A MULTIAGENT SYSTEM FOR JOB-SHOP SCHEDULING

Claudio Cubillos, Leonardo Espinoza and Nibaldo Rodriguez
Escuela de Ingeniería Informática, Pontificia Universidad Católica de Valparaíso, Av. Brasil 2241, Valparaíso, Chile

Keywords: AOSE, PASSI, Planning & Scheduling, Agent System, JSSP.

Abstract: The present work details the design and implementation of a multiagent system devoted to the dynamic Job Shop Scheduling problem. The agent system tackles the planning and scheduling of jobs and their corresponding operations on a set of available machines. The system has been modeled with the PASSI agent-based software development methodology and implemented over the JADE agent platform.

1 INTRODUCTION

In a multiagent system (MAS) diverse agents communicate and coordinate generating synergy to pursue a common goal. Hence as modeling artifact, agent-based systems borrow their key characteristics from us, humans, and our societies.

Therefore, multiagent systems can be seen as a natural evolution from Distributed Artificial Intelligence (DAI) and Distributed Computing (DC). This higher level of abstraction has allowed agents to tackle the increasing complexity of nowadays open software systems where integration, transparency and interoperation among heterogeneous components are a must.

For this technology to get more mature and widespread, the use of agent-oriented software engineering (AOSE) methodologies and tools are a key factor of success.

Hence, the present work describes the design of a multiagent system using a particular AOSE methodology called PASSI (Cossentino et al., 2003).

The chosen domain for the system corresponds to the job-shop scheduling problem under a dynamic scenario in which job requests coming from clients must be processed on-the-fly and where changes can occur due to changes in the environment.

2 THE JSSP PROBLEM

The traditional Job-Shop Scheduling Problem (JSSP), can be described by a set of n jobs $\{J_j\}_{1\leq j\leq n}$ which is to be processed on a set of m machines $\{M_r\}_{1\leq r\leq m}$. Each job has a technological sequence of machines to be processed.

The processing of job J_j on machine M_r is called the operation O_{jr}. Operation O_{jr} requires the exclusive use of M_r for an uninterrupted duration p_{jr}, its deterministic processing time, and each operation O_{jr} has pre-assigned materials $\{W_i\}_{1\leq i\leq k}$. In addition, each job has a due-date $\{D_j\}_{1\leq j\leq n}$.

A schedule is a set of completion times for each operation $\{c_{jr}\}_{1\leq j\leq n;1\leq r\leq m}$ that satisfies those constraints. The considered JSSP involves the scheduling of n jobs J on the m machines M and consuming k materials W while minimizing the total tardiness regarding the due-dates.

On the other hand, the dynamic variant of the problem adds the fact that the jobs to be processed are not known in advance and that they must be scheduled as they arrive.

It is one of the most hard NP-complete combinatorial optimization problems.

2.1 Related Work

Diverse proposals of agent-based systems can be found in literature tackling the job-shop or production scheduling problem.

In (Saad et al., 1995) a Production Reservation approach was proposed by using a bidding mechanism based on the Contract Net Protocol - CNP (Smith, 1978) to generate the production plan.

In AARIA (Parunak et al, 1997), the manufacturing capabilities (e.g. people, machines, and parts) are encapsulated as autonomous agents and use a mixture of heuristic scheduling techniques:

Cubillos C., Espinoza L. and Rodriguez N. (2008).
A MULTIAGENT SYSTEM FOR JOB-SHOP SCHEDULING.
In *Proceedings of the Tenth International Conference on Enterprise Information Systems - SAIC*, pages 148-153
DOI: 10.5220/0001705301480153

forward/backward scheduling simulation scheduling, and intelligent scheduling.

In (Maturana et al., 1999) the adaptive multi-agent manufacturing system architecture called MetaMorph combined the CNP with mediator-centric federation architecture was presented.

Other recent CNP-based solutions can be found in (Váncza, 2000) (Maturana et al., 1999) (Lim, 2002) and (Usher, 2002).

One of the contributions of the present work is to provide a more formal design of multiagent system devoted to Job-shop scheduling using the PASSI methodology.

3 PASSI METHODOLOGY

PASSI is a step-by-step methodology for designing and developing multi-agent societies. Its name stands for a Process for Agent Societies Specification and Implementation. PASSI integrates design models and concepts from both OO software engineering and artificial intelligence approaches using the UML notation.

The models and phases of PASSI encompass anthropomorphic representation of system requirements, social viewpoint, solution architecture, code production and reuse, and deployment configuration supporting mobility of agents. The design process with PASSI is supported by the PASSI ToolKit (PTK, 2005) to be used as an add-in for Rational Rose.

Figure 1 shows PASSI methodology consisting of five models plus twelve steps in the process of building multi-agent. These are briefly described in the following. Please refer to (Burratato, 2002) for a more detailed description.

System Requirements Model. Corresponds to an anthropomorphic model of the system requirements in terms of agency and purpose. It involves 4 steps: a Domain Description (D.D.), an Agent Identification (A.Id.), a Role Identification (R.Id.), and a Task Specification (T.Sp.),

Agent Society Model. It considers the social interactions and dependencies among the agents involved. It considers 3 additional steps: an Ontology Description (O.D.), a Role Description (R.D.), and a Protocol Description (P.D.).

Agent Implementation Model. Provides the solution architecture in terms of classes and methods and considers: an Agent Structure Definition (A.S.D.) and an Agent Behavior Description (A.B.D.)

Deployment Model. Describes a model of the distribution of the parts of the system across hardware processing units and the migration between processing units.

4 THE AGENT SYSTEM

The multiagent job-shop scheduling system stands over the Jade Agent Platform (Bellifemine et al., 1999), which provides a full environment for agents to work.

In the following subsections, the agent system is described making reference to the most relevant PASSI steps and artefacts, while considering space restrictions.

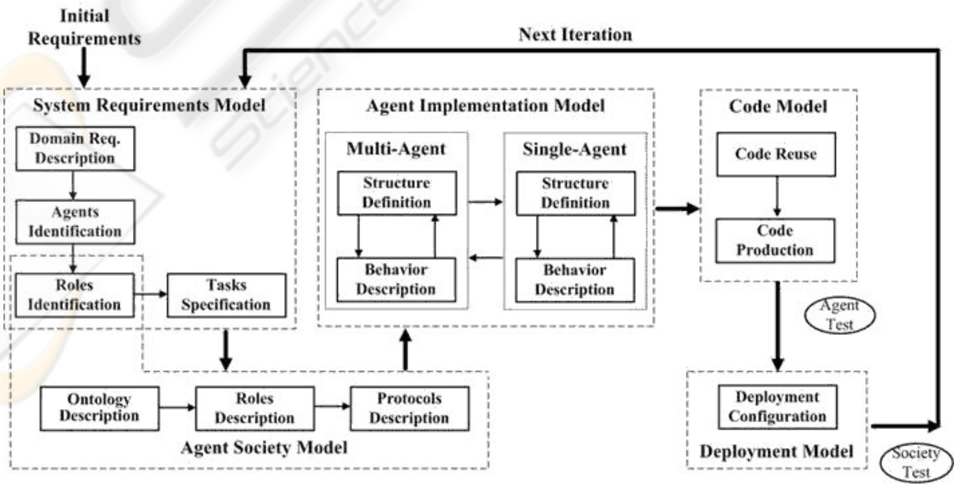


Figure 1: PASSI diagram showing Models and steps required (Burrafato, 2002).

4.1 Agent Identification (A.Id.)

In this step the use cases capturing the system requirements are grouped together to conform an agent. The diagram in Figure 2 shows the identified use cases for this job-shop system and the leveraged agents.

Firstly, the Client agent is a GUI agent in charge of the communication between an actual client and the rest of the system, providing the possibility of generating a job order, and to communicate inbound/outbound eventualities regarding such order due to changes in the environment (e.g. order modification/cancellation from client, order delay/reject from the system).

Machine agents encapsulate each real machine, being primarily in charge of its schedule management. This involves processing requests coming from Order agents and performing the scheduling process.

For this, it carries out a search in the solutions state space by implementing an optimization heuristic. In the actual system, a search algorithm presented by (Yoo et al., 2002), inspired in simulated annealing was implemented.

On its turn, Order agents are devoted to the job order management, its breakdown into operations, the request of necessary materials for each operation execution to Stock agents, and the request to Machine agents for the scheduling of each operation.

For the interaction with the Stock agents the FIPA Query Interaction Protocol (FIPA. 2002b) standard is used. In the latter case, the FIPA Request Interaction Protocol is used (FIPA, 2002a).

4.2 Task Specification (T. Sp.)

In this phase the scope is to focus on each agent's behavior, decomposing it into tasks, which usually capture some functionality that forms a logical unit of work and generating cohesion. Therefore for each agent an activity diagram is developed containing what that agent is capable of along the diverse roles it performs. In general, an agent will be requiring one task for handling each incoming and outgoing message.

As example, a portion of the tasks of the Order agent are depicted in Figure 3. The diagram shows five tasks on the right that constitute the Order agent capabilities. The *ReceiveOrder* task handles Client

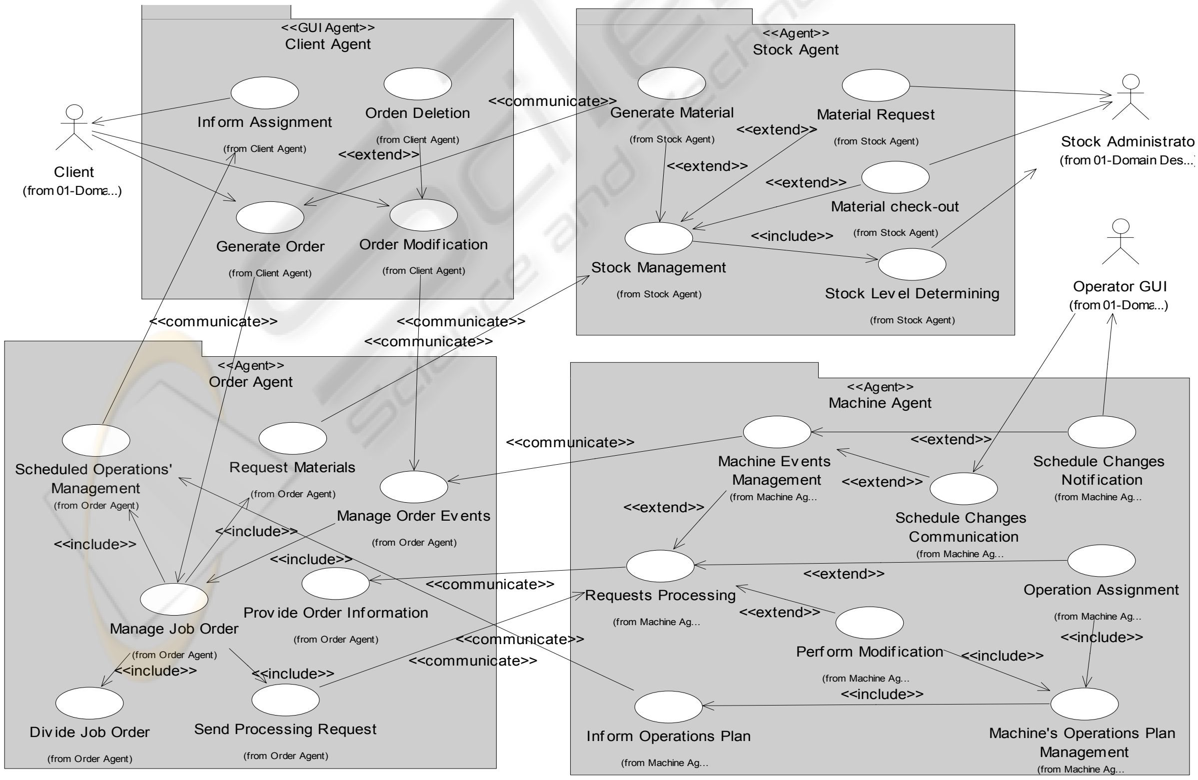


Figure 2: Agent Identification Diagram for the Job-shop scheduling system.

messages which request for an order to be processed. This one calls the *DivideOrderIntoOperations* task which splits the order into its corresponding set of operations on different machines. Each operation is forwarder to the *SendOperation* in charge of constructing and sending the request message to the Machine agent. The message is handled by the *OperationScheduling* task of the Machine agent.

In the following, once the machine has found a scheduling position of that operation, it answers back the Order through a *ReportOperationSchedule* task. Such a message is handled by the *ReceiveResponseFromMachine* task in the Order side. In this step an evaluation is performed in order to check whether if the operation corresponds to the last one within the order. In such a case, the order has been scheduled completely; otherwise still remain operations to be scheduled on machines.

After processing the last operation, the *ReportingOrderStatus* task is called, being in charge of collecting all the schedules of the order's operation set and informing the Client the actual schedule of its order. Finally, the Client handles the above message through its *ReceiveOrderInformation* task.

4.3 Ontology Description (O.D.)

In this step the agent society is described from an ontological perspective, providing them with a common knowledge of the job shop domain, and thus, enabling communication among involved actors and agents.

The following Figure 4 shows a portion of the Domain Ontology Description providing the concepts necessary for mutual agent understanding within the society.

On the center the *JobOrder* is depicted, which is decomposed into *Operations*. Other related concepts are *MaterialList, Stock* and *StockLevel.*

In the middle-upper part of the diagram we can identify on the left the Machine concept and on the right the Client concept. Both have *Events* and a *Utility Function* associated.

5 IMPLEMENTATION

As stated before, a system was implemented over the Jade Framework. In addition to the described agents, other simulation agents were created in order to coordinate the correct creation, execution and destroy of the agents along the diverse experiments and runs.

Other implementation issue regards the development of GUIs for Machine and Order agents. The above Figure 5 shows a screenshot of two Machine agent GUI's. The machines 1 and 11 are depicted in the foreground while having the Jade platform GUI at the background.

The GUI of each Machine agent shows a grid indicating the Job Order ID (e.g. Orden_C_8) and the task ID, that is, the relative order of the operation within the set of operations of the job order.

The grid also details the starting and processing times together with the Early and Latest Start Times (EST & LST respectively).

In the case of the Order GUI a similar approach was taken. In this case the grid shows the list of all the tasks (or operations) for the given Job order indicating times and the corresponding machines on each case.

Regarding the benchmark data for the experimentations, Figure 6 shows the format of the plain .txt file used to feed the Order and Machine agents. The example shows the 15x15 example (15 machines, 15 orders) from (Taillard, 1994).

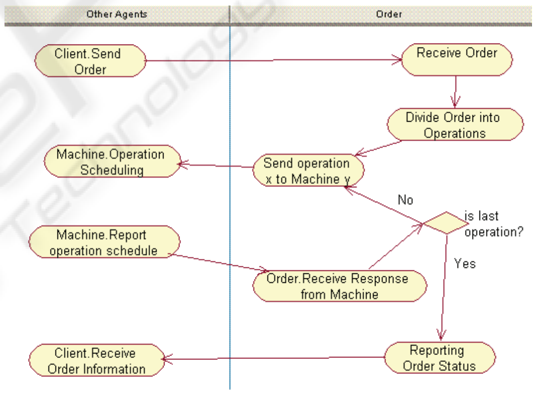


Figure 3: Part of the Task Specification Diagram for the Order Agent.

6 CONCLUSIONS

The design of an agent-based software system for dynamic job shop scheduling was described.

The agent formalization with PASSI promotes the architecture maintainability, its ability to cope with newer requirements and the possibility to scale and integrate other actors and systems.

Next steps consider the testing with benchmark data tackling diverse scenarios and topologies of distribution and the implementation of diverse scheduling algorithms (e.g. Genetic algorithm, tabú search, sa).

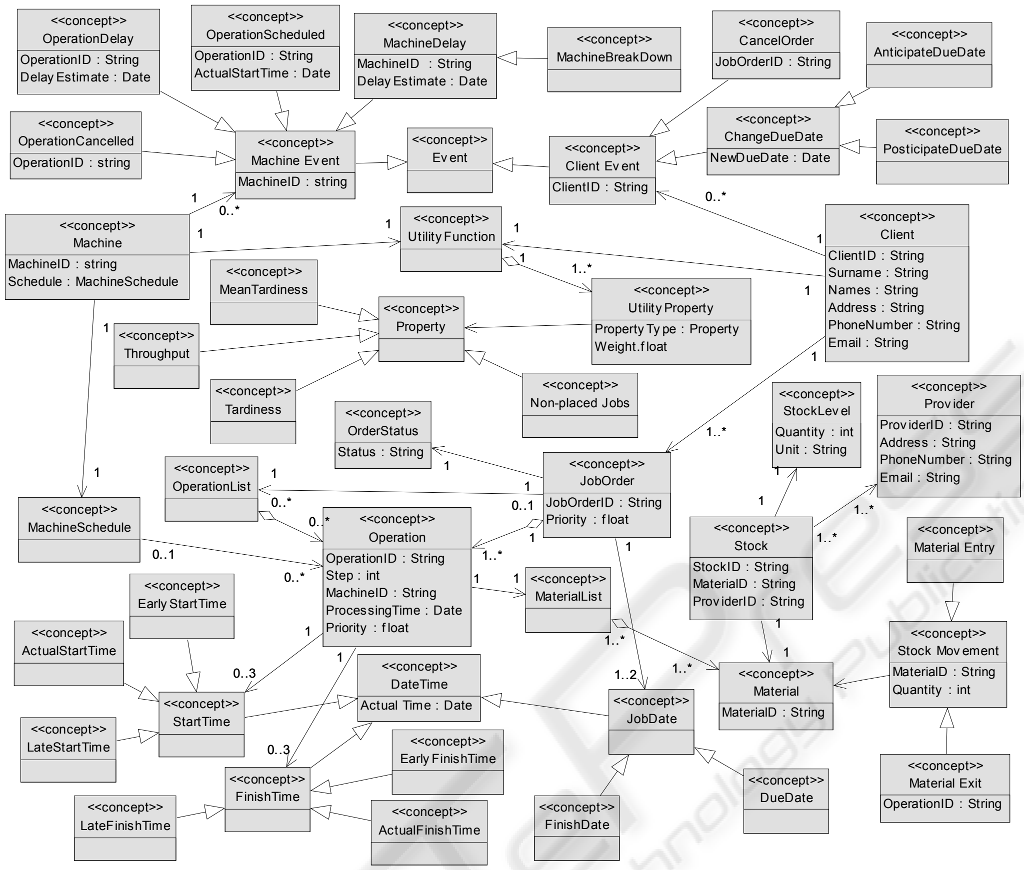


Figure 4: Part of the Domain Ontology Diagram for the Job-shop system.

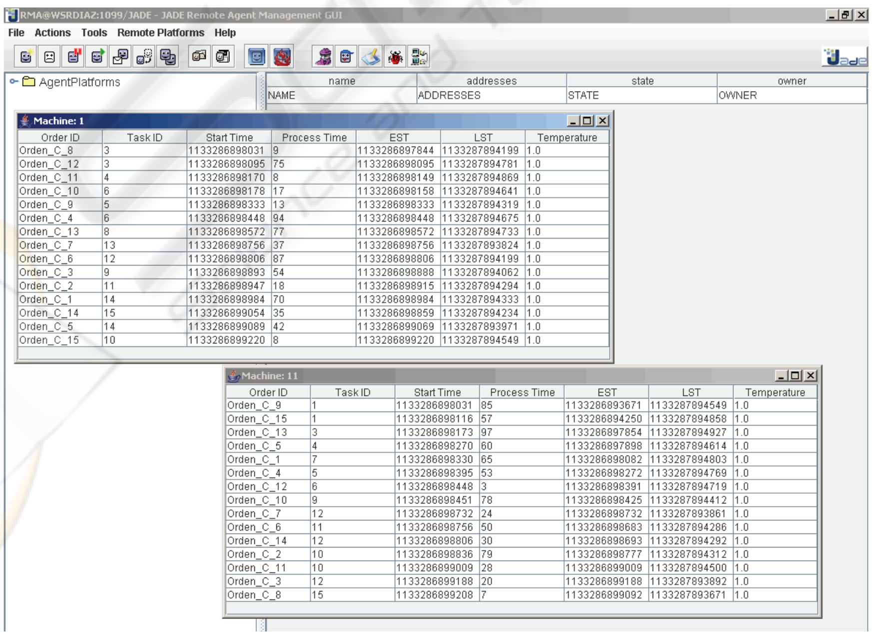
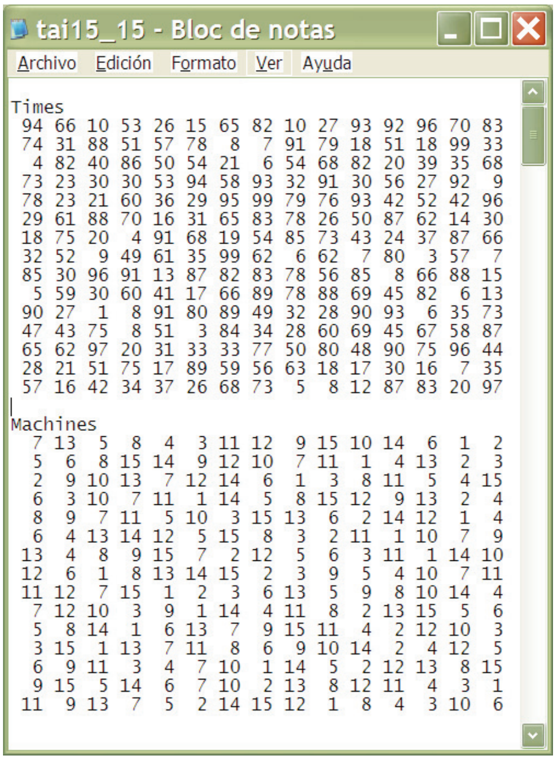


Figure 5: Screenshot of Order Agents GUI under a JADE Platform.



```
tai15_15 - Bloc de notas
Archivo Edición Formato Ver Ayuda

Times
 94 66 10 53 26 15 65 82 10 27 93 92 96 70 83
 74 31 88 51 57 78  8  7 91 79 18 51 18 99 33
  4 82 40 86 50 54 21  6 54 68 82 20 39 35 68
 73 23 30 30 53 94 58 93 32 91 30 56 27 92  9
 78 23 21 60 36 29 95 99 79 76 93 42 52 42 96
 29 61 88 70 16 31 65 83 78 26 50 87 62 14 30
 18 75 20  4 91 68 19 54 85 73 43 24 37 87 66
 32 52  9 49 61 35 99 62  6 62  7 80  3 57  7
 85 30 96 91 13 87 82 83 78 56 85  8 66 88 15
  5 59 30 60 41 17 66 89 78 88 69 45 82  6 13
 90 27  1  8 91 80 89 49 32 28 90 93  6 35 73
 47 43 75  8 51  3 84 34 28 60 69 45 67 58 87
 65 62 97 20 31 33 33 77 50 80 48 90 75 96 44
 28 21 51 75 17 89 59 56 63 18 17 30 16  7 35
 57 16 42 34 37 26 68 73  5  8 12 87 83 20 97

Machines
  7 13  5  8  4  3 11 12  9 15 10 14  6  1  2
  5  6  8 15 14  9 12 10  7 11  1  4 13  2  3
  2  9 10 13  7 12 14  6  1  3  8 11  5  4 15
  6  3 10  7 11  1 14  5  8 15 12  9 13  2  4
  8  9  7 11  5 10  3 15 13  6  2 14 12  1  4
  6  4 13 14 12  5 15  8  3  2 11  1 10  7  9
 13  4  8  9 15  7  2 12  5  6  3 11  1 14 10
 12  6  1  8 13 14 15  2  3  9  5  4 10  7 11
 11 12  7 15  1  2  3  6 13  5  9  8 10 14  4
  7 12 10  3  9  1 14  4 11  8  2 13 15  5  6
  5  8 14  1  6 13  7  9 15 11  4  2 12 10  3
  3 15  1 13  7 11  8  6  9 10 14  2  4 12  5
  6  9 11  3  4  7 10  1 14  5  2 12 13  8 15
  9 15  5 14  6  7 10  2 13  8 12 11  4  3  1
 11  9 13  7  5  2 14 15 12  1  8  4  3 10  6
```

Figure 6: Part of the Task Specification Diagram for the Order Agent.

ACKNOWLEDGEMENTS

This work is part of Project No. 209.746/2007 entitled "Coordinación en una sociedad multiagentededicada a la programación y control bajo ambiente dinámico", funded by the Pontifical Catholic University of Valparaíso (www.pucv.cl).

REFERENCES

Bellifemine, F. et al. 1999. JADE - A FIPA Compliant Agent Framework. C-SELT Internal Technical Report.

Burrafato, P., and Cossentino, M. 2002. Designing a multiagent solution for a bookstore with the PASSI methodology. In 4th International Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS-2002).

Cossentino, M., P. Burrafato, S. Lombardo, and L. Sabatucci. 2003. Introducing pattern reuse in the design of multi-agent systems. In Agent Technologies, Infrastructures, Tools, and Applications for E-Services, eds. R. Kowalszyk, J. P. Mu ller, H. Tianfield, and R. Unland, LNAI 2592, pages 107--120, Berlin: Springer-Verlag.

FIPA. 2002a. FIPA Request Interaction Protocol Specification, Standard, version H, 2002-12-06 Available at http://www.fipa.org/specs/fipa00026/

FIPA. 2002b. FIPA Query Interaction Protocol Specification, Standard, version H, 2002-12-06 Available at http://www.fipa.org/specs/fipa00027/

Fischer, K. 1994. The design of an intelligent manufacturing system. Proc. of CKBS'94, University of Keele, England, pp. 83–99.

Lim, M. K. and Zhang, Z. 2002. Iterative multi-agent bidding and co-ordination based on genetic algorithm. Proceeding of 3 Complex Systems, and E-Businesses, Erfurt, pp. 682--689.

Maturana F, Shen Weiming, Norrie D. 1999. MetaMorph: an adaptive agent-based architecture for intelligent manufacturing. International Journal of Production Research, 37: 2159-2173,.

Parunak, H.V.D., Baker, A.D. And Clark, S.J. 1997. The AARIA agent architecture: an example of requirements-driven agent-based system design. Procs. 1st Int. Conf. on Autonomous Agents, Marina del Rey, CA.

PTK. 2007. The PASSI Toolkit (PTK) Available at http://sourceforge.net/projects/ptk. Accessed on 09/10/2005

Saad, A., Biswas, G., Kawamura, K., Johnson, M. E. Salama, A. 1995. Evaluation of contract net-based heterarchical scheduling for flexible manufacturing systems. Procs. Int. Joint Conf. on Artificial Intelligence, Workshop on Intelligent Manufacturing, Montreal, Canada, pp. 310-321.

Smith, R. G. and R. Davis. 1978. Distributed Problem Solving: The Contract Net Approach. Proceedings of the 2nd National Conference of the Canadian Society for Computational Studies of Intelligence.

Taillard. E. D. 1994. Benchmark problems. available at: http://ina2.eivd.ch/Collaborateurs/etd/problemes.dir/ordonnancement.dir/ordonnancement.html

Usher, J.M. 2002. Negotiation-based Routing in Job Shops via Collaborative Agents. World Manufacturing Congress, Distributed Intelligence in Technology, Economic and Social Applications: An International Symposium, Workshop on Agent-Based Process Planning and Scheduling, Rochester, New York.

Váncza, J. and Márkus, A. 2000. An agent model for incentive-based production scheduling. Computers in. Industry, Vol. 43, Issue 2,pp. 173-187.

Yoo, M. J., Müller, J. P. "Using Multi-agent System for Dynamic Job Shop Scheduling". 4th Int. Conf. on Enterprise Information System (ICEIS 2002), Ciudad Real, Spain, April 2002.