Stefan Schulte, Julian Eckert, Nicolas Repp, Ralf Steinmetz: An Approach to Evaluate and Enhance the Retrieval of Semantic Web Services. In: IEEE: 5th International Conference on Service Systems and Service Management (ICSSSM'08), p. 237-243, IEEE, June 2008. ISBN 9781424416721.

An Approach to Evaluate and Enhance the Retrieval of Semantic Web Services

Stefan Schulte, Julian Eckert, Nicolas Repp and Ralf Steinmetz, Fellow, IEEE

Multimedia Communications Lab (KOM), Technische Universität Darmstadt, Germany
e-mail: schulte@kom.tu-darmstadt.de

***Abstract*—In order to achieve the (semi-) automatic composition of Web services to business processes and workflows respectively, it is necessary to make use of semantic information regarding the capabilities and further characteristics of particular Web services. In recent years, a couple of approaches have been proposed to achieve (semi-) automatic annotation, retrieval, and composition of Semantic Web services; however, differences in Web service standards and repositories used for the evaluation of these approaches lead to a lack of both in-depth evaluations and comparability of the proposals. In this paper, we present a workbench to test and evaluate the performance of annotation, retrieval, and composition algorithms for Semantic Web services. As a proof of concept we introduce SEM.KOM, a prototypical implementation based on standardized Web service technologies.**

Index Terms—**Semantic Web services, Service Interface Matching, Service Matchmaking**

I. Introduction

In order to achieve the (semi-) automatic composition of Web services to business processes and workflows, it is crucial to address the retrieval of appropriate services. Only if these services can be identified is it possible to address further problems, such as, Quality of Service [1], and to include a service in a workflow. Unfortunately, a syntactic description of a Web service's functionalities and characteristics is only sufficient if all possible parties (i.e., service providers, service brokers, and service requestors) use the exact same vocabulary. It is quite unlikely that this requirement is fulfilled even in a corporate environment. Hence, it is necessary to enrich Web service descriptions with semantic annotations.

Since the first presentation of semantic markup of Web services in 2001 [2], the (semi-) automatic annotation, retrieval, and composition of Semantic Web services (SWS) has been a research topic of great interest. Several different approaches to achieve these goals have been proposed. However, due to differences among chosen Web service technologies and repositories, there are currently no extensive comparisons of the miscellaneous proposals. As stated by [29], only little effort has been put into such evaluations.

Furthermore, the proposed approaches often refer to a particular domain or intention of their authors. Even though techniques used for SWS annotation could be used for SWS retrieval as well, they are only seldom transferred from one area of application to another.

One aspect that is often considered to be only of secondary importance is the performance of the proposed algorithms with regard to computation time. Nevertheless, this quality aspect is especially relevant if addressing real time applications involving a large number of potential Web services [23].

Furthermore, the applicability of the proposed approaches for SWS retrieval to real world problems has to be questioned due to the current lack of appropriate ontologies.

To overcome these issues, we present SEM.KOM, a workbench for testing and evaluating the performance of SWS annotation, retrieval, and composition algorithms. Our aim is to identify the "best of breed"-approaches in terms of computation time, precision, and retrieval.

The remaining part of this paper is structured as follows. In the next section, basic concepts of SWS retrieval are presented and relevant related work is introduced. A means of evaluating (semi-) automatic annotation, retrieval, and composition of SWS is described in Sect. III. Then in Sect. IV, we introduce our evaluation workbench SEM.KOM. The paper closes with a summary of our findings and an outlook on our future work.

II. SWS Retrieval

In recent years, different organizations and collaborations between companies have achieved the standardization of XML-based Web service technologies. In particular, the *Web service Description Language* (WSDL) [3] has been established as one of the key building blocks in the Web service stack. Further standards like the *Web Services Business Process Execution Language* (WS-BPEL [4]) and *Universal Description Discovery and Integration* (UDDI [5]) are also widely accepted as standardized Web service technologies.

Because these technologies are primarily motivated by the interoperability of software components over the Web, they rely on the pure syntactical description of Web services' functionalities and properties. Therefore, several suggestions for how to enhance the specifications with semantic annotations have been made in order to establish SWS, e.g., *Semantic Annotations for WSDL and XML Schema* (SAWSDL [6]). A BPEL extension for SWS has been proposed by [7].

This work is supported in part by the E-Finance Lab e.V., Frankfurt am Main, Germany (http://www.efinancelab.com).

978-1-42

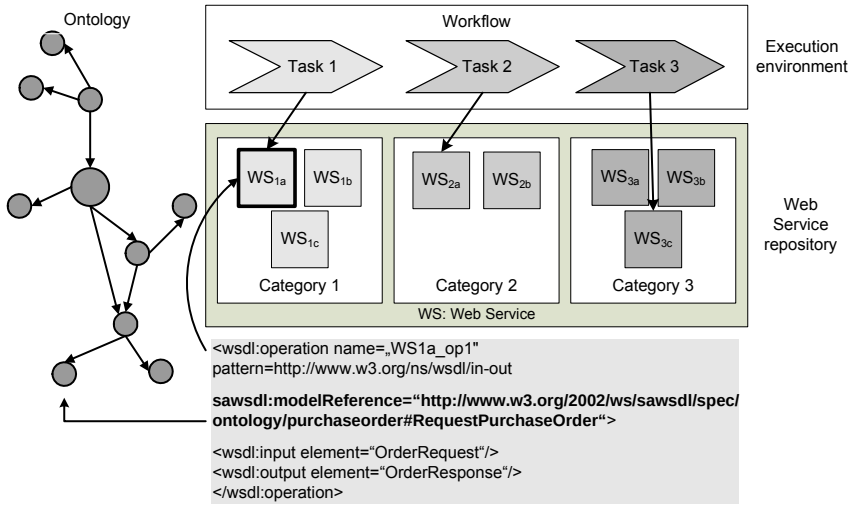

Fig. 1. Semantic annotation of Web services using SAWSDL and an ontology

In the following subsections we will present the basic concepts of SWS retrieval and related work, including a brief illustration of basic constructs in SWS, approaches to and drawbacks for SWS retrieval, and enhanced approaches to SWS retrieval.

A. Basic Constructs in SWS and Standardization Efforts

The Semantic Web vision is based upon the offering of metadata and the association of resources with this metadata. SWS make use of this concept – the metadata is mostly available in an ontology, i.e., a finite, controlled (extensible) vocabulary that provides an unambiguous interpretation of its classes and the strict hierarchical subclass relationships between them [9].

The semantic metadata provided by an ontology has to be linked to the different operations, inputs, outputs, and interfaces of a Web service description.

SAWSDL and OWL-S ("OWL for Services") [8] are popular examples for implementing the concept of semantics for Web services. While SAWSDL adds semantic annotations to WSDL (cp. Fig. 1), its specification does not restrict which language is used to detail the related semantic model. It is possible to embed semantic models in the WSDL document, but in most cases, service developers will employ an external ontology to define semantic descriptions, e.g., using the Web Ontology Language (OWL).

In OWL-S, the semantic concepts are not annotated inside a WSDL description. On the contrary, a Web service description in OWL-S consists of three parts: a *profile ontology*, a *process ontology*, and a *grounding ontology*. The first ontology describes what the service does and the second how the service is used. The actual description of how to interact with the service is described in the third ontology; here, each OWL-S atomic process is mapped to a corresponding WSDL operation [8].

In contrast to this *explicit* or *formal* capability representation of Web services, *implicit* semantics are not represented by concepts in an ontology. Implicit semantic information might be the result of a state transformation but can also be derived from syntactic textual descriptions of a Web service's capabilities. Hence, on the one hand, it is not necessary to provide an ontology, but on the other hand, it is more difficult to identify capabilities from an implicit representation [10]. Today, most research efforts focus on the use and facilitation of explicit semantic information; nevertheless, it might be useful to fall back on implicit semantic information (cp. Sect. IID.).

B. Approaches to SWS Retrieval

SWS retrieval is based on a matchmaking engine, i.e., an algorithm that finds the best fitting Web services for a precise service request. There is no limitation on the technologies used by the algorithm, the form of the request, the number and the sequence of the "best fitting" services, or which service feature is retrieved. If there were perfect semantic annotations and no lack of information, we would easily get a match between a request and an existing Web service description; however, this is relatively uncommon, especially in heterogeneous environments with a large number of service providers and requestors.

In such an environment, it is more likely that at least one of the parties involved does not know how to request or advert a particular service correctly [11]. Furthermore, it is still possible that a requested service is simply non-existent but that the requested functionalities can still be provided by one or more other services.

Hence, several authors have proposed different kinds of matchmaking based on the degree of conformity between requests and Web service descriptions. For example, [10] and [11] propose the matching of capabilities: A service is deemed to be of use for the requestor if all outputs requested are matched by the outputs advertised and if all inputs needed by the service advertised can be covered by the inputs provided by the requestor. Matches between inputs/outputs requested and advertised are categorized into

SWS are motivated by the automation of information use and dynamic interoperability [8].

exact, *plug in*, *subsumes*, and *fail* matches [11]. This way, it is possible to arrange fitting services by the degree to which they match the inputs/outputs requested.

The service request is expressed as a Web service description that perfectly meets the request; a query in terms of keywords or the ability to browse a service repository is not provided. Hence, it is necessary to identify these inputs and outputs prior to the actual service retrieval, making it more difficult for uninformed users to identify appropriate services.

The four categories *exact*, *plug in*, *subsume*, and *fail* may also be employed to measure the degree to which an advertised Web service can be met by a request. A detailed implementation of this approach is presented in [15]. The authors enhance the four mentioned categories by *intersection*. Still, it is not possible to assess, for example, which of two *plug ins* meets the request better. Xu et al. propose using the semantic distance between concepts in an ontology to extend this categorization and introduce a feasibility to rank Web services [16].

The presented approaches have in common that their ultimate goal is the automatic service composition. For the automatic composition of services and workflows to be possible for a huge number of processes, it is necessary to have many services at hand. Papazoglou suggests the concept of *service markets*, where such services could be offered and requested [12]. In such markets, the involved service providers, service requestors and service brokers rely on an extensive ontology in order to advertise and find the available services. Hence, a clearly defined ontology is the key requirement of the aforementioned approaches to SWS retrieval. This necessity is increased by the annotation of fine-granular objects like inputs and outputs in a Web service description with semantic concepts.

While the aforementioned approaches are necessary when regarding completely automatic service composition, the user could be involved in semi-automatic service composition by identifying proper services a priori. In the following subsection, we present two major drawbacks, which in our opinion, lead to the conclusion that a scenario without extensive service markets and generally accepted ontologies is more realistic.

C. Drawbacks for SWS retrieval

Despite the allure of large service markets where numerous similar services are advertised by different vendors and almost every possible service is provided [12], it is very unlikely that this vision will ever come true for more than clearly defined small domains. In most application areas it will be quite to the contrary – potential service users will browse and search specialized repositories in order to identify useful services.

The second drawback for the aforementioned approaches to SWS retrieval is the shortage of appropriate ontologies. Hepp identifies four reasons why ontologies have not yet had the impact the research community estimated: firstly, because the real world changes fast, it is difficult to keep ontologies up to date; secondly, there is a lack of economic incentives for the individuals contributing to an ontology; thirdly, an ontology is usually created by a small community but intended to be used by a large community, which relies heavily on the ontology documentation in order to understand the semantics, making it quite likely that parts of the community will not comprehend the ontology and will therefore not be able to use it; and finally there is the problem of intellectual property rights, especially if already existing (industrial) standards are adapted in an ontology [14].

All things considered, the development of ontologies is seriously constrained by these technical, social, economic, and legal drawbacks. It might be possible for legal restrictions to demand the introduction of open standards (as in the energy domain [13]), thereby enforcing the development of ontologies and consequently of extensive service markets; however, in most domains this will not be the case. Nevertheless, most of the approaches to SWS retrieval mentioned in previous sections rely heavily on an extensive and accurate ontology in order to semantically annotate Web services.

Hence, it is necessary to enhance current approaches to SWS retrieval by methods that do not have a high degree of dependence on such ontologies.

D. Enhancing existing approaches to SWS

There are several possibilities for overcoming the aforementioned obstacles. Firstly, it is possible to accept the shortage of appropriate ontologies for current SWS retrieval approaches and to push the development of new ontologies towards a more coarse-granular level. Unfortunately, this would also lead to a less exact semantic annotation of Web services, especially when regarding fine-granular parts of the service description.

The second approach is to enhance the explicit capability representation of Web services by the use of implicit semantics. Klusch et al. introduce the concept of *hybrid* SWS matching, which uses both logic-based reasoning (as introduced in [10]) and information retrieval techniques [17].

Semantic annotation on a very coarse-granular level is introduced in ASSAM, e.g. [18],[19]. A similar approach is proposed by [21]. Even though the authors focus on the annotation of SWS, the presented techniques could also be utilized in SWS retrieval [20]. Corella et al. use heuristics instead of machine learning algorithms to achieve the goal of semi-automatic WS classification [22].

In our opinion, both approaches have to be combined because fully automated service retrieval (and hence, service composition) is very hard to achieve if addressing more than clearly defined small domains. Therefore, it seems necessary to change the current view on SWS retrieval. Obviously, SWS retrieval should not be limited to the support of automatic service composition. It is also necessary to develop methods that assist a human being in finding fitting services for a given task even if there is a lack of appropriate ontologies.

This could be done by using a keyword-based search that deploys the syntactic and semantic description of a service or by using a taxonomy that allows the service requestor to

manually browse a service repository. Even though there are reservations about the enforceability of ontologies (cp. Sect. IIC.), existing ontologies should still be used to support the retrieval process. If semantic annotations and a clearly defined ontology are available, these information should be utilized.

An approach for applying these considerations to an evaluation workbench for SWS will be presented in Sect. III.

III. A Workbench to Test and Evaluate SWS Retrieval

In Sect. II we presented current approaches to SWS retrieval and annotation. These approaches differ in the employed algorithms and technologies. Hence, it is difficult to compare their performance with regard to quality metrics like precision and recall [24]. Furthermore, the performance in terms of computation time is often neglected. In order to identify a "best of breed"-approach, it is necessary to test the proposed algorithms in the same environmental setup and with the same set of Web services.

Thus, we present a workbench capable for testing, comparing, and evaluating different approaches to SWS retrieval (cp. Fig. 3).

A. Evaluation workflow

To obtain comparable evaluation results of different approaches to SWS retrieval, it is necessary to deploy exactly the same experimental setup in all test runs. Hence, a workbench must be compatible to be used with different input formats and retrieval algorithms. Furthermore, it is desirable to compare evaluation results of different test runs and may be necessary to substitute the test data set in the service repository.

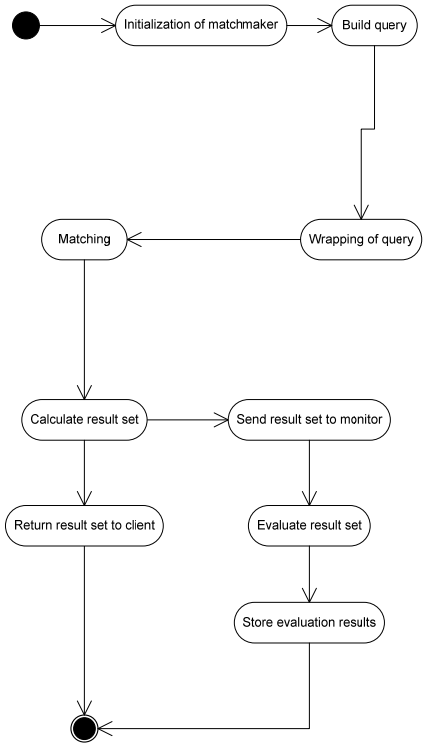

Fig. 2. Activity diagram of matching algorithm evaluation

All things considered, the evaluation workflow has to be made up of the following activities (cp. Fig. 2):

1. *Initialization of matchmaker*: The matching engine(s) is/are imported and initialized; the services and ontology are read in from the repository; a reasoner (if applicable) is provided.
2. The service request is phrased and the *query is build*.
3. The *query is wrapped* respectively converted into a form that can be parsed by the matchmaking engine.
4. The *matching* against all the services loaded from the repository takes place. Performance measures like computation time are metered.
5. A *result set is calculated* and delivered to the requester who is waiting for his query. Furthermore, the monitor is provided with the results.
6. The *result set is evaluated* and compared with former results (optional).
7. Finally, the *evaluation results are stored* in the file system or database and presented to the workbench user.

B. Components

We propose the following architecture to map the workflow presented in the previous section (cp. Fig. 3):

- The *request wrapper* is deployed to allow the application of different kinds of queries. A query could be formulated as a "perfect answer", i.e., a service that would match the request perfectly.

When regarding different Web service standards like SAWSDL, OWL-S, and WSDL, the service request has to be transferred into a consolidated form. It should also be possible to supply a keyword-based query.

- All available services are governed by the *service repository*, which should be able to store all available data about a particular service. As UDDI is per se not able to store semantic information, another solution has to be found. Because we are talking about a testing environment, the repository could be made up of files in a directory or a database. However, it is important that the repository be able to deliver a number of service descriptions to the matchmaker engine if requested.
- The actual *matching engine* is the core component of this architecture. It is possible to choose from different retrieval algorithms or combine them in order to achieve better retrieval results. This component is involved in the evaluation process (cp. Fig. 2) after the service request (i.e., query) has been converted by the request wrapper. It deploys the repository interface to get the descriptions of services in the repository. When the matchmaking process is finished, this component returns a result set that could include all tested services and a corresponding matching score or just a number of "matching" services. The assessment of whether or not a match or matching score is right or wrong should be controllable by the user via parameters.
- A *result monitor* evaluates the result set delivered by the matching engine. The assessment of whether or not a match or matching score is correct depends on the aforementioned parameters. The monitor gives a comparison of matching results, if a former evaluation has taken place. It is possible to store the evaluation results in a database.

A prototypical implementation of this workbench for evaluating SWS retrieval algorithms is presented in the next section.

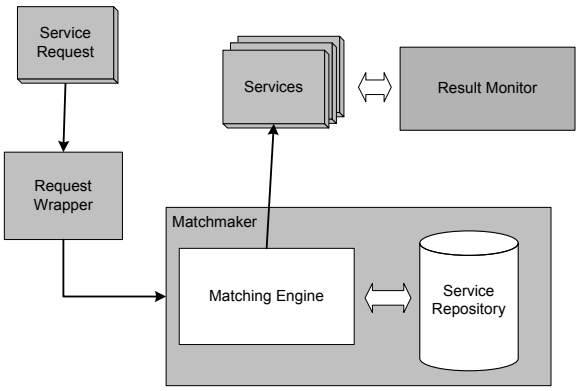

Fig. 3. Workbench architecture

IV. SEM.KOM – Prototypical Implementation

The current version of SEM.KOM is a prototypical implementation of the workbench architecture presented in Sect. III. While we are permanently enhancing the capabilities of our architecture, we have not yet implemented all possible features of SEM.KOM at the moment. In particular, we are planning to put more retrieval algorithms into effect.

Currently, the components mentioned in Sect. IIIB. have been realized in SEM.KOM as follows:

- The *request wrapper* uses an RDF/XML format [26] to convert the service request into a comparable format. It is possible to post the request in terms of a complete OWL-S description. The *Jena Semantic Web Framework* [27] (version 2.5.4) is used to read and write RDF-statements. In order to parse OWL-S, we use OWL-S API 1.1.0 beta (http://www.mindswap.org). Data from OWL-ontologies is directly read from the corresponding files.
- It is possible to choose from two approaches to SWS retrieval in the *matching engine*: the implementation of logic-based reasoning as presented in [10] or keyword-based search. A combination of these approaches is also provided.
- The *service repository* is available in the form of files in a directory and can be accessed via an interface that wraps all advertised services. At the moment, we are deploying OWLS-TC (version 2.2 [25]) as the dataset for testing. In OWLS-TC, every query is associated with a set of relevant services. Hence, the decision if a service in the result set fits the query can be made automatically.
- The *result monitor* provides the quality metrics precision, recall, F1 score (which is the harmonic mean of precision and recall), and average query response time and stores them and all corresponding metadata (i.e., service request, applied service repository).

This paper shows the implementation status at a specific point of time (i.e., April 2008), and therefore provides a snapshot perspective on SEM.KOM. All capabilities of this workbench are constantly being enhanced. The future development of SEM.KOM is presented in the next section.

V. Conclusion and Future Work

In the era of globalization and sophisticated customers, flexible business processes are crucial for the long-term success of an enterprise. One way to achieve flexible business processes is to utilize the (semi-) automatic composition of Web services to processes and workflows. Therefore, it is necessary to get detailed information about the capabilities of these services. Unfortunately, most standards in the Web services stack provide only syntactical information. Hence, it seems consequential to enhance current standards by semantic annotations.

In recent years, a couple of approaches to semantic annotation of Web services have been proposed, with OWL-S, SAWSDL and WSDL-S being the most popular. Different algorithms have been implemented to retrieve Web services based on the semantic information provided by these standards. Unfortunately, in most cases these algorithms are tested within a proprietary experimental setup, making it difficult to compare the different approaches to SWS retrieval. Furthermore, while the performance in terms of precision and recall is frequently evaluated, there is often a lack of evaluations regarding the performance in terms of computation time. In any case, the comparability of evaluations is limited due to different experimental setups, technologies, and test data sets used.

Consequently, we decided to implement SEM.KOM, a software architecture that can be used to evaluate, test, and compare SWS retrieval algorithms. In this paper, we presented the preliminary work to reach the goal of comparable evaluation results regarding SWS retrieval algorithms. Furthermore, we introduced the current implementation status of SEM.KOM.

In the future, we will enhance the capabilities of SEM.KOM with special regard to the following points:

- It will be possible to provide the service request in alternative formats, e.g., as an SAWSDL-based service description.
- More approaches to SWS retrieval will be added to our matchmaker and evaluated. This especially includes approaches that do not make use of logic-based matchmaking (as introduced by [10] and [11]) but have a different focus, for example, machine-learning algorithms as presented in [18]-[22] or case-based reasoning [28].
- If SAWSDL establishes itself as the primary SWS standard, it might be useful to replace the current dataset for testing with a SAWSDL-based service repository.
- At the moment, it is only possible to address ontologies that are available in OWL. In the future, we will apply more ontology standards.
- SEM.KOM will be endowed with a GUI.

We will present results from the evaluations in the near future. Furthermore, we plan to illustrate a more detailed view of SEM.KOM's features in a future publication and make SEM.KOM available to the research community.

As the evaluation of given approaches to SWS retrieval is not an end in itself, we will combine and enhance current approaches in order to improve the evaluation results and support the vision of (semi-) automatic service composition.

References

[1] R. Berbner, T. Grollius, N. Repp, O. Heckmann, E. Ortner, R. Steinmetz, "An approach for Management of Service-oriented Architecture (SoA)-based Application Systems," in *Proceedings of the Workshop Enterprise Modelling and Information Systems Architectures (EMISA 2005)*, Klagenfurt, Austria, 2005, pp. 208-221.

[2] S. A. McIlraith, T. C. Son, and H. Zeng, "Semantic Web services," *IEEE Intelligent Systems*, vol. 16, no. 2, pp. 46-53, 2001

[3] D. Booth, C. K. Liu, "Web services Description Language (WSDL) Version 2.0 Part 0: Primer," WC3 Recommendation, 2007. Available: http://www.w3.org/TR/2007/REC-wsdl20-primer-20070626.

[4] A. Alves, A. Arkin, S. Askary, C. Barreto, B. Bloch, F. Curbera, M. Ford, Y. Goland, A. Guízar, N. Kartha, C. K. Liu, R. Khalaf, D. König, M. Marin, V. Mehta, S. Thatte, D. van der Rijn, P. Yendluri, and A. Yiu, "Web services Business Process Execution Language Version 2.0," OASIS Standard, 2007. Available: http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html.

[5] L. Clement, A. Hately, C. von Riegen, T. Rogers (eds.), „UDDI Version 3.0.2 – UDDI Spec Technical Commitee Draft," OASIS Standard, 2004. Available: http://uddi.org/pubs/uddi-v3.0.2-20041019.htm.

[6] J. Farrell and H. Lausen, "Semantic Annotations for WSDL and XML Schema," WC3 Recommendation, 2007. Available: http://www.w3.org/TR/2007/REC-sawsdl-20070828/.

[7] J. Nitzsche, T. van Lessen, D. Karastoyanova, and F. Leymann, "BPEL for Semantic Web services," in *Proceedings of the 3rd International Workshop on Agents and Web services in Distributed Environments (AWeSome'07)*, November 2007.

[8] D. Martin, M. Burstein, D. McDermott, S. McIlraith, M. Paolucci, K. Sycara, D. L. McGuinness, E. Sirin, and N. Srinivasan, "Bringing Semantics to Web services with OWL-S," *World Wide Web*, vol. 10, iss. 3, September 2007, pp. 243-277.

[9] D. L. McGuinness, "Ontologies come of age," in D. Fensel, J. Hendler, H. Lieberman, et al. (eds): *Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential*. MIT Press, USA, 2003.

[10] K. Sycara, M. Paolucci, A. Ankolekar, and N. Srinivasan, "Automated discovery, interaction and composition of Semantic Web services," *Journal of Web Semantics*, vol. 1, no. 1, 2003, pp. 27-46.

[11] M. Paolucci, T. Kawamura, T. R. Payne, and K. Sycara, "Semantic Matching of Web services Capabilites," in *Proceedings of the First International Semantic Web Conference (ISWC 2002)*, LNCS 2342, 2002, pp. 333-347.

[12] M. P. Papazoglou, "Service-Oriented Computing: Concepts, Characteristics and Directions," in *Proceedings of the 4th International Conference on Web Information Systems Engineering (WISE 2003)*, IEEE Computer Society, Washington, DC, 2003, pp. 3-12.

[13] M. Uslar, "The Common Information model for utilities: An introduction and Outlook on Future Applications," in *Proceedings of the XML-Tage 2006*, Berlin, Germany, 2006, pp. 135-148.

[14] M. Hepp, "Possible Ontologies – How Reality Constrains the Development of Relevant Ontologies," *IEEE Internet Computing*, vol. 11, no. 1, January/February 2007, pp. 90-96.

[15] L. Li and I. Horrocks, "A Software Framework For Matchmaking Based on Semantic Web Technology," in *Proceedings of the 12th International Conference on World Wide Web (WWW2003), Budapest, Hungary, 2003, pp. 331-339*.

[16] B. Xu, P. Zhang, J.-Z. Li, and W.J. Yang, "A Semantic Matchmaker for Ranking Web services," *Journal of Computer Science and Technology*, vol. 21, no. 4, July 2006, pp. 574-581.
[17] M. Klusch, B. Fries, M. Khalid, and K. Sycara, "OWLS-MX: Hybrid OWL-S Service Matchmaking," in *Proceedings of the 1st Intl. AAAI Fall Symposium on Agents and the Semantic Web*, 2005.
[18] A. Heß and N. Kushmerick, "Learning to Attach Semantic Metadata to Web services," in *Proceedings of the 2nd International Semantic Web Conference (ISWC2003)*, 2003.
[19] A. Heß, E. Johnston, and N. Kushmerick, "Semi-Automatically Annotating Semantic Web services (Extended Abstract)," in *Proceedings of the Workshop Information Integration on the Web (Int. Conf. Very Large Data Bases)*, 2004.
[20] M. Bruno, G. Canfora, M. Di Penta, and R. Scognamiglio, "An Approach to support Web service Classification and Annotation," in *Proceedings of the International Conference on E-Technology, E-Commercae and E-Service (EEE 2005)*, Hong Kong, IEEE Computer Society, pp. 138-143.
[21] N. Oldham. C. Thomas, A. Sheth, and K. Verma, "METEOR-S Web service Annotation Framework with Machine Learning Classification," in *Proceedings of the 1st International Workshop on Semantic Web services and Web Process Composition (SWSWPC'04), Part of the 2nd International Conference on Web services (ICWS'04)*, San Diego, CA, USA, 2004.
[22] M.-Á. Corella and P. Castells, "Semi-Automatic Semantic-Based Web service Classification," in *Business Process Management Workshops*, LNCS 4103, 2006, pp. 459-470.
[23] R. Berbner, M. Spahn, N. Repp, O. Heckmann, and R. Steinmetz, "Heuristics for QoS-aware Web service Composition," in *Proceedings of the 2006 IEEE International Conference on Web services (ICWS 2006)*, Chicago, IL, USA, pp. 72-79.
[24] C. J. van Rijsbergen, "Information Retrieval," London, UK, Butterworth, 1979.
[25] M. Klusch, "OWLS-TC-v2.2," 2007, Available: http://projects.semwebcentral.org/projects/owls-tc
[26] F. Manola and E. Miller, "RDF Primer," W3C Recommendation, 2004, Available: http://www.w3.org/TR/rdf-primer/
[27] J. J. Carroll, I. Dickinson, C. Dollin, D. Reynolds, A. Seaborne, and K. Wilkinson, "Jena: Implementing the Semantic Web recommendations," in *Proceedings of the 13th international conference on the World Wide Web. Alternate track papers & posters (WWW Alt. '04), New York, NY, USA, pp. 74-83.*
[28] D. Thakker, T. Osman, and D. Al-Dabass, "Semantic-Driven Matchmaking and Composition of Web services Using Case-Based Reasoning," in *Proceedings of the 5th IEEE European Conference on Web services (ECOWS'07)*, Halle, Germany, 2007, pp. 67-76.
[29] U. Küster, H. Lausen, and B. König-Ries, "Evaluation of Semantic Service Discovery – A Survey and Directions for Future Research," in *Preliminary Proceedings of the 2nd Workshop on Emerging Web services Technology (WEWST)*, Halle, Germany, 2007, pp. 37-53.