

Konsistentes Verschalten Petri-Netz-basierter Komponenten beim Entwurf verteilter Steuerungen in CNet

Nils Hagge, Bernardo Wagner
Institut für Systems Engineering
Fachgebiet Echtzeitsysteme
Universität Hannover
Appelstraße 9A
D-30167 Hannover
Telefon/Telefax: +49-511-762-19898/-4012
{hagge|wagner}@rts.uni-hannover.de

Abstract: Dieser Beitrag gibt eine kurze Einführung in das CNet-Komponentenmodell. Dabei handelt es sich um eine Modellierungssprache zum Entwurf von verteilten Steuerungen auf Basis von Petri-Netzen mit der Möglichkeit zur Hierarchisierung, Modulbildung und Wiederverwendung vergleichbar mit der Funktionsbausteinsprache nach IEC 61499, jedoch deutlich geringem Sprachumfang. Ziel ist die automatische Generierung von Steuerungsprogrammen zur verteilten Ausführung einschließlich der erforderlichen Kommunikationselemente. Die Schnittstellen von CNet-Komponenten werden aus neu eingeführten Eingangsstellen und Ausgangstransitionen gebildet. Das Verhalten der Komponenten wird mit einer speziellen Klasse gefärbter Petri-Netze – genannt PNet – formuliert. Es gilt die starke Schaltregel. Für die Verschaltung der Komponenten untereinander wird Rückwirkungsfreiheit gefordert. Dies bedeutet, dass die äußere Beschaltung einer Ausgangstransition deren Schalten nicht verhindern darf. Dieser Beitrag gibt ein formales Verfahren an, das bei bekanntem Komponentenverhalten nachweisen kann, ob bei einer gegebenen Verschaltung von Komponenten unter angenommener starker Schaltregel keine Ausgänge (Ausgangstransitionen) blockieren.

Stichworte: Petri-Netze, Komponenten, Analyse, Erweiterter Erreichbarkeitsgraf, Konsistentes Verschalten

1 Einleitung

CNet leitet sich aus „Component Net“ ab und stellt eine überwiegend grafische Modellierungssprache zum Entwurf komplexer verteilter Steuerungen basierend auf Petri-Netzen dar [Wu2002]. CNet ermöglicht Modularisierung, Hierarchisierung und Wiederverwendung. Das zentrale Sprachelement von CNet bildet die CNet-Komponente (Abb. 1(c)), welche sowohl zur Modellierung der Steuerungsausgaben als auch zur Beschreibung des Prozessverhaltens und zur Kapselung von Kommunikationsmitteln zum Zweck der Verteilung eingesetzt wird. Die Schnittstellen von CNet-Komponenten setzen sich aus so genannten *Eingangsstellen* und *Ausgangstransitionen* zusammen. Dies sind neue Sprachelemente, die sich aber stark an die bekannten Elemente Stelle und Transition der Petri-Netze anlehnen. Erstere sind eine spezielle Form

von Stellen (Abb. 1(a)), die von außerhalb einer Komponente mit Marken belegt werden können. Letztere sind Transitionen (Abb. 1(b)), die von außerhalb einer Komponente nur über Postkanten mit anderen Stellen verbunden werden dürfen. Diese Vereinbarung sorgt dafür, dass sich ungeschaltete Schnittstellenelemente „neutral“ verhalten in Bezug auf die Funktionalität der übrigen Elemente. Dies beruht auf der Tatsache, dass unverbundene Ausgangstransitionen immer schalten, wenn ihre (inneren) Vorbedingungen erfüllt sind, insbesondere auch dann, wenn es keine äußeren Nachbedingungen gibt. Dual dazu modellieren unverbundene Eingangsstellen Bedingungen oder Situationen, die niemals eintreten. Diese Festlegungen begünstigen die Wiederverwendung von Komponenten, da komplexe Komponenten auch dann instanziiert werden können, wenn nicht die volle Funktionalität benötigt wird. Dies wäre nicht möglich, wenn Ausgänge aus Stellen modelliert würden, da eine unverbrauchte Marke am Ausgang zu einer internen Blockade der Komponente aufgrund der starken Schaltregel führen würde.

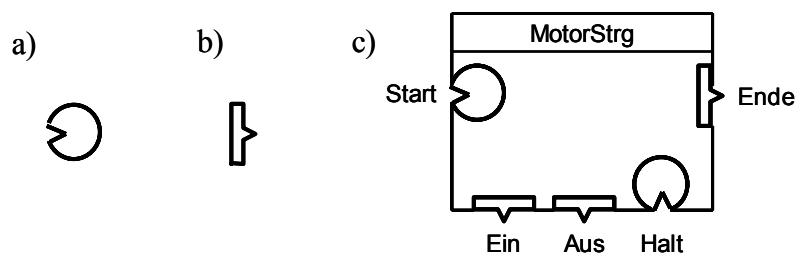


Abbildung 1: CNet-Sprachelemente a) Eingangsstelle b) Ausgangstransition c) Komponente

Die farbigen Marken in CNet können zusätzliche Informationen über den Prozess oder den Zustand der Steuerung ausdrücken, welche nicht in der Struktur der Netze enthalten sind oder sein sollen. Dazu werden die Eingangsstellen sowie die einfachen Stellen mit einer Farbmenge beschriftet. Diese Farbmenge bezeichnet einen strukturierten Datentyp, der die zu speichernde Information wie in einer imperativen Programmiersprache näher beschreibt und eingrenzt (vgl. [Jen1992]).

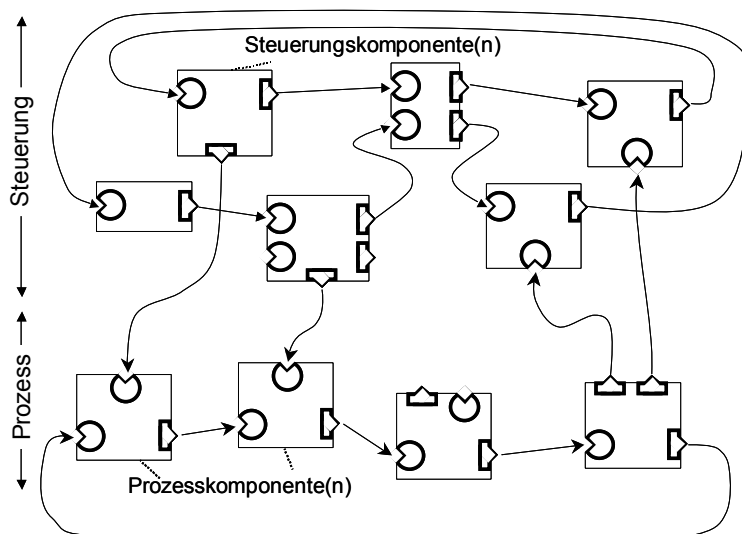


Abbildung 2: Komponentennetzwerk bestehend aus Steuerungs- und Prozesskomponenten

In Abb. 2 wird beispielhaft ein Komponentennetzwerk gezeigt, das sich in zwei Teile, das Steuerungs- und das Prozessmodell, gliedert. Die Verbindung zwischen Steuerungs- und Prozesskomponenten erfolgt ebenfalls über Eingangsstellen und Ausgangstransitionen. Schnittstellenelemente, die zur Kommunikation der Steuerung mit dem Prozess dienen, werden in der grafischen Darstellung der Komponenten an den vertikalen Kanten eingezeichnet. Zur Generierung des Steuerungscode werden nur die Steuerungskomponenten benötigt, während die Prozesskomponenten durch sogenannte Treiberkomponenten ersetzt werden, welche den tatsächlichen Prozesszugriff implementieren. Weitere Erläuterungen dazu können [HW2005b] entnommen werden. Schwerpunkt von CNet ist der grafische Entwurf und die werkzeuggestützte Generierung von nebenläufigen verteilt ausführbaren Steuerungsprogrammen in Java. Eine vollständige Verifikation ist im Allgemeinen zu komplex, daher beschränkt sich die Analyse auf Komponenten oder Teilnetze mit dem Ziel, Informationen zu erhalten, die für eine fehlerfreie Codegenerierung benötigt werden.

2 Verhaltensanalyse

Petri-Netze haben gegenüber anderen in der Steuerungstechnik üblichen Beschreibungsmitteln den Vorteil neben der grafischen Darstellung auch mathematisch formal erfassbar zu sein. Dies bildet die Grundlage für zahlreiche Ansätze zur formalen Verifikation eines Steuerungsentwurfs hinsichtlich bestimmter aus einer Spezifikation geforderter Eigenschaften. Dies hat eine besondere Bedeutung beim Entwurf sicherheitskritischer Applikationen. Ein bekanntes Verfahren ist die Erreichbarkeitsanalyse zur Ermittlung sämtlicher möglicher durch das System einnehmbarer Zustände. In realen Aufgaben ist diese Vorgehensweise häufig nicht praktikabel, da obwohl in beschränkten Netzen auch die Erreichbarkeitsmenge beschränkt ist, die Analyse komplexerer Netze meist auf eine nicht mehr handhabbare Menge an Zuständen („Zustandsexplosion“) führt.

2.1 Analyse der Komponenten

Durch die Hierarchisierung und Komponentenbildung in CNet erhält man die Möglichkeit, kleinere überschaubare Teilnetze getrennt zu betrachten. Dabei ist zu berücksichtigen, dass die verhaltensbeschreibenden Netze in den CNet-Komponenten aufgrund der Schnittstellenelemente keine in sich geschlossenen Petri-Netze darstellen und daher das bekannte Verfahren zur Ermittlung des Erreichbarkeitsgraphen (vgl. [Ab1990]) nicht angewandt werden kann, weil es die neuen Netzelemente Eingangsstelle und Ausgangstransition nicht berücksichtigt. In [HW2005b] wurde das Verfahren in der Weise erweitert, dass das Belegen einer Eingangsstelle als besonderer Schaltvorgang mit einer niedrigeren Priorität gegenüber Transitionen betrachtet wird. Dadurch wird in Kombination mit dem Schaltzwang erreicht, dass alle möglichen internen Schaltvorgänge abgeschlossen sind, bevor externe Ereignisse wahrgenommen werden. Der auf diese Weise gewonnene Graph wird „Erweiterter Erreichbarkeitsgraph“ (EEG) genannt. Sind die Ausführungszeiten für die einzelnen Schaltvorgänge bekannt, können aus dem EEG die Reaktionszeiten einer Komponente auf die Eingangsereignisse ermittelt werden. Der Vorteil der Wiederverwendung von Komponenten bezieht sich auch auf die Komponentenanalyse. Der einmal er-

mittelte EEG kann zusammen mit der Komponentenbeschreibung in einer Bibliothek abgelegt werden.

2.2 Konsistente Verschaltung der Komponenten

Neben der korrekten Funktion der Komponenten eines Steuerungsentwurfs muss sichergestellt sein, dass Komponenten nur im zulässigen Kontext eingesetzt werden. Dazu gehört einerseits, dass die Farbmengen der Kantenausdrücke und der mit ihnen verbundenen Eingängen oder Ausgängen kompatibel zueinander sind. Dies ist vergleichbar mit der Typkompatibilität von aktuellen und formalen Parametern bei Unterprogrammaufrufen in imperativen Programmiersprachen. Darüber hinaus wird in [Wu2002] gefordert, dass die äußere Beschaltung von Komponenten keine Rückwirkungen auf die Schaltfähigkeit der Ausgangstransitionen und inneren Transitionen der Komponenten erzeugt. Diese Forderung beruht auf der Tatsache, dass die Ausgangstransitionen einer CNet-Komponente als nur mit einem unidirektionalen Informationsfluss behaftet verstanden werden sollen. Im Allgemeinen findet an den Postkanten aufgrund der *starken Schaltregel* ein bidirektionaler Informationsfluss statt. Beim Überprüfen der Schaltfähigkeit einer Transition gelangt die Information, ob noch ausreichend Kapazität im Nachbereich vorhanden ist, von den Stellen zur Transition, während beim Schalten der erzeugte Farbwert von der Transition über die Postkanten zu den Stellen gelangt. Eine Überprüfung der Aktiviertheit einer Ausgangstransition ohne Betrachtung der Kapazität an der externen Postkante entspräche jedoch der Anwendung der *schwachen Schaltregel*. Da aufgrund der informationstragenden Marken in CNet in der realen Implementierung die Kapazitäten von Stellen stets endlich ist, kann nicht von der starken Schaltregel abgewichen werden, ohne den Verlust von Marken an Komponentenausgängen zu riskieren. Daher muss auf andere Weise sichergestellt werden, dass von Ausgangstransitionen erzeugte Marken stets von den betroffenen Stellen aufgenommen werden können. In [Wu2002] werden dazu informelle Festlegungen getroffen, aus denen sich Entwurfsregeln ergeben, die dieses Verhalten sicherstellen sollen. Das Befolgen und Überprüfen dieser Entwurfsregeln durch den Anwender gestaltet sich jedoch erfahrungsgemäß schwierig. Ziel einer Beschreibungssprache und der mit ihr zur Verfügung gestellten Werkzeuge sollte es daher immer sein, solche Anforderungen möglichst vollautomatisch zu überprüfen. Daher wird im Folgenden eine Konsistenzregel formuliert, die eine spätere formale Überprüfung der korrekten Verschaltung von Komponenten ermöglichen soll.

3 Konsistenzregel zur Verschaltung von Komponentenausgängen

Abb. 3 zeigt einen Ausschnitt aus einem CNet-Systemmodell. Das System bestehe aus dem Komponentennetzwerk $\Sigma = (C, F_0)$, welches sich aus den Komponenten $C = \{c_1, c_2, \dots, c_n\}$ und deren Verschaltungen, ausgedrückt durch die Menge der externen Postkanten $F_0 = \{f_1, f_2, \dots, f_n\}$, zusammensetzt. CNet-Modelle können neben den Komponenten auch noch Stellen und Transitionen enthalten, die sich außerhalb einer Komponente befinden und die mit Komponentenschnittstellen kommunizieren. Dies wird hier der Übersicht halber weggelassen. In Abb. 3 werden nur die beiden Komponenten c_1 und c_2 einschließlich ihrer internen Imp-

lementierung sowie die Kanten $f_1 = (a_1, e_1)$ und $f_2 = (a_2, e_2)$ dargestellt. Die Kante f_1 ist einschließlich der Schnittstellenelemente, die sie verbindet, hervorgehoben. Mit der Konsistenzregel, die im Folgenden vorgestellt wird, kann nachgewiesen werden, dass die Verbindung eines Ausgangs a einer Komponente c mit dem Eingang e einer anderen Komponente c' über eine Kante $f = (a, e)$ rückwirkungsfrei ist. Das bedeutet, dass die Kapazität des Eingangs e so gewählt wurde, dass alle Marken, die durch Schaltvorgänge am Ausgang a erzeugt werden, aufgenommen werden können.

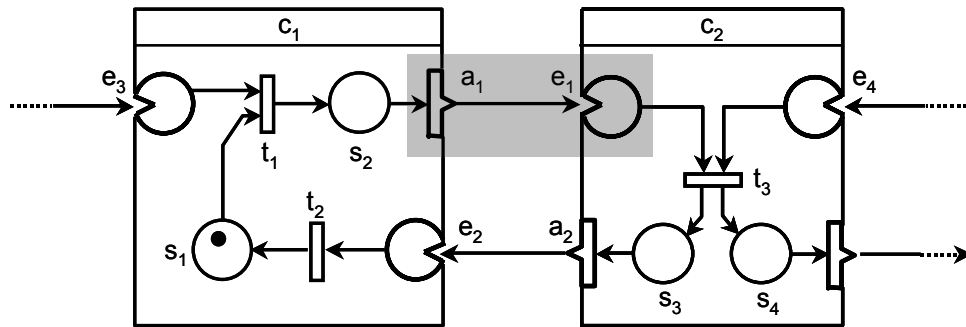


Abbildung 3: Ausschnitt aus einem Systemmodell – hervorgehoben ist eine Komponentenverbindung

Um den formalen Nachweis führen zu können, werden folgende Größen und Symbole benötigt:

- k : Kapazität des Eingangs des zu untersuchenden Ausgangs-Eingangs-Paars (e, a)
- S : Vereinigungsmenge aller Stellen und Eingänge aller Komponenten C
- T : Vereinigungsmenge aller Transitionen und Ausgänge aller Komponenten C
- $m_0(s)$: Anfangsmarkierung der Stelle oder des Eingangs s
- $W(s, t)$ bzw. $W(t, s)$: Gewicht der Prä- bzw. Postkante zwischen s und t

Die Rückwirkungsfreiheit des Ausgangs-Eingangs-Paars (a, e) ist sichergestellt, wenn sich die folgenden Größen

- S_{ref} Referenzstellenmenge $S_{ref} \subseteq S$ mit $e \in S_{ref}$,
- $V: S_{ref} \rightarrow \mathbb{N} \setminus \{0\}$ Gewichtungsfunktion, die jeder Stelle und jedem Eingang der Referenzstellenmenge eine positive ganze Zahl zuordnet,
- $b \in \mathbb{N}$ eine nicht negative ganze Zahl

so bestimmen lassen, dass folgende Bedingungen erfüllt sind:

$$\sum_{s \in S_{ref}} V(s) \cdot m_0(s) = V(e) \cdot k - b \quad (1)$$

$$\sum_{s \in S_{ref}} V(s) \cdot W(s, a) \geq V(e) - b \quad (2)$$

$$\sum_{s \in S_{ref}} V(s) \cdot (W(s, t) - W(t, s)) = 0 \text{ für alle } t \in T \quad (3)$$

Wenn sich eine Lösung finden lässt, gibt b den minimal verbleibenden Platz im Eingang e . Für das gezeigte Beispiel in Abb. 3 gilt:

$$a = a_1; \quad e = e_1; \quad k = 1 \quad (4)$$

Als Kandidat für die Referenzstellenmenge kommt die Menge der auf dem Kreis liegenden Stellen und Eingänge, die den betrachteten Eingang enthalten, in Frage, weil die Struktur des Kreises eine Rückkopplung des Eingangs auf den Ausgang enthält. Dies ist eine Grundvoraussetzung dafür, dass überhaupt eine Aussage über das Schaltverhalten eines Ausgangs in Abhängigkeit eines nachgeschalteten Eingangs möglich ist. Wir setzen:

$$S_{ref} = \{e_1; s_3; e_2; s_1; s_2\} \quad (5a)$$

$$V(s) = 1 \quad \forall s \in S_{ref}; \quad b = 0 \quad (5b)$$

und prüfen:

$$(1) \Rightarrow V(e_1) \cdot m_0(e_1) + V(s_3) \cdot m_0(s_3) + V(e_2) \cdot m_0(e_2) + V(s_1) \cdot m_0(s_1) + V(s_2) \cdot m_0(s_2) = \\ 0 + 0 + 0 + 1 \cdot 1 + 0 = V(e) \cdot k - b = 1 \cdot 1 - 0 \quad (6.1)$$

$$(2) \Rightarrow V(s_2) \cdot W(s_2, a) \geq V(e_1) - b \\ \Rightarrow 1 \cdot 1 \geq 1 - 0 \quad (6.2)$$

$$(3) \Rightarrow V(s_1) \cdot (W(s_1, t_1) - W(t_1, s_1)) + V(s_2) \cdot (W(s_2, t_1) - W(t_1, s_2)) = 0 \wedge \\ V(e_2) \cdot (W(e_2, t_2) - W(t_2, e_2)) + V(s_1) \cdot (W(s_1, t_2) - W(t_2, s_1)) = 0 \wedge \\ V(s_3) \cdot (W(s_3, a_2) - W(a_2, s_3)) + V(e_2) \cdot (W(e_2, a_2) - W(a_2, e_2)) = 0 \wedge \\ V(e_1) \cdot (W(e_1, t_3) - W(t_3, e_1)) + V(s_3) \cdot (W(s_3, t_3) - W(t_3, s_3)) = 0 \wedge \\ V(s_2) \cdot (W(s_2, a_1) - W(a_1, s_2)) + V(e_1) \cdot (W(e_1, a_1) - W(a_1, e_1)) = 0 \\ \Rightarrow 1 \cdot (1 - 0) + 1 \cdot (0 - 1) = 0 \wedge \quad 1 \cdot (1 - 0) + 1 \cdot (0 - 1) = 0 \wedge \\ 1 \cdot (1 - 0) + 1 \cdot (0 - 1) = 0 \wedge \quad 1 \cdot (1 - 0) + 1 \cdot (0 - 1) = 0 \wedge \\ 1 \cdot (1 - 0) + 1 \cdot (0 - 1) = 0 \quad (6.3)$$

Die Gleichungen (6.1) bis (6.3) sind erfüllt. Daraus ergibt sich, dass die Kapazität des Eingangs e_1 mit $k = 1$ ausreichend ist, so dass Ausgang a_1 immer schalten kann, wenn die internen Vorbedingungen erfüllt sind.

4 Formaler Nachweis der Konsistenzregel

Für den Beweis wird auf eine Hilfskonstruktion zurückgegriffen. Statt der direkten Verbindung des Ausgangs a mit dem Eingang e über eine Postkante wird ein Netz bestehend aus zwei Stellen p und q sowie einer Transition t_0 gemäß Abb. 4 eingesetzt, das explizit die Kapazität k des Eingangs e nachmodelliert. Es ist leicht einzusehen, dass dadurch das gewünschte Verhalten erzwungen werden kann:

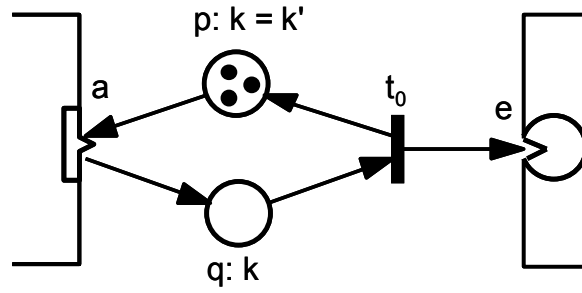


Abbildung 4: Hilfskonstrukt zur expliziten Modellierung der Eingangskapazität

Die beiden Stellen p und q der Kapazität k bilden zusammen eine S-Invariante. Die Stelle p weist eine Anfangsmarkierung von k Marken auf, d.h., sie ist vollbelegt. Für jeden Schaltvorgang von a wird aus p eine Marke entnommen, so dass maximal k Schaltvorgänge infolge möglich sind. Die Stelle q puffert die erzeugten Marken. Für jede vom Eingang e abgezogene Marke erhält der Ausgang a eine Schaltberechtigung in Form einer Marke in p zurück. Wenn gezeigt werden kann, dass die Stelle p *strukturell redundant* [Be1986] ist, bedeutet das, dass das Schaltverhalten des Netzes nach Entfernen von p einschließlich aller Kanten von und nach p identisch mit dem Schaltverhalten des gegebenen Netzes ist. Mit anderen Worten heißt das, dass die Markierung von p niemals das Schalten von a verhindert hat. Demnach sind in p stets genügend Marken vorhanden, um ein Schalten von a zu ermöglichen. Da diese Marken die verbleibende Kapazität im Eingang e modellieren, folgt daraus, dass die Kapazität stets ausreichend ist. Es muss angemerkt werden, dass die Kante von p nach a kein legales CNet/PNet-Konstrukt darstellt. Dieses Hilfskonstrukt dient hier nur der Stützung der Beweisführung.

In [Be1985] [Be1986] wird ausgeführt, dass die Stelle $p \in S$ eines Petri-Netzes $PN = (S, T, W, m_0)$ genau dann *strukturell redundant* ist, wenn eine (möglicherweise leere) Teilmenge $I \subseteq S$, eine Gewichtungsfunktion $V : I \cup \{p\} \rightarrow N \setminus \{0\}$ sowie ein $b \in N$ existieren, so dass die drei folgenden Bedingungen erfüllt sind:

$$V(p) \cdot m_0(p) - \sum_{s \in I} V(s) \cdot m_0(s) = b \quad (\text{i})$$

$$\forall t \in T : V(p) \cdot W(p, t) - \sum_{s \in I} V(s) \cdot W(s, t) \leq b \quad (\text{ii})$$

$$\forall t \in T : \exists c_t \in N : V(p) \cdot C(p, t) - \sum_{s \in I} V(s) \cdot C(s, t) = c_t \quad (\text{iii})$$

Darin bedeutet $C(p, t) = W(t, p) - W(p, t)$ den Markenzuwachs auf Stelle p beim Schalten der Transition t .

Sei $\Sigma = (C, S_0, T_0, F_0, m_{0,0})$ ein in CNet/PNet modelliertes System bestehend aus den Komponenten C , den freien Stellen S_0 und Transitionen T_0 sowie den Kanten F_0 und der Anfangsmarkierung $m_{0,0}$ der Stellen S_0 . Die Gesamtmenge aller Stellen und Eingangsstellen im System Σ ergibt sich zu

$$S = \left(\bigcup_{c \in C} S(c) \right) \cup S_0 .$$

Entsprechend bestimmt sich die Menge aller Transitionen und Ausgangstransitionen zu

$$T = \left(\bigcup_{c \in C} T(c) \right) \cup T_0 .$$

Betrachtet wird das verbundene Ausgangs-Eingangs-Paar $(a, e) \in F_0$, welches durch das Konstrukt aus Abb. 4 ersetzt wird. Damit ergibt sich für das modifizierte Petri-Netz des flachen Systems $\Sigma' = (S', T', W', m_0')$

$$S' = S \cup \{p, q\}$$

$$T' = T \cup \{t_0\}$$

$$W'(s, t) = \begin{cases} 1 & \text{für } (s, t) = (p, a) \\ 1 & \text{für } (s, t) = (q, t_0) \\ W(s, t) & \text{sonst} \end{cases}$$

$$W'(t, s) = \{(a, e) \mapsto 0; (a, q) \mapsto 1; (t_0, e) \mapsto 1; (t_0, p) \mapsto 1; \text{sonst} \mapsto W(t, s)\}$$

$$m_0'(s) = \{p \mapsto k; q \mapsto 0; \text{sonst} \mapsto m_0(s)\}$$

Die Anwendung der Bedingung (i) auf das modifizierte Petri-Netz Σ' liefert:

$$V(p) \cdot m_0'(p) - \sum_{s \in I} V(s) \cdot m_0'(s) = b \quad (4)$$

Mit Einsetzung von $m_0'(p) = k$, Beschränkung der Menge I auf eine Teilmenge der Stellen S des nicht-modifizierten Systems Σ und Umformung ergibt

$$\sum_{s \in I} V(s) \cdot m_0(s) = V(p) \cdot k - b \quad (4a)$$

Bei der Anwendung von Bedingung (ii) sind drei Fälle zu unterscheiden:

ii.1) $t = a$

$$V(p) \cdot \underbrace{W'(p, a)}_{=1} - \sum_{s \in I} V(s) \cdot W'(s, a) \leq b \quad (5)$$

$$\sum_{s \in I} V(s) \cdot W(s, a) \geq V(p) - b \quad (5a)$$

ii.2) $t = t_0$

$$V(p) \cdot \underbrace{W'(p, t_0)}_{=0} - \sum_{s \in I} V(s) \cdot W'(s, t_0) \leq b \quad (5b)$$

Ungleichung (5b) ist trivial erfüllt, da $W'(s, t_0)$ nur für $s = q$ einen Wert ungleich null liefert, da aber $q \notin I$ ist, ergibt sich die Summe insgesamt zu null.

ii.3) $t \in T' \setminus \{a, t_0\}$

$$V(p) \cdot \underbrace{W'(p, t)}_{=0} - \sum_{s \in I} V(s) \cdot W'(s, t) \leq b \quad (5c)$$

Ungleichung (5c) ist ebenfalls trivial erfüllt, da p keine Vorkanten außer zu a hat, ergibt sich der Minuend der linken Seite zu null. Die Summe im Subtrahenden kann nur positive Werte oder null annehmen, so dass sich die linke Seite von (5c) insgesamt ein negativer Wert oder null ergibt. Da b definitionsgemäß nicht negativ ist, ist Ungleichung (5c) stets erfüllt. Für Bedingung (iii) müssen ebenfalls diese drei Fälle unterschieden werden:

iii.1) $t = a$

$$V(p) \cdot \underbrace{W'(a, p)}_{=0} - \underbrace{W'(p, a)}_{=1} - \sum_{s \in I} V(s) \cdot (W'(a, s) - W'(s, a)) = c_a \quad (6)$$

$$-V(p) + \sum_{s \in I} V(s) \cdot (W'(s, a) - W'(a, s)) = c_a \quad (6a)$$

iii.2) $t = t_0$

$$V(p) \cdot \underbrace{W'(t_0, p)}_{=1} - \underbrace{W'(p, t_0)}_{=0} - \sum_{s \in I} V(s) \cdot (W'(t_0, s) - W'(s, t_0)) = c_{t_0} \quad (7)$$

Die Transition t_0 hat gemeinsame Kanten mit den Stellen p und q sowie dem Eingang e . Da gefordert wird, dass $p, q \notin I$ und $e \in I$ vereinfacht sich Gleichung (7) zu

$$V(p) - V(e) = c_{t_0} \quad (7a)$$

iii.3) $t \in T' \setminus \{a, t_0\}$

$$V(p) \cdot (W'(t, p) - W'(p, t)) - \sum_{s \in I} V(s) \cdot (W'(t, s) - W'(s, t)) = c_t \quad (8)$$

Da p nur gemeinsame Kanten mit a und t_0 besitzt, ergibt sich der Minuend in Gleichung (8) zu null. Wie oben kann W' durch W ausgedrückt werden und es ergibt sich:

$$\sum_{s \in I} V(s) \cdot (W(s, t) - W(t, s)) = c_t \quad (8a)$$

Da gezeigt werden soll, dass die Hilfsstelle p vollständig isoliert werden kann, d.h., dass alle Kanten von und nach p entfallen können, muss für alle $t \in T'$ gelten: $c_t = 0$. Damit ergibt sich aus Gleichung (7a)

$$V(p) = V(e) \quad (9)$$

Des Weiteren vereinfachen sich die Gleichungen (4a) (5a) zu

$$\sum_{s \in I} V(s) \cdot m_0(s) = V(e) \cdot k - b \quad (10)$$

$$\sum_{s \in I} V(s) \cdot W(s, a) \geq V(e) - b \quad (11)$$

Dies entspricht den aufgestellten Bedingungen (1) und (2). Die Einsetzung von Gleichung (9) in (6a) (8a) führt zu

$$\sum_{s \in I} V(s) \cdot (W'(s, a) - W'(a, s)) = V(e) \quad (12)$$

$$\sum_{s \in I} V(s) \cdot (W(s, t) - W(t, s)) = 0 \text{ für } t \in T \setminus \{a\} \quad (13)$$

Es gilt $W'(e, a) = W(e, a) = 0$. Aber wegen $W'(a, e) \neq W(a, e)$ kann in Gleichung (12) W' nicht direkt durch W ersetzt werden. Stattdessen muss der Summand für den Fall $s = e$ wie folgt korrigiert werden:

$$\sum_{s \in I} V(s) \cdot (W(s, a) - W(a, s)) + V(e) \cdot \underbrace{W(a, e)}_{=1} - V(e) \cdot \underbrace{W'(a, e)}_{=0} = V(e) \quad (14)$$

Gleichung (14) vereinfacht sich zu

$$\sum_{s \in I} V(s) \cdot (W(s, a) - W(a, s)) = 0$$

und führt unter Berücksichtigung von Gleichung (13) auf die geschlossene Form

$$\sum_{s \in I} V(s) \cdot (W(s, t) - W(t, s)) = 0 \text{ für alle } t \in T, \quad (15)$$

was der dritten notwendigen und hinreichenden Bedingung gemäß Gleichung (3) für die konsistente Verschaltung eines Ausgangs-Eingangs-Paars entspricht. ■

5 Zusammenfassung

In diesem Beitrag wurde das CNet-Komponentenmodell als komponentenbasierte Modellierungssprache für verteilte Steuerungssysteme, deren Schnittstellenelemente sich aus den Elementen der Petri-Netze ableiten, kurz vorgestellt. Weitere und tiefgreifendere Details sind [Wu2002] [HW2004] [HW2005a] [HW2005b] zu entnehmen. Ein Vorteil von Petri-Netzen und daraus abgeleiteter Formalismen ist die Möglichkeit, mit formalen Mitteln bestimmte Eigenschaften nachzuweisen und so sicherheitskritische Anlagen bereits während des Entwurfs verifizieren zu können. Allgemein bekannt ist die Erreichbarkeitsanalyse der klassischen flachen Petri-Netze [Ab1990] sowie das als „Zustandsexplosion“ bekannte Problem der exponentiell wachsenden Erreichbarkeitsmenge bei Zunahme der Anzahl an Stellen. Die hier vorgestellten CNet-Komponenten begegnen dem Problem in der Weise, dass sie jeweils nur kleinere Netze mit wenigen Elementen enthalten, für die eine Erreichbarkeitsanalyse mit vertretbarem Aufwand realisierbar ist. Außerdem fällt dieser Analyseaufwand nur *einmal* während des Entwurfs der Komponente und nicht bei jeder Verwendung an. Das bekannte Verfahren zur Erreichbarkeitsanalyse wurde erweitert, so dass es die CNet-Schnittstellenelemente berücksichtigt [HW2005b]. Das

Ergebnis wird „Erweiterter Erreichbarkeitsgraph“ genannt. Der Hauptteil dieses Beitrag befasst sich mit der korrekten Verschaltung der Komponenten untereinander. Dazu wird eine *Konsistenzregel* formuliert, mit der gezeigt werden kann, dass Marken, die an Ausgangstransitionen erzeugt werden, stets von der angeschlossenen Stelle aufgenommen werden können, so dass dies nicht vor dem Schalten geprüft werden muss. Der Wegfall dieser Prüfung führt zu einer deutlichen Vereinfachung bei der Codegenerierung und letztendlich auch zur Reduktion des notwendigen Datenverkehrs im Netzwerk oder auf dem Feldbus bei der verteilten Ausführung. Bisher gab es informelle Entwurfsregeln [Wu2002], die der Anwender zu befolgen hatte, um die Rückwirkungsfreiheit von Ausgängen zu garantieren. Weiterführende Überlegungen bezüglich der Formulierung der Konsistenzregel werden dahingehend angestellt, die Bedingungen so umzuformulieren, dass sie ohne Kenntnis der inneren Beschreibung der Komponenten geprüft werden können. Dazu ist es erforderlich, die strukturellen Informationen, die in den Bedingungen (1-3) geprüft werden, auf einen oder mehrere für die Komponente charakteristische Werte zu konzentrieren und auf die Schnittstelle abzubilden.

6 Literatur

- [Ab1990] Abel, Dirk: *Petri-Netze für Ingenieure: Modellbildung und Analyse diskret gesteuerter Systeme*. Springer-Verlag, Berlin, 1990
- [Be1985] Berthelot, Gérard: *Checking Properties of Nets Using Transformations*. Lecture Notes in Computer Science. **Volume** (222): Advances in Petri Nets 1985, S. 19-40, 1986
- [Be1986] Berthelot, Gérard: *Transformations and Decompositions of Nets*. Lecture Notes in Computer Science. **Volume** (254): Advances in Petri Nets 1986, S. 359-376, 1987
- [HW2004] Hagge, Nils; Wagner, Bernardo: *Mapping reusable control components to Java language constructs*. Proc. 2nd IEEE International Conference on Industrial Informatics, S. 108-113, Berlin, 2004
- [HW2005a] Hagge, Nils; Wagner, Bernardo: *Java Code Patterns for Petri Net Based Behavioral Models*. Proc. 3rd IEEE International Conference on Industrial Informatics, Perth, 2005
- [HW2005b] Hagge, Nils; Wagner, Bernardo: *A New Function Block Modeling Language Based on Petri Nets for Automatic Code Generation*. IEEE Transactions on Industrial Informatics **Volume** (1), S. 226-237, 2005
- [Wu2002] Wurmus, Harald: *CNet – Komponentenbasierter Entwurf verteilter Steuerungssysteme mit Petri-Netzen*. Dissertation, Universität Hannover, Fachbereich Informatik, Hannover, 2002