

Re-Designing the Hearing Aid User Experience Using the Open Speech Platform

By Tamara Zubatiy

Advisors: Dr.Harinath Garudadri, Dr.Scott Klemmer,
Krishna Vastare

The Honors Program In Cognitive Science
Cognitive Science Department
Univresity of California, San Diego

Table Of Contents

Introduction	2
Needfinding	2
Hearing Loss	2
The Open Speech Platform	4
Reusable Web Page Scripts Summary Table	6
Ideation and Reframing	6
Storyboarding	7
Prototyping and Implementation	8
Researcher Page Webapp	8
Wireframing	9
Implementation	9
NAL-NL2 Prescription Webapp	11
Wireframing	11
Implementation	12
Four-Alternative Forced Choice Task Webapp	13
Wireframing	13
Implementation	14
ABX Webapp	16
Paper Prototyping	16
Implementation	17
Future Work	18
Acknowledgements	19
References	19

Re-Designing the Hearing Aid User Experience

Using the Open Speech Platform

By Tamara Zubatiy

Advisors: Dr.Harinath Garudadri, Dr.Scott Klemmer, Krishna Vastare

Introduction:

Over the last several quarters, I have been working with Dr.Harinath Garudadri, Dr.Scott Klemmer, and the Open Speech Platform team to create several web apps that enable real time communication with a hearing aid using an embedded web server. The focus of this write-up is to reveal the design process methods that were used in the creation of four specific webapps: the researcher page for amplification parameters, the NAL-NL2 prescription app, the Alternative Forced Choice task, and the ABX task.

Needfinding:

Hearing Loss

Hearing loss is a growing problem in America. About 25% of people aged 65-74 and 50% of people 75 and older have disabling hearing loss (National Institute for Deafness and other Communication Disorders). The World Health Organization (WHO) defines disabling hearing loss as “hearing loss greater than 40 decibels (dB) in the better hearing ear in adults and a hearing loss greater than 30 dB in the better hearing ear in children” (WHO). Hearing loss of this severity results in an increase in cognitive load during simple tasks such as listening to two people speaking at the same time. This is the competing speaker problem, and its study has shed light on the complicated problems that hearing loss patients face. As a result of this increased cognitive load in tasks that were once considered simple, many people feel depressed and experience quickened cognitive decline. In fact, “hearing loss is independently associated with incident all-cause dementia”, suggesting that hearing loss is a very serious problem in our culture (Lin et al.) This problem is especially pressing because according to the U.S. Census Bureau, the number of people over 65 is expected to double between 2012 and

2050, revealing that hearing loss will continue to affect the lives of many Americans in the future. (Census Bureau)

Unfortunately, hearing aid technology is not adequate to meet the needs of people with hearing loss, and 80% of adults who could benefit from hearing aids don't use them for various reasons (McCormack and Fortnum). Users in this study reported reasons for abandonment being anything from aesthetics to improper fit to not being worth the money. Commercial hearing aids can be very expensive (thousands) and require the help of an audiologist to fit and program. Dr. Arthur Boothroyd, from the SDSU School of Speech, Language and Hearing sciences gave me the following anecdote when illustrating the failures of commercial hearing aids: one of his patients really likes to hug people. However, when she gets closer to people, the noise profile of sounds around her changes, and the acoustics work differently because she has changed place in space. As a result, whenever she leans in to give somebody a hug, she hears a high pitched ringing from all of the feedback. She came to see Dr. Boothroyd and explained this problem, so he created a setting for her called the "Hug Setting". The idea would be that each time she would go in to hug somebody, she could click a button on her hearing aid and it would adjust the settings to Hug Setting. The catch is that Hug Setting doesn't do that much for amplifying sounds and improving hearing, it just prevents the feedback. There exists the need for multiple independent settings to account for the acoustics in various physical and noise environments. The kicker of the anecdote is that this woman only uses her Hug Setting the majority of the time, because she doesn't want to be awkward and have to press the button in case she does need to hug somebody. This means that the rest of the time she experiences higher cognitive load from not being able to hear well in common situations. As we have learned, this may lead to cognitive decline, but also underscores the humanity of hearing loss patients. Hearing loss patients need a way to have their hearing aid settings adjust to the environment around them, and they need to be able to respond to audiological tests to assess their hearing in the real world around them.

One may blame the current state of the hearing aid abandonment on audiologists and speech scientists, the ones responsible for fitting patients to hearing aids. However, audiologists are just limited in the ways in which they can control hearing aids to begin with. The problem stems from the fact that commercial hearing aid manufacturers do not open all of the features of the hearing aid for manipulation. Additionally, it is not the fault of the audiologist that he/she works in an office or lab, where the noise environment is relatively stable and often optimized. As a result, when the audiologist fits a hearing aid in such an environment, it often doesn't work in any other environment or situation. Even if the audiologist were to simulate different noise environments in the lab, fit each one, and save all the settings to buttons on the hearing aid, the limitations

of the hearing aid would take over. Finally, audiologists typically only get about an hour per patient, which is just not enough time to thoroughly explore the settings they would need for each noise environment they experience. Audiologists need a way to monitor and control settings in real time, in order to test subjects in real noise conditions remotely. They also need to be able to manipulate more parameters than are available in commercial hearing aid software.

Commercial hearing aids have very poor battery life, and it gets worse if you press the buttons more often. Additionally, even the most expensive hearing aids have a maximum of 5 settings, which isn't a realistic representation of the real time environment which changes constantly. To further complicate the situation, even if users do switch through their 5 settings, they often kill the battery in the process, leading to frustration and more cognitive load. The buttons are also very small, which is a problem for aging people who are losing their dexterity.

The Open Speech Platform

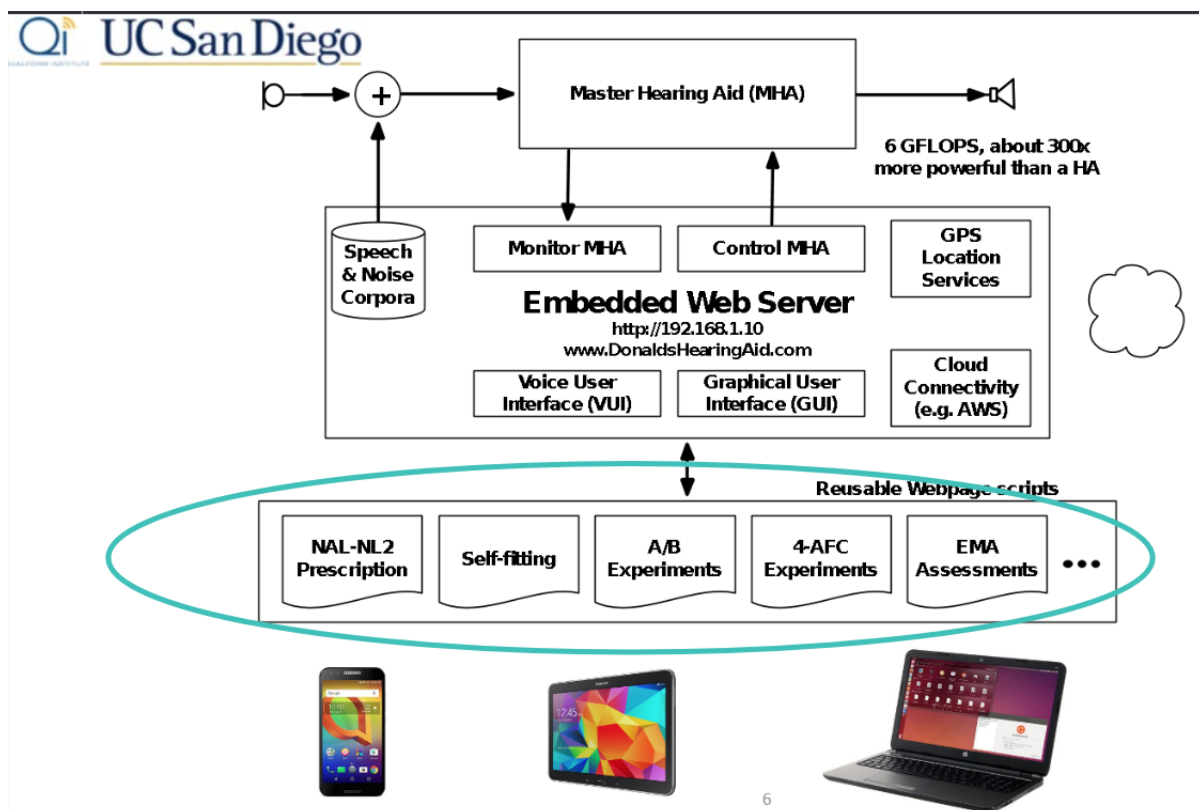
The Open Speech Platform is an open sourced toolkit, developed by Dr. Harinath Garudadri's team, to enable the control and monitoring of a hearing aids' settings in real time from any browser. The platform uses an embedded web server to communicate from apps that control it to the hearing aid itself. It has a custom set of algorithms called the Master Hearing Aid (MHA) that emulate existing hearing aid softwares for amplification, noise cancellation and feedback management. Prior to my entry into the project, the only way to control the hearing aid was with an android tablet. Android is a very inaccessible programming language, so audiologists who had the platform were not able to modify the app for their own research. Thus, the need of the open speech platform was to create tools with a lower barrier to entry in order to enable audiologists to modify them more easily and run their own experiments. Finally, the embedded web server will eventually live on a Snapdragon chip, the hardware inside many Android smartphones. This means that this platform is 300X more powerful than a current commercial hearing aid, and will support all of the interactions necessary for an optimal user experience.

I sent out a Google Form survey to our collaborators at various universities, asking them what sorts of reusable and modifiable web app scripts we should create. I also spent time interviewing our collaborators from Ryerson and SDSU to better understand their unique needs. At Ryerson, they needed a way to fit standard prescriptions, such as the NAL-NL2 prescriptive algorithm, onto the settings of the MHA. They also needed a way to easily run multiple choice tests and collect users' responses after. At SDSU, they

were experimenting with modifying the settings, and needed a very easy way to control the amplification parameters, to transmit a particular set of parameters, and to save sets of parameters as “states”, so that they could be loaded back in. They currently only had the Android control app, and were struggling because it was difficult to modify the software after their Android engineer got another job, and because the current webapp was poorly designed and had many usability issues. (More in Ideation section) Here is a diagram of the components of the Open Speech Platform:

Figure 1: Open Speech Platform Diagram

The Master Hearing (MHA) aid is a set of algorithms that performs the functions of a hearing aid: amplification of sounds, noise management, and feedback cancellation. The embedded web server is used to communicate between the MHA and the user. My contribution is circled in blue: the reusable web app scripts that can be modified to fit the needs of researchers. The purpose and function of each webapp is detailed in the summary table below.



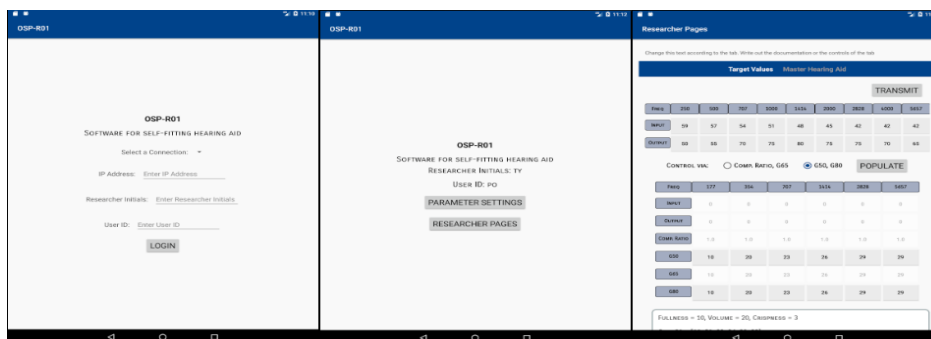
Reusable Web Page Scripts Summary Table:

Webapp	Needs it Fulfills
NAL-NL2 Prescription	Ability to fit hearing aid on the OSP using a canonical algorithm from any browser. Converts prescription into MHA settings. Can be modified and transmitted remotely. Uses HTML, so can be easily modified to track any additional information
A/B Experiments (ABX app)	Ability for user to respond to questions in an A/B format. Suitable for many audiological tests. Sound processing happens through MHA. Enables responses from any browser. Can be deployed remotely and in the user's environment. HTML, easily modifiable.
4-AFC (Four alternative forced choice task)	Ability for users to respond to a common multiple choice question task from any browser. Sound processing happens through MHA. HTML, easily modifiable.
Amplification Parameters Page (not in diagram)	Ability for audiologists to manipulate all available amplification parameters that modify the settings of the MHA from any browser. Ability to transmit settings remotely. Ability to save and load existing states to save time. HTML, easily modifiable.

Ideation and Re-Framing:

Before I started in on creating something new, I needed to see what the current implementation was. Here are some screenshots of the Android interface that was being used before the development of the Webapps. Its major limitations were the high barrier to entry of Android programming as well as the limited freedom in modifying the settings of the hearing aid. Furthermore, the app could only be accessed from an android device, limiting the user base.

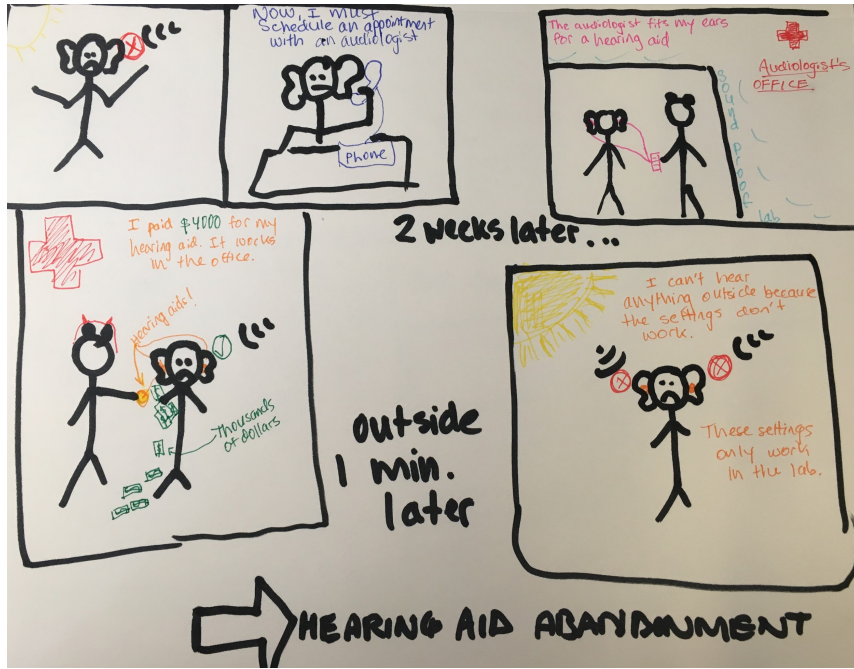
Figure 2: The Android User Interface - The most left screenshot is a login page that records key information about a session using the Android App. The following page



offers little information and is confusing unless the person was already familiar with our app layout. One control screen is shown.

Storyboarding

Figure 3: In this storyboard, you see a hearing loss patient realizing that they can't hear. They



do the normal thing and schedule an appointment with their audiologists, The audiologist conducts a thorough 1 hour exam in a special lab room and prescribes a prescription onto a \$4000 hearing aid. As soon as the patient walks out of the office a minute later, he can't hear again because the settings that were fixed in the quiet room do not generalize to the noisy environment outside. This illustrates the common problem of hearing aid abandonment.

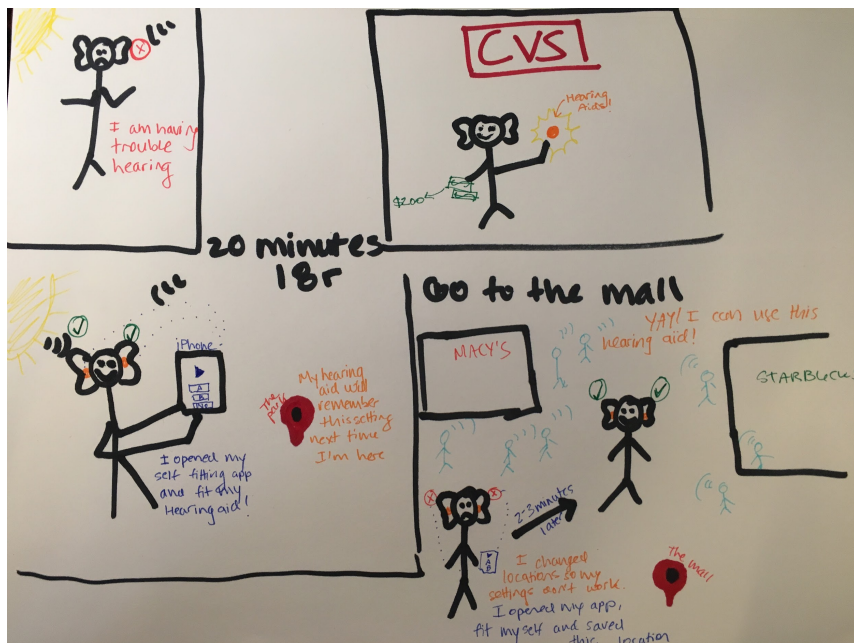


Figure 4: In this storyboard, a hearing loss patients realizes his hearing loss and heads to CVS to buy an over the counter hearing aid. Thanks to the webapps I've designed to work with the OSP, this patient can simply load up a prescription webapp and program his own hearing aid from his location. He saves the settings for this location so they will be loaded in automatically next time (future feature). The patient then goes to the mall, which has a different noise

environment. His settings don't work here, but he uses the app to quickly change and saves this location as well. This demonstrates the power of the OSP to get from setting, hearing loss, to satisfaction, ability for the patient to change his own hearing aid settings remotely.

Prototyping and Implementation

Researcher Page Webapp

I first looked at the existing Android implementation of the researcher page and analyzed what its flaws were.

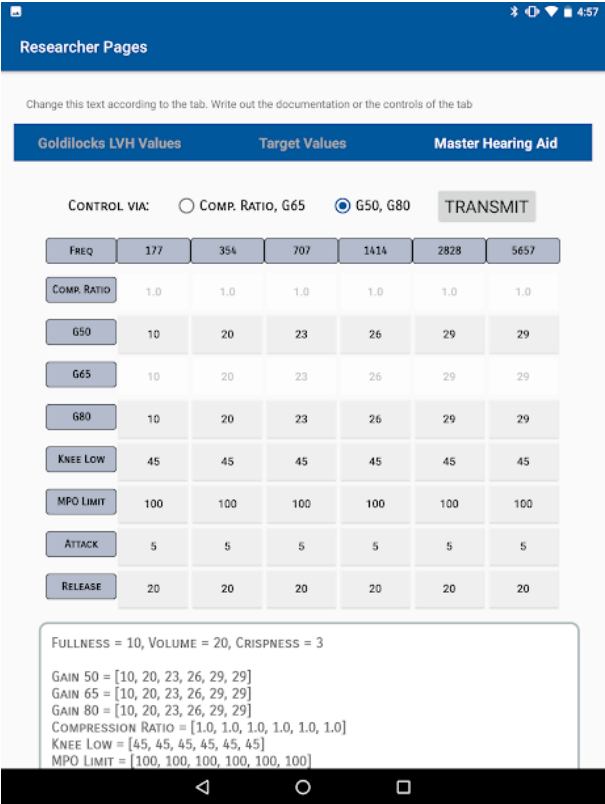


Figure 5: The Android OSP researcher page. All of these are settings in the OSP prescription.

In the current implementation, all of the buttons are very close together on an Android touch screen. Additionally, in order to change a value, one needs to click on the associated box and then use a slider to change the number. This means that you need at least 5 clicks to edit a value. Additionally, one is limited when using this implementation because it is only accessible from an Android device.

Our collaborators reported that they often accidentally touched the wrong box. They also complained about how many clicks it takes to make any edits. They mentioned that it was hard to see which element they were editing at any given time.

Wireframing

Figure 6: Researcher Page Wireframe

Control Via: G50/G80 CompRatio/G65

Parameters						
Frequency	177	354	707	1414	2828	5657
Compression Ratio	90	90	80	80	80	80
G50	90	90	80	70		
G65	90	90	80	70		
G80	90	90	80	70		
Knee Low	90	90	80	70		
Knee High	90	90	80	70		
Attack	90	90	80	70		
Release	90	90	80	70		

Reset Transmit

In the wireframe shown as Figure 6, I outline the key elements that should be present in the final UI, specifically the appropriate settings in a familiar arrangement. We did this intentionally to

minimize the cognitive load of people switching to using the web app from Android. All of the editable fields are clickable and allow users to just enter the appropriate value with a keyboard, minimizing the number of clicks needed to change settings. Additionally, edited squares are highlighted to remind the researcher of what modifications they made. Researchers can instantly press Transmit, and the green highlighting goes away, such that they non-highlighted boxes reflect the current state of the hearing aid. At any point, the researcher can press Reset and return to the previously transmitted state.

Implementation

The images below are screenshots from the debut release of the webapps (early June 2018). Figure 7a demonstrates the power of the usability heuristic of recognition rather than recall, which contributes to lowering the cognitive load experienced by the audiologist when interacting with the webapp. Figure 7b demonstrates the use of the usability heuristic of helping users diagnose and recover from errors. We make it easy for the user to understand that they are not connected, and that the system is working properly.

Figure 7a: Researcher Page UI- This screenshot demonstrates the functionality of signifying which cells have been changed since last transmit. This helps mediate cognitive load when interacting with the interface.

Researcher Page

Frequency	177	354	707	1414	2828	5657
Compression Ratio	0	0	0	0	0	0
G50	0	0	0	0	0	0
G65	0	3	0	0	0	0
G80	0	0	1	2	0	0
Knee Low	0	0	0	0	0	0
Knee High	0	0	0	0	0	0
Attack	0	0	0	0	0	0
Release	0	0	0	0	0	0

Figure 7b: This image demonstrates the signifier that pops up when trying to transmit to a hearing aid when one is not connected.

Researcher Page

Frequency	177	354	707	1414	2828	5657
Compression Ratio	0	0	0	0	0	0
G50	0	0	0	0	0	0
G65	0	3	0	0	0	0
G80	0	0	1	2	0	0
Knee Low	0	0	0	0	0	0
Knee High	0	0	0	0	0	0
Attack	0	0	0	0	0	0
Release	0	0	0	0	0	0

Error! Looks like you're not connected to a hearing aid.

NAL-NL2 Prescription App

Our collaborators at Ryerson expressed the need for an easy to use diagnostic webapp that can be used as a baseline in their experiments. Dr.Garudadri went to National Acoustic Laboratories in Australia over Winter Quarter and they gave us permissioned access to their canonical prescription algorithm, NAL-NL2. This tool allows audiologists to enter certain client parameters (age, gender, tonal language history, audiogram) and generate a prescription in terms of the OSP settings. The webapp then seamlessly enables audiologists to transmit the settings directly to the hearing aid. Although the OSP is open-sourced, we cannot open source the code for this algorithm, so we have included a pre-compiled version in the repository, that cannot be edited. Its functionality is only available to collaborators of the Open Speech Platform.

Wireframing

I started by wireframing the essential components of the user interface, making sure to include the necessary controls for the functionality audiologists needed. Audiologists need to lower their cognitive load, so I wanted to include all of the controls available on one page. There was originally a “calculate” button after the Client’s Audiogram, but I took it out in this wireframe to demonstrate the potential of just automatically populating the table of parameters upon audiogram upload. See figure 8a and 8b below for the NAL-NL2 wireframe.

NALNL2 Webapp

Powered By the Open Speech Platform (OSP) · UCSD

Client's Date of Birth: *

Client's gender: * Female Male

Does the client speak a tonal language?: * No Yes

Has client used hearing aid before?: * Yes No

Client Hearing Aid Type: * CIC ITC ETE BTE

Number of hearing aids: * Unilateral Bilateral

Client's audiogram:*

Control Via:* G50/G80 CompRatio/G65

MHA Parameters

Frequency	177	354	707	1414	2828	5657
Comp Ratio	90	90	80	80	80	80
G50	90	90	80	70		
G65	90	90	80	70		
G80	90	90	80	70		
Knee Low	90	90	80	70		
Knee High	90	90	80	70		
Attack	90	90	80	70		
Release	90	90	80	70		

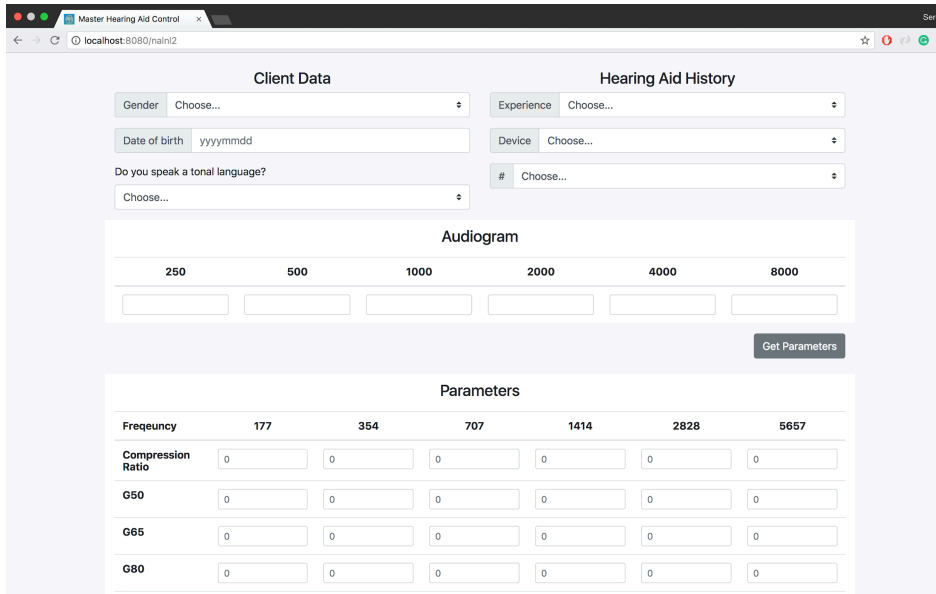
Figure 8a: This is the first half of the wireframe, that is presented in an endless scroll view. It includes the requisite parameter entry fields for the algorithm. Additionally, it offers the possibility to upload an audiogram in an appropriate format.

Figure 8b: This is the second half of the user interface that will be auto populated with the NAL-NL2 prescription in OSP parameters. It enables users to transmit parameters, edit them, and reset them to a previous state. This view demonstrates the consistency between our webapps, as this table looks and feels like the Researcher Page

We discussed these wireframes with our Ryerson colleagues. They told us that it may be more time consuming to upload the file, rather than enter the

audiogram manually. Otherwise, they really liked the experience and said that all of their required functionality was included. The final user interface was coded in HTML, CSS and Javascript. It uses PHP to communicate with the hearing aid by sending the contents of the parameter fields into the OSP. The PHP script sends parameters that are arranged in an intuitive array. In this way, it is very easy to manipulate the UI elements, and to send along parameters to control the actual hearing aids

Implementation



The screenshot shows a web browser window titled "Master Hearing Aid Control" at the URL "localhost:8080/nalnl2". The interface is divided into several sections:

- Client Data:** Includes dropdown menus for "Gender" and "Experience", a text input for "Date of birth" (format: yyymmdd), and a dropdown for "Do you speak a tonal language?".
- Hearing Aid History:** Includes dropdown menus for "Device" and "#".
- Audiogram:** A horizontal axis with frequency markers at 250, 500, 1000, 2000, 4000, and 8000 Hz. Below the axis are six empty input boxes for data entry.
- Parameters:** A table with columns for frequencies: 177, 354, 707, 1414, 2828, and 5657 Hz. Rows include "Compression Ratio", "G50", "G65", and "G80", each with six input boxes.

A "Get Parameters" button is located at the bottom right of the Audiogram section.

Figure 9: Current implementation of NAL-NL2 web app

Pictured in Figure 9 above is the actual reusable web app that we are providing code for in the 2018b release. This app will support the research of our collaborators at SDSU and Ryerson as early as mid June. These researchers will be able to study the efficacy of these generated prescriptions with real patients. They will also inform us of changes that need to be made, so that we can keep iterating together.

This webapp has the most potential for eventually being accessible to individuals. After more research is done, it is possible that hearing aid users of the future will simply be able to go to a drug store, purchase an over the counter hearing aid for hundreds (not thousands) of dollars, and then program the settings of their hearing aids from their smartphones from any location.

Four-Alternative Forced Choice Task (4AFC) Webapp

This is a version of the Alternative Forced Choice Task in which there are four options. It is a commonly used format, and is very easy to adapt to studies with any number of answer choice possibilities. For example, at SDSU, they need a 4AFC but at Ryerson, they are looking for maybe 6 or 9AFC eventually. Additionally, these institutions need the sound files to be going through the OSP, so that they undergo realtime audio processing, just like in a hearing aid. As such, I worked with Krishna to create the API that is being passed along to the OSP for this processing. SDSU and Ryerson both needed ways of getting the score, knowing the word that was selected for each question, and being able to download a task history that includes all of the options in each question, the correct option, and the selected option, shown in Figure 15. They also need to know who was taking the test (Participant ID) and who was administering it (Tester ID), so we added a login page shown in Figure 13. Finally, this webapp is designed to be used by hearing impaired people who are doing tests in their audiologists' office, or maybe eventually in the wild. We needed to design the visual interface for people with little technical skills, poor vision, and poor dexterity.

Wireframing

For the design of this web app, our key concerns were that people with low vision and low dexterity would be using this interface. As such, I designed the buttons and the text to be as big as possible. By increasing the size of the targets, I was hoping to use Fitt's law to my advantage

and minimize the number of mis-clicks people make. Functionality wise, the audio for this webapp comes directly through the OSP, there are no preloaded sound files. Additionally, we wanted to include a signifier for "locking in" an answer choice, so this is simulated by highlighting the clicked on answer choice in green.

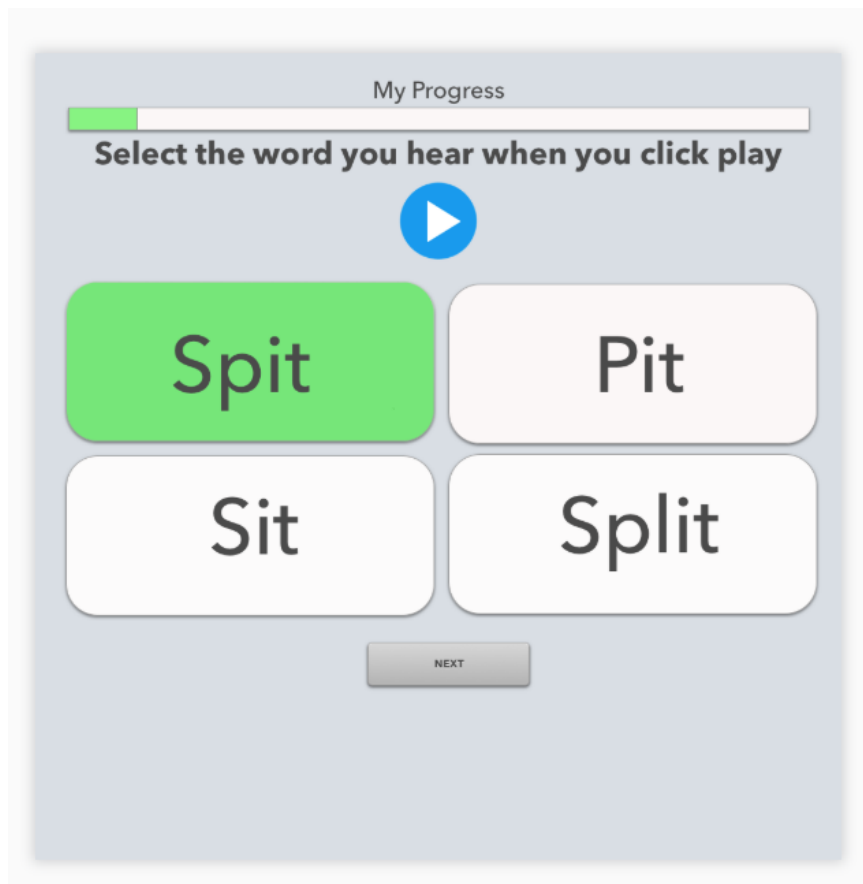


Figure 10: Original wireframe for 4afc created in InVision. Key features: large text, large target sizes. Signifier for knowing what option is chosen

Implementation

When it came time for implementation, we wanted to do several creative things: first, we needed a way to tally up how well people did in the task, so we developed a JSON data structure that is used to populate the form. This JSON is also formatted in the appropriate way to make pass it through the OSP in order to access the appropriate sound files for each page. We instruct users on how to structure their audio files for this task: asking them to create one folder for each questions sound files. The JSON data structure contains the number of the folder that is associated with each question, as well as the correct answer, and the selected answer is appended to the dataset during the task. The data structure also contains instructions to the OSP about how exactly to process and playback the required sounds. We then implemented this UI using HTML, CSS and Javascript with the REACT framework. This allowed us to have more fluid interactions and record the score. PHP is used to communicate with OSP.

```
{
  "REQUEST_ACTION":8,
  "VERSION":2
}
{
  "Word_set":[1-20],
  "Actual_answer":[1-9],
  "Noise":"babble",
  "Ambience_noise":1,
  "Fileplayback_level":0.8,
  "Noise_level":0.1,(dB SPL)
  "Ambience_level":0.1
}
```

Figure 11: 4AFC API that *Is used to communicate with the OSP to create real-time sound processing and load appropriate words into the interface. The additional parameters are OSP controls for manipulating features of the sound.*

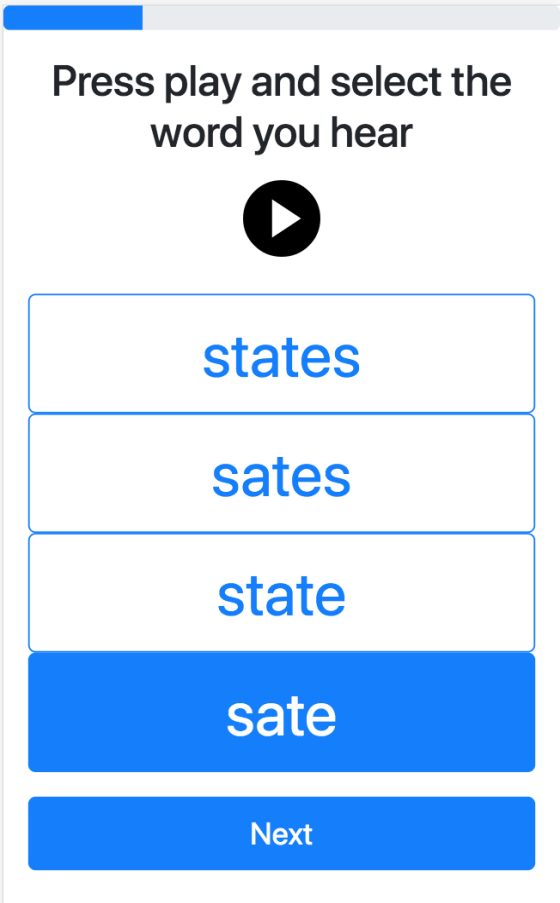


Figure 12: 4AFC Mobile UI

4AFC Web App

→ Log In

Figure 13: 4AFC Desktop Login Page used to capture participant ID and tester ID before task

Press play and select the word you hear



states	sates
state	sate

Next

Figure 14: 4AFC Desktop View shows 4 multiple choice options and offers real time sound processing via OSP

Result : 2 / 4

Tester ID: Tzubatiy
Participant ID: 123456

Question #	Words	Correct Answer	Participant's Answer
1	states, sates, state, sate	states	state
2	peak, teak, peat, teat	teak	peat
3	base, pace, bait, pate	bait	bait
4	mat, bat, mad, bad	bad	bad

[Download Result](#) [Start Over](#)

Figure 15: 4AFC results screen showing tester ID, participant ID, question numbers, question options, correct answer, and selected answer. Option to download data as a CSV is also available.

ABX Webapp

The ABX task has become the standard psychoacoustic test for determining if an audible difference exists between two sound samples (Boley and Lester). As such, our collaborators at SDSU and at the University of Minnesota (new collaborators) have asked us to create a version of it that enables real time audio processing through the OSP. We also created a version with loaded in audio files so that we used to test the behavior flow of the webapp in an experiment for COGS 230: Interaction Design Research. In this experimental procedure, three stimuli are presented. Stimulus "A" is one sound, stimulus "B" is known to be quantitatively different in some way, and the task of the listener is to identify whether stimulus "X" is the same as "A" (i.e. X=A) or the same as "B" (X=B) (Boley and Lester).

Paper Prototyping

For this webapp, we started out by paper prototyping the design of the interface. The first iteration is shown in figure 11a below. To test the behavior flow, we created and conducted a short experiment on just noticeable difference in sound pressure. We randomly selected 6/14 experimental conditions for testing on paper and then assigned each of the 6 a colored marker. Participants were asked to use taps of the marker on the page to simulate clicking, and to circle their answer choices. I used the Wizard of Oz technique by playing the appropriate sound each time the user "clicked" in this experience prototype. Users had a lot of trouble understanding what the task actually was because they were not sure what to listen for. We rapidly iterated and changed the wording to what is shown in figure 11b below. We tested this version on 4 students in the interaction design class, and got better feedback.

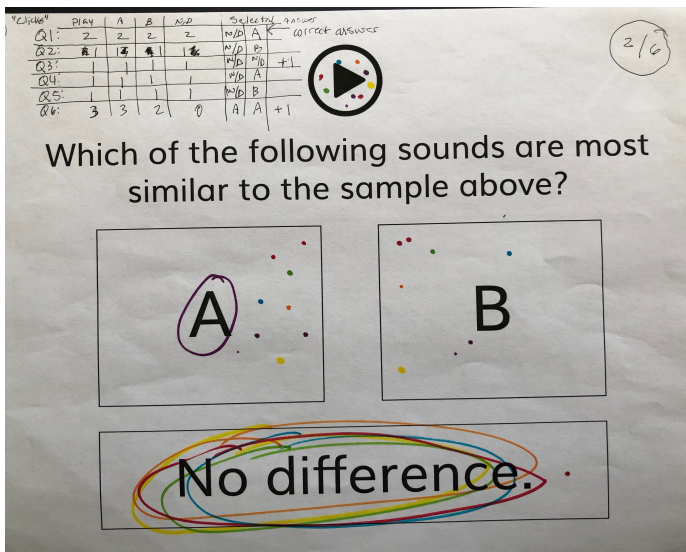


Figure 11a: Paper Prototype V1

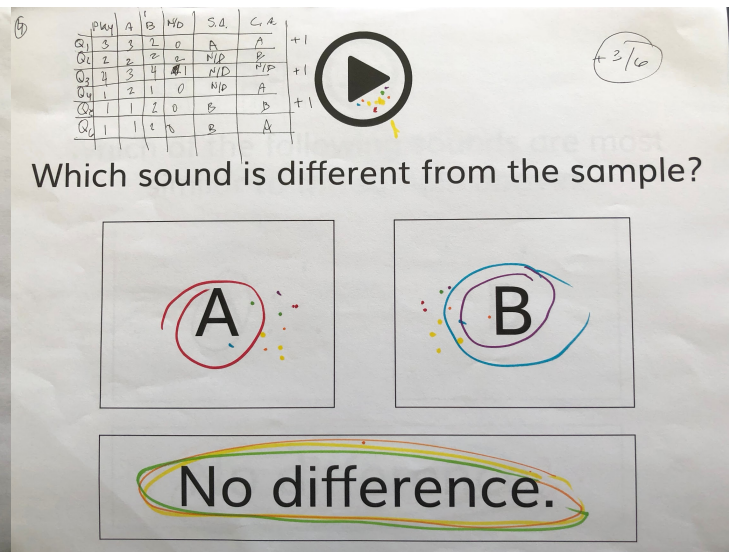


Figure 11b: Paper Prototype V2

Implementation

In the final implementation, we had a login page just like Figure 9, and a very similar behavior flow to the app. For the one used in our experiment, there was no real time audio processing, and the sounds were pre-loaded. The interface was created using HTML, CSS and Javascript with the REACT framework for easy interactions. For the actual ABX that runs on OSP, we use PHP to communicate with OSP. This is a responsive user interface, meaning it has an adaptive mobile version just like 4AFC. Our motivation in doing this is that in the near future, people with hearing loss will be accessing our interfaces from their homes, and not from their audiologists' offices, so we needed it to work on any browser, including mobile.

Figure 12: ABX Mobile view

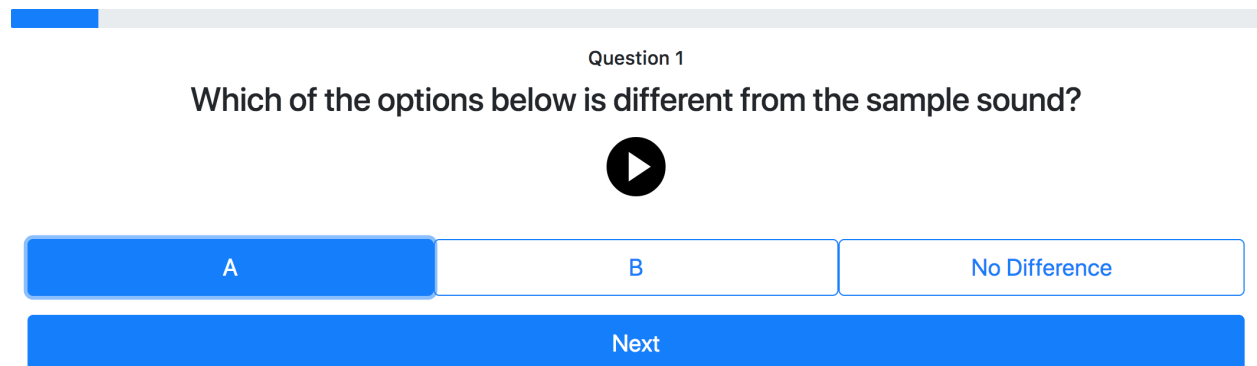
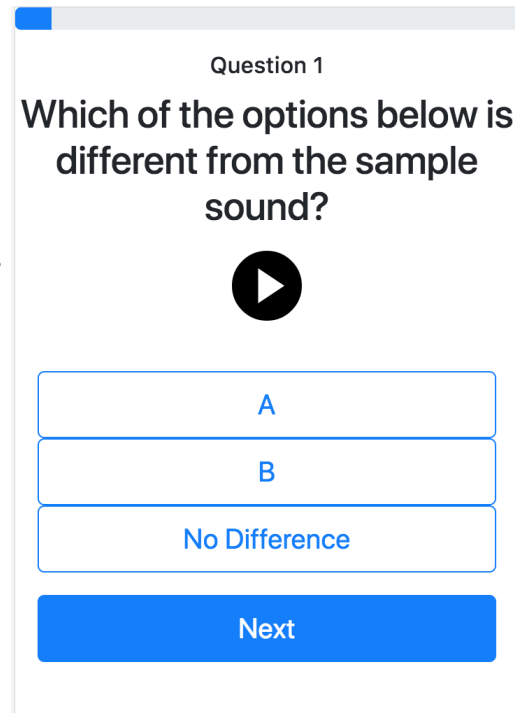


Figure 13: ABX Desktop View demonstrates the signifier functionality of highlighting which answer choice you selected.

During the experiment, we used an accompanying Google form to collect responses, but the OSP version actually has a scoresheet just like the one for 4AFC that is available for download.

People reported frustration with the lag time between stimuli when interacting with the non-OSP version, and the OSP version. As such, a future step may be to remove the play button, and have the 'X' stimulus play just before the A or the B when the user presses one of those two buttons. In this way, there will be minimal lag time between the two sound presentations and users may have a better chance of hearing a difference.

Future Work

The next steps for this project are divided into two tracts: hardware and software. I am not on the hardware side, but the next step there is to create the wearable pendant with the Snapdragon chip. We will be presenting this at International Hearing Aid Conference (IHCON) in mid August 2018. On the side of software, the wearable will enable functionality that has not been possible before. I have already designed all of the webapps we have created so far to be responsive and compatible with mobile devices. This will allow a natural transition into using these same webapps in new ways for other tasks.

The next user interface design push will be for developing an optimized self-fitting paradigm on the OSP to support the research of our SDSU collaborators.

I am also working directly with our collaborators to support their needs as they may arise. Part of this work is helping maintain our forum (shown below).

Figure 14: OSP Website forum

Welcome to the Open Speech Platform (OSP) community. We look forward to your active participation and contributions towards hearing healthcare.

[Register](#) [Log-in](#)

Forum	Topics	Posts	Freshness
1. Getting Started	3	5	1 week ago tperry
2. Hardware	2	2	4 days, 19 hours ago hgarudadri
3. Software	0	0	No Topics
4. Signal Processing	0	0	No Topics
5. Miscellaneous	1	1	1 month, 1 week ago hgarudadri
6. User Applications	1	3	4 days, 19 hours ago hgarudadri

Acknowledgements

I would like to thank Dr. Harinath Garudadri for welcoming me into his electrical engineering lab to do this work. I am very grateful for the incredible OSP team, especially Dr. Ganz Chockalingam, Dr. Arthur Boothroyd, Dr. Carol Mackersie and our other collaborators. I am also grateful for the other students on the OSP project, specifically Krishna Vastare for developing the APIs that communicate from the webapps to the OSP. I would also like to thank Sergio Luna, for his implementation of the PHP scripts that enabled this communication. I would finally like to thank Simon Li for his expertise in REACT and impressive support of our team in such a short time. I want to thank Evan Schmitz for his help in running the ABX experiment during COGS 230. Finally, I would like to thank Dr. Scott Klemmer for teaching me and supporting my work as a designer on this project.

References

- “Deafness and Hearing Loss.” *World Health Organization*, World Health Organization, www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss.
- Lin, Frank R., et al. “Hearing Loss and Incident Dementia.” *Archives of Neurology*, vol. 68, no. 2, 2011, doi:10.1001/archneurol.2010.362.
- Mccormack, Abby, and Heather Fortnum. “Why Do People Fitted with Hearing Aids Not Wear Them?” *International Journal of Audiology*, vol. 52, no. 5, 2013, pp. 360–368., doi:10.3109/14992027.2013.769066.
- United States, Congress, U.S. Census Bureau , et al. “An Aging Nation: The Older Population in the United States Population Estimates and Projections.” *An Aging Nation: The Older Population in the United States Population Estimates and Projections*, U.S. Census Bureau, 2014.
- “Quick Statistics About Hearing.” *National Institute of Deafness and Other Communication Disorders*, U.S. Department of Health and Human Services, 20 Dec. 2017, www.nidcd.nih.gov/health/statistics/quick-statistics-hearing.