MANAGING USER INTERACTION IN AN ONTOLOGY-BASED SYSTEM

Anna Goy and Diego Magro
Dipartimento di Informatica, Università di Torino, C.Svizzera 185, Torino, Italy

Keywords: User interaction, User interfaces, Ontology, Semantic Web, e-Business.

Abstract: In this paper, we present an approach to the management of user interaction within an ontology-based system. In particular, we discuss the case of ARNEIS, an "intelligent" web-based repository of software solutions, that enables software houses to upload a description of their software solutions for business automation, and small-to-medium sized enterprises to find software products or services supporting their activities. To this purpose, as argued in the paper, a particularly meaningful field is Customer Relationship Management, that is thus exploited as a test bed domain. The functionality of ARNEIS is based on an ontological representation of the domain knowledge, which represents the shared conceptual vocabulary to express software descriptions and technological requirements. In this paper we describe our proposal for the management of a user-friendly interaction enabling software houses to upload the semantic representation of the description of their offers. The approach we propose within the ARNEIS scenario represents an example of a more general solution to face the issue of how to build formal representations of resources in an ontology-based IR system.

1 INTRODUCTION

In order to be competitive in the globalized market Small-to-Medium sized Enterprises (SME) have to exploit the support provided by innovative technological solutions to their business. On their side, Information and Communication Technologies (ICT) relies more and more on Internet and Web technologies: web-based solutions are available for almost any kind of application, supported by innovative technologies at different levels, ranging from Cloud Computing infrastructures (Creeger, 2009; Dikaiakos et al., 2009), to Web Services (Alonso et al, 2004), to Semantic Web (Antoniou and Van Harmelen, 2004).

However, the technological support is not enough: in order to be competitive, SME should also take care of new business and management approaches. An example is the way in which they handle their relationships with customers. The new market requires personalized approaches to the single customer, as well as flexible offers, that need to be updated rapidly. Moreover, in order to be aware of the market and customer behavior trends, data about offers, sales, communications with customers, and so on have to be elaborated very rapidly, to support management and marketing decisions. For these reasons, SME pay every day more and more attention to the principles of Customer Relationship Management (Freeland, 2005) and to those ICT products and services supporting it. The key feature of the CRM approach is a one-to-one marketing perspective, i.e. establishing personalized relationships with the single customer, by producing personalized offers, pricing, after-sale services, etc. Moreover, CRM is a field in which technological innovation could bring great benefits since it requires the processing, integration, and analysis of a huge amount of heterogeneous knowledge (about customers, sales, communications, etc.); it also requires effective, fast and integrated communication tools.

Within this scenario, we think that there are two main issues that play a major role:

(1) A clean, complete, and sharable ontology (Guarino, 1997) defining the concepts related to CRM is of paramount importance, in order to support the mentioned knowledge management activities, as well as Enterprise Application Integration; see, for instance, (Benjamins, 2008).

(2) SME would get great benefits from a web-based service supporting an intelligent matching between supply and demand for CRM-related tools.

Moreover, the shared knowledge mentioned in (1) can be exploited in a cross-enterprise scenario like the one outlined in (2). Thus, we designed an architecture for a web-based intelligent system supporting SME in finding suitable software solutions for their business, and we chose CRM as a testbed for this architecture.

On the basis of this architecture, we developed ARNEIS (Advanced Repository for Needs of Enterprises and Innovative Software), a prototype implementation of a web-based repository of software solutions, that exploits Web Services and Semantic Web technologies. ARNEIS users are: (1) ICT companies (i.e., software houses) that offer software solutions for business automation, but can be in trouble in getting in contact with their potential customers; (2) SME that aim at finding software products or services supporting their business and management activities, but lack the know-how to find the right ICT solution that fits their needs.

Such an intelligent repository has to be equipped with a large and detailed knowledge base, e.g. an ontology, which represents the shared conceptual vocabulary; such an ontology is the basis for the matching algorithm used to suggest SME the most suitable software solutions, given their needs.

Moreover, ARNEIS requires a web-based user interface (UI) enabling ICT companies to describe their software solutions and SME to express their requirements and needs and to find suitable software solutions supporting their business.

We claim that both these goals, i.e. building the CRM ontology and defining the system UI, have to be based on a domain analysis that takes into account how users, both from ICT companies and from SME, talk (and think) about CRM activities (see Section 3).

The CRM ontology developed for the ARNEIS project is described in (Magro and Goy, 2008a; Magro and Goy, 2008b), while this paper focuses on the definition of the UI enabling the users to interact with it. In particular, Section 2 briefly describes the system architecture and Section 3 reports the main results of the domain analysis; Section 4 focus on the UI management: it provides a brief survey of the relevant related work in the field, it discusses the UI design choices and it describes the mechanism for UI management. Section 5 briefly discusses some open issues and concludes the paper.

2 ARNEIS ARCHITECTURE

The basic architecture of the ARNEIS system is described in (Goy et al., 2008); Figure 1 shows a simplified version of it.

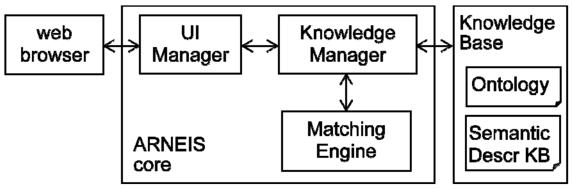


Figure 1: ARNEIS system architecture (a simplified version taken from (Goy et al., 2008c)).

It is a standard three-tiers architecture, where the presentation layer is represented by a standard web browser.

The application logic is contained in the *ARNEIS core*, which includes three main components: the *UI Manager*, the *Knowledge Manager*, and the *Matching Engine*.

The *UI Manager* generates the UI taking as input the concepts, properties and relations provided by the Knowledge Manager and represented in the Ontology; moreover, it collects information provided by the user and forwards it to the Knowledge Manager, that builds the semantic representation of software descriptions and SME requirements.

The *Knowledge Manager* mediates the interaction between the UI Manager and the data layer (its role will be discussed in more detail in Section 4.3). Moreover, it dialogs with the *Matching Engine* (whose description is out of the scope of this paper), by providing it the semantic representations on the basis of which the Matching Engine will calculate the correspondence between SME requirements and software house offers.

The data layer contains the *Knowledge Base*, whose main components are the *Ontology*, representing the system semantic competence about the domain (CRM), and the semantic representation of software descriptions and SME requirements (*Semantic Descr KB*). Both the Ontology and the Semantic Descr KB are written in OWL (http://www.w3.org/2004/OWL/).

The OWL domain Ontology is based on a CRM *Reference Ontology* (Guarino, 1997), described in (Magro and Goy, 2008a; Magro and Goy, 2008b), which in turn is based on DOLCE (http://www.loa-cnr.it/DOLCE.html). The CRM Reference Ontology basically models: (1) business activities (e.g., sales, offers, communications, appointments, etc.); (2) business relationships which the company is involved in (e.g., relationships with actual or potential customers); (3) the knowledge that the company has on (or derives from) business activities and relationships; (4) software supporting business activities and knowledge management.

3 DOMAIN ANALYSIS

Given the architecture described in the previous section, which is the information needed by the UI Manager to generate the UI? And which kind of information should be elicited from the user in order to build a structured and formal representation of the software solutions offered?

To answer this question, we carried out a detailed domain analysis, in order to understand how users (both software houses and SME) talk about CRM.

We analyzed two types of information sources: (1) Documents (e.g., brochures, white papers), produced by ICT companies, describing their CRM software tools. (2) Interviews with (a) salesmen from ICT companies, aimed at eliciting the way they describe software solutions for CRM for SME; (b) managers of SME, in order to understand which concepts and terms they use to think about their activities related to customer management.

The analysis of documents and interviews provided us with the following outcomes:

(1) We identified the main concepts, properties and relations involved in the description of CRM activities in SME, which represent the basic requirements for building the CRM Ontology.

(2) We identified a set of *dialog topics*, i.e. concepts that emerged to be the keys of the description of CRM activities. Within the system, these concepts can be represented as *templates*, i.e. general conceptual patterns that are instantiated with more specific concepts in the different documents/interviews, by means of various linguistic forms. Each template has a corresponding formal semantic representation in terms of the Ontology. Basically, each description of a software solution (or a SME requirement) is an instantiation of a number of such templates (see Section 4.3).

4 USER INTERACTION

As mentioned in Section 2, the system needs a formal representation of the descriptions of software solutions and of technological requirements (both based on the terms defined in the system Ontology) in order to find the software solutions best matching the SME requirements. Since it is unrealistic to ask users to write descriptions in a formal Semantic Web language, such as OWL, the generation of a user-friendly UI turned out to be an issue of major importance.

4.1 Related Work

The importance of the UI to access complex knowledge bases is a topic largely studied in various research fields. Many authors recognize that, recently, there has been an increase of interest in the design of UI for systems based on semantic technologies (see, for instance, the SWUI workshops series). The main issue within this field is, quite obviously, the fact that the user interacting with a formally encoded knowledge base (e.g., an ontology) should not be exposed to the details of such a formal representation. In other words, "semantic technologies should be invisible to users" (Benjamins, 2008), p. 76.

The most traditional approaches facing this issue are devoted to support user queries aimed at searching for information encoded in formal knowledge bases (databases or ontologies). These approaches are mainly based on the translation of keywords (provided by the users) into formal queries (being SQL, or DL, or other formal languages); see, for instance, (Tran et al., 2007b).

Among many approaches to build UI to access knowledge bases, some studies propose to use NL-based UI; e.g. (Bernstein and Kaufmann, 2006), (Cimiano et al., 2008), (Damljanovic et al., 2010). Other approaches propose graphical (web-based) UI; e.g., (Thoméré et al., 2002).

(Bhagdev et al., 2008a) and (Bhagdev et al., 2008b) describe two tools supporting the user interaction with formal knowledge bases: (a) K-Forms is a tool enabling users to define knowledge structure and content through a user-friendly form-based UI; the tool then translate such a knowledge into a formal (RDF/OWL) representation; (b) K-Search is a tool that supports knowledge search and sharing.

In all the mentioned approaches the user input is a query aimed at extracting information from a (formal) knowledge base. In ARNEIS, the two types of users interacting with the system have different goals: the goal of a user from a software house is to provide the system with a description of a software solution, hoping that it will match some user requirements; the goal of a user from a SME is to "explain" the system the SME technological needs in order to find a suitable software solution. Thus, in

ARNEIS, in both cases, the user input is a complex description (of a software product or of a set of requirements) based on an existing ontology. In order to elicit this kind of information the system requires a UI that is more complex than the UI typically provided by the previously mentioned approaches, where the user simply provides the system with a query.

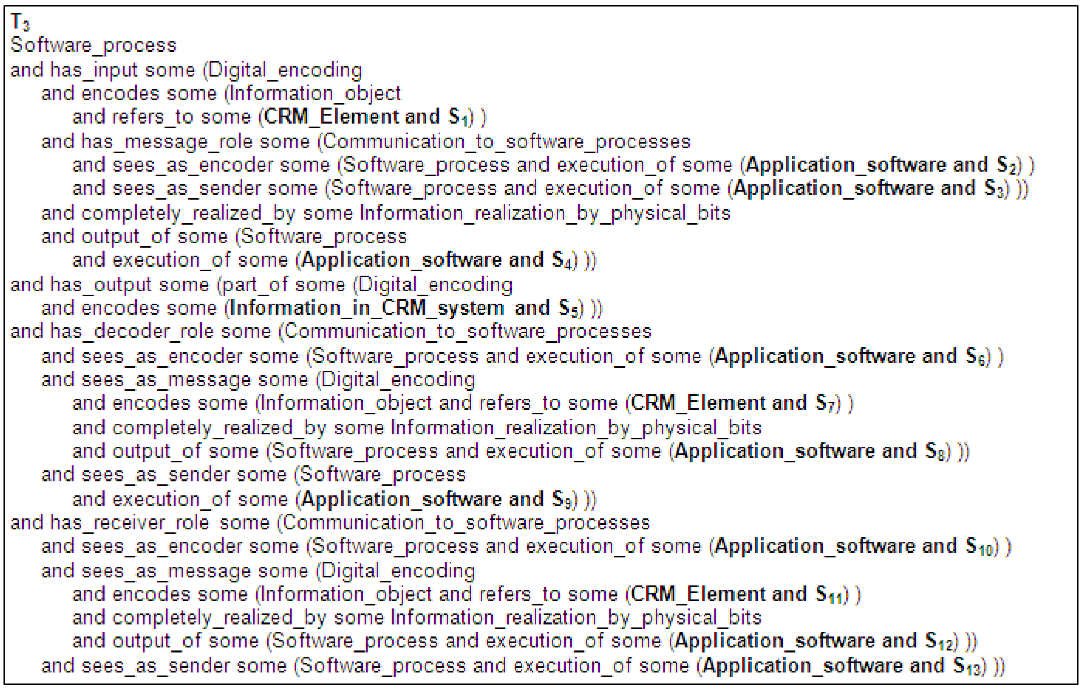
Other works concerned with user interaction with formal representations are authoring tools: in these cases the UI is aimed at enabling the user to design and populate a complex knowledge structure such as an ontology (e.g. Protégé: http://protege.stanford.edu/). With respect to these tools, ARNEIS has, not only a different goal, but also different types of users: in fact, ARNEIS users are not expert in knowledge representation and they could find it difficult to interact with an authoring system, even if provided with a smart (maybe graphical) UI.

Some approaches to Semantic Search propose to build ontology-based Information Retrieval (IR) systems, i.e. systems in which both resources and user queries are represented in a formal semantic language. For example (Tran et al., 2007a) proposes a resource model based on various integrated ontologies, expressed in OWL, enabling resources (documents) representation in terms of ontology elements (i.e., entities and axioms). As discussed by the authors, the issue of how to obtain formal semantic representations of resources is still open. Some steps have been made towards the automatic extraction of such representations from (textual) documents (e.g., Banko et al, 2007), but the results in this field are still poor. As an alternative, "a manual approach can be undertaken" (Tran et al., 2007a), p. 65. In particular, we think that a manual approach actually makes sense in those cases in which resources are not already available as documents. For instance, in a scenario as the one described above, a software house may have only simple brochures describing its software products; it seems to be reasonable to provide it with a tool supporting the construction of a formal semantic representation of their software products, which can be handled as resources by an ontology-based IR system. Our proposal is as a step in this direction.

In the following section we will describe how we faced the issue of the design of the user interaction in ARNEIS. However, the proposed approach aims at providing a general solution to solve the problem of how to build formal representations of resources in an ontology-based IR system.

4.2 User Interface

Concerning the type of UI most suited to our system, we chose on-line forms, since they are very familiar to web users; see (Bhagdev et al., 2008a). However, usually, users from SME, interacting with ARNEIS



```
T3
Software_process
and has_input some (Digital_encoding
    and encodes some (Information_object
        and refers_to some (CRM_Element and S1) )
    and has_message_role some (Communication_to_software_processes
        and sees_as_encoder some (Software_process and execution_of some (Application_software and S2) )
        and sees_as_sender some (Software_process and execution_of some (Application_software and S3) ))
    and completely_realized_by some Information_realization_by_physical_bits
    and output_of some (Software_process
        and execution_of some (Application_software and S4) ))
and has_output some (part_of some (Digital_encoding
    and encodes some (Information_in_CRM_system and S5) ))
and has_decoder_role some (Communication_to_software_processes
    and sees_as_encoder some (Software_process and execution_of some (Application_software and S6) )
    and sees_as_message some (Digital_encoding
        and encodes some (Information_object and refers_to some (CRM_Element and S7) )
        and completely_realized_by some Information_realization_by_physical_bits
        and output_of some (Software_process and execution_of some (Application_software and S8) ))
    and sees_as_sender some (Software_process
        and execution_of some (Application_software and S9) ))
and has_receiver_role some (Communication_to_software_processes
    and sees_as_encoder some (Software_process and execution_of some (Application_software and S10) )
    and sees_as_message some (Digital_encoding
        and encodes some (Information_object and refers_to some (CRM_Element and S11) )
        and completely_realized_by some Information_realization_by_physical_bits
        and output_of some (Software_process and execution_of some (Application_software and S12) ))
    and sees_as_sender some (Software_process and execution_of some (Application_software and S13) ))
```

Figure 2: "(Dynamic) acquisition of data from (another) enterprise application" template.

in order to look for software solutions supporting their CRM activities, are not technical experts and thus they can be in trouble even in filling in large on-line forms, possibly requiring technical skills in order to be completed. Thus, we are studying the possibility of implementing different types of UI for these users: a Natural Language (NL) UI, in which the SME users can freely express their requirements, and a UI based on a graphical representation of business processes (BP), based on the idea that business processes could be the form in which companies think about their business and management activities.

These two alternative UIs are a work in progress. The form-based UI, instead, can be suited for ICT company users, which are probably skilled enough about the functional and technological aspects of their software products or services. Moreover, such users are probably also more used to web-based interaction, often consisting in a sequence of on-line forms. Finally, such users are probably also more motivated to complete a possibly boring task, since they are describing the software solutions they offer, and this can be viewed as an effective promotion of their products and services.

Thus, in the following, we will concentrate on the form-based UI devoted to the acquisition of descriptions of software solutions by ICT companies.

4.3 Managing User Interaction

We claim that the design of the user interaction enabling ICT company users to provide a description of their software solutions could only be based on an analysis of the way in which users talk about CRM. For this reason, in the domain analysis phase, besides Ontology requirements, we also identified dialog topics, representing the key concepts of the descriptions of software solution supporting CRM activities (see Section 3).

In order to clarify how dialog topics are exploited in the system, let's look at an example. One of the templates representing dialog topics we extracted from our analysis corresponds to the concept of "(dynamic) acquisition of data from (another) enterprise application", which means that the described CRM software application can acquire data by directly communicating with another application (within the system, templates do not have a linguistic form, but only a formal, OWL, one. For the sake of readability, here we provide also a rough linguistic "translation"). Such a general concept is instantiated in different documents/interviews with more specific concepts by heterogeneous linguistic expressions: specific instances of this template include more specific concepts in place of "data" and in place of "enterprise application"; for example "real time acquisition of customers info from Ms Outlook", or "(management of) product records acquired from ERP software".

We decided to represent templates as ontological concepts belonging to an *Application Ontology* (Guarino, 1997), specific for ARNEIS, and linked to the upper CRM Ontology.

A template represents a structured concept in which there are some *variables* (*slots*). By default, such variables are filled in by generic concepts (e.g. "data"), that can be replaced by more specific concepts (e.g. "product data").

Figure 2 shows the OWL logical representation (generated by Protégé) of the class representing the template corresponding to the concept of "(dynamic) acquisition of data from (another) enterprise application".

In order to identify slots within a template, we needed a way lo "label" them, possibly without compromising the correctness of the OWL syntax. Thus, we decided to indicate them by adding, to each concept that represents a default slot filler, the expression *and* S_i, where S_i is a unique identifier (automatically generated) for that slot. Thus, for example, in Figure 2 the expression *(CRM_element and* S_1*)* identifies a slot (labeled S_1) filled in, by default, by the *CRM_element* class (intuitively speaking, *CRM_element* represents all the items that are typically involved in CRM: customers, products, orders, sales, and so on). If we aim at expressing the acquisition of data about customers (e.g., "customers info"), in the template instantiation, the class *CRM_element* will be replaced by a more specific one, i.e. *Customer* (subclass of *CRM_element*) and the expression *and* S_1 will be deleted; if we aim at expressing the acquisition of data about products (e.g., "product records"), *CRM_element* will be replaced by *Product_or_service* (subclass of *CRM_element*). Analogously, the class *(Application_software and* S_2*)* can be replaced by the class *Ms_Outlook*, or *ERP*.

At a first glance, this labeling solution may seem to bring to odd meanings. In fact, from a semantic point of view, by writing *Information_object ... refers_to some (CRM_element and* S_1*)*, we are actually saying that an *Information_object* refers to something that is, at the same time, a *CRM_element* and a S_1 (see Figure 2), which is not the meaning we would like to express. Thus, it is important to stress that the expression *and S_i* is used merely as a label, to identify slots within a template. We chose this solution because it preserves the syntactic correctness of the OWL representation. Moreover, it does not produce odd semantic results, since in ARNEIS not instantiated templates are never used in any form of semantic reasoning.

For each template, the UI Manager has to generate a set of web-forms, aimed at eliciting the information needed for filling in the slots. Since, typically, the knowledge on the basis of which such forms are generated does not change in time, web forms are pre-compiled off-line; they are re-generated only in case the knowledge bases are modified.

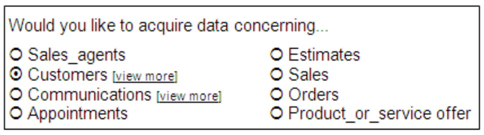
In order to enable the UI Manager to generate the web forms, the Knowledge Manager performs two (nested) steps:

(1) For each template in the Application Ontology, it extracts all the slots by looking for the expressions of the form *(C and S_i)*, where *C* is a named class of the reference Ontology.

(2) For each slot (identified by S_i), the Knowledge Manager extracts from the Ontology all the subclasses of *C* and provides the UI Manager with this information.

The UI Manager, in turn, for each slot, generates a web form in order to ask the user which subclass of *C* (if any) she wants to consider.

For example, Figure 3 shows the form asking the user to select the subclasses of *CRM_element*.

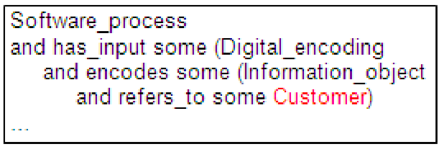


Would you like to acquire data concerning...

○ Sales_agents	○ Estimates
[✓ Customers] [view more]	○ Sales
○ Communications [view more]	○ Orders
○ Appointments	○ Product_or_service offer

Figure 3: web form for filling in the first slot of the template corresponding to "(dynamic) acquisition of data from (another) enterprise application".

The user selects the desired subclass (C_x) and the UI Manager sends this information back to the Knowledge Manager, that substitutes C_x in place of *(C and S_i)* within the corresponding template slot. The result (that refers to the example presented above) is shown in Figure 4.



```
Software_process
and has_input some (Digital_encoding
    and encodes some (Information_object
        and refers_to some Customer)
...
```

Figure 4: part of the instantiated template after the user answer.

There are some issues to be faced in order to make this mechanism work: (1) Which is the proper question the UI Manager should pose to the user (e.g., in Figure 3, "Would you like to acquire data concerning...")? (2) In some cases, the user may want to specify a more specific class (e.g., she wants to acquire data about golden customers, which are represented, within the Ontology, by the class *Golden_customer*, subclass of *Customer*). (3) It is quite common that the suited value for filling in a slot depends on the value assigned to another slot (of the same template). In the example above (see Figure 2), if we fill in the slot S_1 with *Customer*, meaning that we want to acquire information about customers, the slot S_5 (representing the knowledge structure that is modified by the data acquisition) have to be filled in with *Customer_database*. Moreover, many slots within a template are usually filled in by the same concept. For instance, in Figure 2, if we fill in the slot S_1 with *Customer*, the slots S_7 and S_{11} have to be filled in with *Customer* too.

In the following we will explain how we faced the mentioned issues.

(1) *Natural Language Questions.* The knowledge engineer who configures ARNEIS on a new domain is in charge of defining the Application Ontology containing the templates. When doing this, she identifies the slots and, for each slot, she provides a natural language question that will be used by the UI Manager to generate the form referring to that slot. The link between the slot S_i within the template T_j ($T_j.S_i$) and the corresponding question is stored in a *Configuration KB* which is accessed by the UI Manager during the form generation process.

(2) *Indirect Subclasses.* The Knowledge Manager, when extracting from the Ontology all the subclasses of *C* (step (2) mentioned above), actually extracts not only the direct subclasses of *C*, but the whole subtree. The UI Manager, on the basis of this information, includes a link to enable the user to optionally expand the subtree (see "view more" links in Figure 3): in this way she is enabled to select any (direct or indirect) subclass she is interested in.

However, given the complexity of the Ontology, listing all the (named) subclasses of a given class *C* could result in a too long list, containing concepts that are relevant from the formal point of view, but not from the user perspective. For this reason, we added to the Configuration KB, for each slot *(C and S_i)*, identified by the unique label $T_j.S_i$, a list of the subclasses of *C* that should be asked to the user. For example (the Configuration KB is an XML document; here we show its content in pseudo-code instead of XML for the sake of readability):

```
T₃.S₁ → {Sales_agent, Customer, ...}
```

The list of "subclasses to be asked" potentially contains both direct and indirect subclasses of *C*.

(3) *Dependencies between slots*. In order to take into account possible dependencies between slot fillers, the Configuration KB also contains some if-then rules representing such dependencies; for instance:

```
    IF T₃.S₁ = Customer
    THEN T₃.S₅ = Customer_database

    IF T₃.S₁ = Customer
    THEN T₃.S₇ = Customer
```

These dependency rules enable the UI Manager to avoid asking the user useless questions: after having asked the filler for slot S_1, slots S_5 and S_7 will be filled in automatically.

When the user has completed all the forms proposed by the ARNEIS system, the instantiated templates are saved in the Semantic Descr KB (see Section 2): the set of such instantiated templates is the OWL representation of the software solution supporting CRM, proposed by the ICT company.

5 DISCUSSION AND CONCLUSIONS

In this paper, we described the management of the user interaction within the ARNEIS system, an "intelligent" web-based repository of software solutions, that enables ICT companies to upload a description of their software solutions for business automation, and SME to find software products or services supporting their business and management activities. The functionality of such an intelligent repository is based on a semantic representation of the domain knowledge, which represents the shared vocabulary to express both software descriptions and technological requirements. In particular, we described the management of a user-friendly interaction enabling software houses to upload the semantic representation of the description of their products and services. The approach we proposed within the ARNEIS scenario suggests a general solution to solve the problem of how to build formal representations of resources in an ontology-based IR system.

Our approach is strongly based on a domain analysis that takes into account how users talk about their business activities and the software applications that could support such activities. In fact, the mechanism described in this paper relies on templates stored in an Application Ontology and a Configuration KB, that have to be built by the knowledge engineer who configures the system on a domain (e.g., CRM). This task could represent a considerable knowledge acquisition effort. However, the definition of templates and Configuration KB is typically done once and usually does not require frequent updates. Moreover, the Configuration KB definition can be easily supported by a user-friendly tool that provides the knowledge engineer with simple mechanisms supported by system defaults (e.g., the lists of the subclasses of *C* that have to be proposed to the user can be automatically built by default including all the available subclasses; the knowledge engineer will be enabled to simply delete those ones she thinks are not meaningful for the end user).

Since the knowledge acquisition effort is a key aspect of knowledge-based systems like ARNEIS, the development of the mentioned user-friendly tool, supporting the knowledge engineer in the system configuration on new domains, represents our main future work.

REFERENCES

Alonso, G., Casati, F., Kuno, H., Machiraju, V., 2004. *Web Services*. Springer, Berlin.

Antoniou, G., Van Harmelen, F., 2004. *A Semantic Web Primer*. The MIT Press, Cambridge MA.

Banko, M., Cafarella, M. J., Soderland S., Broadhead M., Etzioni, O., 2007. Open information extraction from the web. In *IJCAI 2007, 20th International Joint Conference on Artificial Intelligence*, 2670-2676. Morgan Kaufmann.

Benjamins, V. R., 2008. Near-Term Prospects for semantic Technologies. In *IEEE Intelligent Systems*, 23(1), 76-88.

Bernstein, A., Kaufmann, E., 2006. GINO - A Guided Input Natural Lanaguage Ontology Editor. In *ISWC 2006, 4th International Semantic Web Conference*, 114–157. Springer.

Bhagdev, R., Chakravarthy, A., Chapman, S., Ciravegna, F., Lanfranchi, V., 2008a. Creating and Using Organisational Semantic Webs in Large Networked Organisations. In *ISWC 2008, 6th International Semantic Web Conference*, 723-736. Springer.

Bhagdev, R., Chapman, S., Ciravegna, F., Lanfranchi, V., Petrelli, D., 2008b. Hybrid Search: Effectively combining Keywords and Semantic Search. In *ESWC 2008, XX European Semantic Web Conference*, 554-568. Springer.

Magro, D., Goy, A., 2008a. The Business Knowledge for Customer Relationship Management: an Ontological Perspective. In *Int. ACM Workshop on Ontology-supported business intelligence*. ACM Press.

Magro, D., Goy, A., 2008b. Towards a first Ontology for Customer Relationship Management. In *IEEE/ACM Workshop on Applied Ontologies in Distributed Systems*, 637-643. ACM Press.

Goy, A., Magro, D., Prato, F., 2008. ARNEIS: A Web-based Intelligent Repository of ICT Solutions for E-business. In *Int. ACM Conference on Information Integration and Web-based Application & Services*, 403-406. ACM Press.

Cimiano, P., Haase, P., Heizmann, J., Mantel, M., 2008. ORAKEL: A Portable Natural Language Interface to Knowledge Bases. In *Data & Knowledge Engineering*, 65(2), 325–354.

Creeger, M., 2009. CTO roundtable: Cloud computing. In *Communications of the ACM*, 52(8), 50–65.

Damljanovic, D., Agatonovic, M., Cunningham, H., 2010. Natural Language Interfaces to Ontologies: Combining Syntactic Analysis and Ontology-based Lookup through the User Interaction. In *ESWC 2010, 7th European Semantic Web Conference*, 106-120. Springer.

Dikaiakos, M. D., Pallis, G., Katsaros, D., Mehra, P., Vakali, A., 2009. Cloud computing. Distributed internet computing for IT and scientific research. In *IEEE Internet Computing*, 13(5), 10–13.

Freeland, J., 2005. *The Ultimate CRM Handbook*. McGraw-Hill, New York.

Guarino, N., 1997. Understanding, Building and Using Ontologies. In *Int. Journal of Human-Computer Studies*, 46, 293-310.

Thoméré, J., Barker, K., Chaudhri, V., Clark, P., Eriksen, M., Mishra, S., Portr, B., Rodriguez, A., 2002. A Web-based Ontology Browsing and Editing System. In *AAAI 18th National Conference on Artificial Intelligence*, 927-934. AAAI Press.

Tran, T., Bloehdorn, S., Cimiano, P., Haase, P., 2007a. Expressive Resource Descriptions for Ontology-Based Information Retrieval. In *ICTIR 2007, 1st International Conference on the Theory of Information Retrieval*, 55-68. Springer.

Tran, T., Cimiano, P., Rudolph, S., Studer, R., 2007b. Ontology-based Interpretation of Keywords for Semantic Search. In *ISWC 2007, 5th International Semantic Web Conference*, 523-536. Springer.