ONTOLOGY INFORMATION RETRIEVAL THROUGH NATURAL LANGUAGE INTERFACES

Domingo Senise de Gracia
MSc in Artificial Intelligence
Universidad Politécnica de Madrid
Boadilla del Monte, 28660 Madrid (Spain)
domingo.senise@haitta.com

ABSTRACT

In this article I write about the need to have a more user-centric attitude regarding the ontology retrieval information if our aim is to universalize this kind of knowledge representation, and to this end I introduce two approaches which follow this line of research: FREyA, an English natural language interface developed by the researchers Danica Damljanovic, Milan Agatonovic, and Hamish Cunningham from the Department of Computer Science of the University of Sheffield; and a German natural language interface developed by the researchers Irina Deines and Dirk Krechel from the University of Applied Sciences RheinMain in Germany.

Author Keywords

FREyA, Natural language interface, NLI, ontologies, Closed-domain question answering systems, Linked-Open-Data project, ORAKEL, AquaLog, FrameMapper, PANTO, SPARQL, RDF, QuestIO, Querix, SWAT, Ginseng, GermaNet

THE CHALLENGE

Semantic data is the key for an efficient information retrieval. It relies on a well-defined structure and enables automated processing.

More and more RDF ontologies -containing semantic data- emerge across the web and are getting interlinked through the Linked-Open-Data project. RDF data can be accessed via SPARQL which seems, by now, the only query language which provides a precise access to semantic data.

However, for a common user a query formulation in SPARQL syntax is complicated and difficult. It requires at least medium skills of this coding language and knowledge of the ontology structure.

Owing to this overstraining feeling amongst common users, there is an emerging trend towards a more user-centered search interface aiming at good usability and correct answers.

With large datasets such as Linked-Open-Data available, these user-friendly interfaces will bring the advantages of the semantic data closer to the casual users. Several recent studies have shown user preference to Natural Language Interfaces (NLIs) in comparison to others.

Throughout the last years research has been very active in developing interfaces for accessing structured knowledge: from faceted search, where knowledge is grouped and represented through taxonomies, to menu-guided and form-based interfaces. Nonetheless, these interfaces still require that the user is familiarized with the queried knowledge structure.

Natural Language Interfaces (NLIs), which are often referred as closed-domain Question Answering (QA) systems, have a very important role as they are intuitive for the end users and preferred to keyword-based, menu-based or graphical interfaces. NLIs can process queries formulated in natural language and transform them into a SPARQL query to be processed by the system. They are accepted most by the user, even though they might suffer from worse recall and precision.

RELATED WORK

In 2006 the researchers A. Bernstein and E. Kaufmann from the University of Zurich developed two different controlled language interfaces to the semantic web: SWAT and Ginseng.

SWAT -the Semantic Web ACE Transformer- was an interface which allowed formulating queries in Attempto Controlled English (ACE), a subset of natural English. Each ACE query was translated into a discourse representation structure –a variant of the language of first-order logic– that was then translated into a N3-based semantic web querying language, using an ontology-based rewriting framework.

The approach offered great potential for bridging the gap between the logic-based semantic web and its real-world users, since it allowed users to query the semantic web without having to learn an unfamiliar formal language.

Ginseng -Guided Input Natural Language Search Engine- relied on a simple question grammar which got dynamically extended by the structure of an ontology, to guide users in formulating queries in a language seemingly akin to English. Basing on the grammar Ginseng translated then the queries into SPARQL, which allowed their execution.

In order to transform the original query into SPARQL, the current NLI systems usually apply natural language processing methods, such as tokenization, stemming, part-of-speech tagging and lemmatization to the labels of the domain ontology and to the user query. In general, the user query is analyzed by a constituency parser too.

NLI systems which have a good performance require a customization, such as in the case of ORAKEL; several systems have been developed for which the customization is not mandatory, for instance PANTO, Querix, AquaLog, QuestIO, NLP-Reduce. However, the customization usually improves the recall.

In the case of ORAKEL, customization is performed through the user interaction, using a software called FrameMapper, where the linguistic argument structures -such as verbs or nouns with their arguments- are mapped to the relations in the ontology.

AquaLog is capable of learning the user's jargon in order to improve his experience by the time. Its learning mechanism is good in a way that it uses ontology reasoning to learn more generic patterns, which could then be reused for the questions with similar context.

Querix is another ontology-based question answering system which relies on clarification dialogs in case of ambiguities. Both Querix and PANTO expand their knowledge base by synonyms from WordNet. They store a resource's natural-language information and its synonyms in a gazetteer, which is either a relational database or a hash map. Since these data stores are simply structured, the identification of resources is very fast. On the other hand, complex structured information such as hierarchies cannot be archived. Therefore no other semantic relations besides synonymy are considered in the identification process of these solutions.

Regarding the two solutions I'm going to describe in this article, I must begin saying that Damljanovic, Agatonovic, and Cunningham's system -FREyA- is intended to be used by end-users from the start. This may be seen as an overload since users will be heavily engaged into dialogs, until the system learns enough to be able to automatically suggests the correct answer. Nonetheless the primary goal of the dialog in this system is not only to resolve ambiguities, but also to map question terms to the relevant ontology concepts.

With respect to the German NLI system it uses RDF to structure a resource's natural-language information and its semantic relations. Lexical entries as well as lexical and conceptual relations from the German lexicon GermaNet are modeled in RDF and linked to the domain ontology. As a result, a resource cannot only be identified by its synonyms, but also by any other semantic relation that connects it to the user-query extract.

FREYA

Although many NLIs to ontologies have been developed, those that have reasonable performance are domain-specific and tend to require customisation for each new domain which, from a developer's perspective, makes them expensive to maintain.

FREyA -developed in 2010 and named after Feedback, Refinement and Extended Vocabulary Aggregation- combines syntactic parsing with the knowledge encoded in ontologies in order to reduce the customisation effort. If the system fails to automatically derive an answer, it will generate clarification dialogs for the user. The user's selections are saved and used for training the system in order to improve its performance over time.

Most QA systems contain the classifier module which is used to detect the question category or the type of the question. The successful parsing is based on this identification. However, the syntactic patterns for this classification are usually derived from the dataset which must be large in order to work efficiently.

Damljanovic, Agatonovic, and Cunningham's approach is to use the knowledge encoded in ontologies as the primary source for understanding the user's question, and only then try to use the output of the syntactic parsing in order to provide the more precise answer. Questions do not need to fall within the predefined categories. If the system is not able to automatically derive an answer, it will generate clarification dialogs. The user's choice is then saved and used for training the system in order to improve its performance over time.

Before developing FREyA, Damljanovic, Agatonovic, and Cunningham's developed QuestIO -Question-based Interface to Ontologies, which translated a Natural Language (NL) or a keyword-based question into SPARQL, and returned the answer to the user after executing the formal query against an ontology.

Although this approach used very shallow NLP, it was quite efficient for very small and domain-specific ontologies. It performed quite well for the set of ill-formed and grammatically incorrect questions. However, the trade-off was that many grammatically correct questions which did require more deep analysis remained unanswered, or partially answered. For example, if the question was *What is the largest city in Nevada?* QuestIO could list cities in Nevada, but it ignored the word *largest* which was in this case crucial to deliver semantic meaning. Furthermore, QuestIO displayed the result of executing SPARQL queries as a table in which the user found the answer.

From this experience the Sheffield researchers began wondering how to:

- Improve understanding of the question's semantic meaning.
- Provide the concise answer to the user's question.
- Communicate the system's interpretation of the query to the user.
- Assist the user formulate the query which falls within the boundaries of the system capabilities.

The combination of these four requirements form the core of FREyA.

The functioning of FREyA can be broken down into the four following steps:

1. Identification and verification of ontology concepts
2. Generating SPARQL
3. Identification of the answer type and presenting the results to the user.

4. Learning.

1.- Identification and Verification of Ontology Concepts

Damljanovic, Agatonovic, and Cunningham's algorithm for translating a NL question into the set of Ontology Concepts (OCs) combines the syntactic parsing with ontology reasoning in order to identify the user's information need correctly. In cases when the algorithm does not derive conclusions automatically, it generates suggestions for the user. By engaging the user into the dialog, there is a better chance of identifying her/his information need.

Identification of Ontology Concepts

The knowledge available in the ontology is used to identify the ontology-based annotations in the question, which are called Ontology Concepts. Generated annotations contain links to ontology resources (e.g. URIs). If there are ambiguous annotations in the query, the user is engaged into the dialog. For example, if someone is enquiring about Mississippi, it cannot be derived automatically whether OC refers to *geo:River*, or *geo:State*. To resolve this ambiguity, a clarification dialog is generated where the user selects one of the two. Note that disambiguation rules are applied, which are based on the ontology reasoning before the clarification dialog is modeled. For instance, for the question *Which rivers flow through Mississippi?*, modeling a clarification dialog is not necessary, as due to the context of the question it is derived automatically that Mississippi refers to *geo:State*.

In FREyA it is used GATE application, and an ontology-based gazetteer called OntoRoot Gazetteer to perform this step. OntoRoot Gazetteer relies on the human understandable lexicalizations of ontology resources and therefore, the quality of produced annotations depends directly on them. Seldom ontology resources are followed by human understandable lexicalisations (e.g., labels). In this case NLIs to ontologies would have to translate any additional semantic meanings into the relevant operations with the ontology concepts; e.g. superlative means applying maximum or minimum function to the datatype property value.

Identification of Potential Ontology Concepts

Potential Ontology Concepts (POCs) are derived from the syntactic parse tree, and refer to question terms which could be linked to an ontology concept. Syntactic parse tree is generated by Stanford Parser. Several heuristic rules are used in order to identify POCs. For example, each NP (noun phrase) or NN (noun) is identified as a POC. Also, if a noun phrase contains adjectives, these are considered POCs as well. Next, the algorithm iterates through the list of POCs, attempting to map them to OCs.

Mapping POCs to OCs

A Potential Ontology Concept is mapped to Ontology Concept in two ways:

1. Automatically: if it overlaps with an Ontology Concept in a way that OC spans over POC:

- Both POC and OC refer to exactly the same text span in the question; for example, in *Which rivers flow through Texas?*, *Rivers* can be identified as OC, as referring to the class *geo:River*, while it can also be identified as POC. In this case, POC is automatically mapped to OC.

- POC refers to the text span which is contained within the span to which OC refers ($POC \subset OC$);

2. By engaging the user: when the user verifies it by choosing it from the list of the available suggestions.

Generating Suggestions

Suggestions are generated for each POC which does not overlap with OC, or in cases when POC spans over OC. First, the algorithm identifies the closest OC to this POC by walking through the syntax tree, and then uses ontology reasoning to generate suggestions.

Option *none* is always added to the list of suggestions. This allows the user to ignore suggestions, if they are irrelevant. In the following table sample queries and generated suggestions for relevant POCs are shown:

Query	POC	Closest OC	Suggestions
population of cities in california	population	geo:City	1. city population 2. state 3. has city 4. is city of 5. none
population of california	population	geo:california	1. state population 2. state pop density 3. has low point ... n. none
which city has the largest population in california	largest population	geo:City	1. max(city population) 2. min(city population) 3. sum(city population) 4. none

The task of creating and ranking the suggestions before showing them to the user is quite complex, and this complexity arises as the queried knowledge source grows.

Ranking Suggestions

Initial ranking is based on the string similarity between POC and suggestions, and also based on synonym detection as identified by Wordnet nd Cyc. For string similarity Monge Elkan metrics with Soundex algorithm are combined.

Soundex algorithm compensates for any spelling mistakes that the user makes: this algorithm gives a very high similarity to the two words which are spelled differently but would be pronounced similarly.

2.- Generating SPARQL

After all POCs are resolved, the query is interpreted as a set of OCs. For instance, the query *What is the highest point of the state bordering Mississippi?* would be translated into the list of the following OCs:

geo:isHighestPointOf	geo:State	geo:border	geo:mississippi
PROPERTY	CLASS	PROPERTY	INSTANCE

Next step is generating set of triples from OCs, taking into account the domain and range of the properties. Last step is generating SPARQL query.

3.- Answer Type Identification

The result of the SPARQL query is a graph and, in order to show the concise answer, the answer type of the question must be identified. To achieve this, the output of the syntactic parsing is combined with the ontology-based lookup, coupled with several heuristic rules.

4.- Learning

The used approach is inspired by the Reinforcement Learning methodology in order to improve ranking of suggestions shown to the user.

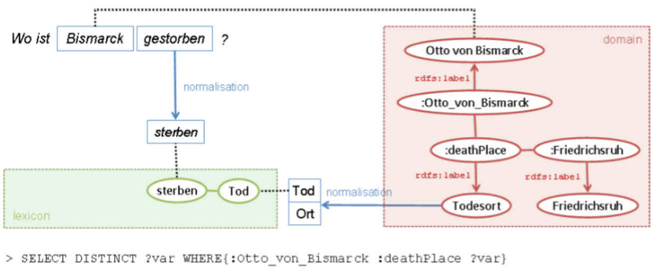
Damljanovic, Agatonovic, and Cunningham use a semi-supervised approach on the one hand to avoid the automatic classification and to allow users to enter queries of any form; and on the other hand, to minimize customisation of the NLI system which would be required upon using supervised learning.

The first important aspect of RL is the identification of the goal to be achieved: in this case the correct ranking of suggestions. Each suggestion has its initial ranking calculated based on synonym detection and string similarity. These are used in the untrained system. Each time the suggestion is selected by the user, it receives a reward of +1 while all alternative ones receive -1. The system then learns to place the correct suggestion at the top for any *similar* questions. *Similar* is identified by a combination of a POC and the closest OC. This increases robustness of the learning mechanism as the learning model is not updated per question, but per each combination of POC and the closest OC. In addition, some generalization rules derived from the ontology are applied. For example, if the closest OC is *geo:Capital*, its superclass *geo:City* is saved in the learning model in order to reuse the same rule for all cities, not only capitals.

THE GERMAN NLI

This NLI system developed in 2013 is able to answer German user queries addressing RDF data labelled in German. Aiming for high precision and recall, this NLI system concentrates on a semantic component which mainly contributes to a correct generation of the SPARQL query. For example, regarding the German question *Wo ist Bismarck gestorben?* ("Where did Bismarck die?"). According to the domain ontology, the expected answer is the resource :Friedrichsruh. Thus, the user query has to be transformed into the following SPARQL query:

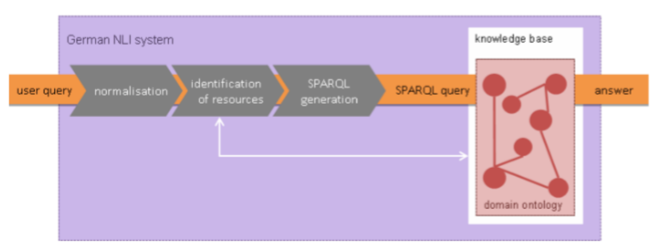
SELECT DISTINCT? var WHERE{:Otto_von_Bismarck :deathPlace ?var}



When translating the user query into SPARQL the identification of resources is a crucial factor. Text-based identification and a semantic-based identification of resources are distinguished. Based on text-based similarity between the query extract ”Bismarck” and the German label ”Otto von Bismarck”, the corresponding resource *:Otto_von_Bismarck* is recognized easily by the system. However, no text-based similarity exists between ”gestorben” (died) and ”Todesort”, which is the label of the required resource *:deathPlace*. A semantic-based similarity indicates a relation between those two terms. Thus, if a phrase or term in the user's query is semantically related to a resource’s label, a special treatment is applied. The algorithm first normalizes the question’s term ”gestorben” and the resource’s label ”Todesort”. The result is the infinitive form ”sterben” (die) and the noun component ”Tod” (death), which is a part of the noun compound ”Todesort”. Then, a German lexicon is used to resolve the semantic relation between ”sterben” and ”Tod”.

1.- System Design

The system consists of two components, namely the translation process and the knowledge base. The translations process transforms the user query into SPARQL concentrating on the identification of resources, since they are essential in the graph pattern of the SPARQL query. A normalization needs to be applied on the user query and the labels first, as illustrated by the system workflow in the figure below. The natural-language-processing methods such as parsing, lemmatizing, part-of-speech tagging and noun-compound splitting constitute the normalization and are accomplished by the constituency parser Stanford Parser, the part-of-speech tagger TreeTagger and the noun-compound splitter BananaSplit. Then, in case of a semantic-based identification a lexicon is used to resolve the path between normalized user-query extract and semantically similar resource. After text-based and semantic-based identification of resources, the translation process concludes by generating the SPARQL query. Thereby, an answer type is determined and triples for the graph pattern are built. Like in many other natural-language interfaces, the answer-type determination occurs dictionary- and rule-based. Finally, the SPARQL query is sent to the domain ontology and the returned answer is passed to the user. Note the whole process in this figure:



Natural-language information resulted by the normalization of a resource’s label is linked to the domain ontology as well as the German lexicon GermaNet, such as needed for an identification of recourse :deathPlace in the previous example.

2.- Expansion of the Domain Ontology

Since GermaNet contains Germans common words, almost every normalized user-query extract can be encountered in the ontology. The normalized user-query extract is the starting point of the path while its semantically similar resource represents the ending point. Hence, if a user-query extract has a semantically similar resource in domain ontology, it can be identified by walking the path of GermaNet's lexical and conceptual relations from starting to ending point. The more paths can be created, the more semantically similar ontology resources can be recognized.

Since the linking of domain ontology and GermaNet can only be achieved when both share the same representation format, mapping GermaNet to RDF needs to be accomplished first.

Considering GermaNet extension -most German's common words are included, most lexical entries of the domain lexicon can either be encountered in it by their canonical name or by their lemma. For each domain-lexicon entry that is available in GermaNet, a connection is established between the corresponding domain resource and the GermaNet entry.

After the linking process almost every domain-lexicon entry, the ending point of the path, is connected to GermaNet. Few unlinked domain-lexicon entries may be encountered, basically caused by two reasons: very domain-specific lexical entries, which are mostly named entities, are not available in lexicons; besides, results generated by the morphological analysis may be incorrect. In that case, either no lemma is returned or a noun compound could not be split or a wrong lemma respectively wrong noun components are found.

3.- Identification of Semantically Similar Resources

The identification of domain-ontology resources starts with the determination of potential resources. Those are user-query extracts for which may exist an equivalent resource in the domain ontology. Nominal phrases, nouns, named entities and verbs are potential resources and can be recognized by part-of-speech tags.

If a similar resource exists in domain ontology, it represents the ending point of the path. Since in the initialization phase of the section of Expansion of Domain Ontology links between domain-ontology resources and lexical entries were established, paths from lexical entries to their similar resources are available in the RDF store. Only those domain-ontology resources, which are very specific, cannot be linked to the lexicon. Nevertheless, those specific resources usually don't have a semantic equivalent and can be encountered by their name in text-based identification.

EVALUTION OF BOTH NLI SYSTEMS

Damljanovic, Agatonovic, and Cunningham evaluated **FREyA** on the 250 questions from the Mooney Geoquery dataset. They evaluated correctness, ranked suggestions, and learning mechanism.

Correctness

The correctness of FREyA was reported in terms of precision and recall.

Table 3 shows the number of questions correctly answered automatically, as opposed to those which have been answered correctly only after engaging the user into at most 2 clarification dialogs. Finally, there is a system failure to answer questions correctly in 7.6% of the time (e.g., questions with negation). Recall and precision values are equal, reaching 92.4%:

Correct			Incorrect
no dialogs	1 dialog	2 dialogs	
72	127	32	19
28.8%	50.8%	12.8%	7.6%

Ranked Suggestions

The Mean Reciprocal Rank (MRR) was reported for the performance of the ranking algorithm. MRR is a statistic for evaluating any process that produces a list of possible responses (suggestions in this case) to a query, ordered by probability of correctness.

It was labeled manually the correct ranking for suggestions which was generated upon running FREyA with 250 questions. This was the gold standard against which the ranking mechanism achieved MRR of 0.81.

Learning Mechanism

From the set of 250 questions, 103 were randomly selected which required one clarification dialog with the user in order to get the correct answer. Then, the initial ranking algorithm was run and compared results with manually labeled gold standard. MRR was 0.72.

103 questions were grouped then by OC, and training and evaluation sets from each group were randomly chosen. This process was repeated two times: these two iterations were independent and they had both been performed starting with an untrained system. The table below shows the structure of the dataset grouped by OC for both iterations:

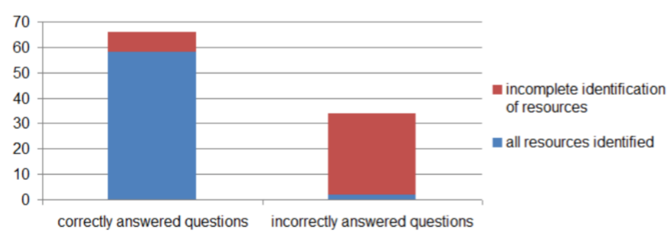
	Iteration 1		Iteration 2	
OC	**Training**	**Evaluation**	**Training**	**Evaluation**
geo:State	26	19	19	26
geo:City/Capital	20	19	19	20
geo:River	12	6	9	9
geo:Mountain	1	0	0	1
total	59	44	47	56

After learning the model with 59 questions from the iteration 1, MRR for the evaluation questions (44 of them) reached 0.98. Overall MRR (for all 103 questions) increased from 0.72 to 0.77. After training the model with 47 questions during the iteration 2, overall MRR increased to 0.79. Average MRR after running these two experiments was 0.78, which shows the increase of 0.06 in comparison to MRR of the initial rankings. Therefore, the learning algorithm improved the initial ranking by 6%.

Regarding the German NLI, the researchers Deines and Krechel used precision and recall, as measurement units, being slightly adapted though. Recall is the number of questions correctly answered by the search interface, divided by the total number of questions. Precision measures the number of questions correctly answered divided by the number of questions which were answered at all.

250 resources which basically specified persons were chosen from the ontology schema of DBpedia 3.7 to evaluate the German NLI system. Resources without a German label were translated semi-automatically. In addition, German labelled resources about 1000 persons were loaded from a SPARQL endpoint. The NLI system was asked 100 questions addressing this data set. 66 questions were answered correctly, 34 either partial or incorrectly. Since the NLI system always returned an answer, recall and precision achieved 66%.

According to the below figure, the evaluation results show clearly a coherence between the identification of resources and a correct answer:



In 58 of 66 correctly answered questions, all resources required for a proper SPARQL generation were identified. On the other hand, in 32 of 34 wrongly answered questions only a partial or none identification of resources was accomplished. Hence, a complete identification of resources contributed essentially to a correct translation of user query into SPARQL.

In a total of 40 questions an incomplete resource recognition was detected. The evaluation results reveal, that in 27 of these 40 questions an identification of resources fails because of missing relations in GermaNet. The lack of especially pertonymy relations, such as between the lexical entries "schreiben" (write) and "Schriftsteller" (writer) or between "heiraten" (marry) and "Ehegatte" (spouse), prevented a path construction which was needed for a semantic-based identification. Hence, questions like *Welche Bücher hat Stephen King geschrieben?* (Which books did Stephen King write?) or *Mit wie vielen Männern war Marilyn Monroe verheiratet*? (How many men Marilyn Monroe was married to?) could not be answered because the resources *:writer* labelled as "Schriftsteller" and *:spouse* labelled as "Ehegatte" could not be found.

Other reasons for incorrect translations of user query were special expressions like the verb "kommen" (to be from) in *Woher kommt X?* (Where is X from?) or the pronoun "derselben" (the same) in *Werden Futurama und Simpsons von derselben Person produziert?* (Are Futurama and Simpsons produced by the same person?). Also transitive verbs like "beeinflussen" (influence) were translated incorrectly, such as both questions *Wen hat X beeinflusst?* (Who did X influence?) and *Wer hat X beeinflusst?* (Who was X influenced by?) produced the same answer.

BUSINESS STATE OF THE ART

In the Spanish scenario, there are two companies which offer these NLI systems to their customers: Inbenta and Anboto,

Inbenta is focused on a field of AI that deals with machines understanding human languages. They take their customers' content and organize it in their knowledge base. This KB organizes the content to all the potential questions humans might ask and generate the corresponding answers. In order to access these answers a contextual trigger is needed to create the connection. Their trigger is a specialized intelligent search engine that searches based on contextual meaning from user questions instead of keywords. This allows their AI to fully understand the meaning of human language.

Anboto provides technological solutions to enable an easy interaction in natural language between human beings and machines. Through open free (non-monitored) dialogue, Anboto makes it possible for customers to obtain and retrieve information, buy products and services, and experience a more natural human-computer interaction. Disabled people will also find this system offers an easier access to those services. Anboto's solutions significantly reduces the direct interaction with human agents for tasks that are repetitive, mechanical, etc.

In the European scenario, we find the company Expert Systems. Its mission is to develop software that comprehends written language in the way people do, with the speed and precision required to discern, manage and utilize large volume of strategic information. Based upon a patented technology that employs millions of definitions, concepts and relationships, its solutions read and understand multiple languages the way people do. This company bought two divisions of the Spanish semantic technology company ISOCO a year and a half ago.

And in the worldwide scenario the two most important companies are IBM Watson and Cycorp.

Cycorp is a leading provider of semantic technologies that bring a new level of intelligence and common sense reasoning to a wide variety of software applications. The Cyc software combines common sense ontology and knowledge base with a powerful reasoning engine and natural language interfaces to enable the development of novel knowledge-intensive applications. As a premier knowledge-based technologies research and development company, Cycorp leverages its cutting edge innovations in knowledge representation, machine reasoning, natural language processing, semantic data integration, and information management and search. The Cyc project was founded in 1984 by Dr. Douglas Lenat as a lead project in the Microelectronics and Computer Technology Corporation (MCC). In 1994, Cycorp was founded to further develop, commercialize, and apply the Cyc technology.

And finally the true star in this play: **IBM Watson**

Watson is a question answering computer system capable of answering questions posed in natural language, developed in IBM's DeepQA project by a research team led by principal investigator David Ferrucci. Watson was named after IBM's first CEO and industrialist Thomas J. Watson.

IBM built Watson to apply advanced natural language processing, information retrieval, knowledge representation, automated reasoning, and machine learning technologies to the field of open domain question answering. According to IBM, more than 100 different techniques are used to analyze natural language, identify sources, find and generate hypotheses, find and score evidence, and merge and rank hypotheses. The sources of information for Watson include encyclopedias, dictionaries, thesauri, newswire articles, and literary works. Watson also used databases, taxonomies, and ontologies. Specifically, DBPedia, WordNet, and Yago were used.

CONCLUSION

Taking into account common users have huge problems even with the simplest Boolean expressions and logically with the formalisms underlying the semantic web, how could we bridge the gap between the semantic web and real-world users, taking into account the state-of-the-art of the technology and considering the advantages and limitations shown upon explaining FREyA and the German NLI?

It's very likely that the trend and final solution will be the use of natural language interfaces (NLIs) to retrieve information from ontologies, since human beings will try to find the most natural way of dealing with machines. Just consider, for instance, the evolvement of the coding language since its inception some decades ago: from machine code -totally unnatural to human beings- to the current syntax of some coding languages, which borrow terms and expressions directly from natural languages.

Considering the state-of-the-art of the technology and hoping an ongoing evolution towards excellent standards of usability, the best solution for the casual and occasional user of ontologies may lie between the freedom of a full NLI and the structuredness of a formal query language, aiming always at implementing eventually the vision of the semantic web into the closest human-centric realization.

Ontologies are currently the unique and best method to take advantage of the full knowledge power of the Internet. If we want to democratize and extend their use -avoiding they may expire as other technologies due to lack of connection with human standards, we should strive for humanizing ontologies and it implies using natural language interface/interaction.

REFERENCES

1. Danica Damljanovic, Milan Agatonovic, and Hamish Cunningham, *Natural Language Interfaces to Ontologies: Combining Syntactic Analysis and Ontology-Based Lookup through the User Interaction,* Department of Computer Science, University of Sheffield.

2. Valentin Tablan, Danica Damljanovic, and Kalina Bontcheva, *A Natural Language Query Interface to Structured Information,* Department of Computer Science, University of Sheffield.

3. Irina Deines and Dirk Krechel, *A German Natural Language Interface for Semantic Search*, University of Applied Sciences RheinMain, Germany.

4. Esther Kaufmann, *Talking to the Semantic Web - Query Interfaces to Ontologies for the Casual User,* Dynamic and Distributed Information Systems, University of Zurich.

5. http://www.ibm.com/smarterplanet/us/en/ibmwatson/

6. http://www.cyc.com/

7. http://www.expertsystem.com/es/

8. http://www.anbotogroup.com/es/

9. https://www.inbenta.com/es