13 A MULTIAGENT BASED CONTROL SYSTEM APPLIED TO AN EDUCATIONAL SHOP FLOOR

José Barata[1], Gonçalo Cândido[2], Filipe Feijão[3]
[1]*Universidade Nova de Lisboa / Uninova – jab@uninova.pt*
[2]*Uninova – gmc@uninova.pt*
[3]*Uninova – fnf@uninova.pt*

This paper addresses the design and implementation of a multiagent based control architecture to support modular reconfigurable production systems. The requirements for plugability of modules (manufacturing components) and product changes were considered and tested against an educational platform based on fischertechnik, which resembles a production system composed of several workstations connected by a crane and conveyors.

1. INTRODUCTION

Multiagent systems (Wooldridge 2002) are becoming more and more a paradigm recognised as important in the context of the manufacturing world, not only at the high level aspects of the Enterprise Information Technology (Camarinha-Matos and Afsarmanesh 2001; Lin et al. 1999) but especially at the lower level aspects that include shop floor control, which is the focus of the work being presented here.

It is now widely recognised that external conditions to production companies, such as the need to cope with short life cycles products, stricter quality requirements, low costs, and customized products, can only be solved if agile shop floors can be built (Barata 2005; Gunasekaran 2001; Vernadat 1999). Considering in particular the assembly domain this need is particularly emphasized in (Barata et al. 2005; Onori 2002a; Onori et al. 2003a; Onori et al. 2003b). What is required is not a solution which tries to accomplish all of the envisaged assembly needs within a closed unit (Flexible Assembly systems) but, rather, a solution which, being based on several reconfigurable, task-specific, process-oriented elements (system modules), allows for a continuous evolution of the assembly system: this is known as the Evolvable Assembly Paradigm (EAS) (Alsterman et al. 2004; Onori 2002b). The main points behind shop floor agility are: 1) the need for modular manufacturing components, 2) the need to cope with product changes, and 3) the need for better operational support.

The need for modular components is directly connected with the multiagent paradigm for two reasons. The first reason is related to modeling and abstraction. In

Please use the following format when citing this chapter:

Barata, J., Cândido, G., Feijão, F., 2006, in IFIP International Federation for Information Processing, Volume 220, Information Technology for Balanced Manufacturing Systems, ed. Shen, W., (Boston: Springer), pp. 119–128.

fact it is quite simple to make the connection between a module and a modular component. The second reason is related to the need for plugability. In fact if a modular system is being considered they must be plugged or unplugged at wish. Therefore a multiagent system seems to be ideal because by definition a multiagent system is adequate to plug and unplug agents (in this case component modules). Another aspect connected to a modular system is the fact that a particular assembly or production system is a composition of modules, which can then be also considered as a coalition of modules. This is another connection between production modules and the agent world. A very well known and established work is in fact the work on agent coalitions (Castelfranchi et al. 1992; Klusch and Gerber 2002; Pechoucek et al. 2002; Shehory and Kraus 1995), which can be applied also here (Barata 2005).

The need to cope with product changes is also directly connected to the multiagent paradigm by the considering that an agent is an entity which is autonomous, reactive, and with social ability. In this case the autonomy makes the agent responsible for its own acts which mean that each manufacturing component or module is individually responsible for the actions over the product. Adding or removing one of these modules does not have a big impact on the overall system structure since each module is a confined individual entity. It must be noted that being confined does not mean any interactions with other modules. In fact the social ability that characterise the multiagent world is fundamental to ensure that the modules interact among themselves.

The need for better operational support means all the tasks that are required to support the shop floor while it is operating. This include aspects such as online reconfiguration of production parameters, maintenance support functionalities, advanced diagnostic systems, advanced user interfaces, etc. All these requirements are also better solved if the multiagent paradigm is considered, because among other aspects it is important to model each manufacturing component or module as an entity that includes all relevant information (maintenance parameters, process related variables, etc) that can be used in supporting the aspects referred before. This shows how the connection between modular component and agents is done. Another important aspect at this level is the autonomy that each agent (module) must possess. In fact, each agent must support simultaneously different behaviours or actions. For instance, an agent representing a certain module participating in a coalition (system) must simultaneously do the following tasks: 1) answer requirements for executing some work (for instance, the request to transport one piece from one point to another received by a gantry robot), 2) keeping monitoring the internal sensors of the module, 3) answering requests from external users for internal parameters values, 4) interacting with other modules for advanced diagnosis functionalities, etc.

From the previous paragraphs the connection between multiagent systems and Evolvable (agile) Shop Floors were made and emphasised in order to create the motivation for the work described in this paper.

The objective of this paper is two fold. The first one is to describe a multiagent based control system that supports an agile shop floor. The second goal is emphasizing that educational platforms such as Fischertechnik (Fischertechnik 2006) if mimicking real production systems are perfectly adequate to test real production systems because it is possible to add or remove modules, without

compromising the general goals, in a way that would be impossible with a real system.

The system used to test the multiagent based control is described in section 2, while the multiagent architecture is described in Section 3. The implementation aspects are presented in section 4 and finally the conclusions are described in Section 5.

2. SYSTEM DESCRIPTION

The production system used to test the multiagent based control system is composed of several machining centres, which are linked by conveyors and a gantry robot.

The MOFA France kit by Staudinger Gmbh (See Figure 1) simulates a flexible manufacturing system as a closed loop manufacturing circuit, achieving various possible situations based on generic manufacturing tasks.

1 – ToolMachine1 : Welder
2 – ToolMachine2 : Driller
3 – ToolMachine3 : Painter
4 – ToolMachine4 : Dryer
5 – MachineTable1 : Rotative Conveyor
6 – MachineTable2 : Slide Conveyor
7 – Conveyor Belt 1
8 – Conveyor Belt 2
9 – Conveyor Belt 3
10 – Conveyor Belt 4
11 – Robot – Crane
12 - Storage Area

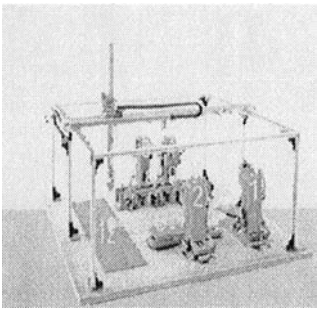

Figure 1 - MOFA France Kit

This kit has 4 machines (adaptable to any kind of manufacture tasks), a storage area, a cartesian robot (crane functions), local transporters and some material presence sensors.

The product which is stored in the storage area must be transported by the crane to the different machines. Some of the machines are also interconnected by conveyors. Some machines, such as machine 1, also include some critical resources such as a rotating table with two positions on the opposite sides of the table. This rotating table is a critical resource since it represents a kind of bottleneck for the machine 1. In the situation in which the machine is processing a piece but a position is vacant the crane can always bring a new piece. If, on the other hand, the table has already one finished piece and is processing one the crane should not bring a new piece before removing the finished one from the rotating table. This is more relevant if optimization of crane movements is to be considered.

Machine 2 is fed by a linear moving table with two places for pieces. This table can move between two positions. When the piece is being processed the other place is available for loading or unloading operations. This means that the crane can load or unload pieces from the machine 2 left or right hand side. One place of the table is always positioned in front of machine 2.

Machine 3 and 4 are connected by 3 bidireccional conveyors and the crane has several feeding places. It is possible therefore to connect pieces from machine 3 to 4 and vice versa using only the conveyors.

The pieces which represent pallets with parts that need to be transformed are very simple. In the case of this work each piece will be considered as an entity (agent) that includes the set of operations that need to be done.

A multiagent based control system was therefore suited in order to deal with critical resources, information exchange, parallel behaviours, sequential actions, as well as providing a clear and amicable graphical interface.

3. ARCHITECTURE

The main goal for the system being created is to provide a multiagent based architecture that could allow the addition and removal of new modules (manufacturing components) and control a system in which for each piece added with different process plans (this means different products) the system is able to realize the necessary operations without any new programming. The system must adapt to the new coming products represented by the pieces, which are also individually represented as agents.

3.1 System Architecture

The architecture needs to fulfill the previous requirements and provide a fully functional system, in addition to integrate several technologies.

The system architecture (see Figure 2) consists of several agents working together, trying to accomplish the production objectives. These agents have diverse abstraction levels according to their role on the environment, and they execute actions of distinct complexity levels. The roles of each production agent are known as skills since they identify what are the capabilities presented on each agent. These skills are important during system creation and also during system operation. The main agents involved are: Controller or Coalition Leader agents, Manufacturing Resource Agents (MRAs), and Agent Machine Interface (AMI).

The shop floor is composed of different Controller Agents, which represent different cell workstations that provide complex skills to other entities, such as pieces following its own process sequence. In the case of the considered scenario there is only one cell, and therefore only one coalition. Basically, these Controller Agents represent a group (or coalition) of manufacturing resource agents (also know as Basic Agents) whose high level requests are coming from the sequential order-build mechanism issued by the Controller Agent. This software mechanism tries to use all the available physical skills, setting up complex behaviours based on more simple ones, offered by lower level layer agents.

Manufacturing Resource Agents (MRA) represent a specific manufacturing component, such as robots, conveyors, machinery, etc. Using an agentified shop floor element, which encapsulates all provided skills, functions, interaction behaviours and internal status, it gives a clear view about a specific component, as well as giving a panoramic visualization of entire shop floor elements.

For some physical agents there is a need to use an AMI (Agent-Machine Interface), since physical agents developed are generic for each type of component and all the hardware connection is specific for each type of manufacturing equipment. So, if all the low level hardware communication is present on AMI and it can interact with its respective physical agent, there is no need of reprogramming. Instead, it is only necessary to configure agent parameters in the shop floor database. In a futuristic approach, these AMIs can be developed by module suppliers and delivered together with newly purchased equipment. In a simplistic form, the AMI connects the physical agent to the hardware, and then just executes orders from directly connected higher level agents. For legacy system, these AMI need to be created, in order to permit physical agent interactions.

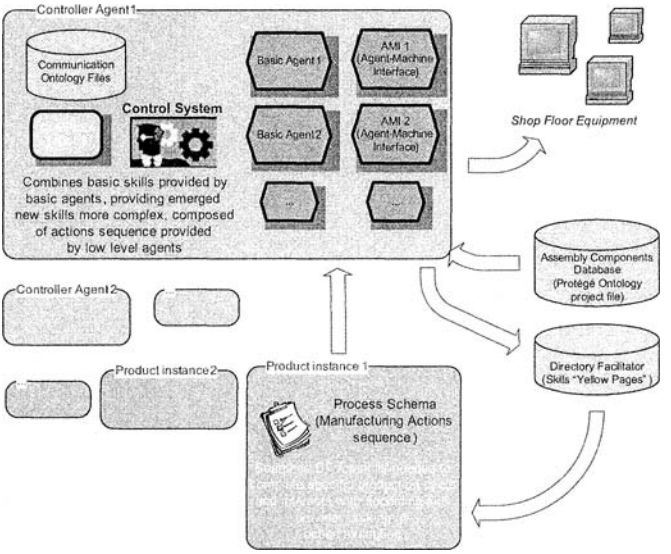

Figure 2 - System Architecture

The shop floor environment is commanded by each unique piece that wants to be processed. Each Piece Agent has its own custom production process pre-defined by user, and can be easily and rapidly changed thru a graphical interface. After the configuration phase, each piece knows its production objectives and behaves accordingly to it. Each piece agent goes thru its process plan order and asks the agent controller for the operations it needs until all of them are carried out.

All the interactions between agents use ACLMessages following FIPA (FIPA 2002) directives that permit necessary information exchange between them according to the immediate situation.

3.2 Manufacturing Resource Agent (MRA)

A MRA is nothing more than a shop floor component abstraction or module. It encapsulates the core of the physical component itself: skills, role and interaction behaviours.

The MRA reacts to external action requests, in order to execute its published skills, issuing all the necessary actions downwards in order to accomplish the requests received from the Controller agent that leads the coalition he is participating in. The MRA architecture is represented in Figure 3.

Again, all the interaction between agents follows previous methods, except the hardware equipment communication that is manufacturer dependent. This last type of communication needs original equipment interface libraries.

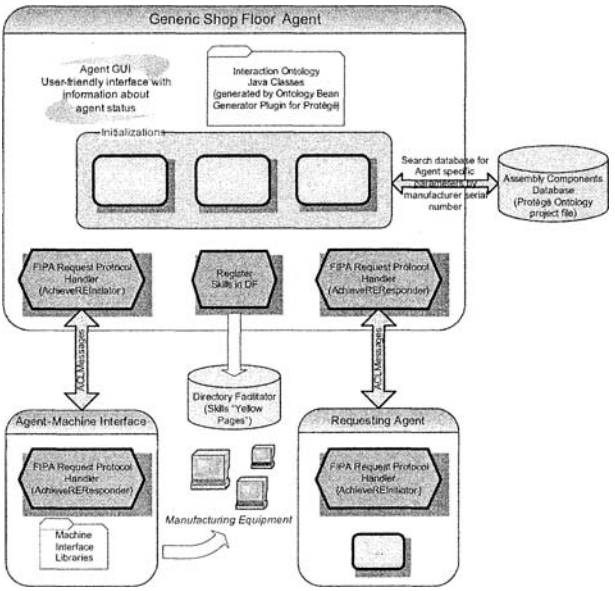

Figure 3 - Generic MRA architecture

3.3 Ontology

Taking into account the dimension and complexity of the system to implement, it was necessary to create a manufacturing components ontology, which takes into account its individual characteristics, as well as the relations established among them. The manufacturing components ontology defines the structure of the components in a production line as well as specific catalog information. At initialization phase, the MRAs search for configuration information in structures defined at this ontological level. MRAs get their configuration information from the database using their serial number.

Therefore, the ontology created, it is more correct in fact to call it Knowledge model because is a derivation from an ontology, is simple but generic. It supports the introduction of new components, adjustment of parameters and allows easy conformity. By using this process, it is easier to support fast alterations in the product, process and disposal of the assembly line, which is a crucial factor nowadays.

Skills are defined as basic or complex. A basic skill implements a service in its more elementary form. A complex skill is composed of basic skills or other lower

level complex skills, supplying services more complex than only the sum of constitutive elements.

Skills have to be considered according to two different perspectives: one defines what the service is, the other defines how it will be executed. A skill can therefore be considered for configuration (what is the service) and for execution (how it will be executed). This information can be found on the skills database and allows other agents to know which type of interaction and necessary parameters are needed to get the action done.

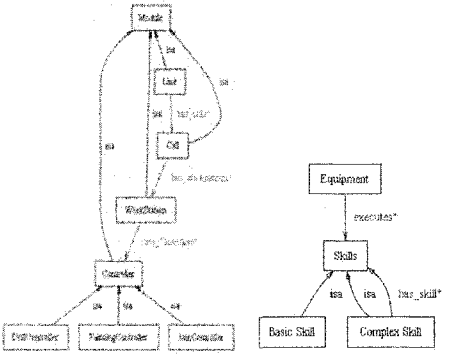

Figure 4 –Assembly Components Ontology relations examples

When an agent is launched, its first behaviour is reading the Assembly Database searching by component type and manufacturer serial name. Once the database entry is found, it contains all catalog information, AMI information, available skills and position on equipment tree. The agent collects that information, assumes a specific physical identity attached to real assembly component, registers skills and it is ready for interaction with other agents.

The Controller Agent database entry defines which agents it represents, which complex skills it affords and what are the interaction parameters. It contains all the relevant information about the coalition (group of MRAs) it is coordinating.

4. IMPLEMENTATION

4.1 Tools

The system was implemented using the JADE 3.3 (Java Agent DEvelopment Framework) (JADE 2001). JADE provides a distributed agent environment system that supports different functionalities such as FIPA specifications, compliant behaviours, and graphical tools for debugging and deployment phase.

For ontology and database development it was used the Protégé Ontology Editor and Knowledge Acquisition System (Protégé-2000 2000). This innate ontology development tool allows easy and prompt creations and updates because it is possible to integrate a Protégé project file in an ordinary java program. Instances of the MOFA France equipment were created in the Assembly Ontology and JADE behaviours were created in each MRA for initialization. In order to create the

communication ontology used for agent interaction, it was used the Ontology Bean Generator *plugin* for automatic conversion from Protégé to Java files to be used by all agents.

All the created code was made on Java JDK 1.4.6, using Eclipse 3.1 Development environment added with JADE and Protégé libraries.

4.2 Hardware

The hardware connection to the multiagent system implementation is composed of several software layers giving consecutively higher level functionalities, which enables a final implementation isolated from low level system.

The connection to the MOFA cell was tricky because the equipment is also used to support classes using traditional ways of programming such as C++ or JAVA. Therefore all the outputs and inputs of the MOFA are connected to an Data Acquisition Card located in one PC. Hence it was required to guarantee concurrency at this lower level. It must be taken into account that the system would be simpler, at this level, if each module (component of the cell) included each own controller.

Using a previously developed Java class for MOFA France kit control in each AMI project, it is possible to have multiple connections to the hardware, enabling the concurrency the system was lacking. Whenever a request arrives, each AMI simply needs to call one of the functions belonging to the developed Java class.

4.3 Interaction

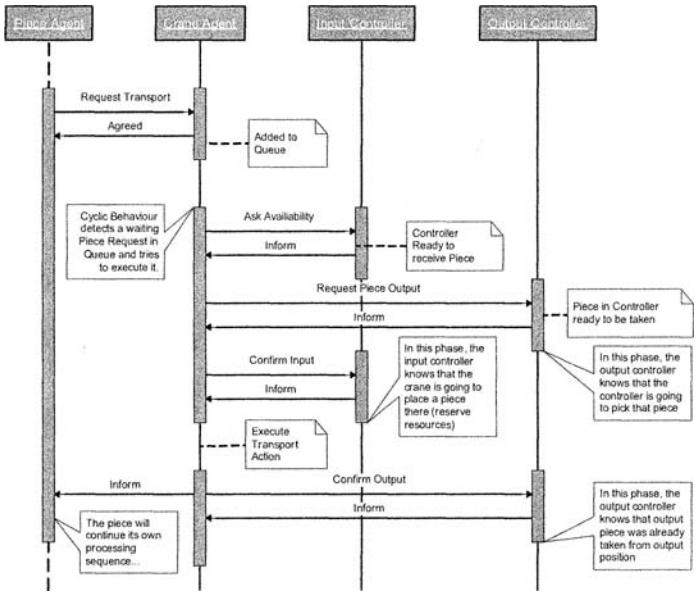

Figure 5: Crane Agent interactions

The cooperation between all the agents implies a robust interaction schema that supports information exchange and deals with error situations to avoid deadlocks and abnormal stopping situations. One of the most vital cases to deal with is the Crane Agent interactions as it consists of an extremely critical resource. In fact, no material can be transported between workstations if it fails (see Figure 5).

The above UML sequence diagram stands for a normal request for transportation made by a piece to the MRA crane agent from one origin (InputController) to a destination (OutputController). If some controller is occupied, the piece is remained on the queue and another request is chosen; then, its feasibility is checked and evaluated, according to its priority. With this approach of checking each request for transportation deadlocks are avoided.

All the messages exchanged follow FIPA Request Interaction Protocol, in SL1 language with communication ontology terms transporting information according to message performative and final objective. As an example, the Piece Agent that wants to be transported from Storing area to Join workstation interacts with Crane Agent using the next ACLMessage content example (see Figure 6).

```
((action
   (agent-identifier
     :name CraneMOFA1@PortatilGoncalo:1099/JADE
     :addresses (sequence http://PortatilGoncalo:7778/acc))
   (PieceAction
     :PieceStatus Incomplete
     :PieceRequest Crane_Transport
     :PieceArgs (sequence unknown Storing Join))))
```

Figure 6: ACLMessage content (PieceAgent to CraneAgent)

Due to space restrictions it is not possible to show all the interactions existing in the created system. However, they follow a similar pattern to this one, in which the major differences reside on the content of the SL1 language.

5. CONCLUSION

Due to space restrictions many important details related to the implementation were not included. The project behind this paper is working at the Electrical Engineering Department of the New University of Lisbon. The authors therefore believe the first goal enunciated at the beginning of the paper was achieved. In fact, a multiagent based control system is controlling the educational MOFA production system, which is composed of various heterogeneous manufacturing modules. Each of these modules was agentified and a multiagent based coalition was created that emulates the system.

The different tests made with different "products" with different process sequences proved the ability of the system to cope with these changes without any reprogramming. Moreover, it was also tested the addition of new models without implications in programming.

The second goal of proving that an educational platform can be useful for testing purposes were also achieved since the experience showed that a lot of effort can be saved if an educational platform is used instead of a real one. Of course, the authors

recognise that the real system can always provide new challenges or "surprises" but to test the concept of product changes and process adaptability the use of such an educational platform is very recommendable.

6. ACKNOWLEDGMENTS

The authors would like to acknowledge the European Commission which has partially funding this work trough the IP EUPASS project and the students Luís Ribeiro, Nuno Veríssimo, Alexandre Silva, João Veríssimo, Rodrigo Guerreiro, José Santos e Rui Milagaia.

7. REFERENCES

1. Alsterman, H., Barata, J., and Onori, M. (2004). "Evolvable Assembly Systems Platforms: Opportunities and Requirements." Intelligent Manipulation and Grasping, R. Molfino, ed., IMG'2004, Genova, 18-23.
2. Barata, J. (2005). *Coalition Based Approach For ShopFloor Agility*, Edições Orion, Amadora - Lisboa.
3. Barata, J., Camarinha-Matos, L. M., and Onori, M. "A Multiagent Based Control Approach for Evolvable Assembly Systems." *INDIN 05 - 3rd International IEEE Conference on Industrial Informatics*, Perth - Australia.
4. Camarinha-Matos, L. M., and Afsarmanesh, H. (2001). "Virtual Enterprise Modeling and Support Infrastructures: Applying Multiagent Systems Approaches." Multi-Agent Systems and Applications, M. Luck, V. Marik, O. Stepankova, and R. Trappl, eds., Springer-Verlag, Berlin, 335-364.
5. Castelfranchi, C., Micelli, M., and Cesta, A. (1992). "Dependence Relations Among Autonomous Agents." Decentralized A.I. 3, E. Werner and Y. Demazeau, eds., Elsevier Science Publishers B. V, Amsterdam, NL, 215-227.
6. FIPA. (2002). "FIPA ACL Message Structure Specification." *XC00061F*, FIPA - Foundation For Intelligent Physical Agents, Geneve.
7. Fischertechnik. (2006). "http://www.fischertechnik.de."
8. Gunasekaran, A. (2001). *Agile manufacturing : the 21st century competitive strategy*, Elsevier, Oxford ; New York.
9. JADE. (2001). "http://sharon.cselt.it/projects/jade/."
10. Klusch, M., and Gerber, A. (2002). "Dynamic Coalition Formation Among Rational Agents." *IEEE Intelligent Systems*, 17(3), 42-47.
11. Lin, F. R., Tan, G. W., and Shaw, M. J. (1999). "Multiagent enterprise modeling." *Journal of Organizational Computing and Electronic Commerce*, 9(1), 7-32.
12. Onori, M. "Evolvable Assembly Systems - A New Paradigm?" *International Symposium on Robotics*, Stockholm, Sweden.
13. Onori, M. "Evolvable Assembly Systems - A New Paradigm?" *ISR2002 - 33rd International Symposium on Robotics*, Stockholm, 617-621.
14. Onori, M., Barata, J., Lastra, J., and Tichem, M. (2003a). *European Precision Assembly - Roadmap 2010*, Assembly-Net Project.
15. Onori, M., Camarinha-Matos, L. M., and Barata, J. (2003b). "European Assembly - Status Report." *Assembly Automation*, 23(1), 8-12.
16. Pechoucek, M., Marik, V., and Bárta, J. (2002). "A Knowledge-based Approach to Coalition Formation." *IEEE Intelligent Systems*, 17(3), 17-25.
17. Protégé-2000. (2000). "http://protege.stanford.edu."
18. Shehory, O., and Kraus, S. (1995). "Coalition Formation among Autonomous Agents: Strategies and Complexity." From Reaction to Cognition, C. Castelfranchi and J. P. Muller, eds., Springer-Verlag, Heidelberg, 57-72.
19. Vernadat, F. B. (1999). "Research Agenda for Agile Manufacturing." *International Journal of Agile Management Systems*, 1(1), 37-40.
20. Wooldridge, M. J. (2002). *An Introduction to Multiagent Systems*, J. Wiley, New York.