

WWWSigTranscribe - Annotation via the WWW

Christoph Draxler

Department of Phonetics and Speech Communication, University of Munich, Munich, Germany
draxler@phonetik.uni-muenchen.de

Abstract

WWWSigTranscribe is a tool for annotating speech signals via the WWW. It was originally developed for the SpeechDat(II) project for an orthographic annotation of telephone speech, and has been extended to handle standard (WAV, AIFF, .au) and multi-channel audio file formats.

The WWW client-server architecture is of particular interest to educational software because it features platform independence, flexible access control, local as well as world-wide access, and a central storage of shared resources.

WWWSigTranscribe is implemented in Java 1.1 in a clean object-oriented design. It features a signal display window and an editing pane with editing buttons. Consistency checkers, e.g. for phonetic annotations, can be implemented easily. Due to its object-oriented design, WWWSigTranscribe can be adapted to new annotation formalisms easily. Because it is implemented in pure Java it supports Unicode which allows the use of IPA symbols or ideographical languages.

The WWWTranscribe toolbox is used at our department for teaching, annotation training, and the actual annotation work in projects.

1 Introduction

Successful education requires skilled teachers, motivated students, and high-quality tools to support both. Education in the speech sciences faces particular challenges:

- Phonetics and speech-related laboratories and departments usually are small and have limited personnel and hardware resources.
- Language sciences, e.g. linguistics, and engineering sciences, e.g. computer science or electrical engineering, require speech courses in their curricula, but have little or no means to offer such courses themselves.
- Finally, speech education requires a lot of practical hands-on experience – practice is everything.

A tool to support the needs of education in speech sciences should therefore provide easy access to speech resources, if possible without large investments in hardware or software both for students and the educational institution.

Teachers should be able to use the tool for demonstrations, and students should be able to work and

practice wherever they want – in lecture halls, in local or remote class rooms, or at home.

This tool should be integrated into an existing infrastructure to simplify the communication between teachers and students, and to avoid extra overhead. Furthermore, such a tool should be adaptable, extensible, and robust.

In speech science education some resources can be shared, others can be distributed. Databases, file servers, or software that requires specific machines typically are shared resources. They are either too large to be distributed, or they may not be distributed due to licence restrictions, or they require that data changes, e.g. due to updates, must be available to everyone immediately.

Data display or speech annotation can be distributed. The processing power of modern PCs is sufficient for computationally expensive signal processing tasks, and so even they can be distributed. Hence a client-server architecture, where a central server is accessed by distributed clients, is well-suited for educational environments.

Note that no tool can eliminate the need for the person to person relationship in education.

2 WWW Client-Server Architecture

The WWW is a client-server system. A WWW client (a browser, e.g. Netscape Navigator, Internet Explorer, Opera) sends a URL (Uniform Resource Locator) to a WWW server via a TCP/IP network, e.g. the Internet. It receives HTML formatted documents which it then displays, or data which is output either by the browser or external helper applications (Fig. 1).

In this client-server architecture, the server implements the data management tasks and computations, whereas the client mainly displays the data.

2.1 Client-side computations

To reduce the traffic between the server and the client it is desirable to execute suitable tasks on the client, e.g. checks for formal consistency to ensure that only data that is formally correct is transferred to the server. The program code for such tasks is embedded in the HTML pages and is downloaded together with them.

Currently, there are two approaches to embedding code within pages: scripts and applets. Scripts are written in an interpreted language, e.g. JavaScript or VBScript. compiled to a machine independent byte-code. This byte-code is then executed in a run-time environment in the browser on the client machine.

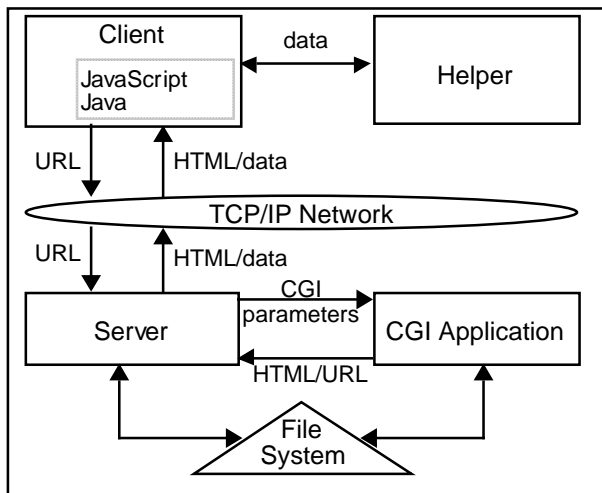


Figure 1 WWW client-server architecture

Scripts are limited to the interface elements already found in a browser window, e.g. buttons, text fields, images, and links. Applets are much more powerful because Java provides a full range of standard libraries including the AWT (abstract window toolkit), a graphics library. The AWT contains layout managers for the display organization, canvasses for drawing, and panels for interactive elements such as buttons, pop-up menus, text fields, and text areas.

2.2 Security

Any executable code downloaded from the Internet is a potential threat to the local machine if it is allowed to spawn further processes, spy on other applications running on the host, or read from or write to the local file system.

Java implements a number of security features: it does not allow direct access to memory, e.g. via pointers. The run-time environment verifies that the code does not contain illegal instructions. In general, applets may not read from or write to the local file system. Communication is restricted to the server from which they originated. Finally, applets can be given a digital signature so that any modification to the applet can be detected before it is run. (Signed applets are considered trusted code, and hence they can be granted access privileges on client machines.)

These strict security features make Java the natural choice for writing applets.

3 WWWTranscribe

At Eurospeech '97 WWWTranscribe, a web-based extensible toolbox for the management and processing of speech corpora was presented [3]. Its primary features are platform independence and modularity. The system has become one of the standard tools for database annotation at the BAS (Bavarian Archive for Speech Signals) [7], e.g. dialect identification [5], [1], annotation of RVG-1 recordings [2], validation of speech in noisy environments, etc.

At the Phonetics department of the University of Munich, WWWTranscribe is used in phonetics lectures, for annotation training, and for the actual annotation work in data collection projects, e.g. SpeechDat [8].

4 WWWSigTranscribe

In WWWSigTranscribe, the WWW server is responsible for the management of the speech and annotation files, it monitors the annotation tasks performed on the clients, and it provides access to a common shared lexicon.

Annotation on a client machine begins with user login and the selection of a speech signal. The server fetches from a database the prompt text corresponding to this signal, generates the HTML code for the transcription window and transfers it to the client (Fig. 2).

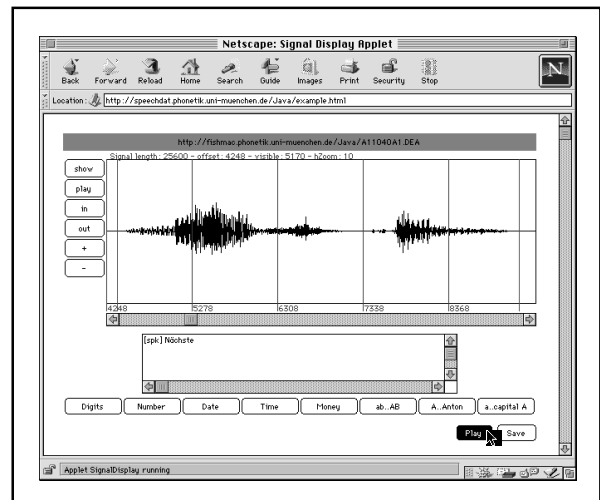


Figure 2 WWWSigTranscribe window

The annotation module WWWSigTranscribe is a dynamically generated HTML form containing an applet. This applet consists of a signal display window with buttons to modify the signal display (show the selected signal fragment, zoom in and out, increase or decrease vertical zoom), text areas for annotation and comments, a series of editing buttons, and buttons to play the signal and to save the annotation.

Initially, the signal is displayed in its display window and the annotation text area contains the text from the prompt sheet. The annotator then clicks on the *Play* button to start the audio signal output. Then he or she edits the annotation field either manually, via keyboard shortcuts, or via the editing buttons (cf section 4.3.1).

4.1 Consistency checking

A click on the *Save* button starts the consistency checker for the annotation text. A tokenizer analyzes the lexical structure of the annotation items, e.g. phonetic labels, orthographic words, or marker symbols. A parser then checks the syntax of the annotation text. Errors are reported to the user via an alert dialog, if possible with the cause of the error and a suggested remedy.

4.2 Dictionary lookup

If an annotation text has been parsed successfully, a lexicon lookup for the transcription items is performed. If an item cannot be found in the lexicon, it can either be inserted as a new entry, or be rejected.

For training, any lexicon lookup is reported immediately so that the user can either correct the annotation text, or confirm the insertion of the new entry into the lexicon. In large-scale annotation projects, lexicon lookup errors are not reported to the annotators so that their workflow is not interrupted. Instead, the erroneous item is forwarded to a lexicon expert for further treatment.

4.3 Implementation issues

WWWSigTranscribe is a Java applet embedded into a browser window and distributed via the WWW. Applet implementations differ from traditional standalone applications. Some of the implementation issues are discussed in more detail in this section.

4.3.1 Hierarchy of Display and Editing Panels

WWWSigTranscribe consists of nested panels, i.e. screen regions with interactive elements, and canvasses for drawing. These panels and canvasses are organized by the standard AWT layout managers which render the applet in the different browser environments.

Each panel or canvas corresponds to an object class. It is thus straightforward to modify the appearance or functionality of the applet or of subcomponents by replacing individual classes.

The top-level layout nesting hierarchy is given in (Fig. 3). `AppletControlPanel` contains the interface elements relevant to the applet as a whole, e.g. a *Play* button to load and play the speech signal (this button does not yet work in the current version – cf. section 4.4.2).

`SignalDisplay` consists of the `SignalDisplayPanel` and the `AnnotationPanel`. The `SignalDisplayPanel` contains the `SignalDisplayControlPanel` and the `SignalDisplayWindow`. The `SignalDisplayControlPanel` contains the interface elements relevant to the signal display window, i.e. buttons to show the current selection, and horizontal or vertical zoom in and out. The `SignalDisplayWindow` consists of a canvas on which the oscillogram curve is drawn, and a horizontal scrollbar. Interaction with this display is limited to setting the left and right boundaries of the speech signal by mouse click, and scrolling through the signal.

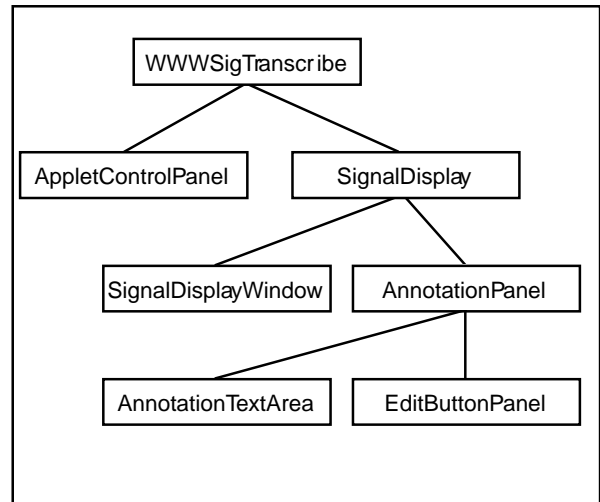


Figure 3 WWWSigTranscribe Panel Hierarchy

`AnnotationPanel` is divided into the `AnnotationTextArea` and the `EditButtonPanel`. The buttons in `EditButtonPanel` update the annotation text held in the text area, e.g. change digits to text strings in orthographic annotations. `AnnotationTextArea` contains two text areas, one for the current annotation text, the other for comments; it is enhanced by the handling of special keyboard events such as keyboard shortcuts (cf. section 4.4.1).

4.3.2 Consistency checking

Different annotation formalisms require different consistency checkers. For SpeechDat-type orthographic annotations, or SAM-PA [9] phonemic transcriptions, a finite state automaton is sufficient, while other annotation formalisms, e.g. based on XML [10], may need the expressive power of context-free grammars. The implementation of the consistency checker is based on Java *interfaces*.

4.3.3 Dictionary lookup

Dictionary lookup can be performed either on the client or the server. For client-side dictionary lookup the applet must either be granted access to the local file system, or the dictionary must be held within the applet. Access to the file system should not generally be granted, and including the dictionary in the applet dramatically increases its size and thus the amount of data to transfer.

In WWWSigTranscribe dictionary lookup is performed on the server. It is relatively easy to maintain a single shared dictionary in one location, and any updates are available to all clients immediately, avoiding inconsistencies. Furthermore, dictionary lookup creates only little network traffic and can be performed when saving the annotation.

4.3.4 Client-server communication

Saving an annotation text, retrieving a speech signal, and dictionary lookup all require data transfer between the client and the server. Both the applet and the

surrounding browser can communicate with the server. If the applet communicates with the server, then it must be capable of interpreting the data returned by the server. On the one hand this allows the implementation of a highly efficient data exchange mechanism, but on the other hand such a mechanism has to be implemented manually. If the browser communicates with the server, then the built-in capabilities of the browser can be exploited, and standard interfaces can be used, e.g. the cgi (Common Gateway Interface).

In WWWSigTranscribe the applet is in fact embedded into an HTML form. The *Save* button is not part of the applet, but of the HTML form. Clicking on this button means submitting the contents of its fields to the server. In the current implementation, the server then returns a new HTML form with a new copy of the applet.

4.4 Current restrictions

WWWSigTranscribe originally was implemented in Java 1.0 and has been ported to Java 1.1. The major changes needed concern the handling of events – the event model of Java 1.1 differs considerably from that of version 1.0.

4.4.1 Keyboard shortcuts

An important problem for the implementation of keyboard shortcuts is that they are dependent on the client's operating system. Keyboard shortcuts often have a special meaning in the windowing environment on the client machine, and they are thus intercepted and interpreted before reaching the applet.

4.4.2 Audio output

Audio output in Java 1.1 is still rudimentary. The only built-in class is `audioClip` which can access only .au formatted signal data, and supports only playing entire signals – output of signal fragments is not yet possible. In Java 1.2 these restrictions are lifted, but Java 1.2 is not yet available for all platforms.

As a temporary solution, there is a second *Play* button in the HTML form. Pressing this button loads the audio file and passes it on to a helper application for output. Clearly, this is inefficient: the signal has to be transferred once for display, and then again for audio output.

4.4.3 Maximum signal length

In the current version of WWWSigTranscribe the signal data is stored in an array of type `short`. In Java, arrays must have a fixed length, and so the size of the signal to be displayed is limited by the length of the array. This restriction can be overcome by storing the signal in a `Vector` which is a built-in Java class.

5 Conclusions and Outlook

WWWSigTranscribe is a first step towards a toolbox of Java applets for annotation and segmentation via the WWW. The software is currently employed in its second, Java 1.1, version. A new version is currently

implemented for SpeechDat-Car multi-channel data; here, the signal display features synchronous tracks for the five recording channels.

Java is sufficiently fast on modern workstations (PowerMacintosh, LINUX or Windows PCs), and although no effort has gone into optimizing WWWSigTranscribe for speed, it is fast enough to perform annotations comfortably.

Finally, Java supports UniCode, a two-byte character encoding for most writing systems, including the IPA characters and Japanese and Chinese character sets. Hence, in principle any language can be transcribed in its appropriate alphabet with WWWSigTranscribe. Due to its object-oriented design, extending WWWSigTranscribe to handle different annotation systems is straightforward.

References

- [1] Burger, S., Draxler, Chr. (1998) Identifying Dialects of German from Digit Strings, *1st Int'l Conference on Language Resources*, Granada
- [2] Burger, S., Schiel, F. (1998) RVG-1 – A Database of Regional Variants of Contemporary German, *1st Int'l Conference on Language Resources*, Granada
- [3] Draxler, Chr. (1997) WWWTranscribe - A Modular Transcription System based on the WWW, *Eurospeech 97*, Rhodes
- [4] Draxler, Chr. (1998) WWWSigTranscribe – A Java Extension of the WWWTranscribe Toolbox, *1st Int'l Conference on Language Resources*, Granada
- [5] Draxler, Chr., Burger, S. (1997) Extracting Region Information from Digit Strings in Telephone Speech, *Eurospeech 97*, Rhodes
- [6] Flanagan, D. (1997) *Java in a Nutshell*, O'Reilly, Cambridge, MA
- [7] Schiel, F., Draxler, Ch. & Tillmann, H. G. (1997): The Bavarian Archive for Speech Signals: Resources for the Speech Community; in: *Proceedings of the EUROSPEECH 1997*, Rhodos
- [8] Höge, H., Tropsch, H., Winski, R., van den Heuvel, H., Häb-Umbach, R., Choukri, K. (1997) European Speech Databases for Telephone Applications, *ICASSP 97*, Munich
- [9] Wells, J. (1998) Standards, Assessment, and Methods: Phonetic Alphabets, <http://phon.ucl.ac.uk/home/sampa/home.htm>
- [10] W3 consortium, eds: T. Bray, J. Paoli, C. M. Sperberg-McQueen (1998) Extensible Markup Language (XML) 1.0, <http://www.w3.org/TR/REC-xml>