FORMAL SPECIFICATION COLUMN
Part 19, January 2003
Hartmut Ehrig

ON THE RELEVANCE OF HIGH-LEVEL NET PROCESSES

HARTMUT EHRIG
Technical University of Berlin
Franklinstraße 28/29, D – 10587 Berlin
E-mail: {ehrig}@cs.tu-berlin.de

1 General Motivation

The notion of nondeterministic and deterministic processes based on occurrence nets is an essential concept to capture the non-sequential truly concurrent behavior of Petri nets. This concept is well-known for elementary nets and safe place-transition nets and has been generalized to other low-level net classes. Let us call a net class low-level if the firing is based on black tokens only, while a high-level net class has colored tokens which are defined as data elements of a suitable data type. The concept of high-level nets is certainly very useful to model more complex communication based systems, because it allows to use an adequate balance between data type and net features. This avoids to represent even basic data types by nets as it is necessary in the case of low-level nets. For high-level nets, however, the standard technique to define processes is to consider them as processes of the low-level net $Flat(N)$ which is obtained from N via the well-known flattening construction. This low-level notion of processes for high-level nets, however, is not really adequate, because the high-level structure using data types is completely lost. For this reason we have introduced in our paper [1] a new notion of high-level net processes for high-level nets which captures the high-level structure. The key notion is a high-level occurrence net K, which generalizes the well-known notion of occurrence nets from low-level to high-level nets. It is important to note that the flattening of high-level occurrence nets and processes in general does not lead to low-level occurrence nets and processes. This effect is due to so called "assignment conflicts" which can occur in high-level occurrence nets. This means that different assignments for the same transitions may lead to forward or backward conflicts in the flattening. In [1] we have given a syntactical characterization of such assignment conflicts. In fact, it would be possible to restrict the notion of high-level processes to those, where the corresponding high-level occurrence net has no assignment conflicts. Another important difference between low-level and high-level occurrence nets and processes is the fact that there is a unique choice for the initial marking of a low-level occurrence net: this is the marking of all input places with one black token each. In the high-level case it does not make any sense to consider only one initial marking of the input places, but a set of initial markings. This corresponds to a set of possible input data for a procedure in a high-level programming language. Similar to a procedure which in general leads only to a partial function, we cannot expect that a high-level process terminates for all initial markings. Even in the case of a finite high-level occurrence net we may have deadlocks, i.e. we may have a firing sequence which cannot be extended to a complete firing sequence. Here we call a firing sequence of an occurrence net complete if each transition is fired exactly once.

Of course, it is an interesting problem to analyse under which conditions there is a complete firing sequence $s : init \xrightarrow{*} \bullet$ for a given initial marking. Similarly it is interesting to know under which conditions we have deadlock-freeness and uniqueness of the final marking on the output places.

In the case of low-level occurrence nets all of these problems have an easy solution. In the next section we summarize these results and give an outline of how to solve these and related problems in the high-level case. A more detailed presentation will be given in our technical report [2].

In the low-level case the notion of processes has been extended already from Petri nets to graph transformation system [3]. Hence our notion of high-level net processes is also relevant in view of graph transformations in the high-level case, i.e. attributed graph transformation systems.

2 From Low-Level to High-Level Net Processes

For low-level Petri nets the notion of nondeterministic and deterministic processes is an essential concept to capture their non-sequential truly concurrent behavior. Especially in the case of elementary net systems and safe place/transition nets this has been worked out in a fully satisfactory way by Rozenberg, Winskel, Nielsen, Goltz, Reisig, Degano, Meseguer, Montanari and other authors [4,5,6,7,8,9,10,11] leading to different notions of deterministic and nondeterministic processes and to a truly concurrent semantics of Petri nets in terms of prime algebraic domains and event structures.

For finite (deterministic) low-level processes the following behavior is well-known or at least folklore: Given the initial marking consisting of all input places of an occurrence net, there is at least one *complete firing sequence*, where each transition fires exactly once and the final marking consists exactly of all output places. Moreover, the occurrence net is concurrently enabled. This means that for each total order of the transitions which is compatible with the causal order of the occurrence net, there is exactly one such complete firing sequence. In addition, also each place of the occurrence net is visited by each complete firing sequence exactly once and each incomplete firing sequence can be extended to a complete firing sequence (deadlock-freeness). Similar properties are valid for infinite occurrence nets and processes, provided that the set of transitions is countably infinite. Of course, such infinite complete firing sequence do not have a final marking which is equal to all output places, but the output places are approximated by the infinite sequence of markings.

The main challenge is now to find out which properties for the behavior of low-level occurrence nets are still valid in the high-level case, or can be obtained under suitable additional assumptions.

In our paper [1] we have defined high-level processes (AHL-processes) for algebraic high-level nets (AHL-nets) and the flattening of AHL-nets as well as AHL-processes already. We have pointed out that due to so called "assignment conflicts" the flattening of an AHL-process is in general not a low-level process.

Moreover, there is no canonical initial marking for AHL-occurrence nets, because in general there are different meaningful markings of the input places. For this reason we study marked AHL-occurrence nets $(K, INIT)$, where K is an AHL-occurrence net and $INIT$ is a set of markings of the input places of K. We say that K is *enabled* for $init \in INIT$ if there is a complete firing sequence $s : init \xrightarrow{*} \bullet$, where completeness means that each transition in K is fired exactly once. In fact, it is an important problem to analyse under which conditions K is enabled for some $init \in INIT$, similar to the problem whether a procedure is well-defined for given input data.

We are able to show in [2] that K is enabled for $init \in INIT$ if and only if there is an *instantiation* L of $(K, init)$. An *instantiation* L of $(K, init)$ is a low-level occurrence net $L \subseteq Flat(K)$ where the net structures of L and K are equal and $init$ is the set of all input places of L. In general, there may be none, one or several instantiations L for $(K, init)$, but we are able to give sufficient conditions for existence and uniqueness.

The next interesting question is which properties for the behavior of low-level occurrence nets are still valid in the high-level case, provided that $(K, init)$ is at least enabled. In fact we are able to show in this case that $(K, init)$ is concurrently enabled, each complete firing sequence $s : init \xrightarrow{*} \bullet$ visits each place and each transition of K exactly once, and terminates for finite K with a final marking on the output places of K, while for countably infinite K the final marking of the output places is approximated. But in general the final marking is not uniquely defined for each $init \in INIT$ and we may have deadlocks. These problems, however, can be avoided if the AHL-occurrence net K has *functional assignments*, i.e. even for different consistent assignments of the variables for a transition the data on the output places of the transition are functional dependent on the data of the input places. Moreover, we can ensure that $(K, init)$ is enabled and deadlock-free if K has *full assignments*, i.e. for each choice of data on the input places of a transition there is at least one consistent assignment matching this choice of data.

Finally let us analyse the relationship between the flattening $Flat(K)$ of an AHL-occurrence net and all the instantiations L of $(K, init)$ for $init \in INIT$. In general $Flat(K)$ may contain places and transitions which do not belong to an instantiation. But if $(K, INIT)$ is *flat-reachable*, i.e. each item of $Flat(K)$ is visited by at least one complete firing sequence $s : init \xrightarrow{*} \bullet$ with $init \in INIT$, then $Flat(K)$ can be represented by the union of all instantiations of $(K, INIT)$ and vice versa. In general, however, different instantiations are not disjoint, but overlap with each other. In spite of this, we are able to characterize under which conditions $Flat(K)$ can be represented as disjoint union of all instantiations of $(K, INIT)$, where for each $init \in INIT$ there is a unique instantiation $L(init)$ for $(K, init)$ in [2]. We show that these conditions are satisfied for the AHL-occurrence net of a high-level process for the dining philosophers. On the other hand, there are also meaningful cases of high-level processes, where the corresponding AHL-occurrence net K has assignment conflicts and $Flat(K)$ is not the union of all instantiations of $(K, INIT)$. Finally let us point out that several possible extensions of high-level processes are discussed in section 4 of our paper [1].

References

1. H. Ehrig, K. Hoffmann, J. Padberg, P. Baldan, and R. Heckel. High-level net processes. In W. Brauer, H. Ehrig, J. Karhumäki, and A. Salomaa, editors, *Formal and Natural Computing*, LNCS 2300, pages 191 – 219. Springer, 2002.
2. H. Ehrig. Behaviour and Instantiation of High-Level Net Processes. Full Technical Version. Technical Report 2003-01, Technical University of Berlin, 2003. To appear.
3. A. Corradini, U. Montanari, and F. Rossi. Graph Processes. *Special Issue of Fundamenta Informaticae*, 26(3,4):241–266, 1996.
4. M. Nielsen, G. Plotkin, and G. Winskel. Petri Nets, Event Structures and Domains, Part 1. *Theoretical Computer Science*, 13:85–108, 1981.

5. U. Goltz and W. Reisig. The Non-Sequential Behaviour of Petri Nets. In *Information and Computation*, volume 57, pages 125–147. Academic Press, 1983.
6. G. Rozenberg. Behaviour of Elementary Net Systems. In W. Brauer, W. Reisig, and G. Rozenberg, editors, *Advances in Petri Nets 1986*, pages 60–94. Springer Verlag Berlin, LNCS 254, 1987.
7. G. Winskel. Petri nets, algebras, morphisms, and compositionality. *Information and Computation*, 72:197–238, 1987.
8. G. Winskel. Event Structures. In W. Brauer, W. Reisig, and G. Rozenberg, editors, *Petri Nets: Applications and Relationships to Other Models of Concurrency*, pages 324 – 392. Springer, LNCS 255, 1988.
9. P. Degano, J. Meseguer, and U. Montanari. Axiomatizing Net Computations and Processes. In *Proc. of LICS'89*, pages 175–185, 1989.
10. J. Engelfriet. Branching Processes of Petri Nets. *Acta Informatica*, 28:575–591, 1991.
11. J. Meseguer, U. Montanari, and V. Sassone. On the Semantics of Place/Transition Petri Nets. *Mathematical Structures in Computer Science*, 7:359–397, 1997.