Loose Semantics of Petri Nets*

Julia Padberg[1] and Hans-Jörg Kreowski[2]

[1] Technische Universität Berlin,
Fakultät IV, Informatik und Elektrotechnik,
Berlin, Germany
padberg@cs.tu-berlin.de
[2] Universität Bremen,
Fachbereich für Mathematik und Informatik,
Bremen, Germany
kreo@tzi.de

Abstract. In this paper, we propose a new,loose semantics for place/transition nets based on transition systems and generalizing the reachability graph semantics. The loose semantics of a place/transition net reflects all its possible refinements and is given as a category of transition systems with alternative sequences of events over the net. The main result states that each plain morphism between two place/transitions nets induces a free construction between the corresponding semantic categories.

1 Introduction

Petri nets are one of the most thoroughly investigated approaches with a multitude of extensions and variants. They are one of the most prominent specification techniques for modeling concurrency and have a wide range of application areas in practice. In this paper, we introduce a new semantics for Petri nets which is based on transition systems. The semantics of a net is given by a class of models corresponding to all possible refinements of a net with respect to transition refinement. In this sense, it is a loose semantics as known and well accepted in the area of data type specification (see, e.g. [24]).

The semantics we define here is developed in view of system specification. It is suitable for relating different stages of refinement. This is obviously important for the vertical structuring, but as well for horizontal structuring with abstraction mechanisms like parameterization and modularization.

The reachability graph is a standard model of a place/transition net describing all possible sequences of firings of transitions starting from an initial marking. Our new semantics generalizes this net semantics in such a way that a firing of a transition can be refined by sequences of events. Moreover, we allow alternative possibilities for each such refinement. Typical examples of alternative sequences are the interleavings of independent events. Altogether, the loose semantics of a place/transition net consists of the class of transition systems with alternative sequences of events including the reachability graph. This class forms a category

* Research partially supported by the EC Research Training Network SegraVis.

H.-J. Kreowski et al. (Eds.): Formal Methods (Ehrig Festschrift), LNCS 3393, pp. 370–384, 2005.

in a natural way. As the main result of the paper, we show that each plain morphism between two place/transition nets induces a free construction between the corresponding semantic categories. This is the key result that allows one to consider Petri nets as building blocks of parameterization and modularization.

We continue this paper by introducing state transition systems that capture our idea of alternatives and refinement. In Section 3, we show that transitions systems over Petri nets can be considered as a loose semantics of the corresponding Petri net. Next we show that we obtain a free functorial construction of the place/transition net semantics, based on contravariant forgetful functor. Subsequently in Section 5 we treat the relation to other approaches in some detail and discuss at last the impact of a loose semantics for Petri nets in Section 6.

2 Transition Systems with Alternative Sequences

In this section, we recall the notion of state transition systems and add a new feature to them: a relation of alternative sequences of events. Transition systems with alternative sequences will be combined with place/transition nets in the next section.

A state transition system $STS = (S, E, TS, \widehat{s})$ is given by a set of states S, a set of events E, the set of transitions $TS \subseteq S \times E \times S$, and the initial state $\widehat{s} \in S$.

If one reads the events along the paths in state transition systems, one gets sequences of events. More formally, we write $s_0 \xrightarrow{w} s_n$ if there is some sequence of transitions $(s_{i-1}, e_i, s_i) \in TS$ for $i = 1, .., n$ and $w = e_1 e_2 ... e_n \in E^*$ or if $w = \lambda$ and $s_0 = s_n$.

Next we want to consider some of these sequences of events as alternatives to each other. To make this precise, let $AS \subseteq E^* \times E^*$ be some relation on E^* and AS^{Con} denote its congruence closure, i.e. the closure of AS that is reflexive, symmetric, transitive, and congruent with respect to concatenation. Moreover, let $E^\diamond$ denote the quotient factoring E^* through AS^{Con} and $[\_] : E^* \to E^\diamond$ the canonical function with $[\_](w) = [w]$ for all $w \in E^*$ where $[w] = \{w' | (w, w') \in AS^{Con}\}$ is the congruence class of $w \in E^*$.

Definition 1 (Transition Systems with Alternative Sequences). *A transition system with alternative sequences $TSA = (S, E, TS, \widehat{s}, AS)$ is given by a state transition system $(S, E, TS, \widehat{s})$ and the relation of alternative sequences $AS \subseteq E^* \times E^*$ subject to the following consistency condition:*

$$\forall w' \in [w] : s \xrightarrow{w} s' \iff s \xrightarrow{w'} s'$$

The consistency condition ensures that alternatives are alternatives at all states. So, they are global alternatives in the following sense: Whenever there is a state where the sequence w occurs the alternative sequence $w' \in [w]$ has to occur as well.

Next we examine morphisms between transition systems with alternative sequences. We allow mapping one event $e_1 \in E_1$ to a congruence class of sequences of events by a morphism $f_E : E_1 \to E_2^\diamond$ with $f_E(e_1) = [w]$. This

denotes the refinement of one event by alternative sequences of events. The morphism $f_E : E_1 \to E_2^\diamond$ can be extended uniquely by ${f_E}^\diamond : E_1^\diamond \to E_2^\diamond$ defined for $w = e_1 \cdot ... \cdot e_n \in E_1^*$ by $f^\diamond([w]) = f(e_1) \cdot ... \cdot f(e_n)$, where the concatenation of congruence classes is defined by the congruence class of the concatenation, i.e. $[u] \cdot [v] = [uv]$.

Definition 2 (TSA-Morphisms). *Given transition systems with alternative sequences $TSA_i = (S_i, E_i, TS_i, \widehat{s}_i, AS_i)$ for $i = 1, 2$, then a TSA-morphism is given by $f : TSA_1 \to TSA_2$ with $f = (f_S, f_E)$ and $f_S : S_1 \to S_2$ and $f_E : E_1 \to E_2^\diamond$ such that the following conditions hold:*

1. Existence of a path: *For all $(s_1, e_1, s_1') \in TS_1$ and for all $e_2^1 \cdot e_2^2 \cdot ... \cdot e_2^n \in f_E(e_1)$ there is a path $f_s(s_1) \xrightarrow{e_2^1} s_2^1 \xrightarrow{e_2^2} s_2^2 \xrightarrow{*} s_2^n \xrightarrow{e_2^n} f_s(s_1')$.*
2. Reachability of initial state: *We have $\widehat{s_2} \xrightarrow{*} f_S(\widehat{s_1})$.*
3. Preservation of alternatives: *Given $(w, w') \in AS_1$ then we have $f_E^\diamond([w]) = f_E^\diamond([w'])$ for the unique extension ${f_E}^\diamond : E_1^\diamond \to E_2^\diamond$ and $w \in E_1^*$.*

Then we obtain:

- *Composition $g \circ f : TSA_1 \to TSA_3$ of the morphisms $f : TSA_1 \to TSA_2$ and $g : TSA_2 \to TSA_3$ is given by the composition of its components with $(g \circ f)_S = g_S \circ f_S$ and $(g \circ f)_E : g_E^\diamond \circ f_E$, where ${g_E}^\diamond : E_2^\diamond \to E_3^\diamond$ is the unique extension.*
- *Identity $id_{TSA} : TSA \to TSA$ is given by $id_{TSA} = (id_S, [\_]_E)$.*

Hence, we have the category **TSA** *of transition systems with alternative sequences.*

Note, condition 3 obviously implies $f_E^\diamond([w]) = f_E^\diamond([w'])$ for any $w' \in [w] \in E_1^\diamond$ (see [21]) and the composition is well-defined as we have congruence with respect to concatenation.

Example 1 (Transition Systems with Alternative Sequences). Here we give a short example of some transition systems with alternative sequences, where we concentrate on the events and depict the states merely as $\bullet$, and the initial state by $\to \bullet$. The numbers adjacent to the states are merely used to illustrate morphisms later on. First, we investigate the examples in Fig. 1 to illustrate our notion of morphisms and subsequently we give an interpretation of the example.

All states are mapped injectively. TSA_1 is mapped to TSA_2 by f, where $f_E(s) = [s] = \{s\}$ and $f_E(d) = [d] = \{d\}$. Preservation of alternatives is satisfied as $AS_1 = \emptyset$. The TSA morphism $g : TSA_2 \to TSA_3$ is defined for the events by $g_E(s) = [t] = \{t, uv\}$, $g_E(s') = [s'] = \{s'\}$, $g_E(d) = [d] = \{d\}$, and $g_E(d) = [d] = \{d\}$. Preservation of alternatives is satisfied since we have $g_E^\diamond([sd]) = [td] = \{td, uvd, s'd'\} = [s'd'] = g_E^\diamond([s'd'])$. The composition $g \circ f$ is for the events obviously given by $g_E^\diamond \circ f_E(s) = g_E^\diamond([s]) = [t] = \{t, uv\}$ and $g_E^\diamond \circ f_E(d) = g_E^\diamond([d]) = [d] = \{d\}$. Again preservation of alternatives is satisfied since $(g_E \circ f_E)^\diamond([sd]) = g_E^\diamond \circ f_E^\diamond([sd]) = [td] = \{td, uvd, s'd'\} = [s'd'] = (g_E \circ f_E)^\diamond([s'd'])$. The interpretation of this example is that the transition system TSA_1 describes a simple system with the following events s for start, d for distribute, r for

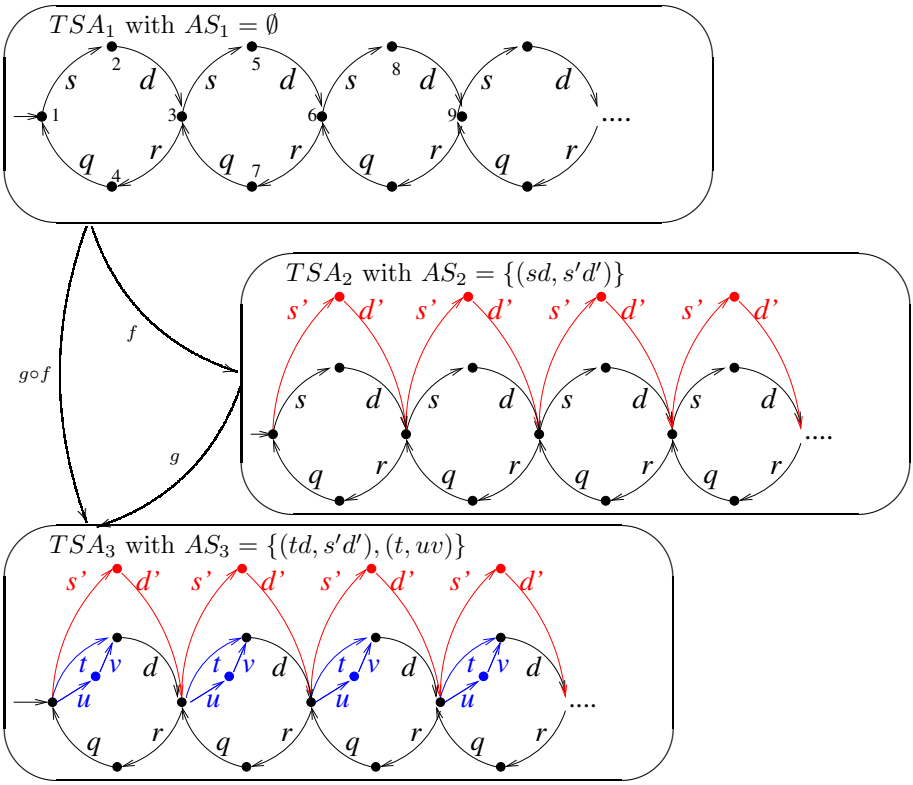


Fig. 1. Transition systems.

receive, and q for quit. These events follow each other as depicted in Fig. 1. The transition system TSA_2 states that the sequences of events sd and $s'd'$ are alternatives. Whenever one of both occurs at a certain state the other does so too. This describes independently of the syntactic specification that a system has different, but equally desired sequences of events. As in the case of our example, they need not be induced by single events. In TSA_2 two alternatives for starting and distributing namely sd or $s'd'$ result in the same state, and they do that in any case. Nevertheless neither s and s' nor d and d' are alternatives. By the morphism $g : TSA_2 \to TSA_3$ we refine the event s by $[t] = \{t, uv\}$, that is s can be expressed either by the event t or the sequence uv.

3 Transition Systems with Alternative Sequences over Net Systems

In this section, we associate place/transition systems with transition systems with alternative sequences. We use place/transition nets in the usual way with weighted arcs so that the pre- and post-domains of transitions as well as the

markings are multisets over the sets of places (as the algebraic notation in [14]). Given a set P, let the set of finite multisets over P be denoted by $P^{\oplus}$.

Then a place/transition net is given by $N = (P, T, pre, post, \widehat{m})$ where P is the set of places, T the set of transitions, $pre, post : T \to P^{\oplus}$ are mappings associating a pre- and a post-domain to each transition, and $\widehat{m} \in P^{\oplus}$ is the initial marking.

The set of finite multisets over P is the free commutative monoid over P. An element $w \in P^{\oplus}$ can be presented either by the natural function $w : P \to \mathbb{N}$ or as a linear sum $w = \sum_{p \in P} \lambda_p \cdot p$, and we can extend the usual operations and relations on $\mathbb{N}$ as $\oplus$, $\ominus$, $\leq$, and so on to $P^{\oplus}$. Moreover, we need to state how often is a basic element with in an element of the free commutative monoid given. We define this for an element $p \in P$ and a linear sum $w = \sum_{p \in P} \lambda_p \cdot p \in P^{\oplus}$ with $w_{|p} = \lambda_p$ for $p \in P^{\oplus}$ and $w_{|Q} = \sum_{p \in Q} \lambda_p \cdot p$ for a subset $Q \subseteq P$.

The pre-set $\bullet x$ and the post-set $x\bullet$ are defined as usual, and so is the set of reachable markings $[\widehat{m}>$.

The set of enabled transitions is $[T> = \{t \in T | m[t> m' \text{ for some } m \in [\widehat{m}>\}$.

We now define transitions systems that can be viewed as models of a net, where a refinement of the enabled transitions and the representation of the states relate the net to the transition system. In particular, we allow refinements of transitions to be equivalence classes of alternative sequences.

Definition 3 (Transition Systems with Alternative Sequences over Place/Transition Nets). *A transition system with alternative sequences TSA over a place/transition net $N = (P, T, pre, post, \widehat{m})$ consists of a transition system with alternative sequences $TSA = (S, E, TS, \widehat{s}, AS, rep, ref)$ and two functions $rep : [\widehat{m}> \to S$ and $ref : [T> \to E^{\diamond}$ subject to the following conditions:*

1. Representation of markings: *The function $rep : [\widehat{m}> \to S$ represents the reachable markings.*
2. Refinement of transitions: *The function $ref : [T> \to E^{\diamond}$ refines the transitions.*
3. Reachability of initial marking: *We have $\widehat{s} \xrightarrow{*} rep(\widehat{m})$.*
4. Existence of a path: *For all $m[t> m'$ with $m \in [\widehat{m}>$ we have a path for all $w \in ref(t)$ so that $rep(m) \xrightarrow{w} rep(m')$.*

Note that we may have $ref(t) = [\lambda]$ only for transitions where for all $m, m' \in [\widehat{m}>$ with $m[t> m'$ we have $rep(m) = rep(m')$. Next we establish the category of transition systems with alternative sequences over N. Hence, this category is the loose semantics of a net N.

Definition 4 (Category TSA(N) of Transition Systems with Alternative Sequences over N). *The category* **TSA(N)** *of transition systems with alternative sequences over the place/transition net $N = (P, T, pre, post, \widehat{m})$ is given by the class of transition systems $TSA = (S, E, TS, \widehat{s}, AS, rep, ref)$ over N, and by TSA-morphisms $f : TSA_1 \to TSA_2$ with $rep_1 : [\widehat{m_1}> \to S_1$ and $ref_1 : [T_1> \to E_1^{\diamond}$ (resp. $rep_2 : [\widehat{m_1}> \to S_2$ and $ref_2 : [T_1> \to E_2^{\diamond}$) satisfying the following conditions:*

1. Preservation of representation: $f_S \circ rep_1 = rep_2$.
2. Preservation of transition refinement: $f_E^{\diamond} \circ ref_1 = ref_2$.

Now we have a class of transition systems for each net. Moreover, it is a category so we have morphisms, that denote refinements of events with alternatives. To illustrate this new type of net semantics, we now present the well-known producer-consumer net and discuss its loose semantics.

Example 2 (Producer-Consumer). In Fig. 2 we have the well-known producer-consumer net with the transitions s for start, d for distribute, r for receive, and q for quit. This net is obviously closely related to the transition systems in Fig. 1. This producer-consumer net PCN denotes a part of the category **TSA**(**PCN**) of transition systems over this net.

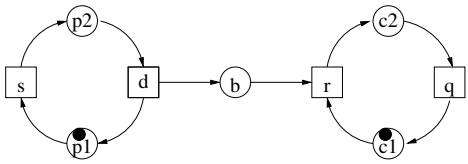


Fig. 2. Producer-Consumer net PCN.

The transition system TSA_1 is extended by the representation function $rep_1 : [\widehat{m}> \rightarrow S_1$ with $rep_1(p1 \oplus c1) = 1$, the follower marking is mapped by $rep_1(p2 \oplus c1) = 2$, $rep_1(p1 \oplus b \oplus c1) = 3$, and so one, where the numbers denote the states of transition system TSA_1 in Fig. 1. The transitions are refined trivially by themselves, $ref_1(s) = [s] = \{s\}$, $ref_1(d) = [d] = \{d\}$, and so forth. The transition systems TSA_2 and TSA_3 with suitable representations and refinements are clearly transition systems over PCN.

The transition system TSA_1 is isomorphic to the reachability graph of PCN and hence it is initial in this category (see Section 5).

The loose semantics for some place/transition net N comprises all possible refinements, where we use refinement in a very broad sense: A transition can be refined by various alternatives of event sequences including the empty sequence. The only requirement is that the source of these event sequences needs to be the representation of the marking before firing the transition, and the target of the sequence needs to be the representation of the marking after firing the transition. The initial object is the usual reachability graph $R(N)$ of a net N (see Section 5). So, the classical semantics of a net is a distinguished member of the loose semantics and any transition system TSA in **TSA**(**N**) is a refinement of $R(N)$, as there is a unique morphism from $R(N)$ to TSA.

4 Free Construction of the Loose Semantics over Plain Morphisms

Based on the algebraic notion of Petri nets [14] we use simple homomorphisms that are generated over the set of places. These morphisms map places to places and transitions to transitions. Morphisms are the basic entity in category theory; they can present the internal structure of objects and relate the objects. So they are the basis for the structural properties a category may have and can be used successfully to define various structuring techniques.

Definition 5 ((Plain) Morphisms). *A plain morphism $f : N_1 \to N_2$ is given by $f = (f_P, f_T)$ with $f_P : P_1 \to P_2$ and $f_T : T_1 \to T_2$ so that $pre_2 \circ f_T = f_P^{\oplus} \circ pre_1$ and post analogously.*

Moreover, for the initial marking we have for all $p \in P_1$:
$\widehat{m_1}(p) \leq \widehat{m_2}(f_P(p))$ for the natural function associated to a multiset.

Lemma 1 (Plain morphisms preserve firing). *Plain morphisms $f : N_1 \to N_2$ preserve firing in the following sense:*

$$m[t > m' \text{ implies } f_P^{\oplus}(m)[f_T(t) > f_P^{\oplus}(m) \text{ for } m, m' \in P_1^{\oplus} \text{ and } t \in T_1.$$

Then we define $\widehat{f_P} : [\widehat{m_1} > \to [\widehat{m_2} >$ with $\widehat{f_P}(m) = f_P^{\oplus}(m) \oplus m_2^R$ where we have $\widehat{m_2} = f_P^{\oplus}(\widehat{m_1}) \oplus m_2^R$.

Note, by induction over the length of the firing sequence we can show that $\widehat{f_P}$ is well-defined and preserves firing as well: $m[t > m'$ with $m \in [\widehat{m_1} >$ implies $\widehat{f_P}(m)[f_T(t) > \widehat{f_P}(m')$

Theorem 1 (Forgetful Functor of Transition Systems with Alternative Sequences over N). *A plain morphism $f : N_1 \to N_2$ induces the following forgetful functor (if necessary subscripted with the corresponding net morphism) $V = V_f : \mathbf{TSA}(\mathbf{N_2}) \to \mathbf{TSA}(\mathbf{N_1})$. This functor $V(TSA_2) = TSA_1$ is defined by $TSA_2 = (S_2, E_2, TS_2, \widehat{s_2}, AS_2, rep_2, ref_2)$ with $rep_2 : [\widehat{m_2} > \to S_2$ and $ref_2 : [T_2 > \to E_2^{\diamond}$, where $TSA_1 = (S_2, E_2, TS_2, \widehat{s_2}, AS_2, rep_1, ref_1)$ and we have*

- *the following representation*
  $rep_1 := rep_2 \circ \widehat{f_P} : [\widehat{m_1} > \to S_2$, and
- *the following refinement*
  $ref_1 := ref_2 \circ f_T : [T_1 > \to E_2^{\diamond}$.

A TSA-morphism $h : TSA_2 \to TSA_2'$ is mapped by $V(h) = h$.

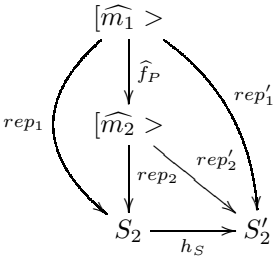
Proof. TSA_1 is a transition system over N_1:

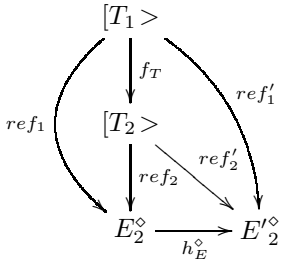
1. Representation: rep_1 is well-defined.
2. Refinement: ref_1 is well-defined.
3. Reachability of of initial state: $\widehat{s_2} \xrightarrow{*} rep_2(\widehat{m_2}) = rep_2(\widehat{f_P}(\widehat{m_1})) = rep_1(\widehat{m_1})$
4. Existence of a path:
   for any $m[t > m'$ with $m \in \widehat{m_1} >$ we have $\widehat{f_P}(m)[f_T(t) > \widehat{f_P}(m')$ and hence there is the path $rep_2(\widehat{f_P}(m)) \xrightarrow{ref_2 \circ f_T(t)} rep_2(\widehat{f_P}(m'))$ that is the path $rep_1(m) \xrightarrow{ref_1(t)} rep_1(m')$.

Given TSA-morphism $h : TSA_2 \to TSA_2'$ then $VP(h) : TSA_1 \to TSA_1'$ with $V(h) = h$ is well-defined:

1. preservation of representation :
   $h_s \circ rep_1 = h_S \circ rep_2 \circ \widehat{f_P} = rep_2' \circ \widehat{f_P} = rep_1'$

2. preservation of transition refinement:
   $h_E^\diamond \circ ref_1 = h_E^\diamond \circ ref_2 \circ f_T = ref_2' \circ f_T = ref_1'$ See the diagrams below:



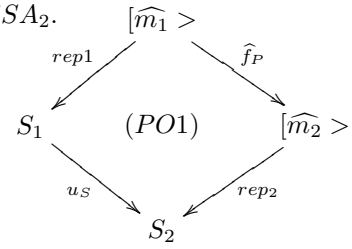


$\surd$

Theorem 2 (Free Functor of Transition Systems with Alternative Sequences over N). *A plain net morphism $f : N_1 \to N_2$ induces the following free functor $F = F_f : \mathbf{TSA}(\mathbf{N_1}) \to \mathbf{TSA}(\mathbf{N_2})$. This functor $F(TSA_1) = TSA_2$ is defined by $TSA_1 = (S_1, E_1, TS_1, \widehat{s_1}, AS_1, rep_1, ref_1)$ and $TSA_2 = (S_2, E_2, TS_2, \widehat{s_2}, AS_2, rep_2, ref_2)$ as given in the proof.*

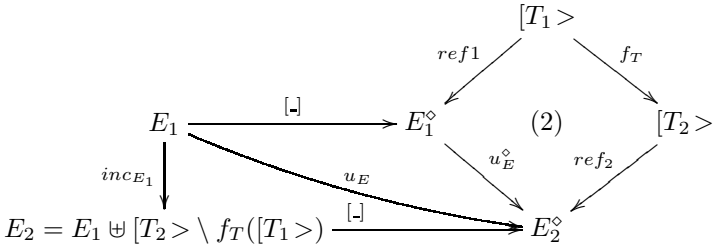
A TSA-morphism $h : TSA_1 \to TSA_1'$ is mapped by $F(h) = \overline{h}$.

Proof. 1. First we give the construction for TSA_2.



In **Set** we construct the pushout PO1 below and obtain S_2, and hence $rep_2 : [\widehat{m_2}> \rightarrowtail S_2$. We define $\widehat{s_2} = u_S(\widehat{s_1})$.
The construction of E_2 is given by $E_2 = E_1 \uplus [T_2> \setminus f_T([T_1>)$ in **Set**.
Then we define $AS_2 = AS_1 \uplus \{(w_1, w_2) |$ $f_T(t_1) = f_T(t_2)$ for some $w_1 \in ref_1(t_1)$ and $w_2 \in ref_1(t_2)\}$.
Then we have $E_2^\diamond$, and we define $u_E := \lfloor \_ \rfloor \circ inc_E : E_1 \to E_2^\diamond$ and hence $u_E^\diamond : E_1^\diamond \to E_2^\diamond$. This is well-defined as $E_1 \subseteq E_2$ and $AS_1 \subseteq AS_2$.
We now define $ref_2 : [T_2> \to E_2^\diamond$ by

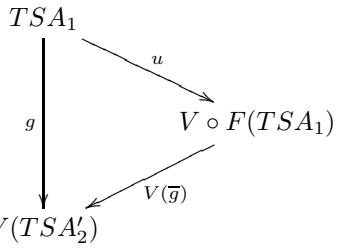
$$ref_2(t) := \begin{cases} [t] & t \notin f_T([T_1>) \\ u_E^\diamond \circ ref_1(t') & t = f_T(t') \text{ and } t' \in [T_1> \end{cases}.$$



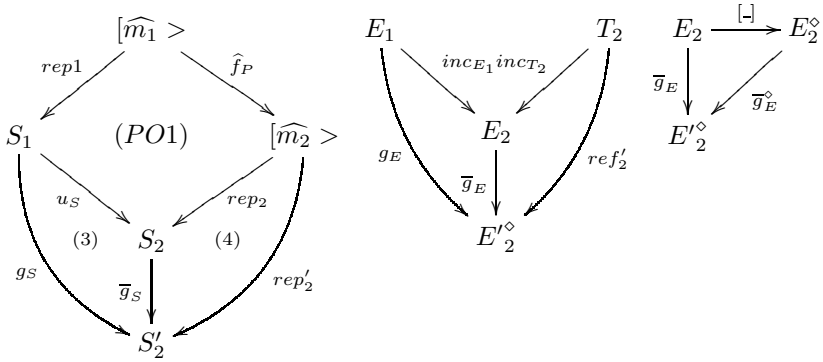
The square (2) commutes due to the quotient construction, since we have $E_2^\diamond = E_{2|AS_2^{Eq}}^*$.
We define $TS_2 \subseteq S_2 \times E_2 \times S_2$ using the transition system TS_1, all new firing paths of N_2 and then construct all alternatives event sequences inductively:

(a) If $(s, e, s') \in TS_1$
then $(u_S(s), e, u_S(s)) \in TS_2$.
(b) If $m[t> m'$ in N_2 and $t \notin f_t([T_1>)$
then $(rep_2(m), t, rep_2(m')) \in TS_2$.
(c) If $m[t> m'$ in N_2, $m \notin \widehat{f_P}(\widehat{m_1})$, $t = f_T(t_1)$, and with some $m_1[t_1> m_1'$
then for all $w = e_0....e_n \in ref_2(t)$
with $rep_1(m_1) \xrightarrow{e_0} s_1...s_n \xrightarrow{e_n} rep(m_1') \in TS_1$
we have $(rep_2(m), e_0, u_S(s_1)) \in TS_2$ and
$(u_S(s_n), e_n, rep_2(m')) \in TS_2$.
So, we have $TSA_2 = (S_2, E_2, TS_2, \widehat{s_2}, AS_2, rep_2, ref_2)$.
It is obviously well-defined.

2. We have a free construction:
There is $u : TSA_1 \to V \circ F(TSA_1)$.
Note that, $V \circ F(TSA_1) = V(TSA_2)$
$= (S_2, E_2, TS_2, \widehat{s_2}, rep_2 \circ \widehat{f_P}, ref_2 \circ f_T)$.
So we define $u = (u_S, u_E)$ where u_S and u_E are given in PO1 and (2) above.
u is well-defined as $u_S \circ rep_1 = rep_2 \circ \widehat{f_P}$ and $u_E^{\diamond} \circ ref_1 = ref_2 \circ f_T$.



Given $g : TSA_1 \to V(TSA_2')$ in $\mathbf{TSA(N_1)}$ defined by $g = (g_S, g_E)$ with $g_S : S_1 \to S_2'$ and $g_E : E_1 \to E_2'^{\diamond}$ then we have to construct $\overline{g} : TSA_2 \to TSA_2'$ in $\mathbf{TSA(N_2)}$. We have $\overline{g}_S$ induced by PO1.
And we obtain $\overline{g}_E : E_2 \to E_2'^{\diamond}$ due to the coproduct E_2.



$\overline{g}$ is well-defined in $\mathbf{TSA(N_2)}$, as (4) commutes,
and we have $\overline{g}_E^{\diamond} \circ ref_2 = \overline{g}_E^{\diamond} \circ [\_] \circ inc_{T_2} = \overline{g}_E \circ inc_{T_2} = ref_2'$.
Now we prove that $g = V(\overline{g}) \circ u$:
We have $g_S = \overline{g}_S \circ u_S$ due to (3).
And we have $\overline{g}_E^{\diamond} \circ u_E = \overline{g}_E^{\diamond} \circ [\_] \circ inc_{E_1} = \overline{g}_E \circ inc_{E_1} = g_E$ $\surd$

Example 3 (Refining the Producer-Consumer).

In Fig. 3 we again have the producer-consumer net PCN. This net is refined by the morphism $f : PCN \to PCN'$ to the producer-consumer net PCN where we can directly feed and empty the buffer. The morphism f maps the states and transitions injectively. Now, we have the categories **TSA**(**PCN**) and **TSA**(**PCN**′) that are related by the forgetful and the free functor as given in the Theorems 1 and 2. In Fig. 4 we illustrate the two functors, that form the adjunction $F \dashv V$.

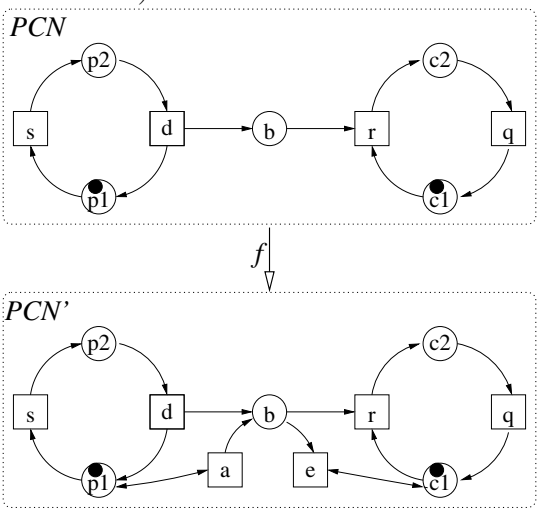


Fig. 3. PCN and PCN'.

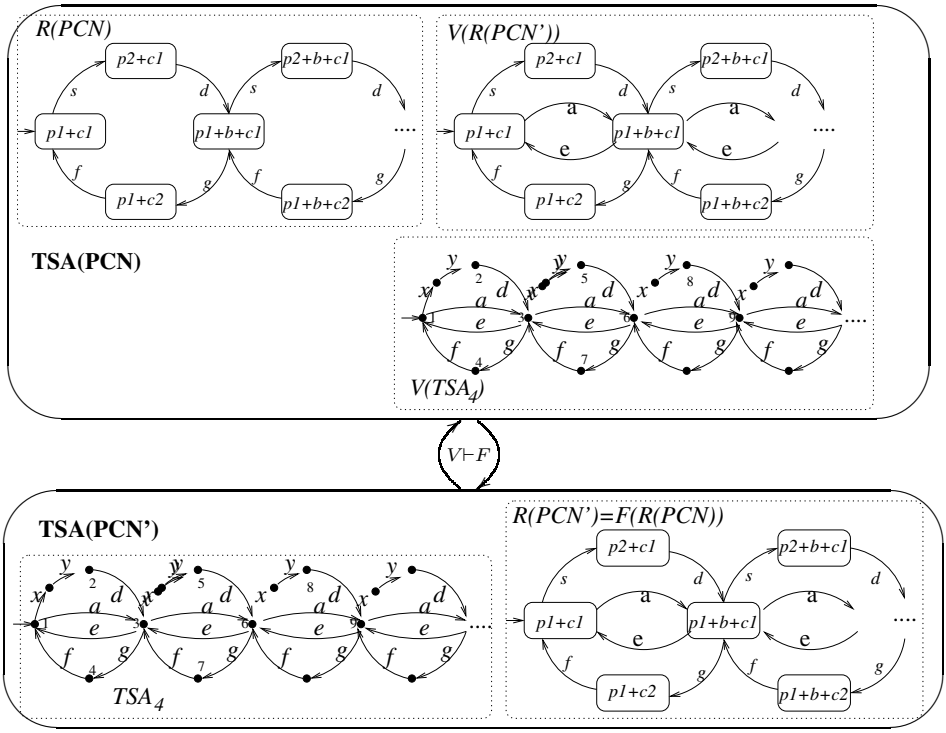


Fig. 4. Free and Forgetful Functor.

We have the transition system $R(PCN)$ in **TSA**(**PCN**), that is the reachability graph of the net PCN. This transition system $R(PCN)$ is mapped by the free functor F:**TSA**(**PCN**)→**TSA**(**PCN**′) to the transition system $F(R(PCN))$. This is isomorphic to the transition system $R(PCN')$. $F(R(SynPCN))$ is constructed by adding the reachable markings and the new transitions in the net N_2. The forgetful functor V : **TSA**(**PCN**′) → **TSA**(**PCN**) maps the transition system TSA_4 to the transition system $V(TSA_4)$ and $R(PCN')$ to $V(R(PCN')$ by keeping the transition system and redirecting the representation and the refinement functions.

5 Related Work

In the course of the last 40 years there has been developed a lot of different Petri net semantics: reachability or marking graph [23, 6], event structures [17, 3, 18], trace languages [13, 10, 15], partial orders semantics [2, 12], and others more. All these semantics have in common that they relate a Petri net to one semantic object.

In this section we relate our loose semantics to the two closest Petri net semantics, namely reachability graph and trace languages, in a provisional way. Since most of the above semantics are related to each other in a significant way (see [25, 19]) the results from the discussion below can be adopted accordingly.

Reachability Graph of Place/Transition Net Systems. The reachability graph of a place/transition net is given by the reachable markings and the firing transitions in-between. Hence, a suitable definition of the reachability graph is a transition systems with the empty alternative sequence of events. So, more formally the reachability graph $R(N) = ([\widehat{m} >, T, TS, \widehat{m}, \emptyset)$ with $TS \subseteq [\widehat{m} > \times T \times [\widehat{m} >$ defined by $TS = \{(m, t, m') | m[t > m'$ for any $m \in [\widehat{m_1} >\}$ of a place/transition net N is a TSA over N, where $rep(m) = M$ and $ref(t) = [t] = \{t\}$. Moreover, the reachability graph $R(N)$ is initial in the category **TSA**(**N**) for the proof see [21]. This means, that every transition system over N is **TSA**(**N**) can be considered a refinement of the reachability graph along a unique morphism.

Trace Equivalences. The relation to trace equivalences is the following.

Local trace equivalences [10, 15] denote the set of independent events following a sequence of events. So in a sense the interleaving of independent events are alternative sequences. But our approach states alternatives globally (in Definition 1).

Let us denote with $|\_|_e : E^* \to \mathbb{N}$ the family of functions, that counts the number of times an event $e \in E$ occurs in a sequence.

So given a transition system with alternative sequences, we can compute multisets of independent events. A multiset $m \in E^{\oplus}$ consists of independent events, if for any linearization $v \in Lin(m) = \{w \in E^* | m_{|e} = |w|_e$ for all $e \in E\}$ we have $Lin(m) \subseteq [v]$.

The other way round we can consider for the set of alternatives AS the set of linearization $Lin(m)$ of each multiset $m \in M$, the set M of multisets of independent events i.e. $AS = \bigcup_{m \in M} Lin(m) \times Lin(m)$.

6 Discussion of the Impacts of a Loose Semantics

Parameterized Petri Nets Based on the ideas of parameterization for data types, Petri nets can be parameterized by distinguishing a subnet as the parameter. Analogously to algebraic specifications, we map the formal parameter net PAR by an inclusion to the target net TAR. In Fig. 5 we have the formal parameter net, that denotes the transition s can be replaced by an actual parameter net.

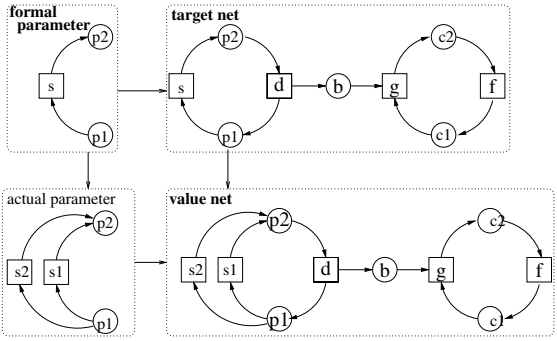
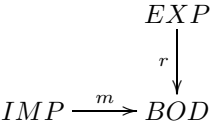


Fig. 5. Actualization.

The semantics of a parameterized Petri net (PAR, TAR) maps each transition system over the PAR to the corresponding transition net over TAR. The semantics is given by the free functor $F : \mathbf{TSA}(\mathbf{PAR}) \to \mathbf{TSA}(\mathbf{TAR})$. In Fig. 5 the net is refined by the actual parameter, where the transition s is substituted by the subnet containing the transitions $s1, s2$.

Constructor Semantics for Petri Net Modules. Most attempts to Petri net modules (among others [4, 5, 11]) do not provide Petri nets as interfaces. For a not so recent survey see [1]. There are either places or transitions, but no full Petri nets in the interface. When modeling software components these notions of Petri net modules are not powerful enough, since they do not allow specifying behavior in the interfaces. Our approach to Petri net modules [20] has been achieved by a transfer from the concept of algebraic module specifications presented in [7]. The main motivation for our approach to Petri net modules is the modeling of component-based systems. The component concept as suggested in [16, 26] for *Continuous Software Engineering (CSE)* is the underlying concept for our approach.

A Petri net module $MOD = (IMP, EXP, BOD)$ consists of three Petri nets, namely the import net IMP, the export net EXP and the body net BOD. Two Petri net morphisms $m : IMP \to BOD$ and $r : EXP \to BOD$ connect the interfaces to the body.

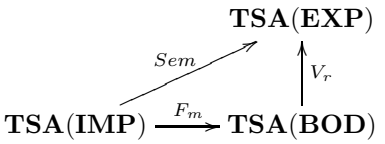


The import interface specifies resources which are used in the construction of the body, while the export interface specifies the functionality available from the Petri net module to the outside world. The body implements the functionality

specified in the export interface using the imported functionality. The import morphism m is a *plain morphism* and describes how and where the resources in the import interface are used in the body. The export morphism r is a *substitution morphism* and describes how the functionality provided by the export interface is realized in the body. The class of substitution morphism is as generalization of plain morphisms, where a transition is replaced by a subnet. Nevertheless, the forgetful functor constructions can be given for substitution morphisms as well (explicitly in [21]).

In [22] a transformation-based semantics for Petri net modules has been introduced based on the transformation-based approach to generic components [8]. There the semantics is defined based on all transformations the import may undergo. The advantage of our approach is that it is constructive: The semantics of a module is based on the loose semantics presented in this paper. It gives for each possible transition system over the import net the according transition system over the export net.

So we obtain a functor mapping the category of transition systems over the import net **TSA**(**IMP**) to the category of transition systems over the export net **TSA**(**EXP**).



This semantic functor naturally depends on the morphisms that relate the interfaces to the body of the module. We define the functor $Sem : \mathbf{TSA}(\mathbf{IMP}) \to \mathbf{TSA}(\mathbf{EXP})$ by $Sem = V_r \circ F_m$. V_r and F_m are constructed using the morphisms r and m. Now, we have the constructive semantics of a Petri net modules analogously to [7]. Each model of the import net is mapped to the corresponding model of the export. This module semantics takes some transition system over the import net and then constructs freely along the plain morphism $m : IMP \to BOD$, yielding a transition system over the body net BOD. Then it forgets along ' the forgetful functor V_r the internal details of the body and only represents the part specified by the export net EXP. Based on this notions we then obtain directly: internal and model correctness, compositional semantics with respect to module operations as union, composition (as given for Petri net modules [20]) or general module operations, based on schemes.

References

1. L. Bernadinello and F. De Cindio. A survey of basic net models and modular net classes. *Advances in Petri Nets 1992*, LNCS 609,pp: 304–351, Springer, 1992.
2. E. Best and J. Desel. Partial order behaviour and structure of Petri nets. *Formal Aspects of Computing*, pages 123–138, 1990.
3. A. Corradini, H. Ehrig, M. Löwe, U. Montanari, and F. Rossi. An event structure semantics for safe graph grammars. In *Proc. PROCOMET'94, IFIP TC2 Working Conf., San Miniato 1994*, pages 417–439. IFIP TCS, 1994.
4. S. Christensen and L. Petrucci. Modular analysis of Petri nets. *Computer Journal*, 43(3):224–242, 2000.

5. J. Desel, G. Juhás, and R. Lorenz. Petri Nets over Partial Algebras. In H. Ehrig, G. Juhás, J. Padberg, and G. Rozenberg, editors, *Advances in Petri Nets: Unifying Petri Nets*, volume 2128 of *LNCS*. Springer, 2001.
6. J. Desel and W. Reisig. Place/transition Petri nets. In W. Reisig and G. Rozenberg, editors, *Lectures on Petri Nets: Basic Models*, pages 122–173. Springer Verlag, LNCS 1491, 1998.
7. H. Ehrig and B. Mahr. *Fundamentals of Algebraic Specification 2: Module Specifications and Constraints*, volume 21 of *EATCS Monographs on Theoretical Computer Science*. Springer Verlag, Berlin, 1990.
8. H. Ehrig, F. Orejas, B. Braatz, M. Klein, and M. Piirainen. A Generic Component Concept for System Modelling. In R. Kutsche, H. Weber (Eds.)*Proc. FASE 2002: Formal Aspects of Software Engineering*, LNCS 2306, pages 33–48. Springer, 2002.
9. P. Huber, K. Jensen, and R.M. Shapiro. Hierarchies in Coloured Petri Nets. In G. Rozenberg, editor, *Advances in Petri nets 1990*, LNCS 483, pages 313–341. Springer, 1991.
10. P. W. Hoogers, H. C. M. Kleijn, and P. S. Thiagarajan. A trace semantics for Petri nets. *Information and Computation*, 117(1):98–114, 1995.
11. G. Juhás and R. Lorenz. Modelling with Petri modules. In B. Caillaud, X. Xie, and L. Darondeau, Ph.and Lavagno, editors, *Synthesis and Control of Discrete Event Systems*, pages 125–138. Kluwer Academic Publishers, 2002.
12. E. Kindler. A compositional partial order semantics for petri net components. In Azéma, P. and Balbo, G., editors, *18th Int. Conf. on Application and Theory of Petri Nets*, LNCS 1248, pages 235–252. Springer-Verlag, 1997.
13. A. Mazurkiewicz. Basic notions of trace theory. In de Bakker, J.W. et al., editors, *Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency.*, LNCS 354, pages 285–363. Springer, 1989.
14. J. Meseguer and U. Montanari. Petri Nets are Monoids. *Information and Computation*, 88(2):105–155, 1990.
15. R. Morin and B. Rozoy. On the semantics of place/transition nets. In *CONCUR 99*, LNCS 1664, pages 447–462. Springer, 1999.
16. H. Müller and H. Weber, editors. *Continuous Engineering of Industrial Scale Software Systems*. IBFI, Schloß Dagstuhl, Dagstuhl Seminar Report #98092, 1998.
17. M. Nielsen, G. Plotkin, and G. Winskel. Petri Nets, Event Structures and Domains, Part 1. *Theoretical Computer Science*, 13:85–108, 1981.
18. M. Nielsen and V. Sassone. Petri nets and other models of concurrency. In W. Reisig and G. Rozenberg, editors, *Lectures on Petri Nets I: Basic Models*, LNCS 1491, pages 587–642. Springer, 1998.
19. M. Nielsen, V. Sassone, and G. Winskel. Relationships Beween Models of Concurrency . In G. Rozenberg, J.W. de Bakker, W.-P. de Roever, editors, *A Decade of Concurrency*, pages 425 – 476. LNCS 803, 1993.
20. J. Padberg. Petri net modules. *Journal on Integrated Design and Process Technology*, 6(4):105–120, 2002.
21. J. Padberg. Transition systems with alternatives: an approach to a loose semantics of place/transition nets. Technical Report 2003-18, Technical University Berlin, 2003.
22. J. Padberg and H. Ehrig. Petri net modules in the transformation-based component framework. *Journal of Logic and Algebraic Programming*, 2003. submitted.
23. W. Reisig. *Petri Nets*, volume 4 of *EATCS Monographs on Theoretical Computer Science*. Springer Verlag, 1985.

24. H. Reichel. Specification semantics. In E. Astesiano, H.-J. Kreowski, and B. Krieg–Brückner, editors, *Algebraic Foundations of System Specification*, IFIP State–of–the–Art Reports, chapter 5, pages 131–158. Springer Verlag, 1999.
25. B. Rozoy. On distributed languages and models for concurrency. In G. Rozenberg, editor, *Advances in Petri Nets*, LNCS 609, pages 267–291. Springer, 1992.
26. H. Weber. Continuous Engineering of Communication and Software Infrastructures. In J.P. Finance (ed);*Fundamental Approaches to Software Engineering (FASE'99)*, LNCS 1577, 1999, pages 22–29. Springer Verlag, Berlin, Heidelberg, New York, 1999.