Formal Modeling the Low-level Interaction Between Holons

VALENTIN VLAD, ADRIAN GRAUR, CALIN CIUFUDEAN, CONSTANTIN FILOTE
Electrical Engineering and Computer Science Department
University of Suceava
Universitati 13, RO-720229, Suceava
ROMANIA
{vladv / adriang / calin / filote}@ eed.usv.ro

Abstract: This paper is focused on formal modeling of low-level interaction between holons. Using as formal methods the Place/Transition nets and Net Condition/Event systems (NCESs) we have developed some models for a communication and control mechanism between three production facilities sharing a common resource. The resource is represented by a shared route area on which each production facility delivers its ready pallets and the control mechanism has to assure the correct access to this route in order to avoid the collision of pallets.

Key-Words: Formal modeling, Petri nets, Net Condition/Event systems, Holon.

1 Introduction
Holonic manufacturing systems (HMS) represent a novel paradigm based on concepts developed by Arthur Koestler to living organisms and social organizations [1]. The basic elements of a HMS are called holons, a term coined from the Greek words "holos" (the whole) and "on" (a particle). This combination reflects the fact that a holon may act both as an autonomous entity (the whole attribute) and as a part of a larger whole (the particle attribute).

As it is shown in [2] at least three separate information processing levels can be distinguished in a factory entirely based on holonic principles:

- real-time control, tightly connected with the physical level of manufacturing equipment;
- production planning and scheduling;
- supply chain management, integrating the particular plant with its external entities (suppliers, customers, partners, etc.).

Based on these levels, there are several architectures proposed for holons that combine both real-time control technologies (like the IEC 61499 standard [12]) and Multi-Agent Systems (MAS) technologies (for reasoning activities and high-level communication). Such an architecture is presented in Fig.1.

Fig. 1. Holonic architecture combining both real-time and MAS technologies (from [2])

In a holon equipped with both a lower level real-time component and a higher-level intelligent component, three communication channels should be considered [3]:

- intra-holon communication between the function block part and the intelligent component;
- inter-holon communication that is aimed at communication between the intelligent components of multiple holons;
- a direct communication channel between real-time components of neighboring holons.

Our paper is focused on some aspects of formal modeling and verification of low-level interaction...
between holons, taking as a practical example the control and communication mechanism between three production facilities sharing a common resource. This work is inspired from [4], where the authors introduced a Petri net based approach for defining robust and reliable communication between the high and low-level sub-holons within a device holon. In this paper we developed the formal modeling example by means of two kinds of Petri nets: Place/Transition Petri net, and Net Condition/Event Systems (NCESs), as a more effective modeling method.

The outline of the paper is as follows: the next chapter states theoretical background of the NCES formalism, the third chapter proposes a working scenario for the theoretical issues discussed above, the fourth chapter gives the formal models for the working scenario, and finally, in the last part several conclusions are given.

2 Net Condition/Event Systems

For computer science formal verification provides the correctness of programs by using mathematical methods and models. Unlike testing via simulation the formal verification can explore complete set of system’s state space and mathematically prove that no undesirable or dangerous behavior occurs [6].

Formal verification is applied to a previously built model using an appropriate finite-state or hybrid formalism, e.g. finite-state machines, Petri nets, etc. Petri nets are widely used in formal modeling as a method for synthesizing and specifying control programs [4]. There are many different kinds of Petri nets, each one having its advantages and limitations and being more appropriate for a specific domain or application: Elementary Net systems, Place/Transition systems, Net Condition/Event systems (NCESs) [6, 7], etc. For our work we have chosen as main method the Net Condition/Event systems, given the similarities between these systems and the IEC 61499 standard.

The formalism of Net Condition/Event systems was introduced by Rausch and Hanisch [10] as a modular extension of Signal-net systems (SNSs), informally described in the next section.

2.1 Signal-Net Systems

This section briefly gives some informal definition of Signal-net systems, according to [6]. More formal definition may be found in [11].

A Signal-net system is a place/transition model similar to Petri nets. Basic artifacts of the place/transition models are: places, which bear tokens; (net) transitions, and arcs connecting places with transitions and transitions with places, known as token flow arcs. SNS in addition have two types of arcs: event arcs from transitions to transitions and condition arcs from places to transitions.

2.1.1 Semantics

The semantics of Signal-net systems is defined by the firing rules of net transitions. There are several conditions to be fulfilled to enable a net transition to fire.

First, as in the ordinary Petri nets, an enabled transition has to have a token concession, i.e. all the flow arcs from its pre-places have to be enabled. A flow arc is enabled when the token number in its source place is not less than its weight.

In addition to the flow arcs from places, transitions may have incoming condition arcs from places and event arcs from other transitions. A transition is enabled by condition signals if all source places of the condition signals are marked by at least one token. With respect to incoming event arcs a transition can have either OR or AND mode (event signal sensitivity mode). In the OR mode to fire a transition it is necessary that at least one event arc lead a signal-event whilst in the AND mode all event arcs must lead a signal-event.

Several SNS transitions can fire simultaneously. A set of simultaneously firing net transitions is called step. A step is formed by first picking up a nonempty subset of enabled spontaneous transitions, and then by adding as many as possible enabled transitions which are forced to fire by event signals produced by the transitions already included in the step.

2.1.2 Discrete timing

The concept of discrete timing is applied to the SNS as follows: to every pre-arc \([p, t]\) of the transition \(t\) we attach an interval \([l, h]\) of natural numbers with \(0 < l < h < \infty\). The interval is also referred to as permeability interval.

The interpretation is as follows: every place \(p\) bears a clock \(u(p)\) which is running if the place is marked \((m(p) > 0)\), and is switched off otherwise. The clocks run at the same speed measuring the time the token status of its place has not been changed. If a firing transition \(t\) removes a token from the place \(p\) or adds a token to \(p\), the clock of \(p\) is turned back to 0. A (marking-enabled) transition \(t\) is time-enabled only if for any pre-place \(p\) of \(t\), the clock at place \(p\) shows a time \(u(p)\) such that \(l(p,t) < u(p) < h(p,t)\).

2.2 Net Condition/Event Systems - NCES

The general idea of Net Condition/Event formalism is to design a system as a set of modules with a particular dynamic behavior and their interconnections via signals [6].
Once designed, the modules can be re-used over and over again. Each module has inputs and outputs of two types:
1. Condition inputs/outputs carrying information on marking of places in other modules, and
2. Event inputs/outputs carrying information on firing transitions in other modules.
Condition input signals as well as event input signals are connected with transitions inside the module.
The NCES concept provides a basis for a compositional approach to build larger models from smaller components. The “composition” is performed by connecting the inputs of one module to the outputs of another module, as shown in Fig. 2.

The result of the composition of two NCES, \( N_1 \) and \( N_2 \), is an NCES \( N_{1+2} \) obtained as a union of the components. If such a module has no external inputs then it represents a Signal-net system (SNS).
NCES can precisely follow the structure of popular block diagram modeling and implementation languages, such as state flow of MATLAB/Simulink and the function blocks of the IEC 61499 standard [9].

3 The Working Scenario
As we already have mentioned in the Introduction, our paper is focused on an example of defining a formal model for a control and communication mechanism which has to assure the exclusive access of three production facilities to a shared resource.
The considered example is built around a Bosch Rexroth conveyor with three flows, presented in Fig. 3a.

Along the conveyor belts three production facilities, called as \( A, B \) and \( C \) are considered to be placed, as is illustrated in Fig. 3b.

Fig. 3. a) Bosch Rexroth conveyor with three flows; b) Schematic of the system

Each production facility is placed alongside a stop gate with two proximity sensors. Considering the case of machine \( A \), it is placed alongside the stop gate \( VE_4 \), which have attached the proximity sensors \( S_9 \) and \( S_{10} \).
A working scenario for the machine \( A \) may by as follows: when a pallet arrives, it is stopped by the gate \( VE_4 \) and its presence is detected by the sensor \( S_9 \). Now the production facility can execute some operations to the components transported on the pallet. After finishing its work, the production facility \( A \) will set free the pallet by opening the gate. The departure of the sensor is confirmed by the activation of the sensor \( S_{10} \). Its activation will cause the gate to go back in the blocked state in order to stop the next arriving pallet.
When a pallet leaves a production facility it enters in a shared route area, marked in the Fig. 3b with the gray color. The critical route area is considered to be formed by the routes between each production facility \( (A, B \) and \( C) \) and the proximity sensors \( S_{15} \) and \( S_{16} \). In order to avoid the collision of the pallets released by production facilities, their access in the shared area must be controlled such as one pallet enters at a time. Once this pallet leaves the critical area, another pallet can be released by a production facility. As a result, a...
A mechanism for communication and control must be developed, which has to assure the correct access of production facilities to the shared route area. The next sections will present some formal models for such a mechanism.

4 Formal Models

4.1 Formal Model Based on Place/Transition Systems

Based on the concepts presented in [4] a model for the communication and control mechanism was firstly developed, using as formal method a classical form of Petri nets: Place/Transition systems. This model is depicted in Fig. 4 and the notations for the PNs are mentioned in Table 1.

The control mechanism is consisting of four distributed software components: three of these are considered to be placed into the production facilities’ controllers, and the fourth, having the role of a supervisor, may be located into a different embedded device.

![Diagram of the access control mechanism based on P/T systems](image)

**Fig. 4.** Model of the access control mechanism based on P/T systems

<table>
<thead>
<tr>
<th>Places/transitions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(p_2, p_3, p_{10})</td>
<td>Shared places having the meaning of messages passed from production facilities (A, B) and (C) to the supervisory component. Through these messages the production facilities may request access to the shared route.</td>
</tr>
<tr>
<td>(p_5, p_6, p_{11})</td>
<td>Shared places having the meaning of messages passed from the supervisory module to the production facilities (A, B) and (C) respectively. Through these messages the machines are granted with access to the shared route in order to deliver one pallet.</td>
</tr>
<tr>
<td>(t_1, t_3, t_9)</td>
<td>Transitions meaning that a new pallet has became ready to be delivered. As a consequence, the corresponding production facility has to send a message to the supervisor component in order to get access to the shared route.</td>
</tr>
<tr>
<td>(t_5, t_6, t_7)</td>
<td>Transitions through that the supervisor component grants the route access to a certain production facility.</td>
</tr>
<tr>
<td>(t_2, t_4, t_{10})</td>
<td>Transitions inside the models of production facilities, signifying the action of releasing a ready pallet.</td>
</tr>
<tr>
<td>(p_8)</td>
<td>After granting the route access to a production facility, the supervisor component will wait in this state for the shared route to become free. This will happen when the pallet occupying the route will pass one of the proximity sensors (S_{15}) and (S_{16}).</td>
</tr>
<tr>
<td>(t_8)</td>
<td>Transition meaning that the shared route has become free.</td>
</tr>
</tbody>
</table>
4.2 Formal Model Based on NCES

A more effective and intuitive way for modeling the above presented mechanism is to use the NCES models. The condition and event signals, available in NCES allow a more clear specification of the messages passed between modules. Also, the modularity of NCES improves the models development process due to the possibility of reusing previously defined models. In Fig. 5 is depicted the NCES version of the communication and control mechanism.

The components implied in the control mechanism are modeled as modules, called as AccessCtrl A, AccessCtrl B, AccessCtrl C and Access Supervisor. The first three modules, considered to be located in the production facilities’ controllers, are identical. Using the NCES tools one can build a single model of this type, save it into a library and then use its instance as often as is needed.

The AccessRQ output of the AccessCtrl A module serves as incoming condition for the transition $t_5$ inside the Access Supervisor module. That means that the transition $t_5$ may become enabled only if the place $p_3$ contains a token (the software component is not busy) and the AccessRQ signal is true. Firing of the transition $t_5$ leads to the generation of the event Access_1, and the token from the place $p_3$ will pass to the place $p_4$. The event Access_1 serves as incoming event for the transition $t_2$ of the module AccessCtrl A and may be interpreted as a message through which the supervisory module grants the route access to the production facility $A$. The event forces the firing of transition $t_2$, which may be seen as an action through that the production facility $A$ release the waiting pallet. After the firing of transition $t_2$ the condition output AccessRQ becomes false (the token is no longer in $p_2$) and the module enters in a waiting state for a new pallet to be ready (now the token is in the place $p_1$). The conflict generated by the simultaneous activation of more than one transition from

![Fig. 5. Formal model of the control mechanism based on NCES](image-url)
the set \( \{t_5, t_6, t_7\} \) is assumed to be solved by an internal algorithm of the supervisory module.

Finally, the place \( p_4 \) represents the state where the supervisory module waits for the shared route to become free. Outgoing from this state is possible by firing of transitions \( t_6 \) or \( t_7 \), i.e. through the activation of one of the proximity sensors \( S_{15} \) and \( S_{16} \).

### 4.3 The Autonomous NCES Model

In order to analyze an NCES module it has to be an autonomous one, i.e. to have no external inputs. Such a module represents a Signal-net system.

It may be seen that the model depicted in Fig. 5 is not autonomous because it has five not assigned inputs, subject of external signals. Therefore we added to our model four modules (\( \text{Sim}_A \), \( \text{Sim}_B \), \( \text{Sim}_C \) and \( \text{CnvModel} \)), such as the resulted model do not need any external signal (see Fig. 6).

The modules of type \( \text{Sim}_X \) simulate in a simple manner the activity of the production facilities \( A \), \( B \) and \( C \), considering the processing time corresponding to each machine being 5, 4 and 3 time units respectively. A pair of modules \( \text{Sim}_X \) and \( \text{AccessCtrl}_X \) are grouped and encapsulated in a new module having the type \( \text{Machine}_X \).

The module \( \text{CnvModel} \) is designed to model the conveyor behavior since a production facility releases a pallet until it leaves the shared route (moment signalized through the activation of one of the condition output \( S_{15} \) and \( S_{16} \)). The time for transporting a pallet from any production facility to the proximity sensors \( S_{15} \) or \( S_{16} \) is considered to be 4 time units.

**Fig. 6.** The autonomous model of the control mechanism

### 5 Conclusion

Formal methods are widely used in modeling of distributed and concurrent systems in order to assure the correct system behavior by means of formal verification techniques.

This paper has exemplified the use of Petri net based formal methods for modeling the low-level interaction between holons (in our example represented by the production facilities and their associated intelligence). It was shown that NCESs represent an appropriate modeling solution for this problem due to their particular
characteristics: modularity and synchronization of transitions by means of signals.

References: