Integrated Approach to Modelling and Analysis using RTCP-nets*

Marcin Szpyrka[1] and Tomasz Szmuc[1]

Institute of Automatics,
AGH University of Science and Technology,
Al. Mickiewicza 30, 30-059 Kraków, Poland
mszpyrka@agh.edu.pl, tsz@agh.edu.pl

Abstract. RTCP-nets are a subclass of coloured Petri nets formed in order to support specification, design, validation, and verification of embedded systems. The advantages of the nets are directly visible in rapid modelling of the so-called rule-based control systems that are widely applied. A method of embedded systems' modelling based on RTCP-nets has been presented in the paper. The formalism is supported by software tool called *Adder*. This tool provides an integrated environment supporting formal specification and design of embedded systems.

1 Introduction

Correctness of a *real time system* is a difficult task due to a high concurrency level and additional time requirements [1]. These features cause, that applications of formal modelling and verification of such systems could be reasonable, especially when the use of these methods is supported by the corresponding software tools. A wide class of real time systems perform on the basis of a set of rules, which are used to compute outputs in response to current state of inputs that are monitored in such system environment. This set of rules specified in the analysis phase as functional requirements may be formally described, and then incorporated into the system model.

A large number of formalisms has been proposed for real-time systems ([1], [2]), e.g. process algebras, Petri nets, temporal\real-time logics, timed automata. Petri nets are one of the most widespread formal methods used in software engineering. The presented approach is based on a subclass of coloured Petri nets (CP-nets [3]), the so-called RTCP-nets. RTCP-nets ([6]) are an adaptation of CP-nets in order to make modelling and verification of real-time systems' easier and more efficient. The main features of RTCP-nets in comparison to timed CP-nets are presented below.

- A priority value is attached to each transition.
- A set of arcs is defined as a relation (multiple arcs are not allowed). Two expressions are attached to each arc: a weight expression and a time expression. For any arc, each evaluation of the arc weight expression must yield a single token belonging to the corresponding type (colour); and each evaluation of the arc time expression must yield a non-negative rational value.

* Research supported from a KBN Research Project No.: 4 T11C 035 24

– Time stamps are attached to places instead of tokens. Any positive value of a time stamp describes how long a token in the corresponding place will be inaccessible for any transition. It is possible to specify how old a token should be so that a transition may consume it.

Hierarchical RTCP-nets are based on the construct used for hierarchical CP-nets. Substitution transitions and fusion places ([3]) are used to combine pages but they are a mere designing convenience. A special *canonical form* of hierarchical RTCP-nets has been defined to speed up and facilitate drawing of the models ([6]). RTCP-nets in canonical form are composed of four types of subnets with precisely defined structures: *Primary place pages* are used to represent active objects (i.e. objects performing actions) and their activities. *Primary transition pages* are oriented towards activities' presentation and are second level pages. *Linking pages* belong to the functional level of a model. They are used (if necessary) to represent an algorithm that describes an activity in detail. Moreover, a linking page is used as an interface for gluing the corresponding D-net into a model. *D-nets* are used to represent rule-based systems in the Petri net form.

CASE tools for RTCP-nets called *Adder Tools* are being developed at AGH University of Science and Technology in Kraków. The tools are a free software covered by the GNU Library General Public License. It has been implemented in the GNU/Linux environment by the use of the Qt Open Source Edition. The software can be compiled and run in Mac OS X or Windows systems. *Adder Tools* contain: *Designer* – for design and verification of rule-based systems, *Editor* – for design of RTCP-nets, and *Simulator* – for simulation of RTCP-nets. *Adder Tools home page*, hosting information about current status of the project, is located at *http://adder.ia.agh.edu.pl*.

2 System design process

This paper presents a development process, based on RTCP-nets, that can be used for modelling of real-time embedded systems. A general scheme of RTCP-nets design process is illustrated in Fig. 1. The approach is consistent with general rules of object system development but some elements of structural modelling have been additionally included. The main stages of RTCP-nets design process are given bellow.

1. The first stage consists of system requirements' definition and verification. For the considered control systems, requirements are usually described by the use of decision tables (see [7]).
2. System decomposition is the first step of a model development. It starts with distinguishing objects that constitute the system and with defining attributes that describe their features. Objects are divided into active and passive ones. Construction of primary place pages for active objects ends this development stage.
3. The next stage deals with description of model dynamic that is especially important for reactive systems. Primary transition pages are constructed at this stage. After completion of this stage, RTCP-net represents all elements (objects) that constitute the modelled system and all its activities.
4. The last stage is related to development of functional aspects of the system. Linking pages and D-nets (if necessary) are used for this purpose.

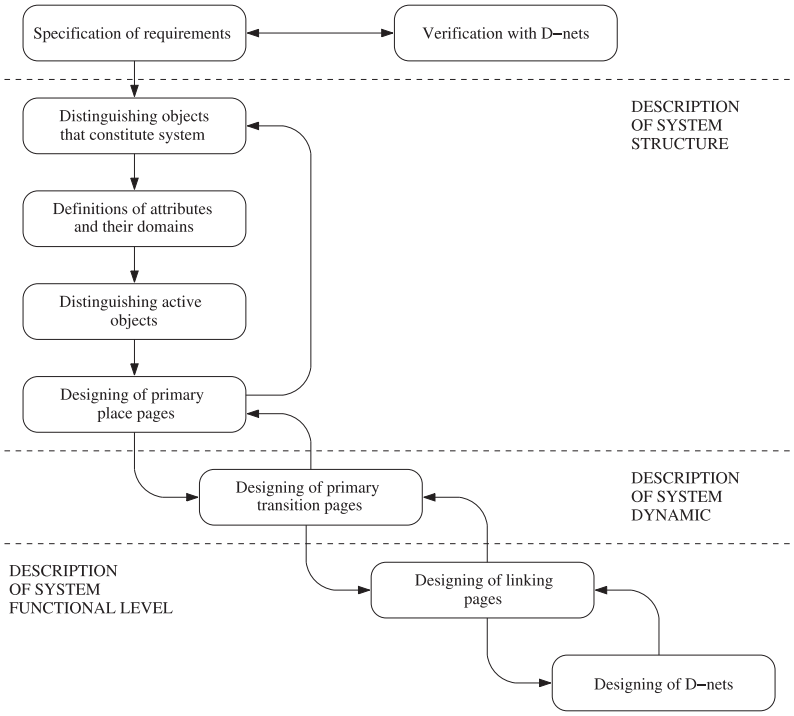

Fig. 1. The RTCP-net development process model

3 Case study

Let consider an office heating/air conditioning control system. A driver turns a heater and an air conditioner on/off to keep the office temperature between $T - 2°C$ and $T + 2°C$, where T is the required temperature. T is set depending on the particular day (D), month (M), and hour (H). All relationships between these attributes are represented in the form of a rule-based system presented in Fig. 2 (the table is based on the example presented in [4]). D, M, and H are conditional, while T is a decision attribute. Each row of the table represents a generalized decision rule ([7]). For example, the first rule means: *If it is any day of the week, the month is: December or January or February, and the time is before 9 or after 17, then the required temperature is equal to* $14°C$.

Any rule-based system is useful when it satisfies certain formal requirements. For intuition, a decision table is considered to be *complete* if for any possible input situation at least one rule can produce a decision. A decision table is *deterministic* if no two different rules can produce different results for the same input situation. The last property means that any dependent (non-necessary) rules were removed. *Adder Designer* enables users to verify a decision table properties automatically.

Design of an RTCP-net starts with distinguishing objects that constitute the modelled system. Any object is represented by a place. For each object, a list of its attributes and their types are defined. The Cartesian product of the defined types specifies the corresponding place type. The following active objects can be distinguished in the

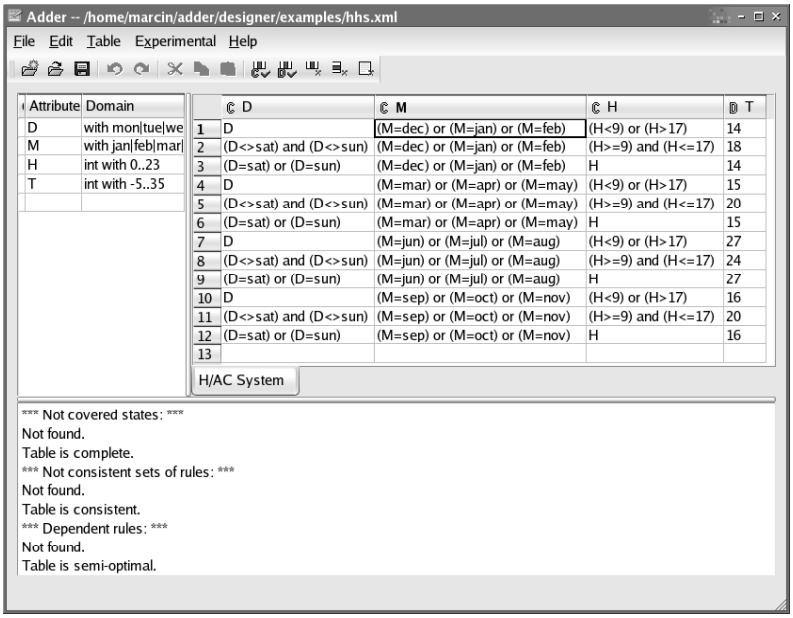

Fig. 2. Example of *Adder Designer* session

considered system: *Driver, AirCond, Heater, Sensor, Office*. The last object represents the system environment. Examples of definitions of some colours (types) are as follows:
color Temperature = int with -5..35;
color Temperature2 = with normal — heat — cold;
color State2 = with ok — failure;
color DriverState = product Temperature2 * State2;

Primary place pages are used to represent active objects. Any such page is composed of one place representing the object and one transition for each object activity. Such a page for the *Driver* is presented in Fig. 3 a). Other primary place pages are designed in a similar way. Transitions placed on primary place pages are usually substitution ones. For each of them a *primary transition page* is drawn. Such a page contains all the places, the values of which are necessary to execute the activity. A primary transition page for the *Measurement* activity is shown in Fig. 3 b). The place *Timer1* is used to guarantee, that the temperature will be measured every 15 seconds. If expressions of arcs surrounding such a transition are not enough to describe the activity, a linking page (and D-net if necessary) must be constructed. D-net form of the decision table presented in Fig. 2 is shown in Fig. 3 c). To include such a D-net into the model a linking page should be constructed. Such a page is used to gather all necessary information for the D-net and to distribute the results of the D-net activity. Connections among all pages are represented by the use of a page hierarchy graph (see Fig. 4).

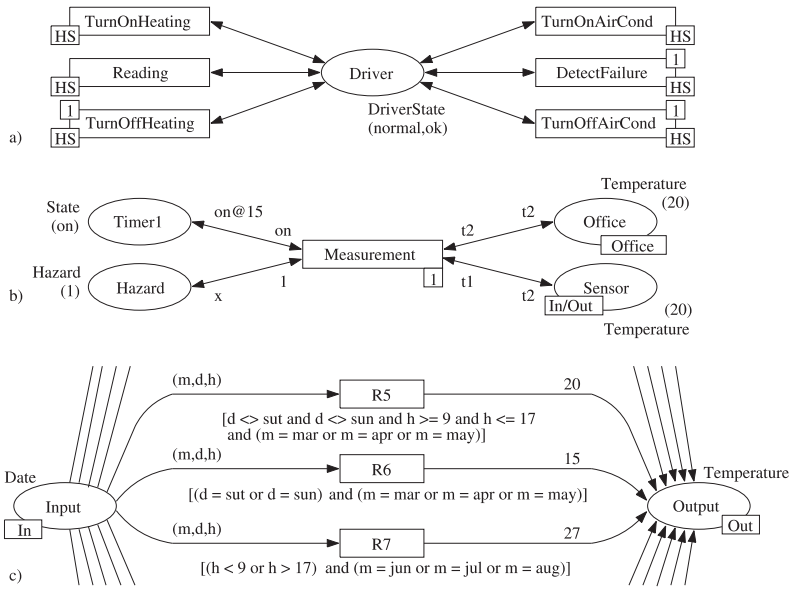

Fig. 3. a) *Driver* primary place page, b) *Measurement* primary transition page, c) Part of D-net

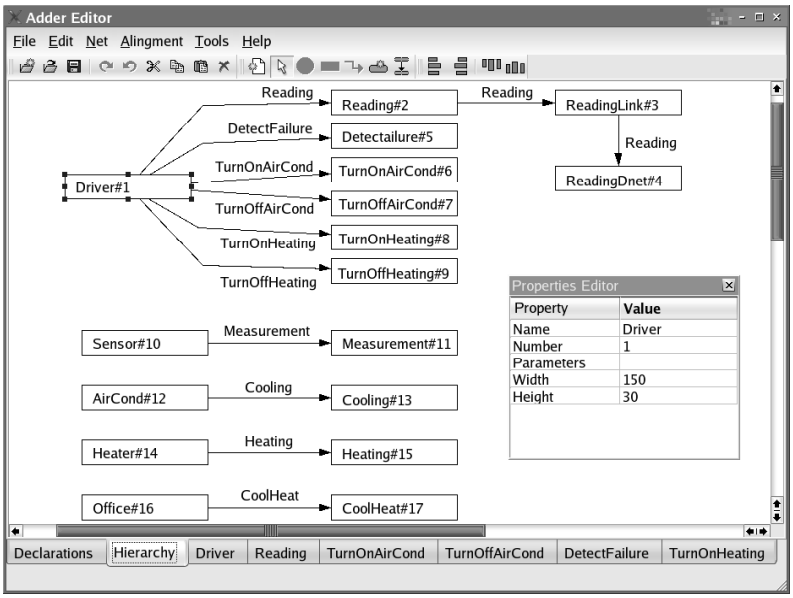

Fig. 4. Example of *Adder Editor* session

4 Model verification

Verification of RTCP-nets may be carried out in two ways. First of all, simulation is used to check how a model works. Adder Simulator supports both interactive and automatic simulation of an RTCP-net. The tool generates a textual report.

Formal verification of RTCP-nets is based on coverability graphs. Two states are considered to *cover* each other, if both have the same markings and the same *level* of tokens accessibility, i.e. in both states, for each place the tokens are already accessible or we have to wait the same number of time units to remove them. A coverability relation is defined on the set of all states and the coverability graph contains only one node for each equivalence class of the relation. If the set of reachable markings of an RTCP-net is finite and each type is finite, then it is possible to construct a finite coverability graph representing the set of all reachable states regardless of the fact the set is finite or infinite ([5]). The coverability graph for an RTCP-net provides similar capabilities of analysis of the net properties as the full reachability graph. For example, the following conclusions are resulted from the graph analysis:

- The heater and air conditioner never work at the same time.
- The heater never works if the temperature is *heat*.
- The air conditioner never works if the temperature is *cold*.
- If the temperature changes from *heat* to *normal*, the air conditioner is immediately turned off.
- The air conditioner (heater) works at least $30s$ and can work incessantly.

5 Summary

A proposal of a method for real-time systems modelling has been presented in the paper. It uses two subclasses of CP-nets – D-nets and RTCP-nets. D-nets are used for the formal definition and verification of system specification, and then are built into system model in the design phase. RTCP-nets are used as the modelling language in the design phase. The presented approach is supported by the so-called *Adder Tools*. The two kinds of Petri nets and the related tools constitute basis for the proposed method for development of real-time systems.

References

1. Cheng A. M. K. *Real-time Systems. Scheduling, Analysis, and Verification*. Wiley Interscience, New Jersey, 2002.
2. Heitmeyer, C., Mandrioli, D. (Eds.) *Formal Methods for Real-Time Computing*. Jonh Wiley & Sons, Chichester, 1996.
3. Jensen K. *Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use*, volume 1-3. Springer-Verlag, 1992-1997.
4. Ligęza A. *Logical Foundations of Rule-Based Systems*. Springer-Verlag, Berlin, 2006.
5. Szpyrka M. Analysis of RTCP-nets with reachability graphs. *Fundamenta Informaticae*, 2006 (to appear).
6. Szpyrka M. Fast and flexible modelling of real-time systems with RTCP-nets. *Computer Science*, 6:81–94, 2004.
7. Szpyrka M., Szmuc T. D-nets – Petri net form of rule-based systems. *Foundations of Computing and Decision Sciences*, 31(2):157-167, 2006.