CLEI ELECTRONIC JOURNAL, VOLUME 20, NUMBER 2, PAPER 1, AUGUST 2017

Lightweight Semantic based Approach to Assess Readability of Web Service Interfaces*

Alan De Renzis, Martin Garriga, Andres Flores, Alejandra Cechich
GIISCO Research Group, National University of Comahue
{alanderenzis,martin.garriga,andres.flores,alejandra.cechich}@fi.uncoma.edu.ar

and

Cristian Mateos, Alejandro Zunino
ISISTAN Research Institute, UNICEN University
{cristian.mateos,alejandro.zunino}@isistan.unicen.edu.ar

Abstract

A Web Service has an interface described in a machine-processable format (specifically WSDL). Service providers expose their services by publishing their WSDL documents. Service consumers can learn about service capabilities and how to interact with services. Service descriptions (WSDL documents) should be ideally understood easily by service stakeholders so that the process of consuming services is simplified. In this work we present a practical metric to quantify readability in WSDL documents. We adapted and extended an existing ontology-based semantic readability metric to focus on WSDL documents by using WordNet as a lightweight concept hierarchy. We have validated our approach by performing both qualitative and quantitative experiments. The first one consists of a controlled survey with a group of service consumers. The results showed that consumers (software engineers) required less time and effort to analyze WSDL documents with higher readability values. The second experiment compares our approach with two ontology-based approaches. The third experiment compares the readability values of a dataset of real-life service descriptions before and after rewriting them. The results showed the effectiveness of our approach to assess readability of Web Services interfaces.

Keywords: Web Services, Readability, Service Interface, Service description quality, WordNet.

1 Introduction

The Service-Oriented Computing (SOC) paradigm has experienced an ever-increasing adoption for building distributed, inter-organizational applications in heterogeneous environments [1]. Mostly, the software industry has adopted SOC by using Web Service technologies. A Web Service is a program with a well-defined interface that can be published, located and invoked by using standard Web protocols [2].

Typically, service interfaces and communication protocols are represented in the form of WSDL[1] (Web Services Description Language) documents. WSDL is an XML-based language for specifying service descriptions. A WSDL document is defined by a service provider, published in a service registry, and then used by service consumers both to figure out service capabilities and interact with the specified Web Service. The goal is that service consumers can reason about the functionality of a Web Service without knowing service implementation details and by only looking at its associated WSDL document [3]. Intuitively, this can be achieved provided the document is readable/understandable, i.e., it does not suffer from specification problems that might obscure the purpose of the service to consumers. Moreover, when such documents are published in a registry, services must be discovered prior to be consumed. The process of discovering Web Services requires automatically matching a query against potential candidate services in a registry, and then

*This work is an extended version of work published at CLEI'16 – DOI:10.1109/CLEI.2016.7833369
[1]`http://www.w3.org/TR/wsdl`

manually inspecting resulting candidates to select the desired result. For this process to be effective, service providers must suply meaningful WSDL documents upon developing services [4].

Particularly, from a linguistic point of view, readability is defined as *"the level of ease or difficulty with which text material can be understood by a particular reader who is reading that text for a specific purpose"* [5]. In the context of Web Services, service descriptions (WSDL documents) should be ideally understood easily by service stakeholders so that the process of consuming services is simplified. Then, producing readable service descriptions can be used as a strategy by service providers to attract service consumers [6]. Current research in the field is indeed compatible with the idea of generating readable service descriptions, since they address other relevant quality attributes such as maintainability [7], discoverability [8] and adaptability [9].

However, quantifying readability and hence providing a metric broadly applicable for WSDL documents is still a major challenge. Previous work in the topic [6, 10] requires a service domain ontology – structured as a concept hierarchy modeled via languages such as OWL[2]/RDF[3] – to compute readability. Ontologies aim to describe all relevant aspects related to general services which are accessible through a Web Service interface, with the ultimate goal of enabling the (total or partial) automation of the Web Service lifecycle – e.g., discovery, composition, monitoring, etc. [11]. However, domain-specific ontologies are often avoided in real life applications because they are difficult to specify [12]. Thereby, the lack of complete and relevant domain-specific ontologies hinders the applicability of ontology-based approaches in practice [13, 14].

Therefore, the main contribution of this paper is to propose a practical metric to quantify readability in WSDL documents. Following with our previous work in the field [15], we adapt and extend the semantic readability metric proposed in [10] to focus on WSDL documents by using WordNet [16] as the underlying concept hierarchy. WordNet is a domain-independant lexical database of the English language. Thus, a domain-specific, "heavy" ontology for each domain is not necessary. The readability metric is applied over services descriptions – truly accessible to service consumers – and does not consider their implementation code – certainly unavailable.

To validate the proposed readability metric, we performed qualitative and quantitative experiments. First, a controlled survey with a group of human service consumers. They were asked to analyze a dataset of WSDL documents to determine whether or not they were readable in general terms – i.e., without considering our metric. The survey results were compared with the readability values obtained by calculating the readability metric. The results showed that software engineers required less time and effort to analyze WSDL documents with higher readability values. The second experiment compared the proposed readability metric with two ontology-based WSDL readability metrics. The goal was to observe the effectiveness of our lightweight WordNet-based approach with regard to heavyweight ontology-based approaches. Lightweight approaches can be considered simplistic and practical while they may be less effective to deal with domain-specific aspects, regarding heavyweight approaches [17]. The comparison was performed using the available experimental results of the two ontology-based approaches. Obtained results showed that our readability values are consistent with that of the other approaches. This suggests that our metric is suitable to assess quality of WSDL documents in readability terms. The third experiment compares readability values of a dataset of real-life WSDL documents from the industry before and after modifying them by applying well-known best programming practices. Results show a readability improvement for rewritten WSDL documents.

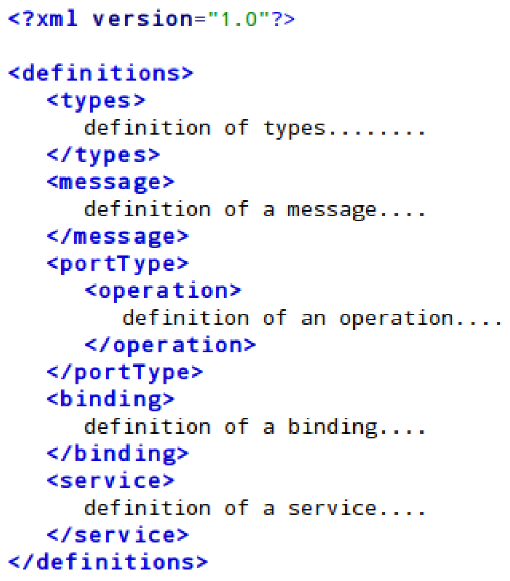
The rest of this paper is organized as follows. Section 2 gives some background on readability concepts. Section 3 introduces the Web Services centered readability metric proposed in this work. Section 4 presents the experiment in the form of a survey. Section 5 presents the second experiment which compares existing readability metrics with regard to our metric. Section 6 details the third experiment which compares the readability values of a dataset of real-life WSDL documents from the industry before and after rewriting them. Section 7 discusses relevant related work. Conclusions and future work are presented afterwards.

2 Background

A service development life-cycle, as any other regular kind of software component, consists of several phases. Within these, the service design phase comprises service interface specification using WSDL. Several important concerns should influence design decisions to result in good service interface designs [18]. Many of the problems related to the efficiency of standard-compliant approaches to Web Service discovery stem from the fact that the WSDL specification is incorrectly or partially exploited by providers [19].

[2]Web Ontology Language. `https://www.w3.org/2001/sw/wiki/OWL`

[3]Resource Description Framework. `https://www.w3.org/RDF/`



```
<?xml version="1.0"?>

<definitions>
   <types>
      definition of types........
   </types>
   <message>
      definition of a message....
   </message>
   <portType>
      <operation>
         definition of an operation....
      </operation>
   </portType>
   <binding>
      definition of a binding....
   </binding>
   <service>
      definition of a service....
   </service>
</definitions>
```

Figure 1: WSDL Structure

2.1 Web Service Description Language (WSDL)

When employing Web Services, a provider describes each service technical contract, a.k.a. its interface, in WSDL. WSDL is an XML-based language designed for specifying service functionality as a set of abstract operations with inputs and outputs, and to associate binding information so that consumers can invoke the offered operations. WSDL allows providers to describe two parts of a service: what it does (its functionality) and how to invoke it. The first part reveals the service interface that is offered to consumers, while the second part specifies technological aspects, such as transport protocols and network addresses. Consumers use the functional descriptions to match third-party services to their needs, and the technological details to invoke the selected service [20].

Figure 1 shows the structure of a WSDL document. With WSDL, service functionality is described as one or more port-type that arranges different operations that exchange input and return messages. Main WSDL elements, such as *porttypes*, operations and messages, must be labelled with unique names. Optionally, these WSDL elements might contain documentation as comments. Messages consist of parts that transport data between consumers and providers of services, and vice-versa. Exchanged data is represented using XML according to specific data-type definitions in XML Schema Definition (XSD), a language to define the structure of an XML element. XSD offers constructors to define simple data-types (e.g., `integer` and `string`), restrictions, and both encapsulation and extension mechanisms to define complex elements. XSD code might be included in a WSDL document using the *types* element, but alternatively it might be put into a separate file and imported from the WSDL document or even other WSDL documents afterward.

2.2 Plain-text oriented Readability

The original readability model from [10], which is based upon "heavy" ontologies, computes two main features of a plain-text document, namely document *scope* and document *cohesion*, according to the presence/absence of domain terms in a document. The vocabulary of the domain is structured as a concept hierarchy, where the more specific the terms the deeper they appear in the hierarchy. The terms in the document which have a match in the concept hierarchy are regarded as *domain concepts*, otherwise they are *non-domain concepts*. The concept hierarchy that captures the domain terms can be a domain-specific ontology or taxonomy. The model also considers *difficulty* (of comprehension) of single words in a document.

The *scope* is defined as the coverage of domain-specific concepts in the document. The original model calculates the scope considering the *average depth* of all the concepts in the document. If the *average depth* is high the document scope will be low. Thus, the document will be less readable. Unfortunately, the original model does not provide any range or threshold to determine "high" and "low" *depth* values.

The *cohesion* feature refers to how much focused is the document on a particular topic. It can be computed as the shortest path among each pair of terms according to the given concept hierarchy. This reflects the semantic relation among the domain terms in the document. According to the model, the more cohesive the domain terms in the document, the more readable the document.

Finally, the *difficulty* feature is related with the comprehension of words in a document. It is calculated in the original model by using the *Dale-Chall's readability index* [21]. This formula indicates the percentage of *difficult* words in the document using a list of approximately 3,000 *familiar* (non-difficult) words.

Formula 1 shows the overall Concept-Based Readability score (CRS), which is calculated upon *scope*, *cohesion* and the *Dale-Chall's readability index* (DaCw).

$$CRS(d) = Scope(d) + Cohesion(d) + DaCw(d)^{-1} \tag{1}$$

$$Scope(d) = e^{-(\sum_{i=1}^{n} depth(c_i))} \tag{2}$$

$$Cohesion(d) = \frac{\sum_{i,j=1}^{n} sim(c_i, c_j)}{numberOfAssociations} \tag{3}$$

$$DaCw(d) = \% - Of - Difficult - Words \tag{4}$$

For more details of the concept-based readability score we refer interested readers to [10].

Example To illustrate the original readability model, next we describe a simple example involving a document d composed of three words: `compact`, `motorcar` and `truck`. We consider the concept hierarchy presented in Figure 2 .

- *Scope*: The depth of the word `compact` is 3, while the depth of `motorcar` and `truck` is 2. Therefore the scope is: $scope(d) = e^{-(3+2+2)} = 0.000911$.
- *Dale-Chall's Index*: The words `compact` and `truck` do not belong to the Dale-Chall's list, then the percentage of *difficult* words is 66%.
- *Cohesion*: the cohesion value is calculated as follows:

$$cohesion(d) = \frac{\sum_{i,j=1}^{3} sim(c_i, c_j)}{3}$$

where

$$sim(compact, motorcar) = 0.77$$

$$sim(compact, truck) = 0.3$$

$$sim(motorcar, truck) = 0.48$$

then

$$cohesion(d) = \frac{0.77 + 0.3 + 0.48}{3} = 0.52$$

- Readability value: finally, the readability value is calculated as follows:

$$CRS(d) = 0.000911 + 0.52 + 66^{-1} = 0.536$$

CRS applied to WSDL An existing approach in the context of readability for Web Services [6] has applied the notions defined in the original model defined by [10] upon WSDL documents in a straightforward way. In this approach, all words included in a WSDL document are analyzed to determine if they are domain terms with respect to a domain-specific ontology previously selected. However, from the analysis of the readability model, we conclude the following issues:

- The original readability model is strongly dependent of a concept hierarchy (in the form of a domain-specific ontology). However, it is known that the task of building (or discovering) a specific ontology for a domain requires a large effort even by skillfull developers [14, 13, 22]. This hinders the adoption of such readability model in practice.
- The original readability model targeted plain-text documents. Thus, it is unaware of the structural information included in a WSDL document. Structural information involves each comprising part of a WSDL document, namely operations, port types, messages, data type definitions, etc. As this information is known a-priori, it is possible and desirable to analyze it when calculating readability. Then, the overall comparison extent is limited to only compare certain subsets of relevant terms.

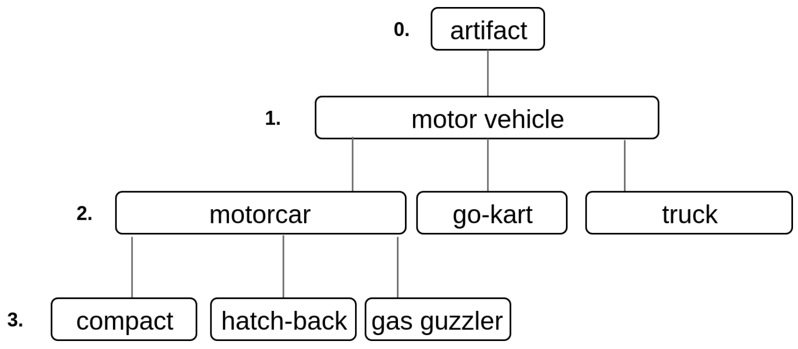


Figure 2: Vehicles hierarchy

3 Readability on Web Services

This section discusses the details of the readability metric for WSDL documents. When applied in the context of service discovery and inspection, the readability value enables the discovery of services which are more readable, further easing the adoption of Web Services in practice.

3.1 Web Service centered Readability

To overcome the issues mentioned in Section 2.2, we decided to adapt the original readability model to the context of Web Services, particularly for WSDL documents. In addition, we decided to use WordNet [16] as the underlying concept hierarchy. WordNet is a domain-independant lexical database of the English language that is structured as a lexical hierarchy – a tree with root node {`entity`}. Terms are grouped in synsets (synonym sets) that represent the same lexical concept. Several relationships connect different synsets, such as hyperonymy/hyponymy, holonymy/meronymy and antonymy. The hierarchy links more general synsets like {`furniture, piece_of_furniture`} to increasingly specific ones like {`bed`} and {`bunkbed`}. Thus, the category `furniture` includes `bed`, which in turn includes `bunkbed`; conversely, concepts like `bed` and `bunkbed` make up the category `furniture`. To calculate relationships between concepts, we use the JWNL Java library[4].

In our approach, every term included in a WSDL document is analyzed with the help of WordNet. The original model (described in Section 2.2) defines domain and non-domain terms. In our Web Service centered model, for a given WSDL document, those terms which belong to the WordNet dictionary are considered as *existing terms* (equivalent to domain terms in the original model), thus they are used as input to calculate the readability value; otherwise they are *non-existing terms* (i.e., non-domain terms in the original model). As stated, using WordNet as the underlying concept hierarchy enables considering terms belonging to different domains for the readability calculation. We considered that any domain-specific term that is not recognized by WordNet is a non-existing term.

Each operation in a WSDL document is individually analyzed to calculate its readability value. Then, the average value for all operations in a WSDL document constitutes the overall readability value of the service. This value represents the difficulty with which a WSDL document can be understood by a service consumer. In other words, this value represents the difficulty for a human reader to understand the only specification available for a given Web Service.

3.2 Concept hierarchy selection

Domain independency was the main feature that we expected from a concept hierarchy, thus extending the metric application range. Many helpful alternatives are available, as WordNet, BabelNet [23], SUMO [24], DOLCE [25], DISCO [26] and Schema.org [27]. Main features of each concept hierarchy are described as follows.

WordNet [16] is the most widely used lexical database of the English language. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked by means of conceptual-semantic and lexical relations. The resulting network of meaningfully related words and concepts can be navigated.

BabelNet is a very large, wide-coverage multilingual semantic network. This approach integrates lexicographic and encyclopedic knowledge from WordNet and Wikipedia. Machine Translation is applied to enrich the resource with lexical information for all languages. BabelNet brings together the strengths of WordNet -

[4]`https://web.stanford.edu/class/cs276a/projects/docs/jwnl/javadoc/`

i.e., being highly structured and providing labeled lexico-semantic relations - with those of Wikipedia - i.e., providing large amounts of semantic relations, multilinguality and continuous collaborative updates [23].

SUMO (Suggested Upper Merged Ontology) is an upper level ontology that has been proposed by the IEEE Standard Upper Ontology Working Group, with collaborators from the fields of engineering, philosophy, and information science. SUMO provides definitions for general-purpose terms and acts as a foundation for more specific domain ontologies [28]. A mapping from WordNet synsets to SUMO has also been defined, showing that their coverage of English concepts is equivalent [24]. The SUMO ontology defines high-level concepts like Object, ContinousObject, Process, Quantity, Relation, and so on, providing axioms in first-order logic that describe properties of these concepts and relations among them.

DOLCE ontology is a formal foundational ontology developed as a top-level ontology in the WonderWeb project [29] (a finished project to develop the infrastructure required for the large-scale deployment of ontologies as the foundation for the Semantic Web). The goal of DOLCE is to provide a common reference framework for WonderWeb ontologies to facilitate sharing of information among them. Within its representation, DOLCE aims at capturing "ontological categories underlying natural language and human common-sense" [30]. In contrast with "lightweight" ontologies, often with a minimal terminological structure (e.g., a taxonomy) fitting the needs of a specific community, the main purpose of foundational ontologies is to negotiate meaning, either for enabling effective cooperation among multiple artificial agents, or for establishing consensus in a mixed society where artificial agents cooperate with human beings [31].

DISCO (DIStributed COllocations) is a pre-computed database of collocations and distributionally similar words. DISCO's Java library[5] allows retrieving the semantic similarity between arbitrary words. The similarities are based on the statistical analysis of very large text collections (e.g., Wikipedia), through co-occurrence functions. For each word, DISCO stores the first and second order vectors of related words using a Lucene[6] index. To analyze the similarity between two words, DISCO retrieves the corresponding word vectors from the index and computes the similarity according to a similarity formula based in co-occurrences [26].

Schema.org[7] is a collaborative community-driven set of schemas for structured data on the Internet. In 2011, the major search engines Bing, Google, and Yahoo (later joined by Yandex) created Schema.org to provide a single schema across a wide range of topics that included people, places, events, products, offers, and so on. The idea was to present webmasters with a single vocabulary. Over 10 million sites use Schema.org to markup their web pages and email messages, such as Google, Microsoft, Pinterest and others. The vocabularies are developed by an open community process. A shared vocabulary makes it easier for webmasters and developers to decide on a schema and get the maximum benefit for their efforts, by sharing the markup accross multiple consumers [27].

Using WordNet enables to exploit as many information as possible gathered from WSDL documents, without requiring any extra semantic specification for services, such as ontologies, usually unavailable in practice. As we early mentioned, WordNet is domain-independant, and its concept hierarchy can be used to assess WSDL documents coming from any domain category. In fact, current Web Service repositories such as Mashape.com and ProgrammableWeb.com organize published services in dozens of categories as diverse as Messaging, Advertising, Real Estate, Security and Financial, just to name a few. However, we consider that in a future work we could define another metric proposal based on any of the mentioned alternatives and to provide a helpful comparison between each other.

3.2.1 Concepts extraction

In the concepts extraction step, all terms from words and identifiers from WSDL documents are pre-processed and identified according to WordNet.

We use the Java API of the Membrane SOA model[8]. The extracted terms belong to identifiers included in specifics elements (parts) in WSDL documents: *Definition, Data types, Message, Operation, Port type, Port and Service.* A list of terms is extracted from each operation part – e.g., a list of message terms, a list of sequence terms, and a list of data type terms, among others. To extract terms from identifiers, we make use of an algorithm for semantic separation of terms defined in a previous work [32]. Identifiers are normally restricted to a sequence of one or more letters in ASCII code, numeric characters and underscores ("_") or hyphens ("-"). The algorithm supports the rules in Table 1 – i.e., the usual programming naming conventions. By using WordNet, a semantic level has been added to be aware of cases that do not follow those conventions. The algorithm analyzes identifiers, recognizing potential terms (uppercase sequences and

[5]`http://www.linguatools.de/disco/disco-download_en.html`
[6]`https://lucene.apache.org/core/`
[7]`http://schema.org/`
[8]`http://www.membrane-soa.org/soa-model-doc/1.4/java-api/parse-wsdl-java-api.htm`

Table 1: Rules for Decomposing Identifiers

Notation	Rule	Source	Result
JavaBeans	Splits when changing text case	`getZipCode`	`get Zip Code`
Special tokens	Splits when "_" or "-" occurs	`get_quote`	`get quote`

lowercase sequences). Then, WordNet is used to analyze all the potential terms and determine the most adequate term separation.

All possible terms from a WSDL document are mapped to concepts in the WordNet hierarchy. Those terms that do not belong to the WordNet dictionary are considered as *non-existing terms*. The outcome of this step is a collection including the lists of terms of all WSDL elements for each operation. This structured collection is crucial to calculate the Document Cohesion feature, which is explained latter in this section.

3.2.2 Document Scope

The original readability model (Section 2.2) defines document scope feature as the coverage of domain concepts in a document. As the number of domain-specific concepts in a document increases, the readability of the document decreases. However, we have redefined this feature for the context of Web Services, in particular considering WSDL documents, since:

- Using specific terms in WSDL documents reduces ambiguity about service functionality and makes them self-explanatory [19]. Thus, we consider that WSDL documents with more domain-specific terms are more readable.
- As discussed in Section 2.2, the approach in [6] applies the readability formula from the original model directly to WSDL documents. However, the obtained values for document scope in their case studies are too small, very close to zero (for example 2.31^{E-92}).

All in all, we redefined the *scope* as the specificity degree of the concepts in a WSDL document. The deeper the concepts of a document appear in the WordNet hierarchy, the more readable the document is, since the concepts will be more specific and ambiguity will be less likely [33]. The scope of a document d is computed according to Formula 5, as the average depth of all concepts in the document divided by the maximum depth of the concept hierarchy – i.e., 16 in the case of WordNet. Scope values range between 0 and 1.

$$Scope(d) = \frac{\frac{\sum_{i=1}^{n} depth(t_i)}{n}}{MaxTreeDepth} \tag{5}$$

where $depth(t_i)$ is the depth of concept t_i extracted from the document d w.r.t. the WordNet hierarchy, $MaxTreeDepth$ is the maximum depth of the WordNet hierarchy, and n is the total amount of existing terms.

Example Figure 3 shows an excerpt of the WordNet hierarchy where, for example, the depth of the concept `enclosure` is 4, and the depth of the concept `birdcage` is 6. These values indicate that the concept `birdcage` is more specific than the concept `enclosure`, and then the use of `birdcage` might lower the chance of a misinterpretation.

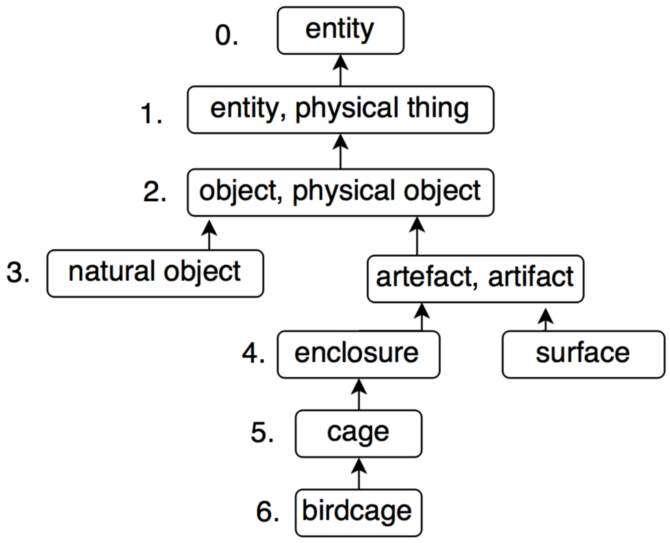


Figure 3: WordNet hierarchy example

3.2.3 Document Cohesion

Document Cohesion refers to how much focused is a document on a particular topic. Similarly to document scope, we redefined *cohesion* for the Web Services context. The original readability model analyzes the relations among each pair of concepts included in a plain-text document. Since a WSDL document is a structured document, we decided to exploit such structure. For this, we defined *Document Cohesion* as the average operation cohesion considering each operation included in the WSDL document. Operation cohesion is computed considering the semantic relationship between the existing terms in an operation signature, which is reflected by the shortest path among such terms in the WordNet concept hierarchy. The signature of an operation includes operation name, input, output and data-types in its definition.

Operation Cohesion is calculated according to Formula 6. The *opCohesion* value increases as the *length* between the concepts decreases. Essentially, the more cohesive the domain terms in the document, the more readable the document.

$$opCohesion(O_i) = \frac{\sum_{i,j=1}^{n} sim(t_i, t_j)}{n} \tag{6}$$

with n>1, i<j, and

$$sim(t_i, t_j) = -log \frac{length(t_i, t_j)}{2 * MaxTreeDepth} \tag{7}$$

where $length(t_i, t_j)$ is shortest path between t_i, t_j in the WordNet hierarchy, $MaxTreeDepth$ is the maximum tree depth (16).

Finally, document cohesion is calculated according to Formula 8:

$$cohesion(d) = \frac{\sum_{i=1}^{n} opCohesion(o_i)}{n} \tag{8}$$

where n is the number of operations in document d. Cohesion values range between 0 and 5.

Example Let us consider a WSDL document from the Car Rental domain. Figure 4 shows an excerpt of the WordNet hierarchy containing concepts from such domain. The *length* between the concepts `compact` and `truck` is 3, and the length between `compact` and `motor vehicle` is 2. These values indicate that `compact` is more similar to `motor vehicle` than to `truck`. Thus, the joint use of the domain concepts `compact` and `motor vehicle` in an operation name or an operation comment, will be more cohesive than the joint use of `compact` and `truck`.

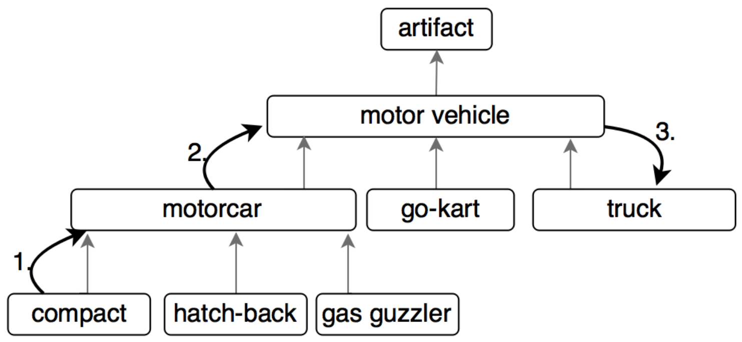


Figure 4: Length between `compact` and `truck` in the WordNet hierarchy

3.2.4 Simplified Dale-Chall's Readability Index

The simplified Dale-Chall's readability index is a formula that considers the percentage of difficult words in a document [21]. The rationale behind this index is that the length of the sentences in a document and the difficulty of words are correlated to the overall difficulty of reading and understanding the document. Since the concept-based readability model measures readability at word level, sentence-level complexity is not applicable and hence only word difficulty is considered. Words in the document are identified as either *familiar* or *unfamiliar* (difficult) words. That is, they are familiar words if they can be found in the Dale-Chall's list, which contains approximately 3,000 familiar words. Otherwise, they are regarded as unfamiliar or difficult words. The Dale-Chall's readability index of a document d can be computed according to Formula 9, and ranges from 0 to 1.

$$DaCw(d) = \frac{100 - PDW}{100} \tag{9}$$

where PDW is the percentage of difficult words among the terms extracted from d.

3.2.5 Web Service Readability metric

The Web Service Centered Readability (WSCR) metric for a WSDL document d determines the readability score by considering the three features explained above: the document scope, the document cohesion and the simplified Dale-Chall's index. Readability is computed according to Formula 10. This value ranges between 0 and 7, representing the worst and best WSCR value respectively.

$$WSCR(d) = Scope(d) + Cohesion(d) + DaCw(d) \tag{10}$$

Example Let us consider a simple WSDL document for the apartment rental domain with one operation, as shown in Figure 5. The first step for the readability calculation is *concepts extraction*:

- operation terms: [`reserve`, `apartment`, `breakfast`]
- message terms: [`reserve`, `apartment`, `breakfast`, `request`]
- type terms: [`apartment`, `breakfasts`, `meal`, `id`, `date`, `housing`, `description`, `rooms`, `number`]

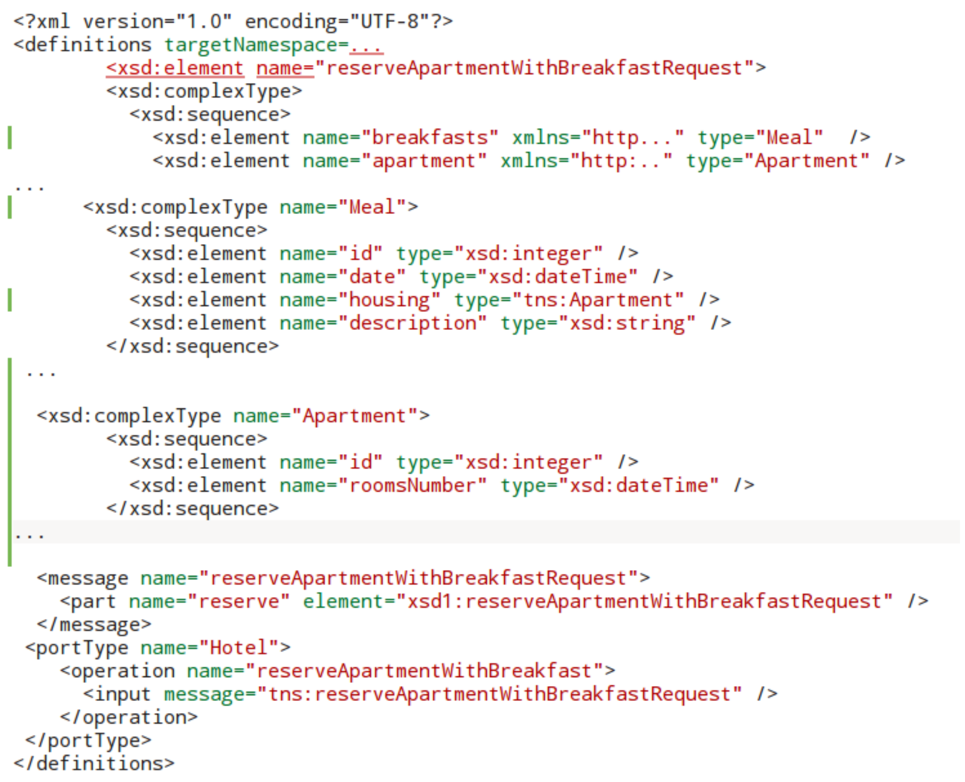
To calculate the *scope*, it is necessary to calculate the depth of each concept. For example:

$$depth(apartment) = 7$$

$$depth(breakfast) = 8$$

$$depth(room) = 5$$

Considering all the concepts in the operation definition, the total number of terms is 16. Then, the scope is calculated as follows:



```
<?xml version="1.0" encoding="UTF-8"?>
<definitions targetNamespace=...
        <xsd:element name="reserveApartmentWithBreakfastRequest">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="breakfasts" xmlns="http..." type="Meal"  />
            <xsd:element name="apartment" xmlns="http:.." type="Apartment" />
...
      <xsd:complexType name="Meal">
        <xsd:sequence>
          <xsd:element name="id" type="xsd:integer" />
          <xsd:element name="date" type="xsd:dateTime" />
          <xsd:element name="housing" type="tns:Apartment" />
          <xsd:element name="description" type="xsd:string" />
        </xsd:sequence>
 ...

  <xsd:complexType name="Apartment">
        <xsd:sequence>
          <xsd:element name="id" type="xsd:integer" />
          <xsd:element name="roomsNumber" type="xsd:dateTime" />
        </xsd:sequence>
...
  <message name="reserveApartmentWithBreakfastRequest">
    <part name="reserve" element="xsd1:reserveApartmentWithBreakfastRequest" />
  </message>
 <portType name="Hotel">
    <operation name="reserveApartmentWithBreakfast">
      <input message="tns:reserveApartmentWithBreakfastRequest" />
    </operation>
 </portType>
</definitions>
```

Figure 5: WSDL document for apartment rental domain

$$Scope(d) = \frac{\frac{5+6+...}{16}}{MaxTreeDepth} = \frac{7.18}{MaxTreeDepth} = 0.44$$

being $MaxTreeDepth = 16$.

On the other side, the *Simplified Dale-Chall's Readability Index*

$$DaCw(d) = \frac{100 - 18}{100} = 0.82$$

since an 18% of the terms are not included in the familiar words list.

Finally, the *cohesion* is calculated by comparing terms that are not in the same WSDL section. For example, the concept `room` from the list of type terms is compared with the concept `apartment` from the list of operation terms. Then,

$$length(room, apartment) = 4$$

given by the shortest path between `room` and `apartment` in the WordNet hierarchy. Then,

$$sim(room, apartment) = -log\frac{4}{2 * MaxTreeDepth}$$

where $MaxTreeDepth = 16$ is the maximum depth of the WordNet tree. Therefore,

$$sim(room, apartment) = 0.9$$

Considering the concepts `meal` and `breakfast`, $length(meal, breakfast) = 1$ since `meal` is an hyperonym of `breakfast`. Then, $sim(meal, breakfast) = 1.5$.

In the last step, the operation cohesion is calculated as:

$$cohesion(op) = \frac{1.5 + 0.9 + ...}{71} = 1.3$$

where 71 is the total number of concept relationships. Recall that the *document cohesion* is the average of operation cohesion values. In this case, it is 1.3 because the WSDL document is composed by only one operation.

Finally the *Web Service centered readability (WSCR)* is calculated as follows:

$$WSCR(d) = Scope(d) + Cohesion(d) + DaCw(d) =$$

$$0.44 + 1.3 + 0.82 = 2.56$$

WSCR range is between 0 and 7. We may initially think that 2.56 is a low value in readability terms. As we pointed out, 7 is the best value, however, to obtain this value it is necessary to simultaneously meet the following conditions:

- All WSDL term depths have to be 16.
- The distances between terms have to be 0 – i.e., equal terms or synonyms.
- All terms have to belong to *Dale-Chall's* list.

These conditions are not possible in a real-life WSDL document. Therefore, a WSDL document with a readability value of 2.56 is actually readable, but it can certainly be improved.

4 Experiment 1: Survey Evaluation

In the first experiment, we have conducted a survey to validate the Web Services centered readability model presented in Section 3. This survey involved 6 groups of participants from the IT sector. Participants included professionals, practitioners, teachers and students. Each participant received WSDL documents from different domains, including WSDLs with either low or high readability values according to our metric. Participants were not concerned with the readability concept prior to take the survey. Then, we measured time and effort required to analyze the functionality contained in the operations defined by the WSDL document. Following, we detail the preparation, distribution and results of the experiment.

Hypothesis Let n be WSDL documents that implement similar functionalities. A service consumer will easily analyze (in terms of time and effort) the self-contained functionality in those WSDL documents with higher readability values.

4.1 Survey preparation

Dataset The dataset consisted of 6 original WSDL documents that belong to two domains:

- Domain 1 – eCommerce. Services in this domain provide functionality to manage customers and their purchase orders – i.e., operations to add customers, add orders or associate orders to customers.
- Domain 2 – Hotel. Services in this domain provide functionality to manage hotel bookings – i.e., operations to book a room, associate breakfast to a room or list free rooms.

Each document comprises three operations and their corresponding data-types, which are specific to each domain. Moreover, the original WSDL documents presented medium readability values (between 2 and 3). We rewrote each document improving their readability values.

After rewriting documents, the readability value significantly increased for the rewritten WSDL documents: an increment of 43.6% for the eCommerce domain and 24.96% for the Hotel domain. Therefore, we came to expect an analogous reduction in the effort required by developers to understand the more readable versions of the employed WSDL documents.

Survey form design For each WSDL document (12 in total) we built a form, containing questions about both the functionality contained in the WSDL document and the required effort to understand such functionality, in terms of the analyzed parts of the WSDL. For each operation in the document, we included two questions:

- Select the correct functionality between three possible options. This question determines if the WSDL is descriptive enough to deduce the operation's functionality. For example, for the operation "*store*" from the eCommerce domain the options are:
  1. Stores an order corresponding to a Customer.
  2. Stores a Customer in the system.

3. Stores the system state when the operation is executed.

- Select the required effort in terms of the analyzed elements of the WSDL. Each option refers to how many parts (operation names, data-types, parameters, messages and so on) were analyzed to deduce the operation's functionality. Concretely, the three options (for every operation) are:
  1. Operation name and input/output messages only.
  2. Operation name, input/output messages and also parameters.
  3. Operation name, input/output messages, parameters and also data-types defined with their attributes.

We also saved the start and finish time to fill a form, to compute the elapsed time participants needed to complete each form. We considered this as an indicator of reading effort.

Form distribution In this survey, 54 persons participated as experimental subjects, where 20 were advanced IT students and 34 were teachers and professionals from the IT sector. We grouped the participants in 6 groups of 9 persons each. Then, we distributed the WSDL documents – as experimental objects – among the participants. We assigned 4 WSDL documents to each group: 2 original WSDL documents (with low readability values) of one domain, and 2 rewritten WSDL documents (with high readability values) of the other domain. Thus, the experimental objects have two possible values: original and rewritten. The goal of this distribution was to make a "blind" survey, where the participants did not know about "good and bad" WSDL documents, in readability terms. In addition, we did not introduce participants into readability concerns nor into features to improve readability. Formally, for the experimental design, we followed the guidelines from [34, 35, 36]:

- A *balanced factorial* experiment, where the same number n of observations are gathered for each experimental subject. In this case, the number of observations are the questions in each form. All participants answered the same number of questions – 6 per each of their 4 assigned WSDL documents (24 in total).
- *Confused interaction* in the groups, where only a portion of the WSDL's combinations is assigned to each group [36]. As mentioned earlier, we assigned to each participant 2 original WSDL documents (with low readability values) of one domain, and 2 rewritten WSDL documents (with high readability values) of the other domain. This avoids the experimental threat of the learning effect: when the participant changes from WSDL documents with low readability values to WSDL documents with high readability values, the domain switching prevents comparing with each other. Also, this allows to perform a randomized experiment with a relatively small experimental population, which is not completely randomizable otherwise.

We developed the forms using the Google forms platform, and then we distributed the forms and guidelines by e-mail. We hosted the WSDL files in an institutional server from our University to ensure their availability. Table 2 summarizes the list of WSDL files with their corresponding domain. Table 3 summarizes the distribution of forms and WSDL files per group.

4.2 Results

Two weeks after distributing the forms, we started to collect the results. We measured the results in terms of the following metrics:

Hit rate This metric collects the answers for the first question about operation's functionality. It assumes a binary value: success (when the operation's functionality is pointed out correctly) or failure (when it is not). Figure 6a shows the hit rate for the first question about operation functionality:

- The participants selected the correct operation functionality for the 94% of the rewritten WDSL documents.
- The participants selected the correct operation functionality for the 68% of the original WDSL documents.
- For the eCommerce domain, the improvement in the hit rate between original and refactored WSDL documents was of 32%.
- For the Hotel domain, the improvement in the hit rate between original and refactored WSDL documents was of 14%.

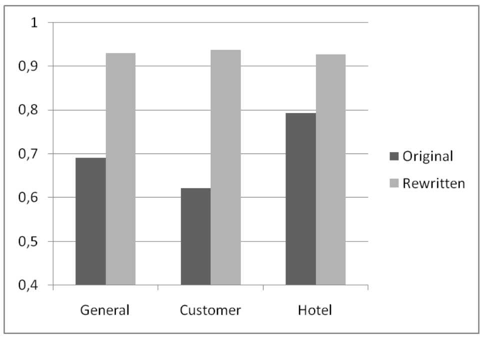
Table 2: WSDL files used in the experiment

Domain	**WSDL**	**Form**
Customer (original)	`CustomerSBO-1`	`goo.gl/forms/q2MXgTtFOu`
	`CustomerSBO-2`	`goo.gl/forms/JQh4yszcOk`
	`CustomerSBO-3`	`goo.gl/forms/4I6p5D8AXT`
Customer (rewritten)	`CustomerSBO-1`	`goo.gl/forms/oJAClS4xas`
	`CustomerSBO-2`	`goo.gl/forms/4rkTvGaKjX`
	`CustomerSBO-3`	`goo.gl/forms/twlG8WTSGp`
Hotel (original)	`Hotel-1`	`goo.gl/forms/ouNwjkAPkM`
	`Hotel-2`	`goo.gl/forms/R4SCyPFe9g`
	`Hotel-3`	`goo.gl/forms/igcdp9dYwd`
Hotel (rewritten)	`Hotel-1`	`goo.gl/forms/XTVeNoZhiQ`
	`Hotel-2`	`goo.gl/forms/7zAQF0tOI2`
	`Hotel-3`	`goo.gl/forms/TeTES3r9Ym`

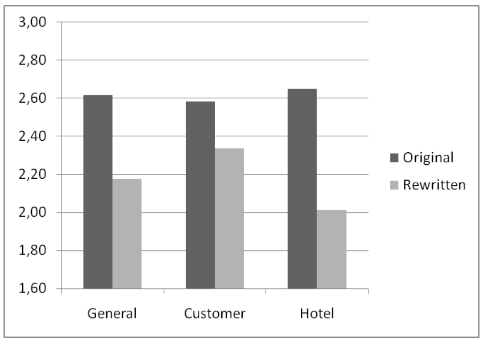
Table 3: Forms distribution per group

Group	**WSDLs**	**Group**	**WSDLs**
1	Customer-1-O	4	Customer-1-O
	Customer-2-O		Customer-3-O
	Hotel-1-R		Hotel-1-R
	Hotel-2-R		Hotel-3-R
2	Customer-1-R	5	Customer-2-O
	Customer-3-R		Customer-3-O
	Hotel-1-O		Hotel-2-R
	Hotel-3-O		Hotel-3-R
3	Customer-1-R	6	Customer-2-R
	Customer-2-R		Customer-3-R
	Hotel-1-O		Hotel-2-O
	Hotel-2-O		Hotel-3-O

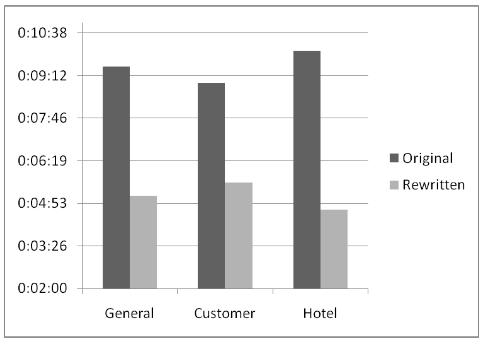
O: Original, R: Rewritten



(a) Hit Rate



(b) Average Effort



(c) Average time

Figure 6: Survey results for original and refactored WSDL documents

Effort This metric collects the answers for the second question about the effort of analyzing the WSDL document. It is an integer that is mapped to the following qualitative values, according to the selected option for the corresponding question:

1. Low effort. The participant only needed to analyze operation name and messages. Effort = 1.
2. Medium effort. The participant needed to analyze operation name, messages and parameters. Effort = 2.
3. High effort. The participant needed to analyze operation name, messages, parameters and complex data-types with their attributes. Effort = 3.

As stated earlier, each possible option of the second question was mapped to a numeric value according to required effort to deduce the operation functionality, ranging from 1 (low effort) to 3 (high effort). Figure 6b shows the average effort that the participants needed to deduce the operations functionality:

- The average effort for refactored WSDL documents was 2.15 (medium).
- The average effort for original WSDL documents was 2.65 (medium - high).
- The results present an effort reduction of 12% for the eCommerce domain and 32% percent for the Hotel domain.

Time This represents the elapsed time between starting and submitting each form. It assumes a continuous value using the format MM:SS (minutes and seconds). Previously, we indicated to the participants that the form should be completed without interruption.

Figure 6c presents the average time taken to analyze the WSDL documents and answer the forms. The results show that:

- The average time to analyze refactored WSDL documents was 5 minutes.
- The average time to analyze original WSDL documents was 9:30 minutes.
- The refactored WSDL documents presented a time reduction of 40% and 55% for eCommerce and Hotel domains, respectively.

4.3 Discussion

The experimental results presented in this section validate the experimental hypothesis, as participants required less time and effort to analyze WSDL documents with higher readability values. In particular, we can emphasize the aspects explained below.

Regarding the relationship between readability values and experimental results, Table 4 shows the readability improvement (percentage) for rewritten and original WSDL documents, and their correlation with the improvements reported in the experiment. We observed an overall readability improvement of 34% (considering both domains). This improvement relates to the results observed in hit rate, effort and time – which were improved in 24%, 21% and 46% respectively in the rewritten WSDL documents.

Table 4: Readability, hit rate, effort, and time improvement (%) for rewritten WSDL documents

	Readability (%)	**Hit rate (%)**	**Effort (%)**	**Time (%)**
Overall	34	24	21	46
Customer	44	33	11	38
Hotel	25	14	32	54

It is worth mentioning that a possible threat to the validity of an experiment involving people, such as the one performed, is Demand Characteristics [37]. Demand Characteristics result from cues in the experimental environment or procedure that lead participants to make inferences about the purpose of the experiment and to respond in accordance with (or in some cases, contrary to) the perceived purpose. Software engineers are inherently problem solvers. When they are told to perform a task, the majority will try to figure out what is expected from them and perform accordingly. Demand Characteristics influence a participant's perceptions of what is appropriate or expected and, hence, their behavior [36]. In the context of this experiment, the participants may figure out the notion of "good" and "bad" WSDLs (more and less readable documents). Thus, the participants will indicate a lower effort when analyzing readable WSDL documents. Since the experiment adopted Confused interaction in the groups, the learning was reduced and this implies that the deduction was indirectly reduced.

Another experimental threat is the language. Original and rewritten documents were implemented using concepts of the English language. However, even when all participants had some formal English knowledge, they were not native English speakers. Therefore, due to the lack of proficiency in the language, this could generate a slight noise to understand the WSDL documents and complete the survey.

5 Experiment 2: Comparison with other WSDL readability metrics

In the second experiment, we have compared the proposed readability metric with two ontology-based readability metrics proposed in [38, 6] for WSDL documents. The goal is to observe the effectiveness of our approach with respect to heavyweight ontology-based approaches. On this regards, lightweight approaches (e.g., WordNet-based) can be considered practical and less complex, but they may lack to effectively deal with domain-specific aspects [17]. For this experiment, we have used the available experimental results of the two ontology-based approaches.

5.1 Overview of ontology-based approaches

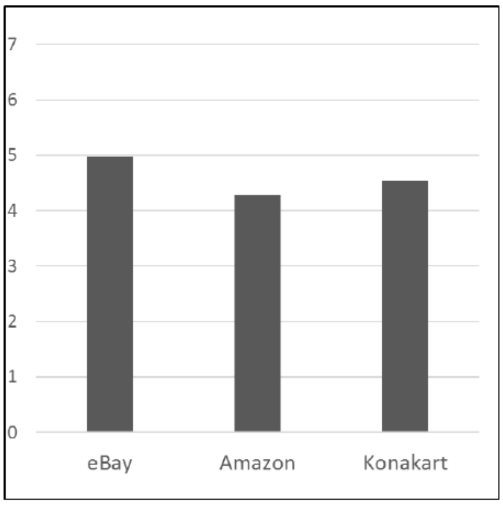
On the one hand, we considered the approach in [6], already presented in Section 2.2. In that work, the authors applied the notions from the model defined by [10] upon WSDL documents in a straightforward way. The main process consists in analyzing all words included in a WSDL with respect to a domain-specific ontology previously selected.

On the other hand, we considered the approach presented in [38]. This work is based on the approach above – i.e., in [6]. The authors extended the same process by using the WordNet dictionary as a complement to the domain-specific ontology. Similarly to the original process, after extraction of key concepts, they are mapped to the domain ontology. However, for many cases, extracted words do not map to any ontology concept. Hence, in this approach, synonyms of such words are retrieved from the WordNet dictionary. Those synonyms are intended to be mapped to the domain ontology. Finally, mapped synonyms are used as replacement to the non-mapped extracted words. Nevertheless, the original readability formula holds.

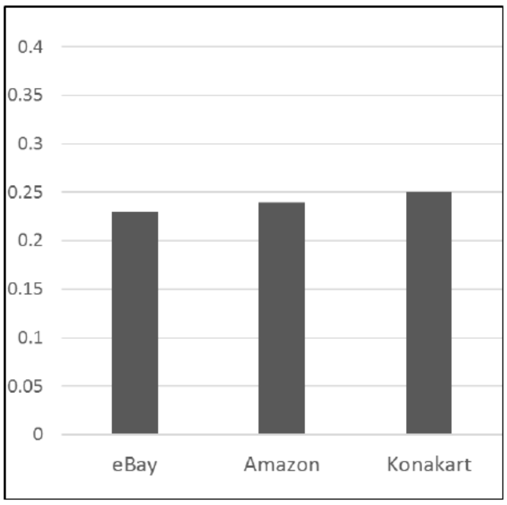
5.2 Execution and results

To compare our work with the readability approaches mentioned above, we analyzed the same WSDL documents evaluated in both approaches, which belong to the e-commerce domain: Konakart[9], Amazon[10] and eBay[11]. Regrettably, another WSDL also evaluated in those approaches is no longer available since the company (Click and Buy[12]) announced termination of commercial activity on 2016. Considering that both approaches are ontology-based, for the e-commerce domain they used the GoodRelations ontology[13]. GoodRelations is a standardized vocabulary for product, price, store, and company data. The latest version is available in RDF/XML format.

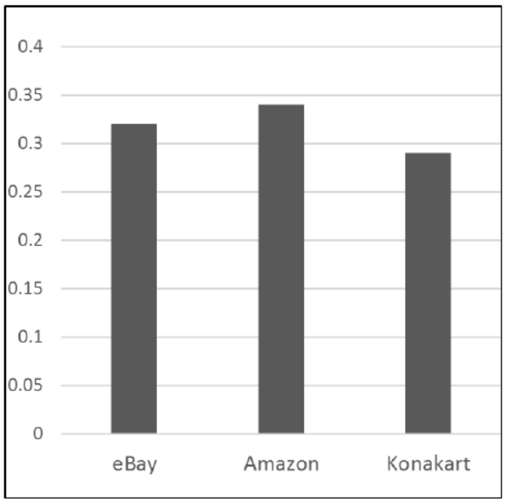
It is important to mention that the range of our readability formula is different from the range of the other approaches' formula. While our range is between 0 and 7, the range of the other formula is between 0 and 3. Then, readability results could not be directly compared. However, we analyzed the obtained results in terms of the sense given by each readability approach individually. Figures 7a, 7b and 7c show the obtained results from our approach, the ontology-based approach [6] and the extended ontology-based approach [38], respectively.



(a) Our approach



(b) Ontology-based approach [6]



(c) Extended ontology-based approach [38]

Figure 7: Readability values for e-commerce WSDL documents

The following aspects can be highlighted from the obtained results:

[9] `http://www.konakart.com/konakart/services/KKWebServiceEng?wsdl`
[10] `https://webservices.amazon.com/AWSECommerceService/AWSECommerceService.wsdl`
[11] `developer.ebay.com/webservices/latest/ebaySvc.wsdl`
[12] `https://www.clickandbuy.com/`
[13] http://www.heppnetz.de/projects/goodrelations/

- Our approach: readability values are between 4.2 and 5. We can observe that these values are similar in readability terms. Besides, as we mentioned in Section 3.2.5, these values are actually above the expected medium readability values.
- Ontology-based approach: readability values are between 0.23 and 0.25. We can observe that the WSDL documents are also very similar in readability terms, since the obtained values are very close to each other. However, the authors do not provide information about what they consider as high/low readability values.
- Extended ontology-based approach: readability values are between 0.29 and 0.32. We can observe that these values are very close to each other as well, with a slight difference with respect to the ontology-based approach. Again, the authors do not provide information about what they consider as high/low readability values.

5.3 Discussion

As we have pointed out in the last section, we can not directly compare values between the three approaches. Even though, we can observe that the readability values for WSDL documents of eBay, Amazon and Konakart kept an evident similarity in the context of each of the three approaches. According to our approach, those WSDL documents present a high quality in readability terms. In the other two approaches, those WSDL documents had the highest readability values with respect to Click and Buy service. Thereby, we may assume that for such approaches, these three WSDL documents are highly readable as well.

As a final interpretation of the obtained results, we emphasize that our approach is able to assess readability with similar precision than ontology-based approaches, even when we prescind from a domain-specific ontology. As we pointed out in Section 3.1, domain-specific ontologies are difficult to specify in order to be truly useful [12]. This is the case with the experiment in the extended ontology-based approach [38]. In that approach, WordNet is used to complement the lack of certain e-commerce concepts in the GoodRelations ontology.

6 Experiment 3: Document rewriting by expert developers.

The goal of the third experiment was to measure the Web Service centered readability of WSDL documents in comparative terms. We hypothesize a likely relationship between the Readability value and documents improved by experts.

6.1 Experiment configuration and execution

To evaluate the Web Service centered readability model we used a dataset of 84 WSDL documents of real-life Web Services extracted from [39]. Notice that this dateset is divided into 26 main categories (with severral subcategories) – e.g., business, communication, engieneering, flights. We have implemented a Java application to calculate the readability values for all WSDL documents in the dataset. Then, a group of experts rewrote each subset of the documents – with low readability values – by applying well known coding practices – e.g. those provided in [40, 41]. Finally, we compared the readability values from the original and rewritten documents.

Using as input the WSDL documents in the dataset, we executed the Web Service centered readability procedure. The results consist on the readability value $WSCR(d_i)$ for each WSDL document d_i – as defined in Section 3.1.

After calculating the readability values for the 84 documents, we selected a subset of 29 documents with the lowest readability values to be rewritten. The remaining have a relatively high readability value (greater than 4). It is not worthy to rewrite these documents because only a slight improvement could be obtained – demanding a large effort.

An academic group of Web Service experts with knowledge in the explained readability notions manually rewrote each document (from the selected subset) by applying well known coding practices. For each WSDL document in the subset, a unique rewritten document was produced – i.e., 29 rewritten WSDL documents. The only rule that we established, was not to affect the intended functionality offered by the services. Thereby, they were allowed to change the name of operations and descriptions of data, but without altering neither the structural aspect nor the semantic of the service elements. The rewriting criteria was free as it was not prescribed in any guideline. Then, we calculated the readability value for each rewritten WSDL document. Finally, we compared readability values of rewritten documents with respect to the value of their corresponding original documents. Assuming as hypothesis that the rewriting improves the readability, our

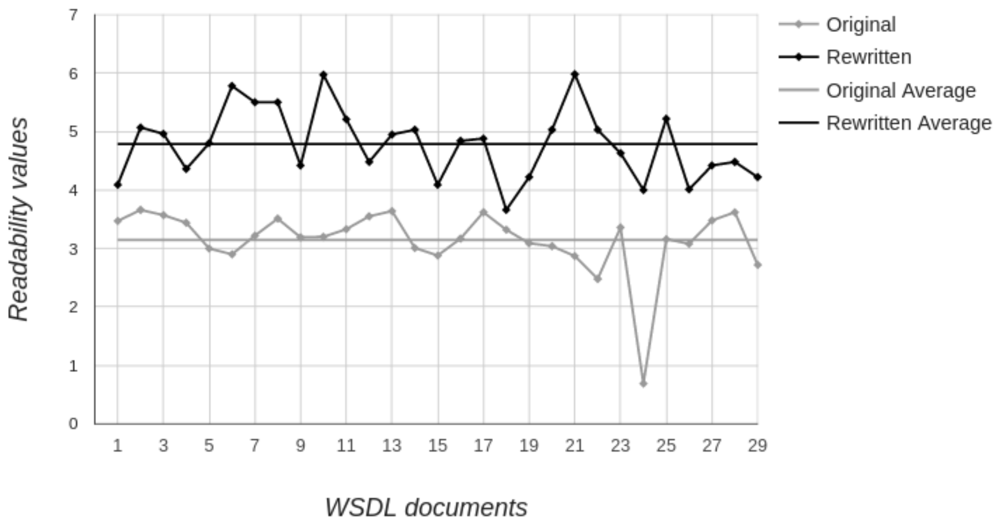


Figure 8: Readability values from original and rewritten WSDL documents (higher is better)

goal was to quantify these improvements over real-life documents. This provides a precise notion about the potential impact of the readability approach upon real-life WSDL documents.

6.2 Results and Discussion

Figure 8 shows the readability values from original and rewritten WSDL documents. The X axis represents the 29 WSDL documents and the Y represents the readability value for each version of the documents (original and rewritten). Documents were arranged without any specific order into the chart. The following aspects can be highlighted from the obtained results:

- Original documents presented an average readability value of 3.14 and a standard deviation of 0.56. It is worth noticing that original readability values in the considered dataset are high as the WSDL documents are relatively well-written. For example, the terms used in identifiers are well formed and contain existing words, but are not significant enough to infer the corresponding functionality for the operation.
- Rewritten documents presented an average value of 4.78 and a standard deviation of 0.61.
- All rewritten documents were improved in readability terms according to our formula.
- The improvements were not homogeneous. Some documents were highly improved compared to others.

According to the first experiment, it is most likely that documents with higher readability values are easier to read, thus demanding less time and effort to detect key functionality and to be analyzed. The results show an average readability improvement of 52% for rewritten WSDL documents. By drawing a parallel with the first experiment (survey), the results suggest that rewritten documents would require less time and effort to be analyzed by service consumers. In addition, the standard deviations for both versions (original and rewritten) did not present a significant change. Finally, the obtained improvements were not homogeneous. This is because the rewriting criteria was dependant of each expert/developer. Also, the specific domain of each service is a factor that really impacted in the document rewriting. This is because each developer has different experience and knowledge of each domain. Thus, for a developer it is easier to rewrite a document from a known domain. This is evidenced since the used dataset [39] comprises a wide range of domains, that as we early mentioned includes business, communication, engineering and flights among others.

7 Related work

Certainly, our work is related to a number of efforts that can be grouped into two broad classes. On the one hand, there is a substantial amount of research concerning to readability-oriented approaches applied to several areas. On the other hand, several approaches are focused on improving the quality of WSDL documents considering different attributes. In this sense, our approach is related to both kind of efforts since we share similar goals: proposing a readability metric and improving the quality of WSDL documents.

7.1 Readability-oriented approaches

In [42] the authors propose a bilingual (English and Chinese) assessment scheme for Web page and Web site readability based on their textual content, and conduct a series of experiments with real Web data to evaluate the scheme. The authors adopt two classical formulas for readability (one per language) defined in [43, 44]. In this approach the Web site readability is considered as an indicator of the overall difficulty level of a site, and it is defined considering the readability of the pages that compose it. Unlike our work, this approach neither include a semantics basis nor assesses the relations between the words that compose the document – the adopted readability formulas are merely syntactic-based.

The work in [10] (introduced in Section 2.2) presents a concept-based model for text readability. In addition to textual content of a document, the model considers how the domain-specific concepts contained in the document affect its readability. Using domain-specific concepts allowed the authors to define a readability formula developed for textual materials. However, as we pointed out in Section 3.1, the lack of complete and relevant domain-specific ontologies hinders the applicability of ontology-based approaches in practice [13, 14, 45]. For this reason, we adapted and extended the readability measure by means of WordNet [16] as the underlying concept hierarchy.

The work in [6] (also introduced in Section 2.2) applied the model from [10] upon WSDL documents in a straightforward way. In addition, the work in [38] (introduced in Section 5) extends the process defined in [6] by using the WordNet dictionary as a complement to the domain-specific ontology. Considering that Web Services are developed by third-parties, their descriptions do not necessarily include concepts from a domain-specific ontology. Thereby, for concepts that do not map in the domain ontology, synonyms of such words are retrieved from the WordNet dictionary. Then, those synonyms are mapped to the domain ontology, as replacement to the non-mapped extracted words. Even when the general process was extended, the authors did not change the original readability formula. For both ontology-based approaches in [6, 38] we may pointed out that their readability models are limited by being unaware of the structural information included in a WSDL document. Therefore, we decided to exploit this information when calculating readability, as it is available and known a-priori.

Lastly, in [46], the authors propose to measure document readability from two levels. First is the surface level readability that relates to the surface content. It can be assessed by a series of classical readability features. Beyond the surface content, a higher level, namely the topic level readability, reflects whether it is easy for a user to comprehend the hidden topics in documents. Thus, they propose a topic-based readability model, which can be used to enhance domain-specific Information Retrieval by considering both the surface and topic level readability of documents. In this work, we use as one of its basis the hypothesis that the overall lower taxonomy depths of identified topics in the specific taxonomy would indicate a better document readability. Then, [46] also employs the average tree depth of the identified topics to calculate the metric.

7.2 Service Descriptions assessment approaches

Several efforts address the problem of the quality of WSDL documents – which should not be confused with considering QoS of the corresponding Web Service. The work in [47] found a high correlation between well-known object-oriented metrics measured directly in the code implementing services, and the occurrences of "anti-patterns" in the corresponding WSDL specifications. Anti-patterns represent a set of indicators of poor quality service interfaces. The work proposes a practical approach of best practices to guide professional developers in obtaining better WSDL designs that align with the technologies and techniques for building services.

The work in [48] presents a model of semantic annotations for describing the Web services and the request and an algorithm which discovers and composes the Web services. In this approach, ontological engineering plays a leading role in adding semantics into a service description. As we early mentioned, it is known that the task of discovering a specific ontology for a domain requires a large effort even by skillfull developers [14, 13, 22] .

The work in [49] presents a metric for understandability of WSDL descriptions called *WSDL Understanding Degree* (WSDLUD). The authors define a criteria tree for each part of WSDL descriptions composed by the following characteristics: (i) Type Understanding Degree, (ii) Message Understanding Degree, (iii) Port Type Understanding Degree, (iv) Binding Understanding Degree and (v) Service Understanding Degree. The principal difference with our approach is that the WSDLUD metric does not compare correlation between words that should be related, such as words included in names of messages and their corresponding operations.

8 Conclusions and Future Work

Broadly, readability is defined as the level of ease or difficulty with which text material can be understood by a particular reader who is reading that text for a specific purpose [5]. In the context of Web Services, service descriptions should be understood with ease so that these descriptions can be used as a strategy by service providing organizations to attract service consumers [6], and simplify the process of finding and reusing external functionality from client applications.

Motivated by the problems of metrics designed for natural text documents to quantify Web Service readability, this paper presented a readability metric for WSDL documents. We propose the use of WordNet as the underlying concept hierarchy, in replacement of a domain-specific ontology for each possible domain.

As qualitative evaluation, we conducted a survey. The experiment has shown that it is easier, in terms of required time and effort, to analyze the self-contained functionality in the WSDL documents with higher readability values. Thus, such documents can be used by service providers to attract potential consumers effectively.

In addition, a quantitative experiment was performed. Our approach was compared with two ontology-based approaches [6, 38] that directly apply the original readability model in [10]. We observed that the obtained readability value ranges, according to the three approaches – for the same set of WSDL documents – are consistent. This evidences the effectiveness of our approach to assess readability of WSDL documents without depending on a domain-specific ontology.

Finally, in the last experiment we calculated readability values for WSDL documents in a public dataset. Then, a subset of documents was rewritten by professional coders and re-calculated readability values. The readability values from original and rewritten documents were compared, showing that the readability values increased in average 52% for the latter. According to the experiments, it is arguable that documents which acknowledge the readability notions, not only may have higher readability values but also may reduce the understandability effort by consumers, as shown by the different experiments in this work. When faced with little readable documents, consumers were forced to make a (likely) large effort on deducing the functional capabilities of such services. Therefore, the proposed readability model is potentially useful for its application in industry and real-life scenarios of service consumption.

As future work we are planning to exploit and integrate the previously mentioned semantic basis, DISCO [26, 32] for readability calculation. DISCO could be an alternative or complement to WordNet as the concept hierarchy that allows calculating document cohesion and scope. In addition, another similar alternative database to explore is Google n-gram [50].

In this work we have shown how to measure the readability of WSDL documents. Also, we compared well written documents and poorly written documents regarding their readability. However, we did not address how to improve the readability of documents, which is crucial in order to apply our approach massively in the industry. In a future work we will focus on a set of practical guidelines to improve the readability of documents.

Finally, we intend to apply our approach to Rest Web Services [45]. If we follow the approach proposed by IBM [51], where a Rest Web Service is described with WSDL 2.0, then our approach will be directly applicable. Otherwise, we will have to adapt our approach to the context of Restful Web Services and its associated description languages, such as RAML, OpenAPI (formerly Swagger) and API Blueprint.

Acknowledgment

This work is supported through research grants PICT-2012-0045 by ANPCyT and PIP 2013-2015, code 11220120100185CO and UNCo–Reuse (04-F009) and CONICET scholarships. We would like to thank the IT professionals, teachers and students who participated in the survey. We would also like to thank to the anonymous reviewers for their valuable comments to improve the quality of this paper.

References

[1] J. Erickson and K. Siau, “Web service, service-oriented computing, and service-oriented architecture: Separating hype from reality,” *Journal of BD Management*, vol. 19, no. 3, pp. 42 – 54, 2008. [Online]. Available: http://dx.doi.org/10.4018/jdm.2008070103

[2] M. Bichler and K. Lin, “Service-oriented computing,” *Computer*, vol. 39, no. 3, pp. 99–101, 2006. [Online]. Available: http://dx.doi.org/10.1109/MC.2006.102

[3] J. O. Coscia, C. Mateos, M. Crasso, and A. Zunino, “Refactoring code-first web services for early avoiding {WSDL} anti-patterns: Approach and comprehensive assessment,” *Science of Computer Programming*, vol. 89, Part C, pp. 374 – 407, 2014. [Online]. Available: http://dx.doi.org/10.1016/j.scico.2014.03.015

[4] M. Crasso, J. M. Rodriguez, A. Zunino, and M. Campo, “Revising WSDL Documents: Why and How,” *IEEE Internet Computing*, vol. 14, no. 5, pp. 48–56, 2010. [Online]. Available: http://dx.doi.org/10.1109/MIC.2010.81

[5] J. Pikulski, “Readability,” 2012, avaliable: http://www.eduplace.com/state/author/pikulski.pdf.

[6] P. Sripairojthikoon and T. Senivongse, “Concept-based readability of web services descriptions,” in *Advanced Communication Technology (ICACT), 2013 15th International Conference on*, 2013.

[7] D. Baski and S. Misra, “Metrics suite for maintainability of extensible markup language web services,” *IET Software*, vol. 5, no. 3, pp. 320–341, 2011. [Online]. Available: http://dx.doi.org/10.1049/iet-sen.2010.0089

[8] C. Mateos, M. Crasso, A. Zunino, and J. O. Coscia, “A stitch in time saves nine: Early improving code-first web services discoverability,” *International Journal of Cooperative Information Systems*, vol. 24, no. 02, p. 1550004, 2015. [Online]. Available: https://doi.org/10.1142/S0218843015500045

[9] M. Garriga, A. Flores, C. Mateos, A. Zunino, and A. Cechich, “Service selection based on a practical interface assessment scheme,” *International Journal of Web and Grid Services*, vol. 9, no. 4, pp. 369–393, October 2013. [Online]. Available: https://doi.org/10.1504/IJWGS.2013.057469

[10] X. Yan, D. Song, and X. Li, “Concept-based document readability in domain specific information retrieval.” New York, NY, USA: ACM, 2006, pp. 540–549. [Online]. Available: http://doi.acm.org/10.1145/1183614.1183692

[11] D. Roman, U. Keller, H. Lausen, J. De 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://dl.acm.org/citation.cfm?id=1412350.1412357

[12] C. Mateos, J. Rodriguez, and A. Zunino, “A tool to improve code-first web services discoverability through text mining techniques,” *Software: Practice and Experience*, 2014. [Online]. Available: http://dx.doi.org/10.1002/spe.2268

[13] D. Bouchiha, M. Malki, A. Alghamdi, and K. Alnafjan, “Semantic web service engineering: Annotation based approach,” *Computing and Informatics*, vol. 31, no. 6, pp. 1575–1595, 2012. [Online]. Available: http://www.cai.sk/ojs/index.php/cai/article/viewArticle/1332

[14] M. Crasso, A. Zunino, and M. Campo, “A survey of approaches to web service discovery in service-oriented architectures,” *Journal of Database Management (JDM)*, vol. 22, no. 1, pp. 102–132, 2011. [Online]. Available: https://dx.doi.org/10.4018/978-1-4666-2044-5.ch005

[15] A. D. Renzis, M. Garriga, A. Flores, A. Cechich, C. Mateos, and A. Zunino, “Assessing readability of web service interfaces,” in *2016 XLII Latin American Computing Conference (CLEI)*, Oct 2016, pp. 1–12. [Online]. Available: http://dx.doi.org/10.1109/CLEI.2016.7833369

[16] G. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. Miller, “Introduction to Wordnet: An On-line Lexical Database,” *International Journal of Lexicography*, vol. 3, no. 4, pp. 235–244, 1990.

[17] L. Ding, P. Kolari, Z. Ding, and S. Avancha, “Using ontologies in the semantic web: A survey,” in *Ontologies*. Springer, 2007, pp. 79–113. [Online]. Available: 10.1007/978-0-387-37022-4_4

[18] M. Papazoglou and W. V. D. Heuvel, “Service-oriented design and development methodology,” *International Journal of Web Engineering and Technology*, vol. 2, no. 4, pp. 412–442, 2006. [Online]. Available: https://dx.doi.org/10.1504/IJWET.2006.010423

[19] J. M. Rodriguez, M. Crasso, A. Zunino, and M. Campo, "Improving web service descriptions for effective service discovery," *Science of Computer Programming*, vol. 75, no. 11, pp. 1001–1021, 2010. [Online]. Available: https://dx.doi.org/doi:10.1016/j.scico.2010.01.002

[20] J. L. Ordiales, C. Mateos, M. Crasso, and A. Zunino, "Anti–pattern free code–first web services for state–of–the–art java wsdl generation tools," *International Journal of Web and Grid Services*, vol. 9, no. 2, pp. 107–126, 2013, in Press. [Online]. Available: https://dx.doi.org/10.1504/IJWGS.2013.054108

[21] J. Chall and E. Dale, "Readability revisited: The new dale-chall readability formula," *Brookline BookslLumen Editions*, 1995.

[22] M. Garriga, A. Flores, A. Cechich, and A. Zunino, "Web services composition mechanisms: A review," *IETE Technical Review*, no. ahead-of-print, pp. 1–8, 2015. [Online]. Available: http://dx.doi.org/10.1080/02564602.2015.1019942

[23] R. Navigli and S. P. Ponzetto, "BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network," *Artificial Intelligence*, vol. 193, pp. 217–250, 2012. [Online]. Available: https://doi.org/10.1016/j.artint.2012.07.001

[24] A. Pease, I. Niles, and J. Li, "The suggested upper merged ontology: A large ontology for the semantic web and its applications," in *Working notes of the AAAI-2002 workshop on ontologies and the semantic web*, vol. 28, 2002. [Online]. Available: https://www.aaai.org/Papers/Workshops/2002/WS-02-11/WS02-11-011.pdf

[25] E. Bottazzi and R. Ferrario, "Preliminaries to a dolce ontology of organisations," *International Journal of Business Process Integration and Management*, vol. 4, no. 4, pp. 225–238, 2009. [Online]. Available: https://doi.org/10.1504/IJBPIM.2009.03228

[26] P. Kolb, "Experiments on the difference between semantic similarity and relatedness," *Proceedings of the 17th Nordic Conference on Computational Linguistics - NODALIDA'09*, May 2009.

[27] R. V. Guha, D. Brickley, and S. Macbeth, "Schema.org: Evolution of structured data on the web," *Commun. ACM*, vol. 59, no. 2, pp. 44–51, Jan. 2016. [Online]. Available: http://doi.acm.org/10.1145/2844544

[28] I. Niles and A. Pease, "Towards a standard upper ontology," in *Proceedings of the international conference on Formal Ontology in Information Systems-Volume 2001.* ACM, 2001, pp. 2–9. [Online]. Available: https://dx.doi.org/10.1145/505168.505170

[29] C. Masolo, S. Borgo, A. Gangemi, N. Guarino, A. Oltramari, and L. Schneider, "The wonderweb library of foundational ontologies," 2002.

[30] N. F. Noy, "Semantic integration: a survey of ontology-based approaches," *ACM Sigmod Record*, vol. 33, no. 4, pp. 65–70, 2004. [Online]. Available: https://dx.doi.org/10.1145/1041410.1041421

[31] A. Gangemi, N. Guarino, C. Masolo, A. Oltramari, and L. Schneider, "Sweetening ontologies with dolce," in *International Conference on Knowledge Engineering and Knowledge Management.* Springer, 2002, pp. 166–181. [Online]. Available: https://dx.doi.org/10.1007/3-540-45810-7

[32] A. De Renzis, M. Garriga, A. Flores, A. Cechich, and A. Zunino, "Semantic-structural assessment scheme for integrability in service-oriented applications," in *Computing Conference (CLEI), 2014 XL Latin American*, Sept 2014, pp. 1–11. [Online]. Available: https://dx.doi.org/10.1109/CLEI.2014.6965175

[33] E. N. Zalta and S. Abramsky, "Stanford encyclopedia of philosophy," 2003. [Online]. Available: https://plato.stanford.edu/archives/win2012/entries/davidson/

[34] N. Juristo and A. Moreno, *Basics of software engineering experimentation.* Springer Publishing Company, Incorporated, 2010. [Online]. Available: https://dx.doi.org/10.1007/978-1-4757-3304-4

[35] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in software engineering.* Springer Science & Business Media, 2012. [Online]. Available: https://dx.doi.org/10.1007/978-3-642-29044-2

[36] R. E. Kirk, *Experimental design.* Wiley Online Library, 1982.

[37] M. Orne, "Demand characteristics and the concept of quasi-controls," *Artifacts in Behavioral Research: Robert Rosenthal and Ralph L. Rosnow's Classic Books*, p. 110, 2009.

[38] K. S. Abid, A. S. Abid, and M. M. Ansari, "A better approach for conceptual readability of wsdl," in *Multimedia Big Data (BigMM), 2015 IEEE International Conference on.* IEEE, 2015, pp. 260–263. [Online]. Available: https://doi.org/10.1109/BigMM.2015.52

[39] A. Heß, E. Johnston, and N. Kushmerick, "Assam: A tool for semi-automatically annotating semantic web services," in *The Semantic Web–ISWC 2004.* Springer, 2004, pp. 320–334. [Online]. Available: https://dx.doi.org/10.1007/978-3-540-30475-3_23

[40] J. M. Rodriguez, M. Crasso, C. Mateos, and A. Zunino, "Best practices for describing, consuming, and discovering web services: a comprehensive toolset," *Software: Practice and Experience*, vol. 43, no. 6, pp. 613–639, 2013. [Online]. Available: http://dx.doi.org/10.1002/spe.2123

[41] D. Aguilera, C. Gómez, and A. Olivé, "A complete set of guidelines for naming uml conceptual schema elements," *Data & Knowledge Engineering*, vol. 88, pp. 60–74, 2013. [Online]. Available: https://doi.org/10.1016/j.datak.2013.09.001

[42] T. P. Lau and K. Irwin, "Bilingual web page and site readability assessment," in *Proceedings of the 15th international conference on World Wide Web.* ACM, 2006, pp. 993–994. [Online]. Available: https://doi.org/10.1145/1135777.1135981

[43] S.-j. Yang, *A readability formula for Chinese language.* University of Wisconsin–Madison, 1970.

[44] R. Flesch, "A new readability yardstick." *Journal of applied psychology*, vol. 32, no. 3, p. 221, 1948.

[45] M. Garriga, C. Mateos, A. Flores, A. Cechich, and A. Zunino, "{RESTful} service composition at a glance: A survey," *Journal of Network and Computer Applications*, vol. 60, pp. 32 – 53, 2016. [Online]. Available: http://dx.doi.org/10.1016/j.jnca.2015.11.020

[46] W. Zhang, D. Song, P. Zhang, X. Zhao, and Y. Hou, "A sequential latent topic-based readability model for domain-specific information retrieval," in *Information Retrieval Technology.* Springer, 2015, pp. 241–252. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-28940-3_19

[47] C. Mateos, M. Crasso, A. Zunino, and J. L. Ordiales, "Detecting WSDL bad practices in code–first Web Services," *International Journal of Web and Grid Services*, vol. 7, no. 4, pp. 357–387, 2011. [Online]. Available: https://doi.org/10.1504/IJWGS.2011.044710

[48] H. N. Talantikite, D. Aissani, and N. Boudjlida, "Semantic annotations for web services discovery and composition," *Computer Standards & Interfaces*, vol. 31, no. 6, pp. 1108–1117, 2009. [Online]. Available: https://doi.org/10.1016/j.csi.2008.09.041

[49] M. Berón, H. Bernardis, E. Miranda, D. Riesco, M. Pereira, and P. Henriques, "Languages, applications and technologies: 4th international symposium, slate 2015, madrid, spain." Springer International Publishing, 2015, ch. WSDLUD: A Metric to Measure the Understanding Degree of WSDL Descriptions. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-27653-3_9

[50] J. Michel, Y. Shen, A. Aiden, A. Veres, M. Gray, J. Pickett, D. Hoiberg, D. Clancy, P. Norvig, and J. Orwant, "Quantitative analysis of culture using millions of digitized books," *Science*, vol. 331, no. 6014, pp. 176–182, 2011. [Online]. Available: https://dx.doi.org/10.1126/science.1199644

[51] L. Mandel, "Describe rest web services with wsdl 2.0," *Rational Software Developer, IBM*, 2008.