System Requirements Specification

Android Interface - Firebird API

CS308 Project

Aditya Ayyar - 09005001
Darshan Kapashi - 09005004
Siddhesh Chaubal - 09005008
Pararth Shah- 09005009

March 2, 2012

Contents

1 Introduction

We plan to provide a library to android app developers for developing apps for the firebird. The API will allow programmers to focus on developing the application rather than harping on minute technical details of the hardware. It will make the functionalities of firebird accessible to the entire programmer community by providing a simple higher level interface to firebird.

The purpose of this document is to present a detailed description of the Firebird API. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, and the constraints under which it must operate.

1.1 Definitions,Acronyms and Abbreviations

Term	Definition
Bot / Robot	The electro-mechanical machine that is guided by the user.
Software Requirements Specification	A document that completely describes all of the functions of aproposed system and the constraints under which it mustoperate.
User	The person who is using.our API.
SDK	Software Development Kit
IDE	Integrated Development Environment.
API	Application Programming Interfaces.
AVR Studios	IDE for c programming for the robot.
Android	An open source operating system in cell phones; the platform on which user will make apps for the bot.
USB	Universal Serial Bus
Bluetooth	A medium of wireless communication between Firebird V and the controlling device. Use bluetooth module for the Firebird from Nex Robotics

1.2 References

- Android Developer Reference
  http://developer.android.com/reference/packages.html
- USB interface tutorial covering basic fundamentals bluetooth
  http://www.eeherald.com/section/design-guide/esmod14.html
- An Introduction to Bluetooth Programming
  http://people.csail.mit.edu/albert/bluez-intro/
- E-Yantra
  http://www.e-yantra.org

2 Overall Description

2.1 System Environment

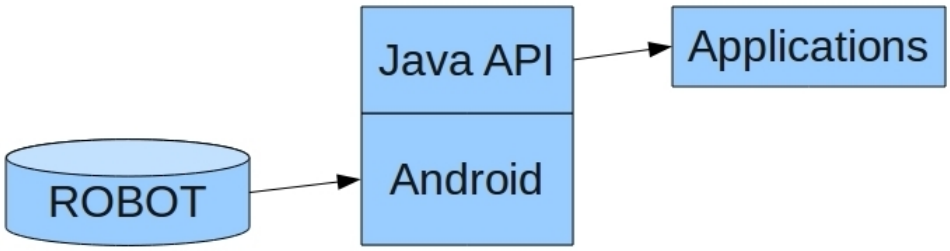

Figure 1: System Environment

- There are 2 interfaces in this system.
- One interface sits on the robot and listens to any communication from the Android device, over a USB or Bluetooth wireless link.
- Another interface, the Java API, sits on top of this Android device.
- The Java API exposes the functionalities of the robot which can be used by any application on the device.

2.2 Product Perspective

- The robot (Firebird V) is controlled using some communication channel.
- The communication channel may be a USB serial communication port or Bluetooth.
- An application on the Android device will invoke the firebird API on the Android.
- The API will process the command given and use its communication module to talk to the robot.
- The robot will listen (using its communication module) and react appropriately (generate some output or provide input).

2.3 Product Functions

This section outlines the use cases for a user in this system.

2.3.1 Read Input

1. Application invokes the API. e.g. Firebird.getReading(PORTC);
2. API implementation will create a message like "command=get_input,parameter=PORTC"
3. This message will be sent to the Firebird.
4. Firebird decodes this message and takes required action. In this case, gets value on PORTC
5. Firebird transmits back the value to the device.
6. The earlier invoked API function returns this value to the application.

2.3.2 Generate Output

1. Application invokes the API. e.g. Firebird.moveForward();
2. API implementation will create a message like "command=move_forward"
3. This message will be sent to the Firebird.
4. Firebird decodes this message and takes required action. In this case, turns on both motors in forward direction.

2.4 User characteristics

- The user (i.e. Application Developer) is expected to have only some basic knowledge about working of sensors, motors, etc. and be able to code in Java to do reasonably complex tasks with the Firebird using an Android device.
- The purpose of this project is to make programming the Firebird bot very convenient for existing Java programmers, who have some experience with building Android applications.

2.5 Constraints

1. If the communication channel being used is bluetooth, the Android device and the Firebird must be within bluetooth range.
2. If the communication channel being used is USB serial communication, the Android device should preferably be mounted on the Firebird.
3. Communication delays may occur which can affect performance.

2.6 Assumptions and Dependencies

The bandwidth of the communication channel should be able to handle the load of communications i.e. downlink and uplink.

2.7 Requirement Subsets

1. Transmission of Commands
2. Interpretation of Commands
3. Signal generation based on Commands
4. Action performed based the Signals
5. Response based on the environment of the Robot

2.8 Testcases

An android app with the following features:

- Display various sensor values from Firebird on the controlling device
- Print text to the lcd display of Firebird.
- Move the bot.
- Ring the buzzer occassionally.

3 Details

3.1 Functionality

- This project provides a generic Java API to access all the capabilities of the Firebird Robot through USB and/or wireless communication
- The Java API will abstract all commands for manipulating the robot, including reading of port/sensor values, controlling motor speed, setting timers/interrupts, printing to LCD screen, etc
- The generic Java API can be access via any Android phone or a PC supporting Java Runtime Environment
- We provide an Android Interface built on top of this API to enable Android App Developers to build innovative apps making use of the Firebird bot
- The Android interface will allow Android developers to manipulate the Firebird robot without worrying about the actual details of the bot
- We will also provide a library of higher level functions for performing simple tasks on the bot, eg. move forward by x cm, rotate bot by x degrees, etc
- This library will build on top of the Java API and is intended to help novice programmers who are not acquainted with Firebird programming, while advanced users can directly use the Java API to access low level functionality of the Firebird to customize their apps.

3.2 Supportability

3.2.1 Basis for future work

- The main goal of our project is to provide completely reusable code which will help Java programmers with varying amount of exposure in working with the Firebird, to easily build innovative apps
- The success of this project is more dependant on how well the code can be used by other Java programmers to build interesting apps for the Firebird
- So the project is designed as a collection of modules which implement a specific task and which are interchangeable as required

3.2.2 Distinguished modules of the project

Each module performs a certain task. The following are the major modules:

1. Firebird On-Board Interface
2. Communications Link Interface (USB or Bluetooth)
3. Java API
4. Android API
5. Simple Tasks Library
6. Applications Layer

The next section describes these modules in detail.

3.2.3 Documentation

Documentation in terms of embedded comments and also an explicit documentation will help any future developers. Each module will be supported with in-depth documentation regarding its functionality, use-case and dependence on other modules.

3.2.4 Design Constraints

The restrictions being implemented in the code are primarily due to those imposed of the system by the hardware being used. They are as follows:

- The latency of communication between Android device and Firebird puts a limit on the rate at which commands can be transferred to the bot
- This will restrict the apps to be designed such that the rate of transfer of commands and data required is less than the rate supported by the communication link
- In case the bluetooth communication link is not fast enough then the USB serial port can be used by mounting the phone on the bot
- However this means that the phone cannot run interactive apps which require inputs from the user during execution

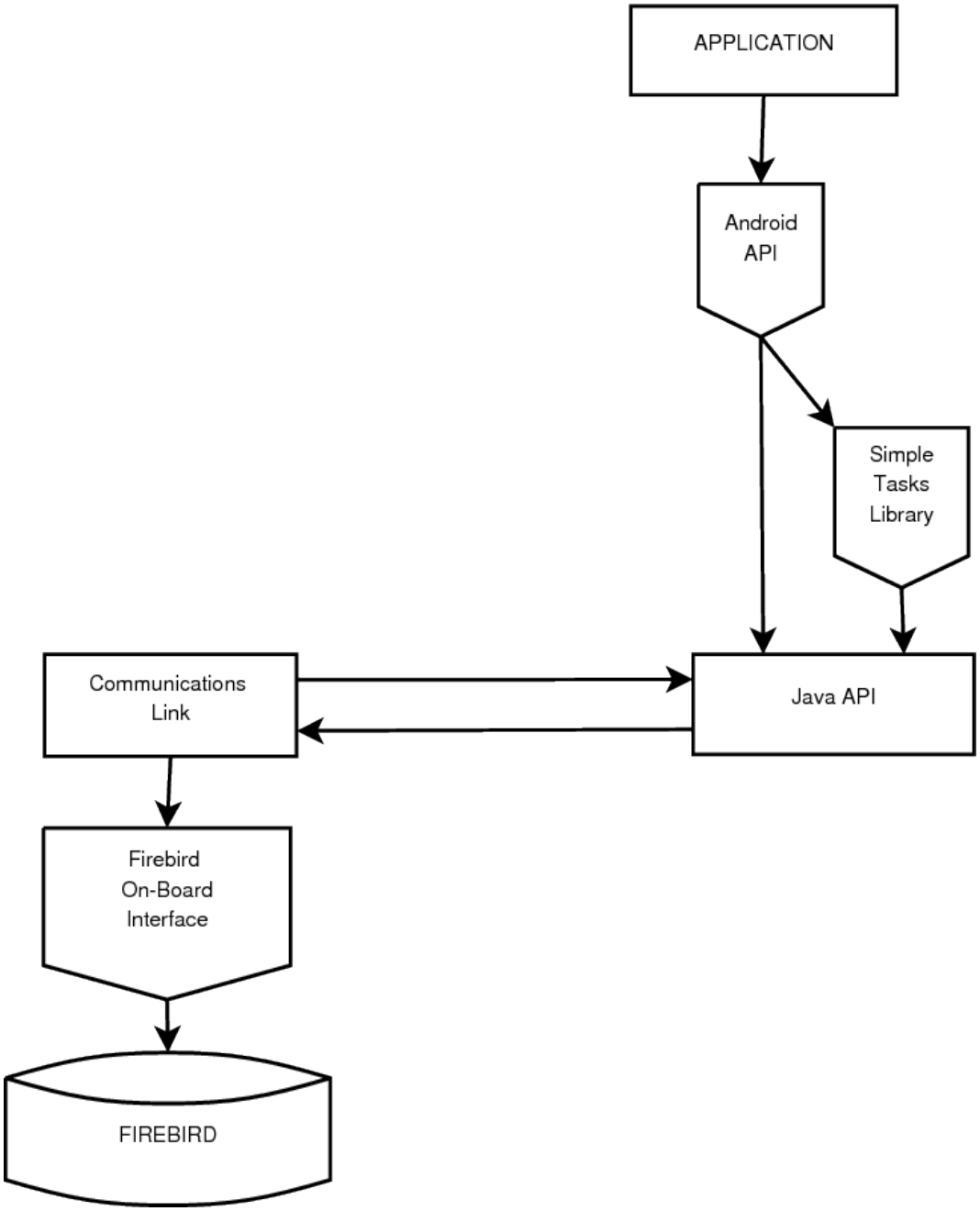

Figure 2: Module Architecture

3.3 Interfaces

3.3.1 Firebird On-Board Interface

- This module is the one which actually communicates with the hardware of the bot
- It implements all the low level functionality, ie reading/writing to port values, manipulating motors, etc
- The code will be written in Esterel and compiled into C code which is burnt on the bot
- It is basically a set of functions with each functions doing a specific task
- The code will run in a loop waiting for commands from the Communications Link

3.3.2 Communications Link Interface

- This module abstracts the communication over a serial USB port or Bluetooth link
- It will define a message format which is used to send/recieve messages between the Firebird On-Board interface and the Java API
- The interface code will be divided into two layers, one present on-board the Firebird, and other one to be installed on the Android phone (or PC) which is controlling the bot

3.3.3 Java API

- This is the most important module which provides Java functions for manipulating low level functionality of the bot
- It will talk to the Firebird Interface via the Communications Link to send/recieve command and data messages
- The exact format of functions will be provided in a separate specification sheet which is to be used by all programmers while implementing their applications

3.3.4 Android API

- This module provides a native Android API to build apps on the Android phone for manipulating the Firebird
- It abstracts all the functionality required to communicate with the hardware, so that developers can include it like any other Android API in their app
- The API is designed to have an interface to other Android APIs so that it is easily integrable in the Android apps, alongside other native APIs for using accelerometer, GPS, magnetometer, etc of the Android phone

3.3.5 Simple Tasks Library

- This library builds on the Java API to provide functions for accomplishing simple tasks on the Firebird, like moving the bot by x cm, rotating by x degrees, etc
- This is useful for programmers who do not know the ports, sensors, and other hardware specific details of the Firebird
- It can be used with the Android API to build apps by combining the simple tasks for doing something useful with the Firebird

3.3.6 Applications Layer

- This is the final layer which builds upon the previous layers to build innovative apps that perform useful tasks combining the features of Firebird bot and Android phone
- For programmers without any experience with Firebird, the Android API and Simple Tasks Library should be used to implement the app
- Advances users who require greater flexibility and control over the Firebird functionality should use the Java API to access the low level features of Firebird

4 Quality Control

4.1 Ensuring correct behaviour

The bot will correctly execute the command given by the app. In case of failure to do so, an exception will be thrown, facilitating the application to take suitable action.

4.2 Precision of sensor inputs

Precise values of sensor inputs will be provided to the application as per request.

5 Risk Management

5.1 Communication latency

Slow communication speed may cause a deviation in the stipulated behaviour due to various latencies.
Fall back plan: Calibrate the bot for various functionalities showing deviation in behaviour.

5.2 Implementation of interrupts

There is no trivial method for the application to realise an interrupt has occurred in the robot.
Fall back plan: We can hard code the interrupts in the robot, so that they function as they would have done in a simple firebird program.