



Bilkent University

Department of Computer Engineering

Senior Design Project

Project Short Name: Clerk

Project Specifications Report

Ahmet Malal, Ensar Kaya, Faruk Şimşekli, Muhammed Salih Altun, Samet Demir

Supervisor: Uğur Doğrusöz

Jury Members: Prof. Varol Akman and Prof. Çiğdem Gündüz Demir

Innovation Expert: Mehmet Surav

Project Specifications Report

Oct 14, 2019

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2.

Contents

1	Introduction	2
1.1	Description	2
1.2	Constraints	3
1.2.1	Implementation Constraints	3
1.2.2	Economic Constraints	3
1.2.3	Ethical Constraints	4
1.2.4	Sustainability Constraints	4
1.2.5	Social Constraints	4
1.2.6	Language Constraints	4
1.3	Professional and Ethical Issues	4
2	Requirements	4
2.1	Functional Requirements	4
2.1.1	Receiving Voice Input	4
2.1.2	Converting Voice Input to Text Commands	4
2.1.3	Providing Microsoft Word Functionality	5
2.1.4	Read-back Functionality for Error Checking	5
2.1.5	Document Navigation	5
2.2	Non-functional Requirements	5
2.2.1	Reliability	5
2.2.2	Efficiency	6
2.2.3	Extensibility	6
2.2.4	Compatibility	6
2.2.5	Usability	6

1 Introduction

Microsoft Word is one of the most popular word-processing programs around the world. It is used primarily to create various types of documents that you can print and publish, such as books, papers and reports. When you open a file in Microsoft Word, it can be edited using various features that Word provides. Currently, these features are accessible through use of some input devices, such as mouse and keyboard.

In this way, it is assumed that people who are going to use Microsoft Word, should be able to use these devices in a conventional way. However, some people can't use these devices effectively or at all. Some people might not want to use them. Some people can't use their hands or are visually impaired. There should be a reliable way for these people who also may want to create documents and write reports, poems, and maybe a book.

Let us consider a visually impaired person. They should be able to create, delete, and save files; type and delete sentences; change the font and type of the text; change the position of the cursor and the alignment of the paragraph; change the color of the words; insert images. Briefly, they should be able to use basic functionalities of a program like Microsoft Word. There is no tool currently available that provides use of major functionalities of Microsoft Word without the use of mouse and keyboard together.

In the market, there is an add-in Dictation for Microsoft Word, which enables you to write without using the keyboard using speech recognition. However, this is the only feature of this add-in. There are also plenty of web applications such as Speechnotes that allow users to take voice notes and convert to text. These applications can't be used to create complicated documents. Additionally, in most of them, there is still a need to use a mouse to make use of other functionalities.

Our goal is to build an application that can work entirely by way of voice commands, eliminating the need to use mouse and keyboard while editing documents in Microsoft Word.

1.1 Description

Clerk will be an add-in for Microsoft Word designed for users who aren't able to or don't want to edit documents using keyboard and mouse. It will work on any platform that can run Word. The user will be able to dictate their commands by giving voice input from an input device. Clerk will take commands from the user's voice input and apply them in Word. For instance, when the user says, "Dear John", the program will insert the words "Dear John" to where the cursor currently is. Additionally, it will allow users to execute common functionality like copy, paste, delete, select a portion of text, as well as save the file, make selected text bold or italic. For instance, when the user says, "Save the file as PDF", it will save the file in PDF format.

Clerk will be useful especially for visually impaired people who are not able to use the keyboard and mouse devices. Our application will let them edit documents using only voice

commands.

Clerk will also have features to detect errors and misunderstandings, and confirm spelling of words. These features will work by reading the specified text/paragraph or spell a specific word if the user commands the application to do so. If Word detects a spelling error or encounters a word that is not in the dictionary, the application will alert the user. Also, for visually impaired people it is not possible to use the mouse to navigate inside the document. To ease the use of Word for them, we will have commands to set the position of the cursor by specifying the location where they want the cursor to be. For example, “Second Paragraph First Sentence” will take the cursor to the first sentence of the second paragraph.

We are going to use Word JavaScript API to develop the Microsoft Word add-in [1]. HTML, JavaScript and CSS will be used for implementation of our add-in, which is the standard way add-ins are built for Microsoft Word.

In short, Clerk will allow users to use Microsoft Word without using a keyboard or mouse.

1.2 Constraints

1.2.1 Implementation Constraints

- Clerk will be developed as a Microsoft Word Add-in application.
- Javascript, HTML, and CSS will be used to develop the add-in.
- Git and GitHub will be used for version control and collaboration.
- Web Speech API will be used for speech recognition and synthesis [2].
- Word Javascript API will be used in development of the add-in.
- Common API will be used to control the objects and metadata in Word documents [3].
- The application will be built under Object Oriented Programming paradigms.

1.2.2 Economic Constraints

- Clerk will be an open-source Microsoft Word Add-in that will be available to use for free to all Word users and developers.
- Free and open-source libraries and APIs will be used.
- The web page of the application will be on the Github domain, which is free.

1.2.3 Ethical Constraints

- Personal information and private data will not be shared with any third parties.
- Microsoft Word already encrypts the user data, which is accessible only after authentication.
- Voice input from the user won't be used for anything other than converting the text and parsing to understand commands.

1.2.4 Sustainability Constraints

- To improve the usability of the application, feedback from the users will be considered.
- The application will be updated by the developers according to feedback taken from the users.

1.2.5 Social Constraints

- Even though the main audience of the application is visually impaired people, the application can be used by everyone.

1.2.6 Language Constraints

- Turkish or English languages will be supported by the application.

1.3 Professional and Ethical Issues

- The consent of the user is required since we will take their voice input and process it.
- The application will not store the voice input content in any way. So, there is no way we can share data with third parties.
- Licenses for third-party APIs and libraries will be checked before usage.

2 Requirements

2.1 Functional Requirements

2.1.1 Receiving Voice Input

- The user will be able to give voice input to the application using any voice input device.

2.1.2 Converting Voice Input to Text Commands

- The input received will be converted to text to be parsed.
- The text input will be parsed to understand the contents. The application will figure out which functionality the user wants to use, such as typing, copying, pasting, or saving the file.

2.1.3 Providing Microsoft Word Functionality

- After understanding what the user wants to do by parsing the command, the application will be able to execute it.
- The user will be able to command for any functionality provided by Microsoft Word. These functionalities include copy/paste, saving files, inserting images/figures/tables, navigating the contents, adding/removing text, and changing format of the text, e.g. font.

2.1.4 Read-back Functionality for Error Checking

- The user will be able to request a read-back of a word, sentence, or paragraph from where the cursor currently is. This means that the program will convert the text in the document to speech and read it to the user.
- The user will be able to stop the read-back and ask for a spelling of a word. This will help find typos.
- The read-back functionality will be integrated with errors found using Microsoft Word's error checking mechanisms, e.g. underlining a word in red that isn't in the dictionary, so that the user will be alerted to these errors real time.
- The user will be able to set the speed at which the words are read by the read-back feature.

2.1.5 Document Navigation

- The application will provide satisfactory navigation inside the document. It will have commands to be able to navigate to certain sections, pages, paragraphs or sentences. It will also provide functionality to search for phrases within a document, and navigate through occurrences of those phrases.

2.2 Non-functional Requirements

2.2.1 Reliability

- The application must be able to convert voice to text with acceptable accuracy. Errors and misunderstandings must happen rarely. When they do happen, the user needs to be alerted appropriately.
- The application must not randomly crash. It must be able to recover from most errors. It must be able to communicate the errors and the reasons they occurred to the user as much as possible.

2.2.2 Efficiency

- The application must not take more than 5 seconds to process and respond to any voice command. This requirement includes speech to text delay and parsing. Longer delays could turn the user away from use of the application.

2.2.3 Extensibility

- It must be easy to develop new features and add new functionality into the application. This requires the application to be sufficiently modular, which will be achieved by using Object Oriented Programming paradigms and appropriate design patterns.

2.2.4 Compatibility

- The application will be able to work on multiple platforms that can run Word such as Windows, Mac, iPad and web browser. This feature is provided by Microsoft Office themselves [4].

2.2.5 Usability

- Once the application is launched, the user should be able to perform all the functionality using only voice commands from an audio input device. There should not be a need to use any other devices. This is especially important since our target audiences include people who aren't able to use these devices.

References

- [1] “Word JavaScript API Overview - Office Add-ins | Microsoft Docs,”
<https://docs.microsoft.com/en-us/office/dev/add-ins/reference/overview/word-add-ins-reference-overview?view=word-js-preview>, Accessed: 2019-10-11.
- [2] “Web Speech API - Web APIs | MDN,”
https://developer.mozilla.org/en-US/docs/Web/API/Web_Speech_API, Accessed: 2019-10-12.
- [3] “office package - Office Add-ins | Microsoft Docs,”
<https://docs.microsoft.com/en-us/javascript/api/office?view=word-js-preview>,
Accessed: 2019-10-13.
- [4] “Office Add-ins Documentation - Office Add-ins | Microsoft Docs,”
<https://docs.microsoft.com/en-us/office/dev/add-ins/>, Accessed: 2019-10-12.