"Talking" Triples to Museum Chatbots

Savvas Varitimiadis[1], Konstantinos Kotis[1], Dimitris Spiliotopoulos[2(✉)], Costas Vassilakis[2], and Dionisis Margaris[3]

[1] Department of Cultural Technology and Communication, i-Lab, University of the Aegean, Mytilene, Greece
{svaritimiadis,kotis}@aegean.gr

[2] Department of Informatics and Telecommunications, University of the Peloponnese, Tripoli, Greece
{dspiliot,costas}@uop.gr

[3] Department of Informatics and Telecommunications, University of Athens, Athens, Greece
margaris@di.uoa.gr

Abstract. The paper presents recent work on the design and development of AI chatbots for museums using Knowledge Graphs (KGs). The utilization of KGs as a key technology for implementing chatbots raises not only issues related to the representation and structuring of exhibits' knowledge in suitable formalism and models, but also issues related to the translation of natural language dialogues to and from the selected technology for the formal representation and structuring of information and knowledge. Moreover, such a translation must be as transparent as possible to visitors, towards a realistic human-like question-answering process. The paper reviews and evaluates a number of recent approaches for the use of KGs in developing AI chatbots, as well as key tools that provide solutions for natural language translation and the querying of Knowledge Bases and Linked Open Data sources. This evaluation aims to provide answers to issues that are identified within the proposed MuBot approach for designing and implementing AI chatbots for museums. The paper also presents Cretan MuBot, the first experimental KG/Ontology-based AI chatbot of the MuBot Platform, which is under development in the Heracleum Archaeological Museum.

Keywords: Knowledge Graphs · RDF triples · NLP · Chatbots · Museums

1 Introduction

The paper presents recent work on the design and development of AI chatbots for museums using KGs. A KG mainly describes real world entities and their interrelations, organized in a directed graph [1, 2]. This new interactive technological trend for museums has been driven by recent observations and studies reporting that museum visitors are not impressed in the long-run by simple virtual guided tours or tours that use high-tech technological applications such as Augmented, Virtual and Mixed Realities (AR/VR/MR), but they are interested in gaining knowledge about the exhibits in a human-like, interactive and conversational manner [3, 4]. Furthermore, in comparison

M. Rauterberg (Ed.): HCII 2020, LNCS 12215, pp. 281–299, 2020.
https://doi.org/10.1007/978-3-030-50267-6_22

with Machine Learning (ML) systems, KGs and their integration to AI chatbots in every domain provides to the AI chatbots better rational dialogues that could eventually lead to more meaningful AI applications. ML focuses on prediction accuracy and scoring learning algorithms. Common sense knowledge or reasoning is out-of-scope of ML systems and for this drawback ML systems are lacking in credibility [5–7]. However, the combination of KGs and ML is an intriguing future research work.

In a museum visiting experience, visitors must be able to chat with 'smart' exhibits, ask questions in natural forms (through text or voice) and receive audible or written answers. The utilization of KGs as a key technology for implementing chatbots in this setting raises not only issues related to the representation and structuring of exhibits' knowledge in suitable formalism and models, but also issues related to the translation of natural language dialogues to and from the selected technology for the formal representation and structuring of information and knowledge [8]. Moreover, such a translation must be as transparent as possible to visitors, towards a realistic human-like question-answering process [4].

For chatbot framework/application to be effective and useful for the museum or any other domain, it must combine recent AI technological advances (Semantic Web, Linked Open Data, Knowledge Graphs, Natural Language Processing/Generation, Machine Learning) with museum needs and purposes. Natural Language Processing (NLP) and Natural Language Generation (NLG) techniques enable computers to segment, assign meaning, and analyze human communication in its natural forms and to give the users the ability to chat with the smart exhibits in a natural way [9, 10].

A museum chatbot must fulfill several specific requirements and characteristics. It must be simple, informative, accurate and precise. It must have strong conversational skills and provide meaningful content. It may be entertaining and should be able to engage the audience in the whole experience/tour duration. It may be positive if there is a capability by the chatbot to provoke users to find and learn more, but at the same time to be sensitive and understanding on human emotions [11–13].

Engineers must be able to configure the chatbot for 'talking' in human-like manner, and at the same time 'taking' in triples i.e., to retrieve and present structured knowledge utilizing Resource Description Framework (RDF) triple stores and the Linked Open Data (LOD) cloud. Furthermore, it must be available anytime, have an attractive interface and to be easy to use/interact with [11–13]. RDF and Web Ontology Language (OWL) are the two basic descriptive Semantic Web technologies that play a crucial role in the formal representation and structuring of data, information and knowledge [14].

The presented work aims to address challenges related to the following questions:

1. How can museum visitors 'talk' to museum exhibits in the most natural way, to learn about them?
2. How can human-exhibit dialogues be used for the user to retrieve the exhibit knowledge about themselves and about other related/connected exhibits, utilizing linked and open datasets?
3. What is the most appropriate technological trend to use for the most efficient conversation between visitors and 'smart' exhibits?

In relation to the above questions, the contribution of this paper is threefold:

1. To introduce the novel concept and interactive technological trend of AI chatbots for museums, as an alternative to the high-tech technological AR/VR/MR applications, towards supporting the learning of knowledge about their exhibits in a human-like, interactive and conversational manner,
2. To review the approaches and necessary tools for KG/Ontology-based AI chatbots and Query/Answering (QA) systems, as an alternative candidate to the ones based on ML techniques (e.g. Dialogflow),
3. To introduce a novel KG-based framework for chatbot-human interaction in a smart museum environment and the MuBot approach to the design, implementation and evaluation of such a framework.

The rest of the paper is structured as follows: Sect. 2 presents the related work concerning the development and use of AI chatbots and QA systems based on KGs and/or ontologies. Also, this section provides a critical description of key tools for NLP/NLG and KGs/Ontologies transformation/translation. Section 3 presents the MuBot approach, the main purpose, the special features, and the architectural design. In Sect. 4, the Cretan MuBot is presented, the first experimental chatbot of the MuBot approach, as an evaluation case study in the Heracleum Archaeological Museum. Section 5 discusses future steps in the development of the MuBot approach and concludes the paper.

2 Related Work

In this section, selected related approaches to AI chatbots/QA systems that are based on KGs and/or ontologies, as well as selected research and commercial tools for NLP/NLG and KGs/Ontologies transformation/translation, are presented. The aim of our work is to evaluate all these approaches and tools for their usability and effectiveness, in the context of the proposed MuBot approach.

2.1 Related Approaches to AI Chatbots/QA Systems Based on KGs and/or Ontologies

KGs are considered as a new AI technological trend that originates to the basic principles of the Semantic Web and the construction of Knowledge Bases (KBs). As KGs are in an evolution process, several definitions can be found [1, 2]. A definition quoted from recent literature is: "A knowledge graph (i) mainly describes real world entities and their interrelations, organized in a directed graph, (ii) defines possible classes and relations of entities in a schema, (iii) allows for potentially interrelating arbitrary entities with each other and (iv) covers various topical domains" [15].

The term KG was re-introduced as such at 2012 by Google, which has developed the famous Google's Knowledge Graph [16]. The most famous open and commercial KGs are Freebase, Wikidata, DBpedia, Yago, Google's Knowledge Graph Yahoo!'s Knowledge Graph, Microsoft's Satori and Facebook's Entities Graph [15]. KGs have many advantages and can be used for a variety of tasks such as relationship prediction

systems, search engines and question/answering agents [2]. KGs are flexible, can be easily updated and related to new data in a smart way, contain semantic information, rely on ontologies and may be queried in natural language [2, 16]. KGs may also be used in synergy with other available AI-driven technologies such as NLP, NLG, OWL and RDF datasets [2]. KGs have the ability to respond to NLG questions through database query languages. AI chatbots and QA systems can exploit this new opportunity of KGs and become 'smarter' by gaining unlimited access to stored and structured knowledge [9].

There are many proposed approaches to the recent bibliography that introduce KGs to AI chatbots and QA systems. Hallili [10] has proposed the SynchroBot, a dialog system that a) has connectivity to robust and flexible KBs and KGs for extracting information, and b) could use NLP tools to interpret user's questions and NLG techniques to provide proper answers. This ability differs from other QA systems which are either focusing in providing a logical conversation ability to the users, without caring for the richness of their knowledge source, or, on the other hand, can only provide accurate answers without any conversation skills.

OntBot was an earlier ontology-based chatbot that relied on the approach proposed by Hallili [17]. OntBot used the Protégé platform [18] in order to develop an ontology template which gathers knowledge from e-commerce website APIs. Users can place their questions to a dialog manager that uses NLP such as Facebook Wit.Ai. The dialog manager connects to the ontology template with Python language, searches the KBs and KGs and provides the proper answer to the user with the help of NLG techniques from the dialog manager [19].

Another similar approach was implemented for the healthcare domain and the goal was to produce a framework that can assist patients by providing them an AI chatbot with strong conversational skills and a robust Knowledge Base source [20, 21].

A clear statement of KGs as a part of AI chatbot is provided at the proposed approach of 'GRANK.AI.' KGs platform. GRAKN.AI is described as a KG database that uses machine reasoning to simplify data processing for AI Chatbots and other applications. Question querying is done through Graql, a knowledge-oriented graph query language for retrieving information, and for performing graph analytics and automated reasoning. The proposed approach is using DialogFlow with all its NLP, NLG, ML components, as the dialogue manager component. DialogFlow is connected to GRAKN.AI KGs and searches for the proper answer to the user's question [9]. It is a good example approach for demonstrating the combination of KGs and ML technologies.

Summarizing, the integration of KGs to AI Chatbots and Q/A systems is a task that can boost the AI chatbots effectiveness. More crucial though is the development and evaluation of NLP/NLG components that could become a reliable interpreter of natural language input or output.

2.2 Tools for NLP/NLG and KGs/Ontologies Translation

The tools that are presented and reviewed in this paper are covering key parts of the proposed approach and focus on the following questions:

1. How to recognize the entities and their relationships in a statement?
2. How to translate natural language text to a KG/ontology?

3. How to translate natural language questions to SPARQL queries?
4. How to translate RDF triples (returned from the SPARQL query) to natural language text?

Name Entity Recognition

Word Sense Disambiguation (WSD) and Name Entity Recognition (NER) for the learning of ontologies from text, are difficult tasks to accomplish and automate. Several tools were developed that can recognize the entities and the classes of given user statements in natural language.

The NLP group at Stanford University developed the Stanford Named Entity Recognition (NER), a Java-based tool used for information extraction from users' text. NER tries to find and classify atomic entities in text into predefined categories such as the names of persons, organizations, locations, expressions of times, quantities, monetary values, percentages, etc. [22]. An online demo is published at http://nlp.stanford.edu:8080/ner/. Using the example expression "Snake Goddess figurine was found at Knossos palace in Crete" as input to the tool, entity "*Location*" for the words "Knossos, Crete" and "*Organization*" for the words "Snake Goddess" respectively, were recognized.

A more advanced tool for parsing plain text is Stanford CoreNLP which provides a set of grammatical analysis NLP tools developed by Stanford NLP group, such as the part-of-speech (POS) tagger, the named entity recognizer (NER), the parser, the open information extraction tool (OPENIE) and others [23, 24]. An online demo of the CoreNLP tool is found at http://corenlp.run/. The tool was evaluated with the expression "Snake Goddess figurine was found at Knossos Palace, in Crete" and successfully returned a set of outputs by every integrated subtool i.e., NER tool recognized the entities as described above, while OPENIE analyzed and recognized the relationships between them.

Another tool is Babelfy which implements a graph-based approach to Name Entity Recognition and Linking, and WSD, based on a light identification of the possible meanings presented in graphical semantic interpretations. Babelfy is demonstrated in an online tool at http://babelfy.org/. When the phrase "Snake Goddess figurine was found at Knossos palace in Crete" was used as input, the tool successfully recognized the entities and the concepts, searched for them to LOD cloud sources such as Wikipedia, Babelnet and DBpedia, and fetched the definitions of every entity organized in a graph representation [25].

Natural Language Text to KGs/Ontologies

Protégé is the most popular and widely used free software for easily building and maintaining ontologies for any domain. Users can edit domain ontologies, share their ontology to others, create a knowledge graph representation of them, connect them with external sources, design and implement SPARQL queries, export the developed ontologies in several formats such as RDF/XML, RDF/OWL and others [18]. Protégé is also a Web service for registered users at https://webprotege.stanford.edu [26]. Protégé has been used in our use case scenario to manually create the Cretan MuBot evaluation knowledge base (schema and museum data). The output is available at https://github.com/KotisK/muBotOnto-example. Protégé is providing excellent services in manually building a knowledge base but lacks automation in the creation of ontology schemas from user input statements.

FRED is a tool for automatically producing RDF/OWL ontologies and linked data from natural language sentences. It is implemented in Python and is available as Representational state transfer (REST) service [27] and as a Python library suite. FRED is using a big set of established NLP components to produce RDF/OWL ontologies and knowledge graphs. The provided results are enriched with NER and WSD techniques [28]. FRED tool is available online at http://wit.istc.cnr.it/stlab-tools/fred/demo/ and was used to produce RDF/OWL ontology and a KG for the expression "Snake Goddess figurine was found at Knossos palace in Crete". The example output is included at https://github.com/KotisK/muBotOnto-example. The FRED tool provided an RDF ontology that has some basic similarities with our custom-made Protégé ontology, as entities, classes and properties are similarly recognized to both ontologies. FRED is using RDF triples that are only derived from DBpedia KB, while our Protégé ontology uses custom-made properties and entities. FRED tool is been used in several semantic web applications and will be considered as a component to the proposed MuBot approach.

Natural Language Text to SPARQL Queries

AutoSparql was developed by the Research Group Agile Knowledge Engineering and Semantic Web (AKSW) at University of Leipzig (http://aksw.org/Projects/AutoSPARQL.html). Its main goal is to provide users an easy way to place questions to a knowledge base. Specifically, AutoSparql can convert a natural language question to a SPARQL query, which can then retrieve the answer from a given RDF triple store. The software uses the Query Tree Learner (QTL) algorithm which is described by its creators as a light-weight learning algorithm [29]. The QTL algorithm can use various NLP techniques for creating sophisticated semantic representations of the given questions. In addition, the system and the algorithm can be trained and become smarter in providing the proper SPARQL queries. Furthermore, the user can connect with open RDF resources such as DBpedia and others [29–31]. AutoSparql software is not active now but its code is available at https://github.com/AskNowQA/AutoSPARQL. During evaluation, runtime errors were reported at compile time. Future work includes fixing and testing the tool in order to be used as candidate component of the proposed MuBot approach.

FREyA software is an interactive natural language interface that is used for querying ontologies. FREyA uses syntactic parsing in combination with the ontology-based lookup in order to interpret the question of the user. FREyA is also able to use Linked Open Data such as DBpedia. In addition, the user gets further involved and trains the system as his choices are used in order to improve its performance over time [32, 33]. This strong involvement of the user in the training of the system is consider as a drawback, as the naivety of the user about certain complex issues of data modelling, without any assistance, may not lead to the best results [30]. FREyA is not fully active now but its code is available at https://github.com/danicadamljanovic/freya. The code is outdated (Sesame java framework has evolved to Eclipse RDF4J framework which requires further installation efforts, which are left for future work).

QUEPY is a python framework that can transform natural language questions to SPARQL queries. The transformation from natural language to SPARQL queries is done by using at first a special form of regular expressions and then using a convenient way to express semantic relations. The input question is parsed using a library called

REfO (Regular Expressions for Objects). The rest of the transformation is handled automatically with the use of the Natural Language ToolKit (NLTK), a python platform that is included in the QUEPY framework and other techniques, to finally produce RDF triples and SPARQL queries [34, 35]. The QUEPY demo is not fully developed and supported. The code and the documentation of the tool is found at https://quepy.readthedocs.io/en/latest/. The software needs significant effort in order to be installed and work properly.

In our future work we aim to reuse and proceed to any modifications to the abovementioned tools in order to make them functionable components of our approach.

SPARQL Query and RDF Triples to Natural Language

SPARQL2NL is a tool that was developed by the Agile Knowledge Engineering and Semantic Web (AKSW) research group at University of Leipzig (http://aksw.org/Projects/SPARQL2NL.html). The SPARQL2NL tool allows the verbalization of SPARQL queries by converting them into natural language. The tool uses LOD sources such as DBpedia and performs several improvements to the given results of the queries that help users to choose the suitable answer in a natural way, without getting involved with ontologies and SPARQL queries' syntax [29, 36]. The SPARQL2NL project is not active now but its code is available at https://github.com/AKSW/SPARQL2NL. The installation of SPARQL2NL software encountered the same compile errors as the AutoSparql software.

The Spartiqulator system provides a similar functionality as the SPARQL2NL tool. Its main scope is to verbalize SPARQL queries in order to create natural language expressions that are readable and understandable by the common user. The Spartiqulator system refers only to RDFa and RDFS sources and uses several NLG techniques for the verbalization of SPARQL queries to suitable answers for the users [37, 38]. There is an active online demo of the tool at https://aifb-ls3-kos.aifb.kit.edu/projects/spartiqulator/. The tool has been tested with SPARQL queries from our Cretan MuBot use case scenario. For our example SPARQL query (*Where was Snake Goddess found?*), which can be found at https://github.com/KotisK/muBotOnto-example, Spartiqulator system provided the answer "*Found ats of snake goddess"*. The given answer has obvious syntax and grammatical mistakes and also lacks the *location* property which is derived from http://dbpedia.org/ontology/ that is not supported by the Spartiqulator system. The same limitations were also noticed with other example SPARQL queries that have been used for testing the tool.

SPARQLtoUser is a tool that can produce a representation of a provided SPARQL query to the end-users. The tool supports multilingual input and refers to any domain. The tool tries not only to verbalize the query, but also to find a more schematic representation of it. The user can choose the SPARQL query that provides the most accurate answer. SPARQLtoUser code can be found at https://github.com/WDAqua/ SPARQLtoUser [39]. SPARQLtoUser tool is also used as a QA webservice available at http://www.wdaqua.eu/qa. The webservice is accepting input from the users in NLG and is providing them several possible answers along with their SPARQL queries, allowing them to select the most suitable one. It also provides a probability score to each answer, asking users to affirm it or not. The QA webservice refers and gets its answers from Wikidata, DBpedia, DBLP and OpenStreetMap KDs, but cannot understand all user's

questions and answer them directly and in a sophisticated way. SPARQLtoUser tool returned "*no answer*" for our example query *"Where was Snake Goddess found at?"*. However, for the term "*Snake Goddess*" the tool returned the correspondent Wikipedia data entry.

Sparklis is a Semantic Web tool that helps users to explore SPARQL endpoints by guiding them in the interactive building of questions and answers, from simple ones to complex ones. There is an online tool at http://www.irisa.fr/LIS/ferre/sparklis/osparklis.html. Users can build a query without having any knowledge of the SPARQL language. Users can see the SPARQL query that has been created and get a proper answer [40, 41]. The tool has been evaluated with the questions from our Cretan MuBot use case. Unfortunately, due to the fact that the tool uses only profound SPARQL endpoints such as DBpedia, it could not provide proper answers. The user is not able to place a free question and must follow a guided structure [42]. The tool works well only if you shape a query such as "Give me an island in Greece?". The tool can retrieve the possible answers by providing all the islands in Greece. But in the case of questions such as "I want to find the most famous island in Greece?" the tool fails.

LD2NL is an open-source holistic NLG framework that facilitates the verbalization of the three key languages of the Semantic Web (RDF, OWL, and SPARQL) into NL. LD2NL framework builds upon the open source code of SPARQL2NL tool, which presented in a previous paragraph [43]. The LD2NL can generate either a single sentence or a summary of a given resource, rule, or query [44]. The LD2NL is an ongoing project that is not fully released as a portable application. Its code could be found at https://github.com/dice-group/LD2NL.

We aim to reuse and proceed to any modifications to the abovementioned tools in order to make them functionable components of the proposed MuBot approach.

3 The MuBot Approach

The architecture of the MuBot approach that is proposed in this paper relies on three main components:

- a Knowledge Base component that utilizes Semantic Web technology (RDF, SPARQL query language, OWL ontologies) for knowledge representation, linking, reasoning and querying,
- NLP component for interpreting users input from natural language to RDF/SPARQL,
- NLG component for creating the proper well-defined human-like answers.

The aim of the approach is to develop an AI chatbot able to conduct dialogues/conversations, such as the following:

- **User:** Where was the Snake Goddess figurine found?
- *Chatbot:* The Snake Goddess was found at Knossos Palace.
- **User:** And where is Knossos Palace located?
- *Chatbot:* Knossos Palace is in Crete.

Moreover, an additional objective of the proposed approach is to customize such a conversation in a way that it looks like discussing with the exhibit itself, in case where visitors are found to be nearby (in close range) with it. In this case, the chatbot takes over the role of the exhibit (it becomes the exhibit). For instance, the following dialogue is presented in a museum visitor standing close to the Snake Goddess exhibit, where the chatbot now has taken over the role of the Snake Goddess exhibit:

– **User:** Where were *you* found?
– *Snake Goddess (as chatbot):* I was found at Knossos Palace.
– **User:** And do you know where Knossos Palace is located?
– *Snake Goddess (as chatbot):* I do. Knossos Palace is located in Crete.

Providing a personality trait to an AI chatbot, contributes towards a more engaging experience for the visitors as first person communication is always more effective [45]. In future work authors will evaluate and reuse tools that can provide personality traits to the MuBot approach.

3.1 KG-Based Chatbot-Human Interaction in a Smart Museum Environment

In the following paragraphs, the interaction (questioning/answering) between a chatbot of a smart museum and its visitors that want to interactively learn about its exhibits, is presented.

For demonstration purposes, Fig. 1 depicts the interaction of two visitors (human entities) and the chatbot (software entity) about two exhibits. This choice has been made in order to point out the capacity of the proposed approach to setup the chatbot in a way that can lookup/search for the requested knowledge in a distributed manner, i.e., among the network of interconnected/interrelated exhibits. Such a network is built offline by the curators that are familiar with the 'stories' that each exhibit can tell the visitors. This interconnection/interrelation is formally put in a knowledge index structure that the chatbot has access for lookup/search during the QA processing. So, if a question cannot be answered by the chatbot, due to lack of knowledge that a smart exhibit may have (i.e., its KG is missing the related RDF triples), the chatbot may lookup/search this index for relative knowledge that resides in the KGs of another smart exhibit or in the LOD cloud.

The following entities 'live' in the example smart museum environment:

- Human (H): the visitor in the museum,
- Chatbot (CH): the chatbot software agent,
- Exhibit (Exh): the actual artefact in a museum e.g., a painting, a figurine, etc.,
- Smart Exhibit (SExh): the virtual exhibit in the smart environment of a museum, a virtual entity represented in a conceptual model and encoded in a formal language such as OWL.

In terms of data, information and knowledge, the following are considered to be the required in such a setting: a) Knowledge Graph (KG), b) Natural Language Question (NL-Q), c) Natural Language Answer (NL-A),d) Formal Language Question (FL-Q),

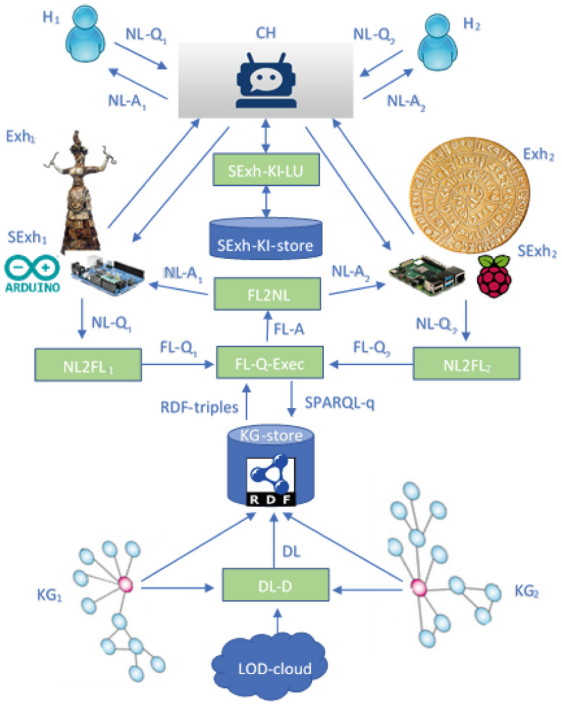

Fig. 1. Visitors interaction with Smart exhibits through the CH application

e) Formal Language Answer (FL-A), f) SPARQL query (SPARQL-q), g) RDF triples (RDF-triples), h) Data Links (DL).

In terms of storage, the setting involves: a) Knowledge Graphs in RDF triples store (KG-store), b) a "Smart Exhibits" Knowledge Index store (SExh-KI-store), and c) the Linked Open Data cloud (LOD-cloud).

Finally, the processing of data, information and knowledge requires the following components: a) Natural to formal language conversion (NL2FL), b) Formal to natural language conversion (FL2NL), c) Smart Exhibits' Knowledge Index Lookup (SExh-KI-LU), d) Formal Language Question Execution (FL-Q-Exec), e) Data Links Discovery (DL-D).

In Fig. 1, an example representation of how two museum visitors interact through the chatbot dialog manager with two "smart" exhibits is depicted. The process is as follows:

1. The visitors, as they use the chatbot application and reach the exhibit of their interest (Exh1, Exh2), interact with the CH by setting a NL-Q.
2. The CH searches at the SExh-KI-store, with the use of SExh-KI-LU, and finds the stored basic information about the exhibits.
3. The CH, with its NL2FL component, translates the NL-Q (1, 2) of the visitors to a formal syntax and sends them to the FL-Q-Exec. The NL2FL component is using NLP techniques as the NER.

4. At the FL-Q-Exec, FL-Q (1,2) are transformed to SPARQL-q which are looked-up to KG-store for retrieving the right answer.
5. KGs, that are designed for the description of the SExhs, are stored at the KG-store (as RDF triples). If an answer could not be found to the exact matched stored SExhs KGs at the KG-store, an additional search could be conducted to other SExh KGs of the museum KG-store or with the use of the DL-D component to connected LOD cloud sources.
6. When the matched RDF triples are retrieved, the FL-Q-Exec sends the FL-A to the FL2NL component.
7. At the FL2NL component, NLG techniques are used for the transformation of the FL-As to NL-As for the SExHs.
8. Finally, the visitors receive the NL-As through the CH app and could proceed to the next NL-Q.

3.2 The MuBot General Architecture

The proposed MuBot approach aims to provide museums the opportunity to create simple, interactive and human-friendly chatbots for their visitors. Visitors will be able to use a chatbot application that will be created through the MuBot platform, to chat with a 'smart' exhibit when they are in front of (or close to) it. They will be able to ask questions through text or voice (in natural language) and receive audible or written answers. The basic components of the proposed general MuBot architecture are presented in Fig. 2 and are described in brief at the following paragraphs.

Dialogue Manager
The dialogue manager is the first component of the proposed architecture and it consists of the front-end graphical user interface that the museum visitors use in order to have a conversation with the chatbot or the exhibits, in contrast to the back-end infrastructure. The dialogue manager could be implemented either as a custom-made solution or reuse a commercial (or free) existing chatbot interface that follows all the UX and UI standards for creating chatbots.

NLP
The NLP component is a crucial part of the architecture as it takes the natural language questions (NL-Qs) and transform them to machine-readable formal questions (FL-Qs). Related work [10, 46] describes three aspects of NLP identification:

– Expected Answer Type (EAT), which in the proposed architecture is handled by the Content Type Recognizer (CTR), represents the answer to the question,
– the property that relates the question with the possible answer, and
– the recognized NE (Named Entity) which in the proposed architecture is handled by the Content Entity Recognizer (CER) and represents the subject of the question made.

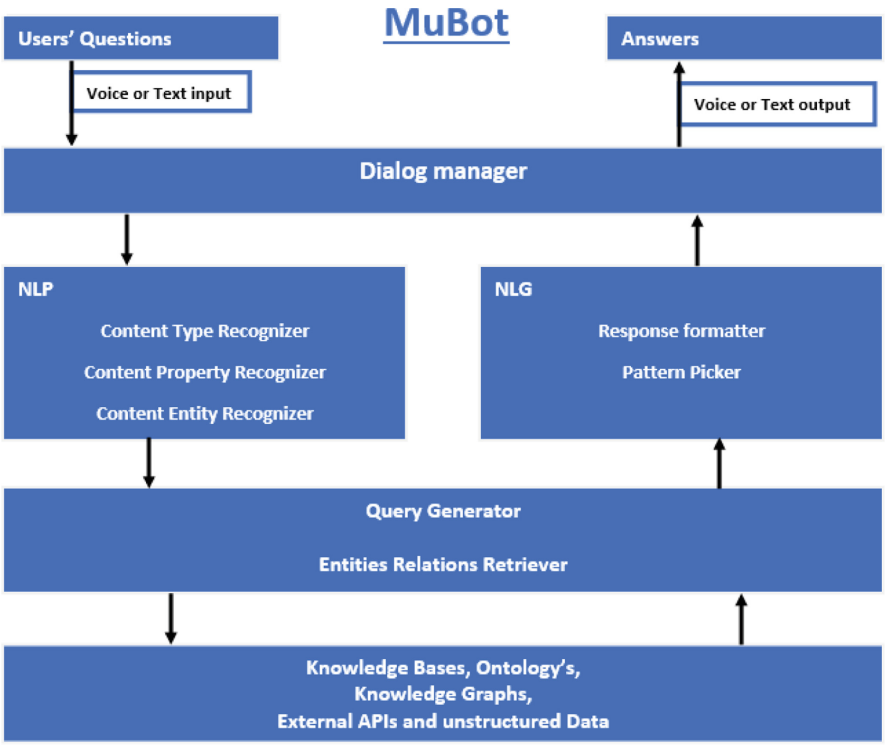

Fig. 2. The MuBot general architecture.

Query Generator and Knowledge Base Retriever
This component transforms the FL-Qs to SPARQL-queries with the FL-Q-Exeq component and searches the KGs Store or the LOD cloud for the proper knowledge. The FL-Q-Exeq receives the provided knowledge in RDF triples and transform it to FL-As.

Knowledge Base (KB)
The KB store is the storage of the KGs. If the search to the KB does not return any results, the DL-D tool will search for possible answers from external Data Links and the LOD cloud. When the knowledge is retrieved, the right answer returns to the FL-Q-Exeq as RDF triples. The connection/linkage of the MuBot architecture with external KGs, such as DBpedia, and the reasoning engine (inferencing mechanism) provides/adds a range of additional (inferred) RDF triples that multiplies the ability of CH in understanding users' questions.

NLG
This component of the proposed architecture uses NLG techniques in order to pick a response pattern that matches with the query-provided triples and proceeds to the final answer formatting by providing user the proper human-like NL-As.

4 Evaluation

The proposed MuBot architecture is under deployment and evaluation with example cases in the Archaeological Museum of Heraklion. Specifically, the Cretan MuBot use case scenario has the visitors chatting with a famous exhibit of the museum, the "Snake Goddess" figurine. For the specific scenario an example ontology was manually engineered and stored to the knowledge base. An example model (in .owl and .ttl serializations) and related queries in SPARQL can be accessed at https://github.com/KotisK/muBotOnto-example. In the following paragraphs, the Cretan MuBot use case scenario and the related queries that have been engineered for this purpose are demonstrated.

4.1 Knowledge Base

The KB uses an example MuBot ontology engineered with the use of the Protégé tool for the specific scenario, with IRI: http://i-lab.aegean.gr/ontologies/mubotOnto/ and prefix "mbo". It uses ontologies that are derived from external sources such as the DBpedia ontology, with IRI: http://dbpedia.org/ontology/ and prefix "dbo" (Fig. 3).

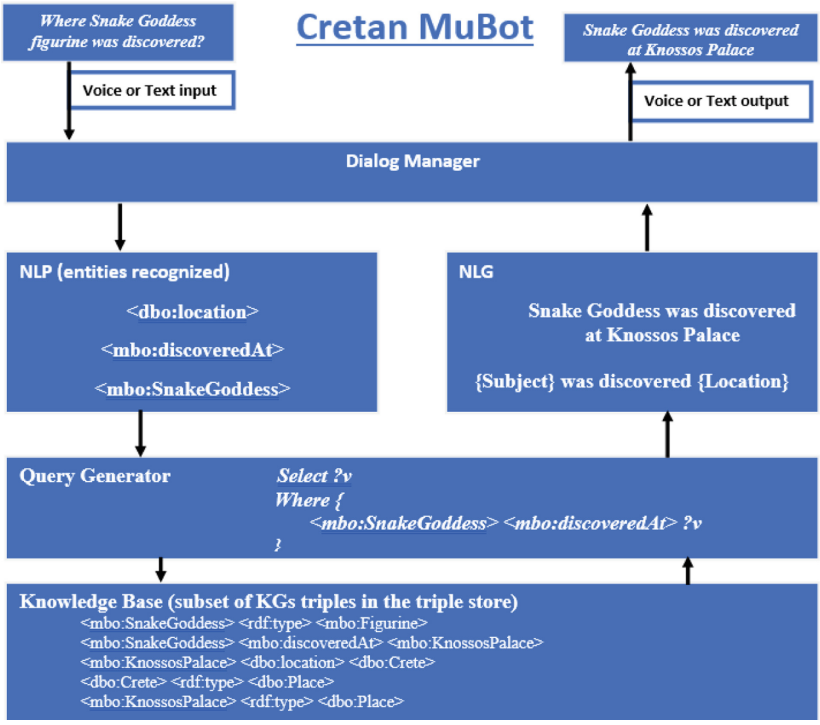

Fig. 3. The Cretan MuBot KG-oriented instantiation.

The MuBot experimental ontological model data is described in the triples form <subject> , <predicate> , <object> . In our example, the statement "*Snake Goddess figurine was discovered at Knossos Palace"* is expressed with the following RDF triples:

a) *<mbo:SnakeGoddess> <rdf:type> <mbo:Figurine>*
b) *<mbo:SnakeGoddess> <mbo:discoveredAt> <mbo:KnossosPalace>*

By extending and enriching the model with external DBpedia semantic data and the MuBot ontology, the following triples can be added:

c) *<mbo:Figurine> <mbo:discoveredAt> <dbo:Place>*
d) *<mbo:KnossosPalace> <dbo:location> <dbo:Crete>*
e) *<dbo:Crete> <rdf:type> <dbo:Place>*
f) *<mbo:KnossosPalace> <rdf:type> <dbo:Place>*

The classes that are produced for the example instantiation are: *mbo:Figurine* and *dbo:Place*, and the properties are: *mbo:discoveredAt*, *rdf:type*, and *dbo:location.* Furthermore, synonyms could be added to the classes and properties, allowing the system to be able to understand different questions with the same meaning.

4.2 Question Understanding with NLP

The users of the MuBot platform can place their questions through textual or vocal input methods. Voice to text transformation tools can be used for vocal input (such tools are out of the scope of the presented work). The provided text input can be interpreted using several NLP techniques and tools, as described in the Related Work section. In the presented example the question made is "*Where Snake Goddess figurine was discovered?"*. The CTR is *mbo:location*, the property is *mbo:discoveredAt* and the CER is *mbo:SnakeGoddess* and *mbo:Figurine.*

When all the natural language input is processed, the component is generating a formal language query (in our case in SPARQL) that produces results (answers) from the KB. The example formal query follows:

```
PREFIX mbo: <http:/i-lab.aegean.gr/ontologies/mubotOnto/>

PREFIX dbo: <http://dbpedia.org/ontology/>

SELECT? v? z

WHERE {

mbo: SnakeGoddess mbo:discoveredAt ?v.

?v dbo:location ?z.
}
```

After query execution and pattern matching, the following triples are matched and returned:

a) <mbo:SnakeGoddess> <mbo:discoveredAt> <mbo:KnossosPalace>
b) <mbo:KnossosPalace> <dbo:location> <dbo:Crete>

In a more elaborated example engineered to demonstrate the inferencing/reasoning capabilities of the proposed approach, the following statement is defined (in terms of properties and property hierarchy):

c) *<mbo:discoveredAt> <rdfs:subPropertyOf> <mbo:foundAt>*

The restriction of *mbo:foundAt* object property has as domain the class *mbo:Figurine*, and as range the class *dbo:Place*. Inferencing will add the following inferred triple in the model:

d) *<mbo:SnakeGoddess> <mbo:foundAt> <mbo:KnossosPalace>*

This allows for querying the inferred model (tested in Snap SPARQL plugin of Protégé 5.5) with the following queries also:

I. "*Where Snake Goddess figurine was found?*":

```
PREFIX mbo: <http:/i-lab.aegean.gr/ontologies/mubotOnto/>

PREFIX dbo: <http://dbpedia.org/ontology/>

SELECT ?v ?z

WHERE {

        mbo:SnakeGoddess mbo:foundAt ?v.

        ?v dbo:location ?z.
}
```

The returned data in variables are: ?v = mbo: KnossosPalace, z? = mbo:Crete.

II. "*What was found in Crete?*":

```
PREFIX mbo: <http:/i-lab.aegean.gr/ontologies/mubotOnto/>

PREFIX dbo: <http://dbpedia.org/ontology/>

SELECT ?s ?v
```

```
WHERE {

        ?s mbo:foundAt ?v.

        ?v dbo:location mbo:Crete.

}
```

The returned data in variables are: ?s = mbo:SnakeGoddess, v? = mbo: KnossosPalace, allowing for NLG module (described in next section) to provide answers such as "*Snake Goddess was found in Knossos Palace, in Crete"*.

The example Cretan Mubot model (in .owl and .ttl serializations) and the related queries in SPARQL can be accessed from https://github.com/KotisK/muBotOnto-example.

4.3 NLG Answers

The last step of the process in the MuBot architecture is the creation/formation of the appropriate answer to the given question(s).

In the presented example, the NLG component will be able to generate the following answers:

a) "*Snake Goddess was found at Knossos Palace*"
b) "*Knossos Palace is located in Crete*"

Or better, by synthesizing the two statements above:

c) "*Snake Goddess was found at Knossos Palace, in Crete*" (replacing 'is located in' with, 'in').

The generation processes could be more creative if it can exploit additional semantic properties that, for example, could return multimedia files like a video of the figurine, as a response.

5 Conclusion

The design and development of an AI museum chatbot with the use of KGs is a big challenge as there are several distinct tasks to be considered in order to provide credible answers to questions in a human-like manner. The provided review of key tools and KGs-based AI chatbot approaches aims to provide knowledge related to the proposed architectural components that must be considered for the implementation and deployment of the MuBot approach. AI chatbots must be designed in a way that museum visitors will be facilitated to interact with exhibits the way they could possibly interact with a museum curator or a guide. Moreover, museum chatbots should have strong

conversational skills in order to engage users in their cultural experience. This can be achieved by seamlessly accessing and consuming knowledge through KGs stores and the LOD cloud.

The deployment of the Cretan MuBot use case provides the opportunity to evaluate the proposed approach and highlight the steps to future work. Such work is focusing in further evaluating NLP/NLG related modules i.e., evaluation of key tools that transform natural language (visitor questions) to SPARQL queries and the returned triples (knowledge) back to natural language (answers to visitor), in the most effective and human-centered manner. In addition, further work should be conducted in defining the needs and the requirements of museum visitors and museum curators by the proposed AI museum chatbot approach. This might be considered as the most intriguing part of our future work as MuBot approach must not only provide credible answers in a human-like manner but should also develop personality traits and skills that will engage the users and assist them in the understanding of the presented museum/cultural knowledge.

References

1. Yan, J., Wang, C., Cheng, W., Gao, M., Zhou, A.: A retrospective of knowledge graphs. (2018). https://doi.org/10.1007/s11704-016-5228-9
2. Bonatti, P.A., Decker, S., Polleres, A., Presutti, V.: Knowledge graphs: new directions for knowledge representation on the semantic web. Rep. Dagstuhl Semin. **8**, 29–111 (2019). https://doi.org/10.4230/DagRep.8.9.29
3. Roussou, M., Perry, S., Katifori, A., Vassos, S., Tzouganatou, A., McKinney, S.: Transformation through provocation? 1–13 (2019). https://doi.org/10.1145/3290605.3300857
4. Schaffer, S., Gustke, O., Oldemeier, J., Reithinger, N.: Towards chatbots in the museum. In: CEUR Workshop Proceedings, vol. 2176, pp. 1–7 (2018)
5. Tresp, V., Ma, Y., Baier, S.: Machine learning with knowledge graphs. In: Proceedings of the Twelfth International Workshop on Neural-Symbolic Learning Reasoning, vol. 2017, pp. 1–48 (2014)
6. Nickel, M., Murphy, K., Tresp, V., Gabrilovich, E.: A review of relational machine learning for knowledge graphs. Proc. IEEE **104**, 11–33 (2016). https://doi.org/10.1109/JPROC.2015.2483592
7. Pommellet, T., Lécué, F.: Feeding machine learning with knowledge graphs for explainable object detection. In: CEUR Workshop Proceedings, vol. 2456, pp. 277–280 (2019)
8. Androutsopoulos, I., Spiliotopoulos, D., Stamatakis, K., Dimitromanolaki, A., Karkaletsis, V., Spyropoulos, C.D.: Symbolic authoring for multilingual natural language generation. In: Vlahavas, I.P., Spyropoulos, C.D. (eds.) SETN 2002. LNCS (LNAI), vol. 2308, pp. 131–142. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-46014-4_13
9. Orth, A.: Building chatbots with dialogflow and GRAKN.AI. https://blog.grakn.ai/chatbots-and-grakn-ai-67563c64cfde
10. Hallili, A.: Toward an Ontology-Based Chatbot Endowed with Natural Language Processing and Generation Amine Hallili, vol. 7271 (2014). To cite this version: HAL Id: hal-01089102
11. Akma, N., Hafiz, M., Zainal, A., Fairuz, M., Adnan, Z.: Review of chatbots design techniques. Int. J. Comput. Appl. **181**, 7–10 (2018). https://doi.org/10.5120/ijca2018917606
12. Radziwill, N.M., Benton, M.C.: Evaluating quality of chatbots and intelligent conversational agents (2017)
13. Shawar, B.A., Atwell, E.: ALICE chatbot: trials and outputs. Comput. Sist. **19**, 625–632 (2015). https://doi.org/10.13053/CyS-19-4-2326

14. Cahn, B.J.: Chatbot literature review. Thesis (2017)
15. Paulheim, H.: Knowledge graph refinement: a survey of approaches and evaluation methods. Semant. Web **8**, 489–508 (2017). https://doi.org/10.3233/SW-160218
16. Stichbury, J.: WTF is a Knowledge Graph? (2017)
17. Al-Zubaide, H., Issa, A.A.: OntBot: ontology based ChatBot. In: 2011 4th International Symposium on Innovation in Information and Communication Technology, ISIICT 2011 (2011). https://doi.org/10.1109/ISIICT.2011.6149594
18. Musen, M.A.: The protégé project. AI Matters. **1**, 4–12 (2015). https://doi.org/10.1145/2757001.2757003
19. Vegesna, A., Jain, P., Porwal, D.: Ontology based chatbot (for E-commerce website). Int. J. Comput. Appl. **179**, 51–55 (2018). https://doi.org/10.5120/ijca2018916215
20. Kamateri, E., Meditskos, G., Symeonidis, S., Vrochidis, S.: Knowledge-based intelligence and strategy learning for personalised virtual assistance in the healthcare domain. In: 13th International Conference on Advanced Semantic Processing (2019)
21. Meditskos, G., et al.: Towards an ontology-driven adaptive dialogue framework. In: MARMI 2016 - Proceedings of the 2016 ACM 1st International Workshop Multimedia Analysis and Retrival for Multimodal Interaction, Co-located with ICMR 2016, pp. 15–20 (2016). https://doi.org/10.1145/2927006.2927009
22. Krishnan, V., Ganapathy, V.: Named entity recognition (2005)
23. Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S., McClosky, D.: The stanford CoreNLP natural language processing toolkit, pp. 55–60 (2015). https://doi.org/10.3115/v1/p14-5010
24. Angeli, G., Premkumar, M.J., Manning, C.D.: Leveraging linguistic structure for open domain information extraction. In: ACL-IJCNLP 2015 - 53rd Annual Meeting Association for Computational Linguistic and 7th International Joint Conference on National Language Processing. Proceedings Conference on Asian Federation National Language Processing, vol. 1, pp. 344–354 (2015). https://doi.org/10.3115/v1/p15-1034
25. Moro, A., Raganato, A., Navigli, R., Informatica, D., Elena, V.R.: Entity linking meets word sense disambiguation: a unified approach, vol. 2, pp. 231–244 (2014)
26. Horridge, M., Gonçalves, R.S., Nyulas, C.I., Tudorache, T., Musen, M.A.: WebProtégé: a cloud-based ontology editor. In: Web Conference 2019 - Companion World Wide Web Conference WWW 2019, pp. 686–689 (2019). https://doi.org/10.1145/3308560.3317707
27. Neumann, A., Laranjeiro, N., Bernardino, J.: An analysis of public REST web service APIs. IEEE Trans. Serv. Comput. (2018). https://doi.org/10.1109/TSC.2018.2847344
28. Gangemi, A., Presutti, V., Reforgiato Recupero, D., Nuzzolese, A.G., Draicchio, F., Mongiovì, M.: Semantic web machine reading with FRED. Semant. Web **8**, 873–893 (2017). https://doi.org/10.3233/SW-160240
29. Lehmann, J., Bühmann, L.: AutoSPARQL: let users query your knowledge base. In: Antoniou, G., et al. (eds.) ESWC 2011. LNCS, vol. 6643, pp. 63–79. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21034-1_5
30. Unger, C., Bühmann, L., Lehmann, J., Ngomo, A.C.N., Gerber, D., Cimiano, P.: Template-based question answering over RDF data. In: WWW'12 - Proceedings of the 21st Annual Conference on World Wide Web, pp. 639–648 (2012). https://doi.org/10.1145/2187836.2187923
31. Höffner, K., et al.: TBSL question answering system demo. In: Proceedings of the 4th Conference on Knowledge Engineering Semantic Web (2013)
32. Damljanovic, D., Agatonovic, M., Cunningham, H.: Natural language interfaces to ontologies: combining syntactic analysis and ontology-based lookup through the user interaction. In: Aroyo, L., et al. (eds.) ESWC 2010. LNCS, vol. 6088, pp. 106–120. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13486-9_8

33. Damljanovic, D., Agatonovic, M., Cunningham, H.: FREyA: an interactive way of querying linked data using natural language. In: García-Castro, R., Fensel, D., Antoniou, G. (eds.) ESWC 2011. LNCS, vol. 7117, pp. 125–138. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-25953-1_11
34. Bansal, R., Chawla, S.: An approach for semantic information retrieval from ontology in computer science domain. Int. J. Eng. Adv. Technol. (IJEAT) **4**(2), 2249–8958 (2014)
35. Sæbu, T.S.: OptiqueNLQF: A natural language query formulation system based on semantic technologies (2015)
36. Ngomo, A.C.N., Bühmann, L., Unger, C., Lehmann, J., Gerber, D.: SPARQL2NL - verbalizing SPARQL queries. In: WWW 2013 Companion – Proceedings of the 22nd International Conference on World Wide Web, pp. 329–332 (2013)
37. Ell, B., Vrandečić, D., Simperl, E.: SPARTIQULATION: verbalizing SPARQL queries. In: CEUR Workshop Proceedings, vol. 913, pp. 50–60 (2012)
38. Ell, B., Harth, A., Simperl, E.: SPARQL query verbalization for explaining semantic search engine queries. In: Presutti, V., d'Amato, C., Gandon, F., d'Aquin, M., Staab, S., Tordai, A. (eds.) ESWC 2014. LNCS, vol. 8465, pp. 426–441. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-07443-6_29
39. Diefenbach, D., Dridi, Y., Singh, K., Maret, P.: SPARQLtoUser: did the question answering system understand me? In: CEUR Workshop Proceedings, vol. 1932, pp. 1–8 (2017)
40. Ferré, S.: SPARKLIS: a SPARQL endpoint explorer for expressive question answering. In: CEUR Workshop Proceedings, vol. 1272, pp. 45–48 (2014)
41. De Beaulieu, C.: What' s new in SPARKLIS, pp. 1–4 (2016)
42. Ferré, S.: Sparklis: an expressive query builder for SPARQL endpoints with guidance in natural language. Semant. Web **8**, 405–418 (2017). https://doi.org/10.3233/SW-150208
43. Ngonga Ngomo, A.-C., Bühmann, L., Unger, C., Lehmann, J., Gerber, D.: Sorry, I don't speak SPARQL, pp. 977–988 (2013). https://doi.org/10.1145/2488388.2488473
44. Ngonga Ngomo, A.-C., Moussallem, D., Bühmann, L.: A holistic natural language generation framework for the semantic web, pp. 819–828 (2019). https://doi.org/10.26615/978-954-452-056-4_095
45. Zumstein, D., Hundertmark, S.: Chatbots: an interactive technology for personalized communication and transaction. Int. J. WWW/Internet **15**, 96–109 (2018)
46. Cabrio, E., Cojan, J., Aprosio, A.P., Magnini, B., Lavelli, A., Gandon, F.: QAKiS: An open domain QA system based on relational patterns. In: CEUR Workshop Proceedings (2012)