Software Requirements Specification

*This document outline is based on the IEEE Standard 830-1993 for Software Requirements Specifications.*

*This document was created in part by Steve Mattingly (smattingly@computer.org).*

*This document should specify what functions are to be performed on what data to produce what results at what location for whom.*

*A properly written SRS limits the range of valid designs, but does not specify any particular design.*

*A good SRS is*

* *Correct (accurately captures the “real” requirements)*
* *Unambiguous (all statements have exactly one interpretation)*
* *Complete (where TBDs are absolutely necessary, document why the information is unknown, who is responsible for resolution, and the deadline)*
* *Consistent*
* *Ranked for importance and/or stability*
* *Verifiable (avoid soft descriptions like “works well”, “is user friendly”; use concrete terms specify measurable quantities)*
* *Modifiable (evolve the SRS only via a formal change process, preserving a complete audit trail of changes)*
* *Traceable (cross reference with source documents and spawned documents)*

*The paragraphs written in the “Comment” style are for the benefit of the person writing the document and should be removed before the document is finalized.*

**Version: Draft**

**September 11, 1998, Changes: August 2013**

**Revision Chart**

*This chart contains a history of this document’s revisions. The entries below are provided solely for purposes of illustration. Entries should be deleted until the revision they refer to has actually been created.*

*The document itself should be stored in revision control, and a brief description of each version should be entered in the revision control system. That brief description can be repeated in this section. Revisions do not need to be described elsewhere in the document except inasmuch as they explain the development plan itself.*

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Primary Author(s)** | **Description of Version** | **Date Completed** |
| Draft | TBD | Initial draft created for distribution and review comments | TBD |
| Preliminary | TBD | Second draft incorporating initial review comments, distributed for final review | TBD |
| Final | TBD | First complete draft, which is placed under change control | TBD |
| Revision 1 | TBD | Revised draft, revised according to the change control process and maintained under change control | TBD |
| Revision 2 | TBD | Revised draft, revised according to the change control process and maintained under change control | TBD |
| etc. | TBD | TBD | TBD |

**Contents**

**1.** **Introduction**

1.1 Purpose

1.2 Scope

1.3 Definitions, Acronyms, and Abbreviations

1.4 References

**2.** **Overall Description**

2.1 Product Perspective

*2.1.1* *System Interfaces*

*2.1.2* *User Interfaces*

*2.1.3* *Hardware Interfaces*

*2.1.4* *Software Interfaces*

*2.1.5* *Communications Interfaces*

*2.1.6* *Memory Constraints*

*2.1.7* *Site Adaptation Requirements*

2.2 User Characteristics

2.3 Constraints

2.4 Assumptions and Dependencies

**3.** **Specific Requirements**

3.1 Software Product Features

*3.2.1* *Feature 1*

3.2 Performance Requirements

3.3 Software System Attributes

*3.3.1* *Reliability*

*3.3.2* *Availability*

*3.3.3* *Security*

*3.3.4* *Maintainability*

*3.3.5* *Portability*

3.4 Logical Database Requirements

3.5 Other Requirements

**4.** **Appendices**

# Introduction

## Purpose

The purpose of this document is to provide a detailed description of the requirements of the mobile application “NAME” for use in hang gliding for pilots and event organizers. It will illustrate the system constraints, interface and interactions with external sensors, mobile sensors and the user.

## Scope

The “NAME” is a mobile application based around external sensors gathering information on humidity, temperature and speed with the combination of GPS it will be able to track the flight route of the pilot live and show a directional path as well as headed direction. Accumulating all the information, post-flight, to show statistics per certain point on the route, based on speed, altitude, temperature and humidity.

The application is an alternative, more comfortable and easier way for pilots to track their flights and add of even more useful functions, with the goal of replacing the very expensive tools that are currently needed or just combine those tools with more features.

## Definitions, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| Term | Definition |
| User | Someone who interacts with the mobile phone application. |
| Pilot | The user who controls the hang glider. |
| Hang Gliding | Hang gliding is an air sport in which a pilot flies a light, non-motorised foot-launched heavier-than-air aircraft called a hang glider. |
| GPS | Global Positioning System. |
| GPS-Navigator | An installed software on mobile phone which could provide GPS connection and data, show locations on map and find paths from current position to defined destination. |
| Stakeholder | Any person who has interactions with the system who is not a developer. |
| Downdraft | Air moving downwards, usually within a thunderstorm cell. |
| Updraft | A column of rising air in the lower altitudes of Earth’s atmosphere. |
| MM | Android ® Marshmallow (6.0) |

## References

1. *IEEE Software Engineering Standards Committee, “IEEE Std 830-1993, IEEE Recommended Practice for Software Requirements Specifications”, December 2, 1993.*
2. *https://www.android.com/versions/marshmallow-6-0/*
3. *https://developer.android.com/reference/android/database/sqlite/package-summary*
4. [*https://en.wikipedia.org/wiki/Variometer*](https://en.wikipedia.org/wiki/Variometer)
5. [*https://en.wikipedia.org/wiki/Hang\_gliding*](https://en.wikipedia.org/wiki/Hang_gliding)
6. [*http://xcmag.com/news/encounters-of-the-liquid-kind/*](http://xcmag.com/news/encounters-of-the-liquid-kind/)
7. [*http://www.aerospaceweb.org/question/atmosphere/q0046b.shtml*](http://www.aerospaceweb.org/question/atmosphere/q0046b.shtml)
8. [*https://en.wikipedia.org/wiki/Density\_of\_air*](https://en.wikipedia.org/wiki/Density_of_air)
9. [*https://en.wikipedia.org/wiki/Humidity#Measurement*](https://en.wikipedia.org/wiki/Humidity#Measurement)
10. [*http://www.kids-fun-science.com/what-is-air-pressure.html*](http://www.kids-fun-science.com/what-is-air-pressure.html)
11. [*https://en.wikipedia.org/wiki/Atmospheric\_pressure#Measurement\_and\_maps*](https://en.wikipedia.org/wiki/Atmospheric_pressure#Measurement_and_maps)
12. [*http://xcmag.com/news/getting-up-to-speed/*](http://xcmag.com/news/getting-up-to-speed/)
13. [*http://articles.latimes.com/2007/aug/30/local/me-maccready30*](http://articles.latimes.com/2007/aug/30/local/me-maccready30)
14. [*https://www.sciencedirect.com/topics/earth-and-planetary-sciences/cumulonimbus-clouds*](https://www.sciencedirect.com/topics/earth-and-planetary-sciences/cumulonimbus-clouds)
15. [*https://www.metoffice.gov.uk/learning/clouds/cloud-spotting-guide*](https://www.metoffice.gov.uk/learning/clouds/cloud-spotting-guide)
16. [*https://en.wikipedia.org/wiki/Cumulonimbus\_cloud*](https://en.wikipedia.org/wiki/Cumulonimbus_cloud)
17. https://wiki.sei.cmu.edu/confluence/display/java/Input+Validation+and+Data+Sanitization

# Overall Description

## Product Perspective

This system consists of an offline mobile application for Android®. In its first release, the application will assist the user by tracking various statistics relevant to hang gliding and presenting them off-line of the flight. In further versions, the application will also assist the user on-line the flight.

The application will be using GPS to gather location data during the flight. The location data will be used to track the user in 2D space while in flight and construct the path.

Since a lot of data will be generated by the application, an offline database that is device specific will be used to store the data.

The initial release will gather data about temperature, humidity, speed, acceleration, location, of which, only location will be gathered from the phone device itself. Humidity, temperature and acceleration will be gathered by \_\_[[INSERT NAME HERE]]\_\_ which is provided by SignalGenerix.

### System Interfaces

The product is an accessory to the common devices employed by hang-gliders, professional or not and does not communicate with any of them. The product communicates with a GPS system through the phone for 2D positional tracking, and \_\_[[INSERT NAME HERE]]\_\_ to gather data about temperature, humidity and acceleration.

### User Interfaces

**Navigation Menu**

The system contains a navigation menu with the following entries which represent different screens:

**Home**

Home is the home screen (first screen) that comes up when the user opens the application and it will contain a small presentation of the last flight and its data, weather information and will provide the action to record a flight.

**About**

About displays information about the specific version of the Application.

**History**

History page shows a list of previously recorded flights, along with their date, time and any custom name provided by the user. The entries allow clicking on them which transitions the user to a page specifically for the selected flight and will be referred as Flight.

**Flight**

Flight page presents all the gathered data in graphs where applicable as a function of time, such statistics are humidity, temperature, speed, directional acceleration, altitude and altitude change.

**Statistics**

Statistics presents the average, minimum and maximum statistics for a specific data point. Such statistics are, top 2D speed, maximum altitude decrease/decrease rate, max/min temperature, directional acceleration.

**Settings**

Settings screen provides various options the user can adjust to personalize their experience.

### Hardware Interfaces

\_\_[[Insert name here]]\_\_ will be used to track humidity, temperature and acceleration. \_\_[[insert name here]]\_\_ is provided by SignalGenerix and connection to the hardware device is done over Bluetooth®. Additionally, the communication layer between Bluetooth® and \_\_[[Insert name here]]\_\_ is provided by SignalGenerix.

### Software Interfaces

**Operating System**

The Operating System (OS) is device dependent, however, Android ® Marshmallow (6.0) which will be referred as MM, will be used as baseline due to its market share and compatibility between it, and later versions.

Android MM’s specification exists in reference [2]

**Database**

The system will be using SQLite®. SQLite is provided by the Operating System and as such, version number and specification number vary. Documentation for SQLite is provided by Google (see reference [2])

SQLite is used as a means to store information generated by the application.

### Communications Interfaces

The application will not be communicating with Internet explicitly. The only case of implicit connection is through the GPS system which is highly dependent on the Android device, Operating System and User settings. As the connection is employed by GPS, the application remains agnostic of it.

Additionally, the application will be communicating with [[INSERT NAME]] through Bluetooth® in order to gather the relevant information. Said communication will be established by the OS and managed through the provided API.

### Memory Constraints

The amount of RAM needed should be around 256Mb of RAM to fit most phones in terms of RAM and phone storage at most 512Mb for database and the app. Meanwhile the app is at most 20Mb. But during the development while gathering the data there may be some adjustments on the memory consumption.

### Site Adaptation Requirements

In order to keep the installation easy and simple and because it's an Android app the main Requirement is to have the Play Store installed. Also it is required to have the “product\_name”

and Bluetooth connection for the application to function properly.

## User Characteristics

Users come from various educational levels without a necessarily high technology knowledge, therefor the application must be simple and easy to use. At the same time every hang gliding pilot passes a specific exam to get a license in order to fly. That means the users will be familiar with the hang gliding terminology, weather and the knowledge for flying at different flying sites.

## Constraints

The power consumption is a constraint because some flights can last for hours and the phone battery must handle the flight.

Another constraint is the amount of data gathered and their real time management, because a lot of information will be gathered from the device sensors and will be processed in real time.

## Assumptions and Dependencies

We assume that in the future a "flight mode" user interface will be needed, to provide the pilot with information during the flight and an option for voice commands so the user will be able to interact with the system while flying. Lastly there is a possibility in the future to add an option to connect to social media, so users will be able to share their flights with their friends.

# Specific Requirements

## Software Product Features

### Feature 1: Calculate elevation/altitude

#### Purpose

Users should be able to check, from the record of their flight, the altitude, for any part of the flight route, and the maximum altitude they reached.

[Flight Mode] The user must be able to check the elevation/altitude at any time, in order to decide if they should descent or ascent.

#### Associated Functional Requirements

##### Functional Requirement 1

###### Introduction

The elevation value is one of the most important features, since users of the hang glider have to know how they should act in order to reach the desirable location. Ignorance of this value sometimes can be fatal for users, because there is a maximum height and if they overcome that height they will have to deal with shortage of oxygen.

###### Inputs

The input values will be send to the application from an altimeter

###### Processing

We don’t really need processing since the altimeter is responsible for this job.

###### Outputs

The altitude will be displayed in feet or meters based on users preference. The user can check the altitude in the post flight statistics or from the flight record.

[Flight Mode] The application will check the value of the altimeter and display warnings in case there is a danger involving the user. The altitude should be displayed to the phone’s screen and should be updated every second.

### Feature 2: Calculate rate of descent or climb

* + - 1. Purpose

Humans are not able to sense climb and sink rates (unlike birds or other flying animals). In the past (before the invention of the variometer) sailplane pilots even if they could readily detect abrupt changes in vertical speed, their senses did not allow them to distinguish sink from lift, or strong sink from weak sink. They really had a hard time in the past since the actual climb/sink rate could not even be guessed at, except if there was a fixed visual entity or reference nearby. A fixed reference, like a hillside or the ground, is generally very unprofitable position to be in, for a glider pilot. The most useful forms of lift, like thermal and wave lift, are usually found at higher altitudes and it is very hard for a pilot to detect or exploit them without the variometer.

* + - 1. Associated Functional Requirements
         1. Functional Requirement 1

Introduction

In order to calculate rate of descent or climb a variometer will be used. In hang gliding the variometer is used almost continuously, sometimes it also has audible output, during a normal flight to inform the pilot of rising or sinking air. Usually gliders are equipped with more than one type of variometer. There is a simple type of variometer which does not need an external source of power (which can save up power). On the other hand the electronic type, which needs a power source to be operative during the flight, can provide audio signals (for guidance). This instrument is not used during launching or landing, with the exception the case when the pilot wants to avoid releasing in sink (aerotow).

Inputs

The input values will be send to the application from the variometer.

Processing

We don’t really need processing (other than safety checks for Flight Mode) since the variometer is responsible for this job.

Outputs

User will be able to check from the flight record the rates for descent or climb that were encountered during their flight.

[Flight Mode] The application will check the value of the variometer and display warnings in case there is a strong lift or sink wave. The descent or climb rate should be displayed to the phone’s screen and should be updated every second. Descent and climb rate must be displayed in meters per second or feet per second based on the user’s preference.

### Feature 3: Calculate the position with GPS

* + - 1. **Purpose**

GPS (global positioning system) and a compass can be used to help the navigation during a flight. For competitions, GPS is used to verify if the contestant has reached the required check-points or not.

* + - 1. **Associated Functional Requirements**
         1. Functional Requirement 1: GPS

*Introduction*

The GPS can be used to visualise the location of the pilot on the map and assign some check points (for competition purposes). It allows users to check their current position and help them pinpoint a landing zone nearby. GPS can also be used to track and record the flight of the glider pilot and let them upload it at a later stage to a global flight database (to “Leonardo” for example).

*Inputs*

The input signals will be send to the application from the mobile’s GPS.

*Processing*

The flights will be recorded and saved on the mobile’s memory until they are deleted or uploaded to a database (requires internet connection).

[Flight Mode] Based on the location, the application will be able to synchronise with a global flight database and check which flights have a conflict with the user’s flight to warn them in advance or during the flight.

Outputs

The application will save the flights to files or a database on mobile’s memory and users will be able to upload them to a global flight database, inspect them or delete them.

* + - * 1. Functional Requirement 2: Compass (for Flight Mode)

*Introduction*

The compass is also useful for glider pilots as it helps them to understand to which way they are heading.

*Inputs*

The input for the compass will be send to the application from the mobile’s GPS.

*Processing*

The GPS will do the processing for the compass calculations.

Outputs

[Flight Mode] The compass can be presented on the application’s interface with an arrow pointing to North.

* + 1. **Feature 4: Temperature**
       1. **Purpose**

The temperature, even if it might not seem very important, it can affect the flight. Consider that when a glider pilot lifts higher in the sky the temperature around him/her goes down. So, it is important to keep track of the temperature. Also temperature can affect downdrafts and updrafts.

* + - 1. **Associated Functional Requirements**
         1. Functional Requirement 1

*Introduction*

A thermometer can be used to send temperature values to the application for statistic purposes.

[Flight Mode] It can warn users in case it spots extremely low (or high) temperatures.

*Inputs*

The input will be given to the application from the thermometer.

*Processing*

No processing required.

Outputs

Users will be able to check the minimum, maximum and average temperature of a flight.[Flight Mode] The temperature can be displayed on the screen periodically. In case of extreme temperatures the application will warn the pilot. Temperature will be displayed in Celsius or Kelvin degrees.

* + 1. **Feature 5: Humidity and pressure**
       1. **Purpose**

Temperature, humidity and pressure can affect the density of air. The density of air is one of the main factors which determine if the wing will fly faster or slower. Pressure can also affect the pilot’s health if it gets too extreme. Humidity and pressure can indirectly affect the flight, so we must make sure that those values are being monitored by the application. Also, sometimes we might be able to predict a cumulonimbus (also called thundercloud) based on humidity and temperature, so that we can warn the pilot to take the appropriate measures.

* + - 1. **Associated Functional Requirements**
         1. Functional Requirement 1: Humidity

*Introduction*

A humidistat can be used to send humidity values to the application for statistics of flight records.

*Inputs*

The input will be given to the application from the humidistat.

*Processing*

Humidity can be used to calculate density of air.

Outputs

Humidity values will help the pilot to get a better understanding of the weather and the “nature” of the air that surrounds him/her during the flight. This will improve his experience and knowledge for the next flight.[Flight Mode] The humidity can be displayed on the screen periodically. In case of extreme humidity values the application will warn the pilot.

* + - * 1. Functional Requirement 2: Pressure

*Introduction*

A barometric can be used to send pressure values to the application for statistic purposes.

*Inputs*

The input will be given to the application from the barometric.

*Processing*

Pressure can be used to calculate density of air.

Outputs

Pressure values will help the pilot to connect how he/she felt during a flight with the pressure he/she was experiencing at a specific part of the flight, giving him more experience for the next one.[Flight Mode] The Pressure can be displayed on the screen periodically. In case of extreme pressure conditions the application will warn the pilot.

* + 1. **Feature 6: Calculate velocity**
       1. **Purpose**

During a flight velocity is very important for the pilot. If the velocity is below the minimum value, under certain circumstances, it might cause loss of altitude and a sharp fall of the wing. Velocity is also important during take offs and landings.

* + - 1. **Associated Functional Requirements**
         1. Functional Requirement 1

*Introduction*

For a given rate of climb there is an optimal speed to fly your particular glider (according to MacCready). This is the most efficient speed, if we take into consideration the balance between getting there fast to exploit good climbing potential, but not so quickly that you lose (valuable) height, which takes too long to regain. MacCready has calculated how that speed must be varied to compensate for associated lift and sink en route, and made flights a lot easier for pilots.

*Inputs*

The input will be given to the application from the GPS which can calculate the velocity during flight.

*Processing*

No processing required.

Outputs

Users after the flight can check their velocity of a specific part of the flight and see their highest or lowest speed and when they reached those speeds. [Flight Mode] The velocity must be displayed on the screen all the time. It should be updated every second. It can warn the pilot if the velocity drops below the minimum value. Also we can present the average velocity and maximum velocity at the end of the flight. Velocity will be displayed on meters per second or kilometres per hour, based on user’s preference.

* + 1. **Feature 7: Timer**
       1. **Purpose**

During a flight competition most of the time we need the time duration of a flight to decide who is the winner.

* + - 1. **Associated Functional Requirements**
         1. Functional Requirement 1

*Introduction*

All phones have a timer (clock) which can be used to calculate the time between take off and landing, or between two checkpoints. This information will also be added to the record of each flight. And maybe we can create a leaderboard with the best times (based on the same geographical checkpoints). Other pilots will be able to download the record from the flight, get the checkpoints and maybe try to beat the time record.

*Inputs*

The input will be given to the application from the mobile’s clock.

*Processing*

Calculate the time between two checkpoints and save it to the flight record.

Outputs

At the end of the flight you will be able to check the flight time, and the time between each pair of checkpoints.

* + 1. **Feature 8: Upload/Download flight’s information** 
       1. **Purpose**

During a flight the pilot must be sure that he/she is not entering an other’s flight route, to prevent accidents. How can the pilot know all the flights in a specific area? The global flight databases have these information stored on their servers, waiting for somebody to check them out.

* + - 1. **Associated Functional Requirements**
         1. Functional Requirement 1

*Introduction*

The pilot will be able to download the flights that are happening in a specific area, at a specific time and the application will warn him/her if he/she is heading to a route of another flight. Also there are areas that flights are forbidden for various reasons which will be marked also on the application, so that it can warn the user in advance.

*Inputs*

Users will be able to upload their flight record to a global flight database (like Leonardo) so that they can free some space from the mobile phone’s local memory.

[Flight Mode] The input will be given to the application from a global flight database.

*Processing*

Upload a flight record to a global flight database or download a record to check the statistics.[Flight Mode] Calculate the distance between user’s location and forbidden area and warn him (100m before).

Outputs

A flight record (if the user decides to download one from the global flight database).[Flight Mode] The application will pop up warnings in case the pilot is heading to a route where at the same time another flight is happening.

## Performance Requirements

At any time only one user can have access to the application (on one mobile device). However many users can use the online features of the application (upload flights to a global flight database).

Recorded flights should be stored to a file on the mobile device (if there is enough space), since an internet connection requires a lot of power and we do not want to leave the pilot in the middle of the flight with no battery. We can add a power bank with the sensors (for backup) and warn the pilot in advance when there is little power left.

The application will require at least 256MB of RAM and 512MB of secondary storage to be able to run efficiently.

Future Requirements:

Altitude, velocity and climb/descent rate must be updated every second (without delays).

Humidity, temperature and pressure can be displayed periodically every 3-5 seconds.

The GPS and compass should change dynamically (as a normal GPS would on ground movement) during the flight and update location every second.

## Software System Attributes

### Reliability

[[Insert name]] provided by SignalGenerix is considered reliable. However, like any other hardware interface, its reliability is a function of the manufacturing process and the operational time. Also, the GPS system is considered reliable.

[[Insert name]] is dependent on the mobile’s operating system. Thus, any failure on behalf of the operating system would result in abnormal termination of the application. None actions are taken to restore the application’s before-the-failure state. A database file in SQLite though, is the only component that can roll back, even if the application crash in the middle of a transaction. In this way, information loss is prevented. (https://www.sqlite.org/howtocorrupt.html)

The statistics presented to the user are based on the measurements gathered from the [[Insert name]], so they reflect the real “big picture” of the flight. In addition, any weather data is retrieved from Internet.

### Availability

[Insert name]] is available on any mobile device with minimum Android version Marshmallow 6.0 and has access on PlayStore.

Also, power consumption is a constraint that defines availability.

Regarding user interfaces, there are some constraints. For example, when the user chooses to record a flight, he/she cannot navigate through the other screens. Moreover, maybe there are cases that an interface screen may not respond while others do.

### Security

[[Insert name]]’s database stores information and statistics about the user’s flights. These are considered personal data that should be kept private. Only the user has the authorization to share this data.

Furthermore, storing data in the database should be followed by the sanitization of the input. In this way it is not only ensured that user’s data is properly stored but malicious code injection is also prevented. [17]

### Maintainability

Maintainability phase includes:

* Bug fixes
* Future updates

Software bugs such as errors or faults in the system that cause incorrect or unexpected behavior should be resolved successfully.

Adding extra features or improving the performance of an existing one, might be a future update.

### Portability

[[Insert name]] is usable on Android phones and tablets only.

## Logical Database Requirements

The system will be using SQLite. A database stores:

* Elevation/Altitude (Feet or Meter)
* Temperature (Celsius or Kelvin(?) or Fahrenheit)
* Humidity (%)
* Acceleration (not in the features?)
* Rate of descent or climb (feet per minutes or meters per second)
* GPS positioning
* Atmospheric pressure (N/m2 or Pa)
* Density of air (kg/m3)
* Velocity (m/s)
* Flight duration (minutes or hours)

The user has access to all the past flight records in the database, so he/she can view or share but not edit them. A flight record includes all the listed information above. Flight records are organized based on the date and time they were recorded.

The application is responsible for gathering the data from the sensors and store it in the database using the appropriate format for each category of measurements.

## Other Requirements

# Appendices

*Include supporting detail that would be too distracting to include in the main body of the document.*