AN ARCHITECTURE FOR A MULTI-MODAL WEB BROWSER

Ivano Azzini
Dipartimento di Informatica e Sistemistica, Università di Pavia, Pavia, Italy
ivano@aim.unipv.it

Toni Giorgino
Dipartimento di Informatica e Sistemistica, Università di Pavia, Pavia, Italy
toni@aim.unipv.it

Luca Nardelli
ITC-irst, Pantè di Povo, Trento, Italy
lunarde@itc.it

Marco Orlandi
ITC-irst, Pantè di Povo, Trento, Italy
orlandi@itc.it

Carla Rognoni
Dipartimento di Informatica e Sistemistica, Università di Pavia, Pavia, Italy
carla@aim.unipv.it

1. INTRODUCTION

The very rapid evolution of telecommunication technology is leading to the convergence of fixed and mobile networks and devices. At present, it is very widespread for people to access the Web with Internet connections using HTML and/or WML (*Wireless Markup Language*) browsers, and present portable devices (e.g. PC/PDA, GPRS/WAP phones) offer a range of features (e.g. large memories, graphical displays, friendly user interfaces, communication interfaces, including the possibility to install Internet browsers) that make them suitable for hosting quite all of the applications that can be normally performed by standard PCs. Their main limitation is related to the reduced input/output capabilities, since they frequently lack of an alphanumeric keyboard and have very small displays. In this case, the development of multi-modal browser with voice input/output capabilities and/or using other devices (e.g. graphic pointing, touch screens, small numeric keyboards, etc.) should satisfy a large variety of user requirements.

The idea we propose in this paper consists in the definition (and consequent realization) of an architecture capable of handling multi-modal browsing through the synchronization of HTML and VoiceXML documents. In doing this, we have to consider issues related to the variability of user/terminal profiles, as well as issues related to the layout adaptation to different presentation modalities (e.g. spatial/temporal axes and *hyper-linking*). VoiceXML enables users to browse documents by speaking and hearing on a phone, but does not support a graphic interface, as HTML or WML do. We propose to synchronize different documents through a specific platform instead of adding new features to existing HTML, WML or VoiceXML documents.

This approach has the advantage of allowing, in a quite general way, multi-modal browsing of existing HTML documents by developing corresponding VoiceXML documents. In any case, we do not exclude the possibility of defining an XML schema that will include, on a general basis, both HTML and VoiceXML syntax, thus allowing a multi-modal definition of an application in a single document. What we want to point out is that the system we are proposing can be used in a quite general way, provided that the Markup documents, describing the web service to realize, are correctly interpreted by specific components.

The work presented in this paper has been partially developed inside the E.U. project *Homey* [2, 3]. The purpose of this project is to monitor the clinical state of chronic patients (in particular patients affected by hypertension pathologies), by means of the telephone, as will be described in section 3.1.

Another application under investigation is to use the multi-modal browser for accessing the WebFabIS information system (a system for the data management described in section 3.2). The benefits resulting from the adoption of mobile devices are currently being investigated. Finally, we are going to introduce our technology into the Oncological Electronic Medical Record application developed by the Telemedicine and medical Informatics laboratory of ITC-irst (see section 3).

2. SYSTEM DESCRIPTION

In ITC-irst, a client-server architecture (called *SPINET - SPeech INto Enriched Text)* for developing voice applications has been realized. The SPINET Server (see fig.1) can handle multiple simultaneous connections with applications that need speech recognition resources (e.g. recognition engines, text to speech engines, wave sources, grammar compilers, phonetic transcribe, etc.); the modules of the architecture can be distributed inside a Local Area Network (LAN), the communication among them is handled by means of “sockets”.

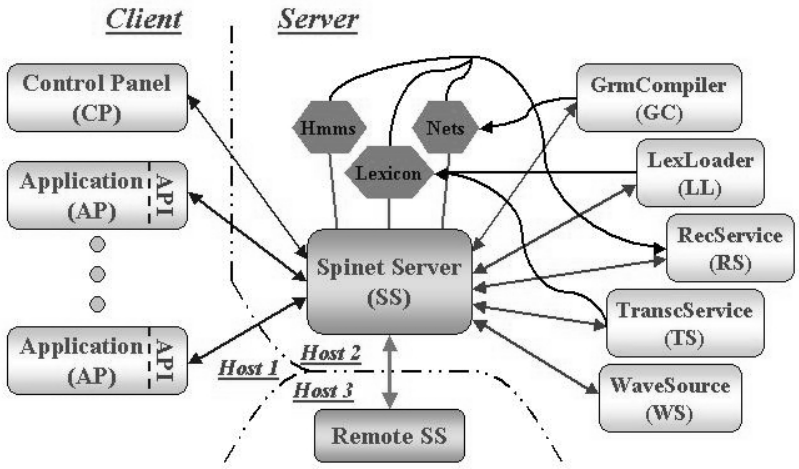

Figure 1: Architecture of Spinet (Speech Into Enriched Text), the ITC-irst speech recogniser.

The developer of a voice application can use a particular resource by using of an Application Programming Interface (*Spinet* API) provided with the *Spinet* package. The *Spinet* API can be accessed by applications developed in both the C++ and Java Programming Languages.
Furthermore an interpreter of the VoiceXML language has been realized and, also in this case, application programmers can access its functions through a Java API.
Recently, ITC-irst is working on Text to Speech (TTS) technology, by using the Festival environment (see http://www.itc.it/IRST/PrgIRST/DITELO.htm). A first release of this TTS can be freely downloaded from http://www.cstr.ed.ac.uk/projects/festival/.

The architecture of the multi-modal browser proposed in this paper is shown in Figure 2.

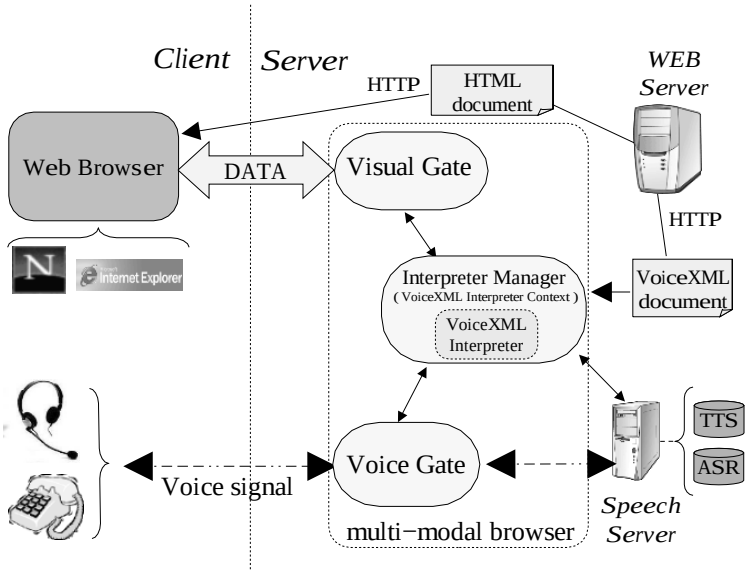

Figure 2: Architecture of the proposed multi-modal browser.

A client uses a traditional Web browser (e.g. Netscape or Internet Explorer) to interpret HTML documents. The client is also able to acquire and transmit the voice signal to the server (voice channel). Note that the speech signal can be transmitted either through PSTN (Public Switch Telephone Network) or through TCP/IP connections; in the first case we need to have telephone boards on the server side while in the second case an application, on the client side, will have to manage the voice channel between the Web server and the client browser.

The server side hosts the following components.

- A conventional Web server, handling requests for HTML documents and the corresponding VoiceXML counterparts. The Web server manages all of the resources needed for ASR and TTS functions (e.g. grammars, speech files, etc.).
- The Speech Server (in our case Spinet), which manages both ASR and TTS engines.
- The multi-modal browser, which integrates our VoiceXML Interpreter.

The multi-modal browser consists of three components:

- a Voice Gate, that answer to the user's requests by means of a Voice Manager (this last one controls the voice channel);
- a Visual Gate, that answer to the user's requests by means of a Visual Manager that controls a TCP/IP connection with the Web browser; this connection allows to synchronize HTML and VoiceXML

documents as we will be explained bellow;

- an Interpreter Manager, corresponding to the VoiceXML Interpreter Context (see www.voicexml.org), that integrates our VoiceXML Interpreter.

Both Managers (voice and visual) interact with the VoiceXML Interpreter through the Interpreter Manager.

Our solution for synchronizing HTML and VoiceXML documents (see Figure 2) consists in opening and maintaining a TCP/IP connection between the Web, on the client side, and the multi-modal browser on the server side.

Information is exchanged along the TCP/IP connection of in order to:

- modify the incoming HTML pages according to the input provided by the user's utterances;
- update the VoiceXML context according to "non voice" input (e.g. provided by the user via keyboard and/or mouse).

For example, when the user utters one or more values for filling fields in a displayed form, the system modifies the loaded Document Object Model (DOM, it allows to access the structure of HTML documents) pages in order to show the uttered data. On the other hand, if the user types in the value of a field, the Web browser communicates the event to the multi-modal browser on the server side through the Visual Manager, which in turn updates the corresponding VoiceXML field item variable.

Two HTML frames are used on the client side:

- the first frame covers the full visible area of the client browser and is used to display the HTML documents.
- the second frame is hidden: a Java applet and a JavaScript (consists of a set of JavaScript functions) are loaded inside it.

The advantage of this solution is that both applet and JavaScript file are loaded only at the beginning of the multi-modal service, therefore the connection between client and server sides is always active.

The JavaScript functions allow both to update DOM objects on demand (e.g. the *value* of an input field), and to dispatch the events occurring when users interact with a Web page (e.g. through a button press or a text field modification). The Java applet manages the TCP/IP connection between the client and the server, by transmitting and receiving commands according to a specific communications protocol. The applet and the JavaScript functions can communicate thanks to the Live Connect technology developed by Netscape™ (presently both Internet Explorer and Netscape Navigators are able to use this technology [4]).

Figure 3 shows how the “non-voice” user input is managed by the multi-modal browser.

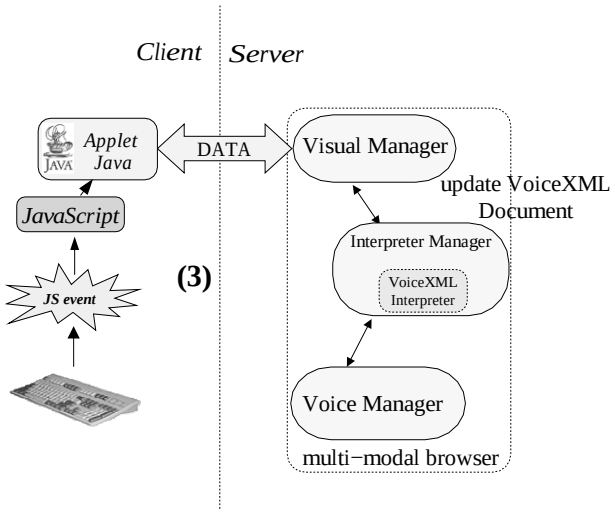

Figure 3: Non-voice interaction.

Essentially, a request for updating VoiceXML documents is sent to the Visual Manager by the applet, which, in turn is asked to do it by JavaScript function calls.
The Visual Manager analyses the request and communicates to the Interpreter Manager to execute the necessary operations.
Similarly a voice interaction generates a recognition event, which the Interpreter Manager uses in order to modify the context of the current VoiceXML document. The multi-modal browser, through the Interpreter Manager and the Visual Manager, uses this recognition event to update the corresponding HTML document through the TCP/IP connection opened with the applet. Finally, the applet calls the JavaScript functions that modify the Web page (see Figure 4).

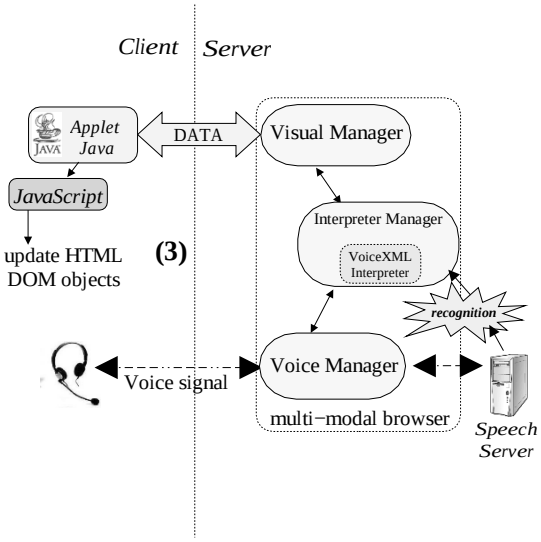

Figure 4: Voice interaction.

On the server side a user-friendly interface allows to control the status of all users by means of control windows.
We want to point out that this architecture allows independence between voice and visual user interactions. When a user ask for a specific modality, the appropriate Gate will first control if the Interpreter Manager for that user is present: in this case the Gate informs the Interpreter Manager about the new added modality (in order to enable the Interpreter Manager to exchange commands or data on the TCP/IP connection or on the voice channel); otherwise, a new Interpreter Manager is created for the new host with the current requested modality; it will be active as long as the user interacts with one of the two modalities.

Note that, there is a problem related to mark-up language syntax: it is not always possible to automatically generate a visual component from the verbal component (e.g. voice prompts, grammars) and vice-versa. We are currently investigating the possibility to include VoiceXML and HTML pages in a single document (a possible solution is to use the XSL markup language proposed in [7]).
Furthermore, in this project we intend to investigate the possibility of distributing the resources in different ways, particularly on the client side (e.g. the ASR/TTS), thus avoiding to transmit the speech signal to the Web server. Our final goal is to use our multi-modal browser on portable device (e.g. PDA and UMTS terminal)

3. APPLICATION SCENARIOS

3.1. The Homey Project

The Homey project is currently investigating the use of voice recognition technology in the context of hypertension care.

Recent guidelines [1] suggest that patients affected by high blood pressure should have their blood pressure values monitored relatively frequently (e.g. once a week). The Homey collaboration has developed a database, which keeps a record of a wide range of clinical data associated to patients under treatment for hypertension. The database stores patient anagraphic records, detailed cardiovascular and familiar anamnesis, and habits. The physician also records the prescribed clinical exams and the respective outcomes, therapies assigned, side effects possibly observed, besides, of course, blood pressure and heart rate values. Most of this data is used to update three standard risk indicators (e.g. Framingham); the history of these indicators can be monitored in time in order to check therapy effectiveness. The database consists of approximately 30 tables, not counting those used for decoding.

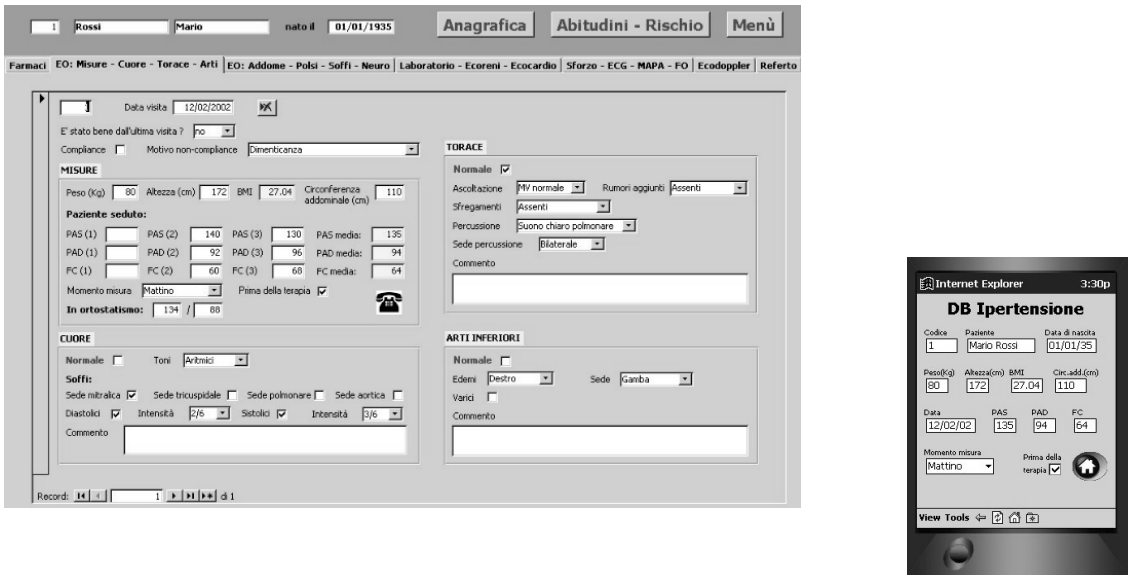

Figure 5: A part of the graphical user interface of the Homey database, and its reduced version suited for PDA displays. The small telephone icon indicates that the data entry was performed by the patient himself, dialling into the automatic dialogue system. The most frequently used input fields will be voice-enabled by the multi-modal technology explained in the article.

While most of the records of the database will be in control of the physician, the database also has an interface towards a telephone gateway (see figure 5). Patients will be able to dial a toll free number from home and enter some relevant data by themselves, such as blood pressure values and side effects of his/her therapy possibly observed. The data-entry happens via a mixed initiative dialogue, which takes place through TTS and ASR technology; the latter is provided by a SPINET server.

We are currently evaluating the possibility of providing to physicians the database voice access as well as to

patients. The idea is that the most frequently used subset of the database could have a multi-modal interface, so that health care personnel could browse and fill it via both hands-on as well as hands-free operation, without the need of explicitly switching the input modality. To achieve this goal, according to what has been discussed above, we will create a version of the database GUI and a description of the desired vocal interaction, respectively in the HTML and VoiceXML languages. These two documents will be generated dynamically according to the appropriate alignment rules so that the input fields and the field variables can be kept synchronized by the multi-modal browser. The dynamic generation of these two documents shall happen on-demand as they will be requested to the Web server during the user interactions. The generation of both files may be handled by established server-side technologies like CGI scripts, JSP and Java Beans.

This multi-modal interface will be first developed and tested on a desktop environment. The ultimate goal of multi-modality in our domain is to make the human-machine interaction more comfortable; we shall therefore study the impact of issues like the avoidance of repetitive or most error-prone inputs, when they could be avoided. We are also evaluating the opportunity of giving PDA devices to health care personnel. Such personnel could benefit a much reduced version of the database input forms, so that they could be displayed properly on wireless networked appliances. Multi-modal browsing could apply to these devices, as well, as long as their Web browser has appropriate support to Java and JavaScript, and the network architecture provides an appropriate sustained bandwidth towards the speech recogniser, or sufficient local sound encoding resources (e.g. real-time GSM compression).

3.2. The WebFabIS Project

WebFabIS is an information system for clean-room microelectronics laboratories. It is aimed to handle technical and organizational information: device recipes, process control, yield indicator, safety control, and resources optimisation. It is used by the clean-room staff in order to have device fabrication instructions and from the research team in order to store and maintain new device design and recipes. It has been developed for R&D labs, but can be easily adapted to production organization, with no structural modifications.

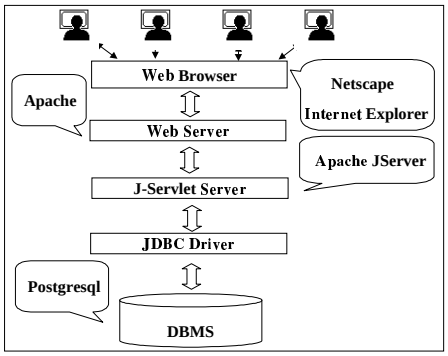

Figure 4: General overview of the WebFabIS architecture

In the microelectronics laboratory where WebFabIS is employed could happen that the operators cannot interact with the system through the keyboard. In this environment the operators wear uncomfortable cloths and gloves and have very often their hands busy with fragile silicon wafers. For this reason the possibility to interact with the system with the voice is much more then a "funny gadget". The architecture adopted for WebFabIS allows an easy integration with the multi-modal browser. In fact both systems are modular, Web

based and realized with Java (WebFabIS makes extensively use of HTML and of the Servlet technique). Note that for each HTML traditional page, the corresponding VoiceXML has to be generated.
WebFabIS has been designed with the most advanced software engineering techniques, e.g. UML and the Object-Oriented paradigm, and can of course be considered a Web application, since it can be accessed through Web browsers with no need of local applications (see Fig. 4). Strong attention has been paid to the modularity and the flexibility of the system. It results highly dynamic thanks to the use of the Java Servlets. Thanks to them, indeed, the same servlet can be used to interface the user to all the types of equipment and process steps.
The tools used to build the first release of the system are:

- a relational database: where all the information of the system are stored, e.g. technological process data, users password (PostgreSQL 6.5.3);
- a database driver: it allows the connection from the web server to the tables (JDBC);
- java and java servlets: used for the implementation of the browser interface (in "html" pages) and the processing of the information (JAVA2, JSDK2.0);
- web server (Apache Server1.3.11);
- java servlet engine: it manages through the Hyper Text Transfer Protocol "http" the communication between the client (user interaction with browser) and the server (database and java code) (Apache JServlet Server 1.1);
- web browser: it allows for user-system interaction (Netscape and Internet Explorer).

Among the quality targets of the system obtained thanks to the design techniques adopted:

- portability;
- access through the net;
- object-orientation;
- WEB based;
- easiness to use also to not trained people;
- it allows the contemporary access, also in write mode, to the same table from different users, without deadlock problems.

3.3. Oncological Electronic Medical Record Application

A distributed Oncological Electronic Medical Record (OEMR) was developed by the Telemedicine and Medical Informatics Laboratory of ITC in close collaboration with oncological specialists. The project (funded by the Italian Health Ministry) started in 1997 and finished in 2000. The aim of the project was to create a computer supported collaborative work (CSCW) environment for supporting the provision of expert advices from the referring cancer center in Trento to three peripheral hospitals.
The network infrastructure is an Intranet based on a Wide Area Network connecting the hospitals
The OEMR has Web-based architecture structured as a series of HTML pages logically linked, so as to reproduce the structure of a conventional medical record. The multimedia medical data (mainly medical images and text data) are stored in relational databases and are accessible through a Web browser from every hospital's ward involved in the care of an oncological patient. The patient data are stored into and retrieved from databases (Microsoft SQL 7) by the Web Server (Microsoft Internet Information Server). This last one dynamically creates the HTLM Pages using ASP (Active Server Pages) technology.
The system was deployed in all the involved wards and is currently used in the daily clinical routine.
There is a great deal of studies regarding the facilities for collecting, storing and consulting medical information.
In this context appears very useful to add voice capability to the existing Oncological clinical record infrastructure.

4. FUTURE WORKS

In this paper we have described the architecture developed in ITC-irst for handling multi-modal interactions with the Web using standard browsers. The architecture is quite flexible and can be adapted to different mark-up language specifications. We are currently developing some applications using the proposed system, specifically for data entry in a clean room and inside hospitals.
Given these applications scenarios, and other ones that we are going to investigate in the next future we intend to collect field data for studying and developing user interaction models capable to represent user preferences and needs. In this context we think it is fundamental to derive models that integrates, in some way, the probabilistic paradigm of spoken language system with the probabilistic methods of text based information retrieval.
Some ideas and results about this will be proposed in next conferences.

5. REFERENCES

[1] *The Sixth Report of the Joint National Committee on Prevention, Detection, Evaluation, and Treatment of High Blood Pressure*. NIH Publication No. 96-4080, November 1997;

[2] Ivano Azzini, Daniele Falavigna, Roberto Gretter, Giordano Lanzola, Marco Orlandi. *First steps toward an adaptive spoken dialogue system in medical domain*, EUROSPEECH 2001, Aalborg, Denmark, September 2001 - IRST Tech. Rep. No. 0104-21;

[3] E.U. Project "HOMEY" *Home monitoring through intelligent dialog system*. EC Fifth Framework project IST-2001-32434.

[4] http://www.netscape.com/eng/mozilla/3.0/handbook/plugins/

[5] G. Booch, J. Rumbaugh, I. Jacobson, UML: The unified modeling language –User Guide (Addison-Wesley 1999)

[6] S. Khoshafian, Object-oriented databases (Wiley, New-York, 1993)

[7] http://www.w3.org/Style/XSL/

[8] L. Ferrario, C. Armaroli, M. Zen "WebFabIS: A Web system for Microelectronics Laboratories Activity" RM2001- iasted conference - Cancun Mexico May 2001