Tackling Complex Natural Language Queries on the Web Contents

Marta Gatius and Horacio Rodríguez

Software Department, Technical University of Catalonia

Abstract. Interfaces tackling complex queries about information on the Web deal with the problem of processing information obtained from different Web sites. This paper discusses a solution to this problem: a domain specific ontology and a wrapper system providing content-based information to access the Web. The presentation of the contents of the Web sources accessed may differ (different words, different units of measure,...). The wrapper system, using an explicit description of the HTML pages to be analyzed, extracts the information and represents it in the ontology. The ontology provides inferential and reasoning capabilities to solve complex requests. The design proposed has been used in a Natural Language Interface system supporting communication about the contents of a collection of domain-restricted Web sites.

1 Introduction

The fast growth of Internet has made available a large amount of information and services. Conventional information seeking technologies suffer from several drawbacks when applied directly to access information on the Web. The main reasons are the following:

1. The information is usually placed in heterogeneous sources with different ways of accessing.
2. Web pages change rapidly (in presentation and accessing).
3. Most Web sites are designed to facilitate browsing and not querying.
4. Many Web pages are generated on the fly, based on user request.

Due to these problems, a lot of systems helping the user accessing Web pages and services have been built. Most of these systems were designed to solve the problem of locating Web sites containing specific information([10]). Nevertheless, the number of systems dealing with the more complex problem of extracting and processing Web contents is also increasing. Several approaches have been followed under different paradigms: Information Extraction ([5]), Information Integration ([9],[12]), Question Answering ([2], [15]) and Semantic Web ([16]).

This paper deals with a complementary task: tackling complex Natural Language (NL) queries about the contents of a collection of domain-restricted Web sources. To achieve this goal the information extracted from several Web sites must be integrated and processed. We propose the use of a domain specific conceptual ontology (CO) to mediate between query and answer. The CO provides the commonsense knowledge required to answer user queries. For example, in the travelling domain, questions using the word *transport*, such as *which transports connect Barcelona to Madrid ?* require information placed in different Web sites, each site containing data of a different type of transport (train, bus,...). In the CO, each type of transport would be represented as a subclass of the class *transport*. The information extracted from the Web would be represented as instances of these subclasses. The inferences and reasoning required to solve complex requests would be done over these instances.

Information in the Internet is obtained by means of wrappers. In the Web environment a wrapper can be defined as a processor that converts information implicitly stored as in a HTML document into information explicitly stored as a data structure for further processing. A wrapper system has been designed to extract the contents of different types of Web sources, such as corporation Intranets, public Intranets, personal pages, etc.

Our system has been used in a Natural Language Interface (NLI) system supporting communication about the contents of a collection of domain-restricted Web sites. NL mode seems especially appropriate to achieve friendly and intelligent interaction ([13]). It is easier to use NL to express complex queries than to use other modes of communication. NLI technology has been widely used for accessing to highly structured knowledge sources, such as databases. However, the incorporation of NLI technology for accessing other types of sources has been quite slow. The main problems NLIs present is the large run-time requirements and the users lack knowledge about the limits of the conceptual/linguistic coverage of the interface. These problems can be overcome in domain-restricted NLIs that incorporate a user guiding. By restricting the language to the domain and by guiding the users about the NL the system accepts, mistakes and misunderstandings in the communication are avoided and the run-time requirements are reduced. For all these reasons, the proposal discussed in this paper was incorporated in a system that generates domain-restricted NLIs (described in [7]and [8]). The system has been applied to allow the users to access different Web sites in Spanish to a couple of domains: travelling and university courses.

A description of the CO is given in Sect. 2. Section 3 describes the wrapper system. The NLI system to the Web contents is described in Sect. 4.

2 The Conceptual Ontology

The CO is the core of this proposal. The CO is designed in a frame-based fashion where basic entities are concepts, attributes describing concepts and operations over these concepts. These three entities are represented in separated taxonomies.

The CO is organized at two levels: the general level, describing the conceptual knowledge common to all domains and the domain level, describing a particular domain. Two sublevels were distinguished on the domain level: the description level and the case level. The description level represents all concepts and attributes involved in the communication for a particular domain as well as all possible operations on those concepts. The concepts and attributes as well as the specific operations required for a domain are defined as subclasses of those in the general level. The case level describes all information about the specific cases (instances) obtained from the Web during the communication process. Concepts are described by a set of attributes. The description of a concept can also incorporate the addresses of Web sites containing particular information about the concept. In case one address represents a request to a Web service[1], the parameters required are also specified.

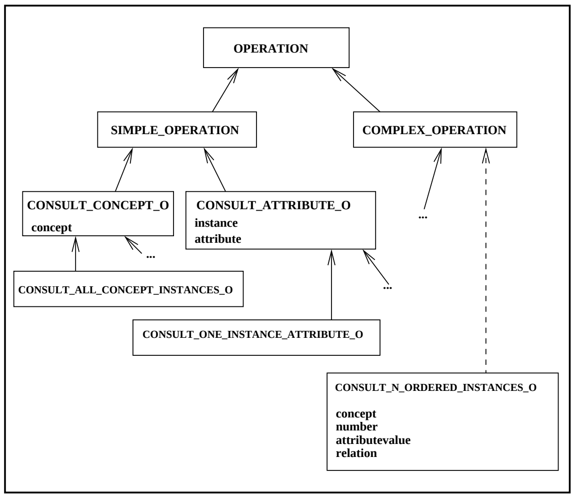

Fig. 1. A fragment of the taxonomy of operations

The operations necessary to answer user questions must be represented in the taxonomy of operations in the CO. There is a basic set of operations of different arity in the CO general level. In the operation description, arguments and preconditions are represented as slots. Subclasses of operations inherit the arguments and preconditions from the upper classes. Semantics of the operations are given by constraining the arguments and preconditions inherited. Operations are classified as simple or complex, as shown in Fig. 1. There are two subclasses of simple operations: *consult_concept_o*, representing the operations querying for the instances of a conceptual class and *consult_attribute_o*, representing those obtaining the values of one or more attributes of a conceptual instance. All operations belonging to the first class include the argument *concept*. Operations in the second class include the arguments *instance* and *attribute*.

Complex operations involve several instances. Complex operations were incorporated to enhance the functionality of the system. Figure 1 shows the com-

[1] What is sometimes called the hidden Internet

plex operation *consult_n_ordered_instances_o*, obtaining all ordered groups of *n* instances of a particular concept satisfying a set of specific constraints. The arguments of the operation *consult_n_ordered_instances_o* represent the concept, the number of instances in the group, the properties of the instances and the relations between the instances in the group. If specialized tasks for a domain are necessary, they must be incorporated in the taxonomy of operations.

Consider, for example, the process of tuning the CO to process requests about trains and buses schedules. The model defined for this domain covers the knowledge required for supporting user queries about information on railways and buses described in the Web. This knowledge has been obtained from the study of a corpus of dialogues registered in the telephone service of *RENFE* (Spanish public Railway Company) as well as the information provided by the Web services of two different companies: *RENFE*[2] and a private company of buses. Figure 2 shows a fragment of a page provided in the Web service of *RENFE*. Given a departure city and an arrival city this service provides a page containing all trains connecting directly the two cities.

File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print
Bookmarks Location: http://horarios.renfe.es/hir
What's Related
RENFE
Horarios y Precios

Fig. 2. A fragment of the RENFE Web page

A few concepts are enough for describing the domain. Most of users queries are about the schedule of a particular train or bus. In user interventions, a transport is usually described by its departure and final stations and, optionally, by other related information as its frequency and date (i.e. *Trains from Barcelona to Madrid on Mondays*). All this information has been incorporated in the description of the concept *transport*. This concept is described by a set of attributes: *departure* , *arrival, timedeparture, timearrival, date, frequency* and *price*. The concept *transport* is the core of the application and has two subclasses:

[2] http://www.renfe.es

the concept *train* and the concept *bus*. The addresses of the Web sites containing information about these concepts were also incorporated into their description.

The operation *consult_connections_o*, obtaining connected transports has been incorporated for this domain . This operation is shown in Fig. 3. It is defined as a subclass of the complex operation *consult_n_ordered_instances_o*, shown in Fig. 1. In the operation *consult_connections_o* several of the arguments of its upper class are restricted to specific values. The value of the argument representing the concept is *transport*. The number of instances in the group is *2*. The value of the argument representing the properties of the instances in the group states that the first transport must leave from the station represented by the variable *Departure* and arrive to a connection station and the second transport must leave from the connection station and arrive to the station represented by the variable *Arrival*. In order to achieve an efficient performance, the connection stations are restricted to those classified as connections in the area(s) of departure and arrival. This constraint is expressed by using the basic predicates *or* and *hasvalue* to represent the value of the attribute *arrival* of the first transport in the group. Finally, the value of the argument describing relations between instances (represented in the slot *relation*), states that the first transport in the group must arrive before the second transport leaves. Relations and predicates are represented in a Prolog like formalism.

```
CONSULT_CONNECTIONS_O

isa: CONSULT_N_ORDERED_INSTANCES_O
concept: transport
number: 2
attributevalue: [[[departure,Departure],
                  [arrival,[pred,Value,[[or,[hasvalue,Departure,area,Area],
                                             [hasvalue,Arrival,area,Area]]
                                         [hasvalue,Area,connecity,Value]]]]

                   [[departure,Value],[arrival,Arrival]]]
relation: [[before,timearrival,timedeparture]]
```

Fig. 3. The operation *consult_connections_o*

The performance of the operation *consult_connections_o* would require accessing to the Web services providing information on trains and buses, the two subclasses of the *transport* class. The data extracted will be represented as instances of the classes *train* and *bus*. The results of this operation would include combinations of two trains, combinations of two buses and combinations of one train and one bus.

3 The Wrappers System

The set of Web sites that would be consulted during communication are selected when adapting the system to a specific domain. Their content is accessed during communication through their corresponding wrappers. Although new languages have been developed to write Web documents in a more structured from (XML, RDF,...), facilitating data access and integration, most Web pages are written in HTML. Wrappers translate these HTML sources to other representation formalisms. Several approaches are being proposed to reduce the cost of implementing wrappers: special languages for writing wrappers ([3]), semi-automated tools ([1],[6]), automatic wrappers generation ([4], [11], [14]).

Any wrapper expressive enough for supporting the operations placed in the CO can be used in this proposal. We have designed and implemented a simple wrapper system that uses an explicit description of the HTML sources to be analyzed. The wrapper system provides a special language for describing Web pages. The description of a Web page consists of three parts:

1. The organization of the page.
2. The textual processing to be done over the data to be extracted.
3. The representation of the resulting data in the CO.

In current implementation, only semi-structured Web pages are considered. The wrapper represents the HTML page code as an HTML syntax tree where nodes are labeled by tag names. The children of each node consists of either other nodes, text or links. Then, following the description of the page, the wrapper traverses the HTML parsed tree to extract the information required. This information, once obtained, is textually processed and represented in the CO.

Several wrapper classes have been implemented for dealing with different extraction tasks. The description of the Web page has to be provided to a class of wrappers allowing, thus, the instantiation of a specific wrapper. The most representative of these classes is described briefly. Wrappers belonging to this class are in charge of parsing a wide range of semi-structured pages where information is represented as lists (or tuples) of attribute values. Wrappers in this class use the HTML tags delimiting the attributes to localize the information to be extracted. Additional information, such as tags delimiting tables and tuples containing the attribute values can also be used. This class is used, for example, to extract the information provided by the Web service of RENFE. This information is presented in a table where each row describes a particular train. Each train consists of a set of attributes describing its number, type, stops (a link to a related page), departure time, arrival time, circulation interval, prices, class and other related features. The specific cases in the organization of HTML pages that this wrapper class covers are described next.

- Tuples of attribute values delimited by tags.
- Tables containing tuples (or rows) of attribute values. Usually, the same specific tags delimit information represented in tables.
- Tuples with empty attributes and tuples with missing attributes.
- Tabular and nested information structure. In a tabular structure, each attribute is represented as an itemized value. In a nested structure, attributes can be represented either as itemized values or as tuples of attribute values.
- Different types of information. By default, the wrappers extract text, although they can also obtain other types of information: Internet addresses, images, codes.

– Irrelevant information. In most pages, there is irrelevant information. In these cases, the description of the page may include specific information to facilitate the extraction of the relevant data: a text placed before (or next) to the relevant data or a set of tags delimiting it.
– Mistakes and incompleteness. The wrappers deal with the existence of void tuples. They also deal with the problem of tags delimiting the attribute values in the source that do not appear in the page description.

Data is extracted from an HTML source following the information about its organization, such as the tags delimiting attributes and tuples. Then, some text processing must be done over this data. The textual processing required for each attribute extracted is given in the page description using a set of predefined types: *text, integer, brackets, list, time, data, weekday,...* There is a default presentation for these types. For example, by default the text will be written in lower case letters, without accents and without spaces.

Finally, the resulting data must be represented in the CO. This information can be represented as new conceptual instances or enriching existing instances. If there is more than one concept described in a page, a different page description will be used for each concept. The description of the page must indicate the name of the concept described as well as the correspondence between the attributes of the tuples in the page and the attributes describing each conceptual instance. Information about a particular instance can occur in more than one page. For example, the departure time and arrival time of a particular train can be in one page and the train stops schedule in a different one.

4 Supporting NL Communication on the Web Contents

As mentioned before, the system described in this paper has been used in a NLI system that generates domain-restricted interfaces to access the Web. This section describes both, the process of building a NLI to a collection of domain-restricted Web sites as well as the functionality of the NLI generated.

4.1 Automatic Building of a NLI for a Specific Domain

Building an interface for a specific domain is an incremental process of acquiring the knowledge bases involved. This knowledge is represented in separate, reusable knowledge bases: the CO, already described, the Linguistic Ontology (LO), representing general linguistic knowledge, and a set of control rules (CRs) representing the general relationships between conceptual knowledge and its linguistic realization. These general knowledge bases are tuned to a particular domain. The approach followed consists of adapting automatically the already existing general linguistic resources to the domain concepts represented in a CO. The knowledge to be incorporated for a specific domain consists of: The domain model in the CO, the lexical realizations of the domain objects appearing in communication (concepts, attributes and operations) in the LO and the descriptions of the Web pages to be accessed.

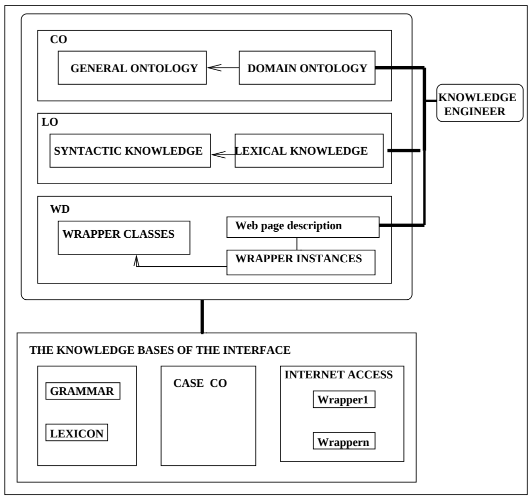

Fig. 4. Building a domain-restricted NLI

Once all this knowledge has been incorporated, the system generates the domain-restricted grammar and lexicon by adapting general linguistic knowledge in the LO to the domain model in the CO. The CRs control this process. Figure 4 shows the system general bases and how they are instantiated for a specific domain. The knowledge bases of the domain-restricted interfaces consist of: the domain-restricted grammar and lexicon, the CO domain model and the wrappers capable to extract information from the Web sites selected. The interfaces using the linguistic resources generated support expressive and precise language adapted to the contents of the Web sites selected.

4.2 A NLI in Action

During communication, the vocabulary used is displayed on the screen. Dynamic entries were incorporated to reduce the number of the lexical entries necessary for a domain. The value of the dynamic entries is set at run-time. There are three different types of dynamic entries: entries representing instances of concepts, entries representing a proper noun or a number that will be requested to the user and entries associated with a specific set of options that will be displayed on the screen at run-time

At each state, only the NL options the system can accept are active. Once the user has selected an option, the interface checks if the option is a lexical entry (it belongs to the interface lexicon) or it is a dynamic entry. Once the final value has been introduced by the user, it is passed to the incremental parser. When the parser returns all items that can be recognized in the next step, the interface updates the NL options that must be active. Alternatively, the user can type the sentence. In this case, the sentence is sent directly to the parser.

Once a whole sentence has been recognized and interpreted by the parser, it is passed to the dialogue controller (DC). The final interpretation of the sentence consists of a consulting operation over the CO. The DC is in charge of executing it. If the necessary information is not found in the CO, the DC obtains the addresses and descriptions of the Web pages containing this information from the Web and activates the corresponding wrapper classes. The wrappers extract the information and represent it in the CO. Then, the user consult is executed again over the CO. The final results are displayed on the screen.

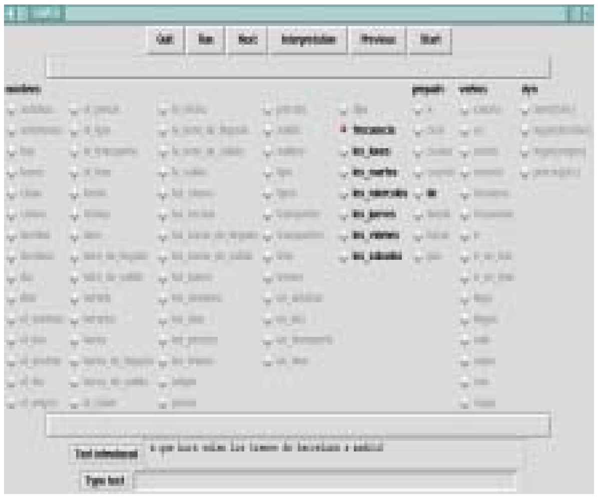

Fig. 5. The NLI generated for the travelling domain

Figure 5 shows the NLI generated for the travelling domain. As can be observed in this figure, once the user has introduced a text either by typing or by choosing the active NL options, only the allowable continuations are available (either for typing or choosing). For instance, once the user has introduced the sentence *A que hora sale el tren de Barcelona a Madrid* two options are displayed on the screen: *frecuencia* (a dynamic entry associated with the week days) and *de* (a lexical entry). In this example, the user has chosen *frecuencia* and the entries corresponding to the week days are shown on the screen.

5 Conclusions

Web query systems deal with the problem of processing information obtained from different Web sites. This paper outlines a solution to this problem: a wrapper system that represents the data extracted from the Web in the CO, thus allowing content-based information access to Web sources. The consistency and power of the CO described provides the NLI system with inferential and reasoning capabilities to tackle complex requests. It could also allow the adapting of the system to the user preferences by adding user constrains to the definition of the operations.

Probably, when languages with well-defined semantics will be used to write Web documents, wrappers following the regularities in presentation format and text extraction heuristics would be replaced by ontologies mappings translating the contents of the Web pages. That would facilitate the data access and integration in a more distributed fashion.

References

1. Ashish, N., Knoblock, C.A.: Wrapper generation for semistructured Internet sources. ACM SIGMOD Workshop on Management of Semi-structured Data (1997)
2. Cardie, C., Ng, V., Pierce, D., Buckley, C.: Examining the Role of Statistical and Linguistic Knowledge Sources in a General-Knowledge Question-Answering System. The Sixth Applied Natural Language Processing Conference (2000)
3. Cohen, W.: Recognizing Structure in Web Pages using Similarity Queries. AAAI (1999) 59–66
4. Cohen, W.: Whirl: A word-based information representation language. Artificial Intelligence, 118, (2000) 163–196
5. Eikvil, L.: Information Extraction from World Wide Web - A Survey. Report 945 (1999). ISBN 82-539-0429-0. Available at: http://www.nr.no/documents/samba /researchareas/BAMG/Publications/webIE/rep945.ps
6. Garcia-Molina, H., Papakonstantinou, D., Quass, A., Rajaraman, Y., Sagiv, Y., Ullman, J., Vassalos, V., Widom, J.: The TSIMMIS approach to mediation: Data models and languages. The journal of Intelligent Information Systems (1997)
7. Gatius, M., Rodríguez, H.: A domain-restricted task-guided Natural Language Interface Generator. Flexible Query-Answering Systems Workshop (1996)
8. Gatius, M.: Using an ontology for guiding natural language interaction with knowledge based systems. Ph.D. thesis. Technical University of Catalonia (2001)
9. Hearst, M.A.: Information Integration Trends and Controversies. Column IEEE Intelligent Systems, 13 (1998) 17–20
10. Kobayashi, M., Takeda, K.: Information Retrieval on the Web. Computing Surveys, 32 (2000)
11. Kushmerick, N.: Wrapper induction: Efficiency and expressiveness. Artificial Intelligence, 118 (2000) 15–68
12. Levy, A.Y.: Combining Artificial Intelligence and Databases for Data Integration. Special issue of LNAI: Artificial Intelligence Today; Recent Trends and Developments (1999)
13. Maybury, M.: Intelligent Multimedia Interfaces. AAAI Press & Cambridge MA: The MIT Press, Menlo Park, CA (1993)
14. Muslea, I., Minton, S., Knoblock, C.: Hierarchical Wrapper Induction for Semistructured, Web-based Information Sources. Conference on Automated Learning and Discovery(CONALD) (1999)
15. Voorhees, E. Overview of the TREC 2001 Question Answering Track. Presentation to the Text REtrieval Conference, USA (2001)
16. W3C Semantic Web Activity: www.w3.org/2001/sw