

Modeling Service Communication with Open Petri Nets^{*}

Chrysafis Hartonas

Dept of Computer Science, TEI Larissa,
41110 Larissa, Greece
hartonas@cs.teilar.gr

Abstract. In this report we extend on the work presented in [2, 5, 9, 7, 8], where the authors proposed a petri net model for service message exchange.

We revisit the composition problem for arbitrary communicating services and refine the model so that it can deal with cyclic nets, therefore modeling recursion as well. Further, we provide a rigorous definition of composition, that removes the dependance of this operation on specific naming conventions on communication ports.

We also revisit the issue of the public view of a service, we introduce a natural notion of abstract behavior graph for a service and propose using it as the public view.

Keywords: Service Oriented Architecture, Service Communication, Service bind operation, Open Petri Nets.

1 Preliminaries

1.1 Background Work

The *Service-Oriented Architecture* (SOA) is an architectural style whose goal is to achieve loose coupling among interacting software components. The main point is that (web) services (the components) are thought of as, precisely, *interacting* in some ways, rather than as executing processes in isolation. Component interaction (service *binding*) necessitates a mechanism of communication (which may be simple data passing) between services. In any case, component interaction raises the issue of appropriately modeling this communicative behavior of services, i.e. proposing a suitable model for the *bind* operation on services.

Extending on a proposal of [1], *open workflow nets* (a variant of petri nets) have been explored in [2, 5, 9] and also in [7, 8] as a means to model service

^{*} Research conducted in the context of the AgentNet project, led by this author. AgentNet is part of a larger research project run by the Graduate Program in Logic, Algorithmics and Theory of Computation of the University of Athens. The project is co-funded by the European Social Fund and National Resources (EPEAEK II) PYTHAGORAS II

composition. The authors proposed to model service composition as a simple *union* operation on nets. However, successfully modeling composition as union, in the reports mentioned, depends on adopting a suitable *naming convention* for communicative actions and communication ports, so that the union operation may have the desired result; this is clearly illustrated in the vending machine example the authors present in both [7, 8], as well as in the examples presented in [2, 5].

We note that, the need for an explicit naming convention becomes of significance when composing services each one of which can perform both input and output on the same communication channel. For example, the vending machine will both input a number of coins (performing a $?€$ action) and output some change (performing a $!€$ action). Similarly for the service client, except in reverse order). To avoid confusion, the respective input and output ports have to be named differently, e.g. $€$ and $€'$ (see [8]). Furthermore, *composability* of services (or of services with clients) is explicitly made to depend on the naming, not of the (complementary) communicative actions, but on this ad hoc naming convention on communication ports, see e.g. [6]. This is fine when analyzing an example, as one is then free to name communication ports in precisely the way that serves the example! However, this ad hoc naming approach is somewhat lacking in formal rigor and largely unnecessary, as we show in this report.

1.2 Objectives of this Report

In this report, we focus on providing a formal account of *composability of services*, on formally modeling the operation of *composition of services* (or service-client composition) and, finally, on providing a formal and abstract characterization of the *public view of a service* [2, 5]. Our goal is both to extend the approach of [2, 5, 7, 8] by treating recursive processes as well (cyclic nets) and, also, to abstract away from naming conventions on communication ports. Further, we examine here the case where composing services on a communication channel, leaves this channel still open for external communication (external concurrent join), or internalizes the port for private communication (internal concurrent join), or, finally, a mixed situation in which communication actions are annotated as internal or external and thereby ports are internalized or not depending on the annotation of the communicative action they correspond to (mixed concurrent join).

We define a class of *open petri nets*, similar in spirit to the open workflow nets of [2, 5, 7, 8] and introduce an operation of *composition (concurrent join)* of nets. The composition operation depends on identifying *matching* input-output transitions. This identification, as we propose it here, abstracts away from any example-specific naming convention for communication ports, and it is defined in terms of a notion of *firing rank sequence* for each input or output transition. The approach we take here can handle both finite and infinite (recursive) processes. Our approach can be used to refine the notions of “public view” [2, 5], or of the “operating guideline” proposed in [7, 8].

1.3 Structure of this Report

In Section 2, we first briefly review the definition of place-transition nets. Subsequently, we extend to a notion of *Open Petri Net*, in Section 2.2. The definition is in spirit, though not in technical detail, similar to the definition of the open workflow nets of [2, 5, 7, 8]. For technical reasons, we review in Section 2.3 the notion of *net firing*, including a notion of *weak firing* (of *weakly enabled transitions*), designed to describe the behavior of a net under the assumption that all requested input becomes available.

Section 3 contains the main technical contribution of this report. Notions of external, internal or mixed open net composition are proposed and technically defined, which rest on the identification of *matching communicative actions*. These operations, we contend, can adequately, and abstractly, model the *bind* operation on services, independently of port naming conventions and for recursive, as well as for finite behavior. Furthermore, they provide a solid ground for formally modeling the *composability* and *compatibility* of services.

In Section 4 we discuss natural notions of behavioral equivalence of oPNs, prove some useful technical properties of the concurrent join operator and, furthermore, we propose a notion of *public view*. We conclude the report with Section 5, where we also point out some directions for further research.

2 Communicating Petri Nets

2.1 Place-Transition Nets

For reader's convenience, we first briefly recall the definition of place-transition nets.

Definition 1 (Place-Transition Petri Nets). Assume disjoint and nonempty sets *Place* and *Trans* of place and transition labels, respectively. A *Petri Net* with place labels in $P \subseteq \text{Place}$ and transition labels in $T \subseteq \text{Trans}$ is a triple

$$\mathcal{A} = (P, \longrightarrow, M_0)$$

where $M_0 \subseteq \mathcal{P}_{nf}(P)$ is called the *initial marking*, and $\longrightarrow \subseteq \mathcal{P}_{nf}(P) \times T \times \mathcal{P}_{nf}(P)$ is the transition relation (the subscript *nf* indicates non-empty finite subsets). A transition $t = (I, a, O)$ will be also denoted by $t = I \xrightarrow{a} O$. The set $I = \text{pre}(t)$ is called the preset of the transition, while $O = \text{post}(t)$ its postset, and $\text{act}(t) = a$ its action. We occasionally also write $Pl(\mathcal{A})$ for the set P of places of \mathcal{A} . \square

Note that, for simplicity, we restrict here to markings with single tokens on places, but this is an inessential restriction.

2.2 Open Petri Nets

To define open nets, assume pairwise disjoint sets *Place*, *Trans* and *Port*.

Definition 2 (Open Petri Nets). An open net is like an ordinary net except for

1. its set of places is contained in the union $Place \cup Port$
2. its action labels are either
 - τ (a silent, internal action), or
 - $c!$ or $c?$, with $c \in C$, a set of communicative action names
3. if $I \xrightarrow{c!} O$ is a transition in the net, then $I \cap Port = \emptyset$ and $O = O' \cup \{x\}$, with $\emptyset \neq O' \subseteq Place$ and $x \in Port$
4. if $I \xrightarrow{c?} O$ is a transition, then $I \cup O \subseteq Place$ and, for some port name $x \in Port$ and set of places $U \subseteq Place$, there is a transition $O \cup \{x\} \xrightarrow{\tau} U$
5. if $I \xrightarrow{\tau} O$ is a transition, then $O \subseteq Place$ and, unless there is a transition $J \xrightarrow{c?} K$ such that $K \cap I \neq \emptyset$, the preset I of τ contains no ports (i.e. $I \cap Port = \emptyset$) □

Intuitively, a transition $t = I \xrightarrow{c!} O \cup \{x\}$ is a transition that executes an asynchronous output action and deposits the output on a communication buffer, named x . The action is asynchronous in the sense that the net will proceed to fire any transitions that became enabled after firing t , without waiting for its output to be consumed. Similarly, a transition $r = J \xrightarrow{c?} K$ is an action that prompts (requests) input on some specific channel c . If such input is provided, through some buffer y associated to channel c , then it is consumed by an internal action $\{y\} \cup K \xrightarrow{\tau} L$.

Note that we have restricted only to visible, communicating actions and any internal action (process) of the service is uniformly designated by a τ action.

It should be also noted that we focus on the communication exchange per se, ignoring the type of content being communicated. Thus, in this setting, what is passed from service to service during communication is merely a “token”.

In the sequel, we write $\mathcal{A} \cong \mathcal{B}$ for two nets, provided there is an injective renaming σ of place (including port) names, such that $\mathcal{A}\sigma = \mathcal{B}$ (where $\mathcal{A}\sigma$ is the net obtained by applying the renaming σ to the places of \mathcal{A}).

In the next Section we recall the standard notion of “enabled transitions” and net firings and, furthermore, we extend to appropriate notions of *weakly enabled transitions* and *weak net firing*. The intention here is to capture the (potential) behavior of an open net, assuming that any requested input becomes available when needed.

2.3 Net Firings

Definition 3. A transition t is enabled at a marking M if $\mathbf{pre}(t) \subseteq M$ (i.e. all places and ports in its preset hold a token). It is weakly enabled if all places (but not necessarily the ports) in $\mathbf{pre}(t)$ are contained in the marking, in other words, if $\mathbf{pre}(t) \subseteq M \cup (\mathbf{pre}(t) \cap Port)$.

Two transitions t, t' are concurrently enabled, if they are both enabled and $\mathbf{pre}(t) \cap \mathbf{pre}(t') = \emptyset$. They are weakly, concurrently enabled if they are both weakly enabled and $\mathbf{pre}(t) \cap \mathbf{pre}(t') = \emptyset$. □

The definition of (weakly) concurrently enabled transitions generalizes in the obvious way to more than two transitions.

A (one-step) *firing* of a net is a transformation of its marking, such that, if T is a set of concurrently enabled transitions, then a non-empty subset S of T is (non-deterministically chosen and) “fires”. Intuitively, “firing” of a transition results in removing the tokens from its **pre** set and depositing them in its **post** set. Technically, if $t = I \xrightarrow{a} O$ is enabled at the marking M , then firing the transition results in the new marking $M' = (M \setminus I) \cup O$ and we write $M \xrightarrow{a} M'$. Similarly for a set $S = \{t_1, \dots, t_n\}$ of concurrently enabled transitions we write $\mathbf{act}(S) = \mathbf{act}(t_1) \cdots \mathbf{act}(t_n)$ and $M \xrightarrow{\mathbf{act}(S)} M'$, where $M' = (M \setminus \bigcup_i \mathbf{pre}(t_i)) \cup \bigcup_i \mathbf{post}(t_i)$.

A sequence of firings

$$M = M_0 \xrightarrow{\mathbf{act}(T_1)} M_1 \xrightarrow{\mathbf{act}(T_2)} \dots \xrightarrow{\mathbf{act}(T_n)} M_n \xrightarrow{\mathbf{act}(T_{n+1})} \dots$$

will be called a *firing sequence*.

A *weak firing sequence* is defined similarly, assuming the T_i 's are weakly enabled sets of transitions and making the input assumptions explicit by listing the input port as if marked.

Definition 4. *A place A of the net is (weakly) reachable, given the marking M , if there exists a finite sequence of firings of (weakly) concurrently enabled sets of transitions $T_i = \{t_{i,1}, \dots, t_{i,m(i)}\}$*

$$M = M_0 \xrightarrow{\mathbf{act}(T_1)} M_1 \xrightarrow{\mathbf{act}(T_2)} \dots \xrightarrow{\mathbf{act}(T_n)} M_n$$

such that $A \in M_n$ (for some $n \geq 0$). □

The reachability of the states (weak, or otherwise) can be represented with a *reachability graph*, a labeled, directed graph (a *transition system*) whose points represent states (sets of marked places, i.e. markings), and arcs represent transitions between two such states.

Before proceeding to define the *concurrent join* of nets (including rigorous definitions for the notions of *rank numbers* and *rank sequence*, we mention a simple notational convention. For a set T of concurrently (weakly) enabled transitions, we write $\mathbf{act}^-(T)$ for the sequence of communication actions in $\mathbf{act}(T)$. If no such actions exist in $\mathbf{act}(T)$, we let $\mathbf{act}^-(T) = 1$. Thus, if $\mathbf{act}(T) = \tau c! \tau \tau d?$, then $\mathbf{act}^-(T) = c!d?$ and if $\mathbf{act}(T) = \tau^n$, then $\mathbf{act}^-(T) = 1$.

3 Concurrent Join of Open Petri Nets

In this section we define the operation of composition of open nets.

3.1 Rank Sequence, Matching Transitions and Composability

The main issue in composing open nets is to formally identify *composable transitions* in the two nets. To achieve this, we define a notion of *rank sequence* for a

communication transition $c!$ or $c?$, to be used in the definition of the concurrent join (composition) of open nets.

Intuitively, the rank sequence (t) of a communication transition t with action $c!$ (similarly for $c?$) consists of rank numbers of order $k \geq 1$, denoted by $(t)_k$, where k indicates the number of times t became enabled in a weak firing sequence for which t becomes enabled at the earliest possible time. More specifically, if $\text{act}(t) = c!$ (and similarly for $c?$), $(t)_k = i$ means that when enabled for the k -th time, exactly $i - 1$ transitions with action $c!$ have become enabled to fire, up to the point where the transition t becomes enabled for the k -th time, at the earliest possible opportunity. The formal definition follows.

Definition 5 (Rank Sequence). *The rank sequence of a communication transition t , denoted by (t) , is the increasing sequence of its rank numbers.*

A natural number $i \geq k$ is a σ -rank-number of order $k \geq 1$ for t , if σ is a weak firing sequence

$$M = M_0 \xrightarrow{\text{act}^-(T_1)} M_1 \xrightarrow{\text{act}^-(T_2)} \dots \xrightarrow{\text{act}^-(T_n)} M_n \xrightarrow{\text{act}^-(T_{n+1})}$$

such that

- there exist $j_1, \dots, j_k \in \{1, 2, \dots\}$ such that
 - if $a < b$, then $j_a < j_b$
 - t is weakly enabled (concurrently with other transitions) in each of T_{j_1}, \dots, T_{j_k}
 - if $s < j_b$, for some b , and t is weakly enabled (concurrently with other transitions) in T_s , then $s = j_a$, for some $a < b$
- if $\text{act}(t) = c!$ (and similarly if $\text{act}(t) = c?$), then there are exactly $i - 1$ transitions $t^{(1)}, \dots, t^{(i-1)}$ with $\text{act}(t^{(r)}) = c!$, for each $r = 1, \dots, i - 1$ such that $t^{(r)} \in T_{j_i}$ for some j_i .

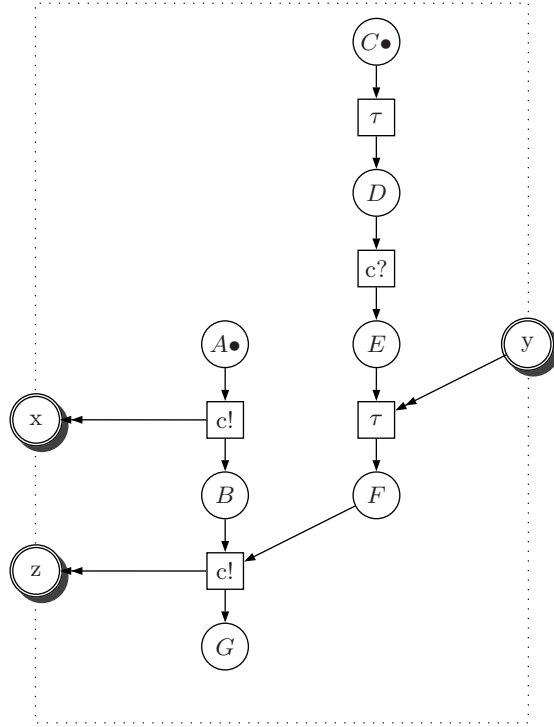
The number i is the rank number of order k for t (denoted by $i = (t)_k$), provided that i is the least σ -rank-number of order k for t . \square

An example will help clarify the content of the definition. First, recall that the definition of a weak firing sequence is meant to capture firings in which all required input is assumed to be provided.

Example 1. Consider the net in Figure 1. For concreteness, we list below all seven weak firing sequences in the example net.

$$\begin{aligned} & \{A, C\} \xrightarrow{c!} \{x, B, D\} \xrightarrow{c?} \{x, B, E, y\} \xrightarrow{1} \{x, B, F\} \xrightarrow{c!} \{x, z, G\} \\ & \{A, C\} \xrightarrow{c!} \{x, B, C\} \xrightarrow{1} \{x, B, D\} \xrightarrow{c?} \{x, B, E, y\} \xrightarrow{1} \{x, B, F\} \\ & \quad \xrightarrow{c!} \{x, z, G\} \\ & \{A, C\} \xrightarrow{1} \{A, D\} \xrightarrow{c!} \{x, B, D\} \xrightarrow{c?} \{x, B, E, y\} \xrightarrow{1} \{x, B, F\} \\ & \quad \xrightarrow{c!} \{x, z, G\} \\ & \{A, C\} \xrightarrow{1} \{A, D\} \xrightarrow{c?} \{A, E, y\} \xrightarrow{c!} \{x, B, E, y\} \xrightarrow{1} \{x, B, F\} \\ & \quad \xrightarrow{c!} \{x, z, G\} \\ & \{A, C\} \xrightarrow{1} \{A, D\} \xrightarrow{c?} \{A, E, y\} \xrightarrow{1} \{A, F\} \xrightarrow{c!} \{x, B, F\} \end{aligned}$$

Fig. 1. Example for the rank sequence of communicative transitions



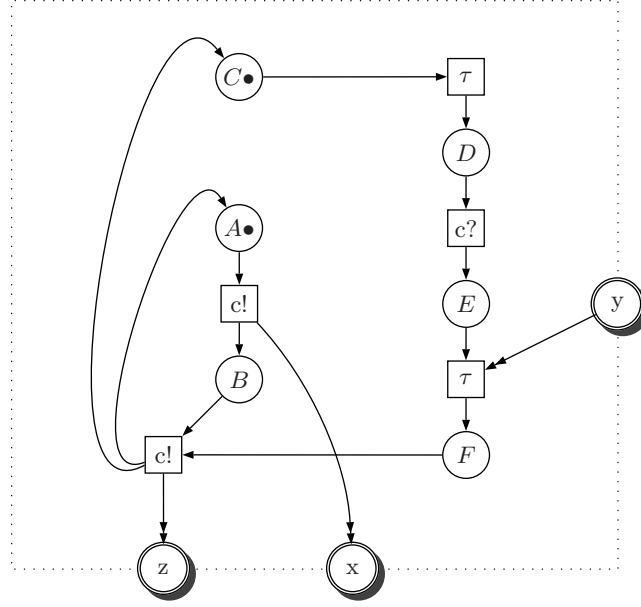
$$\begin{aligned}
 & \xrightarrow{c!} \{x, z, G\} \\
 \{A, C\} & \xrightarrow{1} \{A, D\} \xrightarrow{c!c?} \{x, B, E, y\} \xrightarrow{1} \{x, B, F\} \xrightarrow{c!} \{x, z, G\} \\
 \{A, C\} & \xrightarrow{1} \{A, D\} \xrightarrow{c?} \{A, E, y\} \xrightarrow{c!} \{x, B, F\} \xrightarrow{c!} \{x, z, G\}
 \end{aligned}$$

Let $t_1 = \{A\} \xrightarrow{c!} \{x, B\}$, $t_2 = \{D\} \xrightarrow{c?} \{E\}$ and $t_3 = \{B, F\} \xrightarrow{c!} \{z, G\}$. Then the rank number of order 1 for t_1 is 1, for t_2 it is again 1, while for t_3 it is 2. Since each transition becomes weakly enabled exactly once, the rank sequences of the transitions have each exactly one member, as listed.

Now we modify this net by removing the place G and modifying the transition t_3 to $t_3 = \{B, F\} \xrightarrow{c!} \{A, C, z\}$, so that the net acquires a recursive behavior and firing of the last output transition re-replaces tokens at the initial places A, C . The modified net is shown in Figure 2.

Then each communication transition becomes weakly enabled infinitely many times. The first time t_1 is enabled it has rank number 1. The second time it becomes enabled, it is preceded by exactly two previously enabled and fired transitions, namely t_1 itself (when it first fired) and t_2 , so that its rank number

Fig. 2. Example 2 for the rank sequence of communicative transitions



of order 2 (its rank number the second time it becomes enabled) is 3 (it is the third time a $c!$ transition is enabled to fire in the net). Hence the firing sequence is $(t_1)_1 = 1$ and for $k > 1$, $(t_1)_k = k + 1$.

For t_2 , each time through the loop the only $c?$ transition that has previously fired may be itself and therefore $(t_2)_k = k$.

For t_3 , the first time it becomes enabled it is preceded by exactly one $c!$ transition (this is t_1), hence it is the second time a $c!$ transition becomes enabled in the net, i.e. $(t_3)_1 = 2$ and in general $(t_3)_k = 2k$. \square

Now let \mathcal{A}, \mathcal{B} be two nets and $t = I \xrightarrow{c!} O \cup \{x\}$ an output transition in \mathcal{A} . Consider its rank sequence (t) . The rank number $(t)_1$ designates the number of output transitions that have become enabled up to, including, the point when t becomes enabled (at the earliest possible opportunity) for the first time. Turning to \mathcal{B} , let $r = J \xrightarrow{c?} K, K \cup \{y\} \xrightarrow{\tau} L$ be an input prompt transition, followed by a transition that consumes the input when provided. Consider its rank sequence (r) . Assuming the nets are to be composed concurrently, under what conditions should it be possible and permissible for output from t to be fed as input to r ? The Definition that follows captures our proposed notion of *matching transitions* and, therefore, of *composable services*.

Definition 6 (Matching Transitions). Let \mathcal{A}, \mathcal{B} be two nets, $t = I \xrightarrow{c!}_A O \cup \{x\}$ an output transition in \mathcal{A} and $r = J \xrightarrow{c?}_B K, K \cup \{y\} \xrightarrow{\tau}_B L$ be an

input prompt transition in \mathcal{B} , followed by a transition that consumes the input when provided. If there exist $m, k \leq \min\{\text{length}(t), \text{length}(r)\}$ such that for all $n \leq \min\{\text{length}(t), \text{length}(r)\}$, $(t)_{m+n} = (r)_{k+n}$, then we shall call t and r matching transitions. \square

Matching transitions are precisely the *composable transitions*, i.e. precisely those pairs of transitions in the two nets that can be *composed*, allowing for output from one net to flow, as input, in the other net.

3.2 External Join of Nets

In this Section we define the external concurrent join of oPNs. Both the internal and mixed concurrent join are then easily definable variants, discussed in Section 3.3.

To make the concept simple to understand, we first define a notion of composition at a single pair of matching transitions.

Definition 7. Let \mathcal{A}, \mathcal{B} be two nets, $t = I \xrightarrow{c!}_A O \cup \{x\}$ an output transition in \mathcal{A} and $r = J \xrightarrow{c?}_B K, K \cup \{y\} \xrightarrow{\tau}_B L$ a matching input prompt transition in \mathcal{B} . Assume the set of place and port names of \mathcal{A} is disjoint from the corresponding set of \mathcal{B} (otherwise, injectively rename them). Let also z be a new port name (not occurring in either \mathcal{A} or \mathcal{B}). The join $\mathcal{A} \otimes_{t,r} \mathcal{B}$ of the nets at the matching pair (t, r) is the net obtained as the union of \mathcal{A}, \mathcal{B} , after renaming both ports x (in \mathcal{A}) and y (in \mathcal{B}) to z . In other words, let $\mathcal{A}' = \mathcal{A}_{/x \leftarrow z}$, $\mathcal{B}' = \mathcal{B}_{/y \leftarrow z}$ and $\mathcal{A} \otimes_{t,r} \mathcal{B} = \mathcal{A}' \cup \mathcal{B}' = \mathcal{A}_{/x \leftarrow z} \cup \mathcal{B}_{/y \leftarrow z}$. \square

Technically, the renaming operation (place fusion) $/_{B \leftarrow A}$ is a unary operator with $\mathcal{A}_{/B \leftarrow A}$ being the net obtained by renaming all places or ports labeled by B , using the new label A (which may already be used to label a place or port in the net) and appropriately modifying the transition relation and marking of the net. The net $\mathcal{A}_{/B \leftarrow A}$ is rigorously defined as follows:

- The places of $\mathcal{A}_{/B \leftarrow A}$ are exactly $(Pl(\mathcal{A}) \setminus \{B\}) \cup \{A\}$
- $M_{0, \mathcal{A}_{/B \leftarrow A}} = \begin{cases} M_{0, \mathcal{A}} & \text{if } B \notin M_{0, \mathcal{A}} \\ M_{0, \mathcal{A}} \setminus \{B\} \cup \{A\} & \text{otherwise} \end{cases}$
- The transition relation is derived from that of \mathcal{A} , in the obvious way.
 - If $I \xrightarrow{u}_A O$ is a transition in \mathcal{A} and $(I \cup O) \cap \{B\} = \emptyset$, then $I \xrightarrow{u}_A O$ is a transition in $\mathcal{A}_{/B \leftarrow A}$
 - If $B \notin I \xrightarrow{u}_A O \cup \{B\}$ is a transition in \mathcal{A} , then $I \xrightarrow{u}_A O \cup \{A\}$ is a transition in $\mathcal{A}_{/B \leftarrow A}$
 - If $I \cup \{B\} \xrightarrow{u}_A O \not\neq B$ is a transition in \mathcal{A} , then $I \cup \{A\} \xrightarrow{u}_A O$ is a transition in $\mathcal{A}_{/B \leftarrow A}$
 - If $I \cup \{B\} \xrightarrow{u}_A O \cup \{B\}$ is a transition in \mathcal{A} , then $I \cup \{A\} \xrightarrow{u}_A O \cup \{A\}$ is a transition in $\mathcal{A}_{/B \leftarrow A}$.

The operation generalizes to more than one equations in the obvious way:

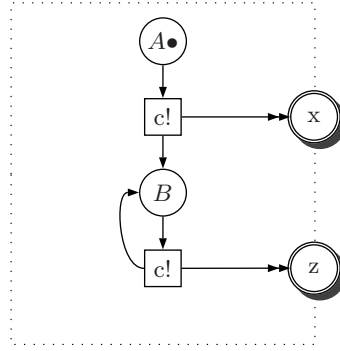
$$\mathcal{A}_{/B \leftarrow A, D \leftarrow C} = (\mathcal{A}_{/B \leftarrow A})_{/D \leftarrow C}$$

and similarly for more equations.

An example will clarify the content of Definitions 6 and 7, as well as the issues involved in net composition.

Example 2. Consider the two nets in Figures 3, 4.

Fig. 3. Example for the join of two nets: The net \mathcal{A}



The net \mathcal{A} of Figure 3 performs an initial output transition on channel c and then gets into a loop, continuously performing output on c .

Let $t_1 = \{A\} \xrightarrow{c!} \{x, B\}$ and $t_2 = \{B\} \xrightarrow{c!} \{z, B\}$ be the two transitions of \mathcal{A} .

The net \mathcal{B} of Figure 4 executes a silent action, prompts for input and, if input is received, it consumes it and repeats this behavior.

Let $r = \{D\} \xrightarrow{c?} \{E\}$ be the input transition of \mathcal{B} .

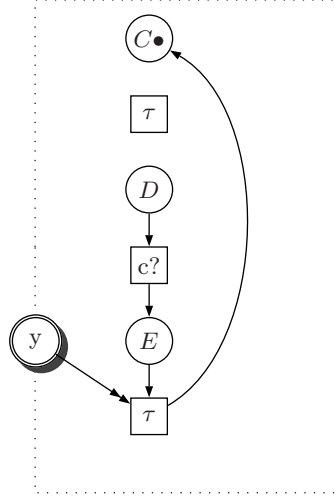
Then we have $\text{length}(t_1) = 1$, $(t_1)_1 = 1$, $\text{length}(t_2) = \omega$, $(t_2)_k = k + 1$, $\text{length}(r) = \omega$ and $(r)_k = k$.

Notice that

$$(t_1)_1 = 1 = (r)_1$$

$$(t_2)_k = k + 1 = (r)_{k+1}$$

Fig. 4. Example for the join of two nets: The net \mathcal{B}



By Definition 6 both (t_1, r) and (t_2, r) are composable pairs of transitions. To obtain the desired composition, let z_1, z_2 be two new port names. Let

$$\begin{aligned}
 \mathcal{A}^{(1)} &= \mathcal{A}_{/x \leftarrow z_1} & \mathcal{B}^{(1)} &= \mathcal{B}_{/y \leftarrow z_1} \\
 \mathcal{A}^{(2)} &= \mathcal{A}_{/z_1 \leftarrow z_2} & \mathcal{B}^{(2)} &= \mathcal{B}_{/z_1 \leftarrow z_2} \\
 &\text{and} \\
 \mathcal{A} \otimes \mathcal{B} &= \mathcal{A}^{(2)} \cup \mathcal{B}^{(2)} \\
 &= \mathcal{A}_{/x \leftarrow z_1, z_1 \leftarrow z_2} \cup \mathcal{B}_{/y \leftarrow z_1, z_1 \leftarrow z_2}
 \end{aligned}$$

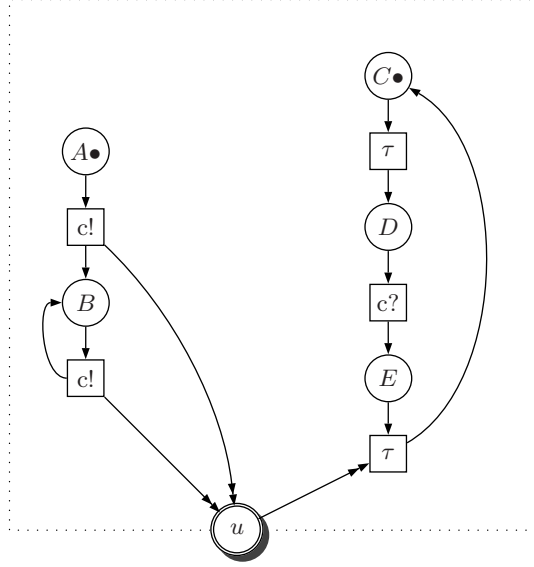
The resulting composite net is shown in Figure 5. The reader may wish to experiment with variations of the example, with more than one initial $c!$ moves on the left, before getting into the loop, and with several $c?$ transitions on the right, before getting into the input loop as well. \square

Incidentally, it is not hard to also model the example of the vending machine, of [7]. We leave this to the interested reader.

Having illustrated, with examples, the intended general notion of composition of open petri nets, we proceed to the formal definition, after a small technical Lemma.

Lemma 1. *Let \mathcal{A}, \mathcal{B} be two nets and J be the set of all matching pairs of transitions (transitions t of \mathcal{A} , paired with matching transitions r in \mathcal{B}). There exists a renaming function σ , defined on the port labels of \mathcal{A}, \mathcal{B} , such that in $\mathcal{A}\sigma, \mathcal{B}\sigma$ all matching pairs in J have their associated communication ports tagged by the same label.*

Fig. 5. Concurrent join of the nets of Figures 3 and 4 (without hiding the communication port)



Proof. If J is finite, then obvious. Otherwise, let j_0, \dots, j_n, \dots be an enumeration of the members of J and for each j_i , let (x_i, y_i) be the pair of port names of the two matching transitions in j_i . Let also $(z_i)_i$ be a sequence of new port names (not occurring in either \mathcal{A} or \mathcal{B}). Define a sequence of partial renaming functions as follows:

- σ_0 : $\text{dom}(\sigma_0) = \{x_0, y_0\}$ and $\sigma_0(x_0) = z_0 = \sigma_0(y_0)$
- σ_{n+1} : Having defined σ_n , let
 $\text{dom}(\sigma_{n+1}) = \text{dom}(\sigma_n) \cup \{x_n, y_n\}$
Case $x_n, y_n \notin \text{dom}(\sigma_n)$
Then extend σ_n , setting $\sigma_{n+1}(x_n) = z_n = \sigma_{n+1}(y_n)$
Case $\{x_n, y_n\} \subseteq \text{dom}(\sigma_n)$
Then let $\sigma_{n+1} = \sigma_n$
Case $x_n \in \text{dom}(\sigma_n)$, but $y_n \notin \text{dom}(\sigma_n)$
Let $\sigma_{n+1}(x_n) = \sigma_n(x_n) = \sigma_{n+1}(y_n)$
Case $x_n \notin \text{dom}(\sigma_n)$, but $y_n \in \text{dom}(\sigma_n)$
Let $\sigma_{n+1}(x_n) = \sigma_n(y_n) = \sigma_{n+1}(y_n)$
- σ : Define $\sigma = \bigcup_n \sigma_n$

By definition of σ , if $(t, r) = j_k$ is the k -th matching pair, then in $\mathcal{A}\sigma$ and $\mathcal{B}\sigma$ the matching transitions t, r have their corresponding ports tagged by the same label.

Definition 8 (External Concurrent Join of Open Nets). Let \mathcal{A}, \mathcal{B} be two nets, assuming their sets of ports and places are disjoint (if not, injectively rename each one of them) and let σ be the port renaming function of matching transitions, as in Lemma 1. The external concurrent join of \mathcal{A}, \mathcal{B} , is defined as

$$\mathcal{A} \otimes \mathcal{B} = \mathcal{A}\sigma \cup \mathcal{B}\sigma$$

where recall that $\mathcal{A}\sigma$ is the net obtained from \mathcal{A} , but with communication port names modified by σ . \square

3.3 Internal and Mixed Join

In Section 3.2 we defined an operation of external concurrent join, in which communication ports remain available for further external communication. In this Section we slightly modify our definitions and obtain notions of *internal concurrent join* (hiding the ports after composition) and *mixed concurrent join* (selectively hiding or not the communication ports, depending on the nature of the communication to take place).

To achieve this, the definition of oPN's needs to be slightly modified, allowing for a distinction between *external* and *internal* communication ports, Port_{int} , Port_{ext} , so that internalization of all (or some) ports after composition can be performed. But also, by introducing a distinction between private and public (internal and external) communicative actions.

The case of internalizing all ports after composition of matching transitions is straightforward, by letting $\mathcal{A}|\sigma$ (where σ is the renaming function of Lemma 1) be the net obtained by first renaming according to σ , but also by removing from its set of external communication ports the range of σ and adding it to its set of internal communication ports.

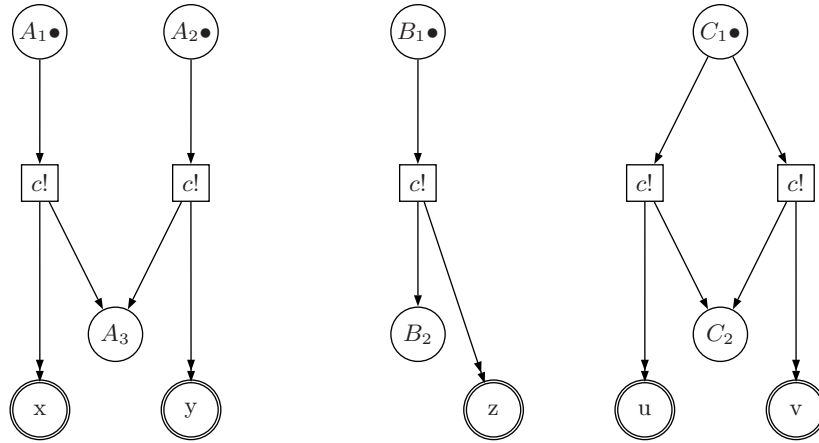
For the case of a mixed concurrent join, communicative actions can be assigned an annotation, e.g. $!c: \text{in}, ?d: \text{ex}$ (or no annotation, meaning that both private and public communication are acceptable). This affects the definition of matching (composable) transitions, since now the additional requirement that the annotations agree must be imposed. Furthermore, the set of composable transitions for two oPNs splits into two sets J_{in}, J_{ex} and similarly for the renaming function σ of Lemma 1, splitting to σ_{in}, σ_{ex} . The mixed concurrent join can be then simply defined as $\mathcal{A}\sigma_{ex}|\sigma_{in} \cup \mathcal{B}\sigma_{ex}|\sigma_{in}$. Details are left to the interested reader.

4 Behavioral Equivalence of Open Nets and Public View of Services

4.1 Motivation

In this Section we investigate notions of behavioral equivalence of oPN's, which appear to be natural given the intended interpretation of an oPN as a Service. For motivation, consider the three nets in Figure 6.

Fig. 6. Example for the Behavior of Nets



Each of the example nets can output on channel c . The fact that, in the first net, we have drawn a separate output buffer for each output transition is immaterial when it comes to joining this oPN with any net that expects input on c . Indeed, as an example, consider the oPN with transitions $\{X\} \xrightarrow{c?} \{Y\}$, $\{Y, a\} \xrightarrow{\tau} \{Z\}$ (a net that performs input on c and the input buffer is called a). Given our definition of concurrent join of oPN's, the two output buffers and the input buffer will be all identified to one. A similar observation applies to the third net in Figure 6. Roughly then and intuitively at the moment, the behavior of all three nets of Figure 6 is the same and this should be reflected in the *public view* of the nets.

4.2 Abstract Behavior Graph

In Section 2.3, if T is a set of concurrently enabled transitions and $S \subseteq T$ is a set of non-deterministically selected transitions that fire concurrently, $S = \{t_1, \dots, t_i\}$, we wrote $\text{act}(S) = \text{act}(t_1) \cdots \text{act}(t_i)$. In other words, $\text{act}(S)$ was regarded as a string (or multiset, since order is unimportant). We define here a notion of reduced reachability graph, in which multisets are replaced by sets. Similarly for the reduced sets $\text{act}^-(T)$. Further, the only transitions we consider are those corresponding to maximal sets of concurrently weakly enabled transitions and where all transitions fire simultaneously.

Definition 9 (Reduced Reachability Graph). Let \mathcal{A} be an oPN and M a state (marking) of the net. A state N of the net is a successor state of M if there exists a maximal set of concurrently weakly enabled transitions $T = \{t_1, \dots, t_k\}$ and N results from M by simultaneously firing all the t_i . We then write $M \xrightarrow{\text{act}^-(T)} N$, where $\text{act}^-(T)$ is the reduced set of action names (omitting internal action names) of the transitions t_i in T . A state N is a descendant of a state M if there is a finite sequence of successor states to the initial state, such that the end state is N .

The reduced reachability graph of \mathcal{A} is a rooted, labeled directed graph, whose nodes are the descendants of the root node (the initial state of the net) and where an edge exists from node M to node N just in case N is a successor state of M via a maximal set T of concurrently weakly enabled transitions. Edges are labeled by the reduced sets of actions of the transitions in T . \square

As an example, the reduced reachability graphs of the nets in Figure 6 are the following:

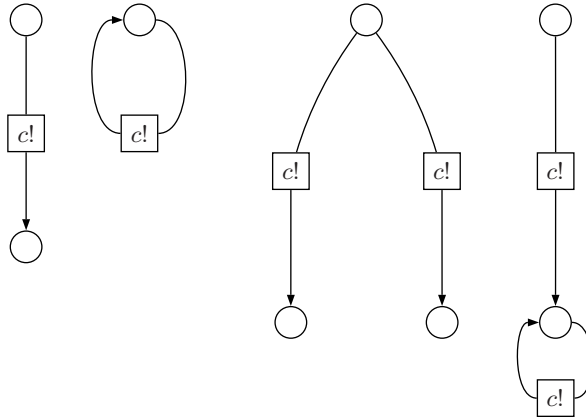
$$\begin{array}{ll} \text{first net} & \{A_1, A_2\} \xrightarrow{c!} \{A_3, x, y\} \\ \text{second net} & \{B_1\} \xrightarrow{c!} \{B_2, z\} \\ \text{third net} & \{C_1\} \xrightarrow{c!} \{C_2, u\} \\ & \{C_1\} \xrightarrow{c!} \{C_2, v\} \end{array}$$

In the case of oPNs that we investigate in this report, the significant behavior of oPNs relates exclusively to their input/output behavior (and any other, internal, behavior is hidden by a silent invisible action τ). Intuitively, then, all nets in Figure 6 are behaviorally equivalent: they can all perform output on the same channel c , and get to a state where output is available for consumption. In the sequel, we discuss a natural notion of oPN equivalence.

The reduced reachability graph of an oPN \mathcal{M} still contains redundant information. For example, if an output on channel c can be performed at a state S , leading to a successor state S' , then the information that output is available at state S' is redundant. This is particularly true because, the number and labeling of output ports through which output from channel c is available is immaterial, given that any oPN \mathcal{N} that can synchronize with \mathcal{M} , will fuse any number of output places of \mathcal{M} at state S' to exactly one, identifying it with its input place. In the sequel, we define a natural notion of *behavior graph*, taking into consideration the remarks just made.

Definition 10 (Behavior Graph of an oPN). Let \mathcal{A} be an oPN and \mathcal{R}_A its reduced reachability graph. Its behavior graph \mathcal{G}_A is obtained from the reduced reachability graph \mathcal{R}_A by removing from each node all port labels. Further, by injectively renaming finite sets of place labels, state names are represented with single tags, like S , S' etc. \square

Fig. 7. The behavior graph of the oPN's of Figure 6, followed by the behavior graph of $S \xrightarrow{c!} S$. The third graph is the behavior graph of a net like the last oPN to the right, at Figure 6, where the two $c!$ actions lead to different places C_2 and C_3 , rather than to a single one. The fourth graph corresponds to the oPN of Figure 3.



Thus, the behavior graphs of the three oPN's turn out to be the graphs

$$\begin{aligned} \text{first net } S_1 &\xrightarrow{c!} S_2 \\ \text{second net } S'_1 &\xrightarrow{c!} S'_2 \\ \text{third net } S''_1 &\xrightarrow{c!} S''_2 \end{aligned}$$

For comparison, consider the net with a single transition $\{A\} \xrightarrow{c!} \{A, x\}$, which outputs on channel c and repeats its behavior. Its behavior graph is then simply $S \xrightarrow{c!} S$.

Graphically, we represent behavior graphs by drawing circles for states, with arcs labeled by sets of actions. Thus, the three oPN's of Figure 6 have the behavior graph shown first on the left in Figure 7, where we place the arc label in a box.

Intuitively, the first and third behavior graphs do not really represent different behavior, but rather different ways to represent this behavior. To capture this intuition, we may turn to the standard notion of transition system bisimilarity, since behavior graphs are really labeled transition systems. For our present purposes, a weaker notion of *trace equivalence* is sufficient.

Definition 11. We say that the nets are trace equivalent, in symbols $\mathcal{M} \simeq \mathcal{N}$, if and only if their behavior graphs have the same set of maximal paths. \square

As an example, note that the first and third graphs in Figure 7 are bisimilar and so are the second and fourth graphs in the same Figure, hence their associated open nets are behaviorally equivalent.

Further, notice that the same pairs of behavior graphs have exactly the same sets of maximal paths, hence their associated open nets are also trace equivalent.

Definition 12 (Abstract Behavior Graph). An abstract behavior graph is defined as the equivalence class, under trace equivalence, of a behavior graph. \square

Our proposal then, regarding the *public view* of a service, is to precisely take its abstract behavior graph (any representative of the class) as the *public view* of the service.

5 Conclusions and Further Research

This report focused exclusively on the issue of *composability* and *composition* of services, represented as open petri nets. By introducing a technical notion of *firing rank sequence* we have produced a completely rigorous account of composability and composition of services. Further, the notion of abstract behavior graph we proposed provides an account for the *public view* of services and the means to decide on *compatibility* of services.

By contrast, we have not addressed the issue of *compositional* definition of services. As noted in the literature (see e.g. [9]), it is advantageous to address this issue by combining the process algebraic with the petri net approach to modeling services. We have recently worked on this issue in another report [4], where we define a class of open nets appropriate for the modeling of a value-passing process algebra (value-passing CCS), thus contributing in this direction as well.

References

1. W.M.P. van der Aalst, “A class of petri nets for modeling and analysing business processes”, Computing science report 95/26, Eindhoven University of Technology, 1995.
2. F. Leymann, D. Roller, and M. Schmidt. Web services and business process management. *IBM Systems Journal*, 41(2), 2002.
3. Hartonas, C., Kodokostas, D. and Kokkinos, K., “Petri Net Semantics for Communicating Agents”, in *Annals of Mathematics, Computing & Teleinformatics*, vol 1-4, pp 31-40, 2006.
4. Hartonas, C. and Kodokostas, D., “Open Net Semantics for Value-Passing Processes”, Technical Report, TEI Larissa, 2007.
5. A. Martens. *Verteilte Geschäftsprozesse - Modellierung und Verifikation mit Hilfe von Web Services*. PhD thesis, Institut für Informatik, Humboldt-Universität zu Berlin, 2004.

6. Axel Martens, “On Compatibility of Web Services”, ???
7. Massuthe, P., Reisig, W. and Schmidt, K., “An Operating Guideline Approach to the Service Oriented Architecture”, in *Annals of Mathematics, Computing & Teleinformatics*, vol 1-3, pp 35-44, 2005.
8. Kathrin Kaschner, Peter Massuthe, and Karsten Wolf, “Symbolic Representation of Operating Guidelines for Services”, *Petri Net Newsletter*, 72:21-28, April 2007.
9. Twan Basten, *In Terms of Nets: System Design with Petri Nets and Process Algebra*, PhD thesis, Eindhoven University of Technology, 1998 (ISBN 90-386-0631-1).