# Introduction

# Problem background

-Elderly more disposable income but less access to the internet due to problems with hands etc.

-With time, more and more old people will be computer literate and will thus need to be able to access computers or the internet in particular.

-Common uses of the internet for the elderly: Shopping, email, banking, Skype etc.

-Larger problem: Finding a set of techniques that can be used to improve the usability of computers for the elderly.

# Investigation purpose and scope

-Use the web browsing as a means of determining a set of methods that can be used to improve the usability of the internet for the elderly. Beyond this, such techniques can be used to improve the general usability of computers for the elderly.

Main means of improving computer usage is the use of speech recognition. Issue with regard to speech recognition is: What sort of visual annotations and voice commands are most suited to the elderly

The primary area of investigation was the issue of whether or not numerical referencing (assigning numbers to links on web pages) would perform better or be less confusing to elderly uses over spoken link names (referring to either their name or a key word within a link name).

-idea that has a simple set easy to pronounce words will perform better.

-idea that numbers are less conducing than having to read out entire link name or specific words

# Requirements

## Fundamental requirements

Iteration 1

....

Iteration 2

...

Iteration 3

...

-Determine whether or not spoken link names or spoken links perform better

-Determine which is preferred by the elderly users

-Determine whether visual feedback or voice feedback is more preferable for the elderly (iteration 1, minor)

## Secondary requirements

-Determine the feasibility of having voice processed on the cloud instead of on local machines

# Success criteria

-get an indication of which voice referencing technique performs better for the elderly

-Get results that give an indication of the preferred means of controlling the web pages by voice

-Does visual feedback give some sort of improved performance or improve user experience?

# Constraints

-Cost: Constrained to use freeware tools.

-Time: Limited time to conduct tests and develop software. Lots of time required to train the elderly...

-Test subjects: The elderly are not always keen to currently conduct experiments on computers

-Health: Finding test subjects who have sufficient hearing, eyesight and intellectual consistency to conduct the test.

-Skill: Most elderly people are not computer literate

# Assumptions

-Assume that all test subjects are not computer literate. This meant that earlier tests had to be very sequential in nature.

-More complicated tests, complete guidance

# Investigation procedure

-construct basic test to derive basic preference

-conduct more specific tests to confirm previous results and elicit more specific user preference

-Conduct a test on a facsimile website which incorporates multiple features from previous tests in an attempt to gain a more qualitative assessment of user experience on this more complicated site.

# Sensitivity to test subjects

-Repetition limited to three times per mistake

-If individuals discomforted at any stage of the test, they were permitted to stop (some had heart problems)

# Ethics

# Tools

Free tools used.

-Initially, Eclipse was used to write the JavaScript and run the client code locally.

-Eclipse also ran very slowly on our machines

-unpredictably and problems that arose meant that an alternative had to be sought out. The loosely typed language of JavaScript meant the usual advantages of IDE’s are not really experienced...

-Chose to use Note++ to do syntax checking

-ran IIS server

-Version control: Used Git and Github to perform merges and backups

-Toggl used to keep track of the time spent on the project

# Group work and division

-Git

-from when Git was used (initial merge)

-Division

# Problems encountered

-Unpredictable behaviour with IIS

-Going through proxy server, vocabulary populated with unknown words

-Internet connectivity and the problem with testing. One of the tests had to be aborted

-Finding test subjects who were willing

# Time breakdown

-History with Toggl

-Due dates for different iterations

-Meeting deadlines?

-Meeting with supervisor?

# Results

## First iteration

## Second iteration

## Third iteration

## Interpreting the results

# Implementation

## Implementation overview

The developed code essentially evolved through the three iterations of the project. For this reason, the implementation will be discussed in light of the most complex iteration, Iteration 3. Most of the functionality remaining in Iteration 3 could be used in earlier iterations as well, provided that the appropriate parameters are set. However, since no more changes were required in earlier iterations of the project as iterations progressed, the *JS* code used for each iteration was separated for each iteration.

## Using the speechAPI

SpeechAPI [ref] provides a flash object, *SpeechAPI.swf*, which can be embedded onto an HTML page using a *SWFObject.* This flash object is then used to stream audio between the client and server. When the server sends a callback back to the client, the *SPeechAPI.js* code interfaces with the flash object.

-Limited vocab

-closest matching

-advance features not used

## Main method descriptions

### Controlling flow and notifications

The main processing of results from the server (including setting up connections) is conducted in *speechprocessor.js*. Since two primary streams of tests were undertaken, two main courses of the program are determined by the state of the Boolean variable, *numericalReferenced*. If this variable is true, the program will divert to a stream which performs the required functions for numerical functions. Otherwise, the code diverts on a course that is implicitly, one of spoken name referencing.

Another Boolean variable which helps determine the course of the code is the variable, *confrimationMode*. This variable is set to true every time a result is returned indicating that a link should be followed or the result is interpreted as a certain command that requires confirmation.

### Confirmation mode

Confirmation, in the programs context, means that the user must either say, “Yes” or “No”. If confirmation mode (*confrimationMode* global) is set to true and the user confirms positively, a link is followed or the command is performed, otherwise, the code will refresh the pages. Confirmation was a feature only introduced at Iteration 3 of the program. It was introduced to compensate for misinterpreted user commands. From the results of the earlier iterations it was clearly evident that results would often be returned as something they are not. If a result is interpreted by the speech engine as being significant in nature, that is it will result in some form of navigation, a confirmation is required of the user (so for commands “Home”, “Backwards”,””Forwards” and all link selection).Although users preferred not to have to perform a confirmation step (See Section XXX), it is at this stage necessary.

### onLoaded callback

The processing that takes place from onLoaded, is the setup of the required vocabulary needed by the speech engine (words that it will recognise and return text results for). The vocabulary goes through to iterations to determine what words are actually needed.

Firstly, the total vocabulary is determined (See Figure 2 in Appendix XX). From here, the specifically needed words in the vocab are slected from the words list (note: This is only necessary for the numerical referencing mode as the total amount of vocab –numbers –needs to winnowed down to anly cater for the specific numbers of links on a page. All outputs were run through this additional function tom make the code easier to read.) See Figure 3.

Once the specific vocab for a web page has been determined, the speech engine gets loaded with the derived vocab using speechapi.setupRecognition.

### determineSpecificConfirmations/LinkNames/Numerical string

…

Do diagram…

# Variation from plans

# Critical analysis

## Tradeoffs

Problems

## Recommendations

-finding the key words within link names dynamically. Can this be done to an extent that will provide adequate clarity sufficient distance between words?

-What sort of words is commonly misinterpreted?

-trained speech engine to deal with elderly voice requirements

-Hand held activation?

-Future iterations:

-Automatic button pressing

-Numbering starts again for different sections (confirm whether people like it or not?)

-Colour calibration for links colour and link highlighting, setting confirmation settings

-Do all of this on another website? (more complicated?)

-Possibly extend from mock up to real website to test feasibility of methods in real applications (real websites)

# Conclusion

# Bibliography

APPendix:

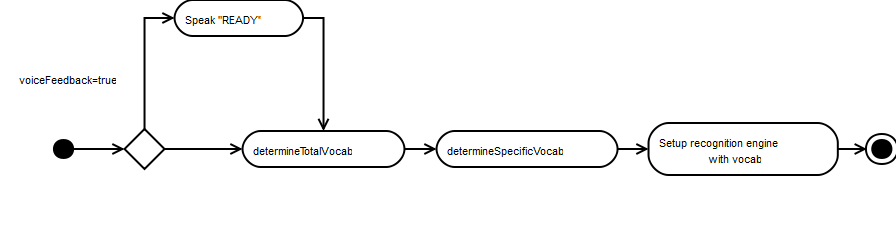


Figure : OnLoaded Overview

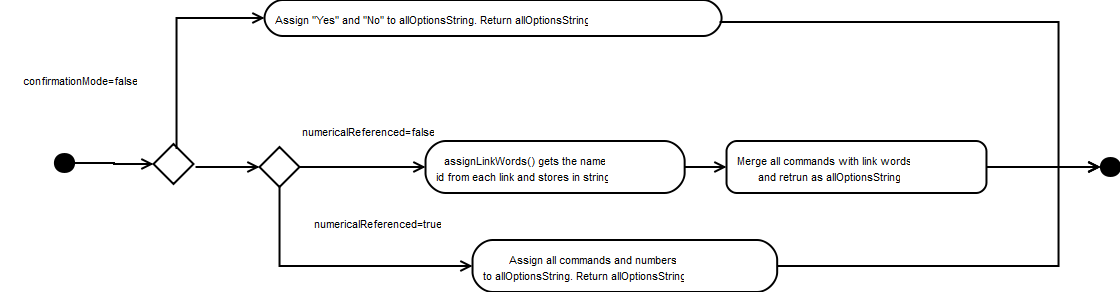


Figure : Determine total vocab overview

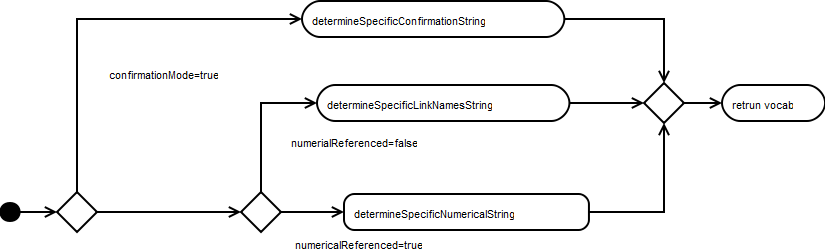


Figure :Determine specific vocab