28

# MAS AND SOA: COMPLEMENTARY AUTOMATION PARADIGMS

Luís Ribeiro, José Barata, Pedro Mendes
*UNINOVA / UNL, Quinta da Torre, 2829-516 Caparica, Portugal*
*{ldr,jab,pcm}@uninova.pt*

*This document surveys existing research in emergent concepts and technologies supporting the establishment of what are expected to be future automation systems. Multiagent systems (MAS) and Service Oriented Architectures (SOA) are currently the most promising concepts in this matter. The author's experience in the implementation and study of SOA and MAS for distributed automation systems suggests that there are substantial benefits in converging both paradigms and technologies In this context, the goal of the present work is to unveil their strengths and weaknesses and propose the unification of complementary features as a mean to provide unprecedented support to the study, modelling, design and implementation of complex distributed systems.*

## 1. INTRODUCTION

Pushed by market instability and turbulence, modern enterprises are expected to adopt innovative business methodologies to gain flexibility and agility and remain high end competitors in a globalized market.

As customers' demands rise higher in respect to diversity, exclusivity and quality of the goods, the impact of emerging requirements is deeply felt at the shop floor level. Virtually all the recent control approaches and paradigms advocate the use of distributed intelligence to maximize enterprise's agility and flexibility: bionic manufacturing systems (BMS) (Ueda 1992), holonic manufacturing systems (HMS) (Babiceanu and Chen 2006; Bussmann and Mcfarlane 1999; Gou et al. 1998; Van Brussel et al. 1998), reconfigurable manufacturing systems (RMS) (Koren et al. 1999; Mehrabi et al. 2000), (EAS) (Barata et al. 2006b; Frei et al. 2007b; Onori 2002; Onori et al. 2005; Onori et al. 2006) and evolvable production systems (EPS) (Barata et al. 2007a; Barata et al. 2007b; Frei et al. 2007a).

The industrial sector, traditionally conservative in respect to technological changes, is aware of the potential of application of such technologies and paradigms and two concepts seem to be in the research frontline: Service Oriented Architectures (SOA) and Multiagent Systems (MAS). The slow shift to SOA and MAS is accompanied by an increase in the offer of embedded tiny devices that will effectively support the establishment of intelligent automation environments. In fact, there will not be real implemented adaptive and reconfigurable systems without tiny embedded controller able to support MAS/SOA.

The success of the approaches earlier described partially depends on the possibility of having embedded MAS/SOA-ready devices. There are, however, computational limits and constraints to render the use of such technologies and devices cost effective that necessarily need to be overcome. Although is difficult to quantify and access the performance of the different approaches it is the authors' experience-based belief that there are substantial advantages in merging the best of both worlds (MAS and SOA).

The subsequent sections are organized as follows: section 2 details the concept of SOA and surveys existing research; section 3 presents a similar analysis for MAS; section 4 overviews the authors' applied research in distributed manufacturing systems that empirically suggests a favourable outcome of merging the best of MAS and SOA and in section 5 the main conclusions are discussed.

## 2. SERVICE ORIENTED ARCHITECTURES

The subject of Service Oriented Architectures (SOA) is vast, complex and multidisciplinary.

The definition of SOA is far from being agreed as a search in the literature easily confirms. Contact points between the numerous definitions frequently include the following topics:

- Autonomy: there are no direct dependencies between the services and they are structurally decoupled.
- Interoperability: is achieved by, rather than detailing the operations performed by the service provider, specifying an interface that describes the services being hosted and the interaction patterns considered.
- Platform Independence: ideally the services are described using text-based formats (XML(Bray et al. 2006), WSDL(Christensen et al. 2001), ebXML, etc). These representations are not tied to a particular computer architecture, operating systems, programming language or technology and can be easily decoded by any system.
- Encapsulation: services provide self-contained functionalities that are exposed by user defined interfaces hiding unnecessary details. By composing and orchestrating services a very complex level of functionality can be offered through a clean and simple interface.
- Availability/Discovery: the services can be published in public registries and made available for general use.

As an emerging modelling paradigm for distributed systems SOA is often confused with a wide range of networked information technologies. In this context, Web Services are the preferred mechanism for SOA implementation.

The Web Services Working group of the World Wide Web Consortium (W3C) defines Web Service as(Booth et al. 2004): "a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web Service in a manner prescribed by its description using SOAP(Box et al. 2000) messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards." The comparison to SOA is, in this case, immediate and natural as SOA was born through the convergence of several web technologies.

Although a significant share of the research in SOA focus on modelling and supporting inter enterprise relationships, there is a favourable convergence of factors that are rendering it attractive in the establishment of automated networks of devices namely: the availability of affordable and high performance embedded devices, the expansion and low cost of Ethernet based networks and its acceptance in the industrial domain, the ubiquitous nature of the Internet, the existence of lightweight, platform agnostic communication infrastructures, etc.

This has triggered several European projects in the field including industry's heavy weights. Among them one may mention as examples: SIRENA (SIRENA 2006) – award winning project that targeted the development of a Service Infrastructure for Real time Embedded Networked Applications (Jammes and Smit 2005); SODA (SODA 2006) – creation of a service oriented ecosystem based on the Devices Profile for Web Services (DPWS) framework developed under the SIRENA project; SOCRADES (SOCRADES 2006) – development of DPWS-based SOA for automation systems; InLife (InLife 2006) – including a test case that explores service oriented DPWS-based diagnosis on distributed manufacturing systems (Barata et al. 2007c; Barata et al. 2007d).

The DPWS (Chan et al. 2006), whose initial publication dates from May 2004, is especially relevant for automation environments, as it defines the minimal Web Service's implementation requirements for: secure message exchange, dynamic discovery, description and subscribing and eventing. DPWS is in this context directed to tiny low cost computing devices.

In (Papazoglou et al. 2005) a roadmap for SOA is presented and identifies the following research areas: Service Foundations, Service Composition, Service Management, Service Design and development. The main open challenges, also described in the roadmap include: dynamically reconfigurable run-time architectures, services discovery, autonomic composition of services and orchestration, self-* (self-configuring/healing/diagnosing…) services, design principles for engineering service applications. While some of these challenges can be eased by emergent standards such as BPEL4WS (Andrews et al. 2003) and WSCI (Arkin et al. 2002) others, specifically service composition in dynamically reconfigurable run-time architectures, are harder to tackle specially in heterogeneous systems, which are typically SOA's target environments.

Additionally, as stated in (Huhns and Singh 2005) most Web Services specifications do not properly support transactions which constitutes a significant implementation barrier in a wide range of systems. Other frequent limitations include:

- Code explosion – when there exists interaction between many heterogeneous services
- Reprogramming – under those circumstances the introduction of a new service with an unknown service description leads to the reprogramming of every service that interact with it.

## 3. MULTIAGENT SYSTEMS

Most definitions for agents are of functional nature and relate to their authors' background and the systems under study. Nevertheless, it is possible to isolate a common set of characteristics widely accepted(Camarinha-Matos and Vieira 1999):

- Autonomy – an agent is autonomous when it is able to act alone without help from third parties (like other agents or humans).
- Sociability – an agent must be able to communicate with other agents or even other entities.
- Rationality – an agent can reason about the data it receives in order to find the best solution to achieve its goal.
- Reactivity – an agent can react upon changes in the environment, changing its behaviour accordingly.
- Proactivity – a proactive agent has some control on its reactions basing them on its own agenda and objectives.
- Adaptability – an agent is capable of learning and changes its behaviour when a better solution is discovered adapting itself to changes in the environment.

Furthermore as detailed in (Wooldridge and Jennings 1995) several agent architectures have been categorized according to the prevalence of certain characteristics ranging from purely deliberative to purely reactive and hybrid.

While SOA is attractive for automation in part due to the possibility of seamlessly integrating systems at an interface of proxy level, the MAS contribution has been mainly related with “what lies behind the interface”.

The agent characteristics earlier pinpointed allow a Multiagent System to behave as a dynamic social network of problem solvers that provide a result that is often bigger than the sum of individual contributions.

Given the large scope of applicability of such a system numerous environment implementations (A-globe 2007; ABLE 2007; JADE 2007; JATLite 2007) have been developed focusing distinct agent models and communication mechanisms.

A consistent standardization effort in defining agent communication languages (ACL), interaction protocols and overall integration of heterogeneous agents systems has been deployed The Foundation for Intelligent Physical Agents (FIPA)

As shown in (Bussmann et al. 2004; Marik and Mcfarlane 2005; Monostori et al. 2006) automation domain denotes a potential of MAS application with effective advantages over, currently used centralized solution in respect to: feasibility, robustness and flexibility, reconfigurability and redeployability. In (Marik et al. 2005 ) is shown that at least 25% of industrial automation problems can be efficiently solved by using an agent based approach. In (Monostori et al. 2006) a thoroughly overview of the use of MAS in manufacturing is addressed and the major strategic directions are indicated. In particular, the aspects of support for emergence and embodied intelligence to support highly adaptable and reconfigurable (evolvable) manufacturing systems are emphasized. There are, however, several weaknesses in MAS when applied to industrial systems, namely: rather limited time for the decision making, constraints given by the properties of the physical equipment as well as limited number of acceptable manufacturing structures.

## 4. MERGING MAS AND SOA

Although there is a lack of measurable evidence on the strengths and weakness of both concepts some issues systematically pop-up during the implementation of distributed control systems for flexible assembly cells. Two installations have been used to test MAS and SOA: the MOFA educational shop floor (Barata et al. 2006a) (Figure 1) and the NOVAFLEX pilot assembly cell (Barata 2003; Barata et al. 2007c; Cândido and Barata 2007; Ribeiro 2007) (Figure 2).

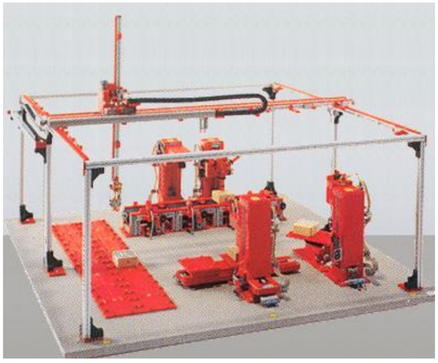

*Figure 1*. The MOFA educational shop floor

*Figure 2*. The NOVAFLEX cell Assembly cell

In either case each participant in the assembly process (robot, gripper, conveyor, tool magazine, etc) was abstracted as an agent or a service that interact among in the completion of cooperative tasks.

The implementation work was mainly pc-based (Figure 3) therefore computational power was not a constraint. However, the differences between SOA and MAS were immediately felt. Although both paradigms support the idea of distributed autonomous entities and provide an effective modeling metaphor for complexity encapsulation, SOA emphasizes contract-based descriptions of the hosted services and does not provide a reference programming model. MAS, on the other hand, support well established methods to describe the behavior of an agent. The automation environments considered are typically heterogeneous and the lack of a structured development model/template renders system designing, implementation and debugging harder. The fact that agents are regulated by internal rules that support the implementation of social behaviour is a clear advantage. This is of major importance when considering systems that undergo dynamic runtime changes which is the case of the production paradigms earlier referred. SOA, on the other hand, is typically supported by widely used web technologies and assures interoperability with a wide range of systems and can easily spawn over the internet. Most well known MAS platforms are optimized for LAN use and are restricted to compliance with well defined but less used interoperability standards

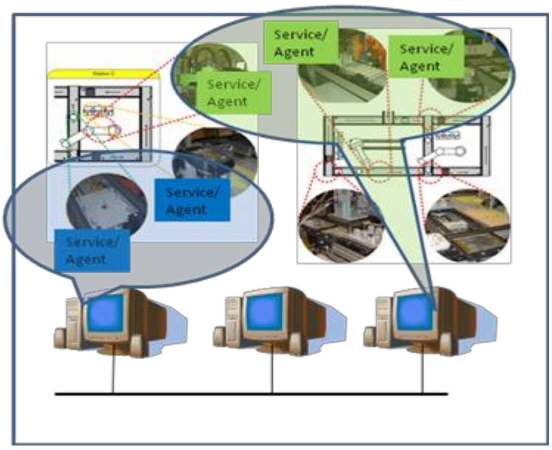


*Figure 3*. Each service or agent abstracting a different device run on a pc.

Recently in an extension to (Barata et al. 2006a), after successfully running JADE-LEAP (JADE 2007) on a GUMSTIX (GUMSTIX; Wooldridge 2002) device, it was possible to control some of the MOFA shop floor components from an agent inside that device. This experiment is closer to the systems envisioned by the future automation paradigms earlier mentioned where each participant in the process has local processing power (Figure 4).

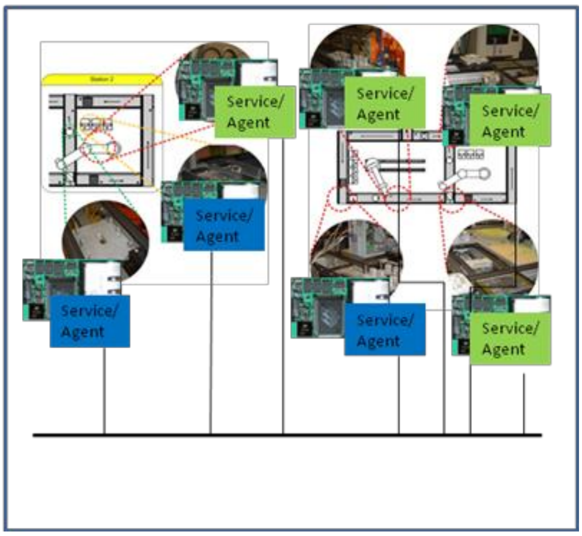


*Figure 3*. Each service or agent abstracting a different device runs on a local controller on the device itself.

Unfortunately the computational requirements of the java virtual machine in addition to the ones of the JADE-LEAP platform introduced a significant overkill that tremendously reduced the performance of the system. . Emergent frameworks like DPWS provide high performance Web Service support for devices with limited resources without constraining services implementation but the inners describing the behaviour of the service have coded from scratch.

Table 1 presents a comparative analysis between MAS and SOA. The selected characteristics included the conceptual and technology related features that are most relevant for the systems under study.

*Table 1*. Comparative Analysis between SOA and MAS

| Characteristics | SOA | MAS |
|---|---|---|
| Basic Unit | Service | Agent |
| Autonomy | Both entities denote autonomy as the functionality provided is self-contained | |
| Behaviour description | In SOA the focus is on detailing the public interface rather than describing execution details. | There are well established methods to describe the behaviour of an agent. |
| Social ability | Social ability is not defined for SOA nevertheless the use of a service implies the acceptance of the rules defined in the interface description | The agents denote social ability regulated by internal or environmental rules |
| Complexity encapsulation | Again, the self-contained nature of the functionalities provided allows hiding the details. In SOA this encapsulation is explicit. | |
| Communication infrastructure | SOA are supported by Web related technologies and can seamlessly run on the internet. | Most implementations are optimized for LAN use. |
| Support for dynamically reconfigurable run-time architectures | Reconfiguration often requires reprogramming | The adaptable nature of agents makes them reactive to changes in the environment. |
| Interoperability | Assured by the use of general purpose web technologies. | Heavily dependent on compliance with FIPA-like standards. |
| Computational requirements | Lightweight implementations like the DPWS guarantee high performance without interoperability constraints | Most implementations have heavy computational requirements |

The discussion around the fusion of MAS and SOA is not fundamentally new. However the research focus has been in enabling agents in existing systems to request, provide or manage web services (Greenwood and Calisti 2004; LIAO et al. 2004; Lyell et al. 2003; Maamar et al. 2003).

In the work developed by (Shen et al. 2005) an Agent-Based Service-Oriented Integration Framework was implemented in which web services were used as the backbone of some of the agents used in their case study. However, this integration framework is aiming business transactions and e-Business and therefore not targeting low cost computationally limited resources such as the ones under study.

In a first attempt to merge MAS and SOA in a lightweight environment for embedded devices, DPWS has been used to provide Web Services interfacing functionalities, while state control and the execution model has been borrowed from the Agent concept.

The approach taken is currently running as nearly forty independent entities, spread across several computers in the NOVAFLEX's cell, that interact in the completion of assembly tasks. In each entity Web Services are providing:

- Data encapsulation
- Communication support
- Complexity encapsulation

- Service Publishing and Discovery

Behind the Web Services interface the agent inspired code takes care of:

- Structured communication with the adequate semantics
- State control and interactions' monitoring
- Generic execution of process plans

In this manner the entities running in the system denote a memory footprint of few kilobytes and yet deliver the adequate performance in the tasks under execution while being pooled by external system that gather extra information concerning the device (documentation and life cycle parameters). Currently this code is being installed in embedded devices to test its performance.

## 5. CONCLUSIONS

It is expected that future automation environments denote a complex and distributed nature. Current approaches are on the edge of becoming obsolete in supporting the expected requirements. Researchers are currently structuring the paradigms and technologies that will support these systems and, in this context, the demand and use of network intelligent devices is growing and new systems need to be developed to accompany also that shift in technology. Multiagent systems and Service Oriented Architectures currently align as the main candidates for that purpose. The comparative analysis in the previous section unveiled contact points, weaknesses and strengths. Paradigmatically both concepts target similar systems and are supported by well structured standards and development environments. Nonetheless, each misses significant complementary functionalities provided by the other.

As the number of SOA/MAS ready devices is expected to increase the need for good tools and methods to support the integration of both worlds become mandatory. The authors are currently developing solutions for this goal.

## 6. REFERENCES


1. A-globe. "A-globe." http://agents.felk.cvut.cz/aglobe/. 2007
2. ABLE. "Agent Building and Learning Environment." http://www.alphaworks.ibm.com/tech/able 2007
3. Andrews, T., Curbera, F., Dholakia, H., Goland, Y., Klein, J., Leymann, F., Liu, K., Roller, D., Smith, D., Thatte, S., Trickovic, I., Weerawarana, S. "Business Process Execution Language for Web Services Version 1.1.", 2003
4. Arkin, A., Askary, S., Fordin, S., Jekeli, W., Kawaguchi, K., Orchard, D., Pogliani, S., Riemer, K., Struble, S., Takacsi-Nagy, P., Trickovic, I., Zimek, S. "Web Service Choreography Interface (WSCI) 1.0 W3C Note.", 2002
5. Babiceanu, R., Chen, F. "Development and applications of holonic manufacturing systems: a survey." Journal of Intelligent Manufacturing, 2006; 17: 111–131.
6. Barata, J. "Coalition Based Approach for Shop Floor Agility," PhD, Universidade Nova de Lisboa, Monte da Caparica, 2003.
7. Barata, J., Cândido, G., Feijão, F. "A multiagent based control system applied to an educational shoop floor." BALANCED AUTOMATION SYSTEMS in Manufacturing and Services, Niagara Falls, Canada, 2006a.
8. Barata, J., Frei, R., and Onori, M. "Evolvable Production Systems Context and Implications." International Symposium on Industrial Informatics, IEEE, Vigo. 2007a
9. Barata, J., Onori, M., Frei, R., and Leitão, P. "Evolvable Production Systems: Enabling Research Domains." International Conference on Changeable, Agile, Reconfigurable and Virtual Production, Toronto, Canada. 2007b
10. Barata, J., Ribeiro, L., and Colombo, A. W. "Diagnosis using Service Oriented Architectures (SOA)." International Conference on Industrial Informatics, IEEE, Vienna. 2007c
11. Barata, J., Ribeiro, L., and Onori, M. "Diagnosis on Evolvable Production Systems." International Symposium on Industrial Electronics, IEEE, Vigo. 2007d
12. Barata, J., Santana, P. F., and Onori, M. "Evolvable Assembly Systems: A Development Roadmap." IFAC Symposium on Information Control Problems in Manufacturing, Saint-Etienne, France. 2006b
13. Booth, D., Hass, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C., and Orchard, D. "Web Services Architecture, W3C Working Group Note 11 February 2004.", 2004
14. Box, D., Ehnebuske, D., Kakivaya, G., Layman, A., Mendelsohn, N., Frystyk Nielsen, H., Thatte, S., Winer, D. "Simple Object Access Protocol (SOAP) 1.1, W3C Note 08 May 2000.", 2000
15. Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E., Yergeau, F. "Extensible Markup Language (XML) 1.0 (Fourth Edition), W3C Recommendation 16 August 2006.", 2006
16. Bussmann, S., Jennings, N. R., and Wooldridge, M. Multiagent Systems for Manufacturing Control: A Design Methodology, Springer, Berlin. 2004
17. Bussmann, S., Mcfarlane, D. C. "Rationales for Holonic Manufacturing." Second International Workshop on Intelligent Manufacturing Systems, Leuven, Belgium, 177 - 184.
18. Camarinha-Matos, L. M., Vieira, M. "Intelligent Mobile Agents in Elderly Care." Journal of Robotics and Autonomous Systems, 1999; 27(1-2): 59-75.
19. Cândido, G., Barata, J. "A Multiagent Control System for Shop Floor Assembly." International Conference on Industrial Applications of Holonic and Multiagent Systems, Regensburg, Germany. 2007
20. Chan, S., Conti, D., Kaler, C., Kuehnel, T., Regnier, A., Roe, B., Sather, D., Schlimmer, J., Sekine, H., Thelin, J., Walter, D., Weast, J., Whitehead, D., Wright, D., Yarmosh, Y. "Devices Profile for Web Services." http://schemas.xmlsoap.org/ws/2006/02/devprof/. 2006
21. Christensen, E., Curbera, F., Meredith, G., Weerawarana, S. "Web Services Description Language (WSDL) 1.1, W3C Note 15 March 2001." 2001
22. Frei, R., Barata, J., Di Marzo Serugendo, G. "A complexity theory approach to evolvable production systems." Internacional Conferecene in informatics and control, automation and robotics, Angers, France. 2007a
23. Frei, R. M., Ribeiro, L., Barata, J., and Semere, D. "Evolvable Assembly Systems: Towards User Friendly Manufacturing." International Symposium on Assembly and Manufacturing, IEEE, Ann Arbor, USA. 2007b
24. Gou, L., Luh, P. B., Kyoka, Y. "Holonic Manufacturing Scheduling Architecture, Cooperation Mechanism and Implementation." Computers in Industry, 1998; 37(3): 213-231.
25. Greenwood, D., Calisti, M. "Engineering Web Service - Agent Integration." 2004 IEEE International Conference on Systems, Man and Cybernetics. 2004
26. GUMSTIX. "GUMSTIX." http://www.gumstix.com.
27. Huhns, M. N., Singh, M. P. Service-Oriented Computing: Key Concepts and Principles. Internet Computing 2005; 9(1), 75-81.
28. InLife. "Integrated Ambient Intelligence and Knowledge Based Services for Optimal Life-Cycle Impact of Complex Manufacturing and Assembly Lines." http://www.uninova.pt/inlife/, 2006.
29. JADE. "Java Agent Development Framework." http://www.jade.tilab.com, 2007.
30. Jammes, F., Smit, H. "Service-oriented architectures for devices - the SIRENA view." International Conference on Industrial Informatics, Perth, Western Australia, 140- 147.

31. JATLite. "Java Agent Template, Lite." http://java.stanford.edu/, 2007.
32. Koren, Y., Heisel, U., Jovane, F., Moriwaki, T., Pritchow, G., Ulsoy, A. G., Van Brussel, H. Reconfigurable Manufacturing Systems. CIRP Annals 1999, 48.
33. LIAO, B.-S., GAO, J., HU, J., CHEN, J.-J. "A federated multi-agent system: Autonomic control of Web Services." Proceedings of the Third International Conference on Machine learning and Cybernetics, Shangai.
34. Lyell, M., Rosen, L., Casigni-Simkins, M., Norris, D. "On software agents and Web services: Usage and design concepts and issues." Proceedings of WSABE 2003.
35. Maamar, Z., Sheng, Q., Benatallah, B. "Interleaving Web services composition and execution using software agents and delegation." Proceedings of WSABE 2003.
36. Marik, V., Mcfarlane, D. C. "Industrial adoption of agent-based technologies." Intelligent Systems 2005; 20(1): 27-35.
37. Marik, V., Vrba, P., Hall, K. H., Maturana, F. P. "Rockwell automation agents for manufacturing." International Conference on Autonomous Agents, The Netherlands, 107 - 113
38. Mehrabi, M. G., Ulsoy, A. G., Koren, Y. Reconfigurable Manufacturing Systems and their Enabling Technologies. International Journal Manufacturing Technology and Management 2000; 1: 113-130.
39. Monostori, L., Váncza, J., Kumara, S. R. T. Agent-Based Systems for Manufacturing. CIRP Annals 2006; 55(2).
40. Onori, M. "Evovlbale Assembly Systems - A New Paradigm?" 33rd International Symposium on Robotics Stockholm, 2002.
41. Onori, M., Alsterman, H., Barata, J. " An architecture development approach for evolvable assembly systems." International Symposium on Assembly and Task Planning: From Nano to Macro Assembly and Manufacturing, Montréal, Canada, 19-24.
42. Onori, M., Barata, J., Frei, R. "Evolvable Assembly Systems Basic Principles." Conference on Information Technology for BALANCED AUTOMATION SYSTEMS in Manufacturing and Services, Springer, Ontario, Canada, 2006.
43. Papazoglou, M. P., Traverso, P., Dustdar, S., Leymann, F., Krämer, B. J. "Service-Oriented Computing: A Research Roadmap." Dagstuhl Seminar Proceedings 05462 (SOC).
44. Ribeiro, L. "A Diagnostic Infrastructure for Manufacturing Systems," New University of Lisbon, Lisbon, 2007.
45. Shen, W., Li, Y., Hao, Q., Wang, S., Ghenniwa, H. "Implementing Collaborative Manufacturing with Intelligent Web Services." Proceedings of the 2005 The Fifth International Conference on Computer and Information Technology, 2005.
46. SIRENA. "Service Infrastructure for Real-time Embedded Network Applications." http://www.sirena-itea.org/Sirena/Home.htm, 2006.
47. SOCRADES. "Service-Oriented Cross-layer infRAstructure for Distributed smart Embedded devices." http://www.socrades.eu/Documents/AllDocuments/default.html, 2006.
48. SODA. "Service Oriented Device and Delivery Architecture." http://www.soda-itea.org/Home/default.html, 2006.
49. Ueda, K. "A concept for bionica manufacturing systems based on DNA-type information." PROLAMAT, IFIP, Tokyo, 1992.
50. Van Brussel, H., Wyns, J., Valckenaers, P., Bongaerts, L., Peeters, P. Reference Architecture for Holonic Manufacturing Systems: PROSA. Computers in Industry 1998; 37: 255-274.
51. Wooldridge, M., Jennings, N. R. Intelligent Agents - Theory and Practice. Knowledge Engineering Review 1995; 10(2): 115-152.
52. Wooldridge, M. J. An Introduction to Multiagent Systems, J. Wiley, New York, 2002.