# Asynchronous Composition of Petri Nets via Places*

(extended abstract)

Nikolay Anisimov and Alexey Kovalenko

Institute for Automation and Control Processes
Far East Branch of the Russian Academy of Sciences
5 Radio Str., Vladivostok, 690041, Russia
phone: +7 4232 310 418; fax: +7 4232 310 452
e-mail: anisimov@iapu2.marine.su

**Abstract.** This paper addresses the problem of defining Petri net calculi. The need for defining different problem-oriented calculi and their complexity is stressed. To avoid cumbersome and clumsy definitions of net operations we propose to use a very small set of auxiliary basic operations. One of them, called $s$–merging, is introduced in this paper. Some standard properties of this operation are proved. For example, using $s$–merging we define Petri net calculus for specification communication protocols including sequential composition, choice, iteration, and disruption operations in a simple and elegant way. It is shown that some standard properties of the operations directly follow from properties of $s$–merging.

## 1 Introduction

It is well-known that Petri nets provide a very suitable formal model for parallel and distributed systems at the basic level. At the same time, due to a lack of compositionality, they are not widely used for real-world applications of industrial size. One of the ways to enhance the compositionality of Petri nets is to develop a Petri net calculus (PN-calculus) for a specific area of application. However, all known attempts of building PN-calculi for real-world applications resulted in very complex definitions which are very difficult to deal with. To overcome this problem, we suggest formal tools that will help designers to build their own problem-oriented PN-calculi. In this paper we present one of these tools – basic net operation of place merging (s-merging). Using this operation, a designer can easily define several problem-oriented net operations for PN-calculus.

One of the most promising direction in the theory of Petri nets is the so called compositional approach and algebraic approach in particular. The last means an introduction of Petri net calculus, which usually consists of a carrier set of nets, together with a set of net operations. There are several reasons for this interest. First, using operations on nets one can define a "true-concurrency" semantics of different parallel languages, whose original semantics are defined within an interleaving approach. For instance, there are attempts for defining net based semantics both for well-known abstract languages like CCS, TCSP, ACP and for more practical ones like `OCCAM` and LOTOS. Second, PN-calculus is of great interest in its own right, without relation to any language. Once defined, it allows one to construct and to specify nets of a large size quite naturally and simply. Sometimes, it is possible to derive some properties, or any other useful information about designed nets, during the process of construction. We should mention some such Petri net calculus [4, 5, 7, 6]. Much in the net algebraic approach has been done within the DEMON/ESPRIT and Caliban projects.

Obviously, an actual set of operations in any PN-calculus depends of the area of the application of the calculus. For example, it is clear that for `OCCAM` and for LOTOS, we should define different sets of net operations. Therefore, we need some techniques for defining different PN-calculi. But, if we have a closer look at any existing PN-calculus, we can see that definitions of almost all net-operations are very long and clumsy, while their intuitive meanings are quite clear. All known attempts of building PN-calculi for real-world applications resulted in very complex definitions which are very difficult to work with. Moreover, it takes a great deal of time and efforts to prove even some standard properties of these operations, e.g. associativity, commutativity, congruence of an equivalence w.r.t. these operations, etc. As an example, we mention our own experience with defining a set of net operations for a communication protocol specification [1]. As a result, we have obtained a quite complex set of definitions and auxiliary notions. Other efforts to define them, in a more elegant and simple way, was not crowned with a success. Something should be done to cope with this problem.

To do so in these notes we present the following idea: once we are unsuccessful with defining net operations in a simple way, it might be reasonable to define a very small set of auxiliary operations that may be quite complex. However, at the same time, this auxiliary set may enable us to define easily the main set of net operations. Moreover, once having proved all important properties of these auxiliary operations, we can automatically

* This work is supported by the Russian Fund for Basic Research (Project No. 96-01-0177).

derive properties of other operations. Here we introduce one such operation, called *place merging*. Using this operation, it is easy to define various well-known operations like *sequential composition, choice, iteration*, etc. Then we show how to do so by using the just mentioned example of the net algebra for communication protocol specification [1]. The other auxiliary operation of transition merging is reported in [2] and can be used for the definition of *parallel composition* of nets.

## 2 Basic definitions and notations

In this section we give a set of basic definitions and notions from the Petri net theory. Let $A$ be a set. Then $\mathcal{P}(A)$ will denote the powerset of $A$. Let $\rho \subseteq \mathcal{P}(A)$, then denote $\|\rho\| = \bigcup_{x \in \rho} x$. A multiset over a set $A$ is a function $\mu : A \to \{0, 1, 2, ...\}$. A set of all multisets over $A$ will be denoted as $\mu A$.

**Definition 1.** Let $Act$ be an *alphabet* of labels. Then *a labelled net* is a tuple $N = \langle S, T \rangle$ where

1. $S$ is a finite set of *places* with typical elements $s_1$, $s_2$, ...
2. $T \subseteq \mathcal{P}(S) \times Act \times \mathcal{P}(S)$ is a finite set of *transitions* with typical elements $t_1$,$t_2$, ...

Note that this definition slightly diverges from the conventional ones [8] while defines the same objects. This will simplify defining net operations. We also use the following notation.

Let $\langle S, T \rangle$ be a labelled net and $t \in T$, $s \in S$, $T' \subseteq T$, $S' \subseteq S$, then denote:

1. Given a transition $t = \langle Q_1, a, Q_2 \rangle \in T$, then ${}^\bullet t = Q_1$ is called *preset* of $t$, $t^\bullet = Q_2$ *postset*, and $\underline{t} = a$ *a name* of the transition. Write ${}^\bullet t^\bullet = {}^\bullet t \cup t^\bullet$.
2. ${}^\bullet(T') = \bigcup_{t \in T'} {}^\bullet t$, $(T')^\bullet = \bigcup_{t \in T'} t^\bullet$, ${}^\bullet(T')^\bullet = {}^\bullet(T') \cup (T')^\bullet$
3. ${}^\bullet s = \{t \mid s \in t^\bullet\}$, $s^\bullet = \{t \mid s \in {}^\bullet t\}$, ${}^\bullet s^\bullet = {}^\bullet s \cup s^\bullet$
4. ${}^\bullet(S') = \bigcup_{s \in S'} {}^\bullet s$, $(S')^\bullet = \bigcup_{s \in S'} s^\bullet$, ${}^\bullet(S')^\bullet = {}^\bullet(S') \cup (S')^\bullet$.

So we can write $t = \langle {}^\bullet t, \underline{t}, t^\bullet \rangle$.

Define marking of a net as usual $M : S \to \{0, 1, 2, ...\}$. A transition $t \in T$ is said to be enabled at the marking $M$ *iff* ${}^\bullet t \leq M$. An enabled transition may occur with the marking $M$ to yield a new marking $M'$, write $M[t\rangle M'$, where $M'(p) = M(p) \setminus {}^\bullet t \cup t^\bullet$ for each $s \in S$. If $v = t_1 t_2 ... t_n \in T^*$, then $M[v\rangle M'$ means that $\exists M_1, M_2, ..., M_{n-1} : M[t_1\rangle M_1[t_2\rangle ... M_{n-1}[t_n\rangle M'$. We will write $M(w\rangle M'$, $w \in Act^*$ if $\exists v \in T^* : M[v\rangle M'$ and $\sigma(v) = w$. A marking $M$ is called *safe* if $M(s) \leq 1$ for all $s \in S$.

We will use a conventional graphical representation of nets in which places are represented as circles, and transitions as boxes with directed arcs. Names of the transitions will be placed inside the appropriate boxes. Sometimes, it is convenient to define labelled nets up to isomorphism.

**Definition 2.** Two labelled nets $N_1 = \langle S_1, T_1 \rangle$ and $N_2 = \langle S_2, T_2 \rangle$ are said to be *isomorphic*, write $N_1 = N_2$, *iff* there is a bijection $\omega : S_1 \to S_2$ such that

$$t \in T_1 \iff \langle \omega({}^\bullet t), \underline{t}, \omega(t^\bullet) \rangle \in T_2$$

We will also need a simple operation of union two disjoint nets.

**Definition 3.** Let $N_1 = \langle S_1, T_1 \rangle$, $N_2 = \langle S_2, T_2 \rangle$ be two disjoint nets, i.e. $(S_1 \cup T_1) \cap (S_2 \cup T_2) = \emptyset$, then the *disjoint union* of these nets is a new net $N = N_1 \uplus N_2 \stackrel{def}{=} \langle S_1 \cup S_2, T_1 \cup T_2 \rangle$

## 3 Merging of places

In this section we introduce a basic operation of *place merging*. Previously let us define a notion of *s–access point*.

**Definition 4.** Let $N = \langle S, T \rangle$ be a net. Then a *s-access point* is a set $\rho \subseteq \mathcal{P}(S)$ such that $\forall M, M' \in \rho : M \subseteq M' \Rightarrow M = M'$.

In other words, $s$–access point is a set of subsets on $S$ such that these subsets are mutually incompatible with respect to inclusion. Intuitively, a $s$–access point can be thought of as a set of safe markings, and a reachability one of them makes a synchronization possible. For this reason, elements of $\rho$ will be denoted by the letter $M$, possibly with indexes.

**Definition 5.** Let $N = \langle S, T\rangle$ be a net defined over $Act$ with $s$–access points $\rho_1$ and $\rho_2$. Denote $S^1 = \bigcup_{M\in\rho_1}$, $S^2 = \bigcup_{M\in\rho_2}$, $S^{12} = S^1 \cup S^2$. Then s–merging of the net $N$ w.r.t. $\rho_1$ and $\rho_2$ builds a new net $N' = S_{merg}(N, \rho_1, \rho_2)$ defined over $Act$ where

1. $S' = S \setminus S^{12} \cup S^1 \times S^2$;
2. $T' = \{\langle Q_1, \underline{t}, Q_2\rangle \mid M_1 \in \rho_1, M_2 \in \rho_2\}$ where

$$Q_1 = {}^\bullet t \setminus S^{12} \cup \bigcup_{M'\in\rho_1} ({}^\bullet t \cap M') \times M_2 \cup \bigcup_{M''\in\rho_2} M_1 \times ({}^\bullet t \cap M'')$$

$$Q_2 = t^\bullet \setminus S^{12} \cup \bigcup_{M'\in\rho_1} (t^\bullet \cap M') \times M_2 \cup \bigcup_{M''\in\rho_2} M_1 \times (t^\bullet \cap M'')$$

The idea of synchronization via places is itself quite simple. If a net reaches a marking from one $s$–access point, $M \in \rho_1$, then there is the possibility to jump into one of the markings of an other point, $M' \in \rho_2$. Note that this jump is non-deterministic. To implement this scheme of synchronization, we first multiply synchronized sets of places, obtaining $S^1 \times S^2$. Second, each transition surrounding merging places is split, each copy of it being generated for one possible marking. For example, if a transition $t \in T^1 \setminus T^2$ then it is copied $|\rho_2|$ times where each copy implements a jumping into marking $M$.

According to the above definition, the operation of $s$–merging substantially changes the structure of a net. However, if we previously have defined other access points, then after applying the operation we have to redefine them for a new net, trying to keep their intuitive meaning. The next definition accomplishes this transformation.

**Definition 6.** Let $N = \langle S, T\rangle$ be a net, $\rho_1 \subseteq \mathcal{P}(S^1)$, $\rho_2 \subseteq \mathcal{P}(S^2)$, $\rho \subseteq \mathcal{P}(S')$ its $s$-access points. Then the transformation of $\rho$ onto the net $N' = S_{merg}(N, \rho_1, \rho_2)$ is a new $s$–access point:

$$\tilde{\rho} = \left\{M \setminus S^{12} \cup (M \cap S^1) \times M_2 \cup M_1 \times (M \cap S^2) \mid M \in \rho, M_1 \in \rho_1, M_2 \in \rho_2\right\}$$

**Definition 7.** Let $N_1$ and $N_2$ be two disjoint nets with $s$–access points $\rho_1$ and $\rho_2$, respectively. Then $s$–merging of these nets with respect to $\rho_1$ and $\rho_2$ is a new net

$$N = (N_1 \ {}_{\rho_1}\!\oplus_{\rho_2} N_2) \stackrel{def}{=} S_{merg}(N_1 \uplus N_2, \rho_1, \rho_2)$$

*Example 1.* Fig.1 show the net $N = \langle S, T\rangle$ where $Act = \{a, b, c\}$,

1. $S = \{s_1, s_2, s_3, s_4\}$,
2. $T = \left\{\langle\{s_1\}, a, \{s_2\}\rangle, \langle\{s_2\}, c, \{s_3\}\rangle, \langle\{s_1\}, b, \{s_4\}\rangle\right\}$.

There are two $\rho$-access points $\rho_1 = \{\{p_3\}, \{p_4\}\}$, $\rho_2 = \{\{p_1\}\}$. As a result of $s$-merging operation on $N$ w.r.t $\rho_1$ and $\rho_2$ we obtain a new net $N' = s{-}merg(N, \rho_1, \rho_2) = \langle S', T'\rangle$, which, intuitively, performs an iteration on net $N$. According to the definition we have:

1. $S' = \{s_2\} \cup \{s_3, s_4\} \times \{s_1\} = \{s_2, s_{31}, s_{41}\}$, where $s_{i\ldots n} = s_i \ldots s_n$.
2. Transitions from $T$ are split as follows:
   $\langle\{s_1\}, a, \{s_2\}\rangle \longmapsto \langle\{s_{31}\}, a, \{s_2\}\rangle$ and $\langle\{s_{41}\}, a, \{s_2\}\rangle$;
   $\langle\{s_2\}, c, \{s_3\}\rangle \longmapsto \langle\{s_2\}, c, \{s_{31}\}\rangle$;
   $\langle\{s_1\}, b, \{s_4\}\rangle \longmapsto \langle\{s_{31}\}, b, \{s_4\}\rangle$ and $\langle\{s_{41}\}, b, \{s_4\}\rangle$.

The resulted net is presented on Fig.1. Access points $\rho_1$ and $\rho_2$ are extended onto $\tilde{\rho}_1 = \left\{\{s_{31}\}, \{s_{41}\}\right\} = \tilde{\rho}_2$.

## 4 Some properties of $s$–merging operation

In this section we prove some useful properties of $s$-merging operations. First, it is not hard to show the commutativity of $s$–merging (of course, up to isomorphism).

**Proposition 8.** *Let $N$ be a net and $\rho_1$ and $\rho_2$ its $s$–access points. Then*

$$S_{merg}(N, \rho_1, \rho_2) = S_{merg}(N, \rho_2, \rho_1)$$

[htbp]

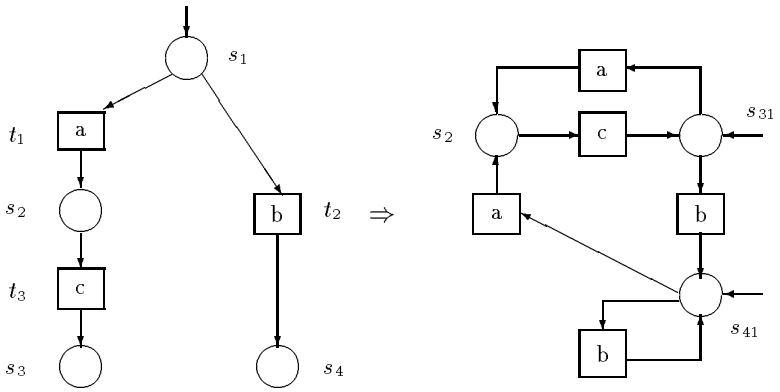


**Fig. 1.** Example of $s$–merging operation

The next important standard property is associativity. Unfortunately, in general, this property is not valid:

$$S_{merg}\big(S_{merg}(N,\rho_1,\rho_2),\tilde{\rho}_3,\tilde{\rho}_4\big) \neq \big(S_{merg}(S_{merg}(N,\rho_3,\rho_4),\tilde{\rho}_1,\tilde{\rho}_2\big)$$

One can be convinced in it taking the case where $S^1 \cap S^3 \neq \emptyset \neq S^2 \cap S^4$ that results in even different sets of places. Fortunately, for the binary version of s-merging associativity is valid.

**Theorem 9.** *Let $N_1$, $N_2$ and $N_3$ be nets and $\rho_1$ is a $s$–access point of $N_1$, $\rho_2$, $\rho_3$ of $N_2$ and $\rho_4$ of $N_3$. Then*

$$\big((N_1\ {}_{\rho_1}\oplus_{\rho_2}\ N_2)\ {}_{\tilde{\rho}_3}\oplus_{\rho_4}\ N_3\big) = \big(N_1\ {}_{\rho_1}\oplus_{\tilde{\rho}_2}\ (N_2\ {}_{\rho_3}\oplus_{\rho_4}\ N_3)\big)$$

.

The next, and very important property, is a congruence of an equivalence w.r.t. $s$–merging operation. In this paper we take a well-known notion of bisimulation based on arbitrary interleaving, see e.g. [3]. Here we generalize a notion of initial markings, substituting it by a set of $s$–access points $(N,\Pi)$ where $\Pi = (\rho_1,\rho_2,...,\rho_n)$, $n \geq 1$. Each $s$–access point can be thought of as a set of alternative markings.

**Definition 10.** Let $(N_1,\Pi_1)$ and $(N_2,\Pi_2)$ be nets with $s$–access points, $\Pi_i = (\rho_{i1},...,\rho_{in})$, $i \in \{1,2\}$. The nets are said to be *bisimular equivalent*, write
$(N_1,\Pi_1) \approx (N_2,\Pi_2)$, *iff* there is a relation $R \subseteq \mu S_1 \times \mu S_2$ such that:

1. for each $j \in \{1,...,n\}$
   (a) $\forall M_1 \in \rho_{1j} \quad \exists M_2 \in \rho_{2j}$: $(M_1,M_2) \in R$
   (b) $\forall M_2 \in \rho_{2j} \quad \exists M_1 \in \rho_{1j}$: $(M_1,M_2) \in R$
2. whenever $(M_1,M_2) \in R$ and $w \in Act^*$ then
   (a) $M_1(w\rangle M_1' \Rightarrow \exists M_2'$: $M_2(w\rangle M_2'$ and $(M_1',M_2') \in R$
   (b) $M_2(w\rangle M_2' \Rightarrow \exists M_1'$: $M_1(w\rangle M_1'$ and $(M_1',M_2') \in R$

Now we can state without proof one of the main result of the paper.

**Theorem 11.** *If $(N_1,\Pi_1) \approx (N_2,\Pi_2)$ where $\Pi_i = (\rho_{i1},..,\rho_{in})$, $i \in \{1,2\}$, then*

$$(S_{merg}(N_1,\rho_{1i},\rho_{1j}),\Pi_1) \approx (S_{merg}(N_2,\rho_{2i},\rho_{2j}),\Pi_2)$$

## 5 Petri net entity calculus

In this section we demonstrate how easy and convenient it is to use $s$–merging for the definition of some non-trivial net operations. As an example we take an algebra of macronets for the specification of communication protocols with simplified operation of parallel composition [1].

**Definition 12.** A Petri net entity (or entity for short) is a tuple $E = \langle N,h,l,d\rangle$ where

1. $N$ is a labelled PT-net, and
2. $h$, $l$, $d$ are $\rho$-access points called *head*, *tail* and *internal* access points, respectively

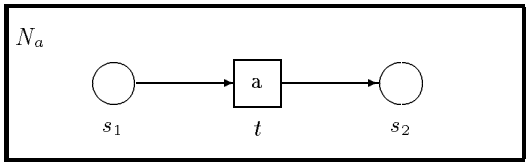


**Fig. 2.** Atomic entity $N_a$

Intuitively, the entity is nothing else but a net equipped with information for future compositions. Here, head and tail access points carry the information about initial and final states of the net. An internal point contains the information about all states of a net that will be needed for the disruption operation.

**Definition 13.** Let $E_1 = \langle N_1, h_1, l_1, d_1\rangle$ , $E_2 = \langle N_2, h_2, l_2, d_2\rangle$ be entities with nets $N_1 = \langle S_1, T_1\rangle$ and $N_2 = \langle S_2, T_2\rangle$. Let also $Act$ be an alphabet. Then a Petri net entity calculus on $Act$ is defined as follows

1. Let $a \in Act$. An *atomic entity* is an entity $E_a = \langle N_a, h, l, d\rangle$ where
$$N_a = \langle \{s_1, s_2\}, \{t\}, \{\langle \{s_1\}, t, \{s_2\}\rangle\}, \{(t, a)\}\rangle$$
where $h = \{\{s_1\}\}$, $l = \{\{s_2\}\}$, $d = \{\{s_1\}, \{s_2\}\}$, see Fig.2.
2. A *sequential composition* of entities $E_1$ and $E_2$ is an entity
$$(E_1; E_2) = \langle (N_1 \ {}_{l_1}\oplus_{h_2} \ N_2), \tilde{h}_1, \tilde{l}_2, \tilde{d}_1 \cup \tilde{d}_2\rangle$$
3. A *choice* of entities $E_1$ and $E_2$ is an entity
$$(E_1 [] E_2) = \langle (N_1 \ {}_{h_1}\oplus_{h_2} \ N_2), \tilde{h}_1 \cup \tilde{h}_2, \tilde{l}_1 \cup \tilde{l}_2, \tilde{d}_1 \cup \tilde{d}_2\rangle$$
4. An *iteration* of the entity $E_1$ is an entity
$$*(E_1) = \langle S_{merg}(N_1, h_1, l_1), \tilde{h}_1, \tilde{l}_1, \tilde{d}_1\rangle$$
5. A (pure) *parallel composition* of entities $E_1$ and $E_2$ is an entity
$$(E_1 \| E_2) = \langle (N_1 \uplus N_2), h_1 \otimes h_2, l_1 \otimes l_2, d_1 \otimes d_2\rangle$$
where $\rho_1 \otimes \rho_2 = \{x \cup y \mid x \in \rho_1, y \in \rho_2\}$
6. A *disruption* of entities $E_1$ and $E_2$ is an entity
$$(E_1 \circ E_2) = \langle (N_1 \ {}_{d_1}\oplus_{h_2} \ N_2), \tilde{h}_1, \tilde{l}_2, \tilde{d}_1 \cup \tilde{d}_2\rangle$$

So the calculus whose definition in [1] required about ten pages, here takes only a few lines.

Note that the disruption operation is slightly different from a well-known disabling operation used in LOTOS where an entity in a final state can not be disrupted. One can easily define this version of the operation as follows:

$$(E_1 \ [\rangle \ E_2) = \langle (N_1 \ {}_{d_1'}\oplus_{h_2} \ N_2), \tilde{h}_1, \tilde{l}_2, \tilde{d}_1 \cup \tilde{d}_2\rangle \text{ where } d_1' = d_1 \setminus l_1$$

Now we can prove some pleasant properties of the net operations.

**Theorem 14.** *Let $E_1$, $E_2$, $E_3$ be Petri net entities. Then*

1. $(E_1 [] E_2) = (E_2 [] E_1)$
2. $((E_1; E_2); E_3) = (E_1; (E_2; E_3))$
3. $((E_1 [] E_2) [] E_3) = (E_1 [] (E_2 [] E_3))$
4. $((E_1 \circ E_2); E_3) = (E_1 \circ (E_2; E_3))$
5. $((E_1 \circ E_2) \circ E_3) = (E_1 \circ (E_2 \circ E_3))$

Using the theorem 11 we can also easily derive the congruence of bisimulation equivalence with respect to all net operations. This result is of great practical importance because it allows one to substitute subnets by equivalent ones.

**Theorem 15.** *An equivalence of entities is a congruence relation with respect to sequential composition, choice, iteration and disruption operations, i.e. if $E_1 \approx E_1'$ and $E_2 \approx E_2'$ then*

1. $(E_1; E_2) \approx (E_1'; E_2')$
2. $(E_1 [] E_2) \approx (E_1' [] E_2')$
3. $(E_1 \circ E_2) \approx (E_1' \circ E_2')$
4. $*(E_1) \approx *(E_1')$

## 6 Concluding remarks

In this paper we offered formal tools for building different problem- oriented Petri net calculus. The more general idea underlining this paper can be formulated as follows: as a rule, one who needs a net calculus takes one of the basic net models and directly defines a set of net operations. We suggest another scheme, offering not only a basic model, but a small set of net operations that allows one access to nets and help to define one's own calculi.

Of course, sets of auxiliary operations have been used before by several authors. We can mention Kotov's merging operation [6], Cartesian product of places and operations of addition and removal of places and transitions [4]. However, all of them were quite simple and did not reduce the complexity. We, however, deliberately defined a complex auxiliary operation, that would essentially simplify our future work. Such an approach would also be useful in the implementation of any net algebra [2]. Once implemented in subroutines, auxiliary operations allow easy implementation of all other operations of Petri net calculus.

## References

1. N.A.Anisimov. An Algebra of Regular Macronets for Formal Specification of Communication Protocols. *Computers and Artificial Intelligence,* **10** (1991), 541–560.
2. N.Anisimov, M.Koutny, On Compositionality and Petri Nets in Protocol Engineering, in: *Protocol Specification, Testing and Verification XV*, P.Dembiński, M.Średniawa eds. (Chapman & Hall, 1996), pp.71-86
3. E.Best, R.Devillers, A.Kiehn, and L.Pomello. Concurrent Bisimulations in Petri Nets. *Acta Informatica,* **28**(1991) 231-261.
4. E.Best, R.Devillers, J.G.Hall. The Box Calculus: a New Causal Algebra with Multi–label Communication. In: *Advances in Petri Nets 1992,* ed.G.Rozenberg, *Lecture Notes in Computer Science,* **609** (Springer–Verlag, 1992) 21–69.
5. R.J.Van Glabbek and F.W.Vaandrager. Petri Net Models for Algebraic Theories of Concurrency. *Lecture Notes in Computer Science,* **259** (Springer–Verlag, 1987) 224–242.
6. V.E.Kotov. An Algebra for Parallelism Based on Petri Nets. *Lecture Notes in Computer Science,* **64** (Springer–Verlag, 1978) 39–55.
7. P.E.Lauer, R.H.Campbel. Formal Semantics of a Class of High-Level Primitives for Coordinating Concurrent Processes. *Acta Informatica,* **5**(1975) 297-332.
8. W.Reisig. Petri Nets: An Introduction. *EATCS Monograph on Theoretical Computer Science* (Springer–Verlag, 1985).
9. G.Rozenberg (ed). Advances in Petri Nets 1992. *Lecture Notes in Computer Science,* **609** (Springer–Verlag, 1992) p.472.

This article was processed using the LaTeX macro package with LLNCS style