A Personal Agents Hybrid Architecture for Question Answering featuring Social Dialog

Miguel Coronado, Carlos A. Iglesias and Alberto Mardomingo

Abstract—**There exists a growing trend in using NLIs (Natural Language Interfaces) that ranges from research to commercial products. Conversational agents beneath these interfaces have become more sophisticated, being able to either perform a task in behalf of the user or give a precise response to a question as Question Answering systems do. When combining Conversational Agents with QA capabilities the maintenance cost exponentially increases. In this paper we propose a hybrid architecture for a Question Answering system that features social dialog. We claim that including social dialog in QA systems increases users satisfaction and makes them easily engage with the system. Finally, we present an evaluation that supports these hypotheses.**

I. Introduction

In the last few years, there exists a growing trend in using Natural Language Interface (NLI) [1], not only in research but also in commercial products such as Apple Siri. Conversational agents have evolved from rather simple systems that do their best effort to maintain a conversation, to personal assistants that understand users requests and perform tasks on their behalf. Once again Siri, Google Now or Microsoft's Cortana are a few enlightening examples.

Understanding natural language usually involves using grammars and semantics, statistical methods, or simply templates [2] to identify keywords from the input phrase to comprehend what users request. Whichever the approach, researchers have to face the continual challenge of reducing the cost of maintaining these services while they are extended to embrace wider knowledge bases. Including assistance on the mix, also involves modeling the actions that the system will take. For instance, every time a new kind of consult is supported, e.g. "What is the weather like in London?" the system have to i) be adapted to understand new concepts (cities), ii) model the action that will be taken (weather request) and iii) incorporate the logic to execute that action (consult the weather forecast service). Some systems [3] were designed aiming to reduce the cost of maintainability, but these only cover the first aspect of the former list.

A second challenge NLI designers need to tackle consist in introducing social dialog into the system. Many NLI systems are indeed Question Answering (QA) systems, since they only interact with users by providing an answer to the questions they formulate. While the effectiveness of some QA systems is comparable to human's [4], they do no effort for chatting with the user, and it has been proved that including social dialog in NLI increases user's satisfaction in several domains such as eLearning [5], [6].

Having identified these two challenges, the main contribution of this paper is proposing an architecture for a conversational agent that addresses both. It features flexible conversation, easy maintainability and domain portability. These contributions are supported by the experiments described in the evaluation section. In designing the solution a conjunction of several technologies is used. In order to face maintainability we designed the system from a semantic web perspective aligned with the Information Retrieval (IR) module. Finally, to feature flexible conversations we used chatterbot technologies ruled by the agent system.

The rest of the paper is structured as follows. First, we describe the related work in section II. In section III we describe the architecture we propose, outlining the purpose of each of the modules involved. Then, we present a use case where we explain the details of the scenario we select to develop the implementation (section IV). In section V we describe the evaluation process to support our contribution. And finally, we expose the conclusions we obtained from this work.

II. Background

While Natural Language technology advances, its application to other research fields is increasing fast, being also applied to commercial applications and services. NLIs offers a set of advantages in relation to prior interfaces: intelligence, bi-directionality, interactivity, goal-driven dialog, and proactivity among others.

Developing a system with a NLI usually involves using several other technologies, as it will be shown in the in the architecture we propose in section III. Therefore, in the rest of the section we will describe the state of the art of some of the technologies involved.

A. QA Systems

Research in QA has a great career. Nevertheless, its ultimate goal has not been achieved yet. Although the basic steps of the process tend to remain the same, several different approaches were made in order to tackle the problem from different views –thus facing their main withdraws.

QA systems may be grouped by the technologies they rely on. Specially in relation to the use of Semantic Web (SW) and

Natural Language Understanding (NLU). According to Lopez et al. [7], *semantic QA takes queries expressed in Natural Language (NL) and a given ontology as input, and returns answers drawn from one or more Knowledge-Base (KB)s that subscribe to the ontology*. It also points out that these systems vary according to the degree of domain *customization* that is required as well as the subset of NL they understand

Indeed, in the state of the art we find many different examples of systems that cover the whole range. Systems include customization in order to perform some adjustements that overcome language *ambiguity* and deal with *ontology peculiarities*, e.g. different ontologies may reference a person as an instance of *Person* from the *foaf* ontology, or simply by its full-name, as a literal. Systems like Aqualog [3], PANTO [8], FREyA [9], Querix [10], NLP-Reduce [11] or QuestIO [12] do not perform any customization, i.e. they aim to be suitable for any domain. The most usual strategy among them is generating the lexicon and ontology annotations on demand, when the KB is loaded. Thus, although they are not bound to a certain field of knowledge, they need to specify the ontology that covers the domain. On the other edge, some systems rather increase their performance and recall at a cost of being valid for a specific niche of knowledge, e.g. QACID [13] or ORAKEL [14]. This customization often requires the knowledge of an expert to configure and adequate the system, sometimes in an iterative process [14]. It is also widespread to constrain the language supported, as Aqualog or ORAKEL do, in order to avoid ambiguity in the lexicon.

Concerning NLU, the complexity of the questions supported by systems vary substantially. NLP-Reduce or QuestIO implement pattern matching or bag-of-words approaches, since they lack of syntactic analysis. The advantages of this approach are that they can process ill-formed questions[1] and they are ontology-independent. Nevertheless, including grammars is the preferred choice, thus provides mechanism to deal with disambiguation or to support compositional semantics (e.g. ORAKEL). FREyA considers a trade-off, its NLU relies on the ontology lexicon as primary source, and only use syntactic analysis to address disambiguation.

To make use of the information available at the Linked Open Data (LOD) cloud, semantic QA systems need to be *portable*, i.e. they can easily adapt to new domains or ontologies of the same domain. Only PowerAqua [15] and the latest version of FREyA [16] are able to use the distributed sources of the SW and have been tested using an *open-domain* scenario.

B. IR systems

Traditionally, QA systems include an IR module. However, not only because of its complexity, but also because of its usefulness, some bibliography addresses them as separate systems ([17] classifies them as a kind on QA system). QA-IR stands as a great interface for retrieving information – i.e. integrating heterogeneous data sources under a common interface, the so called NLIs. These techniques have evolved from its early application for accessing databases in natural language language to its application to the Web [18], [19], [20], enabling the automatic construction of very large knowledge bases [21].

The notion of introducing semantics to extract information from the web was first introduced by Katz et al [22]. They proposed using an object-property-value data model that corresponds naturally to both user questions and online content, with the same principles of the SW. It addresses the challenge of providing information in a way that matches how users thinks and how the domain knowledge is structured for computer processing [23].

Within this scenario, *crawlers* are a useful tool to discover relevant documents and their relation with formerly retrieved documents [24]. Moderns crawlers not only extract structured information from the web pages they visit, but also they learn to assign visit priorities to web pages, guiding the crawling process [25]. Semantic crawlers constitute another variation of classic crawlers which select the next page to scrape using different criteria based on the *semantic distance* of the pages to the topic that is being extracted.

C. Dialog Agents and chatterbots

Chatterbots technologies have been on researchers agenda almost since the existence of computers. From the point of view of Natural Language Processing (NLP), several different approaches have been adopted during the years [26], [27], [28]. Unlike QA systems, conversational agents often work with closed knowledge bases [29]. Since they are usually tailored to a particular domain, they have a set of queries they understand, that are related to that domain. The simplest and easiest-to-implement technologies are based on this approach, and they still have great performance [11], [12]. When restricted to a domain, conversations agents usually understand more complex queries, and provide more detailed responses at a cost of harder domain portability [30].

The years of experience has shown that the social dialog capabilities, that a dialog agent introduces in NLI interfaces, may be applied to support the users on performing a particular task, to persuade them in favour of a particular option (e.g. healthier habits) or to enhance learning capabilities. Tourism industry may algo benefit from dialog agents capabilities, presenting museum exhibitions to users [31] or working as recepcionist in a virtual hotel [32]. In all these scenarios, the agent is proactive and it taking the initiative to guide the conversation towards the achievement of a certain goal. In addition, all cited papers support the idea that the users' satisfaction of using the NLI is increased when social dialog is included.

III. ARCHITECTURE

In this section we describe the overall architecture of a QA system that features social dialog in a learning environment, and has a reduced maintainability cost. The architecture constitutes the main contribution of the paper. As we hypothesized before, QA systems provide better user experience when they feature social dialog in addition to answering users' questions. Moreover, in a learning environment, having a system that traces users' learning process and supports it by taking the initiative also enhances user satisfaction. We identified the following requirements for the architecture: (i) present the users

[1] ill-formed questions are very common when users talk to a machine in a NLI

a simple interface to gather their questions (input) and show them the answers (output); (ii) classify each input according to the speech act it performs [33]; (iii) track dialog interactions answering according to the topic and previous utterances; (iv) implement a QA system being able to search the document library and extract a short answer; (v) semiautomatically extend the documents library, scrapping external documents producing semistructured indexes [34]; and (vi) follow up student learning process and use it to improve the learning experience.

The diagram shown in Figure 1 presents the global architecture of the system. Four main modules may be identified: Conversational Agent, Teaching Agent, Question Answering, and Information Extraction Agent (in addition to the speech act classifier that simply routes the query to the module that should first process it). In the rest of the section we will discuss the function of each module.

The speech act classifier receives the user queries. It provides an endpoint that the user-agent (or different user-agents) uses to post the requests and obtain responses (i.e. the system interface). The speech act classifier routes the user query to the conversational agent or the QA modules depending on the speech act detected. Nevertheless, the architecture may be extended with additional modules that handle other speech acts. This classifier is usually implemented using a supervised machine learning approach with algorithms such as decision trees or naive Bayes that provide decent results for this purpose [35], [33]. The classifier carries out a preliminar analysis of the query, since other modules will perform a deeper analysis of the input depending on the speech act they handle.

A. Conversational Agent

The *Conversational Agent* is responsible for handling the social dialog of the conversation (also referred by other authors as small talk or chit chat). It also traces the topic of the conversation that stores in the fact KB, along with the former utterances and pills of information learned or devised from the user. It is responsible for understanding the whole meaning of the user query when he/she omits information already provided in previous utterances. The Input Analyzer inside the Conversational Agent, performs a script based analysis of the input by matching regular expressions against the input. Some advanced implementations of script based analyzers also use dictionaries and perform Part of Speech (PoS) tagging and parsing.

The Conversational Agent, by means of the Answer Generator, is also in change of generating the answers that will be sent to the users. These are also stored in the dialog scripts. In the most common scenario, the small talk input is analyzed by the Input Analyzer and tells the Answer Generator to given a particular answer in response. However, the QA and the Teaching Agent modules can also instruct the Answer Generator to send a response to the user. In those cases, the Conversational Agent generates the answer according to the topic and former utterances, giving it a particular touch when needed. Besides the textual response that is presented to the user, the answer is decorated with metadata to provide further information about the state of the Conversational Agent. Typically, they indicate the mood of the agent, its facial expressions and gestures (possibly implemented using Behavior Markup Language (BML)), etc.

B. Question Answering

The *Question Answering* module takes part with those user queries where he/she asks for a particular piece of information, i.e. as with a regular QA system. The QA Analyzer uses domain-specific grammars to extract the precise meaning of the query. This is, it is classifies the type of query, extracts the relevant concepts, and categorize them according to the ontologies considered. The effectiveness of the process depends of the accuracy and degree of detail of the grammars applied, which includes the precision of the concept categorization. The QA Analyzer combines general-scope dictionaries with domain-specific ones to enhance its effectiveness. If there is no grammar that can be applied to the query, the analysis simply does not return any outcome.

As with regular QA systems, the IR is consulted to obtain the relevant documents where the answer is contained. The IR works with a set of documents that has been previously indexed. In this case, given the semantic nature of the IR system, it also acts as a SPARQL endpoint that may be queried for precise pieces of information. Thus, it supports a dual working mode depending on the nature of the query. With semantic queries the results are more precise and the information returned has better structure, the better categorization of the fields returns the more accurate results. Semantic queries also enable the use of linked data not only for enhancing results but for query expansion. When the semantic IR is not available –either because the incoming query is too general, or because there is no relevant structured documents–, the IR module will do its best to return a piece of information as accurate as possible. At least, it retrieves a set of documents that are related to the query. It will try to categorize the nature of the documents, which bring the category of the concept searched, that can be used to expand the query and reformulate the query. If no relevant document is returned by the IR, the QA is not capable to give a response, and thus the Answer Generator will inform the user.

Moreover, the Semantic IR may also be queried by external modules, e.g. the Teaching Agent. The treatment given to the query is the same as when it comes from the QA Analyzer. Finally, the conversational agent may derive a query to the QA in those cases where the Speech Act Classifier missclassifies the query, and more frequently with those utterances where the user asks for more information. In this case, the QA system needs further information to perform the document retrieval. Thus, the conversational agent will expand the query and route it to the QA.

C. Information Extractor Agent

In case there is no relevant document for the query performed, the system will try update the KB. The *Information Extractor Agent*'s main function consists on analyzing unstructured documents in order to extract fields, categorize them and generating a semistructured document. This is a slow process, so it cannot be performed in near real-time; instead, unresolved query may trigger its execution, that will be available for future queries. This automatic information

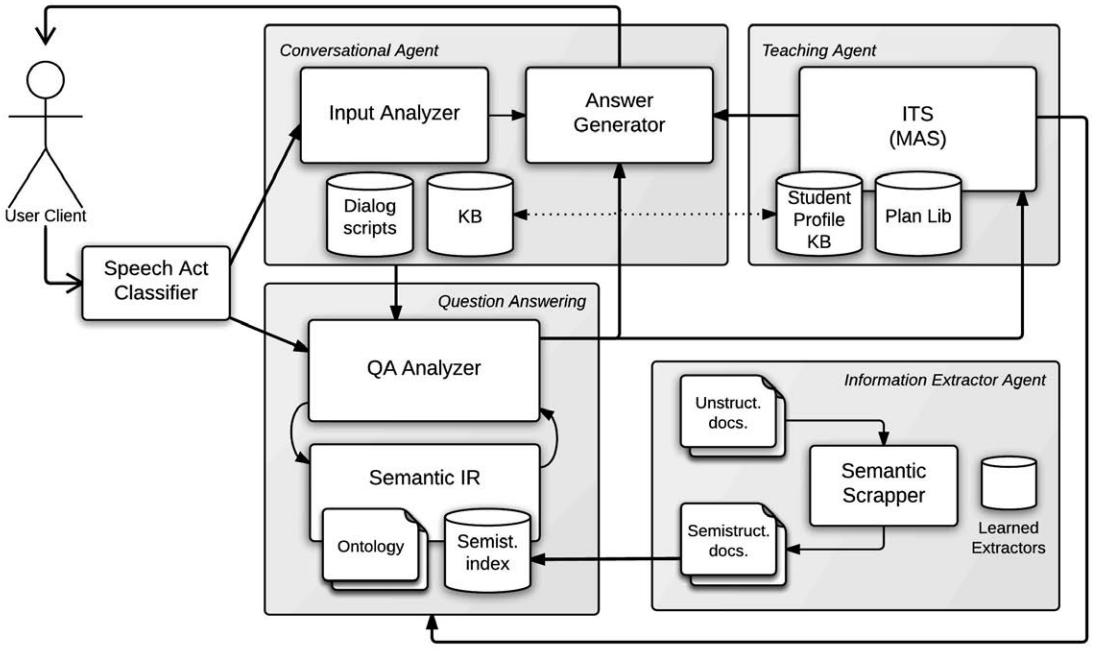

Fig. 1. Global view of the architecture proposed

extraction mechanism is a best-effort process that relies on the information on the sources, and the ontologies used to map that information. Semantic scrappers such as Scrappy [36] may boost this process. Alternatively, a system administrator may also manually include documents on the IR index, but also mark them to be processed by the Information Extractor Agent and index them afterwards.

D. Teaching Agent

The *Teaching Agent* supervises the learning process of the users, and is in charge of improving their learning experience. To do so, it gathers information about all interactions of the user with the system. This information is obtained from the KB of the Conversational Agent (or derived from it) which is synchronised with the Student Profile KB of the Intelligent Tutoring System (ITS). An ITS is usually implemented as a Multi Agent System (MAS) of Belief Desire Intention (BDI) agents. In that case, the Student Profile KB is included in the Belief Base (BB) of the agent.

The plan library rules the behavior of the agents. For instance, a plan may consist on suggesting the user a particular concept to learn, and it may be triggered when the ITS detects that the user has repeatedly asked about some concepts that are related among them and to the concept suggested. In this case, it uses information from the taxonomy of concepts, and measures the similarity of them. The information about the concept is obtained from the semantic IR, since the ITS may also consult the it to get particular information to send to the user (e.g. examples, explanations, formal definitions, etc.). As with the rest of the modules, the Answer Generator from the Conversational Agent module ultimately generates the response that is sent to the user.

IV. Case Study: Teaching a Programming Language

In this section we describe the use case scenario as well as some details of the architecture implementation we just described. The main goal of this section is showing the benefits of the architecture regarding development cost and maintainability of the corpus.

The scenario selected is in the scope of e-Learning. It represents a personal assistant that plays the role of a first-hand help for students (similar to a quick reference guide) that are learning Java programing language in Freshman year. It is able to maintain a conversation with the students. When asked about topics around the subject, it is able to find the related concept and explain it to the student. In that case, the conversation may also show more complex interactions, i.e. the assistant may take a proactive role and drive the conversation either suggesting the students to check related topics that may be on their interest, or asking them some questions related to those topic to test if they properly learnt them. Since it stores the students progress, in successive conversations, the same concepts will not be suggested as relevant. The NLI systems is developed to be used in Spanish. The user interface of the assistant is shown in Fig.2 and it is available online [2].

All the interactions just described that are related to social dialog[3] are conducted by the Conversation Agent. We used ChatScript[4] to develop the flexible dialogue capabilities of the personal assistant. We selected this framework because it provides dialog managing capabilities, it follows an NLU approach based on rules, supports functions & db access withing the rules, and features hot rule base update. We developed a

[2]http://demos.gsi.dit.upm.es/gsibot/
[3]it is also referred as chit-chat
[4]http://chatscript.sourceforge.net/

Fig. 2. Learning personal assistant user interface

corpus that gives a suitable response to lots of social dialog interaction. The response includes additional metadata such as the mood of the bot, or the url of the document from which the response was extracted.

The Question Answering system is implemented using an instance of Apache SOLR. The incoming queries are analysed using a custom parser build up using the standard elements. The fields defined in the schema were derived from the concepts modelled in the Java Learning Objects Ontology (JLOO) [37]. The indexing process were carried out using 368 documents about Java concepts in Spanish. The Speech Act Classifier is implemented by another ChatScript instance. We took this approach instead of using a machine learning implementation (e.g. a Support Vector Machine) because having ChatScript already in the system made it easier, and the precision in this usecase was of 83% [5].

The decision of initiating new interactions and suggesting related topics to the user is made by the Teaching Agent, developed using AgentSpeak [38]. Since its belief base is synchronised with the knowledge base of the Conversational Agent, it contains the information retrieved (or deduced) from the interactions with the user. In order to illustrate how the different knowledge bases may look like, listing 1 shows ChatScript's KB state at a certain time, and listing 2 shows AgentSpeak's BB state at the same time. They show how several facts from the knowledge base of ChatScript are replicated into AgentSpeak's (indicating its source). In addition, it shows that some beliefs are derived; these will trigger the plans from the agent plan library that suggest the user additional topics to check out. The agent infers which topics should be suggested extracting related concepts according to the semantic structure of the JLOO used, and filtering those the user already asked for.

```
( user212 asked_about abstract_class )
( user212 asked_about loops )
( user212 asked_about while_loop )
```

Listing 1. Chatscript knowledge base excerpt

```
( user212 asked_about abstract_class )[source(c_ag)]
( suggest user212 concept(interface) )
( suggest user212 concept(abstract_method) )
( user212 asked_about loops )[source(c_ag)]
( suggest user212 concept(while_loop) )
( suggest user212 concept(for_loop) )
( user212 asked_about while_loop )[source(c_ag)]
( user212 followed_suggestion concept(while_loop) )
( suggest user212 example(for_loop) )
```

Listing 2. AgentSpeak belief base synchronised with Chatscripts

We selected scrappy[6] [36] as a semantic scrapper. It can be configured using custom ontologies and change them depending on the documents scraped. Beside, it features a ontology discovery working mode, since when no ontology is provided, it structures the ontology according to the inline-semantics from the document, and returns the corresponding ontology along with the scrapped data.

Finally, it is important to recall that the main goal of this section is to show that the architecture is feasible, and to prove its benefits regarding development cost and maintainability of the corpus. Since utterances asking for concepts and definitions are handled by a semantic QA, indexing additional documents increases the scope at no cost. In addition, the textual search capabilities of SOLR are used as a fallback, and are fully domain independent. Thus, incorporating concepts of a different domain does not require other templates or parsers, unless we aim to support additional question types. On the other hand, the reference implementation of the corpus is not focused on boosting learning experience of the student. Thus, the implementation is not comparable to others at learning support [5], [6].

V. Evaluation

As already stated, one of the contributions of this paper is proposing an hybrid architecture for conversational agents with QA capabilities. We claim including social dialog in QA systems increases users satisfaction and makes them easily engage with the system. To test these hypotheses, we conducted an empirical study in which participants interact with three different configurations of the system and evaluate its experience through the process. To reduce subjectivity we gathered several events regarding participants interaction with the system e.g. average number of interactions.

We proposed the participants to interact with two different configurations of our prototype. They are presented a QA configuration and a Conversational Agent configuration (both with similar interfaces in order to avoid the effect of the possible confounding variable). The QA search for the information in the documents available and shows the result. Indeed, with this configuration, the QA system is working as a IR since it does not extract any information from the document. This is simple but yet valid and useful configuration since the documents we worked with were relatively small. The second configuration consist on a conversational agent, that have access to the same information as the former system (and uses the same IR module), with the added features of social dialog and proactive

[5]Computed using the logs collected during testing with humans including those from the evaluation described in the next section

[6]https://github.com/josei/scrappy

recommendation of related topic and suggesting examples. In both cases, some answers will be served with suggestions of related topics and examples to ask for. Instead of implementing four different configurations of the system, this approach was selected because the influence of the suggestions can still be measured, and otherwise the learning and fatigue effects of within-subjects evaluation with too many interfaces (that are very similar) may falsify the results.

In summary, the study is a 2x2 factorial within-subjects design. Independent variables are the use of social dialogue and enabling suggestions (only answers or answers with suggestions), while the outcome variables are the use natural language to structure the questions to the system, the use of social dialog to chat with the agent, the impact of the suggestions in subsequent interactions, the number of interactions until the user considers the task to be completed in both cases, and their satisfaction with the learning and questioning experience.

We hypothesise that participants will prefer using natural language rather than keywords search to query the system, and that they will mostly follow the suggestions given, specially in the Conversational Agent configuration. Particularly we state the following hypotheses:

- **Hypothesis 1:** Participants will interact more times with the conversational agent interface than with the QA interface.
- **Hypothesis 2a:** Participants will use natural language queries significantly more times than keyword queries to consult the system.
- **Hypothesis 2b:** In particular, participants will use natural language queries slightly more times with the conversational agent interface than with the QA interface.
- **Hypothesis 3a:** Participants will prefer to follow suggestions and ask for related topics significantly more times rather than asking for different topics.
- **Hypothesis 3b:** In particular, participants will prefer to follow suggestions when these are offered so that they have to accept or reject them.
- **Hypothesis 4:** Participants who engage in social dialogue will show significantly more satisfaction in the questionnaire.

A. Participants

Participants ranged in age from 20 to 30 years old, with most under 25 (mean=24.19, SD=3.9). 15% were female, and 69% were students. All indicated that they had experience using computers and knowledge of the domain of the study (java programming language). None had participated in any previous study involving conversational agents.

B. Procedure

The 12 participants that volunteered for this evaluation process were randomly assigned to one of two groups, varying the order in which they will evaluate the two interfaces for counterbalancing (50% of the participants will test the QA configuration first and then the Conversational Agent). Each group receives a evaluation guide and a questionnaire. Each participant is given the task consisting on questioning the system (to add extra motivation we encouraged them to try to trick it or find a relevant topic where to which the system has no answer). They are requested to perform this task with both system configuration. The evaluation consist on: first, the participants fill in a demographics questionnaire; second, they are given the general instructions to follow in the whole evaluation process; then they are given the task and the url of the system they will test first (according to the group the belong to); after that, they are requested to fill in a questionnaire of satisfaction about the first system configuration; then they are given the url of the second system to test (and the same task to perform); a second satisfaction questionnaire about the second system configuration is delivered; and finally they are requested to fill in a global satisfaction questionnaire. The average time for performing the whole evaluation process was 9.13 minutes (SD=4.26).

C. Measurements

During the process, two different measurements were taken: the questionnaires of satisfaction delivered three times during the process, and the interaction measurements taken from the logs resulting from the interaction of each participant with the system. The questionnaires and the interaction measurements were paired using unique session identifiers. After the evaluation process the logs were computed to obtain five metrics: the number of total interactions of each participant with each system configuration, the number of suggestions received by each participant with each configuration, the number of suggestions followed by each participant with each configuration, the format of each participant's query (Natural language or keywords) with each configuration, and the number of participants that interacted using social dialog with each configuration.

D. Results

1) Number of interactions: The average number interactions per user with the two different interfaces was 8.09 (QA) and 12.9 (Conversational Agent). This was 59.5% more interactions in average with the Conversational Agent interface. The mean and standard deviation of the number of interactions with both systems are shown in Fig. 3. The ANOVA we performed indicated that the difference was statistically significant ($F_{1,11}$=7.32, p=0.012 < .05). No group effect were observed regarding the counterbalancing. These results support our first hypothesis, concluding that users appreciate NL interaction and it is shown in a change of their behavior: encouraging them to keep on interacting with the system. This is also relevant in a eLearning environment, where it could push students to learning more or at least intensify their curiosity.

2) Usage of different type of queries: We split the user queries into two groups: those formed NL and those that basically formed of keywords. Although in different measure 75% of the participants used a keyword query at least once, and all of them used NL queries. The average number of NL queries per use was 6.79, 1.26 times greater than the average number of keyword-based queries. It is statistically significant ($F_{1,11}$=9.82, p=0.004 < .005) that prefer NL to consult rather than keyword-based queries. This is also verifiable when we

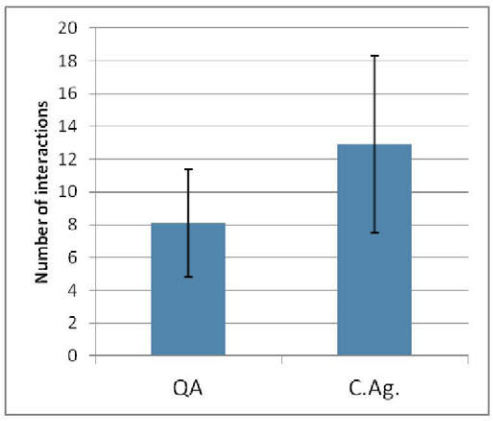

Fig. 3. Average number of interactions with both systems

split the data of both interfaces. Resulting p values are 0.011 and 0.008 for QA and conversational agent configurations respectively. This supports hypothesis 2a. However, it is not statistically significant that users use more often NL than keywords with the Conversational Agent than with the QA configuration, so hypothesis 2b cannot be validated. Figure 4 shows this distribution, and it can be appreciated that the number of interactions increases with the Conversational Agent configuration for both query types. We concluded this is because, users in general have their own preferences about use natural language or keywords (or possibly influenced by the nature of the interface used) regardless of the nature of the answers received. It is important to point out that only two out of 12 users in the experiment used more often keyword queries than NL queries Here there exists a subtle group effect, since users from Group B (which interacted first with the conversational agent) slightly reduced the usage of Keyword queries when they interacted with the QA interface.

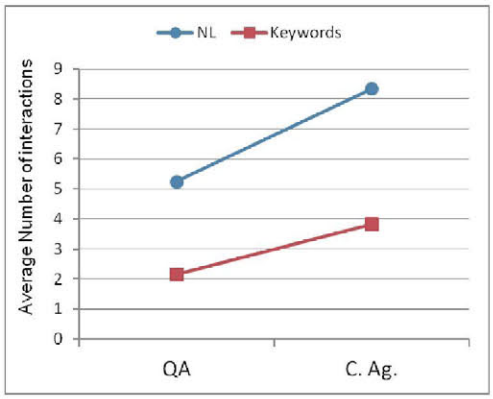

Fig. 4. Average number of interactions per user using each query type for each system configuration

3) Impact of suggestions: An average of 57% of the times a user was given a suggestion he/she followed that suggestion in the next interaction. Besides, users rated the usefulness of suggestions with 3.41 over 4. However, 96% of the suggestions followed were close-ended questions, e.g. "Do you want to check for *inheritance* too?". Thus we can conclude that users are interested in suggestions when they only need to accept or reject them (hypothesis 3b), but not that they follow suggestions of any type (hypothesis 3a). In out experiment, we decide to give suggestions in pairs (e.g. "you may be interested in searching for topic A or topic B") and it may be not the best approach to persuade the user to follow them.

4) Satisfaction: According to the results shown in Fig. 5, no correlation between interaction metrics and user's satisfaction is appreciated. Thus we cannot validate hypothesis 4.

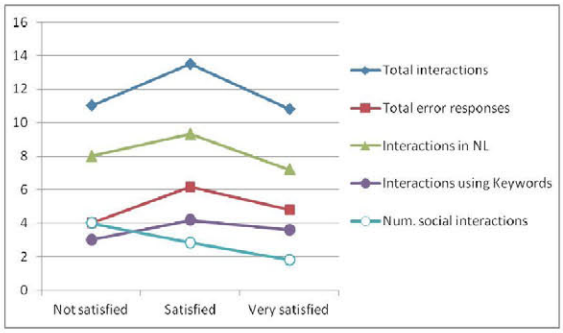

Fig. 5. User interaction metrics sorted by its satisfaction

VI. Conclusions

In this paper we identified that there were two challenges not yet approached at the same time in the state of the art: QA systems and personal assistants do not usually feature social dialog (despite it boosts users' engagement and increases its satisfaction), it may be caused to the high cost of maintaining NL systems. We reviewed several approaches of QA systems that independent of the domain but all of them lack of social dialog.

On the one hand, we proposed a personal agent architecture for QA that features social dialog and has a low maintainability cost. Since utterances asking for concepts and definitions are handled by a semantic QA, indexing additional documents increases the scope at no cost. In addition, the textual search capabilities of SOLR are used as a fallback, and are fully domain independent. Thus, incorporating concepts of a different domain does not require other templates or parsers, unless we aim to support additional question types.

On the other hand, with the prototype we developed, we could perform a evaluation to support our hypothesis about the benefits of social dialog and system proactiveness.

Throughout the evaluation process we validated three of the four hypothesis we formulated beforehand. We concluded that i) users appreciate NL interaction and it is shown in a change of its behavior, significantly increasing the number of interactions with the system when it features social dialog; ii) users prefer to query the system using NL queries than using keyword-based queries; and iii) users appreciate and take into consideration suggestions of relevant topics only if they are offered so that they only have to be accepted or rejected.

As future work, we plan to conduct a longitudinal study, gathering information during a long period to measure the long term evolution of student's behavior. To do so, the system will be improved, making it more proactive, including more flexible dialog and enhancing its document library so its knowledge base covers as many questions as possible.

ACKNOWLEDGEMENTS

This research work is supported by the Spanish Ministry of Economy and Competitiveness under the R&D project CALISTA (TEC2012-32457) and, the European Union under the Horizont 2020 project MixedEmotions (H2020 141111). The authors whish to thank Javier Herrera who took part in the development of an early stage of the architecture presented as part of his master thesis.

REFERENCES

[1] M. a. Hearst, "'Natural' search user interfaces," *Communications of the ACM*, vol. 54, no. 11, p. 60, 2011.

[2] A. R. X. R. C. Zaenen, "Language Analysis andUnderstanding," in *Survey of the state of the art in human language technology*, A. R. X. R. C. Zaenen, Ed., 1997, pp. 95–138.

[3] V. Lopez, V. Uren, E. Motta, and M. Pasin, "AquaLog: An ontology-driven question answering system for organizational semantic intranets," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 5, no. 2, pp. 72–105, 2007.

[4] M. D. Smucker, J. Allan, and B. Dachev, "Human question answering performance using an interactive document retrieval system," in *Proceedings of the 4th Information Interaction in Context Symposium on - IIIX '12*. New York, New York, USA: ACM Press, 2012, p. 35.

[5] S. Chaudhuri, R. Kumar, I. Howley, and C. P. Ros, "Engaging Collaborative Learners with Helping Agents," in *Proceeding of Artificial Intelligence in Education (AIED' 09)*, 2009.

[6] A. Kerly, P. Hall, and S. Bull, "Bringing chatbots into education: Towards natural language negotiation of open learner models," *Knowledge-Based Systems*, vol. 20, no. 2, pp. 177–185, 2007.

[7] V. Lopez, V. Uren, M. Sabou, and E. Motta, "Is Question Answering fit for the Semantic Web ?: A survey," *Semantic Web*, vol. 2, no. 2, pp. 125–155, 2011.

[8] C. Wang, M. Xiong, Q. Zhou, and Y. Yu, "PANTO : A Portable Natural Language Interface to Ontologies," *The Semantic Web: Research and Applications*, vol. 4519, pp. 473–487, 2007.

[9] D. Damljanovic, M. Agatonovic, and H. Cunningham, "Natural Language Interfaces to Ontologies : Combining Syntactic Analysis and Ontology-Based Lookup through the User Interaction," *The Semantic Web: Research and Applications*, vol. 6088, pp. 106–120, 2010.

[10] E. Kaufmann, A. Bernstein, and R. Zumstein, "Querix : A Natural Language Interface to Query Ontologies Based on Clarification Dialogs," in *5th International Semantic Web Conference (ISWC 2006)*, no. November, 2006, pp. 5–6.

[11] E. Kaufmann, A. Bernstein, and L. Fischer, "NLP-Reduce : A naïve but Domain-independent Natural Language Interface for Querying Ontologies," 2007.

[12] V. Tablan, D. Damljanovic, and K. Bontcheva, "A Natural Language Query Interface to Structured Information," *The Semantic Web: Research and Applications*, vol. 5021, pp. 361–375, 2008.

[13] O. Ferrández, R. Izquierdo, S. Ferrández, and J. L. Vicedo, "Addressing ontology-based question answering with collections of user queries," *Information Processing & Management*, vol. 45, no. 2, pp. 175–188, 2009.

[14] P. Cimiano and P. Haase, "Porting Natural Language Interfaces between Domains An Experimental User Study with the ORAKEL System ," in *Proceeding IUI '07 Proceedings of the 12th international conference on Intelligent user interfaces*, 2007, pp. 180–189.

[15] V. Lopez, V. Uren, M. Sabou, and E. Motta, "Cross Ontology Query Answering on the Semantic Web : An Initial Evaluation," in *K-CAP '09 Proceedings of the fifth international conference on Knowledge capture*, 2009, pp. 17–24.

[16] D. Damljanovic, M. Agatonovic, and H. Cunningham, "FREyA : An Interactive Way of Querying Linked Data Using Natural Language," in *The Semantic Web: ESWC 2011 Workshops*, G. García-Castro, Raúl and Fensel, Dieter and Antoniou, Ed., Galway, Ireland, 2012, pp. 125–138.

[17] W. Lu, J. Cheng, and Q. Yang, "Question Answering System Based on Web," in *2012 Fifth International Conference on Intelligent Computation Technology and Automation*. Ieee, 2012, pp. 573–576.

[18] P. Atzeni, G. Mecca, and P. Merialdo, "Semistructured and structured data in the web: Going back and forth," *SIGMOD Record*, vol. 26, no. 4, pp. 16–23, 1997.

[19] C. A. Knoblock, S. Minton, J. L. Ambite, N. Ashish, I. Muslea, A. G. Philpot, and S. Tejada, "The ariadne approach to web-based information integration," *International Journal of Cooperative Information Systems*, vol. 10, pp. 145–169, 2001.

[20] J. Hammer, J. Cho, R. Aranha, and A. Crespo, "Extracting Semistructured Information from the Web," in *Proceedings of the Workshop on Management of Semistructured Data*. Tucson: Stanford Infolab, 1997, pp. 1–8.

[21] F. Suchanek, D. Saarbruecken, and G. Weikum, "Knowledge Harvesting in the Big-Data Era," in *SIGMOD '13 Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*. ACM, 2013, pp. 933–937.

[22] B. Katz, S. Felshin, D. Yuret, A. Ibrahim, J. Lin, G. Marton, A. J. Mcfarland, and B. Temelkuran, "Omnibase : Uniform Access to Heterogeneous Data for Question Answering," in *Natural Language Processing and Information Systems*. Springer Berlin Heidelberg, 2002, pp. 230–234.

[23] B. J. Grosz, D. E. Appelt, P. A. Martin, and F. C. Pereira, "TEAM: an experiment in the design of transportable natural-language interfaces," *Artificial Intelligence*, vol. 32, no. 2, pp. 173–243, 1987.

[24] J. Cho, "Crawling the Web: Discovery and maintenance of large-scale Web data," no. November, 2001.

[25] S. Batsakis, E. G. Petrakis, and E. Milios, "Improving the performance of focused web crawlers," *Data & Knowledge Engineering*, vol. 68, no. 10, pp. 1001–1013, Oct. 2009.

[26] J. Weizenbaum, "Eliza—a computer program for the study of natural language communication between man and machine," *Commun. ACM*, vol. 9, no. 1, pp. 36–45, Jan. 1966.

[27] M. Mauldin, "Chatterbots, tiny muds, and the turing test," in *Proc. Natl Conf. AI (AAAI-94)*.

[28] A. Neves, F. Barros, and C. Hodges, "iaiml: a mechanism to treat intentionality in aiml chatterbots," in *Tools with Artificial Intelligence, 2006. ICTAI '06. 18th IEEE International Conference on*, Nov 2006, pp. 225–231.

[29] B. A. Shawar and E. Atwell, "A comparison between alice and elizabeth chatbot systems," *Raport instytutowy, University of Leeds*, 2002.

[30] A. Mollá, Giego (Macquarie University and S. Vicedo, Jose L. (University of Alicante, "Special Section on Restricted-Domain Question Answering," *Computational Linguistics*, no. October 2006, 2007.

[31] S. Kopp, L. Gesellensetter, N. C. Krämer, and I. Wachsmuth, "A conversational agent as museum guide–design and evaluation of a real-world application," in *Intelligent Virtual Agents*. Springer, 2005, pp. 329–343.

[32] M. Coronado, C. A. Iglesias, and G. Hernández, "3DTour - Mundos virtuales 3D aplicados al sector turístico," in *Actas IX Jornadas de Ingeniería Telemática (JITEL 2010)*, 2010, pp. 300–303.

[33] C. Moldovan, V. Rus, and A. C. Graesser, "Automated speech act classification for online chat," *CEUR Workshop Proceedings*, vol. 710, pp. 23–29, 2011.

[34] R. Delbru, S. Campinas, and G. Tummarello, "Searching web data: An entity retrieval and high-performance indexing model," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 10, pp. 33–58, 2012.

[35] B. Samei, H. Li, F. Keshtkar, V. Rus, and A. C. Graesser, "Context-Based Speech Act Classification in Intelligent Tutoring Systems," in *Intelligent Tutoring Systems*. Springer, 2014, pp. 236–241.

[36] J. I. Fernández-Villamor, C. A. Iglesias, and M. Garijo, "A Framework for Goal-Oriented Discovery of Resources in the RESTful Architecture," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 44, no. 6, pp. 796–803, June 2014.

[37] M.-C. Lee, D. Y. Ye, and T. I. Wang, "Java learning object ontology," in *Advanced Learning Technologies, 2005. ICALT 2005. Fifth IEEE International Conference on*. IEEE, 2005, pp. 538–542.

[38] R. H. Bordini, J. F. Hübner, and M. Wooldridge, *Programming multi-agent systems in AgentSpeak using Jason*. John Wiley & Sons, 2007, vol. 8.