LE 12
1

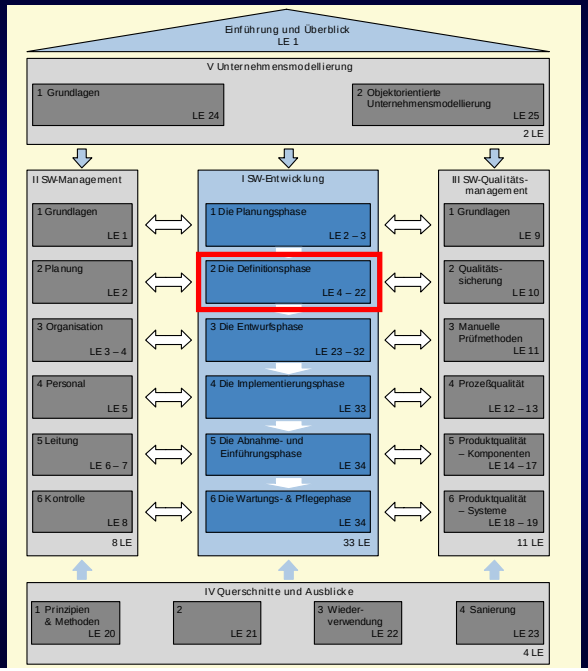
Software-Technik

2 Die Definitionsphase
Zustandsorientierte Sicht 2
Petri-Netze

Prof. Dr. Helmut Balzert
Lehrstuhl für Software-Technik
Ruhr-Universität Bochum

© Helmut Balzert 2001

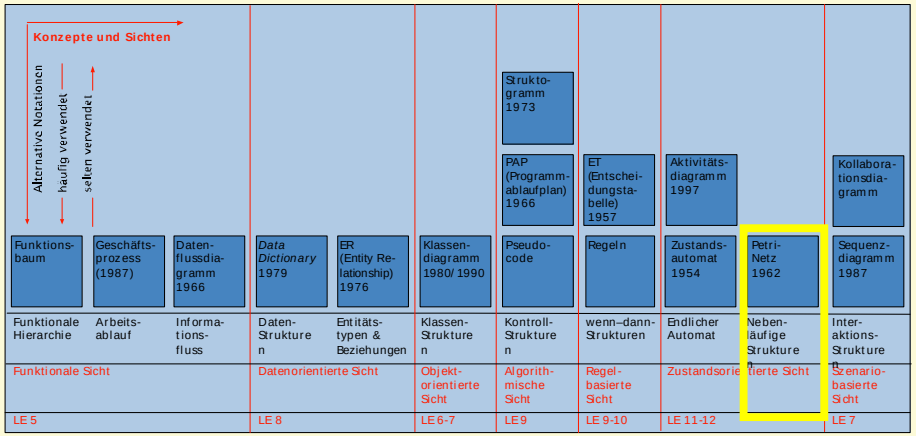
LE 12
2



LE 12
3

Inhalt

- **Basiskonzepte: Zustandsorientierte Sicht**



LE 12
4

Inhalt

2.17 Petri-Netze
- 2.17.1 Grundlagen
- 2.17.2 Bedingungs/Ereignis-Netze
- 2.17.3 Stellen/Transitions-Netze
- 2.17.4 Prädikat/Transitions-Netze
- 2.17.5 Hierarchische Petri-Netze
- 2.17.6 Zeitbehaftete Petri-Netze
- 2.17.7 Strukturelemente und Strukturen von Petri-Netzen
- 2.17.8 Methodik
- 2.17.9 Analyse und Simulation von Petri-Netzen
- 2.17.10 Wertung
- 2.17.11 Einsatz von CASE-Werkzeugen.

LE 12 5

2.17 Petri-Netze

- Zur Historie
  - Prof. Dr. Carl Adam Petri
    *12.7.1926 in Leipzig
    Professor an der Universität Hamburg
  - Erfinder der Petri-Netze
  - Entwicklung einer allgemeinen Theorie diskreter Systeme, die auf den Konzepten der Nebenläufigkeit, der Verteiltheit und der asynchronen Kommunikation aufbaut.

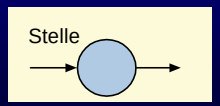
LE 12 6

2.17.1 Grundlagen

- Petri-Netz
  - Gerichteter Graph, der aus zwei verschiedenen Sorten von Knoten besteht:
    - Aus Stellen und Transitionen

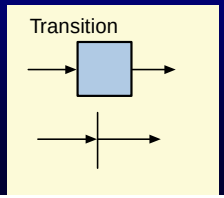
Stellen
(Plätze, Zustände)

entspricht einer Zwischenablage von Informationen



Transitionen
(Hürden, Zustandsübergänge)

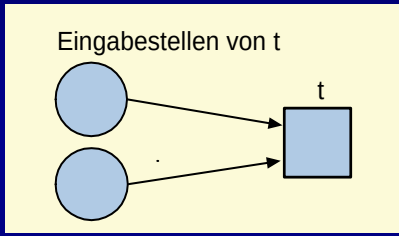
beschreibt die Verarbeitung von Informationen

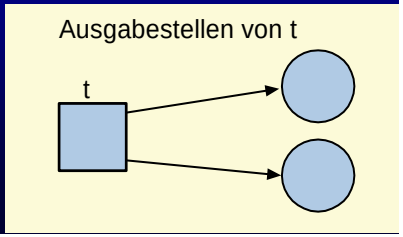


LE 12 7

2.17.1 Grundlagen

- Semantik
  - Kanten dürfen jeweils nur von einer Sorte zur anderen führen
  - Stellen, von denen Kanten zu einer Transition t laufen, heißen Eingabestellen von t
  - Stellen, zu denen – von einer Transition t aus – Kanten führen, heißen Ausgabestellen von t.



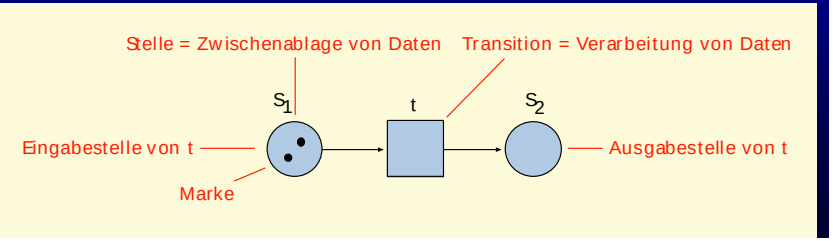


LE 12 8

2.17.1 Grundlagen

- Objekte werden als Marken bezeichnet

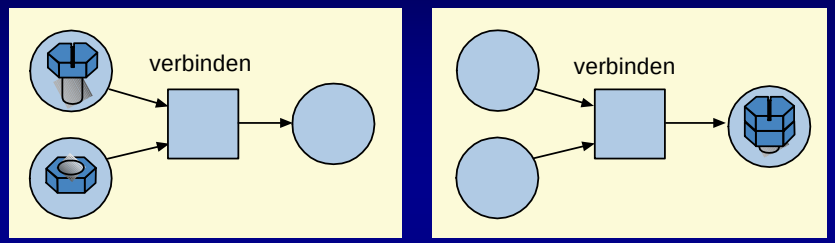
- Schaltregel:
  - a Eine Transition t kann »feuern«, wenn jede Eingabestelle von t mindestens 1 Marke enthält
  - b Schaltet eine Transition, dann wird aus jeder Eingabestelle 1 Marke entfernt und zu jeder Ausgabestelle 1 Marke hinzugefügt



LE 12
9

2.17.1 Grundlagen

- **Anschauliche Vorstellung der Schaltregel**



- **Abhängig von der Art der Objekte:**
  - Bedingungs/Ereignis-Netze (B/E-Netz)
  - Stellen/Transitions-Netze (S/T-Netz)
  - Höhere Petri-Netze.

LE 12
10

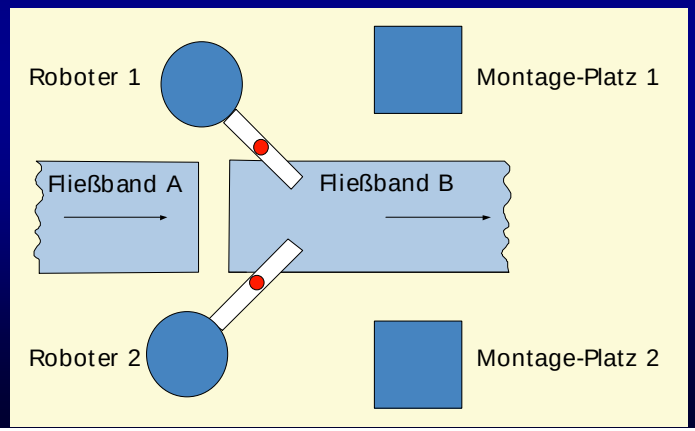
2.17.2 Bedingungs/Ereignis-Netze

- **B/E-Netz**
  - Objekte bzw. Marken sind vom Datentyp boolean
  - Transitionen werden als Ereignisse interpretiert
  - Stellen werden als Bedingungen bezeichnet
  - Jede Stelle kann entweder genau eine oder keine Marke enthalten
  - Zusätzliche Schaltbedingung:
  - C Eine Transition t kann schalten, wenn jede Eingabestelle von t eine Marke enthält und wenn jede Ausgabestelle von t leer ist.

LE 12
11

2.17.2 Bedingungs/Ereignis-Netze

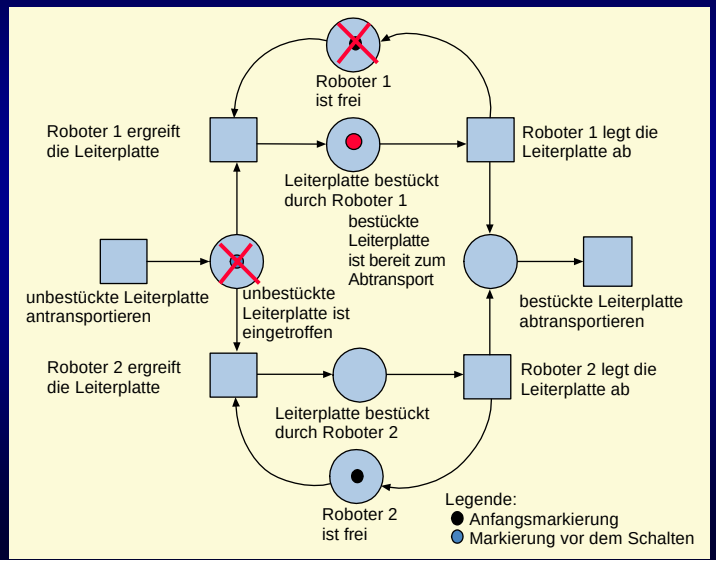
- **Beispiel**
  - 2 Roboter bestücken Leiterplatten mit elektronischen Bauelementen, die auf einem Fließband A antransportiert werden



LE 12
12

2.17.2 Bedingungs/Ereignis-Netze: Beispiel

- B/E-Netz des Bestückungsroboters



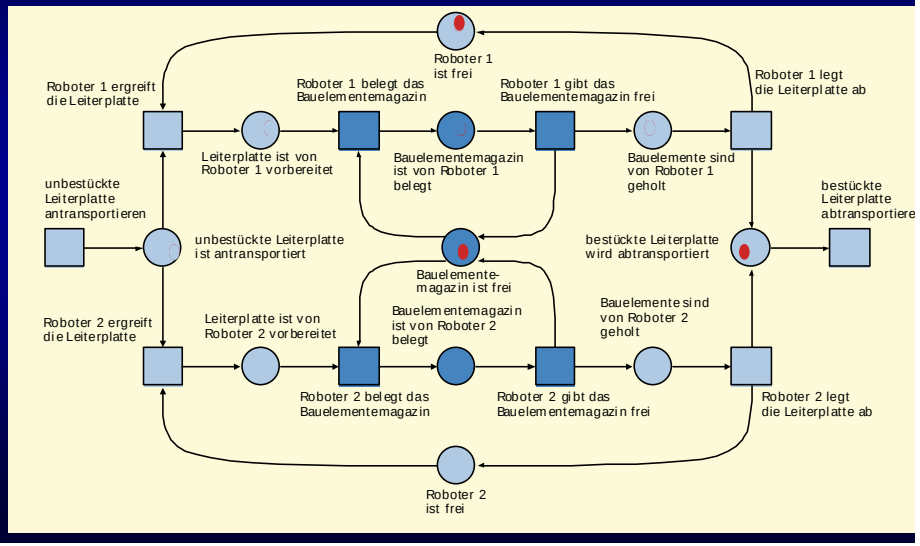
Vor dem Schalten

Nach dem Schalten.

LE 12
13

2.17.2 Bedingungs/Ereignis-Netze

- B/E-Netz mit gemeinsamem Bauelemente-Magazin



LE 12
14

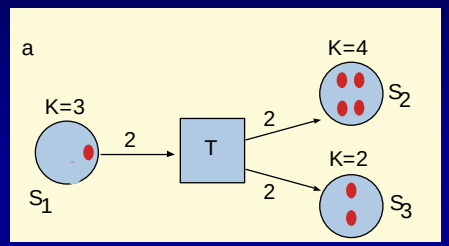
2.17.3 Stellen/Transitions-Netze

- **S/T-Netze** *(P/T Net, Place/Transition Net)*
  - Stellen können **mehr als eine** Marke enthalten (in B/E-Netzen nur eine Marke)
  - Transitionen müssen so viele Marken beim Schalten wegnehmen oder hinzufügen, wie die **Gewichte** an den Pfeilen angeben (in B/E-Netzen nur eine Marke)
  - Soll eine Stelle eine Kapazität größer 1 erhalten, dann wird dies durch **»K = ...«** an der Stelle notiert
  - Die Kapazität definiert die maximale Anzahl von Marken, die auf einer Stelle liegen dürfen.

LE 12
15

2.17.3 Stellen/Transitions-Netze

- Schaltbedingungen bei S/T-Netzen

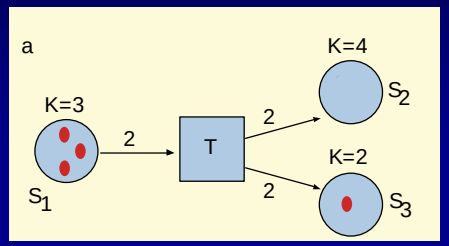


- T kann schalten
  - Anschließend sind in
    - S_1 **1** Marke
    - S_2 **4** Marken
    - in S_3 **2** Marken.

LE 12
16

2.17.3 Stellen/Transitions-Netze

- Schaltbedingungen bei S/T-Netzen

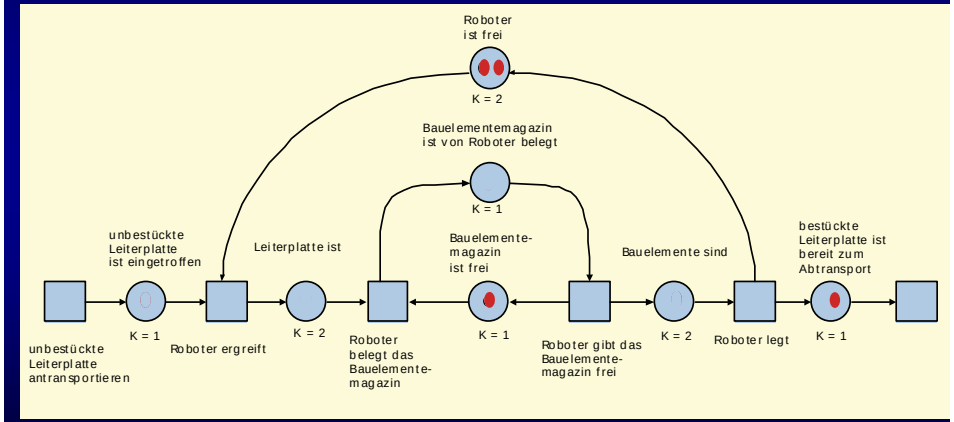


- T kann **nicht** schalten, da in S_3 dann **3** Marken liegen würden
  - Dies ist wegen **K = 2** von S_3 **nicht** erlaubt.

LE 12
17

2.17.3 Stellen/Transitions-Netze

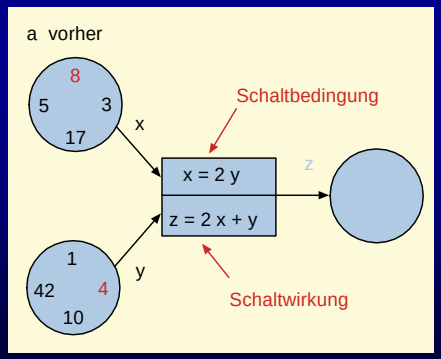
- **S/T-Netz des Bestückungsroboters**

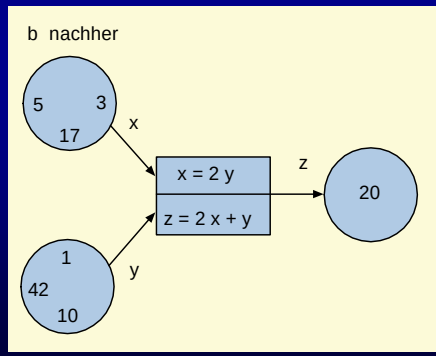


LE 12
18

2.17.4 Prädikat/Transitions-Netze

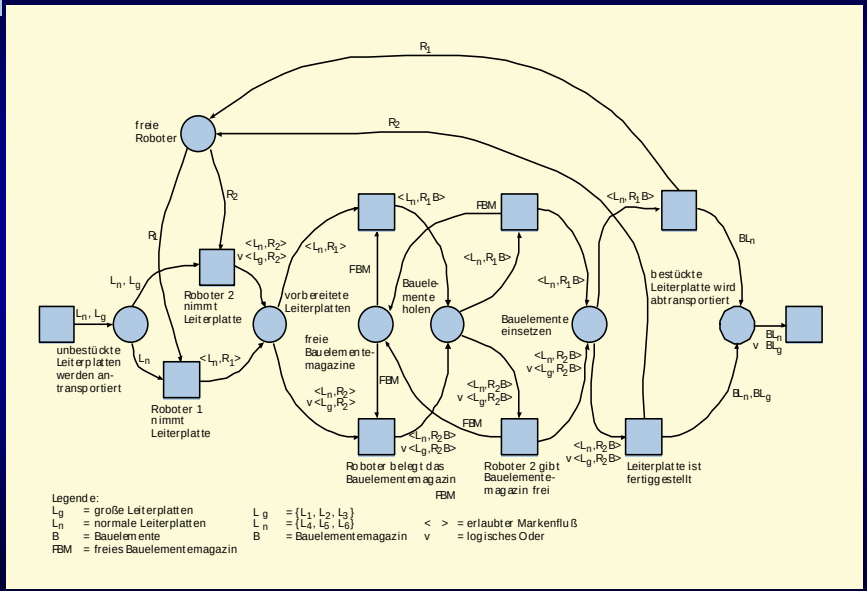
- **Pr/T-Netze**
  - **Verwenden individuelle, »gefärbte« Marken**
  - **B/E- und S/T-Netze verwenden nur »schwarze« Marken, die alle gleich sind**





LE 12
19

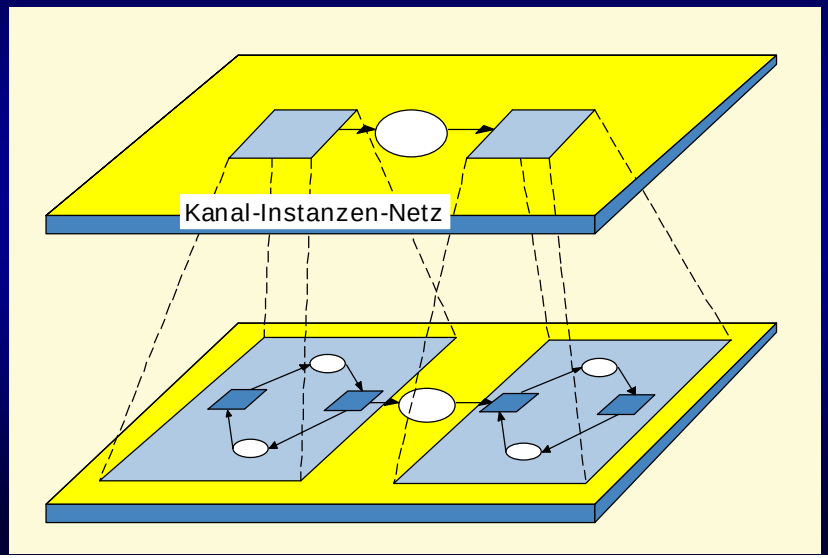
2.17.4 Prädikat/Transitions-Netze



LE 12
20

2.17.5 Hierarchische Petri-Netze

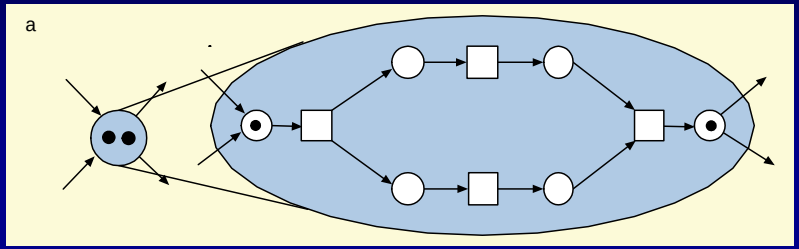
- **Strukturierte Darstellung komplexer Netze**



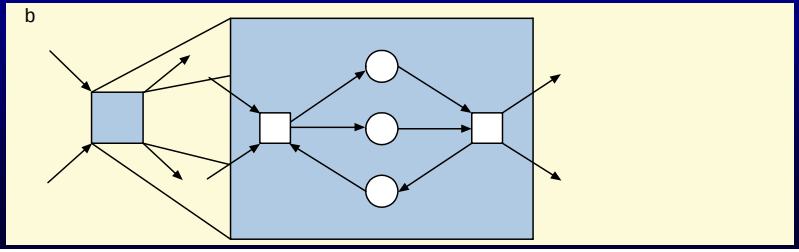
LE 12
21

2.17.5 Hierarchische Petri-Netze

- Markengetreue Verfeinerung einer Stelle

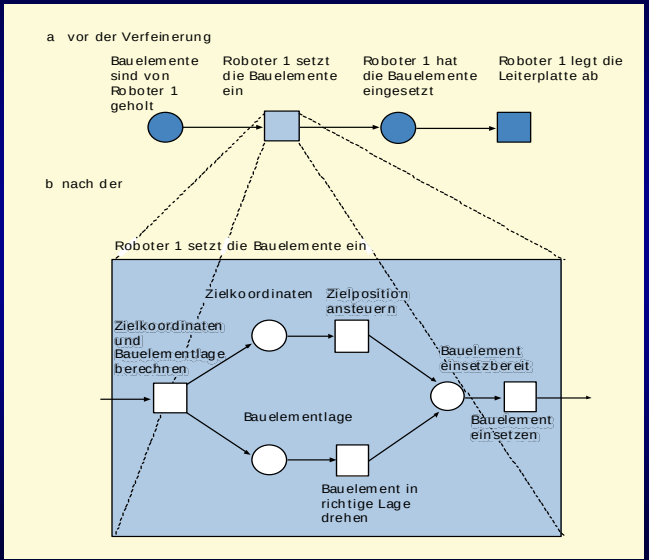


- Markengetreue Verfeinerung einer Transition



LE 12
22

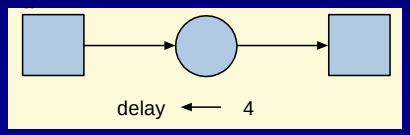
2.17.5 Hierarchische Petri-Netze

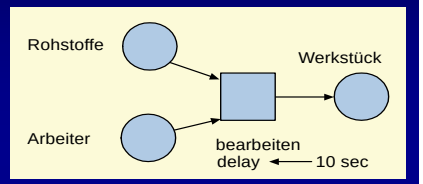


LE 12
23

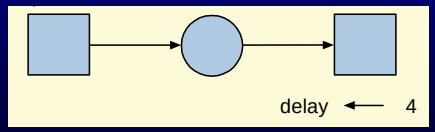
2.17.6 Zeitbehaftete Petri-Netze

- Verschiedene Notationsmöglichkeiten



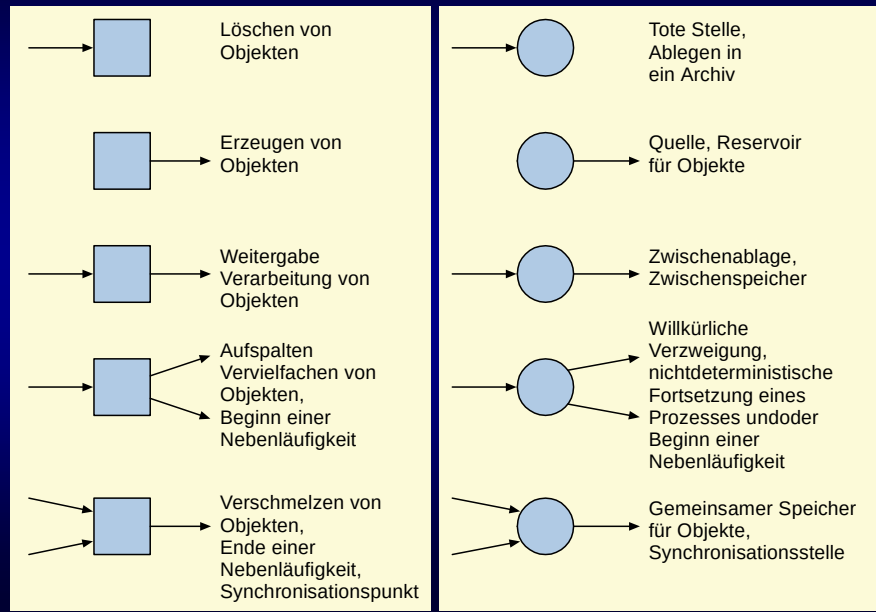


- Die Transition kann erst 4 Zeiteinheiten nach dem Eintreffen der Marke in der Eingangsstelle feuern



LE 12
24

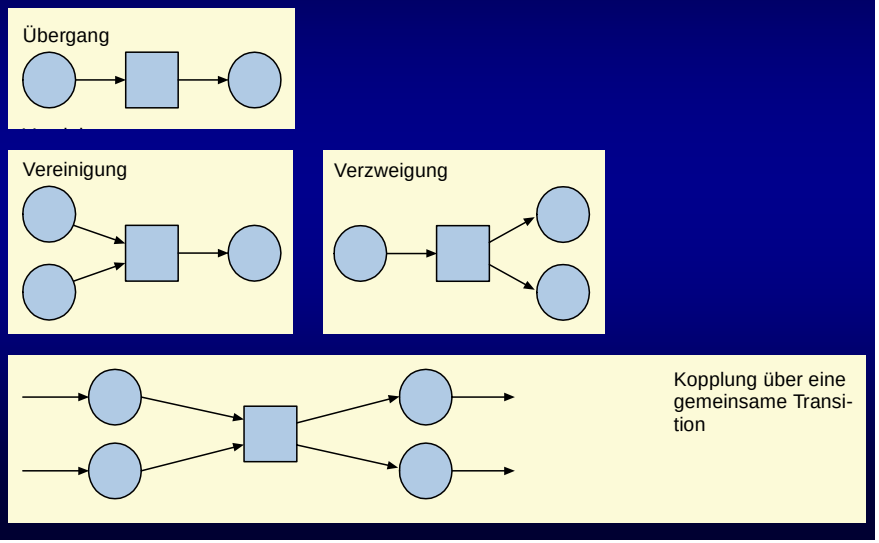
2.17.7 Strukturelemente & Strukturen



LE 12
25

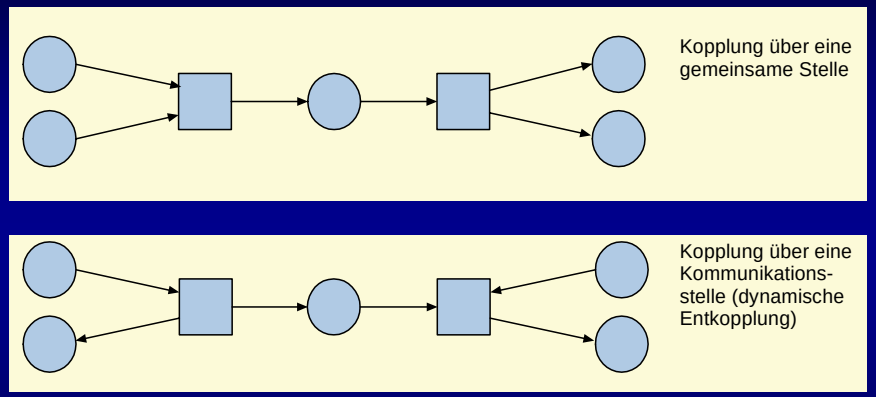
2.17.7 Strukturelemente & Strukturen

- **Strukturen**



LE 12
26

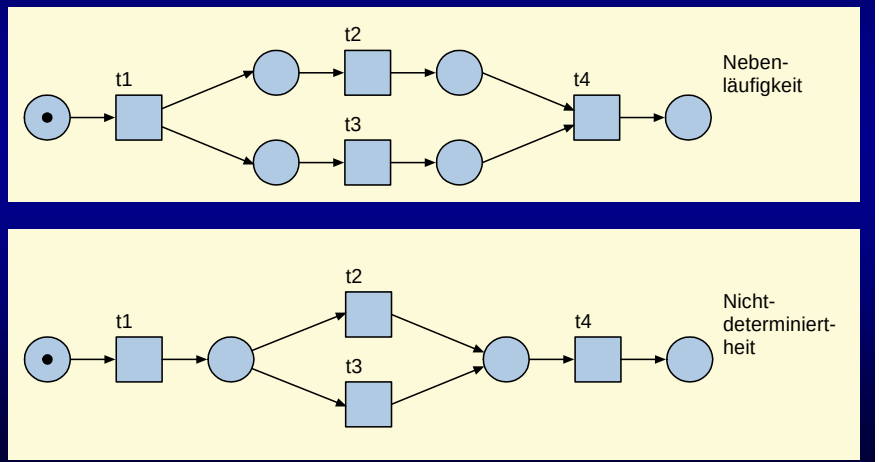
2.17.7 Strukturelemente & Strukturen



LE 12
27

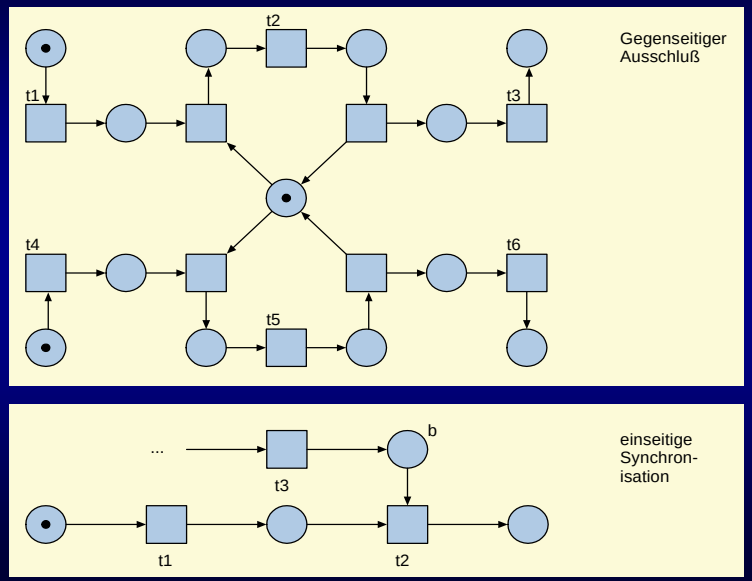
2.17.7 Strukturelemente & Strukturen

- **Typische Anwendungsmuster**
  - **Aufgebaut aus Strukturen**



LE 12
28

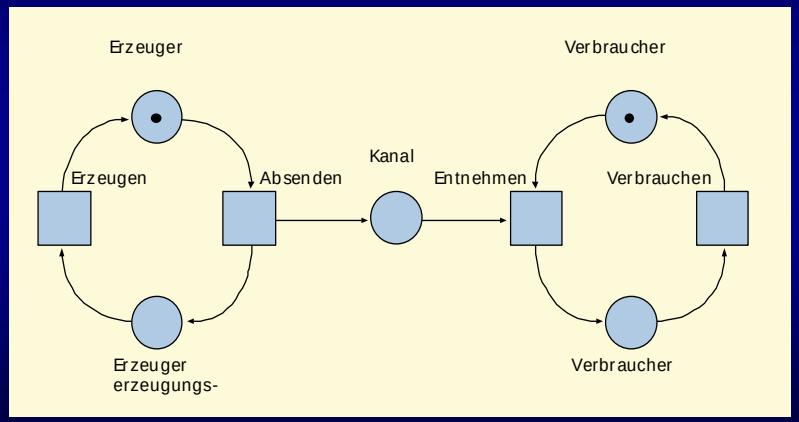
2.17.7 Strukturelemente & Strukturen



LE 12
29

2.17.7 Strukturelemente & Strukturen

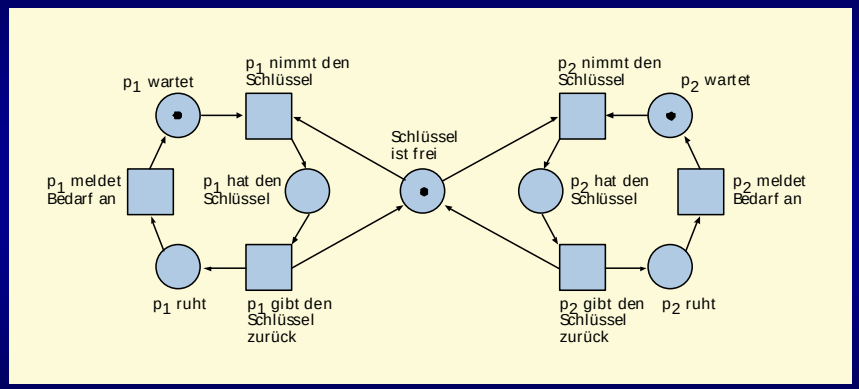
- **Produzenten & Konsumenten** bzw. **Erzeuger & Verbraucher** im Petri-Netz



LE 12
30

2.17.7 Strukturelemente & Strukturen

- **Leser & Schreiber** im Petri-Netz



LE 12
31

2.17.8 Methodik

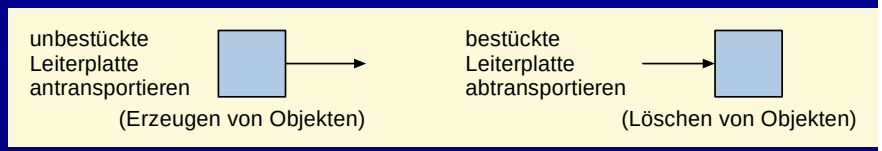
1. Aktive und passive Komponenten identifizieren
2. Beziehungen ermitteln
3. Verfeinerung und Ergänzung
4. Festlegung der Objekte
5. Überlegungen zu Schaltregeln und Schaltwirkungen
6. Netztyp festlegen
7. Anfangsmarkierung festlegen
8. Analyse, Simulation.

LE 12
32

2.17.8 Methodik: Beispiel

1. Schritt: aktive & passive Komp. identifizieren

- Analysiert man die Problembeschreibung der Leiterplattenbestückung, dann werden die Leiterplatten vom Fließband A antransportiert und vom Fließband B abtransportiert

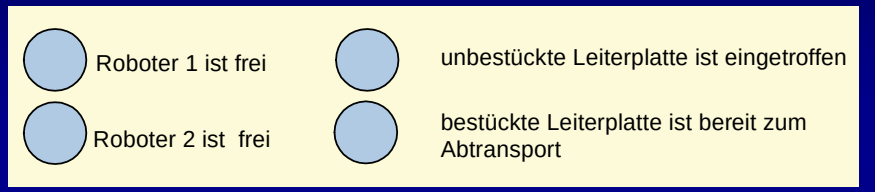


- Für die Problemstellung ist uninteressant, was vor dem Antransport und nach dem Abtransport mit den Leiterplatten geschieht
  - Schnittstellenmodellierung ist problemgerecht.

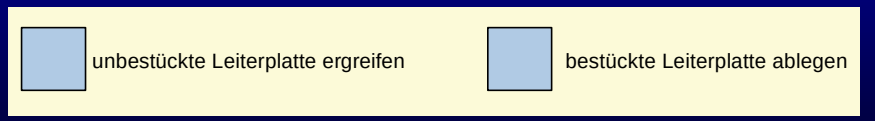
LE 12
33

2.17.8 Methodik: Beispiel

- Anhand der Problembeschreibung lassen sich passive Komponenten identifizieren



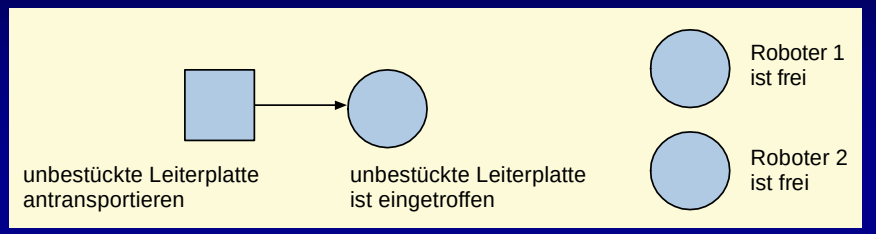
- Es ergeben sich weitere aktive Komponenten

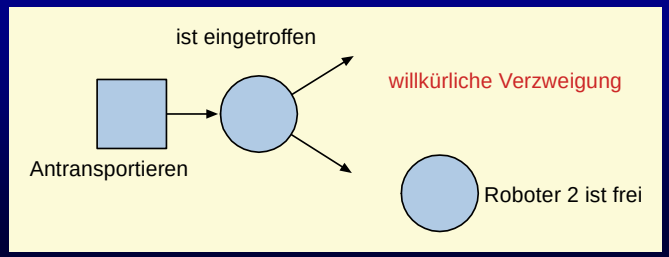


LE 12
34

2.17.8 Methodik: Beispiel

2. Schritt: Beziehungen ermitteln

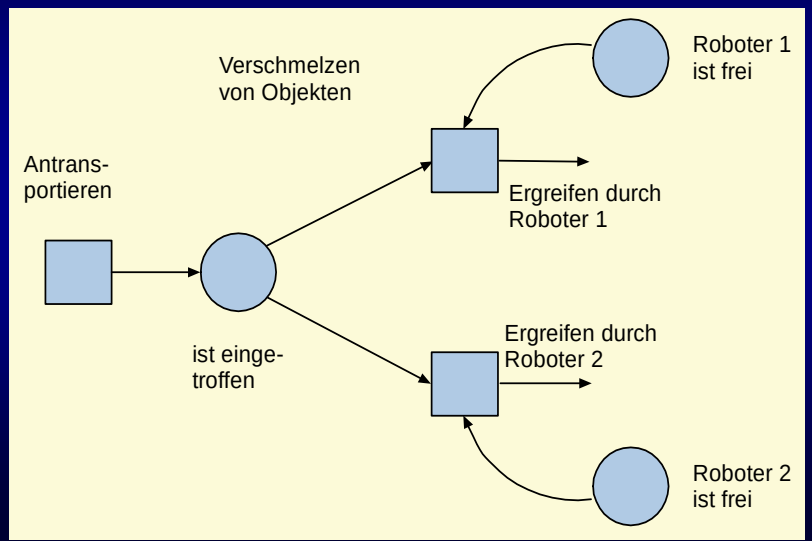




LE 12
35

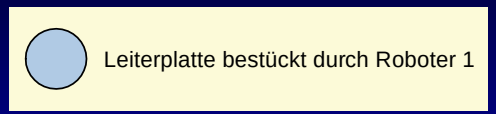
2.17.8 Methodik: Beispiel

- Verschmelzen von Roboterarm und Leiterplatte

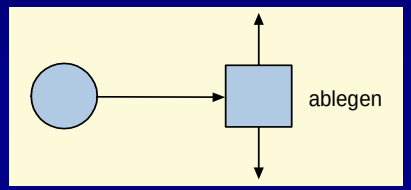


LE 12
36

2.17.8 Methodik: Beispiel

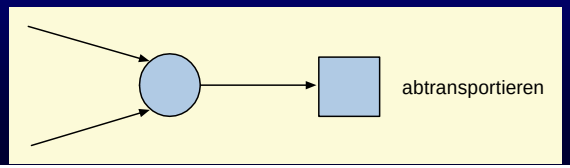


Leiterplatte bestückt?

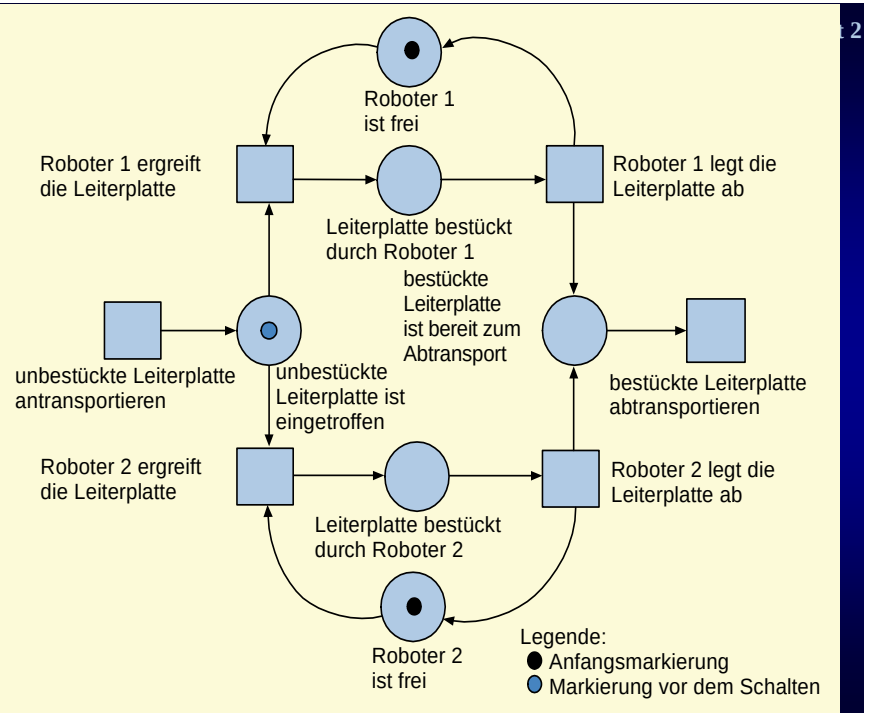


Aufspalten von Objekten

Die getrennt bearbeiteten Leiterplatten werden anschließend wieder auf dem Transportband B vereinigt



LE 12
37



LE 12
38

2.17.8 Methodik: Beispiel

3. Verfeinerung und Ergänzung

- Nach dem Erstellen eines Kanal-Instanzen-Netzes kann man iterativ...
  - Instanzen und Kanäle verfeinern
  - das erstellte Netz verfeinern

4. Festlegung der Objekte

- Welche konkreten Objekte können die Kanäle bzw. Stellen beinhalten?
  - Reichen anonyme Objekte aus oder werden individuelle Objekte benötigt?
  - Daraus ergeben sich mögliche Netztypen.

LE 12
39

2.17.8 Methodik: Beispiel

5. Überlegungen zu Schaltregeln und Schaltwirkungen

- Welche Schaltregeln und Schaltwirkungen sind zur Modellierung nötig?

6. Netztyp festlegen

- Aus den Objekt- und Schaltüberlegungen ergibt sich der benötigte Netztyp

7. Anfangsmarkierung festlegen

- Anfangsmarkierung überlegen

8. Analyse, Simulation

- Netz analysieren und simulieren.

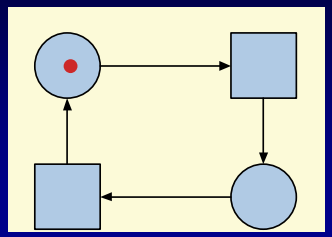
LE 12
40

2.17.9 Analyse und Simulation

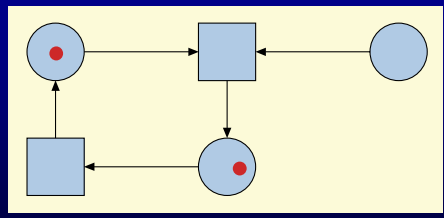
- Typische Fragestellungen zur Analyse:
  - Terminiert das Netz?
    - Können, ausgehend von einer Anfangsmarkierung, stets nur endlich viele Transitionen schalten?
  - Ist jede Transition lebendig?
    - Können, ausgehend von einer Anfangsmarkierung, die Transitionen stets so schalten, daß eine vorgegebene Transition t im weiteren Verlauf nochmals schalten kann?
  - Treten vermeidbare Verklemmungen auf?
    - Gibt es Situationen, in denen keine Transition schalten kann, die aber bei anderer Schaltreihenfolge hätten vermieden werden können?

2.17.9 Analyse und Simulation

- Das Netz ist **lebendig,** da die Transitionen immer abwechselnd schalten

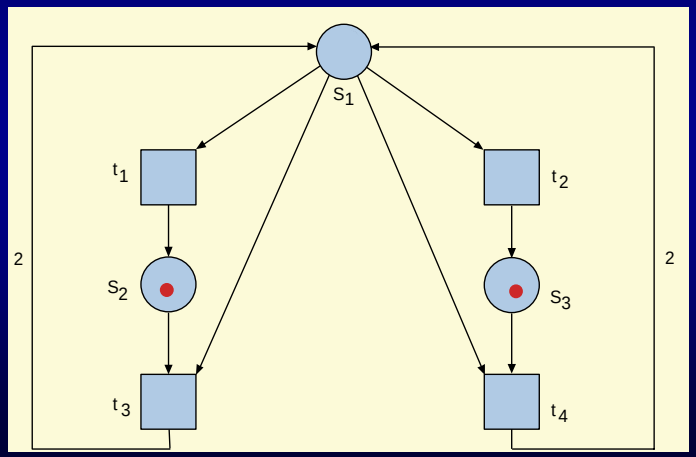


- Das Netz ist **todesgefährdet,** da nach dem 2. Schalten keine weitere Transition mehr schalten kann.



2.17.9 Analyse und Simulation

- Verklemmung *(deadlock)*
  - Wenn eine Stellenmenge – einmal ohne Marken – nie mehr markiert werden kann



2.17.10 Wertung

- Petri-Netze
  - Eignen sich besonders gut zur Modellierung von Systemen mit kooperierenden Prozessen
  - Das Anwendungsspektrum umfasst daher
    - diskrete
    - ereignisorientierte
    - verteilte Systeme
  - Sie werden auch für die Vorgangsmodellierung von Bürovorgängen eingesetzt *(work flow).*

2.17.10 Wertung

- Ähnlichkeiten mit **Zustandsautomaten:**
  - **Stellen** lassen sich als **Zustände** interpretieren
  - **Transitionen** lassen sich als **Zustandsübergange** interpretieren
- Zustandsautomaten
  - Ein System befindet sich zu jedem Zeitpunkt in genau **einem** Zustand
- Petri-Netz
  - Ein System kann sich zu einem Zeitpunkt in **mehreren** Zuständen, dargestellt durch die aktuelle Markenbelegung, befinden.

LE 12
45

2.17.10 Wertung

- Petri-Netze vs. Zustandsautomaten
  - Je nach Struktur des Petri-Netzes können in einem System unabhängig voneinander Zustandsübergänge erfolgen
  - Synchronisationen zwischen nebenläufigen Systemen können durch eine geeignete Netzstruktur erzwungen werden
  - Petri-Netze besitzen...
    - eine größere Mächtigkeit
    - erlauben die Modellierung von Problemklassen, für die Zustandsautomaten nicht geeignet sind.

LE 12
46

2.17.10 Wertung

- B/E-Netze
  - Gut geeignet für die Beschreibung des Kontrollflusses bei kooperierenden Prozessen
    - Dabei wird der augenblickliche Zustand jedes Prozesses durch eine Marke repräsentiert
- S/T-Netze
  - Erweitern die Modellierungsmöglichkeiten, da beliebig viele – aber weiterhin – anonyme Marken pro Stelle abgelegt werden können.

LE 12
47

2.17.10 Wertung

- Pr/T-Netze
  - Sind übersichtlicher als S/T-Netze
  - Modellierungsmächtigkeit ist größer, zumal auch beliebige, kontinuierliche Funktionen benutzt werden können
    - Beispiel: Höhe > 4000 m als Schaltbedingung
  - Nachteil: Schwierigere Handhabung dieser Netze
  - Schwierig, korrekte Schaltbedingungen und Schaltwirkungen zu entwerfen
- Generell gilt:
  - Je größer die Modellierungsmächtigkeit, desto geringer die Analysemöglichkeiten.

LE 12
48

2.17.10 Wertung

- Vorteile
  - + Bestehen aus wenigen und einfachen Elementen
  - + Sind grafisch gut darstellbar
  - + Marken erlauben eine gute Visualisierung des jeweiligen Systemzustands
  - + Besitzen ein solides theoretisches Fundament
  - + Petri-Netze können – im beschränkten Rahmen – analysiert und simuliert werden
  - + Es gibt Petri-Netz-Werkzeuge, die die Erstellung, Analyse, Simulation und Code-Generierung erlauben
  - + Einziges weit verbreitete Basiskonzept zur Modellierung kooperierender Prozesse.

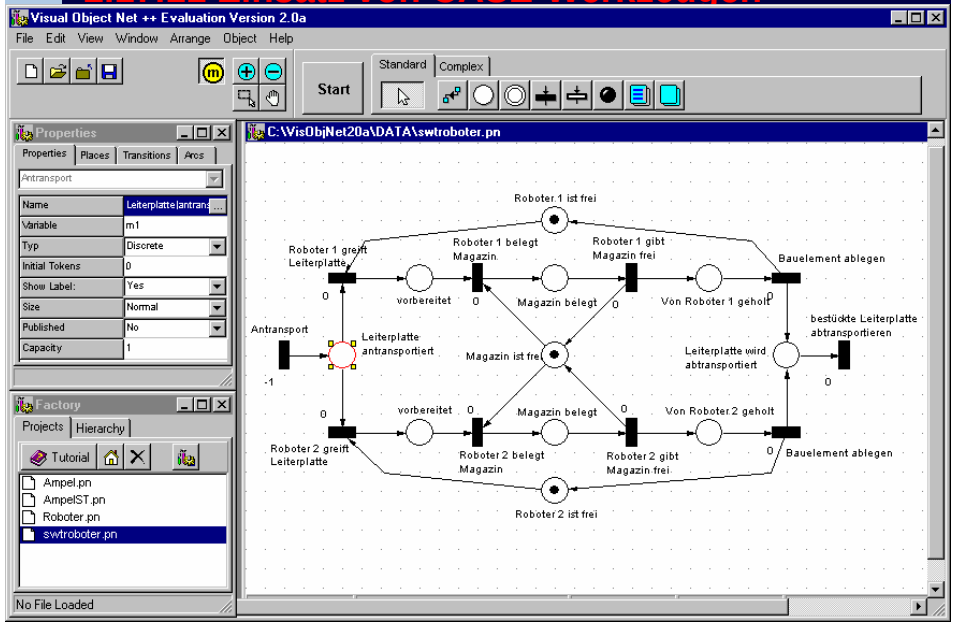
LE 12
49

2.17.10 Wertung

- Nachteile
  - Für die Praxis sind höhere Petri-Netze nötig, für die es keine einheitliche Notation gibt
  - Höhere Petri-Netze sind schwer zu erstellen und zu analysieren
  - Petri-Netze sind mit anderen Basiskonzepten bisher nicht kombiniert worden, d.h. es ist ein vollständig für sich stehendes Konzept
  - Petri-Netze besitzen eine statische Struktur
    - Will man z.B. Vorgänge beschreiben, bei denen dauernd neue Prozesse erzeugt werden (z.B. *task*-Konzept in Ada), dann erweisen sich Petri-Netze als ungeeignet
  - Es gibt keine allgemeine Erstellungs-Methode.

LE 12

2.17.11 Einsatz von CASE-Werkzeugen



LE 12
51

Zusammenhänge

Allgemeine Bezeichnung	grafische Repräsentation	B/E - Netze	S/T - Netz	Pr/T - Netz	Zeitbehaftetes Netz	Hierarchisches Netz
Stelle	●	Bedingung *markiert oder*	Stelle *Kapazität > 0*	Prädikat *Kapazität > 0*	Stelle *Zeitintervall*	Kanal
Transition	▮ *oder* ▮	Ereignis	Transition	Ereignis *Schaltbedingung & Schaltwirkung*	Transition *Zeitintervall*	Instanz
Pfeil (Flussrelation)	→	ungewichtet	gewichtet	gewichtet *konstante und/ oder variable Beschriftung*	je nach Netztyp	je nach Netztyp
Marke	●	Marke *uniform*	Marke *uniform*	Objekt *individuell*	je nach Netztyp	je nach Netztyp

LE 12
52

- Danke!
- Aufgaben

- Diese Präsentation bzw. Teile dieser Präsentation enthalten Inhalte und Grafiken des Lehrbuchs der Software-Technik (Band 1), 2. Auflage von Helmut Balzert, Spektrum Akademischer Verlag, Heidelberg 2001