Software requirements specification: Hiking Tour Assistant

Contents

1. Introduction

The Hiking Tour Assistant (HTA) is a software application designed to assist hikers in recording their hiking tours. This Software Requirements Specification for the Hiking Tour Assistant outlines the functional and non-functional requirements of the software. It describes the features and capabilities of the software, as well as the technical constraints and performance requirements.

1.1 Purpose

The purpose of this document is to describe the purpose and functions of the Hiking Tour Assistant and to lay out the requirements in as much detail as possible. This document also serves as a plan for the project and can be used to check requirements related to different aspects of the product.

The intended audience for this SRS is first and foremost the assistants and other students who will evaluate the mini project. As a result, this document is relatively technical and assumes that the reader has some background in software development and/or requirements engineering.

1.2 Scope

Hiking is a popular activity in Finland. The hiking experience can be improved by tracking the activity by using a smartwatch. The purpose of the Hiking Tour Assistant is to help the user to track their hiking session statistics including travelled distance, step count, and burned calories.

The software implements a smartwatch UI which allows the user to start and stop recording sessions. During a hiking session travelled, distance and step count are displayed. When the session is ended, its statistics are saved to the non-volatile memory of the smartwatch. Only the most recent session is saved i.e., the last session is overwritten as a new session is ended.

In addition to this, a web UI for an external device (a Raspberry Pi in this case) for displaying the statistics, including calories burned calculated on the RPi, is implemented. The external device can read data from the smartwatch over a Bluetooth connection.

Note that the session statistics include only travelled distance, step count and burned calories. The software does not for example record the hiking route.

1.3 Definitions, acronyms, and abbreviations

HTA	Hiking Tour Assistant
RPi	Raspberry Pi
SRS	Software requirements specification

1.4 References

LiLyGo TWatch documentation

RPi documentation

IEEE-830 Standard

1.5 Overview

The rest of the SRS contains the overall description of the product, including description of the functions, operation, intended users and potential constraints and assumptions and definitions of all the specific requirements. A context diagram outlining the interactions between external entities and the software system is also provided.

Section 2 contains the overall description of the HTA, consisting of five subsections: product perspective product functions and the context diagram, user characteristics, constraints, and assumptions and dependencies. Section 3 offers a more detailed explanation of the specific requirements. The section is divided into external interface requirements, user interfaces, functional requirements, performance requirements, design constraints, and software system attributes.

2. Overall description

2.1 Product perspective

The Hiking Tour Assistant is meant to serve as a simple solution for logging hiking statistics using a smartwatch. HTA allows the user to track travelled distance and step count during the session and to move these statistics to an external device (Raspberry Pi) where extended statistics including estimated calories burned can be displayed.

The project can be easily separated into two products: the software for recording the session statistics on the smartwatch and a web UI implemented on the Raspberry Pi to display the session data. These two products communicate with each other over a Bluetooth connection.

System interfaces:

There are no external system interfaces as the software is self-contained on the smartwatch and RPi.

User interfaces:

The smartwatch UI shall be intuitive and easy to use. It allows the users to start & stop hiking sessions and input their step length. The current session statistics are displayed during a hiking session. The user can also access the statistics of the last recorded session. The UI shall also have an option to synchronize the data with the RPi on user command.

The web UI on the RPi shall display the extended session statistics on the screen. In addition to travelled distance and step count, the extended statistics include an approximation of the burned calories during the hike which is calculated on the RPi. In order to calculate the estimated burned calories, the UI should ask the user to input their weight.

Hardware interfaces:

The hardware interfaces on the smartwatch include the screen and buttons which the user can interact with. The hardware interfaces on the RPi include the peripherals such as the mouse, keyboard and monitor.

Software interfaces:

The two main software interfaces are the LilyGo smartwatch firmware and the Raspberry Pi operating system.

Communications interfaces:

The smartwatch should communicate with the RPi via Bluetooth using the ESP32 platform. ESP32 supports Bluetooth v4.2.

Memory constraints:

The Raspberry Pi has 1 GB of LPDDR2 memory and 32 GB of flash storage.

The LilyGo watch has 520 kB of SRAM, 8 MB of PSRAM and 16 MB of flash storage.

As only the latest session statistics are stored, the smartwatch memory capacities should not pose any issues.

Operations:

The smartwatch has four main modes of operation: idle, recording, stopped and synchronizing with RPi. The RPi has four main modes of operation: idle, synchronizing with the smartwatch, calculating the estimated burned calories, and displaying the data.

2.2 Product functions

The main product functions are listed below. These are explained in greater detail in section 3.2 Functional requirements.

- Start & stop hiking sessions
- Record step count and convert it into travelled distance during the hiking session
- Display this data on the smartwatch screen
- Synchronize and store the data with RPi via Bluetooth
- Calculate the estimated number of calories burned during the session on RPi
- Initialize the web UI and show statistics from the last session (travelled distance, step count and burned calories)

2.2.1 Context diagram

Figure 1 shows the context diagram for the Hiking Tour Assistant (HTA). The diagram shows the basic relationships between the HTA and the main hardware interfaces which interact with it i.e., the smartwatch screen, the smartwatch controls and the RPi peripherals. The smartwatch controls are used as inputs for the system, and they can indicate either starting/ending a session, synchronizing the data of the last session, or inputting the users step length in order to calculate the travelled distance. The display shows the user the relevant information, for example steps and travelled distance during the session. The RPi accepts synchronizing the latest session data from the smartwatch over a Bluetooth

connection. After receiving the data, the user's weight is requested and estimated burned calories are calculated. Next, the extended statistics are displayed on the external monitor connected to the RPi.

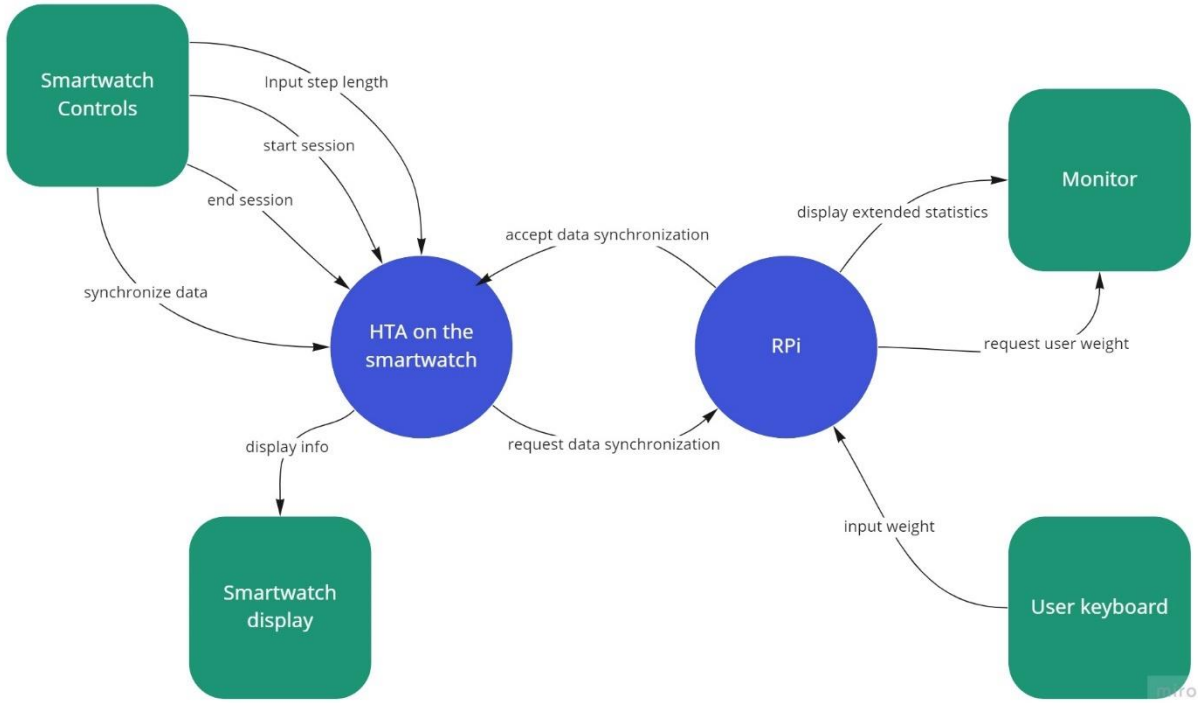

Figure 1: The context diagram of the Hiking Tour Assistant.

2.3 User characteristics

The expected users are normal people who do not necessarily have technical expertise or higher education. The watch should be easily usable and not require more than basic knowledge of such devices.

2.4 Constraints

Regulatory policies: The software shall follow the data collection laws in Finland.

Hardware limitations: The main limitations of the hardware is the memory capacity and processing power of the LilyGo smartwatch.

Interfaces to other applications: Other than the interfaces between the smartwatch and RPi, there are no interfaces to external applications or devices.

Parallel operation: The smartwatch and RPi operate independently, except when they are synchronizing. There are no other parallel operation constraints.

Higher-order language requirements: C++ is used for the development of the smartwatch software.

Signal handshake protocols (e.g., XON-XOFF, ACK-NACK): Bluetooth v4.2 implements the 802.11 RSN 4-way handshake between two devices establishing a connection.

Reliability requirements: The main reliability requirement is that the smartwatch saves the session data into non-volatile flash memory in regular intervals so that if the watch malfunctions (for example the battery runs out) the session data can be recovered.

Criticality of the application: The application is not critical, and possible failures will only be an inconvenience for the user.

Safety and security considerations: The application is not safety-critical.

2.5 Assumptions and dependencies

It is assumed that the software is developed specifically for the LiLyGo smartwatch and RPi platforms, so only the constraints which come with these platforms need to be accounted for. If the goal platforms change the SRS will need to be updated accordingly.

The LiLyGo TWatch Library will be used to implement the smartwatch software. The library provides many useful ready-made functions such as the BMA423_StepCount.

3. Specific requirements

3.1 External interface requirements

3.1.1 User interfaces

Name of item	Session controls	HTA screen	Data synchronization command	Display session data command	Session data display	User step length	User weight
Description of purpose	Start and stop recording sessions	Display hiking session information	Send a data synchronization command	Send a command to the RPi to display the saved session data.	Display the session data saved to the RPi	Determine user step length for distance conversion	Determine user weight for calorie calculation
Source of input	Button on the HTA	N/A	Button on the HTA	Mouse connected to the RPi	N/A	Button on the HTA	Keyboard connected to the RPi
Destination of output	N/A	HTA screen	N/A	N/A	Monitor connected to the RPi	N/A	N/A
Valid range & accuracy	N/A	N/A	N/A	N/A	N/A	0 –10 meters, two decimal accuracy	0 – 1000 kg, one decimal accuracy

Units of measure	N/A	Steps, kilometers	N/A	N/A	Steps, kilometers, calories	Meters	Kilograms
Timing	Starting or ending a session shall have at most a 1 s delay	The screen shall update at 1 Hz frequency or faster	Synchronization attempt shall start within 1 second of pressing the button	The command shall execute with at most a 500 ms delay	Displaying the data shall take at most 5 seconds to complete	Saving the user step length shall take no longer than 500 ms	Saving the user weight shall take at most 500 ms
Relationships to other inputs/outputs	Session results are visible on the HTA screen	Related to the session controls	This command starts the synchronization process	Session data is displayed after executing this command	Related to the display data command	Step length is asked on the HTA screen	When the data is synchronized, calories are calculated based on the user's weight
Screen formats/ organization	N/A	LED matrix with steps and distance in this order	N/A	N/A	Data is displayed in the following order: steps, kilometers, calories	N/A	N/A
Window formats/ organization	N/A	N/A	N/A	N/A	Depends on the connected display	N/A	N/A
Data formats	N/A	Floating point numbers	N/A	N/A	Floating point numbers	Floating point numbers	Floating point numbers
Command formats	Boolean input	N/A	Boolean input	Boolean input	N/A		N/A
End messages	N/A	Session saved message	N/A	N/A	N/A	User step length saved message	User weight saved message

3.1.2 Hardware interfaces

The system shall make use of the HTA screen and peripherals connected to the RPi to achieve the desired functionality.

3.1.3 Software interfaces

The system shall work together with the smartwatch firmware and RPi operating system.

3.1.4 Communications interfaces

Name of item	Bluetooth interface
Description of purpose	Synchronize the data between the HTA and RPi
Source of input	Input data from the HTA
Destination of output	N/A
Valid range & accuracy	N/A
Units of measure	N/A
Timing	Synchronization attempt shall take at most 10 s, otherwise it shall time out
Relationships to other inputs/outputs	The data synchronization command starts the synchronization process
Screen formats/organization	N/A
Window formats/organization	N/A
Data formats	Floating point numbers
Command formats	N/A
End messages	Synchronization result on the HTA and RPi screens

3.2 Functional requirements

In this subchapter, the functional requirements for the system are described through use cases.

Use case 1a: the user starts new session

- When the user presses a button to start the session, the watch starts recording steps.
- The HTA state changes from idle to recording.
- If starting the session fails due to an error, the HTA should be able to recover from it.

Use case 1b: a session is in progress

- The HTA is in the recording state.
- Steps taken by the user are tracked and used to calculate the traveled distance.
- The traveled distance and step count are displayed on the screen as an LED matrix.

- If the session ends due to an error, the HTA should try to save the data gathered so far and be able to recover from the error.

Use case 1c: the user ends the session

- When the session is ended by pressing a button on the HTA, the recorded session data is saved into non-volatile memory.
- The HTA screen shall show a session ended message and then turn off.
- The HTA state changes from recording to stopped and then idle.
- If there is an error when ending the session, the watch should be able to recover from it.

Use case 2: the user sends a synchronization command

- When the synchronization command is given from the HTA, it will try to form a connection with the RPi. It will time out after a certain time if connection cannot be formed. If the connection is formed, it will attempt to synchronize the data.
- The result of the synchronization (success or failure) is displayed on the HTA and RPi screens.
- The states of both the HTA and RPi change from idle to synchronizing and then back to idle.
- In case of an error, both the HTA and RPi should be able to recover from it.

Use case 3: the user sends command to display data saved on RPi

- When a command is given, the data is retrieved from the RPi memory and a web server is launched, where the data is displayed in a readable format.
- If no data is found, the web server shall show a “No session data” message.
- The state of the RPi changes from idle to displaying data and back to idle when the user exits the web page.
- In case of an error, the RPi should be able to recover from it. In this case, the web server should show an error message if possible.

Use case 4: the HTA asks for the user’s step length

- The user inputs their step length on the HTA screen. The system checks if the input is a number in the correct range. If it is, the input is accepted and saved into the memory. Otherwise, it is rejected and asked again.
- This procedure happens on the first startup of the HTA.
- If there is an error at any point, the HTA should be able to recover from it.

Use case 5: the RPi asks for the user’s weight

- The user inputs their weight using the keyboard connected to the RPi. The system checks if the input is a number in the correct range. If it is, the input is accepted and saved into the memory. Otherwise, it is rejected and asked again.
- This procedure happens on the first startup of the RPi.
- If there is an error at any point, the RPi should be able to recover from it.

3.3 Performance requirements

The timings described in chapter 3.1 should hold at least 95% of the time. The HTA and RPi shall save one session at a time into memory. Length of sessions is only limited by the memory constraints of the devices. In practice, the maximum session length is probably never reached.

3.4 Design constraints

3.4.1 Standards compliance

The following standards shall apply to the development of the HTA:

- IEEE-830 std for Software requirements specification
- Bluetooth v4.2 standard

3.5 Software system attributes

3.5.1 Reliability

The HTA shall perform its intended function consistently without errors.

3.5.2 Availability

The smartwatch should save the last session statistics into non-volatile memory so that it can be viewed in the web UI. The statistics of the previous session can be deleted as a new session is started. In this case, the watch should ask the user to confirm that the data from the earlier session can be deleted.

3.5.3 Security

Only the latest session is saved on the device. The communication between the smartwatch and RPi using Bluetooth is only established, when necessary, otherwise Bluetooth should be turned off to minimize security risks. The RPi is not connected to the internet; thus, a potential attacker would need physical access to the RPi to perform any malicious activity, for example steal user data. Therefore, no data encryption is needed.

3.5.4 Maintainability

The code should be modular and commented on as necessary to make it easily readable to increase maintainability.

3.5.5 Portability

Using a commonly used language such as Python and C++ increases the portability of the system.