

Embedded Systems / Eingebettete Systeme

Studiengang Informatik
Campus Minden

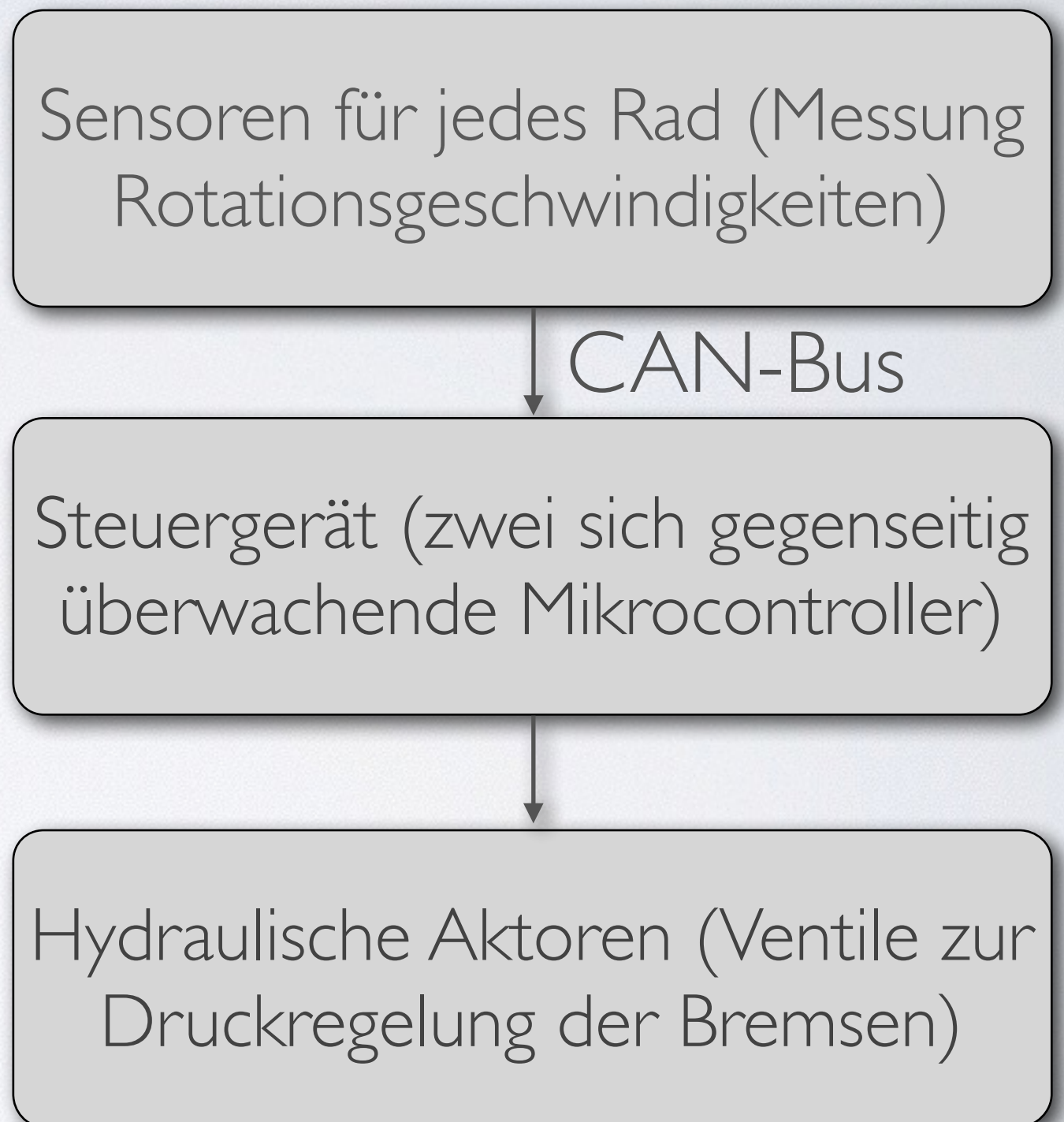
Matthias König



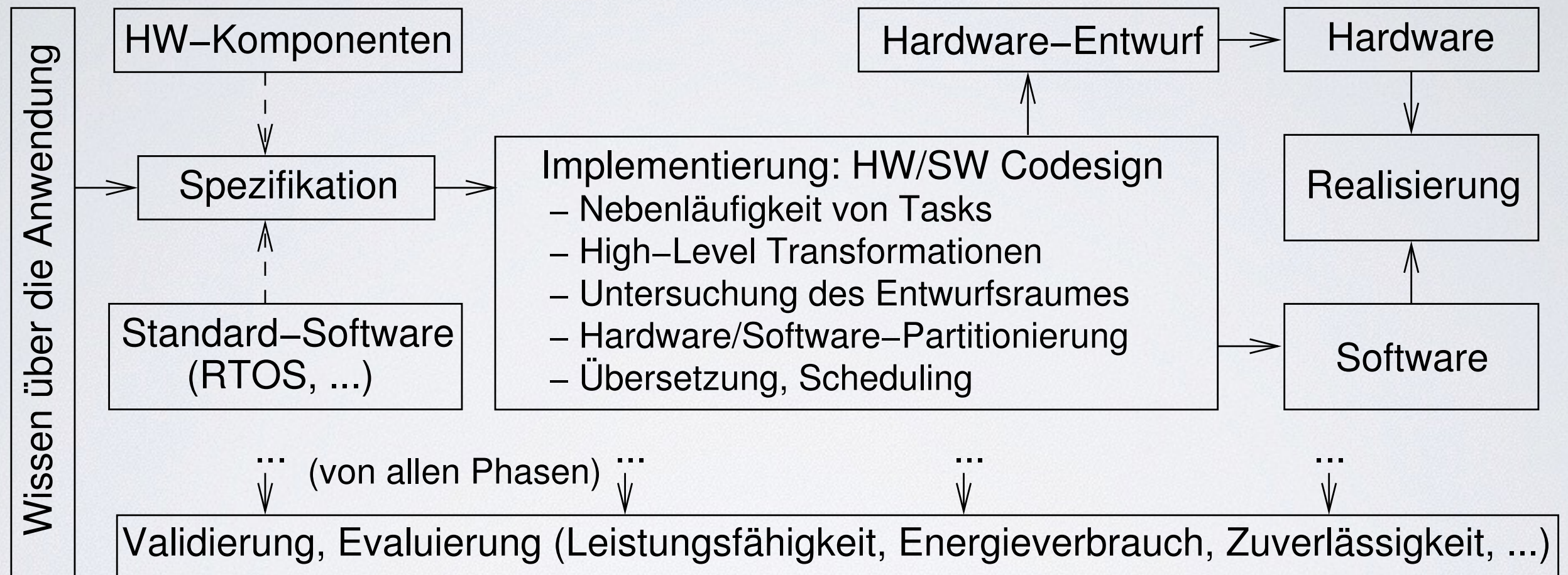
FH Bielefeld
University of
Applied Sciences

Beispiel einer bekannten Anwendung: Antiblockiersystem

- Verhindert beim Bremsen Blockieren der Räder
- Erhöhung der Sicherheit
- Harte Echtzeitanforderung
- Steuergerät (eingebettetes System), vier Geschwindigkeitssensoren, mindestens zwei hydraulische Aktoren



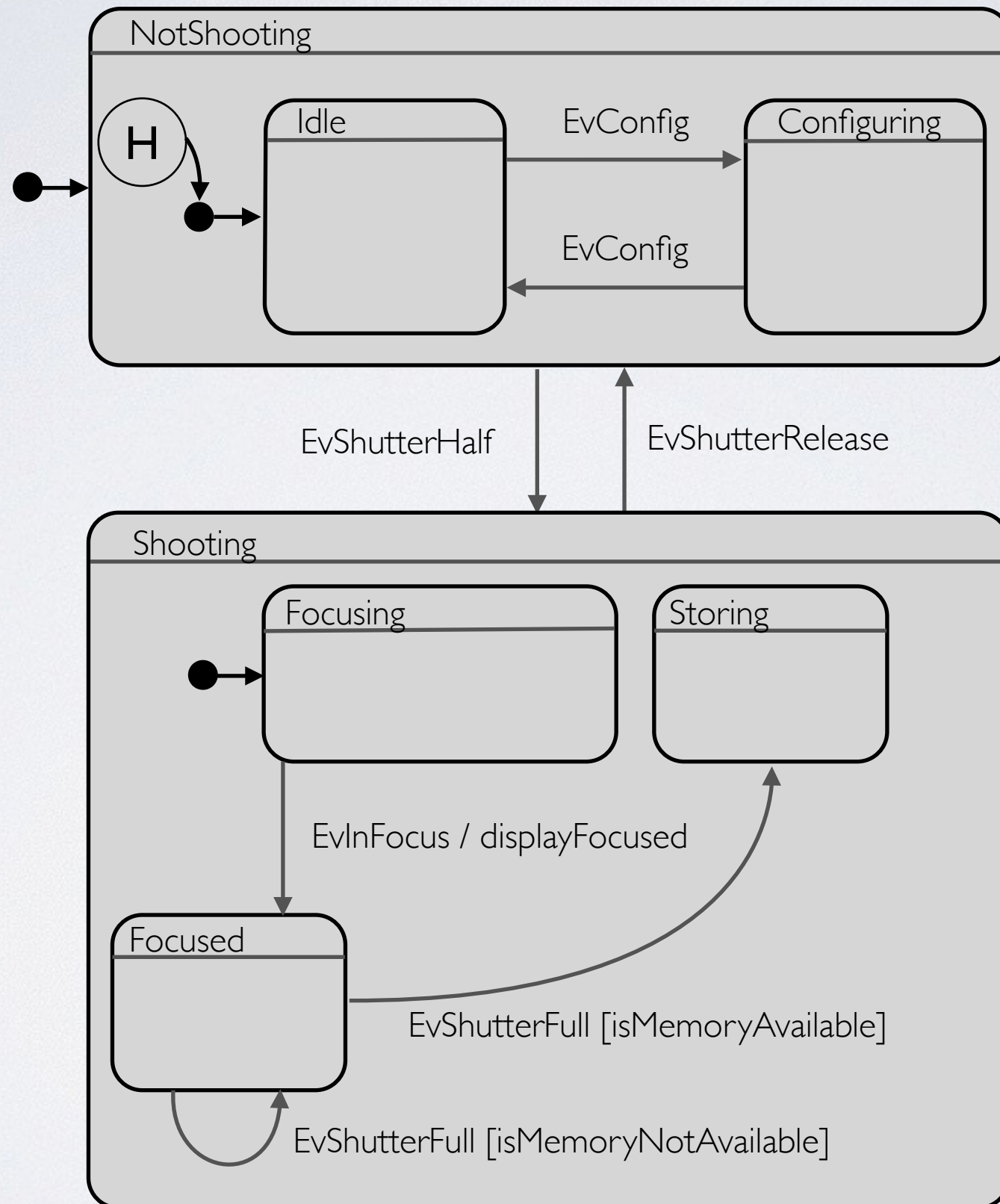
Hardware/Software Codesign



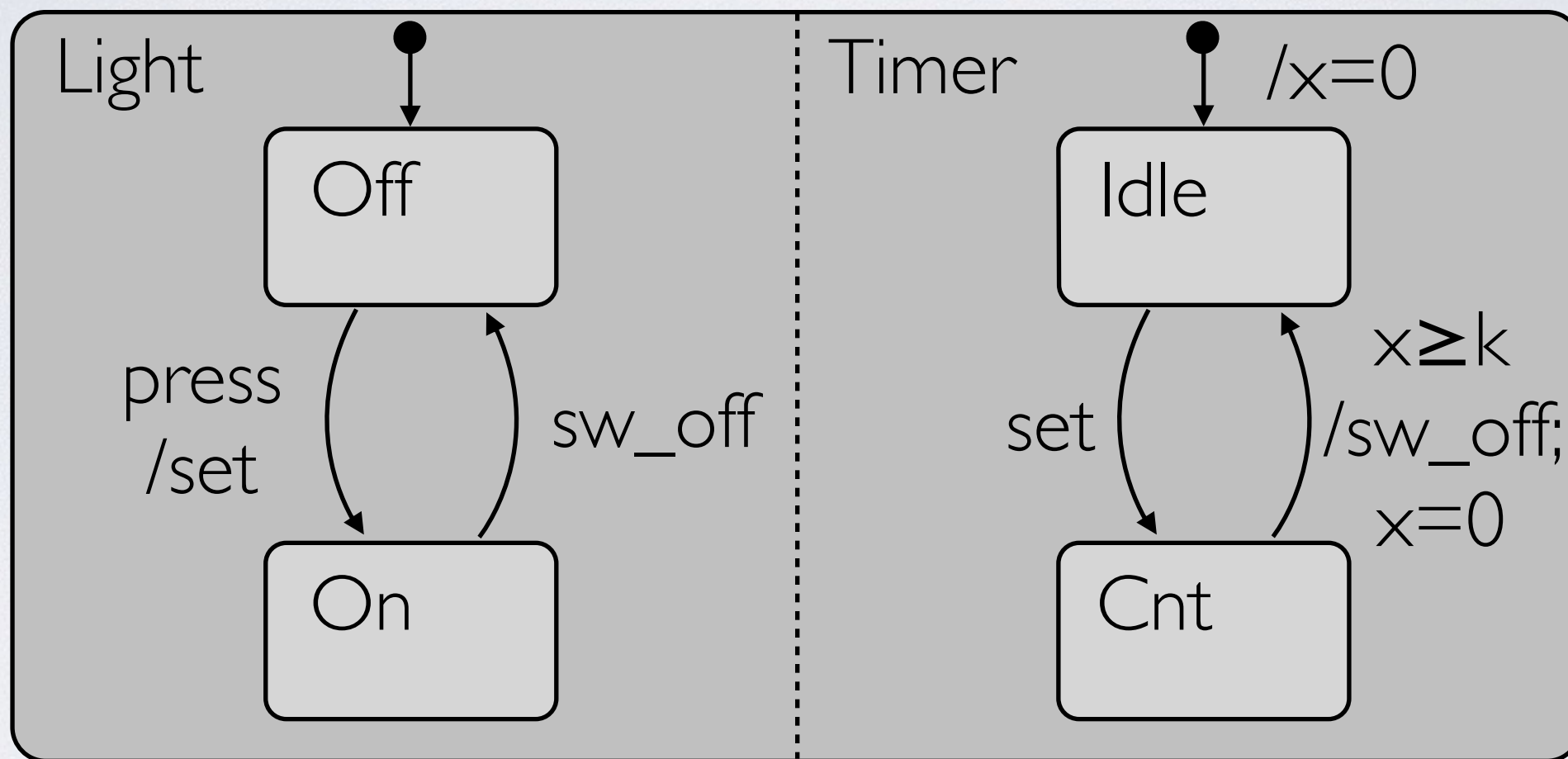
Entwurf Eingebetteter Systeme (nach Marwedel)

[Quelle: Marwedel, Eingebettete Systeme]

Wiederholung: Statecharts



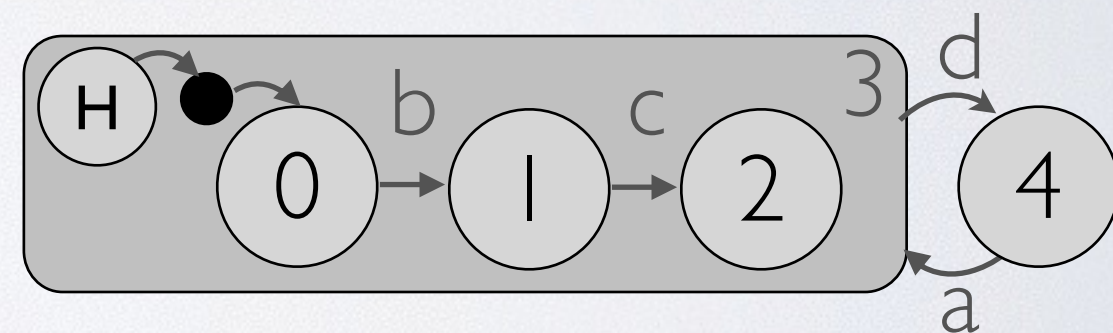
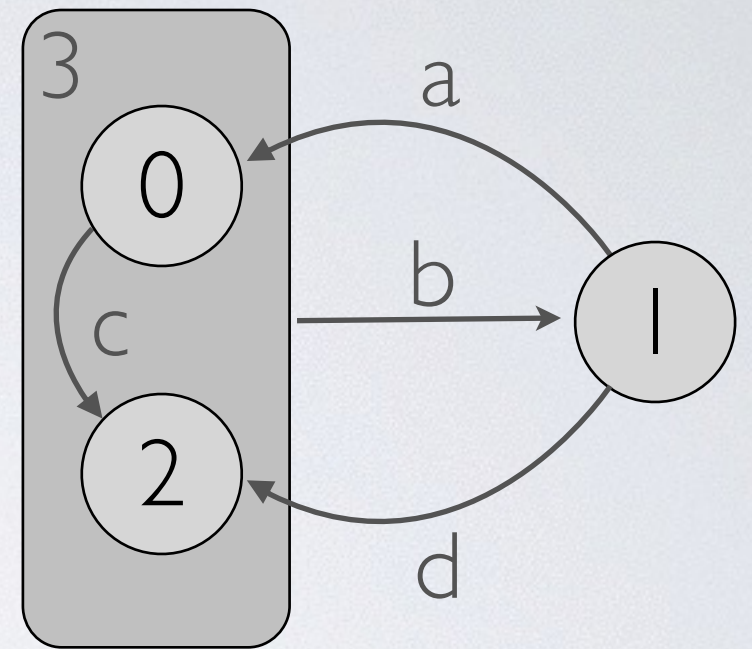
Wiederholung: Statecharts



In Cnt: tick/ $x = x + 1$

Wiederholung: Statecharts

- Erweiterung von endlichen Automaten durch
 - Hierarchie / Modularität
 - Nebenläufigkeit
- Begrenzt einsetzbar für
 - komplexe Berechnungsverfahren
 - verteilte Systeme

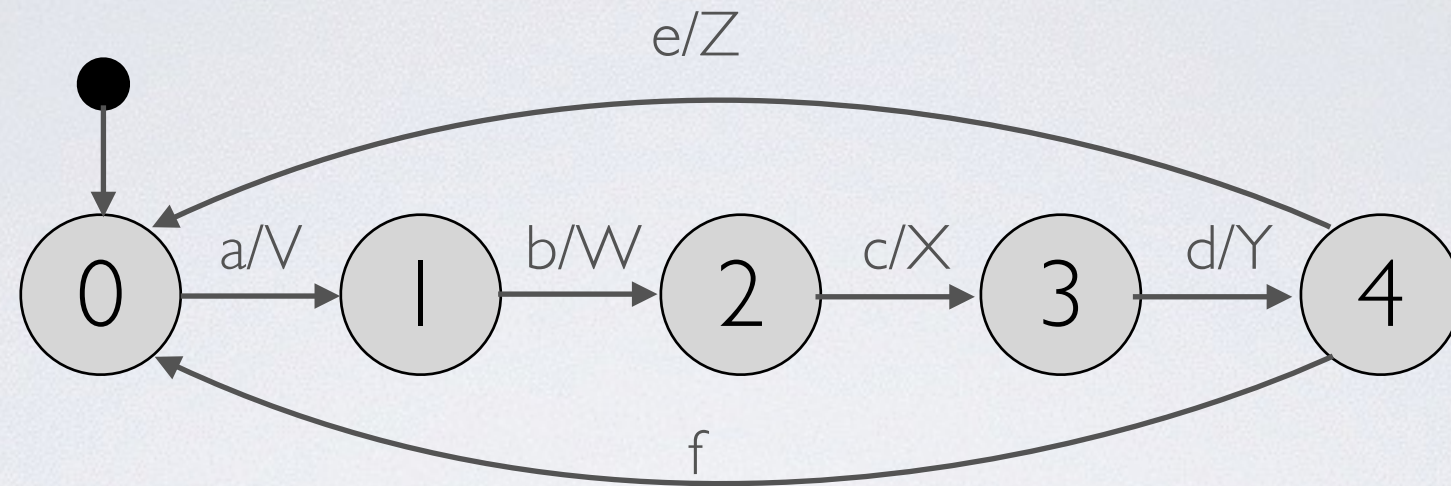


Specification and Description Language (SDL)

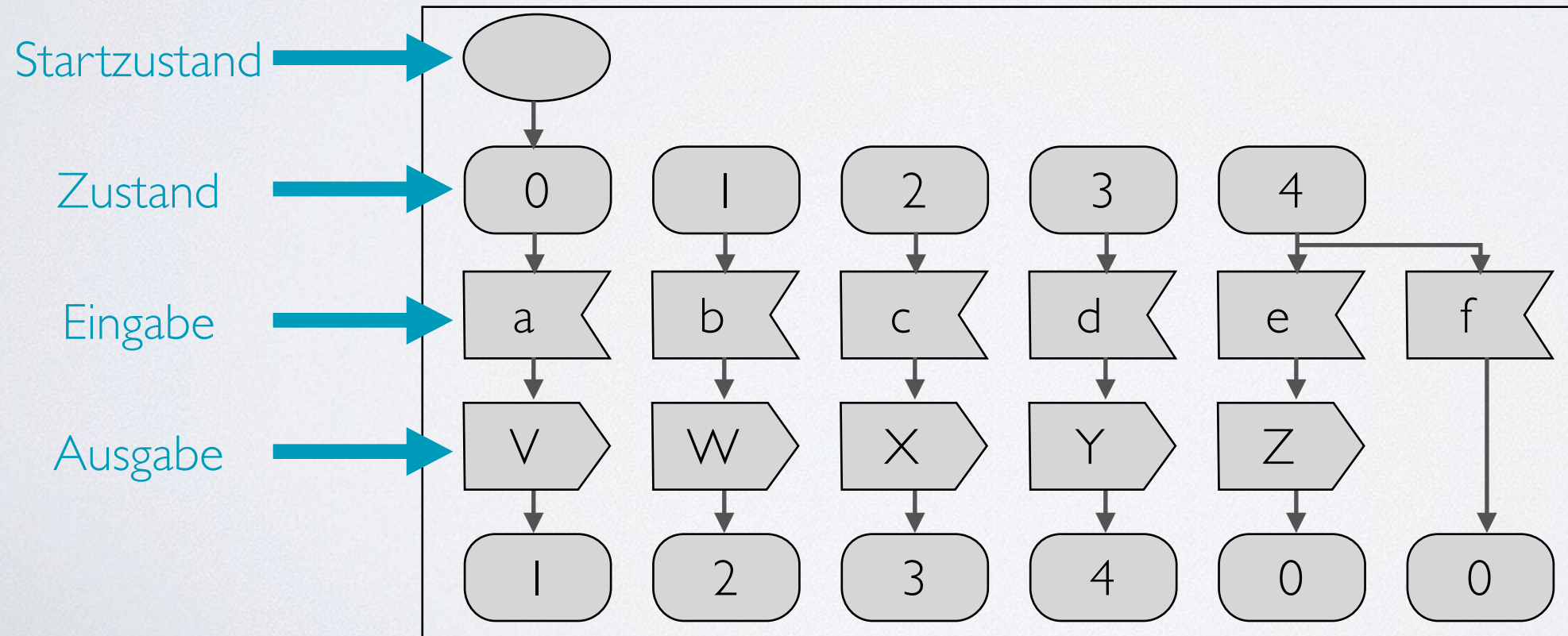
Specification and Description Language SDL

- Für verteilte Systeme geeignete Modellierungssprache
- Im Standard Z.100ff der International Telecommunication Union definiert (erste Version 1980)
- Verbreitet im Telekommunikationsbereich
- Basiert auf
 - Zustandsautomaten
 - Asynchronen Nachrichtenaustausch
- Unterstützt graphische und textuelle Repräsentation

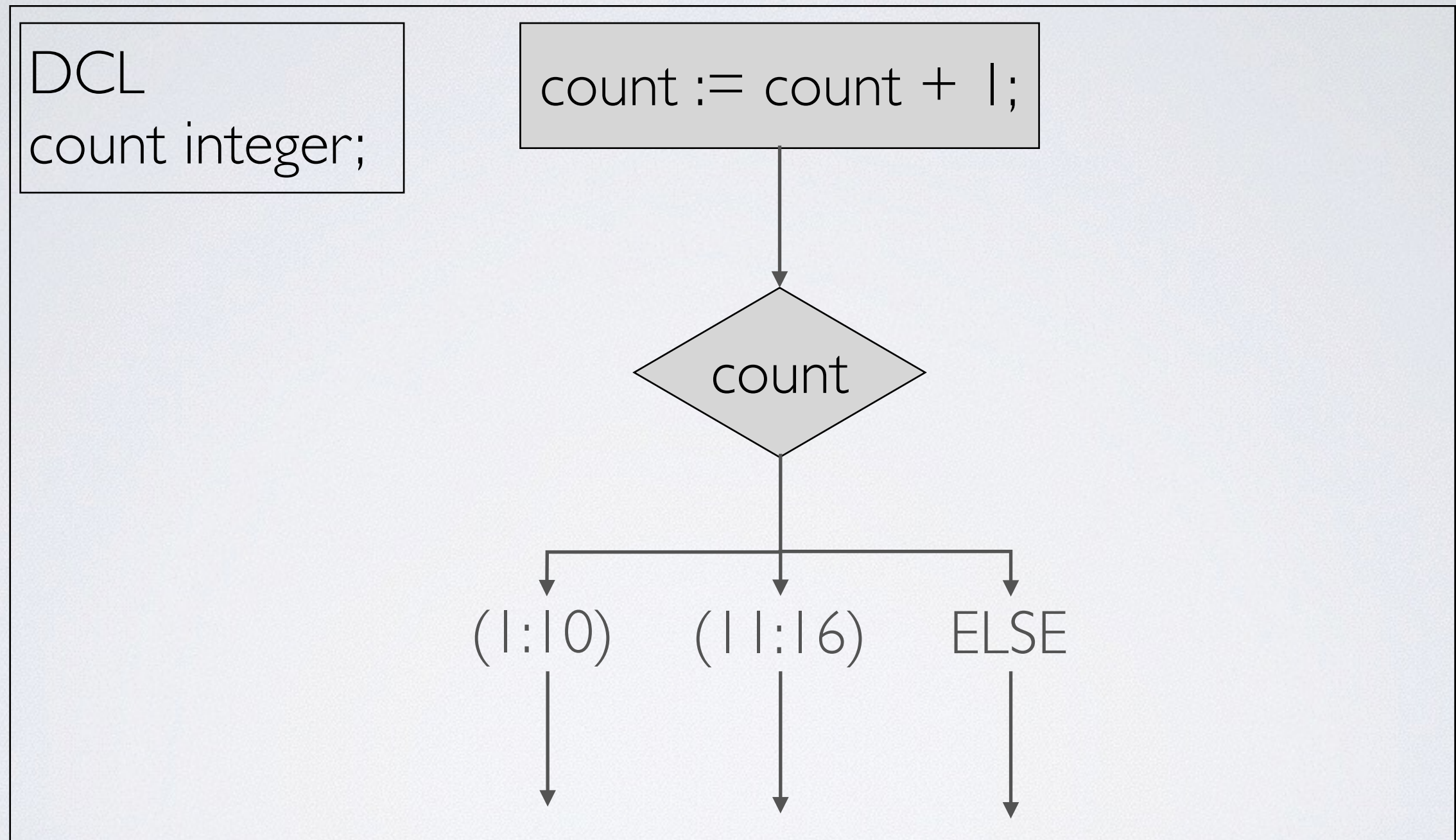
SDL: Prozesse für Zustandsautomaten



Prozess

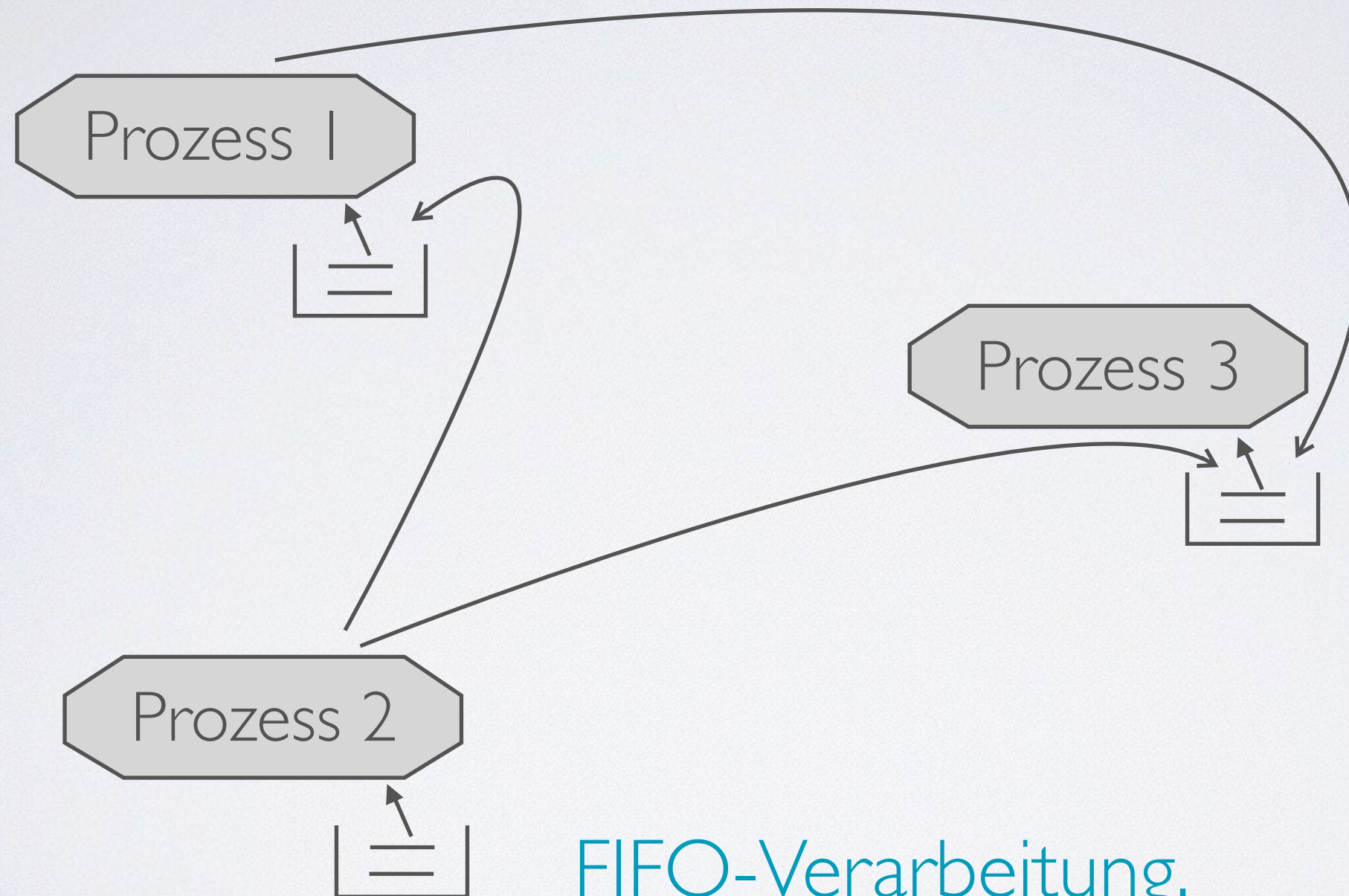


SDL: Zusätzlich Operation auf Daten



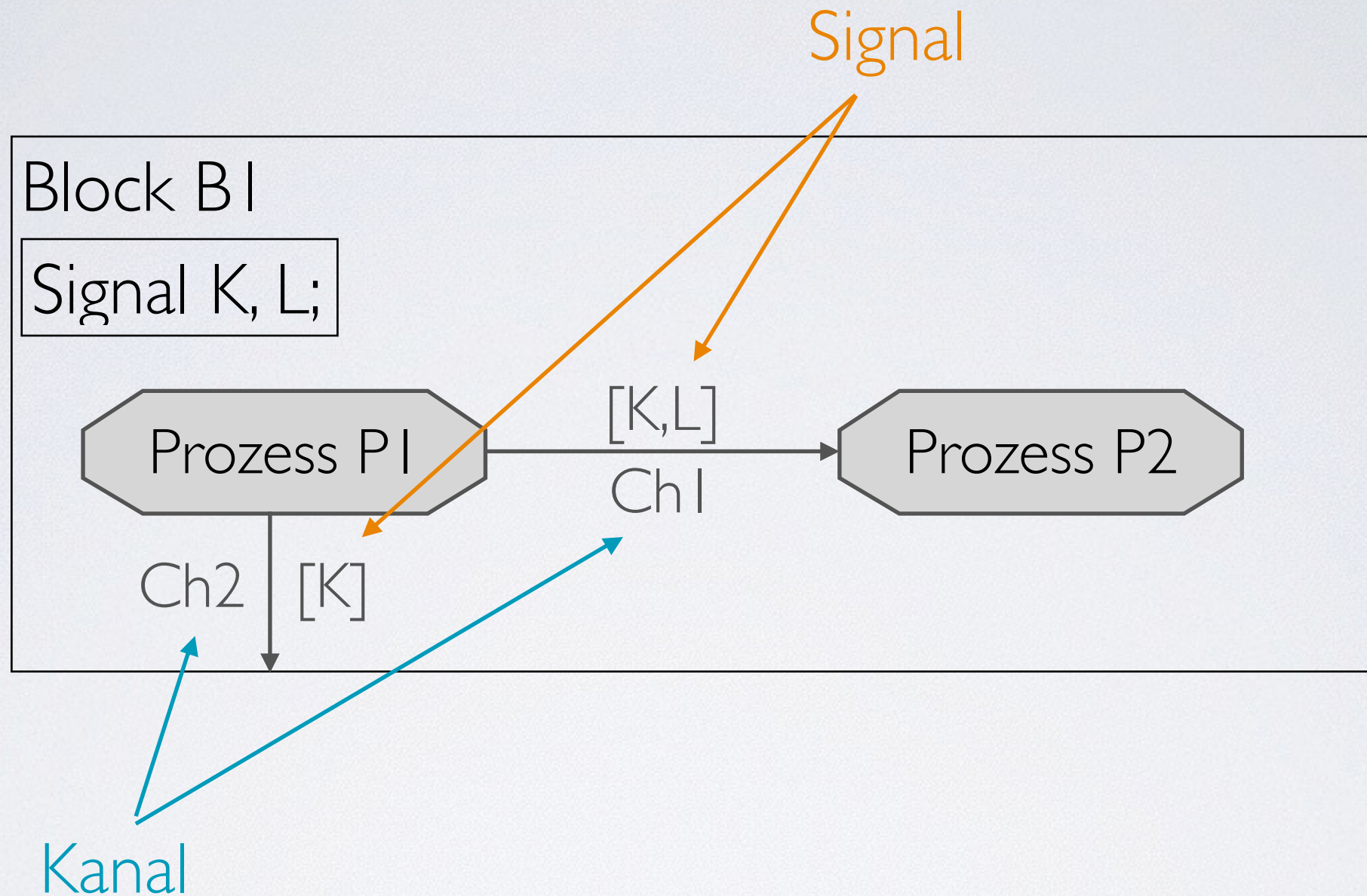
Kann Bestandteil eines Prozesses sein

SDL: Asynchroner Nachrichtenaustausch



FIFO-Verarbeitung,
Prozess prüft, ob Eingabewert passt.

SDL: Prozess-Interaktionsdiagramm

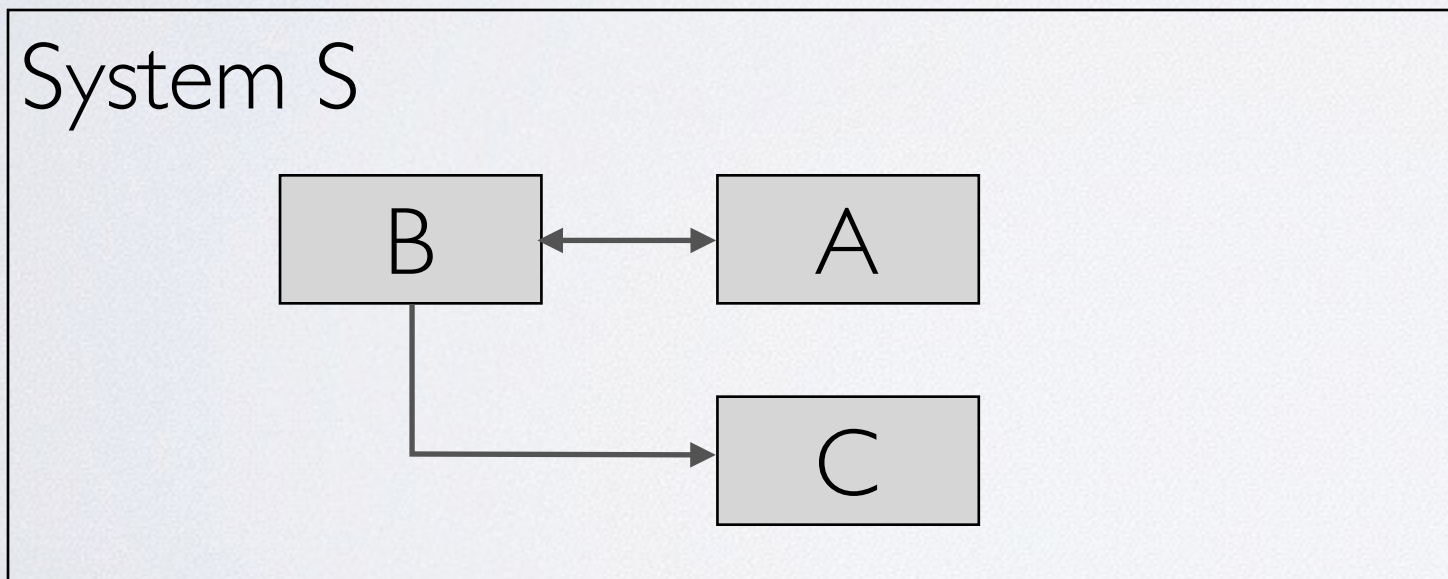
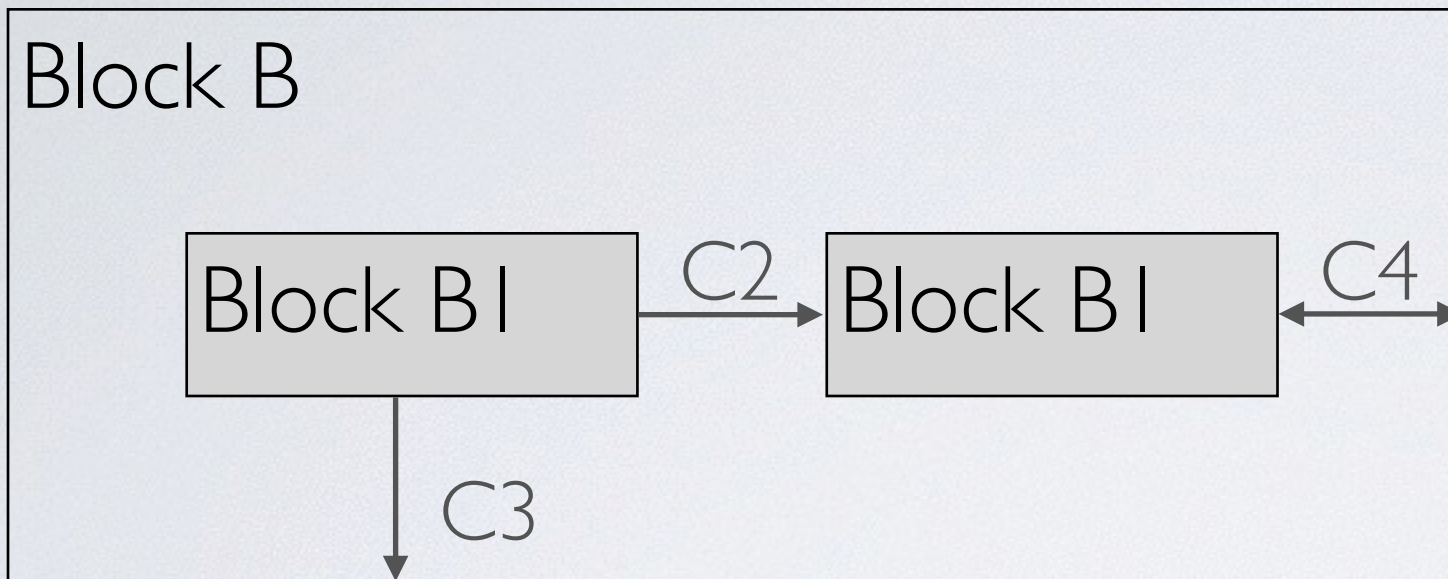


SDL: Angabe des Empfänger

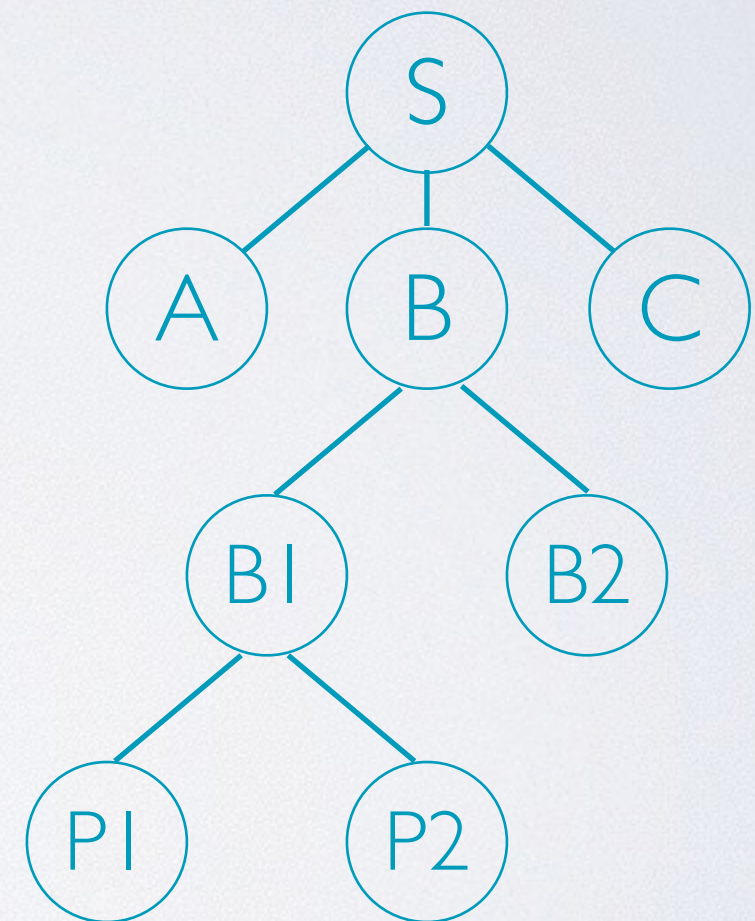
- Durch Prozess-Identifikatoren (auch zu generierten Prozessen/OFFSPRING)
- Explizit
- Implizit



SDL: Hierarchie

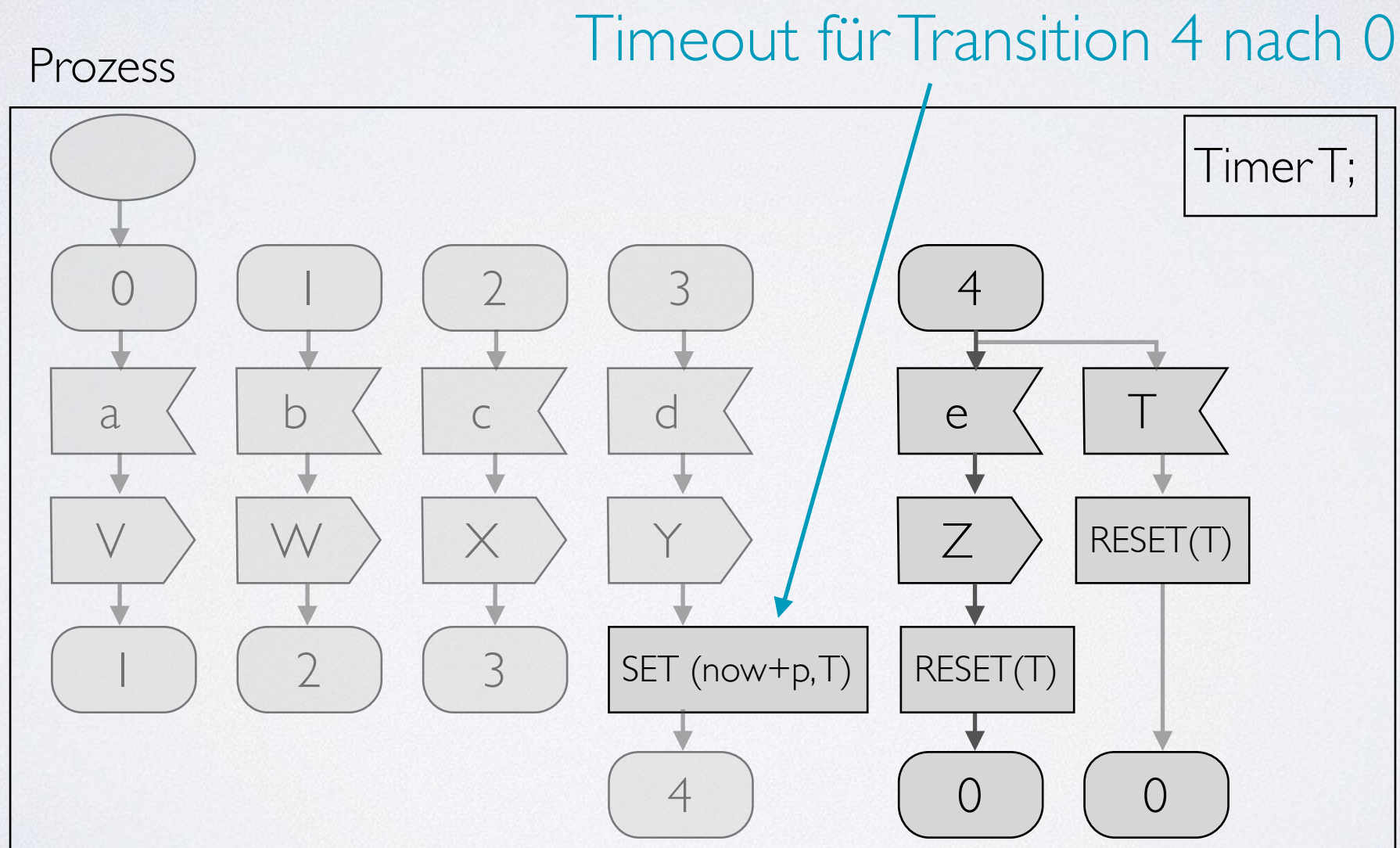


- höchste Ebene: System
- unterste Ebene: Prozess-Interaktion
- kein Verschachteln von Prozessen



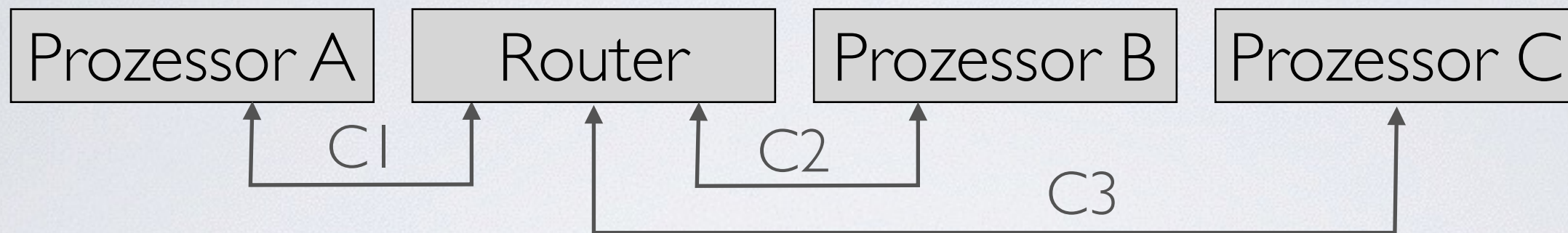
SDL:Timer

- SET definiert Timeout (Einfügen eines Signals in FIFO).
- Signal/Übergang erfolgt bei Zeitablauf, falls vorher keine Signaleingabe.

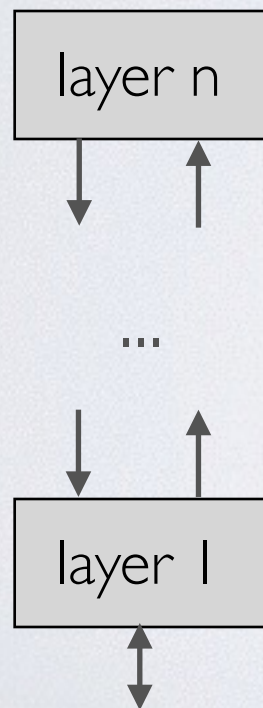


SDL-Beispiel: Computernetzwerk

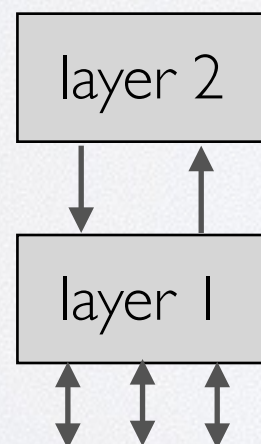
System S



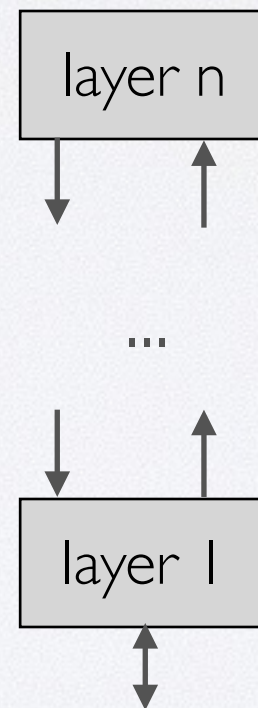
Block Processor A



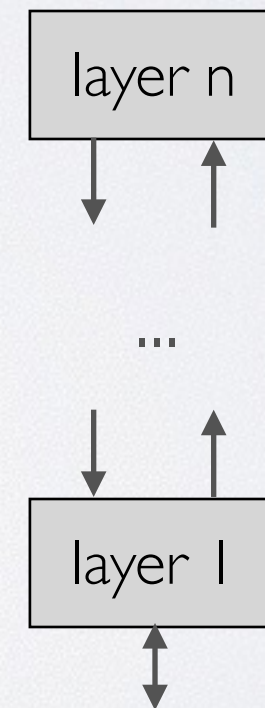
Router



Block Processor B

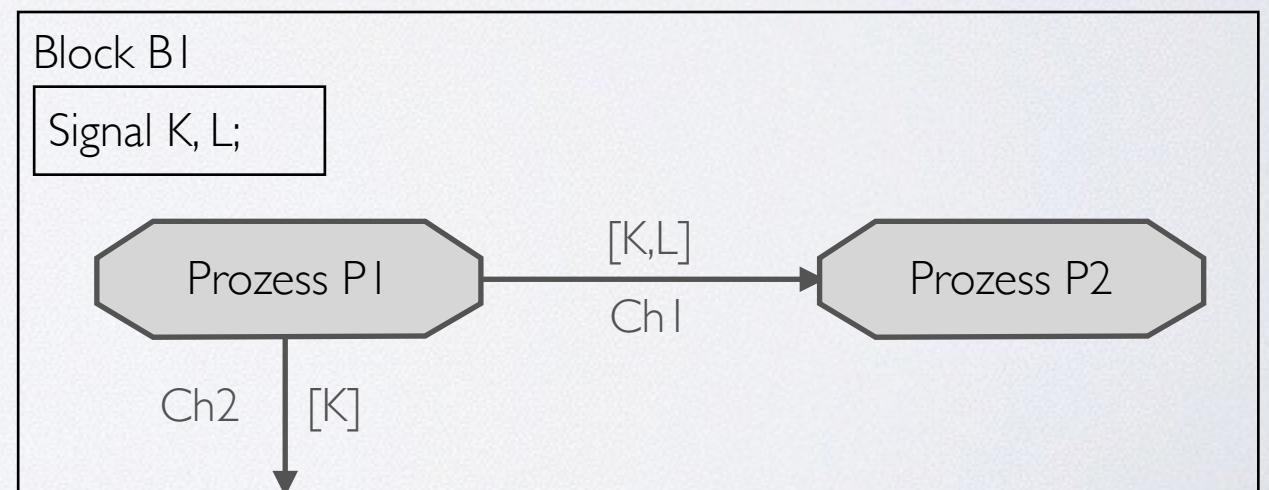
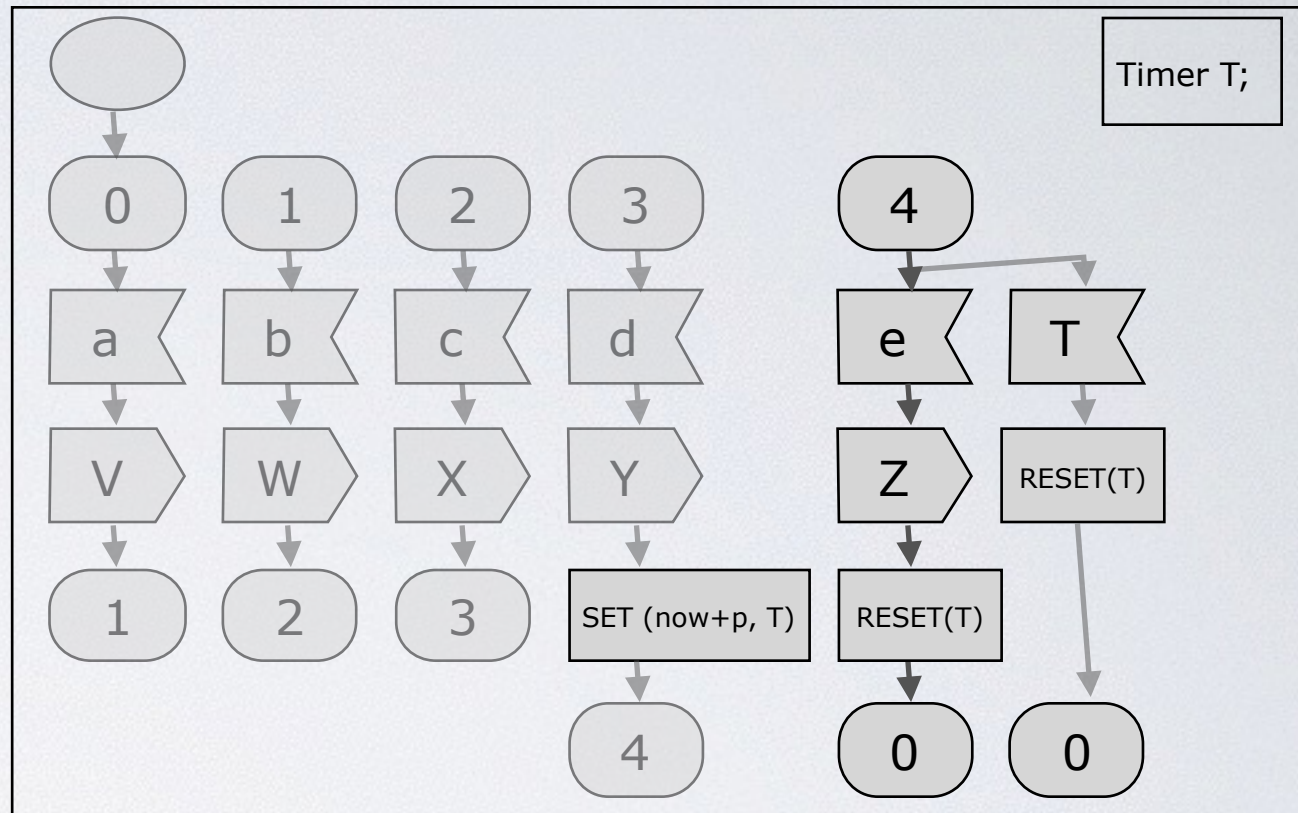


Block Processor C

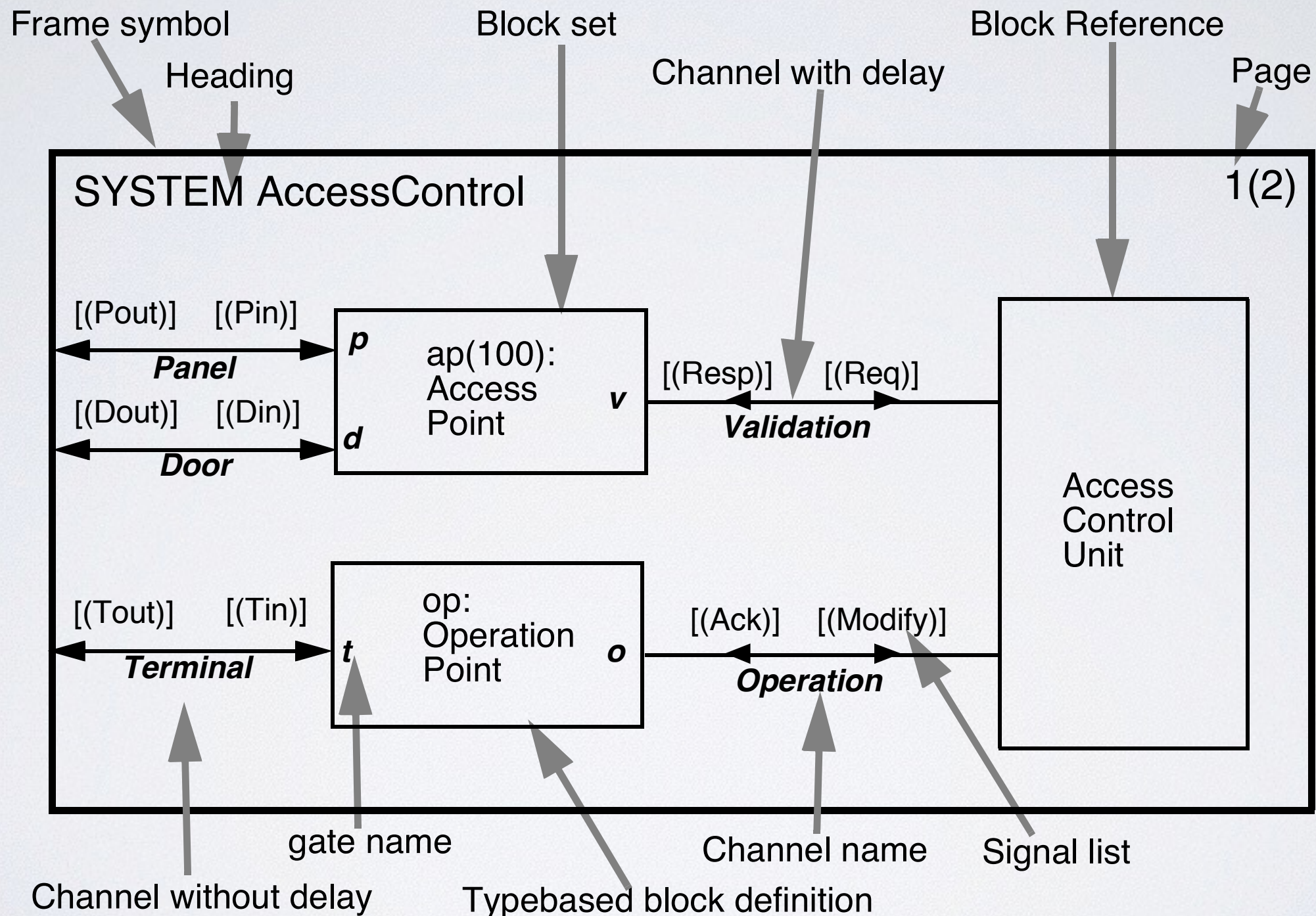


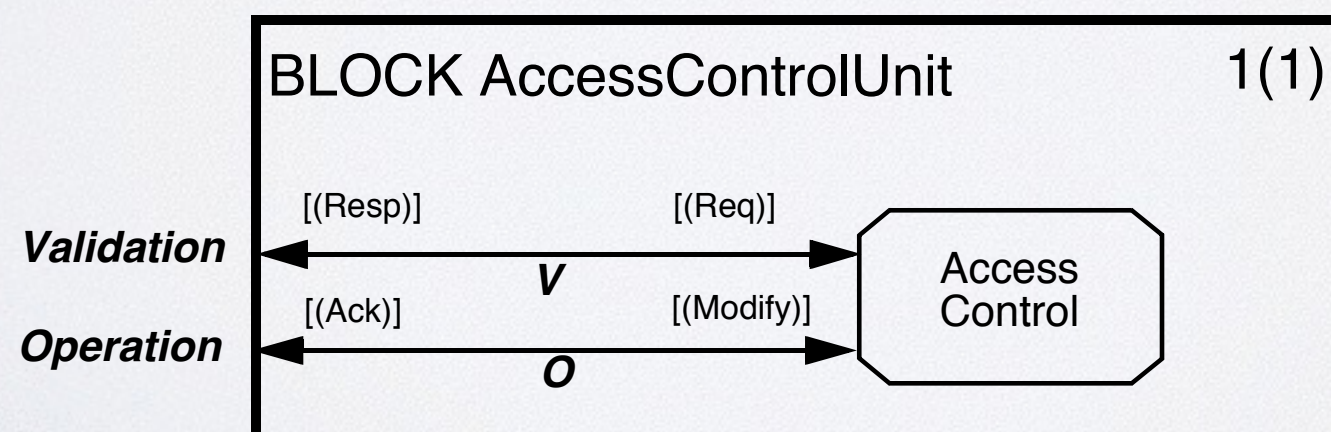
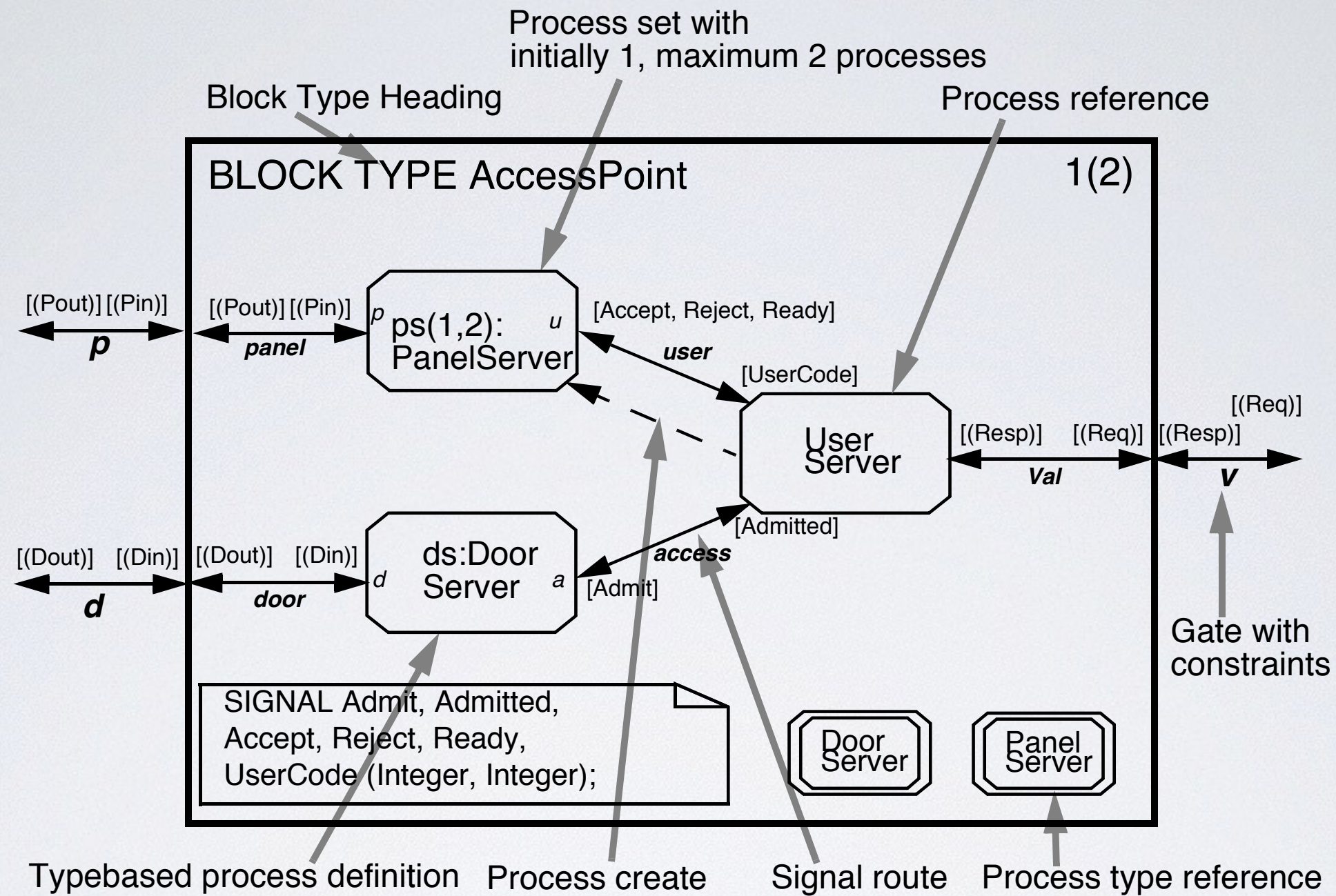
Wiederholung: SDL

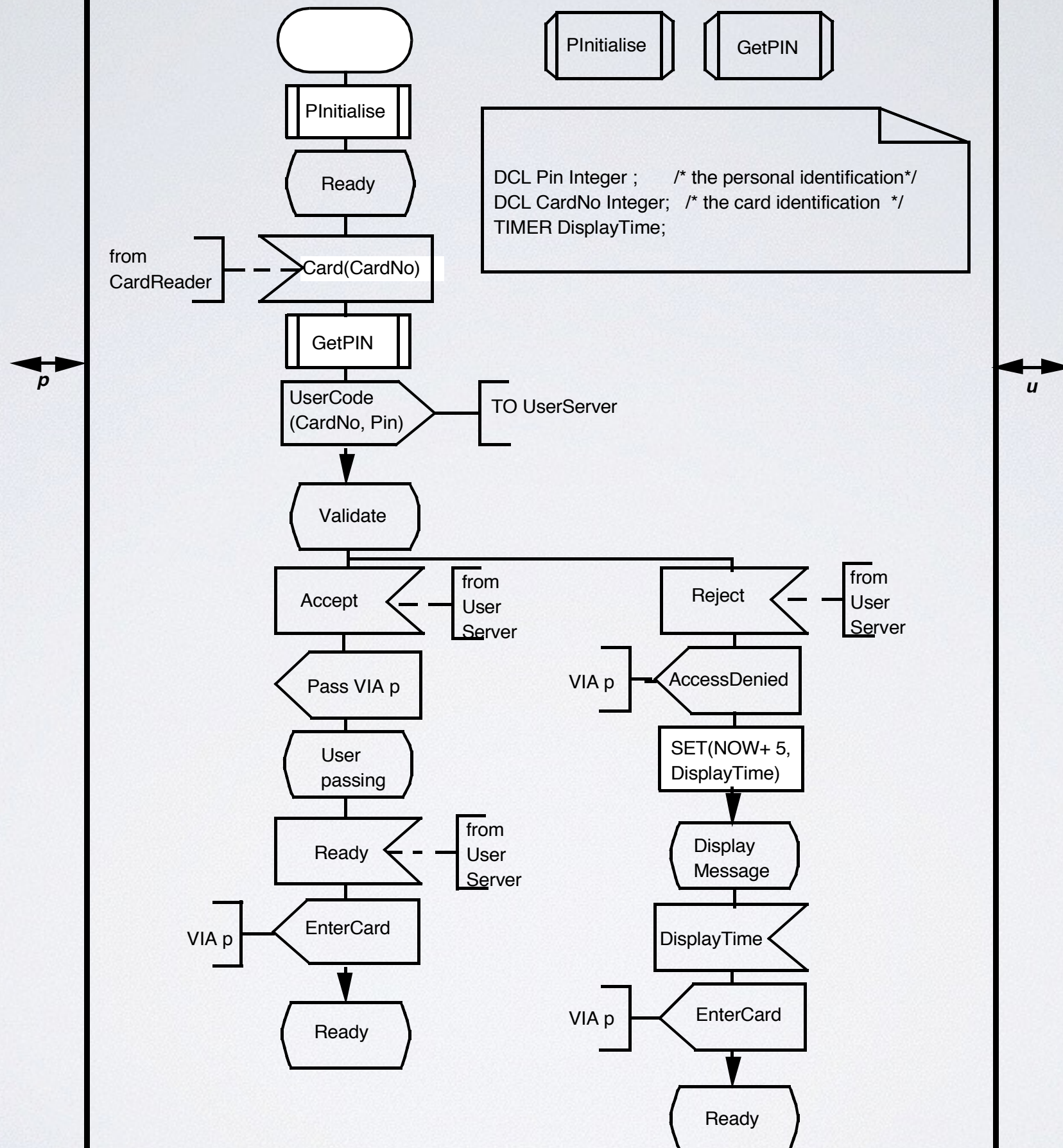
- Basiert auf
 - Zustandsautomaten
 - Asynchronen Nachrichtenaustausch
- Für verteilte Systeme ausgelegt
- Unterstützt graphische und textuelle Repräsentation

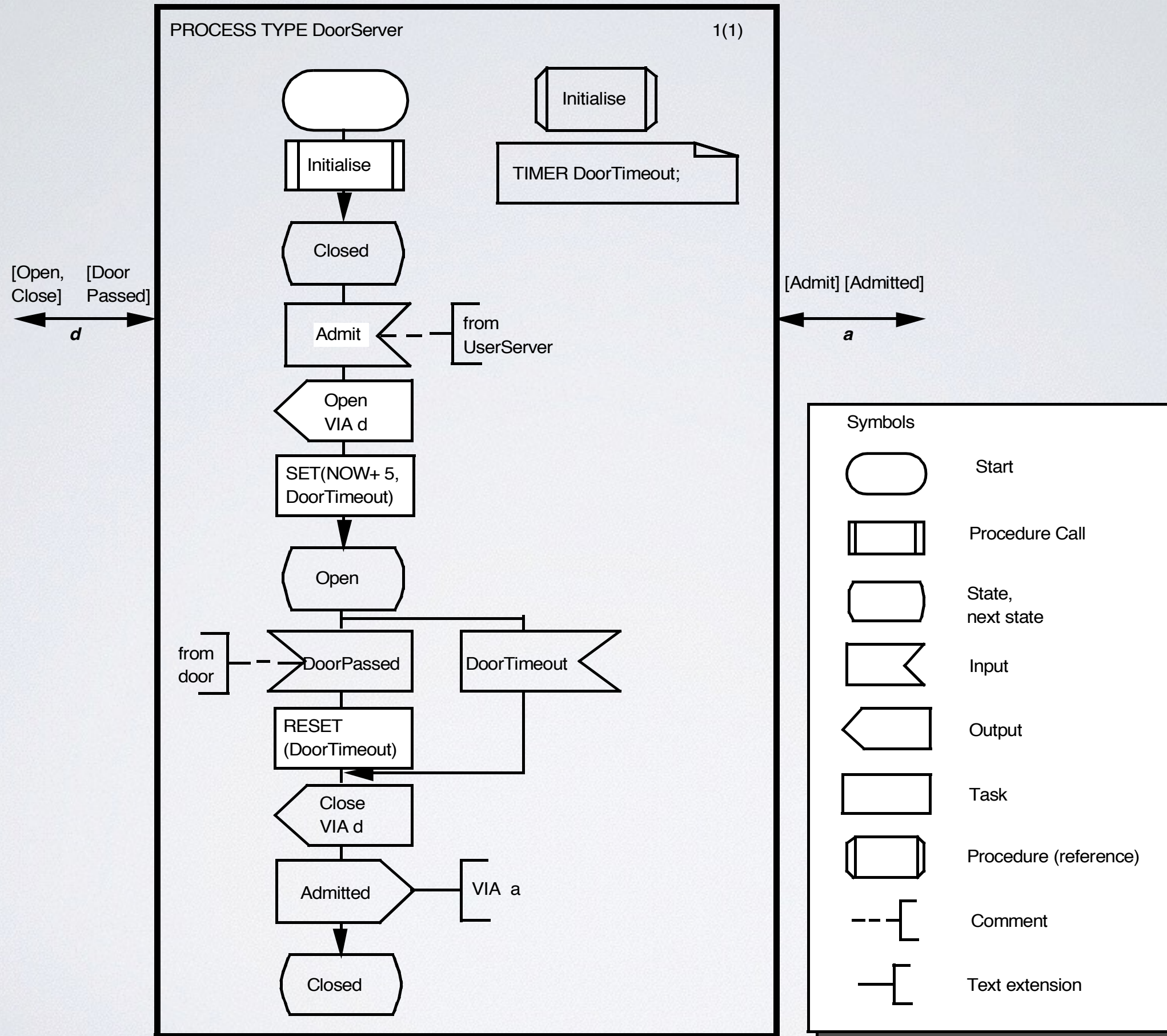


SDL-Beispiel: Zugangskontrolle









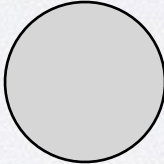





SDL: Zusammenfassung

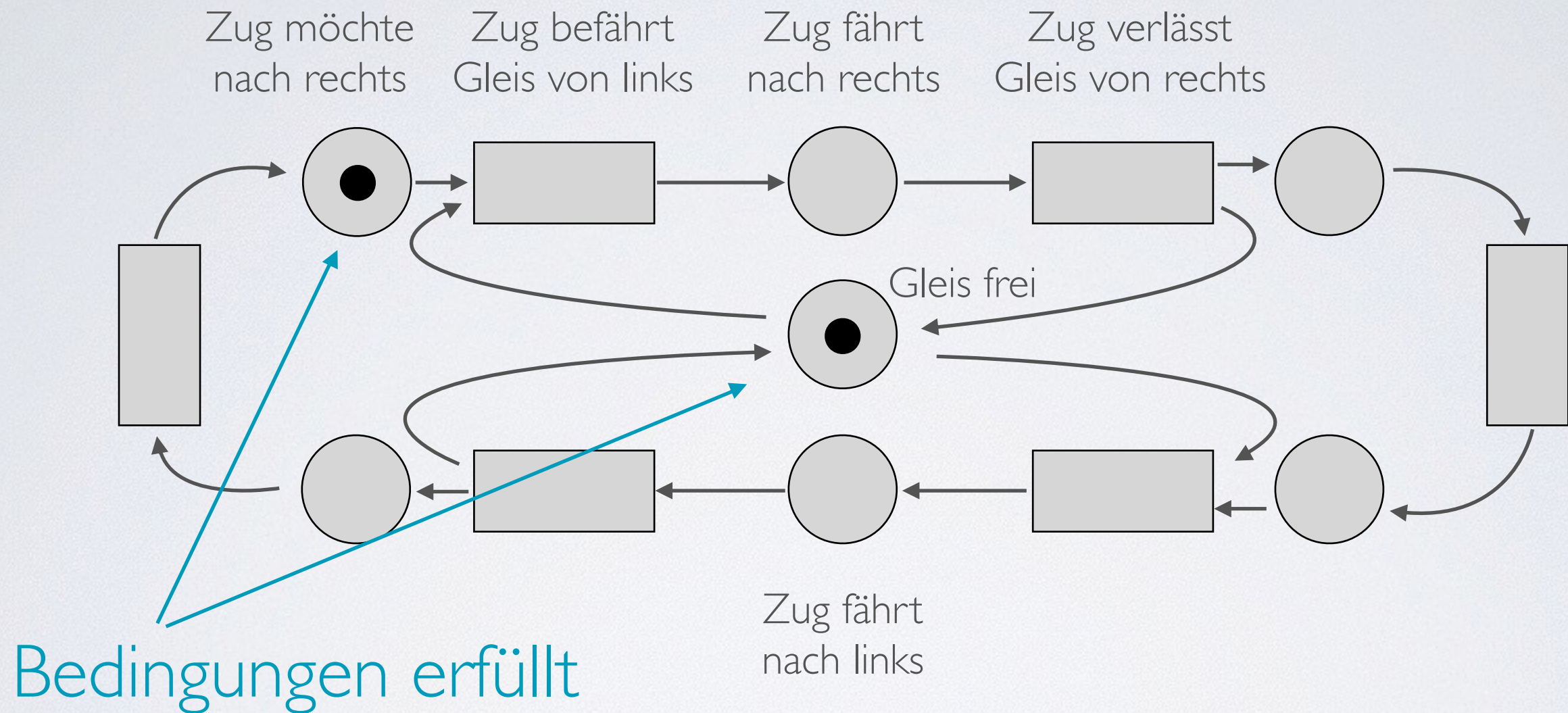
- Modell eines Zustandsautomaten (endlichen Automaten)
- Asynchroner Nachrichtenaustausch (non-blocking)
- Vorteile:
 - Ausgelegt für verteilte Systeme
 - Werkzeuge verfügbar (s. <http://www.sdl-forum.org>)
- Nachteile:
 - Reihenfolge der Signalabarbeitung nicht spezifiziert (nicht deterministisch)
 - Obere Schranke für FIFOs schwer bestimmbar
 - Nur weiche Zeitbedingungen möglich

Petrinetze

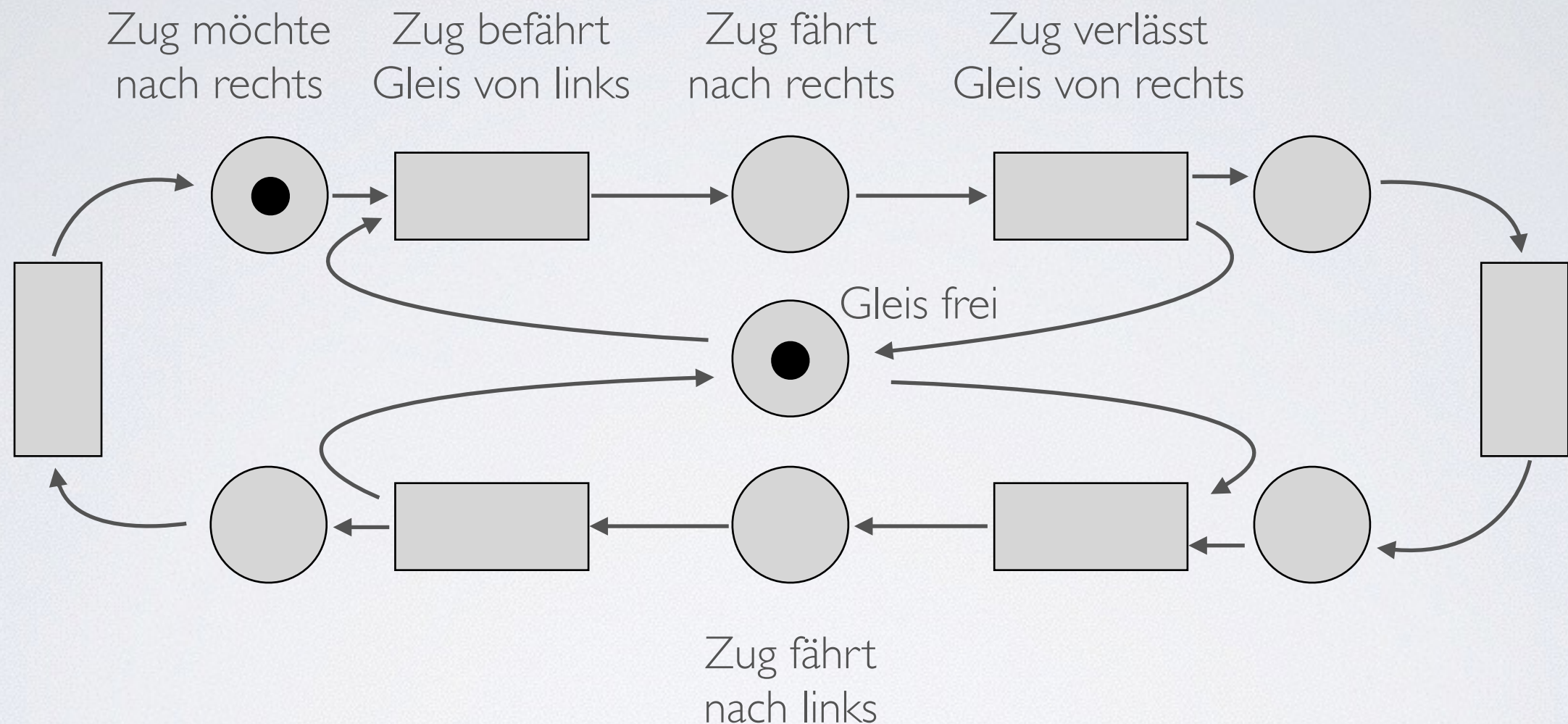
Petrinetze

- Benannt nach Carl Adam Petri (Doktorarbeit 1962)
- Geeignet für verteilte Systeme
- Formale Prüfung möglich
- Kernelemente:
 - Bedingungen (oder Stellen) 
 - Ereignisse (oder Transitionen) 
 - Flussrelationen (oder Kanten) 
 - Marken (oder Token) 

Petrinetz-Beispiel: Eingleisiger Streckenabschnitt

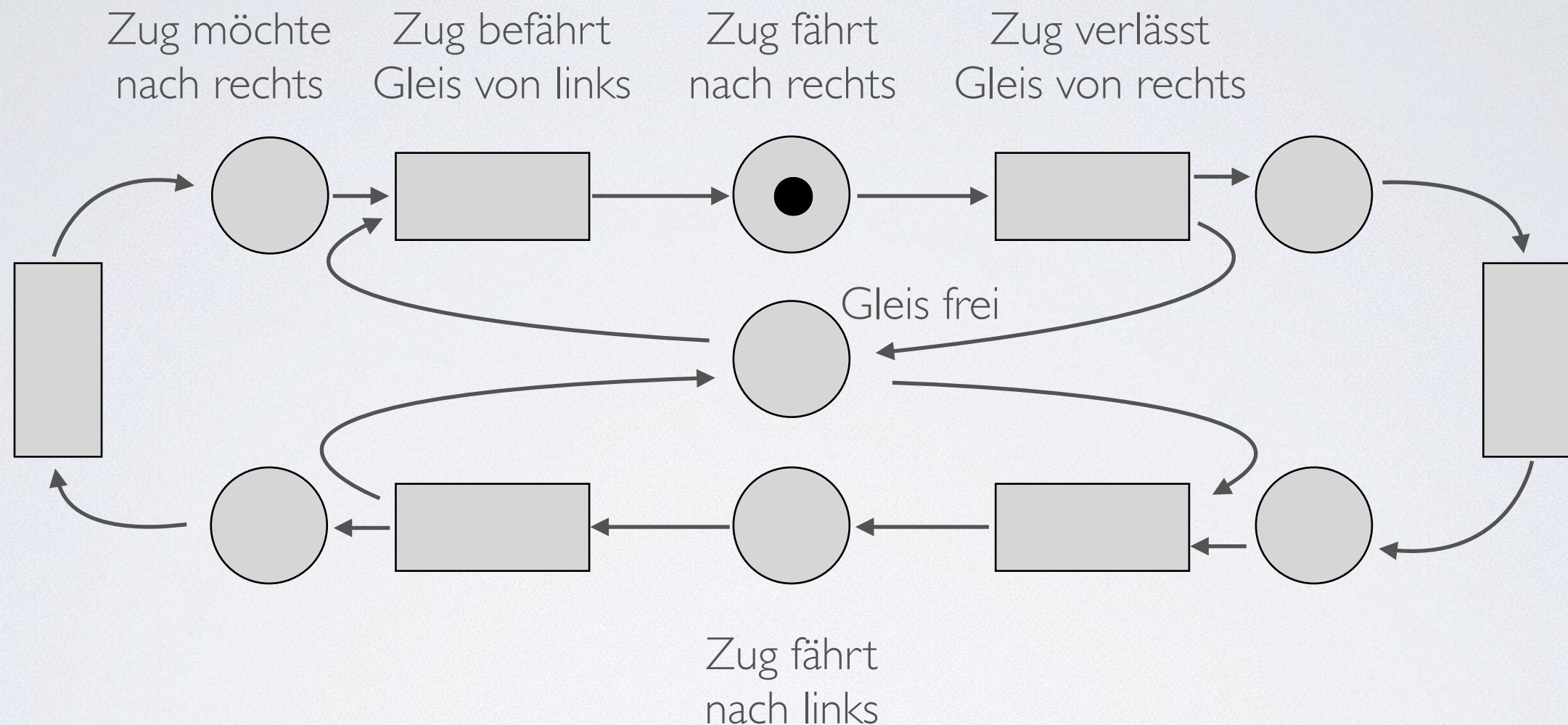


Petrinetz-Beispiel: Eingleisiger Streckenabschnitt



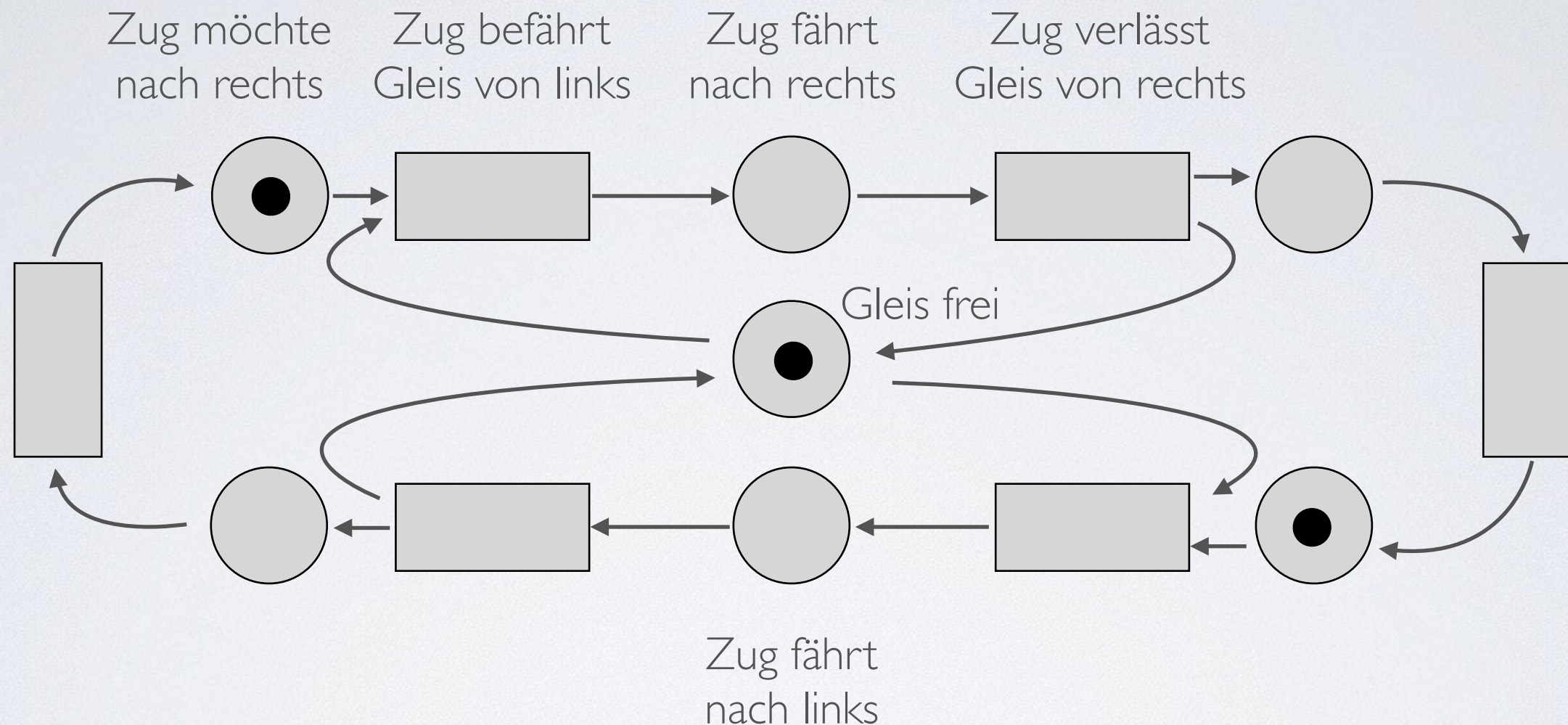
Gleis ist frei, Zug kann fahren.

Petrinetz-Beispiel: Eingleisiger Streckenabschnitt



Gleis ist belegt, Zug verlässt Gleis.

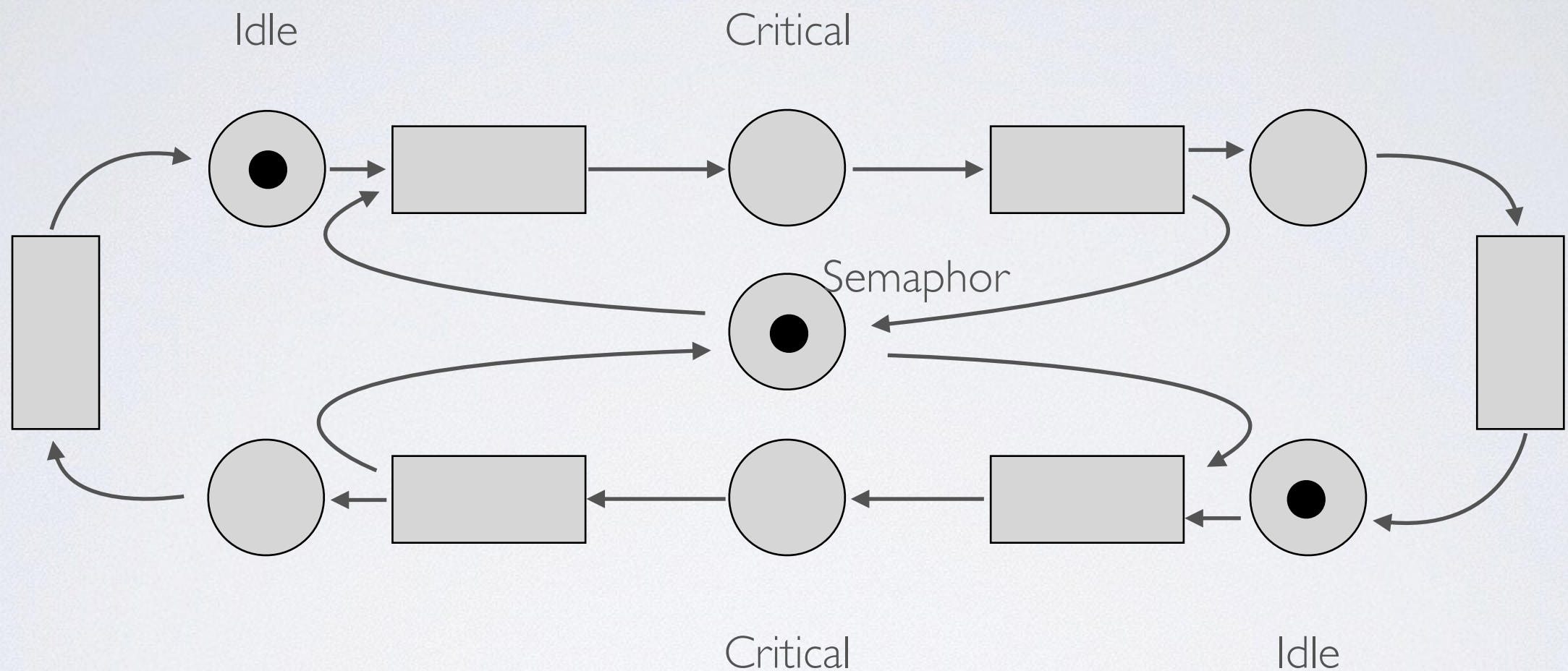
Petrinetz-Beispiel: Eingleisiger Streckenabschnitt



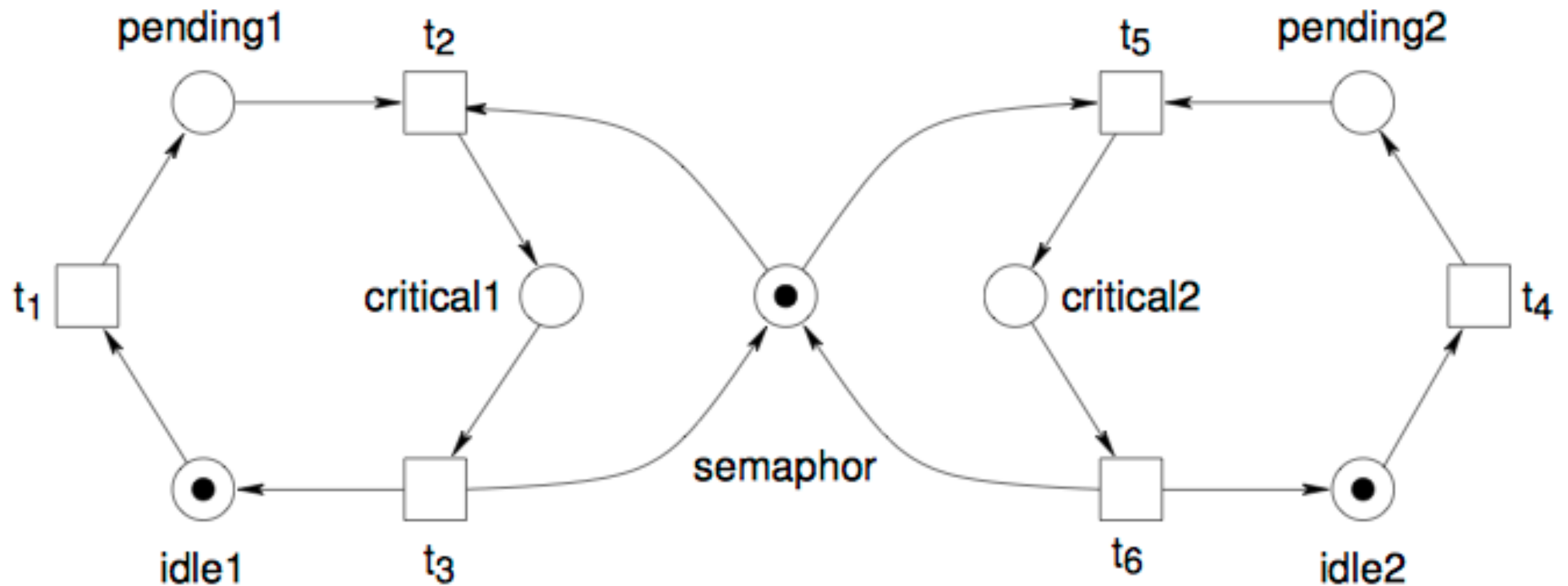
Konfliktsituation!

Beispiel ähnlich zu Zugriff auf gemeinsamen Speicher.

Petrinetz-Beispiel: Semaphore



Petrinetz-Beispiel: Semaphore

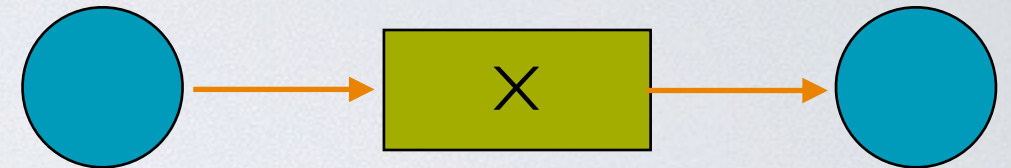


Petrinetze: Arten

- Bedingungs-/Ereignisnetz
 - Höchstens eine Marke pro Bedingung
- Stellen-/Transitionennetz
 - Mehrere Marken pro Bedingung möglich
 - Kantengewichtungen
- Prädikat-/Ereignisnetz
 - Marken sind identifizierbar
 - Bedingungen sind Variablen/Funktionen zuordenbar

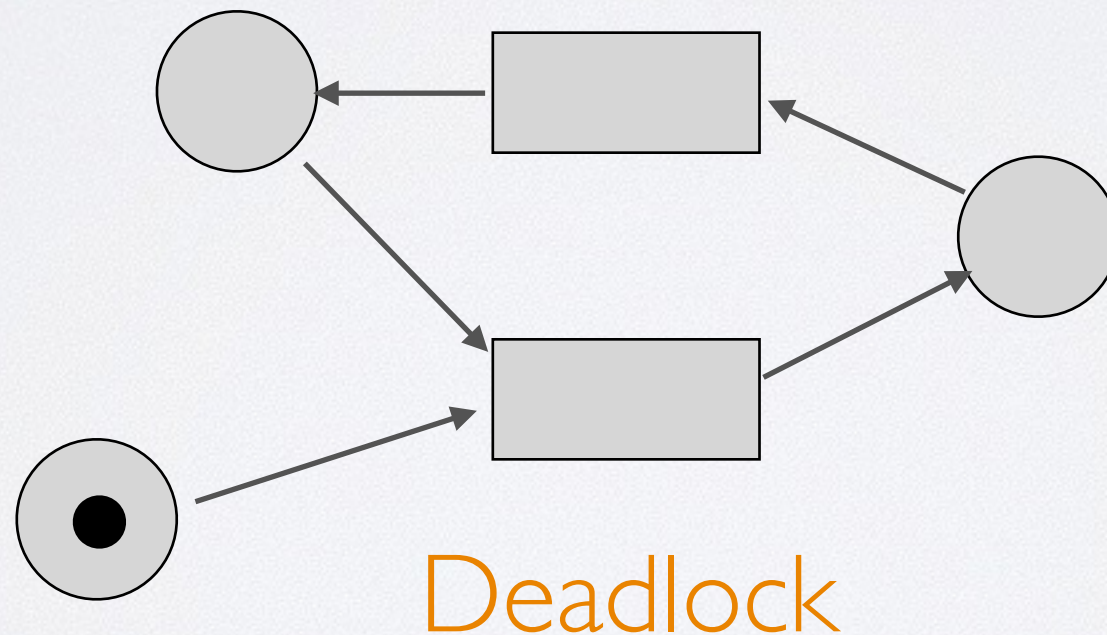
Bedingungs-/Ereignisnetz: Formale Beschreibung

- Netz $N = (C, E, F)$ mit:
 - C und E sind disjunkte Mengen.
 - $F \subseteq (E \times C) \cup (C \times E)$ ist Flussrelation.
- C ist Menge der Bedingungen.
- E ist Menge der Ereignisse.
- $x \in (C \cup E)$: Vorbedingungen $\{y | yFx\}$, Nachbedingungen $\{y | xFy\}$



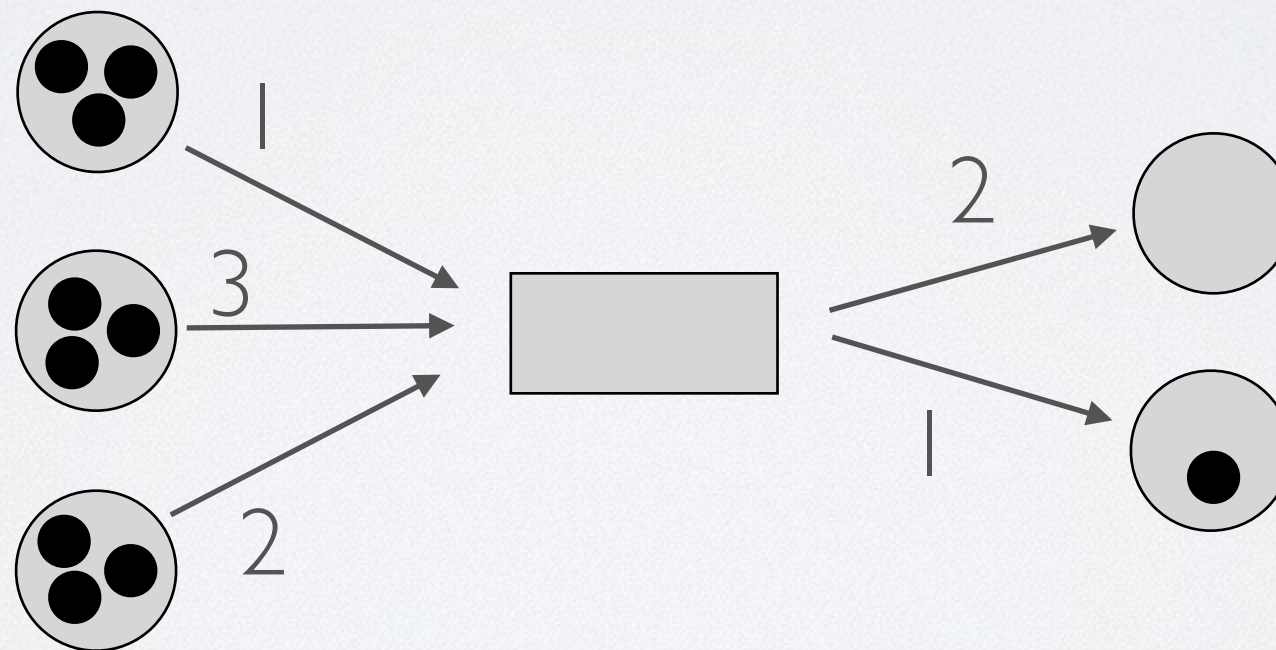
Petrinetz: Formale Beschreibungen

- Verwendung zur Herleitung von Eigenschaften:
 - Erreichbarkeit (ist Zustand von Anfangszustand erreichbar?)
 - Lebendigkeit (kann weiter geschaltet werden?)



Stellen-/Transitionennetz

- Schalten nur, wenn
 - Vorbedingung erfüllt, bei Anzahl Marken \geq Kantengewicht,
 - Nachbedingung erfüllt, Kapazität für Marken ausreichend
- Schalten nicht zwangsläufig (nicht deterministisch, wenn mehrere aktiviert)

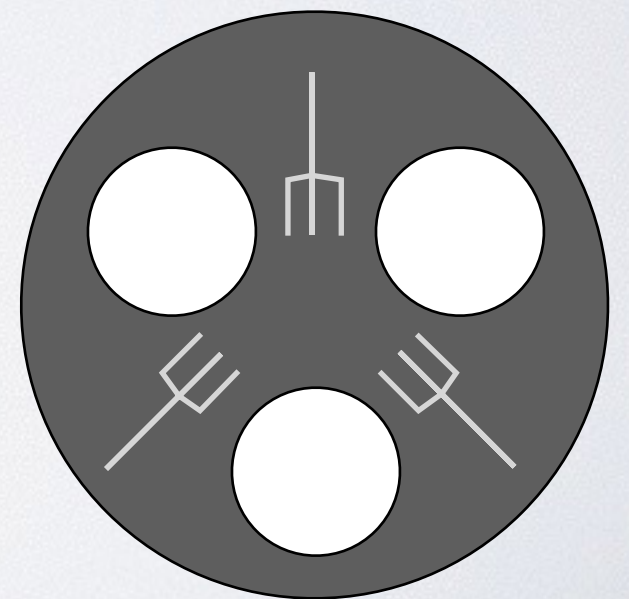


Prädikat-/Ereignisnetz

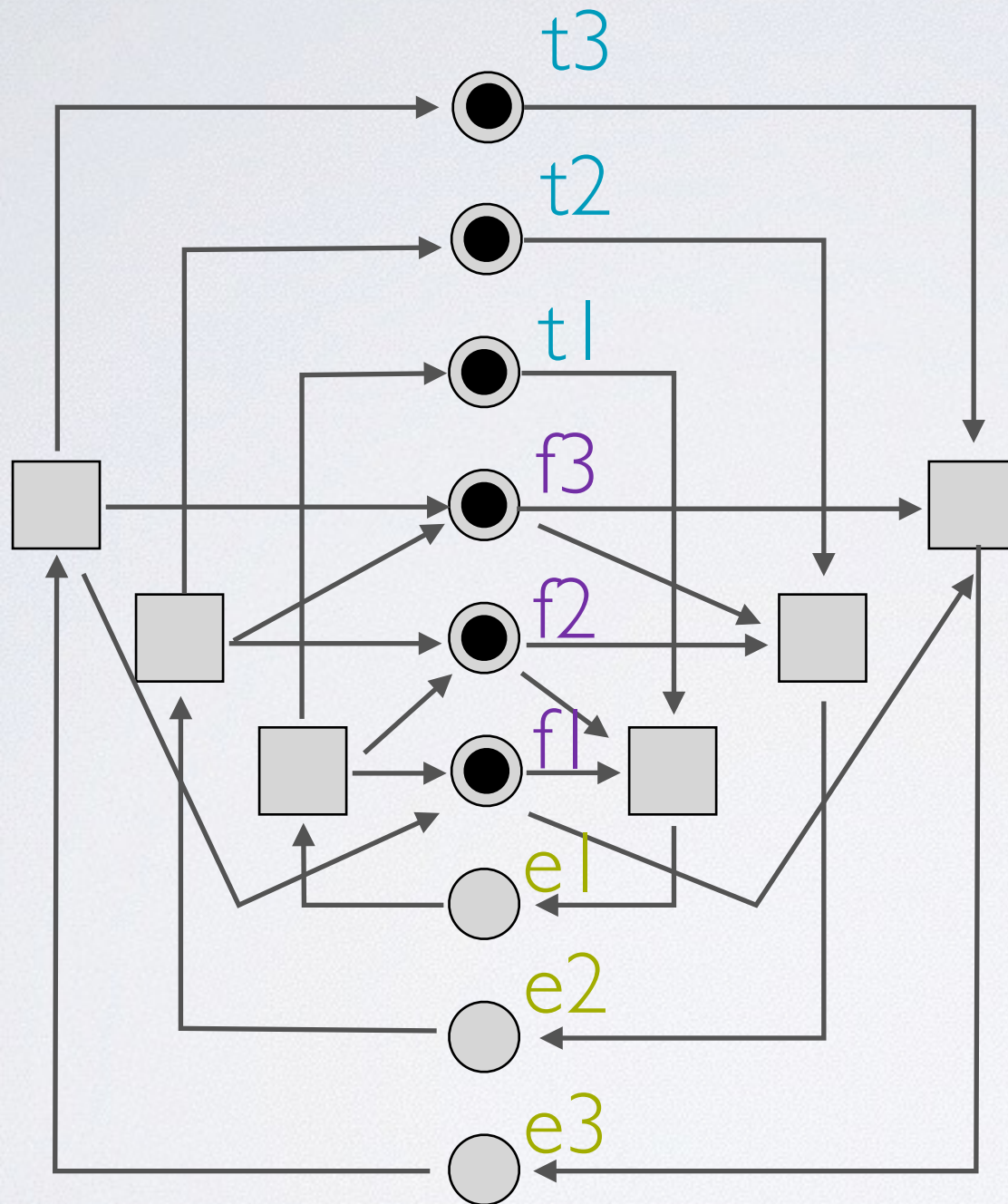
- erreichen Verringerung der Netzgröße
- unterscheidbare Marker
- Bedingungen mit Funktionen versehen

Veranschaulichung am “Philosophenproblem”:

- $n > 1$ Philosophen am runden Tisch
- n Gabeln und n Teller
- Philosoph kann denken oder essen
- Philosoph benötigt zum Essen beide Gabeln neben Teller



Philosophenproblem: Bedingungs-/Ereignisnetz



$x \in \{1, 2, 3\}$

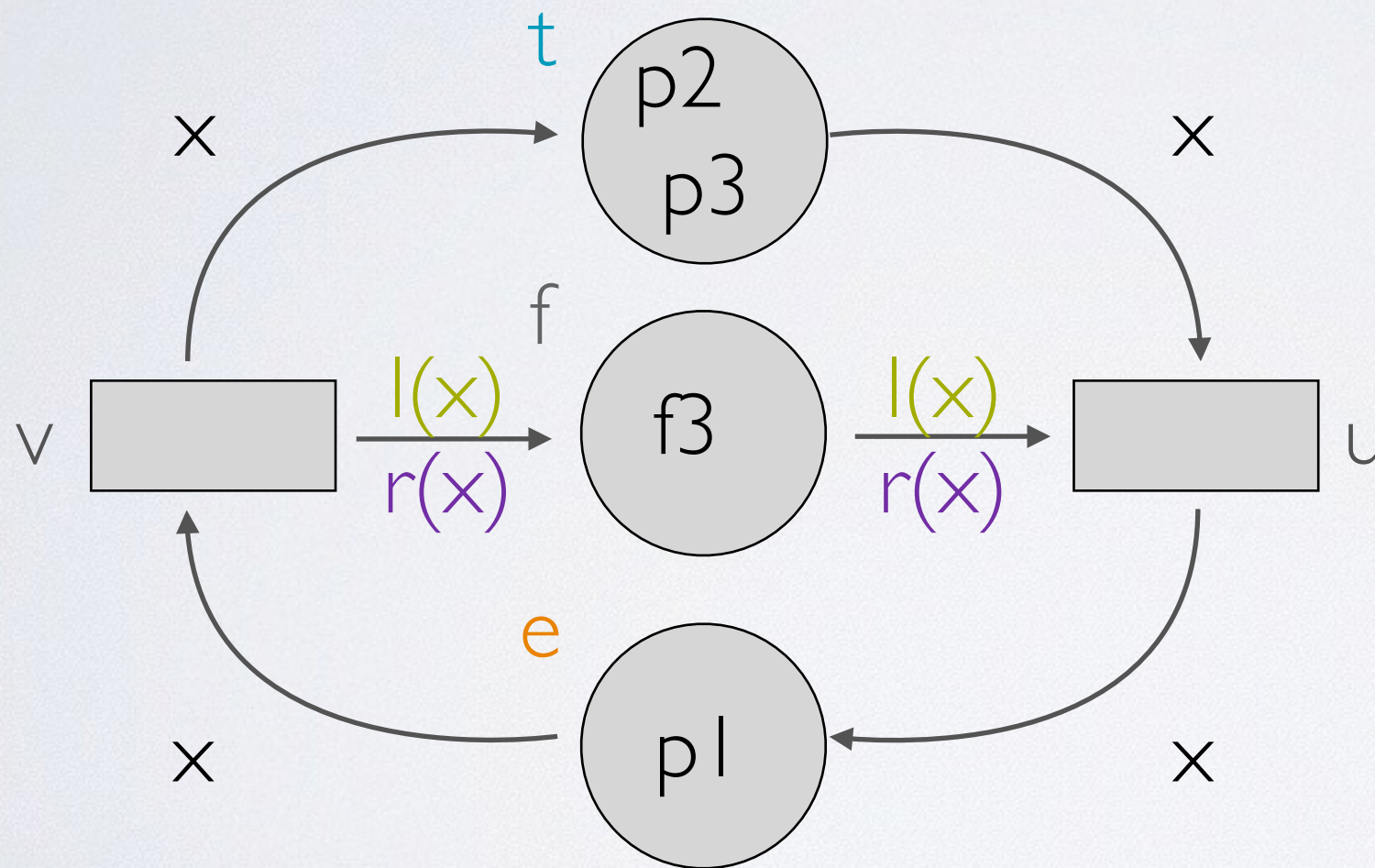
tx : x denkt

ex : x isst

fx : Gabel x ist frei

Philosophenproblem: Prädikat-/Ereignisnetz

- Variablen für Philosophen ($p1, p2, p3$) und Gabeln ($f1, f2, f3$)



x : Philosoph
 l : linke Gabel
 r : rechte Gabel
 t : denkt
 e : isst
 f : Gabel ist frei

Petrinetz-Beispiel: Klimaanlage

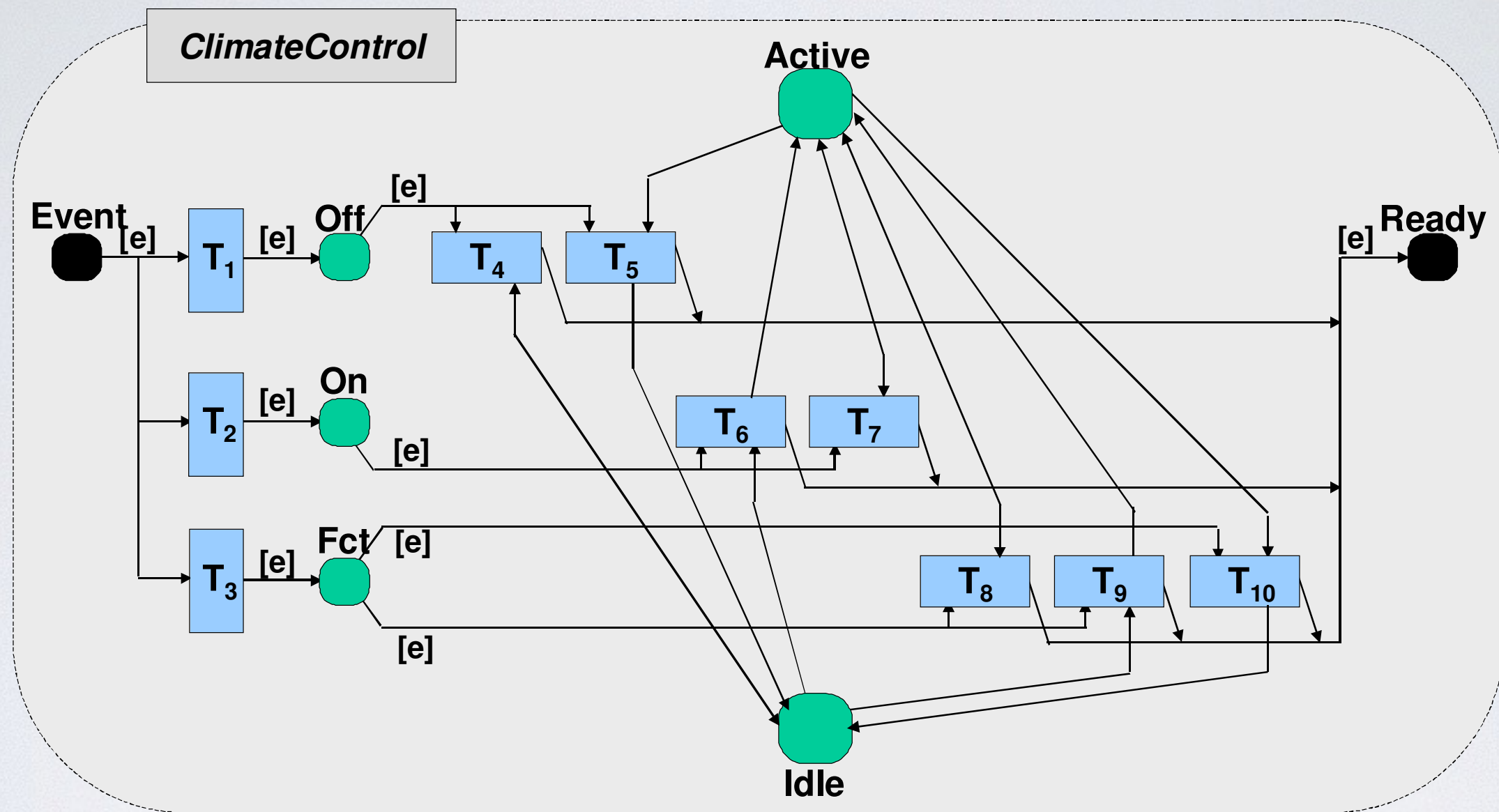
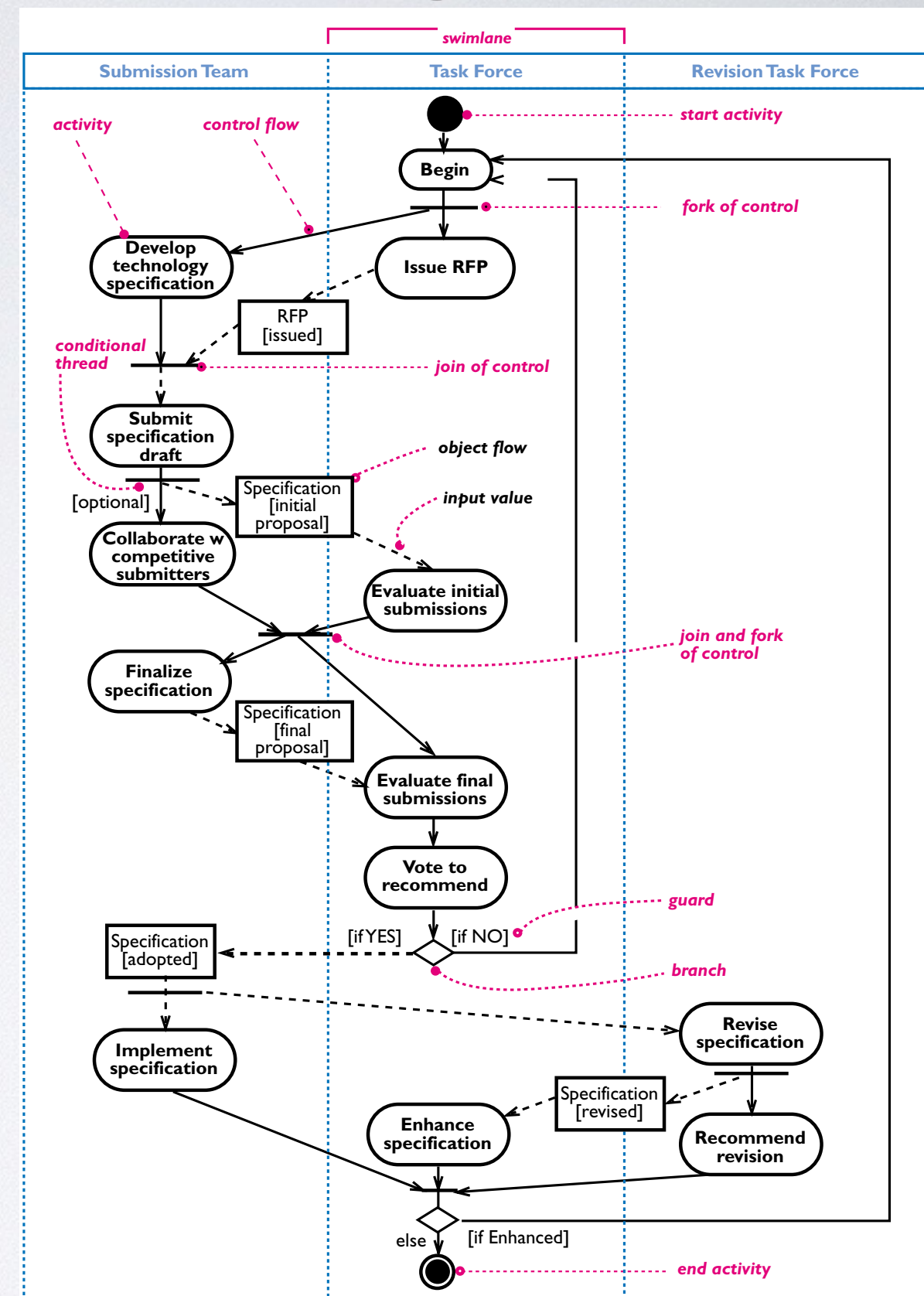


Figure 3: Specification of Climate Control

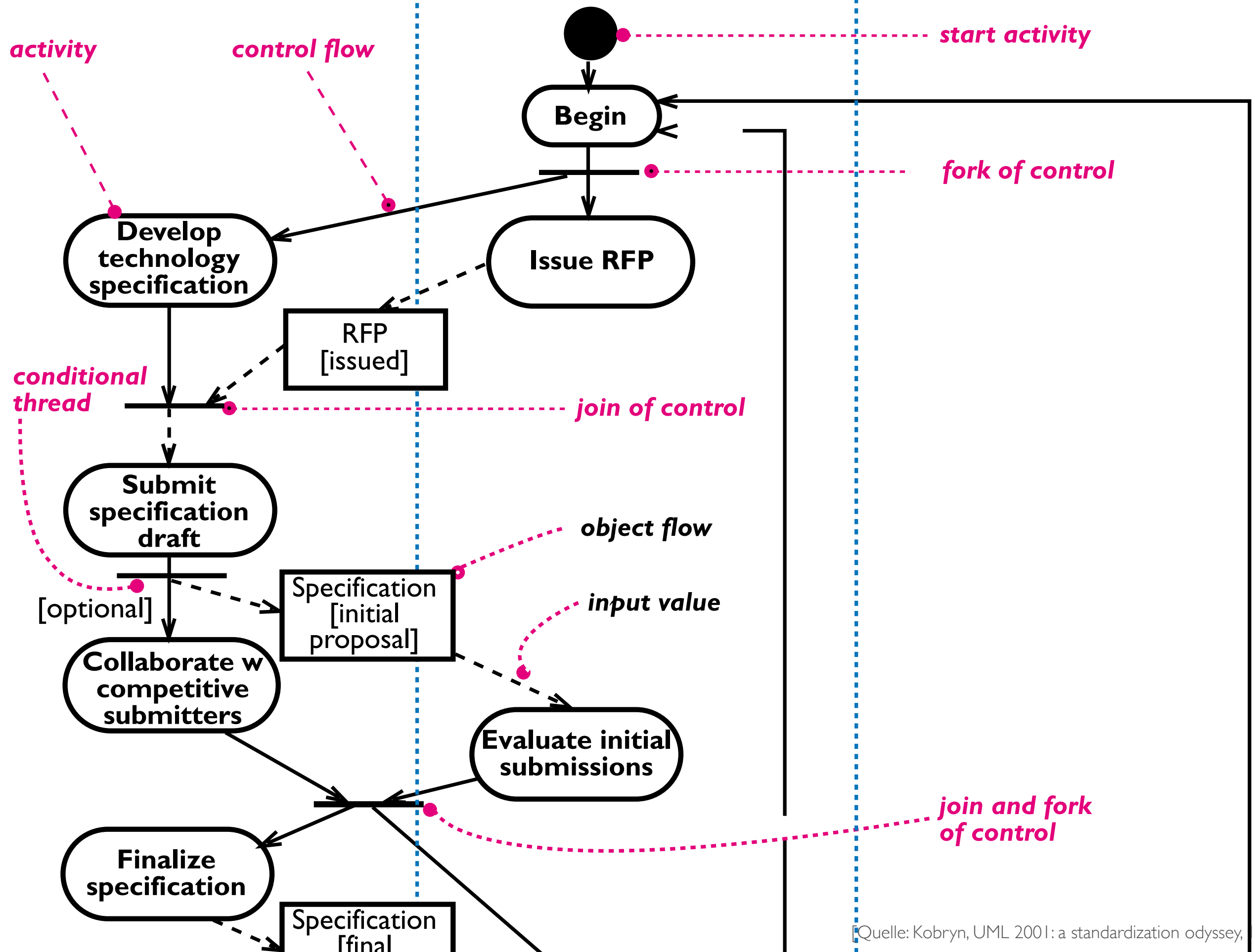
[Quelle: Rust et al., 2002]

Petrinetz-Erweiterungen

- UML Aktivitätsdiagramme können als Erweiterungen von Petrinetze angesehen werden.
- Horizontale Balken stehen für Transitionen/Ereignisse.



[Quelle: Kobryn, UML 2001: a standardization odyssey, 1999]



Petrinetze: Zusammenfassung

- Vorteile:
 - Geeignet für verteilte Systeme
 - Formale Beweise von Eigenschaften möglich
- Nachteile:
 - Keine Unterstützung von Zeitbedingungen
 - Kein Hierarchiekonzept
- UML Aktivitätsdiagramme sind prinzipiell erweiterte Petrinetze

Wenig bekannte Anwendung: Intelligente Toilette

- Embedded System am unerwarteten Platz
- Steuerung von Geruchsabsaugung und Reinigung
- Regelung von Druck- und Temperatursensoren, Ventilen, Boiler, Fön
- Stand-by-Betrieb



[Quelle: M. Schmid, Programmierung per Blockschtbild, Computer & Automation | 4.02.2012]

Literatur / Quellen

- Rolv **Braek**, SDL basics, Computer Networks and ISDN Systems, Vol. 28, Juni 1996, S.1585-1602, Elsevier
- A. **Thums**, G. Schellhorn, F. Ortmeier, W. Reif, Interactive verification of statecharts. In H. Ehrig, editor, Integration of Software Specification Techniques for Applications in Engineering, pages 355 – 373. Springer LNCS 3147, 2004
- Cris **Kobryn**, UML 2001, UML 2001: a standardization odyssey, CACM, Vol. 42, Oktober 1999, S. 29-37
- M. **Schmid**, Programmierung per Blockschaltbild, Computer & Automation 14.02.2012, URL: http://www.computer-automation.de/steuerungsebene/industrie-pc/fachwissen/article/85810/0/Programmierung_per_Blockschaltbild/
- Peter **Marwedel**, Eingebettete Systeme, Springer-Verlag, 2008
- C. **Rust**, F. **Stappert**, R. **Bernhardi-Grisson**, Petri Net Based Design of Reconfigurable Embedded Real-Time Systems, In Kleinjohann, Kim, Kleinjohann, Rettberg, editors, Design and Analysis of Distributed Embedded Systems, pages 41–50. Kluwer Academic Publishers, 2002
- Wikipedia, Anti-lock braking system, URL: http://en.wikipedia.org/wiki/Anti-lock_braking_system
- **Stand aller Internetquellen: 27.04.2012**