ANALYSIS OF INTELLIGENT CHAT BOT WEB-BASED APPLICATION

AMARDEEP KUMAR

Research Scholar, Dept. of Computer Application,

Sri Satya Sai University of Technology & Medical Sciences, Sehore, Bhopal-Indore Road, Madhya Pradesh, India

Dr. Jitendra Seethlani

Research Guide, Dept. of Computer Application,

Sri Satya Sai University of Technology & Medical Sciences, Sehore, Bhopal Indore Road, Madhya Pradesh, India

ABSTRACT

This paper presents the design and development of an intelligent voice recognition chat bot. The paper presents a technology demonstrator to verify a proposed framework required to support such a bot (a web service). While a black box approach is used, by controlling the communication structure, to and from the web-service, the web-service allows all types of clients to communicate to the server from any platform. The service provided is accessible through a generated interface which allows for seamless XML processing; whereby the extensibility improves the lifespan of such a service. By introducing an artificial brain, the web-based bot generates customized user responses, aligned to the desired character. Questions asked to the bot, which is not understood is further processed using a third-party expert system (an online intelligent research assistant), and the response is archived, improving the artificial brain capabilities for future generation of responses.

KEYWORDS: Artificial Intelligence, Chat bot, Computers, Application, Technology

INTRODUCTION

Conventionally web-bots exist; web-bots were created as text based web-friends, an entertainer for a user [1]. Furthermore, and separately there already exists enhanced rich site summary (RSS) feeds and expert content processing systems that are accessible to web users. Text-based web-bots can be linked to function beyond an entertainer as an informer [2], if linked with, amongst others, RSS feeds and or expert systems. Such a friendly bot could, hence, also function as a trainer providing realistic and up-to-date responses. The convenience could be improved if the system is not only text based but also voice-based & voice trained. This is the problem addressed by this paper. A conversation is an assimilation of information where one creates differences and similarities during the duration of a conversation. Depending on the level of intelligence the experience would be enjoyable and a true emulation of a virtual entity. The gradient of intelligence is not the number of correct and incorrect statements but the ability to learn and add to its knowledge base. To create a more user accessible chat system; a simpler input method

using voice is introduced; creating and catering for a more personal and convenient experience. The process of an online chat system would follow a client server approach which acquires the signal and streams it to a server. The input voice is then processed and a response is generated. This process places a large processing requirement on the server's processor and memory resources. This limitation is even more evident when a large number of users are to be simultaneously accommodated on the system. Voice recognition requires a two part process of capturing and analysis of an input signal [3]. While the client utilizes the operating system for an input mechanism to acquire a signal, it is for the client to interpret the signal. This process can alleviate processing from the server and allow the server to generate responses faster than when it has more voice processing requirements. Server response generation can be broken down into two categories: data retrieval and information output. The core focus of this paper is to improve the information output by generating a response that is relevant to the request, factual and personal. This requires aspects of news and an intelligent algorithm to generate informative and user specific responses.

CHATBOTS CONSIST OF THREE MAJOR COMPONENTS:

The user interface, an interpreter and a database. Laven [1] defines Chatbot as a platform that efforts to simulate typed discussion, with the goal of at least provisionally tricking the social into thoughtful they were speaking to other person. Actually, chatbot is a conversational agent that cooperates with operators for a given topic using the natural language. Till date several chatbots have been organised on the internet for the determination of education, consumer service site, supervision, entertaining, etc. The famous existing chatbots are ALICE [2], Siri and Ok Google. The AI based chatbots are famous because they are light weight, easy to configure as well as at low cost. In our paper, we are going to have an application for college purpose which will provide all the information related to college and student queries.

Firstly the bot analyzes user triggered message to the chatbot program, then according it matches reply from the MySQL database, the answer is formulated and send back to the user. Students must select the category listed in a drop down fashion having various options such as admission, faculty details, syllabus, exams etc. Hence, this will avoid student's direct enquiry to college. If any new applicant enquirers for admission and the particulars about any section of the college this bot will assistance to get the answer of enquiry of the applicant. The chatbots that are currently been live in market uses text, voice and emotion intelligence as the input. In this paper, we have used the text as user input. If the present proposes need to be improved, we have to provide some options. For the same, we restart from the basics. There is always need to rethink about the fundamental abilities on which intelligence works.

a) Arithmetic The power to compute is the fundamental of intelligence. It contains arithmetic processes like addition, subtraction, division and so on. Today's machineries do well on this portion. They can help carry out even complex calculations within no time.

b) Comparison, Logic and Reasoning The choice of AI becomes wider when a structure has the capability to apply logic and make Assessments. Current generation PCs can accomplish logical operations nicely with the Values of Boolean algebra.

c) Education, Heuristics and Memory The main objects of AI will be a tool to remember past incidents, learn new things and gain experience. Heuristics implementation in newer software has given the ability for machines to grow, learn and gain experience.

d) Senses It helps to know the environment around us. We humans are lucky enough to have really efficient and effective set of senses. Some animals like dogs are said to have even greater abilities to sense. A working machine with correctly installed equipment's to sense the surroundings will prove to be a great body for its intelligent brain.

e) Perception The output of senses is then processed here. This leads to creativity along with intelligence. We can call a machine with a perception as a distant dream of AI.

f) Consciousness It is a most difficult content to be detailed on. Most difficult task to implement in a machine. Take for an example - How can a physical system come to notice the presence of itself in the world? This question is really very difficult to answer for. Everyone will have their own views.

SYSTEM ARCHITECTURE

The system consists of the following three components: client, server, and content acquisition. The server is a simple object access protocol (SOAP) aware internet application (web service) based on a black box approach. A black box approach isolates the client from interacting with the inner workings of the web service; as opposed to a white box approach, where the inner workings are essential and allows the client to interact with a distributed environment. As shown in Fig. 1, all messages are formatted in an extensible markup language (XML) and encapsulated as a SOAP message pack. The packs are text based allowing for a greater diversity of clients and platforms. The client contains the voice recognition processing module which allows the client to only send and receive plain text.

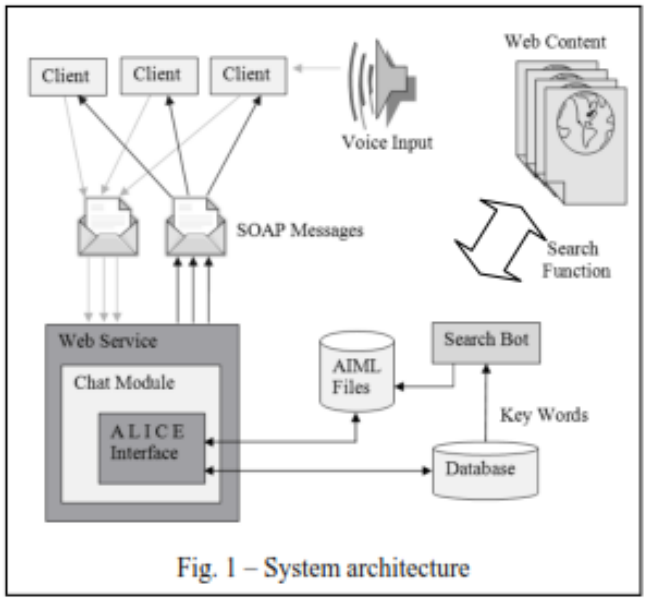

Fig. 1 – System architecture

The web service processes all received queries using the response generation module (based on the Artificial Linguistic Internet Computing Entity (ALICE) [10] system), which makes use of a data repository. The data repository is updated by the content retrieval module to increase the intelligence autonomously (based on an Artificial Intelligence Markup Language (AIML)). With this approach, an external administrator is only required to verify the quality of the self-training function. For a future query, the content is re-processed and incremental updates are made.

SYSTEM SPECIFICATIONS

The system presented in this paper meets the following requirements.

• The client application is easily accessible through the client browser.

• All communications to and from the server is text and XML formatted.

• XML messages conform to a schema which describes the format.

• Communications with the server is black box oriented and parses incoming XML messages seamlessly.

• The user is allowed to register and login to the system allowing for authenticated, personalized and controlled communication with the server.

• The client applet provides the user two options: text input or voice input.

• The self-training AI module prevents a service bottleneck, and therefore prevents modules from competing for resources.

OPEN SOURCE APPROACH

There are a number of available libraries and open source technologies to implement the specifications presented in this paper. This approach allows new implementations of a paradigm of using existing libraries and technologies to create custom implementations using open source libraries.

SYSTEM IMPLEMENTATION

The main language used to develop the demonstrator of this paper is JAVA [1]; and the applet is embedded using HTML. This approach of hiding the JAVA component from the end user creates an illusion of simplicity as shown in Fig. 2. The website contains the embedded applet, and is hosted by Apache web server[2]. The database and website is managed using open source database management software, MySQL [3].

Fig. 2 – Website with integrated JAVA based applet

The applet requires a number of libraries enabling for processing voice inputs. The signed libraries were kept on the hosting website, the same location as the applet. Before the applet launches inside a browser, a loading sequence streamlines the launch of the libraries. The code segment (edited to save space) below illustrates such a loading sequence.

```
<applet code="client.ICB" archive="Client.jar" name="IWBV">
 <param name="cache_archive"
value="Client.jar,lib/cmu_time_awb.jar,lib/cmu_us_kal.jar,lib/cmu
dict04.jar">
<PARAM NAME="cache_archive_ex"
VALUE="Client.jar,lib/cmu_time_awb.jar;preload,lib/cmu_us_kal
.jar;preload,lib/cmudict04.jar;preload">
<property name="freetts.voices"
value="com.sun.speech.freetts.en.us.cmu_time_awb.AlanVoiceDir
ectory"/>
<PARAM name="freetts.voices"
```

value="com.sun.speech.freetts.en.us.cmu_time_awb.AlanVoiceDir

ectory"/>

</applet>

The pre-loading of libraries allows for streamlined operation (i.e. the libraries are not called in an ad hoc basis, halting the sequence of activities). The user is prompted to accept a signature, upon such an acceptance; the applet can securely communicate to the web service. By using the open source development environment NetBeans [7], the applet and the libraries can be digitally signed. This allows the applet to communicate with a web service not located on its source web server. The chat client is interrupt driven and activates upon interaction from a user. This is shown in Fig. 3; where the control unit decides what component to launch next and what function to process.

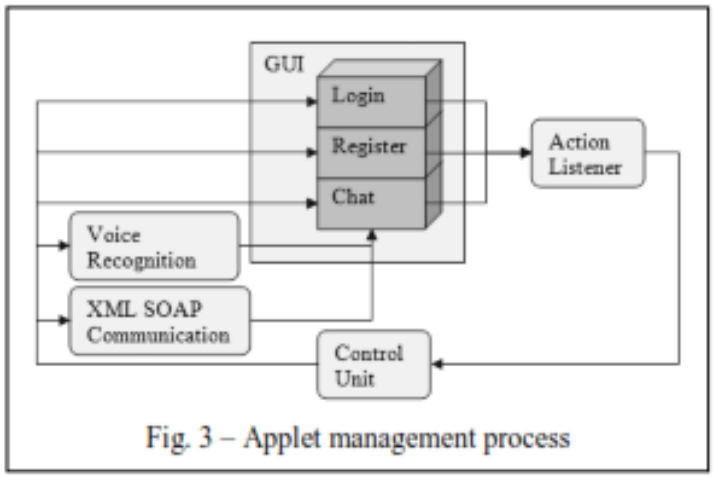

Fig. 3 – Applet management process

The action listener is bound to the buttons (login, register and chat) which is included in the graphical user interface (GUI). Once an event is triggered, either the corresponding SOAP communication module or voice recognition module or interface is activated. The input and output to these modules rely on user input captured via the GUI. Voice recognition is accomplished using an open source library called Sphinx 4 [8]. Speech recognition can be broken into two groups with five subsets in total; speakers and speech styles. Speakers make up single speaking and speaking independently where speech styles include isolated word recognition, connected word recognition and continuous word recognition.

Isolated word recognition requires long breaks between all words to successfully interpret words. Continuous word recognition also requires this but considerably shorter breaks. The last subset of speech styles is continuous speech recognition where one can speak fluently and not require stopping or breaking between words. This has problems interpreting similar vowel in the beginning of words. A vital part of the speech system is contained within a configuration file. As shown in Fig. 4, the configuration file is used to build the recognizer and (or) decoder using the application programming interface (API), including the dictionary. Grammar files which is used in generating the message is an integral part forming part of the acoustic model located in the API.

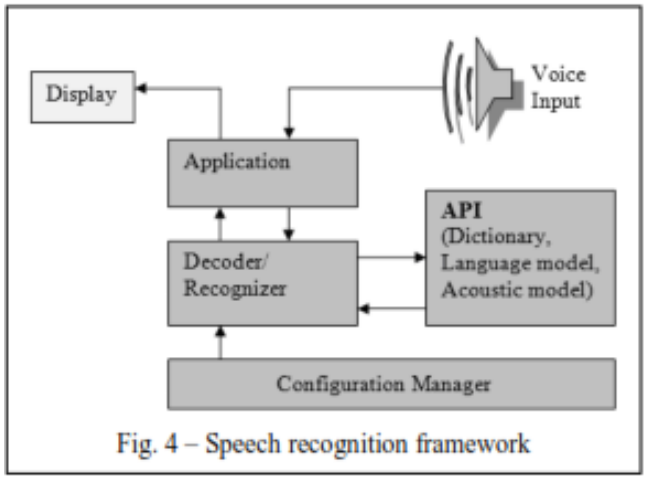

Fig. 4 – Speech recognition framework

The main components of the speech framework consist of the front-end application, decoder, and language model. The front-end acquires and attributes the voice input. The language and acoustic model is used to translate from a standard language (as input to the system) using a dictionary and construction of words located in a look-up table (LUT). A search manager located in the decoder then uses the attributes and LUT to decode the input voice into a result set. From the front-end, the user can activate the configuration manager, loading all components stored in XML format. When running a speech enabled application, the application requires more than 256 Mb in heap size thus placing excessive memory demands on the system Although a more toned down API (Pocket sphinx [9]), focusing on mobile platform is available, this paper focuses of Spinx 4. Communication with the web service is text based and XML-formatted using SOAP. The interface was generated using a JAVA-based architecture for XML binding (JAXB). An XML schema based API can also be published on a website for other developers to develop their own clients. This process of generating the interface is shown in Fig. 5.

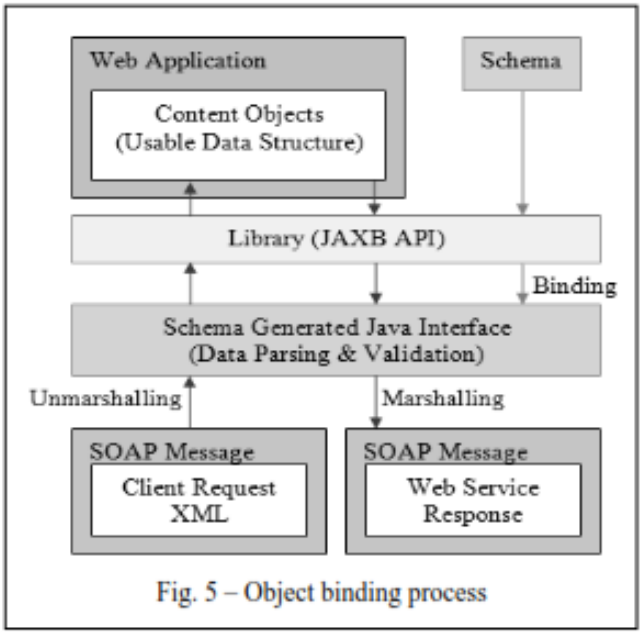

Fig. 5 – Object binding process

All messages are parsed through this interface which creates structure and validates the XML at the same time. The process of parsing the XML to a usable structure is termed marshalling. When an object is created, the XML content

is handled using the object extraction through the interface class created on startup as indicated in the next code segment.

```
JAXBContext.newInstance jaxbContext =

JAXBContext.newInstance( "icbxml" );

Unmarshaller u = jaxbContext.createUnmarshaller();

u.setEventHandler(new ICBXMLValidationEventHandler());

outputPipe.connect(inputPipe);

byte[] bytes = xmlData.getBytes();

outputPipe.write(bytes);

JAXBElement<IcbXml> mElement =

(JAXBElement<IcbXml>)u.unmarshal(inputPipe);

IcbXml icbxmlvar = (IcbXml) u.unmarshal(inputPipe);

IcbXml po = (IcbXml)mElement.getValue();
```

The interface also provides fault response generation if a XML message is invalid. The error location is then encapsulated in XML using the bound object and sent using SOAP. The interface can be regenerated on demand if the requirements or format of the message format changes. The server maintains a large number of users through the process of synchronized threads. The most basic hypertext transfer protocol (HTTP) connection or request for content is done through sockets. When a client connects through a basic connection request (socket connection) the socket creates a new thread pair (incoming, outgoing and processes) to handle the client messaged. All messages pulled from the socket are then processed through the generated interface object.

CONCLUSION

It is really impossible to get all the required data on a single interface without the complications of going through multiple forms and windows. The present college chatbot intends to remove this difficulty by providing a common and user-friendly interface to solve basic queries of college students. The purpose of a chatbot system is to pretend a human conversation. The students can freely ask queries to bot any time. The chatbot provides quick and effective search for answers to the queries. The database holds information about questions, answers, keywords, and logs. We have also developed an interface which will have two parts, one for users and the other for the administration. The core use of threads allowed multiple processing of incoming and outgoing messages to occur without having to create a waiting scenario or unavailable server due to over use or possible congestion. All new connections to the server

spawned a new pair of threads without impacting on other threads related to other users. The use of such a framework is not limited to chat applications only. The true potential lies with information systems that could be built into the existing framework due to the distributed nature. Depending on the requirements of the integration, the necessary additional components can be introduced on the artificial brain level or the AIML file functionality.

REFERENCES

[1]. Augello A. Saccone G. Gaglio S. Pilato G., Humorist Bot: Bringing Computational Humour in a Chat-Bot System. Proceedings of the International Conference on "Complex, Intelligent and Software Intensive Systems (CISIS)", 4-7 March 2008, Barcelona, Spain, pp.703- 708.

[2]. Gambino O. Augello A. Caronia A. Pilato G. Pirrone R. Gaglio S., Virtual conversation with a real talking head. Proceedings of the Conference on "Human System Interactions", 25-27 May 2008, Kraow, Poland, pp. 263-268.

[3]. Vojtko J. Kacur J. Rozinaj G., The training of Slovak speech recognition system based on Sphinx 4 for GSM networks. Proceedings of International Symposium "EL, MAR (Electronics in Marine) focused on Mobile Multimedia", 12-14 Sept. 2007, Zadar, Croatia, pp. 147-150.

[4]. Sun Microsystems, Developer resources for JAVA technology. [Online] http://java.sun.com (Accessed: 30 Oct. 2008)

[5]. The Apache Software Foundation, The Apache HTTP Server Project. [Online] http://www.apache.org (Accessed: 30 Oct. 2008)

[6]. Sun Microsystems, MySQL: The world's most popular open source database. [Online] http://www.mysql.com (Accessed: 30 Oct. 2008)

[7]. Sun Microsystems and CollabNet, NetBeans. [Online] http://www.netbeans.org (Accessed: 30 Oct. 2008)

[8]. Carnegie Mellon University, Sun Microsystems, Mitsubishi Electric Research Laboratories, Sphinx-4 - A speech recognizer written entirely in the JAVA™ programming language, 2004. [Online] http://research.sun.com (Accessed: 30 Oct. 2008)

[9]. Carnegie Mellon University (CMU). Speech at CMU. [Online] http://www.speech.cs.cmu.edu (Accessed: 30 Oct. 2008)

[10].ALICE AI Foundation, Inc. ALICE. the artificial linguistic internet computer entity. [Online] http://www.alicebot.org (Accessed: 30 Oct. 2008)

[11].A. Hoskinson, Ultimate Research Assistant. [Online] http://ultimate-research-assistant.com (Accessed: 30 Oct. 2008)