ENABLING WEB SERVICES COMPOSITION WITH SOFTWARE AGENTS

Mihhail Matskin, Peep Küngas, Jinghai Rao, Jennifer Sampson, Sobah Abbas Petersen
Department of Information and Computer Sciences
Norwegian University of Science and Technology
Sem Sælands vei 7-9
NO-7491 Trondheim, Norway
{misha,peep,jinghai,jjsampson,sobah}@idi.ntnu.no

ABSTRACT
Recent progress in the field of Web services has made it practically possible to publish, locate, and invoke applications across the Web. The ability for efficient selection and integration of inter-organizational and heterogeneous services on the Web at runtime is an important requirement to the Web Service provision. In particular, if no single Web Service can satisfy the functionality required by a user, there should be a possibility to combine existing services together in order to fulfill the request. This trend has triggered a considerable number of research efforts on Web services composition both in academia and in industry. Another trend in modern Web services provision is related to making Web services pro-active. This means that services can be engaged into complex conversations instead of just their simple invocation and response with some result or an error.

In this paper we describe our solution for pro-active Web services selection and composition. In particular, we consider a logic-based method for services composition and marketplace-based system architecture supporting agent communication, negotiation and semantical reasoning in the composition process.

KEY WORDS
Software Agents, Web Services, Services Composition

1. Introduction

Amount of products and services available now on the Web increases dramatically and goes beyond user's ability to analyze them efficiently. At the same time the number of potential customers available via the Internet also increases significantly and starts to be beyond service providers' ability to perform efficient targeted marketing. If no new solutions will be applied the amount of potentially unseen relevant services and non-contacted potential relevant customers will grow dramatically (while the amount of non-relevant offers – spam – will increase). This problem is related to personalization and pro-active service selection and provision.

Another important issue related to development of Web services is their integration and composition. Recent progress in the field of Web services has made it practically possible to publish, locate, and invoke applications across the Web. This is a reason why more and more companies and organizations now implement their core business and outsource other application services over the Internet. In particular, if no single Web service can satisfy the functionality required by a user, there should be a possibility to combine existing services together in order to fulfill the request. The challenge is that Web services can be created and updated on the fly and it is often beyond human capabilities to analyze the required services and compose them manually. The complexity of selecting and composing Web services descends from the following two sources: 1) it is not always easy to define selection criteria for a Web Service; 2) Web services can be developed by different organizations, which provide different offers, so, the ability of efficient integration of possibly heterogeneous services on the Web becomes a complex problem (especially for dynamic composition during runtime).

Web services are considered as self-contained, self-describing, modular applications that can be published, located, and invoked across the Web [1]. Several initiatives in Web services provide platforms and languages for integration of heterogeneous systems. In particular, Universal Description, Discovery and Integration (UDDI) [2], Web Services Description Language (WSDL) [3], Simple Object Access Protocol (SOAP) [4] and part of OWL-S [5] ontology (service profile and service grounding) define standard ways for service discovery, description and invocation (message passing). Some other initiatives such as Business Process Execution Language for Web Service (BPEL4WS) [6], Web Service Flow Language (WSFL) [7] and OWL-S Process Model [5], are focused on representing service composition where process flow and bindings between services are known a priori (OWL-S also provides basis for description of Semantic Web Services). Recent work on WSMO and WSML is a promising initiative towards exploiting semantic-oriented solutions in Web services [31].

In this paper we are focusing on our general conceptual solution to the problem of enabling Web services composition with software agents while for technical details we refer to our other publications.

2. Related work

There are several works on incorporating agents into Web service systems. In particular, Gibbins et al [9] demonstrated usage of DAML-S for Web services descriptions within agents. Another step towards incorporating Web services into agents is proposed by Ardissono et al [10]. Their main contribution is support for more rigorous Web services execution protocols compared to currently prevalent 2-steps protocols (sending input data and then collecting results). The 2-steps protocols are definitely not enough if we engage agent negotiation in service composition.

Architecture for agent-based Semantic Web service composition was also proposed by Ermolayev et al [11]. Since their focus has been set to non-symbolic negotiation, their work could be seen as a complementary part to our work, where we focus on logic-based Web services composition.

Several methods for dynamic composition of Web services have been proposed recently. Most of them fall into one of the following two categories: methods based on pre-defined workflow model and methods based on AI planning.

For the methods in the first category a user should specify the workflow of the required composite service, including both nodes and the control flow and the data flow between the nodes. The nodes are regarded as abstract services that contain search recipes. The concrete services are selected and bound at runtime according to the search recipes. This approach is widely adopted by members of the Information Systems community.

In particular, EFlow [12] is a platform for the specification, enactment and management of composite services. A composite service is modeled by a graph that defines the order of execution among the nodes in the process. The graph is created manually but it can be updated dynamically. It provides the adaptive and dynamic features to cope with the rapidly evolving business and IT environment in which Web services are executed.

Yang et al. [28] proposed a composition logic which dictates how the component services could be combined, synchronized and coordinated.

Tut et al. [29] introduced the use of patterns during the planning stage of service composition. The patterns could be business patterns such as how to model online store-fronts, or generic patterns such as project work process.

Polymorphic Process Model (PPM) [13] supports both reference process and service-based multi-enterprise processes. PPM decouples activity interface from activity implementation, and provides process polymorphism to support their mapping. In PPM, a service is modeled by a state machine that specifies possible states of a service and their transitions. Transitions are caused by service operation (also called service activity) invocations or internal service transitions.

The second category includes methods related to AI planning. They are based on the assumption that each Web service is an action which alters the state of the world as a result of its execution. Since the services (actions) are software components, input and output parameters of a service act as preconditions and effects in the planning context. If the user can specify inputs and outputs required by the composite service then the service (as a composition of available services) is generated automatically by AI planners without knowledge of predefined workflow.

In [14], the authors adapt and extend the Golog logic programming language for automatic construction of Web services. They address the Web services composition problem by provision of high-level generic procedures and customizing constraints.

Waldinger [15] elaborates an idea of service synthesis by theorem proving. His approach is based on automated deduction and program synthesis and has its roots in his early work [16].

McDermott [17] considers closed world assumption in AI planning in composing Web services. He introduces a new type of knowledge, called *value of an action*, which persists and which is not treated as a truth literal. However, while using resource-conscious logics (as we do in our work) this problem is treated implicitly and there is no need to distinguish informative and truth values.

In [18] the SHOP2 planner is applied to automatic composition of Web services, which are provided with DAML-S descriptions. Another planner for automatic Web service construction is presented in [19].

Thakkar et al. [20] consider dynamic composition of Web services using a mediator architecture. The mediator takes care of user queries, generates wrappers around information services and constructs a service integration plan.

SWORD [21] is a developer toolkit for building composite Web services. SWORD does not deploy the emerging service description standards such as WSDL and OWL-S, instead, it uses Entity-Relation (ER) model to specify the inputs and the outputs of Web services.

Sirin et al [22] presents a semi-automatic method for Web services composition.

The main difference between our approach and the above-mentioned methods is that we propose a unified solution to Web services composition problem. In particular, we consider loosely coupled services networks, logic with higher expressive power (Linear Logic [23]) and a framework where methods from both

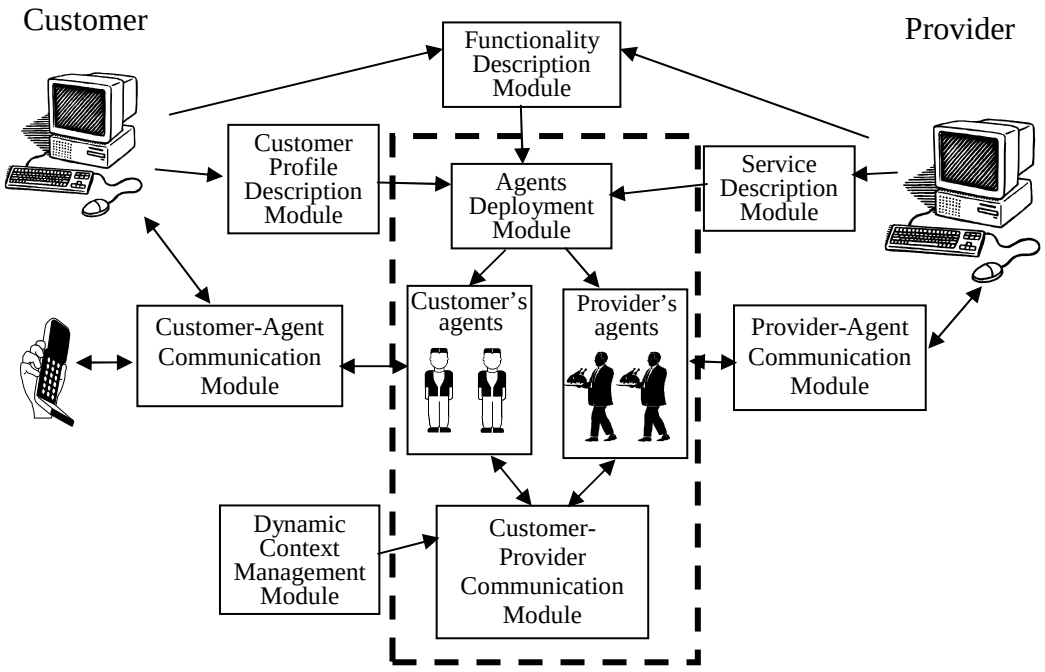

Figure 1. Functional architecture of the system.

above-mentioned categories will work together. Additionally, usage of the logic with higher than classical logic expressive power allows us formally express and utilize richer semantical information (such as non-functional characteristics of Web services, in particular, quantitative attributes) during the reasoning process and distinguish the constraints and facts in qualitative attributes.

3. General Approach

A general functional architecture of the system for agent-enabled Web services selection and composition is presented in Figure 1.

Our approach to supporting Web Service composition is based on providing advance communication and decision making agent-based environment which employs the concept of network of marketplaces [8]. A functionality of the marketplace should include a matchmaking, negotiation, communication and coordination support. The platform implementing the marketplace should have open architecture and allow customization/overriding of the default functionality. We would like to mention that marketplaces are quite usual means for implementation of selling/buying activity for services. However, in our work we apply marketplace also to service composition which (to the best of our knowledge) was not done extensively before. In our approach a marketplace is an infrastructural element which may support a coherent behaviour of agents or serve for conflict resolution via negotiation. There can be many marketplaces for service provision and selection and they can be interconnected (constitute a network). Marketplaces can be used for customers and/or service providers' coalitions formation, for asking a service/product details, for matchmaking, for discrepancy resolution as well as for giving bids and providing special offers. We also assume that they can be used for advertising and information exchange. Each customer and provider may have more than one agent representing him/her in the environment, and marketplaces can be created for gaining consistent behaviour between them. All these features enable pro-active Web services selection and composition. Figure 2 presents an example of a network of marketplaces for a possible service provision situation.

Marketplaces provide a support for composition of services based both on flow models and AI planning.

In particular, the proposed network of marketplaces gives a novel solution to the composition process flow model. The novelty lays in assumption that networks of marketplaces may allow to specify more flexible Web services process and data flows with decentralized control than traditional workflow models.

For AI planning based composition of Web services we employ program synthesis-based approach. If we consider services as software components then the Web service composition can be presented as a software synthesis problem. In comparison with software components, Web services may present a higher abstraction level and they are more loosely coupled. This may allow us to construct a composite service based on a functional specification, without taking into account low level technical details, such as operational environment or communication protocol.

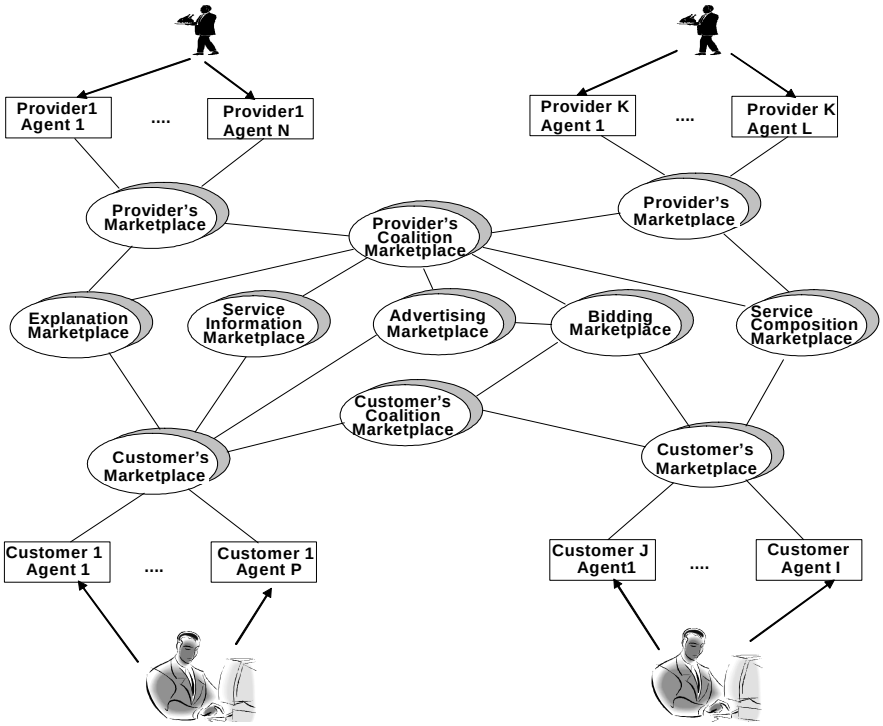

Figure 2. Network of marketplaces for customers and providers

In particular, we develop a method for automated Web service composition which is based on the proof search in multiplicative intuitionistic fragment of Linear Logic (MILL) [27]. The fragment is sound and complete and this guarantees us that the composed services are correct and that all composable services solutions will be found. Choice of Linear Logic (LL) [23] as a formalism for service composition is based on high expressive power of LL for representation of different aspects of Web services. Given a set of existing Web services and a set of functionality and non-functionality attributes, the method finds a composition of atomic services that satisfies the user requirements.

The composition process is as follows. First, a description of existing Web services (written in WSDL or DAML-S/OWL-S) is translated into extralogical axioms in LL, and the requirements to the composite services are specified in form of a LL sequent to be proven. Second, MILL theorem prover is used to determine whether the requirements can be fulfilled by the composition of existing atomic service. If it is possible then the last step is constructing flow models (written, for example, in BPEL4WS or OWL-S) from the generated proofs. We assume that the composite service is ready to be executed when the flow model and description of each atomic service are given. In comparison with other composition methods, our method also utilizes semantical information (such as quantitative and qualitative constraints) in addition to structural service information in service description. This is possible because of the usage of logic with a high expressive power – Linear Logic.

The proposed composition method is to be incorporated into the marketplace infrastructure as a default service provided by the marketplace. However, system architecture will provide an easy "plug-in" of other composition methods as well.

Until now, we assumed only exact match of input and output parameters of atomic services in the composition process. However, in practice services can be composed even if the output parameters of one service do not match exactly inputs of other services. This can be done via semantic reasoning using Domain Ontology as it is shown in Figure 3.

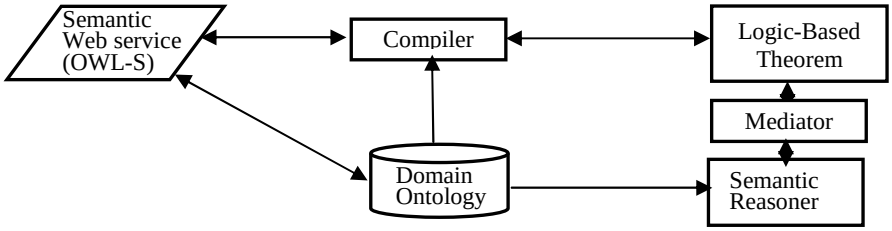

Figure 3. Interaction with semantic reasoner

3. System Components

Main components of our system are designed and prototypically implemented. In particular, the basic marketplace infrastructure was developed in the early versions of the Agora system [19]. It supports both FIPA and KQML specifications for the Agent Communication Language. We are currently working on extending the Agora system design, adjusting it to the needs of pro-active Web services selection and composition and improving its compatibility with modern agent platforms (in particular, with JADE).

The first version of the services composition system based on Linear Logic (as well as details of the method) is presented in [24]. The system utilizes our developed Linear Logic prover and exploits Jena – a Java framework for building Semantic Web services applications. Web services composition method is an essential part of the market place functionality and is to be incorporated both into assistant deployment module and into customer-provider communication module. Having Linear Logic as a basic reasoning method we consider possibility to "plug-in" other reasoning methods for services composition on demand.

Symbolic negotiation method for service selection is presented in [25]. The method uses Partial Deduction in Linear Logic for recognizing new goals and missing services. It is going to be incorporated into matchmaking component of marketplaces as well as into a reasoning component of the agent execution engine.

An essential part of the presentation modules of the system (both customers and services) is based on ontologies. They are used for structuring Web services information on the provider's side and for description of the customers' profiles. Ontologies also play a key role in semantic reasoning. In order to reduce discrepancies between ontologies used by different providers and customers we are working on ontology explanation and ontology mapping methods. An experiment with explanation ontology using the current version of the Agora system is briefly described in [26].

4. Application

Until now we were mainly focused on design and implementation of system components. For performing the system integration (that is under construction now) we are working on developing several application scenarios. Although our scenarios are from "real" application areas we didn't work yet on their practical implementation but rather consider them at the moment as test cases for the system.

The first scenario is using agents for facilitating mobile evidence-based medicine [30]. This might involve the use of software assistant agents to pro-actively communicate availability of relevant clinical cases to clinicians. The software agent may also communicate with other software agents to make informed decisions based on specific patient medical results and documented medical research.

The second scenario is related to Virtual Enterprise formation [32].

Another important work related to application of the system under consideration is related to experiments with composition of existent Web services. Since practically applicable composition requires sufficient amount of semantic knowledge about the composable services (which is not explicitly presented in the WSDL and UDDI specifications) we are working on tools for (semi-) automatic annotation of "non-semantic" Web services.

5. Conclusion and Future Work

In this paper we have shown our solution to enabling Web services selection and composition with software agents. Usage of agents allows supporting pro-activeness and autonomy of the composition process where both parties – customers and providers can play an active role in the process via autonomous operation and negotiation. The proposed architecture allows sufficient flexibility and extensibility in developing different solutions while providing a good set of embedded tools. The first evaluation of the approach and system is quite promising; however, more extensive practical analysis of its advantages and potential benefits should be done when the work on the system will be completed.

For the future work we continue work on integration of the system components and development embedded (default) tools into the system. We also recognize a great importance of semantic enrichment of available services on the Web and put a high attention to work on ontology mapping, ontology explanation and semantic services annotation.

6. Acknowledgements

This work is partially supported by the Norwegian Research Foundation in the framework of the Information and Communication Technology (IKT-2010) program--- the ADIS project.

References:

[1] IBM web services tutorial." [Online]. http://www-106.ibm.com/ developerworks/webservices/

[2] T. Bellwood et al., Universal description, discovery and integration specification (UDDI) 3.0." [Online]. Available: Services: Modeling, Architecture and Infrastructure" workshop in conjunction with ICEIS2003, 2003. http://uddi.org/pubs/uddi-v3. 00-published-20020719.htm

[3] E. Christensen et al., Web services description language (WSDL) 1.1 http://www.w3.org/TR/wsdl/

[4] D. Box et al., "Simple object access protocol (SOAP) 1.1," 2001. http://www.w3.org/TR/SOAP/

[5] D. Martin et al., “OWL-S Semantic Markup for Web Services" November 2004 [Online] http://www.daml.org/services/owl-s/1.1/overview/

[6] F. Curbera et al., Business process execution language for web services (BPEL4WS) 1.0, July 2002, http://www-106.ibm.com/developerworks/ webservices/library/ws-bpel

[7] F. Leymann, Web Services Flow Language (WSFL 1.0), IBM Software Group, http://www-3.ibm.com/software/solutions/webservices/pdf/WSFL.pdf

[8] M. Matskin, Multi-Agent Support for Modeling Co-operative Work. In: T. Yongchareon, F. A. Aagesen, V. Wuwongse (Eds.) Intelligence in Networks. The Fifth International Conference SMARTNET'99, Thailand, Kluwer, 1999, pp. 419-432.

[9] N. Gibbins, S. Haris and N. Shadbolt. Agent-based Semantic Web Services. In Proceedings of the 12th Int. WWW Conf., WWW2003, Budapest, Hungary, 2003, ACM Press, 2003, pp. 710-717

[10] L. Ardissono, A. Goy, and G. Petrone. Enabling conversations with web services. Proc. of the 2nd Int. Conf. on Autonomous Agents and Multiagent Systems, 2003, Melbourne, pp. 819—826.

[11] V. Ermolayev et al. Towards a Framework for Agent-Enabled Semantic Web Service Composition. Int. J. of Web Services Research, 1(3):63-87, 2004

[12] F.Casati and S. Ilnicki and LiJie Jin. Adaptive and Dynamic Service Composition in eFlow, Proc. of 12th Int. Conff on Advanced Information Systems Engineering, 2000, Stockholm, Springer.

[13] H. Schuster, D. Georgakopoulos, A. Cichocki and D. Baker, Modeling and composing service-based and reference process based multi-enterprise processes, Proc. of 12th Int. Conf. on Advanced Information Systems Engineering (CAiSE), 2000, Stockholm, Springer.

[14] S. McIlraith and Tran Cao Son. Adapting Golog for Composition of Semantic Web Services, Proc. of the 8th Int. Conf. on Knowledge Representation and Reasoning, 2002, Toulouse, France

[15] Richard Waldinger. Web agents cooperating deductively. In Proc. of FAABS 2000, Greenbelt, MD, USA, April 5--7, 2000, volume 1871 of LNCS, pp. 50--262. Springer-Verlag, 2001.

[16] Z. Manna and R. J. Waldinger. A deductive approach to program synthesis. ACM Transactions on Programming Languages and Systems, 2(1):90--121, 1980.

[17] D. McDermott. Estimated-regression planning for interaction with web services. In Proc. of the 6th Int. Conf. on AI Planning and Scheduling, Toulouse, France, 2002. AAAI Press, 2002.

[18] D. Wu, B. Parsia, E. Sirin, J. Hendler, and D. Nau. Automating {DAML-S} web services composition using SHOP2. In Proc. of the 2nd Int.l Semantic Web Conference, ISWC 2003, Sanibel Island, Florida, USA, October 20--23, 2003.

[19] M. Matskin, O. J. Kirkeluten, S. B. Krossnes and Øystein Sæle. Agora: An Infrastructure for Cooperative Work Support in Multi-Agent Systems. T. Wagner, O. Rana (eds.) Infrastructure for Scalable Multi-Agent Systems. Springer Verlag, LNCS Volume 1887, 2000, pp. 28-40.

[20] S. Thakkar et al. Dynamically composing web services from on-line sources. In Proceeding of 2002 AAAI Workshop on Intelligent Service Integration, Edmonton, Alberta, Canada, 2002.

[21] S. R. Ponnekanti and A. Fox, SWORD: A Developer Toolkit for Web Service Composition, The Eleventh WWW Conference, 2002, Honolulu, HI, USA

[22] E. Sirin and J. Hendler and B. Parsia, Semi-automatic Composition of Web Services using Semantic Descriptions, Workshop on Web Services: Modeling, Architecture and Infrastructure in conjunction with ICEIS2003

[23] J.-Y. Girard. Linear Logic. Theoretical Computer Science, Vol. 50, pp. 1--102, 1987.

[24] J. Rao, P. Kungas and M. Matskin. "Logic-based Web Service Composition: from Service Description to Process Model". Proc. of the IEEE Int. Conf. on Web Services, ICWS 2004, San Diego, California, USA, July 6-9, 2004, IEEE Computer Society Press

[25] P. Küngas, M. Matskin. Symbolic Negotiation with Linear Logic. In Proceedings of the 4th Int. Workshop on Computational Logic in Multi-Agent Systems, CLIMA IV, Fort Lauderdale, FL, USA, 2004, Springer, LNCS, Vol. 3259, pp. 71-88, 2004

[26] X. Su, M. Matskin and J. Rao. Implementing Explanation Ontology for Agent System. The 2003 IEEE/WIC International Conference on Web Intelligence, Halifax, Canada, October, 2003

[27] P. Lincoln, Deciding provability of linear logic formulas, In: London Mathematical Society Lecture Note Series, Cambridge University Press, 1995, Vol. 222, pp. 109-122

[28] J. Yang and M. P. Papazoglou. Web component: A substract for web service reuse and composition. In Proceedings of the 14th International Conference for Advanced Information Systems Engineering (CAiSE), LNCS 2348, Toronto, Canada, May 2002.

[29] M. Thandar Tut and D. Edmond. The use of patterns in service composition. In The First Workshop of Web Services, e-Business and the Semantic Web, Toronto, Canada, 2002.

[30] J. Sampson, J. An Agent Based Infrastructure for facilitating evidence-based medicine. In Proceedings of ICEIS 6th International Conference on Enterprise Information Systems, (Short paper) Porto, Portugal.

[31] SDK WSMO working group. [Online] http://www.wsmo.org/

[32] S. Petersen and M. Matskin. Agent Interaction Protocols for the Selection of Partners for Virtual Enterprises. V. Marik et al (Eds.): Multi-Agent Systems and Applications III, CEEMAS 2003, Prague, LNAI 2691, Springer, pp. 606-615