An Architecture to Develop Multimodal Educative Applications with Chatbots

Regular Paper

David Griol1,* and Zoraida Callejas2

1 Department of Computer Science, University Carlos III of Madrid, Leganés, Spain
2 Department of Computer Languages and Systems, University of Granada, CITIC-UGR, Granada, Spain
* Corresponding author E-mail: dgriol@inf.uc3m.es

Received 13 Jun 2012; Accepted 10 Jan 2013

© 2013 Griol and Callejas; licensee InTech. This is an open access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Abstract Animated characters are beginning to be used as pedagogical tools, as they have the power to capture students’ attention and foster their motivation for discovery and learning. However, in order for them to be widely employed and accepted as a learning resource, they must be easy to use and friendly. In this paper we present an architecture that facilitates building interactive pedagogical chatbots that can interact with students in natural language. Our proposal provides a modular and scalable framework to develop such systems efficiently. Additionally, we present Geranium, a system that helps children to appreciate and protect their environment with an interactive chatbot developed following our scheme.

Keywords Multimodal Dialogue Systems, Chatbots, Educative Technology

1. Introduction

According to Roda et al. [1], educative technologies should (i) accelerate the learning process, (ii) facilitate access, (iii) personalize the learning process, and (iv) enrich the learning environment. These aspects can be addressed by means of dialogue systems by establishing a more engaging and human-like relationship between the students and the system.

Multimodal dialogue systems can be defined as computer programs designed to emulate communication capabilities of a human being, including several communication modalities (such as speech, gestures, movements, gaze, etc.) [2-5].

Using natural language in educational software allows the students to devote their cognitive resources to the learning task, rather than to working out how to use the interface of the application [6]. Communicative and anthropomorphic characteristics of pedagogical dialogue systems may help support the social dimension of the learning activities, and it has been argued that the social context helps the cultivation of, and motivation for gaining, knowledge [7]. Positive effects on motivation and user engagement have been frequently noted [8-10]. In some cases motivation may be actively supported through deliberate motivational tutoring techniques; in others it may be a result of exposure to a novel technique.

There exist many possibilities for employing dialogue systems for educative purposes, e.g., as assistants,
moderators, or virtual peers [8, 11]. In some systems the student must teach the agent [12], interact with agent peers or co-learners [13], or face troublemakers intended to provoke cognitive dissonance to prompt learning [14].

In this paper we describe an architecture that facilitates the building of chatbot interfaces for educative systems integrating multimodal dialogue systems and animated characters. Our proposal facilitates a rapid and cost-effective development of such systems, providing a modular architecture in which pedagogical and technical contents are seamlessly updated and enhanced. In addition, we describe the application of our proposal to developing a system that helps children to learn about and look after their environment by means of interaction with a multimodal chatbot.

The remainder of the paper is organized as follows. Section 2 describes related research in the development of educative systems that integrate conversational functionalities. Section 3 describes the main characteristics of our architecture. Section 4 shows a practical implementation of our proposal to develop the Geranium educative system. Section 5 presents an evaluation of the system with teachers and children. Finally, conclusions and future work are presented in Section 6.

2. State of the art

As described in the previous section, with the growing maturity of conversational technologies, the possibilities for integrating conversation and discourse in e-learning are receiving greater attention, including tutoring [15], question-answering [16], conversation practice for language learners [10], pedagogical agents and learning companions [17], and dialogues to promote reflection and metacognitive skills [18].

The design, implementation and strategies of dialogue systems employed in e-learning applications vary widely, reflecting the diverse nature of the evolving speech technologies. The conversations are generally mediated through simple text-based forms [9], with users typing responses and questions with a keyboard. Some systems use embodied dialogue systems [19] capable of displaying facial expressions and gestures, whereas others employ simpler avatars [20]. Speech output using text-to-speech synthesis is used in some systems [19], and speech input systems are increasingly available [21, 22].

Dialogue systems as personal coaches integrate information about the domain of the application. Systems of this kind are characterized by the possibility of representing and continuously updating information on users’ cognitive and social state. The main objective is to guide and monitor users in the learning process, providing suggestions and other interaction functionalities not only with the developed application but also with other students. In order to achieve this goal, these applications usually integrate realistic and interactive interfaces.

For example, Grigoriadou et al. [23] describe a system where the learner reads a text about a historical event before stating their position about the significance of an issue and their justification of this opinion. The answers are classified as scientific, towards-scientific or non-scientific, and a dialogue generator produces “appropriate reflective diagnostic and learning dialogue for the learner”. Similarly, in the CALM system [24] the users answer questions on the domain, and state their confidence in their ability to answer correctly. The system infers a knowledge level for the students based on their answers, and encourages the learners to engage in a dialogue to reflect on their self-assessment and any differences between their belief and that of the system about their knowledge levels.

Spoken dialogue systems have also been proposed to improve phonetic and linguistic skills. For example, Vocaliza is a dialogue application for computer-aided speech therapy, which helps in the daily work of speech therapists that teach linguistic skills to Spanish speakers with different language pathologies [25]. In addition, the Listen system (Literacy Innovation that Speech Technology Enables) is an automated Reading Tutor that displays stories on a computer screen and listens to children read aloud [26].

Dialogue systems may also be used as role-playing actors in simulated experiential learning environments. In these settings, the dialogue system carries out a specific function in a very realistic way inside a simulated environment that emulates the real learning environment. However, perhaps the most popular application of dialogue systems in education is tutoring systems. Kumar et al. [27] have shown that agents playing the role of a tutor in a collaborative learning environment can lead to improvement of more than a grade. Some studies [28-30] have evaluated the effect of task-related conversational behaviour in tutorial dialogue scenarios, whereas work in the area of affective computing and its application to tutorial dialogue has focused on identification of students’ emotional states, [31] and using these to improve the choice of task-related behaviour by tutors.

For example, the AutoTutor project [32] provides tutorial dialogues on subjects including university-level computer literacy and physics. The tutoring tactics employed by AutoTutor assist students in actively constructing knowledge and are based on extensive analysis of naturalistic tutoring sessions by human tutors. The technology behind the system includes the use of a dialogue manager, curriculum scripts and latent semantic
Another example of natural language tutoring is the Geometry Explanation Tutor [34], where students explain their answers to geometry problems in their own words. The Oscar conversational intelligent tutoring system [35] aims to mimic a human tutor by implicitly modelling their learning style during tutoring, personalizing the tutorial to boost confidence and improving the effectiveness of the learning experience. The system uses natural language to provide communication about specific topics and dynamically predicts and adapts to a student’s learning style.

Other systems provide a visual representation through an animated bot with gestures and emotional facial displays. These bots have shown to be a good interaction metaphor when acting in the role of counsellors [36, 37], personal trainers [38], or healthy living advisors [39], and have the potential to involve users in a human-like conversation using verbal and non-verbal signals [40]. Due to these features, they can be successfully employed in the pedagogical domain [41], as well as in other domains where it is important to establish long-term relations with the user [42, 39, 40, 43-45].

However, it is difficult to communicate with these agents whenever needed (i.e., when the user is not in front of a computer but he/she nevertheless needs suggestions and advice). Therefore, if we want to support students in a continuous way, the personal advisor should be available and accessible also on a mobile device. Our work represents a step in this direction.

Multimodal dialogue systems are a natural choice for many human-robot applications [46, 47] and are important tools to develop social robots for education and entertainment applications [48]. A mobile robot platform that includes a spoken dialogue system is presented in [49], which is implemented as a collection of agents of the Open Agent Architecture. The dialogue component uses Discourse Representation Structures (DRS) from Discourse Representation Theory (DRT) to represent the meaning of the dialogue between human and robot.

An interaction framework to handle multimodal input and output designed for face-to-face human interaction with robot companions is described in [50]. The authors emphasize the crucial role of verbal behaviour in human-robot interaction. A spoken dialogue interface developed for the Jijo-2 mobile office robot is described in [51]. A microphone array system and a technique of switching multiple speech recognition processes with different dictionaries are introduced to achieve robust speech recognition in noisy office environments.

The TeamTalk human-robot interface is described in [52]. The system is capable of interpreting spoken dialogue interactions between humans and robots in consistent real-world and virtual-world scenarios. An intelligence model for conversational service robots is presented in [53]. The model includes expert modules to understand human utterances and decide robot actions. It also enables parallel tasks, as well as switching and cancelling tasks based on recognized human intentions.

Chatbots trained on a corpus have been proposed to allow conversation practice in specific domains [54]. This may be restrictive as the system may only “talk” in the domain of the training corpus, but the method may be useful as a tool for languages that are unknown to developers or where there is a shortage of existing tools in the corpus language. Fryer and Carpenter [10] argue that chatbots give students the opportunity to use varying language structures and vocabulary (e.g., slang and taboo words), which they may otherwise have little chance to experience.

There is therefore a wide variety of applications of conversational systems in education. Implementations may employ full embodied dialogue systems with emotion or gesture display, synthetic voice output, simple text-based output, dialogue with an accompanying avatar, and many variants or combinations of these.

Due to this variability and the huge amount of factors that must be taken into account, these systems are difficult to implement and typically are developed ad-hoc, which usually implies a lack of scalability. In the next section we describe the proposed architecture to allow the easy development of a multimodal chatbot for pedagogical applications using a modular approach. This way, different alternatives can be considered for each module, and the pedagogical knowledge is separated from the technical details, so that teachers and parents can add new content without having a technical background, whilst the software includes these new data for the interaction with the students.
3. Our architecture to develop pedagogical applications with chatbots

As explained in Section 1, multimodal dialogue systems are responsible for transferring user commands to the chatbot control system, performing speech and visual communication, and reporting task execution results to the users. During the interaction the dialogue system has to manage the dialogue initiative, handle miscommunication, draw inferences between interlocutors' contributions and organize and maintain the discourse.

Systems developed to provide these functionalities typically rely on a variety of components, such as speech recognition (ASR) and synthesis (TTS) engines, natural language processing components (NLU), dialogue management, and graphical user interfaces. As explained in the previous section, laboratory systems usually include specific modules of the research teams that build them, making portability difficult. Thus, it is a challenge to package these components so that they can be easily installed by novice users with limited engineering resources. As such, system deployment is usually limited to hardware within the laboratory, limiting data collection to traditional in-lab user studies.

To ease the development of such systems, we contribute the architecture shown in Figure 1, which allows the development of multimodal applications with a chatbot that can interact with the student orally or through the graphical interface. The systems implemented using the architecture generate personalized questionnaires including selected questions, perform the interaction by means of a conversational animated agent, carry out the corresponding analysis of the students' answers, and provide them with the appropriate feedback.

In our architecture, each of the modules might be considered a “black box”, so that third-party software can be used to ease development. This way, commercial ASR and NLU systems can be employed easily within our architecture.

The student employs the oral interface (which he/she may combine with the GUI) to provide responses to the questions posed by the system. Such oral responses are processed by the ASR, which outputs the most probable word sequence. This sequence is then interpreted by the NLU module, which generates a semantic representation.

Next, the User Answer Analyser obtains the meaning of the user input from the NLU module and checks whether it corresponds to the correct answer defined in the database. Then, it calculates the percentage of success and the set of recommendations to be made to the student. This can be done by means of grammars in which the student’s answer is compared with the reference answer, assigning a specific score and answer each time a coincidence is detected. This way, in our system we provide a balance between accuracy and flexibility in the evaluation process. Test questions provide total reliability in the correction of these answers, while our grammar-based functionality offers the flexibility of natural language.

With this information, the dialogue manager decides on the system response, also considering the confidence measures provided by the ASR and NLU modules. Given
that speech recognition is not perfect, one of the most critical operations of the design of the dialogue manager is related to error handling. Neither speech recognition nor language understanding operates without errors, and so user interfaces are far from being as effective as humans. One common way to alleviate errors is to use techniques aimed at establishing a confidence level for the speech recognition result, and to use this to decide whether to ask the user for confirmation, or reject the hypothesis completely and re-prompt the user.

This way, the dialogue manager might decide to confirm the input, ask again for the information or consider it as valid, then acting in accordance with its correctness. The simplest dialogue model is implemented as a finite-state machine, in which machine states represent dialogue states and the transitions between states are determined by the user’s actions. Frame-based approaches have been developed to overcome the lack of flexibility of the state-based dialogue models, and are used in most current commercial systems. For complex application domains, plan-based dialogue models can be used. They rely on the fact that humans communicate to achieve goals, and that during the interaction the human’s mental state might change [59]. Currently, the application of machine learning approaches to model dialogue strategies is a very active research area [60-61].

The system answer decided by the Dialogue Manager is presented to the student by means of the result generated by the Natural Language Generation and Text-to-Speech Synthesizer and the Multimodal Answer Generation modules. Natural language generation is the process of obtaining texts in natural language from a non-linguistic representation using predefined text messages. Then, the text-to-speech synthesizer transforms the text strings into acoustic signals. There is also the possibility of reproducing pre-recorded prompts stored in the database with generic messages.

The Multimodal Answer Generator is in charge of rendering the multimodal output of the chatbot by coupling the speech output with the visual feedback. The Gesture Selection module controls the incorporation of the animated character’s expressions. In order to do so, it selects the appropriate gestures from the database and plays them to the user.

Finally, the Questionnaire Generator module manages the current interaction to dynamically modify the selected questions according to the information provided by the different modules in the application. The policy proposed for the selection of the questions in each category takes into account the difficulty level and three parameters stored in the learning contents database: incorrect probability increase, correct probability decrease and level threshold.

At the beginning of the interaction with a student, each question with the lowest difficult level in the category is equiprobable, and the probability of the questions of the other difficulty levels is zero. If the student finds the correct answer easily, then the probability that the system chooses the same question in successive interactions decreases with the correct probability decrease factor, whereas if the question is incorrectly answered the probability that the system poses it again to the student increases in the incorrect probability increase and does not decrease until it is answered correctly twice. This way, the content that the students find more difficult to learn is automatically reinforced. Once a certain threshold of the questions in the difficulty level has been correctly answered (level threshold), the questions from the next level are assigned the same probability: that of the most probable question of the previous level plus the incorrect probability increase.

The architecture comprises three databases that contain the learning contents, multimodal expressions of the chatbot and the history of the interaction respectively. The first database stores the questions and answers categorized in different topics. For each question, there is a text, optional multimedia contents (audio and video) and several answers, as well as the difficulty level. For each answer, there is also text and/or multimedia, as well as the positive and negative feedbacks and hints to be provided to the student when he/she selects the respective answer. For each question only one answer is assumed to be correct. The database also contains the parameters employed by the Questionnaire Generator to decide which questions to pose to the students. The second database contains the visual rendering of the chatbot’s gestures and facial expressions, and the third database stores the information about the previous interactions of the user with the system.

The objective is to facilitate the inclusion of new questions and the editing of existing ones, indicating the corresponding responses of the chatbot and making possible the adaptation of the system to new domains. This way, different people can help in the development of the system without requiring expert knowledge in dialogue systems. For example, teachers and parents can enter new questions into the database, and graphic designers can create attractive animations for the chatbot and include them in the corresponding database.

The proposed architecture is inspired by previous works. For example, WebGALAXY [62] is the core architecture in the DARPA Communicator, which was developed as an early effort to make multimodal interfaces accessible via the web. With it, users view a website while speaking over the telephone to a spoken dialogue system which provides the required information. The system’s responses to user queries are spoken over the phone, with
corresponding information also displayed on the web page. It is based on a client-server model in which each module works as a server in a certain task (recognition, understanding, etc.), and the information flow is controlled in a hub that centralizes communication.

Olympus/RavenClaw [63] is an architecture to develop dialogue systems in which each module behaves as an independent server and communicates through the Galaxy hub. The architecture provides default modules such as Sphinx (speech recognizer), Festival (speech synthesizer) and Phoenix (natural language understanding), Rosetta (natural language generation) and Ravenclaw (dialogue management).

The main aim of the USI Project [64], also known as Speech Graffiti, is to design a universal interface independent of the application domain to facilitate the development of dialogue systems. However, the users must have previous knowledge about the protocol in which it is based, which is an intermediate alternative between natural language and directed dialogues.

ATLAS [65] is a library developed in Java to facilitate the development of multilingual and multimodal applications, and is designed as an intermediate layer which separates domain-specific information and the layer which contains the modules of the dialogue system. In order to use it, the speech recognizer and synthesizer must be compliant with the platform.

More recently, efforts have been undertaken to develop standards for deploying multimodal interfaces to web browsers, typically entailing the use of VoIP or local speech recognition. The two major standards under development, Speech Application Language Tags (SALT) [66] and XHTML+Voice (X+V) [67] are both aimed at augmenting HTML with speech interaction. In order to use them, a web browser that supports an interpreter is necessary. Recently, both seem to have lost support and it is difficult to find applications developed in such languages.

Our proposal differs from the previously described architectures in several aspects. First, it does not require use of a telephone, which is awkward in many situations. Second, it does not require special software, as it can be accessed from any of today’s standard browsers. Finally, the architecture provides an efficient means of building educative applications, allowing the rapid authoring of multimodal questionnaires which motivate the students and support the assessment of the students’ learning.

Thus, students can carry out several interactions with the platform, consulting contents corresponding to the different stages defined in the topics of the application, which they can use to evaluate their knowledge and extract conclusions about the results which should be obtained at the end of each of them. The students’ interactions with the chatbot can also provide essential information for both teachers and students. Teachers are provided with a feedback about the degree of the student’s understanding of the different contents. The interaction with the chatbot allows students to develop the ability to put concepts into practice, verifying whether their solution is correct or not, and doing so in an innovative application.

4. An implementation of the proposed architecture: the Geranium system

We have employed the architecture proposed in Section 3 to develop the Geranium pedagogical system, which can be used to make children aware of the diversity of the urban ecosystem in which they live, the need to take care of it, and how they can have a positive impact on it. The system has a chatbot named Gera, a cartoon that resembles a geranium, a very common plant in Spanish homes.

Figure 2 shows a snapshot of the system. As can be observed, it has a very simple interface in which the chatbot is placed in a neighbourhood. There are several buttons to select the type of questions, the chatbot has a balloon that shows the text, image or videos corresponding to the questions and possible answers, and there is a “push-to-talk” button that enables the oral input.

Figure 2. The Geranium system with the Gera chatbot

The activities are grouped in four topics: “in the market”, “animals and plants”, “waste”, and “water and energy”. In the first topic, the children are asked about fruits and vegetables, where the plants come from and the seasons when they are collected. The second topic comprises questions about animals and plants that live in the city, showing photographs, drawings and videos of birds, flowers, trees and leaves and how they change or migrate during the year. In the third topic, the children are asked about recycling, differentiating between the different
wastes and the suitable containers. Finally, the fourth topic deals with good practices to save water and energy at home.

Currently, there are 20 questions per category (a total of 80 questions), although the system can be extended to add new questions with their respective answers to the database. The selection of the questions to be asked at each time is carried out following the policy described in Section 3. As explained before, the three databases in the application store different contents with the aim of separating domain-dependent and domain-independent information so that introducing or editing contents is straightforward for teachers. We have also incorporated a simple tool in php to facilitate access to the database and the management of the contents.

The chatbot poses questions to the children that they must answer either orally or using the graphical interface. Once an answer is selected, the system checks if it is correct. In case it is, the user receives a positive feedback and Gera shows a “happy” (usual case) or a “surprised” (in the case of many correct questions in a row) face. If the answer selected is not correct, Gera shows a “sad” expression and provides a hint to the user, who can make another guess before getting the correct response. Gera has seven expressions: happy, ashamed, sad, surprised, talking, waiting and listening, shown in Figure 3, which can also be extended by adding new multimedia resources to the chatbot expressions database.

![Facial expressions of the Gera chatbot](image)

Figure 3. Facial expressions of the Gera chatbot

As the chatbot changes its expressions, the background also changes. For example, Figure 4 shows the response of the system to an incorrect answer: as can be observed, Gera is “sad” and the background has a grey colour. In the snapshot, the chatbot is providing a hint that will help the user to find the correct answer in the next try.

During oral communication, along with the speech and textual response, the chatbot provides feedback with its expressions. When it is listening to the user, it shows the “listening” face; if the answer selected is not understood, it shows an “ashamed” expression (see Figure 3. There are two additional expressions: “talking” and “waiting”.

![Snapshot of the response to an incorrect answer](image)

Figure 4. Snapshot of the response to an incorrect answer

The current version of the system is web-based. The application consists of the set of components described in the architecture and divided between a server and a client device running a standard web browser. Core components on the server provide speech recognition and speech synthesis capabilities, access to the databases, and a logger component which records user interactions.

This way, Gera is accessible to any user with a standard web browser and a network connection. In addition, the application can be easily accessed not only from desktop, laptop, and tablet computers, but from a variety of mobile devices as well. An audio controller component runs on the client device to capture the user’s speech and stream it to the speech recognizer, as well as to play synthesized speech generated on the server and streamed to the client.

The speech input is activated with a push-to-talk button and the recognition hypotheses generated by the recognizer are passed to the NLU module for processing. Once the input has been processed, the dialogue manager chooses the next system action for which a system answer is generated and synthesized.

The natural language understanding and dialogue management modules have been developed according to the Voice Extensible Markup Language (VoiceXML, www.w3.org/TR/voicexml20), defined by the W3C as the standard for implementing interactive voice dialogues for human-computer interfaces. VoiceXML allows the design of spoken dialogues that feature synthesized speech, digitized audio, recognition of spoken and DTMF (Dual-tone multi-frequency signalling) key input, recording of spoken input, telephony, and mixed initiative conversations.
VoiceXML applications are usually based on the definition of grammars for the NLU module, which delimit the speech communication with the system by specifying a set of utterances that a user may speak to perform an action or supply information, and for a matching utterance returns a corresponding semantic interpretation. Grammars are encoded following the Java Speech Grammar Format (JSGF, www.w3.org/TR/jsgf), supported by any VoiceXML platform, which allows these sentences to be specified in a compact way, and the embedding of semantics into the grammar. In addition, this format allows greater flexibility in terms of grammar structure and debugging. Tags, scopes, and weights are also supported.

In the Geranium system, for each question type there is a grammar template with the usual structure of the responses, and a new grammar is dynamically generated that makes use of the template and contains the exact response options for the actual question. Each of the options has an assigned code which is used also in the GUI and makes it possible to easily control the synchronization between the different modalities employed. Also the template makes it possible to maintain the same structure for the responses to similar questions (e.g., to ask for the name of a bird, a plant or a fruit in a photograph) even when they belong to different categories (e.g., birds are in the category “animals and plants” and fruits are in the category “in the market”). This facilitates the system usage, as it is easier for the users to know what the system expects.

Figure 6 shows an example which corresponds to the snapshot shown in Figure 2 in which the student is asked to tell the name of a bird. As can be observed, the question template is what_is_it.jgf, whereas the exact options for the response are in the question_ex1b3.jsgf grammar, which is generated at run time. Thus, the student utterances can be short, but also more elaborate, as for example: “Starling”, “A sparrow”, “It looks like a pigeon”, “I think it is a sparrow”, or “I am not sure but it can be a swallow”.

File question_ex1b3.jsgf:

```
#JSGF V1.0;
grammar question_ex1b3;
public <question_ex1b3> =
  <.../grammar_templates/what_is_it.jsgf#what_is_it> [out.opt =
  rules,options];
<options> = sparrow [this.out="0"] | pigeon [this.out="1"] | swallow;
  [this.out="2"];
<phrase> = starling [this.out="3"];
```

File what_is_it.jgf:

```
#JSGF V1.0;
grammar what_is_it;
public <what_is_it> =
  <pre_answer> | <options>;
<pre_answer> = [certainty] | [belief] | <phrase>;
<certainty> = “Of course” | “For sure” | “I am sure” | “I know” | (“I am
  not sure” [but!]) | (“I do not know” [but!])
<belief> = “think I believe;
<phrase> = [it is | “can be” | “might be” | “could be” | “may be” | “looks
  like” | “seems [to be]”] (a | an)
```

Figure 5. Transitions to render the chatbot responses in Geranium

Figure 6. Example grammars to recognize the answers to the question posed in Figure 2
The inclusion of static and dynamic grammars makes it possible to implement flexible dialogues with a wide range of possibilities from system-directed initiative to mixed initiative. Static grammars deal with information that does not vary over time, including the templates for the different question types and the grammars to control the exercises flow (e.g., to repeat a question, ask for help or select an option in the menu). Dynamic grammars include information that varies with time and make it possible to easily update and increment the learning contents.

This way, the NLU and dialogue manager modules are simplified by generating a VoiceXML file for each specific question in the system, including the corresponding system prompt and the grammar that defines the valid user's inputs for the prompt. Regarding dialogue management, all the events in the application are controlled using JavaScript. The dialogue manager selects the next system prompt (i.e., VoiceXML file) by following the JavaScript program that determines the order for the set of questions, which is based on the transitions summarized in Figure 5 and VoiceXML finite states.

In addition, we have considered different functionalities that allow the adaptation of the system, taking into account the current state of the dialogue as well as the characteristics of each user. On the one hand, we have captured the main VoiceXML events: noinput (the user does not answer in a certain time interval, or the answer is not sensed by the recognizer), nomatch (the input does not match the recognition grammar or is unrecognized), and help (the user explicitly asks for help).

On the other hand, the management of the previously described events has been fine-tuned using the following properties: ConfidenceLevel, Sensitivity, DocumentFetchHint, GrammarFetchHint, and Bargein. The property ConfidenceLevel allows the accuracy of recognition to be adjusted in order for answers to be accepted. If the confidence score for an utterance is below the specified level, the utterance is rejected. Adjustments to the recognizer can be made with the Sensitivity tag. The valid range for this property is 0.0 (least sensitive to noise) to 1.0 (highly sensitive to quiet input). Thus, if the property is set to a low value, the recognizer is less sensitive to noise, but the student must speak more loudly in order to be recognized. The properties DocumentFetchHint and GrammarFetchHint allow the usage of the cache to be adjusted to make searches either safer or faster. The Bargein property controls whether or not the TTS or audio prompts may be interrupted by a user’s voice input.

Figure 7 shows an example of the application of the described process to generate a VoiceXML file to visualize the question presented in Figure 2. The grammars are shown in Figure 6. As can be observed, the VoiceXML file corresponding to each of the questions can include more than one system prompt. To do this, a prompt counter is defined to track the number of times the prompt has been used since the form was entered. The values for the properties are computed dynamically taking into account the dialogue history.

Regarding the graphical user interface, the system answer generator produces the HTML output for the GUI and the template to be used by the natural language generator to obtain the lexical form of the next system prompt, which is then synthesized. With respect to the input, the visual and oral modalities are synchronized by means of the codes assigned to the answers for each question, both in the HTML form and in the VoiceXML grammars.

5. Evaluation

Task-oriented multimodal dialogue systems are usually evaluated in terms of interaction parameters and subjective judgments [68-71]. Interaction parameters include the technical robustness and core functionality of the system components as well as system performance measures such as task completion rate,
whereas subjective usability evaluations estimate features like naturalness and quality of the interactions, as well as user satisfaction reported in questionnaires and interviews.

We followed this method to evaluate the Geranium system in a primary school with two groups of 20 students, each with a teacher, per age group. The evaluation was carried out in two phases.

In the first phase, the system was used and evaluated by the six teachers for the age groups 8, 9 and 10 years old, who rated the naturalness and pedagogical potential of the system. In the second phase, the system was used by one group of students per age group (60 students in total), who were later interviewed. The transcripts and recordings of the interviews were interpreted by the teachers in order to process the answers and obtain measures of satisfaction, as explained in the following sections.

During both phases, in addition to the subjective judgements, interactive parameters were also recorded and computed to carry out an objective evaluation of the system performance.

5.1 Evaluation with experts

In the first phase of the evaluation, the six primary school teachers rated the system using the questionnaire shown in Table 1. The teachers were told to bear in mind that the system was aimed at children of the same age as their students. The responses to the questionnaire were measured on a five-point Likert scale ranging from 1 (strongly disagree) to 5 (strongly agree). Additionally the experts were asked to rate the system from 0 (minimum) to 10 (maximum); there was also an open question to write comments or remarks.

The results of the questionnaire are summarized in Table 2. As can be observed, the satisfaction with technical aspects was high, as well as the perceived didactic potential. The chatbot was considered attractive and adequate and the teachers felt that the system is appropriate and the activities relevant. The teachers also considered that the system succeeds in making children appreciate their environment. The global rate for the system was 8.5 (in the scale from 0 to 10).

Although the results were very positive, in the open question the teachers also pointed out desirable improvements. One of them was to make the system listen constantly instead of using the push-to-talk interface. However, we believe that this would cause many recognition problems, taking into account the unpredictability of children’s behaviour. This is why most systems directed at children use the push-to-talk metaphor, as in the NICE system, for example [72]. Also, although they considered the chatbot attractive and its feedback adequate, they suggested creating new gestures for the chatbot to make transitions smoother.

### Technical quality

| TQ01 | The system offers enough interactivity |
| TQ02 | The system is easy to use |
| TQ03 | It is easy to know what to do at each moment |
| TQ04 | The amount of information that is displayed on the screen is adequate |
| TQ05 | The arrangement of information on the screen is logical |
| TQ06 | The chatbot is helpful |
| TQ07 | The chatbot is attractive |
| TQ08 | The chatbot reacts in a consistent way |
| TQ09 | The chatbot complements the activities without distracting or interfering with them |
| TQ10 | The chatbot provides adequate verbal feedback |
| TQ11 | The chatbot provides adequate non-verbal feedback (gestures) |

### Didactic potential

| DP01 | The system fulfils the objective of making children appreciate their environment |
| DP02 | The contents covered in the activities are relevant for this objective |
| DP03 | The design of the activities was adequate for children of this age |
| DP04 | The activities support significant learning |
| DP05 | The feedback provided by the agent improves learning |
| DP06 | The system encourages continued learning after errors |

### Table 1. Questionnaire employed for the evaluation of the system by experts

<table>
<thead>
<tr>
<th>Min / max</th>
<th>Average</th>
<th>Std. deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>TQ01</td>
<td>3 / 5</td>
<td>4.17</td>
</tr>
<tr>
<td>TQ02</td>
<td>3 / 4</td>
<td>3.67</td>
</tr>
<tr>
<td>TQ03</td>
<td>4 / 5</td>
<td>4.83</td>
</tr>
<tr>
<td>TQ04</td>
<td>5 / 5</td>
<td>5.00</td>
</tr>
<tr>
<td>TQ05</td>
<td>4 / 5</td>
<td>4.67</td>
</tr>
<tr>
<td>TQ06</td>
<td>4 / 5</td>
<td>4.83</td>
</tr>
<tr>
<td>TQ07</td>
<td>4 / 5</td>
<td>4.83</td>
</tr>
<tr>
<td>TQ08</td>
<td>4 / 5</td>
<td>4.50</td>
</tr>
<tr>
<td>TQ09</td>
<td>4 / 5</td>
<td>4.83</td>
</tr>
<tr>
<td>TQ10</td>
<td>4 / 5</td>
<td>4.67</td>
</tr>
<tr>
<td>TQ11</td>
<td>3 / 5</td>
<td>4.50</td>
</tr>
<tr>
<td>DP01</td>
<td>5 / 5</td>
<td>5.00</td>
</tr>
<tr>
<td>DP02</td>
<td>4 / 5</td>
<td>4.67</td>
</tr>
<tr>
<td>DP03</td>
<td>4 / 5</td>
<td>4.83</td>
</tr>
<tr>
<td>DP04</td>
<td>5 / 5</td>
<td>5.00</td>
</tr>
<tr>
<td>DP05</td>
<td>4 / 5</td>
<td>4.67</td>
</tr>
<tr>
<td>DP06</td>
<td>4 / 5</td>
<td>4.83</td>
</tr>
</tbody>
</table>

### Table 2. Results of the evaluation of the system by experts

From the interactions of the experts with the system we completed an objective evaluation of the application considering the following interaction parameters:
1. Question success rate (SR). This is the percentage of successfully completed questions: system asks – user answers – system provides appropriate feedback about the answer.

2. Confirmation rate (CR). This was computed as the ratio between the number of explicit confirmations and the number of turns in the dialogue.

3. Error correction rate (ECR). The percentage of corrected errors.

The results presented in Table 3 for the described six interactions show that the developed system could interact correctly with the users in most cases, achieving a question success rate of 96.56%. The fact that the possible answers to the questions are restricted made it possible to have a very high success rate in speech recognition. Additionally, the approaches for error correction by means of confirming or re-asking for data were successful in 93.02% of instances where the speech recognizer did not provide the correct answer.

<table>
<thead>
<tr>
<th>SR</th>
<th>CR</th>
<th>ECR</th>
</tr>
</thead>
<tbody>
<tr>
<td>96.56%</td>
<td>13.00%</td>
<td>93.02%</td>
</tr>
</tbody>
</table>

Table 3. Results of the objective evaluation of the experts’ interactions

5.2 Evaluation with children

In the second phase of the evaluation, 60 children (27 boys and 33 girls), 20 per age group of 8, 9 and 10 years old, used the system in sessions of 15 minutes. For the session, each group of 20 students was divided into two groups of 10. The students were supervised by their age group teacher (three teachers in total) and an assistant.

Prior to the session, the teacher explained to his students what the system is about and how it is used. Then, the children were given some minutes to get accustomed to the system, and initial difficulties were resolved by the teacher and the assistant.

Then, they interacted with the system during 15 minutes under the supervision of the teacher and the assistant. To do this, each child had a computer and had to wear a microphone headset. They were allowed to break up the interaction at any time for any reason.

At the end of the session each child was interviewed individually by the assistant in a different room so that they could not hear the responses of the other children. The interview was recorded and the assistant directed it according to the questionnaire outline shown in Table 4, adapting the questions to the age of the subjects (i.e., 8, 9 or 10 years old) and the development of the conversation.

As can be observed, there were mainly two types of questions: those related to the ease of use and likeability of the system, and those related to the learning experience. The interview was designed to be short so that the children could keep their attention during it.

### Interaction experience
- IE01. The system is fun
- IE02. The system is easy to use
- IE03. The chatbot is “personified”
- IE04. The chatbot is useful
- IE05. He/she would use the system again

### Learning contents
- LC01. The questions were easy to understand
- LC02. The questions were easy to answer
- LC03. The system helped to learn new things
- LC04. The system made him/her appreciate the environment

Table 4. Questionnaire outline for the interview with the children

After all the sessions were complete, the three teachers evaluated the 60 interviews mapping the natural language responses of the children onto numerical values on a Likert scale from 1 to 5 corresponding to the questions in Table 4. The agreement between the experts was computed using a kappa coefficient. The value obtained was 0.83, which, in the benchmark by Landis and Koch [73], corresponds to “almost perfect agreement”.

The results of the evaluation are shown in Table 5. As can be observed, the system was rated fairly well and most of the children learned new contents while having fun. In fact, most of the children would like to play again with the system.

<table>
<thead>
<tr>
<th>Questions</th>
<th>Min / max</th>
<th>Average</th>
<th>Std. deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>IE01</td>
<td>4 / 5</td>
<td>4.70</td>
<td>0.26</td>
</tr>
<tr>
<td>IE02</td>
<td>2 / 5</td>
<td>4.33</td>
<td>1.40</td>
</tr>
<tr>
<td>IE03</td>
<td>4 / 5</td>
<td>4.55</td>
<td>0.34</td>
</tr>
<tr>
<td>IE04</td>
<td>3 / 5</td>
<td>4.00</td>
<td>0.73</td>
</tr>
<tr>
<td>IE05</td>
<td>5 / 5</td>
<td>5.00</td>
<td>0.00</td>
</tr>
<tr>
<td>LC01</td>
<td>3 / 5</td>
<td>4.27</td>
<td>0.55</td>
</tr>
<tr>
<td>LC02</td>
<td>3 / 5</td>
<td>4.32</td>
<td>0.41</td>
</tr>
<tr>
<td>LC03</td>
<td>4 / 5</td>
<td>4.84</td>
<td>0.23</td>
</tr>
<tr>
<td>LC04</td>
<td>4 / 5</td>
<td>4.87</td>
<td>0.14</td>
</tr>
</tbody>
</table>

Table 5. Results of the evaluation of the system by the children

We verified that the questionnaire results are not influenced by the sample characteristics, as most of the children were familiar with multimedia applications. We did not find significant differences in the responses of boys and girls. The students were very satisfied with the experience, not only because it facilitated learning but also because it was entertaining for them.

However, we found that the students who had a worse performance answering the questions provided lower values in response to the questions related to the ease of
understanding and answering the system and to the ease of use. These are the questions with lower minimum value (see Table 5), although the average rating was in all cases above 4.

For the objective evaluation, we were interested in evaluating the robustness of the system during the interaction with children. For this reason, we returned to the statistical measures mentioned in Section 5.1.

The results presented in Table 6 for the 60 interactions show that Geranium could interact correctly with the users in most cases. When compared to the results of the experts (Table 3), the system made more recognition errors with children. An analysis of the main problems detected showed that most of these errors were due to the children not holding the push-to-talk button correctly and thus cutting the input, or because they used longer phrases or fillers which were not correctly processed by the system. However, in most cases, these problems could be overcome by confirming or asking again for the data, as shown by the question success rate of 91.53%.

<table>
<thead>
<tr>
<th>SR</th>
<th>CR</th>
<th>ECR</th>
</tr>
</thead>
<tbody>
<tr>
<td>91.53%</td>
<td>21.50%</td>
<td>90.82%</td>
</tr>
</tbody>
</table>

**Table 6. Results of the objective evaluation of the children’s interactions**

6. Conclusions and future work

In this paper we have proposed an architecture to cost-effectively develop pedagogical chatbots. It comprises different modules that cooperate to interact with students using speech and visual modalities, and adapt their functionalities taking into account their evolution and specific preferences.

Following the proposed architecture we have implemented the **Geranium** system, a web-based interactive software with a friendly chatbot that can be used as a learning resource for children to learn about the urban environment.

We have carried out an evaluation of the **Geranium** system with primary school children of 8, 9 and 10 years old and their teachers to assess its ease of use and its pedagogical potential. The study showed a high degree of satisfaction with the system appearance and interface, and the results were very positive with respect to its pedagogical potential.

For future work, we plan to replicate the experiments with a larger group of participants to validate these preliminary results, incorporating the suggestions provided by the teachers. We are also considering the possibility of implementing a version for tablets and digital blackboards.

7. References


