Compositional Modeling of Reactive Systems Using Open Nets*

P. Baldan[1], A. Corradini[1], H. Ehrig[2], R. Heckel[3]

[1] Dipartimento di Informatica, Università di Pisa, Italy
[2] Computer Science Department, Technical University of Berlin, Germany
[3] Dept. of Math. and Comp. Science, University of Paderborn, Germany
{baldan,andrea}@di.unipi.it ehrig@cs.tu-berlin.de reiko@upb.de

Abstract. In order to model the behaviour of open concurrent systems by means of Petri nets, we introduce *open Petri nets*, a generalization of the ordinary model where some places, designated as *open*, represent an interface of the system towards the environment. Besides generalizing the token game to reflect this extension, we define a truly concurrent semantics for open nets by extending the Goltz-Reisig process semantics of Petri nets. We introduce a composition operation over open nets, characterized as a pushout in the corresponding category, suitable to model both interaction through open places and synchronization of transitions. The process semantics is shown to be compositional with respect to such composition operation. Technically, our result is similar to the amalgamation theorem for data-types in the framework of algebraic specifications. A possible application field of the proposed constructions and results is the modeling of interorganizational workflows, recently studied in the literature. This is illustrated by a running example.

1 Introduction

Among the various models of concurrent and distributed systems, Petri nets [16] are certainly not the most expressive or the best-behaved. However, due to their intuitive graphical representation, Petri nets are widely used both in theoretical and applied research to specify and visualize the behaviour of systems. Especially when explaining the concurrent behaviour of a net to non-experts, one important feature of Petri nets is the possibility to describe their execution within the same visual notation, i.e., in terms of processes [5].

However, when modeling *reactive systems*, i.e., concurrent systems with interacting subsystems, Petri nets force us to take a global perspective. In fact, ordinary Petri nets are not adequate to model *open* systems which can interact with their environment or, in a different view, which are only partially specified. This contradicts the common practice, e.g., in software engineering, where a large system is usually built out of smaller components.

* Research partially supported by the EC TMR Network GETGRATS, by the ESPRIT Working Group APPLIGRAPH, by the MURST project TOSCA and by the DFG researcher group Petri Net Technology.

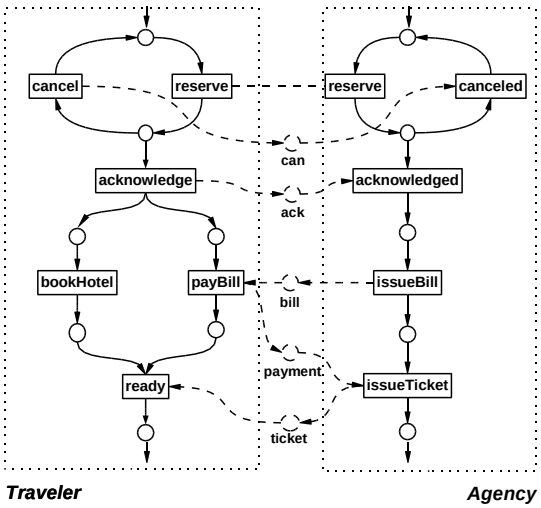


Fig. 1. Sample net modeling an interorganizational workflow.

Let us explain this problem in more detail by means of a typical application of Petri nets, the specification of workflows. A *workflow* describes a business process in terms of tasks and shared resources, as needed, for example, when the integration of different organizations is an issue. A *workflow net* [17] is a Petri net satisfying some structural constraints, like the existence of one initial and one final place, and a corresponding *soundness condition*: from each marking reachable from the initial one (one token on the initial place) we can reach the final marking (one token on the final place). An *interorganizational workflow* [18] is modeled as a set of such workflow nets connected through additional places for asynchronous communication and synchronization requirements on transitions.

For instance, Fig. 1 shows an interorganizational workflow consisting of two local workflow nets Traveler and Agency related through communication places can, ack, bill, payment and ticket and a synchronization requirement between the two reserve transitions, modeled by a dashed line. The example describes the booking of a flight by a traveler in cooperation with a travel agency. After some initial negotiations (which is not modeled), both sides synchronize in the reservation of a flight. Then, the traveler may either acknowledge or cancel and re-enter the initial state. In both cases an asynchronous notification (e.g., a fax), modeled by the places ack and can, respectively, is sent to the travel agency. Next the local workflow of the traveler forks into two concurrent threads, the booking of a hotel and the payment of the bill. The trip can start when both tasks are completed and the ticket has been provided by the travel agency.

The overall net in Fig. 1 describes the system from a global perspective. Hence, the classical notion of behaviour (described, e.g., in terms of processes) is completely adequate. However, for a local subnet in isolation (like Traveler) which will only exhibit a meaningful behaviour when interacting with other subnets, this semantics is not appropriate because it does not take into account the possible interactions.

To overcome these limitations of ordinary Petri nets, we extend the basic model introducing *open nets*. An open net is a P/T Petri net with a distinguished set of places which are intended to represent the interface of the net towards the external world. Some similarities exist with other approaches to net composition, like the *Petri box calculus* [2, 9, 8], the *Petri nets with interface* [12, 15] and the *Petri net components* [7], which will be discussed in the conclusions. As a consequence of the (hidden, implicit) interaction between the net and the environment, some tokens can "freely" appear in or disappear from the open places. Besides generalizing the token game to reflect this changes, we provide a truly concurrent semantics by extending the ordinary *process semantics* [5] to open nets.

The embedding of an open net in a context is formally described by a morphism in a suitable category of open nets. Intuitively, in the target net new transitions can be attached to open places and, moreover, the interface towards the environment can be reduced by "closing" open places. Therefore, open net morphisms do not preserve but reflect the behaviour, i.e., any computation of the target (larger) net can be projected back to a computation in the source (smaller) net.

A *composition operation* is introduced over open nets. Two open nets Z_1 and Z_2 can be composed by specifying a common subnet Z_0 which embeds both in Z_1 and Z_2, and gluing the two nets along the common part. This is permitted only if the prescribed composition is consistent with the interfaces, i.e., only if the places of Z_1 and Z_2 which are used when connecting the two nets are actually open. The composition operation is characterized as a pushout in the category of open nets, where the conditions for the existence of the pushout nicely fit with the mentioned condition over interfaces.

Based on these concepts, the representation of the system of Fig. 1 in terms of two interacting open nets is given by the top part of Fig. 2, which comprises the two component nets Traveler and Agency, and the net Common which embeds into both components by means of open net morphisms. Places with incoming/outgoing dangling arcs are open. Observe that the common subnet Common of the components Traveler and Agency closely corresponds to the dashed items of Fig. 1, which represent the "glue" between the two components. The net resulting from the composition of Traveler and Agency over the shared subnet Common is shown in the bottom part of Fig. 2.

Obviously, one would like to have a clear relationship between the behaviours of the component nets (nets Traveler and Agency in the example) and the behaviour of the composition (net Global in the example). We show that indeed, the behaviour of the latter can be constructed "compositionally" out of the behaviours of the former, in the sense that two deterministic processes which "agree" on the shared part, can be synchronized to produce a deterministic process over the composed net. Vice versa, *any* deterministic process of the global net can be decomposed into processes of the component nets, which, in turn, can be synchronized to give the original process again. Fig. 3 shows two processes of the nets Traveler and Agency, the corresponding common projections over net Common and the process of Global arising from their synchronization.

The synchronization of processes resembles the *amalgamation* of data-types in the framework of algebraic specifications, and therefore we will speak of *amalgamation of processes*. In analogy with the amalgamation theorem for algebraic specifications [4], the main result of this paper shows that the amalgamation and decomposition construc-

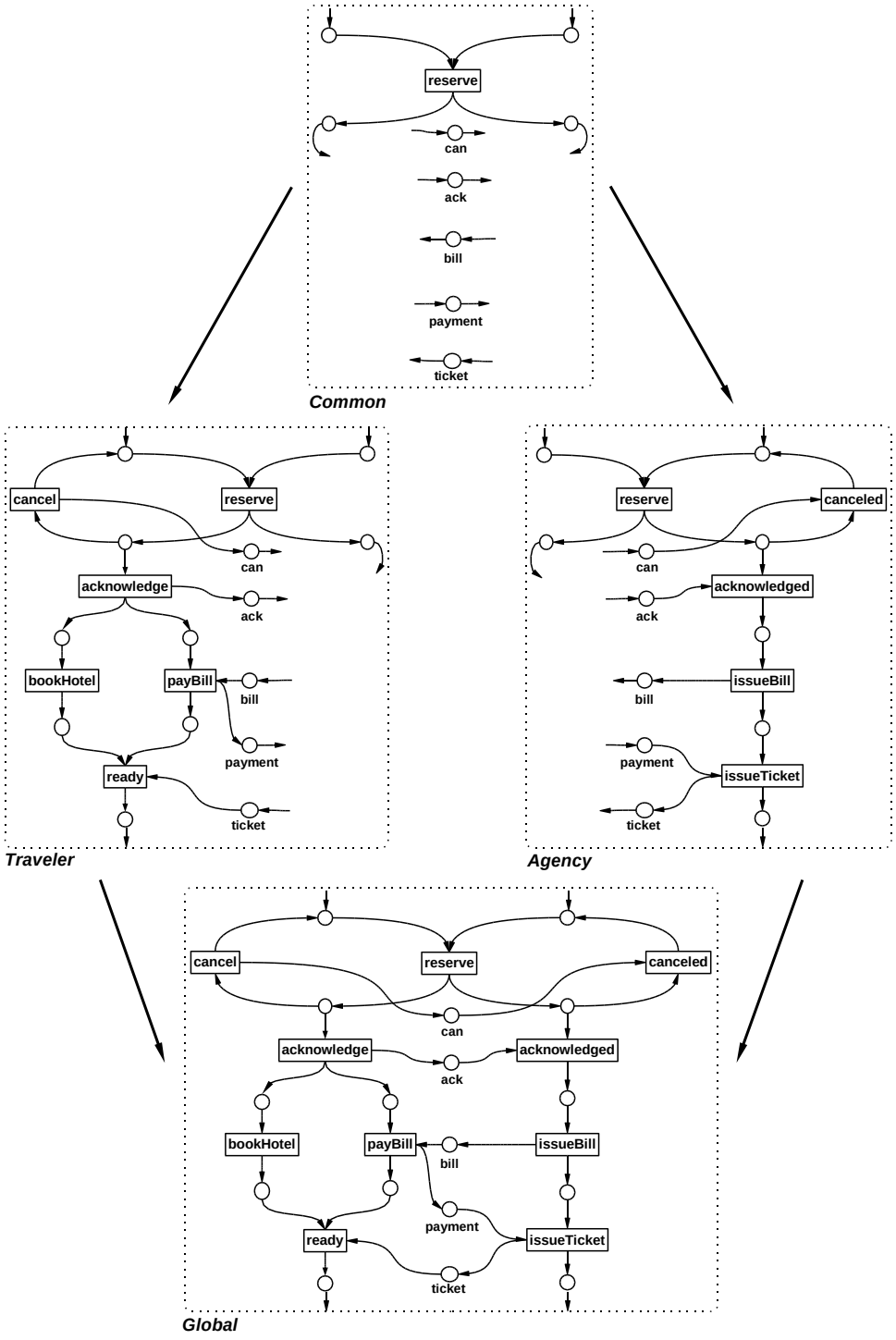


Fig. 2. Interorganizational workflow as composition of open nets Traveler and Agency.

tions mentioned above are inverse to each other, establishing a bijection between pairs of processes of two nets which agree on the common subnet and processes of the net resulting from their composition.

The rest of the paper is organized as follows. Section 2 introduces the open Petri net model and the corresponding category. Section 3 extends the notion of process from ordinary to open nets and defines the operation of behaviour projection. Section 4 introduces the composition operation for open nets. Section 5 presents the compositionality result for the process semantics of open nets. Finally, Section 6 discusses some related work in the literature and outlines possible directions of future investigation. The proofs of the results presented in this paper can be found in [1].

2 Open nets

An *open net* is an ordinary P/T Petri net with a distinguished set of places which are intended to represent the interface of the net towards the external world (environment). As a consequence of the (hidden, implicit) interaction between the net and the environment, some tokens can freely appear in and disappear from the open places. Concretely, an open place can be either an *input* or an *output* place (or both), meaning that the environment can put or remove tokens from that place.

Given a set X we denote by $X^{\oplus}$ the free commutative monoid generated by X and by $\mathbf{2}^X$ its powerset. Moreover for a function $h : X \to Y$ we denote by $h^{\oplus} : X^{\oplus} \to Y^{\oplus}$ its monoidal extension and by the same symbol $h : \mathbf{2}^X \to \mathbf{2}^Y$ the extension of h to sets.

Definition 1 (P/T Petri net). *A* P/T Petri net *is a tuple $N = (S, T, \sigma, \tau)$ where S is the set of places, T is the set of transitions ($S \cap T = \emptyset$) and $\sigma, \tau : T \to S^{\oplus}$ are the functions assigning to each transition its pre- and post-set.*

In the following we will denote by ${}^{\bullet}(\cdot)$ and $(\cdot)^{\bullet}$ the monoidal extensions of the functions σ and τ to functions from $T^{\oplus}$ to $S^{\oplus}$. Furthermore, given a place $s \in S$, the pre- and post-set of s are defined by ${}^{\bullet}s = \{t \in T \mid s \in t^{\bullet}\}$ and $s^{\bullet} = \{t \in T \mid s \in {}^{\bullet}t\}$.

Definition 2 (Petri net category). *Let N_0 and N_1 be Petri nets. A* Petri net morphism *$f : N_0 \to N_1$ is a pair of total functions $f = \langle f_T, f_S \rangle$ with $f_T : T_0 \to T_1$ and $f_S : S_0 \to S_1$, such that for all $t_0 \in T_0$, ${}^{\bullet} f_T(t_0) = f_S{}^{\oplus}({}^{\bullet}t_0)$ and $f_T(t_0)^{\bullet} = f_S{}^{\oplus}(t_0{}^{\bullet})$. The category of P/T Petri nets and Petri net morphisms is denoted by* **Net**.

Category **Net** is a subcategory of the category **Petri** of [10]. The latter has the same objects, but more general morphisms which can map a place into a multiset of places.

Definition 3 (open net). *An* open net *is a pair $Z = (N_Z, O_Z)$, where $N_Z = (S_Z, T_Z, \sigma_Z, \tau_Z)$ is an ordinary P/T Petri net and $O_Z = (O_Z^+, O_Z^-) \in \mathbf{2}^{S_Z} \times \mathbf{2}^{S_Z}$ are the input and output open places of the net.*

The notion of enabledness for transitions is the usual one, but, besides the changes produced by the firing of the transitions of the net, one considers also the interaction with the environment, modelled by a kind of invisible actions producing/consuming tokens in the input/output places of the net. The actions of the environment which produce and consume tokens in an open place s are denoted by $+_s$ and $-_s$, respectively.

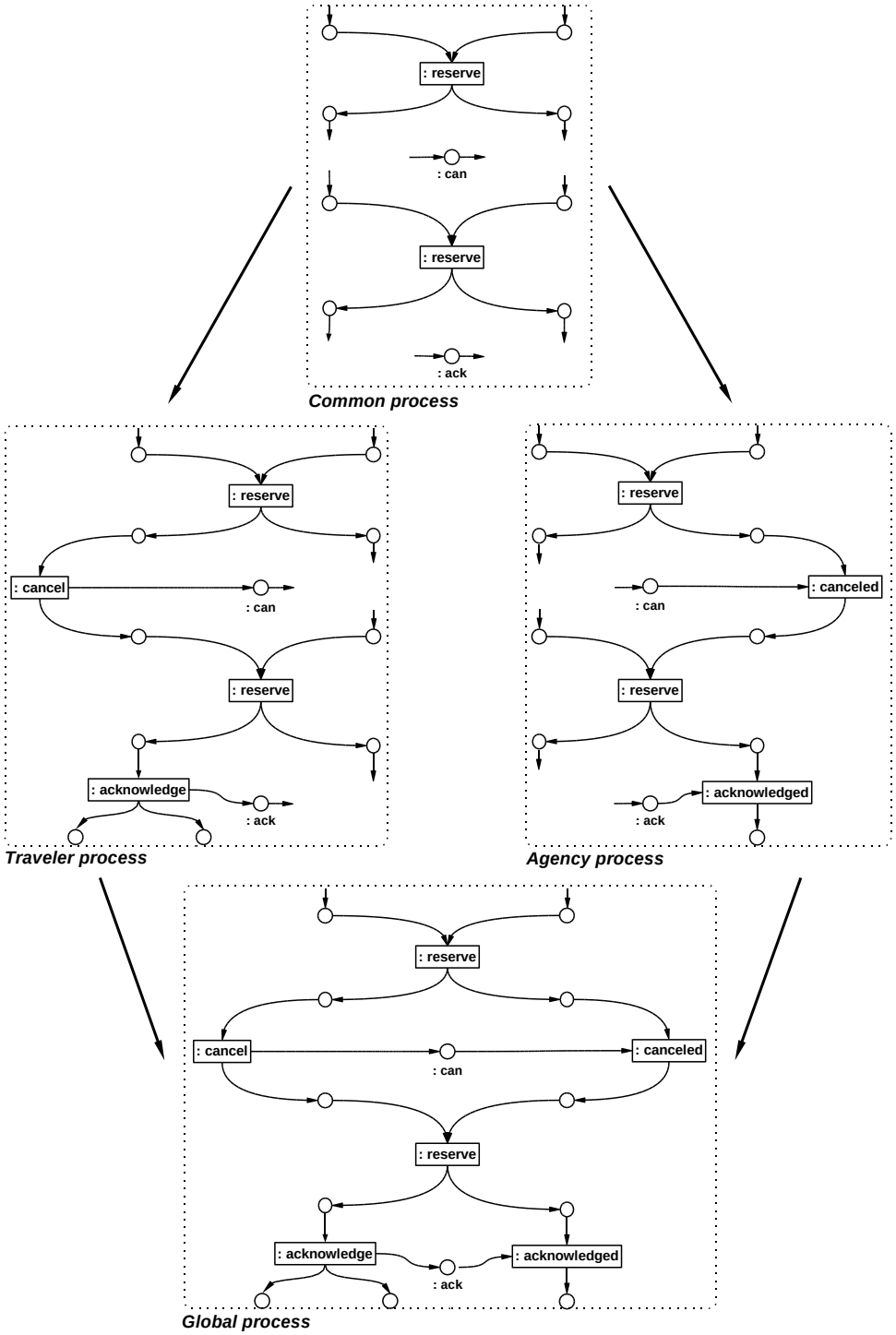


Fig. 3. Amalgamation of processes for the nets Traveler an Agency.

Definition 4 (firing). *Let Z be an open net. A* sequential move *can be (i) the firing of a transition $m \oplus {}^\bullet t \ [t\rangle \ m \oplus t^\bullet$, with $m \in {S_Z}^\oplus$, $t \in T_Z$; (ii) the creation of a token by the environment $m \ [+_s\rangle \ m \oplus s$, with $s \in O_Z^+$, $m \in {S_Z}^\oplus$; (iii) the deletion of a token by the environment $m \oplus s \ [-_s\rangle \ m$, with $m \in {S_Z}^\oplus$, $s \in O_Z^-$. A* parallel move *is of the form*

$$m \oplus {}^\bullet A \oplus m^- \ [A\rangle \ m \oplus A^\bullet \oplus m^+,$$

with $m \in {S_Z}^\oplus$, $A \in {T_Z}^\oplus$, $m^+ \in (O_Z^+)^\oplus$, $m^- \in (O_Z^-)^\oplus$.

Example. The open nets for the local workflows Traveler and Agency of Fig. 1 are shown in the middle of Fig. 2. Ingoing and outgoing arcs without source or target designate the input and output places, respectively. The synchronization transition reserve is common to both nets and the communication places, like can, become open places.

Definition 5 (open net category). *An* open net morphism $f : Z_1 \to Z_2$ *is a Petri net morphism $f : N_{Z_1} \to N_{Z_2}$ such that, if we define $\mathsf{in}(f) = \{s \in S_1 : {}^\bullet f_S(s) - f_T({}^\bullet s) \neq \emptyset\}$ and $\mathsf{out}(f) = \{s \in S_1 : f_S(s)^\bullet - f_T(s^\bullet) \neq \emptyset\}$ then*

$$\textit{(i)}\ f_S^{-1}(O_2^+) \cup \mathsf{in}(f) \subseteq O_1^+ \qquad \textit{and} \qquad \textit{(ii)}\ f_S^{-1}(O_2^-) \cup \mathsf{out}(f) \subseteq O_1^-.$$

The morphism f is called an open net embedding *if both f_T and f_S are injective. We will denote by* **ONet** *the category of open nets and open net morphisms.*

Hereafter, to lighten the notation, we will omit the subscripts "S" and "T" in the place and transition components of morphisms, writing $f(s)$ for $f_S(s)$ and $f(t)$ for $f_T(t)$.

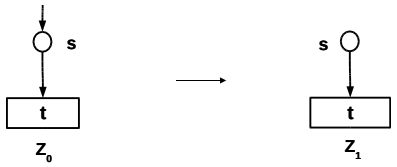
A morphism $f : Z_1 \to Z_2$ can be seen as an "insertion" of net Z_1 into a larger net Z_2, extending Z_1. In other words, Z_2 can be thought of as an instantiation of Z_1, where part of the unknown environment gets specified. Conditions (i) and (ii) first require that open places are reflected and hence that places which are "internal" in Z_1 cannot be promoted to open places in Z_2. Furthermore, the context in which Z_1 is inserted can interact with Z_1 only through the open places. To understand how this is formalized, observe that for each place s in $\mathsf{in}(f)$, its image $f(s)$ is in the post-set of a transition outside the image of ${}^\bullet s$. Hence we can think that in Z_2 new transitions are attached to s and can produce tokens in such place. This is the reason why condition (i) also asks any place in $\mathsf{in}(f)$ to be an input open place of Z_1. Condition (ii) is analogous for output places.

The above intuition better fits with open net embeddings, and indeed most of the constructions in the paper will be defined for this subclass of open net morphisms. However, for technical reasons (e.g., to characterize the composition of open nets as a pushout) the more general notion of morphism is useful.

Example. As an example of open net morphism, consider, in Fig. 2, the embedding of net Traveler into net Global. Observe that the constraints characterizing open nets morphisms have an intuitive graphical interpretation:

- the connections of transitions to their pre-set and post-set have to be preserved. New connections cannot be added;
- in the larger net, a new arc may be attached to a place only if the corresponding place of the subnet has a dangling arc in the same direction. Dangling arcs may be removed, but cannot be added in the larger net. E.g., without the outgoing dangling arc from place can in net Traveler, i.e., if place can were not output open, the mapping in from Traveler into Global would have not been a legal open net morphism.

We said that open net morphisms are designed to capture the idea of "insertion" of a net into a larger one. Hence it is natural to expect that they "reflect" the behaviour in the sense that given $f : Z_0 \to Z_1$, the behaviour of Z_1 can be projected along the morphism to the behaviour of Z_0 (this fact will be formalized later, in Construction 9). Instead, differently from most of the morphisms considered over Petri nets, open net morphisms cannot be thought of as simulations since they *do not* preserve the behaviour. For instance, consider the open nets Z_0 and Z_1 below and the obvious open net morphism between them.



Then the firing sequence $0\ [+_s\rangle\ s\ [t\rangle\ 0$ in Z_0 is not mapped to a firing sequence in Z_1.

3 Processes of open nets

Similarly to what happens for ordinary nets, a process of an open net, representing a concurrent computation of the net, is an open net itself, satisfying suitable acyclicity and conflict freeness requirements, together with a mapping to the original net.

The open net underlying a process is an open occurrence net, namely an open net K such that N_K is an ordinary occurrence net and satisfying some additional conditions over open places. The open places in K are intended to represent tokens which are produced/consumed by the environment in the considered computation. Consequently, every input open place is required to have an empty pre-set, i.e., to be minimal with respect to the causal order. In fact, an input open place in the post-set of some transition would correspond to a kind of generalized backward conflict: a token on this place could be generated in two different ways and this would prevent one to interpret the place as a token occurrence. Similarly, to avoid generalized forward conflicts, output open places are required to be maximal.

Definition 6 (open (deterministic) occurrence net). *An* open (deterministic) occurrence net *is an open net K such that*

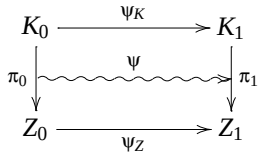
1. *N_K is an ordinary (deterministic) occurrence net, namely (i) for any $t \in T_K$, ${}^\bullet t$ and $t^\bullet$ are sets, rather than proper multisets; (ii) for any $t, t' \in T_K$, if $t \neq t'$ then ${}^\bullet t \cap {}^\bullet t' = \emptyset$ and $t^\bullet \cap t'^\bullet = \emptyset$; (iii) the causal relation $<_K$ defined as the least transitive relation such that $x <_K y$ if $y \in x^\bullet$, for $x, y \in S_K \cup T_K$, is a finitary strict partial order.*
2. *each input open place is minimal and each output open place is maximal w.r.t. $<_K$, i.e., $\forall s \in O_K^+.\ {}^\bullet s = \emptyset$ and $\forall s \in O_K^-.\ s^\bullet = \emptyset$.*

Definition 7 (open net process). *A* (deterministic) process *of an open net Z is a mapping $\pi : K \to Z$ where K is an open occurrence net and $\pi : N_K \to N_Z$ is a Petri net morphism, such that $\pi_S(O_K^+) \subseteq O_Z^+$ and $\pi_S(O_K^-) \subseteq O_Z^-$.*

Note that the process mapping π is *not*, in general, an open net morphism. In fact, the process mapping must be a simulation, i.e., it must preserve the behaviour. Moreover, the image of an open place in K must be an open place in Z, since tokens can be produced (consumed) by the environment only in input (output) open places of Z.

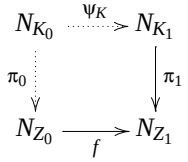
Example. A process for the open net Traveler can be found in the left part of Fig. 3. The morphism back to the original net Traveler is implicitly represented by the labeling (an item $:x$ is mapped to x). Observe that the requirements of minimality for input places and of maximality for output places of a process have a natural graphical interpretation: the absence of backward and forward conflicts extends to dangling arcs, i.e., in total, each place may have at most one ingoing and one outgoing arc.

Definition 8 (category of processes). *We denote by* **Proc** *the category where objects are processes and, given two processes $\pi_0 : K_0 \to Z_0$ and $\pi_1 : K_1 \to Z_1$, an arrow $\psi : \pi_0 \to \pi_1$ is a pair of open net morphisms $\psi = \langle \psi_Z : Z_0 \to Z_1, \psi_K : K_0 \to K_1 \rangle$ such that the following diagram (indeed the underlying diagram in* **Net***) commutes*



Let $f : Z_0 \to Z_1$ be an open net morphism. As mentioned before, it is natural to expect that each computation in Z_1 can be "projected" to Z_0, by considering only the part of the computation of the larger net which is visible in the smaller net. The above intuition is formalized, in the case of an open net embedding $f : Z_0 \to Z_1$, by showing how a process of Z_1 can be projected along f giving a process of Z_0.

Construction 9 (process projection). Let $f : Z_0 \to Z_1$ be an open net embedding and let $\pi_1 : K_1 \to Z_1$ be a process of Z_1. A *projection of π_1 along f* is a pair $\langle \pi_0, \psi \rangle$ where $\pi_0 : K_0 \to Z_0$ is a process of Z_0 and $\psi : \pi_0 \to \pi_1$ is an arrow in **Proc**, constructed as follows. Take the pullback of π_1 and f in **Net**, obtaining the net morphisms π_0 and ψ_K.



Then K_0 is obtained by taking N_{K_0} with the smallest sets of open places which make $\psi_K : N_{K_0} \to N_{K_1}$ an open net morphism, namely $O^+_{K_0} = {\psi_K}^{-1}(O^+_{K_1}) \cup \mathsf{in}(\psi_K)$ and $O^-_{K_0} = {\psi_K}^{-1}(O^-_{K_1}) \cup \mathsf{out}(\psi_K)$, and $\psi = \langle \psi_K, f \rangle$.

Example. The embedding of Traveler into Global in Fig. 2 induces a projection of open net processes in the opposite direction. For instance, the bottom part of Fig. 3 shows a process of Global. Its projection along the embedding of Traveler into Global is shown on the left part of the same figure. Notice how transition acknowledged, which consumes a token in place ack, is replaced in the projection by a dangling output arc: an internal action in the larger net becomes an interaction with the environment in the smaller one.

4 Composing open nets

We introduce a basic mechanism for composing open nets, characterized as a pushout construction in the category of open nets. Intuitively, two open nets Z_1 and Z_2 are composed by specifying a common subnet Z_0, and then by joining the two nets along Z_0. For instance, the open nets for the local workflows Traveler and Agency in the middle of Fig. 2 share the subnet Common, depicted in the top of the same figure, which represents the "glue" between the two components. The net Global resulting from the composition of Traveler and Agency over the shared subnet Common is shown in the bottom part of Fig. 2. This composition is only defined if the embeddings of the components into the resulting net satisfy the constraints of open net morphisms. For example, if we remove the ingoing dangling arc of the place ticket in the net Traveler, the embedding of Common into Traveler would still represent a legal open net morphism. However, in this case the embedding of Traveler into Global would become illegal because of the new arc from issueTicket (see condition (i) of Definition 5).

Formally, given two nets Z_1 and Z_2 and a span $f_1 : Z_0 \to Z_1$ and $f_2 : Z_0 \to Z_2$, the composition operation constructs the corresponding pushout in **ONet**. Category **ONet** does not have all pushouts, while category **Net** does. This corresponds to the intuition that the composition operation can be performed in **Net** and then lifted to **ONet**, but only when it respects the interfaces specified by the various components, e.g., a new transition can be attached to a place only if such place is open (see also [1]).

We start by recalling that for any span $N_1 \stackrel{f_1}{\leftarrow} N_0 \stackrel{f_2}{\rightarrow} N_2$ in **Net** the pushout always exists. It can be defined as $N_1 \stackrel{\alpha_1}{\rightarrow} N_3 \stackrel{\alpha_2}{\leftarrow} N_2$, where the sets of places and transitions of N_3 are computed as the pushout in **Set** of the corresponding components, i.e., $S_3 = S_1 +_{S_0} S_2$ and $T_3 = T_1 +_{T_0} T_2$. The source and target functions are defined by: for all $t \in T_3$, if $t = \alpha_i(t_i)$ with $t_i \in T_i$ and $i \in \{1,2\}$ then ${}^\bullet t = \alpha_i{}^{\oplus}({}^\bullet t_i)$ and $t^\bullet = \alpha_i{}^{\oplus}(t_i^\bullet)$. Next we formalize the condition which ensures the composability of a span in **ONet**.

Definition 10 (composable span). *Let $Z_1 \stackrel{f_1}{\leftarrow} Z_0 \stackrel{f_2}{\rightarrow} Z_2$ be a span of open net morphisms. We say that f_1 and f_2 are* composable *if*

1. $f_2(\mathsf{in}(f_1)) \subseteq O^+_{Z_2}$ *and* $f_2(\mathsf{out}(f_1)) \subseteq O^-_{Z_2}$*;*
2. $f_1(\mathsf{in}(f_2)) \subseteq O^+_{Z_1}$ *and* $f_1(\mathsf{out}(f_2)) \subseteq O^-_{Z_1}$.

In words, f_1 and f_2 are composable if the places which are used as interfaces by f_1, namely the places $\mathsf{in}(f_1)$ and $\mathsf{out}(f_1)$, are mapped by f_2 to input and output open places in Z_2, and also the symmetric condition holds. If, and only if, this condition is satisfied the pushout of f_1 and f_2 can be computed in **Net** and then lifted to **ONet**.

Proposition 11 (pushouts in ONet). *Let $Z_1 \stackrel{f_1}{\leftarrow} Z_0 \stackrel{f_2}{\rightarrow} Z_2$ be a span in* **ONet** *(see the diagram in Fig. 4). Compute the pushout of the corresponding diagram in the category* **Net** *obtaining the net N_{Z_3} and the morphisms α_1 and α_2, and then take as open places, for $x \in \{+,-\}$, $O^x_{Z_3} = \{s_3 \in S_3 \mid \alpha_1^{-1}(s_3) \subseteq O^x_{Z_1} \wedge \alpha_2^{-1}(s_3) \subseteq O^x_{Z_2}\}$. Then $(\alpha_1, Z_3, \alpha_2)$ is the pushout in* **ONet** *of f_1 and f_2 iff f_1 and f_2 are composable.*

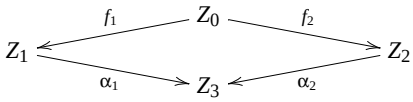


Fig. 4. Pushout in **ONet**.

5 Amalgamating processes of open nets

Let $f_1 : Z_0 \to Z_1$ and $f_2 : Z_0 \to Z_2$ be a composable span of open net embeddings and consider the corresponding composition, i.e., the pushout in **ONet**, as depicted in Fig. 4. We would like to establish a clear relationship among the behaviours of the involved nets. Roughly speaking, we would like that the behaviour of Z_3 could be constructed "compositionally" out of the behaviours of Z_1 and Z_2.

In this section we show how this can be done for processes. Given two processes π_1 of Z_1 and π_2 of Z_2 which "agree" on Z_0, we construct a process π_3 of Z_3 by "amalgamating" π_1 and π_2. Vice versa, each process π_3 of Z_3 can be projected over two processes π_1 and π_2 of Z_1 and Z_2, respectively, which can be amalgamated to produce π_3 again. Hence, all and only the processes of Z_3 can be obtained by amalgamating the processes of the components Z_1 and Z_2. This is formalized by showing that, working up to isomorphism, the amalgamation and decomposition operations are inverse to each other. This leads to a bijective correspondence between the processes of Z_3 and pair of processes of the components Z_1 and Z_2 which agree on the common subnet Z_0.

As a first step towards the amalgamation of processes we identify a suitable condition which ensures that the pushout of occurrence open nets exists and produces a net in the same class. This condition will be used later to formalize the intuitive idea of processes of different nets which "agree" on a common part.

For a given span $K_1 \stackrel{f_1}{\leftarrow} K_0 \stackrel{f_2}{\rightarrow} K_2$ we introduce the notion of causality relation induced by K_1 and K_2 over K_0. When the two nets are composed the corresponding causality relations get "fused". Hence, to avoid the creation of cyclic causal dependencies in the resulting net, the induced causality will be required to be a partial order.

Definition 12 (induced causality and consistent span). *Let $K_1 \stackrel{f_1}{\leftarrow} K_0 \stackrel{f_2}{\rightarrow} K_2$ be a span in* **ONet**, *where K_i ($i \in \{0,1,2\}$) are occurrence open nets. The relation of causality $<_{1,2}$ induced over K_0 by K_1 and K_2, through f_1 and f_2 is the least transitive relation such that for any x_0, y_0 in K_0, if $f_1(x_0) <_1 f_1(y_0)$ or $f_2(x_0) <_2 f_2(y_0)$ then $x_0 <_{1,2} y_0$.*

We say that the span is consistent, *written $f_1 \uparrow f_2$, if f_1 and f_2 are composable and the induced causality $<_{1,2}$ is a finitary strict partial order.*

The next proposition shows that the composition operation in **ONet**, when applied to a consistent span of occurrence nets, produces an occurrence net.

Proposition 13. *Let $K_1 \stackrel{f_1}{\leftarrow} K_0 \stackrel{f_2}{\rightarrow} K_2$ be a composable span in* **ONet**, *where K_i ($i \in \{0,1,2\}$) are occurrence open nets and let $K_1 \stackrel{\alpha_1}{\rightarrow} K_3 \stackrel{\alpha_2}{\leftarrow} K_2$ be the pushout in* **ONet**. *Then $f_1 \uparrow f_2$ if and only if the pushout object K_3 is a occurrence open net.*

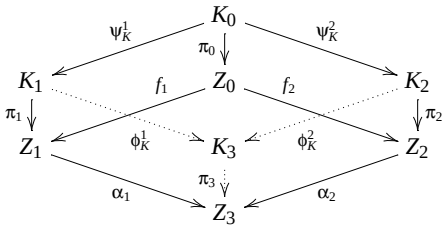


Fig. 5. Amalgamation of open net processes.

Two processes π_1 of Z_1 and π_2 of Z_2 can be amalgamated only when they agree on the common subnet Z_0, an idea which is formalized by resorting to the notion of consistent span of occurrence open nets. In the rest of this section we will refer to a fixed pushout diagram in **ONet**, as represented in Fig. 4, where f_1 and f_2 are a composable span of *open net embeddings*.

Definition 14 (agreement of processes). *The processes $\pi_1 : K_1 \to Z_1$ and $\pi_2 : K_2 \to Z_2$* agree *on Z_0 if there exist projections $\langle \pi_0, \psi^i \rangle$ along f_i of π_i for $i \in \{1,2\}$ such that $\psi^1_K \uparrow \psi^2_K$ (i.e., the span $K_1 \xleftarrow{\psi^1_K} K_0 \xrightarrow{\psi^2_K} K_2$ is consistent). In this case $\langle \pi_0, \psi^1 \rangle$ and $\langle \pi_0, \psi^2 \rangle$ are called* agreement projections *for π_1 and π_2.*

Definition 15 (amalgamation of processes). *Let $\pi_i : K_i \to Z_i$ ($i \in \{0,1,2,3\}$) be processes and let $\langle \pi_0, \psi^1 \rangle$ and $\langle \pi_0, \psi^2 \rangle$ be agreement projections of π_1 and π_2 along f_1 and f_2 (see Fig. 5). We say that π_3 is an* amalgamation *of π_1 and π_2, written $\pi_3 = \pi_1 +_{\psi^1,\psi^2} \pi_2$, if there exist projections $\langle \pi_1, \phi^1 \rangle$ and $\langle \pi_2, \phi^2 \rangle$ of π_3 over Z_1 and Z_2, respectively, such that the upper square is a pushout in* **ONet**.

We next give a more constructive characterization of process amalgamation, which also proves that the result is unique up to isomorphism.

Proposition 16 (amalgamation construction). *Let $\pi_1 : K_1 \to Z_1$ and $\pi_2 : K_2 \to Z_2$ be processes that agree on Z_0, and let $\langle \pi_0, \psi^1 \rangle$ and $\langle \pi_0, \psi^2 \rangle$ be corresponding agreement projections. Then the* amalgamation *$\pi_1 +_{\psi^1,\psi^2} \pi_2$ is a process $\pi_3 : K_3 \to Z_3$, where the net K_3 is obtained as the pushout in* **ONet** *of $\psi^1_K : K_0 \to K_1$ and $\psi^2_K : K_0 \to K_2$ and the process mapping $\pi_3 : K_3 \to Z_3$ is determined by the universal property of the underlying pushout diagram in* **Net** *(see Fig. 5). Hence $\pi_1 +_{\psi^1,\psi^2} \pi_2$ is unique up to isomorphism.*

The amalgamation construction can be given a more elegant (but less constructive) characterization. In fact, process π_3 (and the process morphisms ϕ^1 and ϕ^2) can be obtained by taking the pushout in **Proc** of the arrows $\psi^1 : \pi_0 \to \pi_1$ and $\psi^2 : \pi_0 \to \pi_2$.

The next result shows how each process of a composed net can be constructed as the amalgamation of processes of the components.

Proposition 17 (decomposition of processes). *Let $\pi_3 : K_3 \to Z_3$ be a process of Z_3 and, for $i \in \{1,2\}$, let $\langle \pi_i, \phi^i \rangle$ be projections of π_3 along α_i. Then process π_3 can be recovered as a suitable amalgamation of π_1 and π_2.*

The amalgamation and decomposition results for open net processes are summarized in a theorem which establishes a bijective correspondence between the processes of Z_1 and Z_2 which agree on Z_0 and the processes of Z_3. Let Z be an open net and let $\pi : K \to Z$ be a process. We denote by $[\pi]$ the set of processes of Z isomorphic to π and by $\mathbf{DProc}(Z)$ the set of (isomorphism classes of) processes of Z. Given a span $Z_1 \xleftarrow{f_1} Z_0 \xrightarrow{f_2} Z_2$ in **ONet**, the isomorphism classes of processes of Z_1 and Z_2 which agree on Z_0, denoted by $\mathbf{DProc}(Z_1 \xleftarrow{f_1} Z_0 \xrightarrow{f_2} Z_2)$, is the set

$$\{[\pi_1 \xleftarrow{\psi^1} \pi_0 \xrightarrow{\psi^2} \pi_2] \mid \psi^1, \psi^2 \text{ agreement projections for } \pi_1, \pi_2 \text{ along } f_1, f_2\},$$

where isomorphism of process spans is defined in the obvious way.

Theorem 18 (amalgamation theorem). *Let Z_0, Z_1, Z_2, Z_3 be as in Fig. 4 and assume that the square is a pushout of two composable open net embeddings f_1 and f_2. Then there are composition and decomposition functions establishing a bijective correspondence between* $\mathbf{DProc}(Z_3)$ *and* $\mathbf{DProc}(Z_1 \xleftarrow{f_1} Z_0 \xrightarrow{f_2} Z_2)$.

Example. The amalgamation theorem is exemplified in Fig. 3. Two processes for the component nets Traveler and Agency which agree on the shared subnet Common, i.e., such that their projections over Common coincide, can be amalgamated to produce a process for the composed net Global. Vice versa, each process of the net Global can be reconstructed as amalgamation of compatible processes of the component nets.

6 Conclusions and related work

The compositionality result for the process semantics (Theorem 18) appears to be related to the amalgamation theorem for data-types in the framework of algebraic specifications [4]. There, an amalgamation construction allows one to "combine" any two algebras A_1 and A_2 of algebraic specifications $SPEC_1$ and $SPEC_2$ having a common subspecification $SPEC_0$, if and only if the restrictions of A_1 and A_2 to $SPEC_0$ coincide. The amalgamation construction produces a unique algebra A_3 of specification $SPEC_3$, union of $SPEC_1$ and $SPEC_2$. The fact that the amalgamation of algebras is a pushout in the Grothendick's category of generalized algebras suggests the possibility of having a similar characterization for process amalgamation using fibred categories.

Open nets have been partly inspired by the notion of *open graph transformation system* [6], an extension of graph transformation for specifying reactive systems. In fact, P/T Petri nets can be seen as a special case of graph transformation systems [3] and this correspondence extends to open nets and open graph transformation systems. However, a compositionality result corresponding to Theorem 18 is still lacking in this more general setting.

In the field of Petri nets, several other approaches to net composition have been proposed in the literature. Most of them can be classified as algebraic approaches. A first family considers a category of Petri nets where morphisms arise by viewing a Petri net as the signature of a multisorted algebra, the sorts being the places. Then the semantics is expressed as a categorical adjunction, a fact which ensure its compositionality with respect to operations on nets defined in terms of universal constructions [19, 10].

A second, more recent class of approaches to Petri net composition aims at defining a "calculus of nets", where a set of process algebra-like operators allows to build complex nets starting from a suitable set of basic net components. For instance, in the Petri Box calculus [2, 9, 8] a special class of nets, called *plain boxes* (safe and clean nets), provides the basic components. Plain boxes are then combined by means of operations which can all be seen as an instance of refinement over suitable nets. More precisely, the authors identify a special family of nets, called *operator boxes*. Once a set of operator boxes is fixed, the composition is realized by refining such operator boxes with plain boxes, an operation which produces a net still identifiable with a plain box. The calculus is given a compositional semantics (both interleaving and concurrent). Although based on some common ideas, like the use of interface places, this approach is quite different from ours, since it mainly relies on refinement and it focuses on a special class of nets and on the possibility of defining a kind of process algebra over such nets, with plain boxes as constants and operator boxes as operators.

Another relevant approach in the second family, closer to ours, is presented in the papers [12, 15], which introduce a notion of Petri net with *interface*. The interface is partitioned into an input part, consisting of places, and an output part, consisting of transitions, and it is used to combine different nets, the most basic composition operation consisting of connecting the outputs of one net to the inputs of another net. Then the authors introduce a set of basic combinators which can be used to build terms corresponding to nets with an interface. The *pomset semantics* of nets with interfaces, defined by using a notion of universal context for a net, is shown to be compositional with respect to the net combinators [15]. Despite some technical differences and the different focus, which in these papers is more on the syntactical aspects of the Petri net algebra, Petri nets with interface appear to have several analogies with open nets, and their relationship surely deserves a deeper investigation.

Finally we recall the work in [7] which introduces *Petri net components*, a kind of Petri nets with distinguished input and output places. Components can be combined by means of an operation which connects the input places of a component to the output places of the other, and vice versa. A process semantics is introduced for components and it is proved to be compositional. Components can be viewed as special open nets and the composition operation on components can be defined in terms of the composition operation on open nets. A very interesting idea in [7], which we intend to explore also for open nets, is the definition of a temporal logic, interpreted over processes, which is used for reasoning in a modular way over distributed systems.

The notions of projection and of amalgamation of processes can be extended to general, possibly nondeterministic, processes. We are working on the generalization of the amalgamation theorem to nondeterministic processes, which could represent a first step towards an unfolding semantics for open nets, in the style of Winskel [11, 19], still compositional with respect to our composition operation.

It would be also interesting to extend the constructions and results in this paper to open *high level nets*, which have been already studied on a conceptual level in [14]. Part of the technical background is already available — for instance it has been shown in [13] how to construct pushouts of algebraic high level nets — but a suitable formalization of high level processes is still missing.

Acknowledgement. We are grateful to Ugo Montanari for his insightful suggestions and to the anonymous referees for their helpful comments.

References

1. P. Baldan, A. Corradini, H. Ehrig, and R. Heckel. Compositional modeling of reactive systems using open nets [extended version]. The paper can be downloaded at the address `http://www.di.unipi.it/~baldan/Papers/Soft-copy-ps/open-ext.ps.gz`, 2001.
2. E. Best, R. Devillers, and J. G. Hall. The Petri box calculus: a new causal algebra with multi-label communication. In G. Rozemberg, editor, *Advances in Petri Nets*, volume 609 of *LNCS*, pages 21–69. Springer Verlag, 1992.
3. A. Corradini. Concurrent graph and term graph rewriting. In U. Montanari and V. Sassone, editors, *Proceedings of CONCUR'96*, volume 1119 of *LNCS*, pages 438–464. Springer Verlag, 1996.
4. H. Ehrig and B. Mahr. *Fundamentals of Algebraic Specification 1*. Springer Verlag, Berlin, 1985.
5. U. Golz and W. Reisig. The non-sequential behaviour of Petri nets. *Information and Control*, 57:125–147, 1983.
6. R. Heckel. *Open Graph Transformation Systems: A New Approach to the Compositional Modelling of Concurrent and Reactive Systems.* PhD thesis, TU Berlin, 1998.
7. E. Kindler. A compositional partial order semantics for Petri net components. In P. Azema and G. Balbo, editors, *Application and Theory of Petri Nets*, volume 1248 of *LNCS*, pages 235–252. Springer Verlag, 1997.
8. M. Koutny and E. Best. Operational and denotational semantics for the box algebra. *Theoretical Computer Science*, 211(1–2):1–83, 1999.
9. M. Koutny, J. Esparza, and E. Best. Operational semantics for the Petri box calculus. In B. Jonsson and J. Parrow, editors, *Proceedings of CONCUR '94*, volume 836 of *LNCS*, pages 210–225. Springer Verlag, 1994.
10. J. Meseguer and U. Montanari. Petri nets are monoids. *Information and Computation*, 88:105–155, 1990.
11. M. Nielsen, G. Plotkin, and G. Winskel. Petri Nets, Event Structures and Domains, Part 1. *Theoretical Computer Science*, 13:85–108, 1981.
12. M. Nielsen, L. Priese, and V. Sassone. Characterizing Behavioural Congruences for Petri Nets. In *Proceedings of CONCUR'95*, volume 962 of *LNCS*, pages 175–189. Springer Verlag, 1995.
13. J. Padberg, H. Ehrig, and L. Ribeiro. Algebraic high-level net transformation systems. *Mathematical Structures in Computer Science*, 5(2):217–256, 1995.
14. J. Padberg, L. Jansen, R. Heckel, and H. Ehrig. Interoperability in train control systems: Specification of scenarios using open nets. In *Proc. IDPT*, pages 17–28. Society for Design and Process Science, 1998.
15. L. Priese and H. Wimmel. A uniform approach to true-concurrency and interleaving semantics for Petri nets. *Theoretical Computer Science*, 206(1–2):219–256, 1998.
16. W. Reisig. *Petri Nets: An Introduction.* EACTS Monographs on Theoretical Computer Science. Springer Verlag, 1985.
17. W. van der Aalst. The application of Petri nets to workflow management. *The Journal of Circuits, Systems and Computers*, 8(1):21–66, 1998.
18. W. van der Aalst. Interorganizational workflows: An approach based on message sequence charts and Petri nets. *System Analysis and Modeling*, 34(3):335–367, 1999.
19. G. Winskel. Event Structures. In *Petri Nets: Applications and Relationships to Other Models of Concurrency*, volume 255 of *LNCS*, pages 325–392. Springer Verlag, 1987.