Modeling Multi-Agent Systems Activities Through Colored Petri Nets

An Industrial Production System Case Study

João M. Fernandes
miguel@di.uminho.pt

Orlando Belo
obelo@di.uminho.pt

Departamento de Informática, Escola de Engenharia, Universidade do Minho
Largo do Paço, 4709 Braga Codex, PORTUGAL

KEYWORDS: Multi-Agent Systems, Knowledge Engineering, Distributed Systems, Production Systems Simulation, Colored Petri Nets.

ABSTRACT

This paper presents and discusses an industrial production system model based on Colored Petri Nets. The model was developed in order to analyse the agent's behavior of a distributed multi-agent system oriented to the simulation of the management and control of a specific hardmetal production system plant. This effort help us to create the basis to get a more robust system with means to resist better against eventual conflict, contention or deadlock situations among system's agents. Colored Petri Nets was chosen essentially due to their ability to describe explicitly system's states and actions and for having a semantic which builds upon true concurrency.

1 INTRODUCTION

A quick look to the global software market and a brief analysis on the Artificial Intelligence (AI) research field is sufficient to demonstrate that *Agent Based Technology* (ABT) grew significantly in the last years. Many work has been done in the conceptualization, design and implementation of agent based applications. We observe, day after day, the emergence of new theories, architectures and languages related with ABT arena. These enormous success is probably due to the fact that an agent based system provides high levels of modularity, extendability, asynchronism, decentralized control and opportunistic behavior. Mobile agent applications on electronic commerce activities [1], simulation of hardware systems [2], artificial personal assistants [3], distributed systems to help and support processes of medical care [4] and to assist monitoring in units of medical intensive treatments [5] and distributed computational environments oriented to support cooperation among intelligent agents [6] show that ABT has embraced wide and diversified application domains.

In *Multi-Agent Systems* (MAS) [7] [8] agents are organized by competence areas and distributed according to the needs of the applications and users. Problem solving activities are normally developed in a cooperative way involving the agents that have the expertise and knowledge required to do so. The MAS area provides the basis to study, to specify, to design and to implement applications that involve agent groups and the way agents act, interact, avoid conflicts and develop new forms of cooperation.

2 THE CASE STUDY

Based on the experience and models "imported" from MAS a distributed multi-agent system oriented to the simulation of the management and control of a specific production system of hardmetal products was developed [9]. The main goal of this simulation system (Fig. 1) is to emulate the "behavior" of a specific production plant controled and managed by a group of autonomous agents. The system considers several groups of agents with different knowledge and skills, containing each of them at least one agent. They were developed and distributed according to the main system's activities that were considered to be important for simulation purposes: production orders preparation (POP), raw materials preparation (RMP), pressing (PRE), machining (MAC), sintering (SIN), physical control (PHC), grinding (GRD), final control (FIC), packing and shipment (PAS), and monitoring (MON). According to the real production system, agents are located in specific production points, acting as machine supervisors and managers of their tasks. Each agent integrates the ex-

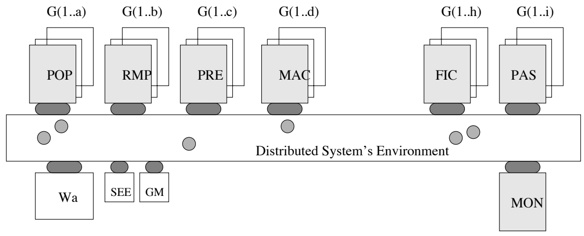

Figure 1: The Production System Simulation Environment.

pertise and knowledge needed to simulate the behavior of each machine during the realization of their main tasks. All the agents involved with the production system simulation, except those related to the production order preparation, are characterized by attributes that define their profiles: a code and an identification, the application area, a list of tasks, and a list of product families. The other agents presented in Fig. 1, namelly the Watcher (Wa), the System Environment Engine (SEE) and the Global Monitor (GM), are provided by the BeAble System [10] and they are in charge to do, respectivelly, global control and surveillance tasks, support the operationallity of the system's environment and system tasks monitorization. The BeAble system is a special kind of integrative system oriented to problem solving activities, involving several distributed and cooperative entities (agents). The system's agents can be knowledge based systems, user interaction platforms, or computational entities able to emulate some kind of physical device.

One of the several analysis tasks performed on the current system's implementation pointed to a redefinition of the global coordination model that supports the present prototype system towards a more efficient and effective one. This effort may help us to create the basis to get a more robust system with means to resist better against eventual conflict, contention or deadlock situations among system's agents. In order to achieve that goal, we need a model with abilities to represent all system's agents tasks, which includes the internal and external ones. Additionally, we also required that such model had a graphical representation, a well-defined semantics, an explicit description of the system's states and actions, and a semantic which builds upon true concurrency. We founded all these requirements in *Colored Petri Nets* (CPN) [11] [12].

3 COLORED PETRI NETS

Petri Nets (PN) were already used by some authors to model agent based systems [13] [14]. Five different approaches can be founded in [15] concerning the application of PNs to model agent based systems. The first one views a PN as a single agent. The PN is used as a state diagram representing the inner states of the agent. A second approach associates an agent with each transition occurred in the system and the whole PN represents an entire MAS. In this case, the behavior of the whole system is given by the occurrence net (reachability graph). In a third perspective, the behavior of each individual agent is described by a separate PN and a mechanism is used to represent the MAS, by combining the several PNs. Usually, the techniques used to combine PNs are place merging and transition merging, but we can also use hierarchical PNs to do so. In this approach, agents are PNs and their interconnections are described also by a PN which represents the next level of the hierarchy. A final view concerns the use of Object PNs to model MAS. In this view, tokens are considered as agents. Although promising, this last approach still lacks the definition of a formal semantics. However, as mentioned earlier, CPNs were selected to model the behavior of the hardmetal production system simulation.

CPNs have proved their usefulness in the description of systems where resource sharing, synchronization, and concurrency occur. CPNs are simultaneously a state- and an event-oriented language. The states of a model are represented by places, while the events are represented by transitions. Each place has an associated data type determining the kind of data which the place may contain. The two main reasons that led us to use CPNs in the system modeling are:

- a CPN model represents a description of the system being modeled and it can be used as a specification or as a presentation.
- CPNs can be analyzed, either by simulation or by formal methods of analysis.

These advantages can be reinforced by other positive characteristics of CPNs [12] such as:

- CPNs have a graphical representation.
- CPNs have a well-defined semantics.

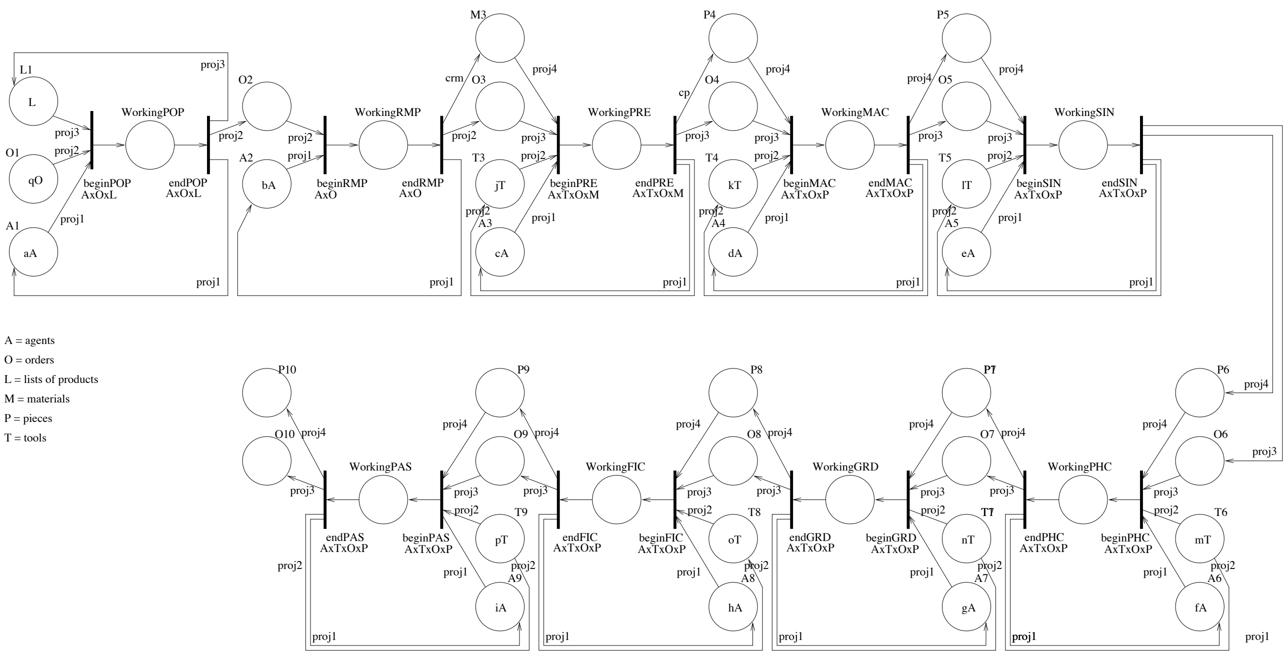

Figure 2: The CPN for the Production System Simulation.

- CPNs are very general-purpose.
- CPNs have very few, but powerful, primitives.
- CPNs have an explicit description of both states and actions.
- CPNs have a semantic which builds upon true concurrency, instead of interleaving.
- CPNs provide hierarchical mechanisms for description.
- CPNs integrate the description of control and synchronization with the description of data manipulation.
- CPNs can be extended to include time concepts.
- CPNs are stable towards changes of the modeled system.
- CPNs can be interactively simulated, with results being presented on the CPN diagram.
- CPNs can have their properties proven by several formal analysis methods.
- CPNs have computer tools supporting their drawing, simulation and formal analysis.

Fig. 2 presents the CPN model used to analyze and simulate the behavior of the hardmetal production system. The CPN uses the notation and assumptions described in [16]. A color, which means a given data type, is associated with each token and it represents a given resource of the production system, namely an agent, a tool, or a production order. In order to explain the CPN in Fig. 2 we choose one of the most relevant tasks in the system: the machining one. It can be described as follows: place $A4$ is initially marked with e tokens of type A (e agents) and place $T3$ is initially marked with l tokens of type T (l tools); a token $< a_i, t_j, o_k, p_l >$ of type $A \times T \times O \times P$ in place $workingMAC$ means that an agent a_i is using a tool t_j doing on p_l an operation listed in production order o_k; when the operation finishes (when transition $endMAC$ fires), the agent becomes idle again (or better, it becomes ready again to work), the tool is released and both the production order and the piece are passed to the next production phase (sintering). Similar considerations could be applied to the other system's tasks, since the CPN developed presents a regular structure.

In Fig. 3, the machining activity (i.e transitions $beginMAC$ and $endMAC$, and place $WorkingMAC$ of Fig. 2) is refined through a new CPN, which means that the CPN in Fig. 2 can be viewed as a top-level CPN. This hierarchical refinement can also be applied to the rest of the system's activities, following similar considerations.

The CPN of the machining activity is composed of five sub-activities: 1) production order analysis, 2) picking pieces, 3) tool preparation, 4) cutting pieces, and 5) quality control. Sub-activities 2 and 3 can run in parallel (or in any sequential order). The cut pieces have their quality level, described in the production order, controled when activity 5 is performed. If the quality is considered to be below the desired level, the piece is rejected and routed to be cut again. If the quality is above the desired level, the piece is accepted and forwarded to the next activity, which is, in this case, the

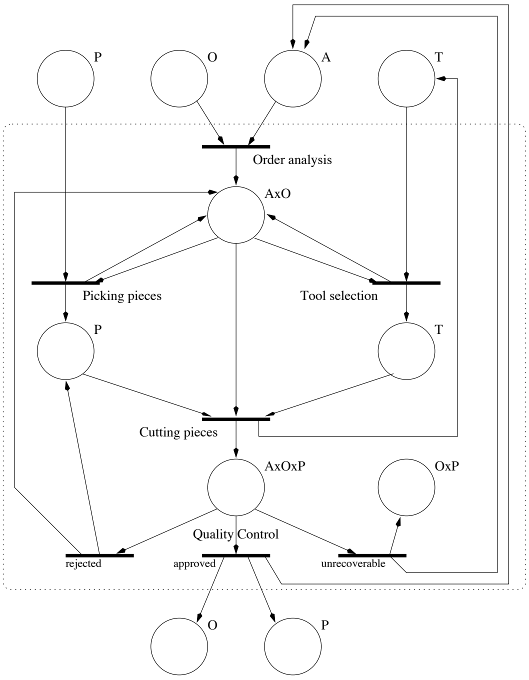

Figure 3: A CPN of the Refined Machining Activity.

sintering. A final possibility is the piece being considered unrecoverable, in which case it will be stored for recycling purposes.

4 CONCLUSIONS

The experiments carried out during the analysis and development of the model based on CPNs gave the possibility to test a priori several coordination strategies and system's behavior. Such way of (previous) control allowed us to optimize the implementation process, reducing the development time and, naturally, the maintenance costs. Additionally, it provides to the developer team a better system's behavior understanding. The possibility to model with CPNs the dynamic creation or deletion of agents along their lifes is among our perspectives for future work.

REFERENCES

[1] C.Munday, J.Dangedej, T.Cross, and D.Lukose. Motivation and Perception Mechanisms in Mobile Agents for Electronic Commerce. In *1st Australian Workshop on Distributed Artificial Intelligence*, pp. 144–158, Canberra, Australia, Nov-95.

[2] J.M.Fernandes, A.M.Pina, and A.J.Proença. Concurrent Execution of Petri Nets based on Agents. In *1st Workshop on Object-Oriented Programming and Models of Concurrency within the XVI Int. Conf. on Applications and Theory of Petri Nets*, Torino, Italy, Jun-95.

[3] T.Mitchell, R.Caruana, D.Freitag, J.McDermott, and D.Zabowski. Experience with a Learning Personal Assistant. *Communications of the ACM*, 37(7):80–91, Jul-94.

[4] J.Huang, N.Jennings, and J.Fox. An Agent-based Approach to Health Care Management. *Applied Artificial Intelligence: An International Journal*, 9(4):401–420, 1995.

[5] B.Hayes-Roth. Architectural Foundations for Real-Time Performance in Intelligent Agents. In *Second Generation Expert Systems*, pp. 643–672. Springer-Verlag, 1993.

[6] E.Oliveira and F.Mouta. A Distributed AI Architecture Enabling Multi-Agent Cooperation. In *6th Int. Conf. on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, Edinburgh, Scotland, Jun-93.

[7] M.Wooldridge and N.Jennings. Intelligent agents: theory and practice. *Knowledge Engineering Review 10*, 1995.

[8] M.Wooldridge and N.Jennings, editors. *Intelligent Agents, Proc. of the ECAI-94 Workshop on Agent Theories, Architectures, and Languages.* Lecture Notes in Artificial Intelligence (890), Springer-Verlag, 1995.

[9] O.Belo. A Hardmetal Tools and Wear Parts Production System Simulation. In *8th European Simulation Symp. (ESS'96)*, Genoa, Italy, Oct-96.

[10] O.Belo and J.Neves. An Architecture for Multi-Agent Systems Communities. In *10th Int. FLAIRS Conf.*, Florida, USA, May-97.

[11] S.Christensen. Coloured Petri Nets: Theory, Tools and Practice. Technical report, Computer Science Depart., Aarhus University, Denmark, Jun-92.

[12] K.Jensen. An Introduction to the Theoretical Aspects of Coloured Petri Nets. In *Lecture Notes in Computer Science (803)*, pp. 230–272. Springer-Verlag, 1994.

[13] T.Murata, P.Nelson, and J.Yim. A Predicate-Transition Net Model for Multiple Agent Planning. *Information Sciences*, 57–58:361–384, 1991.

[14] T.Holvoet. Agents and Petri Nets. *Petri Net Newsletter*, (49):3–8, Oct-95.

[15] I.Lomazova. Multi-Agent Systems and Petri Nets. In *Int. Workshop on Distributed Artificial Intelligence and Multi-Agent Systems (DAIMAS'97)*, pp. 147–152, St.Petersburg, Russia, Jun-97.

[16] R.David and H.Alla. *Petri Nets & Grafcet; Tools for modelling discrete event systems.* Prentice-Hall International, UK, 1992.