System Design

Of

Gesture based Smartphone application for Visually Impaired

Version 1.0

Prepared by

Ahmad Ali FA12-BCS-008

Awais Ahmad FA12-BCS-025

Supervised by

Dr. Yasir Faheem

Department of Computer Science,

CIIT, Islamabad.

February 12, 2016.

**Table of Contents**

[1. Introduction 3](#_Toc441598601)

[2. Use Case Diagram 4](#_Toc441598602)

[3. Functional Requirements 4](#_Toc441598603)

[4. System Architecture 9](#_Toc441598604)

[5. Design Methodology 9](#_Toc441598605)

[6. Data Representation [Diagram + Description] 9](#_Toc441598606)

[7. Design Models [along with descriptions] 13](#_Toc441598607)

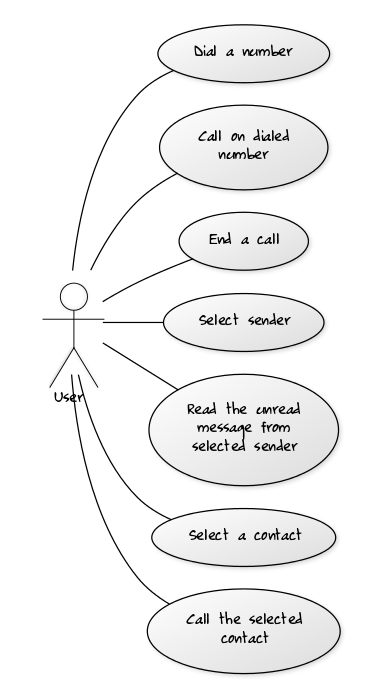
[8. Algorithm & Implementation 16](#_Toc441598608)

# Introduction

This system will be an android application which will work using tap and tilt gestures. The product will capture and manipulate these gestures and set control/functionality for every specific gesture. This project will require use of some existing API’s which will provide the system with features of speech to text, text recognition, haptic response and voice assistance. The product will give control over three list-based applications to the user which include reading unread messages, making a call and contacts. It will facilitate the user with all the basic features that are used daily. However, the visually impaired user would not be able to perform complex operations like playing games, editing a pdf document using this product. There will also be a learning phase in which the system will behave as an adaptable system (user has the control to set priority of most frequently used applications).

|  |  |
| --- | --- |
| **Modules** | **Features** |
| Training | * Sound assistance * Customization |
| Capturing Gesture | * Gesture input * Learning based mechanism for recognizing gestures * Training application according to the user |
| Action on Gesture | * Move to next and previous * Make selection * Launching applications * Sound verifications * Haptic Feedback |
| Call | * Text recognition * Sound verifications * Haptic feedback |
| Message | * Text to Speech (using and existing API) * Sound verifications * Haptic Feedback |
| Contacts | * Selecting contacts * Making call |

# Use Case Diagram



Use Case Diagram for Gesture Based Smartphone Application for Visually Impaired

# Functional Requirements

In this section all the functional requirements for the system are discussed. Every requirement is discussed individually in tabular format.

Table 1: Requirements for Open Application

|  |  |
| --- | --- |
| ID | R-1 |
| Title | Open Application |
| Requirement | When the user shakes the device (when screen is unlocked) the app gets triggered. |
| Source | Interviews and observations. |
| Rationale | Providing the user with an easy method to open app |
| Restrictions and Risk | The screen of the device should be in unlock state to perform shake operation. Secondly, there is a chance that this action is not performed if the user does not shake to the set criteria |
| Dependencies | None |
| Priority | High |

Table 2: Requirements for Setting Application Priority

|  |  |
| --- | --- |
| ID | R-2 |
| Title | Set Application Priority |
| Requirement | The user can set priority for most frequently used application |
| Source | Interviews and observations. |
| Rationale | To make the system adaptable. |
| Restrictions and Risk | None |
| Dependencies | R-1 |
| Priority | Medium |

Table 3: Requirements for Listening for Input

|  |  |
| --- | --- |
| ID | R-3 |
| Title | Listen for input |
| Requirement | Before performing an action, the system should be able to get a gesture for performing that action. So the system should be able to listen for this input (gesture) when required. |
| Source | Discussions |
| Rationale | For correct working of the system |
| Restrictions and Risk | Working accelerometer in the device |
| Dependencies | R-1 |
| Priority | High |

Table 4: Requirements for Recognizing Input Gesture

|  |  |
| --- | --- |
| ID | R-4 |
| Title | Recognize the input gesture |
| Requirement | After getting the gesture (as an input), the system should be able to perform recognition on it to identify if it is a valid gesture or not. |
| Source | Internet and Discussions |
| Rationale | For correct working of the system |
| Restrictions and Risk | The device should be in running state. Secondly, there is a chance that a gesture might not get recognized if the values of the accelerometer does not fall in the recognized gestures category. |
| Dependencies | R-3 |
| Priority | High |

Table 5: Requirements for Performing Action on Gesture

|  |  |
| --- | --- |
| ID | R-5 |
| Title | Perform action on recognized gesture |
| Requirement | After recognizing the gesture, the system should be able to perform an action against it. |
| Source | Discussions |
| Rationale | Allow user to perform the desired task |
| Restrictions and Risk | A specific action can only be performed if it is associated with a recognized gesture. |
| Dependencies | R-4 |
| Priority | High |

Table 6: Requirements for Feedback of Action performed by user

|  |  |
| --- | --- |
| ID | R-6 |
| Title | Feedback of Action performed by the user |
| Requirement | On every input, the system should provide a feedback in terms of a small vibration so that the user is notified that some instruction is received by the device. |
| Source | Discussions and Interviews |
| Rationale | For the system to be user friendly. |
| Restrictions and Risk | The device should support vibration mechanism (hardware). |
| Dependencies | R-1 |
| Priority | Medium |

Table 7: Requirements for Voice Assistance

|  |  |
| --- | --- |
| ID | R-7 |
| Title | Voice Assistance |
| Requirement | At every step, the system should assist the user with voice feedback for better understanding of the flow of events occurring in the system. |
| Source | Observation and discussions |
| Rationale | To make the system user-friendly |
| Restrictions and Risk | All the sound components of the device should be in working state. |
| Dependencies | R-1 |
| Priority | Medium |

Table 8: Requirements for Closing Application

|  |  |
| --- | --- |
| ID | R-8 |
| Title | Close Application |
| Requirement | When the user locks the device (when screen is locked) the app should close. |
| Source | Discussions |
| Rationale | Providing the user with an easy method to close app |
| Restrictions and Risk | Separate button to lock the screen (on device) should be working. Secondly, the auto lock (set on time of inactivity) should be disabled else it will perform the same task. |
| Dependencies | R-1 |
| Priority | Medium |

# System Architecture

The system architecture followed will be the MVC (Model View Controller). We place Activities in the View part of MVC, as it forms the UIs of Android applications and updates them throughout the lifecycle of the Android application. The Services and Content Provider or data are hidden components and monitoring the behavior of the application, they are placed in Model part of MVC. Widget components at kernel are acting as Controller of MVC and interacts with Models and Views. Therefore, we can separate presentation (View) from the logic of application (Model). Frontend activities rely on the XML and Java files and their interactions in Android app programming. Java files/libraries are falling in Model part of MVC, whereas XML belongs to the View part.

# Design Methodology

As it is an android application, classes are involved so we use OOP methodology. We will make classes and call the functions by creating objects. Furthermore, this system follows event driven programming. In this methodology, an action is performed of a specific event and the even listener is constantly waiting for the event to occur. The events may include a touch, gesture, button press etc.

# Data Representation [Diagram + Description]

This project is an android application and uses XML Schema format to represent data of the system.

<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified" xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="manifest">

<xs:complexType>

<xs:sequence>

<xs:element name="uses-sdk">

<xs:complexType>

<xs:simpleContent>

<xs:extension base="xs:string">

<xs:attribute ref="and:minSdkVersion" xmlns:and="http://schemas.android.com/apk/res/android"/>

<xs:attribute ref="and:targetSdkVersion" xmlns:and="http://schemas.android.com/apk/res/android"/>

</xs:extension>

</xs:simpleContent>

</xs:complexType>

</xs:element>

<xs:element name="uses-permission">

<xs:complexType>

<xs:simpleContent>

<xs:extension base="xs:string">

<xs:attribute ref="and:name" xmlns:and="http://schemas.android.com/apk/res/android"/>

</xs:extension>

</xs:simpleContent>

</xs:complexType>

</xs:element>

<xs:element name="application">

<xs:complexType>

<xs:sequence>

<xs:element name="activity">

<xs:complexType>

<xs:sequence>

<xs:element name="intent-filter">

<xs:complexType>

<xs:sequence>

<xs:element name="action">

<xs:complexType>

<xs:simpleContent>

<xs:extension base="xs:string">

<xs:attribute ref="and:name" xmlns:and="http://schemas.android.com/apk/res/android"/>

</xs:extension>

</xs:simpleContent>

</xs:complexType>

</xs:element>

<xs:element name="category">

<xs:complexType>

<xs:simpleContent>

<xs:extension base="xs:string">

<xs:attribute ref="and:name" xmlns:and="http://schemas.android.com/apk/res/android"/>

</xs:extension>

</xs:simpleContent>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

<xs:attribute ref="and:name" xmlns:and="http://schemas.android.com/apk/res/android"/>

<xs:attribute ref="and:configChanges" xmlns:and="http://schemas.android.com/apk/res/android"/>

<xs:attribute ref="and:label" xmlns:and="http://schemas.android.com/apk/res/android"/>

<xs:attribute ref="and:theme" xmlns:and="http://schemas.android.com/apk/res/android"/>

<xs:attribute ref="and:screenOrientation" xmlns:and="http://schemas.android.com/apk/res/android"/>

</xs:complexType>

</xs:element>

</xs:sequence>

<xs:attribute ref="and:allowBackup" xmlns:and="http://schemas.android.com/apk/res/android"/>

<xs:attribute ref="and:icon" xmlns:and="http://schemas.android.com/apk/res/android"/>

<xs:attribute ref="and:label" xmlns:and="http://schemas.android.com/apk/res/android"/>

<xs:attribute ref="and:supportsRtl" xmlns:and="http://schemas.android.com/apk/res/android"/>

<xs:attribute ref="and:theme" xmlns:and="http://schemas.android.com/apk/res/android"/>

</xs:complexType>

</xs:element>

</xs:sequence>

<xs:attribute type="xs:string" name="package"/>

</xs:complexType>

</xs:element>

</xs:schema>

XML SCHEMA for Gesture Based Smartphone Application for Visually Impaired

# Design Models

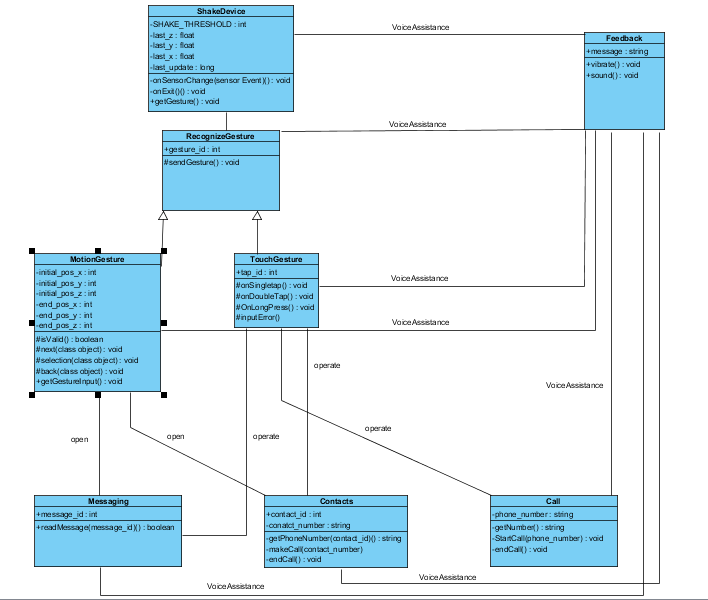
In this section design models (Class Diagram, Sequence Diagram and Activity Diagram) for the system are discussed.

# Class Diagram:

A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

# 7.1.1 Class Diagram for Gesture Based Smartphone Application for Visually Impaired

This system comprises of 8 classes in total in which the main class (shake device) is public. Furthermore Feedback class is also public which has a connection with all the other classes because feedback is used by all the classes. Furthermore there is a protected class RecogniseGesture which has 2 children MotionGesture and TouchGesture. MotionGesture further has 2 children Messages and Contacts and TouchGesture has 3 classes Call, Message and Contacts. Attributes and main function of these classes are shown below.

****

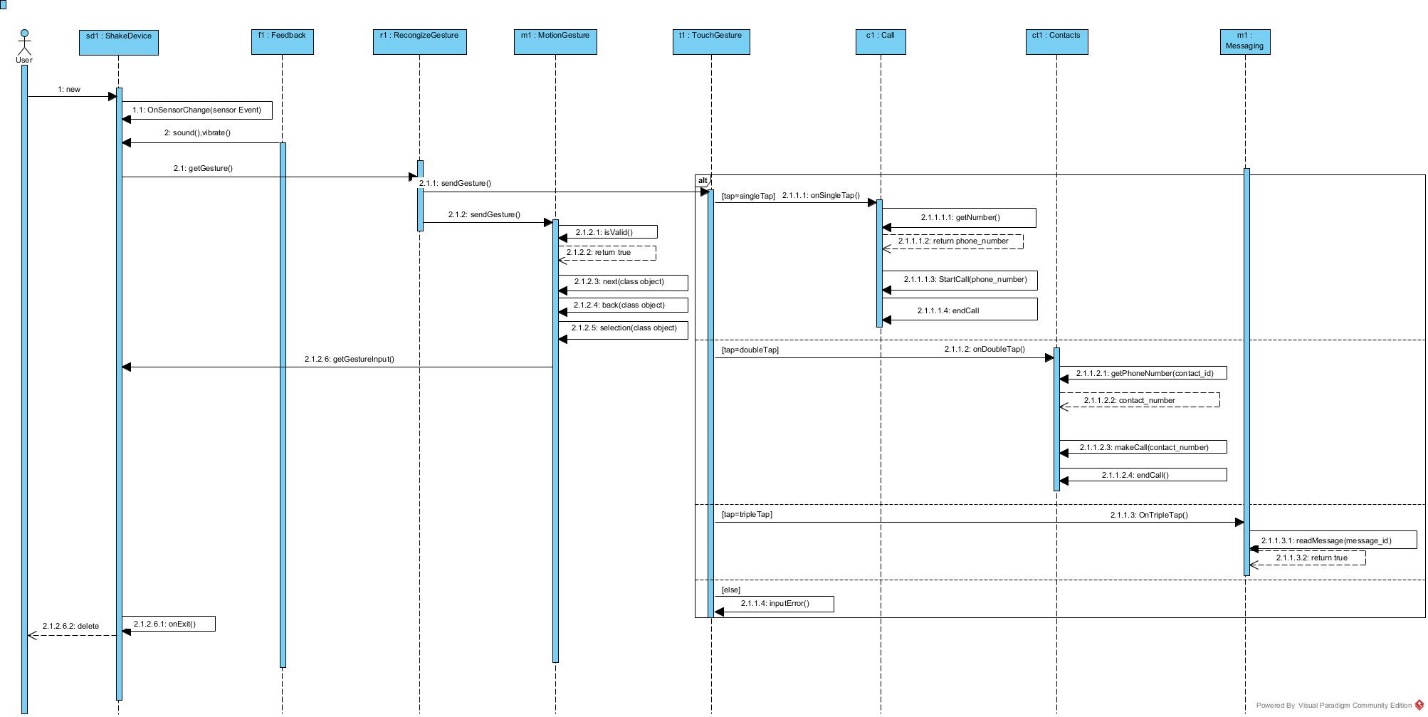
Class Diagram for Gesture Based Smartphone Application for Visually Impaired

* 1. **Sequence Diagram:**

A Sequence diagram is an interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence.

# 7.2.1 Sequence Diagram for Gesture Based Smartphone Application for Visually Impaired

At first, a new instance is created and when the device is shaken, onSensorChange() is called. At every step, feedback is called for haptic feedback and sound assistance to the user. Following this, getGesture() is called and in RecognizingGesture class a gesture is recognized and sent to the respective classes MotionGesture and TouchGesture on the basis of gesture\_id. Then, respective list-based apps (Call, Contacts or Messaging) are opened and specific functions are applied on it according to the user input (gesture) which include phone call, reading a message etc. In the end, if the device is locked (manually from the button), the instance that was first created will be deleted on onExit() call using garbage collector and the app will close.

****

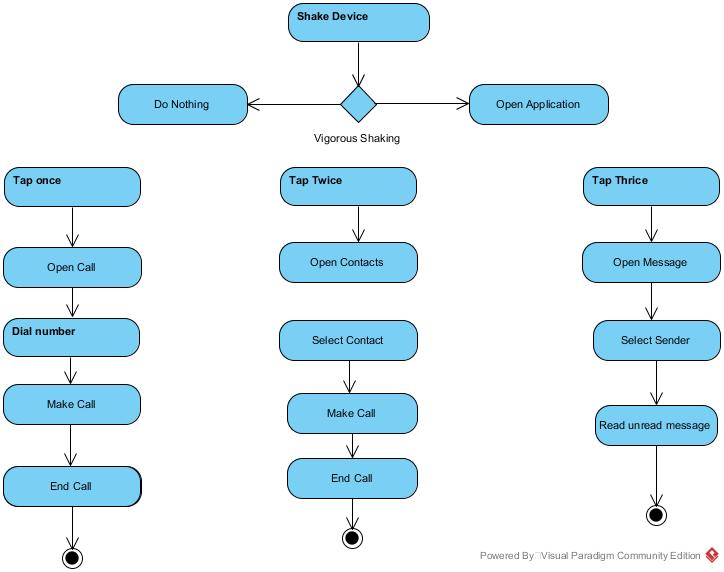
Sequence Diagram for Gesture Based Smartphone Application for Visually Impaired

* 1. **Activity Diagram:**

Activity diagram is an important diagram in UML to describe dynamic aspects of the system. Activity diagram is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system.

# 7.3.1 Activity Diagram for Gesture Based Smartphone Application for Visually Impaired

The activity diagram explains the flow of event as follows. When the screen is open (unlocked) and you shake the device (up to the defined threshold) the app starts. From the main screen of the application, you shift (open) to 3 other apps (Call, Message and Contacts) by tapping as shown in the diagram below. The 3 apps are operated using gestures.

****

Activity Diagram for Gesture Based Smartphone Application for Visually Impaired

# Algorithm & Implementation

The APIs used currently in the early stage of project implementation includes four main APIs of android which are Sensor API, Vibrator API, Widgets API and Sound API. Whereas for the algorithms to be used in the project implementation, a sensing algorithm is used. This algorithm serves the purpose of noise removal by using a low pas filter and a high pass filter, and a threshold value against noise.

When device (in unlocked state) is shaken to a level where it matches the threshold values defined in the system, the GVI application is turned on and main screen appears on the device. The user is then provided with a set of instructions for using the basic features (Call, Contacts and Messages) on tap input. According to the number of taps given by the user, the system opens the set module (One tap for Call, two taps for Contacts and three taps for Messages) and its related screen appears on device. The user is given a voice and vibration feedback for the action performed on the system and when new screen appears on device its set of instructions (voice assistance) are spoken to the user. User has options to navigate cursor (for selection) using volume keys in both Contacts and Messaging. The current cursor position is given to the user in form of voice feedback. In Contacts, if the user gives a button long press input and gives a forward gesture (tilt the device in forward direction), call is dialed to the selected contact. However in Messaging, if the user gives a button long press input and gives a forward gesture (tilt the device in forward direction), the system reads the unread message of that selected sender.

On any stage if the device is shaken vigorously, the user is taken to main screen of GVI application. To improve battery stats, the sensors are being registered on long press of the button. This improves the efficiency of the system and avoids taking unnecessary accelerometer (when not required) of the device to save battery.