Local Context Anchoring in the Modern Web Service Retrieval Model

Konstanty Haniewicz
Poznan University of Economics
Poznan, Poland
konstanty.haniewicz@ue.poznan.pl

***Abstract*—Local Context Anchoring is one of mechanisms supporting the modern Web service retrieval model. Its aim is to provide users with support in the retrieval of data on Web service operations that are beyond boundaries of their Suborganisational Units. Due to the fact that user operates outside his regular environment a mechanism is necessary to make up for the lack of certainty on the structure and content of queried resources. In order to present the mechanism in a satisfactory manner a description of the key concepts that are motivation for modern Web service retrieval is given. Their scope and focus is different from the one presented in majority of publications concerning the Web service domain. The model itself is also introduced along with details on its structure and features important to targeted users.**

***Keywords*-Local Context Anchoring; Web service description; functionality description; knowledge representation; Information Retrieval**

I. INTRODUCTION

In order to satisfy a constant need for up to date data on available resources a novel model for Web service description is proposed. It takes into account a number of initiatives addressing Web service description and retrieval based both on solutions relying on semantic enhancements ([1]–[5]) and those that mainly employ standard Information Retrieval based techniques ([6]–[10]). The critical analysis coupled with feedback gathered from the Information Technology professionals led to the definition of five key aspects of the desired solution which later became a ground work for the definition of the proposed model.

The description of the key aspects is given below.

- **Effectiveness** - is perceived as the ability to cater for a need expressed by a user in a format provided by the available solution. When IR-based solution is taken into account, standard measures of precision and recall should provide the answer to how effective a given solution is. On the other hand, in the case of semantic based solutions, precision and recall are deceiving because when an ontology is being queried it should always provide a complete set of answers. Thus, a single measure cannot be applied and a description of effectiveness in terms of solution specification must be available.
- **Cost** - is an effort that should be spent on a proper description of a desired artifact with an envisioned technology, time spent on learning necessary description techniques and a prognosis on a timewise performance of analyzed solution. This set of properties penalizes solutions that require a lot of effort in preparation for production and use and, in addition, a long time of query matching. Such bias leads to the promotion of solutions with a low level of complication as perceived by the end users.
- **Scalability** - describes how soon and to what degree a performance shall drop along with an increase in a number of handled Web service operations. This measure shall promote solutions that can handle thousands of Web services with tens of thousands operations.
- **Scope** - involves any important additions to a baseline of a Web service description by Web Service Definition Language (WSDL) documents in terms of an identification of a vital, not previously addressed areas of importance to a user, any extension of description besides functional description of Web service operations, such as business key performance indicators [11] and a perspective on Web services different than that of a developer.
- **Purpose statement** - determines whether an initiative allows stating the purpose of a Web service along with its operations. It is understood as a possibility to express a goal of an artifact in question so that it is clear what it does for all the parties involved.

In order to address all of the enlisted aspects, one could not longer work with the traditionally developed models as proved incapable of delivering satisfying results in the majority of the enlisted aspects. A broader discussion is given in the Related Works section.

Thus, the presented model is not a monolithic in the sense of common ontology that had to be designed, produced and deployed by a relatively small and extremely well coordinated team in order to be successful. What is more, it does away with the idea of indexing and extending WSDL documents in order to treat them as a set of regular textual data with specific metadata.

The postulated model aims for a federation of interlocking term networks ordering terms used in the description of Web services and their operations. These terms are gathered on a small scale, in order to make it possible for a relatively small Suborganization Unit (hereafter refereed to as SU) to exhaustively describe their Web service assets with the

 ISBN: 978-1-61208-200-4

vocabulary of their choice that is perfectly understood in SU's context.

The document is organised as follows: first, an overview of the model is given with a special highlight of the mechanism of phrased-based description and its usability for different groups of users; next, a more detailed overview of mechanisms is discussed; following that, a presentation of Local Context Anchoring is given along with validation subsection presenting the experiments' results; summary section is preceded by the Related Works section.

II. Modern Web service description model

The first premise of the presented model is to make a Web service an asset available to a wider range of users in an organization. Thus, its usability is not only based on the mastery of details desired by technology-oriented personnel but also on various features deemed important by business and executive users. The stakeholder environment is depicted in Figure 1. While the three given groups share some needs

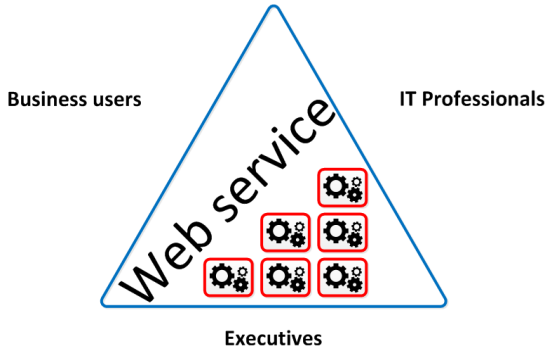

Figure 1. Different interest groups which can benefit from a different approach to a Web service description

one has to explicitly emphasize that business users are more interested in Non Functional Parameters (NFPs) and an actual usage of any given Web service operation in projects of interest to them. They should also be more interested in the purpose of an entity that is crucial for their business processes. This is contrasted with the emphasis given by Information Technology professionals on technical details, such as Inputs, Outputs, Preconditions and Effects, Quality of Services and actual service-providing systems. Executive users might be interested in general costs of invocation, compliance with Service Level Agreements (SLAs) and overall involvement of any given Web service operation across an organisation.

The model itself is build around the notion of phrase-based Web service operation description aided by the above-mentioned NFPs to cater for business-oriented queries. The general phrase-based description is structured as follows:

Web service operation: $\langle(\alpha, \beta, \gamma), \mathbf{nfp}\rangle$

- α – action
- β – object
- γ – action-object supplement
- **nfp** – vector of NFP and its values

The decision to mold it in this particular way is derived from the prevalent lack of purpose statement in the majority of the surveyed initiatives. The purpose statement should be understood as a method of answering the question of what a given Web service operation does in a given context. One can argue that a Web service operation name shall convey this information, or even that a definition of Input and Output values in an ontology used throughout the organisation is sufficient.

Unfortunately, one cannot agree with this due to the fact that names ar often poorly defined and that the connection between the purpose of a Web service operation and its input and output parameters is somewhat remote. Even if when the Web service purpose is stated as a goal encrypted as a series of references to an ontology one can easily check that such definition is unfathomable to an average business user [12].

Instead, the phrase-based description builds on the federated effort of SUs that should possess sufficient knowledge to catalogue their Web service assets with terms and phrases deemed most suitable in their context. The process of cataloguing is semi-automated as the model is able to foresee tools that should accept a number of documents which are to be treated as a reference material to obtain an initial list of important terms that might be included at a later phase.

When accomplished, Local Controlled Vocabulary (LCV) serves as a master list of terms and phrases in a given SU. Web service operations are described with terms originating from LCV. It is very important as there is no guarantee that, when any given SU is preparing its LCV, a number of terms or compound terms used in descriptions will not be repeated. In addition, namespaces allow for customizing the results of Web service retrieval and mapping of terms across an organization.

A syntax of the phrase-query language is given below in the form of Antlr [13] grammar, where actual terms used to denominate phrases and namespaces were substituted by exemplary ones in order to make the syntax brief:

```
grammar phrase_query_grammar;
phrase_query: (a b g) nfp*;
a : 'a:' namespace compound_term;
b : 'b:' namespace compound_term;
g : ('g:' namespace compound_term)+;
namespace : '#' ('aaa'|'bbb'|'ccc');
compound_term :  term+;
term  : 'aaa' | 'bbb' | 'ccc' | 'ddd';
nfp : nfp_el ':' val;
nfp_el : 'nfp1' | 'nfp2' | 'nfp3';
val : sign number;
sign  : '+' | '-' |;
number  : digits '.' digits;
```

 ISBN: 978-1-61208-200-4

```
digits  : ('0'..'9')+;
```

III. Local Context Anchoring

Thanks to the design decisions reported in the previous section, a user faces three possible ways of interaction with the repository built upon the postulated model:

- query based on phrases,
- Web service operation name lookup,
- free query.

The most complicated case is the free query as system has no control over what a user inputs thereinto. What is more, due to the federated nature of the model this is the key scenario when various LCV become integrated.

When querying for a term, a list of matching resources, ordered by one of the phrases, is given. When there is no match, a set of hints is presented based on term references with the employed knowledge representation structure.

The mechanism for Local Context Anchoring (LCA) depends on open data repositories or assorted organisation corpora. In its essence, LCA can be viewed as a specialised Query Expansion algorithm [14] that takes into account organisation specific data.

There is a wide choice of data resources that can be adopted by the individual SUs and organisations as a whole [15].

An organization can possess vast resources on some topics that are key to its objectives and cannot be matched by those available in the open repositories.

The key concept of Local Context Anchoring is probing the available repositories (of open access type or propriety) for terms that coincide with those unmatched with terms used in the descriptions of the Web service operations. These are not to be understood as traditional measures used by Information Retrieval based on statistics of direct neighbour co-occurrence. The unmatched term is queried across the available resources. When matched, its context is probed for terms present in a set of all defined terms across all possible namespaces in addition to a check of the actual descriptions of Web service operations. The context of a search term is understood as a frame that spans for n terms before and after the matched term. The actual number of terms is dependent upon experiments, yet the research performed demonstrates that the frame which matches the length of an average paragraph in English texts is a good choice (100 to 150 terms).

The terms that fall into the frame are normalized and stop words are removed. The mechanism yields best results when multiple matches are found and an the occurrence ranking of coinciding terms can be prepared at a later stage of this algorithm. The retrieval of coinciding terms makes sense only in a situation, where resources responsible for providing the context are rich enough.

A. Auxiliary elements of LCA

The above described mechanism is made more adaptable by the inclusion of local resources that are invulnerable to network latency problems or other access issues.

Best example of the desired kind is Wordnet [16]. Moreover, the ability to include terms more specialised, less specialised and a variety of synonyms enriches the set of possible hints. More, the terms resulting from coinciding term search obtain a higher score if they are matched by the terms obtained from resources such as Wordnet and the like (The Suggested Upper Merged Ontology - SUMO [17] which is integrated with Wordnet or ResearchCYC [18]).

At the time of preparation, a prototype uses the most important open repositories, Web search engine, Wordnet and SUMO version integrated with Wordnet along with a number of resources compiled in such a way that specific domains are better represented.

All of these auxiliary measures are introduced to make the model respond to a user's needs to the furthest possible extent. The model is built upon a presumption that failure to present an answer is the worst case scenario which should be avoided at all costs.

Thus, the mapping across functionalities described in various SUs is used. As discussed above, a SU has a perfect freedom of choice when it comes to a set of terms related to its needs and its particular business environment. Yet, many a time, a situation can occur in which some aspect of functionality was described with a term that had many used synonyms in other LCVs. LCA uses this as an opportunity to draw a set of mappings across various SUs in order to come up with yet another data source that could be used when a free query is to be answered.

As there is no guarantee that entities described with similar terms have similar functionality due to the above emphasized reasons, a decision on similarity has to be made by a query issuing user.

LCA and its auxiliary submechanisms can generate a lot of new possibilities, therefore some restrictions had to be introduced. Throughout the experiments with randomly generated descriptions it became apparent that the search for coinciding or similar terms should be restricted to the first two phrases (α and β). One has to remember that each possibility for α is checked with every possibility for β phrase, which results in quadratic complexity of the whole pass.

The decision was made to avoid exponential expansion of the problem space, especially that γ phrase had no limit on the number of the used terms (although a close inspection of the available Web services can provide grounds for a reasonable arbitrary limit). An attempt to match a term that is unknown a priori and arrives for processing as an outcome of the Local Context Anchoring mechanism resembles efforts of the automatic matching of Semantic Web services. Service composition was proved to be feasible, yet

 ISBN: 978-1-61208-200-4

the performance of this procedure drops significantly with the increase in the number of Web services [19]. Hence, the above constraints.

The whole mechanism can be outlined in the following steps:

- A query is issued by a user;
- The operation is handled by Local Context Anchoring:
  - each term is submitted as a query to open resources;
  - each term is submitted as a query to the available term networks;
  - each item is consulted with the available mappings for Local Controlled Vocabularies;
- The LCA computes preliminary lists of hits from queried resources;
- A List is ranked with the frequencies of hits in given resources;
- A List is modified to satisfy the predefined trust levels associated with resources;
- Matching of ranked terms with the available Web service operation descriptions is performed;
- A List is rectified by the reordering of Web services depending on a level of match (where compound terms are scrutinized);
- A final output is presented to a user.

As a feature of user interface, results can be displayed as a flat list or grouped by a variety of criteria, such as a project to which the matched Web service operations pertain, a namespace of home LCV or a Web service with which it was deployed.

B. Compound term decomposition

In order to balance flexibility and expressivity of the solution, the transformation of singular terms into compound ones was allowed. This is clearly visible in the previously presented syntax of a phrase query.

Compound terms were introduced to prevent the unwanted decrease of the solution's performance as terms built with other terms could be easily implemented. It is only an addition of extra layer of abstraction that allows for storing base terms that does not negatively affect the benefits of the Local Context Anchoring.

Hence, all the compound terms must be built with a tool that stores data on atomic terms used to produce a compound.

Having accomplished this, a reconstruction of terms is a simple procedure that looks a compound term up in a designated register. Users benefit from this feature as they can easily forge new description phrases with terms that suit the character of their SU best. On the other hand, the solution does not lose effectiveness in situations, where a direct match is not present in the repository. Thanks to the discussed features of LCA the Web service operations described even with the most complicated compound terms can still be matched.

C. Result caching

Query caching in Information Retrieval has a very important role as a feature that boosts effectiveness of any IR system [20].

The proposed model also includes mechanisms that allow for the results of queries to be cached. This is mainly dictated by the need of further efficiency gains in terms of execution time. The novelty of the caching mechanisms proposed here is based on the retention of cache data throughout the system's lifecycle. It can be achieved mainly due to a relatively small amount of data concerning Web services. This is to be contrasted with astronomic amounts of data concerning documents in the process of being indexed by common purpose Web search engines.

The essence of this mechanism lies in the fact that every description of a Web service operation is retained in the solution implementing the model and the terms used in the phrases are easily traceable to the previously issued queries. Therefore, once a query has been issued and a set of operations has been retrieved, it should be constantly updated so that the subsequent queries referring to the initial result set could bypass the initial mechanism. This short-circuits the whole process, cutting drastically the number of operations required in order to present the answer to a user.

In order to prevent memory exhaustion, a set of supporting mechanisms is implemented. Such auxiliary mechanisms allow for the coordination of the caching mechanism with the frequency of a particular query. When some previously defined threshold is met, a query result set is cached.

The introduced modification of caching mechanisms satisfies not only the effectiveness, but also a the cost and the scalability. Given that data in cache always cover the complete set of the matching Web service operations, one does not risk a lapse originating from the fact that some important operation was omitted.

The scalability aspect is reinforced by decreasing the size of the search space by the introduction of mapping between frequently issued queries and their constantly updated result sets. What is more, there is no penalty from using cache as there is no possibility of having it outdated.

D. Validation

In order to ensure that the postulated solutions yield desired results a number of experiments was performed. A quality that might be crucial to many, is the validation of effectiveness measured in terms of execution speed of the algorithm responsible for matching user-issued queries against the available descriptions. The experiments were conducted in a moderately fast test bed consisting of a workstation equipped in Pentium Core 2 Duo (Allendale - 2.4 Ghz) processor with 3 GB of RAM. The algorithm was

 ISBN: 978-1-61208-200-4

implemented in the Python programming language (tests run with PyPy implementation version 1.7).

The test scenario was centered on a repetitive retrieval of Web service operation descriptions matching the prepared query. The initial analysis of the solution proposed in the model indicates that the matching process has the worst complexity of $O(n * m)$, where n and m are the numbers of elements from the matched sets.

The Web service operation descriptions used were generated in an automated manner. All of the tests started with as few Web service operation descriptions as 100 and gradually increased up to one million and finally to 10 millions. The upper bound was dictated by the amount of the available memory in the test environment. Every matching operation was performed 100 times with different queries. The initial experiments with the first version of the matching algorithm proved that it was feasible to apply it to the general task. The summary thereof is given in Table I.

Table I
RESULTS FROM EFFECTIVENESS EXPERIMENT MEASURING EXECUTION TIME OF QUERY MATCHING ALGORITHM

Number of descriptions	Average time after ten runs (seconds)
100	0.000053912401
1000	0.000157743692
10000	0.001043230295
100000	0.007972598076
1000000	0.103747159243

Table II
RESULTS FROM EFFECTIVENESS EXPERIMENT MEASURING EXECUTION TIME OF IMPROVED QUERY MATCHING ALGORITHM AGAINST TEST DATA REFLECTING THE NEW STRUCTURE OF WEB SERVICE OPERATION DESCRIPTION

Number of descriptions	Average time after ten runs (seconds)
100	0.000038175809
1000	0.000092013316
10000	0.000257968902
100000	0.001834869384
1000000	0.021018028259
10000000	0.441106071472

The test programme was implemented as a single threaded application. The original run included the following test scenario data:

- 90 α phrase elements to chose from,
- 90 β phrase elements to chose from,
- 90 γ phrase elements to chose from,
- 190 nfp elements to chose from.

As an additional constraint a query had at most 1 alpha phrase element, 1 beta phrase element, from 3 to 7 gamma phrase elements and up to 5 nfp elements. As one can seen, the initial version performed well until a certain number of Web service operation descriptions was to be handled. Since results deviated from the initial assumptions, the code was scrutinized, optimized and extended in order to accommodate additional features.

What is more a hypothesis that Web service operation names should not convey so many phrases was formulated. In order to validate it a survey was conducted with the help of data gathered in the course of the research activities (a corpus of over 50000 WSDL documents). The WSDL documents selected as a means of hypothesis verification were characterized by the highest number of operations. Of course, service architects responsible for the analyzed Web services did not use the function denomination method presented in this work. Fortunately, a number of scrutinized Web services had an easy-to-follow scheme of names, where one could easily distinguish between functional objectives of the terms used for the description. The outcome of the verification can be summarised by the following structure (it is approximated from the eligible WSDLs).

- 50 α phrase elements to choose from,
- 90 β phrase elements to choose from,
- 110 γ phrase elements to choose from.

What is more, one had to arbitrarily establish a number of validated NFPs that should be taken into account. The rationale for this is the fact that every organization is interested in providing of the minimal common set of Key Performance Indicators that makes it possible to compare results globally. The minimal set of 5 enlisted NFPs was used, namely Cost, Availability, Reliability, Performance and Security [21].

It is important to notice, that it is difficult to clearly distinguish between β and γ phrase elements outside the model presented here, where it is tracked throughout the whole process, even if it happens that one term can be used in both phrases, while being a member of different namespaces. Therefore, the above given structure conveys some possible redundancy of terms used in the discussed sets of phrases – their intersection is not empty.

Table II summarizes the results. One can see that the effectiveness was greatly improved in comparison to the initial experiment. The presented implementation handles queries on the body of 10 million descriptions easily and it was measured that it could maintain its effectiveness for a larger corpus by parallelization of the whole procedure. Thanks to the low cost of merging individual sets (an operation of adding a set to another set has a complexity of $O(n + m)$, where n and m are the numbers of elements from summated sets) and cheap partition of corpora into smaller chunks the overhead on the parallelization is very low.

The obtained result data allow one to conclude that the proposed method is effective in terms of execution speed and it scales well up to 100 million descriptions (there were no additional experiments above that number). As mentioned

 ISBN: 978-1-61208-200-4

previously, the additional features were introduced in the improved version. While matching the query against the Web service operation description not only full matches are given, but also those that use the desired terms originating from other namespaces.

IV. Related work

The model and mechanisms presented here are an answer to various issues and challenges raised in the literature on the Web service description [22]–[25]. Historically, the two most important approaches were the Information Retrieval and semantic annotations. If one is to compare the two, it can observed that they are not mutually exclusive as to their global goals but they differ significantly when it comes to the means to present a user with the desired functionality.

In general, the IR-based methods treat WSDLs as text documents, where the desired terms either exist or not. The actual implementation can differ significantly from the common border line by the deployment of various techniques that try to avoid simple term presence verification by virtue of thesauri or other similar means. Nevertheless, the success of a user query is strictly connected to one's choice of query terms. The most important works representing this approach are [6]–[10] .

On the other hand, the semantic-based solutions focus on functionality description with advanced description languages that can perfectly describe user needs in terms of common ontology of concepts. This results in a perfect match of descriptions against user queries with a number of important side effects. This approach is best represented in [1]–[5]. Of importance is the fact that the most precise semantic mechanisms are prone to deficiencies of the cost nature [26].

Apart from the two most important approaches mentioned above, there are solutions that try to leverage their best traits. The means by which the hybrid solutions try to achieve their goals are very different for each case and cannot be summarised successfully for the whole population. The most influential initiatives include [25], [27], [28].

It was decided that each group of the solutions should be ranked against the five key aspects mentioned in the introductory section of this work. The result is available in Table III. The referenced works are only a fraction of those used for the comparison presented below. The total body used was composed of 44 positions (from over twice the number initially considered). All the aspect except Cost should be read as the higher percentage the better, whereas Cost should be understood conversely. The percentage is calculated on the basis of the individual evaluation of the works classified according to the above-mentioned approaches. There were 12 works classified as a pure IR-based approach, 22 classified as the semantic-based approach and 12 were classified as the hybrid approach.

Table III
Quantification of key aspect conformance according to domain literature

	IR based	**Semantic**	**Hybrid**
effectiveness	$\approx 50\%$	$> 75\%$	$\approx 50\%$
cost	$> 50\%$	$< 25\%$	$\approx 50\%$
scalability	$\approx 50\%$	$> 25\%$	$\approx 25\%$
scope	$< 25\%$	$< 25\%$	$\approx 50\%$
purpose	$< 25\%$	$\approx 25\%$	$\approx 50\%$

V. Conclussions

The work presented introduces a flexible and effective Web service description model that, thanks to its structure and features, minimizes the cost of Web service description and allows for a better scalability of the solution. The cost is decreased by the simplification of the description process aided with compartmentalisation of an organisation into sufficiently small units with a clearly defined context used with reference to its Web service assets. The scalability is achieved by resigning from the fully-fledged semantic description that relies on reasoning subsystems characterised by superb precision, yet suffering from low timewise efficiency.

The most important mechanisms were implemented as a prototype in order to measure the effectiveness of various mechanisms. The gathered data demonstrated that query matching with phrased queries was feasible for tens of millions of Web service operations in much less than one second. It was achieved thanks to the structure of query and ability to effectively filter off the unnecessary Web service operations. Local Context Anchoring was initially tested and proved to fetch results in more than 90% of test cases. Yet, the results must be tested manually by users in order to measure their satisfaction with the provided answers.

References

[1] P. Hitzler, M. Krtzsch, B. Parsia, P. F. Patel-Schneider, and S. Rudolph, "Owl 2 web ontology language primer," *W3C Recommendation*, vol. 27, no. October, pp. 1–123, 2009. [Online]. Available: http://www.w3.org/TR/2009/REC-owl2-primer-20091027/

[2] D. Roman, U. Keller, H. Lausen, J. D. Bruijn, R. Lara, M. Stollberg, A. Polleres, C. Feier, C. Bussler, and D. Fensel, "Web service modeling ontology," *Applied Ontology*, vol. 1, no. 1, pp. 77–106, 2005. [Online]. Available: http://portal.acm.org/citation.cfm?id=1412350.1412357

[3] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean, "Swrl: A semantic web rule language combining owl and ruleml," *Syntax*, vol. 21, no. May, pp. 1–22, 2004. [Online]. Available: http://www.w3.org/Submission/SWRL/

 ISBN: 978-1-61208-200-4

[4] R. Akkiraju, J. Farrell, J. Miller, M. Nagarajan, M.-T. Schmidt, A. Sheth, and K. Verma, "Web service semantics - wsdl-s," *International Business*, vol. 2008, no. Version 1.0, pp. 1–42, 2005. [Online]. Available: http://www.w3.org/Submission/WSDL-S/

[5] J. Kopecky, T. Vitvar, C. Bournez, and J. Farrell, "Sawsdl: Semantic annotations for wsdl and xml schema," *IEEE Internet Computing*, vol. 11, no. 6, pp. 60–67, 2007. [Online]. Available: http://oro.open.ac.uk/28624/

[6] J. Zhou, T. Zhang, H. Meng, L. Xiao, G. Chen, and D. Li, "Web service discovery based on keyword clustering and ontology," *2008 IEEE International Conference on Granular Computing*, vol. 1, pp. 844–848, 2008.

[7] M. Bruno, G. Canfora, M. D. Penta, and R. Scognamiglio, "An approach to support web service classification and annotation," *2005 IEEE International Conference on eTechnology eCommerce and eService*, pp. 138–143, 2005.

[8] B. Srivastava, K. Ponnalagu, N. C. Narendra, and K. Kannan, "Enhancing asset search and retrieval in a services repository using consumption contexts," *IEEE International Conference on Services Computing SCC 2007*, no. Scc, pp. 316–323, 2007.

[9] M. Espinoza and E. Mena, "Discovering web services using semantic keywords," *2007 5th IEEE International Conference on Industrial Informatics*, vol. 2, pp. 725–730, 2007.

[10] H. Gao, W. Stucky, and L. Liu, "Web services classification based on intelligent clustering techniques," *2009 International Forum on Information Technology and Applications*, pp. 242–245, 2009.

[11] B. Marr, G. Schiuma, and A. Neely, "Intellectual capital defining key performance indicators for organizational knowledge assets," *Business Process Management Journal*, vol. 10, no. 5, pp. 551–569, 2004. [Online]. Available: http://www.emeraldinsight.com/10.1108/14637150410559225

[12] M. Klusch, *Semantic Web Service Description*. Birkhuser Basel, 2008, vol. 16, no. 2, pp. 31–57. [Online]. Available: http://www.springerlink.com/index/10.1007/978-3-7643-8575-0

[13] T. Parr and K. Fisher, "Ll(*): the foundation of the antlr parser generator." in *PLDI*, M. W. Hall and D. A. Padua, Eds. ACM, 2011, pp. 425–436.

[14] E. M. Voorhees, *Query expansion using lexical-semantic relations*. Springer-Verlag New York, Inc., 1994, pp. 61–69. [Online]. Available: http://portal.acm.org/citation.cfm?id=188508

[15] A. Hogan, J. Umbrich, A. Harth, R. Cyganiak, A. Polleres, and S. Decker, "An empirical survey of linked data conformance," *Journal of Web Semantics (to appear)*, 2012.

[16] C. Fellbaum, *Organization of Verbs in a Semantic Net*. Kluwer, 1999, pp. 93–109.

[17] I. Niles and A. Pease, "Towards a standard upper ontology," *Proceedings of the international conference on Formal Ontology in Information Systems FOIS 01*, vol. 2001, pp. 2–9. [Online]. Available: http://portal.acm.org/citation.cfm?doid=505168.505170

[18] J. Conesa, V. Storey, and V. Sugumaran, "Experiences using the researchcyc upper level ontology," in *Natural Language Processing and Information Systems*, ser. Lecture Notes in Computer Science, Z. Kedad, N. Lammari, E. Mtais, F. Meziane, and Y. Rezgui, Eds. Springer Berlin / Heidelberg, 2007, vol. 4592, pp. 143–155.

[19] J. Rao and X. Su, "A survey of automated web service composition methods," *Semantic Web Services and Web Process Composition*, vol. 3387, no. 2, pp. 43–54, 2005. [Online]. Available: http://www.springerlink.com/index/4M6W37G0JFFK9BV4.pdf

[20] C. Garrod, A. Manjhi, A. Ailamaki, B. Maggs, T. Mowry, C. Olston, and A. Tomasic, "Scalable query result caching for web applications," *Management*, vol. 1, no. 1, p. 550561, 2008.

[21] H. Agt, G. Bauhoff, R.-D. Kutsche, and N. Milanovic, "Modeling and analyzing non-functional properties to support software integration," in *CAiSE Workshops*, ser. Lecture Notes in Business Information Processing, C. Salinesi and O. Pastor, Eds., vol. 83. Springer, 2011, pp. 149–163.

[22] C. V. S. Prazeres, C. A. C. Teixeira, and M. D. G. C. Pimentel, "Semantic web services discovery and composition: Paths along workflows," *2009 Seventh IEEE European Conference on Web Services*, pp. 58–65, 2009.

[23] M. O. Shafiq, R. Alhajj, J. G. Rokne, and I. Toma, "Lightweight semantics and bayesian classification: A hybrid technique for dynamic web service discovery," in *IRI*. IEEE Systems, Man, and Cybernetics Society, 2010, pp. 121–125.

[24] P. Plebani and B. Pernici, "Urbe: Web service retrieval based on similarity evaluation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 11, pp. 1629–1642, 2009.

[25] J. Cardoso, A. Barros, N. May, and U. Kylau, *Towards a Unified Service Description Language for the Internet of Services: Requirements and First Developments*. IEEE, 2010, p. 602609.

[26] M. Sabou, C. Wroe, C. Goble, and G. Mishne, "Learning domain ontologies for web service descriptions: an experiment in bioinformatics," in *Proceedings of the 14th international conference on World Wide Web*, ser. WWW '05. New York, NY, USA: ACM, 2005, pp. 190–198. [Online]. Available: http://doi.acm.org/10.1145/1060745.1060776

[27] U. Chukmol, A.-N. Benharkat, and Y. Amghar, "Enhancing web service discovery by using collaborative tagging system," *2008 4th International Conference on Next Generation Web Services Practices*, pp. 54–59, Oct 2008.

[28] S. Kona, A. Bansal, G. Gupta, and T. D. Hite, "Web service discovery and composition using usdl," *The 8th IEEE International Conference on ECommerce Technology and The 3rd IEEE International Conference on Enterprise Computing ECommerce and EServices CECEEE06*, pp. 65–65, 2006.

 ISBN: 978-1-61208-200-4