Jeremiah Shugrue
Senior Design

Writing 6

Part I
Project Overview

Audiopedia is a program which enables the blind to use Wikipedia solely through voice. The user is able to interact with the website and program by speaking commands into a microphone and receives information through computer generated speech. Wikipedia is a community developed online encyclopedia and one of the most heavily used websites in the world. Audiopedia provides blind users with an unprecedented level of access to Wikipedia and the vast amount of information it contains.

Audiopedia seeks to emulate the browsing experience of a sighted user, which places a premium on speed and a coherent, intuitive design. Ideally a user should be able to quickly pick up and understand the program without any external explanation required. Audiopedia accomplishes this by segmenting each Wikipedia article in ways that allow for quick navigation. For instance, each article is accessible through a list of sections that are created based on Wikipedia's own internal layout.

While the users Audiopedia is designed for may seem like a small niche, 1.3 million Americans are legally blind and 25.2 million adult Americans have trouble seeing. Additionally, there is undoubtedly a demand amongst blind internet users for access to Wikipedia, as around 13% of internet users access it and the number is continually increasing.

Many who are legally blind are unable to use a computer or access the internet without a specialized program and equipment that helps them, such as a screen reader or Braille terminals. Screen readers read the contents of windows in the operating system to the user, including web browsers. Braille terminals are keyboard-like devices that display characters Braille.

Though a user of most screen readers can access Wikipedia, these products are based around breadth of use, concentrating on giving the user access to as many operating system features and websites as possible. As a result of this design philosophy, such programs are unable to take full advantage of the unique structure of Wikipedia. A program designed specifically for interaction with Wikipedia can facilitate more efficient navigation and give the user greater access to Wikipedia's features.

One of the ways Audiopedia takes advantage of Wikipedia's unique structure is by having a list of commands available to the user at all times. Unlike a screen reader in which the user would have to take an inordinate amount of time just to locate the search bar to access a new article, Audiopedia allows users to jump to that feature at any time by issuing the 'search' command. Giving quick access to Wikipedia's features goes a long way towards emulating a sighted user's experience.

Part II

Technical Design

Audiopedia can essentially be broken down into five main components: speech synthesis, speech recognition, Wikipedia interaction, the GUI and the core which ties everything together. The core and the Wikipedia interaction components are where most of the actual code written by myself lies, though modification was necessary for all elements to suit the project layout.

Speech synthesis was handled by the open source library FreeTTS. FreeTTS was very useful in the ease with which it could be implemented; having the program speak a line of text was as easy as issuing one command once the initial setup had been carried out. FreeTTS was also very versatile in the number of options that it allowed for changing the voice, an important feature for blind users who wish to tailor the program to their needs.

Speech recognition was done by the open source library Sphinx4. Sphinx, though very comprehensive in its features and its abilities did present some difficulties for me mainly due to the way I was initially using it. By researching these issues and making some tweaks Sphinx presented itself as a very useful package. Due to these changes much of my initial code for interacting with the voice recognition was thrown out.

Wikipedia interaction was done using the Wikimedia API and a large amount of processing by me. The Wikimedia API allowed me to access the underlying code of any article by making use of its query functionality, though this had to be very carefully processed in order to be useful. The code as I received it was in XML and contained a large amount of extraneous information that needed to be eliminated. I quickly decided that rather than processing the code each time the user made a query about an article, I would segment and store all parts of each article the first time it is loaded. Though this slows down the initial load time for an article it has the benefit of greatly improving speed once it has been loaded. This initial loading also allowed me to make the determination as to what information was available on each article when it initially loads, that way the user does not have to fumble through a number of commands that have no information associated with them if that information just isn't there on an article.

The GUI, which was handled by an open source package known as DJNativeSwing, was a late addition to the project and was implemented strictly for demonstration purposes. My focus with the GUI was to show the audience what is was that the user was accessing so they could make the connection between their own visual experience of the website and the purely audio navigation system that a blind user would be working with. Given my lack of experience with Swing and threading DJNativeSwing

presented a number of issues for me as the deadline drew closer as I had not anticipated having to implement a visual component. I was able to overcome these issues by making some creative changes.

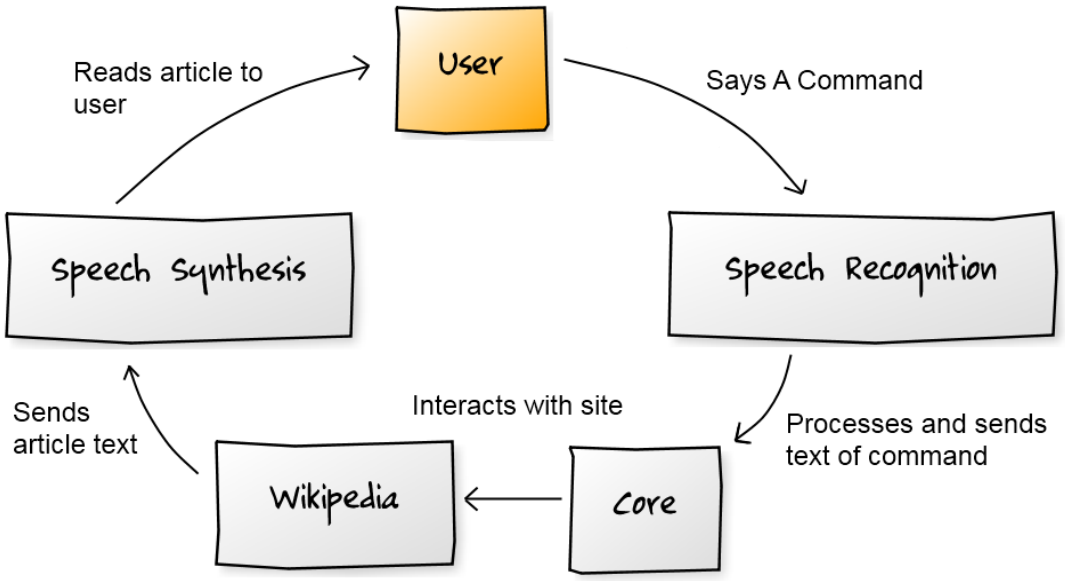
The core of the project is where all other components were tied together. This meant establishing a coherent workflow that only became apparent after I had begun developing the project. I had initially envisioned things as a cycle of issuing commands, processing and receiving text through voice. However, I came to see that it was most helpful to visualize things as more of a branching path, one which changed depending on the commands that the user was using. I therefore divided commands into either 'navigation' or 'article' commands. These differ in that navigation commands deal mainly with moving between articles and using menus while article commands deal with interacting with the articles themselves. By making this distinction I was able to make clear in my mind the idea of the state the user was in.

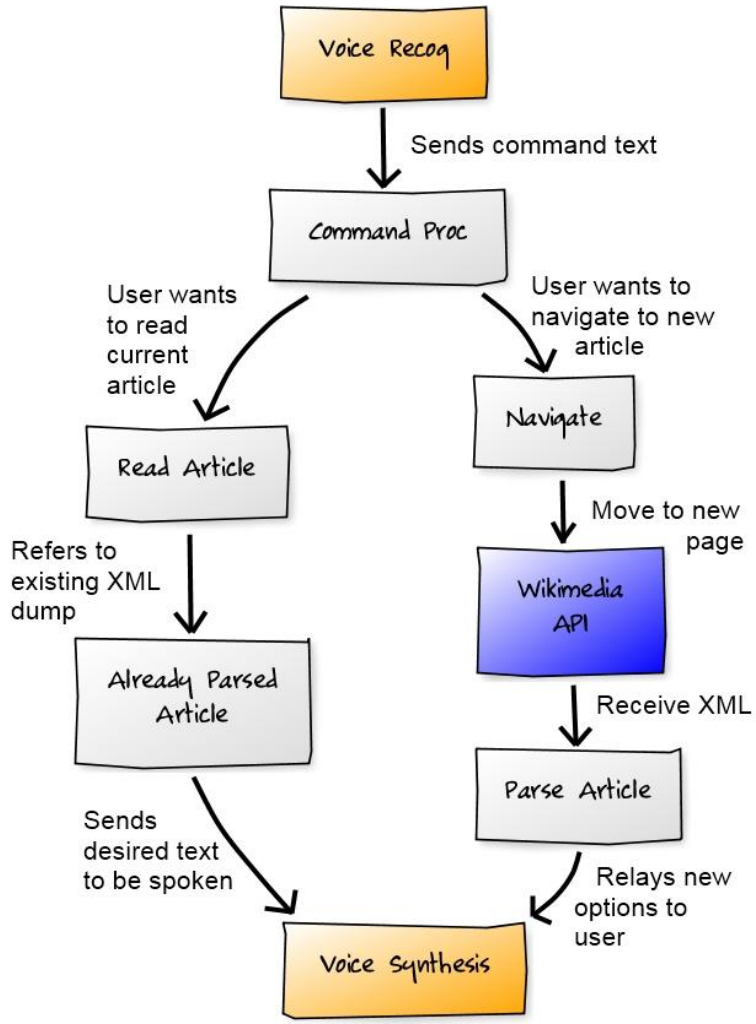
The core acts as glue between the modules of the project, taking the user's commands from the voice recognition, recognizing and performing an action based on that command and ultimately sending the necessary text to the voice synthesis. Along the way it constructs and updates the GUI to reflect the current situation and possibly pulls information from Wikipedia and processes it so that it can be given to the user.

Audiopedia underwent a number of major conceptual changes over its development, mainly the result of necessity as without actually implementing it I was unable to anticipate the necessary changes and problems along the way. One of the largest changes was in how voice commands were handled. Initially I envisioned the user as being prompted for input at various points. However it quickly became clear that this would render the program nearly unusable, as it would be extremely slow and the user would be forced to listen to a large amount of text that he didn't need to hear. As a solution I decided that the user must be able to issue a command at any time, interrupting whatever text was currently being read. This not only allows the user to skip any text he does not wish to here, but allows for the quick navigation aided by commands that I had envisioned.

Part III
Project Chronology

As previously mentioned, by the time I had finished the project the structure and layout had changed quite significantly from when I initially envisioned it. This happened both on a large scale design level and on my actual implementation on some features. The major conceptual change I think can be summed up in the change in my workflow diagrams from an early to a later presentation:


User
Reads article to user
Says A Command
Speech Synthesis
Speech Recognition
Sends article text
Interacts with site
Processes and sends text of command
Wikipedia
Core


Voice Recog
Sends command text
Command Proc
User wants to read current article
User wants to navigate to new article
Read Article
Navigate
Move to new page
Refers to existing XML dump
Wikimedia API
Already Parsed Article
Receive XML
Parse Article
Sends desired text to be spoken
Relays new options to user
Voice Synthesis

The shift from thinking of the program as simply a cycle to more of a point to point process was important as it allowed me to consider that at each point the user must have a consistent process in terms of issuing commands. This lead into another major change, the shift of voice recognition from a prompt to simply allowing the user to interject and issue a command at any time. This was an important step as it greatly improved usability and took the experience to a level that the project needed.

Though I will admit development lagged about halfway through, given that I was unable to get as much work done over winter break as I had anticipated, work really picked up pace after 70%, almost constantly accelerating until the end of the project. This was mainly due to the desire to implement new features towards the end, such as the GUI, which I had not initially planned on and had not given myself the necessary time to do them.

Part IV
Future Speculation

Given more time and resources I think I could greatly improve the user experience and expand my project to truly encompass all features of Wikipedia. I feel that the ability to take advantage of Wikipedia's unique structure is one of Audiopedia's strengths, however there are times where it can be a weakness. While a program which simply emulates a webbrowser would theoretically have access to all features, albeit in a very inefficient and hard to use way, Audiopedia does not have any of that accessibility out of the gate for features that have not been implemented yet in it. While this increases the amount of work that it necessary for making a certain feature available, I feel that it greatly improves the end result.

Given more time I would like to continue to improve and tailor the voice recognition, though I am happy with the current accuracy, I do feel there are some places in which it can be improved. I would also seek to increase the number of options available for changing the speech synthesis, for instance making available a large selection of different voices. For the Wikipedia interaction I would like to allow the user to access embedded audio files in articles and even to gain some information on images that are on a page, making use of the text information that Wikipedia stores on each.

I feel that the modular nature of Audiopedia lends itself well to further development as each component can be worked on mostly separately without the need to worry about breaking another.