42 A SERVICE-ORIENTED FRAMEWORK FOR INTEGRATION OF SHOP FLOOR SCHEDULING AND CONTROL

Kewei Li[1], Chun Wang[1],
Hamada Ghenniwa[1], Weiming Shen[1,2]
[1]*University of Western Ontario*
[1,2]*National Research Council Canada*
kli64@engga.uwo.ca; cwang28@engga.uwo.ca;
hghenniwa@eng.uwo.ca; weiming.shen@nrc.gc.ca

This paper presents an agent-based service-oriented framework to address the challenges for integrating real time shop floor scheduling and control in a multi-workcell environment. The proposed framework enables shop floor to adapt to the changes in the environment dynamically. Shop floor scheduling and control can address dynamic changes either within a workcell, such as machine breakdowns, or cooperatively amongst multiple workcells, for example to delegate a task that cannot be performed locally or to improve the performance of the workcell. A software simulation environment has been implemented to validate the proposed approach.

1. INTRODUCTION

Globalization has driven manufacturing enterprises to shed the security of mass production and shift to a new paradigm which is referred to by different names such as lean production and agile manufacturing. An agile manufacturing system needs to respond to dynamic changes in a timely and cost effective manner.

Dynamic changes can derive from either outside parties in the market, such as the supply side (representing suppliers), demand side (representing customers) or within the enterprise, such as real-time events from the shop floor. To deal with the dynamic changes within the enterprise, the monitoring and control of shop floor has to be fully integrated with manufacturing scheduling system to provide the manufacturing enterprises with the capability to survive in today's dynamic market environments. This research is concerned with integrating Real time shop floor monitoring and control system with Scheduling system. In order to achieve effective real time Shop floor monitoring and control in dynamic manufacturing environments, we propose a service-oriented integration framework including interaction mechanisms which coordinate the behaviors of scheduling system and shop floor control system in a flexible manufacturing multi-workcell environment.

Please use the following format when citing this chapter:

Li, K., Wang, C., Ghenniwa, H., Shen, W., 2006, in IFIP International Federation for Information Processing, Volume 220, Information Technology for Balanced Manufacturing Systems, ed. Shen, W., (Boston: Springer), pp. 395–404.

The manufacturing environment is typically seen as a large, complex man-made system of heterogeneous, interrelated activities (STANESCU, 2002). The close monitoring of the operational performance of various plants, manufacturing shop-floors / cells / machines, as well as their associated instrumentation and control is of increasing strategic importance. Industrial plants are complex open system, the integration of scheduling and control are based on wide-are network of heterogeneous platforms. The integration needs to address the issues such as distribution, heterogeneous, interoperability and open system environment.

This paper presents our research work on the integration for real time shop floor monitoring and control. We apply agent technology to build up a multi-agent system for a single workcell, and Web services technology for the integration of distributed multiple workcells (or plants). The rest of the paper is organized as follows. Section 2 provides a literature review. Section 3 proposes a agent-based service-oriented framework for integration of shop-floor scheduling and control. Section 4 presents our prototype implementation. Section 5 concludes the paper.

2. A REVIEW OF INTEGRATION APPROACHES

2.1 MIDDLEWARE BASED APPROACHES

The integration of the scheduling and control system is a form of distributed information system. There are kinds of ways trying to address the problems and constraints existing in the design and implementing of distributed information systems. The system architecture may vary from 1-tier to N-tier designed in either a bottom-up or top-down manner. Their communication patterns are between synchronous and asynchronous interaction.

Distributed information systems have evolved in response to improvements in computer hardware and networks. Middleware constitutes the basic infrastructure behind any distributed information system (Alonso, 2004). Middleware facilitates and manages the interaction between applications across heterogeneous computing platforms. Conventional middleware platforms can be used for the integration for manufacturing system, such as RPC and related middleware, object brokers, object monitors and message-oriented middleware (MOM).

EAI can be seen as a step forward in the evolution of middleware, extending its capabilities to cope with application integration. When the systems involved were compatible and comparable in their functionality and did not involve many platforms, middleware could be used with out further ado to integrate the servers. Unfortunately, for more ambitious projects, plain middleware platform was not enough. The main limitation was that any concrete middleware platform makes implicit assumptions about the nature of the underlying systems. When these systems are very different in nature and functionality, using conventional middleware to integrate them becomes rather cumbersome, and in some cases simply infeasible. Message brokers may be the versatile platform for Enterprise Application Integration. Message brokers are direct descendants of the platforms for message oriented middleware. They are derived from the new requirements posed by EAI, in terms of supporting the integration of heterogeneous, coarse-grained enterprise applications. Message brokers can address one of the enterprise process automation problems: that of hiding the heterogeneity and the distribution of

enterprise systems to provide a uniform view to the applications that integrate those systems. Examples of leading commercial implementations of EAI platforms today are Tibco ActiveEnterprise, BEA Weblogic Integration, WebMethods Enterprise, and WebSphere MQ. Integration through a message broker entails a number of benefits which are typically characterized as lower development cost, lower opportunity costs and lower maintenance effort. Despite these advantages, message brokers are not a panacea for all application problems. Indeed, there are many drawbacks to implementing EAI solutions using message brokers. The main issue is that software licenses are extremely expensive. Besides licensing costs, companies need to invest in the training of IT personnel and acquire the resources necessary for the installation and operation of the tool. These reasons often discourage small and medium enterprises from adopting EAI platforms, and even from performing the integration at all.

Workflow management systems are the tools used to make the integration logic explicit and more manageable. Workflow management systems (WfMSs) tackle the other side of the application integration problem: that of facilitating the definition and maintenance of integration logic. Examples of leading commercial workflow systems include WebSphere MQ Workflow by IBM, Vitria BusinessWare, Tibco BPM, BEA WebLogic Integration, and Microsoft BizTalk Orchestration.

On the one hand, the type of integration discussed so far has been implicitly limited to LANs. As it is easy to imagine, the problems of application integration are significantly amplified when we consider integration across enterprises and over the Internet. On the other hand, although current technology provides tremendous benefits, existing platforms are still very expensive and quite complex to use and maintain.

2.2 Web-Based Approaches

The increasing use of the Web as a channel to access information systems forced middleware platforms to provide support for Web access. This support is typically provided in the form of *application servers* (SUN's J2EE or Microsoft's .NET based). Application servers are equivalent to the middleware platforms discussed in earlier section. The main difference is the incorporation of the Web as a key access channel to the services implemented using the middleware.

At the presentation layer, application servers conceptually resemble conventional middleware. The functionality provided is similar to that of CORBA, TP monitors, and message brokers. The support for the presentation layer and for the document as the basic unit of transfer is what differentiates application servers from conventional middleware.

Firewalls present a special challenge to integrating inter-enterprise systems across the Internet. Almost all forms of communication usually employed by conventional EAI products cannot traverse a well-configured firewall. The widely accepted solution to get through the firewall is done by a technique known as tunneling, whereby protocols which would be blocked by the firewall are hidden under protocols that accepted by the firewall. Tunneling is used by conventional middleware systems (e.g., GIOP/IIOP over HTTP) to bypass firewalls.

The widespread use of the Web made it very difficult to standardize all forms of Internet exchanges. In the context of the Web, the answer to this problem is the

eXtensible Markup Language (XML). XML addresses the data representation problem by focusing on the syntax, rather than the semantics, of the documents exchanged. However, by providing syntax along with parsing and validation tools, XML lays the foundation on which semantics can be defined.

2.3 Agent-Based Approaches

Multi-agent systems have been the subject of massive amounts of research in recent years and are beginning to find their way into commercial applications (Shen, 2001). An exhaustive overview is beyond the scope of this paper, but essential pointers include (Wooldridge, 2002) and [Luck, 2003].

Early research work on agent-based intelligent manufacturing scheduling and factory control was reported almost 20 years ago (Shen, 1999). Agent Based Manufacturing has become a new paradigm for next generation manufacturing systems (Shen, 2001). Researchers have been applying agent technology to manufacturing enterprise integration and supply chain management, manufacturing planning, scheduling and execution control, materials handling and inventory management. Researchers have conducted an extensive literature review and published the review results already (Shen, 1999; Shen, 2001.).

Many researchers are probing into solutions for enterprise integration and some reached the conclusion that the agent technology provides a nature way to realize enterprise integration effectively. Agent-based enterprise integration has been a very active research area recently. In addition to significant academic researches, some projects have attracted active industrial participation and developed industrial applications. In most projects, software agents are used to encapsulate existing legacy software systems using various middleware approaches.

Security and privacy are critical issues in implementing Internet-enabled agent-based manufacturing systems. Socket communication is actually used for inter-agent message exchange. Although secure sockets could be used to enhance the communication security, an implementation in a real shop floor may have some difficulties because of firewalls;

The agent technology is still considered to be "nice-to-have" in industrial applications by industrial people. It seems to be still a long way to go before it is considered to be "must-have". We believe that the agent technology must be integrated with other technologies such as Web based technologies (including Web Services and Semantic Web) and Grid computing for its wide and successful applications in industry in a near future.

Web services are emerging as a major technology for achieving automated interactions between distributed and heterogamous applications (Tsalgatidou, 2002). Web Services technology is part of the SOC (Service Oriented Computing) paradigm and can be considered as an implementation of the SOC model providing Web applications with a loosely coupled integration approach (Shen, 2005). The W3C (The World Wide Web Consortium) defines a Web service as "... 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 – Web Service Definition Language). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically

conveyed using HTTP with an XML serialization in conjunction with other Web-related standards." (W3C, 2004).The advantage of Web services have already been demonstrated and highlight their capacity to be composed into high-level business processes (Benatallah, 2002). It is argued that composition via service interconnection allows more sophisticated services and applications to hierarchically constructed from primitive ones (Maamar, 2003). Web service can be independent as much as possible from specific platforms and computing paradigms, in addition, it can be easily compassable.

The integration using software agents and Web Services technologies together is able to avoid the weakness of each individual technology, while reinforcing their strengths. It can be a good approach to address the issues which remains in the real time shop floor monitoring and control.

3. SERVICE-ORIENTED FRAMEWORK FOR INTEGRATION OF SCHEDULING AND CONTROL

For the scheduling and control aspect, developing multi-agent systems is carried out in cooperative distributed multi-workcell manufacturing environments. In this context, each work cell (or plant) have a designated autonomous scheduling system based on the local constraints and objectives of individual manufacturing resources (represented by software agents), and is directly connected to the shop-floor's control level.

Based on the analysis in the previous section, we propose this approach, and its name is Service-Oriented Framework for Integration of Scheduling and Control (FISC). It has two parts: a Platform for Integration of Scheduling and Control (PISC) and a Mechanism for Integration of Scheduling and Control (MISC). Technically, the FISC is an abstract extensible skeleton with a set of meta-components and a higher-level management component. According to the particular Multi-workcell scenarios, the FISC can dynamically extended and deployed to multiple work cells, adjust or re-configure itself to initialize the schedule and control parameter, to accept new order or outsource corresponding job to remote work cell in response to the dynamic event happened in the shop floor level (such as machine break down). The PISC is designed to improve the adaptability of integrating platform in the heterogonous distributed multi-workcell environment. The purpose of the MISC is to provide the adaptability from the interaction protocol point of view. The relations among the FISC, MISC and PISC are demonstrated in Figure 1. The PISC includes five parts: the agent factory, general utility, schedule and control platform, web service run time environment, web service interface. The MISC has one major component: the knowledge base. Further detailed description will be given in the following subsections.

We adopt the scheduling model, shop floor control model, Coordinated Intelligent Rational Agent (CIR-Agent) model (Hamada, 2000), and the web service integration approach as the base to construct the FISC. The CIR-Agent model is adopted to design the work cell resource level agent (such as machine agent and operator agent). The web service integration approach is used to construct the multi-workcell environment, and the PISC is considered to be settled on it. The scheduling and control model are utilized to design the mechanism related to the protocols between agents (e.g., protocol between scheduler agent and controller agent,

protocol between control agent and resource agent) and protocols between multiple workcells (e.g., Protocol used for one work cell out source job to a remote work cell). The web service coordination model and service composition model are used as the foundation when designing the algorithms for the multi-workcell integration mechanism determination and composing mechanism in both the PISC and MISC.

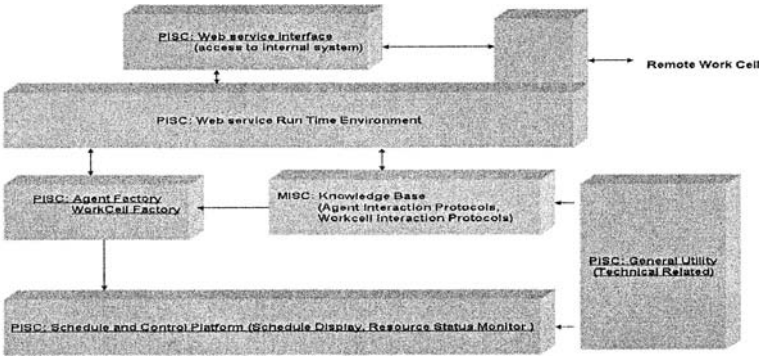

Figure 1 - Service-Oriented Framework for integration of scheduling and control

3.1 Framework for Integration of Scheduling and Control (FISC)

A framework is an extensible high-level structure including a set of reusable components, which can be dynamically reused and instantiated to build a concrete application. The main advantage of a framework is that it can successfully decrease the systematic complexity without sacrificing the flexibility and adaptability of the generated application.

The main goal of the FISC is to provide an open, flexible, Service-oriented and FIPA-compliant (FIPA, 2005) framework to improve the adaptability of the integration platform in the complex flexible manufacturing system (such as multiple workcells in wide area network). In order to achieve the goal, the FISC is structured as two parts: the Platform for Integration of Scheduling and Control (PISC) and Mechanism for Integration of Scheduling and Control (MISC), by which it can improve the adaptability at two levels: the integration platform and participated workcells. In this paper, the adaptability of an integration platform indicates that it can support integration in a heterogeneous, distributed and Wide area network environment For a participated work cell (including the work cell controller and shop floor level resources), the adaptability means that a new instance of controller or resource agent can be instantiated and configured to join the flexible manufacturing system. The PISC provides a scalable and flexible integration platform with the capabilities to schedule, monitor and control in a multi-workcell manufacturing environment. The MISC will include mechanism for protocol selection, service coordination and service composition.

3.2 Platform for Integration of Scheduling and Control (PISC)

The overall usage of the PISC is to provide a flexible platform for scheduling and control integration. In terms of multi-workcell environment, when a new work cell need to be integrated, the platform can be easily deployed on the new work cell, what need be done by the user is to configure the capability of the resource agent through GUI, then register with the work cell controller agent, the new work cell then is integrated into the framework. For the interaction between work cells which happened at the web service level or the interaction between scheduler, controller and resources agent in a single work cell which happened at the agent communication level, MISC is used to deal with the coordination and composition function.

3.3 Mechanism for Integration of Scheduling and Control (MISC)

As briefly mentioned previously, the Mechanism for Integration of Scheduling and Control (MISC) is designed to realize all key functions, such as the function for service coordination, function for service composition and function for agent negotiation protocol selection.

4. PROTOTYPE IMPLEMENTATION

Figure 2 demonstrates a sample deployment of the prototype in a two work cell environment. Currently the cells are simulated in a computer environments, not connected to the real machines directly. The used key implementation technologies are listed below:

- System platform: Linux / Windows
- Programming language: Java 2 Platform Standard Edition 1.4.2 (J2SE)
- Web Service middleware tool: Java WSDP 1.6
- Web Service container: Tomcat 5.0 for Java WSDP 1.6
- Multi-agent Platform: JADE

Within the scope of integration, the dynamic state change of the multi-workcell control system is modeled as a Finite State Machine as shown in Figure 3.

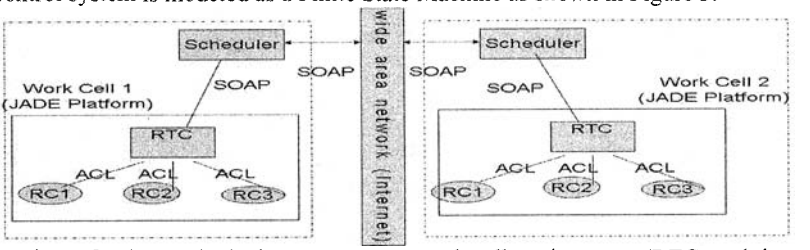

Figure 2 - A sample deployment on two work cell environment (RTC: real time controller; RC1: resource controller 1; RC2: resource controller 2; RC3: resource controller 3; they exist as agents. SOAP: Simple Object Access Protocol; ACL: Agent Communication Language)

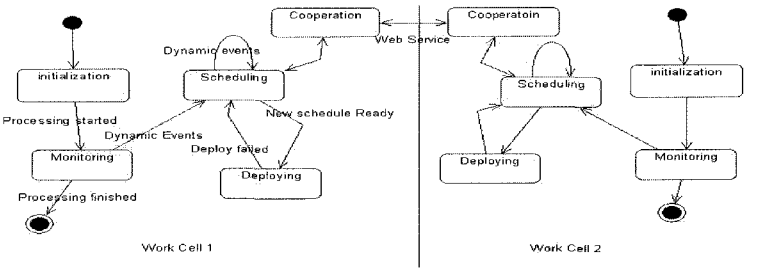

Figure 3 - Finite State Machine model of two work cell control

Figure 4 depicts the multi-workcell service protocol. In this figure, we show two work cells. Real Time Controller of work cell 1 provides Report machine capability service and Report dynamic event service to the Scheduler of Work Cell 1. Scheduler provides Deploy Schedule Service to Real Time Controller. All of these happen within one work cell.

Under certain circumstances, scheduler of Work Cell 1 receives some Dynamic event such as machine down, then scheduler found it is not able to redeploy schedule within this work cell because the resources is not available anymore. Then Scheduler 1 invokes Distribute Job Service of Scheduler 2 in Work Cell 2, assign the needed job to Scheduler 2, Scheduler 2 then deploy schedule within Work Cell 2. This will achieve the goal to distribute work load between multiple workcells.

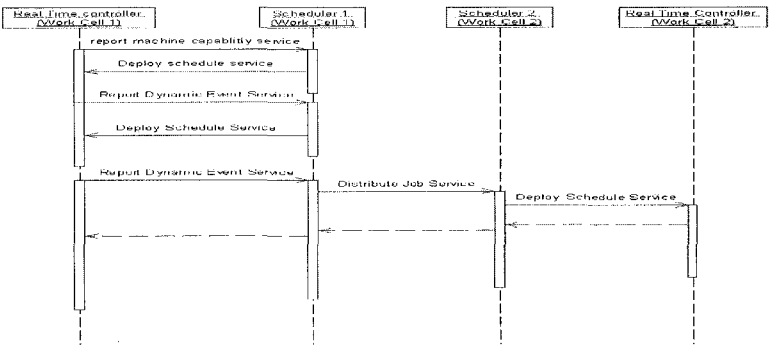

Figure 4 - Two Work Cell's Service protocol in the sample prototype

An implemented working scenario can be represented as follows:

- There is a Work Cell including a RTC Agent, three Machine Agents and one Operator Agent, these agents sit on an Agent platform (JADE Platform 1), a Scheduler Agent which is designated for this Work Cell is running on a remote Agent Platform.
- RTC Agent look up the service in UDDI, and find this designated Scheduler, then bind with the Scheduler.

- RTC Agent report the Work Cell's capabilities to the Scheduler through Web service.
- Scheduler deploys the Work Cell's schedules to RTC through Web Service.
- Within the Work Cell, RTC Agent deploys schedule to different resource Agent through ACL Messages. (Agent communication)
- RTC Reports Work Cell disturbance (such as machine down) in real time to Scheduler through Web service.
- Scheduler receives the disturbance, generates new schedule, and deploys new Schedules to RTC Agent through Web service.
- RTC agent deploys new schedules to different Resource Agent through ACL Messages in real time.
- Work cell 1 resources down, not available anymore; Scheduler 1 assign Job to scheduler 2 in work cell 2 through Web Service. Work Cell 2 accepts the job. This is an example of the multi-workcell scenario.

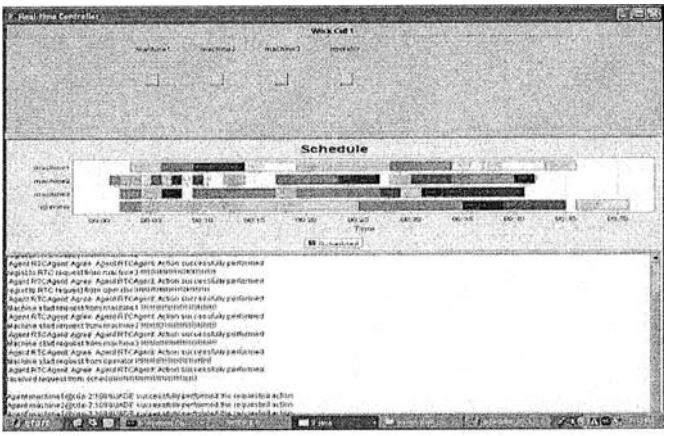

Figure 5 - RTC screen shot (all resources are working properly)

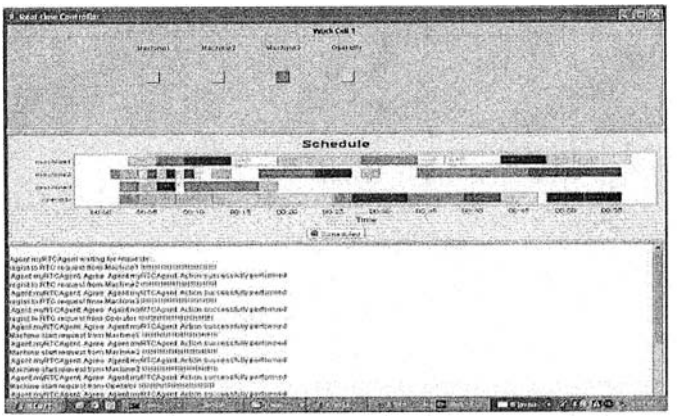

Figure 6 - RTC screen shot (Machine 3 is down)

The above two figures are the screen shot of the RTC (Real Time Controller). The RTC monitors and controls 3 machines and 1 operator. Figure 5 shows that every resource' status is well and their related schedule; Figure 6 shows machine 3 is down at certain moment and the re-deployed schedules.

5. CONCLUSIONS AND PERSPECTIVES

This paper is concerned with developing a framework using software agent and web service technologies for the integration of real-time shop-floor monitoring and control in multi-workcell manufacturing environment.

Web Services offer fundamentally new ways of doing business through a set of standardized tools, and support a service oriented view of distinct and independent software components interacting to provide valuable functionality. The supporting of interoperation makes Web Service a good approach to integrate heterogeneous systems. A Web Service infrastructure solution provides a rapidly deployable and re-configurable cooperative distributed manufacturing environment.

The experimental deployment prototype of the service-oriented integration framework for scheduling and control has shown that it is a promising approach for building up information integration system in flexible manufacturing environments.

6. REFERENCES

1. Alonso, G, Casati, F, Kuno, H, Machiraju, V. Web Services: Concepts, Architectures and Applications, Berlin : Springer, c2004. pp. 67.
2. Benatallah, B, Casati, F. (Guest Editors). "Special Issue on Web Services". Distributed and Parallel Databases, Kluwer Academic Publishers, 2002; 12(2-3).
3. FIPA. The Foundation for Intelligent Physical Agents. http://www.fipa.org/
4. Ghenniwa, H, Kamel, M. "Interaction Devices for Coordinating Cooperative Distributed System". Journal of Intelligent Automation and Soft Computing, 2000; 6(2): 173-184.
5. Luck, M, McBumey, P, Preist, C. Agent technology: Enabling Next Generation Computing. http://www.agentlink.org/roadmap, 2003
6. Maamar, Z, Sheng, QZ, Benatallah, B. "Interleaving web services composition and execution using software agents and delegation". In AAMAS'2003 Workshop on Web Services and Agent-based Engineering, 2003.
7. Shen, W, Li, Y, Hao, Q, Wang, S, Ghenniwa, H. "Implementing Collaborative Manufacturing with Intelligent Web Services". Proceedings of CIT 2005, 2005; pp. 1063 – 1069.
8. Shen, W, Norrie, DH. "Agent-based systems for intelligent manufacturing: A state-of-the-art survey". Knowledge and Information Systems 1999; 1(2): 129-56.
9. Shen, W, Norrie, DH, Barthes, JP. Multi-Agent Systems for Concurrent Intelligent Design and Manufacturing. London: Taylor and Francis; 2001.
10. Stanescu, AM, Dumitrache, I, Curaj, A, Caramihai, SI, Chircor, M. "Supervisory control and data acquisition for virtual enterprise". International Journal of Production Research, 2002; 40(15): 3545-3559.
11. Tsalgatidou, A, Pilioura, T. "An Overview of Standards and Related Technology in Web services". Distributed and Parallel Databases, 2002; 12(2-3): 135-162.
12. W3C Web Service Architecture Working Group. Web Service Architecture Recommendation. http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/. 2004.
13. Wooldridge, M. An Introduction to Multi-Agent Systems. Wiley and Sons, 2002.