Building Service Models and Architectures to include Semantics and Social Networking

Nikolaos Loutas[1, 2]

[1] University of Macedonia, Business Administration Department, 156 Egnatia str, Thessaloniki, 54006, Greece, nlout@uom.gr
[2] National University of Ireland, Galway, Digital Enterprise Research Institute, IDA Business Park, Lower Dangan, Galway, Ireland, nikos.loutas@deri.org

Abstract. Currently thousands of services are available on the Web but the more services become available the harder it becomes to find and use them. Most of those services lack basic metadata that would improve their accessibility and discovery. In this work, we present a two-step approach for discovering services in the Web 2.0 environment. First, a formalized representation of the user's need is created and then based on that the services that match this need are discovered. The approach tries to make the most out of the available state of the art Semantic Web technologies. We use SA-REST as the annotation mechanisms and try to harvest the knowledge that can be extracted from social tagging.

Keywords: service model, web2.0, semantic web, portal, SA-REST, discovery

1 Introduction

Since the 90's when Web Services (WS) were first introduced, research in this field has always been particularly active. The Service Oriented Architecture (SOA) paradigm [1, 2] is until today the prevalent way of building enterprise Information Systems (IS). SOA has become throughout the years both a very attractive research domain and a profitable business. Therefore, many major IT vendors such as IBM, Microsoft and SAP can be found among the key players.

The main idea behind SOA is the ability to use, reuse and share services. As such, SOA bares in principle some resemblance to peer to peer and to distributive systems. Therefore, SOA provides IT architects and developers a unified view over a physically scattered set of services.

There has been a lot of debate so far with respect to the advantages and disadvantages of SOA. In addition to that many question the added value of SOA itself. They support that technological, organizational and financial barriers can hinder the uptake of SOA. Nevertheless one cannot refuse, that a successful SOA implementation can lead to organizational flexibility, greater business agility and resources (code, services and infrastructure) reuse.

IBM proposed first a conceptual architecture for SOA [3]. There, three basic entities are identified, namely the service provider, the service requestor and the

service broker. This conceptual architecture can support the four key functionalities, which are discovering, composing, publishing and invoking a service. Nevertheless, problems in performance, accuracy, (re-)usability etc. make it difficult for the Web Service brokerage model to fulfill its potential.

When in the late 90's the first ideas about semantics were drafted [4], the research community started thinking how semantics could enhance the WS brokerage model. Therefore, various Semantic Web Service (SWS) models were proposed. In the related work section we present an overview of the most popular ones. All these efforts aimed at enabling the dynamic discovery, composition and invocation of WS, thus extending the functionality of the WS brokerage model.

The results so far have shown that SWS efforts have not fulfilled their objectives yet. Currently, their high complexity discourages both technical and business people from adopting such solutions.

The last couple of years Web 2.0 seems to emerge as a new computing paradigm. This new reality influences the way the research community approaches service modelling and SOA. For example, this has created pressure to the SWS community to come up with light-weight approaches, which may lack in expressivity but win in simplicity (i.e. SA-WSDL [5], SA-REST [6], WSMO-Lite [7]), compared to the traditional SWS frameworks (i.e. WSMO [8], OWL-S [9], WSDL-S [10]), whose high expressivity leverages their complexity.

Interestingly within Web 2.0, new types of services appear which can be loosely mapped to the SOA principles. For example, mashups are created in a decentralised manner and REST [11] is used instead of SOAP [12].

In our research, we examine the service both from the conceptual and the technical perspective. From a conceptual perspective, following the OPEN Group definition [2], a service:

- is a logical representation of a repeatable business activity that has a specified outcome (e.g., check customer credit; provide weather data, consolidate drilling reports)
- is self-contained
- may be composed of other services
- is a "black box" to consumers of the service

From the technical perspective, we extend the definition of the WS provided by OASIS [13] and define a (web) service as a software component that may or may not be described via WSDL and is capable of being accessed via standard network protocols such as but not limited to SOAP or REST over HTTP.

The rest of this paper is organized as follows: section 2 presents our motivation while section 3 the related work. In section 4 we introduce our approach. Section 5 presents our current prototype implementation and section 6 presents our future work.

2. Problem statement and Motivation

Although many argue that SOA and Web 2.0 are different and try to pose a dilemma there, with regards to which paradigm should be followed, the reality is rather different. In fact, the discussion shouldn't be about "Web 2.0 *or* SOA" but about

"Web 2.0 *and* SOA" [14]. Their main difference seems to be that SOA is usually restricted by the closed intra-enterprise IS environment, while Web 2.0 is more about openness, open platforms etc.

Nevertheless, at the core of both one can find the ideas of autonomous, distributed services, remixability and reusability. In both, ownership and control issues are discussed and frankly we could say that Web 2.0 is probably the first actual, massive instance of a light-weight SOA or from another perspective Web 2.0 is about the entire Web being a reusable, shareable, public SOA [16].

Therefore, and due to their inherent similarities, the research community is once again trying to find answers with regards to how the four basic SOA functionalities, namely discovery, composition, publishing and invocation, should be implemented in a Web 2.0 environment. For example, thousands of mashups are available at the moment (i.e. [17]) and all indications show that this number will soon scale to millions. All these mashups are capable to provide access to huge amount of distributed content and/or services but how one can find the mashup (s)he really needs, when (s)he needs it. For instance, how can a driver that is running out of gas find if there is a mashup that displays on a map the open gas stations in the area where (s)he is driving?

Considering that sometimes the user does not have a clear view of his need can complicate things more. For example, a customer may know that (s)he wants to buy a laptop or a citizen may know that (s)he wants to drive a car, but both of them do not know which specific service has to be invoked in order to get the desired results. Thus, it is apparent that the user's need has to be somehow formalized before proceeding to the service discovery step.

In the Web 2.0 environment, the problem above becomes even more difficult, compared to SOA, since these new (web) services lack a standardized description from their creators. In SOA the service provider publishes to a broker (in registries like UDDI) a description of the available services in a standardized format to allow clients to look up and match-make these descriptions with what they actually looking for. Though, Web 2.0 neither proposes standardized ways to describe the mashups nor repositories to store these descriptions. The services are generated in a completely decentralized and uncontrolled way and the overall architecture lacks the SOA broker to mediate between service providers and service clients. Thus, a gap can be identified at this point - that is how can the service clients discover the services published by the service providers in the Web 2.0 environment. The fact that in Web 2.0 service providers and service clients are not distinctly separated and the telling the one from the other is rather blurry makes things even more complicated.

Taking into consideration the volatile, dynamic environment of Web 2.0, we propose in this work a solution to bridge the gap identified above. In particular, our main research objective is to propose a conceptual service model that will extend the existing ones by including social semantics and then to propose a framework for facilitating the discovery of Web 2.0 services that will be described with this service model. A detailed description of our approach is presented in section 4.

3. Related work

In this section we present some work about SWS modeling and discovery.

Klusch et. al [18] present OWLS-MX, which is a hybrid SWS matchmaker that complements logic based reasoning with approximate matching based on syntactic Information Retrieval based on similarity computations, and implement their approach for SWS specified in OWL-S. OWLS-MX is based on previous work of the authors presented in [19].

The OWL-S/UDDI matchmaker [20] provides semantic capability matching by expanding UDDI's proliferation into the WS infrastructure and combining it with OWL-S's explicit semantic description of the WS.

Bernstein and Klein [21] model the service using process models, and these process models are placed in a process ontology. Queries are then defined to find all the services whose process models include a given set of entities and relationships.

In [22] SPARQL is used as an expression language for OWL-S. The SPARQL CONSTRUCT form is used as a compact definition of a process result's pre- and post conditions and effects. When the CONSTRUCT is evaluated over an appropriate data set it produces an RDF graph that describes the state of the world after the execution of the process. Software agents can then use such RDF graphs to discover processes which fulfill their goals.

In [23] the use of a WS Peer-to-peer Discovery Service infrastructure is proposed, which provides a decentralized and interoperable discovery service with semantic-level matching capability.

Keller et. al. [24] propose a model for the automatic location of services that considers the static and dynamic aspects of service descriptions and identifies the matching techniques that can be used for their matching.

In [25] a goal-based WS discovery approach with semantic matchmaking is presented. The authors first make the distinction between goal templates as generic objective descriptions and goal instances that denote concrete requests as an instantiation of a goal template and then formally describe requested and provided functionalities on the level of state transitions that denote executions of WS, respectively solutions for goals.

Finally METEOR-S [26] follows an ontology-based approach to organize registries into domains, enabling domain based classification of WSDL-S services, thus improving WSDL-S service publication and discovery. In Lumina [27] the same team tries to enhance their SWS discovery mechanism so that it can also support SAWSDL services. This work extends the TModels of UDDI.

Apart from studying the efforts made so far in the SWS discovery field, we will also look into the work currently done in the social networking domain. We will try to see how we could use the knowledge, which derives from the social annotation of services done by normal web users without a pre-defined ontology, for enhancing service discovery.

In this field, Wu et. Al [28] show how semantics can be derived from social annotations and use these semantics to discover and search shared web bookmarks.

Bao et. Al [29] try to improve web-search using social semantics. To do so, they propose two algorithms: SocialSimRank that calculates the similarity between social

annotations and web-queries; and SocialPageRank that captures the popularity of web pages. A similar interesting research in the same field is carried out by [30].

Our work differs extends and brings together the ones presented earlier. As it will be described later in detail, we propose a two-step semantically enabled service discovery process. First we try to create a formal description of the user's need and then we try to find the services that address this need. From a technological perspective, our work is rather positioned on the intersection of Semantic Web and Web 2.0, which we define as Social Semantic Web, and focuses on SA-REST services. Additionally, our approach has a strong social perspective. Therefore, we investigate how social tagging and the information that can be extracted from the tagclouds and the folksonomies [31, 32] can be used in service discovery.

4. The Pillars of our Approach

This work aims at proposing a conceptual architecture, which will provide a set of models and tools, to overcome the obstacles presented in section 2. In particular, we investigate solutions for the description, categorization, indexing and discovery of services in the Web 2.0 environment. Our basic contributions are presented in the rest of this section.

4.1 Social Annotation of Services

In order to fulfill our objectives, we have to go beyond SOA and extend the typical SOA view on services. As it was presented in sections 1 and 2, all SOA approaches, semantic or not, use implicitly a conceptual service model, which describes the service. Until now this description was provided by the service provider. All these models lack any social characteristic; that is any attribute that is relevant from the users' point of view and is provided by them and not by the service providers. For example, the information regarding the services that a user usually invokes after the execution of a service is something that is not captured in any SOA service model. Similarly, the information that describes the services in which users with similar profiles are usually interested is not exploited using existing SOA models.

Our approach takes into account the social annotation of services so as to cover the missing descriptions from the service providers' side. Thus, a hybrid approach is proposed, where services are annotated both by the service providers and the service clients. Social annotation of services can in this way substitute some of the functionalities of the SOA broker. Social tagging allows users to:

- classify and cluster services according to their functionality/output/ behavior/ use case.
- express their satisfaction with regards to QoS aspects
- personalize their search
- discover services
- get recommendations about related services
- form communities of interest and/or practice

To capture these social service characteristics an extension of the existing SOA-based conceptual model of a service is needed. This new extended model should accommodate all the extra characteristics and properties that a service model may obtain if the social aspect of a service is taken into account.

In our approach we plan to use SA-REST as the annotation mechanism along with a conceptual service model for a specific domain. In the case that we are currently implementing, we are using an extended version of the GEA [36] PA Service Model [37], in order to semantically annotate e-Government services.

4.2 Our World of Services

Since the term services can have many different, often ambiguous or contradicting definitions, we have to scope our research and explicitly define the kind of services that we are interested in.

Our research focuses on services that are semantically annotated using SA-REST. SAREST [6, 38] provides a simple annotation mechanism based on RDFa [39]. SA-REST allows using any service model expressed in RDF-S in order to annotate the XHTML descriptions of the services. These services can either be:

- RESTful services, namely services that follow the principles of the REST architecture [11]; or
- Web Services that have a WSDL file and their semantic description is provided via an XHTML page.

SA-REST's flexibility allows us to use service models of different domains, i.e. e-commerce, e-government, tourism, and combining them with the SA-REST annotation mechanism. As a result, we can apply our approach in different domains and validate its generality, applicability and extensibility.

4.3 The Service Tree Ontologies

After studying different types of services from different domains, we concluded saying that most service types can be boiled down to different service versions. For example, the buy a ticket service has various service versions, i.e. buy a concert ticket, buy a flight ticket etc.

The different service versions of a service type can be organized in a tree like structure, which can be modeled using an ontology [41]. In the root node there is the generic concept of a service and in the next nodes exist different versions of the generic service depending on specific characteristics either of the users who are entitled to each one or of the product.

In our approach we use Service Tree Ontologies (STO) in order to formalize the user's need and find the available services that can address it. At a first attempt to structure the STO, we define that each STO consists of a set of *Nodes*, which can be either *InternalNodes* or *LeafNodes*. LeafNodes represent different service versions of the service type modeled in the STO, while LeafNodes represent the formalized description of the user's need.

Each LeafNode contains a set of one or more questions. The questions are used to

guide the user through the different service versions until the appropriate one is found and thus his/her need has been formalized. The user's answers to each question are the ones that define the next step in the STO. In order to do this, the appropriate reasoning mechanisms have to be employed.

4.4 A Two-Step Semantically Enabled Service Discovery Process

Our approach follows a *two-step semantically enabled* service discovery process. First the need of the user is formalized and then the services that address this need are discovered. Formalizing the users' needs means expressing in some kind of logical formalism the abstract need that users have in their minds. Then this logical expression can be evaluated over a set of services and matched against their attributes, in order to find the appropriate ones. A detailed, concrete and sound expression of the formalized need, would minimize and increase the relevance of the result set.

The first step is implemented by means of a semantically enabled portal where users can express their needs, while the underlying discovery mechanism realizes the functionality of the second step. STO described earlier play a key role in this process.

The conceptual architecture of our approach is presented in Fig. 1. The various components of the architecture are presented below in detail.

The *User Interface* (UI) provides the means for the users to:

(i) express their needs and discover services that address these needs;
(ii) get information about services, i.e. service provider, cost, required input, expected output etc.;
(iii) invoke services;
(iv) find related services and get recommendations about services that match their profile and/or their behavioral patterns;
(v) express their satisfaction with respect to QoS aspects.

The *Service Tree Ontology Finder* (STOF) helps the user find the STO that models the service type that addresses his/her needs. The user types some keywords that describe his/her need and the STOF finds the matching STO(s). The STOF should also support some kind of semantic browsing through the available STOs based on the tags that describe each one.

The *Querying Component* (QC) carries out the structured discussion between the portal and the user. This discussion aims at formalizing the user's need. In order to do so, the QC either poses the questions stored in STO to the user and gathers his/her answers or directly takes the necessary information from the user's profile (if one exists). When the discussion reaches an end, the information is forwarded to the DC.

The *Discovery Component* (DC) forms a SPARQL query based on the information forwarded from the QC and executes the query over the RDF Triples Repository. The query should return all the services that match with the user's request. Then the URLs of these services along with some brief description (i.e. required input, output, cost, service provider etc) are forwarded to the user. Then, the user can invoke any of the returned services.

The *Personalization Component* (PC) is responsible for personalizing the system according to the user's profile and for implementing the system's recommendation functionalities. PC keeps track of the user's behavior, i.e. which service is invoked

after another service etc., so that it can create patterns and cluster related services. Moreover, through the PC, users can update their profiles and express their preferences. Web 2.0 personalization algorithms and mechanisms will be considered, i.e. [33].

The *Tagging Component* (TC) implements the tagging functionalities of the system. It allows users to tag services and it also organizes tags in clusters, thus trying to structure them and find potential relations between them, that is moving from unstructured tagclouds to folksonomies [34, 35]. The TC could also support tag-based browsing through the available services.

The *Crawler* crawls the Web and tries to find SA-REST services. Once a SA-REST service is found, all the sarest attributes (i.e. sarest:input, sarest:output etc.) are extracted as RDF triples and are stored in the RDF Triples Repository.

The three repositories (Ontologies Repository, User Profile Repository and RDF Triples Repository) are used to store the types of data that the system handles.

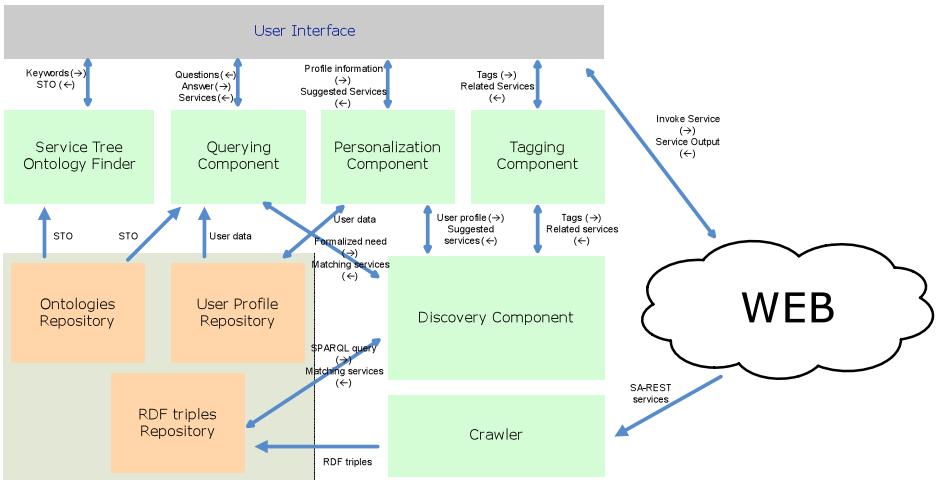

Fig. 1. Conceptual Architecture

5 The e-Government case

In the context of the SemanticGov project [40], we have implemented a prototype of this approach [41]. In this implementation a portal supports the users to identify the formally described goal that expresses their needs and invoke the discovered services that match it. The discovery phase is supported by an online structured discussion. Moreover, the portal helps the users collect information with regards to the service preconditions, input, and result and thus checks their eligibility for the service. From the implementation point of view, this discussion is supported by a set of STOs encoded in WSML and the IRIS reasoner. In the current implementation, the SWS are described by means of WSMO.

6 Future Work

In this paper we presented our two-step approach for discovering services in the Web 2.0 environment. It is expressed by means of a conceptual architecture, which will provide a set of models and tools, for annotating, tagging and discovering services. First a formalized representation of the user's need is created and then based on that the services that match this need are discovered. The approach tries to make the most out of the available state-of-the-art Semantic Web technologies. We use SA-REST as the annotation mechanisms and try to harvest the knowledge that can be extracted from social tagging.

We plan to extend our existing work described earlier and try to combine the GEA PA service model [37] with the RDFa annotation mechanism in order to semantically annotate e-Government services. The XHTML descriptions of the e-Government services will be annotated using mechanism proposed by SA-REST and the concepts of the model, i.e. sarest:service_provider, sarest:precondition, sarest:output etc. Moreover, we plan to extend the PA Service Model, so that it can accommodate user-defined tags. Our crawler will search the Web for SA-REST e-Government services and each time on is found the sarest triples (i.e. <serviceURI sarest:service_provider sp_uri/sp_name>) will be stored in our repository, thus enabling users to perform searches using our SPARQL-based querying mechanism. Moreover, we plan to extend our current prototype, by developing the tagging and recommendation mechanisms and by implementing an SA-REST services editor. Moreover, we will examine the applicability of our work in other domains, since our approach aims to be domain independent.

Finally, we plan to evaluate and validate our work in terms of usability, scalability and performance. Our aim is to involve also users in the evaluation of our prototype. For validating our work the relevant literature will be reviewed so that the most appropriate methods will be identified.

Acknowledgments

This work is carried out under the guidance of Dr. Vassilios Peristeras[1] and Professor Konstantinos Tarabanis[2]. It is supported in part by the SemanticGov (www.semantic-gov.org) and the Ecospace (www.ip-ecospace.org) FP6 projects and by the Science Foundation Ireland project Lion under Grant No (SFI /02/CE1/I131).

References

1. OASIS Reference Model for Service Oriented Architecture, http://www.oasis-

[1] National University of Ireland, Galway, Digital Enterprise Research Institute, IDA Business Park, Lower Dangan, Galway, Ireland, vassilios.peristeras@deri.org

[2] University of Macedonia, Business Administration Department, 156 Egnatia str, Thessaloniki, 54006, Greece, kat@uom.gr

open.org/committees/tc_home.php?wg_abbrev=soa-rm
2. OPEN Group, http://www.opengroup.org/projects/soa/doc.tpl?gdid=10632
3. Gisolfi, D., IBM: *An introduction to dynamic e-business*, http://www.ibm.com/developerworks/webservices/library/ws-arc1/ (2001)
4. Berners-Lee, T., Hendler, J., Lassila, O.L.: The semantic web. Scientific American, 284(5):pp. 34–43, 2001.
5. Kopecky, J., Vitvar, T., Bournez, C., Farrell, J.: SAWSDL: Semantic Annotations for WSDL and XML Schema, IEEE Internet Computing, vol. 11 (6) (2007)
6. Sheth, A., Gomadam, K., Lathem, J.: SA-REST: Semantically Interoperable and Easier-to-Use Services and Mashups, IEEE Internet Computing, vol. 11 (6): pp. 91-94 (2007)
7. Kopecky, J., Vitvar, Zaremba, M., Fensel, D.: WSMO-Lite: Lightweight Semantic Descriptions for Services on the Web Full, Proceedings of 5th ECOWS, pp. 77-86 (2007)
8. Roman, D., Lausen, H., Keller, U. (eds.), Web Service Modeling Ontology (WSMO), Technical report, WSMO Final Draft, http://www.wsmo.org/TR/d2/v1.2/ (2005)
9. Martin, D. et. al.: OWL-S: Semantic Markup for Web Services, W3C Member Submission, http://www.w3.org/Submission/OWL-S/ (2004)
10. Akkiraju, R. et. al.: Web Service Semantics - WSDL-S, W3C Member Submission, http://www.w3.org/Submission/WSDL-S/
11. Fielding, R. T.: Architectural Styles and the Design of Network-based Software Architectures, PhD Thesis in Information and Computer Science, University of California, Irvine (2000)
12. Box, D. et. al: Simple Object Access Protocol (SOAP) 1.1, http://www.w3.org/TR/2000/NOTE-SOAP-20000508/ (2000)
13. OASIS Glossary, http://www.oasis-open.org/committees/wsia/glossary/wsia-draft-glossary-03.htm
14. Schroth, C., Janner, T.: Web 2.0 and SOA: Converging Concepts Enabling the Internet of Services, IEEE IT Professional, vol. 9 (3), pp. 36-41 (2007)
15. Omar, W., Abbas, A., Bendiab, T.: SOAW2 for Managing the Web 2.0 Framework, IT Professional, vol. 9 (3), pp. 30-35 (2007)
16. SOA World Magazine, http://webservices.sys-con.com/read/164532_2.htm (2007)
17. www.programmableweb.com, accessed on April 10 (2008)
18. Klusch, M., Fries, B., Sycara, K.: Automated semantic web service discovery with owls-mx. Proceedings of the AAMAS, pp. 915-922 (2006)
19. Sycara, K., Klusch, M., Widoff, S., Lu, J., Larks, K.: Dynamic matchmaking among heterogeneous software agents in cyberspace. Autonomous Agents and Multi-Agent Systems, 5(2), Kluwer (2002)
20. Sycara, K., Paolucci, M., Anolekar, A., Srinivasan, N.: Automated discovery, interaction and composition of semantic web services. Web Semantics, 1(1), Elsevier (2003).
21. Bernstein, A., Klein, M.: Towards high-precision service retrieval. IEEE Internet Computing, vol. 8(1): 30-36 (2004)
22. Sbodio, M. L., Moulin. C.: SPARQL as an expression language for OWL-S. Proceedings of OWL-S: Experiences and Directions, 4th ESWC (2007)
23. Banaei-Kashani, F., Chen, C.C, Shahabi, C.: Wspds: Web services peer-to-peer discovery service. In Proceedings of ISWS (2004)
24. Keller, U., Lara, R., Lausen, H., Polleres, A., Fensel, D.: Automatic Location of Services. Proceedings of the 2nd ESWC 3532 (2005)
25. Stollberg, M., Keller, U., Lausen, H., Heymans. S.: Two phase web service discovery based on rich functional descriptions. Proceedings of 4th ESWC (2007)
26. Verma, K., Sivashanmugam, K., Sheth, A., Patil, A., Oundhakar, S., Miller. J.: Meteor-s wsdi: A scalable infrastructure of registries for semantic publication and discovery of web services. Information Technology and Management (2004)
27. Lumina - Semantic Web Service Discovery, http://lsdis.cs.uga.edu/projects/meteor-

s/downloads/Lumina/
28. Wu, X., Zhang, L., Yu, Y.: Exploring social annotations for the semantic web. Proceedings of 15th WWW: pp. 417 - 426 (2006)
29. Bao, S., Xue, G., Wu, X., Fei, B., Su, Z.: Optimizing web search using social annotations. Proceedings of 16th WWW: pp. 501 - 510 (2007)
30. Yanbe, Y., Jatowt, A., Nakamura, S., Tanaka, K.: Can social bookmarking enhance search in the web? Proceedings of the 7th ACM/IEEE JCDL pp. 107 - 116 (2007)
31. Specia, L., Motta, E.: Integrating Folksonomies with the Semantic Web. In Semantic Web Research and Applications, vol. 4519: pp. 624 – 639, Springer (2007)
32. Mika, P.: Ontologies are us: A unified model of social networks and semantics Source**,** Web Semantics: Science. Services and Agents on the World Wide Web, vol. 5 (1), pp. 5-15, Elsevier (2007)
33. Eirinaki, M., Lampos, C., Paulakis, S., Vazirgiannis, M.: Web personalization integrating content semantics and navigational patterns. Proceedings of the 6th annual ACM international workshop on WIDM: pp 72 - 79 (2004)
34. Angeletou, S., Sabou, M., Specia, L., Motta, E.: Bridging the Gap Between Folksonomies and the Semantic Web: An Experience Report. Proceedings of 4th ESWC (2007)
35. Begelman, G., Keller, P., Smadia, F.: Automated Tag Clustering: Improving search and exploration in the tag space. Proceedings of 15th WWW: pp. 417 - 426 (2006)
36. Peristeras, V.: The Governance Enterprise Architecture - GEA - for Reengineering Public Administration, PhD Thesis in Business Administration, University of Macedonia, Thessaloniki, Greece (2006)
37. Goudos, S., Loutas, N., Peristeras, V., Tarabanis, K.: Public Administration Domain Ontology for a Semantic Web Services EGovernment Framework, Proceedings of 2007 IEEE SCC, 270-277 (2007)
38. Lathem, J., Gomadam, K., Sheth, A.: SA-REST and (S)mashups: Adding Semantics to RESTful Services, Proceedings of first ICSC, pp. 469-476 (2007)
39. RDFa Primer, http://www.w3.org/TR/xhtml-rdfa-primer/
40. Loutas N., Peristeras V., Tarabanis K.: Providing Public Services to Citizens at the National and Pan-European level using Semantic Web Technologies: The SemanticGov Project, Proceedings of the 6th EEeGov Days, (2008)
41. Loutas, N., Peristeras, V., Goudos, S., Tarabanis, K.: Facilitating the Semantic Discovery of eGovernment Services: The SemanticGov Portal, Proceedings of 3rd VORTE at the 11th EDOC (2007)