International Journal of Computer Information Systems and Industrial Management Applications.
ISSN 2150-7988 Volume 5 (2013) pp. 185-194

The use of Domain Ontologies for Improving the Adaptability and Collaborative Ability of a Web Dialogue System

Marta Gatius and Meritxell González

Software Department, Technical University of Catalonia,
Jordi Girona Salgado 1-3, 08034 Barcelona, Spain
{gatius,mgonzalez}@lsi.upc.edu

***Abstract*: Dialogue systems can be used for guiding the users accessing web services, enhancing the web usability. However, they are expensive to develop and difficult to adapt to different types of web services. The knowledge model of a web service can be seen as the basis to define the semantics of the information to be exchanged among the components of a dialogue system. This approach facilitates the integration of the different types of knowledge involved in human-machine communication and provides a unified framework easier to apply to new web services. Furthermore, the representation of the web service knowledge according to an ontology can enhance the reasoning capabilities of the underlying system. This article describes the use of domain ontologies in a mixed-initiative web dialogue system for improving both its adaptability and its collaborative ability.**

***Keywords*: dialogue systems, web assistants, domain ontologies, communication model, dialogue plans, adaptability, usability.**

I. Introduction

As web is becoming more central to our daily activities the need of web assistants enhancing its usability increases. In many situations we can still find it difficult to access web content because there is a large amount of documents and web services in different languages, and they change rapidly. In order to fulfill the user's need web systems need to be more collaborative and adaptable to different domains, users and languages. Interfaces supporting natural language (henceforth, NL) interaction seem useful for accessing the changing information sources in the web, because they can handle friendly and collaborative communication. Furthermore, the use of human-computer NL interaction can be useful when developing and executing complex software, such as systems involving integration of resources and service composition.

NL modes (text and speech) can support several types of interactions, such as menus (the user is asked to choose an option), form filling (specific information is asked to the user) and commands (the user can express an order). Additionally, language can support phenomena not supported by other modes of communication, such as references to previously evoked entities. Besides, in a simple sentence we can express questions that would require several interactions using other modes, e.g., "*Opera concerts on Saturday night for the next two months*". Furthermore, question-answering interaction can favor reasoning, what can be useful in many different situations, such as in the development of software intensive systems (as described in [1]).

Dialogue systems (henceforth, DSs) are focused on achieving a friendly conversation when guiding the user accessing a specific application or domain. DSs have evolved from simple systems interacting with users in a very restricted way (asking the user simple questions) to complex DSs supporting mixed-initiative dialogues, in which both the user and the system can take the initiative. However, inferring the user's intention becomes complex when the user controls the dialogue because the user's interventions are not restricted to previous system's questions, and the user can even change the conversation topic.

The procedures developed for modeling system-initiative dialogues have limitations for modeling mixed-initiative dialogues. For this reason, many flexible DSs include different types of knowledge: domain conceptual and linguistic knowledge, user models, general dialogue mechanisms (such as clarifications and corrections) and communication plans (defining the steps to follow to solve a domain problem). Additionally, the amount of web documents has also favored the development of new NL resources that can improve NL applications (including DSs). Interesting examples of resources that are being developed in this line are the morphological lexica described in [2] and the framework for event extraction presented in [3].

Adapting existing mixed-initiative DSs to guide the user accessing web applications present several challenges. One of the main limitations is that practical DSs are mostly adapted to the functionality of a specific application and are not easily adaptable to new applications. The main reason is that DSs adapted to the communication needs of a particular application improve their performance because mistakes and ambiguities are reduced. Although there are mixed-initiative DSs, having reusable NL components, most of them are developed for a specific type of application and its adaptation to other types of applications is still difficult. The main problem is that adapting the DS implies the modification of heterogeneous knowledge sources.

The use of the application knowledge model as the basis to define the semantics and the content of information exchanged by the system components facilitates the integration of the different types of knowledge involved in communication and provides a unified system easier to apply to new applications. Increasingly, DSs incorporate ontologies to model the application knowledge. The main advantage of organizing conceptual knowledge according to an ontology is that it favors the system reusability and enhances the reasoning capabilities.

In this paper we focus on the use of domain ontologies in order to improve both the adaptability and the collaborative ability of a mixed-initiative web DS that we had previously developed. The system was designed to guide the user when accessing different types of web services in several languages. One of the main differences between our work and related work on complex conversational systems is that our study is focused on how the user can be assisted when accessing different types of web services and information. For this purpose, we have studied the most appropriate representation of the different types of knowledge involved in the communication that takes place when guiding the user to access the web services: domain-restricted linguistic and conceptual knowledge, service descriptions, general communication tasks, dialogue strategies, as well as information about the user. A complete description of the system design is given in [4].

A prototype of the DS had been previously implemented supporting textual access in Spanish and Catalan. The prototype simulated access to two web services of different type: an informational service on cultural events and a transactional service on large objects collection. The results of the evaluation of the prototype can be found in [5].

Our recent research has been focused on using domain ontologies for improving both the DS adaptability and its collaboration ability. In a previous article [6] we described how the use of domain ontologies could be used to reformulate the user's query (in case that no results satisfying the user's requirements were found) and to summarize web information. This article is an extended description of the improvements on the adaptability and collaborative ability of the system. Related work studying the usability of a flexible web DS (such as the one we developed) in several situations in the medical domain is described in [7].

II. Previous work

The problem of using ontologies for enhancing the web accessibility has focused many relevant works in several fields. In this section we compare our work to other related proposals. In the first subsection we relate our work to other works focused on semantic search and semantic integration to enhance web accessibility. The second subsection compares the system we have developed to other intelligent DSs integrating ontologies.

A. *Using ontologies to enhance web accessibility*

There are relevant works on using domain conceptual knowledge in order to improve interaction with the user when accessing web sources. Because there are many different aspects to be considered in this field, we have selected several examples of relevant works that deal with problems considered in our proposal: presenting the user relevant domain information to be included in the query [8], reformulating the user's query [9], integrating ontologies and web resources to enhance semantic search capabilities [10] and representing formally complex queries and the answer models [11].

The approach proposed in [8] to guide the user when building the query of a search engine consists of an interface presenting a set of menus containing relevant domain data that can be easily adapted. Although this approach can be appropriate for a search engine, it could present limitations for accessing other applications that need richer interaction.

The problem of query reformulation when no results satisfying the users' requirements are found has also focused many works. The approach followed in [9], applied to the discovery of semantic web services consist of extending query terms using WordNet, a general lexical ontology that defines all words (in several languages) and establishes several relations between them. In our work, the queries are also reformulated by extending the terms when no results are found, but instead of using a general ontology, we propose the use of a domain ontology (our approach is based on the use of domain-restricted resources) and further interaction with the user , when necessary (as in the system described in [12]).

The enhancement of semantic search capabilities by facilitating the integration of several web sources is also an active line of research. The work described in [10] is an interesting proposal in this line, using common knowledge about city entities in web sources to enhance semantic search in a domain-specific repository. In our system web taxonomies have also been used to improve the interaction with the user when accessing a web service. For example, a furniture taxonomy (obtained from *ikea* web site) has been integrated into the domain ontology in our DS when adapted to a web service related to the collection of large objects.

The representation of complex queries dealing with information obtained from different web sources is a difficult problem. The work presented in [11] describes a formal model to represent those complex queries and their answer models. Although our DS currently do not support queries as complex as those described in that work, that involve several complex operations (together with their interrelations), we could integrate that work to foster our system capabilities.

Our DS provides access to web services previously selected and properly represented into the system knowledge bases. Future improvements on the DS capabilities could also be achieved by incorporating semantic web techniques, such as those using ontologies for the integration of heterogeneous sources [13], query processing and optimization [14], service discovering and composition [15] and monitoring service based systems [16]. As pointed out in relevant works on service composition (such as [17] and [18]) the natural way to improve them is to include the user into the process. There are several related works focused on this line, such as that on the introduction of annotations to facilitate the user interface as part of the web service description (such as [19]).

B. Related work on Dialogue Systems

There are several DSs using application knowledge model representing domain-specific knowledge [20]. As mentioned before, the use of the application knowledge model as the basis to define the semantics of information exchanged by the system components facilitates the integration of the knowledge bases used as well as its adaptation to new applications. Thus, the use of ontologies representing the application model is especially appropriate in systems supporting several modes of interactions and several languages (such as the SMARTKOM system [21]). It also facilitates the incorporation of advance artificial intelligence techniques (such as the ACTIVE system [22]).

Ontologies representing the domain-specific knowledge have been used for years in systems supporting textual interactions (as well as in other text processing applications) to facilitate the semantic interpretation of sentences, by relating the lexicon to the concepts in the ontologies. More recently, ontologies in DSs have also been used to achieve friendlier interaction. By using domain ontologies, DSs may infer default and misunderstood values from user interventions as well as provide descriptions of domain concepts. The use of ontologies can also improve dialogue in several other forms: it helps to detect differences in what is expected from the user's interventions, such as under/over specification (corresponding to hyperonym or hypononym) and to improve dialogue coherence by reordering system's questions (as described in [23]).

We also propose the use of the domain ontologies to generate summaries describing the results, in case that the number of items that satisfy the user's goal is high. This problem has been approach using different techniques. The most relevant of these works propose the distribution of the results clustered for all the possible query parameters (restrictions), as described in [24]. Other proposals also include the use of a user model to express differences in an effective way, as in [25]. These strategies could have limitations on web systems, because the amount of information is huge and clustering the set of results could be costly. Instead, we propose the use of the domain knowledge in the DS ontologies to select the most appropriate information to generate a summary of the partial results.

The DS described in [26] is another example of related work using domain ontologies to present the most appropriate information, the main difference with respect to our work is that the selection of the ontology knowledge is done using information about the user.

Our DS also incorporates information related to the user's expertise to adapt the dialogue strategies. The DS incorporates an adaptive module that evaluates how well the communication is doing and dynamically adapts the dialogue strategies (initiative and confirmation policies) considering this information. This module could be extended by incorporating additional information on the user profile. Other interesting works to be considered on this line are those on using user and task models to adapt an interface for a complex application (such as [27]), on user models on Web navigation (such as [28]) and on personalized web services (such as [29]).

III. The Dialogue System

As mentioned in the introduction, the mission of the DS that we have developed is to assists the user when accessing the web. The current prototype guides the user to perform two tasks of different type: searching for information on cultural events and stating a date for furniture collection. This section gives an overview of the system.

A. The System Architecture

The DS that we have developed follows a modular architecture, as shown in Figure 1. It consists of four independent components: the NL understanding module (NLU), the NL generator (NLG), the dialogue manager (DM) and the task manager (TM).

During the conversation, every sentence in the user's turn is first processed by the NLU and the resulting interpretation is passed to the DM. The DM is responsible for controlling the dialog to follow the steps to help the user to achieve his/her goals. For this purpose, the DM firsts infers the user's intention from the semantic interpretation of the intervention, the dialogue context and the domain knowledge. Then, the DM determines the next system's actions, which can be any of the following: the generation of a system's respond or accessing the web service. The TM controls the access to the web service. The generation of the system's responds depends on two modules, the DM and the NLG. First, the semantic content of the system's intervention is generated by the DM and then, this semantic representation is passed to the NLG, which generates the system's response in NL.

The DM uses the adaptive module to determine dynamically the degree of the initiative of the system's response. For example, when there are communication problems the system takes the initiative (asking direct questions). Then, when the user answer contains additional relevant data, the initiative is given back to him/her. The adaptive module determines the degree of the DS initiative following a user model that uses relevant cues about how well the communication is doing. The set of cues ranges from task success and repetitions of the same concept to divergence between the user answer and what was expected (i.e., extra or missing information, totally unexpected information) and parser or database error.

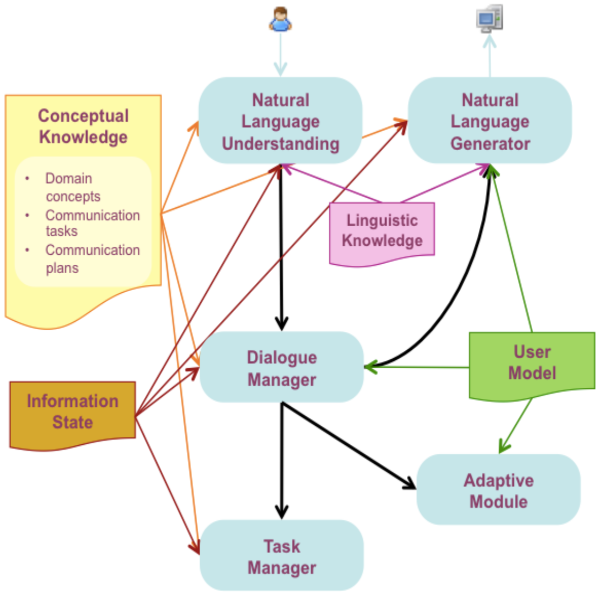

Figure 1. The architecture of the dialogue system

B. *The general knowledge bases*

The different types of knowledge used by our DS are represented in separated knowledge bases. Those knowledge bases consist of the domain ontology (used across all modules), the communication tasks (used by the TM), the general communication plans (used by the DM) as well as the linguistic knowledge bases used by the linguistic modules. In particular, the linguistic knowledge consists of the grammar and the lexicon used by the NLU and the patterns used by the NLG.

Adapting the DS to a new service consists of adapting these general knowledge bases. First, the operations of the new web service become instances of the communication tasks. Then, the domain concepts related to these operations are incorporated into the new domain-dependent ontologies. This initial process is done manually. Next, general plans are adapted to these communication tasks (semi-)automatically. And finally, the general linguistic knowledge is adapted to the communication tasks and domain ontology to obtain the domain-restricted linguistic resources. The adaptation of the data used by all the modules to the service specific knowledge facilitates the exchange of information among the modules.

The next section gives a more detailed description of these general knowledge bases and how they are adapted for a specific web service.

IV. Adapting the general knowledge bases

This section describes the incremental process of acquiring the domain-restricted knowledge that is involved in communication: the domain ontologies, the communication tasks, the dialogue plans and the linguistic resources.

A. *The domain ontology*

As mentioned before, the system uses an ontology representing the conceptual knowledge related to the web services. This domain ontology is used to improve both the adaptability of the DS and its capabilities.

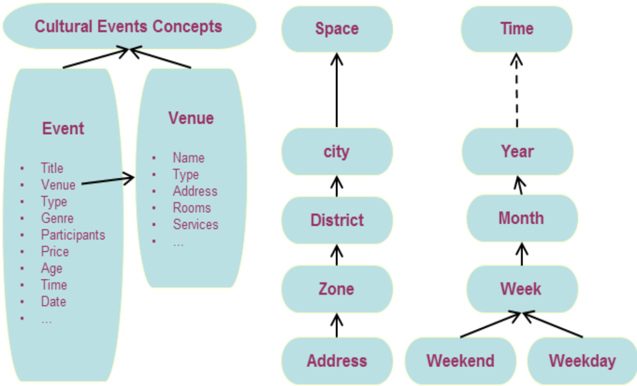

Figure 2. Concepts in the cultural events domain

This ontology consists of top concepts and their attributes representing general knowledge. For each web service, the domain concepts related to the web services operations and their attributes are incorporated as instances of the general concepts. The knowledge in the ontology is used as the basis to define the semantics of the information exchanged by the system components.

Let us consider, for example, the cultural events domain. In order to apply our prototype to this domain, we have studied a web service giving information about the city of Barcelona. This service supported only a single operation: consulting information related to the cultural events that take place in the city. The input parameters of the operation were *what***,** *where***,** *when* and *type*. The advanced search mode included other optional parameters (e.g., *age*). This web service operation is linked to the concept **Event** in the domain ontology, as shown in Figure 2. The parameters of the operation are represented as the attributes **title**, **venue**, **type** and **date**. Other related attributes describing the concept (e.g., **participants, price, age** and **schedule**) are also included because they often appear in the dialogues related to this domain. Additionally, the venue where the event takes place is represented as the concept **Venue** because it is the central item in many dialogues on the domain on cultural events (e.g., which films are shown in the *Central Cinema*). The attributes that describe the concept **Venue** are the **name**, the **address**, the **services** and the **rooms**. The attributes **date** (describing the event) and **address** (describing the venue) are further linked to the representation of the general concepts **Time** and **Space**.

The domain representation can be reused across other services on cultural events. In case that additional information is needed for a new service in the same domain, the domain ontology would be extended. The ontology can thus be used to integrate information related to several web services on the same domain.

Additionally, the domain ontology could also be used to facilitate web service composition. The domain concepts appearing in a web service can be related to other concepts appearing in other web services on different domains, which may not provide the same operations. For example, the information on cultural events includes the venue where an event takes place. Additional data on how to get there (obtained from other web services) could also be given to the user.

B. *The communication tasks*

The application-dependent knowledge appearing in communication is basically related to the operations that the service can perform, i.e., the data needed from the user to perform the task and the resulting information.

In order to facilitate the incorporation of this knowledge for a new web service, we have defined general models for the operations appearing in most common web services. For this reason, we have studied the operations performed by several transactional and informational web services. Notice that one web service may support more than one operation. Three different operations have been identified for transactional services: *submission* (controlling the transaction and presenting the results), *cancellation* (controlling the cancellation of a transaction previously done) and *information* (giving related information). For example, the three operations related to a hotel reservation are: to make the reservation, to cancel a previous reservation and to give information about the reservation or the service itself.

In contrast, informational services usually only perform the operation of giving information. This operation can be decomposed depending upon the type of information that has to be given to the user: *single item*, *list of items*, or *summary*.

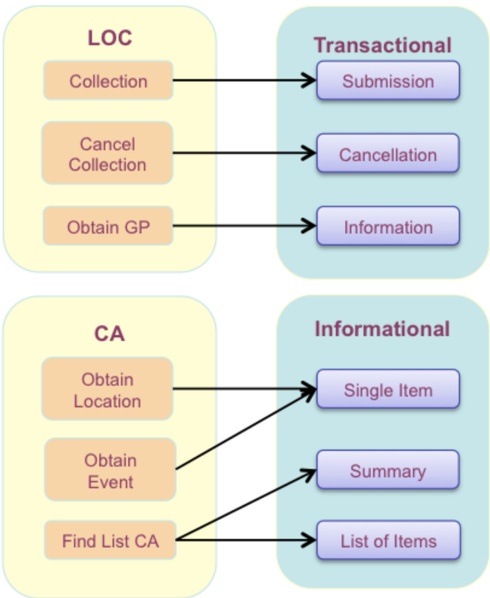

Figure 3. Representation of tasks in web services

Figure 3 illustrates the classification of the operations of the two web services incorporated into the prototype: the informational service giving information on cultural events (CA) and the transactional service stating a date for large objects collection (LOC). Each service operation is defined as instances of the general task representations, and it includes input and output parameters, constraints and conditions. The constraints can apply at different levels: attribute value, attribute relations or the entire task. During the conversation, the user gives values to these parameters, and the TM has to consider all the constraints of the task in order to manage the values given to them.

Considering the information that appears in communication when the DS guides the user to perform the service operations, we have identified and defined the three general types of tasks: **SubmitForm**, **ObtainData** and **FindList**. **SubmitForm** represents the operations making transactions (*submission* and *cancellation*). **ObtainData** represents the operations giving *information*, i.e., returning the description of an item satisfying a given criteria. **FindList** returns a *list of items* or a *summary* of the results.

To facilitate the execution of these tasks we have defined a simple algorithms to processes each type of task. For each service, specific operation constraints are defined. The algorithms that process the tasks **SubmitForm** and **ObtainData** control that there are no conflicts among the parameters. The algorithm processing the **FindList** only controls the number of results.

When a new web service is incorporated, each service operation must be described as an instance of one of the three general tasks. At run time, the TM uses these instances to access the web service and process the information obtained from it. At the beginning of the conversation, the system identifies which service and task the user intends to execute. Then, the system completes the task information considering the data that appear in the subsequent turns. Next, the TM accesses the web service and finally, it processes the results.

As mentioned before, the domain ontology representing service operations and parameters is used as the semantic base to exchange information between the system modules. This semantic information is incorporated into the linguistic resources used by the NLU. Therefore, the semantic interpretations of the user interventions are represented on the grounds of the service task and the domain ontology. From the semantic interpretation of user interventions the TM identifies the service and the task that has to be accessed.

Let's consider, for example, that the user's first intervention is "*Which concerts are there in the Stadium on Saturday?*" The TM has to identify the service to be accessed (cultural events), then the task (find list of events) and the values of the input parameters (*type*: "concert", *where*: "Stadium", *when*: "Saturday"). Then, it executes the task and passes the results to the DM, which controls the system's next actions. For example, the DM controls how results are presented to the user (i.e. a list, a summary, a message) considering the number and type of the data resulting from the execution. In case no items satisfying the user's restriction are found, several strategies to help the user to find the desired information can be followed, as described in next section.

C. *The dialogue plans*

The DM controls the interaction with the user. It controls the information that has to be asked and given to the user and how this process has to be done. In our DS, the DM uses dialogue models based on communication plans. Those models consider the user's utterances as communication actions that are part of a plan that has to be achieved by the user in collaboration with the system.

We consider that the communication that takes place when guiding the user to access web services consists mainly of the information related to the web service tasks and the task's parameters. For this reason, the plans describe, basically, *what* and *when* the information has to be asked and presented to the user.

In order to provide flexibility, the plans in our DS are not structured as flat lists of actions but as sub-plans that are accommodated at run-time considering context information. Communication plans and sub-plans can be decomposed into further sub-plans and actions. Actions are the simplest unit in the communication plan they are used to compose plans and sub-plans. Possible actions are those asking the user information about a single input parameter as well as those giving the value of the output parameters.

The dynamic generation of plans during communication, used by other systems, increases complexity and time processing, reducing the adaptability of the system. To overcome this limitation, in our DS (as in many practical DSs) these communication plans are generated a priori, when a new service is incorporated in the system, and they are stored in libraries. The main drawback that the generation of the plans for each web service presents is that these plans have to be manually written. To solve this problem we have created general templates that facilitate the plan generation for a new service. Basically, those general templates are related to the general communication task models used in our DS to describe the operations involved in transactional and informational web services and described in the previous subsection (The general tasks **SubmitForm, ObtainData** and **FindList**).

The actions needed to carry out **SubmitForm** and **ObtainData** tasks are rather simple. The step of collecting the information consists of asking the user to give the values of a set of slots (parameters) needed to fill a form (the task). Then, the information obtained is passed to the TM and the result of the transaction has to be presented to the user, usually as a text sentence (e.g., "*The transaction has been done successfully*", "Your reservation number is 12345").

The sequence of actions for carrying out **FindList** tasks is complex because different situations have to be considered. The process to obtain information from the user to restrict the search can involve several steps. Although usually there is a set of possible parameters to restrict the search, unlike in the case of transactional services, the user may choose to give the value of only a subset of those possible parameters. However, the user's goal when looking for information on the web is not always clear and can even change during the communication process (even from one turn to the next). Besides, in several cases, results satisfying user's query are not found while in other cases the results obtained are many. For this reason, there is a need for collaborative systems that assist the user when formulating the queries and present the information found in a clear form.

Several considerations have to be done when presenting the results obtained from informational web services. First, the DM has to decide which specific data about the items found has to be presented, i.e., if a complete list or partial description is more appropriate. For instance, the service about cultural event can offer to the user different data about a particular event: the event name, the location, the **data**, the **time**, the **price**, etc.

In the specific case that the DS has found no items satisfying the user's request, the system could guide the user to reformulate the query or, alternatively can do it using different types of knowledge. These strategies are described in the next section.

Several strategies can be followed in the specific case that the number of results obtained is higher than a predefined threshold. Simple responses could present all the results in a row. But when the number of results is high, this approach cannot result very friendly. A more cooperative response could present partial results.

We propose the use of domain ontologies to generate more collaborative responses in the two specific situations mentioned above: when no results satisfying user's requirements are found and when there are many results.

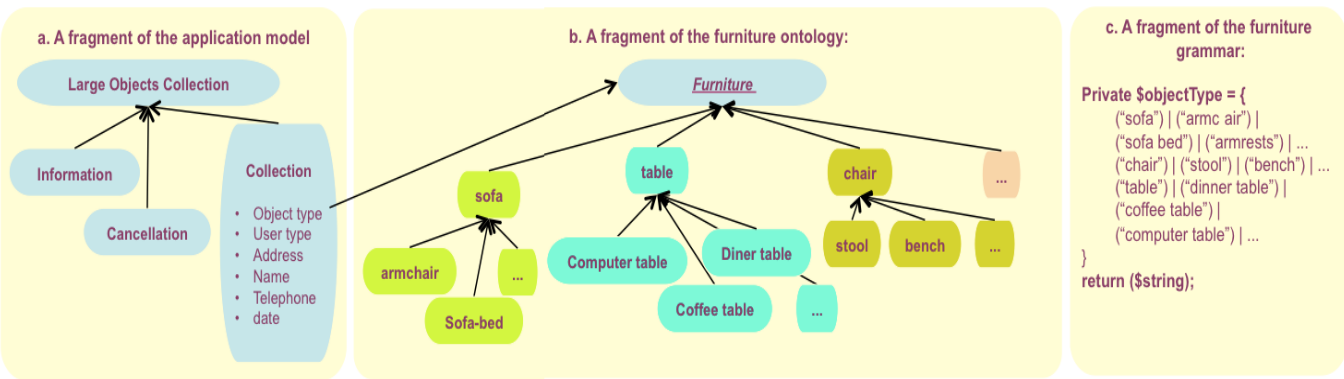

Figure 4. Use of ontologies in the LOC service to obtain domain-restricted linguistic resources

D. The linguistic knowledge

The language modules in our DS (the NLU and the NLG) use general and domain-restricted linguistic resources. The use of domain-restricted linguistic resources limits the space of possible interpretations in language processing but increases robustness and reduces the run-time processing. In our DS, the linguistic resources consist of linguistic structures and terms needed to express questions and answers related to the specific web services and their parameters.

The use of a general description of web services plus an ontology representing domain concepts and relations favors the semi-automatic adaptation of general lexicons and linguistic structures.

Domain taxonomies have also been used in our DS (as in other DSs) in order to obtain all the possible terms that can appear in the user intervention in relation to some specific information. Existing general lexical ontologies, such as *EuroWordnet*, could also be used to obtain related terms, but a lot of work is necessary to choose the appropriate terms for a particular domain among all the possible related terms.

Let us consider, for instance, the service for collecting large objects from houses. This service needs information about the specific type of object that the user wants to get rid of. Figure 4 shows how a fragment of the voice grammar (in VoiceXML formalism) is represented from the collection operation of this service and a domain taxonomy. A taxonomy of furniture has been used to obtain the terms representing the different types of objects. For this specific service, the objects related by the *is-a* relation have been represented as synonyms, linked to the same concept furniture, because there is no need for more specific distinctions considering the specific type of furniture.

When needed, the DS can use the taxonomy to obtain more precise information. For example, it can detect hypernymy in the user's intervention describing the object. In the sentence "*I want to get rid of some furniture*" the concept *furniture* is classified in the objects taxonomy as a hypernym of the information that the service needs, in consequence, in the next turn of the conversation, the DS asks the user to be more specific.

V. Using the domain ontology to enhance the collaborative ability

A. Reformulation of the user's query

Users always expect satisfactory answers from the system. In case that the system cannot offer any result, it should offer possible fruitful alternative. We have defined a set of rules to reformulate the user's query to obtain a more general one (to relax the query constraints) and thus increment the chances to obtain results. The main advantage of this approach is that general relaxation rules can be reused across domains. The following is an example of constraint relaxation rule:

Rule 1. *A conceptual class can be replaced by its upper class.*

U1: I want to go to the Opera tonight.
S1: Sorry there are no opera concerts tonight.
But there is another classical concert:
The Requiem of Brahms in the Auditori.

Dialogue 1. Replacing "Opera" by "classical concerts"

Dialogue 1 illustrates the use of Rule 1 in the domain of the cultural events. In this scenario the user is looking for opera concerts that should take place at a specific date (tonight) and the system does not find any. If the system would have asked the user to give a more general description of the object to be found it could have taken the user several turns to redefine his goal. In order to achieve a more friendly interaction, our DS is able to do this process automatically.

As described in previous section, in the domain ontology the web service operation is linked to the concept **Event** and the operation parameters are represented as the attributes of this concept. In Dialogue 1, the user query includes two restrictions (operation parameters): the event **type** ("*opera*") and the event **date** ("*tonight*"). Applying Rule 1, the system changes the value of query parameter **type** ("*opera*") to its upper class ("*classic concerts*").

Since **Address** and **Date** are general concepts appearing in many web services, we have defined additional specific rules derived from general Rule 1:

Rule 1.1. *The attribute* ***address*** *can be replaced by the more general attributes* ***zone*** *and* ***district****.*

Rule 1.2. *The attribute* ***date*** *can be replaced by the more general* ***week day****,* ***weekend****,* ***week*** *and* ***month****.*

U1: I want to see Opera Aida next Sunday.
S1: Sorry. There is no performance of Opera Aida the next Sunday.
But there is one on Saturday.

Dialogue 2. Replacing "Sunday" by "weekend"

Dialogue 2 illustrates the application of Rule 1.2. There are no results satisfying the restrictions of **title** ("Aida") and **date** ("next Sunday"), but a result is found when the restriction "Sunday" is replaced by the more general "weekend".

Notice that Rule 1 can be applied to any of the restrictions in the query. For each specific domain there is a predefined order of relevance of the operation parameters (restriction candidates in the query) and the restrictions in the query are applied following this order (starting by least relevant). The predefined ordered set of parameters (represented as attributes) in the domain of cultural events is: {**title**, **age**, **date**, **type** and **venue**}.

The Rule 2 is another example of a general rule that can be adapted to each specific domain.

Rule 2. *If the query includes two or more restrictions then one of the restrictions is deleted until results are found.*

U1: Are there musical concerts for children on Sunday in the Auditori?
S1: Ok. I'm searching. I'm sorry, there are no events for children on Sunday at Auditori.
But, there are two other events for children on Sunday.
Results from 1 to 2 follows:
1: Musical festival on Ciutadella
2: Sónar Kids

Dialogue 3. Restriction venue ("Auditori") is deleted

Dialogue 3 illustrates the use of Rule 2. In this example, a set of results partially satisfying the description given by the user is presented. In the examples, there are no musical concerts for children on Sunday in the Auditori, but other musical concerts for children at the same date in other places. The user's query includes four restrictions, that is, the value of four attributes: **type** ("music"), **age** ("for children"), **date** ("Sunday*"*) and **venue** ("Auditori"). The system applies the Rule 2 and eliminates the attribute **venue** (notice that the attributes **age, date** and **type** are more relevant in the cultural domain). The updated query "*Events for children on Sunday*" does find results. If Rule 1 could be applied the venue ("Auditori") would have been replaced by other equipment in the same zone or district. Because no venues in the zone satisfying the query have been found, this restriction is eliminated.

Specific rules for specific domain concepts could also be incorporated. The advantage of defining relaxation rules adapted to a particular domain (and not to a particular service) is that they can be reused across web services related to the same specific domain.

B. Generation of summaries

Another collaborative ability of the DS is to offer a summary of results in case that the system finds too many of them satisfying the user's requirements. The purpose of such summary is to give a friendly response and offer the user the chance to give additional query restrictions that may lead to a satisfactory search.

In order to generate an appropriate summary we have defined an ordered set of the attributes that belong to the ontology concepts representing the service operation. These general ordered set of attributes are the ones to be considered to elaborate a summary and is not the same than the one used in the relaxation rules. Instead, this ordered set consists of the following attributes: first **types** (e.g., types of event, types of restaurant), second **subtypes**, third **locations** and finally **time.** Attributes in this set can be replaced by their subclasses (which depend on each domain). Additionally, the set attributes can be extended for each domain. For example, the information on cultural events can be summarized using the following ordered set of attributes: {event **type**, event **genre**, **zone** and **district**}.

We have also stated several restrictions on attributes that could be considered to produce a proper summary. For instance, attributes that cannot be used for summarization are those already appearing in the query constraints (i.e., explicitly described by the user) and those having an open set of possible values.

The example in Dialogue 4 shows the interaction between the system and a user searching information on cultural events. In this example, a summary has been generated because the results satisfying the user's goal ("*movies today"*) were too many to be presented in one turn. Using the ordered set of attributes, the information has been summarized considering the event **genre** (being the event **type** "cinema").

U1: movies today
S1: OK. I am searching. There are many films today.
But there are 6 events in the dramatic genre,
12 events in the action and terror genre,
8 events in the childlike and adventure genre and
6 events in other genres.
Which genre are you interested in?

Dialogue 4. An example of summary

Conclusion

DSs turn to be useful to assist users to use web services because, in daily tasks, it could be difficult to cope with such a variety and amount of data in the web. One of the main limitations of developing such complex DSs is the high cost of adapting them to different types of web sources (different domains, languages, applications, etc.). To solve this limitation we propose a general organization of the different types of knowledge involved in the communication that takes place when assisting the user: service task models, domain concepts, dialogue plans and linguistic resources. This organization in separated but related general knowledge bases favors the adaptability of the DS to new web services, languages and users. We also describe how the use of domain ontologies enhances both the adaptability of our DS and its collaborative abilities.

The prototype implemented supports text access in Spanish and Catalan to two web services. However, the DS design facilitates its extension to access other web services, languages and modes of communication (such as voice).

Future work will also include the study of how our DS could assist the user in more complex tasks, implying, for example, the composition of several transactional web services.

References

[1] P. Sosnin. "Programming of Human-Computer Interactions in Development of Software Intensive Systems", *International Journal of Computer Information Systems and Industrial Management Applications*, Volume 3, pp. 444-452, 2011.

[2] Z. Detorakis, G. Tambouratzis. "Clustering Techniques for Establishing Inflectionally Similar Groups of Stems", *International Journal of Computer Information Systems and Industrial Management Applications*, Volume 4, pp. 219 -227, 2012.

[3] S. Sangeetha, R.S. Thakur, and M. Arock. "Domain Independent Event Extraction System Using Text Meaning Representation Adopted for Semantic Web", *International Journal of Computer Information Systems and Industrial Management Applications*, Volume 2, pp. 252-261, 2010.

[4] M. González. "Digui: A flexible dialogue system for guiding the user interaction to access web services". *Ph.D. dissertation*, Technical University of Catalonia,

Barcelona, Spain, 2010. [Online]. http://nlp.lsi.upc.edu/papers/mgonzalezthesis10.pdf .

[5] M. Gatius and M. González. "A flexible dialogue system for enhancing web usability". In *Proceedings of the International Conference on World Wide Web (WWW),* pp. 1167–1168, 2009.

[6] M. Gatius and M. González. "Using Domain Knowledge for Fostering the collaborative Ability of a Web Dialogue System". In *Proceedings of the International Conference on Next Generation Web Services Practices (NWeSP),* pp. 129-134, 2011.

[7] M. Gatius and T. Namsrai. "A Conversational System when Accessing Web Sources in the Medical Domain". In *Proceedings of the Conference on Advances in Computers-Human Interactions (ACHI),* pp. 150-154, 2012.

[8] S. Dey, S. Abraham. "User Interface For A Search Engine: A Customized and Multi-domain Approach", *International Journal of Computer Information Systems and Industrial Management Applications,* Volume 4, pp. 169-179, 2012.

[9] D. Mekhzoumi, H. Aliane, O. Nouali. "Queries reformulation for the discovery of semantic web services". In *Proceedings of the International Conference on Next Generation Web Services Practices (NWeSP),* 2011.

[10] W. Guo, S. B. Kraines. "Integrating Knowledge of City Entities Extracted from DBpedia and GeoLite into the EKOSS Failure Cases Repository to Enhance Semantic Search Capabilities", *International Journal of Computer Information Systems and Industrial Management Applications,* Volume 3, pp. 780 -787, 2011.

[11] B. Batouche, Y. Naudet, F. Guinand. "Web Service Complex Request Ontology and it Answers Models". In *Proceedings of the International Conference on Next Generation Web Services Practices (NWeSP),* 2011.

[12] S. Varges, F. Weng, and H. Pon-Barry, "Interactive question answering and constraint relaxation in spoken dialogue systems", *Natural Language Engineering*, Volume 15, pp. 9–30, 2009.

[13] A. Cuadra, M. M. Cutanda. "A Semantic Web-based Integration Framework". In *Proceedings of the International Conference on Next Generation Web Services Practices (NWeSP),* 2011.

[14] S. K. Malik, S. Rizvi. "A Framework for SPARQL Query Processing, Optimization and Execution with Illustrations", *International Journal of Computer Information Systems and Industrial Management Applications*, Volume 4, pp. 208 -218, 2012.

[15] H. Q. Yu, S. Dietze, C. Pedrinaci and D. Liu. "A Linked Data compliant Framework for Dynamic and Web-scale Consumption of Web Services", *International Journal of Computer Information Systems and Industrial Management Applications,* Volume 3, pp. 796-803, 2011.

[16] A.K. Tripathy and M.R. Patra. "Service Based System Monitoring Framework", *International Journal of Computer Information Systems and Industrial Management Applications,* Volume 3, pp. 924-931, 2011.

[17] V. Rozinajová, M. Kasan, P. Návrat. "Towards more effective support of service composition: utilizing AI-planning, semantics and user's assistance". In *Proceedings of the Next Generation Web Services Practices,* pp. 50-55, 2009.

[18] R. Kazhamiakin, M. Paolucci, M. Pistore and H. Raik. "Modelling and Automated Composition of User-Centric Services". In *Proceedings of the Next Generation Web Services Practices, 2011.*

[19] M. Feldmann, G. Hubsch, T. Springer, A. Schill. "Improving Task-driven Software Development Approaches for Creating Service-Based Interactive Applications by Using Annotated Web Services". In *Proceedings of the Next Generation Web Services Practices,* pp. 94-97, 2009.

[20] J. Allen, D.K. Byron, M. Dzikovska, G. Ferguson, L. Galescu, and A. Stent. "Toward conversational human-computer interaction", *AI magazine*, 22(4), pp. 27-38, 2001.

[21] D. Sonntag, R. Engel, G. Herzog, A. Pfalzgraf, N. Pfleger, M. Romanelli, and N. Reithinger. "Smart web handheld –multimodal interaction with ontological knowledge bases and semantic web services", in *AI for Human Computing*, T.S.Huang et al. (Eds.):LNAI 4451, Springer, 2007.

[22] D. Guzzoni, C. Baur, and A. Cheyer. "Active: A unified for building intelligent web interaction assistants". In *Proceedings of the IAT Workshop*, pp. 417–420, 2006.

[23] D. Milward. "Ontologies and the structure of dialogue". In *Proceedings of the Workshop on the Semantics and Pragmatics of Dialogue (Catalog),* pp. 69–77, 2004.

[24] J. Polifroni and M. Walker. "Intensional summaries as cooperative responses in dialogue: Automation and evaluation". In *Proceedings of ACL'08 – HLT,* pp. 479–487, 2008.

[25] A.K. Winterboer, M.I. Tietze, M.K. Wolters, and J.D. Moore. "The user-model based summarize and refine approach improves information presentation in spoken dialog systems", *Computer Speech and Language*, Volume 25(2), pp. 175–191, 2011.

[26] M.E. Foster and M. White. "Assessing the impact of adaptive generation in the comic multimodal dialogue system". In *Proceedings of the IJCAI 2005 Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, 2005.

[27] V. Alvarez-Cortes, V.H. Zárate Silva, B.E. Zayas Pérez, J.A. Ramírez Uresti and A. Quintero Reyes. "User and Task Models Impact on an Adaptive User Interface for the Startup of a Power Plant", *International Journal of Computer Information Systems and Industrial Management Applications*, Volume 4, pp. 300-308, 2012.

[28] G. de la Cruz Martínez and F. Gamboa Rodríguez. "Using User Interaction to Model User Comprehension on the Web Navigation", *International Journal of Computer Information Systems and Industrial Management Applications*, Volume 3, pp. 878-885, 2011.

[29] Z. Jarir, M. Quafafou and M. Erradi. "Personalized Web Services for Web Information Extraction", *International Journal of Web Services Practices*, Volume 5, No.1, pp. 22-31, 2010.

Author Biographies

Marta Gatius was born in Barcelona. She is an assistant professor in the Software Department at the Technical University of Catalunya (Spain). Her research interests include natural language processing and generation, natural language interfaces, dialogue systems, knowledge representation, ontologies and information extraction from the Web. She received her Ph.D. from Software Department at the Technical University of Catalunya in 2001.

Meritxell González was born in Barcelona. She is a researcher in the Software Department at the Technical University of Catalunya (Spain). Her research interests include natural language processing and generation, natural language interfaces, dialogue systems and machine translation and evaluation. She received her Ph.D. from Software Department at the Technical University of Catalunya in 2011.