# Learning Domain Ontologies for Web Service Descriptions: an Experiment in Bioinformatics

Marta Sabou
Dept. of Artificial Intelligence
Vrije Universiteit Amsterdam
marta@cs.vu.nl

Chris Wroe, Carole Goble
Dept. of Computer Science
University of Manchester
{cwroe,carole}@cs.man.ac.uk

Gilad Mishne
Informatics Institute
University of Amsterdam
gilad@science.uva.nl

## ABSTRACT

The reasoning tasks that can be performed with semantic web service descriptions depend on the quality of the domain ontologies used to create these descriptions. However, building such domain ontologies is a time consuming and difficult task.

We describe an automatic extraction method that learns domain ontologies for web service descriptions from textual documentations attached to web services. We conducted our experiments in the field of bioinformatics by learning an ontology from the documentation of the web services used in $^{my}$Grid, a project that supports biology experiments on the Grid. Based on the evaluation of the extracted ontology in the context of the project, we conclude that the proposed extraction method is a helpful tool to support the process of building domain ontologies for web service descriptions.

## Categories and Subject Descriptors

H.3.5 [**Information Storage and Retrieval**]: Online Information Services—*Web-based Services*; I.2.6 [**Artificial Intelligence**]: Learning—*Knowledge acquisition*; I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*Text analysis*; H.4.m [**Information Systems**]: Miscellaneous

## General Terms

Languages, Design, Experimentation

## Keywords

Semantic Web, Web Services, OWL-S, Domain Ontology, Ontology Learning, Ontology Evaluation, Bioinformatics

## 1. INTRODUCTION

Semantic Web Service technology promises to automate web service discovery, composition and integration, tasks that currently need to be performed manually despite the quickly increasing number of on-line services. A common characteristic of all emerging frameworks for semantic web service descriptions (OWL-S [6], WSMO[1], IRS [15]) is that they combine two kinds of ontologies in a service description. First, a *generic web service description ontology*, such as OWL-S, specifies generic web service concepts (e.g., inputs, outputs) and prescribes the backbone of a semantic description. Second, a *domain ontology* specifies knowledge in the domain of the web service, such as types of service parameters (e.g., *Car*) and functionalities (e.g., *CarRental*), that fills in this generic framework. By *domain ontology* we mean a domain specific ontology that is designed and used for the semantic description of web services.

The complexity of the reasoning tasks that can be performed with semantic web service descriptions is conditioned by several factors. First, all web services in a domain should use concepts from the same domain ontology in their descriptions. Otherwise the issue of ontology mapping has to be solved which is a very difficult problem in itself. This requires that domain ontologies should be *generic* enough to provide the needed concepts by any web service in a certain domain. Second, the richness of the available knowledge is crucial for performing complex reasoning. Therefore, the domain ontology should be *rich in semantics.* We conclude that such quality domain ontologies are at least as important as generic web service description ontologies.

Despite their importance, few domain ontologies for web service descriptions exist and building them is a challenging task. A major impediment is the *the lack of guidelines* on how to build such ontologies, what knowledge they should contain and what design principles they should follow. In the bioinformatics domain, for example, different communities used different approaches to build very different ontologies for semantically describing web services [12]. Further, in order to build a generic domain ontology one would need to inspect a large number of web services in that domain. However, no tools are available to support this process.

Our work addresses the current lack of tooling by developing an automatic method that learns a domain ontology for the purpose of web service description from natural language documentations of web services. The method was evaluated in the domain of bioinformatics, in the context of the $^{my}$Grid project[2] [24]. Domain ontology building efforts in $^{my}$Grid prove the difficulty of this task. The project therefore provided a realistic application scenario for our method as well as valuable experimental data for its evaluation.

In what follows we present the $^{my}$Grid project (Section 2), we describe the extraction method (Section 3) and the experimental setup used for its evaluation (Section 4). The results are presented and discussed in Section 5. After positioning our work in the landscape of related work (Section 6), we conclude and point out future work in Section 7.

[1] http://www.wsmo.org/

[2] http://www.mygrid.org.uk/

*WWW 2005*, May 10-14, 2005, Chiba, Japan.
ACM 1-59593-046-9/05/0005.

## 2. BUILDING ONTOLOGIES IN $^{my}$Grid

$^{my}$Grid is a UK EPSRC e-Science pilot project building semantic grid middleware to support *in silico* experiments in biology. Experimental protocol is captured as workflow, with many steps performed by web services. Core to the infrastructure is an ontology for describing the functionality of these services and the semantics of the manipulated data. A key role of the ontology is to facilitate user driven discovery of services at the time of workflow construction. In contrast to efforts such as OWL-S and WSMO, the ontology is not currently intended to support workflow specification, agent-driven automated service discovery, automatic invocation, or monitoring.

Several factors hampered the building of this ontology, transforming the process into a time consuming and difficult activity. First, ontology building in itself is *time consuming*. The ontology was initially built with two months of effort from an ontology expert with four years experience in building description logic based biomedical ontologies. The ontology was built manually in OilEd[3], initially using the documentation for 100 bioinformatics services as a source of relevant terms. These services are part of the EMBOSS(European Molecular Biology Open Software Suite) service collection[4], further referred to as EMBOSS services. A second impediment is the *dynamic nature* of the field. The exponential rise in the number of bioinformatics web services over the past year required a further two months of effort to maintain and extend the ontology. However, its content currently lags behind that needed to describe the 600 services available to the community. Thirdly, *lack of tools* hampered the process. At the time of development, tool support for handling separate ontology modules was minimal, hence the existence of one substantial ontology with distinct subsections covering the domains of molecular biology, bioinformatics, informatics and generic tasks, all under a common upper level structure. Also, no tools existed to facilitate getting an insight in large collections of natural language web service descriptions.

A final, fourth impediment that the ontology curator encountered was the *lack of guidelines* on how to build the domain specific ontology, or indeed how to relate it to upper level ontologies. Since at that time DAML-S (the predecessor of OWL-S) was still under development, the ontology curator devised their own generic web service description schema based on DAML-S but much simplified to reflect narrower requirements. It describes service inputs and outputs but not preconditions and effects because of the data flow orientation of the available services. Four more properties describe the task performed by a service, the method by which it is achieved, and specific applications and resources used. The domain values for each property come from specific subsections of the ontology, mainly the bioinformatics and task subsections. It is these sections that reflect the concepts explicitly mentioned in web service documentation and play the role of a domain ontology. The other sections support formal definition of these concepts and higher level structure. Lacking guidance from the web services field, the curator relied on design principles employed in other large open source biomedical ontologies such as openGALEN [21] and the TAMBIS ontology [1].

A part of this ontology, further referred to as the *application ontology*, provides concepts for annotating web service descriptions in a forms based annotation tool Pedro[5] and is subsequently used at discovery time with or without reasoning to power the search [25].

The ontology building experience in $^{my}$Grid suggests the need of automated tools that support the ontology curator in his work, especially now with the exponential increase of the number of bioinformatics services. We benefitted from work done in $^{my}$Grid to evaluate the extraction method. We used 158 documentations of EMBOSS services (an extended set of the original collection) as a basis for the extraction. The manually built ontology (to which we refer as *Gold Standard*) and the application ontology were used for the evaluation of our method[6].

## 3. EXTRACTION METHOD

Software documentation (such as javadoc style code comments) in general, and web service descriptions in particular, employs natural language in a very specific way. In fact, such texts belong to what was defined as a sublanguage in [9]. A sublanguage is a specialized form of natural language which is used within a particular domain or subject matter and characterized by specialized vocabulary, semantic relations and syntax. Weather reports or real estate advertisements are other examples of sublanguages.

Our extraction method exploits the syntactic regularities which are inherent from the sublanguage nature of web service documentations. Note that in many cases, for example as demonstrated by the presented case study, the web service descriptions also reflect the particularities of the domain language, in our case bioinformatics. Accordingly, our extraction exploits the characteristics of both sublanguages.
The ontology extraction consists of several steps, as depicted in Figure 1. The first stage consists of annotating the corpus with linguistic information that helps in deciding the possible role of each word in the future ontology to be built. In the second step, a set of *syntactic patterns* is used to identify and extract information from the now annotated corpus. The next step, ontology building, transforms the extracted relevant information into ontological constructs. Finally, a pruning step excludes potentially uninteresting concepts from the ontology. In what follows we detail all these extraction steps.

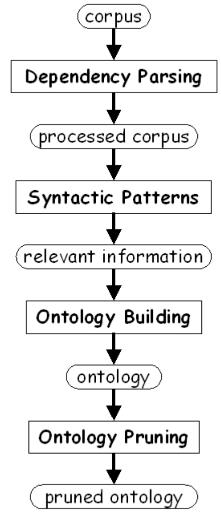

**Figure 1:** Extraction Method.

### 3.1 Dependency Parsing

In previous work we used Part-of-Speech (POS) tags as a basis for our extraction method [23]. This weak linguistic information limited the number and quality of extracted ontological constructs. Therefore, in the current work we opted to exploit richer linguistic information such as pro-

[3] http://oiled.man.ac.uk

[4] http://www.hgmp.mrc.ac.uk/Software/EMBOSS/Apps/

[5] http://pedrodownload.man.ac.uk/

[6] All experimental material is available online at http://www.cs.vu.nl/~marta/experiments/extraction.html

| Position | Word | Lemma | POS | Relation | Head |
|---|---|---|---|---|---|
| 1 | find | find | V | - | - |
| 2 | antigenic | antigenic | A | mod | 3 |
| 3 | sites | site | N | obj | 1 |
| 4 | in | in | Prep | mod | 3 |
| 5 | proteins | protein | N | pcpmp-n | 4 |

**Table 1: An example Minipar output.**

vided by dependency parsing, a commonly used method in computational linguistics to identify dependency relations between words in natural language. A dependency relation is an asymmetric binary relation between a word called **head** and a word called **modifier**.

We use Minipar [11], a state of the art dependency parser with a reported high performance (88% precision and 80% recall with respect to dependency relations). As an example, we list in Table 1 Minipar's analysis for the sentence: *Find antigenic sites in proteins*. For each word in the sentence, the following information is provided : (i) the position of the word in the sentence; (ii) the word as it appears in the sentence; (iii) the lemma of the word ; (iv) the part of speech of the word (e.g. V(verb), N(noun)); (v) the name of the dependency relation between this word and the head (e.g., obj, mod) and (vi) the position of the head word modified by the current word. Accordingly, in the given example *antigenic* is an adjective which modifies the noun *sites*, and *sites* is the object of the verb *find*.

## 3.2 Syntactic Patterns

The previously derived dependencies are explored by several syntactic patterns to identify potentially interesting information for ontology building. We call such patterns syntactic because they are defined on knowledge about the syntax of the sentence. We distinguish three major categories of patterns used to derive different types of information.

**1. Identifying domain concepts.** Domain concepts are generally depicted by the nouns in a corpus. Given the small size of the corpus and the concise style of the web service documentations the majority of nouns denote potentially interesting domain concepts. We extract both unmodified nouns and entire noun phrases (NP). Noun phrase detection is accomplished by extraction patterns that explore the "nn" (depicts a noun modifier of a noun) and "mod" (depicts an adjective modifier of a noun) dependency relations. When such relations are identified, the head noun together with its modifiers are annotated as being a noun phrase. In the example above, our patterns identify <*antigenic site*>[7] as a noun phrase.

**2. Identifying functionalities.** Besides domain concepts, a domain ontology used for web service descriptions needs to specify types of functionalities that are frequently offered in that domain. We observed that, in the majority of cases, verbs identify the functionality performed by a method and nouns closely related to these verbs (in particular their objects) refer to data structures that are involved in some way in that functionality. This observation was successfully used in a previous case study in the domain of RDF storage facilities [23]. Therefore our extraction pattern identifies verbs and their objects as potential information to be added to the domain ontology. If the object is the head of a noun phrase then the whole noun phrase is extracted. Note that this pattern relies on the output of the previous NP extraction pattern. For our example, it would identify <*find*>_<*antigenic site*> as a lexical construct denoting a possible functionality in the bioinformatics domain.

| Position | Word | Lemma | POS | Relation | Head |
|---|---|---|---|---|---|
| 1 | replace | replace | V | - | - |
| 2 | or | or | U | lex-mod | 1 |
| 3 | delete | delete | V | lex-dep | 1 |
| 4 | sequence | sequence | N | nn | 5 |
| 5 | sections | section | N | obj | 1 |

**Table 2: Verb dependency example.**

| Position | Word | Lemma | POS | Relation | Head |
|---|---|---|---|---|---|
| 1 | pick | pick | V | - | - |
| 2 | pcr | pcr | N | nn | 3 |
| 3 | primers | primer | N | obj | 1 |
| 5 | hybridi-<br>zation | hybridi-<br>zation | N | nn | 6 |
| 6 | oligos | oligos | N | conj | 3 |

**Table 3: Noun dependency example.**

This pattern captures the desired information in the majority of cases with a few exceptions. One of the exceptions occurs when several verbs in a sentence refer to the same object. For example, the sentence *Replace or delete sequence sections* suggests that both <*replace*>_<*sequence section*> and <*delete*>_<*sequence section*> are valid functionalities in this domain that we wish to extract. However, Minipar's output does not directly encode the verb-object relation between *delete* and *section* (see Table 2). On the other hand, the analysis denotes that there is a dependency relation between the two verbs of the sentence. Whenever two or more verbs are related by a logical operator they should be bound to a single object (the object of one of the verbs). One of our extraction patterns identifies cases when several verbs are related via the "lexdep" or "conj" relations. These relations denote cases when verbs are related via logical operators such as "or", "and" (e.g., *Reverse and complement a sequence*) or ",". Often there are cases when the logical dependency between more than two verbs is partially specified and we have to explicitly define all dependents based on the transitivity of this relation (e.g. if dependency(v1,v2) and dependency(v2,v3) then dependency(v1,v3)).

Another exception is when several objects are in a conjunctive relation. For example, from *Pick pcr primers and hybridization oligos* we wish to extract both <*pick*>_<*pcr primer*> and <*pick*>_<*hybridization oligos*> functionalities. However, the Minipar output specifies only the first verb-object relation (see Table 3). Nevertheless, knowing that there is a conjunctive relation between *primers* and *oligos* we can deduce that *oligos* also plays an object relation with respect to the verb *pick*. Just as with verbs, we have written a pattern that identifies conjunctive NPs and deduces the additional knowledge. The patterns that identify dependency of verbs and objects are performed before the pattern that identifies functionalities.

**3. Identifying relations.** We observed that prepositional phrases (PP) often express a meronymy (PartOf)

[7] We use this convention to present <*lexical constructs*> extracted from the corpus.

relation between the terms that they interrelate. For example, in Table 1, a preposition relates *sites* and *proteins* suggesting that <*antigenic sites*> are parts of a *protein*. We could translate these relations in ontological relations between concepts. We performed only initial experiments in this direction and the syntactic and semantic ambiguity of prepositions makes it too early to report on this work.

Note that the presented syntactic patterns capture major characteristics of the sublanguage and therefore they identify frequently appearing situations. For this case study we did not implement patterns that exploit less frequent situations (e.g., possessive articles) given the already high coverage of these most generic patterns (as shown by our evaluation in Section 5).

## 3.3 Ontology Building

The ontology building step collects the results of the previous syntactic pattern based extraction. The extracted terms are used for building two different parts of the domain ontology. First, noun phrases are a basis for deriving a data structure hierarchy, having *DataStructure*[8] as the most generic concept. Second, the functionality information (derived by the dependency and functionality extraction patterns) is used for building a functionality hierarchy, having *Functionality* as the most generic concept. Note that an implicit lemmatization is performed because the patterns return the lemmas of the identified terms.

**Building the data structure hierarchy.** We observed that many of the terms mentioned in the analyzed corpus expose a high level of compositionality, in the sense that they incorporate other meaningful terms as proper substrings. Our observation agrees with a recent study [18] of the Gene Ontology terms which proved that 63,5% of all terms in this domain are compositional in nature. Another observation, also stated by this study, is that compositionality indicates the existence of a semantic relation between terms. Namely, a term A contained as a proper substring of a term B is more generic than term B. This translates in the ontological subsumption relationship.

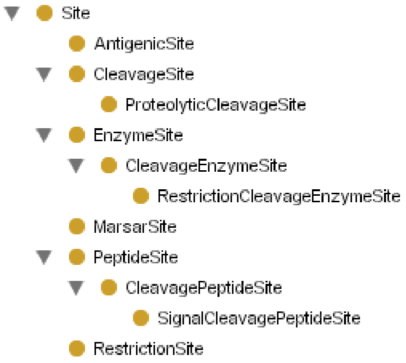

**Figure 2: The *Site* concept and hierarchy.**

The hierarchy building algorithm reflects our observations. It inserts all noun phrases in a hierarchy so that the head noun is the more generic term and terms built by the incremental addition of the noun modifiers (in the order they appear in the text) are considered more specific concepts. As an illustration, Figure 2 depicts the data structure hierarchy built from all the noun phrases with the head *Site*.

**Building the functionality hierarchy.** For building the functionality hierarchy we rely on a simpler algorithm. Each identified distinct verb is added as a concept in the functionality hierarchy (see a snapshot of this hierarchy in Figure 3). Verb-object pairs are added as subtypes of functionalities for that verb. For example, when we identify <*delete*>_<*sequence section*> *Delete* is registered as a type of *Functionality* and *Delete_SequenceSection* is added as a specific type of the *Delete* functionality.

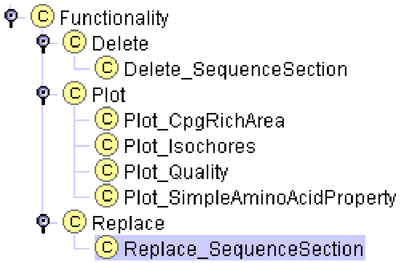

**Figure 3: The *Functionality* hierarchy.**

Naturally, there are other possible ways to represent this functionality information. However, there are no clear guidelines in the field of semantic web services about how these functionality hierarchies should look like. Our conceptualization is close to the OWL-S/IRS/WSMO style of modelling functionalities in a domain by including both the verb of the action and a data element directly involved in the functionality. Note that this step can be rewritten to reflect other ontology building strategies.

**Observations.** Currently we establish a one to one correspondence between phrases in the corpus and derived concepts. However, different words often have the same meaning and they should account for a single concept. Background knowledge of such information would allow a more complex ontology building step: concepts would correspond to distinct senses identified in the corpus rather than distinct lexical tokens. Different lexical constructs denoting the same concept should still be contained in the ontology as lexical realizations of a concept. This is useful for applications that encounter different lexical references to the same concept (e.g., search applications).

A semantic web service description will use concepts from both *Functionality* and *DataStructure* hierarchies creating an implicit link between the used concepts. For example, one service might specify that it performs a *Delete_SequenceSection* activity expecting a *SequenceSection* as its input. Therefore, currently we do not explicitly specify any link between the two hierarchies of the domain ontology. Nevertheless, we observed that formally specifying relations between the two kinds of concepts could lead to deriving extra domain knowledge. We call rules that use the knowledge in the learned ontology to predict new knowledge *semantic patterns*. We believe that they are a promising extension of our technique.

## 3.4 Ontology Pruning

We employ a simple pruning strategy by eliminating any direct specific concept of *DataStructure* that has no specific concepts itself. This heuristic relies on our observation that

[8] Ontology *Concepts* are capitalized and denoted in italics.

complex terms that generate a subhierarchy by decomposition are more likely to be relevant. The evaluation step (subsection 5.2) showed that our intuition was fairly accurate and suggested novel heuristics for performing this step.

**Implementation.** The extraction method is implemented using the GATE [7] framework[9]. We implemented our extraction patterns using JAPE [8], a rich and flexible rule mechanism which is part of the framework. Other modules of our method (e.g., Ontology Building) were declared as GATE Processing Resources. The data used by our method (such as Minipar analysis or structures identified by patterns) is represented as *annotations* on the analyzed texts. Both patterns and individual modules operate on these annotations and represent their output as annotations.

# 4. EXPERIMENTAL SETUP

## 4.1 Corpus

Our experimental corpus consisted of 158 EMBOSS bioinformatics service descriptions. We worked only on the short method descriptions since they are significant for web service descriptions in general being very similar to descriptions found in online web service repositories as SalCentral[10] and XMethods[11]. The detailed descriptions of the EMBOSS services present a specific layout which makes extraction much easier, however using it would have biased our extraction methods towards this particular kind of documentation.

## 4.2 Evaluation

Evaluation of ontology learning methods is a very important but largely unsolved issue, as reported by papers in a recent workshop [3]. There is a general consensus that evaluation is either quantitative or qualitative [2, 17]. *Quantitative evaluation* measures the performance of the various software algorithms that constitute the extraction tool. *Qualitative evaluation* assesses the adequacy of an ontology as a conceptualization of a certain domain.[12]

While quantitative evaluation can be performed in a fairly easy manner by using the well-established recall/precision metrics, qualitative evaluation is more subtle and there are no standard methods for performing such evaluations. A common approach is to compare an automatically extracted ontology with a Gold Standard ontology which is a manually built ontology of the same domain ([22]), often reflecting the knowledge existent in the corpus used for the extraction ([5]). The goal of this approach is to evaluate the degree to which the ontology covers a certain analyzed domain. Another approach is to evaluate the appropriateness of an ontology for a certain task. Initial experiments with such a task-based ontology evaluation are reported in [20]. Finally, [17] advocates a per-concept evaluation by a domain expert. In order to support this process they use simple natural language generation techniques to provide a natural language definition for all concepts learned automatically.

We consider that all these types of evaluation cover important and complementary aspects, and use a combination of them to answer the following questions:

1. *What is the performance of the learning algorithm?*

2. *Is the ontology a good basis for ontology building?*

3. *Does the ontology cover the analyzed domain?*

4. *Does the ontology support a certain task (here: search)?*

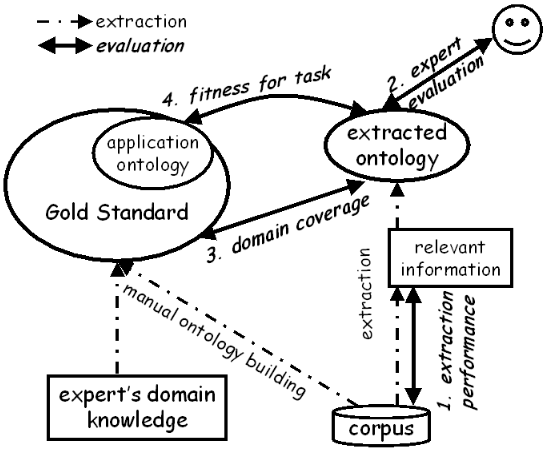

**Figure 4: Evaluation Strategies.**

Figure 4 depicts the kinds of evaluations we perform. We employ quantitative evaluation to asses the performance of our extraction algorithm (marked "1" in Figure 4). From a qualitative perspective, we evaluate several aspects of the ontology. First we rely on the domain expert's concept per concept based evaluation of the ontology to conclude its usefulness for supporting the ontology building task (2). The domain expert is the curator of the Gold Standard ontology. To asses domain coverage we compare the extracted ontology to the manually built Gold Standard which, given its size and complexity, we consider representative for this domain (3). Finally, to get an insight in the degree to which the extracted ontology would support the search task, we compare it to the subset of the Gold Standard that currently supports the $^{my}$Grid search facilities - the application ontology (4). In what follows, we present the methodology and metrics for performing each type of evaluation.

**1. Extraction Performance.** To measure the performance of the extraction modules we manually identified all the lexical constructs to be extracted from the corpus. Then, using the Corpus Benchmark Tool offered by GATE, we compared this set of lexical constructs with the ones that were extracted through syntactic analyzes and pattern based extraction. We use Lexical Recall to quantify the number of relevant lexical constructs that are extracted from the analyzed corpus ($correct_{extracted}$) compared to all constructs to be extracted from the corpus ($all_{corpus}$). Lexical Precision denotes the ratio of correctly extracted constructs over all automatically extracted constructs ($all_{extracted}$).

$$LRecall = \frac{correct_{extracted}}{all_{corpus}} \quad ; \quad LPrecision = \frac{correct_{extracted}}{all_{extracted}}$$

[9] The prototype implementation is available online at http://www.cs.vu.nl/~marta/experiments/extraction.html

[10] http://www.salcentral.com

[11] http://www.xmethods.net

[12] Qualitative evaluation, as defined in ontology learning, can make use of quantitative measures. This slightly conflicts with the general distinction between qualitative and quantitative evaluations.

**2. Ontology Building Support.** Assessing whether the extracted ontology offers a useful basis for building a domain ontology is very important since the main goal of our research is supporting the ontology building task. During the concept per concept analysis of the extracted ontology the domain expert rated concepts *correct* if they were useful for ontology building and were already included in the Gold Standard. Concepts that were relevant for the domain but not considered during ontology building were rated as *new*. Finally, irrelevant concepts, which could not be used, were marked as *spurious*. The higher the ratio between all relevant concepts and all concepts of the ontology, the better it supports ontology building. We express this ratio as an Ontology Precision ($OPrecision$) metric:

$$OPrecision = \frac{correct + new}{correct + new + spurious}$$

Very useful side-results of the expert evaluation were the opinion and suggestions of the expert. They provided valuable ideas for further improvements.

**3. Domain Coverage.** To asses domain coverage we compared the extracted ontology to the Gold Standard. According to [13], one of the few works on measuring the similarity between two ontologies, one can compare two ontologies at two different levels: lexical and conceptual. Lexical comparison assesses the similarity between the lexicons (set of terms denoting concepts) of the two ontologies. At the conceptual level the taxonomic structures and the relations in the two ontologies are compared.

The hierarchical structure of the Gold Standard is complex and very different from the extracted ontology. Several different views are defined using abstract concepts that account for 18% of all its concepts. In contrast, the extracted ontology introduces a limited number of hierarchical relations between domain concepts. Therefore, we only perform a lexical comparison of the two ontologies.

Our first metric denotes the shared concepts between the manual and extracted ontology. This metric was originally defined in [13] as the “relative number of hits” (RelHit), then renamed in [5] to Lexical Overlap (LO). Let $L_{O_1}$ be the set of all *domain relevant* extracted concepts (which is a subset of all extracted concepts rated *correct* and *new* by the domain expert during the second evaluation phase) and $L_{O_2}$ the set of concepts of the Gold Standard. The lexical overlap is equal to the ratio of the number of concepts shared by both ontologies (i.e., the intersection of these two sets - also noted *correct*) and the number of all Gold Standard concepts (noted *all*) (intuitively: recall).

$$LO(O_1, O_2) = \frac{|L_{O_1} \cap L_{O_2}|}{|L_{O_2}|} = \frac{correct}{all}$$

It is interesting to analyze the concepts that are not part of this overlap. Often, the extracted ontology can bring important additions to the manual ontology by highlighting concepts that were ignored during manual creation. We are not aware of any previously defined metric for measuring these additions. Therefore we define the Ontological Improvement (OI) metric as the ratio between all domain relevant extracted concepts that are not in the Gold Standard (noted *new*) and the concepts of the Gold Standard.

$$OI(O_1, O_2) = \frac{|L_{O_1} \setminus L_{O_2}|}{|L_{O_2}|} = \frac{new}{all}$$

| | **Functionality** | **NP** | **All** |
|---|---|---|---|
| $correct_{extracted}$ | 125 | 250 | 375 |
| $all_{corpus}$ | 147 | 312 | 459 |
| $all_{extracted}$ | 145 | 267 | 412 |
| **LRecall** | **85%** | **80%** | **82%** |
| **LPrecision** | **86%** | **94%** | **91%** |

**Table 4: Results of quantitative evaluation.**

We also determine the concepts in the Gold Standard that were not extracted and try to explain why our method failed. Such analysis can give valuable suggestions for the improvement of the method. We define the Ontological Loss metric (OL) as the ratio between non-extracted concepts ($all_{missed}$)and all concepts of the Gold Standard.

$$OL(O_1, O_2) = \frac{|L_{O_2} \setminus L_{O_1}|}{|L_{O_2}|} = \frac{all_{missed}}{all}$$

**4. Fitness for a Task.** A final perspective on ontology quality is the degree to which it supports a certain task. Web service search in the $^{my}$Grid project is powered by a part of the Gold Standard, the *application ontology*. The extracted ontology would support the search task if it would improve the performance of this task. However, since the search task itself is difficult to evaluate, we will simply compare our extracted ontology to the *application ontology*. We use the previously derived three metrics to capture lexical overlap, loss and improvement. We rely on their value to draw our conclusions about the fitness of the extracted ontology for the search task.

# 5. RESULTS

## 5.1 Extraction Performance

The results of this evaluation step, depicted in Table 4, indicate that we can extract the desired lexical information with a great fidelity from the corpus. This shows that the implemented patterns cover the majority of cases. We observed that errors are mostly due to mistakes in Minipar's output (e.g., mistaking adjectives for verbs). Also, we discovered some (less frequent) situations that are not covered by our patterns and for which new patterns can be written.

## 5.2 Ontology Building Support

The results of the expert evaluation are depicted in Table 5. The high ontology precision indicates that we reached our goal to provide a “clean” ontology. Even more, we suggested major additions both for *DataStructures* and *Functionalities*, which are essential for web service descriptions. Further, the pruning mechanism performed satisfactory: it suggested 67 concepts for exclusion out of which only 13 should not have been pruned. We derived several interesting observations from the comments of the domain expert.

**Recall vs. Precision.** It seams that the cleanness of the ontology is not of major importance for the ontology engineer. Often even terms that are not included in the final ontology are useful to give an insight in the domain

|  | **Functionality** | **DataStructure** | **All** |
|---|---|---|---|
| *correct* | 13 | 26 | 39 |
| *new* | 142 | 164 | 306 |
| *spurious* | 25 | 23 | 48 |
| **OPrecision** | **86%** | **89%** | **87%** |

**Table 5: Results of expert evaluation.**

itself and to guide further abstraction activities. We should therefore concentrate to increase the recall of our extraction process even at the expense of the precision of the results.

**Evidence.** The domain expert had difficulties to judge the relevance of certain concepts for the domain (e.g., "name"). It is advisable to provide extra information about each concept to support the decision of the domain expert. For example, providing *evidence* of the use of a concept in the corpus, or its frequency in the corpus.

**Synonymy.** During the evaluation, the expert recognized several potential synonym sets such as: {*find, search, identify, extract, locate, report, scan*}, {*fetch, retrieve, return*}, {*pick, select*}, {*produce, calculate*} or {*reverse, invert*}. Synonymy information is an important piece of knowledge for semantic web services. Especially search and matchmaking services would benefit from knowing which concepts are equivalent. The ontology engineer can decide to include synonymy in his ontology in different ways. For example he can maintain all these concepts and describe their synonymy via an explicit mapping (e.g., owl:equivalentClass). Alternatively, he can choose to maintain one single concept per synonym set and link all lexical synonyms to this concept. Therefore, automatic acquisition of synonymy information remains an important future work.

**Problems.** Throughout the evaluation the expert suggested several possible improvements for ontology building. For example, even if efficient in the majority of cases, the compositionality algorithm should not always be applied. The current implementation leads to cases where the most generic and the intermediary concepts (derived from a composed lexical construct) are not valid domain concepts.

**Abstractions.** The expert often redistributed the extracted domain concepts according to his domain view. For example, two subclasses identified for *Protein* belong to different domains, molecular biology and bioinformatics, and have to be placed in the corresponding hierarchies accordingly. Such abstractions need to be still manually created according to the ontology engineers view on the domain. However, the abstraction step is considerably supported if the expert has an overview of relevant domain concepts.

**Support.** Despite these suggestions, the curator considered the extracted ontology as a useful start for deriving a domain ontology. Several complex structures were fit for inclusion in a final ontology (ex. the *Site* hierarchy in Figure 2), or provided helpful hints in how certain concepts inter-relate. The most appreciated contribution was that the learned ontology even suggested new additions for the manually built Gold Standard.

## 5.3 Domain Coverage

While comparing the extracted ontology to the Gold Standard, we observed that concepts were missed due to four major reasons. Accordingly, we classify missed concepts into four separate categories.

|  | **Gold Standard** | **Application Ontology** |
|---|---|---|
| $all$ | 549 | 125 |
| $correct$ | 39 | 25 |
| $all_{missed}$ | 510 | 100 |
| $missed_{corpus}$ | 3 (0.6%) | 0 (0%) |
| $missed_{external}$ | 360 (70.6%) | 88 (88%) |
| $missed_{abstract}$ | 101 (19.8%) | 6 (6%) |
| $missed_{composed}$ | 46 (9%) | 6 (6%) |
| $new$ | 306 | 27 |
| **LO** | **7%** | **20%** |
| **OL** | **93%** | **80%** |
| **OI** | **56%** | **21.5%** |

**Table 6: Domain coverage and fitness for a task. For each category of missed concepts we show the percentage of the total loss they cause.**

$missed_{corpus}$ denotes concepts that were in the corpus but our extraction method failed to identify them;

$missed_{external}$ denotes concepts of the Gold Standard that are never mentioned in our corpus (e.g., biological structures such as different types of cells, organisms);

$missed_{abstraction}$ denotes concepts that define views, building abstraction levels over the actual domain concepts. For example, services are classified according to their compositionality (composed vs. primitive), type of input or output. These kinds of concepts are also not present in the corpus;

$missed_{composed}$ denotes concepts that are derived from extracted concepts by a modifier (e.g., "id", "report", "record"). Even if one can consider that we half learned these concepts, we will treat them as missed ones.

We obtain a rather low lexical overlap (7%) and a large ontology improvement (56%) as our extracted ontology contains many concepts that are not in the Gold Standard (see the second column of Table 6). These values suggest that the ontology curator worked rather independently from the given corpus during the building of the Gold Standard as he missed many concepts named in the corpus. Post-experimental interviewing of the curator revealed that the examination of the corpus was not meticulous. He used just in time ontology development: concepts were added if needed for describing a certain service. Note also that he worked on a subset of our analyzed corpus (100 service descriptions instead of 158 analyzed by us). Name changes could also account for the mismatch. The curator expanded abbreviations or created a preferred term for several synonyms. He acknowledged that the automatic approach leads to a more faithful reflection of the terms the community uses to describe their services.

Ontology loss was high (93%). Note, however, that the loss is caused by the four major factors external to our extraction method. In fact, our method only missed 3 concepts in the corpus (and this caused 0.6% of the global loss). The major cause for losses (70.6% of global OL) was the fact that the curator included concepts about the fields of biology, bioinformatics and informatics that are not present in the corpus. For this he relied on his expert knowledge and other ontologies in the domain (see Section 2). Another cause for losses were the abstraction concepts of the ontology: 101 out of 549 concepts (18%) constitute the abstraction layers of the ontology. This lead to another partial loss of 19.8%.

## 5.4 Fitness for a Task

The application ontology differs from the Gold Standard in several aspect. First, it is smaller in size, containing only 23% of the Gold Standard's concepts. Second, it includes less abstraction concepts (6 out of 125, i.e., 0.4%) than the Gold Standard (18%). This hints that broad domain coverage and complex views are not of primary importance for the search application.

The extracted ontology has a larger coverage of the *application ontology* (LO = 20%) than of the Gold Standard ontology (7%) (see the last column of Table 6). The losses are again high and due to the absence of many concepts from the corpus.

Computing ontology improvement was not straightforward. It is very difficult to decide which of the new concepts are relevant for the search task. We know, however, that types of actions play an important role in the search application because users can search for web services based on the functionality they offer. Therefore, we only considered new *Functionality* concepts (e.g., *Scan*, *Locate*, *Identify*) excluding their subclasses since this particular application does not make use of composed concepts. Despite these severe reduction of the newly extracted set, we counted 27 potentially useful concepts, leading to an important OI of 21.5%.

# 6. RELATED WORK

Automatic acquisition of semantic web service descriptions was considered by the work of two research teams. Hess and Kushmerick [10] employ Naive Bayes and SVM machine learning methods to classify WSDL files (or web forms) in manually defined task hierarchies. Our work is complementary, since we address the acquisition of such hierarchies. Also, our method does not rely on any manually built training data as the machine learning techniques do. Patil et al. [19] employ graph similarity techniques to determine a relevant domain for a WSDL file and to annotate the elements of the WSDL file. Currently they determine the semantics of the service parameters and plan to concentrate on functionality semantics in the future. They use existent domain ontologies and acknowledge that their work was hampered by the lack of such ontologies.

The ontology learning field offers a wide range of different approaches to ontology acquisition. While most work is targeted on specific domains we are not aware of any efforts that analyze software documentation style texts. Several generic ontology learning tools exist, namely Text-To-Onto [14], OntoLearn [16] or OntoLT [4]. We tried to extract a domain ontology from our corpus using Text-to-Onto, the only tool publicly available. The results (not reported in this paper) were suboptimal due to the strong particularities of our corpus which hampered the efficiency of the generic methods implemented by the tool.

Our method is tailored for dealing with web service descriptions. The extraction patterns presented in this paper exploit the characteristics of both web service descriptions and biological texts. The observations behind the patterns specific to web service descriptions were already used in our previous work in a different domain (RDF storage facilities) [23]. Therefore, we believe that these patterns are tailored on service description sublanguage without being domain dependent.

# 7. CONCLUSIONS AND FUTURE WORK

**Domain ontologies are hard to build.** During our work we gained insight in the severe consequences of the lack of guidelines for building domain ontology. Besides the fact that very different ontologies are built for the same domain (as showed by a previous study [12]), we witnessed a situation when much effort is invested in building a large and complex ontology from which only a small part (23%) is used for implementing semantic web services technology. The domain expert omitted important knowledge available in the textual descriptions of web services due to no tool support to automate this process. Lack of guidelines and tools hampers the development of semantic web service technology and requires special attention from the community.

**The extraction method provides a good starting point for ontology building.** We address these needs by providing an extraction method that supports acquisition of domain ontologies from textual sources. Evaluating the performance of our extraction is a complex issue, which we addressed from different perspectives. According to our quantitative evaluation, the system extracts the desired information from a corpus with high accuracy. From a qualitative perspective, the domain expert considered the extracted ontology as a helpful first step in the creation of a domain ontology. The evaluation indicated that the manual extraction of concepts had been done inadequately by the domain expert. He concluded that automated support for extraction and hierarchical structuring would provide a valuable benefit, and could be used immediately to extend the existing complex, manually built ontology. When comparing the extracted ontology to the Gold Standard we obtained a low coverage of this ontology due to the fact that most of its concepts do not appear in the analyzed corpus. We had a much better coverage of the actual application ontology (20%) since we addressed the extraction of facets that are of major importance to semantic web service technology. We also extracted many new concepts that could considerably enlarge the Gold Standard (to maximize domain coverage) as well as the application ontology (to increase the performance of the search task). The inclusion of these new concepts in the existing ontologies would increase their correlation with the actual description of the existing web services. We consider these results supportive for further developing our extraction method.

One could imagine the use of our method in several application scenarios. We believe it is useful for providing a first insight into a domain and providing a sketch ontology that can be the basis for ontology construction. During our experiments it turned out that such a tool is also useful when an ontology already exists in a domain as it often can suggests new additions that were ignored during manual ontology building. Also, as the domain changes such a method can help to discover newly available domain knowledge and integrating it in the existing ontology. Finally, it can be the basis of harmonization efforts when two existing ontologies are merged: if the method is used on the union of two different corpora that underlined the creation of each ontology it will provide a joint view into the domain.

**Future work.** The presented extraction method is in an early stage of development which leaves plenty of room for future work. We wish to extend the method with more fine-grained extraction patterns to complement the current high coverage patterns. We also intend to exploit other syntactic information such as prepositional phrases. We can also enhance the step of ontology building. We learned from this work that compositionality is a strong characteristic of life science domains, however, our simplistic algorithm often produces invalid concepts. We plan to devise a more complex algorithm. Use of synonymy information during the conceptualisation step is another important development possibility. We would also like to concentrate on strategies that enrich the basic extracted ontology. For example, defining different views on a set of domain concepts or providing a set of patterns that act on the extracted semantic information ("semantic patterns"). Yet another development dimension is the use of external knowledge sources (such as WordNet for synonymy detection or WSDL files for input/ output information) to complement the small corpora. Further, the usability of the extraction tool could be enhanced by providing auxiliary information for each concept such as texts in the corpus where it appears and the frequency of its appearance. Finally, we wish to repeat our experiments in new domains and filter out extraction patterns that are generally valid for any web service description.

## 8. ACKNOWLEDGMENTS

We thank the members of the GATE group for their support in the development of the prototype. We also thank W. van Atteveldt, F. van Harmelen and H. Stuckenschmidt for their comments on earlier versions of this paper.

## 9. REFERENCES

[1] P.G. Baker, C.A. Goble, S. Bechhofer, N.W. Paton, R. Stevens, and A Brass. An Ontology for Bioinformatics Applications. *Bioinformatics*, 15(6):510–520, 1999.

[2] C. Brewster, H. Alani, S. Dasmahapatra, and Y. Wilks. Data Driven Ontology Evaluation. *LREC*, 2004.

[3] P. Buitelaar, S. Handschuh, and B. Magnini. ECAI Workshop on Ontology Learning and Population: Towards Evaluation of Text-based Methods in the Semantic Web and Knowledge Discovery Life Cycle. Valencia, Spain, August 2004.

[4] P. Buitelaar, D. Olejnik, and M. Sintek. A Protege Plug-In for Ontology Extraction from Text Based on Linguistic Analysis. *ESWS*, 2004.

[5] P. Cimiano, S. Staab, and J. Tane. Automatic acquisition of taxonomies from text: FCA meets NLP. *ECML/PKDD Workshop on Adaptive Text Extraction and Mining.* Cavtat–Dubrovnik, Croatia, 2003.

[6] The OWL-S Services Coalition. OWL-S: Semantic Markup for Web Services. White Paper. www.daml.org/services/owl-s/1.0/owl-s.pdf, 2003.

[7] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. GATE: A framework and graphical development environment for robust NLP tools and applications. *ACL*, 2002.

[8] H. Cunningham, D. Maynard, and V. Tablan. JAPE: a Java Annotation Patterns Engine (Second Edition). Research Memorandum CS–00–10, Department of Computer Science, University of Sheffield, 2000.

[9] R. Grishman and R. Kittredge, editors. *Analyzing Language in Restricted Domains: Sublanguage Description and Processing.* Lawrence Erlbaum Assoc., Hillsdale, NJ, 1986.

[10] A. Hess and N. Kushmerick. Machine Learning for Annotating Semantic Web Services. *AAAI Spring Symposium on Semantic Web Services*, March 2004.

[11] D. Lin. Dependency-based Evaluation of MINIPAR. *Workshop on the Evaluation of Parsing Systems, First International Conference on Language Resources and Evaluation*, Granada, Spain, May 1998.

[12] P. Lord, S. Bechhofer, M.D. Wilkinson, G. Schiltz, D. Gessler, D. Hull, C. Goble, and L. Stein. Applying Semantic Web Services to bioinformatics: Experiences gained, lessons learnt. *ISWC*, 2004.

[13] A. Maedche and S. Staab. Measuring similarity between ontologies. *EKAW*, 2002.

[14] A. Maedche and S. Staab. Ontology learning. In S. Staab and R. Studer, editors, *Handbook on Ontologies.* Springer, 2004.

[15] E. Motta, J. Domingue, L. Cabral, and M. Gaspari. IRS-II: A Framework and Infrastructure for Semantic Web Services. *ISWC*, 2003.

[16] R. Navigli and P. Velardi. Learning Domain Ontologies from Document Warehouses and Dedicated Websites. *Computational Linguistics*, 30(2), 2004.

[17] R. Navigli, P. Velardi, A. Cucchiarelli, and F. Neri. Quantitative and Qualitative Evaluation of the OntoLearn Ontology Learning System. *ECAI Workshop on Ontology Learning and Population*, 2004.

[18] P.V. Ogren, K.B. Cohen, G.K. Acquaah-Mensah, J. Eberlein, and L. Hunter. The Compositional Structure of Gene Ontology Terms. *Pacific Symposium on Biocomputing*, 2004.

[19] A. Patil, S. Oundhakar, A. Sheth, and K. Verma. METEOR-S Web service Annotation Framework. *WWW*, 2004.

[20] R. Porzel and R. Malaka. A Task-based Approach for Ontology Evaluation. *ECAI Workshop on Ontology Learning and Population*, Valencia, Spain, 2004.

[21] A.L. Rector and J.E. Rogers. Ontological issues in using a description logic to represent medical concepts: Experience from GALEN. *IMIA Working Group 6 Workshop*, November 1999.

[22] M.L. Reinberger and P. Spyns. Discovering Knowledge in Texts for the Learning of DOGMA-Inspired Ontologies. *ECAI Workshop on Ontology Learning and Population*, Valencia, Spain, August 2004.

[23] M. Sabou. From Software APIs to Web Service Ontologies: a Semi-Automatic Extraction Method. *ISWC*, Hiroshima, Japan, November 2004.

[24] C. Wroe, C. Goble, M. Greenwood, P. Lord, S. Miles, J. Papay, T. Payne, and L. Moreau. Automating Experiments Using Semantic Data on a Bioinformatics Grid. *IEEE Intelligent Systems*, 19(1):48–55, 2004.

[25] C. Wroe, R. Stevens, C. Goble, A. Roberts, and M. Greenwood. A Suite of DAML+OIL Ontologies to Describe Bioinformatics Web Services and Data. *Journal of Cooperative Information Science*, 2003.