Peer-reviewed version available at *Symmetry* **2018**, *10*, 268; doi:10.3390/sym10070268

Article

A Prototype of Speech Interface based on Google Cloud Platform to Access a Semantic Website

Jimmy Aurelio Rosales-Huamaní [1,*], José Luis Castillo-Sequera [2], Juan Carlos Montalvan-Figueroa [1] and Joseps Andrade-Choque [1]

[1] National University of Engineering, Lima, Peru; jrosales@uni.edu.pe; jmontalvanf@uni.pe and jandradec@uni.pe

[2] Department of Computer Science, Polytechnic School, University of Alcala, 28871 Alcalá de Henares, Spain; jluis.castillo@uah.es

* Correspondence: jrosales@uni.edu.pe; Tel.: +51-1-381-5630

Abstract: The main restriction of the Semantic Web is the difficult of the SPARQL language, that is necessary to extract information from the Knowledge Representation also known as ontology. Making the Semantic Web accessible for people who do not know SPARQL, is essential the use of friendlier interfaces and a good alternative is Natural Language. This paper shows the implementation of a friendly prototype interface to query and retrieve, by voice, information from website building with the Semantic Web tools. In that way, the end users avoid the complicated SPARQL language. To achieve this, the interface recognizes a speech query and converts it into text, it processes the text through a java program and identifies keywords, generates a SPARQL query, extracts the information from the website and read it in voice, for the user. In our work Google Cloud Speech API makes Speech-to-Text conversions and Text-to Speech conversions are made with SVOX Pico. As results, we have measured three variables: The success rate in queries, the response time of query and a usability survey. The values of the variables allows the evaluation of our prototype. Finally the interface proposed provides us a new approach in the problem, using the Cloud like a Service, reducing barriers of access to the Semantic Web for people without technical knowledge of Semantic Web technologies.

Keywords: artificial intelligence; semantic web; natural language; Google cloud speech; SPARQL.

---

1. Introduction

The rise of the World Wide Web (WWW) in the last years, has increased the difficulty of finding relevant information about specific study subjects (mainly, because of the ambiguity problem with the terms used in the searching tools). The Semantic Web, usually called Web 3.0 [1], is an attempt to solve that problem by the creation of a data exchanging procedure, that adds a semantic (or meaning) to the existing Web. To provide the Web of a comprehensible meaning for the computers, it is necessary a way to represent the knowledge. For that reason, information gatherings known as ontologies are used.

An ontology is formally defined as a set of concepts organized in hierarchical way, establishing properties and relationships between them, as a set of rules of inferences that allows us automatic manipulation of information.

The use of ontologies results in an efficient way to retrieve information, and to do that so it is necessary the SPARQL (SPARQL Protocol and RDF Query Language, recursive acronym). The problem is that, in many cases, SPARQL is very complicated to handle for an end-user. Experience in information retrieval, demonstrates that users are better at understanding graphical query interfaces rather than simple Boolean queries [2]. There are several works about the generation of Natural Language Interfaces (NLI) to query ontologies that have received wide attention and they allow users to express arbitrarily complex information. A Natural Language Interface is a system that allows users to access information

Peer-reviewed version available at Symmetry 2018, 10, 268; doi:10.3390/sym10070268

stored in some repository by formulating the request in natural language (e.g., English, German, French, etc.) [3]. However, the implementation of NLI involves various problems due to linguistic ambiguities and the development of accurate NLI is highly complicated. In [4] mentioned that NLI help users avoid the burden of learning any logic-based language, offering end users a familiar and intuitive way of query formulation. In [5],[6] mentioned that controlled natural language (CNL) has received much attention due to its ability to reduce ambiguity in natural language. CNL are mainly characterized by two essential properties [7] : 1) their grammar is more than that of the general language, and 2) their vocabulary only contains a fraction of the words that are permissible in the general language.

We believe that adding voice queries to the CNL, we will improve the queries of the end users. This Speech Interface will be the main contribution of our article. For the authors, an easy way to retrieve information, is querying by voice to a website that contains an ontology of a particular domain. This work focus on building a speech interface based on Google Cloud Platform to access a Semantic Website. With this interface, the barriers of accessing to the Semantic Web for people without technical knowledge will be reduced.

Finally, in our work to achieve voice queries, it will be necessary to use the cloud computing because provides resources and services over a network (usually the internet). A feature of cloud computing, is the ubiquitous network access that allows us access from any device such as: an iphone, cell phones and laptops. As results, we present three variables that indicate the performance of our prototype to users. To achieve the results, two programs have implemented, one in Python language for transforming from text into speech and vice versa and another in Java language for the implementation of natural language. Many different queries to the interface were made, for each query several data were taken and different average time responses were obtained.

In the next Section, review related work in the theme. The third Section we have mentioned the problems found in the subject from previous works. The fourth Section presents implement architecture with a Speech interface that can make queries and obtain response by voice using cloud computing services. The fifth Section depicts experimental results that we have obtained with the built architecture. Finally, in Section 6, the conclusions and future work are examined and presented.

2. Related Work

The state of the art is based on papers related to Platforms and Semantic Portals, Natural Language, SPARQL queries, speech interfaces for web and Cloud computing.

In [2] Bernstein, Kaufmann & Kaiser state that users have problems even in the simplest Boolean expressions. To face this issue, it is introduced Ginseng, a quasi-Natural Language interface to query the Semantic Web. Ginseng is based on a simple question and its structure is dynamically extended through an Ontology structure in order to guide users in their query elaboration.

Regardless of Wang, Xiong, Zhou & Yu, presenting a Natural Language interface system called PANTO, a Portable nAtural laNguage inTerface to Ontologies [8]. Which accepts generic natural language queries and outputs SPARQL queries. Based on a special consideration on nominal phrases, it adopts a triple-based data model to interpret the parse trees output by an off-the-shelf parser.

Independently in [9] is presented a semantic based search for model human speech corpora, stressing the search for meanings rather than words. The framework developed embraces the complete recognition and retrieval cycle, from word spotting to semantic annotation, query processing, and search for result presentation. Kaufmann & Bernstein state that the need of making accessible the content to the final users,it is high priority, as more information is stored in knowledge databases and Natural Language interfaces are needed, so they will provide a familiar and convenient environment to data access of the Semantic Web [3].

Then, in [4] is presented OWLPath, a Natural Language-Query editor guided by multilanguage OWL-formatted ontologies. This application allows non expert users easily to create SPARQL queries that can be issued over most existing ontology storage systems. The authors presented a global

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 May 2018 doi:10.20944/preprints201805.0274.v1

Peer-reviewed version available at Symmetry 2018, 10, 268; doi:10.3390/sym10070268

architecture system that is composed of five components and the empirical results were aplicated in two domains: one in the e-finance and the another in e-tourist.
Independently Damljanovic, Agatonovic & Cunningham state with large datasets such as Linked Open Data available, there is a need for more user-friendly interfaces which will bring advantages of these data closer to the casual users [10]. The authors presented a system named FREyA, which combines syntactic parsing with the knowledge encoded in ontologies in order to reduce the customization effort.
In [11] is stated that in many cases making a SPARQL triplet query does not mean a true representation of a query semantic structure in Natural Language. To avoid this problem, the authors propose a novel approach to questions answering over Resource Description Framework (RDF) data that relies on a parse of the question to produce a SPARQL template that directly mirrors the internal structure of the question and that, in a second step, is instantiated by mapping the occurring natural language expressions to the domain vocabulary. Chang, Hung, Wang & Lin mention that Automatic speech recognition (ASR) is a technology which converts the phrases of words spoken by human into text [12]. As a mature technology, ASR has become an alternative input method on many mobile devices, complementing other input methods operated by hands.
In [13] mentioned that cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. Then, [14] mentioned that the cloud computing emerges as a new computing paradigm which aims to provide reliable, customized and QoS guaranteed dynamic computing environments for end-users.
Independently Bukhari & Kim, presents an Ontological Model called HOIEV (Heavyweight Ontology Based Information Extraction for Visually Impaired User) is developed which architecture provides a mechanism to extract accurate information from the ontology which also designs and uses a Speech Command System designed for visual impaired people. The prototype system design not only integrates and communicates among different tools, such as voice command parsers, domain ontology extractors and short message engines, but also, introduces an autonomous mechanism of information extraction (IE) using an ontology [15].
In [16] is presented DEANNA, a framework for Natural Language question answering over structured knowledge bases. Given a Natural Language question, DEANNA translates questions into a structured SPARQL query that can be evaluated over knowledge bases such as Yago, Dbpedia, Freebase, or other linked data sources. DEANNA also analyzes questions and maps verbal phrases to relations and noun phrases to either individual entities or semantic classes.
Independently Pradel, Haemmerlè & Hernandez provide a method to the users for making queries to ontology based knowledge databases that are using Natural Language through query patterns. The approach that present in the paper differs from existing ones in the way that they propose to guide the interpretation processes by using predefined query patterns which represent these query families. The use of patterns avoids exploring the ontology to link the semantic entities identified from the keywords since potential query shapes are already expressed in the patterns [17].
Then, Androutsopoulos, Lampouras & Galanis present a detailed description of NaturalOWL [18], an open source natural language generation (NLG) system that produces English and Greek texts describing individuals or classes of owl ontologies. Unlike simplex verbalizers, which typically express a single axiom at a time in controlled, often not entirely fluent English primarily for the benefit of domain experts, natural owl aims to generate fluent and coherent multi-sentence texts for end-users in more than one languages. The authors concluded that NaturalOwL produces significantly better texts compared to a simpler verbalizer.
Bansal & Chawla developed a system that provides user friendly interface that accepts queries in natural language and extracts data from a ontology in a domain specific to retrieve the desire results [19]. The authors proposed an IRSCSD system (Information retrieval system for computer science

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 May 2018 doi:10.20944/preprints201805.0274.v1

Peer-reviewed version available at Symmetry 2018, 10, 268; doi:10.3390/sym10070268

domain). This system offers advanced querying and browsing of structured data with search for results automatically aggregated and rendered directly in a consistent user interface, thus, reducing the manual effort of users. In the methodology proposed, the authors using a QUEPY framework developed in Python language that is used to transform natural language questions into queries in RDF query language SPARQL.
Finally in [20] is presented a Question Answering (QA) system which combines multiple knowledge bases, with a Natural Language parser to transform questions into SPARQL queries or another query language. The authors demonstrated the feasibility to build such as semantic QA system, the accuracy and relevance of the returned results.

3. Current Problems

From the papers reviewed, the following problems were found.

- The Semantic Web presents a dynamic growing of the knowledge, based on formal logic. However, it is difficult for common users access to this because of they have problems in the construction of the simplest queries [2]
- Linked data initiative encompass structured databases in the RDF data model (Resource Description Framework) from the Semantic Web. Even for expert users, it is quite complex to explore such heterogeneous data [16].
- An increasing quantity of RDF data is issued as Linked Data, so an intuitive way to access those data becomes more important, but the most expressive queries fail to be represented nor answered [11].
- Voice recognition software have become more popular, especially on smartphones, which implies it is needed to work on the translation of Natural Language queries into formal queries [17]

From these problems we have developed our own prototype interface which can solve some of them, this interface supports voice queries, to avoid the difficulty for users without technical knowledge about Semantic Web and SPARQL, this way we try to reduce the access barrier.
Cloud computing services will be needed to make consultation voice queries. In [14] mentioned that cloud computing provide users with services to access hardware, software and data resources, thereafter an integrated computing platform as a service.

4. Implemented Architecture

The proposed architecture is a Speech Interface that can make queries and obtain response by voice, linked to a platform designed with the semantic web tools, OWL, SPARQL, ontologies, etc. At hardware level there are two parts, Raspberry Pi and Web Server. In more abstract level four blocks can be distinguish; Text-to-speech (TTS) and speech-to-text (STT) converter, Web interface, Natural Language Processing (NLP) Module, Knowledge Representation Module. In the implementation of the architecture we use the tools of cloud computing, in our case we use the layer called Platform-as-a-Service (PaaS) that provides user the ability to deploy their applications in the cloud by programming both language and software tools (For example Java, Phyton, Google App). Figure 1 shows the Proposed Architecture.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 May 2018 doi:10.20944/preprints201805.0274.v1

Peer-reviewed version available at Symmetry 2018, 10, 268; doi:10.3390/sym10070268

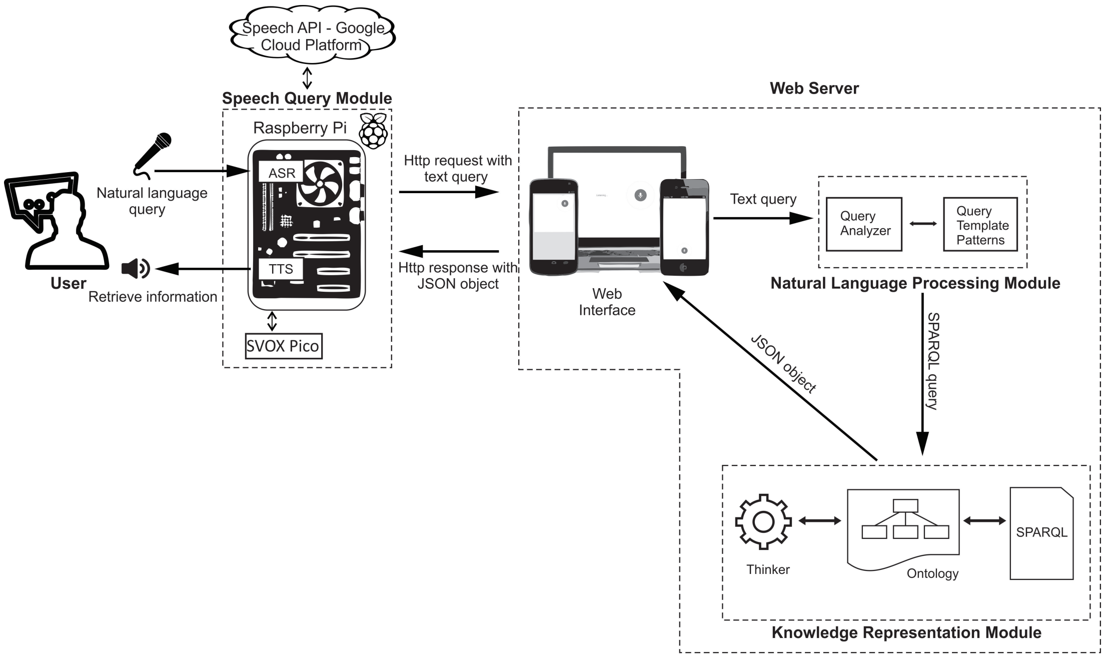

Figure 1. Proposed Architecture

4.1. Implementation of the web server using the Semantic Web tools

The Semantic Web involves many areas of computer science, including Artificial Intelligence, Web Development, Databases, Software Agents, Theoretical Computer Science, Systems Engineering, Computational Linguistics and Pattern Recognition, Document Engineering and Digital Libraries, Human-Computer interfaces, Social and Human sciences [21]. This research, appeals to some of these topics to implement the Web Server. The implementation was made with the following stages:

4.1.1. Knowledge Representation Module

The module consists of an ontology that contains all the concepts of the topic studied [22], states that an ontology encodes the knowledge of and specific domain, in a way it can be understood by a computer; where a domain is a specific area or sphere of knowledge, such as photography, medicine, education, etc. For a quick and easy information retrieve asked by the user, the information is stored inside the ontology. The ontology provides us a class and relationship vocabulary to describe the respective domain. This relationships are expressed in a hierarchical way in which the most general concepts are found in upper zones (super classes) and the most specific concepts are in the lower zones (sub classes). The ontology will be designed in OWL (Web Ontology Language) which is the latest recommendation of World Wide Web Consortium (W3C) [23], and is probably the most popular language for creating ontologies today. The W3C is one of the most active communities regarding the development of standards for the web and a fundamental pillar for the advance implementation and consolidation of the Semantic Web [24]. An increasing number of ontology are being developed and their reusing and sharing offers several benefits. One important benefit is that we can significantly save time and effort by reusing existing ontology instead of building new one every time. However, in our work we have built a new ontological model. This ontology will be based on tourism, an important topic according our national reality. In addition, for ontology modeling, Methontology recommendations will be follow, which is the standard created by the Ontological Engineering Group of the Polytechnic University of Madrid (UPM), which comprises the Specification, Conceptualization, Knowledge Acquiring, Integration, Implementation, Maintenance, Evaluation, and Documentation [25]. Currently there are editors that minimize user effort, either through contextual aids or through

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 May 2018 doi:10.20944/preprints201805.0274.v1

Peer-reviewed version available at *Symmetry* **2018**, *10*, 268; doi:10.3390/sym10070268

friendly graphical interfaces. This facilitates as much as possible the task of creating an ontological model and avoiding to the user typing raw code. In the article Protegé editor is used. Figure 2 depicts the ontology produced by the editor.

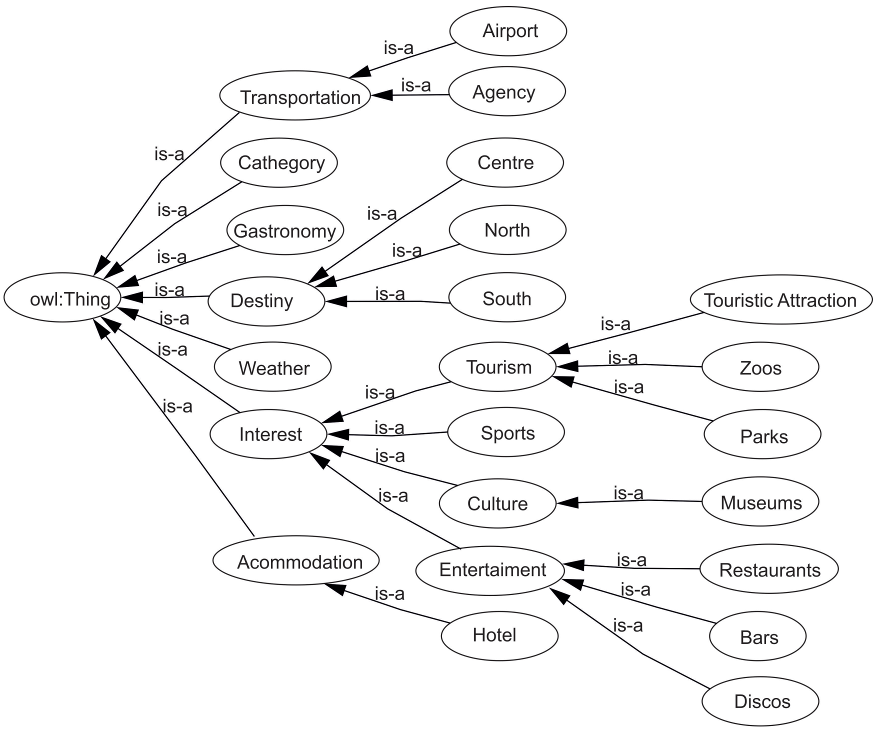

Figure 2. Ontological Model

In our work, we have chosen the tourism sector as the domain of ontology because in our country it is a sector of considerable growth. A trip involves multiple needs such as: lodging, food, transport, etc. Consequently, a tourist should consult information sources that offer different options. As observed in figure 2, our ontological model defines the main classes as Transportation, Category, Gastronomy, Destiny, Weather, Interest and Accommodation. Then, the subclasses, instances, relations and properties of our ontological model are defined, the authors check the consistency of the ontology with tools of the Protegé editor. Finally, we have obtained an OWL ontology that contains all the tourist information applied in special case in Peru that will be used by the system.

4.1.2. Natural Language Processing Module

In the literature reviewed, we observed there are studies that have implemented different natural language interface for the Semantic Web, but in these studies are not considered consultations by voice through interface.
As time goes on, according to [10], software availability of speech recognizing that understands Natural Language will become more and more popular and can be applied in mobile devices. It implies that it must be work on query transformation in Natural Language to formal semantic queries in SPARQL language. Natural Language Processing (NLP) have target to automatically process and translate the language of humans into one that machines can analyze, interpret and retrieve information effectively. In [26], mentioned that Natural Language Processing (NLP) generally requires computationaly and conceptually intensive algoritms relying large amounts of domain-dependent background knowledge, which is, to make things worse, costly to produce.
The Natural Language Processing Module uses a Java program that in essence remove accents, transform into minuscule, separate the phrase in gaps, compare word by word with the templates to

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 May 2018 doi:10.20944/preprints201805.0274.v1

Peer-reviewed version available at Symmetry 2018, 10, 268; doi:10.3390/sym10070268

choose the correct one. These processes are shows in the figure 3.

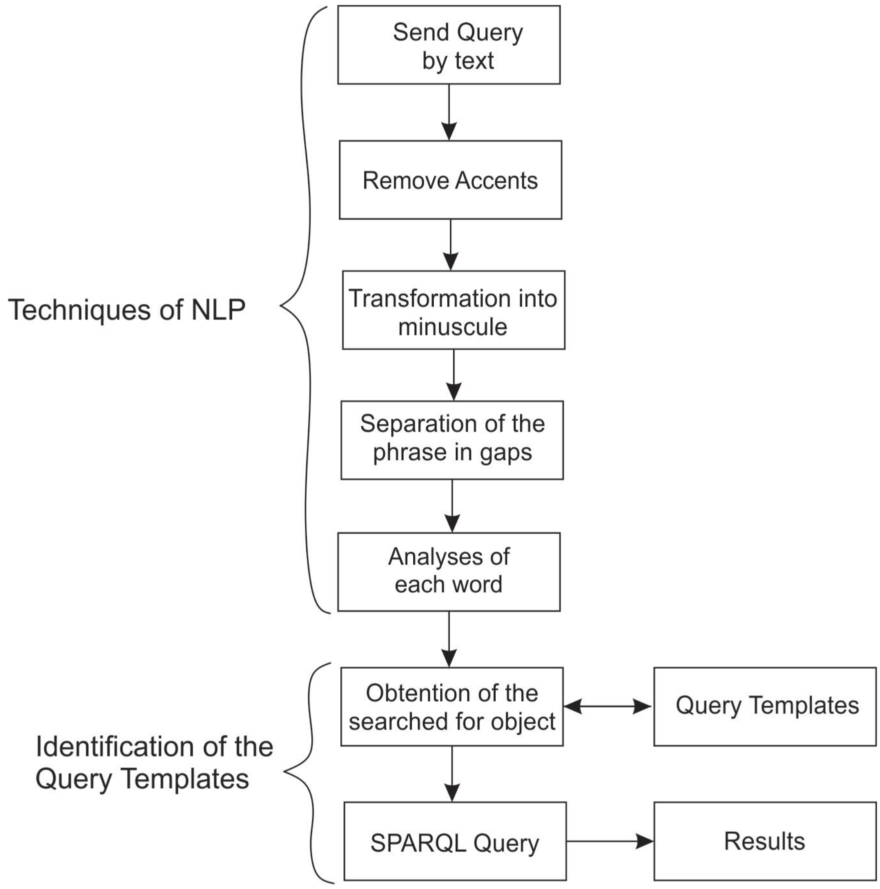

Figure 3. Steps for Natural Language Processing

An important part of the experiment is to identify the query templates use, because of there where Natural Language query is done. In many cases it is impossible for the final user to understand the complex schemes of the Semantic Web in order to express a valid query in SPARQL language. Also, the user needs to know the ontological model, to make queries. There are works around the world that are trying to transform a query from Natural Language into SPARQL query language, identifying the subject, predicate, and object in the sentences of the respective query. In our paper, a variation has been done. An object or variable of the query made by the user has been identified, from comparison criteria of the user's query with the pre-established templates, without many complications in the usage of SPARQL, by using only one variable in the query. This is because the SPARQL query language has possibility of placing a variable instead of an RDF term in the subject, predicate or objects position. There are two internal processes that are detailed as follows and shows in Figure 4.

- Comparison Criteria

In the present research, several patterns of templates have been designed to be converted into SPARQL queries, but all of them developed for the case of tourism in Peru. For example, consider a question like "find hotels in the city of Lima with contain price, name, category, address, and destiny?" A possible SPARQL formulation could consist of the following patterns (joined by shared-variable bindings) ? x has name, ?x has price, ? x has category, ?x has address, ?x has destiny, ?x isa ?place. This complex query, which involves multiple joins, would yield good results but it would be difficult for the user. This would require familiarity with the contents of the information base, which no average user is

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 May 2018 doi:10.20944/preprints201805.0274.v1

Peer-reviewed version available at Symmetry 2018, 10, 268; doi:10.3390/sym10070268

expected to here. In the work, due to complexity mentioned all queries performed by voice have the same format and are expressed as follows: “find hotels in city, find museums in city, find discos in city, find restaurants in city, find transport agencies in city”. The table 1 shows some relationship between the find for a place and the templates developed in Natural Language.

Table 1. How to get information in the system.

Find for a place	Template in Natural Language
Hotels	Find hotels in %Place
Museums	Find museums in %Place
Discos	Find discos in %Place
Restaurants	Find restaurants in %Place
Transport Agencies	Find transport agencies in %Place

For instance, in searching for hotels in the Department of Lima, a voice query “Find hotels in Lima” is made (transformed into text string). Then it will be compared with the sentence defined in Natural Language (NL) “Find hotels in %Place", where %Place is the variable or object that the system recognizes and it will be associated with the SPARQL query template to use. The system will be comparing word by word and finally it will find something that does not recognize and this will be the variable in which a determined value must be assigned. When the system obtains the already defined object as “Lima”, it will send it to the Knowledge Representation Module where the SPARQL query is completed through the Jena Library.

- SPARQL Query

When identified the object, the query in SPARQL language is made. This query will give us the information of all the hotels in Lima, even more it will give us additional information such as: name, price, address and category. In our work, there is a relationship between the sentences to ask with the query in SPARQL language (see table 1).

In the same way, other inquiries can be made to the system and all of them perform the same operations to obtain the desired results. The following listing shows a query to find the museums in the city of Piura.

```
PREFIX table: <http://www.owl-ontologies.com/Ontology1370130534.owl#>
SELECT*
FROM <http://www.owl-ontologies.com/Ontology1370130534.
WHERE {
    ?museums table:name?name.
    ?museums table:address?adress.
    ?museums table:price?price.
    ?museums table:touristAttraction_find_destination?
    touristAttraction_find_destination
    Filter(?touristAttraction_find_destination=:piura)
}
```

Listing 1: SPARQL query

4.1.3. Web Interface

For development of the Web Interface, we use JavaServer Pages (JSP) to interact with HTML and XML. In addition, we use a free and open source Java Framework for building Semantic Web and linked data applications called Jena Apache. Jena is in API for Java language will be used as a developing tool, which permits the management of Ontologies in OWL code. According to [27]

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 May 2018 doi:10.20944/preprints201805.0274.v1

Peer-reviewed version available at *Symmetry* **2018**, *10*, 268; doi:10.3390/sym10070268

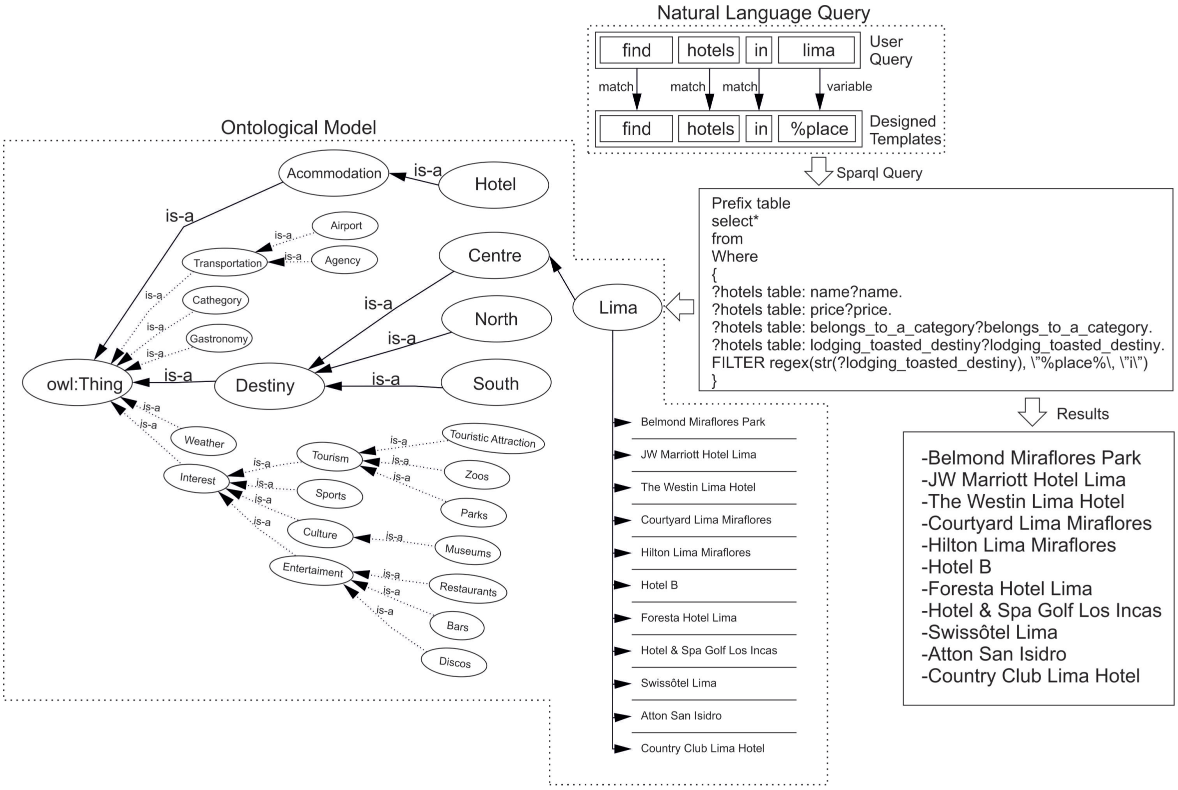

Figure 4. Internal Process

Jena it is necessary for semantic web applications, such as: read, analyze, write and create, navigate and search through an RDF (Resource Definition Framework) chart, query in the RDF database using OWL, ontologies. Furthermore, the platform will be based on the Model – View – Controller scheme, which implement servlets to manage user queries. Such queries can be verified in an Ontological editor using SPARQL [1] recommend, SPARQL that will be used to consult RDF or OWL documents. The Web Interface interact with Raspberry Pi although HTTP requests. Each HTTP request carries a query in text, which is sent to SPARQL generator. It also receives JSON objects from the Knowledge Representation Module. Figure 5 shows a SPARQL query using Jena.

4.2. Implementation of the Speech – Query Module

We use Raspberry Pi 2 in the implementation of the module. Raspberry Pi 2 is a single-board computer, despite its diminutive size, low price and unglamorous appearance, it is a fully functional computer (Partner, 2014). The Raspberry Pi 2 employs a Broadcom BCM2836 SoC with a 900 MHz 32-bit quad-core ARM Cortex-A7 processor, with 256 KB shared L2 cache. We use a USB Audio Sound Card Adapter to create a microphone-in and audio-out jack from the USB port. Raspberry Pi 2 performs text-to-speech conversion, speech-to-text conversion and HTTP requests to the Web Server. To make some transformations of this module we will use cloud computing techniques. In [28] mentioned that thanks to the use of inexpensive, commodity hardware and open source implementations, experimenting with these techniques is easy, even with very low budgets. Adapting cloud computing techniques, however requires a special kind of expertise.

4.2.1. Speech to text conversion (STT)

The conversion process used is the Automatic Speech Recognizing (ASR), which requires appropriate hardware architecture. In [29] it can be seen that exist a considerable number ASR

 doi:10.20944/preprints201805.0274.v1

Peer-reviewed version available at *Symmetry* **2018**, *10*, 268; doi:10.3390/sym10070268

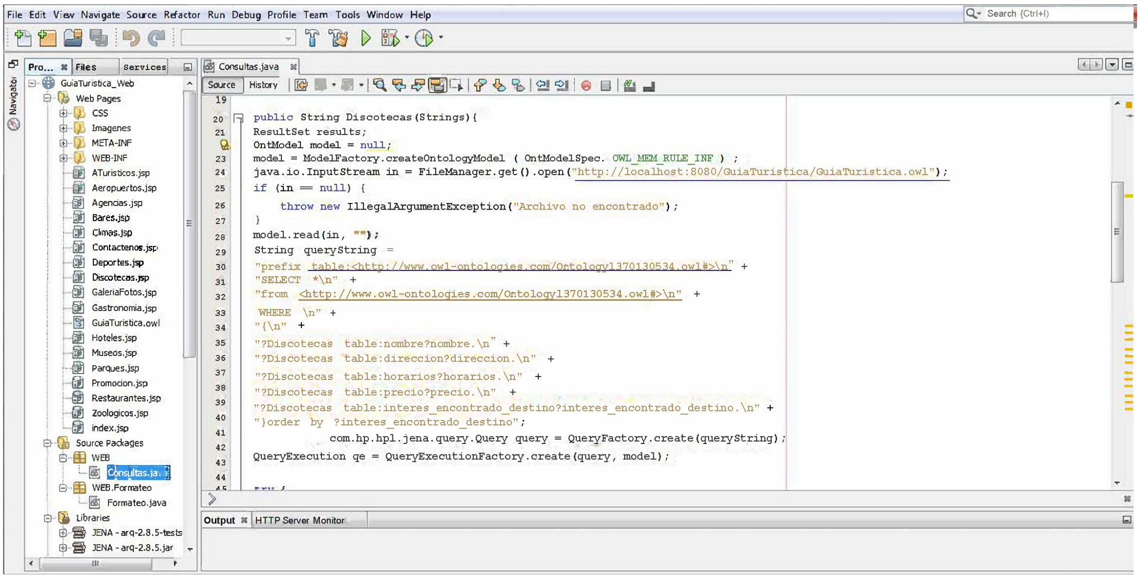

Figure 5. Query SPARQL in Jena

systems, some open source, other private and even based on cloud services. According to [30] thanks to the processing in the cloud, the ASR can be used in devices that do not have a high performance processor and avoid the use of complex processing algorithms. In [31] was presented a system using API Google STT that operate together with reduce board computer Raspberry Pi with excellent results (within 85-90% accuracy). The advantage of using this tiny PC is its small size that "makes ideal" to be embedded anywhere without saturating the available space. On the other hand, the low cost of equipment enables this massive distribution. However, the terminal requires an internet connection. In our case, API Google STT service got our attention because is a cloud computing system and does not compromise the performance of the local computer. This service use "deep learning neural network algoritms" that provides high accuracy in speech recognition. To make the transformation, a Python language program has been designed that separates the audio according into its intensity generating an audio file, in FLAC (Free Lossless Audio Codec) format, where a voice presence is assessed to exist. Then, that file is sent to the Google Cloud Speech service using the Speech Recognition 3.6.0 library, giving a string of text as an answer containing the words of the original audio.

4.2.2. Text to Speech conversion (TTS)

For text to speech conversion (TTS), it is possible to use online and offline engines. Usually, offline engines provide us poor voice quality (and sometimes, a limited number of languages are available). On the other hand, TTS engines online can make a high quality voice synthesis, but require a higher bandwidth of the internet connection and add latency time to the system (which would be better avoided). Fortunately, we found an offline TTS engine, SVOX Pico, which offers an acceptable voice quality. SVOX Pico is a new light-weight Text-to-Speech (TTS) solution. It is designed for integration into mobile phones and other mobile devices. Complementing the SVOX Pico SDK s, the SVOX Pico SDK has a new lean API supporting rapid integration and giving full control over the TTS process. The string is sent to our server with the ontology for processing. The server returns back a JSON object like an answer with the result or results of the searching inside the ontology. Then, the JSON object is separated and read loudly by the TTS engine.

5. Experimental Results

In this section, we provide the experimental results of the performance of Speech Interface in terms of three variables such as: the success rate in queries, the response time of query and the usability

Peer-reviewed version available at Symmetry 2018, 10, 268; doi:10.3390/sym10070268

of the system. The following equipment and software was used for the experimental tests: Laptop HP Corel I5 2.66 GH, DR3 Memory, 4 GB RAM, Raspberry Pi, microphone, sound card, Operative System: Xubuntu 14.04, Web Server: Tomcat, Protegé editor, program in Python Language for transforming from text to speech and vice versa.

The success in our experiment is an operation which "gets" a correct answer to the query. In our work, the success rate in queries was checked. For the calculation of the rate, we performed various questions in the domain of tourism such as: search hotels, museums, restaurants, nightclubs, transportation agencies and cinemas in all cities of Peru. According [32],[33] , the recommended sample size (number of queries) was 30 using the parameters of an estimation of population proportion. Several tests were made and 83.3% of correct results were found. The following table shows these results.

Table 2. The sucess rate in queries

Number of queries	Number of hits	Number of failures	Percentage of hits
30	25	5	83.3

To measure the response time of the query, nine different queries to the platform were made, for each query 10 data were taken and different average time responses were obtained in seconds. The times of every stage of our implementation are shown in table 3:

Table 3. Response time of query in system

Queries	**Server Time (s)**	**Google Cloud Speech API (s)**	**Delay Time (s)**	**Total Time (s)**
Find hotels in Lima	2.380	1.660	2.475	6.515
Find hotels in Piura	2.343	1.986	2.451	6.780
Find hotels in Ica	1.974	1.796	2.107	5.877
Find museums in Lima	0.217	2.126	2.405	4.748
Find museums in Piura	0.179	1.570	2.368	4.117
Find museums in Ica	0.278	2.127	2.466	4.871
Find hotels of 4 and 5 stars in Lima	1.498	2.381	1.707	5.586
Find hotels of 4 and 5 stars in Piura	2.374	2.490	2.496	7.360
Find hotels of 4 and 5 stars in Ica	2.273	1.924	2.449	6.646

As seen in Table 3, many response times of the system have been obtained, by making a program developed in Python language that calculates the time. In each case, it is seen that answers are too small. First, time response of web server is a period of time between the Raspberry Pi send a HTTP request and web server send a HTTP response with JSON object. Second, time response of Google Cloud Speech API is the time spent in Speech-to-Text conversion. Third, time response is Delay Time of system. This delay includes the process of transformation from voice to text and possible errors before the inquiry. To avoid these errors we must performan initial small training of system. To start the trainee system we carry out the following question "Hello Samanta", the system responds “Hello, what do you want?”. From, there we can make queries in natural language, and then the system will start with the transformation into a SPARQL query. As we see in Figure 6, the system spends more time in the delay, All the times they were verified using tests statistics to analyze whether the data used come from a normal distribution. The time that is selected as an indicator in the speech interface is the total time of making the whole operation in the system and this can be used to make future comparisons with other querying interfaces.

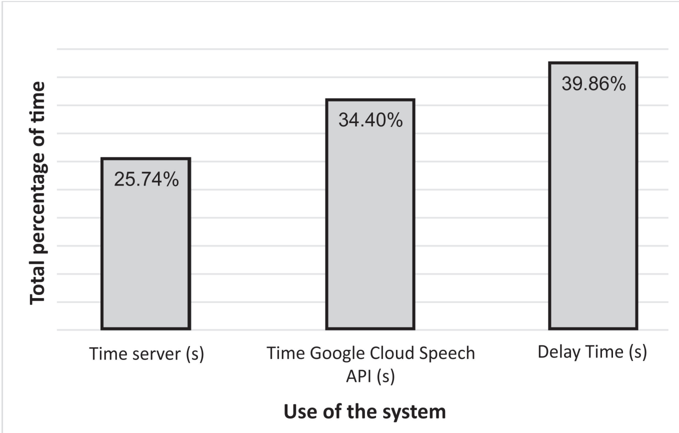

Figure 6. Percentage distribution of query according to use of the system

Assess the usability of the graphical user interface is to evaluate the bi-directional and interactive communicative process between user and system. The usability is defined as the extent to which the product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use and it is an important factor in human-computer interaction [34]. The measurement of the satisfaction of user interface, was conducted based on a small survey with statements on the Likert scale [35]. Some of the questions were:

- "The use of the interface was simple to learn"
  Agree |—| Neutral |—| Disagree
- "Interact with the interface was a frustrating experience"
  Agree |—| Neutral |—| Disagree
- "I think that the interface has all the potential I need"
  Agree |—| Neutral |—| Disagree
- "I think that this interface is very pleasant to work"
  Agree |—| Neutral |—| Disagree

The survey was applied for an average of 30 persons [32],[33]. The results were: 83.33 % agree, 10 % neutral and 6.67 % disagree quite.The following table shows these results.

Table 4. Results of the survey

Agree	Neutral	Disagree	%Agree
25	3	2	83.33

Finally these results indicate that the majority of users agree

6. Conclusions and Future Work

In the paper, we conclude that it is possible to use the tools of the Semantic Web and the use of Natural Language Processing (NLP) , build a prototype interface based on CNL where you can perform voice query. The authors believe that it is possible enrich our variety of queries designing new SPARQL templates (in our research we have 10 SPARQL templates).
In this article verified that with this focus, the gap between the Semantic Web and the real users is overcome because they can make their queries in Natural Language easily without previous knowledge of semantic technologies. Finally, for the evaluation of our designed prototype, three dependent variables were measured such as: the success rate in the query, the query response time, and a usability survey. In the verification of the success rate, quite acceptable results were obtained, with a relatively high success rate. The implemented interface allows to recover requested information

quickly. Experiments made for different templates with patterns in several queries, show good results in obtaining the total response time. After applying the usability survey, the attitude of the users was quite positive and the vast majority of them mentioned that the interface was quite friendly. With the results of the evaluation, we conclude that our interface is helpful to the user.

Currently, our designed system is not portable because is customized to be used in tourism domain. Future work includes some of the limitations that our current prototype still has. First, add new domains. Second, introduces a new framework that helps to perform the queries in Natural Language. Third, consider more variables in the SPARQL queries.

Author Contributions: Jimmy Aurelio Rosales-Huamani and Jose Luis Castillo-Sequera have contributed developing ideas about the Natural Language and Semantic Web. Juan Carlos Montalvan-Figueroa have contributed implementing Python language program. Joseps Andrade-Choque validated the results obtained. All the authors were involved in preparing the manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

1. Antoniou, G.; Van Harmelen, F. *A semantic web primer*; MIT press, 2004.
2. Bernstein, A.; Kaufmann, E.; Kaiser, C. Querying the semantic web with ginseng: A guided input natural language search engine. 15th Workshop on Information Technologies and Systems, Las Vegas, NV, 2005, pp. 112–126.
3. Kaufmann, E.; Bernstein, A. Evaluating the usability of natural language query languages and interfaces to Semantic Web knowledge bases. *Web Semantics: Science, Services and Agents on the World Wide Web* **2010**, *8*, 377–393.
4. Valencia-García, R.; García-Sánchez, F.; Castellanos-Nieves, D.; others. OWLPath: An OWL ontology-guided query editor. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* **2011**, *41*, 121–136.
5. Schwitter, R. A controlled natural language layer for the semantic web. Australasian Joint Conference on Artificial Intelligence. Springer, 2005, pp. 425–434.
6. Smart, P.R. Controlled natural languages and the semantic web **2008**.
7. Muegge, U. Controlled language: The next big thing in translation. *ClientSide News Magazine* **2007**, *7*, 21–24.
8. Wang, C.; Xiong, M.; Zhou, Q.; Yu, Y. Panto: A portable natural language interface to ontologies. European Semantic Web Conference. Springer, 2007, pp. 473–487.
9. Tejedor, J.; García, R.; Fernández, M.; López-Colino, F.J.; Perdrix, F.; Macías, J.A.; Gil, R.M.; Oliva, M.; Moya, D.; Colás, J.; others. Ontology-based retrieval of human speech. Database and Expert Systems Applications, 2007. DEXA'07. 18th International Workshop on. IEEE, 2007, pp. 485–489.
10. Damljanovic, D.; Agatonovic, M.; Cunningham, H. FREyA: An Interactive Way of Querying Linked Data Using Natural Language. ESWC Workshops. Springer, 2011, Vol. 7117, pp. 125–138.
11. Unger, C.; Bühmann, L.; Lehmann, J.; Ngonga Ngomo, A.C.; Gerber, D.; Cimiano, P. Template-based question answering over RDF data. Proceedings of the 21st international conference on World Wide Web. ACM, 2012, pp. 639–648.
12. Chang, Y.S.; Hung, S.H.; Wang, N.J.; Lin, B.S. CSR: A Cloud-assisted speech recognition service for personal mobile device. Parallel Processing (ICPP), 2011 International Conference on. IEEE, 2011, pp. 305–314.
13. Mell, P.; Grance, T. The NIST definition of cloud computing (draft). *NIST special publication* **2011**, *800*, 7.
14. Wang, L.; Von Laszewski, G.; Younge, A.; He, X.; Kunze, M.; Tao, J.; Fu, C. Cloud computing: a perspective study. *New Generation Computing* **2010**, *28*, 137–146.
15. Bukhari, A.C.; Kim, Y.G. Ontology-assisted automatic precise information extractor for visually impaired inhabitants. *Artificial Intelligence Review* **2012**, *38*, 9–24.
16. Yahya, M.; Berberich, K.; Elbassiousni, S.; Ramanath, M.; Tresp, V.; Weikum, G. DEANNA: Natural Language Questions for the Web of Data **2012**.
17. Pradel, C.; Haemmerlé, O.; Hernandez, N. Natural language query interpretation into SPARQL using patterns. Proceedings of the Fourth International Conference on Consuming Linked Data-Volume 1034. CEUR-WS. org, 2013, pp. 13–24.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 May 2018 doi:10.20944/preprints201805.0274.v1

Peer-reviewed version available at *Symmetry* **2018**, *10*, 268; doi:10.3390/sym10070268

18. Androutsopoulos, I.; Lampouras, G.; Galanis, D. Generating natural language descriptions from OWL ontologies: the NaturalOWL system. *Journal of Artificial Intelligence Research* **2013**, *48*, 671–715.
19. Bansal, R.; Chawla, S. Design and development of semantic web-based system for computer science domain-specific information retrieval. *Perspectives in Science* **2016**, *8*, 330–333.
20. El-Ansari, A.; Beni-Hssane, A.; Saadi, M. A Multiple Ontologies Based System for Answering Natural Language Questions. In *Europe and MENA Cooperation Advances in Information and Communication Technologies*; Springer, 2017; pp. 177–186.
21. Euzenat, J. Research challenges and perspectives of the Semantic Web. *IEEE Intelligent Systems* **2002**, *17*, 86–88.
22. Yu, L. *Introduction to the semantic web and semantic web services*; CRC Press, 2007.
23. Schreiber, G.; Dean, M. Owl web ontology language reference, 2004.
24. Del Castillo, J.M.M. *Hacia la biblioteca digital semántica*; Trea, 2011.
25. Fernández-López, M.; Gómez-Pérez, A.; Juristo, N. Methontology: from ontological art towards ontological engineering **1997**.
26. Badia, A. Question answering and database querying: Bridging the gap with generalized quantification. *Journal of Applied Logic* **2007**, *5*, 3–19.
27. Yu, L. *A developer's guide to the semantic Web*; Springer Science & Business Media, 2011.
28. Mika, P.; Tummarello, G. Web semantics in the clouds. *IEEE Intelligent Systems* **2008**, *23*.
29. Duarte, T.; Prikladnicki, R.; Calefato, F.; Lanubile, F. Speech recognition for voice-based machine translation. *IEEE software* **2014**, *31*, 26–31.
30. Stefanovic, M.; Četic, N.; Kovacevic, M.; Kovacevic, J.; Janković, M. Voice control system with advanced recognition. Telecommunications Forum (TELFOR), 2012 20th. IEEE, 2012, pp. 1601–1604.
31. Naeem, A.; Qadar, A.; Safdar, W. Voice Controlled Intelligent Wheelchair using Raspberry Pi. *International Journal of Technology and Research* **2014**, *2*, 65.
32. Berenson, M.; Levine, D.; Szabat, K.A.; Krehbiel, T.C. *Basic business statistics: Concepts and applications*; Pearson higher education AU, 2012.
33. Gallego, C.F. Cálculo del tamaño de la muestra. *Matronas profesión* **2004**, *5*, 5–13.
34. Standard, I. Ergonomic requirements for office work with visual display terminals (vdts)–part 11: Guidance on usability. ISO Standard 9241-11: 1998. *International Organization for Standardization* **1998**.
35. Nielsen, J. *Usability engineering*; Elsevier, 1994.

Sample Availability: Samples of the compounds ...... are available from the authors.