INFRAWEBS SEMANTIC WEB SERVICE DEVELOPMENT ON THE BASE OF KNOWLEDGE MANAGEMENT LAYER

Joachim Nern, Gennady Agre, Tatiana Atanasova, Zlatina Marinova, András Micsik, László Kovács, Janne Saarela, Timo Westkaemper

Abstract: The paper gives an overview about the ongoing FP6-IST INFRAWEBS project and describes the main layers and software components embedded in an application oriented realisation framework. An important part of INFRAWEBS is a Semantic Web Unit (SWU) – a collaboration platform and interoperable middleware for ontology-based handling and maintaining of SWS. The framework provides knowledge about a specific domain and relies on ontologies to structure and exchange this knowledge to semantic service development modules. INFRAWEBS Designer and Composer are sub-modules of SWU responsible for creating Semantic Web Services using Case-Based Reasoning approach. The Service Access Middleware (SAM) is responsible for building up the communication channels between users and various other modules. It serves as a generic middleware for deployment of Semantic Web Services. This software toolset provides a development framework for creating and maintaining the full-life-cycle of Semantic Web Services with specific application support.

Keywords: Semantic Web Services, Fuzzy Set, Ontologies, Case-Based Reasoning

ACM Classification Keywords: H.3.4 Systems and Software: Information networks

Introduction

The Infrawebs open platform and framework is divided into 3 layers:

- a knowledge management layer
- a service development layer
- a service deployment layer

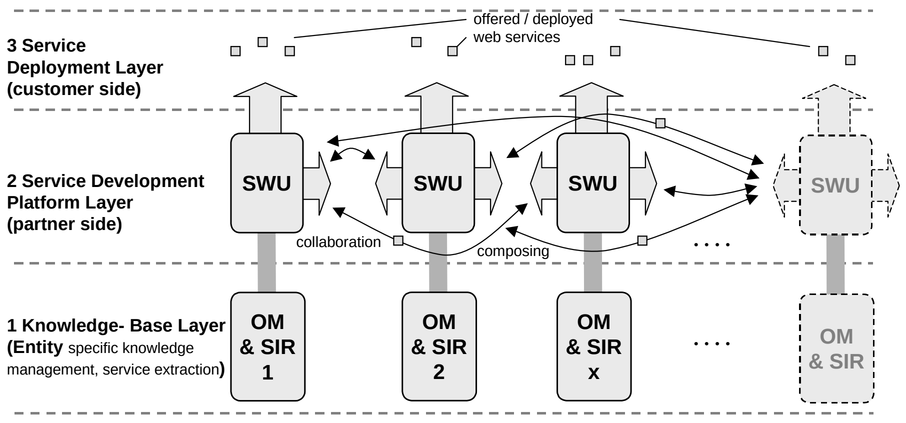
The project's current relation to the state of the art in services engineering and semantic web service area is characterised by a principally innovative approach: based on a bottom-up approach. The project tries to comprehensively design and establish a software tool set as well as a modelling framework covering the full life cycle of semantic web services. The innovative feature is given in the degree of comprehending this life cycle by embedding the cycle components in closed loop structures.

It starts from the "bottom" (knowledge management layer) by providing contemporary as well as future-oriented knowledge management tools, semantic information routing, and enrichment facilities for knowledge objects, which represents the conventional service generation and handling process. This semantic based knowledge management layer is grounded to existing well defined standards like WSDL, SOAP, UDDI and acts as well structured connection to the existing web service related world.

The semantically enriched knowledge artefacts are "shifted to" and "accessed by" a service development layer (Semantic Web Service Unit -SWU) via the novel SPARQL RDF query language. This layer provides tools for creating and composing of Semantic Web Services embedded in a semantic based interoperable middleware, consisting of Semantic Web Service Designer & Composer, Distributed Semantic Web Service Registries, and discovery modules. The INFRAWEBS specific Semantic Web Services are WSMO [WSMO] compliant to parallel European research efforts.

As the top of the overall structure the service deployment layer provides tools for the execution and monitoring of Semantic Web Services. Extracting execution and monitoring information (Quality of Service Brokering) and feeding back this information to the underlying layers guarantees a stable bottom-up to top-down cycle, which inherently optimises itself.

Generated in this way, the open platform (Fig. 1) consists of coupled and linked INFRAWEBS units, whereby each unit provides tools and system components to analyse, design and maintain WEB-Services realised as Semantic-Web-Services within the whole life cycle.



offered / deployed web services
3 Service Deployment Layer (customer side)
2 Service Development Platform Layer (partner side)
SWU
SWU
SWU
SWU
collaboration
composing
....
1 Knowledge- Base Layer (Entity specific knowledge management, service extraction)
OM & SIR 1
OM & SIR 2
OM & SIR x
....
OM & SIR

Fig. 1: Open and extensible development platform for the design, deployment and maintenance of Semantic Web Services as a net of coupled INFRAWEBS units.

As illustrated in Fig.1 the overall design is structured in three main layers:

1) a knowledge management layer for handling service related knowledge artefacts realised as an organisational memory coupled to semantic information routing components (OM&SIR),

2) a service development layer for creating and maintaining Semantic Web Services embedded in a semantic based interoperable middleware, consisting of Semantic Web Service Designer & Composer, Distributed Semantic Web Service Registries, and an agent based discovery module (Semantic Web Service Unit -SWU)

3) a service deployment layer for the execution and monitoring of Semantic Web Services exploiting closed loop feedback information (Quality of Service brokering) provided for distributed decision support issues.

The software tool-set building components map the specific modules of the INFRAWEBS framework. With regard to the SWS technology the position of the project is to leave the conceptual level towards practical and reasonable applicable software tools and components. In the rest of the paper the knowledge management and service development layers are considered in detail.

Knowledge Management Layer

Concerning the knowledge management layer, the base module for the organizational memory (OM) is specified and designed as a Fuzzy Concept Matching OM (FCM-OM). It acts as a repository for semi-structured knowledge artefacts (knowledge content objects).

The OM management module is endowed with tools for knowledge acquisition and knowledge representation [Nern, 2005a]. This module is responsible for the collection, organisation, refinement, and distribution of knowledge objects handled and managed by the service providers. The current specification and realisation of the INFRAWEBS OM is based on the novel results of research activities in the area of Fuzzy Set theory:

Considering the ambiguity, imprecision of concepts (electronic knowledge objects, knowledge content objects of an entity, including the objects handled and received in Internet related environments) a useful approach is the adaptation and application of FCM (Fuzzy Conceptual Matching) methods [Zadeh, 2002]. Within this approach a "concept" is defined (and represented) by a sequence (a set) of weighted keywords. Ambiguity in these concepts is defined by a set of imprecise concepts. Each imprecise concept is defined as a set of fuzzy concepts (using methods of implicit semantics), which is related to "a set of imprecise terms representing the context" [Zadeh, 2002]. Involving and considering also formal semantics, these imprecise terms (words) are "translated" into precise terms (words) formalised as an ontology.

Within the INFRAWEBS project two streams are focused in realising this system component:

- one component of the OM (FCM-OM) is designed following the rules of implicit & soft semantics (using statistical based AI methods like clustering and classification)
- a second component (O-OM) reflects the Knowledge handling based on methods related to formal semantics (Ontologies) – hard semantics [Zadeh, 2002].

The FCM-OM is coupled to the SIR – a Semantic Information Router [Westkaemper, 2005], that is a further module within the first platform layer.

Semantic and non-semantic components of INFRAWEBS are interconnected by SIR, which is responsible for:

- Locating all resources needed for problem solving either in the local SWU or outside.
- Creation of non-semantic content (knowledge objects) by means of semantic content stored in the Distributed SWS Repository.
- Creating an effective system of indexes allowing fast communication between semantic and non-semantic modules of SWU.

- INFRAWEBS proposes a bottom-up approach to the problem of converting regular Web services to semantic ones. Based on some initial description of a Web service to be converted, SIR finds the appropriated WSDL files corresponding to the query and extracts metadata information about the service (in Dublin Core metadata standard) [Dublin Core] from the UDDI (if the service is registered in it). Such metadata is further used by the Service Modeller (a sub-component of the INFRAWEBS CBR Service Designer tool) for filling in non-functional properties of the constructed semantic Web service according to the WSMO Framework.
- The interface between the Semantic Information Router (SIR) and the OM is based on the SOAP protocol. The interface is used to provide WSDL based service descriptions to the OM component and to query for content item references that are related to given service description references.
- The query interface of the SIR component is based on the SPARQL which is a RDF metadata query language designed by the W3C DAWG working group with protocol bindings defined for JDBC, HTTP and other protocol stacks. Technically the SPARQL interface is a metadata query language with syntax close to SQL which offers querying metadata as table and graph result sets. The protocol stack most suitable for J2EE environments is a JDBC type 4 (Java Database Connectivity) compliant drivers. The implementation of a JDBC based protocol binding for SPARQL is named SPARQL4j (http://sourceforge.net/projects/sparql4j).
- Using this RDF query language the connection to the service development layer is performed. Registration of non-semantic atomic services is accomplished via a web-based GUI interface. This interface is mainly used to input WSDL and BPEL4WS based service descriptions and enter additional related non-functional properties. Additionally a UDDI registry interface is provided for the SIR component. This will be established via SOAP as a UDDI Subscription Listener.

Service Development Layer - Semantic Web Unit (SWU)

The main module within the second – the service development – layer is the Semantic Web Unit (SWU) [Atanasova, 2005], [Nern, 2005b], [Agre, 2005]. SWU provides knowledge about a specific domain and relies on ontologies to structure and exchange this knowledge.

The following challenges for developing of SWU have to be taken into account:

- Converting Web Services from available descriptions and domain knowledge (organizational memory) to the semantic ones;
- Composition of Web Services, combining and orchestrating them in order to deliver added-value services;

- Dependencies that arise when a service integrates with external services and becomes dependent on them;
- Integration Processes of several business partners situated on different locations that have to be integrated with each other.

SWU is embedded in INFRAWEBS Environment, responsible for communicating with different users, agents and other SWUs.

SWS Unit ensures designing SWS from the domain knowledge. All knowledge objects from the organizational phenomena influent on the constructed semantic web service and consist of WSDL, BPEL4WS and UDDI files as web service descriptions together with WSML and metadata as ontologies and non-functional properties carriers.

One promising solution to SWS design is to define a library of reusable aspects that would allow the service developer to dynamically instantiate and configure all the needed aspects to deal with different SWS parts. These reusable aspects can be seen as generic templates that can be customized and integrated “on demand” to accommodate to service requirements. This consideration leads naturally to using Case-Based Reasoning approach for service development.

Within the SWU the Designer and Composer modules are responsible for decision supported creation of Semantic Web Services using the Case-Based Reasoning approach. The Designer is a tool for semiautomatic conversion of non-semantic Web services to Semantic Web Services, whereas the Composer enables the semiautomatic creation of new Semantic Web Services via the composition of existing Semantic Web Services.

The architecture of both modules is based on such general principles as:

- Specialization: Each tool is carefully designed based on analysis of specificity of the task it is intended to be used for. It leads to minimization of efforts the service provider should apply for creating a semantic web service. Such minimization is achieved via fully utilization of all available information resources about the service as well as CBR-based mechanism for improving the behaviour of a tool through accumulating and using experience of the service provider to work with this tool.
- User-friendliness: it is assumed that the users of our tools will be semantic Web service providers as well as customers of such services. In both cases the users will not be specialists in first-order logic that is why we implement a self-explained graphical way for constricting and editing of all elements of a semantic web service.
- Intensive use of ontologies: ontologies are the core concept of the Semantic Web technology; however, we consider that creating ontologies for different application domains requires very intensive cooperation of highly qualified domain knowledge engineers and logicians. Both categories of the users do not belong to the range of potential customers of our tool. That is why we assume that our customer will be mainly a user of already created ontologies rather then a creator of new ontologies. However, we foresee that in some cases the service providers have to be able to create some specialized versions of (general) existing ontologies. Means for creating such (restricted) ontologies are also included in our tools.
- Semantic consistency: operation with each tool is organized via ontology-based system-driven interaction with the service creator, which prevents him/her from possible errors and allows being concentrated on the relevant part of knowledge to be acquired. Application of context-sensitive syntactical and completeness checks at each step of the semantic service creation prevents the user from constructing semantically inconsistent and incomplete models.

Designer With the Case-based Designer SWS a service provider creates semantic descriptions of the services on the base of set of ontologies, preferences (QoS) and business logic of services using service design templates (DST).

The Designer consists of several sub-modules responsible for WSMO compliant creation of main elements of INFRAWEBS specific Semantic Web Service.

SWS-Designer has to add the semantic meaning to Web Services about: Data, Functioning, Execution, Discovery, and Selection. This can be done by the following modules: Capabilities editor, Interface editor, Grounding editor via using of DST with appropriate validation and indexing. Creating, storing, and retrieving of similar DST are organized using Case-based Reasoning (CBR) approach. A retrieved template can be further used or adapted by the user for designing the desired functional model of a new semantic Web service and/or to be stored in case-based memory for later re-use.

As a graphical user-friendly tool the Capability Editor facilitates construction and editing of complex WSML-based logical expressions used for representing service capabilities. The BPEL4WS-based editor serves for creating WSMO-based service choreography and orchestration as SWS interface. The Grounding Editor provides facilities for semiautomatic creating of WSMO-based grounding on the base of WSDL descriptions.

A first prototype of an Axiom Editor - an ontology-driven user-friendly tool for graphical creating complex WSML logical expression - was developed as a part of INFRAWEBS SWS Designer. This module is the main part of the INFRAWEBS Designer that is responsible for creating the capability description of a Semantic Web Service according to WSMO framework.

A basic feature of the Designer is the use of Design Service Templates (DST), representing graphical models of capability and functionality (or their parts) of Semantic Web Services, which have been designed by the user in the past.

Composer With the Case-Based Semantic Web Service Composer a service provider constructs SWS semi-automatically in design-time by composing descriptions of existing SWS and using domain knowledge.

The SWS Composer has to resolve two problems during composition: planning of the process (service scenarios) and orchestration of services. It uses the previous service compositions that form the general tasks. Such compositions are represented by Service Composition Templates (SCT).

The SWS Composer provides:

- Similarity-based retrieval of an appropriate semantic service template based on the description of capability of the desired service and description of its functionality
- Semi-automatic adaptation of service functional model based on the results of discovery of sub-services matching the template proxies
- Advertising the created service and its generalization and storing as new template for later re-use.

The Case Base of SCT consists of references to complex service scenarios constructed by SWS Composer in the past and associations between problem solutions (particular description of request for servicing made in the past) and founded solutions.

The Composer presents an interactive approach for composition of WSMO compliant Semantic Web Services. The SCT represents graphical models of the service composition as a control and a data flow between several semantic sub-services given by incomplete description of their capabilities. On the (Service) provider side the Composer enables the creation of a new composed “static” Semantic Web Service by discovering appropriate Semantic Services matching the required capabilities. Selecting of such services is implemented as an interactive system-driven semiautomatic process.

Eclipse RCP (Rich Client Platform) is used for developing the basic platform components; plug-in infrastructure and graphical user interface components, whereas Eclipse GEF (Graphical Environment Framework) is the basis for implementing the graphical editors. Access to WSMO-based repositories (ontologies, Semantic Web Services, etc.) is realized via the WSMO API.

WSMO API - WSMO4J For ensuring compatibility and interoperability within and between the INFRAWEBS framework modules the WSMO API (WSMO4J) is applied. The WSMO4J is an open-source project (distributed under a LGPL licence) with two parts: a) WSMO API - application programming interfaces for WSMO, which allow basic manipulation of WSMO descriptions, e.g. creation, exploration, storage, retrieval, parsing, and serialization and b) WSMO4J - a reference implementation of the WSMO API, including a WSML parser.

One of the major advantages of using the WSMO API in INFRAWEBS is to assure the compatibility and interoperability between the SWS Designer and Composer modules, and the repository component. The distributed SWS registry uses WSMO4J in the process of transforming WSMO element descriptions into RDF triples stored into an RDF triple repository for efficient query and management. Using WSMO4J enables easy integration and interoperability within the framework as well as with the WSMO Studio, thus some of the components can be realized as extensions (plug-ins) for the WSMO Studio.

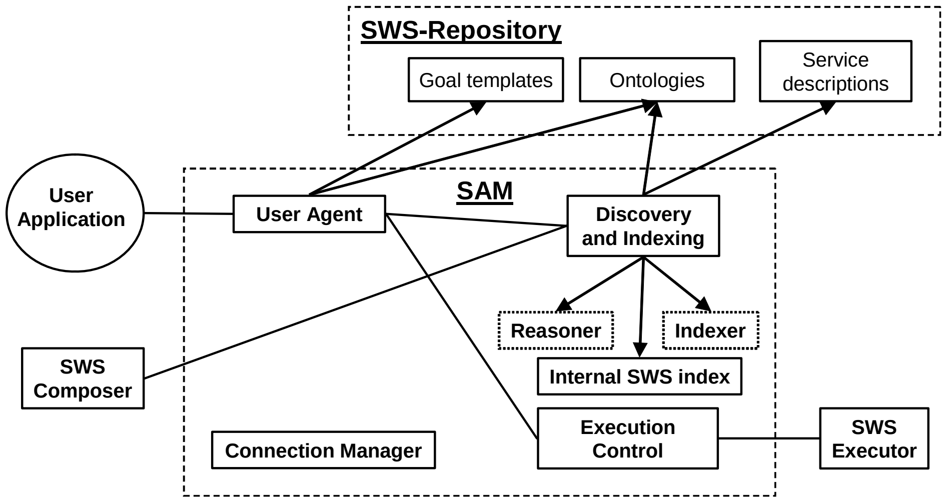


Fig. 2: Internal Architecture of Middleware Layer and its Dependencies with Other Modules

SWS Discovery - SAM The SAM (Service Access Middleware) [Kovacs, 2005] unit is a basic layer of the INFRAWEBS software environment (Fig. 2), guiding the user applications through the steps of semantic web service discovery, selection and execution. The Connection Manager is responsible for building up the communication channels between users and various other modules. The User Application circle denotes the application acting on behalf of the user: for example a GUI interface or an intelligent agent. The User circle and the SWS Composer need to discover and select existing web services with specified capabilities. The Discovery component supports this task, while the Execution Control component oversees the execution of the selected web service in cooperation with the SWS Executor module.

The discovery process is planned as a hybrid approach combining text processing and reasoning. Therefore, this process needs other external information and tools as well: domain ontologies contain the background information about service domains and organizational memories gathered from the INFRAWEBS framework and transformed into Semantic Web compatible format.

The SAM unit within INFRAWEBS acts as a middleware layer for user applications, connecting these applications to the functionality available in the form of Semantic Web Services. An important decision is whether to hide the semantic approach and accompanying logic-based framework of SWS from the user applications or not. A present investigation is to clarify: What level to choose for communication? On the level of plain web services, simple XML data structures are exchanged, which are easy to generate and process, but lack the possibilities of the semantic approach. On the level of semantic web services, facts and rules are exchanged in the form of logical expressions. As the latter case puts extra requirements for the user application, it is decided to find a middle course between the two solutions. Communication is kept on the semantic level, but its form is either hidden or aided with special functionalities and automatic translations.

Apart to SWS discovery the simplest way of discovery is based on keyword matching within the descriptive metadata of web services (similar to the UDDI approach). The most complex method is to logically prove that the web service is able to fulfil the given goal. Currently, response time of discovery process is a significant trade-off for these approaches.

The keyword-based method is improved if the goals and capabilities are used for keyword generation instead of metadata. Goals and capabilities are described using ontology terms, so more homogeneous and precise keywords can be extracted this way. The logic-based methods differ in aspects of goals and capabilities considered. It is simpler and faster to match only by post-conditions, but then matching services might not be executable because of missing or unacceptable information (input).

The project approach is to split the discovery into two phases. In the first phase, the keywords of capabilities are used to filter the possible web services. Then, logical matching is applied only for the filtered services.

The list of matching web services is returned to the user application enriched with descriptive and qualifying metadata.

A Web service is the access point to the real service: its quality determines the quality of access, but might be independent of the quality of service (e.g. automatic translation or flight reservation). Therefore, an iterative selection process (similarly to iterative query refinement in information retrieval) guides the selection of the best service in INFRAWEBS. This is based on a simple two-phase workflow agreement (or business logic) between the web services and the middleware layer.

The Service Access Middleware (SAM) provides support for the usual steps of goal construction, discovery, selection and execution of Semantic Web Services. This support is achieved through a neutral interface, which hides the complexities of Semantic Web Services, therefore applications can be easily adapted to it, and also it can be equivalently used in variants of Semantic Web Services, such as WSMO and OWL-S. SAM also features an iterative selection refinement process for finding not only the suitable web services, but also the best service offers for users' goals.

SWS deployment layer The SWS deployment layer consist of SWS executor that is split up into three main components, namely, the Communication Manager, Choreography Engine and the Invoker [Polleres, 2005] and QoS broker. At present it is defined that the executor should mainly interact with the distributed registry and the SAM components.

Conclusion

The primary objective of the INFRAWEBS project is to develop an ICT framework consisting of several specific software tools, which enables software and service providers to generate and establish open and extensible development platforms for Semantic Web Service based applications [Nern, 2004]. This software tool set facilitates the establishment of virtual development platforms as well as interoperable middleware designed for a semantic and ontology-based handling of Semantic Web Services oriented on given conception WSMO specifications.

One of the goals of the INFRAWEBS project is the development of a SWS full-life-cycle software toolset for creating and maintaining Semantic Web Services with specific application support. An important part of INFRAWEBS is a Semantic Web Unit (SWU) – a collaboration platform and interoperable middleware for ontology-based handling and maintaining of SWS. The SWU provides knowledge about a specific domain and relies on ontologies to structure and exchange this knowledge.

INFRAWEBS Designer and Composer are sub-modules of SWU responsible for creating Semantic Web Services using Case-Based Reasoning approach to fulfil decision support demands.

The architecture of both modules is based on such general principles as:

- service-oriented architecture with bottom-up approach for semi-automatic constructing of semantic web services;
- system driven syntactic consistent and completeness checking;
- past experience utilizing.

The SAM unit within INFRAWEBS acts as a middleware layer for user applications, connecting these applications to the functionality available in the form of Semantic Web Services. These software toolsets are developing for creating and maintaining of full-life-cycle Semantic Web Services with specific application support.

Bibliography

[Nern, 2004] H Joachim Nern, G. Agre, T. Atanasova, J. Saarela. System Framework for Generating Open Development Platforms for Web-Service Applications Using Semantic Web Technologies, Distributed Decision Support Units and Multi-Agent-Systems - INFRAWEBS II. In: WSEAS TRANS. on INFORMATION SCIENCE and APPLICATIONS, 1, Vol. 1, 286-291, 2004.

[WSMO] Web Services Modelling Ontology, SDK WSMO working group, http://www.wsmo.org

[Nern, 2005a] H Joachim Nern, A Dziech, E Tacheva, Fuzzy Concept Sets (FCS) applied to semantic organizational memories within the Semantic Web Service designing and composing cycle, In: Proc. 1st Workshop for "Semantic Web Applications" at the EUROMEDIA 2005, IRIT, Université Paul Sabatier, Toulouse, France, April 2005.

[Zadeh, 2002] Lotfi A. Zadeh. Toward a perception-based theory of probabilistic reasoning with imprecise probabilities. In: Journal of Statistical Planning and Inference 105, 233-264, 2002.

[Westkaemper, 2005] T Westkaemper, J Saarela, H Joachim Nern, Semantic Information routing as a pre-process for Semantic Web Service generation - SIR & OM, In: Proc. 1st Workshop for "Semantic Web Applications" at the EUROMEDIA 2005, IRIT, Université Paul Sabatier, Toulouse, France, April 2005.

[Dublin Core] The Dublin Core Metadata Initiative, http://dublincore.org/

[Atanasova, 2005] Tatiana Atanasova, Gennady Agre, H Joachim Nern, “INFRAWEBS Semantic Web Unit for design and composition of Semantic Web Services INFRAWEBS approach”, In: Proc. 1st Workshop for "Semantic Web Applications" at the EUROMEDIA 2005, IRIT, Université Paul Sabatier, Toulouse, France, April 2005.

[Nern, 2005b] H Joachim Nern, A Dziech, T Atanasova, Applying Clustering and Classification Methods to distributed Decision Making in Semantic Web Services Maintaining and Designing Cycles, EUROMEDIA 2005, In: Proc. Workshop for "Semantic Web Applications", IRIT, Université Paul Sabatier, Toulouse, France, April 11-13, 2005.

[Agre, 2005] Gennady Agre, Tatiana Atanasova, H Joachim Nern, “Case Based Designer and Composer”, In: Proc. 1st Workshop for "Semantic Web Applications" at the EUROMEDIA 2005, IRIT, Université Paul Sabatier, Toulouse, France, April 2005.

[Kovacs, 2005] L Kovacs, A Micsik, “The SUA-Architecture within the Semantic Web Service Discovery and selection process”, In: Proc. 1st Workshop for "Semantic Web Applications" at the EUROMEDIA 2005, IRIT, Université Paul Sabatier, Toulouse, France, April 2005.

[Polleres, 2005] A Polleres, J Scicluna, “Semantic Web Execution for WSMO based choreographies”, In: Proc. 1st Workshop for "Semantic Web Applications" at the EUROMEDIA 2005, IRIT, Université Paul Sabatier, Toulouse, France, April 2005

Authors' Information

Joachim Nern – Scientific coordinator of INFRAWEBS project; big7.net GmbH & Aspasia Knowledge Systems Germany, e-mail: nern@aspasia-systems.de

Gennady Agre – Institute of Information Technologies, Acad. G. Bonchev 29-A, 1113 Sofia, Bulgaria, e-mail: agre@iinf.bas.bg

Tatiana Atanasova – Institute of Information Technologies, Acad. G. Bonchev 2, 1113 Sofia, Bulgaria, e-mail: atanasova@iinf.bas.bg

András Micsik – MTA SZTAKI, H-1111 Budapest XI. Lagymanyosi u. 11, Hungary, e-mail: micsik@sztaki.hu

László Kovács – MTA SZTAKI, H-1111 Budapest XI. Lagymanyosi u. 11, Hungary, e-mail: Laszlo.kovacs@sztaki.hu

Zlatina Marinova – SAI, Ontotext Lab, Sirma Group Corp., 135 Tsarigradsko Shosse Blvd.,1784 Sofia, Bulgaria, e-mail: zlaty@sirma.bg

Janne Saarela – Profium Ltd, Lars Sonckin Kaari 12, 02600 Espoo, Finland, e-mail: janne.saarela@profium.com

Timo Westkaemper – Profium Ltd, Lars Sonckin Kaari 12, 02600 Espoo, Finland, e-mail: timo.westkamper@profium.com