# 

# VoBrowse – Voice Controlled File Browser

## USER’S INSTRUCTIONS

**Main Window Commands:**

1. List files : speaks the content of the current directory opened in the Browser
2. Open <folder name>: opens the folder in the browser
3. Open <file name> : opens the current file in file Browser (see File browse commands for commands to browse the file)
4. Move up: move one directory up
5. Move forward: move forward in history(only if back was an earlier command)
6. Move back : move back in history
7. Find files: starts the file search option. Set focus to search bar.(see Search command section for list of commands available in this mode.)
8. Show bookmarks: lists the bookmarks in the current session.
9. Show history: lists the history in the current session.
10. Show browser: switch tab to directory browser section
11. Turn silent: Stop the current dictation/Speech.
12. Browse drive <drive letter>: set directory browser root to this drive. Only available in windows.
13. Show drives: switch tab to list the drives available on the system. Only available in windows.

Search Commands:

1. A – Z: add the alphabet to the search bar
2. Question mark / period / star: add ? / . / \* to the search bar
3. 0 – 9 : add the corresponding digit to the search bar.
4. Delete single letter: deletes a letter from the search bar
5. Delete all letters: deletes the contents of the search bar
6. Cancel: cancel current search

As letter a-z are very small utterances, the recognition is at times very unreliable. Hence to overcome this problem, following phrases can be used as an alternative and to the same and they put only a single alphabet on the search bar.

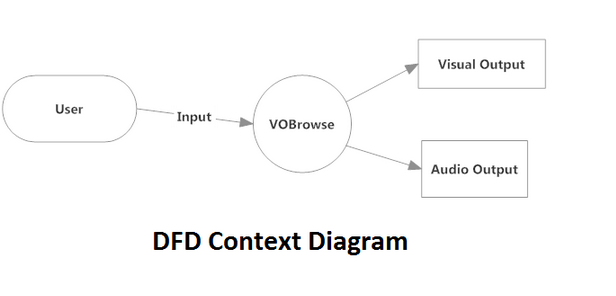
1. a for apple
2. b for ball
3. c for cat
4. d for doll
5. e for elephant
6. f for fish
7. g for girl
8. h for hen
9. i for ink
10. j for joker
11. k for kite
12. l for lion
13. m for monkey
14. n for nest
15. o for owl
16. p for pig
17. q for queen
18. r for rat
19. s for ship
20. t for toy
21. u for umbrella
22. v for van
23. w for wheel
24. x for xylophone
25. y for yak
26. z for zebra

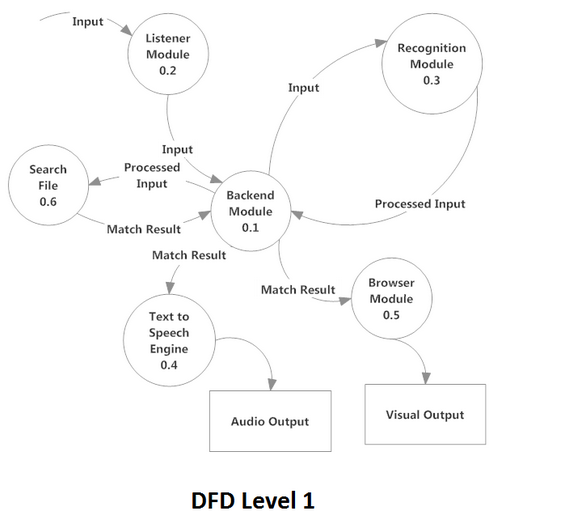
**File Browser commands:**

1. scroll up : scroll up the file browser
2. scroll down : scroll down the file browser
3. font up : increase the font size
4. font down : decrease the font size
5. close : close the file browser

## Technical Details

The following Data Flow Diagram demonstrates the basic flow of information of the software





**The whole working functionalities are divided into certain modules, as shown in the DFD :-**

1. **Listener module**:Mic is used as listener module. It is a hardware device used to input the sound in the machine.

**Working Procedure:**

1. Spoken words enter a microphone.
2. Audio is processed by the computer's sound card.
3. **Recognition module:** It processes on the input signal which it gets from the Backend Module. ***VoiceBase***is a cross platform module, developed by us which handles the task of recognition in the software. We are using two APIs, **PocketSphinx for linux** and **WSR for windows platform** respectively.

*Note: The recognition module of Linux may be improved upon training using acoustic models.*

* **PocketSphinx**:Pocket Sphinx is a small-footprint continuous speech recognition system, freely licensed under a simplified BSD license.

**Features:**

1. Support for semi-continuous, phonetically-tied, and fully continuous acoustic models
2. Model footprint on disk of about 10MB per language.
3. Memory footprint under 20MB for medium-vocabulary continuous recognition.
4. Trigram language models and JSGF finite-state grammars.
5. Acoustic models for English and Mandarin.
6. Small language models for English and Mandarin (simplified and traditional characters)

* **WSR**:Windows Speech Recognition.It uses digital input technology to optimize accuracy.It comes bundled with windows 7 and windows vista

**Features:**

1. Windows Speech Recognition has fairly high recognition accuracy and provides a set of commands that assists in dictation.
2. It requires a bit training to increase accuracy

**Working Procedure:**

1. The software discriminates between lower-frequency vowels and higher-frequency consonants and compares the results with phonemes, the smallest building blocks of speech.
2. The module maintains the grammar. In Linux, building a dynamic dictionary compromises speed greatly. We therefore break the file names into smaller keywords to match them with a limited dictionary. Since the dictionary is huge, we use the ***TRIE Data Structure***for quicker results.
3. The software then generates results and matches them with groups of phonemes, and then to actual words, determining the most likely match.
4. Contextual information is simultaneously processed in order to more accurately predict words that are most likely to be used next, such as the correct choice out of a selection of homonyms such as *merry*, *marry*, and *Mary*.
5. Selected words are arranged in the most probable sentence combinations.
6. The processed sentence is transferred to the backend Module
7. **Search Module:** The Search Module gets its input from the Backend Module. Its unique feature is that it is implemented as an ***asynchronous process****, t*hat is, a user can request a search and at the same time switch tabs. The search keeps happening in the backend. The user can then at a later point of time get back to the result tab.
8. **TTS Module**: Text to speech module for speech synthesis .It converts digital text into a high-quality English-speaking voice. We are using ***festival in Linux*** and ***SAPI in Windows*** as speech synthesis API.

* **Festival**:It is general framework for building speech synthesis systems as well as including examples of various modules. Festival and the speech tools are distributed under an X11-type license allowing unrestricted commercial and non-commercial use alike.
* **SAPI**:The Speech Application Programming Interface or SAPI is an API developed by Microsoft to allow the use of speech recognition and speech synthesis within Windows applications.

**Workingprocedure:**

Document input

|  |
| --- |
| **Text module** |

Generates linguistic structure

Normalises and parses text

|  |
| --- |
| **Speech module** |

Spectral rules

Speech synthesizer parameter values

Speech waveform

|  |
| --- |
| **Voice synthesizer** |

Fig.: Speech output

1. **BrowserModule**: We have built our file browser based on QeFem, Qt-based open source file browser.
2. **Backend Module:** The Backend Module is a multi-threaded process which coordinates the functioning of the software.

**Working procedure:**

1. It keeps listening for the user’s input.
2. Once it gets it, it passes the input to the recognition module for its conversion to an input in machine understandable form.
3. The processed input is then passed to the search module which searches for the input query file.
4. The results of the search module are then passed on to the Audio and Visual Output as required.