Complex System Model based on Multi-Agent Systems and Petri Nets

Karima BELMABROUK, Fatima BENDELLA, Samira Benkhedda
University of Science and Technology of Oran USTO-MB
Oran, Algeria

Abstract - This paper discusses the integration of the multi-agent paradigm and of Petri-Nets of high-level modeling and design of a compact disk production system (PS).

We try first to describe the bases of design methodology using Petri nets of these systems which are by nature complex and distributed. For this, we will introduce a simplified example of a production process. And we will try, then, to define a model of transformation of the corresponding Petri-Net (PN) to a multi-agents system (MAS), passing through an intermediate phase, allowing obviously generating automatically the source code related to the obtained MAS.

The objective of our approach is to take advantage of the power of decomposition of multi-agent systems problems and their ability to represent complex systems, on the one hand, and the ease of modular representation of the complex systems offered by Petri nets the other hand. What will offer us a conceptual modeling clear and realistic of the different agents, formalization and orderly and formal analysis of the global model obtained.

Keywords-Multi Agent Systems (MAS); Petri-Nets (PN); Models; JADE; Complex Systems (CS).

1. Introduction

The global economy has caused a change in customer demands. Indeed they may desire for better quality products with manufacturing delays and delivery shorter and shorter. This trend is forcing companies to constantly innovate to optimize their manufacturing processes [11].

It may be noted that for most companies, the problem is mainly to improve the existing production facilities. We can see that the knowledge of the operation of a system in place is not as simple as you want. Or to control its processes and improve its operations a company must know all the facts of the problem. Modeling a production system to better understand what data are considered in the problem and how they evolve during the successive transformations.

The multi-agent systems have shown their interest to address a problem related to complex systems where the dynamics come into play ecological, economic and social dynamics [2] [6]. But, despite the existence of many applications based on agents, it still lacks the experience to design and construct MAS on an industrial scale [10].

The main objective of this paper is to contribute to the improvement of development of MAS tools, in order to facilitate the task to the programmer, by offering to integrate modeling components of MAS, on an existing platform such as JADE for Java Agent DEvelopment framework [14].

This article has five sections. The first is a literature on some research work in the field of MAS. The next section presents the problem raised. We describe, then, our contribution in order to remedy the problem, using a concrete example of a simplified production process. This example will be treated using PN, first, and using MAS, after, before discussing the mechanism of transformation to be implemented between our two models. Finally, we conclude with a discussion of our proposal and some perspectives for future works.

2. Some existing work

The specification of a MAS involves the identification of a large number of entities and their relationships. Which requires the management of different perspectives of the system. Several issues regarding these aspects have their origin in the presence of conflicting goals and tasks, inconsistencies and unexpected behavior [10]. These problematic configurations should be detected and prevented during the development process to consider alternative ways to treat them.

Approaches for engineering of SMA are numerous [5] [12]. They differ in the concepts used and their respective semantics. Indeed, each approach is influenced by the type of MAS design. The tools offered by each approach also differ and depend on
the semantics and the formalism used to model the MAS. GAIA [12] and OMNI [5] have adopted a formal notation or the least incorporating formal aspects. They have several levels of modeling at various abstraction levels. These approaches allow to refine models of the needs identified from an initial design up.

A formal language OZS was defined in [8] to specify different aspects of the MAS (functional, reactive, temporal, etc.). It allows giving a formal semantics to the organizational concepts used for MAS engineering.

The development environment of the agents platform based on java for multi agent systems (JAFMAS / in) for Java - based Agent Framework for Multi-Agent Systems (JAFMAS) is a graphical development environment that allows modeling and specification of a set of agents with their conversations [3] . In addition, JAFMAS / automatically convert conversation models represented as finite-state-machines competing in a PN for the simple MAS. This PN is then analyzed using an external tool for verifying PN for the following properties: Consistency, liveness and reversibility. If the PN has all these properties, we consider the conversations for this system to be coherent and logically consistent [3]. The work presented in [1], represent the basis for a design approach dedicated to the control of flexible production systems. This methodology integrates joint, multi-agent paradigm and hierarchical colored Petri-nets combined with functional language ML.

3. The issue raised

The vast majority of applications based on agents is built without using agents reusable components and agents can not be generalized [3]. That is why research on methods, tools and platforms for agents is very important for the implementation of the technology agents outside the purely academic field. F. Bousquet and al. [2] were asked the following question: "How to create and operate collective structures from agents who have greater or lesser capacity of representations, including exchanging information, services or goods, which develop contracts and are immersed in a dynamic environment that responds to their actions?"

To answer this question, few methodologies have brought significant real cases to the practice and are assisted by tools. In addition, existing methodologies do not consider certain aspects of the life cycle, usually the analysis and design.

Having a consistent, effective and proven opportunity to enjoy the benefits brought by the technology of agents, in the same manner as methodologies. For the implementation, most methodologies are conditioned by the use of specific agent architecture. Based on the comparative study of some MAS development platforms made by Tony Garneau and Sylvain Delisle in [7], we can infer that:

- Tools providing graphical utilities for the development and implementation were a lack of extensibility and flexibility.
- Those who provide interfaces to implement decrease the effort of programming.
- Most complete tools offer a methodology.
- Most tools do not support any mechanism for data backup.

It should also be noted that:
- The majority of tools have been developed to exploit a particular concept.
- Tools negligent, intentionally or not, several key aspects to the implementation of the MAS.
- This makes it often impossible to use for the development of real systems.

From this and related to the fact that there is no specified methodology for the development of application on the JADE platform [14], we thought the integration of a few design tools for this platform in order to reduce the programming effort and to enable the implementation of relatively complex systems.

For this, we tried to do the following:

1. Choose a formal design for our systems modeling tool: we opted for Petri-Nets, to start and we’ll try after to move on to other tools, subsequently, such as UML. That is why we have found it necessary to present PN in the fourth section.
2. Choose a concrete example of production system, on which we based our study: our choice is focused on a compact disc manufacturing process. Modeling it by Petri nets will be the fifth section.
3. Try to implement MAS corresponding to the example described previously with Petri-Nets under JADE platform, using MAS.
4. Define a transformation model of Petri net to MAS, thus generating the corresponding code.
5. Conduct Integration of a design tool by PN (if an existing tool otherwise proceed with the implementation of a new tool) to the JADE platform.
6. Check and test the validity of the proposed model, trying to model the PN of the treated sample and then generate its MAS.

Other perspectives will be considered, once the proposed model validated.

4. Modeling by Petri-Nets

A. Concepts and definitions [4]

1) Definition 1

A Places-transitions Petri-Net R is defined as a tuple: (P, T, pre, Post) where:
- P is the set of places;
- T is the set of transitions;

...
- Pre $T = P \times N$, an application prior to impact where $Pre(p,t)$ contains the integer "n" value associated with the arc from "p" to "t".
- Post $T = P \times N$, an application of impact back where $Post(p, t)$ contains the integer "n" value associated with the arc from "t" to "p".

**a) About places:**
- "p" is an input place of transition "t" if $Pre(p, t) > 0$
- "p" is an output place of transition "t" if $Post(p, t) > 0$

**b) Matrix representation**
It is easy to construct a matrix representation of a Petri-Net PT. We construct the matrix corresponding to that corresponding to Pre and Post. Lines identify the places and the columns, transitions. An intersection $(i, j)$, in Pre, is the cost associated with the arc leading place 'i' to transition "j", while an intersection $(i, j)$, in Post, is the cost associated with the arc leading the transition "j" instead "i".

**2) Definition 2**
The marking of a Petri-Net is an application $M: P \times N$ giving for each place the number of tokens it contains. The initial marking will be noted $M_B$ and $M(p)$ indicates the marking of the place "p".

**3) Definition 3**
The operation of a Petri-Net is defined as follows. For an "M" marking, transition "t" is known as obtainable (or passable or sensitized) if and only if that is, for all the places "pi", "t", the number of tokens, entries in "pi", $M(ft)$ is greater than or equal to the weight of the arc from "pi" to "t".

**4) Some additional notations:**
- The fact that "t" is obtained from "M" may be noted: $M \begin{array}{c} t > \end{array}$
- The shooting of "t" from "M", giving the new marking note "Me": $M \begin{array}{c} t > Me \end{array}$

**Graphical aspects of Petri-Nets**
A Petri-Net is represented by a graph-oriented with an initial $M_\circ$ marking. It has two types of nodes: places and transitions, connected by an arc-oriented (see Fig. 1). Transitions and places can be interpreted in different ways according to the modeling objective assigned to the Petri-Net [9].

**5. Modeling simplified production system of compact disk with Petri-Nets**
The production of a compact disk is successively in several main stages, which are:

1. **Test phase of the source CD**: is to test the CD provided by the client, subject to the order made.
2. **Galvanizing phase**: is the production of a substrate corresponding to the source CD.
3. **Mastering phase**: used to produce a Master from the substrate of the previous step.
4. **Replication phase**: based on the Master product in the previous step, a new medium called "Stamper" is product in order to use it in an industrial controller to the replication of the number of CDs ordered by the customer.
5. **Printing phase**: The replicated CDs are thus printed.

The Petri-Net shown in Fig. 2 is the first proposed model. It consists of all the places and transitions in the lists below:

**A. List of places**
P1. Source CD to replicate
P2. Source CD under test operation
P3. Defective Source CD
P4. Valid Source CD
P5. MASTER product
P6. Defective MASTER
P7. Valid MASTER
P8. STAMPER product
P9. Defective STAMPER
P10. Valid STAMPER
P11. Replicated CD
P12. Printed CD

**B. List of transitions**
T1. Start process (Job)
T2. Negative Source CD test
T3. Positive Source CD test
T4. Judgment of the Job
T5. Production of the MASTER
T6. Negative MASTER test
T7. Positive MASTER test
T8. MASTER Reproduction
T9. STAMPER Production
T10. Negative STAMPER test
T11. Positive STAMPER test
T12. STAMPER Reproduction
T13. CD Replication
T14. Replicated CD printing
T15. End process
The first model in Fig. 2 is with blocking. This is due to the transition "T4" (judgment of the job) that represents the judgment of the manufacturing process, where the source CD is defective, which means that our Petri-Net is not alive.

To remedy this problem, we propose to proceed with the correction of the defective source CD and re-injection in the process. This can be illustrated by Fig. 3, representing the second proposed model. In this model, the transition "T4" that caused the problem will be replaced by the transition ' T4' "such as:

**T4':** Correction of the defective source CD.

The second model, as proposed, can be reduced by using the first rule of reduction (R1) [4]. This led to the removal of places: P1, P6, P7, P9, P10 and P11, such presented in Fig. 4.

Using this rule we shall ensure the maintenance of the properties of the PN of the second model proposed. The list of places of the reduced Petri-Net will be defined by the set \{P2, P3, P4, P5, P8, P12\} and transitions by the set \{(t1, t15) (t2), (t3), (t4'), (t6, t8), (t7, t9), (t10, t12), (t11, t13, t14)\}.

Fig. 5 represents the marking graph of the third Petri-Net model presented in Fig. 4. It should be noted that this graph is a strongly connected component without outgoing arcs. This means that our Petri-Net is alive and therefore structurally alive.

This graph contains at least one arc, so there is no blockage.

To finish we can conclude that the corresponding Petri-Net is bounded N regardless of the initial marking, which implies that it is structurally bounded.

It is also reversible by definition, because all States are States of home.
Figure 6. Minimal P-Invariants of the third model

Our Petri-Net is alive and bounded, thus, it is persistent and with free choice. There are four basic circuits. So we can deduce four minimal P-invariants (Fig. 6) and a single T-invariant (Fig. 7) which is the identity.

\[
\begin{align*}
\text{T-inv}^4 &= \begin{bmatrix}
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
\end{bmatrix}
\end{align*}
\]

Figure 7. T-Invariant of the third model

The union of supports of the P-inv = \{P_2, P_3, P_4, P_5, P_8, P_{12}\} = P = \Rightarrow Our Petri-Net is conservative.

6. **Modeling simplified production system of compact disk with agents**

An agent is an element of our production system. It can be reactive or deliberative. Reactive if its behavior is directly related to its perceptions by a reflex function: stimulation / answer. This type of agents can be used to model the various agents of our system such as the operators of the various services corresponding to the various phases of production of a compact disk mentioned in the section 5, like the testers agents, the Galvanization phase agents, the Mastering phase agents, the Replication phase agents and the printing phase agents.

Contrary to these reactive agents, the deliberative agents, who argue about the purposes, consider some projection in the future. They ask questions like "What will happen if I do this or that action? And "Will I be happy if this happens?" Naturally, the agent reasoning about its goals is generally much more time to act than a reactive agent.

Two examples of deliberative agents can be considered relating to our production system. The first is the one of the coordination agent. This agent is responsible for the reception of the order of a customer and for its injection in the manufacturing process of compact disks corresponding to the received order. The purpose of this agent is to satisfy all the customers with whom he has to work. For that purpose, he can make decisions concerning the change of priority relative to a certain order, the cancellation of a current job, the change of the quantity of the ordered compact disks, or other.

The second example of deliberative agents is the one of the director agent whose role is to manage the production system. This agent can make strategic decisions such as the executing agents' recruitment, the integration of a new machine of production, the increase of the salary of an executing agent, the negotiation of the cost price of the raw material with the suppliers and others.

The Fig. 8 represents the architecture we proposed to model of our production system.

7. **Discussion**

Relating to our production system presented in the section 5, several rules must be defined to make transition from the proposed Petri Net in the same section to the multi agent system agents proposed in section 6.

We can see, in the Fig. 8 representing the architecture of our multi agent system that the communication between agents is done in two different manners, by sending messages directly or with blackboard.

If we limit ourselves to the communication by sharing information between the reactive agents of our system, we can conclude that this communication between agents reflects the various transitions between the places of our Petri Net presented in section 5. It is clear, also in our example, that the post-conditions of execution of the main task of an agent of test, for example, are the pre-conditions of execution of the main task of an agent of galvanization.

If we consider the communication by sending messages between agents, we can confirm that it is not
possible to model it using Petri Nets, but Multi agents systems are specialized to do it so it is very important to extend all our production systems modeled by Petri Nets to be modeled using agents because of their powerless to reflect the world and to support features which cannot be modeled otherwise.

8. Conclusion and perspectives

Among the benefits that can offer Petri-Nets, we can cite the possibility of obtaining directly executable models. Indeed, models developed by Petri-Net-based design are used directly for simulation and generation of different formal properties of the Petri-Net representing a production system. We were limited, through the example that we have chosen, to specification and design of our simplified production system, using Petri-Net for the first time and we tried after to represent the same example using agents for the second time. The aim of this approach is to start based on a valid model. This was achieved through the model we proposed in Fig. 4 and that faithfully reflects the process of production of compact disks and to propose multi agent architecture, which was presented in Fig. 8, in order to establish a system of transformation, enabling the passage of a model based on places and transitions (Petri-Net) to an agent based model, that was treated in section 6.

The proposed approach offers multiple research perspectives in the field of design related to production and complex systems. In our case, our future work aim to integrate formal methodologies in platform JADE, allowing, thus, specification, verification, validation and design of complex systems. This integration is to provide modeling Petri-Net-based graphics tools to designer of multi agent systems, helping them to develop MAS generic models. These methodologies could be used later for the automatic generation of code thus facilitating the implementation of our multi-agent systems representing different complex systems and their management and operation.

References


