Agent Based Web Service Composition In The Context Of A Supply-Chain Based Workflow

Bastin Tony Roy Savarimuthu, Maryam Purvis, Martin Purvis
Department of Information Science
University of Otago
P O Box 56, Dunedin, New Zealand
{tonyr, tehrany, mpurvis}@infoscience.otago.ac.nz

ABSTRACT
With the advent of Web Services, more and more business organizations make their services available on the Internet through Web Services and also use other services that are available on the corporate Intranet. From the viewpoint of workflow systems, these freely available Web Services and the proprietary intranet-based services should be integrated into individual businesses for their day-to-day workflows. Businesses that use Web Services not only provide the services to their customers but can also use Web Services to customize their internal processing, such as online order placement for raw materials. In this paper we describe the architecture of our agent-based workflow system that can be used for Web Service composition. In the context of an example from the apparel manufacturing industry, we demonstrate how Web Services can be composed and used.

Categories and Subject Descriptors
I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence – *Multiagent systems;* H.4.1 [**Information Systems Applications**]: Office Automation – *Workflow management;*

General Terms
Design

Keywords
Web Services, Multi-agent systems, Workflow systems

1. INTRODUCTION
Most of the commercially available workflow management systems do not offer sufficient flexibility for distributed organizations that participate in the global market. These systems have rigid, centralized architectures that do not operate across multiple platforms ([2], [6]). Employing a distributed network of autonomous software agents that can adapt to changing circumstances would result in an improved workflow management system (WfMS). In the past, WfMS [8] were used in well-defined activities, such as manufacturing, where the processes tend to be more established and stable. But in the current climate WfMS may be used for more fluid business processes, or in processes involving human interactions, such as the software development process. In such situations, it is not always possible to predict in advance all the parameters that may be important for the overall processes. This gives rise to the need of adaptive systems. Our previous works ([1], [7], [10]) describe the advantages of our agent-based framework JBees, such as distribution, flexibility, and the ability to dynamically incorporate a new process model.

The current problem with WfMSs is that they do not integrate the available Web Services very well. With the emergence of Web Services most of the e-commerce transactions that involve both B2B and B2C interactions are available on the Web. The current trend in WfMSs is the seamless integration of these services with the existing workflow infrastructure. In particular we are interested in focusing on Web Service Composition [27]. Web Service Composition refers to the integration of more than one Web Service to realize business functionality. For example, consider the purchase of a book from a online Web Service. The online Web Service authenticates the customer's credit by contacting the customer's bank Web Service. If the customer has enough credit to buy the book, the delivery Web Service has to be contacted. To realize the book order Web Service many Web Services should be composed into a logical unit and executed by the underlying workflow system.

In this paper we explain the architecture of integration of Web Services with our workflow system and also, using an example from the supply chain management perspective, we explain how Web Service Composition is achieved. Supply chain management is one of the key areas in production workflows and we believe that Web Services can be used to improve the turnaround time. The background of our work is described in Section 2. A brief description of our agent-based framework is given in Section 3. In Section 4 we explain different ways of Web Service Composition, namely static and dynamic composition. Using a supply chain example from the apparel manufacturing industry, we explain how Web Service composition is achieved in Section 5. The concluding remarks are presented in Section 6.

2. BACKGROUND
In this section we provide the background information on Coloured Petri nets, Workflow Management Systems, software agents, Web Services and the work related to the composition of Web Services using agents.

2.1 Coloured Petri nets (CPNs)

We use CPN as a formalism to model workflows in our system. The sound mathematical foundation behind Coloured Petri nets (CPNs) makes it a useful tool for modeling distributed systems. Petri nets consist of four basic elements:

- the *tokens* which are typed markers with values
- the *places* that are typed locations that can contain zero or more tokens,
- the *transitions* which represent actions whose occurrence can change the number, locations and value of tokens in one or more of the places connected to them, and
- the *arcs* and their inscriptions that connect places and transitions. A detailed description of CPNs can be found in [9].

2.2 WfMSs and agents

In the context of WfMSs, agent technology has been used in several different ways [16]. In some cases the agents fulfill particular roles that are required by different tasks in the workflow. In these cases the existing workflow is used to structure the coordination of these agents [2, 11-13]. An example of this approach is the work by M. Nissen in designing a set of agents to perform activities associated with the supply chain process in the area of e-commerce [2]. In other cases, the agents have been used as part of the infrastructure associated with the WfMS itself in order to create an agent enhanced WfMS [14, 15]. These agents provide an open system with loosely coupled components, which provides more flexibility than the traditional systems. Some researchers have combined both of these approaches [6], where an agent-based WfMS is used in conjunction with specialized agents that provide appropriate application-related services. In our framework, JBees [7], the WfMS is partitioned among various interacting agents following specified agent interaction protocols. The model associated with a business process is represented using the Coloured Petri net formalism and is executed by a specially designed agent. This agent-based environment facilitates the dynamic incorporation of changed models into the system and thereby assists process reengineering. Advantages of employing agents include the facilitation of inter and intra organizational cooperation and flexibility in choosing processes on the fly and controlling process parameters.

2.3 Web Services

Web Services are software components available on the Internet, which provide certain services that may be of general interest, such as weather monitoring services, currency converters, etc. A large fraction of the Web Services are used within companies protected by corporate firewalls. These corporate Web Services can be accessed for day-to-day business transactions. Examples of these Web Services include banking services and air ticket booking. The workflow process modeler can integrate Web Services with the existing workflow system. For instance, referring to the example above, a process model associated with the travel plan of a tourist may depend upon environmental factors, such as the weather conditions.

2.4 Related Work

Some researchers have integrated agent-based workflow systems with Web Services [17, 18]. However enhancements can be made to improve these approaches. In the research done by Buhler et al. [17, 18], the Business Process Execution Language (BPEL) [19] has been used as a process modeling formalism and a BPEL model is then converted to a Petri net. The problem with this approach, as acknowledged by the authors, is that the demonstration system developed by the researchers so far does not support some of the simple constructs of BPEL4WS. However, in our system the process model is described using a Coloured Petri net that can be directly executed. Our system does not require the conversion of a BPEL process into a Petri net process. The conversion mechanism of a BPEL model to a Petri net model has to be validated to ensure the structural and behavioral equivalence associated with the original model. Also, BPEL only facilitates static Web Service composition. According to our understanding there is no way of providing dynamic Web Service composition using BPEL.

3. EXISTING ARCHITECTURE

Our research is focused on developing an agent-based WfMS, where the work associated with running a WfMS has been partitioned among various collaborating agents that are interacting with each other by following standard agent communication protocols[4]. JBees is based on Opal [5] and uses the CPN execution tool JFern [3]. The processes are modeled using CPNs [9]. Earlier descriptions of JBees can be found in our previous papers [2, 10]. As shown in Figure 1, our enhanced system consists of seven Opal agents, which provide the functionality to control the workflow. The manager agent provides all functionality the workflow manager needs, such as creation and deletion of tasks, roles and process definitions, instantiation of new process instances and creation of resource agents. The process agent executes a process instance. Each resource in the system has its own resource agent. Every resource in the system gets registered to a broker agent that allocates the resources to a particular task. A resource agent is modeled as an interface for a human resource, a non-human resource such as a printer, or the software programs such as a Web Service. A resource agent that is specialized in connecting to a Web Service is a Web Service Agent [22, 23].

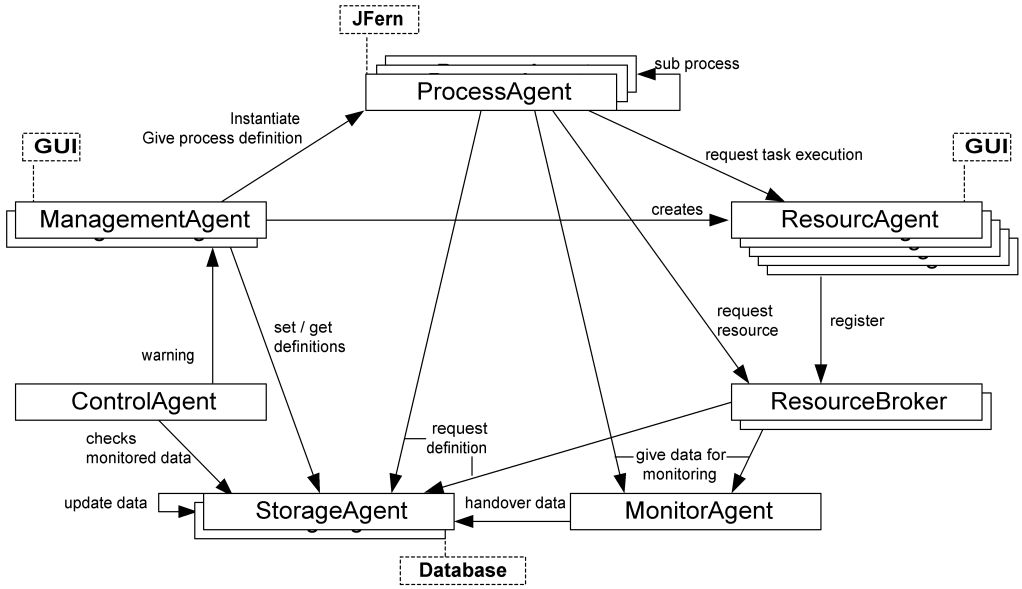

Figure 1: Architecture of agent based workflow system, JBees.

The storage agent manages the persistent data that is needed. The monitor agent collects all the process specific data and sends them to the storage agent. The control agent continuously looks for anomalies to the criteria specified by the human manager and reports the violations to these criteria to the manager agent. The manager agent provides information to the human manager, which can be used for a feedback mechanism.

4. KINDS OF COMPOSITION

The agent based workflow system that we have developed facilitates two kinds of composition, namely static and, importantly, dynamic composition.

4.1 Static composition

The first type of composition is static composition, in which the Service to which the agents will be connected is determined before the workflow execution takes place (prior to run-time). Assume that a travel service providing company offers facilities for tourist information services, air ticket booking, bus ticket booking etc. For internal automation of tasks, it also provides a service that would be accessible by different departments of the company, such as a list of places of interest, their distances from different cities, a list of car rentals etc. In this case, the process model for composing an itinerary which links services offered by different services (airways, bus, local tour organizer) would be a static composition since the existence of such a service is known prior to run-time and the agent can have design-time knowledge of connecting to such a service. Our initial work on static Web Service composition can be found in [22, 23].

4.2 Dynamic composition

Not all services will be known during the design time of the workflow. However, some of the services need to be discovered during workflow execution (run-time). For example, assume that a agent needs to find out the currency conversion rate of a particular currency. There are several service providers who provide such a service in the Internet. The agent now has lots of Web Services to connect to. During run-time the agent can now decide to connect to a reliable Web Service. Another example of dynamic composition is finding the lowest price of an air ticket for a particular destination. There could be various Web Services, each of which provide the information for the tickets. From a dynamic composition perspective the ticket booking agent will connect to all the currently available ticket booking services during run-time and find the lowest price.

In the next section we explain how dynamic Web Service composition can be achieved in our framework through a supply chain example from the apparel manufacturing industry.

5. DYNAMIC COMPOSITION OF WEB SERVICES – EXAMPLE FROM APPAREL MANUFACTURING INDUSTRY

5.1 Business process of an apparel manufacturing industry

Apparel manufacturing is one of important industries in the southern part of India. In 2004, the apparel manufacturing industry in India was worth $5 billion. Out of that the use of paper-based mode for raw material procurement alone was $1 billion. Supply chain management is the most important aspect of this industry. Manufacturers of apparels, such as shirts, depend on the raw materials such as fabric, threads, buttons zippers, etc. Before apparel manufacturing is done, raw materials have to be procured from various third party vendors.

The current trend of placing an order for raw materials, receiving a quotation etc. is paper-based. This leads to unnecessary delay in the production schedule and also affects the profitability of operations [24]. With the availability of

technologies that support Web Services, placing orders for the apparel manufacturing industry can be several magnitudes faster and can be potentially beneficial both to the apparel manufacturers as well as the other parties involved (raw material vendors).

Under such a scheme the vendors (raw material providers) can expose their products through a Web Service. An apparel manufacturer can contact any of the service providers. The overall process model of the apparel manufacturer is the Coloured Petri net model shown in Figure 2.

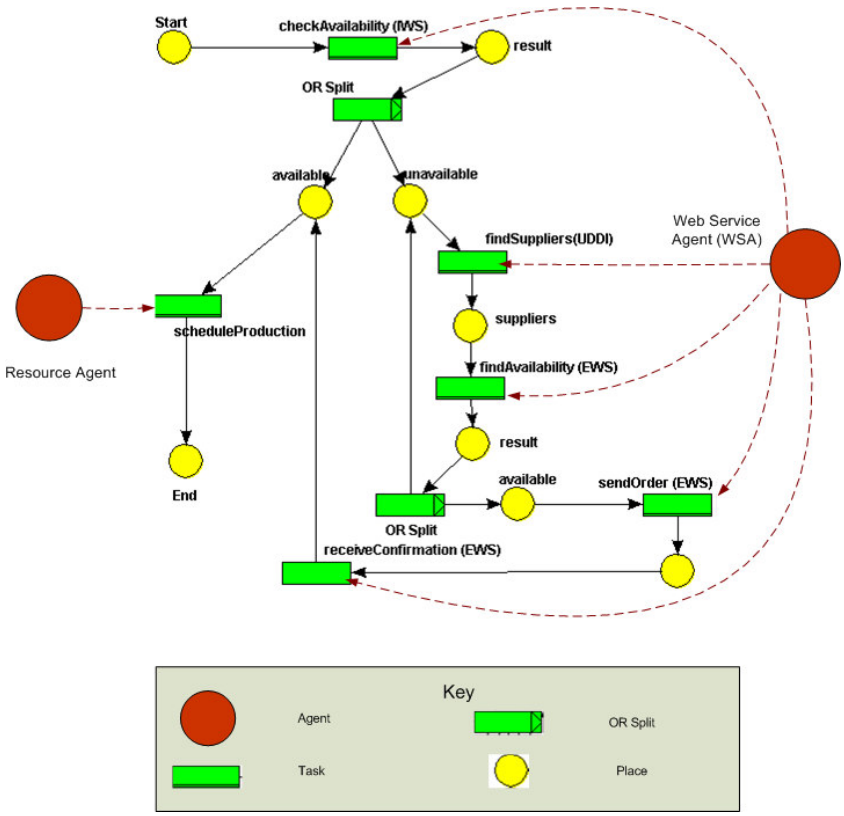

Figure 2. A Web service composition model of apparel manufacturing process using Coloured Petri Net formalism

Apparel manufacturers receive the order details from the large-scale clothing warehouses both in India as well as from abroad. Once the order is received the apparel manufacturers look for the particular components as listed in the order from different vendors.

For example assume that the type of components used for the manufacturing of a shirt involves the selection of a particular kind of fabric (the pattern, design), the color and quality of the thread used for stitching, the kind of button used etc. Currently a tender is called for these components and the vendors supply their quote to the companies. In the present circumstance the company follows a paper-based workflow model, which is time consuming.

We propose that the apparel manufacturing company can make use of dynamic composition of the Web Services. The workflow model for this scenario is given by Figure 2, which can connect to various service providers for placing orders.

The workflow manager creates an instance of the business process model shown in Figure 2. The workflow manager initiates the execution of this workflow instance with the appropriate details (as a job token). At first a Web Service Agent is allocated to check if the components required to produce the ordered clothing are available in the local warehouse (as exposed through Internal Web Service (IWS)). If the components are available, a production schedule is arranged according to the business rules encoded in the *scheduleProduction* transition.

Each of the service providers (vendors) registers their services with the Universal Definition Discovery and Integration (UDDI) registry (as shown by the architectural diagram in Figure 3) [21] using the commonly agreed Web Service Definition Language (WSDL) [20].

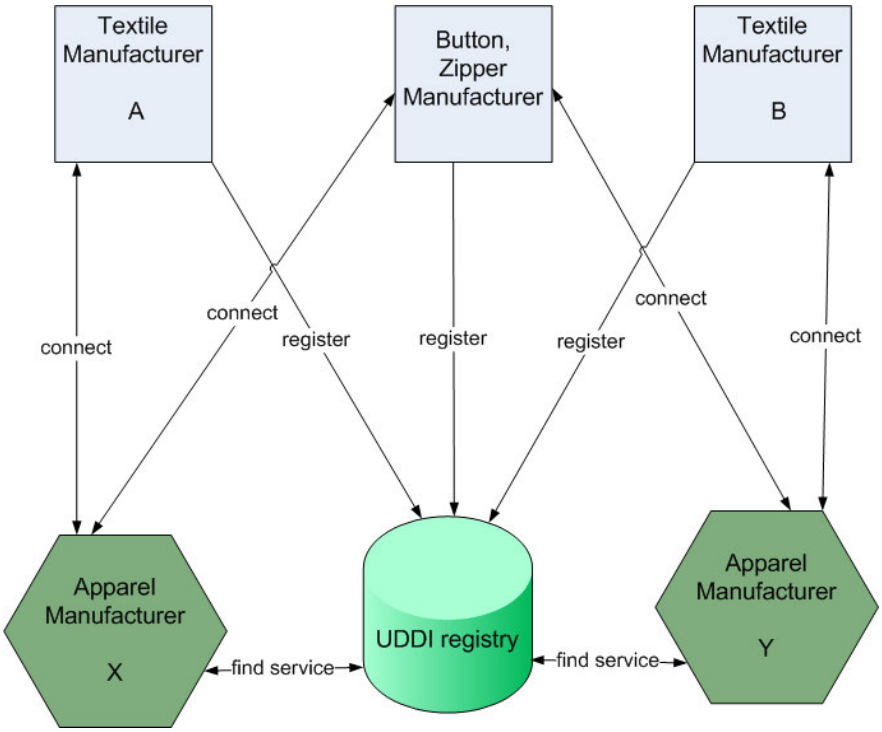

Figure 3. Web Services Architecture of the apparel-manufacturing scenario

Some of the WSDL operations defined by the vendors are given in Figure 4. If the required components are not available, the apparel manufacturer searches the private UDDI registry to find the list of available vendors for each component. This is done by a Web Service Agent (WSA). This agent returns a list of suppliers (URI's). Then, each supplier's Web Service is accessed by various Web Service agents to find out the availability and quote for each component.

The available services on the external Web Services could be a) query for the availability of a product b) query if a required quantity of the desired product is available c) submit query for how long it would take for them to manufacture a particular quantity of a product etc.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <wsdl:definitions targetNamespace="http://localhost:8080/axis/IWSCompanyX.jws"
    xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:apachesoap="http://xml.apache.org/xml-
    soap" xmlns:impl="http://localhost:8080/axis/IWSCompanyX.jws"
    xmlns:intf="http://localhost:8080/axis/IWSCompanyX.jws"
    xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  - <wsdl:message name="getStockRequest">
      <wsdl:part name="productId" type="xsd:string" />
    </wsdl:message>
  - <wsdl:message name="getStockResponse">
      <wsdl:part name="getStockReturn" type="xsd:double" />
    </wsdl:message>
  - <wsdl:message name="findAvailabilityRequest">
      <wsdl:part name="productName" type="xsd:string" />
      <wsdl:part name="quantity" type="xsd:double" />
    </wsdl:message>
  - <wsdl:message name="findAvailabilityResponse">
      <wsdl:part name="findAvailabilityReturn" type="xsd:boolean" />
    </wsdl:message>
  - <wsdl:portType name="IWSCompanyX">
    - <wsdl:operation name="findAvailability" parameterOrder="productName quantity">
        <wsdl:input message="impl:findAvailabilityRequest" name="findAvailabilityRequest" />
        <wsdl:output message="impl:findAvailabilityResponse"
          name="findAvailabilityResponse" />
      </wsdl:operation>
    - <wsdl:operation name="getStock" parameterOrder="productId">
        <wsdl:input message="impl:getStockRequest" name="getStockRequest" />
        <wsdl:output message="impl:getStockResponse" name="getStockResponse" />
      </wsdl:operation>
    </wsdl:portType>
```

Figure 4. WSDL of an Internal Web Service (IWS)

In the *sendOrder* transition, the results from various suppliers are analyzed and the best quote is selected by the workflow manager. Once the suppliers are selected an order is placed using the appropriate service. After the order for all the components for manufacturing a particular kind of apparel has been placed, a production plan is drawn automatically.

5.2 Discussion of the current status of our work

In our architecture we have incorporated appropriate mechanisms for static and dynamic composition. For static composition of Web Services, such as the Internal Web Services (IWS), the Workflow designer compiles the WSDL of the known Web Service and generates required stubs using WSDL2java [25]. Then the designer makes use of the stubs and implements the code needed by each of the transitions in the CPN process model to invoke certain operations on the Web Service.

For dynamic composition, the Workflow designer uses our custom-built library that uses WSDL4J [26] for the dynamic selection and invocation of the operations implemented by the chosen Web Service (EWS). Our implementation supports the simple data types. Currently. We are working towards a library that includes complex data types.

5.3 Advantages of the Web Service composition framework

a) Our framework uses CPNs, which have a formal specification and expressive. For example, proposed CPNs can be examined with tools to validate the reachability of specific nodes.

b) The agent-based framework can be employed to keep track of the reputation of Web Services and the collected data can be used for matchmaking.

c) Our architecture can be used to compose any Web Service available on the Internet, since dynamic composition is supported.

d) The support for monitoring and controlling capabilities in our framework can be used to monitor the performance of the Web Services and store historical data.

e) Semantic Web Services can be incorporated since our framework supports the incorporation of domain-specific ontologies.

f) Our architecture can support agent-based negotiation of services.

6. CONCLUSIONS AND FUTURE WORK

In this paper we have discussed the two types of Web Service composition mechanisms, static and dynamic composition. We have also explained how these mechanisms are implemented in our multi-agent based workflow framework and in particular we have discussed how dynamic Web Service composition can be achieved and used in an apparel manufacturing industry that deals with supply-chain management.

Currently we are working towards supporting complex data types in WSDL for dynamic composition in our framework. We are also working on validating our composition framework by using examples drawn from different workflow domains.

7. REFERENCES

[1] Martin Fleurke, Lars Ehrler, and Maryam Purvis, 'JBees - an adaptive and distributed framework for workflow systems', in Workshop on Collaboration Agents: Autonomous Agents for Collaborative Environments (COLA), Halifax, Canada, eds., Ali Ghorbani and Stephen Marsh, pp. 69–76, http://www.cs.unb.ca/~ ghorbani/cola/proceedings/NRC-46519.pdf, (2003). National Research Council Canada, Institute for Information Technology.

[2] S. Meilin, Y. Guangxin, X. Yong, and W. Shangguang,' Workflow Management Systems: A Survey. ', in Proceedings of IEEE International Conference on Communication Technology, (1998).

[3] Mariusz Nowostawski. JFern – Java based Petri Net framework , 2003.

[4] FIPA, FIPA Communicative Act Library - Specification. 2002. http://www.fipa.org/specs/fipa00037

[5] Martin K. Purvis, Stephen Cranefield, Mariusz Nowostawski, and Dan Carter, 'Opal: A multi-level infrastructure for agent-oriented software development', The information science discussion paper series no 2002/01, Department of Information Science, University of Otago, Dunedin, New Zealand, (2002).

[6] J.W. Shepherdson, S.G. Thompson, and B. Odgers, 'Cross Organisational Workflow Coordinated by Software Agents', in CEUR Workshop Proceedings No 17. Cross Organisational Workflow Management and Coordination, San Francisco, USA, (1998)

[7] Department of information science University of Otago. JBees. http://jbees.sourceforge.net, 2004.

[8] W.M.P van der Aalst and K. van Hee, Workflow Management: Models, Methods, and Systems , MIT Press, 2002.

[9] Jensen, K., Coloured Petri Nets - Basic Concepts, Analysis Methods and Practical Use, Vol. 1: Basic Concepts. EATCS Monographs on Theoretical Computer Science. 1992, Heidelberg, Berlin: Springer Verlag GmbH. 1-234.

[10] Savarimuthu, B.T.R., Purvis, M. and Fleurke, M. (2004). "Monitoring and Controlling of a Multi-agent Based Workflow System". In Proc. Australasian Workshop on Data Mining and Web Intelligence (DMWI2004), Dunedin, New Zealand. CRPIT, **32**. Purvis, M., Ed. ACS. 127-132.

[11] Dastani, M.M., Dignum, M.V., & Dignum, F.P.M. (2003). Role-Assignment in Open Agent Societies.. In (Ed.), *Proceedings of the Second International Conference on Autonomous Agents and Multiagent Systems (AAMAS'03)* Melbourne: ACM Press.

[12] Huang, P. and Sycara, K, 'A computational model for online agent negotiation ', in Proceedings of 35th Hawaii International Conference on System Sciences, (2002).

[13] N.R. Jennings, P. Faratin, T.J. Norman, P. O'Brien, and B. Odgers. Autonomous Agents for Business Process

Management. Int. Journal of Applied Artificial Intelligence, 14(2):145–189, 2000.

[14] H. Stormer. AWA - A flexible Agent-Workflow System . In Workshop on Agent-Based Approaches to B2B at the Fifth International Conference on Autonomous Agents (AGENTS 2001), Montral, Canada, 2001.

[15] M. Wang and H. Wang. Intelligent Agent Supported Flexible Workflow Monitoring System. In Advanced Information Systems Engineering: 14th International Conference, CaiSE 2002, Toronto, Canada, 2002.

[16] G. Joeris. Decentralized and Flexible Workflow Enactment Based on Task Coordination Agents . In 2nd Int'l. Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS 2000 @ CAiSE*00), Stockholm, Sweden, pages 41–62. iCue Publishing, Berlin, Germany, 2000.

[17] Paul Buhler and Jose M. Vidal. Enacting BPEL4WS specified workflows with multiagent systems. In Proceedings of the Workshop on Web Services and Agent-Based Engineering, 2004.

[18] Paul Buhler and Jose M. Vidal. Integrating agent services into BPEL4WS defined workflows. In Proceedings of the Fourth International Workshop on Web-Oriented Software Technologies, 2004.

[19] Business Process Execution Language for Web Services. http://www.ebpml.org/bpel4ws.htm.

[20] Web Services Definition Language, WSDL. http://www.w3.org/TR/wsdl

[21] UDDI (2003). http://www.uddi.org

[22] Savarimuthu, B.T.R., Purvis, M.A., Purvis, M.K. and Cranefield, S., "Agent-based Integration of Web Services with Workflow Management Systems", Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2005), M. Pechoucek, D. Steiner, S. Thompson, (eds.), ACM Press, ISBN 1-59593-093-0, New York (2005), pp 1345 - 1346.

[23] Savarimuthu, B.T.R., Purvis, M.A., Purvis, M.K. and Cranefield, S., "Integrating Web Services with Agent Based Workflow Management System (WfMS)", Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence (WI 2005), A. Skowron, J. Barthes, L. Jain, P. Morizet-Mahoudeaux, J. Liu, and N. Zhong, (eds.), ISBN 0-7695-2415-X, IEEE Press, Los Alamitos, CA (2005) 471-474.

[24] "A Case Study of Tirupur", www.unido.org/file-storage/download?file_id=41692, Accessed on 20th January 2006.

[25] Apache Axis WSDL2Java toolkit, http://ws.apache.org/axis/java/user-guide.html , Accessed on 20th January 2006.

[26] WSDL4J, http://sourceforge.net/projects/wsdl4j, Accessed on 20th January 2006.

[27] Srivatava, Koehler, "Web Service Composition - Current Solutions and Open Problems". ICAPS 2003 Workshop on Planning for Web Services, 2003, pp 28-35