Software Requirements Specification

for

Electronic Barkeeper

Version 1.1 approved

Prepared by Team 8 (Andreas Schön, Krist Stoja & Tobias Bartsch)

Chair of operating systems

19.11.2017

Copyright © 1999 by Karl E. Wiegers. Permission is granted to use, modify, and distribute this document.

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Team 8	11.11.2017	Initial Draft	1.0
Team 8	19.11.2017	Final Version	1.1

1. Introduction

1.1 Purpose

In the original sense the purpose of this project is to develop a way to control a robotic arm via an android device. To give this project a more realistic approach, a rudimentary electronic barkeeper setup is chosen for the development with the Braccio Robotic Arm. This documents provides preliminary information that are necessary for its development.

1.2 Document Conventions

Higher level features are always prioritized whereas lower level features are marked with the keyword “optional”. Optional features must not necessarily be implemented but will be focused if there is enough time to do so. Requirements described in the description follows the conventions of RFC 2119 for the usage special keywords, e.g. “must”, “shall”, “should” and so on.

1.3 Intended Audience and Reading Suggestions

The intended audience for this document is mainly the chair of operating systems in order to track the requirements and the app development team in order track their progress. Of course this document is also produced for those who want to understand the project.
Throughout the rest of this document the project is split into sections. It starts with a description of the project followed by the external interfaces and the system features. This document ends with a short documentation of the non-functional requirements and the assignment of tasks.

1.4 Project Scope

The electronic barkeeper is a standalone research project without any corporate goals or business strategies at this point. The main goal is to enable the Tinkerkit Braccio Robotic Arm to get controlled by speech recognition and voice commands via an android device. Keeping in mind the electronic barkeeper should therefore be able to recognize and match different alcoholic and non-alcoholic beverages in the real world with sample information saved in the database. The recognized drink will then be displayed on the android device.
Optional: In the broader sense the robotic arm should also be able to pass you a wished drink automatically by searching the demanded drink amongst a list of prepared possibilities.

The software being used for development is the Android Studio provided by Google and Arduino IDE provided by Arduino. A GitHub repository will be utilized for the source code and its integrated task management functionalities will be used as well.

2. Overall Description

2.1 Product Perspective

This product is mentally a follow-on project of the previous semester who implemented a controlling mechanism for a robotic arm facilitating the internal sensors of the android device like e.g. gyroscope, accelerometer. This project starts again right from the beginning but with a different goal. This time speech recognition and the practical approach for an electronic barkeeper is part of the scope. Ideally the purpose is to ease a barkeeper's daily work by letting him concentrate on the mixing of the drinks and not on the ordering and delivering process.

2.2 Product Functions

The most basic functions of the product will be the simple movements of the arm in all possible directions utilizing the six servo motors of the robotic arm. Therefore speech recognition will be used. Additionally it will be able to match colors of drinks to previously defined drinks and display them on the android application.

2.3 Operating Environment

The Braccio Robotic Arm will be controlled by the microcontroller Arduino Uno R3. A Bluetooth Shield V2.0 will be attached to the Arduino Uno in order to receive signals from an Android device. Supported versions of Android are 4.1 (Jelly Bean) or higher. The minimum API level will be 19. Additionally it needs the Bluetooth version 4.0 or higher for communicating with the microcontroller.
Optional: If multiple pairing will be included an android version of at least 6.0 (Marshmallow) and Bluetooth version of 5.0 is needed.
Secondary a TCS34725-RGB color sensor will be connected to the microcontroller. Data will be retrieved from a small data base. The software will be developed using the latest version of Android studio and Arduino IDE.

2.4 User Documentation

Mostly this project concentrates on controlling the robotic arm by speech recognition. Thus the user interface will be reduced to a bare minimum. Therefore it does not need an extensive user manual to explain the screens. A simple tutorial in the help section will be implemented to guarantee the understanding of the user interface. Additionally a list and explanation of the possible voice controls will be provided to enable the user.

2.5 Constraints and Assumptions

The constraints of the product are mainly caused by hardware limitations. Depending on the capability of the sensors. The performance for recognizing the drinks will fluctuate. Another

constraint is the version of the Android operating system which has to provide the data from the voice controller. Also the utilization of speech commands for the robotic arm is unknown.

We assume that the sensors provided are compatible with the Arduino Uno chip and provides the expected data. We also assume that the data gathered from the voice controller of Android will deliver strings that we can further use for command identification and movement assignments.

3. External Interface Requirements

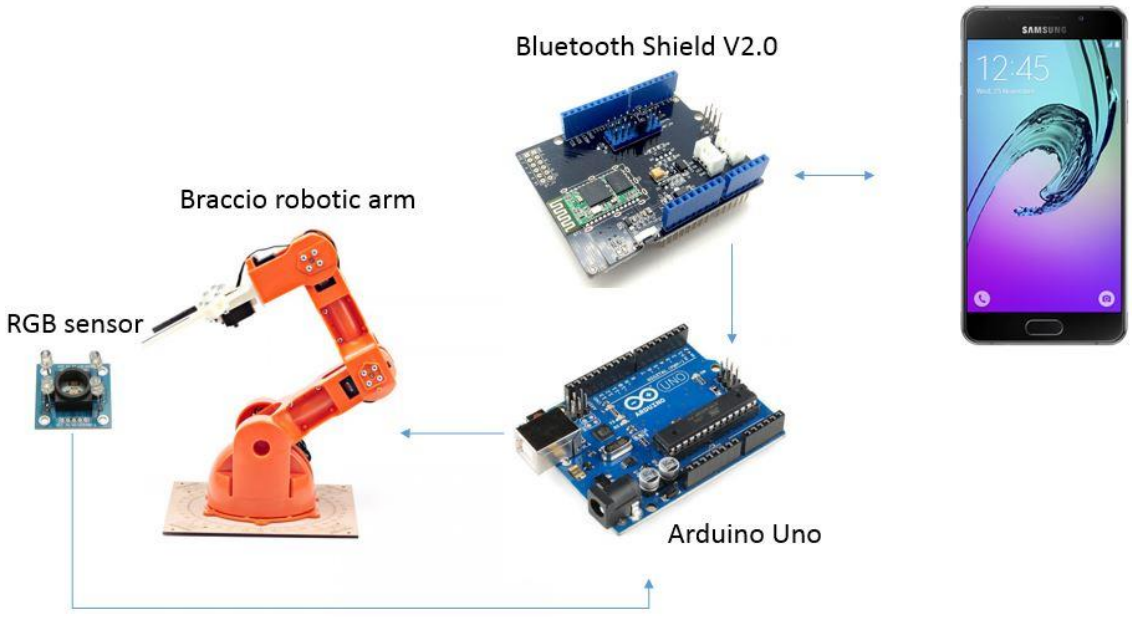
3.1 User Interfaces

For the GUI a simple interface without any user authentication is used. The application follows Google's Material Arts design. The main frame consists of a header and a hamburger item on the left top corner. A help section allows the user to inform himself about the application. The user interacts with the robotic arm via a virtual button in the middle of the screen that initiates the speech recording.

Further details about functional requirements are described in 4.4 User interface.

3.2 Hardware Interfaces

The Arduino Uno R3 is used as the micro controller to connect the robotic arm and the android device. Android devices can be used with Bluetooth 4.0 or higher. Those versions already include Classic Bluetooth, Bluetooth high speed and Bluetooth low energy protocols. In order to get the signal received from our android device, a Bluetooth Shield V2.0 is mounted. To communicate with our Arduino Uno R3, you can choose two pins from Arduino D0 to D7 as Software Serials. Whereas D0 and D1 are Hardware Serial Ports. The robotic arm is connected to the Arduino Uno R3 via a USB Port. The lowest supported Version is USB 2.0. Sensors use the I/O pins of the Arduino as connectors and are limited accordingly. For starters a color sensor, the adafruit-sensore TCS34725 RGB is facilitated.
Optional: A gas alcohol detector is utilized to further increase the detection accuracy.



3.3 Software Interfaces

Google's voice recognition is used to analyse the recorded speech and transform it into a string. This string will then be processed within the Android application. Afterwards a message is sent to the Bluetooth Shield which receives the message and transfers it to the Arduino Uno. Now the Arduino decides whether it activates the robotic arm and/or the sensors.

The Arduino Uno sketch is a small program uploaded to the Arduino Uno which makes it possible to communicate with the Android application and perform actions with the Braccio Robotic Arm.
Libraries are necessary to create commands for the Braccio Robotic Arm within the Arduino Uno sketch.

The used software parts to create and run the system are:

- Android OS
- Google voice recognition
- Arduino Uno sketch
- Arduino Braccio library
- Arduino Servo library
- Arduino Bluetooth Shield library
- Arduino RGB sensor library

4. System Features

The system features are separated in mandatory and optional requirements. The core features are deemed mandatory and so need be realized satisfactorily in the end of the project. Optional features on the other hand can be implemented depending on the possibilities. During the creation of mandatory features, easy extendibility with optional features will be considered.

4.1 Voice control

REQ-1: Users shall be able to control each servo motor (joint) of the robotic arm individually by addressing them via speech recognition.
REQ-2: Users shall be able to use predefined speech recognition commands to control the robotic arm.
REQ-3: Users shall be able to utilize predefined movement patterns.

4.2 Bluetooth connection

REQ-4: The Arduino Uno shall be connected to the Android device via a Bluetooth connection.
REQ-5: Users shall be able to connect the device to the Arduino Uno within the application.
REQ-6: The application shall ask the user after each start of the application to turn on Bluetooth.
REQ-7: After turning on Bluetooth the application shall display the surrounding Bluetooth devices.
REQ-8: Optional: Multiple pairing of more than one robotic arm is possible.

4.3 Sensors

REQ-9: The Arduino Uno shall utilize a RGB colour sensor.
REQ-10: The Arduino Uno shall recognize the RGB code of beverages.
REQ-11: The Android application shall match the RGB code of a beverage to predefined drinks in the database.
REQ-12: Optional: The Arduino Uno shall utilize an alcohol gas detector.
REQ-13: Optional: The Arduino Uno shall recognize the alcohol concentration of beverages.
REQ-14: Optional: The Android application shall match the alcohol concentration of a beverage to predefined drinks in the database.

4.4 User interface

REQ-15: The user interface shall be easy to use.
REQ-16: The user interface shall adhere to material design.
REQ-17: The user interface shall be structured in screens.
REQ-18: The first screen of the application shall be a Bluetooth control centre where the surrounding Bluetooth devices are listed and a button exists to turn your own Bluetooth on and off.
REQ-19: The user interface shall consist of a consistent header.
REQ-20: The user interface shall have a hamburger item in the left top corner that always stays there.
REQ-21: The user interface shall have an own screen solely for speech recording.
REQ-22: Optional: The user interface shall have an own screen solely for manual controlling of the robot's servo motors. This should be realized by slider bars.
REQ-23: The user interface shall have a section that provides an overview over the drinks.
REQ-24: The user interface shall have a details view for each drink that is linked to each picture in the list accordingly.
REQ-25: The user interface shall have a help section that provides the user with an application user guide.
REQ-26: The user interface shall have a list of speech commands for the speech recognition.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

NFREQ-1: Micro bucking of the robotic arm by interfering signals shall be avoided.
NFREQ-2: A stable and sufficient connection between the hardware devices shall be existent.
NFREQ-3: The recognition of a color after expressing the command and the subsequent matching of this color to a beverage shall happen within 10 seconds.

5.2 Software Quality Attributes

NFREQ-4: The source code shall be maintainable.
NFREQ-5: The source code shall be testable.

NFREQ-6: The source code shall adhere to the Google Java Style Guide as well as Android best practices.

6. Mapping tasks to team members

Requirement	Andreas Schön	Krist Stoja	Tobias Bartsch
1	48h	48h	48h
2	16h	16h	16h
3	8h	8h	8h
4		8h	
5		4h	
6			8h
7			4h
9	8h	8h	8h
10	20h		
11			16h
15	8h	8h	8h
16	8h	8h	8h
17	8h	8h	8h
18			8h
19		8h	
20		16h	
21	24h		
23			16h
24			12h
25		24h	
26	16h		
∑	164h	164h	168h