Michał GROBELNY

# TRANSFORMATION OF UML 2.x ACTIVITY DIAGRAMS INTO CONTROL INTERPRETED PETRI NETS IN HARDWARE BEHAVIOURAL MODELLING

***ABSTRACT*** *Behavioural specification is one of the most important steps in embedded systems design. This phase play a key role cause in this step the shape and behaviour of the final product is established. The process can be realized with use of various technologies and tools supporting the phase. Two of the technologies supporting behavioural modelling are UML activity diagrams and Petri nets. The paper presents transformation of UML activity diagrams into control interpreted Petri nets. Transformation is targeted for project in which both technologies are used in parallel. The system described in the paper is realized as a bridge between mentioned modelling technologies fully supporting automation of the transformation process. Moreover, the system enables use of additional techniques such as formal verification or hardware description language code generation.*

**Keywords:** *Petri nets, UML Activity Diagrams, transformation, control process, hardware behavioural modelling*

---

**Michał GROBELNY, M.Sc. Eng.**
e-mail: m.grobelny@weit.uz.zgora.pl

Faculty of Electrical Eng., Computer Sc. and Telecomm.,
University of Zielona Gora

PROCEEDINGS OF ELECTROTECHNICAL INSTITUTE, Issue 251, 2011

# 1. INTRODUCTION

Behavioural specification is one of the most important steps in embedded systems design [1, 2, 14]. In this phase system properties and main functionality goals are specified. Moreover, the higher importance of the first parts of project is due to possible consultations with client. Not always client is a highly professional specialist therefore the simple project behavioural description can help both parts of the consultations. UML activity diagrams [12] are easy understandable and commonly used behaviour description technique. UML is nowadays also used in business areas what may help in client-engineer communication. On the other hand Petri nets [6, 7] are also very common (especially in engineering community) and well supported mathematical technique available for a very long time.

The lack of transformation mechanism between control interpreted Petri nets and UML activity diagrams in control process behavioural specification where motivation to research the topic. First phase of investigation was comparison of both specification techniques [11] to localize fundamental elements and mechanism enabling future basis of transformation rules. Both behavioural specification techniques ensure necessary mechanisms for behavioural modelling of discreet systems such as parallelisms, synchronizations or sequencing of tasks. Furthermore, the fact of influence of Petri nets during the design of UML version 2.0 confirms the similarities.

The proposed translation bridge between two description techniques meets better support in hardware behavioural modelling with verification and synthesis possibilities. Transformation from UML 2.x into Petri nets enables ease and efficient connection between two mentioned technologies making hardware behavioural modelling less complex. Presented method allows to combine the advantages of both types of graphical system specification.

The paper is structured as follows. Section 2 introduces to the topic of transformation from UML activity diagrams into control interpreted Petri nets. Section 3 outlines the necessity of hierarchy in behavioural specification of modern systems, presents transformation complexity of hierarchical structures and presents solution of macroplace representation problem in transformation from Petri nets to activity diagrams. In Section 4 proposition of control process behavioural modelling system is presented. Section 5 concludes the paper.

# 2. TRANSFORMATION

A novel transformation method of logic controllers specification from hierarchical UML activity diagrams into hierarchical control interpreted Petri nets has been proposed [9]. There have been described some transformation techniques in literature so far [4, 13]. However, available solutions do not cover all necessary aspects of control process behavioural modelling. Most of them cover UML and Petri nets in software and business behaviour specification. Since UML 2.0 [16], where activity diagrams where designed inspired by Petri nets, the transformation is more natural. The similarity fact also makes it easier to combine both technologies into one powerful engineering engine.

In proposed solution transformation is realised due to strictly defined rules [8]. Actions of activity diagram correspond to transitions of Petri net. It is different from the approach proposed in [4] where actions are transformed to places. Actions of activity diagram are realised one after the other and there is no defined waiting place between two actions. Whenever a process has to wait between two actions, it is some kind of stopped between them without any graphic marking. Therefore there is no graphical representation in UML activity diagram of Petri net's place. Nevertheless, the lack of this element does not close the way to transformation. Input and output signals which are very important elements of control process descriptions are also provided and transformed between two discussed technologies.

# 3. HIERARCHY

System design decomposition and hierarchical representation of systems is a very important issue in large control systems. Due to the fast technological progress in XXI century, especially in computer sciences, the designs every day become larger. This aspect enforces decomposition of complex designs in hierarchical structures concerning autonomous segments of the design. Thus, to fulfil the discussed requirement the transformation has to handle complex hierarchical structures both in UML activity diagrams and Petri nets.

## 3.1. Transformation complexity

Hierarchical transformation rules are a little bit more complex than simple diagrams transformation. In contrast to UML activity diagrams, Petri nets have

two types of hierarchy representation. There are macrotransitions and macroplaces [10] (see fig. 1). Possible hierarchical transformation scenarios are shown in figure 2.

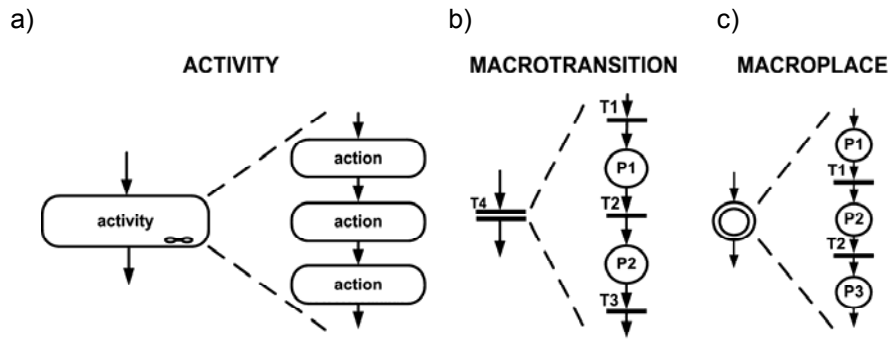


**Fig. 1. Hierarchical structures representation in UML activity diagrams (a) and Petri nets (b), (c)**

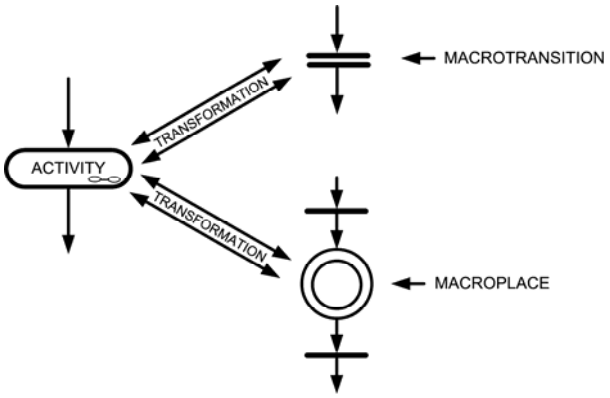


**Fig. 2. Possible transformations of hierarchical structures**

Macroplaces correspond to places of Petri net and are representing static macrostates of modelled system. Macrotransitions correspond to transitions of Petri net and are representing big dynamic change of modelled system which can be divided into elementary actions. A macrotransition can be simply transformed into complex action called in activity diagrams *an activity* (see figure 1 and figure 2). It is one-to-one correspondence and the transformation of the type of hierarchy makes no meaning changes in resulting diagram.

## 3.2. Macroplace problem solution

Due to the fact that there is no representation of place in activity diagrams there is also no simple representation of macroplace. Therefore, author has proposed more complex process realized by swapping of a macroplace into a macrotransition.

Macroplace into macrotransition swapping process is based on expansion of macroplace in existing diagram and surrounding it with two more zero-functionality transitions. Such construction can be then enclosed in a corresponding macrotransition and then simply transformed to complex activity of UML activity diagram. Schematically transformation process of macroplace into macrotransition and next into activity is shown in figure 3.

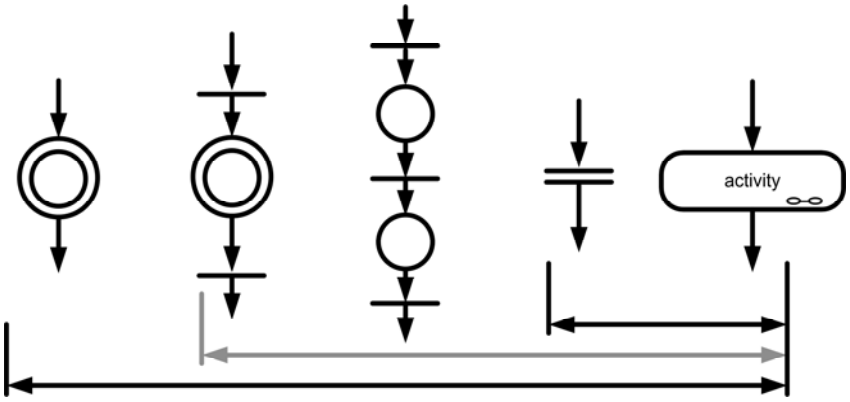


**Fig. 3. Hierarchical structure transformation paths**

Proposed exchange of macroplace into activity of UML activity diagram is a four-stage process. In the first phase macroplace is surrounded by zero-functionality transitions. In the next step macroplace is expanded to regular subnet realisation. In such a way obtained diagram afterwards is bundled in macrotransition. The last fourth step is transformation from macrotransition into UML activity. Process of transformation from macroplace into macrotransition seems to be simple. Backward transformation is considerably more complex. The first question which appears here is when to stop the process. Should it be stopped by receiving macrotransition, macroplace with two additional transitions (second/grey path in figure 3) or just with a macroplace? Activity which was received from macroplace has to be marked to differ it from simple activity received from macrotransition. Additional extra transitions added in macroplace into macrotransition swapping process have to be marked in a special way to remove them in a reverse transformation.

Also the two additional transitions have to be omitted in process behaviour analysis of resulting activity diagram. Both transitions cannot change the meaning of analysed process.

The hierarchical differences of two described technologies make the transformation more complex. Although it is very important to provide all possible mechanisms to enable easy and efficient technology to combine both types of behavioural graphic representation of hardware design.

# 4. CONTROL PROCESS BEHAVIOURAL SPECIFICATION SYSTEM

There has been proposed a system for specifying complex control process behavioural properties. Figure 4 describes basic idea of such a system. The main goal is to ensure full support for designs specified both with control interpreted Petri nets and UML activity diagrams. Designers could use in parallel both techniques in one project. The system can be a bridge between two technologies. Transformation between control interpreted Petri nets and UML activity diagrams will be the basic functionality of described solution. Moreover, usage of commonly used file formats (like e.g. PNML, PNSF3 or XMI) enables preparation of specification diagrams in well-known software environments. UML diagrams could be then prepared in Altova UModel [3] software and modified (after transformation in described system) in WoPeD [15]. Furthermore there will be also a possibility to use VHDL or Verilog code generation to prepare designed project for simulation and synthesis.

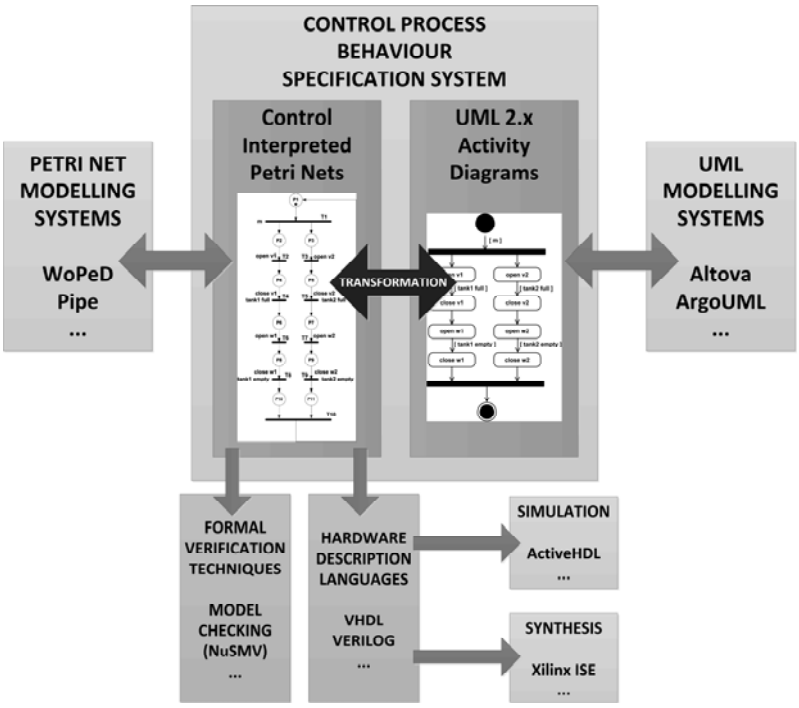


**Fig. 4. Control process designing system**

System provides possibility to export data to formats provided by Petri net formal verification tools. One of the possibilities to formally verify the correctness of designed logic controller behavioural specification could be

model checking [5]. There are mechanisms provided for Petri nets to verify specifications using NuSMV model checker [8]. Specifications prepared in UML activity diagrams could be then transformed to Petri nets and verified. Moreover, projects specified in both techniques in the same time could also use the NuSMV model checking thanks to transformation of UML part into Petri nets.

# 5. CONCLUSIONS

The importance of behavioural modelling phase of logic controller design is very high. Proposed solution is provided to build a bridge between two behavioural modelling technologies. Moreover, there will be a possibility to combine available Petri net analysis techniques with projects constructed using UML activity diagrams. Proposed novel approach extends available transformation techniques [4, 13] making them available in hierarchical control process description. Additionally usage of existing verification techniques may ensure higher quality of project described using UML activity diagrams by transforming them into Petri nets and verify using formal methods e.g. model checking with NuSMV [8]. Moreover, the transformation technique is going to be extended with exception handling mechanisms, which are also very important elements of logic controller design. Furthermore, the author's goal is to deliver fully automatic transformation system, which could be a framework for further improvements of logic controller behavioural modelling. The automation of transformation process is provided to improve usability of described solution.
System enabling transformation of hierarchical diagrams with exception handling will be a fully functional environment for hardware behavioural co-specification using Petri nets and UML activity diagrams with large designs. On the other hand, it will improve communication and coexistence of specialists using in the same project two different specification techniques. On the other hand usage of well-known verification techniques may significantly improve quality of specified control process.

# LITERATURE

1. Adamski M., Karatkevich A., Wegrzyn M. (ed.): Design of embedded control systems, Springer 2005 (USA).
2. Adamski M., Chodan M.: Modelowanie ukladow sterowania dyskretnego z wykorzystaniem sieci SFC, Wydawnictwo Politechniki Zielonogorskiej, 2000 (in Polish).
3. Altova GmbH homepage: http://www.altova.com.

4. Basile F., Chiachio P., Del Grosso D.: Modelling automation systems by UML and Petri Nets, Proceedings of the 9th International Workshop on Disscreet Event Systems, Goteborg, Sweden, 2008, pp. 308-313.
5. Clarke E.M., Grumberg O., Peled D.A.: Model checking, The MIT Press, 1999.
6. David R., Alla H.: Petri Nets & Grafcet. Tools for modeling discrete event systems, Prentice Hall, 1992.
7. Girault C., Valk R.: Petri Nets for Systems Engineering, A Guide to Modelling, Verification and Applications, Springer-Verlag Berlin Heidelberg, 2003.
8. Grobelna I., Grobelny M., Adamski M.: Petri Nets and activity diagrams in logic controller specification – transformation and verification, Mixed Design of Integrated Circuits and Systems – MIXDES 2010, pp. 607-612.
9. Grobelny M., Grobelna I.: Diagramy aktywnosci jezyka UML i sieci Petriego w systemach sterowania binarnego вЂ“ od transformacji do weryfikacji, Pomiary Automatyka Kontrola, nr 10, 2010, pp. 1154-1158.
10. Karatkevich A.: Dynamic Analysis of Petri Net-Based Discrete Systems, Springer-Verlag Berlin Heidelberg, 2007.
11. Grobelny M.: A short comparison between UML Activity Diagrams and Petri Nets in hardware behavioural modelling, X International PHD Workshop – OWD 2008, Conference Archives PTETiS, Vol. 25, pp. 433-436.
12. Object Management Group homepage: http://www.omg.org.
13. Staines T.S.: Intuitive Mapping of UML 2 Activity Diagrams into Fundamental Modeling Concept Petri Net Diagrams and Colored Petri Nets, 15th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems, 2008, pp. 11–200.
14. Gomes L., Barros J.P.: A. Costa: Modeling formalisms for embedded system design, Embedded Systems Handbook, Taylor & Francis Group, LLC, 2006.
15. WoPeD homepage: http://www.woped.org.
16. Wrycza S., Marcinkowski B., Wyrzykowski K.: Język UML 2.0 w modelowaniu systemow informatycznych, Helion, 2005 (in Polish).

*Manuscript submitted 01.06.2011*

# TRANSFORMACJA DIAGRAMÓW AKTYWNOŚCI UML 2.x DO INTERPRETOWANYCH SIECI PETRIEGO STEROWANIA W SPECYFIKACJI BEHAWIORALNEJ SPRZĘTU

Michał GROBELNY

**STRESZCZENIE** *Specyfikacja zachowania systemu jest jednym z kluczowych elementów procesu projektowania sterowników logicznych. Etap ten odrywa ważną rolę ze względu na fakt definiowania*

*kształtu i sposobu zachowania docelowego produktu. Może ona zostać wykonana na wiele sposobów z wykorzystaniem różnych narzędzi wspomagających ten proces. Jednymi z technologii, w których istnieje możliwość opisu zachowania docelowego urządzenia, są diagramy aktywności języka UML i sieci Petriego. Artykuł przedstawia koncepcję transformacji pomiędzy diagramami aktywności języka UML a interpretowanymi sieciami Petriego sterowania. Transformacja dedykowana jest dla projektów, w których inżynierowie wykorzystuję obie wspomniane technologie. Dodatkowo omówiony w artykule system do transformacji ma na celu stworzenie mostu pomiędzy obiema technologiami w pełni automatyzując proces przemieszczania się pomiędzy nimi. Umożliwia on także wykorzystanie dodatkowych narzędzi wspomagających proces projektowania, takich jak formalna weryfikacji czy generowanie kodu w językach opisu sprzętu.*

---

**Michał GROBELNY** received in July 2007 both his M.Sc. in computer science from University of Zielona Góra and Diplom-Informatiker from Fachhochschule Giessen-Friedberg (Germany). Currently he is a Ph.D. student at the Faculty of Electrical Engineering, Computer Science and Telecommunication at the University of Zielona Góra. His current research interest is focused on designing microcontroller systems and using UML and Petri nets for hardware specification. He is currently also working at a company designing vehicle monitoring systems with use of GPS and GSM systems. He is a member of Polish Information Processing Society.