# Extending Audio Notetaker to Browse WebASR Transcriptions

*Roger Tucker*[1], *Dan Fry*[1], *Vincent Wan*[2†], *Stuart Wrigley*[2], *Thomas Hain*[2]

[1] Sonocent Ltd, Chepstow, UK
[2] Department of Computer Science, University of Sheffield, UK
roger,dan@sonocent.com; t.hain,s.wrigley@dcs.shef.ac.uk

## Abstract

The audio annotation tool Audio Notetaker has been extended to allow browsing of transcripts produced with the WebASR system from Sheffield University. The interface has been designed to be usable with as much as 50% recognition error.

**Index Terms**: webASR, transcription, user interface

## 1. Introduction

Audio Notetaker is a software tool for manually annotating audio recordings. It has many thousands of users among whom are qualitative researchers who need to process recorded interviews, but who are inevitably interested in specific parts of the interview. We wanted to see if ASR could help them find and transcribe those useful parts. Although the free-flowing and ambient nature of such recordings makes this a daunting task, we nevertheless wanted to see what could be done with careful UI design and a recognizer optimized for ambient recordings. The recognizer used was a variant of the Sheffield WebASR system [1].

## 2. Audio Notetaker

Audio Notetaker's distinctive feature is that it uses highly sensitive speech detection to visualize spoken phrases in its audio pane, making each phrase both a navigable and editable unit. As well as an audio pane, it has a notes pane and image pane, all of which are displayed alongside each other, but can be individually hidden if not needed. Audio, notes and images can be broken down into common sections, which is the main mechanism in Audio Notetaker for grouping the three kinds of data together. For sections of lesser interest, the notes pane can be used simply to summarize the section's audio; for sections of greater interest full notes or a transcript can be entered.

## 3. WebASR

The webASR service [1] provides web-based access to the AMIDA (http://www.amiproject.org) speech recognition system. There are two distinct modes of access: via a browser or, programmatically, via an HTTP API. Both modes allow: the upload of audio data along with optional segmentation metadata; status checking for currently running ASR jobs; and the retrieval of a transcript annotated with speaker turns. The browser-based interface provides a much richer experience by also allowing full control over existing uploads, ASR processing jobs and transcripts. The webASR service allows the transcript to be downloaded in a number of formats including MLF, PDF and HTML. Integration of the API-based service is facilitated by a wrapper DLL for Microsoft Windows platforms.

The webASR-based speech recognition systems are derived from the AMIDA RT'09 systems [2], but modified firstly to take segmentation input information from Audio Notetaker and secondly to make an additional, parallel, decoding pass using a language model derived from ESDS interview transcriptions (http://www.esds.ac.uk) rather than the AMIDA meeting corpus. The ESDS and AMIDA LMs gave similar error rates on our interviews corpus, but by using both and comparing the two outputs we were able to assign a higher confidence to the 50% of words that appeared in both outputs.

## 4. Initial integration of WebASR

Initially, a fairly simple integration of WebASR was carried out. The DLL was linked in, and an extra menu item was added that allowed the upload of any wav file displayed in the "All Audio Files" file management pane. Audio Notetaker would then poll regularly for the results, and when ready display an icon in the bottom status bar. Clicking the icon would download the results, which would be stored internally but not displayed. We were very reluctant just to display the transcript as a mass of unformatted text with lots of errors as this looks awful compared to the 100% correct formatted text people are used to seeing. Instead the transcript could be viewed by hovering over each segment in the audio pane, and a "speech bubble" would appear, as shown in Fig 1. The webASR speaker segmentation was displayed using colour.

However, while quite effective as a demo, the speech bubble approach was too slow to be effective as a browsing mechanism. It seemed unavoidable that the whole transcription in some form needed to be displayed, and that unlike the notes pane this would need to be linked to the audio pane on a segment-by-segment basis.

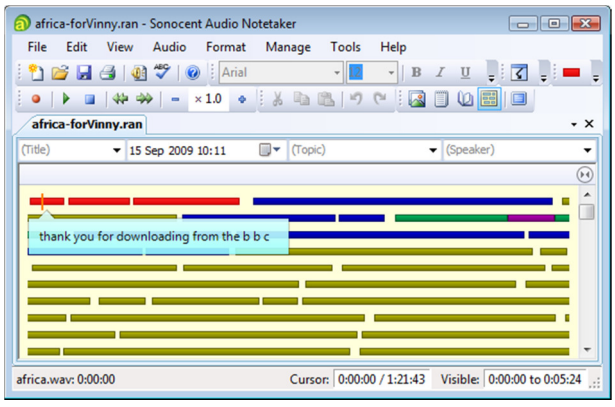


Figure 1: *Speech Bubble interface to transcription.*

† Vincent Wan is now with Toshiba Research Europe Ltd., Cambridge, UK, vincent.wan@crl.toshiba.co.uk

## 5. Transcription pane

To avoid the very negative impact of an unformatted and errorful transcription, we conceived the idea of a multi-level display not unlike that developed by Tucker and Whittaker [3] where less useful words are filtered out and replaced with ellipses. We planned to use some combination of confidence score and inverse frequency score (IFS – frequency in the transcription divided by frequency in the language model), similar to that used in the speech summarization work described in section 3 of [4]. However, the decoder used in webASR doesn't give confidence scores, so we had to rely on a word occurring in both recognition passes to give us a confidence measure. So many words were filtered out by this confidence system that we felt we couldn't apply further IFS processing. Therefore we ended up with a single level of filtering based on confidence only. IFS can easily be added in later if the error rate improves or a more sophisticated confidence scoring system becomes available. A list of user-definable stop words was used to remove any isolated stop words remaining after the filtering process.

We built a fourth pane - transcription pane - to display the two-level transcript. The text in the pane is segmented in an identical way to the audio pane using a special symbol in between the textual phrases. Clicking in the text moves the cursor in the audio pane to the equivalent segment, and clicking in the audio pane causes the equivalent text to be highlighted in yellow. This linking works very well indeed, with these benefits:

- Can choose whether to use the audio or transcription pane to navigate, depending on how useful the transcription is.
- Can easily see speaker turns in the audio pane, so the transcription pane can show just the transcript itself.
- When the transcription pane is in "truncated" mode (i.e. limited to the same length as the audio pane) it is easy to navigate to hidden parts of it by clicking in the audio.

The two levels of filtering are shown in the screenshots below. At first, in the unfiltered transcript, we displayed both recognizer pass outputs where they differed, as well as the audio pane speech bubble, as shown in Figure 2. These only added to the confusion and in the final version we displayed only one of the recognition passes and no speech bubble.

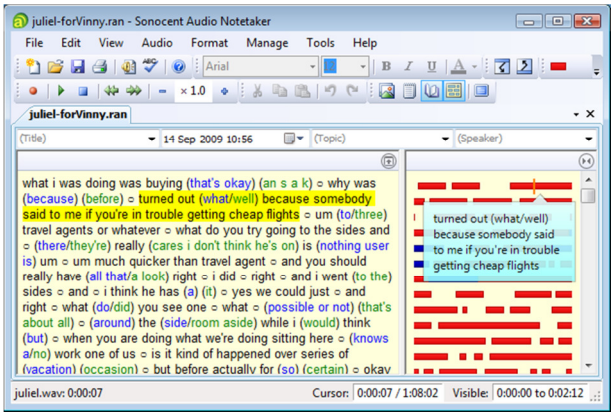


Figure 2: *Interview transcript with no filtering.*

## 6. Correction of transcript

Since parts of the audio will invariably require transcription, we created a correction UI which minimizes the number of keystrokes needed, and always requires fewer strokes than typing in from scratch, no matter how bad the error rate. Phrases are played back one at a time and a series of keyboard shortcuts allow:

- Replacement of the entire phrase by just typing.
- Skipping between words.
- Deleting the word ahead.
- Skipping straight to the end of the phrase.
- Capitalizing the first letter in the phrase and putting a full-stop at the end of the previous phrase.

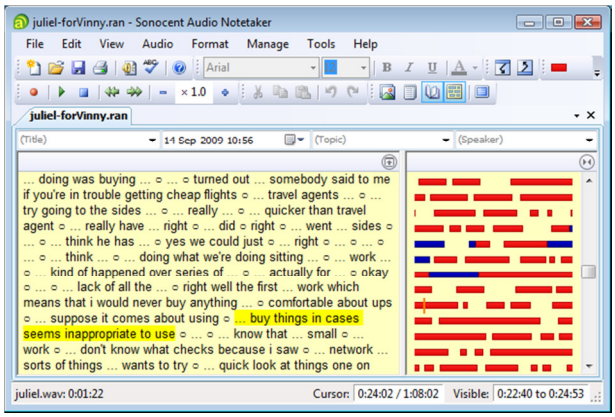


Figure 3: *Interview transcript with filtering.*

## 7. Evaluation

The two levels of transcription were informally evaluated by a qualitative researcher on four of their interviews lasting about 20 minutes each. They reported that they could definitely use the filtered transcription for navigating around their interviews, but perceived the unfiltered transcript as too cluttered with too many wrong words – even though its error rate was just 49% compared to the filtered error rate of 42%. They thought the 50% of words in the filtered output was enough to give a clear reminder of the various parts of the interviews. A more formal test might be useful to determine how the recognition rate and the amount of filtering affect browsing - though the work of [3] shows how hard it is to get concrete results from such tests. However, the fact that the filtered transcription is useful at all is a significant result, given the high error rates.

Our own assessment is that it is phrases rather than words which make the browsing process possible - especially as the recognizer may well miss the really unusual anchor words. Phrases also are easier to spot as erroneous than single words. More thought on the filtering process is needed and whilst it is not obvious that using IFS would help, some kind of extra processing definitely would, if nothing else but to give a more consistent effect for different recognition rates.

## 8. References


[1] Hain, T. et al, "Automatic speech recognition for scientific purposes – webASR", Interspeech'08, 504-507, 2008.

[2] Hain, T. et al, "The AMIDA 2009 Meeting Transcription System" Interspeech'10, 358—361, 2010.

[3] Tucker, S. and Whittaker, S., "Have a say over what you see: evaluating interactive compression techniques", IUI '09: Proceedings of the 13th international conference on Intelligent user interfaces, 37-46, 2009.

[4] Tucker, R. C. F., Hickey, M., Haddock, N., "Speech-as-Data Technologies for Personal Information Devices", Personal & Ubiquitous Computing, vol 7 no. 1, May 2003, 22- 29, 2003.