TIME ANALYSIS OF TIME BASIC NETS USING STRONG TIME SEMANTICS

Attila N. KOVÁCS, Štefan HUDÁK
Department of Computers and Informatics, Faculty of Electrical Engineering and Informatics,
Technical University of Košice, Letná 9, 042 00 Košice, Slovak Republic, tel.: +421 55 602 2518,
e-mail: attila.n.kovacs@tuke.sk, stefan.hudak@tuke.sk

ABSTRACT

Petri nets are a powerful formalism for the specification and verification of concurrent and real-time systems. Real-time systems must be carefully verified because even a small failure in these systems can cause loss of human lives. For this purposes the time reachability analysis of Petri nets with incorporated time issue is more than suitable. The best known and probably the most used Petri nets with incorporated time issue are Time Petri Nets. To perform sophisticated time analyses of systems the Time Basic Nets are far more suitable then Time Petri Nets. This paper presents notions and notations of these nets, their properties and mainly the application of the reachability analysis approach of Time Petri Nets to Time Basic Nets for the purpose of easier time examination.

Keywords: *reachability analysis, real-time systems, time petri nets, time basic nets, time semantics*

1. INTRODUCTION

Nowadays, plenty of systems exist in which time plays a profound role. Such systems are called time-critical systems whose functionalities are defined with respect to time and whose correctness can only be assessed by taking time into consideration. Real-time systems, such as patient monitoring systems, aircraft control systems or traffic control systems, are very common in our everyday life. Even the smallest failure in such a system can cause enormous damages or loss of human lives. That's why these systems must be carefully and precisely verified.

Petri nets are well-suited to model and analyze real-time systems [2] – [19]. Several extensions of Petri nets with incorporated time issue have been already proposed [1, 2, 3, 4, 5, 15]. The most used are Time Petri net, Stochastic Petri nets and Time Basic Nets. Stochastic Petri nets are appropriate for performance evaluation but they do not seem to be useful for modeling and verification of real-time systems where correctness depends on timing bounds. Time Petri nets on the other hand are capable of modeling real-time systems but just a specific part of them. The most suitable extensions of Petri nets are Time Basic Nets, because they have all the advantages of Time Petri nets plus they have a bigger modeling power as it will be demonstrated later.

This paper is organized as follows. In section 2 basic definitions of Time Petri nets are presented. Later on an approach will be presented for finding the reachable markings of this kind of Petri nets. Section 3 deals with Time Basic nets, their definition and two basic time semantics. The approach from section 2 can be easily applied to Time Basic nets. This is shown in section 4 by an example. Section 5 summarizes the achieved goals and we outline further goals of our current research.

2. TIME PETRI NETS

Time Petri Nets (TPNs) are Petri Nets where to each transition a static time interval (SI) is assigned [1] – [4]. The smallest time value of these time intervals is called Static Earliest Firing Time (SEFT) and the largest time value is called Static Latest Firing Time (SLFT). The Static Firing Interval of the transition will be the closed left bounded interval of times between its SEFT and SLFT. For two time intervals $I_1 = [u_1, v_1]$ and $I_2 = [u_2, v_2]$ with $0 \leq u_i \leq v_i \leq +\infty$ we define $I_1 + I_2 = [u_1 + u_2, v_1 + v_2]$ and $I_1 - I_2 = [u_1 - u_2, v_1 - v_2]$.

A *state* in TPNs is a pair $S=(m, I)$ where m is a marking and I is a firing interval set (function) which associates with each enabled transition the time interval in which the transition is allowed to fire.

From the initial state a new state can be reached by a given sequence of firing times corresponding to a firing sequence. Since all time intervals assigned to transitions consist of real numbers the number of reachable states produced by the firing of a single transition is infinite. To handle this problem a *state class* is introduced.

A state class represents all states reachable from the initial state by firing all feasible firing values corresponding to the same firing sequence. More formally, a state class is a pair $C = (m, D)$ in which m is the marking of the class and D is the firing domain of the class, which is defined as the union of the firing domain of all the states in the class. All states in the class have the same marking. A transition t is firable from class $C = (m, D)$ if t is enabled by marking m, and may fire before the minimum of all LFT's related to all enabled transitions. Firing rules in detail can be seen in [3, 4].

2.1. Clock stamped state classes

As we will see later, clock stamped state classes are very helpful in those cases, when we want to find the answer to the question, whether some process or action ends its execution until a specified time [3].

A clock stamped state class (CS-class) is a 3-tuple $C = (M, D, ST)$ where M is a marking; D is a firing domain, i.e., a set of constraints on the values of the time to fire for transitions enabled by current marking M. $D(t_i)$ represents the firing interval of an enabled transition t_i. The left bound of $D(t_i)$ is denoted as $EFT(t_i)$ (earliest firing time) and the right bound of $D(t_i)$ is denoted as $LFT(t_i)$ (latest firing time); ST represents the (global) time interval of the CS-class.

For an enabled transition t_i, $D(t_i)$ gives the global firing time interval of t_i. The word "global" means a relative counting of values to the beginning of the net's execution from the initial CS-class C_0. The initial CS-class is defined as $C_0 = (m_0, D_0, ST_0)$ where m_0 is the initial marking, D_0 contains all the static firing time intervals of the transitions enabled in m_0, and $ST_0 = [0, 0]$. ST represents the global time delay interval in which the net runs from C_0 to current CS-class C.

The following firing rules guide the generation of all reachable CS-classes of a TPN. An enabled transition t_j is said to be firable at CS-class C_k if $EFT_k(t_j) \leq min\{LFT_k(t_i), t_i \in E(C_k)\}$, where $E(C_k)$ is the enabled set at C_k. Let $Fr(C_k)$ be the set of firable transition at CS-class C_k, and let

$$MLFT(C_k) = \min\{LFT_k(t_i), t_i \in Fr(C_k)\} \quad (1)$$

where $MLFT(C_k)$ defines the minimum of latest firing times of all firable transitions in $Fr(C_k)$. The firable transitions in $Fr(C_k)$ can be divided into two groups: a) inherited firable transitions that were firable before C_k is reached and b) new firable transitions that begin firable at C_k. The firing of transition $t_f \in Fr(C_k)$ changes the CS-class to C_{k+1}. If CS-class $C_k = (m_k, D_k, ST_k)$ and $C_{k+1} = (m_{k+1}, D_{k+1}, ST_{k+1})$ then the following steps define transition firing rules:

1. Calculate $D_k(t_f)$, the feasible firing intervals of the firing transition t_f, by shifting right bound of $D(t_f)$ to $MLFT(C_k)$ while keeping its left bound unchanged, i.e.,

$$D_k(t_f) = [EFT_k(t_f), MLFT(C_k)], ST_{k+1} = D_k(t_f) \quad (2)$$

2. The calculation of firing intervals of inherited firable transitions in CS-class C_{k+1} can be done following ways:
   a) Let $m'_{k+1} = m'_k - B(t_f)$ and collect (inherited) firable transitions at m'_{k+1}. Function $B(t_f)$ is responsible for removing tokens from input places of transition t_f.
   b) Let $D_{k+1} = D_k$ and delete from D_{k+1} all entries whose corresponding transitions are disabled by m'_{k+1}.
   c) For each inherited firable transition t_j ($t_j \neq t_f$) at m'_{k+1}, let

$$EFT_{k+1}(t_j) = \max\left(EFT_k(t_j), EFT_k(t_f)\right). \quad (3)$$

3. Calculate the firing intervals of new firable transitions after firing t_f:
   a) Let $m'_{k+1} = m'_k - F(t_f)$ and collect new firable transitions. These transitions are firable in m_{k+1} but not in virtual marking m'_{k+1}. Function $F(t_f)$ is responsible for adding tokens to the output places of transition t_f.
   b) Add into D_{k+1} entries that corresponding new transitions at m_{k+1}: if t_j ($t_j \neq t_f$) is new firable transition at m_{k+1}, then

$$D_{k+1}(t_j) = SI(t_j) + ST_{k+1}. \quad (4)$$

   c) If t_f is still firable at m_{k+1} after its own firing, then

$$D_{k+1}(t_f) = SI(t_f) + ST_{k+1}. \quad (5)$$

Formal proofs and examples for the above mentioned approach can be found in [3]. In section 3 a modified version of this approach will be used to generate the reachable state classes of Time Basic Nets.

3. TIME BASIC NETS

Time Basic nets (TB nets) are a particular case of Time Environment Relationship nets (TER nets) [8]. When we assume that the only types of tokens in TER nets are time values (chronos) then we get TB nets. TB nets have been introduced in [2].

3.1. Definition of Time Basic Nets

A TB net can be characterized as a 6-tuple where *P, T* and *F* are, respectively, the sets of places, transitions, and arcs of nets. The set of places connected with transition *t* by an arc entering *t* is called as the *preset* of *t*. Symbol Θ (a numeric set) is the set of values (timestamps), associated with the tokens. A timestamp represents the time at which the token has been created. In the following, we assume Θ to be the set of non-negative real numbers, i.e., time is assumed to be continuous. Function *tf* associates a function tf_t (called time-function) with each transition *t*. Let *enab* be a tuple of tokens, one for each place in preset of *t*. Function tf_t associates with each tuple *enab* a set of value θ ($\theta \subseteq \Theta$), such that each value in θ is not less than the maximum of the timestamps associated with the tokens belonging to *enab*. At this moment we can define the enabling tuple, enabling time and the firing time.

Given a transition *t* and a marking *m*, let *enab* be a tuple of tokens, one for each input place of transition *t*. If $tf_t(enab)$ is not empty, *enab* is said to be an enabling tuple for transition *t* and the pair *x* = <*enab, t*> is said to be an *enabling*. The triple *y* = <*enab, t, τ*> where <*enab, t*> is an enabling and $\tau \in tf_t(enab)$, is said to be a firing. *τ* is said to be the firing time. The maximum among the timestamps associated with tuple *enab* is the enabling time of the *enabling* <*enab*, *t*>. Firing occurrences, which ultimately produce firing sequences, define the dynamic evolution of the net (its semantics); markings represent the states and transitions represent events of the modeled system.

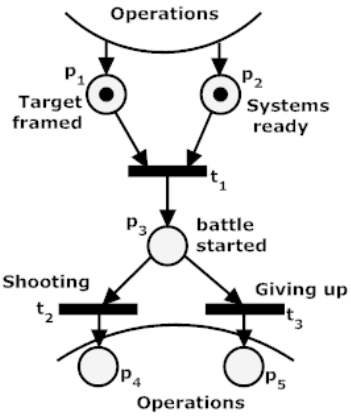
The following statements must hold in TB nets: time never decreases; if the system does not stop, time eventually progresses. More axioms for TB nets can be found in [2].

In TB nets we can distinguish two time semantics: weak and strong time semantics. At this point we will describe the advantages and disadvantages both of them.

3.2. Weak time semantics

The TB net on the Fig. 1 demonstrates a pointing system of a fighter where the pilot can decide whether to shoot a framed target if the pointing system is ready to operate or leave the engaged battle [2].

The presence of a token in place p_1 represents a framed target. Timestamp of the token in this place is the time at which the target was framed. A token in place p_2 represents the readiness of the pointing system for operations.



$$tf_{t_1}(p_1, p_2) = (max(p_1, p_2) \le \tau \le max(p_1, p_2) + 3)$$

$$tf_{t_2}(p_3) = (p_3 \le \tau \le p_3 + 2)$$

$$tf_{t_3}(p_3) = (p_3 \le \tau)$$

Fig. 1 TB net: pointing system of a fighter

Firing the transition t_1 models the start of the battle. A framed target can be shot down if it was not framed too long before the time at which the pointing system becomes ready. This condition is specified by time function tf_{t_1}. If the battle started then the pilot has at most 2 time units to shoot down the framed target. If these 2 time units passes then the pilot losses the chance to shoot down the enemy and can only leave the engaged battle (the pilot must frame the target again to be able to shoot it). These conditions are specified be transitions t_2 and t_3, respectively.

As it was shown in the above example the transition t_2 was not forced to fire. Using weak time semantics the transitions may fire, but they are not forced to do so. The following axioms describe the formal definition of monotonic weak time semantics (MWTS).

Axiom 1: All the times of the firings of an MWTS firing sequence σ must be no less than any of the time stamps of the tokens of m_0.

Axiom 2: All the times of firings of an MWTS sequence σ are monotonically nondecreasing with respect to their occurrence in σ.

Axiom 3: For all $\sigma \in \Theta$ there exists k, $k \ge 0$, such that all firing sequences with at least k firings contain at least one firing whose time is greater than τ, i.e. the number of firings that can occur within a given time interval is bounded.

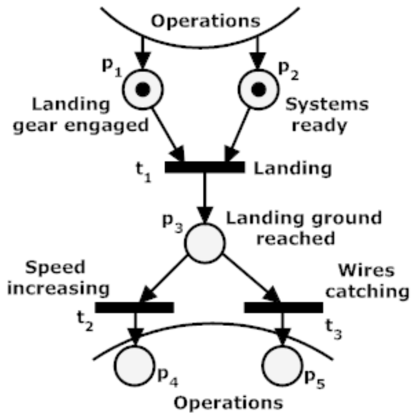
Axiom 1 requires that all firings must occur not earlier than the times associated with the tokens in the initial marking m_0. Axiom 2 describes the monotonicity of the occurrences of firings in the sequence with respect to their firing times. Axioms 1 and 2 capture the fact that time never decreases. Axiom 3 states that if the system does not stop, time eventually progresses, or in other word, there exist no infinitely long firing sequences that take a finite amount of time. This property is often required in real-time system models [6].

3.3. Strong time semantics

Strong time semantics (STS) are used in those cases where some processes must occur in a specified order. This is also the case of TPNs introduced earlier.

At first we will show an example where STS is introduced. Formal definition will be defined later.

TB net on the Fig. 2 shows the landing process on an aircraft carrier. The presence of a token in place p_1 represents the engagement of the landing gears. A token in place p_2 represents the readiness of the systems for landing operations and a token in place p_3 represents the fact that the landing deck was reached by the aircraft. If the aircraft has reached the landing deck there are two possible situations which can happen. The aircraft catches with its landing hook one of the three steel wires that are stretched across the landing deck of the ship. In this situation the aircraft landed safely (token in place p_5). There is also a possibility when the aircraft misses the wires and must get back to the air as soon as possible otherwise it will tumble to the ocean. This situation is modeled by transition t_2.



$$tf_{t_1}(p_1, p_2) = (max(p_1, p_2) \le \tau \le max(p_1, p_2) + 24)$$

$$tf_{t_2}(p_3) = (p_3 \le \tau \le p_3 + 2)$$

$$tf_{t_3}(p_3) = (p_3 \le \tau)$$

Fig. 2 TB net: landing on an aircraft carrier

As it was shown in the previous example the transition t_2 was forced to fire unless disabled by the transition t_3. Using STS the transition must fire until its deadline unless disabled by the firing of some other transition. For STS further axioms must hold in contradiction to MWTS.

Axiom 4: No enabling tuple exists in the initial marking m_0 whose maximum firing time is less than maximum of the timestamps associated with the tokens in m_0. In this case the marking m_0 is called strong initial marking.

Axiom 5: Let σ be a monotonic weak firing sequence of a TB net with a strong initial marking. $\sigma = \langle y_1, y_2, \ldots, y_i, \ldots \rangle$ is a strong firing sequence if and only if for each transition t and for each reachable marking m_i, $1 \le i$, there exists no tuple *enab* enabling transition t in m_i such that the time of firing y_{i+1} is greater than all the firing times of t under tuple *enab*.

As it was mentioned earlier, STS requires enablings to fire within their maximum firing time unless disabled by some other firings occurring before the maximum firing time has expired. The set of firing sequences obtained by STS is a proper subset of the set obtained by MWTS.

3.4. Time interval semantics of Time Basic nets

Until now we used point time semantics to assign a single time value to each token. Instead of time point semantics another time semantic can be used [7].

Interval semantics of TB nets give us the opportunity to assign a time interval (TI) to each token. This time interval specifies the time values in which the tokens can be created. Using TI instead of timestamps gives us a bigger modeling power. Any token (chronos) τ in TI is considered to be a TI $\tau = [\tau_i, \tau_a] \subseteq \mathbb{R}^+$, where $\mathbb{R}^+ = [0, +\infty]$.In TI semantics we replace any enabling tuple $enab = (m(p_1), \dots, m(p_{|preset|}))$ with a corresponding collection of TIs that is called *time interval profile* (TIP). Besides the set operation $\cap, \cup, (\,)^c$ a new operation "+" is defined. For a given constant $c \in \mathbb{R}^+$ and TI $\tau = [\tau_i, \tau_a] \subseteq \mathbb{R}^+$, we have: $c + \tau = [\tau_i + c, \tau_a + c], c \cdot \tau = [\tau_i \cdot c, \tau_a \cdot c]$. For TIs τ' and τ'' we have: $\tau' + \tau'' = \tau \Leftrightarrow \tau = [\tau_i, \tau_a], \quad \tau' = [\tau'_i, \tau'_a], \tau'' = [\tau''_i, \tau''_a], \tau_i = \tau'_i + \tau''_i, \tau_a = \tau'_a + \tau''_a$.

Given TB net N_0 = (P, T, Θ, pre, post, tf, q_0), then tft(enab) has for a given *enab* the unique representation

$$tf_t(enab) = \tau en + tf_t(0) \tag{6}$$

where *τen* is a TI that depends on *enab*, $tf_t(0)$ is a TI that does not depend on *enab*. To put it another way, any t-generated TI τ_t can be represented as a sum of two TIs: τen- the determinate TI that depends on TIP *enab* in question and on *t* (or tf_t) and a constant TI $tf_t(0)$, which depends only on the structure of the TB nets in question. According to the above mentioned unique representation of *enab* some interesting features can be found in [7] and [9].

4. CLOCK STAMPED STATE CLASSES OF TIME BASIC NETS

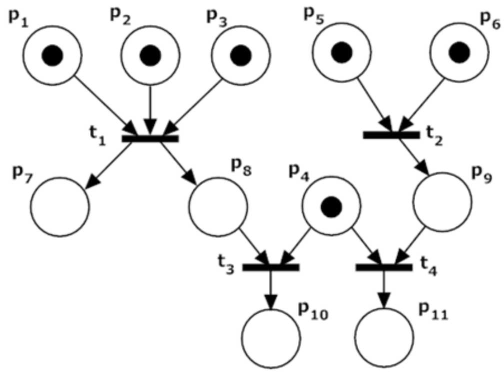
As it was mentioned earlier, reachability problem for time-critical systems is quite different then for ordinary systems. Several researchers tried to solve this crucial problem [7] – [14]. Unfortunately, no general solution of this problem exists for Time Basic Nets.

For the TB net shown on Fig. 3 we will apply the generation rules introduced in section 2. For this TB net a strong time semantic is used.

To use the generation rules we simply replace the names of the places in all time functions with concrete time values. Time functions tf_{t_1} and tf_{t_2} are rewritten as

$$tf_{t_1}(0,0,8) = \max(0,0) + 10 \le \tau \le \max(0,8) + |8-0| + 4 = 10 \le \tau \le 20, \tag{7}$$

$$tf_{t_2}(7,1) = 9 \le \tau \le 1 + 8 + 3 \cdot |7-1| + 4 = 9 \le \tau \le 27.$$



$$m_0(p_1) = \{[0]\}, m_0(p_2) = \{[0]\}, m_0(p_3) = \{[8]\}$$
$$m_0(p_4) = \{[0]\}, m_0(p_5) = \{[7]\}, m_0(p_6) = \{[1]\}$$
$$tf_{t_1}(p_1, p_2, p_3) =$$
$$= \{\tau \mid \max(p_1, p_2) + 10 \le \tau \le \max(p_2, p_3) + |p_3 - p_2| + 4\}$$
$$tf_{t_2}(p_5, p_6) = \{\tau \mid p_6 + 8 \le \tau \le p_6 + 8 + 3 * |p_5 - p_6| + 4\}$$
$$tf_{t_3}(p_4, p_8) = \{\tau \mid \max(p_4, p_8) + 2 \le \tau \le \max(p_4, p_8) + 4\}$$
$$tf_{t_4}(p_4, p_9) = \{\tau \mid \max(p_4, p_9) + 2 \le \tau \le \max(p_4, p_9) + 3\}$$

Fig. 3 Simple TB net with concurrency and synchronisation

The initial CS-class is $C_0 = (m_0, D_0, ST_0)$ where

$$ST_0 = [0,0], m_0 = ([0],[0],[8],[0],[7],[1],0,0,0,0,0), D_0 = \{D_0(t_1) = [10,20], D_0(t_2) = [9,27]\}. \tag{8}$$

From the initial CS-class C_0 the transition t_1 and t_2 is firable. After the firing of transition t_1 from the initial CS-class the next CS-class $C_1 = (m_1, D_1, ST_1)$ can be computed as follows

$$MLFT(C_0) = \min\big(LFT(t_1), LFT(t_2)\big) = \min(20,27) = 20, ST_1 = [EFT_0(t_1), MLFT(C_0)] = [10,20], m_1 = (0,0,0,[0],[7],[1],[10,20],0,0,0), D_1 = \big\{D_1(t_2) = \{\max\big(EFT(t_1), EFT(t_2)\big), LFT(t_2)\} = [10,27]; D_1(t_3) = \max(0,[10,20]) + 2 \le \tau \le \max(0,[10,20]) + 4 = [12,24]\big\}. \tag{9}$$

From the further computation of CS-classes we can create the reachability tree as described on Fig. 4. Questions like "Is the specified marking reachable until 20 time units?" or "Can this situation happen in the time interval [12, 24]?" can be easily answered by this reachabilty tree.

5. CONCLUSIONS

The reachability problem is the most crucial problem in Petri nets. It is closely related to other problems like liveness, deadlock, boundedness or the coverability problem.

For this reason we tried to find the solution for this problem. At first we proposed an approach which solves this problem for TPNs. Later on we introduced TB nets with their different time semantics, such as weak time semantics, strong time semantics and time interval semantics.

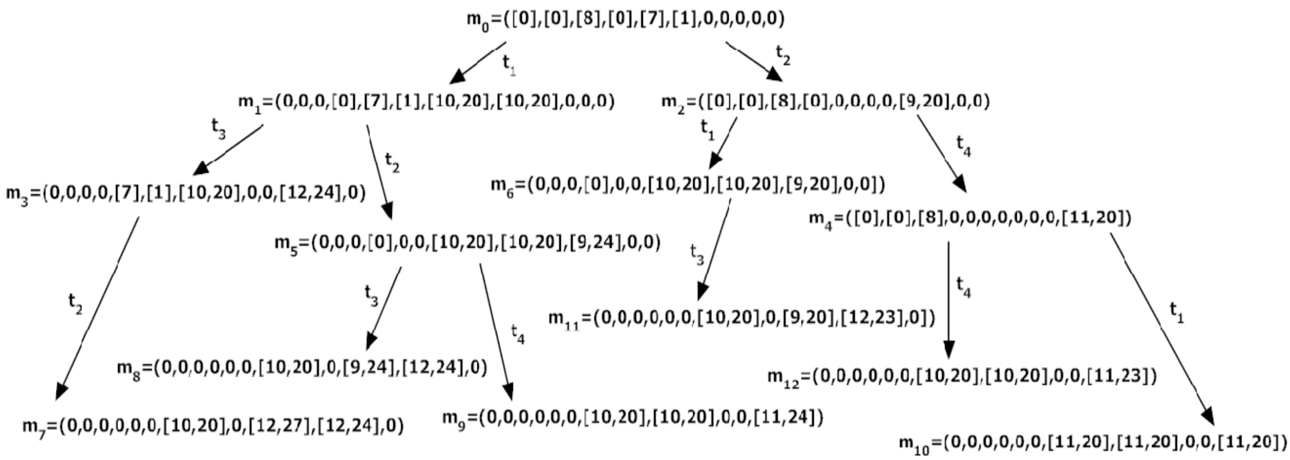


Fig. 4 Reachability tree of the TB net from Fig. 3

For TB nets with STS the approach from section 2 was applied because TB are far more general than TPNs.

Our future work will be focused on the further examination of unbounded TB nets. Currently we are working on a computer tool which will use TB nets to create and verify models of systems.

ACKNOWLEDGMENTS

This work was supported by the Slovak Research and Development Agency under the contract No. APVV-0073-07: "Methods for identification and analysis of security threats in architectures of distributed computer systems and dynamic networks."; and also is supported by VEGA grant project No. 1/0646/09: "Tasks solution for large graphical data processing in the environment of parallel, distributed and network computer systems".

REFERENCES

[1] MARSAN, M. A. – BALBO, G. – CONTE, G. – DONATELLI, S. – FRANCESCHINIS, G.: Modelling with generalized stochastic petri nets, *J. Wiley series in parallel computing*, 1995, 299 pp.

[2] GHEZZI, C. – MANDRIOLI, D. – MORASCA, S. – PEZZÉ, M.: A unified high-level petri net formalism for time critical systems, *IEEE Transaction on Software Engineering*, Vol. 17, No. 2, 1991, pp. 160–172.

[3] WANG, J. – DENG, Y. – XU, G.: Reachability analysis of real-time systems using time petri nets, *IEEE Transation on Systems, Man, and Cybernetics*, Vol. 30, 2000, pp. 725–736.

[4] WANG, J.: Timed petri nets: theory and application, *Kluwer Academic Publisher*, 1998, 282 pp.

[5] GHEZZI, C. – MORASCA, S. – PEZZÉ, M.: Validation timing requirements for time basic net specification, *Journal of Systems and Software*, Vol. 27, 1994, pp. 97–117.

[6] HEINZINGER, T. A. – MANNA, Z. – PNUELI A.: Timed transition systems, *Lecture Notes in Computer Science*, Vol. 600, REX Workshop, 1991, pp. 226–251.

[7] HUDÁK, Š.: Time interval semantics of time basic nets, Proceedings of international conference RSEE'96, Romania, 1996, pp. 1–12.

[8] HUDÁK, Š.: Reachability analysis of systems basad on petri nets, elfa, s.r.o., Košice, 1999, 273 pp.

[9] BAČA, J. – HUDÁK, Š.: De/compositional time reachability analysis, *Proceedings of the 6th International Conference*, Oradea – Felix Spa, Romania, 2001, pp. 60–65.

[10] MAYR, E. W.: An algorithm for the general petri net reachability problem, *Proceedings of the thirteenth annual ACM symposium on theory of computing*, Wisconsin, 1981, pp. 238–246.

[11] KASARAJU, S. R.: Decidability of reachability in vector addition systems, *Proceedings of the thirteenth annual ACM symposium on theory of computing*, California, 1982, pp. 267–281.

[12] BERTHOMIEU, B. – LIME, D. – ROUX, O. H. – VERNADAT, F.: Reachability problems and abstract state spaces for time petri nets with stopwatches, *Discrete Event Dynamic Systems*, Vol. 17, 2007, pp. 133–158.

[13] MADARÁSZ, L. – BUČKO, M. – ANDOGA, R.: Integračné aspekty tvorby a prevádzky systémov CIM (Integration aspects of creation and service of CIM systems), elfa, s.r.o., Košice, 2006, pp. 119–137.

[14] MURATA, T.: Petri Nets: Properties, Analysis and Applications, *Proceedings of the IEEE*, Vol. 77, No. 4, 1989, pp. 541–580.

[15] JENSEN, K.: Coloured Petri Nets: Basic concepts, analysis methods and practical use, Vol. 2, Springer, 1997, pp. 145–168.

[16] ŠIMOŇÁK, S.: Formal methods transformation optimization within the ACP2PETRI tool, Vol. 6, No. 1, *Acta Electrotechnica et Informatica*, 2006, pp. 75–80.

[17] ŠIMOŇÁK, S.: Integrácia formálnych metód s využitím transformácií Petriho sietí a procesných algebier (Integration of formal methods using Petri nets and process algebra transformations), *Dissertation Thesis*, Technical University of Košice, 2003, 112 pp.

[18] TOMÁŠEK, M.: Controlling Communication and Mobility by Types with Behavioral Scheme, *Acta Polytechnica Hungarica*, Vol. 5, No. 4, 2008, pp. 29–40.

[19] TOMÁŠEK, M.: Behavioral Scheme of Mobile Processes, *Journal of Information Control and Management Systems*, Vol. 5, No. 2, 2007, pp. 371–382.

Received December 15, 2009, accepted April 15, 2010

BIOGRAPHIES

Attila N. Kovács was born on 22.8.1984. In 2007 he graduated MSc degree at the Department of Computers and Informatics of the Faculty of Electrical Engineering and Informatics at Technical University in Košice. Since 2007 he is a PhD. student at the Department of Computers and Informatics. His scientific research is focusing on formal methods, especially on Petri nets with incorporated time issue.

Štefan Hudák was born on 25.8.1939. He graduated from Moscow Power Institute (MEI), Faculty of Radioengineering in 1962. He obtained his PhD degree in Technical Cybernetics in 1977 from Slovak Technical University and DrSc degree in Theoretical Informatics in 1997 from T. Schevchenko University, Kiew, Ukraine. He is now Professor of Electrical Engineering and Informatics, Technical University in Košice. His interests are in automata theory, formal description techniques, Petri nets and time-critical systems.