Hearsay: A New Generation Context-Driven Multi-Modal Assistive Web Browser

Y. Borodin, F. Ahmed, M.A. Islam, Y. Puzis, V. Melnyk, S. Feng, I.V. Ramakrishnan, G. Dausch
Department of Computer Science, Stony Brook University, Stony Brook, NY 11794
{borodin, faiahmed, maislam, ypuzis, vmelnyk, songfeng, ram}@cs.sunysb.edu, glenn.dausch@sunysb.edu

ABSTRACT
This demo will present HearSay, a multi-modal non-visual web browser, which aims to bridge the growing Web Accessibility divide between individuals with visual impairments and their sighted counterparts, and to facilitate full participation of blind individuals in the growing Web-based society.

Categories and Subject Descriptors
H.5.2 [**Information Interfaces and Presentation**]: User Interfaces; H.5.4 [**Information Interfaces and Presentation**]: Hypertext/Hypermedia – *architectures, navigation, user issues*

General Terms
Design, Human Factors

Keywords
Screen Reader, Audio Interface, Multi-Modal, Assistive Browser, Blind Users, Web Accessibility.

1. INTRODUCTION
Interacting with the Web has never been easy for people with vision impairments. The dominant assistive tools, screen-readers, compel their users to process information sequentially through simplistic serial interfaces causing information overload; and to make things worse, web-based rich interactive applications are further widening the Web Accessibility divide between the ways blind and sighted people browse the Web.

This demo will present HearSay, a multi-modal non-visual web browser, envisioned as the next-generation assistive technology that will alleviate many Web Accessibility problems. Two years in the making, Hearsay is now a well-tested working system. On the algorithmic side, it embodies a number of robust and scalable techniques based on Information Retrieval and Machine Learning, including: content analysis that partitions web pages into meaningful sections for ease of navigation; context-directed browsing that exploits the content surrounding a link to find relevant information as users move from page to page; detection of actionable objects that help users quickly perform web transactions; detection and handling of changes in web pages, helping users stay focused; statistical models for associating labels with web elements even if they are missing, as in images without alternative text; personalized speech-enabled macros for automating repetitive tasks; statistical language detection, etc. All of the above techniques combined go a long way towards bridging the Web Accessibility divide.

On the interface side, HearSay supports multiple output (audio, visual, Braille) and input (speech, keyboard, touch, phone keypad) modalities. HearSay, the power and usability of which have been demonstrated in a series of user studies with blind subjects, can be used as a desktop application, or can be used remotely via the plain phone. The browser also supports the IBM's Social Accessibility network that brings together end-users and supporters who can collaboratively create and utilize the accessibility metadata.

It is worth mentioning that the HearSay system is an amalgamation of many ideas for non-visual web interaction, including our own prior work on context-directed browsing [3], process models for facilitating accessible web transactions [1], and techniques for making dynamic updates accessible [2]. The demo will present an integrated multi-modal HearSay assistive web browser that embodies the aforementioned ideas.

2. HEARSAY SYSTEM
Figure 1 shows the architecture of the HearSay multi-modal web browser. The 11 inner modules form a customizable pipeline that analyzes and transforms webpage content before presenting the information to the users. The 5 icons at the top of Figure 1 represent various input and output modalities supported by the HearSay browser. Finally, at the bottom of Figure 1, one can see external modules that can be connected to HearSay. We next provide a more detailed bottom-up overview of the HearSay architecture.

Browser Plug-in: Browser Plug-in is a web browser extension that provides seamless integration of all HearSay features with the browser. HearSay currently supports Firefox and Internet Explorer browsers. The plug-in listens on browser and user events (*e.g.*, key presses, tab switches, focus changes, page loads, etc.) and sends them all to the HearSay Application. When the plug-in detects a page-load, it extracts the source of the newly loaded HTML page and enriches it with geometric and style information inferred by the web browser's rendering algorithm. The Content Dispatcher establishes connections with and passes information to and from the Browser plug-in.

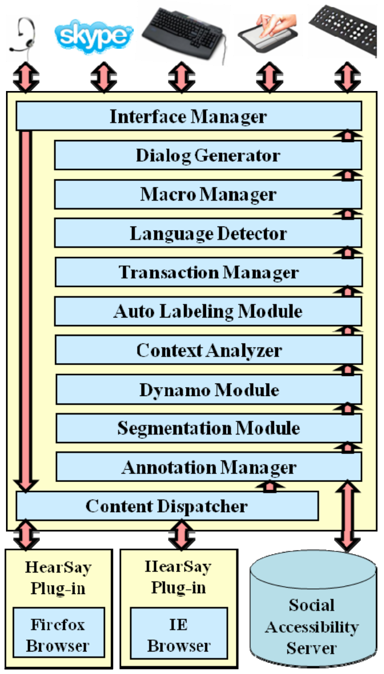


Figure 1 – HearSay Architecture

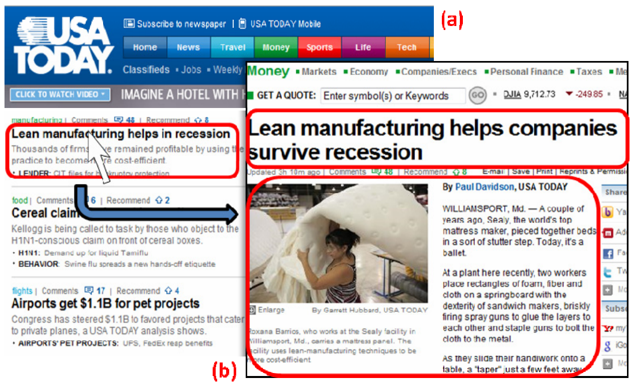
Content Dispatcher: As soon as the HearSay Application is started, the Content Dispatcher module opens a socket where it listens on incoming connections, spawns a new thread for every new connection, and maintains the thread for the duration of the session. The session ID number is used as the identifier for the browser window or is tab connected to HearSay. The Browser Plug-ins send HearSay the events captured in the browser, while HearSay, in its turn, sends the plug-ins commands that control the behavior of the browser. The separation of the main HearSay codebase from the browser implementations carries a number of benefits. Web browsers have grown to be complex applications running JavaScript and Flash, handling cookies, maintaining history, and blocking pop-ups; therefore, outsourcing browser functionality to the established browsers helps the HearSay Application to stay focused on making web content accessible.

Annotation Manager: HearSay users can provide alternative text for images and other content. The Annotation Manager keeps track of user annotations (accessibility metadata) of the web content and interacts with the accessibility metadata server which stores the annotations. The HearSay project has initially maintained its own accessibility database; however, it has recently migrated to the IBM's Social Accessibility [5] Server. Social Accessibility (SA) is a collaborative framework that uses the power of the open community to improve accessibility of the existing web content.

Segmentation Module: The Segmentation Module analyzes the structure and geometric layout of web pages and tries to segment them in a way that will enable blind users to navigate between "meaningful" pieces of information that are "semantically" related (*e.g.*, the news headline and article summary, menu, etc.) HearSay currently implements a variation of the Microsoft's Vision Based Page Segmentation Algorithm (VIPS) [4], which is a top-down algorithm that analyzes the page layout and identifies blocks of information. VIPS tries to find horizontal and vertical separators that are used to reconstruct the semantic structure of the page.

Dynamo Module: The main function of the Dynamo Module is to handle dynamic and static changes in web pages and provide an accessible interface for reviewing the changes. The Dynamo module uses a custom Diff algorithm [2] to compare the web pages visited by the user and mark up all information that changes in the pages. This functionality allows users to focus on the changes and easily skip repeated information, which often includes the links and menus that are present on every page of a web site. This also helps review the differences in web pages that periodically refresh to update their content.

Context Analyzer: The Context Analyzer module plays the central role in identifying and ranking relevant information in web pages. The algorithm [3] analyzes the context of user actions to infer what web content is relevant for the user. The results of the algorithm can be used to find the beginning of main content and identify what information is relevant to the user in web pages (Figure 2). The module collects the text context around the link that was clicked by the user, and then matches it against the destination page. The algorithm tries to find the combination of the best match and the font size to find the starting position for the main content of the page. It also tries to identify all information that has some relevance with respect to the context, which can be reviewed by the user at any time by opening the layer of relevant content.



(a)
(b)
USA TODAY
Lean manufacturing helps in recession
Cereal claim
Airports get $1.1B for pet projects
Money
Lean manufacturing helps companies survive recession

Figure 2 – Finding the Article on USA Today News Web Site

Auto-Labeling Module: The Auto-Labeling Module helps associate labels with web elements (*e.g.*, form elements). Whenever available, the module uses the labels provided by web designers (*e.g.*, *<label for="textboxID">search</label>*); otherwise it uses a probabilistic method for making associations between labels and form elements. The approach described in [6] proposes to use an ensemble of classifiers for the form labeling problem. We go beyond the described work in that we use the context of form elements (and more generally web elements) to identify their labels, even when the labels are missing. We underscore the importance of correct label association for screen-reader users because they cannot fill out unlabeled web-forms without assistance from someone who can see the labels. Needless to say, getting the labels right would mean a substantial increase in the productivity and independence of screen-reader users.

Transaction Manager: The Transaction Manager uses IR-based algorithms for improving accessibility of actionable objects used in web transactions (*e.g.*, 'add to cart', 'check out', 'sign in'). The module collects objects' captions and context (text around them), encodes the collected information as vectors in a vector space model, and uses IR-based methods to determine the concepts that correspond to the given actionable objects. One of the novel aspects of our algorithm is the use of context of transaction-centric objects to identify and classify them, in some cases, even when their captions (*e.g.*, 'add to cart', 'sign in') are missing. Another appealing aspect of the algorithm is its adaptability to emerging concepts resulting from evolution of web services that create new domains of web transactions.

Language Detector: The Language Detector module analyzes web pages to determine the language of the entire page, as well as that of the individual elements. Many websites specify the language with the XHTML *lang* attribute. When this attribute is included, HearSay uses it to switch between the supported languages. In cases when the language is not specified with the *lang* attribute, the module uses statistical analysis to determine the most probable language for the passage. In pages that mix multiple languages, such as wikipedia.org, the system seamlessly switches between languages as it reads the content.

Macro Manager: The Macro Manager allows screen-reader users to automate repetitive browsing tasks performed on a regular basis. For example, checking email, weather or news, paying bills, and shopping – all these could be accomplished more efficiently and faster if users had a way to record and replay their transactions in a flexible way. HearSay users can initiate a macro recording by pressing a shortcut or saying "*Record*", at which point, the browser starts tracking user actions. If the user starts by loading a new page, the macro recording will be page-specific. Conversely, starting from an already opened web page will make the recording page-independent. Once the recording is saved, it can be replayed with the macro player by either entering or speaking the name of the recording, with the player interpreting the macro and simulating the recorded actions at a later stage.

Dialog Generator: The Dialog Generator module converts web page content into interactive dialogs that will be played to the user. Using a collection of VoiceXML dialog templates that are designed to provide standard screen-reading functionalities, as well as the advanced interface features, the module converts any HTML source into an array of spoken elements. By injecting the new content in the executing dialogs at run-time, similar to the way AJAX updates web pages, this dynamic approach to dialog generation allows HearSay to unify the treatment of page-loads and dynamic updates.

Interface Manager: Users interact with HearSay through the Interface Manager, which is a modular multi-platform (Windows, OsX, Linux), extensible VXML interpreter, called vxmlSurfer [7]. The module interprets the VoiceXML dialogs that implement the logic for voicing web content. For every window or tab opened in the target web browser, HearSay instantiates a separate dialog thread in the vxmlSurfer. Switching between browser tabs dynamically switches the dialog threads. This approach helps preserve the dialog context in every tab so that, when the user switches back to an already visited tab, s/he can find it in the same state it was left. The Interface Manager supports a variety of inputs, including keyboard shortcuts, voice commands, gestures (single-finger gestures which are analogous to those supported by Apple's VoiceOver), and phone keypads. The module supports multiple output modalities, including audio, visual, and Braille displays. The interface visually highlights the currently voiced element – a feature useful for people with low vision.

TeleWeb: In addition, HearSay supports remote web browsing via the most ubiquitous communication device, the Phone. TeleWeb integrates a simple and usable phone interface with the intelligent features such as context-directed browsing, template-detection, and macro-replaying. When users call TeleWeb, they first access the main menu, in which they can choose to begin browsing, enter a URL, read TeleWeb help, and access lists of bookmarks, macros, headings, and links. The menu items can be selected by pressing phone keys or by saying the corresponding commands. TeleWeb supports a number of context-sensitive voice commands and keypad shortcuts.

In sum, HearSay is a significant advance over the state-of-the-art screen readers such as Jaws and Window Eyes. HearSay also supports single-touch gestures, similar to those introduced in the recently released VoiceOver screen reader for MacOS. In contrast to VoiceOver, HearSay identifies and reads out "semantic" segments consisting of elements related to the same topic, thereby bringing a higher degree of comprehension. Above all, HearSay's context-based browsing, accessible dynamic updates, macros, and remote access with phones go far beyond what contemporary screen readers have to offer.

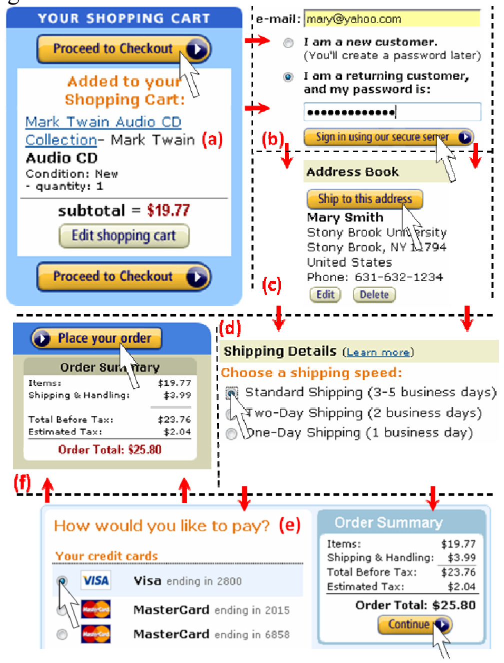
3. DEMONSTRATION SCENARIOS

Using the following three use scenarios we will demonstrate all the novel features of HearSay.

Scenario One: Manuel, who is a visually impaired person and a screen-reader user, starts off his typical day by reading USA-Today on the Web. A common problem he used to face when browsing with a regular screen reader was that it would start reading every web page from the top. Then, he would have to repeatedly press keyboard shortcuts to skip irrelevant content to find the actual news article. Now Manuel browses more efficiently with HearSay.

Upon clicking the news heading (indicated by the mouse cursor in Figure 2-a), Hearsay uses the context of the heading (enclosed by the red box) to find the main section (enclosed by the red box in Figure 2-b) and starts reading from the beginning of the news article, skipping all of the irrelevant content in one go (made possible by the *Context Analyzer*). Manuel can now browse with keyboard shortcuts, voice, or touch. Reading news with HearSay has become a pleasure, and that alone was a good-enough reason for Manuel to switch to HearSay permanently for all web browsing tasks. Within a few weeks, Manuel has become comfortable with all HearSay features.

Manuel, who is bilingual, also frequents Spanish news sites. Taking a break from the U.S. news, he opens a new browser tab and loads his local Spanish newspaper. The *Language Detector* component recognizes Spanish (in spite of the missing language tags) and automatically switches HearSay to Manuel's favorite Spanish voice. While dragging his finger across the touchpad and listening to fragments of content, Manuel stumbles across an article about his favorite Bachata group, Aventura, visiting town with a concert. He pauses there and does a "*pinch-out*" gesture (akin to the iPhone's zoom-in gesture). HearSay responds by reading out the segment enclosing the content related to the topic (identified by the *Segmentation Module*). With a regular screen-reader, Manuel would have to keep pressing keyboard shortcuts all through the browsing session – and now it is a snap. After listening to the news in Spanish, Manuel switches back to the previous tab, and HearSay continues reading the USAToday in English.


YOUR SHOPPING CART
Proceed to Checkout
Added to your Shopping Cart:
Mark Twain Audio CD Collection- Mark Twain (a)
Audio CD
Condition: New
- quantity: 1
subtotal = $19.77
Edit shopping cart
Proceed to Checkout

e-mail: mary@yahoo.com
I am a new customer. (You'll create a password later)
I am a returning customer, and my password is:
••••••••••••••
(b) Sign in using our secure server

Address Book
Ship to this address
Mary Smith
Stony Brook University
Stony Brook, NY 11794
United States
Phone: 631-632-1234
(c) Edit Delete

(d)
Place your order
Order Summary
Items: $19.77
Shipping & Handling: $3.99
Total Before Tax: $23.76
Estimated Tax: $2.04
Order Total: $25.80

Shipping Details (Learn more)
Choose a shipping speed:
Standard Shipping (3-5 business days)
Two-Day Shipping (2 business days)
One-Day Shipping (1 business day)

(f)
How would you like to pay? (e)
Your credit cards
VISA Visa ending in 2800
MasterCard ending in 2015
MasterCard ending in 6858

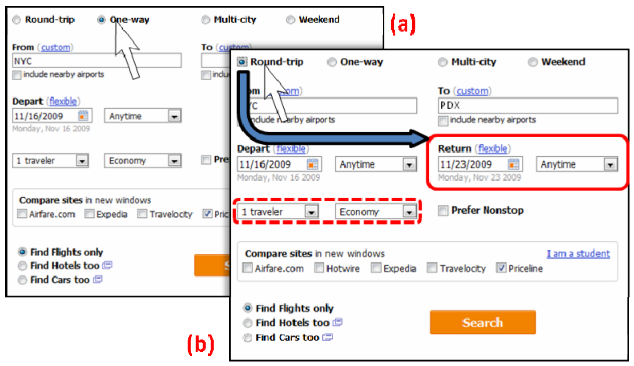
Order Summary
Items: $19.77
Shipping & Handling: $3.99
Total Before Tax: $23.76
Estimated Tax: $2.04
Order Total: $25.80
Continue

Figure 3 – Shopping with Macros on Amazon.com Web

Scenario Two: Mary, who is legally blind, still remembers how tedious web surfing had been with screen-readers. But once she upgraded to HearSay following her friend Manuel's suggestion, doing things online has become much easier. HearSay seems to always take her to the right place of the page and make it very easy for her to go through web transactions such as shopping. In particular, the HearSay's *Transaction Manager* interface allows her to navigate quickly through the items related to the shopping tasks (*e.g.*, searching for a specific item, adding to cart, checking out, etc.).

She realized that shopping can become even easier with macros. Facilitated by the *Macro Manager*, she makes "*checkout*" macros for her favorite shopping websites. While at school, on an impulse, she decides to buy an audio CD from Amazon. Even though she has left her laptop at home, it usually remains on and connected to the Internet. Using the TeleWeb service, she dials into HearSay with her simple cell phone, authenticates herself, and says, "*Open Amazon.*" HearSay recognizes "*Amazon*" as one of Mary's bookmarks, opens the site, and lets Mary navigate it with phone keys and commands.

Once she adds the audio CD to her shopping cart, she just says, "*Play Checkout*," which clues HearSay to find all Mary's "*checkout*" macros and play the one for Amazon. To make sure she does not make any mistakes, Mary runs her macros confirming every action. If she does not want to change any actions, all she has to do is press a key or say "*yes.*" While playing the Amazon "*checkout*" macro, spanning several pages, HearSay clicks on the "*Proceed to Checkout*" button, then enters Mary's e-mail and password and signs in, selects her shipping address, chooses "*shipping speed*", and selects the credit card (Figure 3). Finally, Mary reviews and completes her order.


(a)
Round-trip [✓ One-way] Multi-city Weekend
From (custom): NYC
include nearby airports
Depart (flexible): 11/16/2009 Anytime
Monday, Nov 16 2009
1 traveler Economy
Compare sites in new windows: Airfare.com Expedia Travelocity
[✓ Find Flights only] Find Hotels too Find Cars too

(b)
[✓ Round-trip] One-way Multi-city Weekend
To (custom): PDX
include nearby airports
Depart (flexible): 11/16/2009 Anytime
Monday, Nov 16 2009
Return (flexible): 11/23/2009 Anytime
Monday, Nov 23 2009
1 traveler Economy
Prefer Nonstop
Compare sites in new windows: Airfare.com Hotwire Expedia Travelocity [✓ Priceline]
I am a student
[✓ Find Flights only] Find Hotels too Find Cars too
Search

Figure 4 – Booking a Flight on Kayak.com Web Site

Scenario Three: Bob, who is deaf-blind, is using HearSay to book flights on kayak.com (Figure 4). Because Bob cannot hear, he uses his refreshable Braille display. After filling out the textboxes labeled "*From*" and "*To*" and not finding the "*Return*" date, Bob realizes that he selected "*One-way*" by mistake (Figure 4-a). Instead, he chooses the "*Round-trip*" option, which causes the "*Return*" date textbox to appear on the page (enclosed by a solid red line in Figure 4-b). Thanks to *Dynamo* component, Bob can examine the changes on the page and easily find the "Return" textbox. As he proceeds to fill out the remaining form elements, Bob has no problems determining their roles. Even though several form elements do not have labels (enclosed by dotted red line), the HearSay's *Auto-Labeling* component assigns the labels "Number of Travelers" and "Cabin Class" to these combo-boxes.

4. USER EVALUATION

Over 30 blind users have participated in the HearSay user studies, which were conducted with the purpose of evaluating various features of the HearSay browser. Below, we summarize the qualitative findings and user feedback about the HearSay user interface.

Controlled user studies were conducted to evaluate the Dynamo interface and its ability to help HearSay users cope with dynamic updates, refreshes, and changes in web pages. The evaluation demonstrated that the Dynamo interface has significantly improved subjects' ability to locate dynamic changes in web pages, stay focused on the content in case of page refreshes, and avoid reading repeated information while browsing template-based websites. Human-subjects' experiments were also conducted to evaluate the Context-Directed Browsing approach. The controlled user study has shown that HearSay significantly reduced the time users needed to find main content.

The HearSay user studies have involved the use of the layered interface or simulations of layers. Although some of the blind users found the concept of layers harder to grasp, most of participants enjoyed the easy access to updated content provided through the layered interface. The layered interface has also allowed the participants to access lists of links and headings right on the page, instead of in a separate window. However, screen-reader users would also like to have some options provided by the list view interface, *e.g.*, the ability to alphabetically order the list of items and press any key to access the list elements, starting with the corresponding letter.

Most screen-reader companies have not been able to keep up with the rapidly developing web technologies, and still have basic interfaces that simply read the screen without trying to analyze web content. Therefore, most HearSay study participants welcomed any intelligent features that could save their time in web browsing, provide access to previously inaccessible content, and help them do more than they could with screen-readers. Apart from Dynamo and Context-Directed Browsing, users also praised the ability to create macros, recognize content language, and label form elements.

5. ACKNOWLEDGEMENTS

We would like to thank NSF (Awards: IIS-0534419, IIS-0808678, CNS-0751083) and Sigma Xi (Award: G20071013028487872) for their support of this project, and also Cepstral.com for donating their realistic synthetic voices.

REFERENCES

[1] Sun et al. Model-directed web transactions under constrained modalities. WWW 2006.

[2] Borodin et al. What's New? – Making Web Page Updates Accessible. ASSETS 2008.

[3] Mahmud et al. CSurf: A Context-Driven Non-Visual Web Browser. WWW 2007.

[4] http://www.cs.uiuc.edu/homes/dengcai2/VIPS/VIPS.html

[5] IBM Human Ability and Accessibility Center. http://www-03.ibm.com/able/

[6] Nguyen H., Nguyen T., and Freire J. Learning to Extract Form Labels, In Proceeding of VLDB, 2008.

[7] Borodin, Y. A flexible VXML interpreter for non-visual web access. ASSETS 2006.