CS 319 - Object-Oriented Software Engineering

Analysis Report – Fınal

Genius

Group 15

Asena Rana Yozgatlı	*21000132*
Serhan Gürsoy	*21400840*
Kutbidin Ululu Meder	*21303000*
Mustafa Said Sarı	*21302224*

Deadline: 29/10/2016

Course Instructor: Bora Güngören

Table of Contents

Table of Figures

1. Introduction

In past 20 years, computer/programming technology developed extremely fast. While technology developed that rapidly, for obvious reasons, it started to give users option to personalize these technologies according to their desires. Such personalization become more visible especially in mobile and computer technology. Therefore, right now, a regular user has opportunity to access tons of information with just few clicks. This information can be about something totally new which user don't know or it might about user him/herself, like his/her schedules, mails, contacts. Because of such circumstances, users started to need some software which can help them in order to retrieve exact information that they were looking for. Therefore, this created a new software type which called "Intelligence Personal Assistant Software". Right now, big companies like Google, Apple and Microsoft have their own personal assistant software. Google has Google Assistant, Apple has Siri and Microsoft has Cortana, and these companies are putting too much effort and budget to intelligence personal assistant software because it can change user interactions with information completely.

2. Overview

Genius is an intelligence personal assistant for computers. This Intelligence personal assistant will run at the background as a Windows Service. User will use input box to command assistant and it will execute these commands whether they are related about getting an information from internet or related with personal computer -like starting an application.

Like all other intelligence personal assistants in the industry (Cortana, Siri, Google Assistant) our personal assistant's priority also will be helping user but not being a companion to her/him. Yet, as a secondary goal, we can implement first phases of the natural language processing to give

assistant to reply simple user questions (i.e. “How are you?”, “What is your name?”). However, this won’t be advanced and shouldn’t be confused with other Artificial Intelligence Chat Bots. Although these features can be expanded in near feature, we need to implement basic features which an assistant require to work functionally. In early versions, Genius will only take text-based inputs but in further releases assistant could recognize natural voice without the requirement for keyboard. Assistant will answer questions according to their form whether they are text input or voice input. If question is asked to learn something new but not to execute something, using certain API’s, user can get answer right away. Or user can use assistant to start applications like Word, Paint.

3. Functional Requirements

Launch Program

Launch Program: The user will be able to launch a program by commanding it and giving the name of the application to launch via text-input. If the application is already installed on the computer, then the system will launch the application for the user.

Close Program

The user will be able to close an application by commanding it and giving the name of the application to close via text-input. If the application is already running on the computer, then the system will close the application for the user.

Search

The user will be able to perform search on the web by commanding it and giving the desired phrase to search via text-input. The user should indicate where to search if 'search in one place'

check is not marked. If user inputs Google, then the system will perform a Google search and displays the result on browser. If user inputs Wikipedia then the system will perform a Wikipedia search and displays the results on an additional information window. If Wikipedia search cannot be performed for a reason, the system will ask the user if she/he wants the system to perform a Google search on the desired phrase.

Settings

Settings will be reachable from the Settings button in the main window. The user can set a default search destination, change the default browser.

Help

Help will be reachable from the Help button in the main window. Help will provide information about the current version, developers, main purpose of the program, services provided, usage of the services, and command list.

Small Talk

User will be able to get a response on basic daily talking sentences by giving the sentence via text-input. System will try and find an appropriate response and displays it to user via main window text field. If the system cannot find an appropriate response, it will display an error message indicating that the input is not understood.

4. Nonfunctional Requirements

4.1 Usability Requirements

- The Application should be User-friendly and easy to use.
- The Application should use meaningful names for buttons and icons in order to ease navigation and understandability.
- The Application icon should be visible from the screen corner.
- The application should stay at its corner in case of User transits to another window.

4.2 Reliability Requirements

- The Application should start from the point where it is left in case of power-loss(e.g. the text that was typed by the user should stay)
- The Application should save the progress of the user in case of internet connection problem(e.g. response gotten from internet should stay during internet connection problem)
- The Application should save the progress in case of User closes the application abruptly.
- The Application should save the progress in case of User hides the application abruptly.

4.3 Security Requirements

- The Application should get information for the user from reliable sources and not download viruses to the PC.
- The Application should have and access only to what is allowed by the User.
- The Application should not have an information leakage from users PC to any other repository.

4.4 Performance Requirements

- The Application should gather required information from Internet within a very short period of time.
- The Application's open/close operation should be very fast.
- The Application should work with slow internet connection.

4.5 Supportability Requirements

- The Application software should be very well documented.
- The Application code should be very clearly commented.
- The Application should be supported by all Windows versions.

4.6 Constraints

- The Application should be implemented in C#.
- Adobe Photoshop should be used for graphical usage.

5. System Models

5.1 Scenarios

Scenario 1 : Launch Program Case One

Kadri wants to open an application on his computer with using Genius Assistant. Because Kadri have no idea about how Genius works, clicks "Help" icon at the top of the main page. System shows a text which explains how Genius works. After reading instructions, Kadri presses to “Help” button again or presses anywhere in the application which directs to main page. Then Kadri writes “Open Microsoft Word” to open the program, the application launches that, if there is no program installed on the PC, then Genius informs about the condition.

Scenario 2: Close Program

Oktay was done with his coding in Visual Studio. He was doing some other job on his PC besides coding. While he was doing another job, he wants to close his previous program. In this case, he does not have to go back to previous program and close it, but he just opens Genius Assistant, and writes “Close Visual Studio”, it is that simple. Genius Assistant program made Oktay’s life easier, because the application terminates the entered program by itself.

Scenario 3: Search Google

Ahmet wants to search visual information about engines on Internet. Because he is also busy with something else, he don't want to spend time on going through all searching process, therefore, Ahmet just needs to enter a text that he wants to search for. If the application is even closed, it is a matter of milliseconds to launch the Genius Assistant. Entering the “search engines” in the text

field, he clicks to the “send” button, or clicks the “Enter” button on the keyboard, and the Assistant shows options where to search the entered from: Google, Bing, Wikipedia or Yandex. If Ahmet wants to search about engines on Google, he types “Google”, then Genius Assistant opens default browser with search results page opened.

Scenario 4: Search Wikipedia

Mehmet wants to know more about Pluto. Because he already knows about how Genius works, he simply types "Search Pluto” in the text field on the main page. Then, the application asks which search engine Mehmet wants to use. He thinks that the Wikipedia is the best option, so he types “Wikipedia” when Genius Assistant shows the search options. In less than a second, information about Pluto appears in the default browser’s page. In case, there is no information about Pluto in Wikipedia, then application informs about the case, and asks again if he wants to look for Pluto on the other search engines providing searching options.

5.2 Use Case Model

Use Case Descriptions

Use Case 1

Name: Launch Program

Participating Actors: User

Main Flow of events:

• User writes "run 'the name of program' " to launch the desired program .

• System launches the desired program since its already installed on the computer.

Entry Condition: User should start the application, write run and a valid program name into the text part.

Exit Condition: The desired program is launched successfully.

Alternative Flow of Events:

• User writes the name of the desired program to launch but since it is not a valid name(i.e misspelling the name, entering a program that is not installed on the computer), an error message is displayed and user is asked to writes a valid name.

• User writes a valid name but since the desired program is not able to launch, an error message is displayed and user is asked to writes the name of the desired program to launch.

Use Case 2

Name: Close Program

Participating Actors: User

Main Flow of Events:

• User writes "close 'the name of program' " to close the desired program.

• System closes the desired program since its already running on the computer.

Entry Condition: User should start the application, write close and a valid program name into the text part.

Exit Condition: The desired program is closed successfully.

Alternative Flow of Events:

• User writes the name of the desired program to close but since it is not a valid name(i.e misspelling the name, entering a program that is not running on the computer), an error message is displayed and user is asked to writes a valid name.

• User writes a valid name but since the desired program is not able to close, an error message is displayed and user is asked to writes the name of the desired program to close.

Use Case 3

Name: Search

Participating Actors: User

Main Flow of Events:

• User writes "search 'the string' " to search desired string.

• If user don't click 'search in one place' check mark in settings before, system asks user where would he want to search this; in Google or Wikipedia.

• System launches the default browser and performs a Google search on the desired string, if user choose to search in Google.

• System performs a Wikipedia search and displays the result on screen, if user decided to search in Wikipedia.

Entry Condition: User should start the application, write search and the desired string to search.

Exit Condition: The default browser launched successfully, and the Google search for the desired string is done successfully or the Wikipedia search for the desired string is done successfully.

Alternative Flow of Events:

• If user click 'search in one place' check mark in settings and choose Google or Wikipedia before; when user writes "search 'the string' " , the string is going to search in selected area .

• User writes the name of the desired string to search but since the default browser is not able to launch, an error message is displayed and system returns to application's home page.

• If the string is wanted to be search and cannot found in Wikipedia a message displays and asks " The string couldn't be found. Do you want to search this string in Google? " ; there are yes and no

buttons below this statement. If user click yes, application search this string on Google and display results; if user click no, system close Wikipedia and return to application's home page.

Use Case 4

Name: View Settings

Participating Actors: User

Main Flow of Events:

• User opens the settings panel by clicking " Settings " in menu bar.

• System displays the settings panel on the screen.

• User changes the default browser by selecting the directory of the desired browser.

• System saves the settings.

• User returns to the application's home page by clicking the "Settings " icon.

Entry Condition: User should start the application, and clicks the " Settings " which takes part in menu bar.

Exit Condition: User clicks the " Settings " icon.

Alternative Flow of Events:

• User click 'search in one place' check mark and choose 'In Google' or 'In Wikipedia'. System saves the settings.

• User click 'Open at startup'. System saves the settings.

• User click 'Check for Updates Automatically'. System saves the settings.

• User decided not to change settings clicks the " Settings " icon and return to application's home page.

Use Case 5

Name: View Help

Participating Actors: User

Main Flow of Events:

• User opens the help panel by clicking "Help" icon.

• System displays information about usage of the launcher, closer, searching and the version information of the program, the names and e-mail addresses of the developers.

• User clicks "Help" icon to return to application's home page.

Entry Condition: User should start the application, and clicks the "Help" icon.

Exit Condition: User clicks the " Help" icon.

Use Case 6

Name: Make Small Talk

Participating Actors: User

Main Flow of Events:

• User writes some basic daily talking sentences.

• System response according to this entered sentences.

Entry Condition: User should start the application, and writes basic sentences.

Exit Condition: The small talk is done successfully.

Alternative Flow of Events:

• User writes the sentences but since it is not a valid sentences(i.e misspellings in the sentence, entering a sentence in another languages), an error message is displayed and user is asked to writes a valid sentences.

Use Case 7

Name: Feasibility of Search

Participating Actors: Wikipedia

Main Flow of Events:

• Wikipedia takes the string to be searched.

• Wikipedia process this string and return the result that the search can be perform or not .

Entry Condition: User should start the application, and search desired string in Wikipedia .

Exit Condition: The Wikipedia return result.

Use Case Diagram

Use case diagram is shown in below.

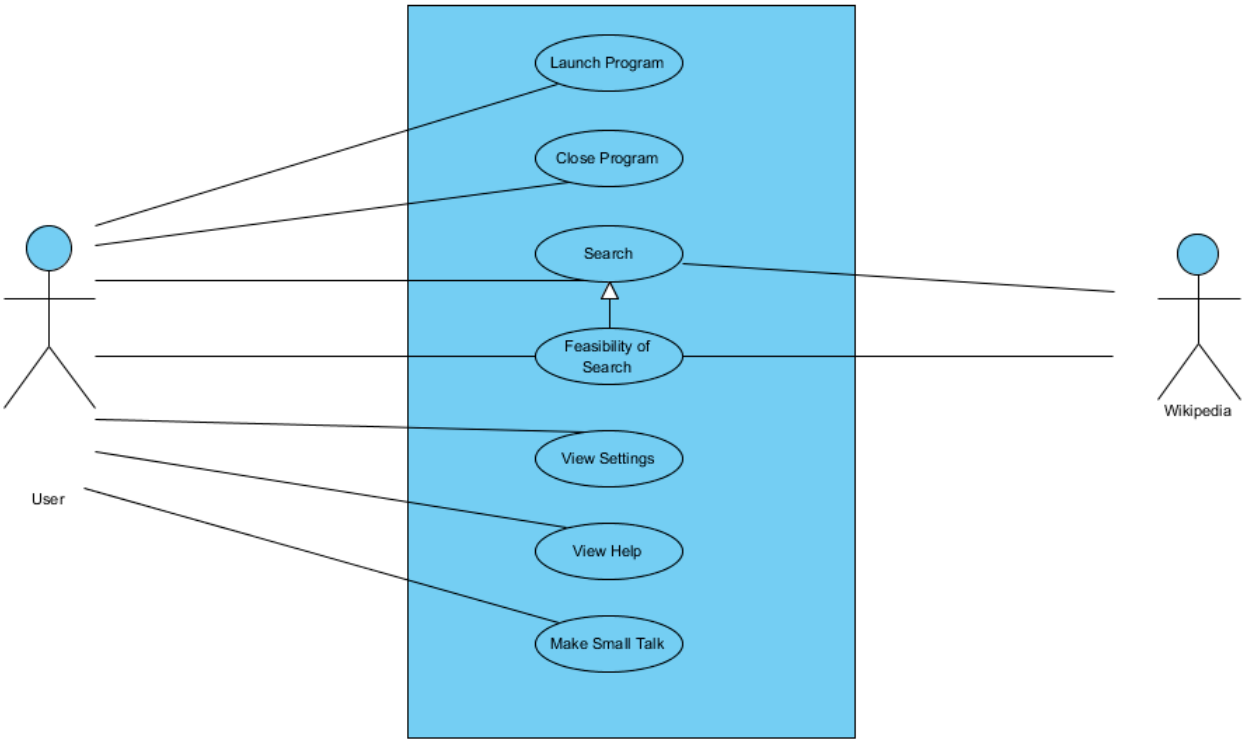

Figure 1

5.3 Object Model

Data Dictionary

Launcher: Launcher is responsible for launching application. Launchers are created by the dispatcher with a specific application information. Launcher needs this information to find and launch the application.

Killer: Killer is responsible for killing application. Killers are created by the dispatcher with a specific application information. Killer needs this information to find and kill the application.

Responder: Responder holds the response information for possible inputs and possible errors. Responder is responsible for picking the appropriate response for an input or an error. It holds a

two dimensional String array to hold the response information for possible inputs, and a one dimensional String array to hold the response information for possible errors.

Searcher: Searcher is responsible for searching web and displaying information. Searchers are created by dispatcher with a specific search information and the search type. There are two types of searches: Google Search, Wiki Search.

Google Search: Google Search uses certain API(s) to perform a search on Google, and displays the information on the desired browser.

Wiki Search: Wiki Search uses certain API(s) to perform a search on Wikipedia, and displays the information on an Info Box.

Job Object: Job Object is responsible for performing the job. It holds the job type information.

Dispatcher: Dispatcher is responsible for understanding and distributing jobs. It holds a one dimensional Job Objects array to hold the active jobs and a two dimensional String array to hold the program information. It is also responsible for getting a response from Responder and displaying it. It gets input from Dialog Box and decides what to do. After deciding it sets output to Dialog Box and performs the job by creating the appropriate job.

Info Box: Info Box is responsible for drawing appropriate windows on screen and displaying the information. It is created by Dispatcher and/or Wiki Search with a specific display information. Info Box displays this information on the window.

Dialog Box: Dialog Box is responsible for getting input from the user and displaying output to user.

Class Diagram

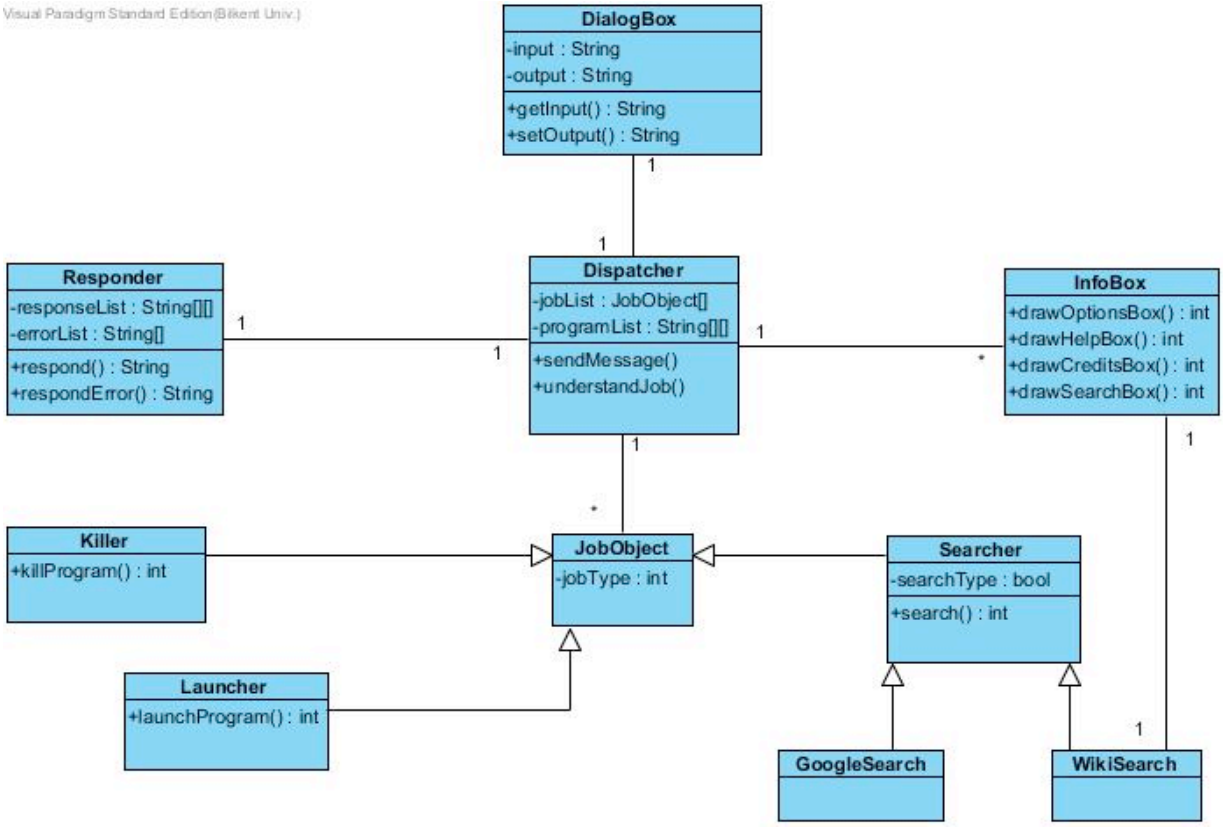

Figure 2

5.4 Dynamic Models

Activity Diagram

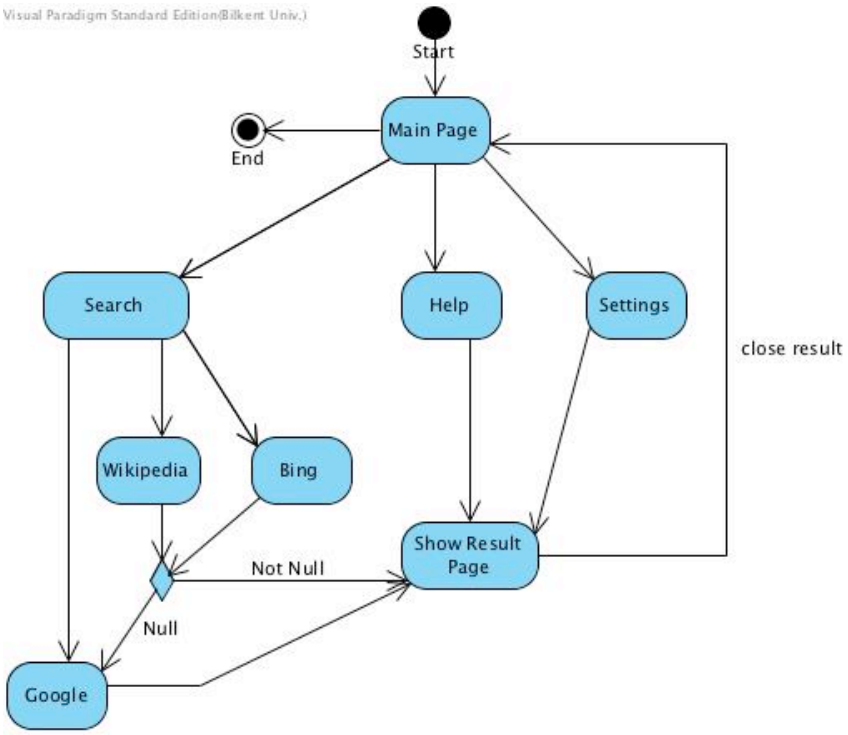

Figure 3

Sequence Diagrams

Diagram 1: Launch Program

This diagram explains a scenario which user launch a program via application. User write open 'this program(excel, paint, spotify, etc.)' into dialog box. Then Dispatcher understand what should to do and answer according to situation, in this case write "Opening 'the program'", and call JobObject , it call launcher and trigger 'launchProgram' method. Lastly, the desired program opening.

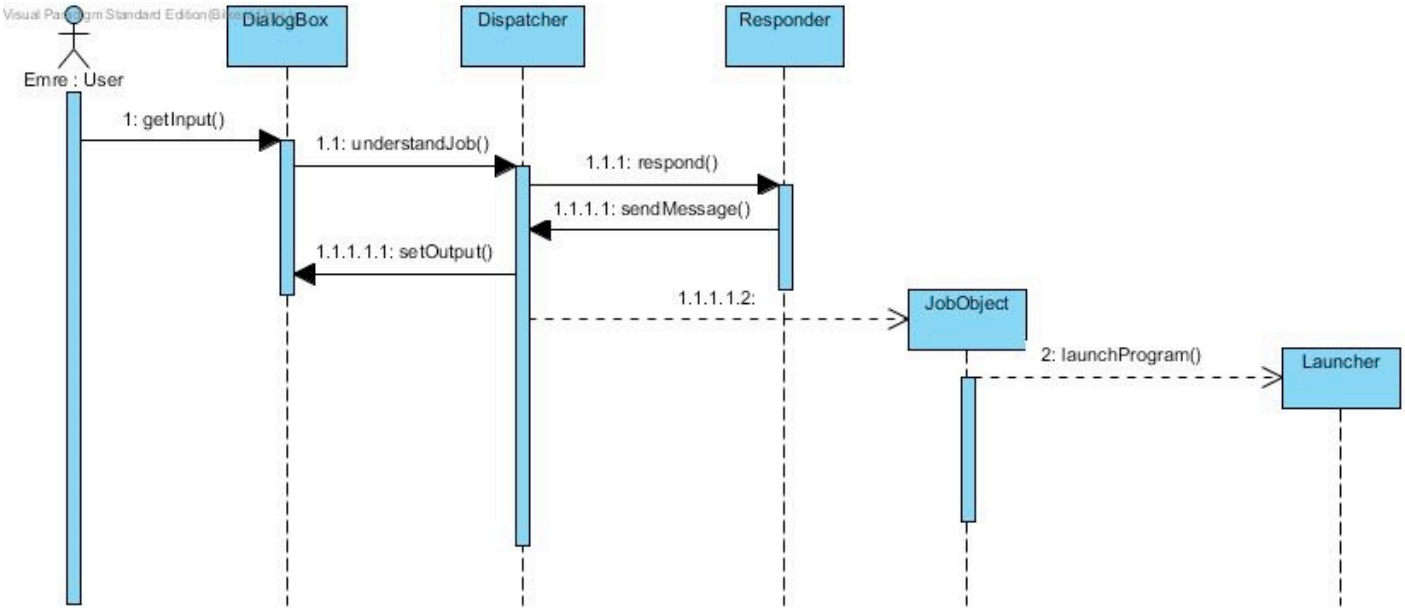

Figure 4

Diagram 2: Wikipedia Search

This diagram explains a scenario which user makes Wikipedia search. User write "search 'this string'" . Dispatcher understand and since in this case 'search in one place' check is not marked in settings, user asked to do choose Google or Wikipedia and user prefer Wikipedia. Then Dispatcher call JobObject and it call Searcher. Searcher start 'search' method and call Wikisearch. After search done, InfoBox is called by Wikisearch and it trigger 'drawSearchBox' method and displays the results of search.

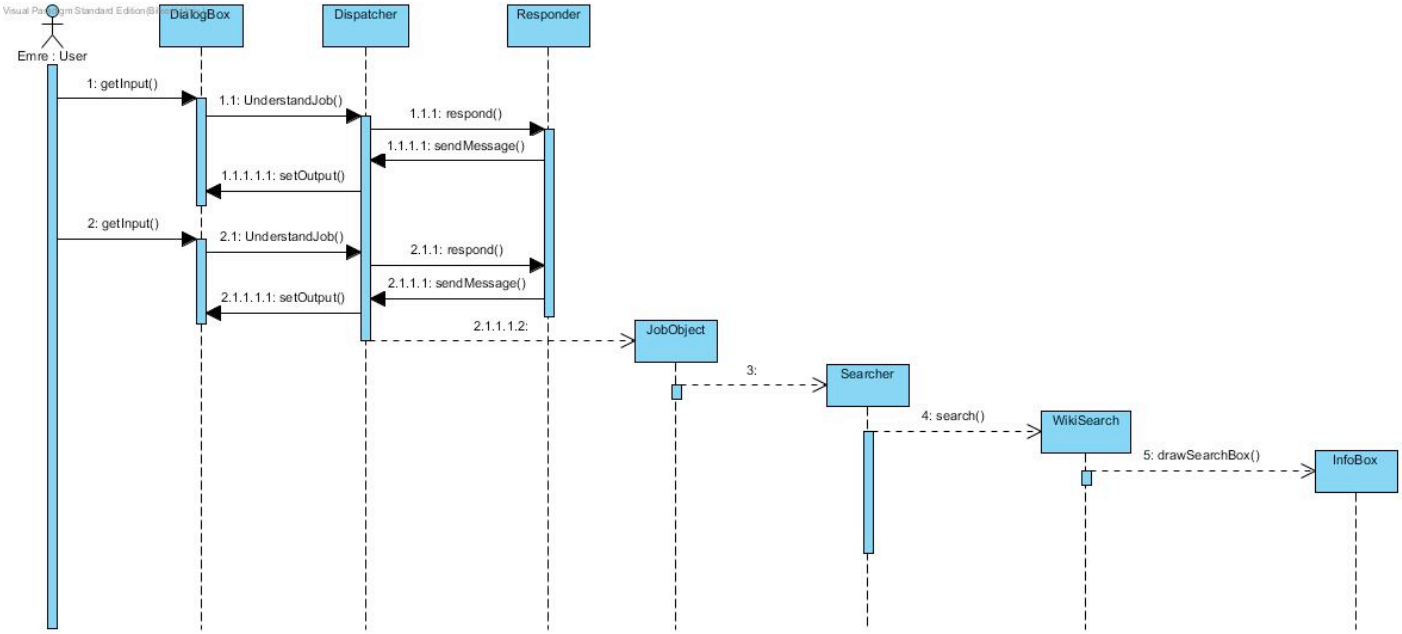

Figure 5

Diagram 3: Google Search

This diagram explains a scenario that user make Google search. User write "search 'this string'" . Dispatcher understand and since 'search in one place' check is marked and Google selected in settings, dispatcher starts Google searching by calling JobObject. Then JobObject call Searcher, it trigger 'search' method and call GoogleSearch. Lastly, search result is shown in default browser.

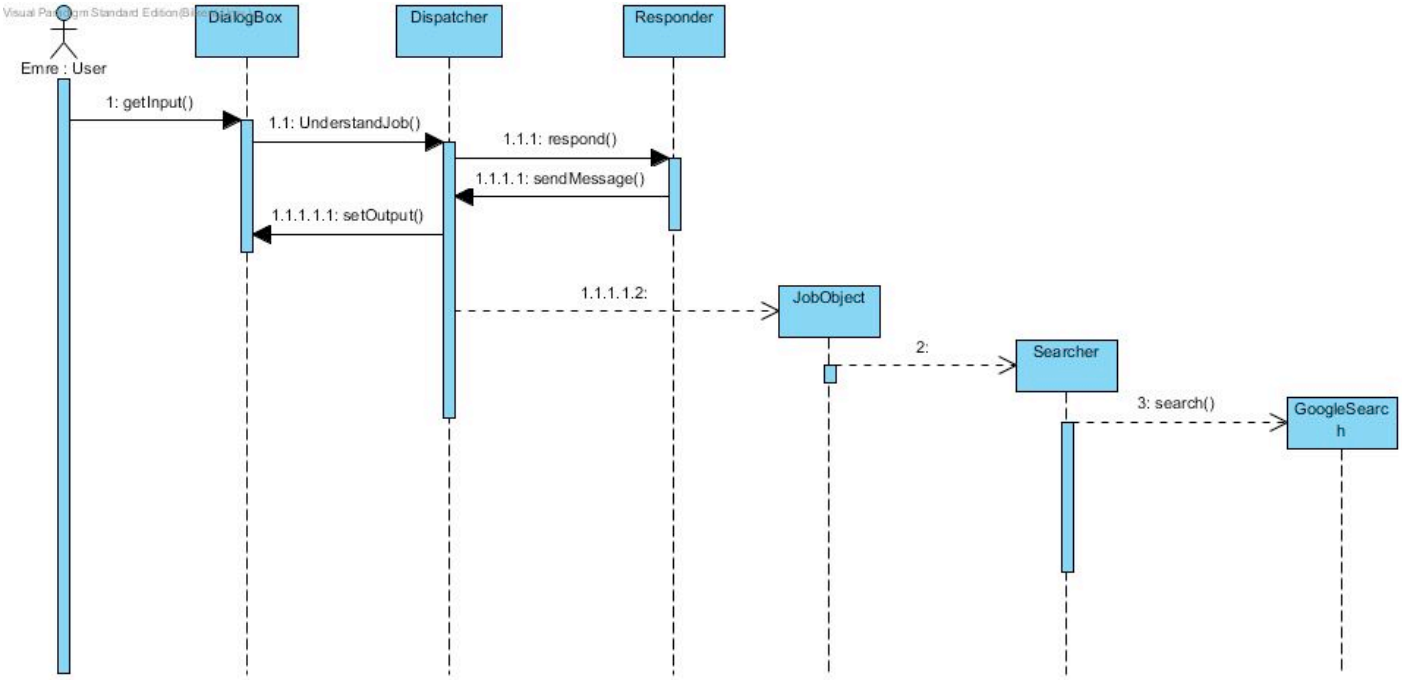

Figure 6

Diagram 4: Change Settings

This diagram explains a scenario which user change settings of application. User click Settings button and change the default search destination or the default browser. Then, Dispatcher understand and save new settings .

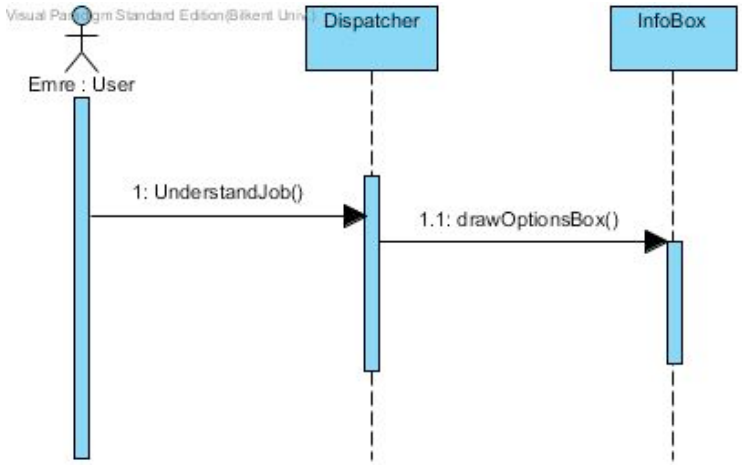

Figure 7

5.5 User Interface

In this part, user interfaces of genius will be examined.

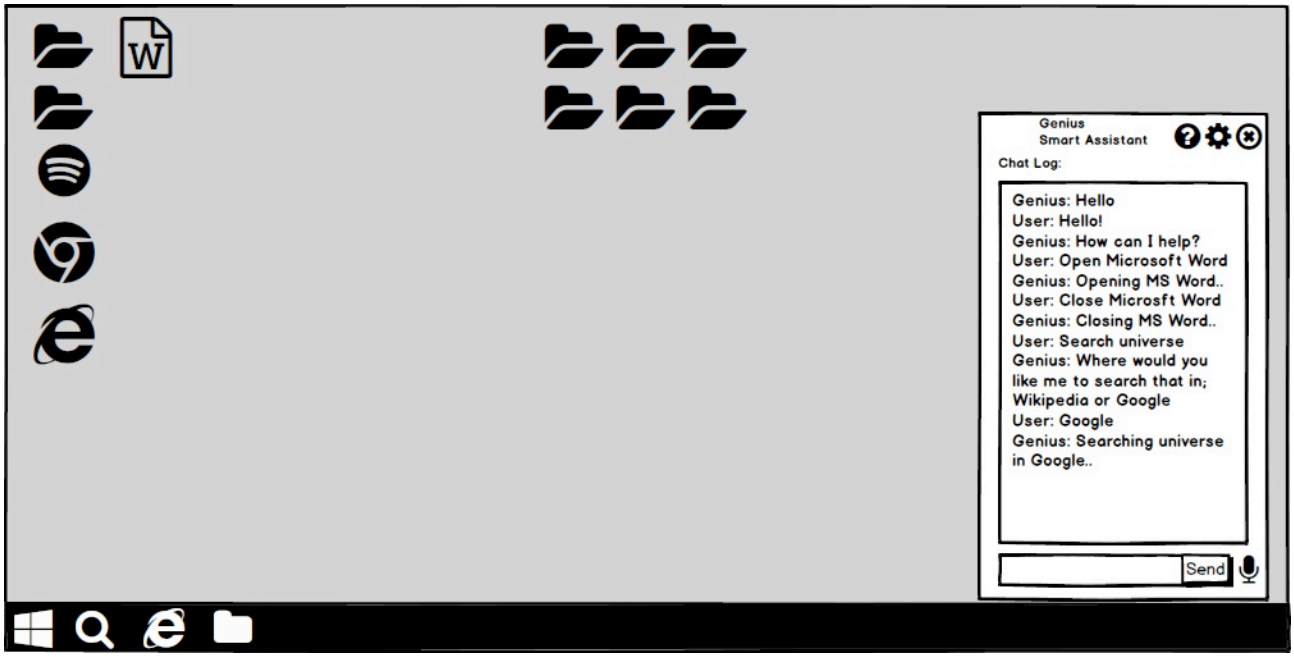

Figure 8

In this mockup of main panel, Genius' main window can be seen. When user types something

it shows in Chat Log area. User can type his/her command to input area at the below of Chat Log area. User can also click Cog icon to see settings or to question mark icon to go help section or simply click to cross icon to shutdown Genius.

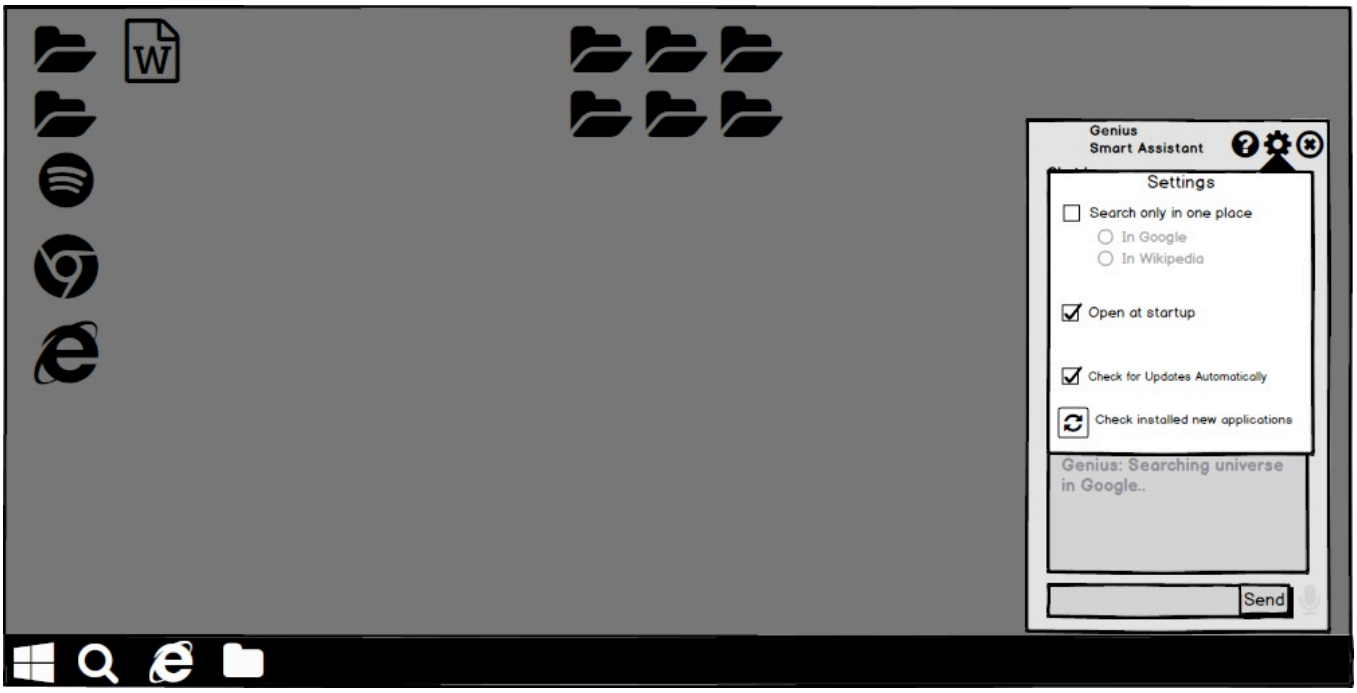

Figure 9

In this settings panel, Genius' settings can be seen. User can preset where Genius going to search input in. It can be either in Search Engine or in Wikipedia. User also can change default search engine if he/she don't want to use Google or vice versa. User can set whether Genius to open at startup or not. User also can change the default setting for Automatic update check. If user installed a new application recently, he/she can click last "Check Installed New Applications" button in order to add them to the list.

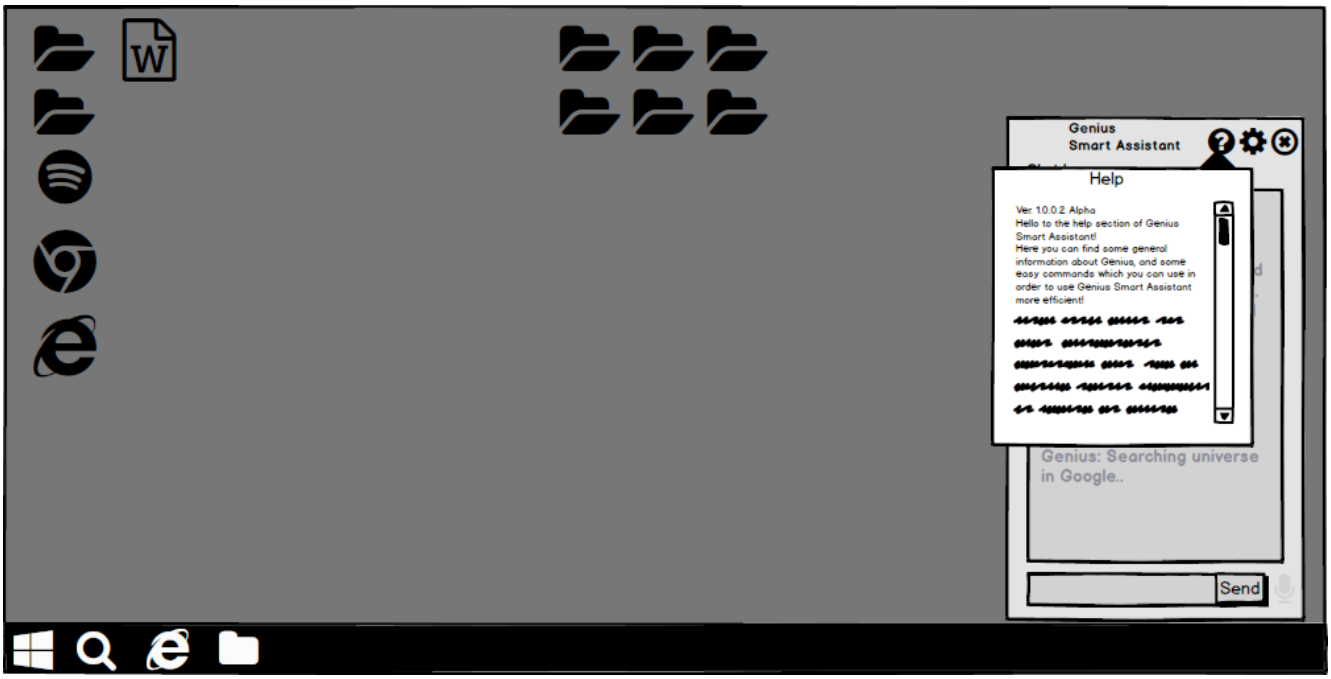

Figure 10

In this help panel, Genius' help section can be seen. In here, user can learn more about Genius. In first line, version of Genius can be seen. There also a credits section at the end where user can see the developers of the Genius. Help section is also useful for user to learn proper commands in order to use Genius more efficient.