Semi-automatic Discovery of Web Services Driven by User Requirements

María Pérez, Ismael Sanz, Rafael Berlanga, and María José Aramburu

Universitat Jaume I, Spain
{mcatalan,isanz,berlanga,aramburu}@uji.es

Abstract. Current research in domains such as the Life Sciences depends heavily on the integration of information coming from diverse sources, which are typically highly complex and heterogeneous, and usually require exploratory access. Web services are increasingly used as the preferred method for accessing and processing these sources. Due to the large number of available web services, the sheer complexity of the data and the frequent lack of documentation, discovering the most appropriate web service for a given task is a challenge for the user.

In this paper we propose a semi-automatic approach to assist the user in the discovery of which web services are the most appropriate to achieve her requirements. We describe the overall framework of our approach and we provide a detailed description of the techniques used in each phase of our approach. Finally, the usefulness of our approach is demonstrated through a Bioinformatics case study.

1 Introduction

In recent years, emergent fields such as the Life Sciences have experimented an exponential growth in the production of data and, consequently, a surge in the need for the creation of new techniques and technologies to analyze these data. This has caused an intense research effort in information integration techniques [9], which have to deal with a number of technical challenges. The first major challenge is the large number of available data sources (the latest reference on publicly available data sets in the Life Sciences [5] lists more than 1200 databases). A second problem is that these data sources have been developed by different institutions and, since there are few standards for data representation, there is a high level of data heterogeneity which causes serious impedance mismatch problems. Finally, the data sources are distributed, and typically available by using Web Services, which provide a limited API for data retrieval and exploration when compared with a full-fledged query language.

This situation creates challenging issues, given the typical process of research in Bioinformatics, which involves integrating data obtained by in-house experiments (for instance, DNA sequencing) with reference databases available on the Internet and queriable using a web page or a web service, and then performing analysis tasks using specific algorithms which may also be already available as web services [2]. To help researchers to find out publicly available services, some

P. García Bringas et al. (Eds.): DEXA 2010, Part I, LNCS 6261, pp. 62–75, 2010.

repositories such as BioCatalogue [1] have been developed; however, searches are hampered by poor documentation and the lack for a standard way of creating annotations. As a consequence, finding data collections and web services which are appropriate for a given research task usually becomes a costly trial-and-error process [11].

Currently, to the best of our knowledge, there is no guide to assist users in the discovery process. There are approaches that focus on the development of interfaces to assist in the location of web services; [10] presents a client engine for the automatic and dynamic development of service interfaces built on top of BioMOBY standard. Other approaches focus on the the discovery of web services that are annotated with a specific vocabulary. This is the case of the ^{my}Grid project[1], whose aim is to provide a controlled vocabulary to make annotations. They have implemented BioCatalogue [1], a Life Sciences web service registry with 1180 registered web services that are meant to be annotated using their Life Sciences ontology, the ^{my}Grid ontology, which should provide in principle a high precision in search. However, most available annotations are just free text, and many web services are not annotated at all. Another issue to be taken into account is that, in many cases, multiple services provide very similar functionality (a particularly insidious example is the multitude of services providing variants of alignments of genes and proteins). In this case, the user has to decide which one is the most appropriate based on diverse quality criteria (availability, coverage of the domain of interest, and so on). To address this problem, assessment techniques must be applied to provide the user with some information about the quality and the functionality of each service [3].

In the literature, web service discovery has been deeply studied, but mainly for traditional business-oriented applications [13]. In scientific domains, such as the Life Sciences, there are very significant differences in skills and expectations with respect to those in business domains. The user in the Life Sciences is a scientist that is assumed to be expert in a highly complex and specific domain, and can be assumed to know exactly what she wants, but may not be an expert in information processing technology – even though she is capable (and usually willing) to deal directly with the integration of the necessary web services. This explains the emergence of specific applications aimed at allowing scientists to design in-silico experiments by combining discovered web services into workflows [4]. Moreover, as [13] remarked, web services in traditional business-oriented applications are usually annotated with a signature (inputs, outputs and exceptions), web service states (preconditions and postconditions) and non-functional values (attributes used in the evaluation of the service), information that facilitates the discovery of web services and their composition. In contrast to these domains, as we have mentioned above, Life Science web services are poorly documented and therefore, it is not possible to apply traditional discovery and composition techniques.

In this paper, we contribute a semi-automatic approach to assist the user in web service discovery, looking for web services that are appropriate to fulfill the information requirements of researchers in the Life Sciences domain. Our aim

[1] `http://www.mygrid.org.uk`

is to make the whole process driven by well-captured requirements, in order to avoid the high costs associated with non-disciplined, non-reusable, ad-hoc development of integration applications. It is important to note that the approach is semi-automatic by design; as we have said before, the user is a scientific expert on the domain which knows the analysis objectives and the steps to attain them.

The remainder of this paper is structured as follows. First, Section 2 presents an overview of the proposed approach. Next, Section 3 focuses on the requirements elicitation and specification phase. Section 4 describes the normalization phase and in Section 5 the web service discovery phase is explained. Then, in Section 6 we present a Bioinformatics case study that showcases the usefulness of our approach, and finally in Section 7, some conclusions and future research lines are summarized.

2 Approach Overview

The overall approach we propose to assist the discovery of web-services based on the users requirements is shown in Figure 1. It consists of three main phases:

1. **Requirements elicitation and specification.** Scientists have information requirements that must be fulfilled by information that is stored in any of the multiple available data sources or even information that is the result of processing data stored in those data sources.

   The main purpose of this phase is to gather the user information requirements and to specify them in a formal way. The user determines her information needs and moreover, gives extra information about her experience and some knowledge describing the steps necessary to do it herself manually. The description of these steps is also included in the formal specification. Section 3 describes the techniques used in this phase.

2. **Normalization.** In the Requirements model, task descriptions are expressed in natural language, and therefore they must be normalized in order to be automatically processed. The normalization consists of a semantic annotation process in which the information of the requirements model, concretely the description of the tasks, is processed and annotated with semantic data. The normalization is carried out in two phases: (i) domain specific annotations, and (ii) application specific annotations. Section 4 provides a more detailed description of the normalization process.

3. **Web services discovery.** The aim of this step is to discover suitable web services that provide the functionality specified by the user-defined tasks in order to fulfill the user requirements. The discovery is based on the annotations made in the previous phase, so it depends largely on the quality of these annotations. In this paper, we focus only on the discovery of web services, but in the future we aim to discover other types of resources such as "naked" databases. Frequently, there is more than one web service providing the same functionality, and therefore the result of this phase is a set of web services per each user-defined task. Section 5 explains the discovery process.

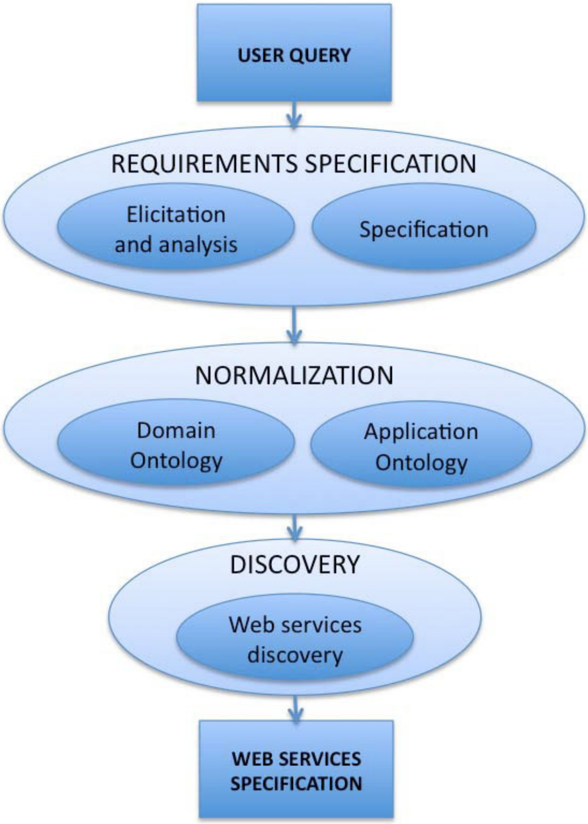


Fig. 1. The phases of the proposed guide

At the end of the process, the user receives as output some sets of web services, one per each specified task, that provide the functionality required by the tasks. The user will then have to select the ones that are the most appropriate for her requirements. In order to be able to decide which are the best ones, which is a complex process outside the scope of this paper, the user may need some assessment techniques. In case the results are not those expected by the user, she can refine the process at the three phases of the guide: (i) Requirements phase: the requirements model is refined by modifying the goals or the tasks, (ii) Normalization phase: another category can be selected to annotate a specific task, or (iii) Web Services discovery phase: a different web service can be selected in the list of discovered services. This process is iterative and it can be executed as many times as required to refine the selection.

3 Requirements Specification

The requirements elicitation and specification process plays a crucial role in the discovery of web services, since the selection is basically driven by the user's information requirements. These requirements must be gathered and formally specified to be the input of the discovery process. The requirements elicitation task is made by the system designer by having personal interviews with the end user. The user determines which information she needs and, moreover, gives extra information about her experience and knowledge describing the steps necessary to do it herself manually. All the provided information is completely independent of the characteristics of web services or other resources. The elicited requirements are analyzed by the system designer in order to detect inconsistencies or missing information. This analysis is made by querying domain ontologies and by interacting with the user.

Once the requirements have been elicited and analyzed, the system designer creates a formal specification of them, called the *Requirements model*, in order to be used in an automatic way in the subsequent phases. This specification is made using the *i** formalism [16,17], which is both a goal-oriented and an agent-oriented language. We use this framework because it provides the functionality required to obtain a formal specification of the user's requirements without taking into account the characteristics of the system. The goals and the tasks of the *Strategic Rationale* (SR) model of the *i** framework capture the user's information requirements and the steps, specified by the user, to achieve those requirements. Here, we generalize the techniques used in [12] to specify the user's requirements in the context of finding appropriate similarity measures for complex information.

4 Normalization

At this stage, the system designer has already gathered the user's information needs and she has specified them in the Requirements model. In this model the information about the tasks is described in natural language, which is hard to process in an automatic way. Therefore, the purpose of this phase is to annotate the tasks descriptions with domain and application ontologies to allow the reconciliation of the requirements of the user with the sources of information (web services) in a process that can be considered as a *normalization* of knowledge. In our case, it is only necessary to normalize the task descriptions, since they will be used to guide the discovery of web services. Among the different techniques to normalize natural language sentences we have chosen to use *semantic annotations*. Our aim is to select the most relevant terms in the task descriptions and use these terms to search for web services. We have divided the semantic annotation process in two steps:

1. **Domain specific annotations.** This step consists of identifying the terms of the task descriptions that are related to the domain in which we are working. For each task, the most relevant terms are retrieved by querying domain ontologies. In our domain, the Life Sciences domain, there are several domain ontologies that can be used to annotate semantically. One example of domain ontology is the UMLS, a metathesaurus that comprises the main biomedical

linguistic resources widely used in health care applications. Another example of Life Sciences ontology is ^{my}Grid ontology [15], which describes the Life Sciences research domain and data related to web services. ^{my}Grid ontology has been implemented in ^{my}Grid project and is aimed to provide a formal vocabulary to describe the web services. In some cases, the user can also decide that she prefers not to use any domain ontology, that is, to use the whole task descriptions expressed in free-text in the next step for tasks classification.

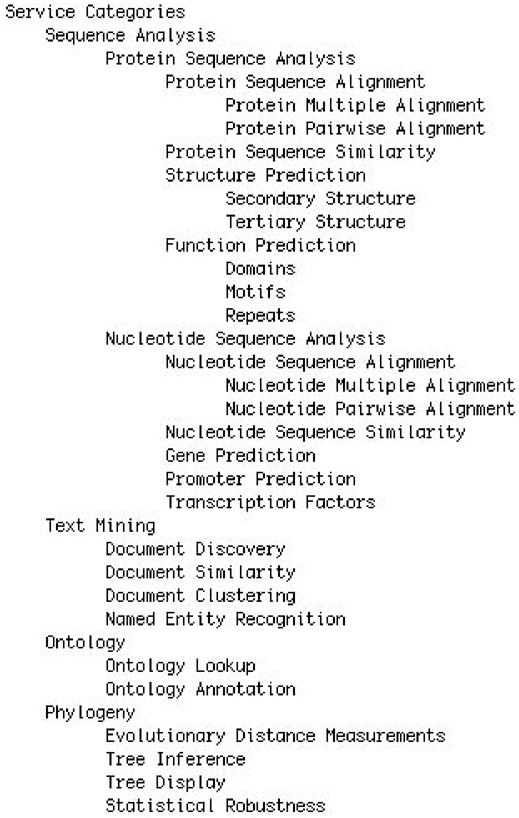
2. **Application specific annotations.** Once the terms related to the domain have been detected, the next step is to use them to query the application ontology in order to determine the type of each task. As we are searching for Life Sciences web services, we apply the taxonomy of categories used by BioCatalogue in order to classify the user-defined tasks. BioCatalogue is a registry of curated Life Science Web Services, whose aim is to provide an easy way for scientists to discover web services of interest. BioCatalogue has a shallow taxonomy of web services categories, and most of the registered web services have at least one category. So, we use this taxonomy to classify the user-defined tasks with the aim of using these categories in the discovery of the web services that are suitable to fulfill the user's requirements. Figure 2 illustrates a fragment of the categories taxonomy of BioCatalogue.

   The final result of this process is a ranked list of annotations for each task. By default, each task is annotated with the category that has the highest similarity score. These annotations have to be validated by the user.

5 Web Service Discovery

Web services in BioCatalogue have four main different types of annotations: descriptions, operations, tags and categories. As a first approximation, we have decided to use categories as the criterion for the discovery process, since they are formally described by a taxonomy and, moreover, they express very well the functionality of the services. For the future, we are planning to generalize this approach by applying techniques for the combination of several types of annotations as criterion for the discovery process, which would result in a more flexible approach [6].

In the Normalization phase, the tasks defined by the user have been annotated with the categories used in BioCatalogue in order to determine the functionality described by the tasks. The discovery process consists in querying BioCatalogue, using its recent launched API, as many times as tasks in the requirements model are. Each query searches for a specific category and retrieves a set of web services that are annotated with this category, that is, web services that are supposed to execute the functionality required by the task. In most cases, the search retrieves more than one service, since there are many services annotated with the same category. In this set of services, maybe some of them do not provide exactly the required functionality and this may be because some categories are too general to describe a specific functionality. So, it is the responsibility of the user to know



```
Service Categories
    Sequence Analysis
          Protein Sequence Analysis
                Protein Sequence Alignment
                      Protein Multiple Alignment
                      Protein Pairwise Alignment
                Protein Sequence Similarity
                Structure Prediction
                      Secondary Structure
                      Tertiary Structure
                Function Prediction
                      Domains
                      Motifs
                      Repeats
          Nucleotide Sequence Analysis
                Nucleotide Sequence Alignment
                      Nucleotide Multiple Alignment
                      Nucleotide Pairwise Alignment
                Nucleotide Sequence Similarity
                Gene Prediction
                Promoter Prediction
                Transcription Factors
    Text Mining
          Document Discovery
          Document Similarity
          Document Clustering
          Named Entity Recognition
    Ontology
          Ontology Lookup
          Ontology Annotation
    Phylogeny
          Evolutionary Distance Measurements
          Tree Inference
          Tree Display
          Statistical Robustness
```

Fig. 2. Fragment of the BioCatalogue Taxonomy

which services are suitable for the tasks and which are not. We are currently working on assessment techniques that could assist the user in the web services selection.

6 Case Study

To prove the usefulness of our approach, a prototype web services discovery guide has been implemented by using the Eclipse EMF modelling framework[2]. The necessary transformations between models have been defined in the ATL[7] language. In this section, we use this prototype to develop a bioinformatics case study extracted from [8]. In this way, we can illustrate how to use our approach to guide the user in a real web service discovery task.

The case study concerns biological research that analyzes the presence of specific genes involved in the genesis of Parkinson's Disease, called LRRK2 genes, in different organisms, in order to know more about the biochemical and cellular functions of these genes. The author studies the presence of the LRRK2 genes in

[2] `http://www.eclipse.org/modeling/emf/`

the organism *N. vectensis*, since previous studies have shown that this organism is a key organism to trace the origin of these genes. The author describes the process step-by-step.We have selected this case study because it describes with detail the techniques used in every step, and it could be useful to validate our guide. However, our intention is not to model a concrete case study, but to offer a guide for more general cases.

Next, there is a short description of each one of the steps of our approach in order to discover the web services that provide the functionality required by the scientist.

1. **Requirements elicitation and specification.** In this first phase the purpose is to gather the user's requirements through a personal interview. The system designer obtains as much information as possible, gathering the user's information requirements and the steps the user would make if she had to make the search herself manually. In this case study the user information requirement is to obtain a comparison of the LRRK2 genes in different organisms. The steps the user would make herself manually are: (*i*) to retrieve the protein sequences of the different domains of the gene in different databases; (*ii*) to predict the gene structure automatically combining the sequences retrieved in (*i*); (*iii*) to align protein sequences to build phylogenetic trees; (*iv*) to build the phylogenetic trees; and (*v*) to analyze the structure of the proteins.

   Figure 3 shows the requirements model obtained as a result of this step and includes all the information elicited by the system designer.

2. **Normalization.** Once the system designer has elaborated the requirements model, the next step is to normalize the description of the user-defined tasks. Next we describe each one of the two steps of the normalization.

   2.1. **Domain specific annotations.** This step consists in selecting the terms relevant to the domain. In this case study, we present both the results of annotating with the ^{my}Grid ontology (OWL ontology) and the results of not using any domain ontology.
   The experiments we have made until now suggest that extracting the most relevant terms is only required when the task descriptions are too verbose, and not when the descriptions are short and simple sentences.
   Table 1 shows a ranked list of concepts of ^{my}Grid ontology similar to the task description terms, in this case *Retrieve protein sequences.* The matching has been made with ISub [14], a string metric for the comparison of names used on the process of ontology alignment and on other matching problems. Each matching has a score that indicates the similarity between the description and the concepts of the ontology.

   In this example the task description is clear and simple, but the matching with ^{my}Grid ontology does not obtain good results. For instance, the top ranked concept has a very low score, 0.114, and it corresponds to the ontology concept *protein sequence id*, which is not very similar to

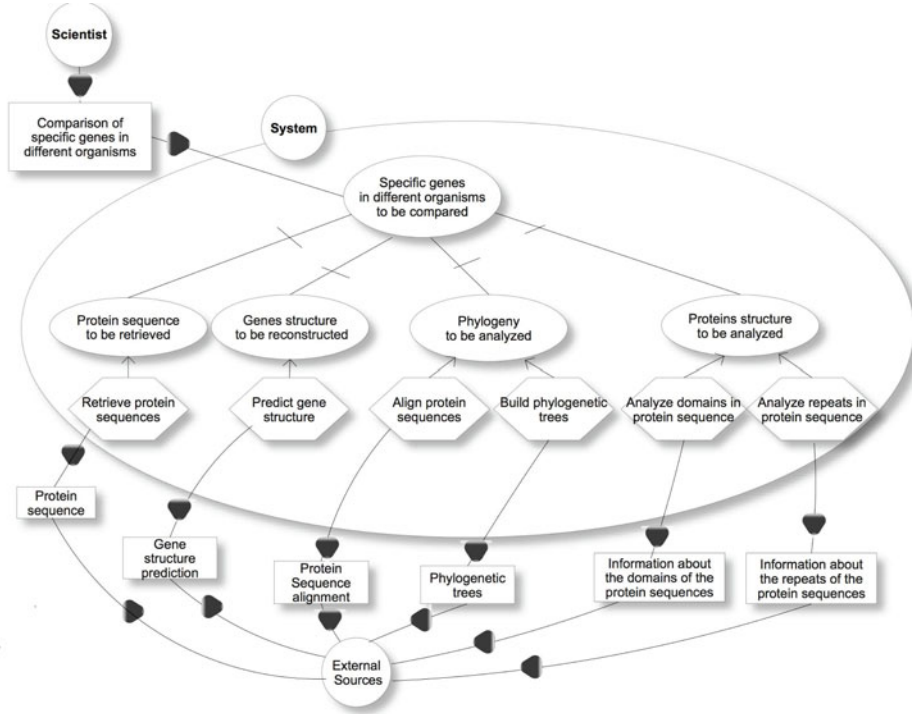

Scientist
Comparison of specific genes in different organisms
System
Specific genes in different organisms to be compared
Protein sequence to be retrieved
Genes structure to be reconstructed
Phylogeny to be analyzed
Proteins structure to be analyzed
Retrieve protein sequences
Predict gene structure
Align protein sequences
Build phylogenetic trees
Analyze domains in protein sequence
Analyze repeats in protein sequence
Protein sequence
Gene structure prediction
Protein Sequence alignment
Phylogenetic trees
Information about the domains of the protein sequences
Information about the repeats of the protein sequences
External Sources

Fig. 3. Requirements model of the case study

Retrieve protein sequences. The scores are low because the sentences of this case do not contain meaningless words and, due to them, the user has decided to classify the tasks without using any domain ontology, that is, using all the words in the task description.

Table 1. Matching between the *Retrieve Protein Sequences* and ^{my}Grid ontology

Concepts	Score
protein_sequence_id	0.11428571428571432
protein_sequence_record	0.08706390861376972
protein_sequence_database	0.07459572248304647

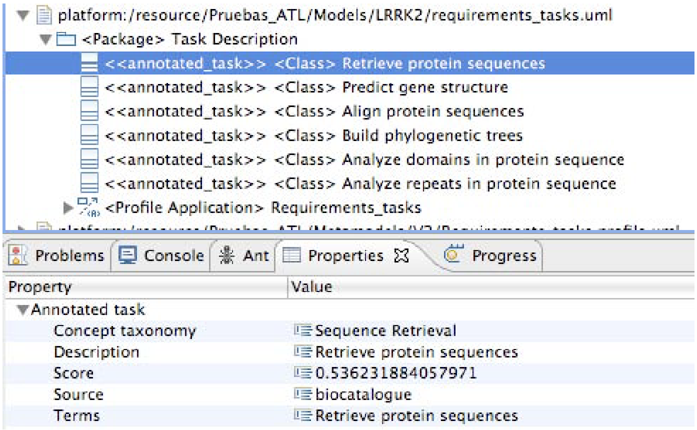
2.2. Application specific annotations. In this step the tasks are classified based on their functionality. This classification is based on the BioCatalogue taxonomy of categories. The matching is not exact, so a ranked list of categories per each task is provided to the system designer. Figure 2 shows a fragment of the web services categories taxonomy of BioCatalogue used to annotate the task descriptions. Table 2 shows the ranked list of the task *Retrieve protein sequences*; this matching has been made also using the ISub metric and, all the terms of the task description have participated in the matching. The category *Sequence retrieval*, which is the one with the highest similarity score, is automatically selected. In case the user does not agree with the selected category, she could select any other category in the ranked list.

Figure 4 shows a screenshot of a fragment of the model produced by our prototype that contains all the tasks annotated with BioCatalogue categories. This fragment shows the details of the annotated task *Retrieve protein sequences*. The task *Retrieve protein sequences* has been annotated with the taxonomy concept *Sequence retrieval* extracted from the source *BioCatalogue* with a score of 0.53652 and the terms used to query the application taxonomy have been *Retrieve protein sequences.*

3. **Web Service discovery.** The web service discovery process is carried out by searching in the BioCatalogue registry services that are annotated with the same category as the user-defined tasks. This search retrieves a set of web services per each user-defined task, and there is no way to know in advance which one is the most appropriate for the user-defined task. So, at this step, the information of each retrieved web service is stored in a model, called *Service model*, which contains a component for each retrieved web service. Each component stores information about the categories, tags, type and WSDL location of the web service.This component is extensible to other required attributes of Web services.

Table 2. Ranked list of categories for the task *Retrieve Protein Sequences*

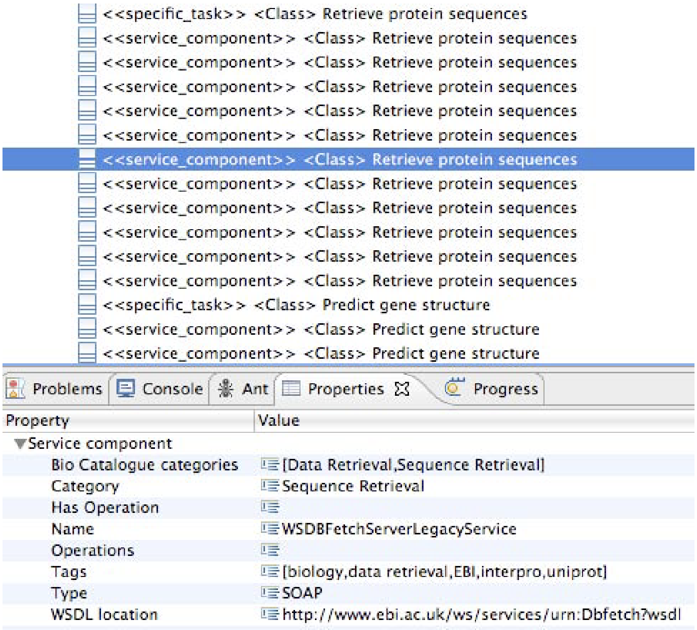
Categories	Score
Sequence Retrieval	0.536231884057971
Sequence Similarity	0.42857142857142860
Sequence Analysis	0.39468690702087295
Sequence Alignment	0.3758921490880254



```
▼ platform:/resource/Pruebas_ATL/Models/LRRK2/requirements_tasks.uml
  ▼ <Package> Task Description
      <<annotated_task>> <Class> Retrieve protein sequences
      <<annotated_task>> <Class> Predict gene structure
      <<annotated_task>> <Class> Align protein sequences
      <<annotated_task>> <Class> Build phylogenetic trees
      <<annotated_task>> <Class> Analyze domains in protein sequence
      <<annotated_task>> <Class> Analyze repeats in protein sequence
    ▶ <Profile Application> Requirements_tasks
```

Property	Value
▼Annotated task	
Concept taxonomy	Sequence Retrieval
Description	Retrieve protein sequences
Score	0.536231884057971
Source	biocatalogue
Terms	Retrieve protein sequences

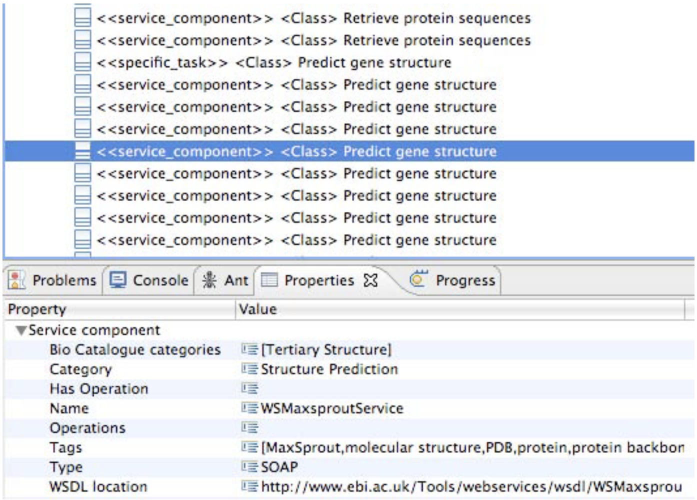
Fig. 4. Normalization of the task *Retrieve protein sequences*



<<specific_task>> <Class> Retrieve protein sequences
<<service_component>> <Class> Retrieve protein sequences
<<service_component>> <Class> Retrieve protein sequences
<<service_component>> <Class> Retrieve protein sequences
<<service_component>> <Class> Retrieve protein sequences
<<service_component>> <Class> Retrieve protein sequences
<<service_component>> <Class> Retrieve protein sequences
<<service_component>> <Class> Retrieve protein sequences
<<service_component>> <Class> Retrieve protein sequences
<<service_component>> <Class> Retrieve protein sequences
<<service_component>> <Class> Retrieve protein sequences
<<service_component>> <Class> Retrieve protein sequences
<<specific_task>> <Class> Predict gene structure
<<service_component>> <Class> Predict gene structure
<<service_component>> <Class> Predict gene structure

Property	Value
▼Service component	
Bio Catalogue categories	[Data Retrieval,Sequence Retrieval]
Category	Sequence Retrieval
Has Operation	
Name	WSDBFetchServerLegacyService
Operations	
Tags	[biology,data retrieval,EBI,interpro,uniprot]
Type	SOAP
WSDL location	http://www.ebi.ac.uk/ws/services/urn:Dbfetch?wsdl

Fig. 5. *WSDBFetchServerLegacyServer* service component. Candidate for the task Retrieve protein sequence.



<<service_component>> <Class> Retrieve protein sequences
<<service_component>> <Class> Retrieve protein sequences
<<specific_task>> <Class> Predict gene structure
<<service_component>> <Class> Predict gene structure
<<service_component>> <Class> Predict gene structure
<<service_component>> <Class> Predict gene structure
<<service_component>> <Class> Predict gene structure
<<service_component>> <Class> Predict gene structure
<<service_component>> <Class> Predict gene structure
<<service_component>> <Class> Predict gene structure
<<service_component>> <Class> Predict gene structure

Property	Value
▼Service component	
Bio Catalogue categories	[Tertiary Structure]
Category	Structure Prediction
Has Operation	
Name	WSMaxsproutService
Operations	
Tags	[MaxSprout,molecular structure,PDB,protein,protein backbon
Type	SOAP
WSDL location	http://www.ebi.ac.uk/Tools/webservices/wsdl/WSMaxsprou

Fig. 6. *WSMaxsproutService* service component. Candidate for the task Predict gene structure.

Figure 5 shows a fragment of the Service model produced by our prototype. In this fragment it can be seen that each user-defined task has a set of discovered web services. In the figure it is shown the information of the web service *WSDBFetchServerLegacyServer* that is a SOAP service whose location is http://www.ebi.ac.uk/ws/services/urn:Dbfetch?wsdl and it is annotated with the tags: *biology, data retrieval, EBI, interpro, uniprot*, and its categories in BioCatalogue are *Data Retrieval* and *Sequence Retrieval.*

Figure 6 shows the information of the web service *WSMaxsproutService* whose category, *Tertiary Structure*, is not the category of the user-defined task, but it is a specialization of it.

This model is provided to the user who has to decide which web service is the most appropriate for her information requirements.

7 Conclusions and Future Work

The approach presented in this paper is focused on domains where applications have to deal with distributed and highly heterogeneous data sources, in which there are few standards for data representation and usually they require an exploratory access. In these domains many web services have been implemented, but due to their distribution and the frequent lack of documentation, they are not easily discovered by potential users. Life Sciences is an example of this type of domain, since many biological data have been released in recent years and, in consequence, many techniques and applications have been implemented to manage that data.

In this paper we have presented a semi-automatic approach to guide the user in the discovery of the appropriate techniques required to fulfill their information needs. We have mainly focused on web services, due to their wide acceptance and popularity within the Life Sciences domain. So, the aim of our approach is to assist the user in the discovery of the web services that provide the necessary functionality to fulfill her information requirements.

Considering the type of end-user the approach is addressed to, an expert on the domain with knowledge and experience to recognise which web services are the most appropriate, one of the main benefits of the approach is that it is a semi-automatic process. In this way, the user is able to change parameters, annotations or automatic selections in each one of the phases in the way she thinks is better, based on her own knowledge and experience or previous results. Thus, we aim to design a process of exploratory search, advising the user in each step, and taking advantage of her previous knowledge.

Currently, we are working on web service assessment techniques. The purpose of this assessment is to assist the user in the web service selection providing her with a set of measures to select the most appropriate among all the discovered services. We are working not only on general QoWS measures, but also on measures that analyze the quality of web services with respect to the user's requirements, for example, measures that analyze the type and the semantics of the results retrieved by a web service, the relevance of the web service with respect to the other user-defined tasks, etc.

Another important issue is the data sources discovery. Nowadays, we are only focusing on web services but in a near future, we also aim to search for data sources that may contain the required information. Their discovery will be based on techniques similar to the ones used for web services discovery and we expect that the main differences will appear in the types of measures necessary to assess the data sources.

Finally, we aim to develop an end-user tool to assist the user during all the process, since her requirements specification to the validation of the results, in order to avoid the presence of a system designer expert.

Acknowledgements

This research has been supported by the Spanish Ministry of Education and Science (grant TIN2008-01825/TIN) and by Universitat Jaume I – Fundació Bancaixa (grant P11B2008-43). María Pérez has been supported by Universitat Jaume I predoctoral grant PREDOC/2007/41.

References

1. Belhajjame, K., Goble, C., Tanoh, F., Bhagat, J., Wolstencroft, K., Stevens, R., Nzuobontane, E., McWilliam, H., Laurent, T., Lopez, R.: BioCatalogue: A Curated Web Service Registry for the Life Science Community. In: Microsoft eScience Conference (2008)
2. Burgun, A., Bodenreider, O.: Accessing and integrating data and knowledge for biomedical research. Med. Inform. Yearb. 2008, 91–101 (2008)
3. Cardoso, J., Sheth, A.P., Miller, J.A., Arnold, J., Kochut, K.: Quality of service for workflows and web service processes. Web Sem. 1(3), 281–308 (2004)
4. Stevens, R., Goble, C., Pocock, M., Li, P., Hull, D., Wolstencroft, K., Oinn, T.: Taverna: a tool for building and running workflows of services. Nucleic Acids Research 34(Web Server issue), 729–732 (2006)
5. Cochrane, G.R., Galperin, M.Y.: The 2010 Nucleic Acids Research Database Issue and online Database Collection: a community of data resources. Nucleic Acids Research 38, 1–4 (2010)
6. Prez, M., Sanz, I., Berlanga, R.: Measure selection in multi-similarity xml applications. In: 3rd International Workshop on Flexible Database and Information System Technology, FlexDBIST-08 (2008)
7. Jouault, F., Kurtev, I.: Transforming models with atl. In: Bruel, J.-M. (ed.) MoDELS 2005. LNCS, vol. 3844, pp. 128–138. Springer, Heidelberg (2006)
8. Marín, I.: Ancient origin of the Parkinson disease gene LRRK2. Journal of Molecular Evolution 64, 41–50 (2008)
9. Mesiti, M., Jiménez-Ruiz, E., Sanz, I., Berlanga, R., Valentini, G., Perlasca, P., Manset, D.: Data integration issues and opportunities in biological XML data management. In: Open and Novel Issues in XML Database Applications: Future Directions and Advanced Technologies. IGI Global (2009)
10. Navas-Delgado, I., Rojano-Muñoz, M., Ramírez, S., Pérez, A.J., León, E.A., Aldana-Montes, J.F., Trelles, O.: Intelligent client for integrating bioinformatics services. Bioinformatics 22(1), 106–111 (2006)

11. Anderson, N.R., Lee, E.S., Brockenbrough, J.S., Minie, M.E., Fuller, S., Brinkley, J., Tarczy-Hornoch, P.: Issues in biomedical research data management and analysis: needs and barriers. J. Am. Med. Inform. Assoc. 14(4), 478–488 (2007)
12. Pérez, M., Casteleyn, S., Sanz, I., Aramburu, M.J.: Requirements gathering in a model-based approach for the design of multi-similarity systems. In: MoSE+DQS '09: Proceeding of the First International Workshop on Model Driven Service Engineering and Data Quality and Security, pp. 45–52. ACM, New York (2009)
13. Rao, J., Su, X.: A survey of automated web service composition methods. In: Cardoso, J., Sheth, A.P. (eds.) SWSWPC 2004. LNCS, vol. 3387, pp. 43–54. Springer, Heidelberg (2005)
14. Stoilos, G., Stamou, G. B., Kollias, S.D.: A string metric for ontology alignment. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 624–637. Springer, Heidelberg (2005)
15. Wolstencroft, K., Alper, P., Hull, D., Wroe, C., Lord, P.W., Stevens, R.D., Goble, C.A.: The mygrid ontology: bioinformatics service discovery. Int. J. Bioinformatics Res. Appl. 3(3), 303–325 (2007)
16. Yu, E.: Modelling Strategic Relationships for Process Reenginering. PhD thesis, University of Toronto, Canada (1995)
17. Yu, E.: Towards modelling and reasoning support for early-phase requirements engineering. In: RE 1997, vol. 85, pp. 2444–2448 (1997)