SOCIETY FOR SCIENCE AND EDUCATION
UNITED KINGDOM

TMLAI TRANSACTIONS ON
MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE

VOLUME 5 NO 4, 2017
ISSN 2054-7390

Semantic Web Service Discovery Framework using Multi-Agents System and NLP Techniques

A. Essayah, M. Youssfi, E. Illoussamen, K. Mansouri, M. Qbadou
Laboratory: Signals, Distributed Systems and Arti¬cial Intelligence (SSDIA)
ENSET Mohammedia, Hassan II University, Casablanca.
Bd Hassan II, Mohammedia, Maroc.
essayah.ahmed@gmail.com, med@youssfi.net, illous@hotmail.com, khmansouri@hotmail.com, qbmedn7@gmail.com

ABSTRACT

As a consequently increase amount of web services available on the web, automatic discovery presents a great challenge, in order to satisfy this requirement, semantic discovery approaches based on ontologies have been developed. However, end-users do not have knowledge about semantic web languages to express their requirements.

In this paper, we propose an automatic discovery framework based on multi-agent systems and natural language processing techniques to match a user request. The framework allows semantic matching through a set of semantic web services in order to enhance the accuracy of the discovery pattern, and to find a relevant match to the user request composed of keywords written in natural language. The use of multi-agent systems provide us the possibility to decrease the run-time by parallelizing simultaneous tasks, also, to implement various natural language processing techniques and matchmaking algorithms, and finally, they allow us to measure the users satisfaction through their behaviors dashboard analyze.

Keywords—Semantic web Service, Ontology, web service composition, functional and non-functional properties, Distributed information systems, natural language processing.

1 Introduction

Nowadays, with the emergence of web services, semantic web and service oriented architecture (SOA)[21], the number of web services over internet is rising, a demand increases for an automatic web service discovery framework that can provides high relevant services to the user requirements expressed using queries specified by functional and non-functional properties [19].

Web services are implemented through standard technologies: WSDL [14][25], UDDI [8], and SOAP[2]. These technologies facilitate the description, discovery, and communication between services. However, this basic infrastructure encode only a syntactic representation of what is expected by, and returned from, a service (its inputs and outputs), and does not allow to automate related web service management tasks such as discovery, selection, composition, invocation and orchestration.

DOI: 10.14738/tmlai.54.3183
Publication Date: 15th August, 2017
URL: http://dx.doi.org/10.14738/tmlai.54.3183

Web service discovery [10] is the process of searching relevant services to a user query. This process usually consists of matching a user request to a set of web service descriptions, in order to retrieve highly appropriate services to the user requirements.

Many syntactic and semantic web service discovery approaches have been proposed [16][1][13][15]. However, they present various limitations. First, some proposed frameworks require that the both of the web service provider and requester use the same ontology domain to allow the sematic matchmaker to measure the match degree between the user query and the web service description. Also, most of domain ontologies are mono-language. That means other web services described by different description languages will not be considered. Another limitation of some proposed approaches is that they require the requestor to be familiar with semantic web languages and express his requirements using a specific semantic description language such as WSMO [11], WSDL [24], OWL or OWL-S [7].

We address these limitations by providing a new framework for discovery of semantically enriched web services, the proposed framework allows end-user to search web services described using several semantic web description languages, through a request expressed in natural language keywords. The discovery process of our framework can be summarized as follows:

Understanding user requirements using natural language techniques [23] such as word sense disambiguation [22], part-of-speech tagging [4], tokenization, sentence segmentation, Stop word removal, lemmatization and Stemming to establish the user request context.

Extracting keywords from semantic or syntactic web service description to specify the web service context.

Measuring similarity degree between user request context and web service context.

Ranking the selected web services using several matchmaking algorithms considering functional and non-functional properties (e.g. price, location, service availability, Quality of Service ...)

Measuring the user satisfaction to adapt the discovery mechanism to the user preferences for later requests.

Multi-agent systems [5] provide a new method to analyze, describe and implement sophisticated applications that need to distribute tasks between autonomous entities. The use of Multi-agent system allows us to adopt various natural language processing techniques such as PoS [4], WSD [22] … and discover web services described in different web service description languages like WSDL [11], WSMO [11], OWL-S [7] …, with a minimum adaptation for Worker Agents. Agents ensure a good portability because they can be deployed in several types of devices including mobile devices.

The remaining of this paper is organized as follows. In section 2 we provide an overview of the related work performed in the area of semantic web service discovery, whereas in section 3, we provide some background knowledge about technologies that are integrated in our framework, namely NLP techniques, web service description languages and Multi-agent systems. In section 4 we present an overview of the proposed framework then we discuss in details its architecture. Finally, section 5 concludes the paper.

2 Related Works

Many research have been made to create a web service discovery framework. Generally, all the proposed approaches are based on syntactic or semantic matchmaking algorithms.

M. Paolucci, T. Kawamura, T.R. Payne, and K. Sycara [16] proposed a semantic matchmaking algorithm consisting of a similarity degree obtained by matching the user request inputs and outputs to the web service inputs and outputs. Four different matching degrees are possible: Exact Match (When the user request inputs/outputs match exactly the web service inputs/outputs), Plugging Match (When the web service inputs/outputs plug the user request inputs/outputs), Subsumed Match (When the user request inputs/output subsume the web service inputs/outputs), and otherwise, a Fail degree is generated.

A. Bener, V. Ozadali, and E. Ilhan [1] extended Paolucci's matchmaking algorithm by proposing an algorithm that performs as well as inputs/outputs matching, preconditions, effects and service context. J.M. García, D. Ruiz, A. Ruiz-Cortés [13] proposed to improve the run-time by adding a preprocessing stage to discard all web services that not contain the concepts expressed by the user request. Using SPARQL this preprocessing mechanism improve the run-time, but it has a negative influence in term of precision.

A knowledge based approach for semantic web services discovery have been presented in [15], the mechanism is based on knowledge and multi-level search for semantic web services discovery. Which an intelligent broker utilizes a knowledge based system and multi-level search to overcome the drawbacks of syntactic and semantic discovery. The multi levels of service discovery allows them to improve the probability of accurate matching.

Our proposed framework considers all the end-users as a non-technicians, we extract concepts from a keyword-base user request using NLP techniques and we define automatically the user request and the web service context, all cited algorithms are effective for single web service description language, instead of from our approach that can be used to discover services described in various web service description languages such as OWL-S, WSMO, WSDL,…

3 Background

In this section we briefly describe some concepts that are integrated in our proposed framework, we start by presenting some natural language processing techniques used to process a requestor requirements expressed using a human language, Then, we define some semantic web concepts and languages used to describe syntactically and semantically a web service, and finally we finish this section by presenting an overview about multi-agent systems.

3.1 Natural language processing techniques (NLP).

Refers to the use and ability of computers to process and understand sentences expressed in a human (natural) language such as English, NLP [23] is a growing area of research with several applications in different fields. Technically our proposed framework employs a set of natural language processing techniques which are listed below:

Tokenizing[4]:Word tokenizing is an important part of text processing, every natural language needs to normalize the text and chop it up into pieces called tokens, text tokenization may also ignores some characters such as punctuation. For example, a sentence like “Hotel in Casablanca Morocco, with a good price” can be divided into 8 tokens {“Hotel”, “in”,“Casablanca”, “Morocco”, “with”, “a”, “good”,“price”}.

Stop word removal: stop word is a predefined set of words from a language that are excluded from the vocabulary because they don’t give a relevant semantic information. An example of stop words list is shown in Figure 1.The last example will be composed of five tokens after applying Reuters-RCV1 stop word list {“Hotel”, “Casablanca”, “Morocco”, “good”, “price”}.

a	an	and	are	as	at	be	by	for	from
has	he	in	is	it	its	of	on	that	the
to	was	were	will	with					

Figure.1. Stop list of 25 semantically non-selective words which are common in Reuters-RCV1

Part-of-speech Tagging[23]: PoS tagging is the process of tagging words based on their context with its corresponding part-of-speech (PoS). Many PoS Tagger are available such as Brill’s rule-Based PoS Tagger [4] from natural language toolkit library (NLTK) [3].

Stemming and lemmatization: the aim of stemming and lemmatization process is to reduce tokens into their stem form called lemma. Depending stemmers/lemmatizers tokens stemming and lemmatization can be done various techniques such as, Terms normalizing (e.g. are, is, am = be), equivalence classes definition, asymmetric expansion (e.g. Hotel, hostel, motel, resort = hotel), affixes chopping (e.g. development, developments, developing, developer = develop). Many words lemmatizers/stemmers are available, the most common for English language is Porter Stemmer [20] and WordNet lemmatizer from NLTK library [3].

Word sense disambiguation [22]: after tokens lemmatization a word sense disambiguation process takes place to identify which meaning of a polysemy lemma is employed in a sentence. WSD algorithms use the sentence’s context and previous disambiguated words to calculate a similarity score.in our case, word sense disambiguation will be applied to the set of lemmas extracted from the user request and from the web service description, producing two sets of two sets of disambiguates meanings that can be used in the discovery and matchmaking algorithms.

3.2 Web service description languages

To promote the automation of web service management tasks, a number of different semantic languages have been created such as, WSMO [11] and OWL-S [7], to provide a more powerful method for web service discovery, selection, composition, invocation and orchestration by describing functionalities of services in a machine interpretable form.

WSDL (Web Service Description Language)[14]: is a W3C (World Wide Web Consortium) standard for syntactic web service description, WSDL 2.0[24] used to describe the web service interface and the message exchange patterns (Structure of SOAP messages).

OWL-S (Ontology Web Language for Services)[7]: formerly DAML-S, is an ontologies-based web service description language, in OWL-S each service is described using three XML-Based parts which are: Service Profiles (to describe the web service capabilities), Service Model (describing web service behaviors) and Service Grounding (to provide details about the web service invocation).

WSMO (Web Service Modelling Ontology)[11]: provides a semantic modelling language for web services capabilities description. In WSMO, a web service is described using four core elements, which are: Ontologies, Goals, Web services and Mediators.

UDDI (Universal Description Discovery and Integration)[14]: is a standard web service registry, that allows services providers through an API to publish his services, and services requesters to search for desired web services. UDDI offers a syntactic description of its content. However, it does not present semantic models to understand the user requests and web service capabilities.

3.3 MULTI-AGENT systems

The equations are an exception to the prescribed specifications of this template. You will need to determine whether or not your equation should be typed using either the Times New Roman or the Symbol font (please no other font). To create multileveled equations, it may be necessary to treat the equation as a graphic and insert it into the text after your paper is styled.

Multi-agent system (MAS) [5] is a collection of interacting agents that are used to solve complex and distributed problems. Agents are loosely coupled which allows adding new functions or removing existing features easily by creating or destroying agent without affection the whole system structure. In our case, the following features of MAS are especially interesting:

Goal orientedness: An agent is able to handle complex tasks, it should decide in which order and in which way the task should be processed. That feature allows us to adopt various natural language processing techniques and discover web services described in different web service description languages with a minimum adaptation for Worker Agents.

Autonomy: Agents are highly autonomous, they operate without human or other agents' intervention. Which allows us to add and remove various functionalities to our framework without affection the system.

Portability: Agents can be deployed in various devices that are supporting JADE/LEAP [5], such as mobile devices, personal computers, servers.

4 The Proposed Framework

4.1 Semantic Web Service Discovery And Composition Framework

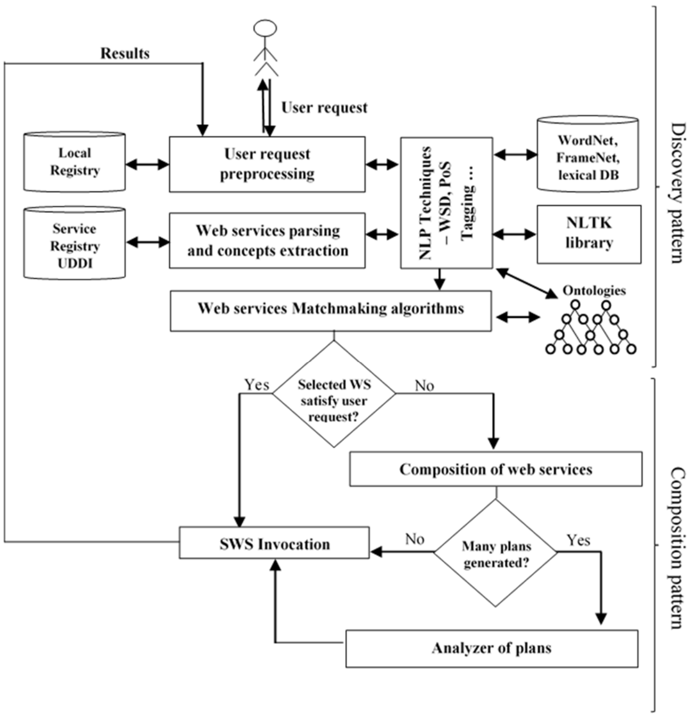
Results
User request
Local Registry
User request preprocessing
Service Registry UDDI
Web services parsing and concepts extraction
NLP Techniques – WSD, PoS Tagging …
WordNet, FrameNet, lexical DB
NLTK library
Ontologies
Web services Matchmaking algorithms
Yes
Selected WS satisfy user request?
No
Composition of web services
SWS Invocation
No
Many plans generated?
Yes
Analyzer of plans
Discovery pattern
Composition pattern

Figure.2. Web service discovery & composition framework global architecture.

Our proposed framework shown in figure 2 aims to discover web services using various web service discovery techniques in order respond to a keyword-based user request expressed using natural languages. When no selected web service matches the user requirements, a service composition process take place to create new functionality by combining the features offered by other existing services. Our proposed framework consists of two different patterns:

- Discovery pattern: The web service discovery pattern consists of keywords-based discovery process for searching web services described using semantic or syntactic languages, the discovery pattern use NLP techniques to establish a match between a user request expressed using keywords and a semantic or a syntactic web service description.
- Composition pattern: The composition mechanism take place when no selected web service meet the user requirements, Our approach based on workflow, consider a composite service as a business process. A composite service (composite plan) includes a set of atomic services as well as controls and data exchange between them. Similarly, a business process is composed of a set of elementary structured activities, as well as the execution order between them.

In this article, the focus is on the discovery pattern, the composition process will be the subject of future works.

4.2 Web Service Discovery Pattern

To satisfy the user requirements, the web service discovery pattern shown in figure 3, uses many components. First, a proxy agent which is the interface between the user and the platform, it's capable of receiving the user request, looking for a response in the local registry, communicating with intermediate agents, executing services, delivering results to the user, measuring user satisfaction and saving the relevant solution to the user request as an history in the local registry. The proxy agent can be hosted by personal computers, servers and mobile devices that are supporting JADE/LEAP [5] such as smart phones, cars, smartwatch and all other wearable devices. Second, a NLP container that allows keywords-based discovery process for searching web services described using semantic or syntactic languages. And finally, a discovery container that uses various web service description languages parsers to extract web service descriptions from the web service register, and different matchmaking algorithms to match between a user request expressed using keywords and a semantic or a syntactic web service description.

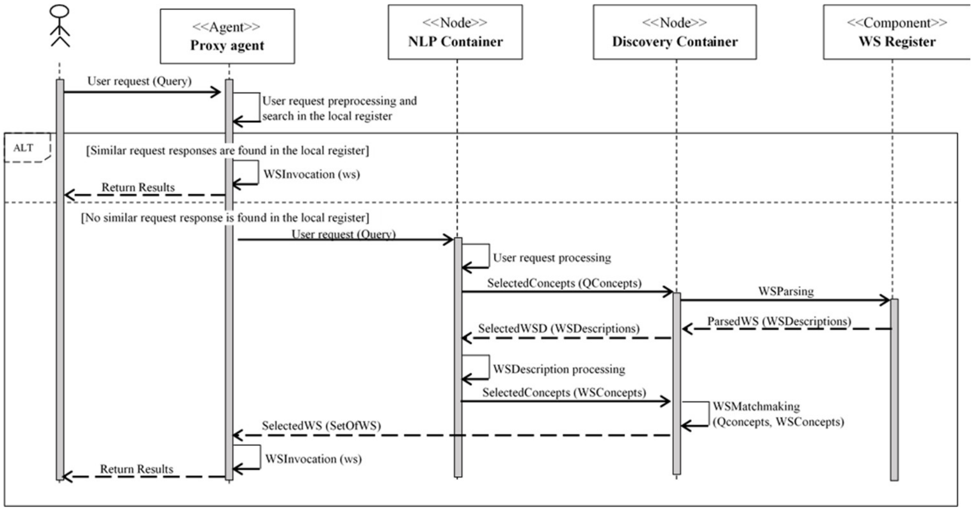

Figure.3. Web service discovery pattern sequence diagram

4.2.1 Natural language processing (NLP) Container:

A service requestor express its requirements using a collection of words. Because a word can have various senses (e.g., pen can be used to mean either an instrument for writing or drawing with ink or a female swan).

NLP container as shown in figure 4, is used to process the user request and the web service descriptions and extract the context and the meanings. The NLP mechanism start by receiving a user query from the proxy agent, the intermediate agent, is the interface between the NLP Container agents and the other system's components. The intermediate agent initiates the user request and the web service description processing by sending a tokenizing request to the NLTK library, to normalize the text and chop it up into pieces called tokens. After receiving tokens the intermediate agent calls a PoS tagging method to tag tokens based on their context with its corresponding part-of-speech (PoS), then we remove the stop words. The intermediate agent then creates WAn worker agents corresponding the number of tokens, each worker agent send a stemming request to the NLTK library to reduce token into its stem form called lemma. In order to identify which meaning of a polysemy lemma is employed, worker agents invoke a word sense disambiguation method, selected concepts are send to the intermediate agent to be delivered as a set of user query concepts (QConcepts) or web service description concepts (WSConcepts) to the discovery container to start the discovery process.

4.2.2 • Discovery Container:

The web service discovery container shown in figure 5, aims to select web services from the service registry, parse and read web service descriptions to extract functional and non-functional concepts and attributes and finally rank selected web services using matchmaking algorithms.

After receiving a discovery request from the NLP Container, the intermediate agent start by creating different Parser agents, each parser agent use specific web service description reader and parser to be able to read and extract relevant information through web service description written in different languages. For OWL-S a parser like Sesame [18] or Jena [12] can helps in this process and for WSMO which is written using WSML parser like WSMO4J [8] can be used. Once the parsing process finish, a sets of relevant web services description related words are selected and sent to the NLP Container (figure 3) to be processed. The NLP Container define the meanings of each word by applying a various NLP techniques like word tokenizing, Stemming and lemmatization and word sense disambiguation. When the NLP mechanism is done, the discovery container receive again a set of selected user request and web services description meanings to match. The goal of matchmaking is to process the user request and the web service description concepts and measure the similarity degree between them. A ranked set of the selected web services is generated. Finally, the proxy agent invokes the selected web service and measures the user satisfaction to adapt the discovery mechanism to the user preferences for later requests.

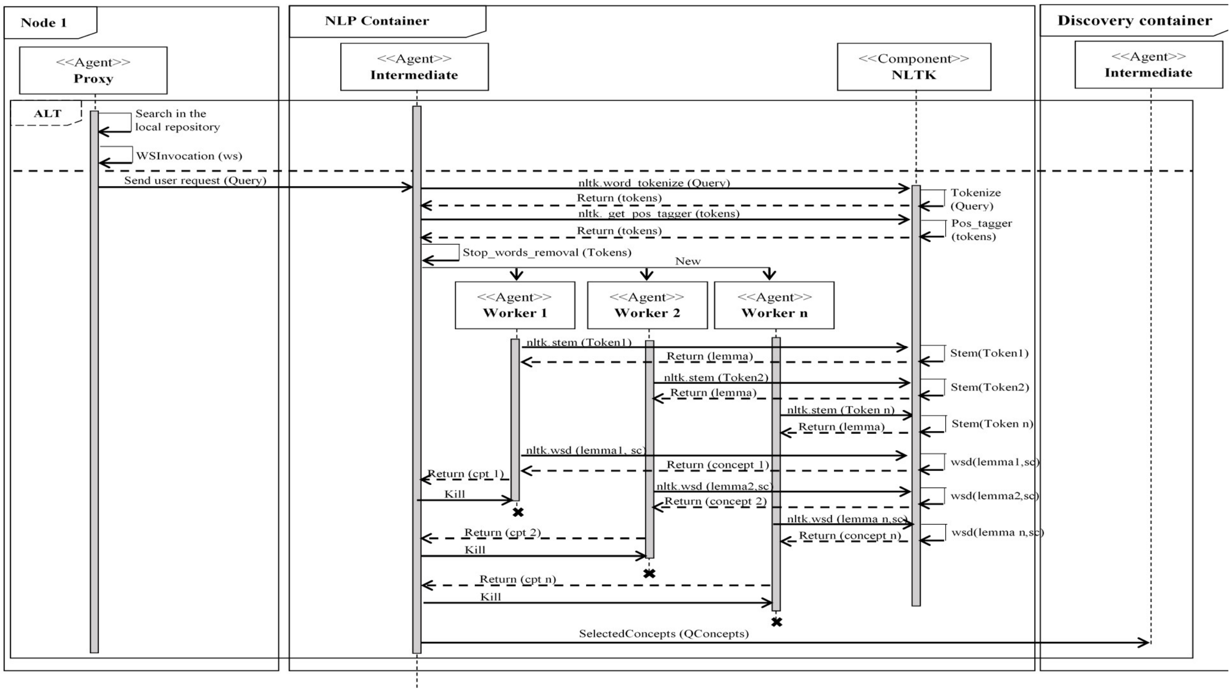

Figure.4. NLP Container sequence diagram

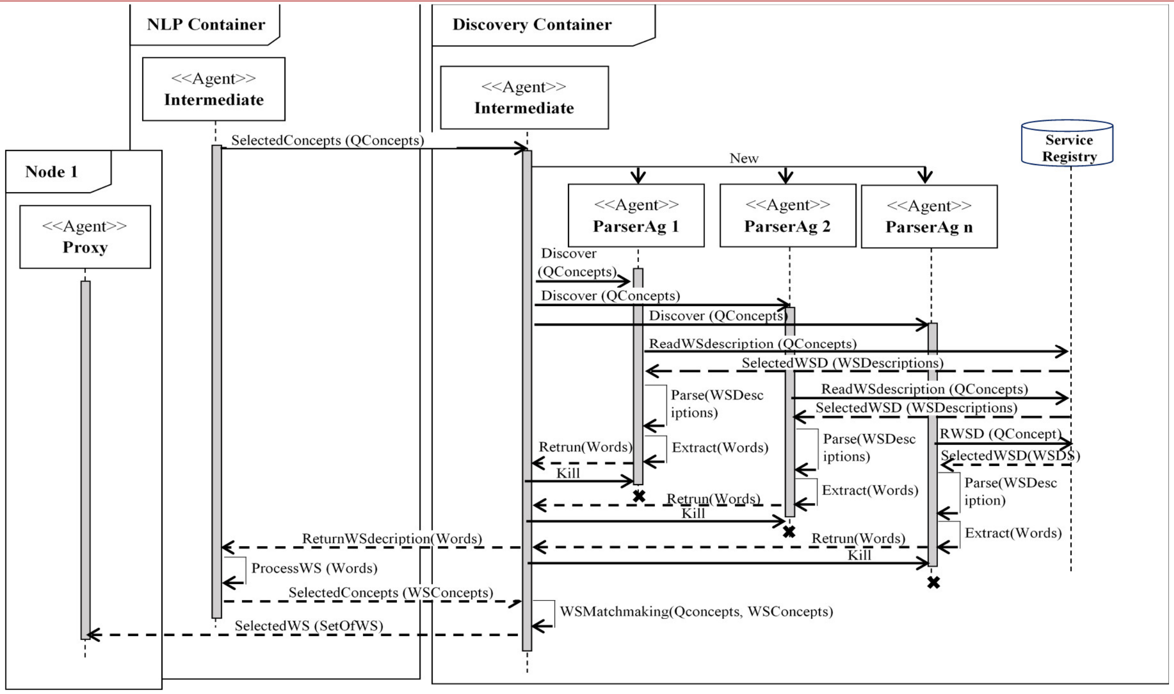

Figure 5. Discovery Container sequence diagram

5 Conclusion and Perspectives

In this paper we presented a framework that allows web service discovery based on user request expressed using human language.

Our proposed framework considers the end-user as non-technician, in fact, we don't require any knowledge related to the web service technologies, the discovery process start by applying NLP techniques to extract concepts from a keyword-base user and then defining automatically the user request and the web service context. Furthermore, our proposed system uses various web service matchmaking algorithms to find the best match to the user requirements. The use of Multi-agent system permits us to implement several web service description languages parsers and execute parallel tasks in order to improve the run-time.

To extend our proposed framework, composition mechanism is required. Future work includes more details about our composition pattern based on workflow and considering a composite service as a business process.

REFERENCES

[1] A. Bener, V. Ozadali, and E. Ilhan, Semantic matchmaker with precondition and effect matching using SWRL. Expert Systems with Applications, 36(5), 9371–9377, (2009).

[2] A. Navarro, A. da Silva, A Metamodel-based definition of a conversion mechanism between SOAP and RESTful web services, Computer Standards & Interfaces, Volume 48, Pages 49-70, ISSN 0920-5489? November 2016.

[3] Bird, Steven, Edward Loper and Ewan Klein (2009), Natural Language Processing with Python. O'Reilly Media Inc. (2009)

[4] Brill, E., 1992. A simple rule-based part of speech tagger. In: Proceedings of the Workshop on Speech and Natural Language. Association for Computational Linguistics, pp. 112–116.

[5] C. Castelfranchi, Y. Lespérance, Developing Multi-agent Systems with JADE Intelligent Agents, In Intelligent Agents VII Agent Theories Architectures and Languages, Vol. 1986,(2001)

[6] D. D. Lewis,Y. Yang, T. Rose, and Li, F. RCV1: A New Benchmark Collection for Text Categorization Research. Journal of Machine Learning Research, 5:361-397, 2004. http://www.jmlr.org/papers/volume5/lewis04a/lewis04a.pdf.

[7] D. Martin, M. Paolucci, S. McIlraith,M. Burstein,D. McDermott, D. McGuinness and al. Bringing semantics to web services: The OWL-S approach. Semantic Web Services and Web Process Composition; p 26-42, 2005.

[8] EU IST, FIT-IT. (2008). WSMO4J API. Available from<http://wsmo4j.sourceforge.net/>.

[9] H. C. Boas, "Bilingual FrameNet Dictionaries for Machine Translation", in Proceedings of the Third International Conference on Language Resources and Evaluation, Las Palmas, vol. IV, pp. 1364-1371. 2002.

[10] H. Cancela, A. Cuadros-Vargas, A. D. Renzis, M. Garriga, A. Flores, A. Cechich, A. Zunino, CLEI 2015, the XLI Latin American Computing ConferenceCase-based Reasoning for Web Service Discovery and Selection, Electronic Notes in Theoretical Computer Science, Volume 321, Pages 89-112, 2016.

[11] Hai H. Wang, Nick Gibbins, Terry R. Payne, Domenico Redavid, A formal model of the Semantic Web Service Ontology (WSMO), Information Systems, Volume 37, Issue 1, Pages 33-60, March 2012.

[12] HP Labs Semantic Web, (2009). Jena. Available from<http://jena.sourceforge.net/>.

[13] J.M. García, D. Ruiz, A. Ruiz-CortésImproving semantic web services discovery using Sparql-based repository filteringWeb Semantics: Science, Services and Agents on the World Wide Web, 17, pp. 12–24 (2012).

[14] M. B. Juric, A. Sasa, B. Brumen, I. Rozman, WSDL and UDDI extensions for version support in web services, Journal of Systems and Software, Volume 82, Issue 8, Pages 1326-1343, August 2009.

[15] M. El Kholy and A. Elfatatry, "Intelligent broker a knowledge based approach for semantic web services discovery," Evaluation of Novel Approaches to Software Engineering (ENASE), 2015 International Conference on, Barcelona, pp. 39-44, 2015.

[16] M. Paolucci, T. Kawamura, T.R. Payne, K. Sycara, Semantic matching of web services capabilitiesThe Semantic Web–ISWC 2002, pp. 333–347, Springer (2002)

[17] OASIS, 2004. UDDI Version 3.0.2, UDDI Spec Technical Committee.

[18] openRDF.org. (2009). Sesame. Available from<http://www.openrdf.org/>.

[19] P. Li, M. Comerio, A. Maurino and F. D. Paoli, "Advanced Non-functional Property Evaluation of Web Services," Web Services, 2009. ECOWS '09. Seventh IEEE European Conference on, Eindhoven, 2009.

[20] Porter, Martin F. An algorithm for suffix stripping. Program 14 (3): 130-137.1980.

[21] R. J. Rabelo, O. Noran and P. Bernus, "Towards the Next Generation Service Oriented Enterprise Architecture," 2015 IEEE 19th International Enterprise Distributed Object Computing Workshop, Adelaide, SA, 2015, pp. 91-100.

[22] R. Navigli, & P. Velardi. Structural semantic interconnections: A knowledgebased approach to word sense disambiguation.IEEE transactions on pattern analysis and machine intelligence (Vol. 27, pp. 1075–1086). IEEE Computer Society (2005).

[23] Venkat N. Gudivada, Dhana Rao, Vijay V. Raghavan, Chapter 9 - Big Data Driven Natural Language Processing Research and Applications, In: Venu Govindaraju, Vijay V. Raghavan and C.R. Rao, Editor(s), Handbook of Statistics, Elsevier, Volume 33, Pages 203-238, 2015.

[24] W3C, 2007. Web Services Description Language (WSDL) Version 2.0, W3C Recommendation.