DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
THE UNIVERSITY OF TEXAS AT ARLINGTON

ARCHITECTURAL DESIGN SPECIFICATION
CSE 4316: SENIOR DESIGN I
FALL 2019

WINTER SOLDIER
OUT REACH ROBOT

JESEE CALZADA
SEDICK IYAMAH
PRATIKSHYA DEVKOTA
JESWIN MATHEW
ALEXANDER WINDELER

Revision History

Revision	Date	Author(s)	Description
0.1	10.01.2015	GH	document creation
0.2	10.05.2015	AT, GH	complete draft
0.3	10.12.2015	AT, GH	release candidate 1
1.0	10.20.2015	AT, GH, CB	official release
1.1	10.31.2015	AL	added design review requests

CONTENTS

List of Figures

List of Tables

1 INTRODUCTION

This products main concept can be broken down into main layers, the tablets application along with its communication modules and the robotic arm and it's software. This document will lay out the path itself the drawing takes from the tablet to the arm within a certain specified time, along with the details of the subsystems in each level.

2 System Overview

The overall structure of the project consists of two main layers. The main path the drawing is planned to take from the application to the robotic arm is the following: The user has to first make the drawing itself in the Android application, Inkscape, and saves the drawing as SVG, which would be transferred to a specified path automatically for images in the tablet itself. Then an android application made by our group/team itself would retrieve this image from that specific path and send it to a Python IDE for Android and convert the SVG to G-code. This G-code is sent the robotic itself so it can work on mimicking the drawing.

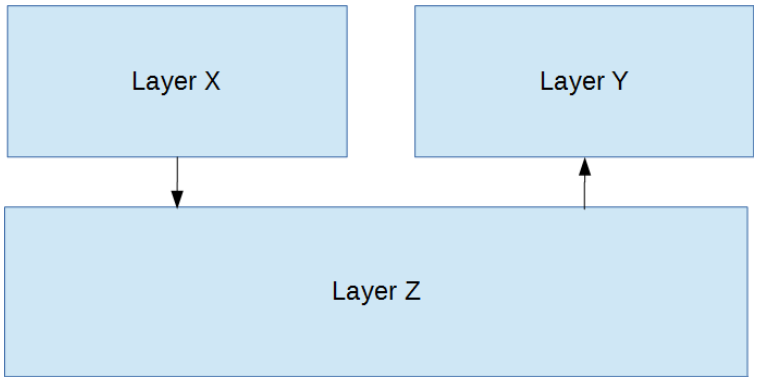

Figure 1: A simple architectural layer diagram

2.1 Tablet Layer Description

As described above, this layer focuses on the interactions between the tablets drawing app (Infinite Design/Adobe Illustrator). After the student finishes the drawing on Infinite Design (or appropriate drawing app), the user will get the option to export the PNG file to SVG format. This will be then saved onto the path where images are stored on the tablet.

2.1.1 Communications Module

The current plan is to create an developed by the team itself using Android studio for the purpose of sending the drawings from the tablet's drawing app to the Marlin software interacting with the robot. The application acts as the communication module between the drawing app and the arm. This application will retrieve the SVG file from the following path: /storage/emulated/0/DCIM/Camera, and send it to the Qpython IDE for compiling the Python scripts for converting the SVG file to G-code.

2.2 Robotic Arm Software Layer Description

This layer consists of the software that will be used to feed control messages to the robotic arm hardware. It connects via USB to the tablet layer through the communications layer and runs on the hardware within the arm hardware layer. There is only one subsystem within the Arm Software Layer which will be the Marlin Software Subsystem. The Marlin software is an open source software made to be a driver for 3D printers.

2.3 Hardware Layer

The hardware layer consists of an Arduino Mega 2560, a Ramps 1.4 board, Motor drivers and the SCARA 4 axis arm robot. The Marlin firmware from the Robotic Arm software layer has to work on the Arduino board to be able to correctly control the arm's movements using G-Code.

3 SUBSYSTEM DEFINITIONS & DATA FLOW

This project has three main subsystems based on our current plan. The main flow of the drawing is as shown in the diagrams itself, starting from the application layer to Marlin Software and finally the Hardware layer. The application layer has two subsystems, the drawing app and the communication module. These two subsystems work in the tablet itself and they are described previously with its major functions implemented. The next is the Robotic Arm Software layer. For this layer we will work with Marlin.ino file in Arduino IDE and configure the Arduino Mega 2560 with it. Finally comes the Hardware layer which consists of the Arduino Ramps 1.4 board, motor drivers and the Scara Robotic arm itself. The RAMPs board receives control voltages from the Arduino as determined by the Marlin software. These control signals are used to determine how to control the arm's motors when mimicking the drawing.

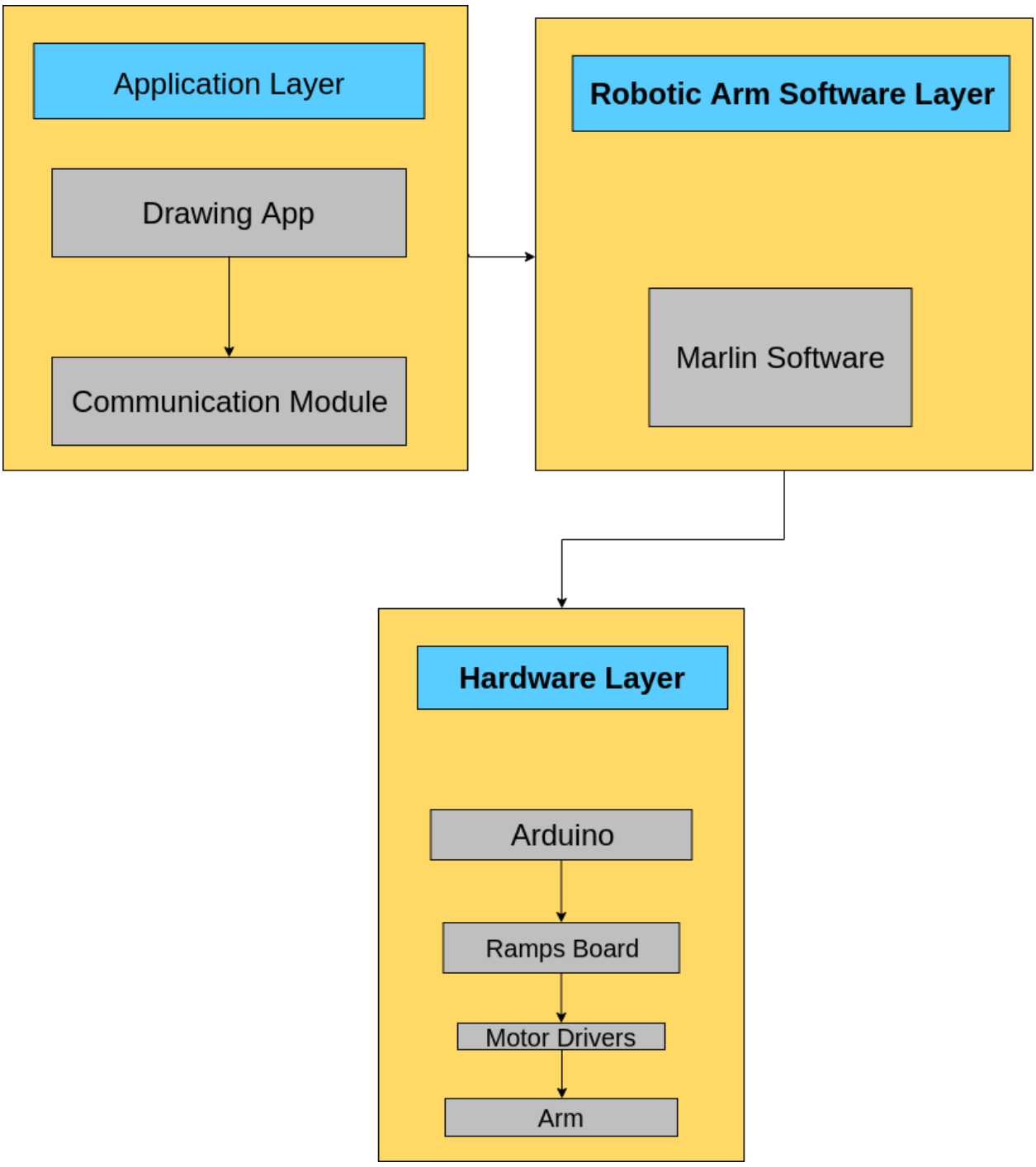

Figure 2: Drawing's flow/path from application to Arm

4 Application Layer Subsystems

The layer is one of the software layers in this project. It deals more with user's input. It consists of two subsystems Tablet subsystem and the Communication module.

4.1 Tablet Subsystem

The Tablet subsystem is the interaction between the user and the communication module.The user will draw a drawing in the tablet using an app called "Infinite design". The drawing will be then saved as a SVG file. This subsystem will send the SVG file to the communication module subsystem.

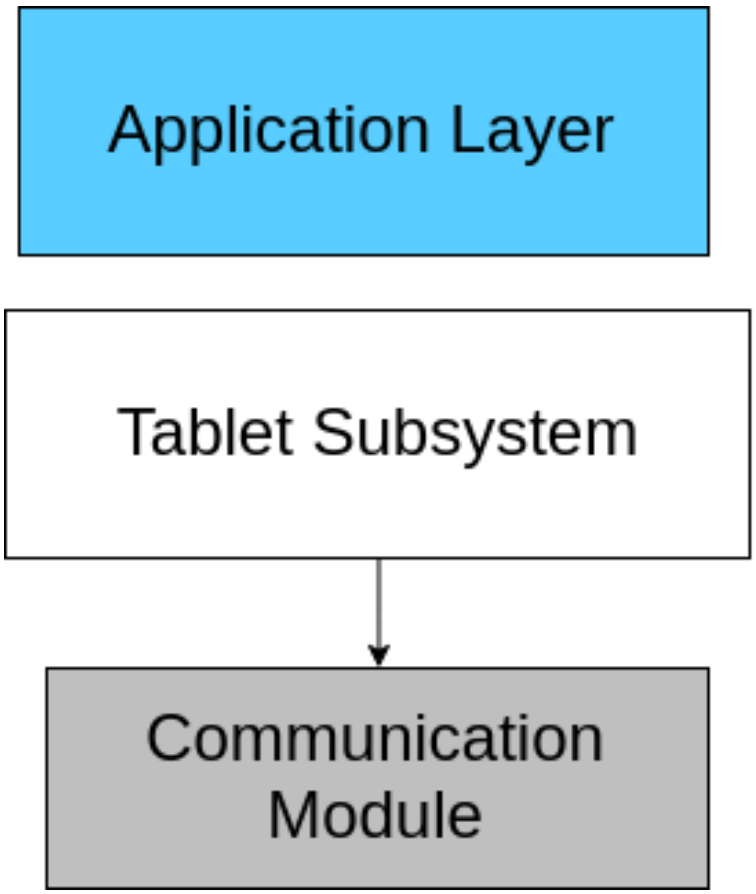

Figure 3: Example subsystem description diagram

4.1.1 Assumptions

The app shall be an android app. The drawing app will save the image as SVG file.

4.1.2 Responsibilities

"Infinite Design" drawing app will be used to draw on the tablet. The drawing will be then saved as a SVG file. The saved file will be then sent to communication module through an USB port.

4.1.3 Subsystem Interfaces

There will be two interfaces.
(1) The drawing app on the tablet.
(2) The USB port through which the SVG is sent to the communication module.

Table 2: Subsystem interfaces

ID	Description	Inputs	Outputs
#01	The drawing app on the tablet.	Drawing	SVG file
#02	The USB port	SVG file	N/A

4.2 Communication Module/App

The communication app/module is a planned application to be built by the software team in our project for handling the SVG file coming in from the drawing app and processing it as necessary. It also takes in input from the robot when a drawing is received properly or in incorrect format.

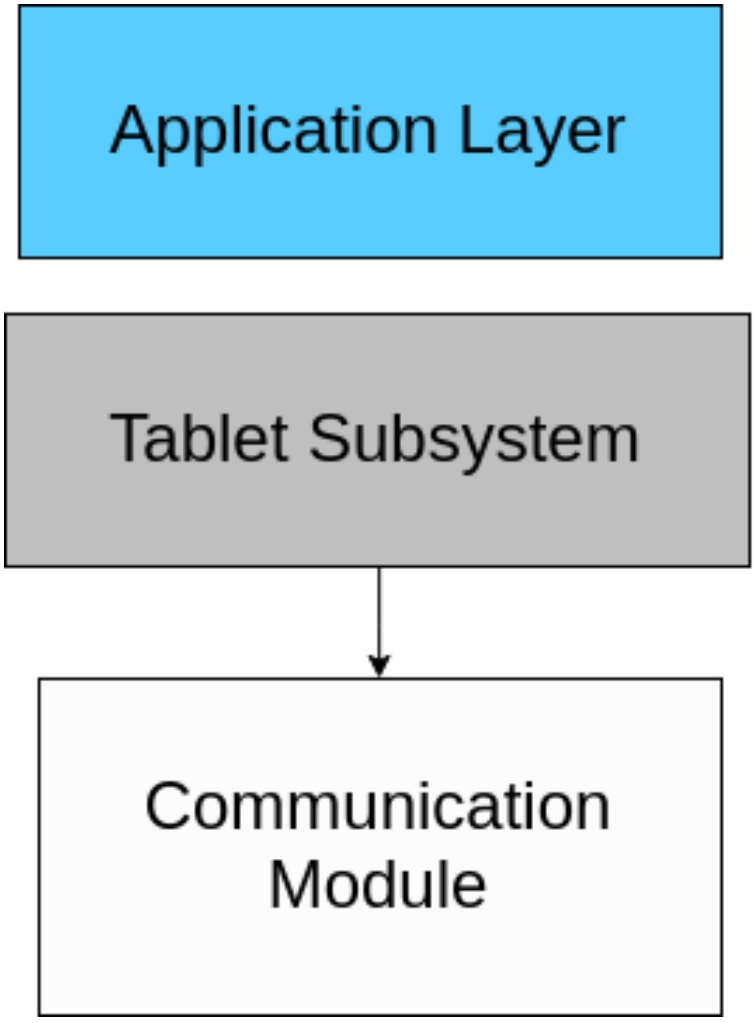

Figure 4: Communication subsystem diagram

4.2.1 Assumptions

The input file is of SVG format.

4.2.2 RESPONSIBILITIES

When the communication module/app receives the SVG file from the tablet's drawing app, it will send the file to a Python IDE running the scripts necessary to convert the SVG file to G-code. Then the app will interact with the Marlin software by sending the G-code to it and waiting for feedback input from the robotic arm itself. In addition, this app will also discard any files as necessary which are not in proper SVG format before running the Python scripts on it.

4.2.3 SUBSYSTEM INTERFACES

The two interfaces are 1) Receiving the SVG file and storing it in memory, 2) Interface for running the SVG file through the python scripts.

Table 3: Subsystem interfaces

ID	Description	Inputs	Outputs
#03	Drawing/image storage	SVG file	N/A
#04	Interface for loading SVG file to run on Python IDE	SVG file	G code

5 Robotic Arm Software Layer Subsystems

The Arm Software Layer consists of the software that will be used to control the subsystems within the Hardware layer. The Arm Software layer consists of one subsystem, the Marlin software subsystem. This subsystem receives GCode from the communications subsystem via USB and uses these instructions to command the components in the Hardware system. The commands sent will be interruption based movement commands. The Marlin software will run on an Arduino board within the Hardware system.

5.1 Marlin Software Subsystem

The Marlin software subsystem is the subsystem that controls sends commands to the components within the Hardware system to move the robotic arm.

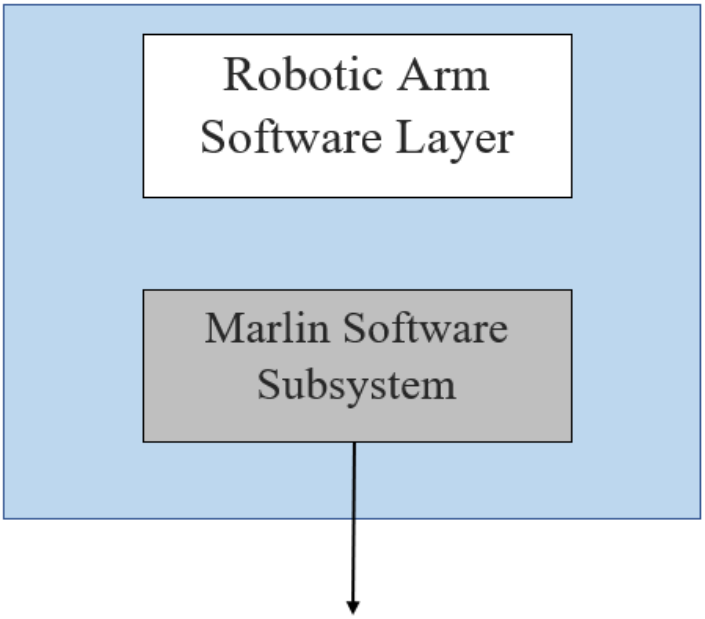

Figure 5: Marlin Software subsystem description diagram

5.1.1 Assumptions

The Marlin software is stated to be compatible with Arduino boards. It can be assumed that it is compatible with the one that will be used for this system.

5.1.2 Responsibilities

The Marlin software subsystem oversees controlling the Arduino board by feeding it commands to move the arm. It can read GCode inputs which will be received from the communication subsystem. The Marlin subsystem can control every aspect of the Arduino board, including domains such as lights and temperature, although these aspects will not be applicable to this project.

5.1.3 SUBSYSTEM INTERFACES

Table 4: Subsystem interfaces

ID	Description	Inputs	Outputs
#05	USB communication between the communication subsystem and the Marlin software subsystem	GCode from the Python IDE.	None
#06	Communication between the Marlin software subsystem and the Hardware system components.	None	Interruption based movement commands.

6 Hardware Layer Subsystems

The hardware layer consists of an Arduino Mega 2560, a Ramps 1.4 board, Motor drivers and the SCARA 4 axis arm robot.

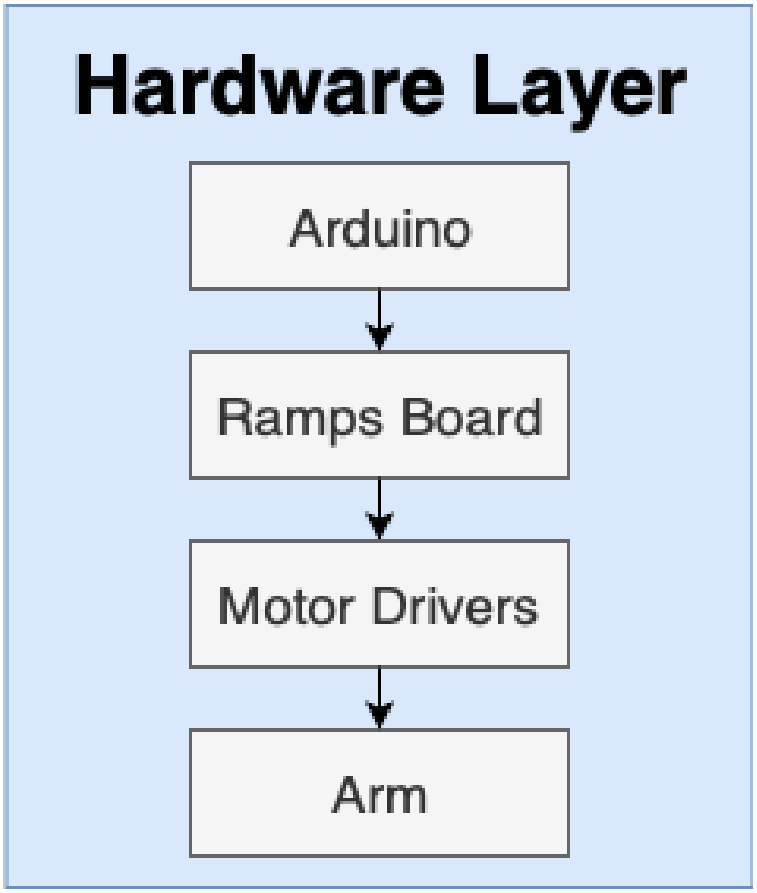

Figure 6: Example subsystem description diagram

6.1 Arduino

The Arduino Mega 2560 runs the Marlin Firmware and communicates with the ramps board.

6.1.1 Assumptions

It is assumed the Marlin Firmware will work correctly on the Arduino Mega 2560 as per its github page. It is also assumed that the Marlin Firmware will be able to control the arm correctly using given G-code.

6.1.2 Responsibilities

The Arduino Mega will be running the Marlin Firmware. The Arduino will be used to send control voltages to the Ramps board, the control voltages will be determined by the software.

6.1.3 Arduino Connections

Table 5: Arduino Connections

ID	Description	Inputs	Outputs
#01	Communication Wire	G-code data	Received Data Confirmation
#02	Ramps Board	N/A	Arm Control Signals

6.2 Ramps Board

The RepRap Arduino Mega Polulu Shield amplifies the control signals received from the arduino to send to the motor drivers in order to control the motors.

6.2.1 Assumptions

It is assumed that the RAMPS Board will work correctly with the arduino.

6.2.2 Responsibilities

The Ramps board will amplify the control signals from the arduino and send them to each of the motor boards in order to control the robotic arms movement.

6.2.3 Ramps Board Connections

Table 6: Ramps Board Connections

ID	Description	Inputs	Outputs
#01	Arduino	Arm Control Signals	N/A
#02	Motor Drivers	N/A	Motor Control Signals

6.3 Motor Drivers

The motor drivers take signals in order to control the motors of the robot.

6.3.1 Assumptions

It is assumed that the motor boards supplied witht the robotic arm are the correct ones and that they will work with the Ramps board.

6.3.2 Responsibilities

The motor drivers will take in control signals and supply the power needed to the motors in order to operate them.

6.3.3 Motor Drivers Connections

Table 7: Motor Drivers Connections

ID	Description	Inputs	Outputs
#01	Ramps board	Motor Control Signals	Power to control motor
#02	Power supply	Power	N/A

6.4 Arm

The arm is controlled by the movement of its motors

6.4.1 Assumptions

It is assumed that the robotics arm be predictable in its movement once it is configured correctly.

6.4.2 Responsibilities

The robotic arm houses 4 electric motors that control its movement. The arm it self does not take any input or output signals.

6.4.3 Arm Connections

Table 8: Arm Connections

ID	Description	Inputs	Outputs
#01	N/A	N/A	N/A

REFERENCES