Aware-D : Voice Recognition-based Driving Awareness Detection

Software Design Document

Yudi Andrean Phanama

Cécile Duthoit

Date: (04/12/2015)

Department of Electrical Engineering, Faculty of Engineering

University of Indonesia

Depok, Indonesia

2015

TABLE OF CONTENTS

1. INTRODUCTION

1.1 Purpose

This document expands the functionality described by the features in the Software Requirements Specifications (SRS) v0.2. Each feature discussed will describe the existing functionality of our voice recognition-based driving awareness detection system (Aware-D), and describe the additional classes, attributes and methods to be implemented.
In order to make the reading of this paper easier for the readers, we will consider that our main user is a man, which allows us not to need to precise “him or her”.

1.2 Scope

As already mentioned in Software Requirements Specification (SRS), Aware-D is a voice-recognition-based android application which detects if the driver of a car seems to be aware enough before driving, and while driving. The app will give simple awareness-testing questions through voice and accept answers from the driver’s voice which help to identify if the driver is still aware or not while driving, and help the driver to keep his awareness level while driving.

The user must initiate the app to start it and answer the questions provided by the app through voice and by voice. The user also has to answer periodic questions provided by the app to make sure he stays aware. The app will alert the driver if certain error threshold (quantity of wrong answer) is passed by the driver in form of alarming sound, and optional SMS sending to a chosen person.

1.3 Overview

The purpose of this document is to help the reader visualize the solution to the project presented. This document verifies how the design meets the requirements stipulated in the SRS document through design viewpoints. The design viewpoints will cover all design elements presented before.

By using information from IEEE 1016-1998, this document will provide a direct approach to the development of this project hence reducing feature creep and pointedly determine the quality of the design.

The Software Design Document is divided into 8 sections with various subsections. The sections of the Software Design Document are:

1. Introduction
2. System Overview
3. System Architecture

4. Data Design
5. Component Design
6. Human Interface Design
7. Requirement Matrix
8. Appendices

1.4 Reference Material

- *IEEE Recommended Practice for Software Design Descriptions*, IEEE Std 1016-1998 (Revision of IEEE Std 1016-1987) Available at http://web.nps.navy.mil/~nschneid/is3020/PDF/1016-1998.pdf
- Strickland, Sarah, Sample for Software Engineering Design Document
- IEEE, *IEEE 1016* Software Design Document (SDD) Template for CENG491

1.5 Definitions and Acronyms

- **DB – Database**, a structured data store
- **SDD – Software Design Description,** is a document that completely describes all of the function of a proposed system and the constraints under which it must operate.
- **SRS – Software Requirement Specification**, is a description of a software system to be developed, laying out functional and non-functional requirements, and may include a set of use cases that describe interactions the users will have with the software.
- **UML – Unified Modeling Language**, is a standard notation for the modeling of real-world objects as a first step in developing an object-oriented design methodology.

2. SYSTEM OVERVIEW

2.1 Architectural Design

The system is coded with Java programming language by using Android Studio integrated development environment. We will use SQLite for the phone database. Android SDK will be used for android application development.

2.2 Application Overview

The main goal of the project is to create a mobile app to help the user to drive safely by making sure that he is aware by monitoring him using questions asked from the app's voice recognition system. The app will ask questions to the user before driving and while the user is driving, and if any error threshold of the wrong answer for the question is exceeded, the app will alert the driver, and send SMS to the chosen contact to notify the contact that the driver is suspected not aware when driving and might be dangerous for himself and all the road users.

2.3 Design Languages

Unified Modeling Language (UML) is used for graphical representations of the viewpoints of the project in System Architecture, Data Design and Component Design parts.

3. SYSTEM ARCHITECTURE

3.1 Architectural Design

The app will be developed for Android Devices in form of Android Application package (apk). Figure 1 presents the deployment diagram. The app will use the smartphone SMS sending and will store data into the smartphone database.

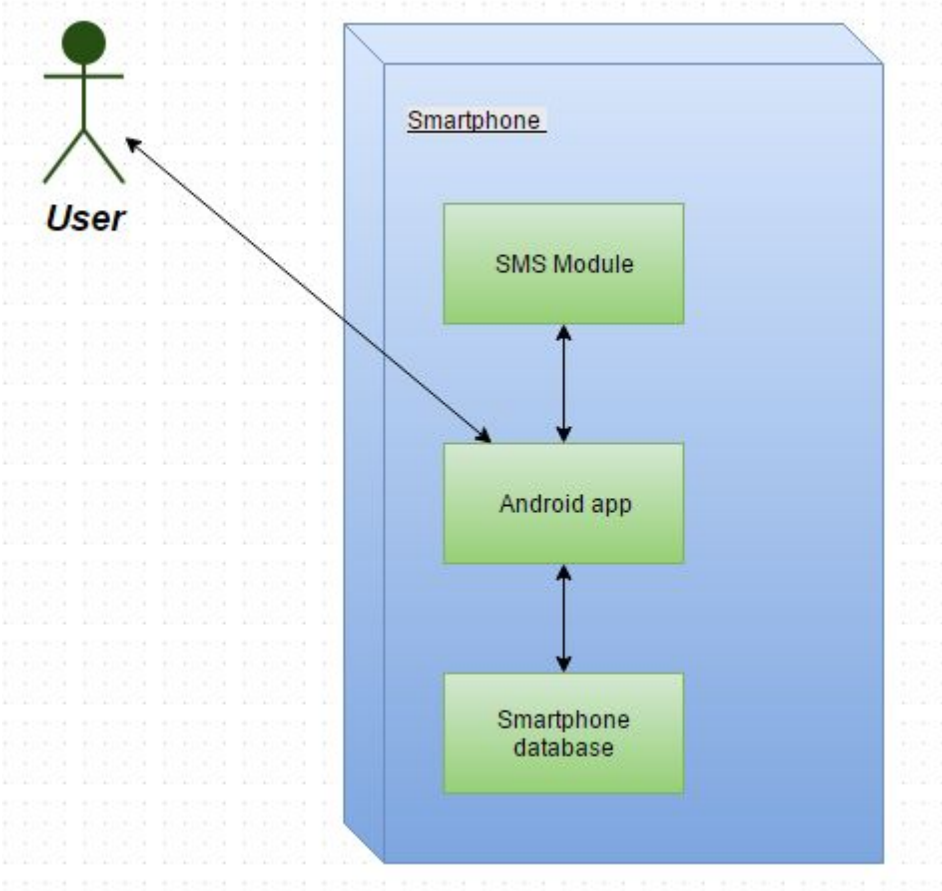

Figure 1. DATS Deployment Diagram

3.2 Decomposition Description

The following section describes in sequence diagram the components design for the system. Every interaction is hierarchically drawn from the user to the parts of the system.

3.2.1 Questioning System Component

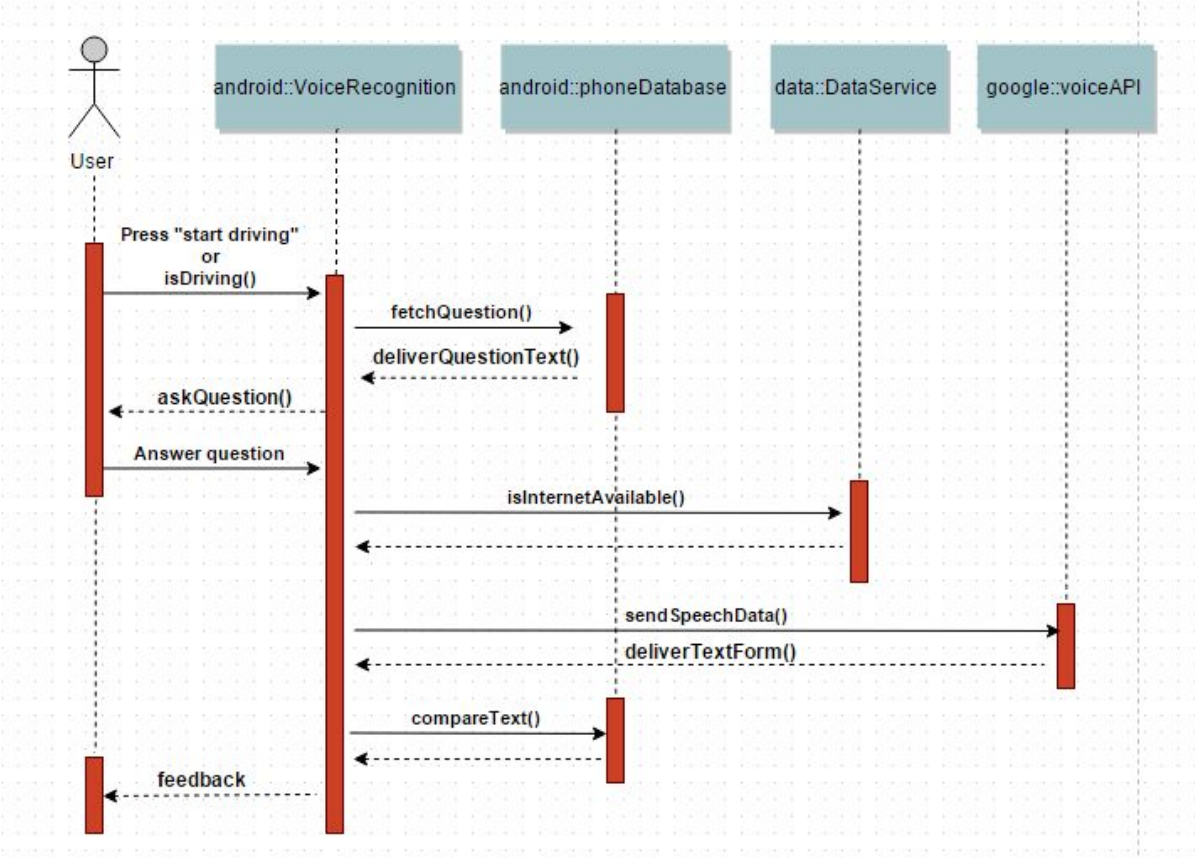

Figure 2.1. DATS Questioning system component sequence diagram

3.2.2 SMS System Component Sequence Diagram

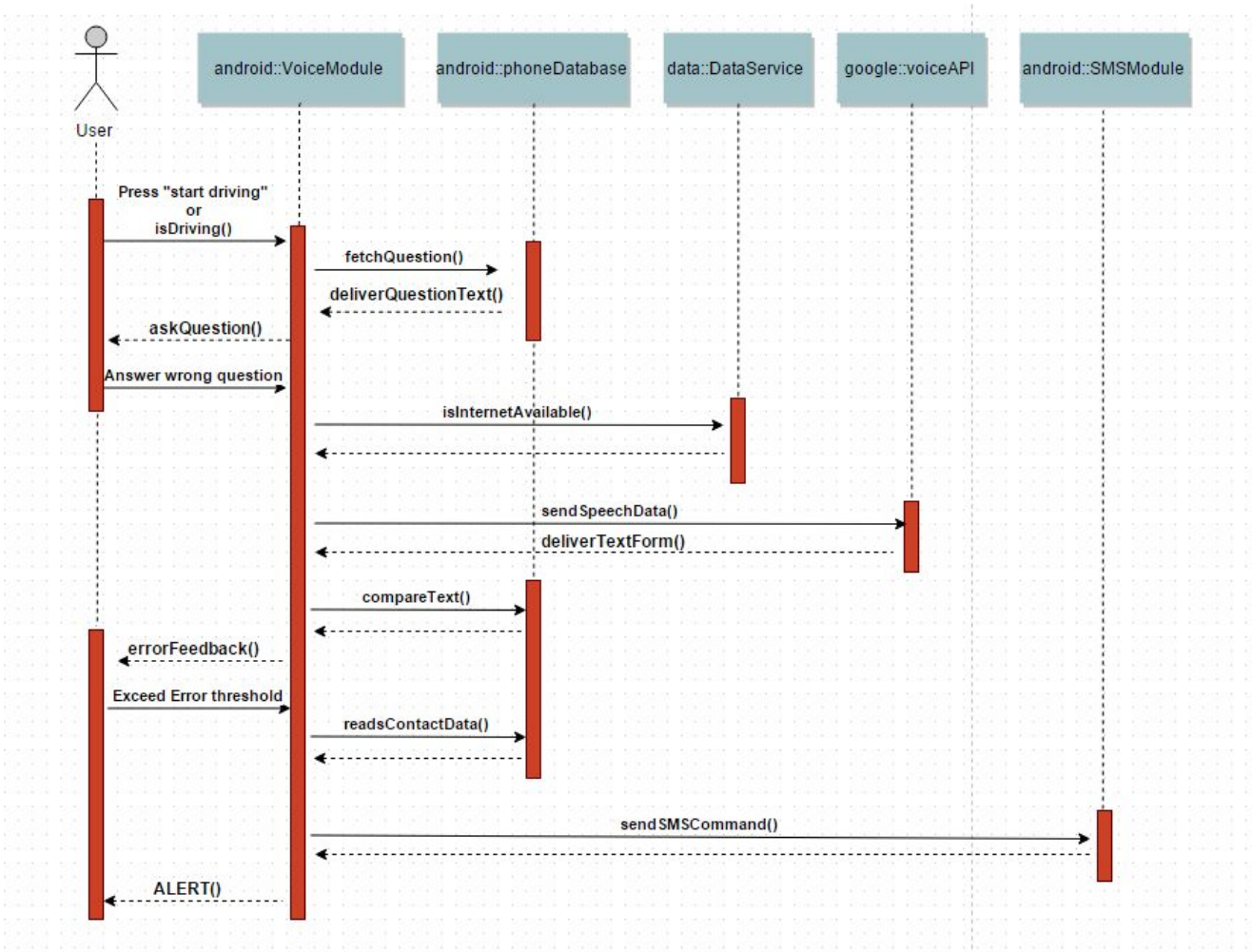

Figure 2.2. DDTS SMS system sequence diagram

The two diagrams above describes the Questioning component and the SMS function component of the system. The questioning component interacts with the user in voice interface using Google Speech API and the phone's SQLite database. The SMS system is describing how the system alerts the selected contact number to notify by the user if the user breached the threshold preset and analyzed by the system.

3.3 Design Rationale

We are making a system with Android platform because Android is currently the most used mobile Operating System in the world capable of doing voice recognition and database in the phone. We decided to use voice recognition for minimal touch interaction from the user and the system because the users will be drivers driving in a car, the minimal touch interaction is intended for the maximum focus for the users' driving.

4. DATA DESIGN

4.1 Data Description

The system will consist of three main data tables: the user parameters (username and contacts to be alerted in case of a loss of awareness detected while driving), data for statistics (results of the previous use of Aware-D) and the questions data (questions and correct answers). The database itself will use SQLite database for Android platform, and will be accessed by the objects created in the java program. The data objects are described in this section:

Session Parameters:
date: date of the session.
user_isDriving: indicates if the user is driving or not.
error_count: the number of wrongly answered questions. Used for analysis.
right_count: the number of correctly answered questions. Used for analysis.
session_id: the identification code of the driving session, each session will be logged.

Question:

question_id: the identification number of the question.
question_text: the text form of the question.
answer_text: the text form of the preset answer.

User_Parameters:
contact_number: the selected contact number(s) to alert with SMS if highly suspicious driving behavior is detected.
contact_name: name of that contact.
username: name of the user, in order to indicate who is the driver to the contact.

4.2 Data Dictionary

Question:
question_id: varchar(3)
question_text: bigtext
answer_text: text

Session Parameters:
date: Date
user_isDriving:boolean
error_count:varchar(2)

right_count:varchar(2)
session_id: varchar(10)

User_Parameters:
contact_number: varchar(15)
contact_name: varchar(20)
username: varchar(20)

5. COMPONENT DESIGN

The following section defines and explains important designs which describes the software, which are shown by diagrams below.

5.1. ARCHITECTURAL DIAGRAM

The architectural design of the application uses the layer concept. Each layer interact in horizontal way. Contents in application layer interact with its upper or lower layer.

5.1.1. User

The user will directly interact to the application layer. User defined as the source of the input who can give inputs to the interface layer. User will be directed to the lower layers by the interface layer.

5.1.2. Database

The database is an entity which stores every single data in the software. Users are not able to directly access the database to retrieve datas. The controller layer is connect between users and database layer since it's retrieve the input from the controller layer.

5.1.3. Interface

This layer gives visualization of the applications. The data who stored in database will visualize here. User could be give input to be forwarded to the controller layer for further query processing.

5.1.4. Controller

This layer controls inputs and give outputs from the interface to the appropriate part of the model.

5.1.5. Model

This layer accepts inputs from the controller layer, integrating them to particular access the database and fetch datas regarding to the queries submitted.

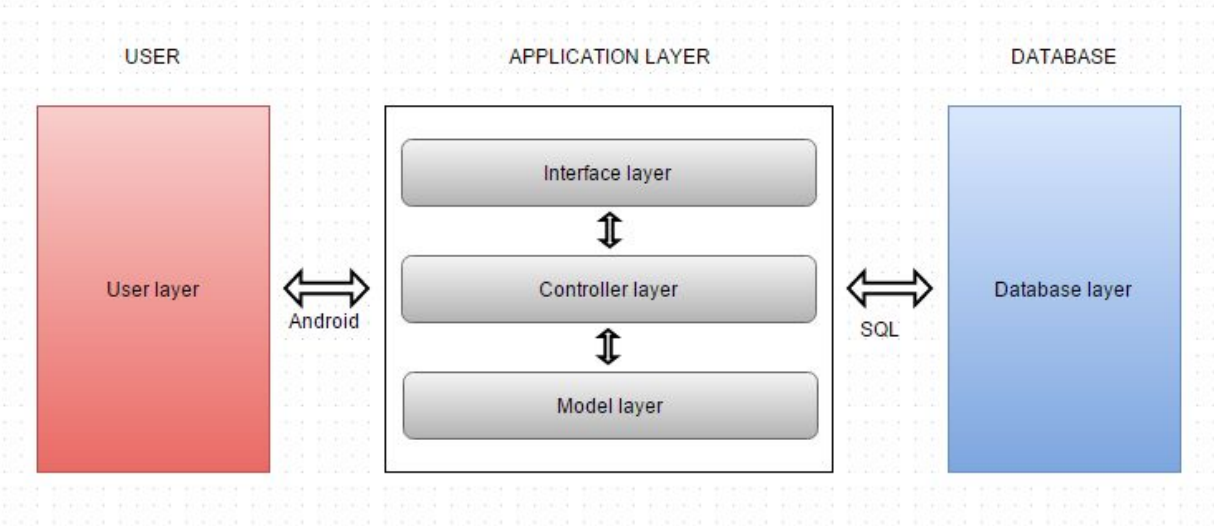

Figure 3. Aware-D Architectural Diagram

5.2. DATAFLOW MODEL

Figures 4.1 and 4.2 present the flowchart, which shows the flow of data in form of algorithm of the software.

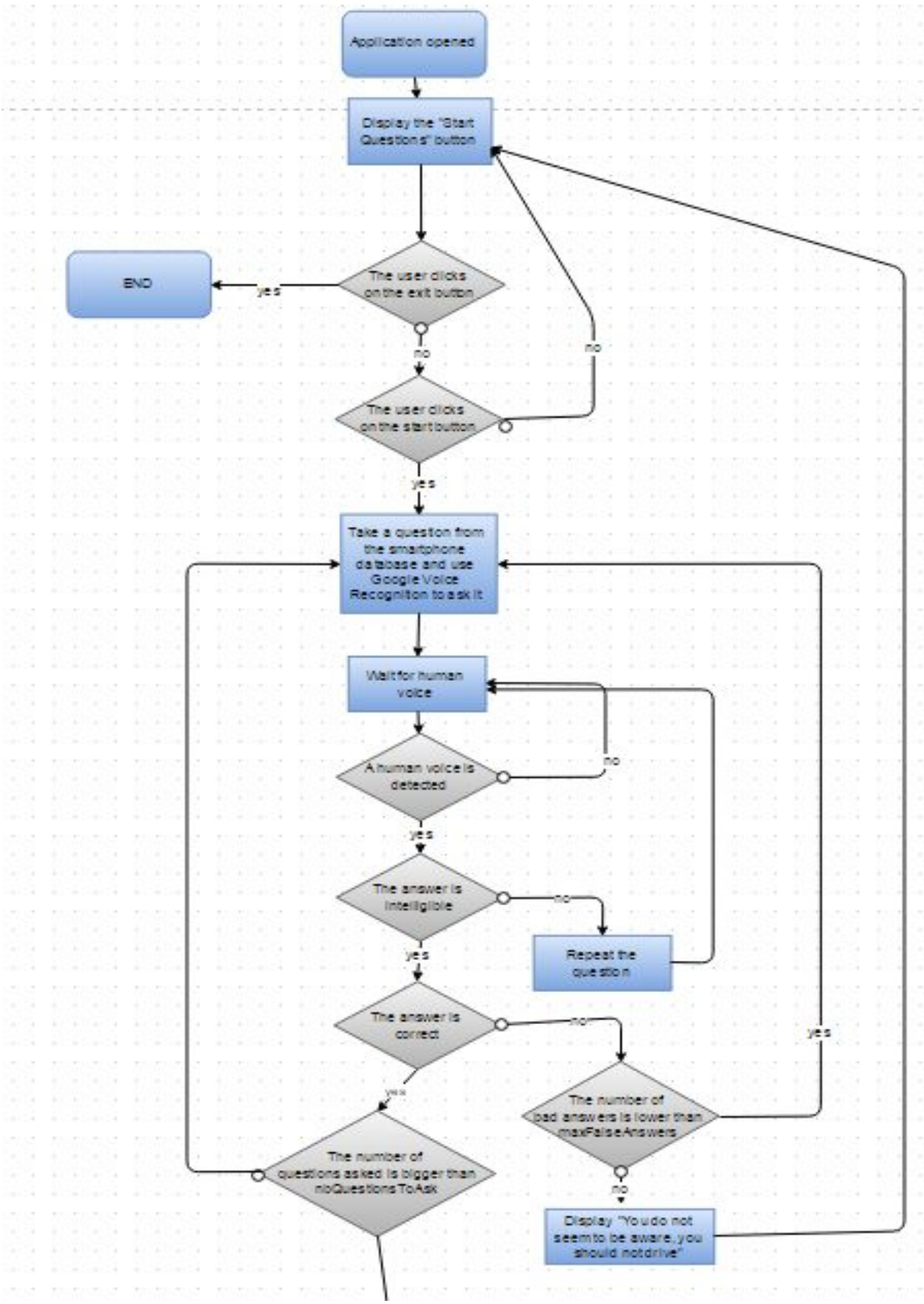

Figure 4.1. DATS Flowchart Diagram (Questions before driving)

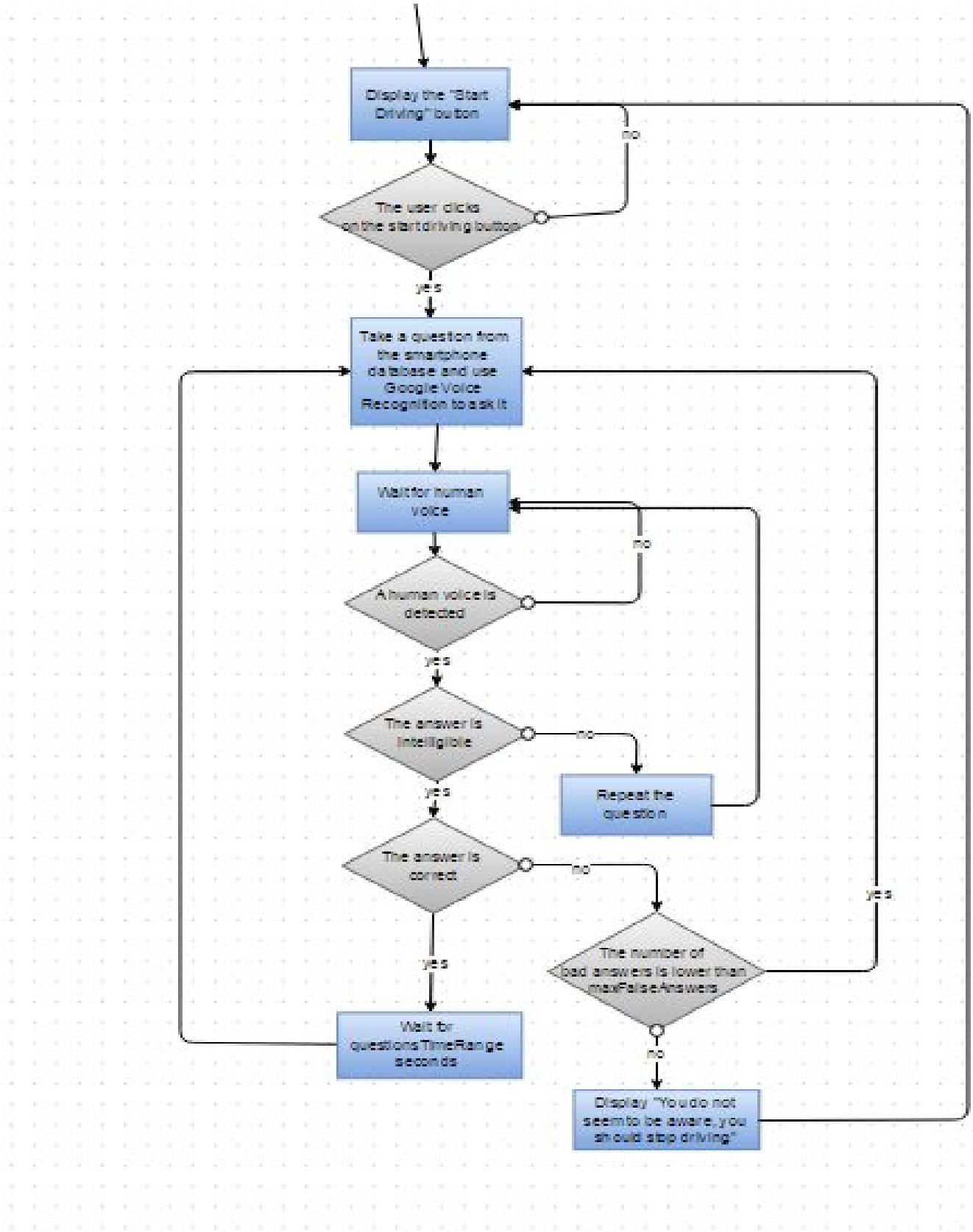

Figure 4.2. DATS Flowchart Diagram (Questions while driving)

5.3. DEPLOYMENT DIAGRAM

Figure 4 shows the deployment diagram, describing the interaction of physical entities in the system. The user interacts with the phone, and the phone will interact with the database in the phone for inputs and outputs triggered and designated to the user.

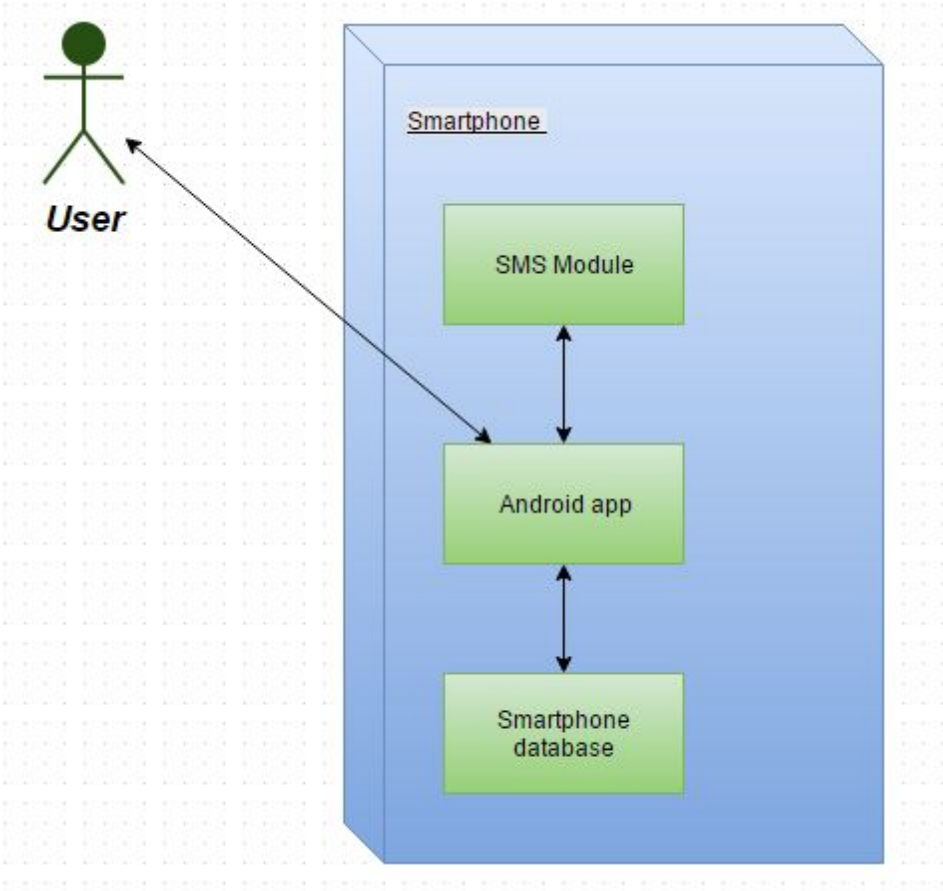

Figure 5. DATS Deployment Diagram

5.4. CLASS DIAGRAM

The class diagram shows the relation of the classes/objects in the system

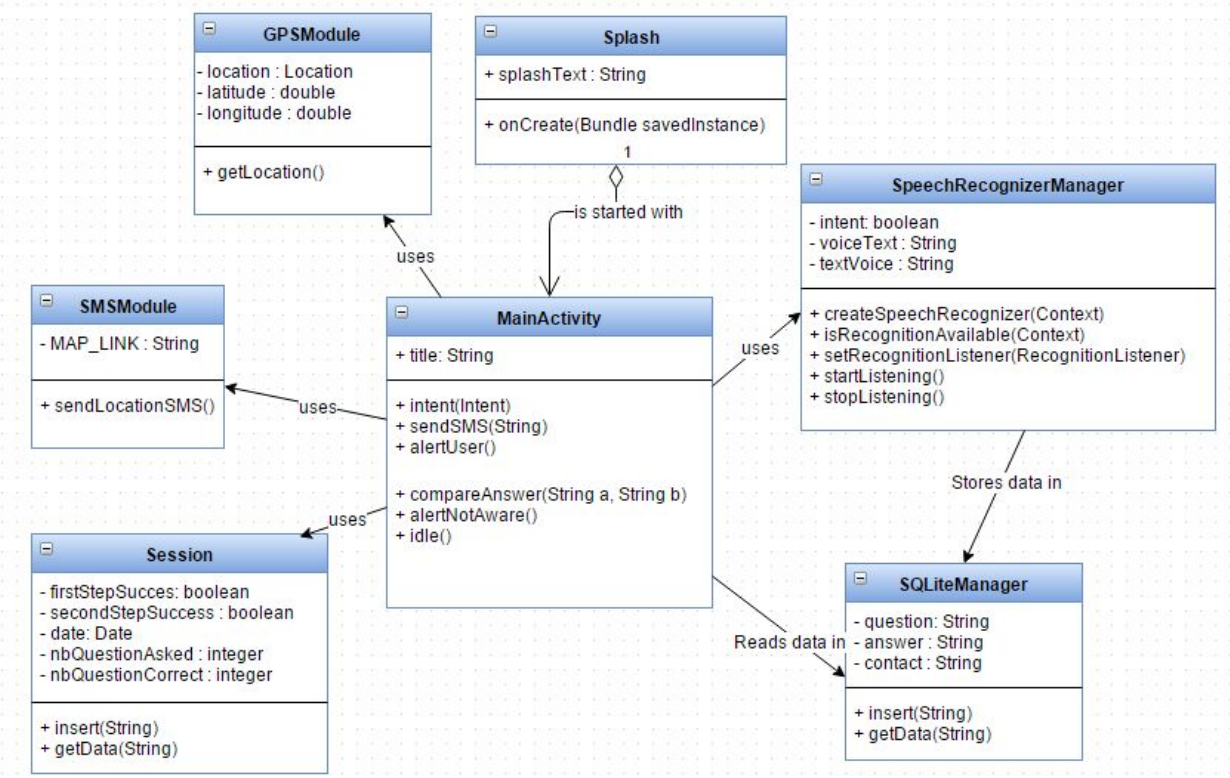

Figure 6. DATS Class Diagram

5.4.1 Splash

Is the splash screen class for creating the splash screen for the app. The app is started with this screen.

5.4.2 MainActivity

The main class of the system, handles the control and input/output from and to the user with RecognitionManager and AnswerManager.

5.4.3 AnswerManager

Calculates the text comparison between the input from the user stored in the database and the answer text preset in the database.

5.4.4 RecognitionManager

Handles the voice recognition, converting from voice to text and from text to voice using Google Speech API. Storing its data in the SQLite database.

5.4.5 SQLiteManager

This class handles the input/output to and from the database.

5.5. USE-CASE DIAGRAM

Describes how the user interacts with the system in steps.

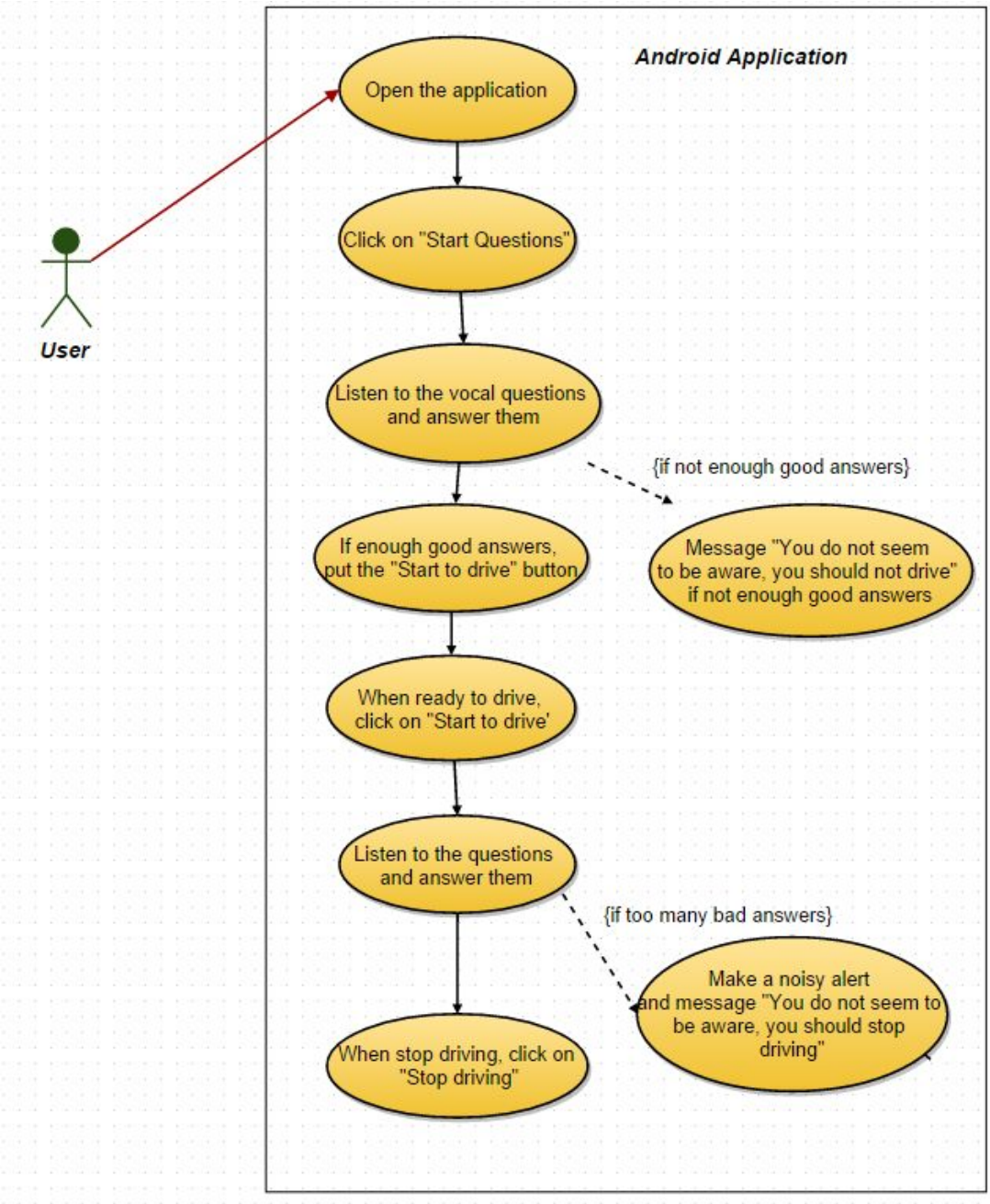

Figure 7. DATS Use Case Diagram

5.6. SEQUENCE DIAGRAM

The diagram describes the whole system's interaction with the user, sequentially, including the graphical user interface (GUI), the SMS and Voice component, and the database in the phone.

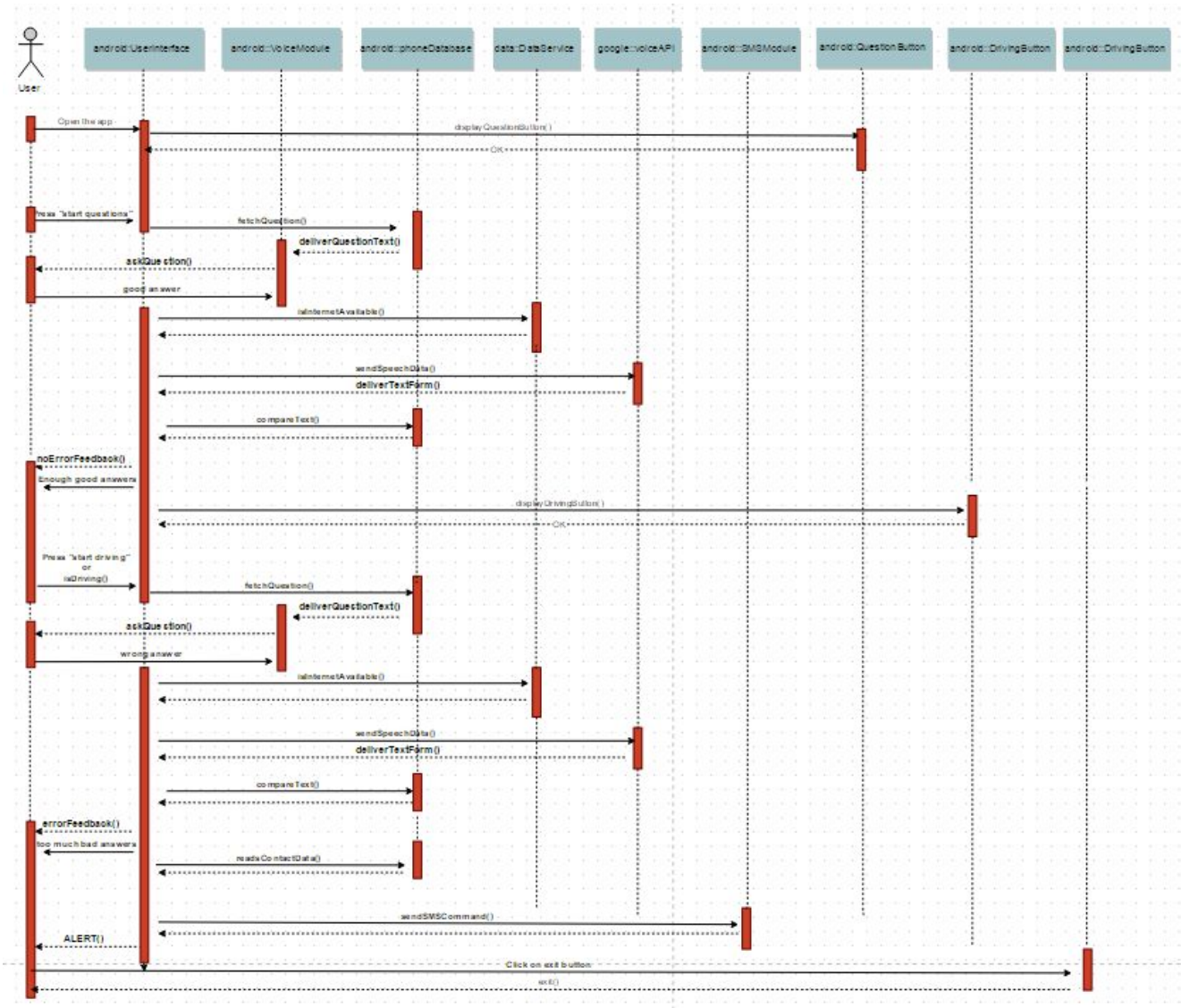

Figure 8. DATS Global system sequence diagram

6. HUMAN INTERFACE DESIGN

6.1 Overview of User Interface

The user will be prompted with the Start button when the application starts. For is first use, the user will have to enter his name and the contact(s) (name and phone number) he wants to be alerted in case of loss of awareness detected during the drive. Once the user clicks the Start button, he will not need to touch his smartphone anymore because everything will be explained by voice. At any time, if the user wants to end the running Aware-D session, he just has to click the End button and the vocal questions will end. The GUI is very simple because the driver must stay focused on the road and does not need to watch his smartphone.

6.2 Screen Images

6.2.1 Main vue Screen

6.2.2 Settings pop-up windows vue

6.2.3 Don’t Drive Screen

6.2.4 Stop Driving Screen

6.3 Screen Objects and Actions

This section briefly describes the interfaces and interface components of Aware-D. Image screenshot of each chapter could be seen in chapter 6.2

6.3.1 Main vue Screen

Main vue screen will be the first screen which the users see when the application starts.

6.3.2 Settings pop-up windows vue

Settings pop-up windows vue screen will be displayed if the user wants to set or modify his name or the contacts to be alerted in case of a loss of awareness while

driving detected.

6.3.3 Don’t Drive Screen

This screen will show up if the user cannot answer the first three questions before driving, and will encourage him to take some rest before start driving.

6.3.4 Stop Driving Screen

This screen will show up if the app detects a loss of awareness while driving, and will encourage the driver to stop driving and take some rest.