beispiel von AutoFocus

Prof. Dr. Dr. h.c. Manfred Broy
Gemeinsam mit Dr. Bernhard Schätz
Fakultät für Informatik, TU München
SS 2007

Modellintegration

Modellintegration: Von Teilsichten zur Gesamtsicht

- Ein Modell ist eine Abstraktion: Definiert eine Sicht auf ein System
- Typische Sichten
  - Datensicht: Datenmodell
  - Schnittstellensicht: Black Box Modell
  - Zustandssicht: Zustandsmaschinen
  - Prozesssicht: Ablaufmodell
  - Architektursicht: Architekturmodell

Aufgaben der Modellintegration

- Zusammenspiel unabhängiger Modellsichten
  - Konsistenz
  - Vollständigkeit
- Zusammenspiel redundanter Modellsichten
  - Konsistenz
- Zusammenspiel der Modellierungstechniken einer Modellierungssprache

Beispiel Konsistenz Daten-/Zustandsmodell

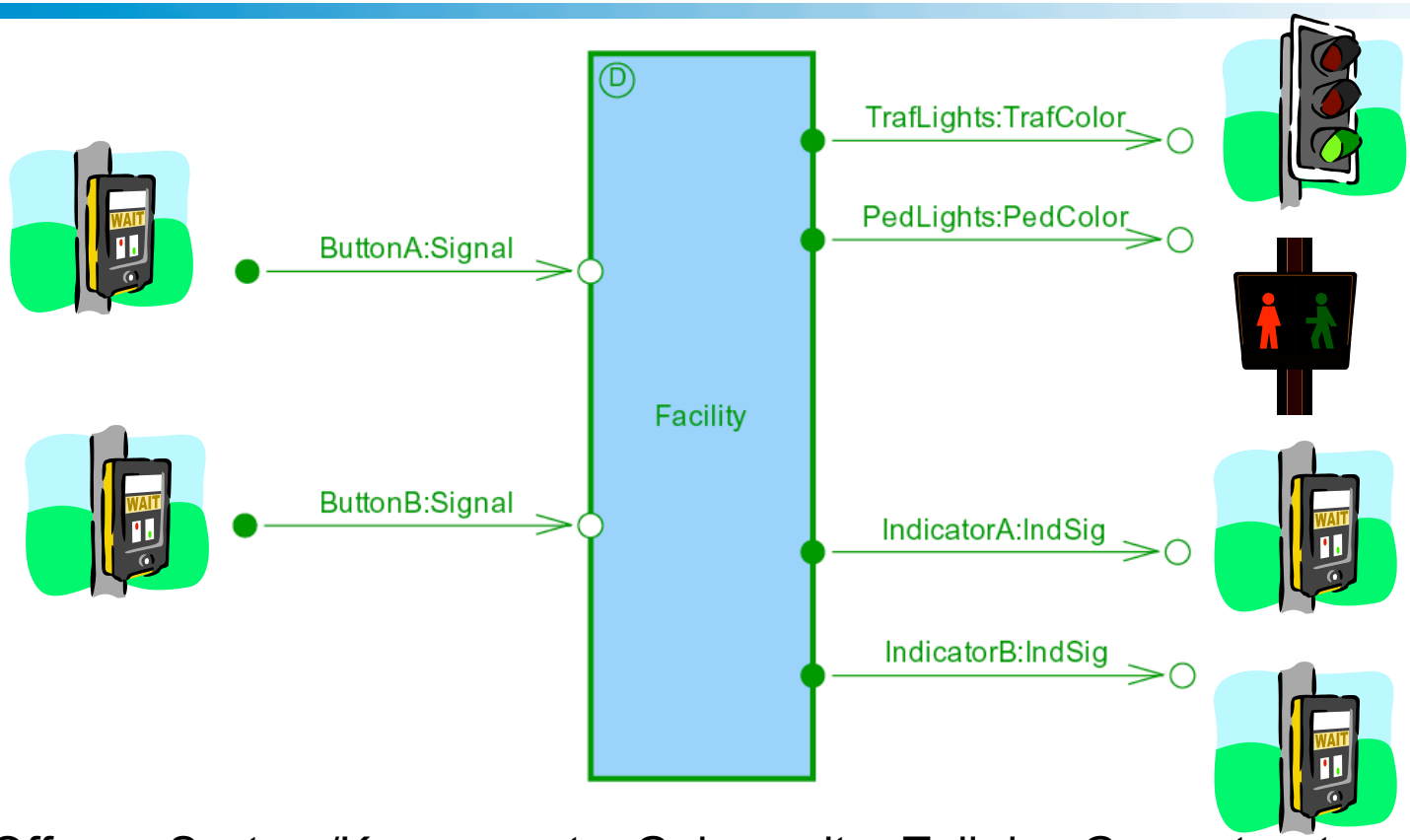
- Eine Zustandsmaschine mit Ein/Ausgabe arbeitet mit Daten
  - Zustandsattribute
  - Nachrichten für Ein/Ausgabe
- Ein Datenmodell definiert eine
  - Menge von Sorten/Datentypen
  - Menge von Operationen darauf
- Konsistenz: Die Typen/Sorten werden wie durch ihre Operationen festgelegt verwendet
- Vollständigkeit: Alle verwendeten Typen/Sorten und Operationen sind definiert

Das Beispiel: AutoFocus

Wir demonstrieren die Problematik am Beispiel

- AutoFocus: Ein Werkzeug für modellbasierte Entwicklung von Software für eingebettete Systeme

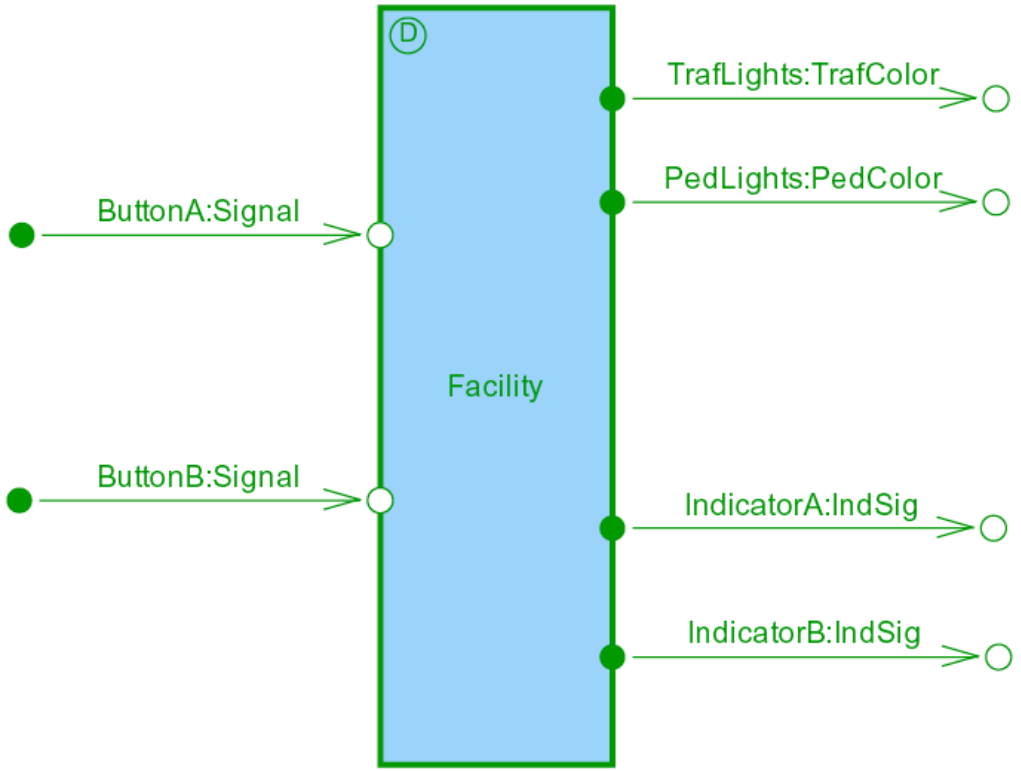
Konzept: Offenes System/Komponente



Offenes System/Komponente: Gekapselter Teil des Gesamtsystems

- Kopplung mit Umgebung über Schnittstelle
- Umgebungsunabhängig beschreibbares Verhalten

Konzept: **Schnittstelle**



Syntaktische Schnittstelle: Interaktionspunkte Komponente/Umgebung

- Richtung der Interaktion (Ein-/Ausgabe)
- Art der Interaktion (Nachrichtentyp)

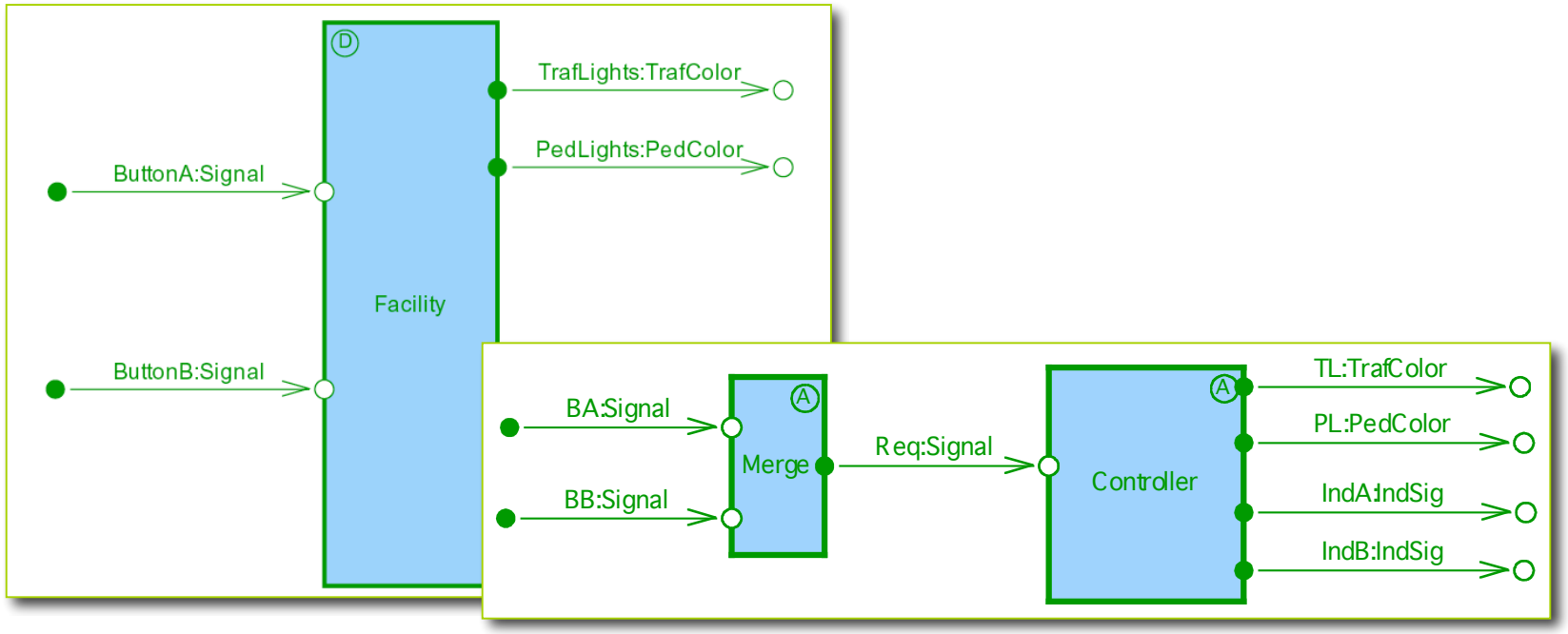
Kompatibilität zum Datenmodell

- Die auftretenden Nachrichten sind im Datenmodell deklariert

Kompatibilität bei Verhaltensmodellierung

- Soweit ist nur ein „syntaktische Schnittstelle“ des Systems/der Komponente beschrieben
- Das „Innere“ (das Verhalten) des Systems/der Komponente wird beschrieben durch
  - Architektur
  - Zustandsmaschine
  - Interaktionsdiagramme
  - Black Box Spezifikation (in AutoFocus nicht implementiert)
- Kompatibilität: Das „Innere“ passt zur Syntaktischen Schnittstelle

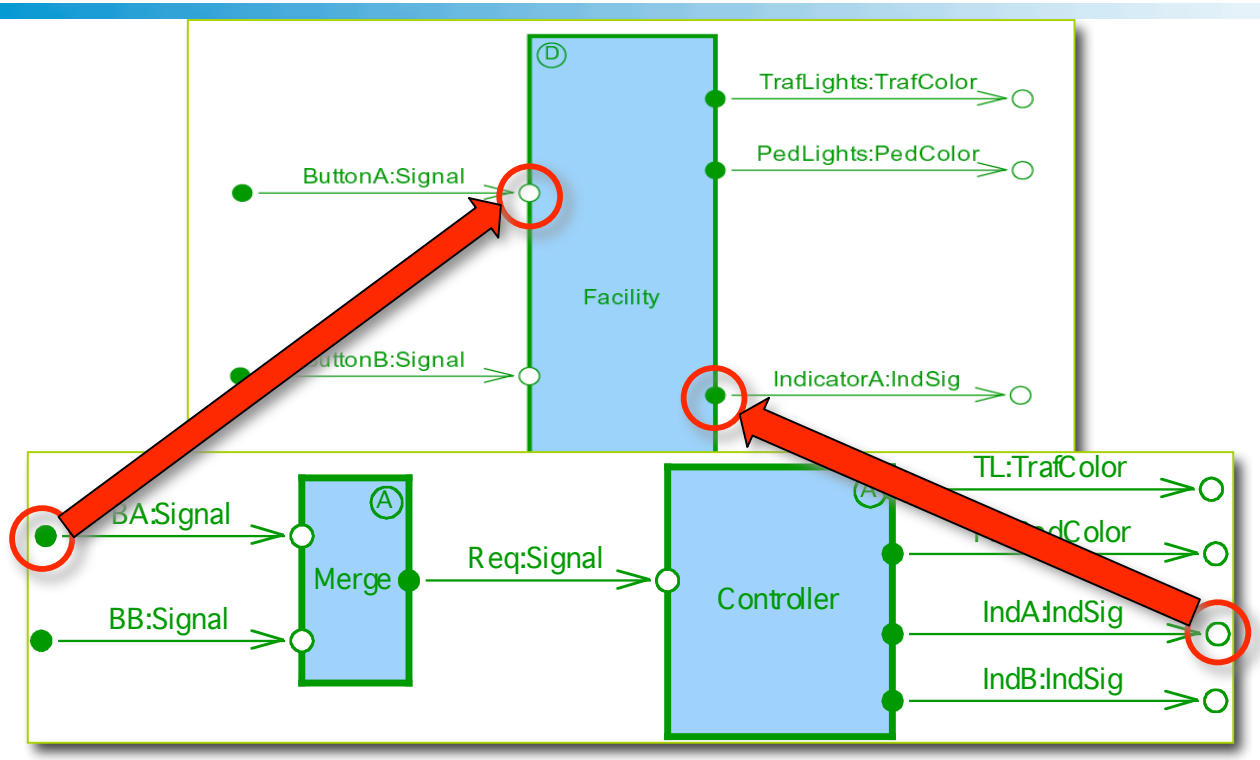
Konzept: Architektur und Hierarchie



Hierarchie: Strukturelle Komponentenarchitektur

- Interne Komponentenstruktur („glass box“)
- Unterkomponenten und interne Kanäle
- Interne und externe Schnittstelle

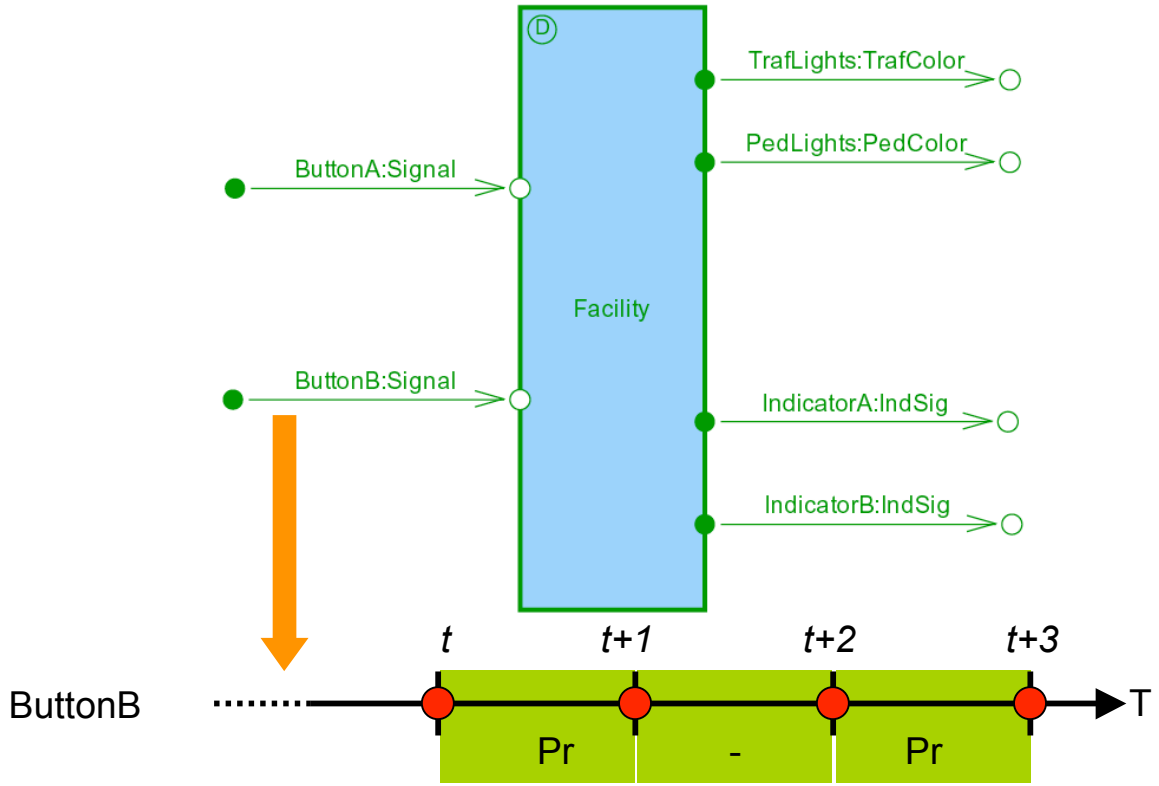
Integration: Kopplung **Externe/interne Sicht**



Integration: Schnittstellensicht

- Interne Eingabe = externe Ausgabe
- Interne Ausgabe = externe Eingabe

Konzept: **Signal**



Signal: Kanalhistorie

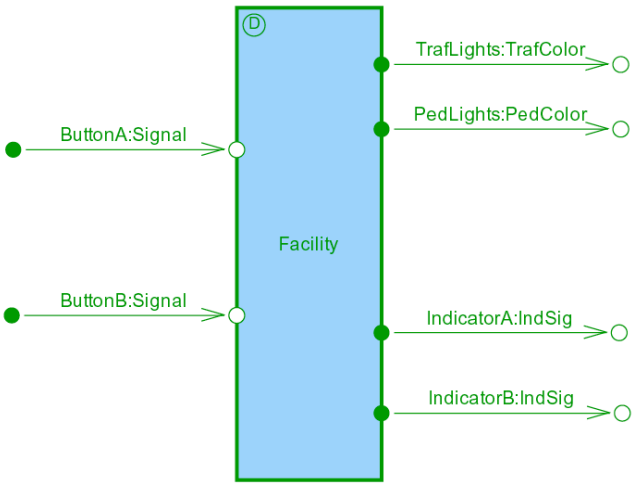
- Gezeitete Beobachtung des Kanals mit globalem Zeittakt
- Definition der beobachteten Nachrichten pro Zeittakt

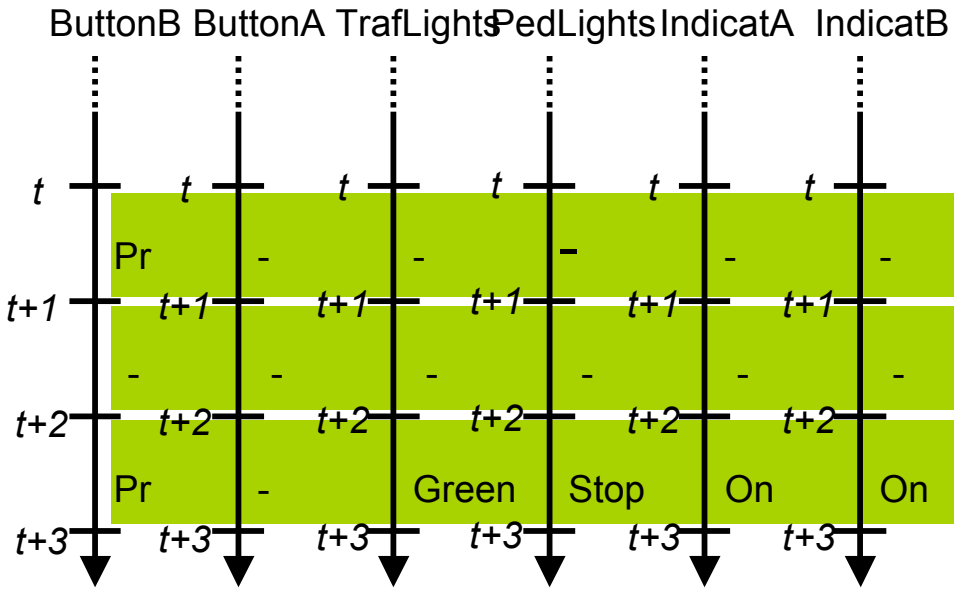
Modell: Strom (wie gehabt)

Strom: (Potentielle unendliche) Nachrichtensequenz

- Daten: Trägermenge $M = \{m_1, m_2, m_3, ...\}$
- Nachrichten: Datum oder Absenz $M_\perp = M \cup \{ \perp \}$
- Ströme: Sequenzen über $M_\perp$
  - Endliche Ströme: $M_\perp^*$
  - Unendliche Ströme: $M_\perp^\infty$
  - Potentiell unendliche Ströme: $M_\perp^\omega = M_\perp^* \cup M_\perp^\infty$
  - Funktional: $M_\perp^\omega = \mathbb{N} \rightarrow M_\perp$
- Varianten: M^ω (ungezeitet), $(M^*)^\omega$ (schwach kausal)
- Spezialfall Autofocus: Genau eine Nachricht pro Zeitintervall

Konzept: **Ablauf**





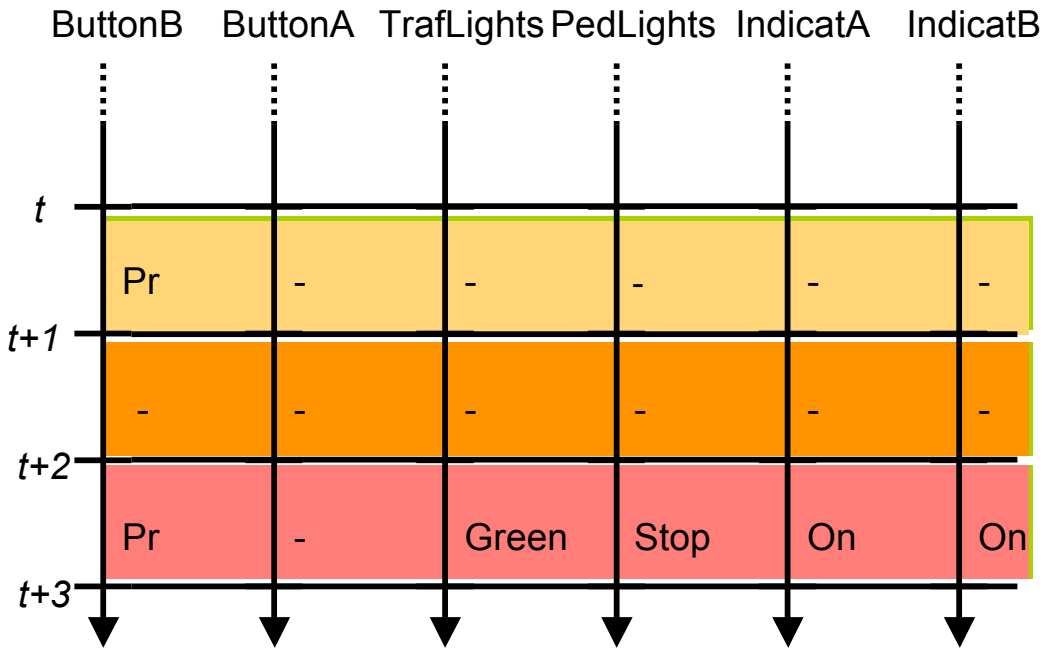
Ablauf: Signalhistorie Komponente

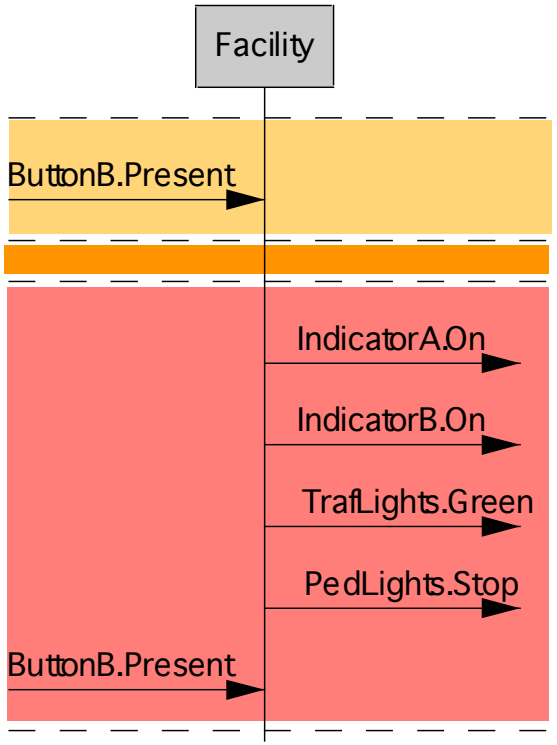
- Alle Schnittstellenelemente
- Alle Zeitschritte

Interaktionsdiagramme

- Ein Interaktionsdiagramm entspricht in unserer Terminologie eine Ablauf (Prozesssicht)
- Das Interaktiondiagramm beschreibt einen Ablauf (Beispiel) und dabei welche Nachrichten in welchen Zeitintervallen auf welchen Kanälen fließen

Konzept: **Interaktionssequenz**





Sequenz: Beschreibung Teilablauf

Modell: Ablauf

(Syntaktische) Schnittstelle: Typisierung

- Komponentenbezeichner: K
- Kanalbezeichner: $i_1,...,i_m,o_1,...,o_n$
- Nachrichtentyp: $I_1,...,I_n,I_1,...,I_n$

(Semantische) Schnittstelle: Verhalten

- Abläufe: $I_{1,\perp}^{\infty} \times ... \times I_{m,\perp}^{\infty} \times O_{1,\perp}^{\infty} \times ... \times O_{n,\perp}^{\infty}$
- Verhalten: $B(i_1,...,i_m,o_1,...,o_n)$:
  $B: I_{1,\perp}^{\infty} \times ... \times I_{m,\perp}^{\infty} \times O_{1,\perp}^{\infty} \times ... \times O_{n,\perp}^{\infty} \rightarrow \mathbb{B}$

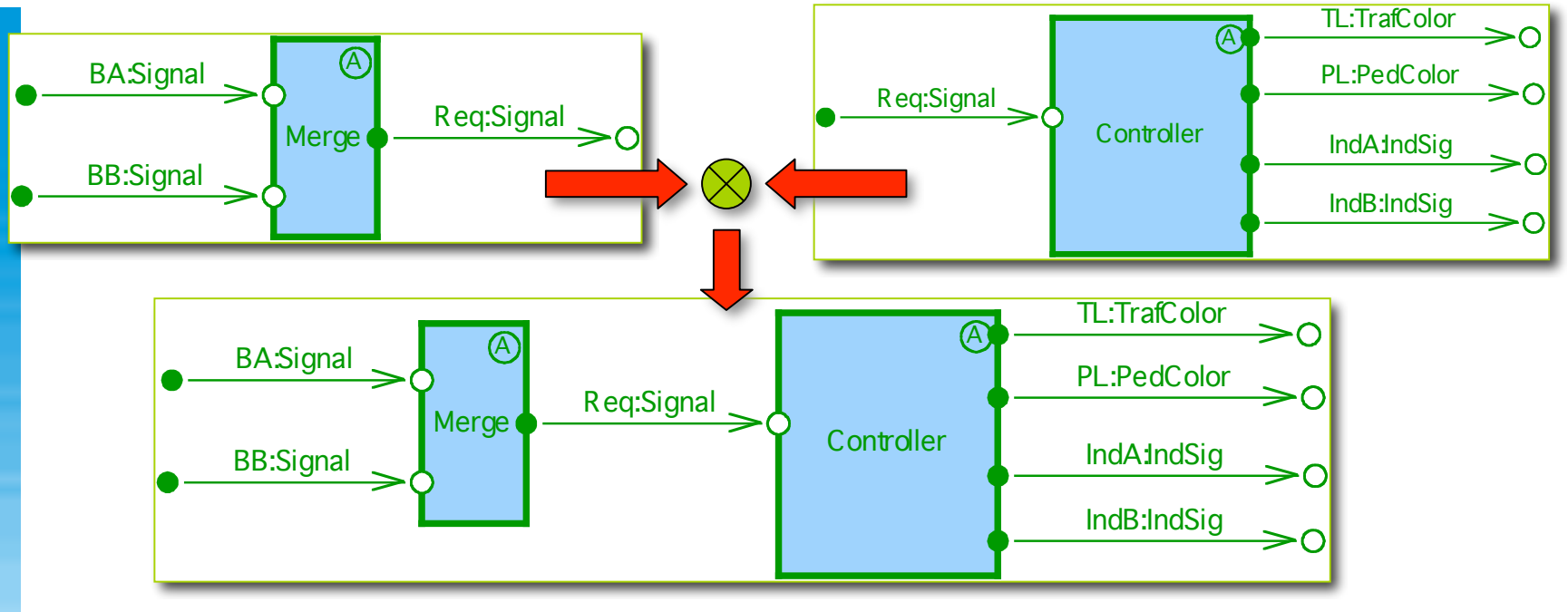
Zusicherungen für Kommunikationshistorien

- Die Spezifikation eines System/Komponenten- verhaltens durch eine Formel $B(i_1,...,i_m,o_1,...,o_n)$ ist syntaktisch korrekt, wenn
  - Die Formel nur über die vorgesehen Kanäle spricht
  - In der Formel die Kanalnamen für Ströme des ensprecenden Datentyps verwendet werden
- Darüber hinaus gibt es noch semantische Bedingungen
  - Keine Widersprüche in der Formel
  - Keine Beschränkungen der Eingabe:
    $\forall\, i_1,...,i_m.\ \exists\, o_1,...,o_n.\ B(i_1,...,i_m,o_1,...,o_n)$
  - Keine Widerspruch zur Kausalität

Komposition

- Syntaktische Kompatibilität
  - Kanalnamen und Kanaltypen passen zusammen

Konzept: Komposition



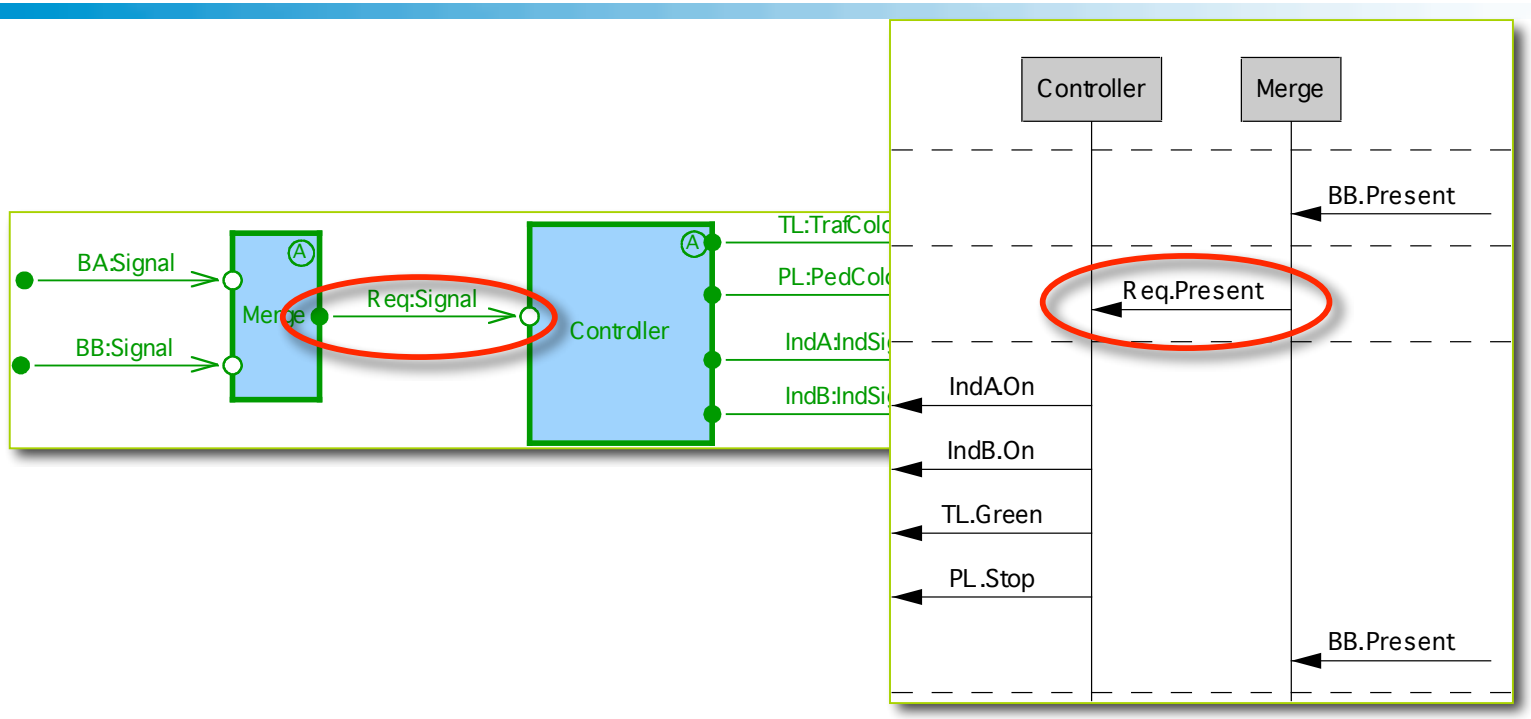
Komposition:

- Kopplung Komponenten mittels gemeinsamer Kanäle
- Komponenteninteraktion mittels Nachrichtenaustausch

Verhalten von Architekturen

- In den Architekturen gibt es Interaktion zwischen den Komponenten
- Diese Interaktion kann wieder durch Interaktionsdiagramme dargestellt werden
- Die so dargestellten Beispielabläufe müssen zu dem Verhalten der Komponenten passen

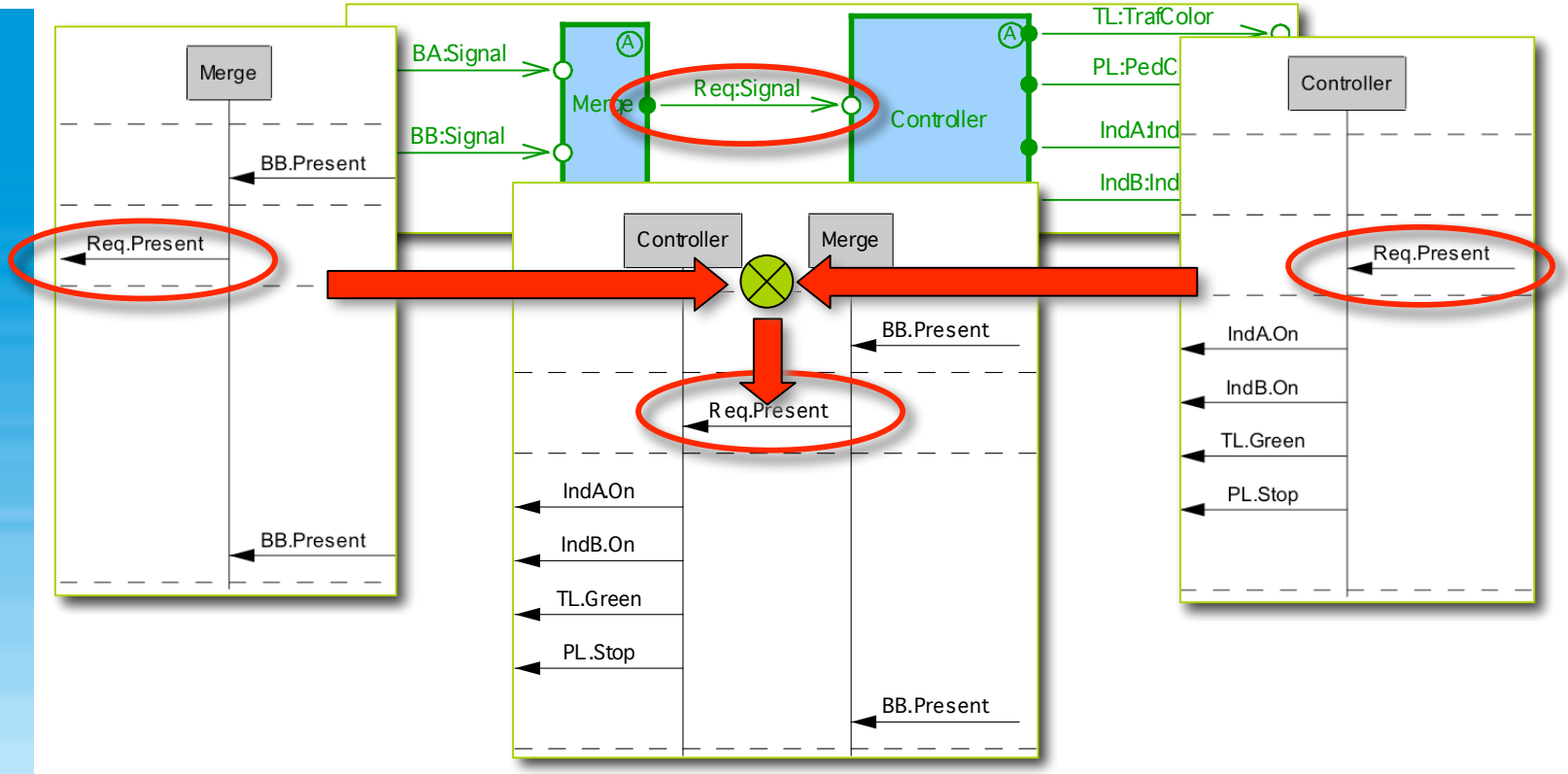
Konzept: **Verteilter Ablauf**



Ablauf: Signalhistorie komponiertes System

- Abläufe Komponenten
- Synchronisation gemeinsamer Kanäle

Konzept: **Kompositionalität**



Kompositionalität: Verhaltensbestimmung

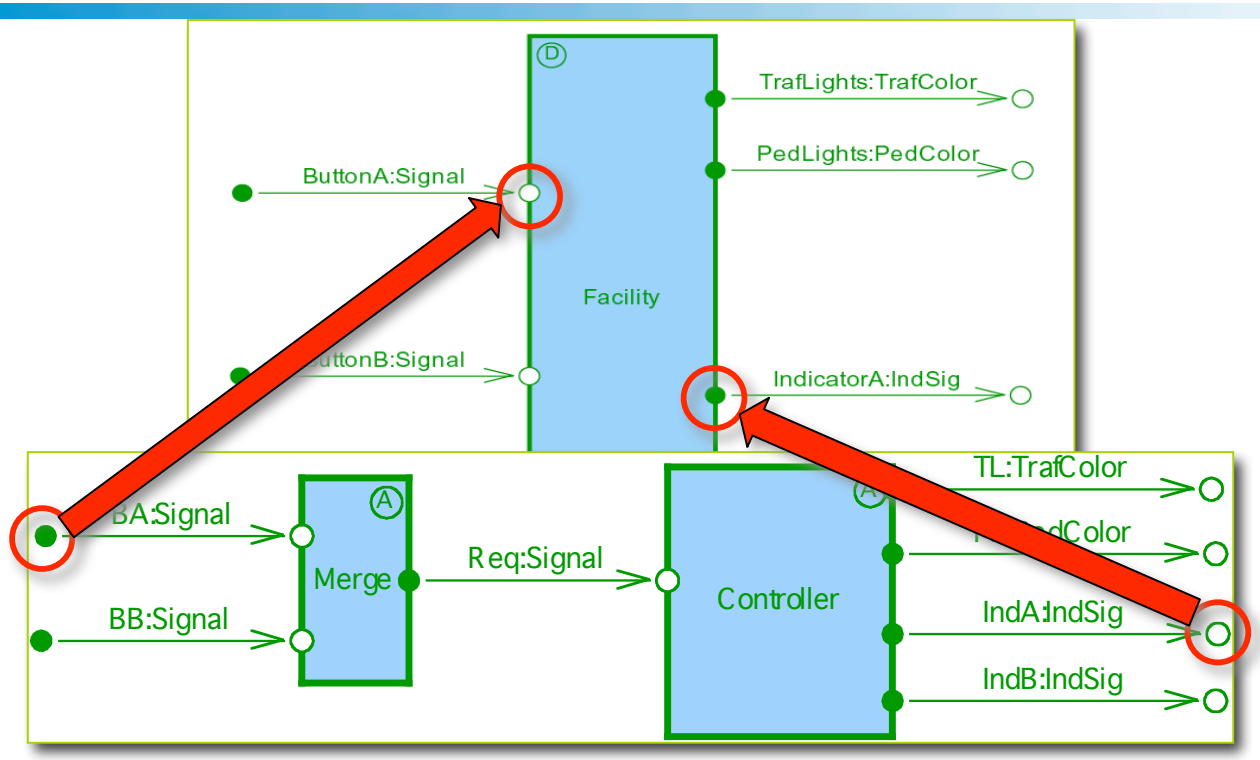
- Von: Verhalten Einzelkomponenten
- Zu: Verhalten Gesamtsystem

Modell: Komposition

Komposition: K || K‘

- Schnittstelle K: $i_1,...,i_m,o_1,...,o_n$
- Schnittstelle K‘: $i‘_1,...,i‘_{m‘},o‘_1,...,o‘_{n‘}$
- Synchronisation $i_1 = o‘_1,...,i_k = o‘_k$, $i‘_1 = o_1,...,i‘_{k‘} = o_{k‘}$
- Schnittstelle K||K‘: $i_{k+1},...,i_m$, $i‘_{k‘+1},...,i‘_{m‘},o_1,...,o_no‘_1,...,o‘_{n‘})$
- Verhalten $K||K‘(i_{k+1},...,i_m, i‘_{k‘+1},...,i‘_{m‘},o_1,...,o_no‘_1,...,o‘_{n‘}) = K(i_1,...,i_m,o_1,...,o_{n,}) \wedge K‘(i‘_1,...,i‘_{m‘},o‘_1,...,o‘_{n‘})$
- Zusätzlich: Verstecken der internen Kanäle durch Existenzquantoren

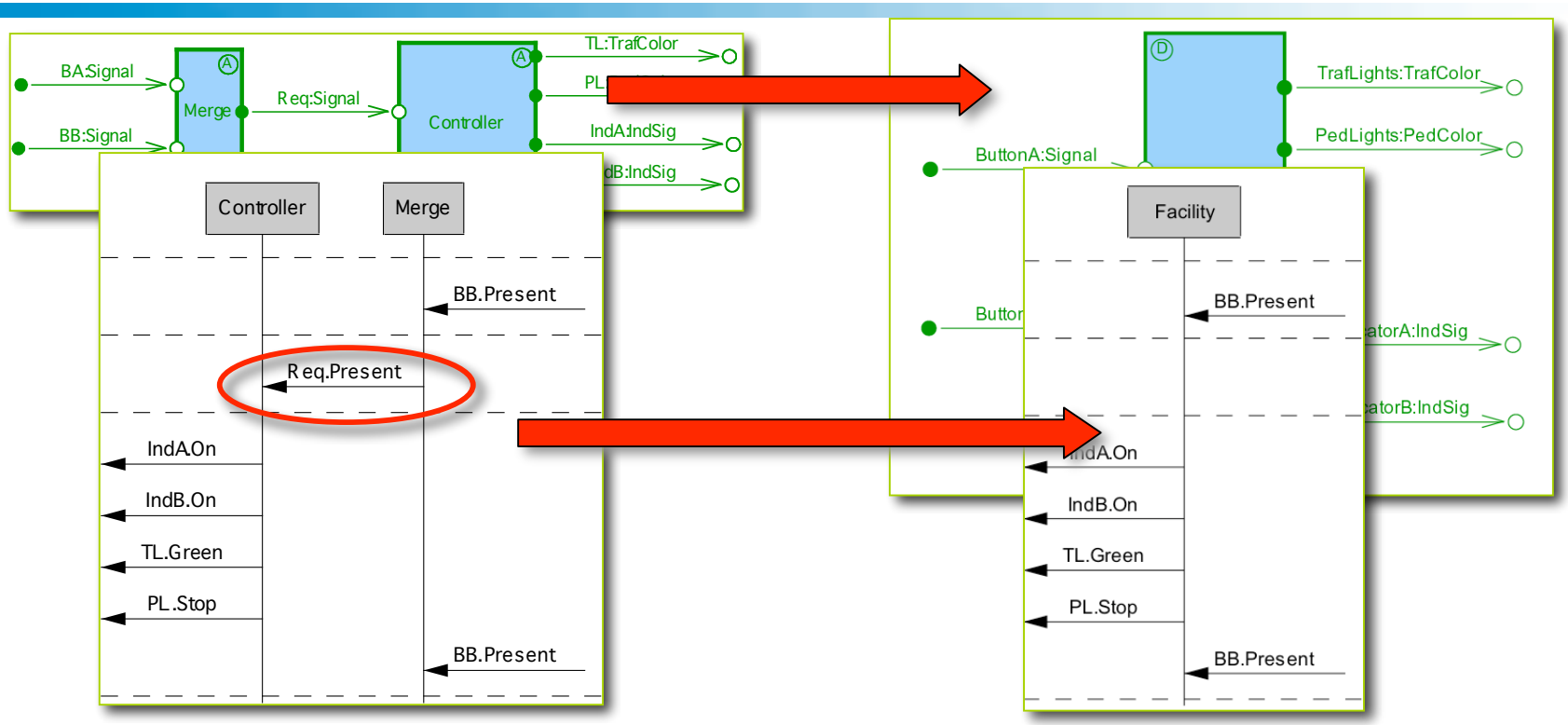
Konzept: **Kapselung**



Kapselung: Verbergen interner Struktur

- Unterkomponenten
- Interne Kanäle

Konzept: **Abstraktion**



Abstraktion: Verbergen interne Kommunikation

- Von: Abläufe Glass-Box
- Zu: Abläufe Black Box

Modell: Abstraktion

Komposition: $K\backslash\{i_1,\ldots,i_k,o_1,\ldots, o_{k'}\}$

- Schnittstelle K: $i_1,\ldots,i_m,o_1,\ldots,o_n$,
- Schnittstelle $K\backslash\{i_1,\ldots,i_k,o_1,\ldots, o_{k'}\}$: $i_{k+1},\ldots,i_m, o_{k'+1},\ldots, o_n$
- Verhalten $K\backslash\{i_1,\ldots,i_k,o_1,\ldots, o_{k'}\}(i_{k+1},\ldots,i_m, o_{k'+1},\ldots, o_n) =$
  $\exists i_1,\ldots,i_k,o_1,\ldots, o_{k'} . K(i_1,\ldots,i_m,o_1,\ldots,o_n)$

Komposition von Spezifikationen

Eingabekanäle

Ausgabekanäle

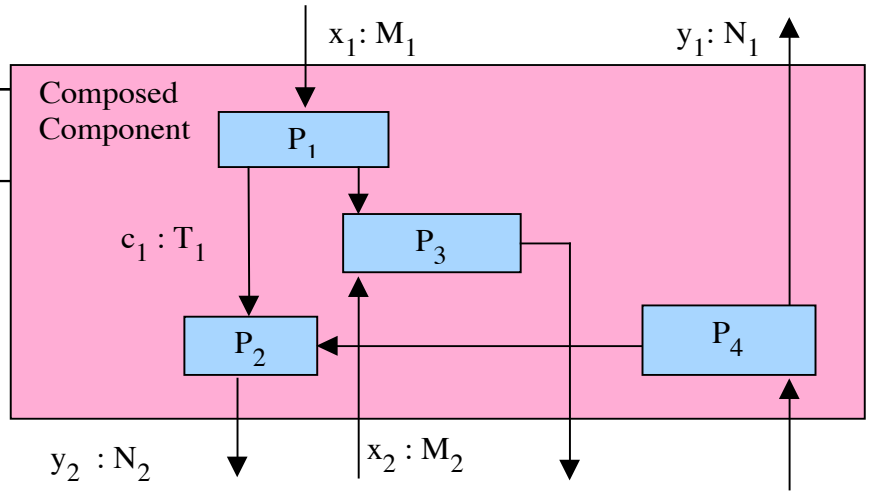
Interne Kanäle

Composed components spec

in $x_1: M_1 , x_2: M_2 , \ldots$

out $y_1: N_1 , y_2: N_2 , \ldots$

$\exists c_1, c_2 , \ldots : P_1 \wedge \ldots \wedge P_n$



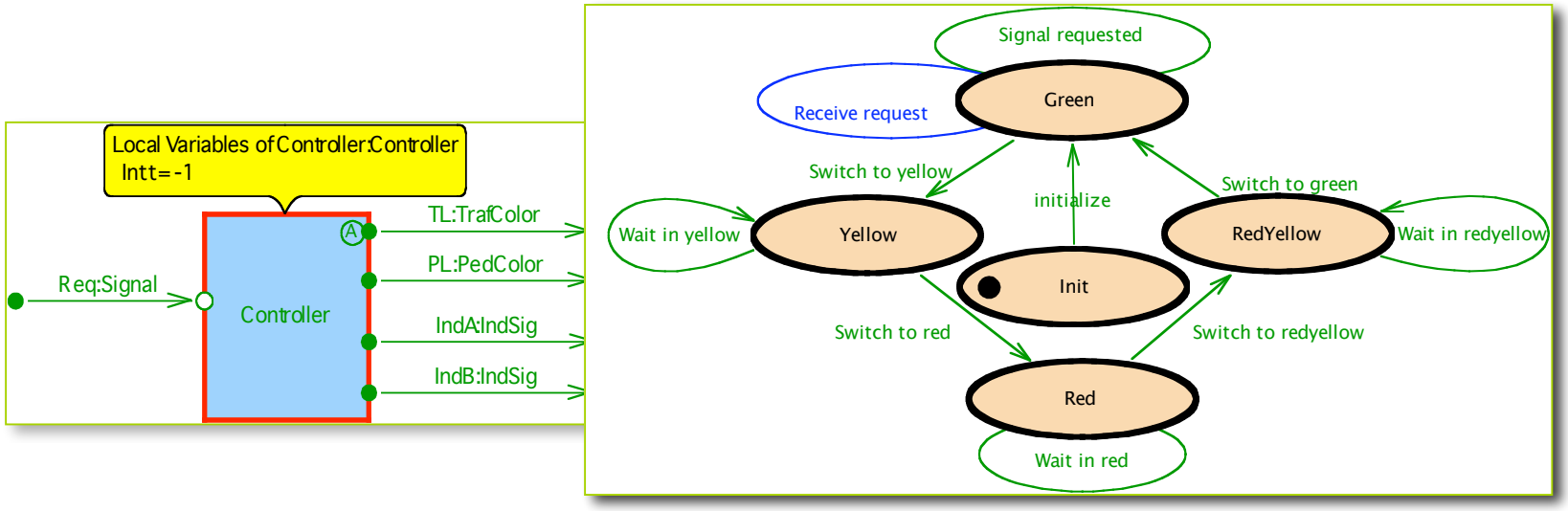
Systemkomposition = logisches UND

Channel Hiding = Existenzquantifikation

Zustandsmaschinen

- Das Verhalten eines Systems/einer Komponente kann auch durch Zustandsmaschine mit Ein/Ausgabe beschrieben werden

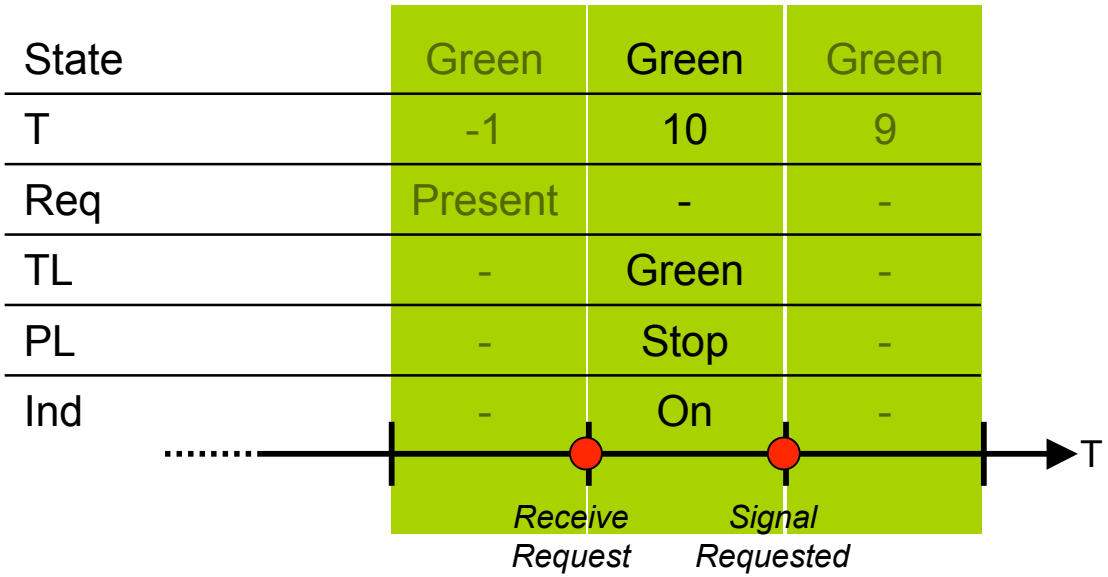
Konzept: **Zustandsautomat**



Automat: Zustandsorientierte Verhaltensbeschreibung

- Zustand: Betriebsmodi, Variablen , Eingabe Ausgabe
- Übergang: Lese-/Schreibe-Aktion

Konzept: **Zustand**



Zustand: Beobachtung zwischen zwei Übergängen

- Ports: aktuelle Nachrichten
- Variablen: aktuelle Werte

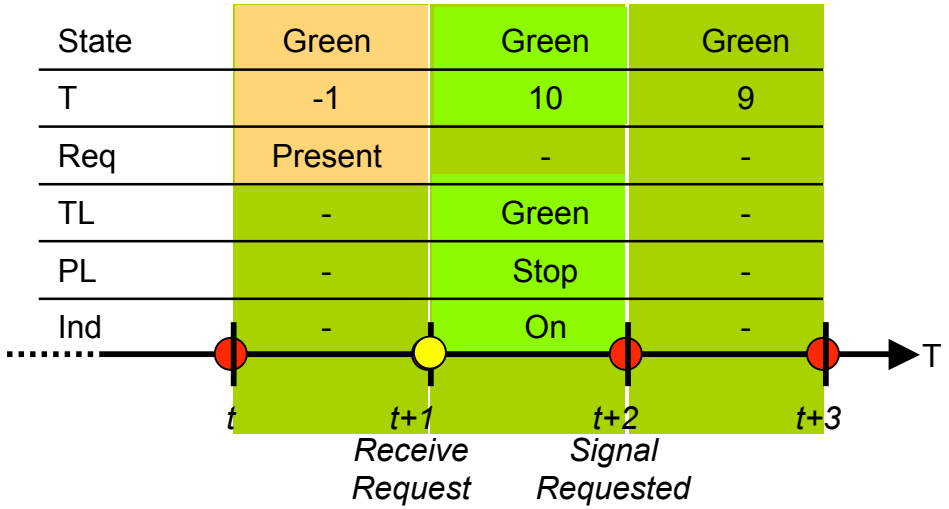
Konzept: **Übergang**

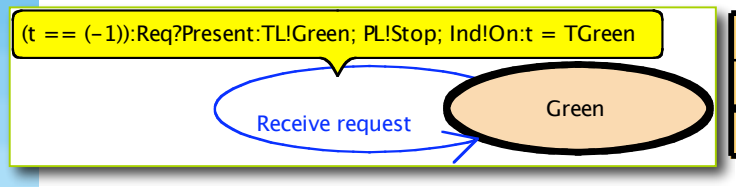
Vorzustand			Nachzustand				
Modus	t	Req	TL	PL	Ind	t	Modus
Green	-1	Present	Green	Stop	On	TGreen	Green

Übergang: Relation Vorzustand/Nachzustand

- Vorzustand: Modus, Variablen, Input Ports
- Nachzustand: Output Ports, Variablen, Modus

Konzept: **Ausführung**





Pre-State			Post-State				
State	t	Req	TL	PL	Ind	t	State
Green	-1	Present	Green	Stop	On	TGreen	Green

Ausführungsschritt: Lesen Vorzustand/Schreiben Nachzustand

- Vorzustand: Kontrollzustand, Datenzustand, Eingabe
- Nachzustand: Ausgabe, Datenzustand, Kontrollzustand

Modell: Ausführung

Automat: Struktur

- Zustand: S (s_0 Startzustand)
- Eingabe: $i_1,...,i_m$
- Ausgabe: $o_1,...,o_n$
- Übergangsrelation: $T(s,i_1,...,i_m,o_1,...,o_n,s)$:
  $S \times I_1 \times ... \times I_m \times O_1 \times ... \times O_n \times S \rightarrow \mathbb{B}$

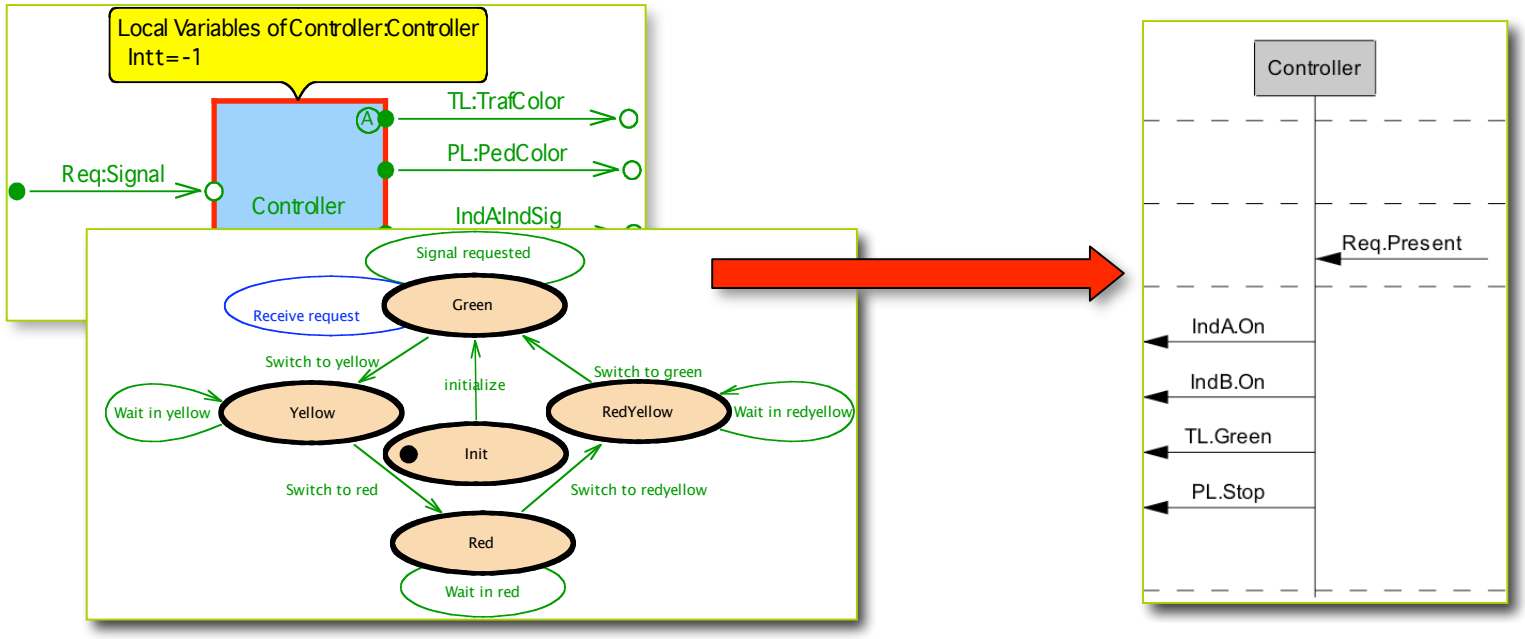
Automat: Verhalten

- $A(s,i_1,...,i_m,o_1,...,o_n)$: $S^\infty \times I_{1,\perp}{}^\infty \times ... \times I_{m,\perp}{}^\infty \times O_{1,\perp}{}^\infty \times ... \times O_{n,\perp}{}^\infty \rightarrow \mathbb{B}$
- $A(s, i_1,...,i_m,o_1,...,o_n) =$
  $s.0 = s_0$ and $o_1.0 = \perp \wedge ... \wedge o_1.0 = \perp \wedge$
  $\forall t.\ T(s.t,i_1.t,...,i_m.t,o_1.(t+1),...,o_n.(t+1),s.(t+1))$

Ablauf einer Maschine

- Eine Zustandsmaschine mit Ein-/Ausgabe erzeugt
  - für jede Eingabehistorie
  - Einen Ablauf bestehend aus
    - einem Strom von Zuständen
    - einerAusgahistorie

Konzept: **Ablauf Zustandsautomat**



Ablauf Zustandsautomat: Einschränkung Ausführung

- Einschränkung auf Schnittstellen
- Kapselung von Daten und Modus

Modell: Ablauf

Automat: Ausführungsablauf

- $K(i_1,\ldots,i_m,o_1,\ldots,o_n)$: $I_{1,\perp}^{\infty} \times \ldots \times I_{m,\perp}^{\infty} \times O_{1,\perp}^{\infty} \times \ldots \times O_{n,\perp}^{\infty} \rightarrow \mathbb{B}$
- $K(i_1,\ldots,i_m,o_1,\ldots,o_n) = A(s,i_1,\ldots,i_m,o_1,\ldots,o_n) \setminus \{s\}$

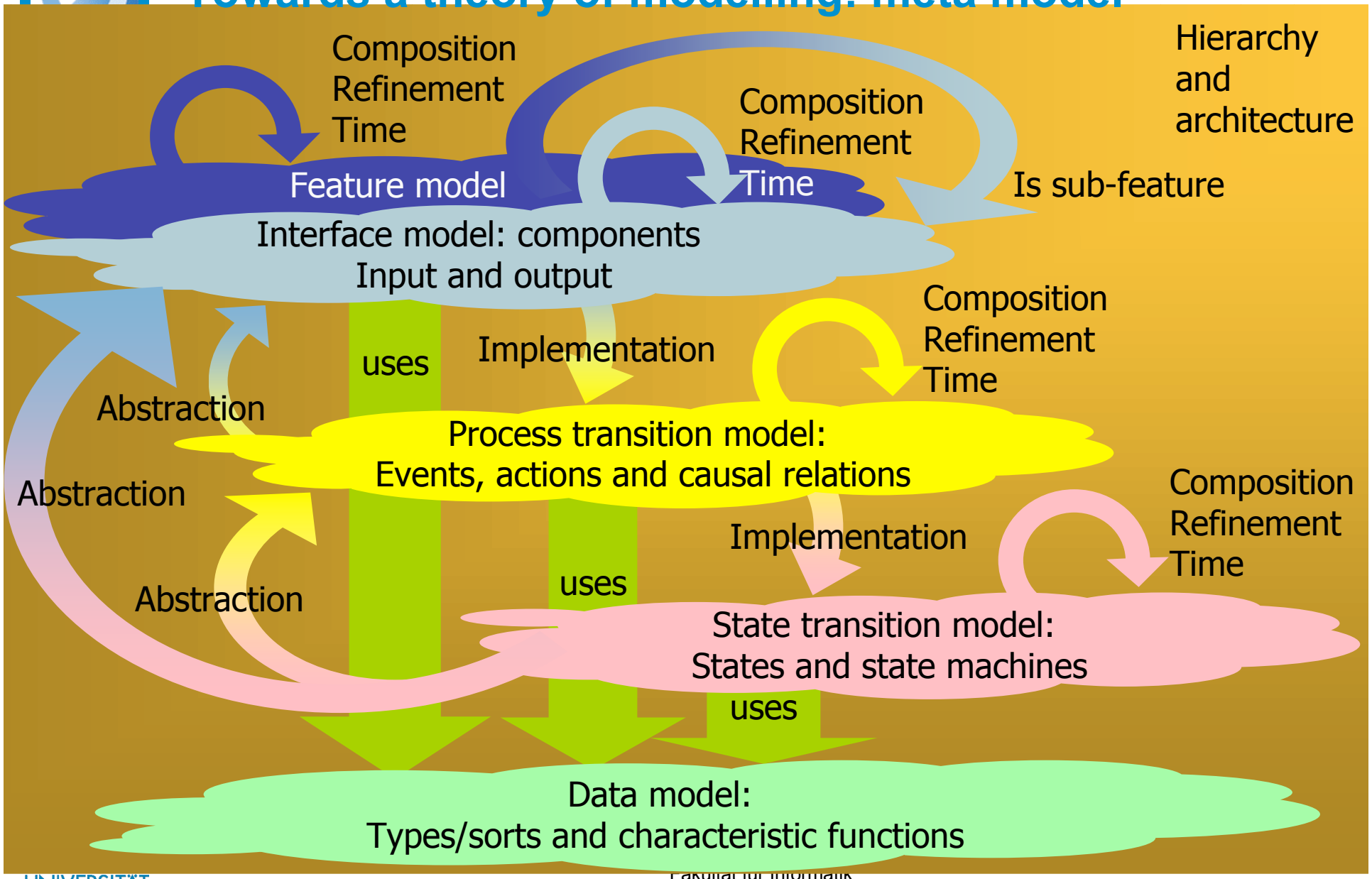
Zusammenfassung

Integration von Modellen:

- Abläufe (Ströme) als gemeinsames Modell
- Modularer Modellbegriff
  - Abläufe komponierter Komponenten aus Abläufen der einzelnen Komponenten
  - Abläufe gekapselter Komponenten aus Abläufen der offenen Komponenten
  - Abläufe implementierter Komponenten aus Abläufen der Zustandsmaschinen der Komponenten

Modulare Integration von Systemmodellen erlaubt modulare Systementwicklung

Towards a theory of modelling: meta model



Systemmodell

Ein System

- hat eine syntaktische Schnittstelle (Ein-/Ausgabekanäle)
- wird in Verhalten und/oder Struktur dargestellt
  - IF: durch stromverarbeitende Funktion beschrieben eine logische Formel (Schnittstellenspezifikation)
  - ZM: eine Zustandsmaschine, beschrieben durch ein Zustandsübergangsdiagramm
  - AR: eine Architektur, beschrieben durch ein Netzwerk von Komponenten (für die wiederum Verhaltensbeschreibungen existieren)
  - PR: durch Mengen von Abläufen, beschrieben durch eine Menge von Interaktionsdiagrammen

Übergänge zwischen Darstellungen

Zwischen jeder der Darstellungen gibt es Übergange

	IF	ZM	AR	PR
IF				
ZM				
AR				
PR				

Wechsel zwischen Systemsichten

- Ist für jede Komponente einer Architektur eine Zustandsmaschine gegeben, so kann die Architektur als Zustandsmaschine aufgefasst werden.
- Jede stromverarbeitende Funktion definiert eine Zustandsmaschine und umgekehrt
- ...

Beispiel für ein Verträglichkeitseigenschaft ...

Sei

- Abs die Schnittstellenabstraktion für Zustandsmaschinen
- || die Komposition auf Zustandsmaschinen
- $\otimes$ die Komposition für Schnittstellen

$$Abs((\Delta 1, \sigma 1) \,||\, (\Delta 2, \sigma 2)) =$$

$$Abs((\Delta 1, \sigma 1)) \otimes Abs((\Delta 2, \sigma 2))$$