IFAC MCPL 2007
The 4th International Federation of Automatic Control Conference on Management and Control of Production and Logistics
September 27-30, Sibiu - Romania

A COLOURED PETRI NET APPROACH FOR THE MANAGEMENT OF OPERATING MODES IN DISCRETE EVENT SYSTEMS

Belhassen Zouari[1], Riadh Frefita[1] and Eric Niel[2]

[1] *LIP2 Lab, Faculté des Sciences de Tunis, 2092, Tunisia*
belhassen.zouari@fst.rnu.tn
[2] *Ampère, INSA-Lyon, F 69621, France*

Abstract: This paper addresses the problem of operating mode management in Discrete Event Systems, which is studied within the general framework of dependable systems for which robustness is a key requirement. Specification of system behaviour is ensured through the use of Coloured Petri nets (CP nets) on the basis of a multi-model design. Particular transitions are specified as switching events that allow the system to switch to a different operating mode, when exceptional events occur, such a failure, loss of a resource, or failed resource recovery. The adopted methodology considers the aspects of mode activation/deactivation, starting state and handling of resource states common to multiple operating modes. An algorithm is provided to generate a global CP net permitting easy implementation of existing methods and tools.

Keywords: Discrete event systems, Operating modes, Reconfiguration, Coloured Petri nets, Supervisory control, Robustness, Formal analysis.

1. INTRODUCTION

Widespread expansion of highly automated systems in industry, transport, telecommunications and domestic appliances has increased the need for application reliability and robustness in the area concerned. This requirement is even more significant for embedded and/or critical systems, in which unforeseen failures may have serious consequences. Methods offering advantageous solutions to safe control and failure reactivity include those based distinctively on *operating mode management*. This technology aims to maintain acceptable system operation despite failures occurring. Acceptable operation level may be determined from risk analysis and must permit other resource reconfigurations. Such failures are typically total or partial loss of system components or resources. Under these circumstances, the system will switch to degraded mode, in which it ensures minimum operation under a revised control policy.

Extensive research has considered operating mode management for Discrete Event Systems (DES) (Hamani *et al.*, 2004; Asarin *et al.*, 2000; Charbonnaud *et al.*, 2001 ; Nourelfath and Niel, 2004; Kamach *et al.*, 2003). In (Hamani *et al.*, 2004), characterisation and mode switching are defined but without formal validation of the switching mechanism and deadlock avoidance. Within the scope of hybrid systems, other studies (Asarin *et al.*, 2000) have proposed methodology for synthesizing switching controllers and have defined a switching condition to prevent unwanted states. Musgrave, *et al.* (1997) have introduced real-time controller accommodation. In this approach, control reconfiguration involves switching within a set of controllers in relation to the appropriate model. Real-time accommodation has been applied to industrial systems such as a reusable rocket engine (Musgrave *et al.*, 1997) and control reconfiguration for high-speed ships (Rauch, 1995). Charbonnaud, *et al.* (2001) have introduced a multi-controller structure featuring operating mode detection and an accommodation loop. In (Kamach, *et al.*, 2003), a multi-model approach has been adopted for managing operating modes. An automaton is associated with each operating mode and a switching mechanism inducing trace memorisation is defined.

This paper addresses the problem of operating mode management in Discrete Event Systems. In common with (Kamach *et al.*, 2003), we adopt a multi-model approach, in which a different model is associated with each operating mode. This allows definition of separate behaviour for each model and different control strategy for each operating mode. In contrast, each model in our approach is a behaviour description represented by a Coloured Petri net (CP net)(Jensen 1997) rather than an automaton representing the model state graph (Kamach, *et al.*, 2003). Each CP net is assumed to represent the behaviour of an operating mode under specific control based on supervisory control theory (Ramadge and Wonham, 1989). Numerous Petri net-based supervisory control studies (Krogh, 1987; Giua and DiCesare, 1994; Ghaffari, *et al.*, 2002; Zouari, *et al.* 2004, 2005; Fanti, *et al.*, 2006; etc.) have led to generation of a controlled model itself described by a Petri net. Throughout this work, the common idea is that the generated controller consists of additional

places that are accurately connected to some controllable transitions to play the role of appropriate constraints that restrict the plant model behaviour to only desired states.

One operating mode management feature is that it allows a system to switch from one operating mode to another one, when exceptional events occur. Such events may be failures, resource losses or failed resource recoveries. In our approach, switching events are modelled by particular transitions that disable the current operating mode and enable a new one. The objective is to allow formal specification and analysis of the whole system such as deadlock/livelock freeness and reachability properties. The adopted methodology needs to consider important aspects involving mode activation/deactivation, target operating mode starting state and handling of resource states common to multiple operating modes.

An algorithm is introduced to generate a global CP-net that allowing easy use of the existing analysis methods and tools.

The remainder of the paper is organised as follows: Section 2 presents the adopted multi-model approach and related hypothesis. Section 3 introduces the CP-net based methodology for the management of multi-model operating mode systems. Section 4 presents an experimentation of the method.

2. A MULTI-MODEL BASED APPROACH FOR OPERATING MODE MANAGEMENT

In this section, we present the hypothesis and concepts adopted for the management of operating modes in a DES. The multi-model approach involves representing a complex system by several simple models. Each one is a partial description of the system behaviour in an activated operating mode. Typically, a system involves a nominal mode and a finite set of degraded modes. The set of all operating modes is denoted $O=\{om_1,om_2,...,om_{|O|}\}$, where $|O|>1$ and where om_1 is assumed to be, by convention, the initially activated mode (i.e. the nominal mode).

For each operating mode om_i, we associate a CP-net $<P_i,T_i,K_i,D_i,W_i^-,W_i^+,\Phi_i,M_{0,i}>$[1] describing the system behaviour under its proper control.

2.1 Supervisory Control of operating modes

As with each operating mode is associated a CP-net, we assume the control constraints are ensured by appropriate added CP-net places and arcs. This can be obtained by applying known Petri net based techniques of controller synthesis. We can cite (Zouari, *et al.*, 2004; Ezpeleta, *et al.*, 2002) where the control synthesis is achieved by *structural methods*, as for instance, those based on place invariant or siphon computation. These ones allow avoiding state space exploration. On the other hand, several techniques (Krogh, 1987; Ghaffari, *et al.*, 2002; Zouari, *et al.*, 2004, 2005; etc.) are behavioural and compute the admissibility, as for instance, those using Theory of regions. The common feature of the majority of these techniques is: from a Petri net description of the plant model, we apply a technique to generate additional places and arcs that are connected to the original structure to provide the controlled model.

2.2 Proper component vs Common component

We consider the following assumption: a process is made up of several components and not all components are used in every operating mode. For instance, from a nominal mode and following a loss of a component, the system switches to a degraded mode in which some of the previously used resources are maintained. Hence, a component may be a power unit, a machine, an AGV or a simple buffer. In our approach, a component is a *sub-structure* or a part of a CP-net. It is an *abstract* view that describes the behaviour of a *sub-system* in a given operating mode. It may correspond to a physical component, to a part of it or a collection of physical ones. In all cases, the definition of a CP-net component is a designer point of view.

An important feature related to the concept of component is whether it is *commonly* used by several operating modes. Hence, if a component is used in more than one operating mode, it is called a *common component*; otherwise it is a *proper component*.

In (Kamach *et al.*, 2003), a tracking concept is introduced so as to maintain a *trace* of events that have occurred for the common components. A recording mechanism is used in order to memorize history of events. This is necessary to determine the starting states when switching from a mode to another. This difficulty is due to the fact that evolving states of common components is not taken into account by the used automata model.

In our approach, a common component consists of a subset of places and transitions, from a given mode CP-net, which is *re-used* in another mode CP-net. The evolving of *marking* of common component places is equivalent to *traces* previously quoted. In this work, we do not care about the follow-up of common components as these ones are implicitly handled by the reachability marking graph (also called occurrence graph) of Petri nets.

From the point of view of the implemented tool, a common component is defined when the designer uses the same name of places or transitions in different sheets (i.e. different mode CP-nets). This is facilitated by interface graphical techniques as sub-structure selection, copy and paste operations.

2.3 Switching events

A switching event occurs when the component functioning conditions do not meet any more the requirements of the current operating mode. A component failure or a recovery from a failure corresponds to switching events. A switching event leads to quitting the current operating mode and entering another one.

In our approach, a switching event is represented by a user defined CP-net transition in a given mode. Similarly to any other CP-net transition, a switching transition changes the marking of the current mode CP-net. However, a switching transition holds additional information on the target operating mode.

[1] In Section 3, formal definition of CP-nets and related notions is presented.