Perspective Methods and Tools for the Design of Distributed Software Systems Based on Services

Marek Paralič

Department of Computers and Informatics, Faculty of Electrical Engineering and Informatics (FEI), Technical University in Košice
Letná 9, 042 00 Košice, Slovakia
Marek.Paralic@tuke.sk

Abstract: In this paper the current research activities of the Distributed team at the Department of Computer and Informatics, FEI, TU in Košice are described. Our focus is to propose and verify new methods and tools, which could contribute to the design and implementation of integrated, pervasive and collaborative services. Pervasive computing is an important research area whose challenges require a thorough rethinking and revision of conventional software design ideas. In reality, there is little consensus and very little basic understanding of the underlying issues and their interactions to produce useful solutions. Our research activities aim to perform the research necessary to contribute to this understanding.

Keywords: service-oriented computing, agent technology, pervasive computing, semantic web services

1 Introduction

Research at the Department of Computers and Informatics in the area of distributed software systems design is supported by the Slovak National Grant Agency and in the past covered also topics such as: methods and tools for design of the distributed programming systems [1] or design and implementation of a technological platform for creating multi-agent applications with hierarchical structure of mobile and static agents [2][3]. Our research in last decade has been focused on methods and methodologies in the area of distributed software systems with strengthened emphasis on agent technologies, whereby the major results achieved include the following:

- architecture that combines mobile agent paradigm with formal foundation of concurrent constraint programming, logic variables and true distributed systems with shared distributed data structure [1],

- definition of simple form for formal expression, control and analysis of dynamic properties (communication and mobility) of mobile software applications [3],
- tool for modelling of multi-agent system that respects security requirements of agents and places, which utilizes modified calculus of mobile ambients [3],
- agent based scheme for generic service for static and mobile users [2],
- multi-agent platform for support of design, implementation and maintenance of flexible distributed services for mobile users, which was verified on the set of simple services designated to the university communities [2][13].

One of the current high interesting areas in the distributed systems is the problem of building and running pervasive computing services. Pervasive computing is an important research area whose challenges require a thorough rethinking and revision of conventional software design ideas. In reality, there is little consensus and very little basic understanding of the underlying issues and their interactions to produce useful solutions. Our research activities aim to perform the research necessary to contribute to this understanding.

Pervasive computing can be seen as a method of enhancing computer use by making many computers available throughout the physical environment, but making them effectively invisible to the user. Pervasive computing system should faced up to the following features [4]: (i) pervasive – it must be everywhere, (ii) embedded – it must live in our world, sensing and affecting it, (iii) nomadic – it must allow users and computations to move freely, according to their needs, (iv) adaptable – it must provide flexibility and spontaneity in response to changes in user requirements and operating conditions, (v) powerful, yet efficient: it must free itself from constraints imposed by bounded hardware resources, addressing instead system constraints imposed by user demands and available power or communication bandwidth, (vi) intentional – it must enable people to name services and software objects by intent, and (vii) eternal – it must be available all the time.

In our current research activities we focus on the methods and tools, which could support the design and implementation of pervasive services, whereby we utilise our experiences with mobile agents and agent technologies in general, service-oriented architecture and technologies of the semantic web. The following sections bring the basic characteristics of the applied technologies – agent technology, service-oriented design based on web services and semantic web technologies relevant for our purposes. In the conclusion our future plans are briefly sketched.

2 Agent Technologies

Agents can be defined to be autonomous, problem-solving computational entities capable of effective operation in dynamic and open environments. Agents are often deployed in environments in which they interact, and sometimes cooperate with other agents (including both, people and software) that have possibly conflicting aims. Such environments are known as multi-agent systems [5]. The term multi-agent system is now used for all types of systems composed of multiple autonomous components showing the following characteristics [6]: each agent has incomplete capabilities to solve a problem, there is no global control system, data is decentralized and computation is asynchronous. Agent technology enables the realization of complex software systems characterized by situation awareness and intelligent behaviour, a high degree of distribution, as well as mobility support. Agent technology has the potential to play a key role in enabling intelligent applications and services by improving automation of routine processes, and supporting the nomadic users with pro-active and intelligent assistance based on principles of adaptation and self-organization [7].

On the field of standardization provides The Foundation for Intelligent Physical Agents (FIPA)[1] a comprehensive set of specifications from messaging and directories, through system management, agent mobility, up to well-formed semantic communication languages for agents. One of the most popular FIPA-compliant agent frameworks is JADE (Java Agent DEvelopment Framework)[2] with the LEAP (Lightweight Extensible Agent Platform) enhancement that enables to run agent platform on small devices like PDAs and mobile phones [9]. Further activities represent OMG[3], conception for communication and cooperation, design languages for agent-based systems with supported tools like AUML[4]. The goal of such activities is to improve the analysis and specification of the multi-agent systems.

Some of the key industrial application areas for agent-based systems are [10]: monitoring and control of physically highly distributed systems; real-time control of high-volume, high-variety, discrete manufacturing operations and transportation and material-handling systems. Another application area for agent technology is building of smart environment based on pervasive computing – e.g. EasyMeating system [11] that explores the use of FIPA agent technologies, Semantic Web ontologies, logic reasoning, and security and privacy policies. Agent technology can open the way to new application domains while supporting the integration of existing and new software, and make the development process for such applications easier and more flexible [7][12][12].

---

1 The Foundation for Intelligent Physical Agent (FIPA), http://www.fipa.org/
2 Java Agent DEvelopment Framework (JADE), http://jade.cselt.it/
3 Object Management Group, http://www.omg.org
4 Agent UML, http://www.auml.org

Despite many of relevant results, multi-agent systems have not become widespread as industrial and commercial applications. In order to bridge the gap between agent technology and methodologies or technologies accepted for real world applications some efforts have been performed. A survey of agent-oriented methodologies can be found in [14]. Some of the most interesting results are Gaia [15] (methodology for agent-oriented analysis and design supporting macro/societal level as well as micro/agent level aspects) and MaSE [16] (object-oriented approach for support of the complete software lifecycle from problem description to realization), which we also applied when developing our own agent-based systems (e.g. [12]).

3 Service-oriented Computing and Web Services

Services are means for building distributed applications and are used to build service-based applications. In this context the service composition is of great importance, whereby it should be supported by every aspect of services – they can be described, selected, engaged, collaborated with and evaluated. Service-oriented computing is a general topic, more specifically in the context of WWW technologies we are speaking about Web Services.

As defined by the World Wide Web Consortium [17], a Web Service is a software system identified by a URI (RFC 2396), whose public interface and bindings are defined and described using XML. Its definition can be discovered by other software systems. These systems may then interact with the Web Service in a manner prescribed by its definition, using XML-based messages conveyed by Internet protocols. Web Services rely on the functionalities of publish, find, bind [18] and the components of a Web Service Model include Service Providers, Service Broker and Service Requester. Web Services are defined by their interfaces in particular about how they describe their functionality, how they register their presence, and how they communicate with other Web Services. People who want to use Web Services could connect to the Universal Description Discovery and Integration (UDDI) centre to search for required services. The information about the Web Services described by Web Service Description Language (WSDL) can be acquired. And the user could use the Simple Object Access Protocol (SOAP) to transfer the requirement information and receive the real service.

As described above, core Web Service technology (WSDL, UDDI) defines formal interface contracts, describing the message syntax, but does not address the semantics of those interfaces. This means, that the meaning of the exchanged data is not formally described. Since the emerging Semantic Web and Web Services have a similar target audience, namely application clients, therefore they are intended for automated processing and share a common base technology (XML), it is obvious to apply semantic web techniques to web services. The Resource Description Framework (RDF) is particularly intended for representation of metadata about web resources in general and web services in particular and represents a notation to express structured metadata which has also a XML-based format representation (RDF/XML).

On higher level of abstraction there are several web ontology languages which are based on RDF notation and are used for knowledge modelling, i.e. define vocabularies and express classes of information by means of the vocabulary and their inherent relationships. The most applied ontology language is OWL (based on DAML, etc.). For the specific application field of web services, languages like OWL-S are evolving. These languages are derived from OWL and enable to define richer semantic for service specifications. This can enable richer, more flexible automation of service provision and use, and support the construction of more powerful tools and methodologies [19]. Another example of research activities in the same area is the Web Service Modelling Ontology (WSMO) and its reference implementation The Web Service Execution Environment (WSMX) [20].

4 Service Compositions in Open Environment

Desirable abstractions for service composition in open environments should exploit opportunities offered by those services:

(i) because services are autonomous, we should not require them to be subservient to other services,

(ii) because services are heterogeneous, we should develop expressive, standardizable representations (presently this is done for data, but not for processes and policies),

(iii) because services are long-lived, evolving and operate in environments that produce exceptions, representations should handle such situations,

(iv) because services can be cooperative, abstractions would represent how they behave in the awareness of the behaviours of other services.

Current approaches are connected to low-level invocation of services – they are not specially geared for enabling composition. Services are integrated through method invocation without regard to any higher-level constraints. Semantic web technologies make use of prior work done in workflow management systems, artificial intelligence approaches for planning, formal process models, multi-agent planning and description logic. The objectives for the development of Semantic Web services are to enable reasoning about Web services, planning compositions of Web services and automating the use of services by software agents. The goal is to make Web services unambiguously interpretable by a computer. Examples of ontologies used for description of Web services are already mentioned OWL-S or WSML. Using these ontologies, a Web service can advertise its functionality to potential users. A request for a service would then be matched against the Web service's advertisement via a matchmaking process, because the objectives of such a request are expressed as goals, which are high level descriptions of concrete tasks. Every requester expresses its goal in terms of its own ontology, which provides the means not only for human to understand the goal, but also for a machine to interpret is as a part of requester's ontology. Similar to the goal description, all semantic web services have their descriptions in their own ontology. In case the ontologies differ, the mediation process should be applied. Mediation is also utilized in case of heterogeneous communication protocols of involved services.

There are many formalisms for describing a business process, such as BPMI or BPML and OWL-S was designed to be neutral with respect to a particular formalism. It instead just provides the necessary vocabulary and properties for a process model. Both the OWL-S service model and WS-BPEL provide a mechanism for description of a business process model. However, they can be contrasted in terms of their expressiveness, representations, semantics, discovery support, execution support and fault handling [21].

Conclusions

In the centre of our attention are methods and tools for building of integrated services in a distributed environment, whereby as basic constructing elements the agent-oriented design and service-oriented design are used. To achieve the goal – design of pervasive integrated services – the agent technology is applied, whereby the concept of the higher-level agent – ambient – is proposed. Ambients support the process of building dynamic workflows of services, whereby semantic description of services is used. Utilisation of semantic information offers improved capabilities in terms of their expressiveness, representations, discovery support, execution support and fault handling. During our future research we strive to give precise semantic of identified workflow patterns, which will be integrated into tools aimed to support design and execution of integrated pervasive services.

Acknowledgement

Research described in this paper is supported by the project '***Integrated distributed applications based on ambients – higher-level agents***' Nr. 1/3135/06 of the Slovak Grant Agency.

References

[1] Paralič, M.: Mobile Agents Based on Concurrent Constraint Programming, Joint Modular Languages Conference (JMLC 2000), September 6-8, 2000, Zurich, Switzerland, Lecture Notes in Computer Science, Vol. 1897, pp. 62-75

[2] Paralič, M., Krokavec, M.: Multi-Agent Applications Design Support. Journal of Information and Organizational Sciences, Vol. 27, No. 2, 2003, ISSN 0351-1804, pp. 101-108

[3] Tomášek, M.: Interactions of Mobile Agents in the Distributed System, in Journal of Information, Control and Management Systems, Vol. 3, Nr. 2, 2005, ISSN 1336-1716, pp. 173-182

[4] Liu, R., Chen, F., Yang, H., Chu, W. C., Yu-Bin Lai: Agent-based Web Services Evolution for Pervasive Computing, in Proceedings of 11th Asia-Pacific Software Engineering Conference (APSEC 2004), November 30-December 3, 2004, Busan, Korea, pp. 726-731

[5] Luck, M., McBurney, P., Preist, Ch.: Agent Technology: Enabling Next Generation Computing, A Roadmap for Agent Based Computing. AgentLink, 2003, http://www.agentlink.org/roadmap

[6] Jennings, N. R., Sycara, K., Wooldridge, M.: A Roadmap of Agent Research and Development, in Journal of Autonomous Agents and Multi-Agent Systems, Vol. 1(1), 1998, pp. 7-38

[7] Bauer, B., Odell, J.: UML 2.0 and Agents: How to Build Agent-based Systems with the new UML Standard, Journal of Engineering Applications of Artificial Intelligence, Volume 18, Issue 2, March 2005, pp. 141-157

[8] Fallah-Saghrouchni, A. E., Suna, A.: Programming Mobile Intelligent Agents: an Operational Semantics, in Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'04), 2004, pp. 65-71

[9] Berger, M., Bouzid, M., Buckland, M., Lee, H., Lhuillier, N., Olpp, D., Picault, J., Sheperdson, J.: An Approach to Agent-based Service Composition and Its Application to Mobile Business Processes, IEEE Transactions on mobile computing, Vol. 2, No. 3, July-September 2003, pp. 197-205

[10] Mařík, V., McFarlane, D.: Industrial Adoption of Agent-based Technologies. IEEE Intelligent Systems, January/February 2005, pp. 27-35

[11] Chen, H., Perich, F., Chakraborty, D., Finin, T., Joshi, A.: Intelligent Agents Meet Semantic Web in a Smart Meeting Room, in Proceedings of 3rd International Joint Conference on Autonomous Agents and Multiagent Systems, July 2004, pp. 854-861

[12] Cooney, D., Roe, P.: Mobile Agents Make for Flexible Web Services, in Proceedings of the 9th Australian World Wide Web Conference AUSWEB 2003, July 5-9, 2003, pp. 385-391

[13] Paralič M., Paralič J.: Data Analysis for Agent-based Mobile Services, Journal of Information and Organizational Sciences, Vol. 29, No. 1, 2005, ISSN 0351-1804, pp. 53-61

[14] Shehory, O., Sturm, A.: Evaluation of Modeling Techniques for Agent-based Systems. In Proceedings of the 5th International Conference on Autonomous Agents, Montreal, Canada, June 2001, pp. 624-631

[15] Zambonelli, F., Jennings, N. R., Wooldridge, M.: Developing Multiagent Systems: The Gaia Methodology, ACM Transactions on Software Engineering and Methodology, Vol. 12, No. 3, July 2003, pp. 317-370

[16] Deloach, S. A., Wood, M. F., Sparkman, C. H.: Multiagent Systems Engineering. International Journal of Software Engineering and Knowledge Engineering. Vol. 11, No. 3, 2001, pp. 231-258

[17] Web Service Architecture Requirements, W3C Working Group, February 2004, http://www.w3.org/TR/2004/NOTE-wsa-reqs-20040211/

[18] Huhns, M. N.: Agents as Web Services. IEEE Internet Computing, 6(4), 2002, pp. 93-95

[19] Martin, D., et al: Bringing Semantics to Web Services: The OWL-S Approach, SWSWPC 2004, Lecture Notes in Computer Science 3387, 2005, pp. 26-42

[20] Haller, A., Cimpian, E., Mocan, A., Oren, E., Bussler, Ch.: WSMX – A Semantic Service-oriented Architecture, in Proceedings of the International Conference on Web Services (ICWS 2005), IEEE Computer Society Orlando, Florida, USA, 2005, pp. 321-328

[21] Ankolekar, A., Martin, D., McGuinness, D., McIlraith, S., Paolucci, M., Parsia, B.: OWL-S' Relationship to Selected Other Technologies, November 2004, http://www.w3.org/Submission/2004/SUBM-OWL-S-related-20041122/