PROPOSAL OF A FRAMEWORK FOR PRODUCTION PLANTS REMOTE CONTROL: A PRELIMINARY TEST CASE

Romeo Bandinelli, Mario Rapaccini
romeo.bandinelli@siti.de.unifi.it,
rapaccini@ing.unifi.it
Università di Firenze
Dip. di Energetica "Sergio Stecco"
Sez. Impianti e Tecnologie Industriali
Via Cesare Lombroso 6/17
50134 Firenze
Italy
tel +39-055-4796722
fax +39-055-4224137

Sergio Terzi, Marco Macchi
sergio.terzi(marco.macchi)@polimi.it,
Politecnico di Milano
Department of Economics, Industrial and Management Engineering
Piazza Leonardo da Vinci 32
20133 Milano
Italy
tel +39-02-23992803
fax +39-02-23992700

KEYWORDS

Parallel and Distributed Simulation, High Level Architecture, Inter Process Communication, Remote Factory

ABSTRACT

This paper illustrate a proposal of a framework for production plants remote control. The architecture has been developed under HLA-RTI environment, with the use of <*next event*> as paradigm for time management. XML as been used as the formalism for both information coding and non-persistent data-structuring, while persistent objects are been represented as HLA objects. The proposal framework can be used in conjunction with any COOTS simulator software, and has been tested in a local environment with Simple++ simulator software.

1 INTRODUCTION

The continuous increase of ICT applications is causing a radical change within the manufacturing industry. The effects on structures and processes are visible to everybody, for either world-wide enterprises or local industries. Summarising, this process is determined by some common issues: globalization (activities have to be managed with reference to a global environment), interconnection (coordination is possible through structured communications among remote groups) and e-manufacturing (industrial processes are computer-based controlled). Technological drivers occurring in this development are focusing significant investments, especially where headquarters and productive sites are spread in a wide territory, thus having the necessity for a narrow integration. In such this distributed environment, the main decision management process might be deployed in a distributed way, since to preserving the independence of each actor from one side, but also in order to provide a coherent creation of value. Within this distributed scenario, tools for performance analysis and supply-chain processes design/management are specifically required.

With reference to the mentioned issues, the paper aims to illustrate a proposal of a framework for production plants remote control to be adopted for the distributed management and coordination of productive nodes; in particular, this objective is achieved by the use of web-based distributed simulation. In order to provide a coherent presentation of the work, the paper will be organized as follow: § 2 introduces the research idea where the framework was developed and how the work was conducted; § 3 analyses the available technologies adopted into the proposed framework and illustrates the proposed framework; § 4 describes the preliminary test case developed according to the proposed framework; § 5 reports some conclusions and highlights further developments.

2 THE RESEARCH IDEA

The present paper aims to illustrate a preliminary research work conducted in the area of remote factory control adopting a distributed simulation approach. This work was elaborated thanks to the contributions and the knowledge of the international research group of fourth *Special Interest Group* (SIG4) of the IMS-NoE community [8], specifically interested in the establishment of a remote scheduling factory control. Within the SIG4 community, the whole research idea is generally named *Remote Factory* project.

The main idea (*Fig. 1*) of the *Remote Factory* project deals with the establishment of a virtual arena, physically provided by web-based and parallel and distributed technologies, where one industrial plant could be emulated/simulated in terms of its physical resources, while the PP&C logics are reproduced in a detached environment. Thank to this separation and to simulation technologies, over an emulated plant could be executed and tested more and more PP&C logics, in

Proceedings 15th European Simulation Symposium
Alexander Verbraeck, Vlatka Hlupic (Eds.)

ISBN 3-936150-28-1 (book) / 3-936150-29-X (CD)

order to identify the best solution using a kind of a benchmarking approach. In such a way, PP&C experts of the enterprise headquarter could be able to identify *a priori* PP&C solutions for each industrial plant and for the whole SC, avoiding inefficient local decisions[1].

The present paper, in particular, concerns with a preliminary research work freely carried on by two SIG4 members in order to investigate one efficient solution for the adoption of the distributed simulation approach: University of Florence (UNIFI) and Politecnico di Milano (POLIMI).

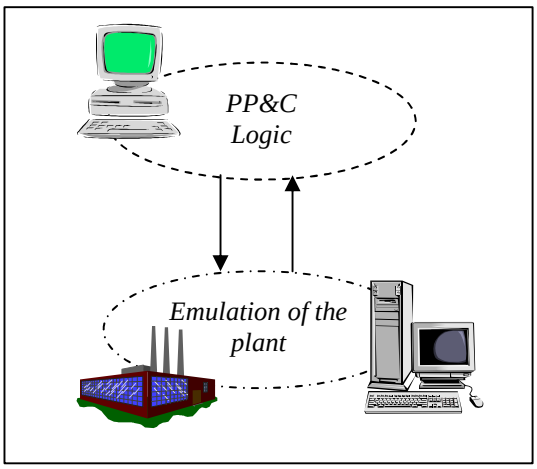

Figure 1 – Remote Factory idea

The developed distributed architecture was tested on a simple industrial test case, where one industrial plant was emulated in a physical model (PM) connected, in a distributed simulated environment, to a logical control model (LM), based on the protocols-negotiation multi-agent logic by Solberg and Lin [9].

3 THE PROPOSED ARCHITECTURE

As mentioned, the *Remote Factory* idea deals with the adoption of a PDS (*Parallel and Distributed Simulation*) environment, where two separated models (PM and LM) could interact.

3.1 Requirements for an architecture for remote factory control

Establishing a remote factory control means to apply a remote management over a production plant, managing PP&C decisions. Therefore, a remote control means, at first, to define which kind of information might flow from one *Control System* (reproduced into a logical model - LM) to the *Production Plant* (emulated in a simulation environment) and, if it is needed, vice versa (*figure 2*).

Physically, a control system defines, using its internal rules and logics, which kind of *production plan* might be performed by resources of the plant (e.g. work-centres, docks, lines) in order to satisfy a due performance (e.g. due-date timing). Work of the control system is to define job scheduling (sequencing, loading, dispatching), communicating its taken decisions to plant resources in terms of *Tasks* to be performed (e.g. "Job X in machine Y, for Z time-units"). Defined *Tasks* can be communicated one or more times, depending to the scheduling system ontology. In fact, traditional scheduling tools elaborate only one general *production plan* for a due production time period (e.g. for one shift, or one day). On the contrary, advanced scheduling solutions (e.g. *Multi-Agent Systems* MAS – see par. 4, or *Genetic Algorithm* GA) try to continuously elaborate a new *production plan* following what happens into the plant, re-defining scheduling *tasks*.

By the production plant side, a different kind of information might be achieved, corresponding to the resources status description. This information is required for setting up the scheduling algorithms of the traditional tools, while the most advanced solutions (e.g. MAS) need it continuously, in order to follow up production plant history.

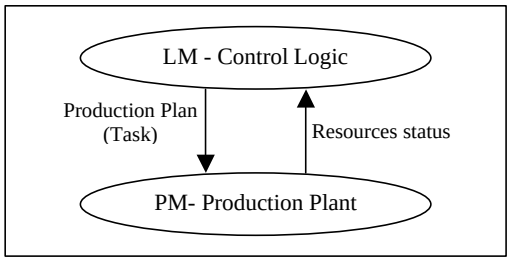

Figure 2 – Remote control information exchange

This way, the needed architecture for remote factory control might:

(i) enable a distributed (simulation) environment for remote management,

(ii) consider the different kind of information flows,

(iii) provide to each model (PM and LM) the requested elements (in terms of IN and OUT flows),

(iv) be able to manage diverse sort of models in terms of plant dimensions (number of resources), for PM, and scheduling ontology, for LM.

In the next paragraph, the proposed architecture will be illustrated in terms of technological solutions.

3.2 Technical foundations of the architecture

The needed PDS distributed environment was identified in HLA (*High Level Architecture*, [12]). As known, HLA is the most important PDS framework, recently defined as a IEEE standard, that was originally developed by the U.S. Department of Defence for

[1] The *Remote Factory* idea deals also with problems different from the remote headquarter control; in particular, at the present the main interest of the *Remote Factory* project (and of the SIG4 group) deals with the establishment of such a benchmarking service for solving the dichotomy which afflicts the world of scheduling research, where PP&C experts are totally detached from industrial reality; other information about this could be read in [8].

military purposes. Within the HLA framework, a distributed simulation is accomplished through a "federation" of concurrent "federates" (distributed models), interacting between themselves by means of a shared data model, specified in a proprietary language (OMT - *Object Modelling Template*) and federation services (basically time and data distribution management services). The federation services are provided by the *Run Time Infrastructure* (RTI) software tool, based on the HLA interface specifications. HLA has been chosen instead of other framework like CORBA [1], RMI [7] or DEVS [3] because of its robustness and the mature time management approach. Moreover, HLA has been adopted by UNIFI and POLIMI in a previous project [10]. Simple++ [5] has been chosen as simulation software, either for the physical model than for the logical one. Simple++ has been chosen because of its diffusion among COOTS (*Commercial Off Of The Shelves*) simulation tools, representing a typical environment that a future Remote factory user could adopt, with a complete support for object oriented programming and a user-friendly interface.

As known, nowadays totally HLA-compliant simulator's commercial software doesn't exist. So, it's not advisable to define an architecture where a totally HLA-compliant simulator is needed, because this choice would force the use of a specific simulator written in C++ or Java, and not a commercial tool. For these reasons the introduction of a component between the simulator and HLA environment was needed. The realization of this add-on could be done according two ways. The first one can be summarized in the definition of a *Delegated Simulator* module. This module is responsible for all the logic of information exchange between federates. The second solution proposes the introduction of a software "living" between the simulator and the RTI. This software, called *Proxy*, has the responsibility to guarantee the communication between the RTI environment and the simulator, and vice versa[2]. For this work, the second way has been chosen, with the use of a *Proxy*, thanks to the flexibility that it provides. The *Proxy*, written in java, was responsible for the information exchange between the simulator and the RTI. While the simulator has a synchronous way to communicate by TCP-IP, RTI has an asynchronous way: the *Proxy* had to store information coming from RTI and transmit it to the simulator as soon as possible and vice versa.

A clear separation from information regarding persistent objects (i.e. SM work centres' state) and not-persistent objects (i.e. production plan) has been done. While the firsts have been implemented as RTI objects, instantiated at the beginning of the simulation and destroyed at the end, the second one has been developed as HLA interactions. This choice allows an easier management of the time, with an improvement of performances in comparison to an architecture without interactions and a correct time-sequence information exchange.

Moreover, this proposal aims to differentiate the communications from the PM to the LM and vice versa. In fact, while the firsts ones have been transmitted as specific values of the HLA objects' attributes, the second ones have been implemented with HLA interactions, with the use of XLM as the formalism for both information coding and not-persistent objects data-structuring. In the proposed architecture, XML is used in order to communicate the production plan (from the controller to the plant), production executions and statistics (from the plant to the controller). For this reason, a C++ library for coding and encoding XML strings has been written and loaded into Simple++.

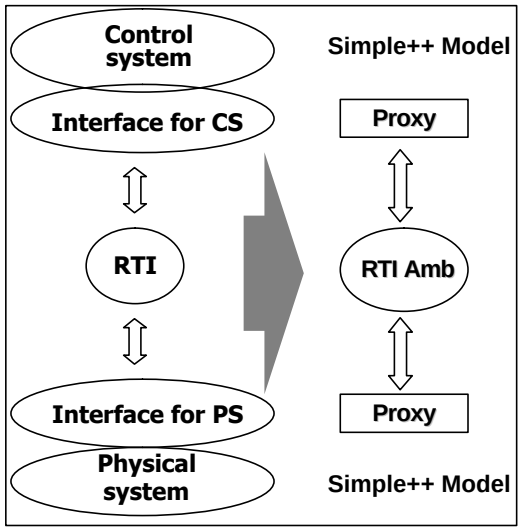

Figure 3 – Overall vision

An overall vision of the architecture is summarized in figure 3.

The XML schema used in order to communicate *production plan* is reported in figure 4.

```
<?xml version="1.0" standalone="yes"?>
 <task>
  <load>
   <lot_ID>1</lot_ID>
   <processor_ID>54CE</processor_ID >
<start_time>1:00:00.0000</start_time >
   <duration>10:00:00.0000</duration>
   <job_ID>8</job_ID>
  </load>
…
</task>
```

Figure 4 – XML schema for production plan

The use of XML inside an HLA environment extends generality of contents of messages; moreover, adding more lot-related information would be very easy.

[2] More information about "Delegate Simulator" and "Proxy" can be found in [11]

4 THE TEST CASE

As a test-case, we adopted an advanced-scheduling MAS solution, remotely controlling a shop-floor. Thus, the test-case was composed by two main components: (i) the shop-floor plant simulation, and (ii) the shop-floor MAS control logic, implemented with Simple++.

This architecture will be described starting from the shop-floor, and dividing it into three areas: (i) the shop-floor structure, (ii) the shop-floor control flows and (iii) the shop-floor control execution.

4.1 The shop-floor structure

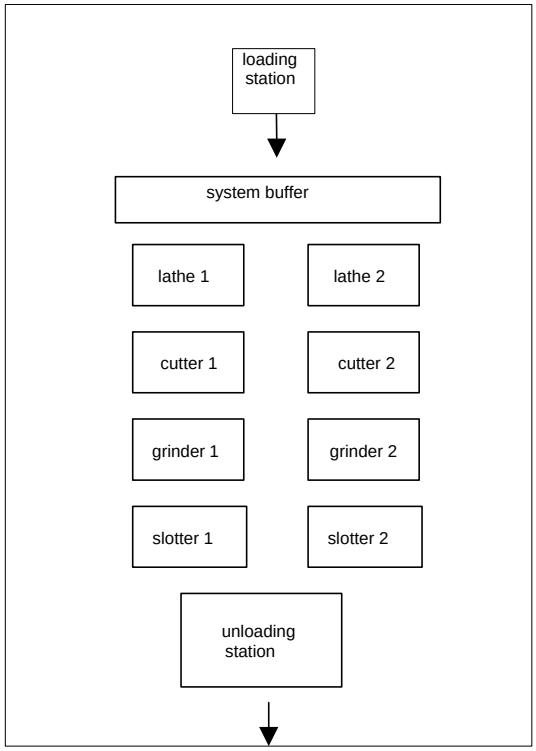

Figure 5 – Shop flow structure

The shop-floor is composed by 8 stations, grouped two by two. Each 4-station lay-out is dedicated to a specific process, as depicted in Fig. 5.
In order to implement reusable and modular architecture, some general-purpose classes were developed within the PM. In fact, the class *Processor* can represent four different objects:

- a buffer in,
- a buffer out,
- a native processor, able to describe any real work-centres,
- a data storage, able to store information about shop-orders execution.

This class was used to implement all the resources available in the PM of the described test-case, as well as the "*system buffer*", that is a virtual buffer where inactive jobs are moved. Other classes were used in order to generate entities (*Job* and *Part*) and manage material flows.

4.2 The shop-floor control flows

The PM is able to receive a *production plan* and to storage it in the data storages of the work-centres. Then, each processor analyzes the work-order according to a FIFO logic. Moreover, statistical information about job completion is stored to be transmitted later to the LM.

4.3 The shop-floor control execution

According to the proposed architecture, two data flows have been identified: the tasks flow from the PM to the LM, and the PM system *events* (i.e. resource status) from the LM to the PM.

Any occurrences relevant to production scheduling (i.e. set-up completed, start of material loading, end of processing, breakdown/failures of work-centre, maintenance beginning, restoring time) were stored as an "event". This information were then required from the LM, in order to either re-schedule or confirm the *production plan*.

In order both to minimize clock stop and to avoid inconsistence states of the PM, the information transmitted during a simulation run has been minimized. All the information used for statistics analysis were stored into the PM and communicated to the LM at the end of each run, while all the productions parameters needed for scheduling were recorded into the logical model. The flow from the PM to the LM contains only the necessaries information of the jobs, while the other flow contains only the update state of the status's attribute of each machinery.

4.4 Example of execution

At the beginning of the simulation run, the proxy creates a number of HLA objects of type "processor" equal to the number presented in the physical model. During the simulation run, HLA objects are synchronized with the real state of the PM by the proxy. This is done by a synchronous socket between the PM and the proxy. Every time a job is executed, the simulation clock is stopped, and socket is opened in order to communicate the new state of the objects. Then, proxy communicates this updating to the other federators with HLA <*next event*> time-management logic. A typical step of the execution's process could be described as follows.

Every time a work activity is completed, an event that stops the simulation clock occurs. The updated state of the PM is communicated to the Proxy, that stores it temporary, and then updates the RTI environment. Since the controller subscribed the needed objects at the beginning of the simulation run, consequently RTI delivers information to the LM. The LM achieves all the information to start the negotiation for the successive working for the job that caused the stop event. The scheduling process can be activate also by a fictitious event generated by the PM, in order to simulate the dynamic of the end of a scheduling

process. This is done with the introduction of a system class (system processor), that is programmed in order to generate an event every time a negotiation's process ends in the LM (depending on the contract net logics).

When these events occur, the PM causes the simulation clock to stop, and this operation permits the PM to receive the results of the scheduling process (task). If the negotiation process finishes without the assignment of any task, the LM communicates, as a result, another fictitious event, that will occur when the next scheduling process finishes. The tasks, that represent the job's order, are communicated to the PM with the use of RTI interaction (communication class), where the attributes (type of working, job's identification, working center's identification, beginning time of the working, working's length) are stored as an XML string. The presence of the system processor solve also a logic's weakness: during the simulation run, it's possible to have the PM completely free from processing. In this state the PM doesn't generate any event, so the simulation clock would be never stopped, in order to receive working order: in this case the fictitious event solves the problem.

The proposed architecture is based on discrete-event distributed simulation, using HLA <NextEvent> paradigm for time management, where an unimportant <LookAhead> value is associated to the LM, while a <LookAhead> proportional to the predicted time for the sheduling process is associated to the LM. As a consequence, the PM is in time advance if compared with the global simulation clock (federation time), this coinciding with the LM clock. As already said, the run is stopped when any system event happens, and status changings are published by the PM. Then, an authorization to go until the next event is requested to the RTI. RTI gives the authorization after all the messages have been delivered to the interested federates. With this logic, the PM will not stop again until the successive event, so it'll not be able to receive others tasks in the meanwhile. During this time, LM can be:

- waiting for the following event, so waiting to go on after the last production orders have been executed;
- waiting for publishing production orders the PM is going to receive.

If LM is in the state 2), the publication of the production plan will be causing LM to go on and to reach state 1). For both the states, when the LM's clock will start again, a new production's plan will be elaborated as a consequence of the PM status-changing.

Figure 6 shows the way an XML string is transmitted and the way the proxy is able to manage the *<next event>* HLA command. Figure 7 shows a log made by Simple++ about the PM production plant.

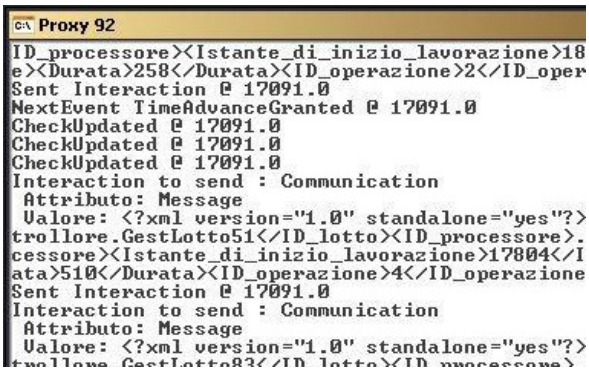

Figure 6 – Output of the proxy during the simulation

	ID_processore	Istante_di_inizio_l	Durata
Lotto52	.Milano.Controllore.Fresa2	3:12:25.0000	5:35.0000
Lotto44	.Milano.Controllore.Stozzatrice1	3:13:43.0000	3:35.0000
Lotto52	.Milano.Controllore.Stozzatrice1	3:18:00.0000	3:35.0000
Lotto53	.Milano.Controllore.Rettifica2	3:51:48.0000	8:30.0000
Lotto51	.Milano.Controllore.Tornio1	4:44:44.0000	12:00.000
Lotto42	.Milano.Controllore.Tornio1	4:56:44.0000	12:00.000
Lotto54	.Milano.Controllore.Fresa1	3:22:10.0000	5:35.0000
Lotto44	.Milano.Controllore.Tornio1	5:08:44.0000	10:00.000
Lotto026	.Milano.Controllore.Fresa1	3:30:44.0000	7:49.0000
Lotto55	.Milano.Controllore.Rettifica2	4:00:18.0000	7:05.0000

Figure 7 – Output of Simple++ order table during the simulation

5 CONCLUSIONS AND FURTHER RESEARCHES

This paper proposes a web-based, HLA-compliant simulation framework for production plans remote control. Adoption of this framework lets the user to choose any COOTS simulators for system modelling, neither binding the simulation execution nor the modelling procedures to a specific kind of technology. An important aspect of this work is the generality of the described framework. Particularly, the use of XML as the standard for the information exchange allows scalability and full unbinding to the users in system modelling. Specially, we defined a standard that differentiate communication regarding persistent objects, like work centers, from non persistent objects, like job orders.

Tests demonstrated the possibility and the conceptual correctness of the architecture. Surely, a more intensive set of tests will be useful in order to verify framework robustness. Our tests were made in a local LAN, with a single production plan and without stochastic elements. Nor the internet velocity has been considered neither the CPU performances has been tested. Even if this standard can't be considered fully tested, it solves some critical issues in the distributed simulation area. Firstly, the architecture for information exchanging among federates solves the synchronization problem, also recurring in previous works [4]. Then, the combined use of HLA interaction and XML guarantees the right sequence in the information's arriving. Last but not least, the modularity approach and object oriented

programming of each element of the system permit the full separation of the information management.

As future developments, an in-depth study of the architecture with different types of LAN would be very interesting, in order to evaluate the effect of delay in information delivering. It would also be hoped the introduction of stochasticity in the physical model, in order to evaluate it, and finally a full integration in the *Remote Factory* project idea would be the natural continuation of this work.

6 REFERENCES

[1] Zeigler, Ball, Cho, Lee, Sarjoughian, 1999, "Implementation of the DEVS Formalism over the HLA/RTI: Problem and Solution.

[2] Huang Xueqin, Miller John A., 22-26 Aprile 2001, "Building a Web-Based Federated Simulation System With Jini and XML", Simulation Symposium, 2001. Proceedings, pages 143-150.

[3] Zeigler, Kim, Buckley, 1999, "Distributed Supply Chain Simulation in a DEVS/CORBA executionenvironment", Proceeding of the 1999 Winter Simulation Conference.

[4] Carofiglio Andrea, Di Benedetto Paolo, 2001, "Il Progetto REMOTE FACTORY: Utilizzo della web based simulation per il benchmarking di sistemi di schedulazione e controllo",2001

[5] Tecnomatix Technologies
Homepage URL http://www.tecnomatix.com

[6] Bettini G., Rapaccini M., Tucci M.,- Automatic Modelling Of Manufacturing Systems With Conventional Stochastic Discrete Events Simulation Languages, Proceedings of 9th European Simulation Simposium, ESS97, Passau (D), 19-22th October 1997, pp. 411-415.

[7] Page, Moose, Griffin, 1997, "Web based simulation in SimJava using Remote Method Invocation", Proceeding of the 1997 Winter Simulation Conference.

[8] IMS-Network of Excellence (IMS-NoE, 2003), www.ims-noe.org

[9] Solberg J.J., Lin G.Y.J. (1992). Integrated shop floor control using autonomous agents. IIE Transactions, Vol. 24, No. 3, pag. 57-71

[10] Wild Web Integrated Logistics Designer, 1999-2000, Research Project funded by M.U.R.S.T.

[11] M. Tucci, R. Revetria, Different Approaches in Making Simulation Languages Compliant with HLA Specification, Proceedings of SCSC 2001, pp. 622-628, Orlando (FL), july 15–19, 2001 (ISBN 1-56555-241-5)

[12] Defence Modeling and Simulation Office (DMSO), 2001, DMSO High Level Architecture Homepage URL http://hla.dmso.mil/

[13] Cavalieri S. M. Macchi, S. Terzi, (2002), Benchmarking Manufacturing Control Systems: Development issues for the performance measurement system. In: Proceeding at IFIP Performance Measurement Workshop, Hanover, Germany

[14] XML, (2003), www.w3.org

7 AUTHORS BIOGRAPHY

Romeo Bandinelli took his Laurea Degree in Mechanical Engineering at Florence University in April 2002 discussing the thesis "Remote Factory Control with Distributed Simulation". Actually, he is a PhD student of University of Florence, Department of Energetic, Plants and Industrial Technologies Section. His current research interests are Parallel and Distributed Simulation applied to industry and supply chain context, ICT, business process re-engineering .

Sergio Terzi is a PhD student of Politecnico di Milano, Department of Economics, Industrial and Management Engineering, Laboratory of Production Systems Design and Management. He is also taking is PhD in conjunction with CRAN laboratories, University of Nancy I, France. He received his B.S. in Management Engineering degrees from the University of Castellanza in 1999 and from the same university he received his M.Sc. degrees in Economics in 2002. His current research interests are Distributed Simulation applied to industry and supply chain context, Technologies enabling Product Lifecycle Management within SME and Modelling of Production Systems.

Mario Rapaccini took his Laurea Degree with honors in Mechanical Engineering at Florence University in april 1996. He is a professional engineer since 1996. In may 2000 he achieved Ph.D. discussing the thesis Advanced tool for configuration and impact assessment of Integrated Municipal Solid Wastes Management Systems. Currently, he's assistant professor in SSD ING-IND/35. Research topics covered are: managerial economics and business organisation, ICT, simulation modelling and analysis (SM&A), business process re-engineering (BPR). He's fellow of AiIG, ANIMP, AIRO and ANIPLA.

Marco Macchi graduated in October 1997 in Management and Production Engineering at Politecnico di Milano. He is currently researcher at the Department of Economics, Industrial Management Engineering at Politecnico di Milano. His current fields of interest are Design and Automation of Manufacturing Systems, Modelling and Simulation of Manufacturing Systems, Application of Multi-Agent Systems, Computer Integration in Manufacturing Systems Engineering, Maintenance Management. He has published more than 20 papers on international journals and national and international conference proceedings. He is member of Special Interest Group on Advanced Techniques in Production Planning and Control of WG 5.7 of IFIP.

The paper is the result of a joint work conducted by the authors; Romeo Bandinelli wrote par. 3.2 and 4, Mario Rapaccini par. 1, Sergio Terzi par. 2 and 3.1, Marco Macchi contributed to par. 5.