<http://www.diva-portal.org>

Postprint

This is the accepted version of a paper published in *Computers & electrical engineering*. This paper has been peer-reviewed but does not include the final publisher proof-corrections or journal pagination.

Citation for the original published paper (version of record):

Cheniki, N., Belkhir, A., Atif, Y. (2016)

Mobile services discovery framework using DBpedia and non-monotonic rules.

Computers & electrical engineering, 52: 49-64

<http://dx.doi.org/10.1016/j.compeleceng.2016.02.016>

Access to the published version may require subscription.

N.B. When citing this work, cite the original published paper.

Permanent link to this version:

<http://urn.kb.se/resolve?urn=urn:nbn:se:his:diva-12065>

Mobile services discovery framework using DBpedia and non-monotonic rules

Nasredine Cheniki^a, Abdelkader Belkhir^b, Yacine Atif^c

^aUniversity of Sciences and Technology Houari Boumedine USTHB, Algeria

^bLaboratory of Computing Systems University of Sciences and Technology Houari Boumedine, USTHB, Algeria

^cSchool of Informatics, University of Skövde, Sweden

Abstract

Mobile services are in constant evolution thanks to mobile device performances and advances in wireless networks. As a result, there is a need to supply efficient discovery processes by enriching service descriptions with semantics through ontologies. In this paper, we propose to use ontologies that cover multiple domains, provided by DBpedia (a semantically-structured version of Wikipedia) to annotate mobile services across different domains. DBpedia is the nucleus for the Web of Data, contains millions of triples described in multiple languages and provides lightweight ontologies, in addition to open tools that we use to develop a user-centric mobile services discovery framework. This approach enables a semantic-based matching process using the resources, concepts and categories of DBpedia, as well as a ranking mechanism using non-monotonic rules to allow users defining alternative choices based on the context and the quality of the service information. Experimental results show that our mobile service discovery framework provides high levels of recall and precision performances.

Keywords: Mobile, services, discovery, context information, QoS information, semantic search, non-monotonic rules, DBpedia, lightweight ontologies.

1. Introduction

PortioResearch [1] revealed that mobile customers reached 6.5 billion subscribers in the beginning of 2013 and it is expected to attain 8.5 billion by the end of 2016. It indicates also that handsets industry will continue to grow in the next couple of years as more users are getting broadband access to Internet services through their mobile devices. On the other hand mobile devices' performance is constantly increasing and mobile communication networks are rapidly growing. This rapid evolution has resulted in a wide proliferation of pervasive environments in which, mobile services are accessible, at anytime and from anywhere. Because they improve consumers' daily life, these services are massively expanding across our mobile ecosystems and are expected to provide further novel experiences and additional levels of personalization opportunities to consumers. Efficient discovery mechanisms to locate intended services in mobile and dynamically changing environments are expected to meet the challenges of these advances in mobile services' provisioning.

Service discovery is the process of finding the appropriate services by evaluating a service query [2]. This process usually consists in matching a user query to a set of service definitions, in order to retrieve targeted services. In a mobile environment, a user request should include context information as well (such as location, time and preferences), in addition to Quality of Service (QoS) information (such as price, rank and service availability). These criteria play an important role in the discovery process and help delivering the right services to the right user at the right time, in the right place, and with right expectations.

Mobile service discovery that is based on semantic matching techniques has received a significant attention from researchers in recent years. Considering concepts and their relationships, semantic-matching could overcome the limitations of traditional matching models that are based on purely syntactic-matching. Semantic matching relies on ontologies which are conceptual models describing usually a specific domain knowledge. Ontologies are widely used to describe services as well as their context and their intrinsic QoS information. The formal language representation of these semantic attributes allows a common understanding and facilitates information sharing, integration and reasoning [3].

Previous research works have used either generic or domain-specific ontologies to enrich their mobile service descriptions [4] [5] [6] [7] [8] [9]. These ontologies usually cover a restricted number of domains, are costly in terms of maintainability [10] [11] and some of them are considered to be over-expressive for semantic search or discovery processes. In addition, due to the variety of mobile services, they could not be well covered by domain-specific ontologies, and as domains evolve, old service descriptions become obsolete. On the other hand, most of domain ontologies are mono-language; if multilingual discovery is considered, an extra effort is needed, for example by translating offline concepts and properties into a target language.

In this paper ¹, we propose to overcome these limitations by annotating mobile services using DBpedia [10], a multi-domain knowledge base extracted from Wikipedia, to enable the design and development of a semantic Web services discovery framework based on structural containers (concepts, resources and categories). DBpedia covers multiple domains with a large number of concepts, instances and categories that could be used to describe a considerable range of services. It combines advantages of multi-domain as well as domain-specific ontologies, by linking its instances to concepts from hand-crafted ontologies to enable rich semantic-applications development. Besides, DBpedia evolves dynamically as Wikipedia changes. Hence, DBpedia is constantly updated and new information is regularly added. Our proposed discovery framework supports multiple languages, where users have the ability to search for mobile services across different languages thanks to the internationalization characteristics of DBpedia.

The proposed approach is user-centric, where the discovery solution involves service providers who describe their services in terms of offered capabilities. Users issue requests using keywords without getting involved in the semantic details that are used to annotate provisioned services for effective discovery. The framework provides a ranking mechanism that sorts mobile services according to user's context and QoS expectations. The user defines a list of alternative choices represented by non-monotonic rules in order to select services that satisfy his/her context and QoS requirements, based on an extended context ontology.

The contributions of this paper can be summarized as follows :

- A user-centric mobile services discovery framework based on the consumption of Linked-Data provided by DBpedia and non-monotonic rules. DBpedia which enfolds cross-domain ontologies is used to describe functional properties (capabilities) of mobile services.
- To represent non-functional properties of mobile services, we propose an extended ontology that represents context and Quality of service information., to enable:
- A service ranking and filtering process for mobile services based on non-monotonic semantic Web rules, used by consumers to represent their non-functional mobile service attributes.

In this work, we consider mobile services as any service that could be published in a mobile environment, for the purpose of being discovered by mobile users from their devices. Context information distinguishes this type of services from traditional ones. Service providers are able to announce the provision of concrete services that could be discovered from mobile devices, such as restaurant, taxicab, healthcare or mechanic services, in addition to software services.

The remaining sections of this paper are organized as follows: Section 2 outlines some of the related research works. Section 3 introduces a motivational scenario illustrating the importance of mobile services discovery. Then, Section 4 introduces MobiSO, a mobile services specification of an ontology to describe and annotate mobile services, their context and related QoS information. Section 5 discusses semantic annotations and the integration of DBpedia

¹This article is an extended version of [12]

in mobile services specifications. Section 6 discusses semantic Web rules focusing on non-monotonic rules. Section 7 reveals the architecture of the proposed discovery framework. Section 8 provides a prototype implementation of the proposed discovery framework along with an overview of the used technologies and tools. This is followed by an evaluation of the proposed framework focusing on discovery effectiveness and performances. Finally, Section 9 concludes the paper with a summary of our work and a proposed set of future research perspectives.

2. Related work

Traditional UDDI based discovery approaches have raised limitations; obviously they have not been designed to support semantic search or adapt discovery processes to take into consideration mobile services context and QoS properties, which are important to extend the scope of discovery beyond traditional fixed points promoted by UDDI specification. Subsequently, mobile services discovery has drawn a great focus from a wide range of research communities. EASY [4] services are described using a proposed OWL based ontology called EASY-L. A set of conformance relations EASY-M (Matching) based on EASY-L were developed to perform context and QoS based service matching. The proposed discovery solution extends the base algorithm for semantic service matching developed by Paolucci et al. [13]. ConQo [5] enhances services discovery by considering context and QoS information. Authors proposed a new ontology for context information called WSMO-Context in order to extend a QoS-based ranking algorithm. Mustafa Adacal et al. [6] proposed an agent-based mobile Web services discovery approach where each service is described with a generic service ontology that allows service providers to refer to a specific domain and a service class with functional and non-functional attributes. A simple matching algorithm is proposed based on a matching service domain and related classes in addition to contextual and preferences properties. ProMWS [7] proposes pre-fetching strategies to predict users query in order to enhance mobile peer-to-peer service discovery. The solution provides a simple semantic matching algorithm that relies on a single common ontology. Toninelli et al. [8] describe services by their provided capabilities that are annotated through domain specific ontologies. Authors extend Paolucci et al. [13] matching algorithm to develop a semantic capability-based matching algorithm. Niazi and Mahmoud [9] integrated their own service ontology into Delivery Context Ontology released by W3C to develop a semantic matching and ranking algorithm which is based on user profile, preferences and device capabilities described using the proposed ontology.

Previous works bring interesting solutions for mobile services discovery, however to the best of our knowledge none of them tackled discovery aspect using multi-domain ontologies to match functional properties of mobile services, and semantic non-monotonic rules to rank their non-functional parameters.

3. Motivating Scenario

Adam is traveling on the highway; during the road, a crash occurred between him and a car driving in front of him. Adam rushed out of the car after the shock to reassure the situation of the car owner he collided with and whom he found completely fainted. Adam takes out his smart phone to quickly look for the nearest mobile doctor on the same road using MobiSDiscovery (an application that facilitates mobile services discovery). The application contacted the server that contained information about persons providing services who were within walking distance of the accident.

Unfortunately the application could not find any doctor, but on the other hand, it found a nurse within a walking distance. Adam asked him to provide medical relief service, so a message alert was received by the nurse who opened the MobiSDiscovery application, he received a urgent request for medical service and identified the place which asked him to help and head directly to him. The nurse arrived and provided first aid to the injured before the ambulance is up there; which often required a long time by virtue of its distance from the scene resulted in rescuing the injured. After check on the status of the patient, who was transferred to the hospital, Adam finished up that his car had got a malfunction and must be taken to the nearest mechanic. He used the same application to look for the nearest car carrier with lowest price. Fortunately, he found the service he was looking for in a walking distance from him and then the service owner was heading for to the scene after Adam summoned the owner of the service by the MobiSDiscovery application.

The above scenario illustrates some benefits of semantic mobile services discovery. Semantic-based discovery offers more efficient results for users by extending user query with more semantically-related terms and concepts (for

example searching for a doctor may lead to any healthcare stuff). In addition, taking context (such as geographic position) and quality of service (such as price) properties increased reliability and efficiency of the discovery issue. In this work we offers a such discovery capabilities by providing semantic, context and quality of service based discovery.

4. MobiSO, an ontology for mobile services description

Many research efforts have been conducted to develop context and QoS information models for mobile services and applications. These models are initially created to express devices' capabilities and user preferences. They are based on eXtensible Markup Language (XML), Resource Description Framework (RDF) [14], Resource Description Framework Schema (RDFS) [15] or Ontology Web Language (OWL) [16]. Semantic Web technologies (RDF, RDFS, OWL) are widely used because they support interoperability, sharing, integration and reasoning [3] processes.

In order to describe devices' capabilities and features, XML models have been used at first. However, models based on XML do not provide any semantics and thus they have been rapidly substituted with semantically-enriched RDF(S)/OWL models. Composite Capabilities Preference Profile (CC/PP) [17] is a W3C recommended RDF model to describe device specifications and user preferences. It can be used by content providers to adapt their content for various devices. User Agent Profile (UAProf) [18] is a specialization of CC/PP usually used by content providers to output adequate format for different devices. RDF models allow common understanding and facilitate content customization. However, they still lack expressiveness and do not describe the semantic relations of their different components, which result in limited semantic inferences.

Ontology-based models (mainly based on RDFS/OWL) empower RDF models to overcome the above limitations. Context Ontology Language (CoOL) [19] is a generic context description language created based on a proposed model called Aspect Scale Context Model (ASC) to provide context interoperability and facilitate service discovery. However, this ontology is not suitable to describe abstract context information [20]. CoDAMoS [21] is another generic and extensible ontology that defines important reusable context ontologies such as User, Environment, Platform and Service.

In Web service domain, Web Service Modeling Ontology (WSMO) and OWL-S are two widely-used ontologies allowing semantic descriptions of Web services to enable the automation of service discovery and composition. WSMO-Context [5], WSMO-QoS [22] and WSMO-M (Mobile) [23] extend WSMO in order to support context and/or QoS information, while OWL-S has been extended in [24] to add context attributes that enable contextual and inherent service compositions in pervasive environments. WSMO and OWL-S models are heavyweight solutions and present high complexity in terms of definition and processing.

The above models exhibit different drawbacks for contextual information. Authors in [20] proposed mIO! an ontology network composed from a set of ontologies developed to model context information for contextual applications and services. This ontology contains 433 classes, 277 object properties, 156 datatype properties and 364 instances. It is actually made up of eleven ontologies: Service, Provider, User, Environment, Location, Device, Time, Interface, Network, Source, Role. Many of these ontologies have reused existing ontologies such as FOAF, SOUPA, Delivery and CoDAMoS. mIO! Ontology Network was designed to facilitate the creation of extensions for concrete use cases.

We have developed an extension that fits our requirements called MobiSO (Mobile Services Ontology). In our ontology (see Figure 1), we differentiate context information from QoS information by defining two adequate classes *ContextInformation* and *QoSInformation*. *MobileService* class represents mobile services and extends the generic class *Service*. *ConcreteService* class is defined to distinguish concrete services from software services. *MobileServiceProvider* class models providers of mobile services.

ContextInformation class contains information such as location, address and current time, while *QoSInformation* class includes attributes such as price, service availability days and rate. More information could be added from mIO! Ontology as needed; for instance, we can incorporate Device ontology to *ContextInformation* class to describe device capabilities.

In order to enrich semantically functional description of mobile services, we have introduced three annotation properties: *resourceReference*, *conceptReference* and *categoryReference*. These properties will be used to map an external resource (individual), concept (class) or category reference to a service definition (described later in section 5). This approach is inspired from SAWSDL (Semantic Annotations for WSDL and XML Schema) [25], which extends Web service description documents (WSDL) with pointers to external semantic definitions. Each service in

Figure 1. Essential parts of Mobile Service Ontology (MOBISO) Model

MobiSO is described with external reference(s) and associated to optional capabilities that are themselves described by external references. We adopt Capability-based service description as it allows the development of user-centric discovery algorithm [8].

Listing 1 illustrates our model with a description of a medical service associated with its context and QoS information (using Turtle syntax). This service is identified by an instance of *ServiceIdentification* class, which contains a name property (lines 11,15 and 16). The described service has context information revealing its geographic position (lines 19,22-28), and it has also three supported languages (line 20). The quality of service information is described by service providing days (line 31), and its price (line 32-36).

5. Semantic Annotations with DBpedia

Tim Berners Lee (the inventor of the Web) along with other colleagues [26] have introduced the semantic Web initiative that aims to structure Web contents to make it machine-accessible and understandable by adding descriptive information (meta-data) to existing Web resources (Web pages, Web services, images, etc.). This process is called semantic annotation, where semantically well-defined terms are associated to Web resources in order to remove information ambiguity and allow software agents to understand, exchange, integrate and discover information automatically. This initiative has been adopted in many research projects aiming at structuring Web contents. Wikipedia, an open gigantic source of knowledge publicly available on the Web, has been targeted by researchers to automatically extract its content and make it structured and machine-understandable. DBpedia [10] is one of the projects that extracts Wikipedia information in more than 100 languages to build a structured multilingual and a multi-domain knowledge base.

Every Wikipedia article is transformed into a DBpedia resource, annotated with a set of properties extracted from the article page. Every resource is represented with unique Uniform Resource Identifier (URI), to avoid ambiguity. Resources URIs are identified with the URI <http://dbpedia.org/resource/>.

Listing 1. "Mobile medical service described with MobiSO using Turtle syntax"

```

1  @prefix mobiso: <http://www.semanticweb.org/MobiSO#> .
2  @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
3  @prefix price: <http://www.ontologydesignpatterns.org/cp/owl/price.owl#>.
4  @prefix time: <http://www.w3.org/2006/time#>.
5  @prefix location: <http://www.cenitmio.es/ontologies/Location.owl#>.
6  @prefix space: <http://pervasive.semanticweb.org/ont/2004/06/space#>.
7  @prefix geo-measurement:
8  <http://pervasive.semanticweb.org/ont/2004/06/geo-measurement#>.
9
10 mobiso:Service1 a mobiso:ConcreteService ;
11 mobiso:identifiedBy mobiso:ServiceIdentification1;
12 mobiso:hasContextInformation mobiso:contextInfo1 ;
13 mobiso:hasQoSInformation mobiso:QoSInfo1.
14
15 mobiso:ServiceIdentification1 a mobiso:ServiceIdentification ;
16 foaf:name "Medical service".
17
18 mobiso:contextInfo1 a mobiso:ContextInformation ;
19 location:hasLocation mobiso:nrLocationRelation1 ;
20 mobiso:preferredLanguage "Ar"^^xsd:language,"En"^^xsd:language.
21
22 mobiso:nrLocationRelation1 a location:NaryRelationHasLocation ;
23 location:obtainedByMethod location:GPS ;
24 space:hasCoordinates mobiso:location1.
25
26 mobiso:location1 a geo-measurement:LocationCoordinates;
27 geo-measurement:latitude "36.838429"^^xsd:double ;
28 geo-measurement:longitude "6.642822"^^xsd:double .
29
30 mobiso:QoSInfo1 a mobiso:QoSInformation ;
31 mobiso:hasServiceProvidingDays time:Sunday, time:Thursday ;
32 mobiso:hasPrice mobiso:price1.
33
34 mobiso:price1 a price:Price;
35 price:hasValue "250.0"^^xsd:float;
36 price:hasCurrency mobiso:DZD.

```

Wikipedia categories are also extracted and transformed into a semantically related categorization mechanism organized by SKOS [27] (Simple Knowledge Organization System) and used to classify DBpedia resources. All extracted information is added to the knowledge base as RDF triples. DBpedia resources are already interlinked to more than thirty external datasets for various purposes making it a central nucleolus for the Web of Data. For instance, to improve resources classification quality, YAGO2 [11], UMBEL² and WordNet [28] have been introduced as typing schemas.

DBpedia project maintains also a community driven effort to develop a consistent ontology based on Wikipedia. The resulting ontology consists of 529 classes, 927 object properties, 1290 datatype properties and 116 specialized datatype properties.

DBpedia knowledge base is reachable via different access mechanisms: Linked Data principles [29] which relies on HTTP and URL to retrieve resource descriptions, SPARQL endpoints which allows the execution of SPARQL queries over DBpedia knowledge base, or by downloading datasets sliced into different parts. More details about DBpedia project can be found in [10].

In our work, we propose to use DBpedia instead of single-domain ontologies to annotate mobile service descrip-

²<http://umbel.org/>

tions in order to develop an effective and universal semantic service discovery solution. We are driven by the following motivations:

- DBpedia covers multiple domains containing a large number of concepts and resources that allow annotations of a wide range of services in different domains. Each resource is referred to with a unique URI which allows a non-ambiguous services annotation. In addition, each resource contains textual description (titles and abstracts) which helps to choose the right service annotation.
- Contributors are constantly updating Wikipedia, either by adding new information or by modifying existing contents. This will have an impact on DBpedia as well since it depends on Wikipedia, and thus new resources and domains are covered. Furthermore, DBpedia communities are actively working to improve DBpedia data quality.
- DBpedia resources are classified within different classification schemas, such as DBpedia Ontology, YAGO2 [11], UMBEL, FreeBase.com, WordNet, etc. These hand-crafted ontologies provide a high precision knowledge that could be used to effectively annotate and search/match mobile services from different domains.
- DBpedia categories are organized and semantically related by SKOS vocabulary. They could be used to semantically arrange mobile services for effective retrieval.
- DBpedia extracts facts from Wikipedia, which covers content in more than 100 languages. Thus, DBpedia is a multilingual knowledge base that we can use to develop a discovery algorithm supporting different languages.
- A set of tools are built around DBpedia knowledge base. DBpedia Spotlight [30] is an automatic configurable annotation system for text documents that detects selected DBpedia resources in a given text. We reckon that using this system to guide service providers in a semi-automatic or even full-automatic semantic annotation process of their services will lead to increased simplicity and flexibility of service provisioning. On the other hand, DBpedia lookup³ is another tool that provides resources related to a specific keywords. This tool helps user to specify their request in terms of keywords (which will be converted into corresponding resources) instead of diving into semantic languages details.
- Two kinds of ontologies exist already for semantic annotations: heavyweight and lightweight ontologies. Heavyweight ontologies are generally used to model complex and knowledge-intensive domains such as biology or astronomy. Lightweight ontologies on the other hand are used for data integration or semantic search. DBpedia provides the second type of ontologies which usually lack expressiveness, may contain errors and induce omissions. However, they could fit specific applications such as semantic search where lack of expressiveness is tolerated for the benefit of scalability [31].
- Due to performance and network communication evolutions, mobile user could not only act as service consumers but also as service providers [32]. To ensure a user-centric services consuming and provisioning, we believe that using lightweight ontologies and especially those provided by DBpedia, will enable this perspective, because users are more familiar with encyclopedic topics and thus they would use it to annotate or discover services.

Listing 2 shows a set of mobile services annotated using DBpedia resources, categories and YAGO concepts as well. External references in service definitions describe generic functionality of the service, while capabilities describe specific functionalities. Service1 is a concrete service that provides a certain type of biopsy available in a given medical clinic. The service is annotated with two DBpedia resources: *Clinic* and *Biopsy*, with their associated categories: *Biopsy*, *Clinics*, *Types_of_healthcare_facilities* Service2 is also a concrete service that offers certain type of investigating facilities (medical tests) available in a given hospital. It is annotated with two DBpedia resources : *Hospital* and *Medical_test* in addition to one YAGO concept *MedicalTests* and set of categories. Service3 is a software service that informs about sandwich items that are available in a given grocery store and their available quantity. It is

³<http://wiki.dbpedia.org/Lookup>

described by *Grocery_store* DBpedia resource and its associated categories which describe main provided functionality of the service. Service specific functionalities are described with two capabilities (sandwich and quantity).

Listing 2. "mobile services descriptions annotated with DBpedia"

```

1  @prefix mobiso: <http://www.semanticweb.org/MobiS0#> .
2  @prefix category: <http://dbpedia.org/resource/Category:> .
3  @prefix dbpedia: <http://dbpedia.org/resource/> .
4  @prefix yago: <http://dbpedia.org/class/yago/> .
5  @prefix mioService: <http://www.cenitmio.es/ontologies/Service.owl#>.
6  mobiso:Service1 a mobiso:ConcreteService;
7  mobiso:resourceReference dbpedia:Clinic,dbpedia:Biopsy;
8  mobiso:categoryReference category:Biopsy,category:Clinics,
9  category:Types_of_healthcare_facilities,
10 category:Pathology,
11 category:Surgical_procedures.
12 mobiso:Service2 a mobiso:ConcreteService;
13 mobiso:resourceReference dbpedia:Hospital,dbpedia:Medical_test;
14 mobiso:conceptReference yago:MedicalTests;
15 mobiso:categoryReference category:Medical_tests,
16 category:Pathology,
17 category:Hospitals,
18 category:Types_of_healthcare_facilities.
19
20 mobiso:Service3 a mobiso:SoftwareService;
21 mobiso:resourceReference dbpedia:Grocery_store;
22 mobiso:categoryReference category:Food_retailers,
23 category:Supermarkets,
24 category:Grocers,
25 category:Shops,
26 category:Sales_occupations;
27 mobiso:hasCapability mobiso:cap1, mobiso:cap2.
28
29 mobiso:cap1 a mobiso:capability;
30 mobiso:name "Sandwich";
31 mobiso:resourceReference dbpedia:Sandwich.
32 mobiso:cap2 a mobiso:capability;
33 mobiso:name "Quantity";
34 mobiso:resourceReference dbpedia:Quantity.

```

Based on these DBpedia annotations, we have developed a semantic matchmaking algorithm which is discussed in Section 7.

6. Non-monotonic Semantic rules for Mobile Service Discovery

W3C in its mission to structure Web content and transform it into a Web of Data has achieved a considerable progress by standardizing a Resource Description Language (RDF) to describe Web content and RDFS/OWL to knowledge driven by this content in specific domains.

In order to balance expressive power and computation complexity, ontologies were introduced in the semantic Web basically as restricted fragments of first order logic. To overcome this limitation and enhance representation capabilities that enable powerful inferences on top of formal ontologies [33], semantic rules were introduced.

Rules in general have the form if conditions, then conclusion defined in two parts: antecedents and consequents. If all conditions in the antecedent clause are determined to be true, then all conclusions in the consequent clause are implied or applied. The following example illustrates how the relation Uncle is represented with rules.

$$brother(X, Y), childOf(Z, Y) \rightarrow uncle(X, Z)$$

Figure 2. Mobile services discovery architecture

Two kind of logic rules exist: monotonic and non-monotonic. Monotonic rules consider that conclusions remain valid even when the knowledge base is updated with additional information, which is not the case for non-monotonic rules. Non-monotonic rules are based on non-monotonic logic [34] that allows reasoning with incomplete and conflicting information.

By combining DATALOG with parts of OWL, a semantic Rules definition Language (SWRL) was proposed to enable the description of Horn-Like rules extending syntax and semantics of OWL [35]. However, SWRL remains monotonic and has been extended in [34] to support non-monotonicity and allow the definition of more expressive rules. This work empowers SWRL with four new operators: Not operator to express negative facts; NotExists quantifier to ask for missing facts in the knowledge base (when used in the antecedent of the rule) and remove facts (when used in the consequent); Dominance operator to establish priorities among rules; and Mutex operator to establish exclusions during rule executions.

In our work, we propose to use non-monotonic semantic rules for the following reasons:

- Rules reason on top of ontologies; new knowledge (in our case context and QoS information) is being inferred.
- Mobility and dynamicity of mobile users in mobile environments may lead to incomplete and/or ambiguous context information generally gathered from sensors [36]. Non-monotonic rules have the ability to deal with this situation where incomplete knowledge may raise inconsistencies and conflicts.
- There are numerous potential service applications of non-monotonicity including personalization, brokering and bargaining [37] allowing users to express alternative choices to select most appropriate services thanks to the priority ranking of rules.

7. Proposed Semantic discovery framework

In order to perform effective semantic mobile services discovery based on DBpedia and non-monotonic rules, we have proposed the following architecture (illustrated in Figure 2). It is divided into three different parts, service provisioning, service request and service matchmaking, where each part is composed from a set of different components, the role of each one is described next.

7.1. Services Provisioning Manager

This component guides services providers in the service provisioning process. Services could be described with textual description (in case of concrete services) or with WSDL and OWL-S formats (in case of software services). For the second case this component attempts to extract useful phrases or keywords that best describe the services. Textual description or extracted phrases/keywords are then passed to DBpedia Spotlight which returns their corresponding candidate annotations (in form of DBpedia resources), so the service provider chooses appropriate annotations that fit his service description. After that, the component get all related information of selected annotations such as concepts and categories hierarchy from DBpedia knowledge base and store them into a local store (DBpedia cache).

7.2. Request Handler

To ensure user-centric search approach and to overcome the problem of limited input capabilities of mobile devices, a user specifies his queries by typing keywords, where each keyword describes a desired service functionality or capability. Then, he chooses and edits semantic rules that specify his context and QoS requirements. A mobile application guides the user to accomplish the above task. The mobile application gets also the last fresh context information from user device to generate appropriate user request which is sent to the the "Request Handler" component. This component is responsible of parsing user request to extract keywords, in addition to rank rules and fresh contextual information in order to propagate each extracted information to the appropriate component before starting the discovery process.

7.3. Keyword/Resource Translator

This component receives extracted keywords from "Request Handler" and translates them into a corresponding DBpedia resources using DBpedia Lookup service or using a simple SPARQL query on DBpedia cached data. Keywords can be specified in user language before translated into equivalent English-based DBpedia resources. An additional language processing task may be performed (for example, by consulting dictionaries to get keyword synonyms), if no corresponding resource was found.

Algorithm 1: Semantic matching of Mobile services function

Input: A set DBpedia resources R each of which represents a user desired capability

Output: set of matched services $S' \subseteq S$

```

1 Function Match( $R$ )
2   foreach  $s \in S$  do
3      $scores = 0$ 
4     foreach  $r \in R$  do
5        $scores += similarity(r,s)$ 
6     end
7     if  $scores > 0$  then
8        $s.semanticWeight += Log(1 + scores)$ 
9        $S' = S' \cup \{s\}$ 
10  end
11 return  $S'$ 

```

Algorithm 2: similarity function

Input : DBpedia resource r and a service s
Output: a similarity *score* indicating semantic service annotations and a resource
Data: A is the set of annotation resources of s
 T is the set of annotation concepts of s
 Δ is the set of annotation categories of s
 ℓ : number of categories hierarchy levels should be retrieved

```

1 Function Similarity( $r, s$ )
2  $A = \text{get\_annotation\_resources}(s)$ 
3  $score = 0$ 
4 if  $r \in A$  then
5    $score = 1$ 
6 else
7    $C = \text{get\_concepts\_of\_resource}(r)$ 
8    $T = \text{get\_annotation\_concepts}(s)$ 
9    $S_{concepts} = \text{Tversky}(C, T)$ 
10   $\beta = \text{get\_categories\_of}(r, \ell)$ 
11   $\Delta = \text{get\_annotation\_categories}(s)$ 
12   $S_{categories} = \text{Tversky}(\beta, \Delta)$ 
13   $score = \text{AVG}(S_{concepts}, S_{categories})$ 
14 return  $score$ 

```

7.4. Services Matchmaker component

The "Services Matchmaker" component receives user query in the form of resources from Keyword/Resource Translator component and attempts to match the query to find semantically related mobile services. Algorithm 1 illustrates an abstract matching process. It acquires as input a list DBpedia resources describing desired capabilities, in order to output a list of semantically matched services. Each matched service has a semantic weight value indicating its semantic relevance to the user request.

Algorithm 1 starts by iterating through all available services in order to calculate their semantic weight (using *similarity* function). If returned score from *similarity* function is positive, we apply a log to the final score to alleviate the score value.

Similarity function (algorithm 2) is called to find degree of similarity between a DBpedia resource r and the annotations of the service s based on Tversky model [?] as follows:

- First, all service annotation resources are retrieved (using *get_annotation_resources()* function). If the resource r belongs to the set of the annotation resources of s , then the score will be 1.
- If the service is not annotated with the resource r , we use its concepts and categories to calculate the score based on Tversky model. After retrieving resource r concepts and using *get_concepts_of_resource* and services concepts using *get_annotation_concepts* respectively, first score is calculated using Tversky function (algorithm 3). Tversky function calculates the similarity score between two list of features based on the commonalities and differences of compared lists. With the same manner, a second similarity score between resource r categories and services categories is calculated.
- The final score is obtained then by computing the average between concepts matching score and categories matching score.

We indicate that super or sub concepts or categories of a given resource at a given level are included in the same level to have the same importance during matchmaking process.

Algorithm 3: Tversky function

Input : Two lists of features F_1 and F_2
Output: a similarity value sim between 0 and 1

- 1 **Function** Tversky(r, s)
- 2 $C = \text{commonElements}(F_1, F_2)$
- 3 $U_{F_1} = \text{uniqueElements}(F_1, F_2)$
- 4 $U_{F_2} = \text{uniqueElements}(F_2, F_1)$
- 5 $sim = \frac{|C|}{|C| + |U_{F_1}| + |U_{F_2}|}$
- 6 **return** sim

7.5. Services Ranking component

“Services Matchmaker” component produces a list of relevance services that matches user needs. However, some of these services may not satisfy user context and QoS requirements. Therefore, we propose to rank matched services based on non-monotonic rules (introduced in section 6) describing alternative choices of services based on context and QoS information. This component uses a rules reasoner engine that provides monotonic and non-monotonic rules inference service. It uses also context and QoS information manager to get required user context and QoS information values involved in the ranking rules. So, this component starts by extracting required information from context and QoS database before it executes user-provided rules. A set of predefined ranking rules could be offered to the user (for example rules that privilege nearest services), so he has the ability to customize and edit them to fit his needs. It is also possible to allow users to exchange their rules created using adequate user interface.

Let us illustrate the following rules describing a desired user ranking choice. Consider three rules $r1$, $r2$ and $r3$ where $r1 > r2$, $r1 > r3$, i.e. the first rule have high priority over $r2$ and $r3$ ($r1$ must be executed before $r2$ and $r3$). The rule $r1$ eliminates any service has price greater than 300, so rule $r2$ and $r3$ will be fired only on services satisfying $r1$. Rule $r2$ gives two points rank to any service has price between 70 and 150, while $r3$ gives only one point to services having price greater than 150.

$$\begin{aligned}
 r1 : & \text{Service}(s), \text{hasQoS Information}(s, \text{QoS Info}), \text{hasPrice}(\text{QoS Info}, \text{price}), \text{price} > 300 \rightarrow \text{Eliminated}(s, \text{true}) \\
 r2 : & \text{Service}(s), \text{Eliminated}(s, \text{false}), \text{hasQoS Information}(s, \text{QoS Info}), \text{hasPrice}(\text{QoS Info}, \text{price}), \text{price} > \\
 & \quad 70, \text{price} < 150 \rightarrow \text{Rank}(s, 2) \\
 r3 : & \text{Service}(s), \text{Eliminated}(s, \text{false}), \text{hasQoS Information}(s, \text{QoS Info}), \text{hasPrice}(\text{QoS Info}, \text{price}), \text{price} > 150 \rightarrow \\
 & \quad \text{Rank}(s, 1)
 \end{aligned}$$

It is worth mentioning that rules could be used to automatically eliminate services that impose mandatory requirements. For instance, if a user’s device does not support NFC (Near field communication) and a particular service requires this feature, it should be removed from resulting ranked service list.

7.6. Rules reasoner engine

Rules reasoner component provides semantic reasoning services to facilitate rules execution for ranking component, context and QoS manager. It deduces and returns new context and/or QoS information by executing received semantic monotonic/non-monotonic rules.

7.7. Context and QoS manager

Context and QoS manager maintains users and services contextual and quality of service information. It collects, stores and updates context information in a database in order to be retrieved as needed. In fact, this information is stored into an RDF store since we have used ontologies models. This component is able to use rules engine to reason upon context and QoS information in order to infer new knowledge. For example, if a service defines specific providing days, context and QoS manager can decide that this service is not available out of defined range, so it could be ignored during discovery process.

8. Implementation and evaluation

We have implemented our proposed framework focusing on the discovery process. It is worth noting that service provisioning and multilingual discovery are out of the scope of this research. Each component of the framework was created as separate module. Jena ⁴ is the core library for the framework. It is used to request Information about resources, categories and concepts via DBpedia SPARQL endpoints ⁵. Service descriptions, context and QoS information are stored in a local Jena TDB RDF store, and retrieved using Jena ARQ and SPARQL 1.1. Jena has a generic rules engine that we have used in conjunction with JNOMO ⁶ which extends Jena to support non-monotonic rules execution.

In the following of this section, we describe used test collection and how services are annotated. Then, we present comparative experimental results shedding the light on how our framework improves discovery effectiveness of mobile services. We conclude this section with performance analysis.

8.1. Semantic matchmaking effectiveness

8.1.1. Test collection and Semantic Services Annotation

To evaluate semantic matchmaking effectiveness, we rely on data provided by OWL-S TC ⁷ version 4. OWL-S TC includes a set of 1083 OWL-S 1.1 semantic services from nine different domains (education, medical care, food, travel, communication, economy, weapons, geography and simulation). Also, it provides 42 test queries associated with binary and graded relevance sets to conduct performance evaluation experiments. In addition, we collect a part of our test collection set from online public repositories such as <http://www.webapifinder.com/> and <http://www.programmableweb.com>.

Our collection set is made from 170 services, 123 services have been transformed from the OWL-S TC collection and 47 are accumulated from the public repositories. Services are classified in the categories: food, medical care, travel, communication, economy, emailing and question/answer.

We annotate services by passing each service textual description to DBpedia Spotlight that returns annotation candidates for the service. Then, we pick out the best candidates that we judge relevant to the service description. In some annotation cases, DBpedia Spotlight is incapable to retrieve the appropriate annotation resources. So, we try to get missed annotations by searching on DBpedia knowledge base.

During the annotation process we noticed that DBpedia is able to annotate almost all services. However, some concepts are missing for the annotation; for example *PREPAREDFOOD* concept do not have a corresponding resource in DBpedia. On the other hand, there were cases where annotation of OWL-S TC services could be improved; for instance in *food_price_AnimalFoodservice.owl*s service description, food is used to annotate the input, where it could be annotated by *Animal.feed* resource from DBpedia.

Figure 3 depicts statistics about DBpedia references used to describe services semantically. It is obvious that categories are dominating references since almost every resource is classified into one or more categories. We noticed that concepts are less used compared to resources and categories because not all resources are typed. The majority of concepts are from Yago and DBpedia ontology (dbpedia-owl) as showing Figure 4.

8.1.2. Discovery Effectiveness criteria

To evaluate the accuracy of our discovery framework, we use precision and recall. Precision is the ratio of the number of relevant services retrieved to the total number of retrieved services. Recall is defined as the ratio of the number of relevant services retrieved to the total number of relevant services in the services directory. Their mathematical definitions are as follows :

$$Precision = \frac{|\{relevant\ retrieved\ services\}|}{|\{retrieved\ services\}|}$$

$$Recall = \frac{|\{relevant\ retrieved\ services\}|}{|\{all\ relevant\ services\}|}$$

Figure 3. Statistics about used references

Figure 4. Statistics about used concepts

In order to evaluate returned ranked services and make comparison between different retrieval systems, 11-point precision-recall curve is used. This graph plots the interpolated precision of an information retrieval (IR) system at 11 standard recall levels. If we have a system that contains n services, we calculate precision and recall for each top k retrieved service, where $k \in [1, n]$. Then, we plot interpolated precision for each recall r level where $r \in \{0.0, 0.1, 0.2, \dots, 1.0\}$. The interpolated precision P_{interp} at recall level r is defined as the highest precision for any recall level $r' \geq r$:

$$P_{interp}(r) = \max_{r' \geq r} p(r') \quad (1)$$

⁴<http://www.jena.apache.org>

⁵<http://dbpedia.org/sparql>

⁶<http://sourceforge.net/projects/jnomo/>

⁷<http://projects.semwebcentral.org/projects/owls-tc/>

Figure 5. 11-point curve for 24 semantic and keyword based queries

8.1.3. Evaluation Results

To measure services retrieval effectiveness, we have calculated interpolated precision of 24 distinct request over the 170 annotated services taking into consideration the top 20 result ($k = 20$). We execute each request for three different category levels. Then, we compare results with keyword-based search performed by Apache Lucene⁸ tool which is an open source project that provides a Java based high-performance, full-featured text search engine library. Apache Lucene uses TF-IDF (Term Frequency-Inverse Document Frequency) which is scoring model used as weighting factor to calculate documents relevance regarding a user query.

We rely on OWL-S TC binary relevance set to classify services as relevant or not relevant, or we judge relevancy by asking whether we would be able to directly use the service to obtain at least part of the information or functionality that the request is interested in.

Figure 5 illustrates obtained results plotted in 11-point curve. It shows that semantic search with different categories have a higher result over keyword-based search. On other hand, using more categories hierarchy enable to get more relevant services. We have performed semantic search for maximum three levels, however, we think that incorporating more levels may affect negatively results relevancy because higher levels may include more generic or specific categorization concepts.

Despite semantic search gives an overall high relevant values, we found in some queries that keyword search gives higher relevance due to the miss typing or to the bad classification of some DBpedia resources.

Experiments show that proposed framework is able to retrieve semantically related services. For instance, when a user searches for a hospital service, other semantically closed services such as clinic and care organization services appear in the search results which is not the case for keyword based search. We noticed also that different synonyms have a unique DBpedia resources thanks to Wikipedia Redirects, this will have impact on the annotation process as well as on the discovery, for example: *Publishes*, *Published*, *Publishing_firm*, *Publication_company*... all redirects to the same DBpedia resource *Publishing*.

8.2. Performance

In this subsection, we present performance evaluation results for services matchmaking and rules execution. Experiment were conducted on a machine equipped with Ubuntu 14.04 64 bit operating system, Java 7, Intel i7 2 GHz x

⁸<http://lucene.apache.org/>

Figure 6. Matchmaking execution time using SPARQL queries

Figure 7. Matchmaking execution time using indexed services

4 processor and 8 GB RAM.

8.2.1. Matchmaking scalability

The experiments were done using Web services collection presented above. For each DBpedia resource used to annotate services, we store locally all their related categories and concepts including their hierarchies. Stored data contains 50795 categories and 783 concepts.

At first, we implement matchmaking algorithm using SPARQL queries. Figure 6 shows measurement results. The analysis of the results reveals that matchmaking services is time consuming process (from 6 to 29 seconds depending on number of used resources). Obtaining categories and concepts of a particular resource takes a reasonable time (from 5 ms to 78 ms).

Thus, indexing services and parallelizing execution are crucial to ensure scalability. Using SPARQL queries we get categories and concepts of a particular resource, then we match them to indexed services. The index strategy shows a considerable gain in terms of execution time compared to SPARQL-based matchmaking, as it is shown in Figure 7 (a query that contains eight resources will take roughly 250ms).

Figure 8. Non-monotonic rules execution time

8.2.2. Non-monotonic context rules execution performance

To evaluate performance execution of non-monotonic rules describing users alternative choices, we have executed a set of 10 rules containing more than 50 clauses. We ensure that 90% of the rules should be fired. Figure 8 shows that rules execution time is acceptable since number of candidate services for ranking process should be decreased by semantic services matchmaking component.

9. Conclusion and future work

Services discovery is an essential aspect to correlate both services providers and users. In mobile environments, developing an effective service discovery becomes a more challenging issue; while context and QoS information may provide distinct results. To respond to this requirements, we have proposed in this paper, an extended ontology called MOBISO used to describe context and QoS information of mobile services. To overcome keyword-based search limitations, a semantic discovery framework based on DBpedia, a multi-domain multilingual knowledge base, and on non-monotonic rules have been proposed. Our discovery framework exploits mobile services descriptions annotated with DBpedia resource, concepts and categories to retrieve semantically related services each of which ordered by a semantic weight value. Then, it applies a ranking process based on contextual and QoS information defined through non-monotonic rules. Evaluation results of our discovery approach shows that services search effectiveness is considerably ameliorated compared to keyword-based search. Besides, semantic rules provided a flexible mechanism allowing users to filter and rank services according to their contextual and QoS needs. Our solution focuses on user-centricity; trying to take user away from semantic technologies details thanks to DBpedia and its tools such as DBpedia Lookup and DBpedia Spotlight.

As future work, we plan to deploy our framework to the cloud as it provides necessary computation resource to ensure scalable and reliable services discovery especially for semantic-based approaches. Moreover, we are convinced that combining syntactic search with semantic search will improve our proposed semantic services search. We plan also to use more linked data to improve typing mechanism of DBpedia resources since some of them lack of adequate concepts. We believe that including more annotation sources by mining Linked-Data, which is a rich source of related and connected information, will improve discovery effectiveness.

References

- [1] Portio Research Mobile Factbook 2013.
URL <http://www.portioresearch.com/media/3986/PortioResearchMobileFactbook2013.pdf>
- [2] T. A. Farrag, A. I. Saleh, H. A. Ali, Semantic web services matchmaking: Semantic distance-based approach, *Computers & Electrical Engineering* 39 (2) (2013) 497–511.

- [3] H.-L. Truong, S. Dustdar, A survey on context-aware web service systems, *International Journal of Web Information Systems* 5 (1) (2009) 5–31. doi:10.1108/17440080910947295.
- [4] S. B. Mokhtar, D. Preuveneers, N. Georgantas, V. Issarny, Y. Berbers, EASY: Efficient semAntic Service discoverY in pervasive computing environments with QoS and context support, *Journal of Systems and Software* 81 (5) (2007) 785–808. doi:10.1016/j.jss.2007.07.030.
- [5] I. Braun, A. Strunk, G. Stoyanova, B. Buder, ConQo—A Context-And QoS-Aware Service Discovery, in: *Proceedings of the IADIS International Conference WWWInternet 2008*, 2008, pp. 432–436.
- [6] M. Adacal, A. Bener, Mobile Web services: a new agent-based framework, *IEEE Internet Computing* 10 (3) (2006) 58–65. doi:10.1109/MIC.2006.59.
- [7] S. K. Chii Chang, Sea Ling, ProMWS: Proactive mobile Web service provision using context-awareness, in: *8th IEEE International Workshop on Managing Ubiquitous Communications and Services*, IEEE, 2011, pp. 69–74.
- [8] A. Toninelli, A. Corradi, R. Montanari, Semantic-based discovery to support mobile context-aware service access, *Computer Communications* 31 (5) (2008) 935–949. doi:10.1016/j.comcom.2007.12.026.
- [9] R. Niazi, Q. H. Mahmoud, An Ontology-Based Framework for Discovering Mobile Services, in: *2009 Seventh Annual Communication Networks and Services Research Conference*, IEEE, 2009, pp. 178–184. doi:10.1109/CNSR.2009.36.
- [10] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, C. Bizer, DBpedia - A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia, *Semantic Web* 6 (2) (2014) 167–195.
- [11] J. Hoffart, F. M. Suchanek, K. Berberich, G. Weikum, YAGO2: A Spatially and Temporally Enhanced Knowledge Base from Wikipedia, *Artificial Intelligence* 194 (2013) 28–61.
- [12] N. Cheniki, B. Abdelkader, Y. Atif, Mobile Services Discovery Using Multi-domain Ontologies and Non-monotonic Rules, in: *International Conference on Next Generation Computing and Communication Technologies (ICNGCCT)*, Dubai, 2014, pp. 119–124.
- [13] M. Paolucci, T. Kawamura, T. Payne, K. Sycara, Semantic matching of web services capabilities, in: *The Semantic Web ISWC 2002*, Vol. 2342 of *Lecture Notes in Computer Science*, Springer, 2002, pp. 333–347. doi:10.1007/3-540-48005-6_26.
- [14] G. Klyne, J. Carroll, Resource description framework (RDF): Concepts and abstract syntax, W3C recommendation, W3C (Feb 2004). URL <http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/>
- [15] R. Guha, D. Brickley, RDF vocabulary description language 1.0: RDF schema, W3C recommendation, W3C (Feb. 2004). URL <http://www.w3.org/TR/2004/REC-rdf-schema-20040210/>
- [16] F. van Harmelen, D. McGuinness, OWL web ontology language overview, W3C recommendation, W3C (Feb. 2004). URL <http://www.w3.org/TR/2004/REC-owl-features-20040210/>
- [17] M. Butler, C. Woodrow, J. Hjelm, L. Tran, F. Reynolds, H. Ohto, G. Klyne, Composite capability/preference profiles (CC/pp): Structure and vocabularies 1.0, W3C recommendation, W3C (Jan. 2004). URL <http://www.w3.org/TR/2004/REC-CCPP-struct-vocab-20040115/>
- [18] User Agent Profile Specification. URL <http://www.wapforum.org/what/technical/SPEC-UAPProf-19991110.pdf>
- [19] T. Strang, C. Linnhoff-Popien, K. Frank, CoOL: A context ontology language to enable contextual interoperability, in: *Distributed applications and interoperable systems*, Springer, 2003, pp. 236–247.
- [20] M. Poveda Villalon, M. C. Suárez-Figueroa, R. García-Castro, A. Gómez-Pérez, A context ontology for mobile environments, in: *Proceedings of Workshop on Context, Information and Ontologies - CIAO 2010 Co-located with EKAW 2010*, 2010.
- [21] D. Preuveneers, J. V. D. Bergh, D. Wagelaar, A. Georges, P. Rigole, T. Clerckx, E. Berbers, K. Coninx, K. D. Bosschere, Towards an extensible context ontology for ambient intelligence, in: *Proceedings of the Second European Symposium on Ambient Intelligence*, Springer, 2004, pp. 148–159.
- [22] L.-H. Vu, M. Hauswirth, F. Porto, K. Aberer, A search engine for QoS-enabled discovery of semantic web services, *International Journal of Business Process Integration and Management* 1 (4) (2006) 244–255.
- [23] N. A. Saadon, R. Mohamad, WSMO-M: NFP-aware web service discovery for mobile computing, in: *2011 Malaysian Conference in Software Engineering*, IEEE, 2011, pp. 70–75. doi:10.1109/MySEC.2011.6140646.
- [24] S. B. Mokhtar, D. Fournier, N. Georgantas, V. Issarny, Context-aware service composition in pervasive computing environments, in: *Rapid Integration of Software Engineering Techniques*, Springer, 2006, pp. 129–144.
- [25] J. Kopecky, T. Vitvar, C. Bournez, J. Farrell, Sawsdl: Semantic annotations for wsdl and xml schema, *Internet Computing*, IEEE 11 (6) (2007) 60–67.
- [26] T. Berners-Lee, J. Hendler, O. Lassila, The semantic web, *Scientific American* 284 (5) (2001) 28–37.
- [27] A. Miles, B. Matthews, M. Wilson, D. Brickley, Skos core: Simple knowledge organisation for the web, in: *Proceedings of the 2005 International Conference on Dublin Core and Metadata Applications: Vocabularies in Practice*, DCMI '05, Dublin Core Metadata Initiative, 2005, pp. 1–9.
- [28] G. A. Miller, WordNet: a lexical database for English, *Communications of the ACM* 38 (11) (1995) 39–41.
- [29] C. Bizer, T. Heath, T. Berners-Lee, Linked data-the story so far, *International Journal on Semantic Web and Information Systems (IJSWIS)* 5 (3) (2009) 1–22.
- [30] P. N. Mendes, M. Jakob, A. García-Silva, C. Bizer, DBpedia Spotlight: shedding light on the web of documents, in: *Proceedings of the 7th International Conference on Semantic Systems*, ACM, 2011, pp. 1–8.
- [31] Jim Hendler, Introduction to the Semantic Web. URL http://videlectures.net/iswc08_hendler_ittsw/
- [32] K. Elgazzar, P. Martin, H. Hassanein, Mobile Web Services: State of the Art and Challenges, *International Journal of Advanced Computer Science and Applications (IJACSA)* 5 (3) (2014) 173–188. doi:http://dx.doi.org/10.14569/IJACSA.2014.050324#sthash.3sDf2f14.dpuf.
- [33] M. Dean, Semantic web rules: Covering the use cases, in: *Rules and Rule Markup Languages for the Semantic Web*, Vol. 3323 of *Lecture Notes in Computer Science*, Springer, 2004, pp. 1–5. doi:10.1007/978-3-540-30504-0_1.
- [34] D. Nute, Defeasible logic, in: D. M. Gabbay, C. J. Hogger, J. A. Robinson (Eds.), *Handbook of Logic in Artificial Intelligence and Logic*

- Programming (Vol. 3), Oxford University Press, Inc., New York, NY, USA, 1994, pp. 353–395.
- [35] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosz, M. Dean, SWRL: A semantic web rule language combining OWL and RuleML, Member submission, W3C (May 2004).
URL <http://www.w3.org/TR/2004/REC-xml-20040204>
- [36] T. Strang, C. Linnhoff-Popien, A context modeling survey, in: Workshop on Advanced Context Modelling, Reasoning and Management, UbiComp - The Sixth International Conference on Ubiquitous Computing, Nottingham/England, 2004.
- [37] G. Antoniou, A. Bikakis, G. Wagner, A system for nonmonotonic rules on the web, in: Rules and Rule Markup Languages for the Semantic Web, Vol. 3323 of Lecture Notes in Computer Science, Springer, 2004, pp. 23–36. doi:10.1007/978-3-540-30504-0_3.