A method for designing dialogue systems by using ontologies

http://www.producao.usp.br/handle/BDPI/48886

Downloaded from: Biblioteca Digital da Produção Intelectual - BDPI, Universidade de São Paulo
A Method for designing Dialogue Systems by using Ontologies

Ewerton José Wantroba  
Department of Computer Science  
Institute of Mathematics and Computer Sciences  
USP - University of São Paulo, Brazil  
Email: wantroba@icmc.usp.br

Roseli Ap. Francelin Romero  
Department of Computer Science  
Institute of Mathematics and Computer Sciences  
USP - University of São Paulo, Brazil  
Email: rafrance@icmc.usp.br

Abstract—The use of ontologies in Computer Science has as main purpose to allow sharing and reusing of knowledge. It is important that the concepts present in the ontology have in fact a formal specification. This article presents a methodology for designing Dialogue Systems, turning easier the task of building the knowledge base for a dialogue system. A method to design and aggregate existing ontologies in these systems are also being proposed. For this, pattern matching, state of art natural language processing tools, thesaurus and the language AIML are also used. The proposed approach has been applied to a receptionist system and it will be shown through the several experiments performed a satisfactory performance of the proposed dialogue system.

I. INTRODUCTION

An ontology is a knowledge structure used to represent (formally) and share domain knowledge through modeling and establishing a framework about relevant concepts and the semantic relationships among these concepts [1], [2] and [3].

Ontology structures explicitly represent domain knowledge by using a format comprehensible by a machine. They can be incorporated into computer applications and systems, facilitating the annotation data [4], the decision making process [5], information retrieval and natural language processing [6]. In addition, they serve as part of the Semantic Web [7]. Ontologies have also the potential to support the process of clinical decision making (CDS), increasing the reuse of data and knowledge systems [8].

Ontology development, according to the principles of the developing ontology, can potentially facilitate interoperability and reuse. On the other hand, designing ad hoc ontologies, without the use of development standards, have created an environment in which there are numerous ontologies with limited ability for both communicate amongst themselves and the reuse of the knowledge [1], [9] and [10].

Although there is no consensus on how to develop ontologies, several approaches have been described to better share some common elements of development. Likewise, while formal evaluation methods have the potential to maximize the benefits of ontologies within the areas of computer science, philosophy and life, there is no standard approach to assess the quality of ontologies, from the perspective of their intrinsic or extrinsic value characteristics (ie, the usefulness of a particular task) [11], [12], [13].

This paper presents an architecture for dialog system which includes a methodology for building dialogue using ontologies. In Section II is described the complete architecture for the dialog system, wherein the dialogue module which makes use of ontologies is inserted. In Section III is explained, in details, the methodology proposed. Then the inference engine consisting of two main methods making use of natural language processing tools along with the thesaurus WordNet is presented in Section IV. For validating the proposed methodology was applied for a receptionist system capable to answer questions about information relatet to Institute of Mathematics and Computer Sciences - ICMC-USP and the results are presented in Section V.

II. ARCHITECTURE

It has been proposed and developed an architecture for a dialogue system (Figure 1). The proposed architecture is independent of the domain of language and can be used in any dialogue system. The advantages of its use include the easy inclusion and updating of knowledge, represented in the AIML files format [14] in this work, but the informatio can be stored using other ways (such as a database). The inference engine mechanism, presented in Section IV), is suitable to use for receptionists systems, but can also be adapted to other dialogue systems.

The architecture is composed by four modules. The first one is the interface module for establishing the communication between the user and the desktop avatar or browser avatar, the lastest allows that people all over of world accessing the recepcionist system. The second is Web Module for providing the encoding and decoding of information received by the interface module. The third one is the voice module allowing voice and textual interaction and enables the system to recognize and synthetize the voice. It is optional, due to its complexity, but highly recommended, as the benefit of its use in relation to a system that allows only textual interaction. The fourth module is the dialogue module, that is responsible for storing the knowledge base in AIML file format [14] and running the inference engine (Section IV). It is capable for holding multiple users simultaneously connected with Avatar Valerie, without regard to conflict of information.

Based on the knowledge representation model presented in Section [15], responsible by the inferences about the information, the dialog module contains the following steps:

1) parsing: to verify if the matching of word by word occurs, using the algorithm of Matching Behavior
implemented by the parser available at Language AIML [14]. In this work, we have adopted the parser ProgramD (http://aitools.org/Programd) due to the fact that it facilitates the web communication.

2) *semantics:* instead of producing semantic trees, a XML tree is constructed through the own structure of language AIML.

3) *knowledge base and structures of the real world:* they are inserted by the developer of `<category>`’s AIML, which contains information of the language, structure of questions and answers external database.

---

**III. BUILDING THE KNOWLEDGE BASE USING ONTOLOGIES**

The most important task that a chatterbot have to perform is to be able to communicate with humans through natural language. The method of knowledge construction and organization required for this task is of paramount importance in order to the chatterbot can be acceptable. The choice of method will directly affect the system’s functionality and limitations. In this section, we will explain how the knowledge base has been created by using ontologies.

For the construction and organization of the dialogue in Portuguese referring to the Institute of Mathematics and Computer Sciences (ICMC-USP), it has been developed and used the methodology explained to follow. Structured data used are provided with a database *MySql* provided by ICMC-USP. Further, unstructured information are also used obtained through *FeedsRSS*, pages of ICMC-USP and such data first go through a process of structuring data as it follows:

1) **Build the Ontologies:**
   a) *Build the Classes* from the structured data contained in database, such as, Name, Room, E-mail,....).
   b) *Build the Generalizations "is-a"* related to the classes to their respective superclasses. (i.e. EMail is an address; Classroom is a local;). How much more generalizations are created at this stage richer domain knowledge will be. Another important fact is about the generalizations which can be changed later without to affect the construction of the dialogue afterwards. All classes and generalizations known by the receptionist developed system are shown in Figure 2.

   ![Diagram](image)

   **Fig. 2.** The classes and generalizations of the ontology.

   c) *Build the Object Properties and Data Properties,* such properties represent relationships between individuals of the ontology. Each property has:
   i) *Name:* usually a verb. (Ex. studies)
   ii) *Domain:* classes of individuals belonging to the domain (eg. Professor)
   iii) *Range:* classes of individuals belonging to the range (eg. Interest Area). If it is a Date Property then these values are considered as literal values and do not classes, eg. [Seminar] isDescribed string
   iv) *characteristics:* they can be of the following types:
      A) *Functional:* If a property is functional for a particular individual *a*, there may be up to one individual *b* that is related to *a* through that individual property;
      B) *Inverse functional:* If a property is an inverse functional, this means that the functional property is its inverse. For the individual *a* can exist at most one related to *a* through individual property individual;
      C) *Transitive:* If a P transitive property relates the individual "a" to individual "b", and also a individual "b" to individual "c", it is inferred that the individual
"a" is related to the individual "c" according to P;

D) **Symmetric**: If a P property is symmetric, and relates a individual "a" to individual "b", then the individual "b" is also related to the individual "a" through the property P.

In Figure 3 it is shown some properties created for the ontology developed in this work.

2) **Design patterns will match with user inputs and the corresponding answers in natural language.**

The AIML language’s formalism were used at this stage. A category is created for each input pattern and AIML file contains as many categories as are the properties of the ontology. Each AIML category contains:

a) **Pattern**: It is created a pattern for each Property of the ontology. An example of input pattern is presented to follow:

   - `Professor[studies][InterestArea].`  
   \[ < patterns > PROFESSOR* studies \]

   The reverse would be automatically constructed:

   - `InterestArea[isStudiedBy][Professor].`  
   \[ < patterns > InterestArea* isStudiedBy PROFESSOR \]

b) **Template**: Each template contains the answer format providing information to the user in natural language based on ontology. Example:

   \[ < template > The professor* studies those Interest Areas: \]

   \[ ontologia.Inferencia(Propriedade = studies; Individual = * ) \]

   (The inference above returns all the Interest Areas of the "Professor *")

**Note**: A mechanism automatically creates all the patterns based on the ontology properties. It will be the developer responsibility to write the template for all the patterns. Otherwise, the system will only display the data as a response.

The proposed method for establishing the dialogue to turn the system able to act as a receptionist at ICMC-USP will be presented in the next section.
IV. INFECTION ENGINE

Algorithm 1 Inference engine.

Require: Ontologies and WordNet, Models PLN (Sentences detector, tokenizer, POS Tagger) and User Input
{Uses a model of detector sentences}

detectSentences(userInput);  
{uses a tokenizer to separate words} 
	words ← tokenizer(sentence); 
{Part of Speech tagger classifies words according to their grammatical classes}

grammaticalClasses ← posTagger(words); 
while all words are not analyzed do 
	if word is a noun then 
		synonymous ← returnSynonymous(word); 
	{the WordNet returns all similar words} 
	candidateWords ← word + synonymous; 
	{checks whether the words are classes in the ontology} 
	candidateClasses ← returnClasses(candidateWords); 
	{Search in the ontology if the words are individuals of classes} 
	candidateIndividuals ← returnIndividuals(candidateWords); 
	{Classes of individuals candidates} 
	ClassesCandidateIndividuals ← returnClasses(candidateIndividuals); 
	end if 
end while 

{call the algorithm 2 for finding the correct AIML pattern based on candidate Classes, candidate Individuals, ClassesCandidateIndividuals} 

pattern ← findPattern(...); 
template ← returnTemplate(pattern); 
return template

Algorithm 2 Finding the AIML pattern that contains the response correspondent to the users input, based on Classes and Individuals already elucidated.

Require: Classes candidateClasses, Individuals candidateIndividuals, Classes ClassesCandidateIndividuals 

desambigua(candidateIndividuals); 
{disambiguation of individuals by keyword} 
{find all Super Classes of individuals candidates} 
SuperClassesCandidateIndividuals ← returnSuperClasses(ClassesCandidateIndividuals); 
{find all Sub Classes of individuals candidates} 
SubClassesCandidateIndividuals ← returnSubClasses(candidateClasses); 
{Only if there is an individual then it is the individual about whom you want to obtain information} 
if candidateIndividuals.size == 1 then 
	individual ← candidateIndividuals[0]; 
	individualClasse ← returnClasse(individual); 
{performs permutation among individual, candidate classes, SuperClassesCandidateIndividuals, SubClassesCandidateIndividuals} 
permuations ← permutation(); 

properties ← findProperties(permuations); 
{if there be any among the permutation property then there is no matching} 
if properties.size == 0 then 
	return "No Match"; 
end if 
{there is more than one property, then the user needs to choose which among them} 
if properties.size > 1 then 
	return "There are more than one Match"; 
end if 
{There is one Match} 
if properties.size == 1 then 
	property ← properties[0]; 
	classeRange ← properties.getRange(); 

AIMLpattern ← < pattern > individualClasse + 

property + classeRange < /pattern >; 
return AIMLpattern; 
end if 
{if there is more than one person then the user must resolve the ambiguity} 
if candidateIndividuals.size > 1 then 
	return "Ambiguity"; 
end if

In order a dialogue contained in AIML file can be inferred by a robot, it is necessary to use interpreters, which are tools that can be found in (http://aiml-programr.rubyforge.org), such as, Program P (http://alicebot.sweb.cz/) and Program D (http://aitools.org/Programd) among others. We have opted to use the interpreter Program D, which is able to maintain a dialogue with multiple users simultaneously and futher to facilitate its use by a Web service.

For interaction in English language, all the knowledge base provided by ALICE chatterbot was used. Its creator [14] proposed that the personality of a chatterbot, developed
using AIML, is easily transcribed by defining variables called properties. These properties are presented in a configuration file. The Avatar Valerie contains 75 properties. In Figure 4, it is shown a part of the file that defines the personality variables of Avatar.

Once added all knowledge of A.L.I.C.E. Avatar into Valerie system, the matching of patterns among the entries in English is given by Matching Behavior Algorithm contained in Program D interpreter.

The interaction in Portuguese language allows to the user to get the information regarding the ICMC-USP, based on the detailed proposal in Section III. For the matching of input patterns with knowledge base of chatterbot, Valerie, was proposed the algorithm 1 presented to follow.

The first step of Algorithm 1 consists of pre-processing the user’s input. Then is obtained and used only the first sentence, being that each word is separated and classified by tokenizer, and by Part of Speech tagger. After, every word is analyzed and the nouns are selected. These will be the candidate words to be individuals of the ontology classes. The goal at this point is to find the AIML pattern that will contain the answer to a presented question. The details are stored as AIML patterns and are presented in Section III. The method to find the pattern matching between the Classes and individuals, discovered in the phrase, is detailed in Algorithm 2 as it follows.

It is assumed that the user will always be asking a question about an individual present in the ontology. The answer is contained in a property of the concerned individual. Then, we need to find out what is the property that contains the answer.

This, in algorithm 2, is basically a permutation among the class of the individual candidate, their super classes, sub classes and other classes found in the standard input. The goal is to find a ontology property that contains the answer. It is noteworthy that the method to disambiguate individuals was presented in Section III.

V. EXPERIMENTS AND RESULTS

For validating the proposed system, a questionnaire was applied to ICMC-USP community, which speaks brazilian portuguese. It contains five scenarios of interaction with the system, in which the user should answer some questions.

A total of 15 questions were suggested to the user among the five scenarios, in which the user should ask to the system (the Avatar) to get the information. For each question, the user should answer if the system returned the information required by ticking one of three possible answers: "Yes, the system hits the first attempt"; "Yes, but the system hits after several attempts"; and "No, the system did not respond".

The system accuracy obtained was 70% having the Avatar answered correct on average 42% of the questions on the first attempt and 28% after a few tries. On average, the system was not able to answer only in 30% of the cases, which usually involved atypical spelling of proper names. This results are showed in Figure 5.

Noteworthy the questions that got the best and the worst accuracy rate. The system performed better on the questions about a Seminar entitled "Allocation Of Tasks And Communication For The Coordination Of Robots", responding to 70% of the questions on the first attempt, 20% after a few tries and 10% the system did not respond. Also, it presented 70% accuracy on the first attempt in question about the teacher called "Roseli Aparecida Francelin Romero", with 10% accuracy after some trial and not responding in 20% of the tests. In Figure 6, it is shown the accuracy of the proposed system answering questions that not include unusual proper names.

The worst performance was in question about a Seminar presented by "Fabio Ruffino", where the system failed to respond in 70% of the cases, and hitting the 30% of the tests on the first attempt, this being the only one in which the system does not respond to more than 50% of the tests. This is due to the name "Ruffino" with the two letter "f" is not usual in Brazil. In Figure 7, it is shown the system accuracy responding questions containing unusual proper names.

VI. CONCLUSION

In this work, a method was proposed for the dialog construction based on ontologies for incorporating in an avatar to turn it able to act as a receptionist system. This proposal can be easily extended to other types of systems. This methodology was applied and tested in the context of a receptionist for ICMC-USP and presented a satisfactory accuracy. The use of methodologies for building such a system, and in conjunction with the proposed pattern for matching of the input and output information allowed to the system to be able to answer questions based on an ontology. This system will be used
as a recepcionist at ICMC-USP, providing information about teachers, seminars and other items of the user interest. The testing showed that system has a high accuracy on questions in which key words (nouns) are easily recognized by the speech recognizer (e.g. the questions about Seminars and about the teacher names, and a low accuracy in questions containing unusual proper names.

Thanks to the use of ontologies, there is a possibility of building an Avatar with its own personality and reusing the knowledge of A.L.I.C.E. This is possible due to easy inclusion of properties separately from knowledge base itself. Originally the software A.L.I.C.E. only allowed interaction by text, and thus, the use of their knowledge base on any system capable of adding features as voice and facial expressions, turns it to be more attractive, intelligent and easy to use.

The availability of the dialogue system in the Web provided an easier the interaction with the user. Considering that the development of microarray chatterbots is a cyclical process in which each time the system is improved based on the previous interactions, the greater the range of individuals, more suitable the system will become.

As future work, we intend to increase the intelligibility of the system. One way to improve the dialogue is to increase the complexity of the ontology and hence adding more complex properties for the search. It is noteworthy that to do this, it is necessary an improvement in the stage of preprocessing user input, with improved techniques for Natural Language Processing. We also intend to use, develop and incorporate techniques allowing the detection of repeated questions by the user and alert him about them.

ACKNOWLEDGMENT

The authors would like to thank Carnegie Mellon University for grant AVATAR Valerie and CNPq and FAPESP for financial support, which enabled the realization of this work.

REFERENCES