On Structuring Events for IOPT Net Models

Rogério Campos-Rebelo[1,2], Anikó Costa[1,2], and Luís Gomes[1,2]

[1] Universidade Nova de Lisboa, Faculdade de Ciências e Tecnologia, Portugal
[2] UNINOVA – Centro de Tecnologias e Sistemas, Portugal
{rcr,akc,lugo}@uninova.pt

Abstract. This paper presents a proposal for structuring events for system models expressed using IOPT nets (Input-Output Place-Transition Petri nets). Currently, a non-autonomous event within an IOPT model is defined based on change of input signals with respect to a specific threshold, when two consecutive execution steps are considered. New types of events are proposed allowing the definition of an event activated not only by crossing a fixed threshold, but also considering a change in associated signal value on a specific amount (belonging to an interval of values). The concept is further extended allowing the definition of an event based on signal values presented on previous execution steps. The proposal results on a classification of several types of events, namely threshold events, momentum events, impetus events, as well as delayed events and logical events. Usage of these types of events allows improvements in terms of expressiveness and compactness of the resulted model.

Keywords: Petri nets, embedded systems, human-system interaction.

1 Introduction

Since its presentation by Carl Adam Petri in 1962 [1], Petri Nets have suffered many changes and developments, which have led to the growth of their use in various areas. Engineering is one of those areas that take advantage of features such as the ability to model synchronization, concurrency, or conflicts [2], or the simultaneous visualization of the structure and behavior of the system [3]. These features allow Petri nets to capture the dynamics of the system, making them useful in simulation [4].

The inclusions of non-autonomous characteristics in Petri nets models have significantly contribute to increase the impact of their use when compared with other modeling formalisms used for embedded systems design [5]. These non-autonomous characteristics allow the connection of the net with other environments, making them very suitable for modeling of system's controller behavior.

There are a set of non-autonomous Petri net classes applicable for this kind of application areas. Some examples are the Interpreted Petri nets[6], synchronized Petri nets [7] or IOPT nets (Input-Output Place-Transition nets) [7].

After working with these formalisms, it is easy to conclude that whenever a complex interface modeling is needed the model tends to become complex and with

L.M. Camarinha-Matos, S. Tomic, and P. Graça (Eds.): DoCEIS 2013, IFIP AICT 394, pp. 229–238, 2013.

hard interpretation, due to limited number of modeling primitives to explicitly handle human-system interaction.

In this paper a set of proposals to define different types of events are presented addressing the effectiveness of Petri nets modeling for human-system interaction systems, having also application within controllers modeling.

On top of the traditional way to define an event (as a signal trespassing a threshold level), new ways to define an event associated with signal changes are proposed based also on concepts of amplitude and changing speed of variation on signal values.

In this sense, starting with the concept of event definition based on a signal threshold when comparing its value on two consecutive execution steps, the new concepts are proposed considering a larger observation window (not only two consecutive execution steps), as well as reasoning on speed and acceleration of the signal values changes. This led to the definition of impetus events and momentum events, as well to the definition of delayed events and logical events, resulting from the pre-processing of signal variations.

An overview on regular and new events is provided in this paper. A simple potential application for this new modeling attitude uses the model of a joystick controlling a plane simulator as inspiring source, and can be used to informally support the rationale around each new type of event, illustrating modeling benefits for their usage and potential compactness of resulting models. Due to lack of space, it is not possible to present detailed models.

2 Relationship to Internet of Things

In this paper a set of additions to the non-autonomous part of Petri nets models allowing a structuring on events is proposed. These additions directly support a more compact modeling of human-system interactions, permitting also a simpler implementation. With the growth of internet and its opening to the general public, rather than just for those who have expertise in the area, brought new problems to solve like how to develop easier and more enjoyable interfaces to facilitate the interaction of the people with the systems.

Petri nets are in a good position to provide solutions to these problems, as tools are available to model distributed systems and to support a good way to answer to these modeling challenges.

The ideas proposed in this paper leads to an evolution in the modeling of the human-system interactions with Petri nets facilitating the modeling of new interfaces within the “internet of things” concept.

3 IOPT Nets

The IOPT nets, proposed in [8], are a low-level and non-autonomous Petri nets class, extending Place Transition nets and allowing modeling of interaction with the environment through inputs and outputs (events and signals). These signals and

events are associated with the transition firing and places marking, allowing communication with the environment.

The IO extension adds a minimal set of notations to Place-Transition nets that allow the specification of control of discrete event systems. The IOPT nets support specification of:

- External input signals and external input events constraining transition firing, as well as generation of external output events as a result of transition firing (Mealy style).
- External output signals associated with marking of places (Moore style).

As common in modeling controller systems with discrete events, the evolution of the model is possible only in specific moments, called "ticks", and the period between two "ticks" is called step (which is of paramount importance along this paper). All transitions that are enabled and ready will fire at the same step ("maximal step"). In this sense, the synchronized Petri nets paradigm [9] is adopted, where a transition is enabled to fire whenever the marking of its input places provides adequate marking, and is ready to fire whenever associated conditions on signals and events are evaluated as true.

The IOPT nets may also accommodate mechanisms for model structuring [10] supported by decomposition and composition techniques, as in [11].

Currently there are two types of events defined: autonomous events (in the sense that they come directly from environment), and non-autonomous events (in the sense that they are automatically generated after processing of external signals). Only the later ones are considered for this paper. A non-autonomous input event or simply input event from now on, is associated with one input signal and is associated with a threshold level. It is the starting concept and will be formally presented in next section under the classification of “threshold event”.

4 Events Definition

In this section the definitions of the existing threshold events are presented, followed by the definitions of the new types of events. Notation used in this section considers representation of signal X, being x_n the value of signal X at execution step n, and x_{n-1} the value of signal X at previous execution step, and x_{n-p} the value of signal X occurring p execution steps before.

4.1 On Threshold Events

Two types of threshold events are defined: the “up” event and the “down” Event, as proposed in [8].

The event “up”, defined against the definition of the threshold level K, occurs when the value of the associated signal in the last execution step was lower or equal to the level K and the current value is higher than the level K. This event is presented and defined in (1). Similarly, the event “down” occurs when the value of associated

signal in the last execution step was higher than the level K and the current value is higher or equal to the level K. Expression (2) presents the mathematical definition and a representation of this type of signals.

Table 1. Threshold events definitions

	Definition	
	$evx^{+}(k) = (x_n > k) \wedge (x_{n-1} \leq k)$	(**1**)
	$evx^{-}(k) = (x_n \leq k) \wedge (x_{n-1} > k)$	(**2**)
	$evx^{\pm} = evx^{+} \vee evx^{-}$	(**3**)

A third type of event, an event "up or down" can be composed from the previous two types of events, is defined in expression (3) and reflects a situation where a crossing of the threshold level is observed.

4.2 On Momentum Events

The new type of events (the momentum events) is associated with reasoning on the difference of values of signal X along consecutive execution steps (where the concept of acceleration can be associated with this type of events, if X is related with position representation). So, let $dx = x_n - x_{n-1}$. Starting with equations in Table 1 related with threshold events *evx*, the momentum events *evdx* defined in Table 2 are obtained applying the same reasoning to *dx* as to *x* previously.

Table 2. Momentum events definitions

Definition	
$evdx^{+}(k) = (x_n - x_{n-1}) > k \wedge (x_{n-1} - x_{n-2}) \leq k$	(**4**)
$evdx^{-}(k) = (x_n - x_{n-1}) \leq k \wedge (x_{n-1} - x_{n-2}) > k$	(**5**)
$evdx^{\pm}(k) = evdx^{+}(k) \vee evdx^{-}(k)$	(**6**)

Expression (4) presents the definition of momentum event "up". This event detects that the variation of the difference on the signal values becomes higher than a specific value k. It means that in x_{n-1} the momentum value was lower or equal than k and in x_n the value becomes higher than k.

In the same way, expression (5) presents the momentum event "down". Similarly, this event can detect a variation of the difference of the signal X lower than a specific value k. It means that in x_{n-1} the momentum value was higher than k and in x_n the value becomes lower or equal than k.

Similarly to (3), a momentum event "up or down" is defined allowing the detection of the crossing of the k value, without taking into account the direction. This event is composed by event "up" (4) and "down" (5) and is defined in (6).

4.3 On Impetus Events

Impetus events are events that occur with the analysis of the differences in the signal magnitude. Four types of basic events can be defined, according with the signal magnitude and the variation direction:

- evxdpu event defined in (7), associated with a signal having a large positive increase rate, larger than k.
- evxdpd event defined in (8), associated with a signal having a small positive increase rate, smaller than k.
- evxdnu event defined in (9), associated with a signal having a large negative increase rate, larger than k.
- evxdnd event defined in (10), associated with a signal having a small negative increase rate, smaller than k.

On top of these four types of events, it is possible to define composed impetus events, detecting if the variation is between two levels, k_1 and k_2, and not only below or above a value k. Thus, two additional events are defined:

- evxdp event defined in (11), associated with a variation of the signal between two values (k_1, k_2) and composed by evxdpu and evxdpd events, considering a positive variation on the associated signal.
- evxdn event defined in (12), associated with a variation of the signal between two values (k_1, k_2) and composed by evxdnu and evxdnd events, considering a negative variation on the associated signal.

Note that in the events (11) and (12) the values k_1 *and* k_2 can be equal. If this occurs, the generated event is associated with a specific value k on the difference of values of signal X.

Table 3. Impetus events definitions

Definition	
$evxdpu\ (k) = \lvert x_n - x_{n-1}\rvert \geq k \wedge (x_n > x_{n-1})\ ,\ k>0$	(**7**)
$evxdpd\ (k) = \lvert x_n - x_{n-1}\rvert \leq k \wedge (x_n > x_{n-1})\ ,\ k>0$	(**8**)
$evxdnu\ (k) = \lvert x_n - x_{n-1}\rvert \geq k \wedge (x_n < x_{n-1})\ ,\ k>0$	(**9**)
$evxdnd\ (k) = \lvert x_n - x_{n-1}\rvert \leq k \wedge (x_n < x_{n-1})\ ,\ k>0$	(**10**)
$evxdp\ (k_1, k_2) = evxdpu(k_1) \wedge evxdpd(k_2)$	(**11**)
$evxdn\ (k_1, k_2) = evxdnu(k_1) \wedge evxdnd(k_2)$	(**12**)
$evxdmu\ (k) = \lvert x_n - x_{n-1}\rvert \geq k \wedge (x_n \neq x_{n-1})\ ,\ k>0$ $\equiv evxdpu\ (k) \vee evxdnu\ (k)$	(**13**)
$evxdmd\ (k) = \lvert x_n - x_{n-1}\rvert \leq k \wedge (x_n \neq x_{n-1})\ ,\ k>0$ $\equiv evxdpd\ (k) \vee evxdnd\ (k)$	(**14**)
$evxdmn = (x_n = x_{n-1}) \equiv \ulcorner(evxdmu \vee evxdmd)$	(**15**)

As previously defined for threshold and momentum events, two additional events can be defined based on the magnitude of the difference, namely:

- evxdmu event defined in (13) that detects if the signal had a variation higher that a certain value k; this event can be obtained by composition of the two "up" events, evxdpu and evxdnu.
- evxdmd event defined in (14) that detects if the signal had a variation lower that a certain value k; this event can be obtained by composition of the two "down" events, evxdpd and evxdnd.

Note that for these events to occur there must have some variation on the signal, so it is needed that $x_n \neq x_{n-1}$.

Finally, to conclude on impetus events, an event detecting no variation in the signal is defined in expression (15).

4.4 Delayed Events

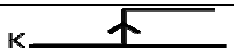
In all cases presented in the previous sub-sections, the analysis of the event is made taking into account the signal value at current execution step and the signal value at immediately prior execution step(s).

In order to accommodate adequate modeling flexibility (particularly for human-system interaction modeling), generation of events may depend on the analysis of signal values not so close in time. In these cases, it can be of interest to consider an event which takes into account a larger number of execution steps in its trigger condition, leading to the comparison of the current value x_n with the value p-execution steps before (x_{n-p}).

This modeling strategy, relying on the analysis of a delayed signal, can be applied to all previously proposed events.

Applying this strategy to threshold events as in Table 1, Table 4 is obtained.

Table 4. Threshold delayed events definitions

	Definition	
	$evx^{+}(k) = (x_n > k) \wedge (x_{n-p} \leq k)$	(**16**)
	$evx^{-}(k) = (x_n \leq k) \wedge (x_{n-p} > k)$	(**17**)
	$evx^{\pm} = evx^{+} \vee evx^{-}$	(**18**)

Likewise, applying the delayed strategy to momentum events as in Table 1, Table 5 is obtained.

Table 5. Momentum delayed events definitions

Definition	
$evdx^{+}(k) = (x_n - x_{n-1}) > k \wedge (x_{n-p+1} - x_{n-p}) \leq k$	(**19**)
$evdx^{-}(k) = (x_n - x_{n-1}) \leq k \wedge (x_{n-p+1} - x_{n-p}) > k$	(**20**)
$evdx^{\pm} = evdx^{+} \vee evdx^{-}$	(**21**)

Finally, applying the same strategy to impetus events as in Table 3, Table 6 is obtained.

Table 6. Impetus delayed events definitions

Definiton	
$evxdpu\ (k) = \lvert x_n - x_{n-p} \rvert \geq k \wedge (x_n > x_{n-p})\ ,\ k>0$	(**22**)
$evxdpd\ (k) = \lvert x_n - x_{n-p} \rvert \leq k \wedge (x_n > x_{n-p})\ ,\ k>0$	(**23**)
$evxdnu\ (k) = \lvert x_n - x_{n-p} \rvert \geq k \wedge (x_n < x_{n-p})\ ,\ k>0$	(**24**)
$evxdnd\ (k) = \lvert x_n - x_{n-p} \rvert \leq k \wedge (x_n < x_{n-p})\ ,\ k>0$	(**25**)
$evxdp\ (k_1, k_2) = evxdpu(k_1) \wedge evxdpd(k_2)$	(**26**)
$evxdn\ (k_1, k_2) = evxdnu(k_1) \wedge evxdnd(k_2)$	(**27**)
$evxdmu\ (k) = \lvert x_n - x_{n-p} \rvert \geq k \wedge (x_n \neq x_{n-p})\ ,\ k>0$ $\equiv evxdpu\ (k) \vee evxdnu\ (k)$	(**28**)
$evxdmd\ (k) = \lvert x_n - x_{n-p} \rvert \leq k \wedge (x_n \neq x_{n-p})\ ,\ k>0$ $\equiv evxdpd\ (k) \vee evxdnd\ (k)$	(**29**)
$evxdmn = (x_n = x_{n-p}) \equiv \ \urcorner(evxdmu \vee evxdmd)$	(**30**)

These events have similar characteristics as the single step events but their usage allows a tuning with time constants associated with the specific modeling situations.

4.5 Logical Events

The next proposed type of complex events is based on the preprocessing of signals evolution history and allows the identification of specific sequences on those signals evolution. They are classified as logical events. Due to space restrictions, only two types of logical events are proposed, having two instances, the "up-down" logical event, and the "down-up" logical event, with and without hysteresis.

These events are of big interest to support production of compact models and rely on the detection of a "cycle" on the level of the signal.

An "up-down" event occurs when the signal value starts below or at level *k* and goes above *k*, returning to level k afterwards. The "up-down" event occurs in the execution step when signal returns or cross level *k*. This event is presented and defined in expression (31)

A "down-up" event occurs when the associated signal starts above level *k*, reaches level *k* or below, and returning above level k afterwards. This "down-up" event is defined in expression (33), and occurs when the signal return above level *k* again.

Delayed logical events can naturally be defined as extensions to the above proposed events, as defined in expression (32) and expression (34), respectively, where *p* determines the pulse width (in terms of number of execution steps). A particular case of interest is observed when *p* could hold an arbitrary value, allowing the detection of a complete cycle on signal values.

Table 7. Logical events definitions

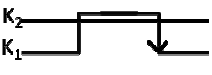
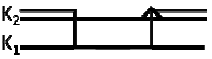
	Definition	
	$evx^{+-}(k) = (x_n \le k) \wedge (x_{n-1} > k) \wedge (x_{n-2} \le k)$	(**31**)
	$evx_p^{+-}(k) = (x_n \le k) \wedge (x_{n-1} > k) \wedge (x_{n-1-p} \le k) \wedge (\forall_{q \in [1,p]} : x_{n-1-q} > k)$	(**32**)
	$evx^{-+}(k) = (x_n > k) \wedge (x_{n-1} \le k) \wedge (x_{n-2} > k)$	(**33**)
	$evx_p^{-+}(k) = (x_n > k) \wedge (x_{n-1} \le k) \wedge (x_{n-1-p} > k) \wedge (\forall_{q \in [1,p]} : x_{n-1-q} \le k)$	(**34**)

A second extension to the referred events, with particular interest for non-Boolean signals, is the definition of a hysteresis on associated signal.

In this case, the events are defined using two levels (k_1 and k_2), corresponding to the hysteresis window.

The "up-down" event with hysteresis, presented in expression (35), is generated when associated signal starts at or below k_1, goes above k_2, and return at or below k_1. Similarly, the "down-up" event with hysteresis, presented in expression (36), is generated when associated signal starts above k_2, goes at or below k_1, and return above k_2.

Table 8. Logical events with hysteresis definitions

	Definition	
	$evx_p^{+-h}(k_1,k_2) = (x_n \le k_1) \wedge (x_{n-1} > k_1) \wedge (x_{n-1-p} \le k_1) \wedge (\forall_{q \in [1,p]} : x_{n-1-q} > k_1) \wedge (\therefore_{l \in [1,p]} : x_{n-1-l} \ge k_2)$	(**35**)
	$evx_p^{-+h}(k_1,k_2) = (x_n > k_2) \wedge (x_{n-1} \le k_2) \wedge (x_{n-1-p} \le k_2) \wedge (\forall_{q \in [1,p]} : x_{n-1-q} \le k_2) \wedge (\therefore_{l \in [1,p]} : x_{n-1-l} \le k_1)$	(**36**)

5 Brief Discussion

The initial type of events, based on a threshold level, only allow modeling of very simple dependencies in terms of analysis of input signals evolution. This means that one event is associated with a specific signal threshold value. All of the other modeling possibilities need to be explicitly represented through sub-models augmented with annotations integrating signal dependencies. This solution allows the modeling of these situations, but leads to a complex model.

Returning to the inspiring example of a joystick controlling an airplane simulator, situations as quick changes in the controls that should lead to different simulation results are difficult to model if only the threshold event is available. On the other hand, the set of proposed new events will adequately support a compact and expressive modeling of the different reactions of the user when using the joystick. In the referred example some examples to of use of each type of event can be found.

For example, whenever the speed at which the joystick is moving is above a certain value, a momentum event can be used considering a K value equals to the value that should not be crossed. When the signal changes from a lower value to a higher value of K, an event is generated.

On the other hand, whenever one wants to identify variations in the joystick speed, it is possible to create an impetus event with the values that wants to detect. The event is generated whenever the speed variation is higher than the set value regardless of the previous value.

Additionally, whenever detection of activation-deactivation sequence of a pushbutton is required, an up-down logical event associated with the pushbutton signal can be used. This event only occurs when the button has just been released.

Any of these previous events could be a delayed event as well. It would be used if the analysis is being done at a rate slower than the analysis rate of the system.

The introduction of the concept of analysis of signal evolution in more than one execution step allows the possibility of introducing some filtering on signal variation.

The proposed logical event paves the way to complex pre-processing of signals, allowing embedded detection of special sequences on signal evolution.

These characteristics make the model easier to produce, understand, and maintain, being easily integrated in automatic code generation tools, as the ones available at http://gres.uninova.pt applicable to IOPT nets.

6 Conclusions and Further Work

The use of these new types of events provides effective modeling capabilities when using IOPT nets models. It allows the creation of smaller models for the same system, making them easier to read and easy to interpret.

Another advantage of the use of such systems lies in the possibility to increase the processing speed of the implemented system using code generation tools.

These event updates are proposed intrinsically matched with IOPT net models, but can also be applied and useful in other classes of Petri nets, both high-level and low-level Petri nets classes.

Other possibility that can be interesting to be further analyzed is the possibility of create events depending on more than one signal, allowing a new level of complex events.

Acknowledgments. This work was partially financed by Portuguese Agency ”FCT - Fundação para a Ciência e a Tecnologia” in the framework of project PEst-OE/EEI/UI0066/2011.

The first author was supported by a Grant from project ARTEMIS/FCT-100261 SIMPLE project, co-funded by Portuguese FCT (Fundação para a Ciência e a Tecnologia) and EU ARTEMIS Joint Undertaking.

References

1. Petri, C.A.: Kommunikation mit Automaten. Technische Hochschule, Darmstadt (1962)
2. Peterson, J.L.: Petri Nets. ACM Comput. Surv. 9(3), 223–252 (1977)
3. Zurawski, R., MengChu, Z.: Petri nets and industrial applications: A tutorial. IEEE Transactions on Industrial Electronics 41(6), 567–583 (1994)
4. Murata, T.: Petri nets: Properties, analysis and applications. Proceedings of the IEEE 77(4), 541–580 (1989)
5. Gomes, L., Barros, J.P., Costa, A.: Modeling Formalisms for Embedded Systems Design. In: Zurawski, R. (ed.) Embedded Systems Handbook, ch. 5, pp. 5–1, 5–34. CRC (2005) ISBN 0849328241
6. Ramirez-Trevino, A., Rivera-Rangel, I., Lopez-Mellado, E.: Observability of discrete event systems modeled by interpreted Petri nets. IEEE Transactions on Robotics and Automation 19(4), 557–565 (2003)
7. Giua, A., Di Cesare, F.: Easy synchronized Petri nets as discrete event models. In: Proceedings of the 29th IEEE Conference on Decision and Control (1990)
8. Gomes, L., Barros, J.P., Costa, A., Nunes, R.: The Input-Output Place-Transition Petri Net Class and Associated Tools. In: 2007 5th IEEE International Conference on Industrial Informatics (2007)
9. David, R., Alla, H.: Petri Nets and Grafcet: Tools for Modeling Discrete Event Systems. Prentice Hall (1992)
10. Gomes, L., Barros, J.P.: On structuring mechanisms for Petri nets based system design. In: Proceedings of the IEEE Conference on Emerging Technologies and Factory Automation, ETFA 2003 (2003)
11. Jensen, K.: A brief introduction to coloured Petri Nets. In: Brinksma, E. (ed.) TACAS 1997. LNCS, vol. 1217, pp. 203–208. Springer, Heidelberg (1997)