SOFTWARE REQUIREMENTS SPECIFICATION FOR TSL-KINECT

SPONSORED BY INNOVA

D-BUG

DUYGU ARALIOĞLU 1630581

BEDİA ACAR 1630508

ZÜLFÜ ULAŞ ŞAHİN 1560796

GÜLNUR NEVAL ERDEM 1630797

Table of Contents

1. INTRODUCTION

This document is a software requirement specification of Turkish Sign Language Recognition Using Microsoft Kinect Project sponsored by INNOVA. Through this document, we are going to provide all specifications and functionalities of the project. This document will mention the functionality, that is what the resulting application is supposed to do, external interfaces which interacts the users, performance, attributes, that is if the application is portable, or maintainable, and design constraints imposed on the implementation such as implementation language, input specifications and output expectations.

1.1. Problem Definition

The main problem that our project aims to solve is the communication problem between speech-impaired people and the others. As those people cannot express themselves with the words, they have many difficulties during their daily life. Since almost all of the normal people do not know sign language and cannot understand what speechless people mean by their special language, tasks such as shopping, settling affairs at a government office are so difficult that speech-impaired people cannot handle by their own.

This problem is very broad and many solutions and implementation can be raised. A solution could be teaching sign language to everyone, yet it is very obvious to be an inefficient and even non-applicable one. Since speechless people can understand other people by lip-reading (or even hearing since being speechless does not mean being deaf also) the main problem is that normal people do not understand them. Thus a more suitable solution should be in the manner that makes speechless people’s language understandable by the other people. For the language of speechless people to be understood by others, we need the gestures performed by speech-impaired people to be recognized and turned into a form that the others can understand.

In this project we will use Kinect as a solution to this problem. Kinect for Xbox

360, or simply Kinect (originally known by the code name *Project Natal*), is a motion sensing input device by Microsoft for the XBox 360 video game console. Based around a webcam-style add-on peripheral for the Xbox 360 console, it enables users to control and interact with the Xbox 360 without the need to touch a game controller, through a natural user interface using gestures and spoken command. [5] Thus Kinect's image recognition capability can be a very promising solution fo our problem.

The system that solves our problem will supply the gestures of the speech-impaired people as the input to Kinect, then take the output from Kinect and turn it to a form that ordinary people can understand. The image processing part of gesture recognition will be done by Kinect which will reduce the problem to classifying the outputs and matching them with their meanings.

1.2. Purpose

The aim of this document is to specify the features, requirements of the final product and the interface of TSL recognition using Microsoft Kinect.

It will explain the scenario of the desired project and necessary steps in order to succeed in the task. To do this throughout the document, overall description of the project, the definition of the problem that this project presents a solution and definitions and abbreviations that are relevant to the project will be provided.

The preparation of this SRS will help consider all of the requirements before design begins, and reduce later redesign, recoding, and retesting. If there will be any change in the functional requirements or design constraints part, these changes will be stated by giving reference to this SRS in the following documents.

1.3. Scope

The project which is going to be presented in this document is called Turkish Sign Language Recognition via Microsoft Kinect. This application is planned to be used by speechless people in order to ease their life, and also government offices that should serve all of its citizens equally,private companies that want to reach and serve speechless people as well, coperations and foundations which aims to help speech-disordered people.

The application will work as the person whose gestures to be recognized (called “user” here after) stand in front of the Kinect XBox. For Kinect to specify the special points on the body, the user should stand in a certain position in order to Kinect track his/her body.After tarcking the user’s body, Kinect is ready to use.It will provide an output of what it sees in front of its camera in the form of the orientations of joints on the body.

The project will use the output come from Kinect and will match them with the pre-defined gestures.In order to define gestures, we will use Kinect again, and store the output of each gesture to a database.After this when a user performes a gesture, the program will try to determine and match it with a pre-defined gesture.If succesful , it will give the meaning of the gesture in text format.

In this project it is aimed to recognize Turkish Sign Language and translate it into a text in Turkish only. Other languages will neither be considered as input language(the sign language) nor as output language(the language that the output text in).

Obviously, our program will have some limitations like gestures based on finger movements so, the program will not cover abstract words of TSL. For the beginning the program will recognize only 10 gestures pre-defined. Yet, it will be possible to define more gestures once it is proved that the program works well enough.

The program will work on PC environment mainly. By connecting Kinect to PC, the user will be able to use our program to convert TSL to text and/or use it to learn TSL.

1.4. User and Literature Survey

There is a conflicting information on how many hearing-impaired people in Turkey. According to the report of the United Nations, this number is 2.5 million. However, according to the Ministry of National Education number of the hearing-impaired in our country is only 400,000. According to the same report, 120.000 of this population are children of whom only 7.000 go to school. [4] There is no information on the number of speech-impaired people separately.Considering these numbers, the need for a TSL recognizer is obvious.

Since Kinect is a new technology, software development environment for it is not well progressed yet and the projects via Kinect are still in its infancy. As far as we researched on the subject, we haven't encounter any commercial product for TSL Recognition. However, there are several student projects about gesture recognition by using Kinect in other languages like American, Arabic and French etc. CopyCat is a platform to collect gesture data for American sign language (ASL) recognition system. This project serves the same purpose with the project explained in this document. It is a practical application that aims to help deaf children develop working memory and improve language skills by using Kinect skeleton features . Another project that we have encountered was developed for French Sign Language. The thing that takes our attention to this project was the fact that it builds a neural network and "self-learning" aspects of the project will make it more and more functional without any more coding.

There is no such a claim that our project is completely different from the existing ones with respect to using technologies and methods, however the uniqueness of the project presented in this document comes from being the first one developed for Turkish Sign Language.

1.5. Definitions and Abbreviations

SRS: Software Requirements Specification

TSL : Turkish Sign Language

SDK: Software Development Kit

NUI: Natural User Interface

FANN: Fast Artificial Neural Network Library

1.6. References

[1] IEEE Std 830-1998: IEEE Recommended Practice for Software Requirements Specifications

[2] The site of Fast Artificial Neural Network library http://leenissen.dk/fann/wp/

[3] The Kinect XBox official site http://support.xbox.com/en-US/kinect/getting-started

[4] This site includes the statistics of deaf people percentage in Turkey , http://uretim.meb.gov.tr/egitekhaber/s83/bilisimdunyasi.htm

[5] The information about Kinect in Wikipedia, http://en.wikipedia.org/wiki/Kinect

[6] Software process model http://en.wikipedia.org/wiki/Iterative_and_incremental_development

1.7. Overview

In the following section of this document, we will focus on overall description of the system. This part explains the product perspective, product functions and constraints, assumptions and dependencies of the desired application. Third part of the document explains the specific requirements of the system. Specific requirements is divided into three parts namely interface requirements, functional requirements and non-functional requirements of the system. The fourth section of this document is data model and description part. This chapter will include information and data domain of the system and its organization. The behavioral model and its descriptions are provided in the fifth part including the state diagrams. After the behavioral model the team structure and introduction of the team members who are responsible from the project will be stated in the next section. Final part is the conclusion part which gives a brief summary of the whole document.

2. OVERALL DESCRIPTION

This part gives information about product perspective, product functions and constraints, assumptions and dependencies respectively.

2.1. Product Perspective

TSL-Kinect will serve to speech-impaired people to learn TSL easily and not to have difficulty in communication by providing their gestures converting to the text for people who does not know TSL.

There will be two functionalities which are for communication and education. The communication module is for translating gestures to the text. Gesture will be searched among gestures that have been introduced to the system. The education module will be some kind of game and has a menu which shows phrases/words correspond to pre-defined gestures. User will choose one of them and perform the gesture. Then, whether the gesture performed correctly or not, user will get a feedback from this interface.

In terms of hardware, only PC and Microsoft Kinect is necessary. Gestures will be captured by Microsoft Kinect so no image processing tasks required. Our product aims to work in PC environment.

In terms of software, we will choose Microsoft official SDK. The constraints are:

- Operating System: Windows 7 or Windows 8 Developer Preview
- Programming Language: Visual Basic, C++ or C#
- Development Environment: Microsoft Visual Studio 2010

2.2. Product Functionalities

TSL-Kinect will have two main functionality; namely communication and education. Both of these modules have two common stages:

- Capturing the body movements

Microsoft Kinect Technology provides the real time position and depth information about multiple points in human body. Kinect provides us x, y and z coordinates of 24 joint points of the user.

- Associating gestures with the corresponding words/phrases.

The program will compare the movement performed by the user with predefined movements in its database.

The difference between two modules is their output forms.Education module decides whether the gesture is performed correctly and returns a true/false value whereas the communication module displays the meaning of the performed gesture.

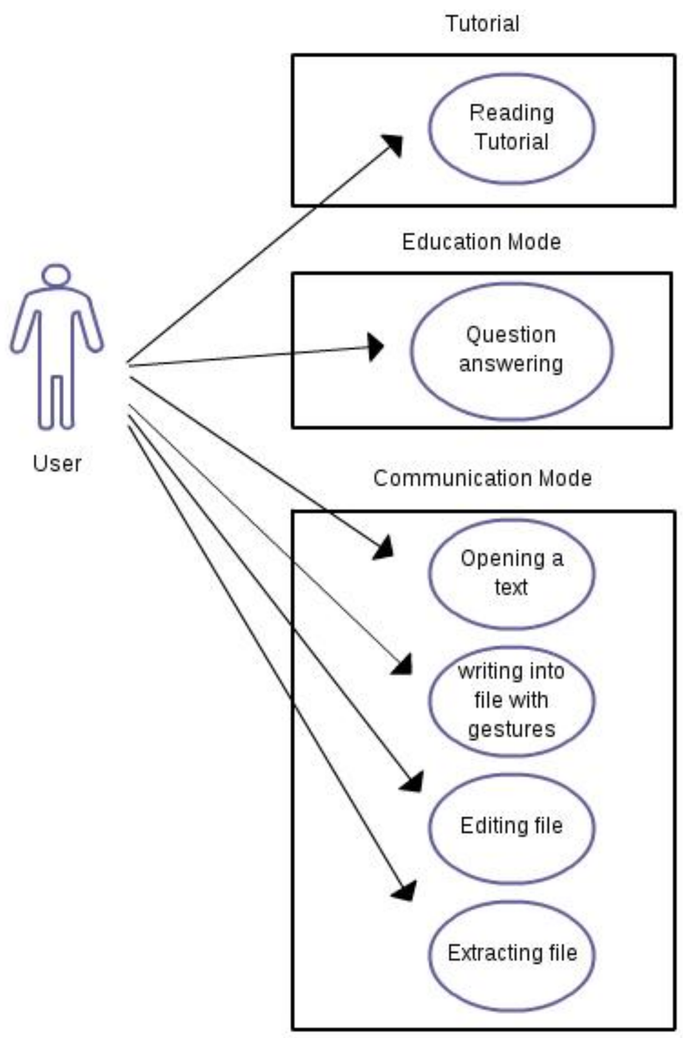


Figure 1 : Use Case Diagram

2.3. Constraints, Assumptions and Dependencies

In order to maintain the accuracy, following assumptions are made:

- The user should stand between 1.8 to 2.4 meters away from the sensors of Kinect.[http://www.xbox.com/en-US/Kinect/GetStarted]
- The Kinect should be properly set up according to its manual.
- The room that the Kinect is set up should be large enough and there must not be many furniture around in order to avoid noise.
- The body tracking phase should be properly done by the user.
- The program will work on PC environment mainly.

The reactivity of the program depends on the Kinect's recognition ability. The application does not have any safety and security concerns.

3. SPECIFIC REQUIREMENTS

The specific requirements of this project will be considered in following three subsections, which are interface, functional and non-functional requirements.

3.1. Interface Requirements

User interface will contain one main menu and two submenus. When the user click the installed executable icon of the program, the main menu will be displayed first. In the main menu there will be four buttons. One button is for the education part of the program, one is for the communication part, and the other two buttons are for quiting the program and showing the tutorial. Tutorial will be a short text document which explains to the user how program will be used. User will select the desired function by mouse-clicks. In addition, there will also be a "Çıkış" button in main menu in order to leave the application. If the user clicks the "Eğitim" button, user will be directed to the education mode screen. In this screen there will be three buttons. This module will display the meaning of a pre-defined gesture randomly by clicking "Yeni" button. "Geri" button will be used to return to the main menu. If the user clicks the "İletişim" button, user will be directed to the communication mode screen. In this screen there will be six buttons. In this module, the user will perform TSL gestures and will get the meaning of the gesture on the screen if there is a match.These meanings will be written one after another on a text field. If the user wants to save this text in a file “Dışa Aktar” button will be used. Besides, user can also load an existing text file by clicking “Aç” button. “Temizle” button will be used to clear this text field. “Geri” button is also available in communication module and its function is the same with the one in education mode.

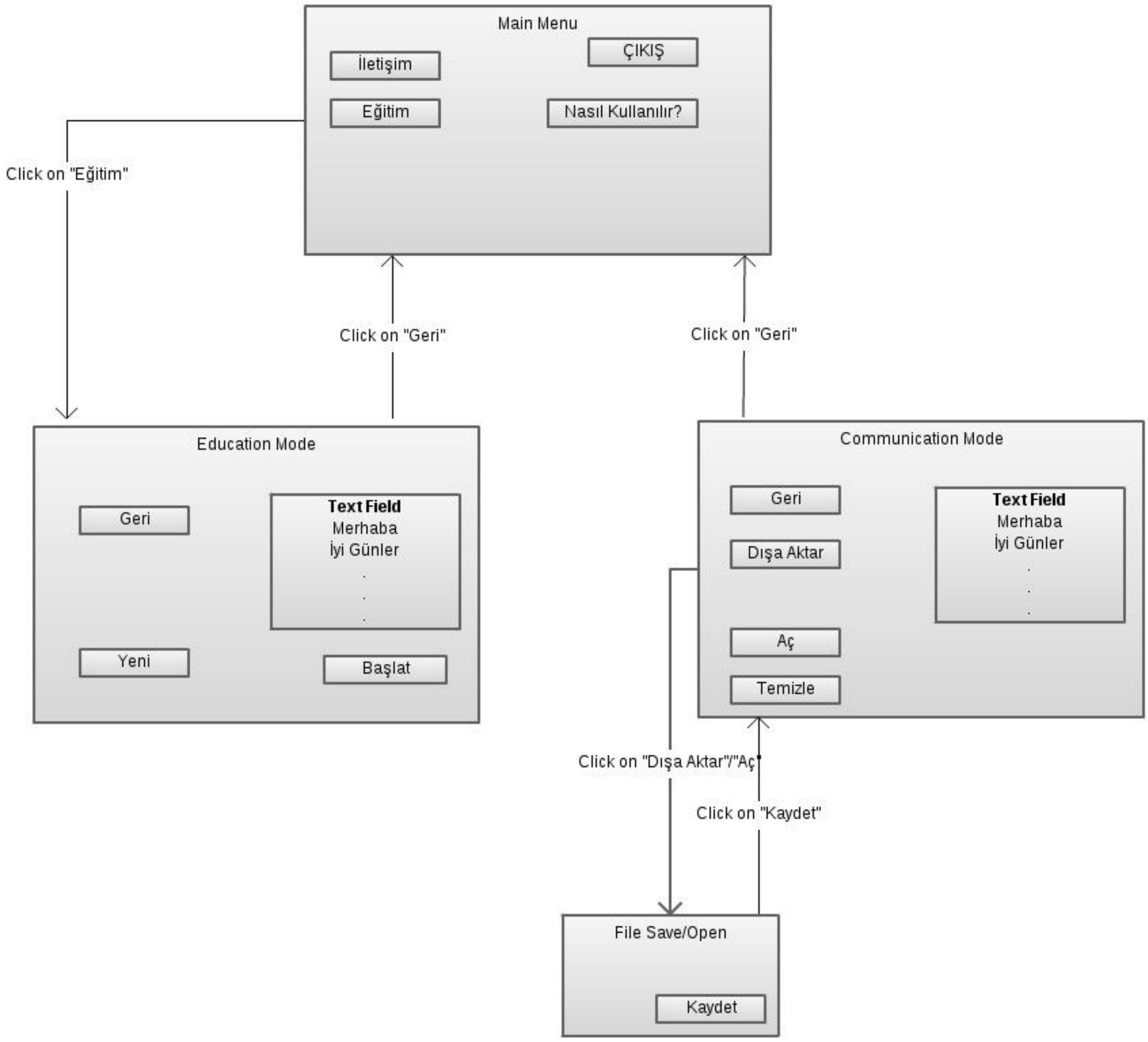


Figure 2: Interface Relations Diagram

3.2. Functional Requirements

3.2.1. Functional Requirements for Communication Mode

3.2.1.1. Background Information

This feature will translate the recognized gesture into the textual meaning of the gesture and display the translated text to the user.

3.2.1.2. Description

System will recognize the appropriate movement of the body parts and will search its database to match the movement with the pre-defined gestures. After matching system will add the meaning of the sign to the opened file.

3.2.1.3. Stimulus and Priority

Data Flow:

Normal Flow of Events:

1) User selects the communication mode from the main menu
2) User opens a file
3) User performs the movement
4) Gesture is recognized and has a match
5) The text is added to the file and displayed to the user.

Alternative Event Flow-I:

4) Gesture could not be recognized due to background noise or standing out of the Kinect's distance range.
5) User gets a feedback explaining the correct position or existence of noise in the background.

3.2.1.4. Functional Requirements

REQ-1: Performed movement must be a pre-defined movement in order to get the meaning of it as return value.

REQ-2: User should be positioned 6 (1.8 m) to 8(2.4 m) feet away from the sensor.

3.2.2. Functional Requirements for Education Mode

3.2.2.1. Background Information

This feature will give the text first and then expecting the corresponding gesture to be performed by the user, the program will try to recognize the gesture done by the user and telling whether it is performed correctly or not.

3.2.2.2. Description

This functionality will use the same pattern recognition with communication mode. After user performed the asked movement, the system will compare the performed movement with the desired movement. As a result of this comparison system will return a text message according to the correctness of the movements. This text will be written to the file and displayed to the user.

3.2.2.3. Stimulus and Priority

Data Flow:

Normal Flow of Events:

1) User selects the education mode from the main menu.
2) The meaning of the sign is displayed.
3) User performs the asked sign.
4) Gesture is recognized and comparison is done.
5) User gets a feedback about the accuracy of the movement.

Alternative Event Flow-I:

4) Gesture could not be recognized due to background noise or standing out of the Kinect’s distance range
5) User gets a feedback explaining that the correct position and existence of noise in the background.

3.2.2.4. Functional Requirements

REQ-1: User should be positioned 6 (1.8 m) to 8(2.4 m) feet away from the sensor.

3.2.3. Functional Requirements of User Interface Controlling

In this part the functionality of all the features provided by interface, in other words all the buttons in main menu and sub-menus will be explained.

3.2.3.1 Tutorial

3.2.3.1.1 Background Information

This feature will display an uploaded text document which describes how the program will be used and inform user about the necessary pre-conditions in order to succeed while using the program.

3.2.3.1.2 Description

Showing the tutorial will be done in the main menu by clicking the "Nasıl Kullanılır?" button.

3.2.3.1.3 Stimulus and Priority

Data Flow:

Normal Flow of Events:

1) User clicks the "Nasıl Kullanılır?" button in the main menu.
2) Manual for how to use is displayed.
3) User clicks the close icon right corner of the manual text and return to the main menu.

Alternative Event Flows:

There is no alternative event flow for this functionality.

3.2.3.1.4 Functional Requirements

There is no specific functional requirement for this feature.

3.2.3.2 Quit

3.2.3.2.1 Background Information

The user will be able to exit the application whenever he wants.

3.2.3.2.2 Description

Quiting the system will be done by clicking the "Çıkış" button which is placed in the main menu.

3.2.3.2.3 Stimulus and Priority

Data Flow:

Normal Flow of Events:

1) User clicks the “Çıkış” button in the main menu.

2) Application is closed.

Alternative Event Flows:

There is no alternative event flow for this functionality.

3.2.3.1.4 Functional Requirements

There is no specific functional requirement for this feature.

3.2.3.3 New

3.2.3.3.1 Background Information

This feature is for displaying a new random gesture meaning and it will be available only in education mode.

3.2.3.3.2 Description

User will be able to perform another gesture whenever he wants by clicking the “Yeni” button.

3.2.3.3.3. Stimulus and Priority

Data Flow:

Normal Flow of Events:

1) User clicks the “Yeni” button in the education mode.

2) A new string for the meaning of gestures is displayed on the screen.

3) User perform the movement corresponding to the displayed meaning.

4) After user performance result is displayed on the screen.

Alternative Event Flows:

There is no alternative event flow for this functionality.

3.2.3.1.4 Functional Requirements

There is no specific functional requirement for this feature.

3.2.3.3 Export to File

3.2.3.3.1 Background Information

This feature is for exporting the current opened file into a text file for later usage.

3.2.3.3.2 Description

User will be able save text format of the performed TSL whenever he wants.

3.2.3.3.3. Stimulus and Priority

Data Flow:

Normal Flow of Events:

1) User clicks the “Dışa Aktar” button while in the communication mode.
2) File browser window appears.
3) User fills the destination and file-name field of the browser.
4) File is saved successfully and user return to communication mode.

Alternative Event Flows:

3) File cannot be saved due to existence of same file-name.
4) User changes the file-name and retry.
5) File is saved successfully and user return to communication mode.

3.2.3.1.4 Functional Requirements

There is no specific functional requirement for this feature.

3.2.3.3 Open

3.2.3.3.1 Background Information

This feature opens an existing file.

3.2.3.3.2 Description

User will be able to open an existing file in order to manipulate the file via TSL.

3.2.3.3.3. Stimulus and Priority

Data Flow:

Normal Flow of Events:

1) User clicks the "Aç" button icon while in communication mode.
2) File browser window appears.
3) User browse the desired file.
4) File is successfully opened and ready to additions with respect to names of the performed gestures.

Alternative Event Flows:

There is no alternative event flow for this functionality.

3.2.3.1.4 Functional Requirements

REQ-1: File that is desired to be opened should exists.

REQ-2: File that is desired to be opened should be in appropriate format for the application.

3.2.3.3 Clear

3.2.3.3.1 Background Information

This feature is for deleting the text file.

3.2.3.3.2 Description

User will be able to delete the meanings of recently performed gestures.

3.2.3.3.3. Stimulus and Priority

Data Flow:

Normal Flow of Events:

1) User clicks the “Temizle” button icon in the communication mode.

2) The text field on the screen is cleared.

Alternative Event Flows:

There is no alternative event flow for this functionality.

3.2.3.1.4 Functional Requirements

There is no specific functional requirement for this feature.

3.2.3.3 Back

3.2.3.3.1 Background Information

This feature is for returning to the main menu.

3.2.3.3.2 Description

User will be able use back icon in both communication and education mode. After clicking to the back icon system will be redirected to the main menu.

3.2.3.3.3. Stimulus and Priority

Data Flow:

Normal Flow of Events:

1) User clicks the back icon while in the communication mode or education mode.

2) User return to the main menu.

Alternative Event Flows:

There is no alternative event flow for this functionality.

3.2.3.1.4 Functional Requirements

There is no specific functional requirement for this feature.

3.3. Non-Functional Requirements

3.3.1. Performance Requirements

Performing TSL via Kinect should be used by a single person. Our system should run on 32 bit (x86) or 64 bit (x64) Dual-core 2.66-GHZ or faster processor. It should not exceed 2 GB RAM.

3.3.2. Design Constraints

For the time being, we will use Microsoft Kinect SDK to control the input stream coming from Kinect and C# programming language in the Visual Studio & .NET environment will be used together with several additional libraries such as NUI for skeleton tracking and FANN for neural network implementation. Software architecture will be based on real time continuous gesture recognition methods. However these are not strictly decided constraints, there will be some changes in SDK or programming language etc. Any alteration that we made in the design constraints will be mentioned in the following documents.

4. DATA MODEL AND DESCRIPTION

In this section, we give information about data models of the project.

4.1. Data Description

In this part, we will give information about the data objects related to this project, the relationship among them, the attributes of the data objects and the complete data model with functions of data objects.

4.1.1. Data Objects

There are 6 data objects namely Input Handler, Recognizer, Interface, Database, Communication and Education.

- **Input Handler object:** This object will take joint coordinates from Kinect Sensor with related library functions. It will arrange I/O strategies of Kinect input stream, eliminate poor quality inputs, and order continuous skeleton coordinates and their time data into a format that recognizer will work on it.

- **Recognizer object:** This object takes arranged skeleton information from Input Handler Object and tries to match them with pre-defined gestures in the Database Object.
- **Interface object:** This object contains interface information, namely information of the main menu to build the corresponding user interface.
- **Database object:** This object contains datasets of pre-defined gestures.
- **Education object:** This object will randomly generate gesture questions to user and give a time to performance to the user. If user can perform gesture correctly in the specified time, it will give information to Interface object to display the result in the form of correct/wrong.
- **Communication object:** This object will take determined gestures from recognizer than it will add appropriate strings to currently opened file which is displayed on the screen. Also this object can delete words from displayed file, open new file, save and export current one.

4.1.2. Relationships

Input handler object contains a list of upper body joint positions and time intervals of the performed movements. This list will be sent to the recognizer object to see if there is a match with the pre-defined gestures(input_generator()). Recognizer object will initialize itself by using Database object (neural_net_initializer()) and after that it will perform recognition on transformed position stream that comes from Input Handler(recognize()) and send detected gesture names to the current active module. User Interface object will regulate menus/submenus (current_mode()), and will display what education and communication objects dictate. Two objects, Education and Communication will activate Input Handler (activate_input_handler()) and use the names of gestures that comes from Recognizer(set_gesture_name()).

4.1.3. Complete Data Model

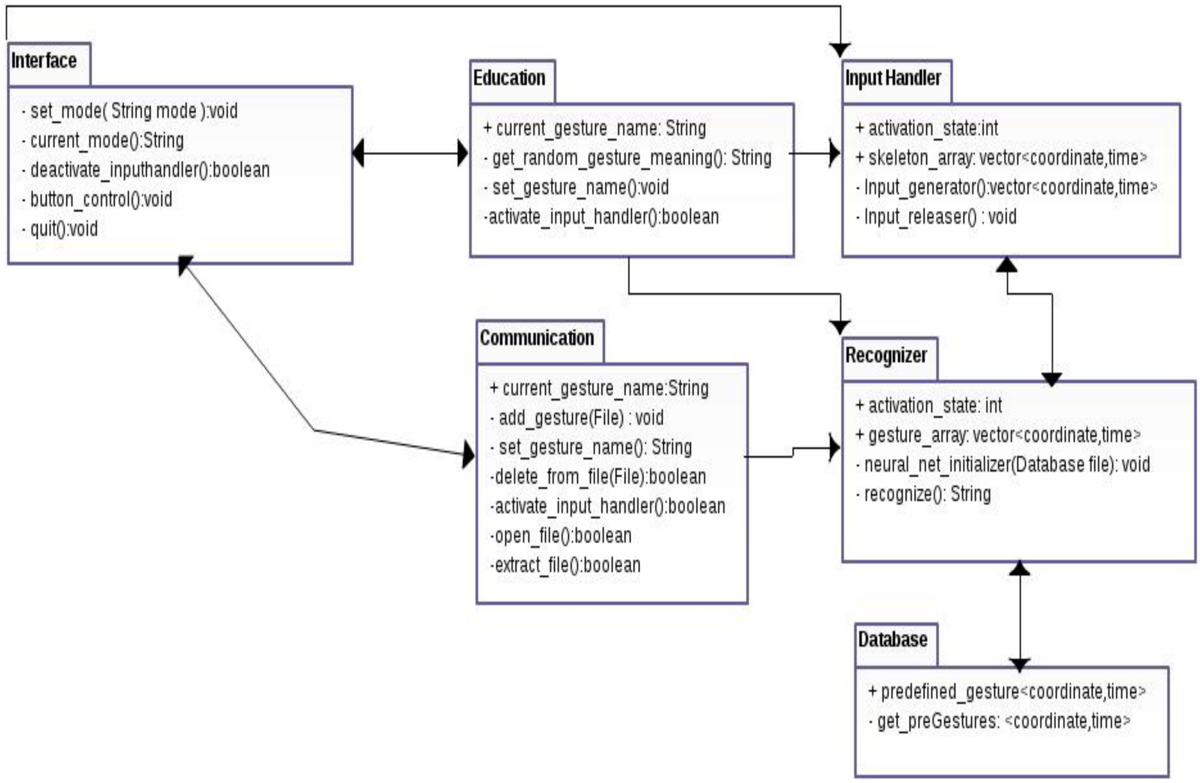


Figure 3: Class Diagram

4.1.4. Data Dictionary

Skeleton Array: The data array that stores all the time values and the corresponding joint positions.

Current Gesture Name: It contains last received gesture name.

Activation State: It is a control variable for checking if a module is active or not.

Predefined Gesture: It is an array resides in Database created for initializing neural network.

5. BEHAVIORAL MODEL AND DESCRIPTION

5.1. Description for Software Behavior

When the program is started only the Interface module is initiated at the beginning. This module controls all other modules and buttons. After user selects one of the communication or education mode in the main menu, corresponding module will be activated; and also this module will trigger input handler and recognizer to start operation. Input handler will start to process skeleton position stream that comes from the Kinect SDK only if one of the communication or education mode is active. While input handler is in operation, it will constantly transform skeleton position stream and send positions-related times arrays to recognizer. Also, it will release saved skeleton positions when enough time is passed or recognizer sent a signal when a match is occurred. Recognizer will take transformed position-time information from input handler and tries to match them with predefined gestures. If matching is successful, it will send corresponding gesture name string to current active module (education or communication), and a warning signal to input handler that tells which part of input stream is matched with a gesture in order to releasing operation. Communication module's essential operation is to open file, save new words which comes from recognizer and give content of current file to interface object to display. Education module will choose random names of predefined gestures and send them as a question to interface object. It will count some time, and wait for recognizer to give it the same name with question asked.

Interface object will handle all visualization operations of the program. When a submenu is opened, corresponding module will be initialized by this module. Also, if active mode creates some string to visualize, interface object will take and show it. All the behaviors of the program can be understood or described by the changes of the interface.

5.2. State Transition Diagram

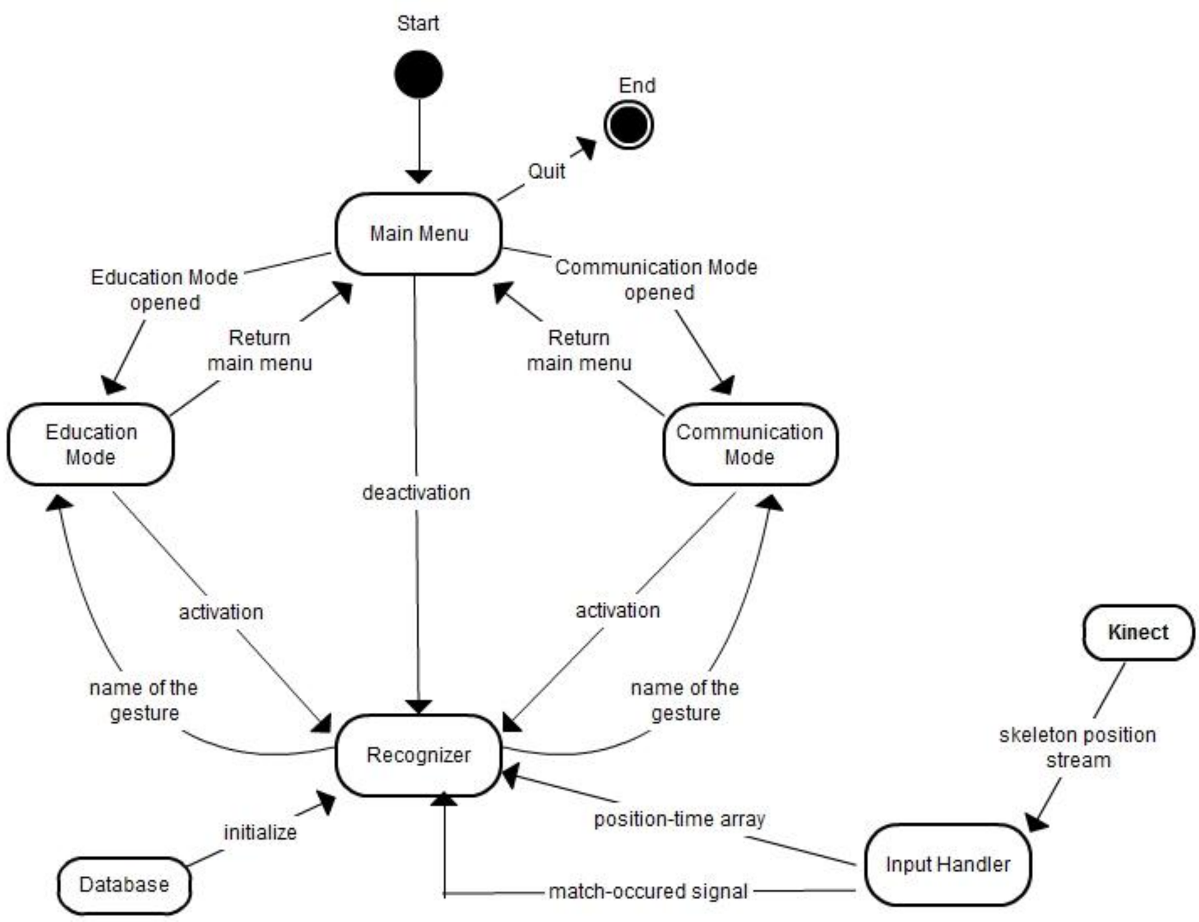


Figure 4: State Diagram

6. PLANNING

6.1. Team Structure

Duygu ARALIOĞLU: Researcher,Software Engineering

Bedia ACAR: Researcher,Software Engineering

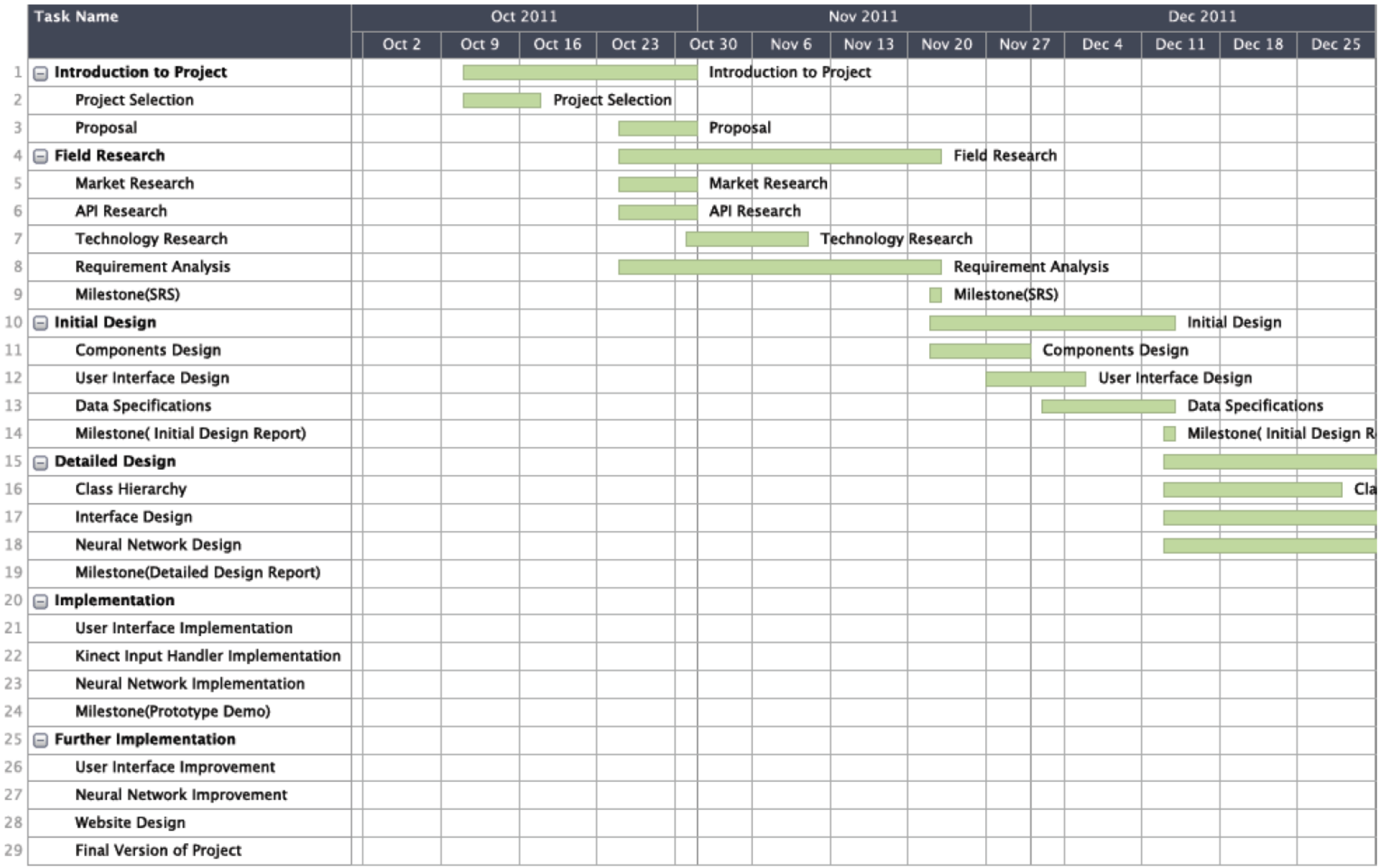
Ulaş ŞAHİN: Researcher,Software Engineering

Gülnur Neval ERDEM: Researcher,Software Engineering

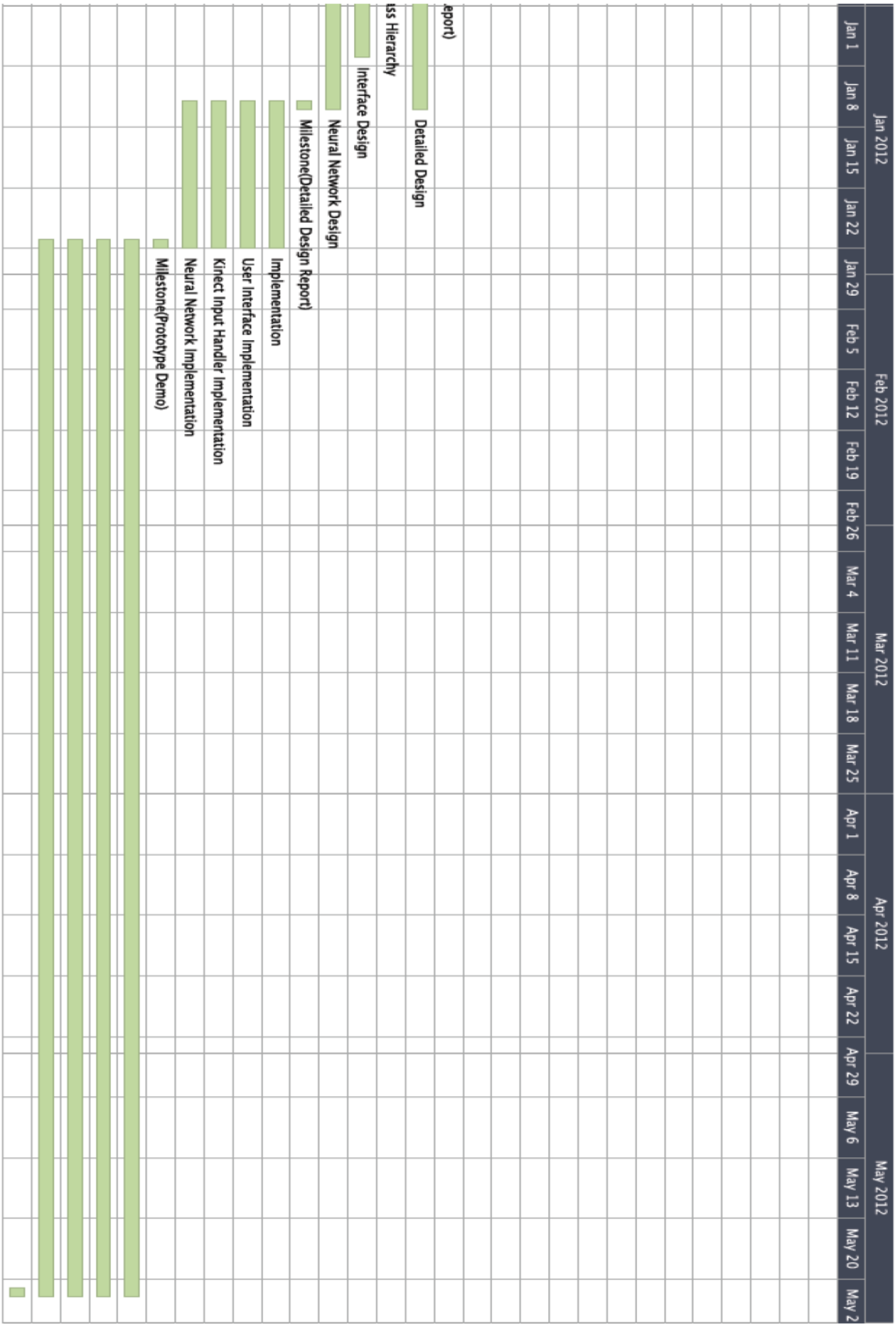
In our team every member's opinion has the equal importance and priority. Thus, our team has "Democratic de-centralized" structure. Literature research will be done by all of the group members in order to create discussion platform. Since every member's experience and knowledge of developments tools are nearly the same and everyone will contribute the development of each module. Modules will be divided into segments on their own and every one will do his/her own part.

6.2. Estimation (Basic Schedule)

The Gantt chart that shows the tentative schedule of the project is as follows with the tasks and dates included:



Task Name
Oct 2011
Nov 2011
Dec 2011
Oct 2
Oct 9
Oct 16
Oct 23
Oct 30
Nov 6
Nov 13
Nov 20
Nov 27
Dec 4
Dec 11
Dec 18
Dec 25
1 Introduction to Project
2 Project Selection
3 Proposal
4 Field Research
5 Market Research
6 API Research
7 Technology Research
8 Requirement Analysis
9 Milestone(SRS)
10 Initial Design
11 Components Design
12 User Interface Design
13 Data Specifications
14 Milestone(Initial Design Report)
15 Detailed Design
16 Class Hierarchy
17 Interface Design
18 Neural Network Design
19 Milestone(Detailed Design Report)
20 Implementation
21 User Interface Implementation
22 Kinect Input Handler Implementation
23 Neural Network Implementation
24 Milestone(Prototype Demo)
25 Further Implementation
26 User Interface Improvement
27 Neural Network Improvement
28 Website Design
29 Final Version of Project


Jan 2012
Feb 2012
Mar 2012
Apr 2012
May 2012
Jan 1
Jan 8
Jan 15
Jan 22
Jan 29
Feb 5
Feb 12
Feb 19
Feb 26
Mar 4
Mar 11
Mar 18
Mar 25
Apr 1
Apr 8
Apr 15
Apr 22
Apr 29
May 6
May 13
May 20
May 2
eport)
Detailed Design
ss Hierarchy
Interface Design
Neural Network Design
Milestone(Detailed Design Report)
Implementation
User Interface Implementation
Kinect Input Handler Implementation
Neural Network Implementation
Milestone(Prototype Demo)

6.3. Process Model

After our research on the process model we decided on Iterative and Incremental Model. Since Waterfall processing model discourages revisiting and revising any prior phase once it's complete we do not want choose it. We have seen that Iterative and Incremental development is at the heart of a cyclic software development process developed in response to the weaknesses of the waterfall model. It starts with an initial planning and ends with deployment with the cyclic interactions in between. It follows a similar process to the plan-do-check-act cycle.

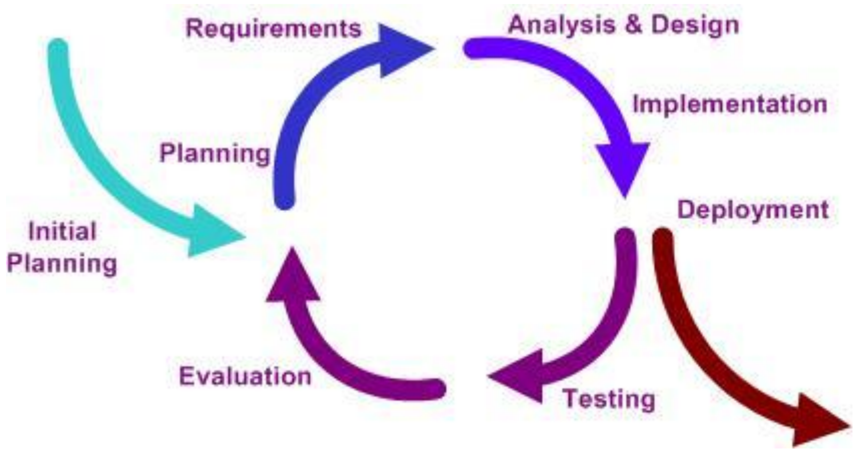


Figure 5: Iterative and Incremental Process Model

7. CONCLUSION

This software requirement specifications document gives information about the project TSL which implement Turkish Sign Language by using Kinect. We first defined the real world problem and give the explanation of our proposed solution for that. This explanation includes basic functionality of the application, interface requirements of the application, performance, attributes, and design constraints imposed on the implementation. In the overall description part, all functional, non-functional and interface requirements of the program is explained in detail. The information about data objects, their attributes and complete data model was also provided. In behavioral model part, relationships between functions and user and characteristics are modeled. Finally, the introduction of team members, expected schedule and process model of our team was covered.