

Car POLY

Final Report

5 조 Go with Ko

정보컴퓨터공학과 **201020217** 강동현

정보컴퓨터공학과 **201020268** 박준영

정보컴퓨터공학과 **201020283** 송범진

정보컴퓨터공학과 **201020340** 전현성

목차

1. 프로젝트 목표.....	3
2. 분석.....	4
3. 응용 시스템 제작 (설계 및 구현).....	4
3.1. 응용 시스템의 구조	4
3.1.1. Arduino Sensing	4
3.1.2. Android.....	5
3.1.3. Embedded Board	7
3.2. 세부 알고리즘의 개요.....	9
3.2.1. Arduino Sensing	9
3.2.2. Android.....	9
3.2.3. Embedded Board	10
3.2.4. Server	13
3.3. 전체 알고리즘의 개요.....	14
4. 변동사항.....	16
5. 시험평가.....	16
5.1. 프로그램 컴파일 환경 및 방법, 실행 방법.....	16
5.1.1. 프로그램 컴파일 환경 및 방법.....	16
5.1.2. 실행 방법	17
5.2. 테스트 및 개선 사항.....	17
5.3. 미 구현 사항.....	17
6. 첨부.....	18
6.1. 조원 별 역할 분담	18
6.2. 회의록.....	18
6.3. 일정	20

1. 프로젝트 목표

사람과 자동차 간의 **Communication service**의 구현을 목표로 한다. **Embedded Device**와 **Arduino**를 이용하여, 실제 자동차의 일부 기능을 구현하고, 사람이 스마트폰을 활용하여, 자동차와 **Communication**을 가능하도록 한다. 구체적인 **Project**의 목표는 다음과 같다.

- **스마트키의 구현**
기존 시중에서 존재하는 스마트키의 기능을 스마트폰으로 구현한다. 사용자는 스마트폰을 이용하여, 자동차의 문을 열고, 닫을 수 있다.
- **다른 사용자에게 권한 부여**
사용자는 본인이 아닌 다른 사용자에게 자동차에 대한 권한을 부여할 수 있다.
- **자동사고 신고 시스템**
자동차는 사고가 발생할 위험이 큰 운송수단이다. 하루에도 수많은 사고가 발생한다. 본 Project에서는 사용자에게 위급한 상황이 발생했을 때, 자동으로 신고하는 기능을 구현한다. 또는 사용자가 자동차를 사용하고 있지 않을 때에도 자동차에 대한 사고 시, 사용자에게 통보한다.

위와 같은 자동차에 대한 **IOT**의 구현을 목표로 한다.

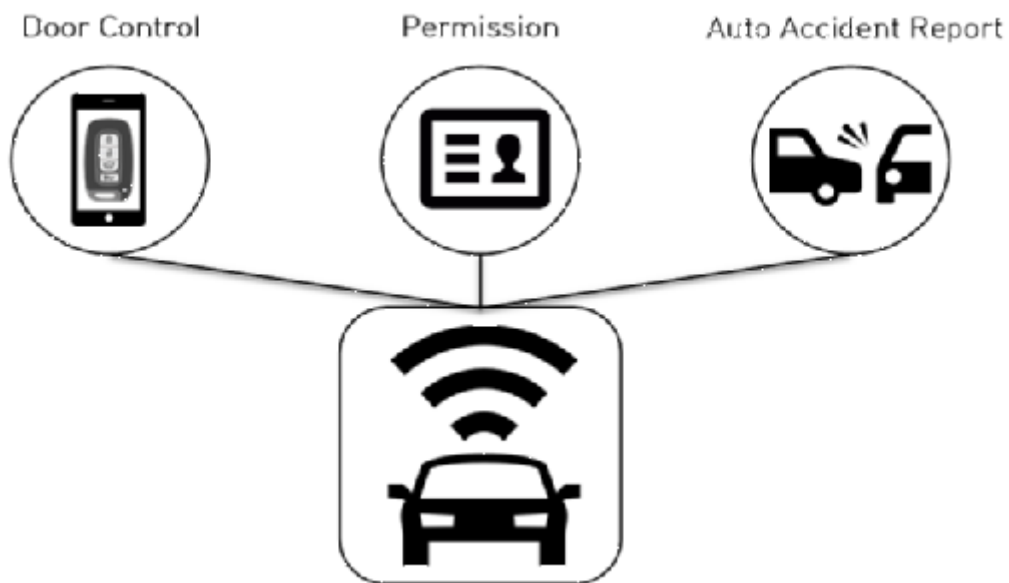


Figure 1 기능적 개요

2. 분석

본 Project에서는 사용자가 직접 사용하는 스마트폰과 Embedded Device 와 Arduino 를 이용하여 구현한 자동차로 나눌 수 있다. 이 두 개체는 인터넷을 통하여 Communication 한다. 아래 그림은 본 Project 의 전반적인 개요를 나타낸다.

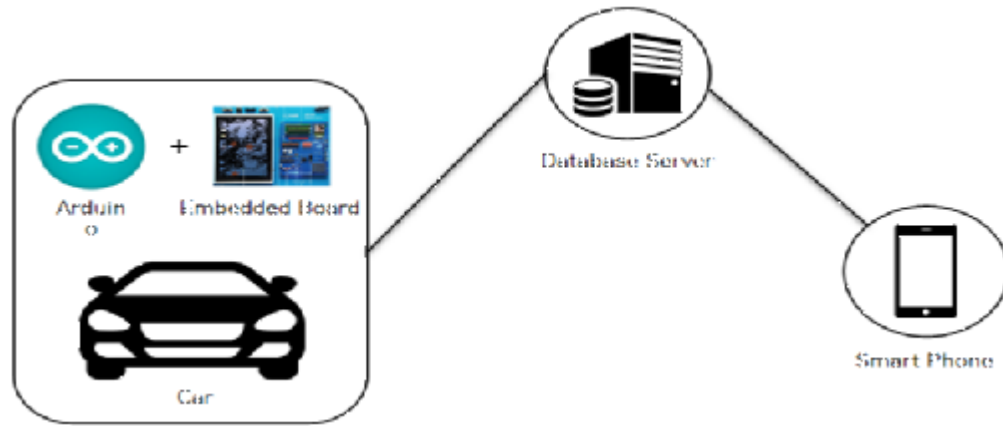


Figure 2 전체 개요

먼저, 자동차를 나타내는 Arduino 와 Embedded device 를 보면, Arduino 는 자동차 외부에 위치한 센서로 이용한다. 이 장치에 충격이나 충돌을 통해, 입력이 이루어지면, Arduino 자체에서 충격을 분석하여, 정차 사고와 주행 사고를 구분하여 Embedded Device 에 전달한다. 이때 Arduino 는 진동 센서와 가속도 센서, 두 종류의 센서를 사용하는데, 정차 사고는 진동 센서에서만 입력을 받을 때, 주행 사고는 두 가지 센서 모두 입력 받을 때, 구분한다. Embedded device 는 Arduino 로부터 전달받은 data 를 서버에게 전달한다.

사용자가 직접 접하게 되는 스마트폰에서는 휴대폰 번호와 설정한 비밀번호를 통해 자동차와 연결한다. 스마트 폰을 통해 사용자는 자동차의 문을 열고 닫을 수 있으며, 해당 차량에 대한 권한을 다른 사용자에게 부여할 수도 있다. 권한 기능을 구현하게 됨으로써, 자동차도 컴퓨터처럼 다중 계정(one-to-many)을 등록할 수 있다. 또 다른 기능으로는, 사용자는 스마트폰 Application 을 통하여, 자동차에 대한 최근 사용정보도 확인할 수 있고, 한 대의 차량에 여러 명의 사용자가 권한을 갖는 것처럼, 한 명의 사람도 여러 대의 차량을 사용할 수 있다.

3. 응용 시스템 제작 (설계 및 구현)

3.1. 응용 시스템의 구조

3.1.1. Arduino Sensing

Arduino 는 Embedded device 와 함께, 자동차를 구현한다. Arduino 를 이용하여, 자동차에게 가해지는 충격을 측정하고, 이에 대한 정보를 Embedded device 로 전송한다. Arduino 로 측정한 충격량은 입력된 Sensor 에 따라 두 가지 상황으로 바뀌는 데, 이는 다음과 같다.

구분	Vibration Sensor	Vibration Sensor + Gyro Sensor
상황	정차 시 사고 발생	주행 시 사고 발생

Table 1 Arduino Sensors

Arduino 내에서 Data 를 측정하기 위한 프로그램의 구조는 다음과 같다. 아래의 그림과 같이 최초의 setup 작업을 마친 뒤, 두 개의 Thread 를 생성한다. 이 두 Thread 는 각자

반복적으로 Sensor로부터 측정된 값을 받아온다. 각각의 Thread는 입력이 있을 시에 Global Variable인 gyro와 vibe라는 int형 변수를 변환시킨다. Loop()는 Thread로부터 변경된 Global Variable이 변경되어 vibe가 1이 되거나, vibe가 1이 되고, gyro 또한 1이 되는 순간까지 반복되어 상태를 확인한다. 그러다 변수에 변화가 생기면, vibe가 1이 되는 상황을 정지상태의 사고로, vibe와 gyro가 1이 되는 상황을 주행상태의 사고로 판단하여 그에 따른 정보를 Embedded device에 전송한다.

3.1.2. Android

Android가 본 Project에서 제공하는 응용 서비스는 다음과 같다.

	<div data-bbox="963 766 1070 799" data-label="Section-Header"> <h4>내 정보</h4> </div> <p>사용자는 Application을 실행시키고 Login 과정을 거치면, 본인의 정보를 받을 수 있다. 이는 사용자가 권한을 갖고 있는 모든 차량의 정보를 포함하고, 현재 이용할 차량 또한 포함된다.</p>
	<div data-bbox="885 1348 1152 1382" data-label="Section-Header"> <h4>최근 차량 사용 시간</h4> </div> <p>Server와의 Communication을 통하여, 가장 최근의 차량을 이용한 시간을 알 수 있다.</p>

	<p>권한 부여</p> <p>Android 를 통해, 현재 control 하고 있는 자동차에 대한 권한을 다른 사람에게 줄 수 있다. 권한을 받은 다른 사용자는 권한 부여 기능 이외에 자동차에 대한 권한을 갖게 된다. 다른 사용자에게 권한을 부여하는 기능은 Owner 의 절대적인 기능이다.</p>
	<p>자동차 문 개폐</p> <p>Android 는 사용자로부터 자동차 문 control 에 대한 정보를 받는다. 받은 정보를 Server 에게 전달하고 자동차의 문 상태를 변경한다.</p>
	<p>회원가입</p> <p>최초 본 System 을 사용할 때, 회원가입을 실시한다. 이때 사용자는 사용할 차량의 Owner 여야 하며, 일반 Guest 권한으로는 회원가입을 실시할 수 없다.</p>


	권한추가
	<p>권한 메뉴에서 권한 추가 버튼을 누르면 다음과 같이 화면이 출력된다. 이는 회원가입과 비슷한 형태이며, 차량의 Owner 권한을 가지고 있는 사용자가 본인 차량에 대한 권한을 타인에게 부여할 수 있는 기능이다.</p>

Table 2 Android 실행 화면

3.1.3. Embedded Board

	Tablet 화면
 	<p>초기 실행시에는 좌측 로그인요청화면이 출력된다. 로그인을 진행하면 우측 그림처럼 고리가 연결되어 로그인이 되었음을 알려주고, Lock/Un-Lock, Power-On/Power-Off 상태에 따라 맞는 이미지를 출력한다.</p>
 	<p>시동기능 – Full color LED, Dip switch</p> <p>시동이 꺼져있을때는 좌측 그림처럼 Full color LED 에 빨간불이 켜져있다. Dip switch 의 첫번째 스위치를 올리면 시동상태가 on 으로 바뀌고 우측 그림처럼 Full color LED 에 초록불이 켜진다. 다시 dip switch 를 내리면 좌측 그림처럼 다시 빨간불이 켜진다.</p>
	<p>연료기능 – BUS LED</p> <p>최초 로그인시에 BUS LED 를 모두 켜워서 연료가 가득 찬 것을 보여준다. 이후 특정시간마다 1 칸씩 LED 를 꺼주고, 마지막 1 칸 남았을 때 모두 깜빡거리며 사용자에게 연료가 거의 다 떨어졌음을 경고해준다.</p>
	<p>사용시간 – 7-segment</p> <p>새로운 사용자가 로그인할 때마다 7segment 를 초기화해서 사용자가 시스템을 이용한 시간을 보여준다.</p>

	<p>사용자 권한 - OLED</p> <p>사용자의 permit에 따라 owner 라면 OLED에 CarPOLY 로고와 함께 Owner 라고 출력해주고, guest 라면 Guest 라고 출력해준다. 새로운 사용자가 로그인할 때마다 해당 사용자의 permit을 확인해서 이미지를 바꿔준다.</p>
	<p>거울기능 - Camera, Touch panel</p> <p>Touch panel과 Camera를 이용하여 거울 기능을 구현하였다. Tablet 화면을 터치하면 touch panel이 터치를 인식해서 camera를 실행시킨다. Camera는 5초간 거울을 보여주고 5초가 지나면 다시 기존의 frame buffer 화면으로 돌아간다.</p>
	<p>사고 자동 신고기능 - Dot matrix, Text LCD, Buzzer, Key matrix</p> <p>아두이노 센서로 정차중사고, 주행중사고 상황을 인식한다. 정차중사고인 경우 Text LCD에 정차중사고, 보험사에 연락한다고 출력해주고 부저를 울려주고, dot matrix로 반짝반짝해준다. 주행중사고인 경우에도 마찬가지로 부저를 울리고 dot matrix를 반짝반짝해주고, Text LCD에 주행중사고, 119와 보험사에 연락한다고 출력해준다. 추가로 사고상황이 아님에도 자동으로 신고를 할 수도 있으므로 Key matrix 입력으로 사용자가 신고를 취소하는 기능도 넣었다. 운전중에 정확한 버튼을 누르는 힘들기 때문에 Key matrix의 어떤 버튼이든 누르면 신고를 취소할 수 있다.</p>

Table 3 Embedded Board 구조

3.2. 세부 알고리즘의 개요

3.2.1. Arduino Sensing

Arduino 는 자동차 내의 충격을 인지하는 센서 역할을 한다. 수집한 data 를 Arduino 가 정차 사고, 주행 중 사고로 판단하여, Embedded device 에 전송한다. Arduino 의 전반적인 flow chart 는 다음과 같다.

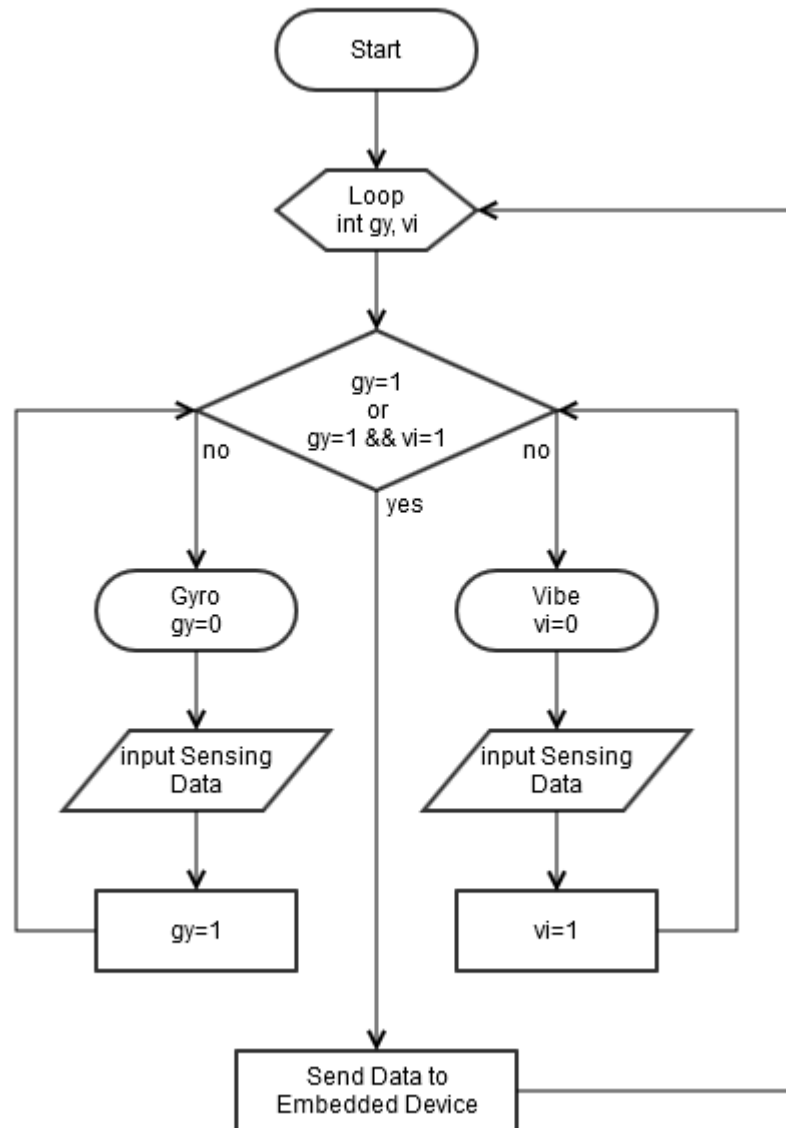


Figure 3 Arduino Flowchart

3.2.2. Android

Android 는 DB server 와의 Communication 이 주된 작업이다. Server 와 Communication 을 시도하는 순간은 다음과 같다.

1. Login
2. Join
3. 권한 부여
4. Door control

5. Logout

먼저, **Application** 실행 시, **Login** 화면이 나타난다. **Login** 화면에서는 **Login** 을 하거나, **Join** 버튼을 눌러, 회원가입 화면으로 전환할 수 있다. 먼저 회원가입을 보면, **Server** 로 회원가입 시 필요한 정보들을 전송한다. 이때, 휴대폰 번호로 중복 체크 또한 실시한다. **Login** 을 시도할 경우, **Server** 에 입력된 휴대폰 번호와 비밀번호를 판단하여 **Login** 을 실시한다. **Login** 과정시, 사용자와 차량에 대한 대략적인 정보를 받아온다. 권한 부여 기능에서는 회원가입과 같은 방법으로 실시한다. **Door Control** 시에도 현재 **Android** 내의 **Door** 상태와, **Server** 상의 **Door** 상태를 모두 확인하여, 동기화를 한다. **Logout** 은 **Logout** 을 함과 동시에 **Server** 에 **Login** 상태를 변경해주어, **Embedded device** 가 해당 정보를 수신할 수 있게 한다. **Android** 의 전체적인 **flow chart** 는 다음과 같다.

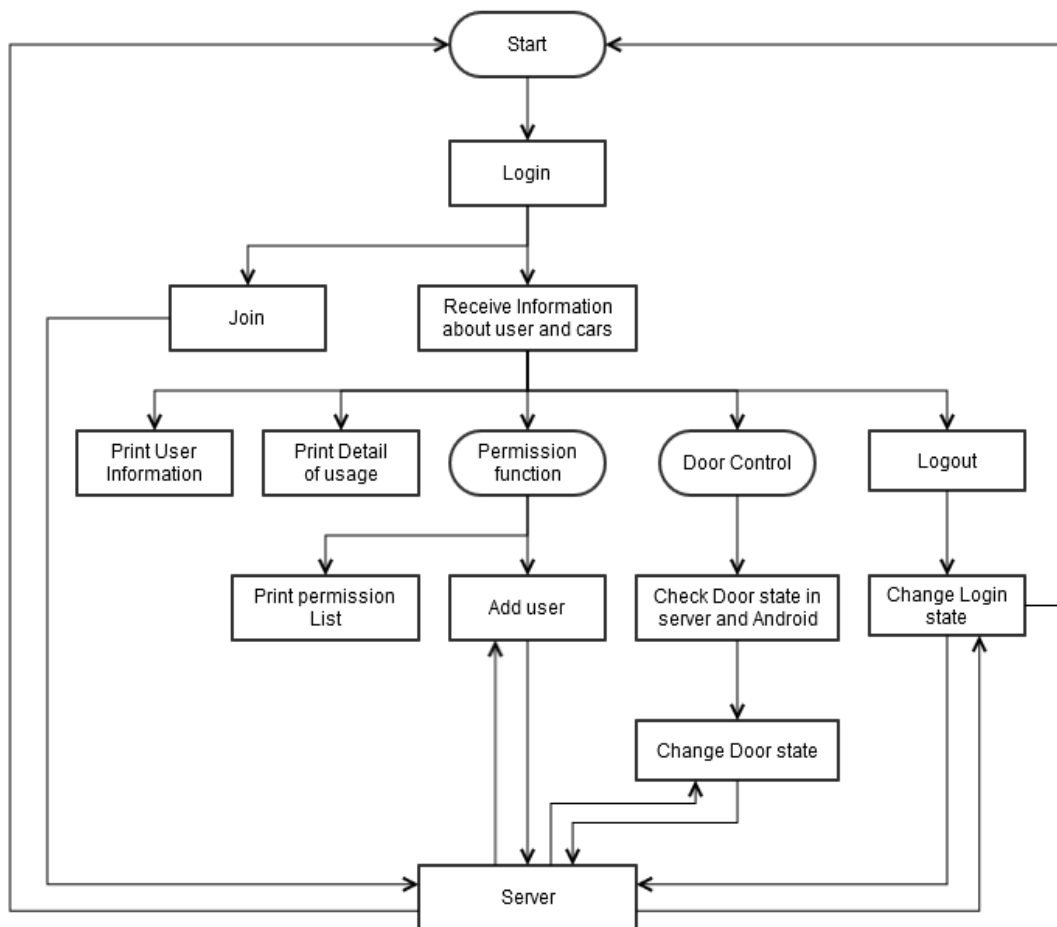


Figure 4 Android Flowchart

3.2.3. Embedded Board

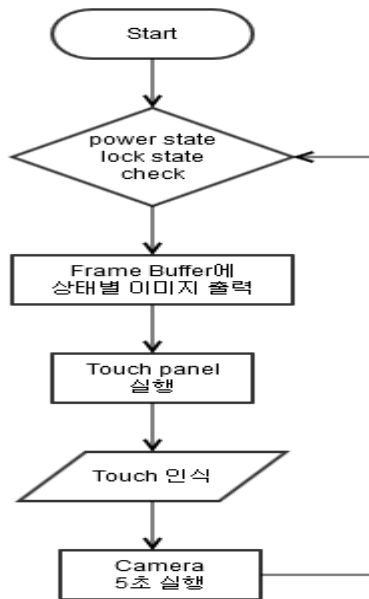


Figure 5 거울기능 Flowchart

거울 기능

미러기능의 경우는 Embedded Device 의 TouchPannel 의 기능을 이용한 것이다. TouchPannel 를 Touch 하면 조건문에 따라 카메라 모듈을 실행시키고 이를 Frame Buffer 를 이용, Tablet 화면에 출력하게 된다. 미러기능이 시작되면 약 5 초후에 자동적으로 종료된다.

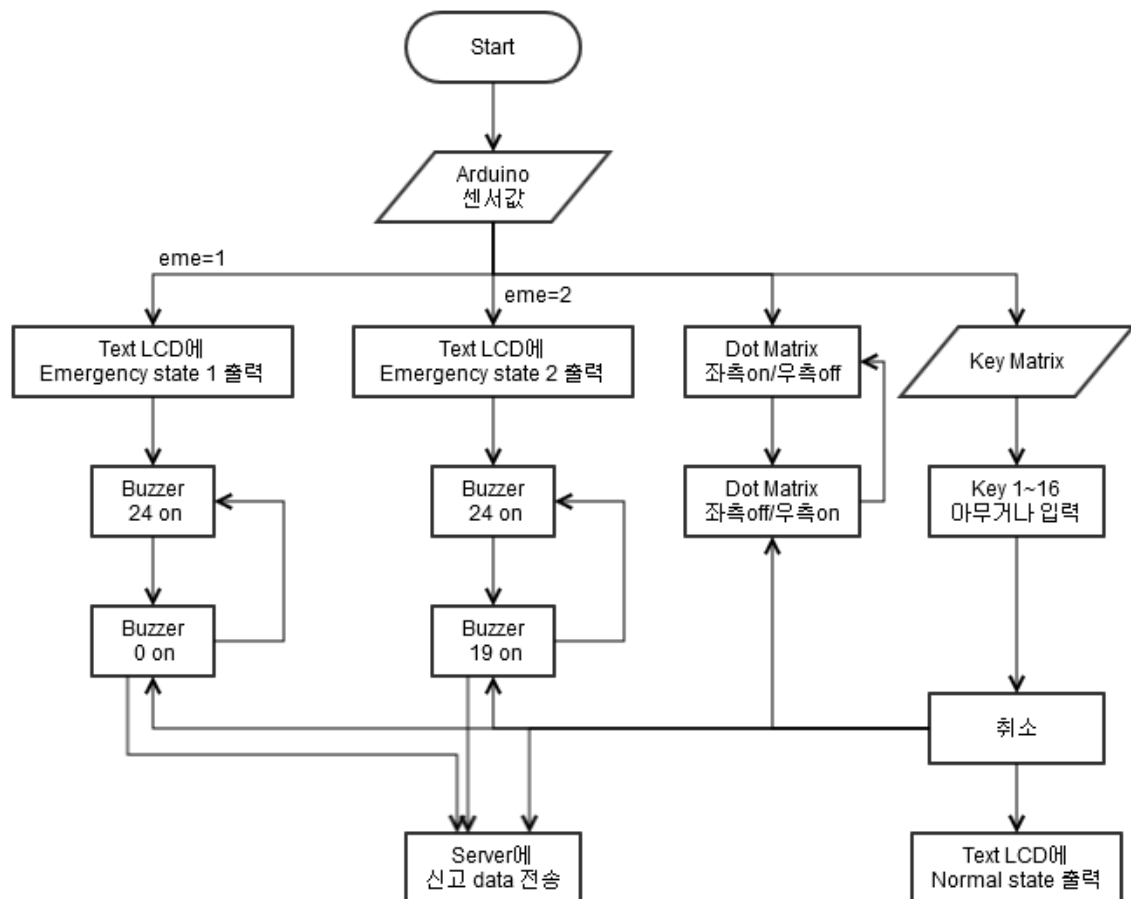


Figure 6 사고발생 Flowchart

사고발생시 경고, 자동신고기능

신고의 경우는 Arduino 의 자이로 센서와 진동센서를 이용, 정보를 받아온다. 진동센서에서만 충격이 감지되는 경우 정차중 사고로 탐지하여 1 이라는 값을, 자이로센서의

급격한 변화와 진동센서에서의 충격이 감지되는 경우에는 2 라는 값을 Embedded Device 로 전송하고 Device 에서는 해당 데이터를 eme 변수에 저장하게 된다. thread 로 돌고 있는 Buzzer 와 Dot matrix 는 eme 의 값의 변화에 따라 각자 정해진 동작을 하게 된다. 이 때, 9 초 내의 Key matrix 의 버튼을 누르게 되면 동작이 정지되고, 신고 기능이 취소 되어 DB 로 신고 전송값이 전송되지 않고, Key matrix 의 입력값이 없다면, 정상적으로 신고처리 되게 된다.

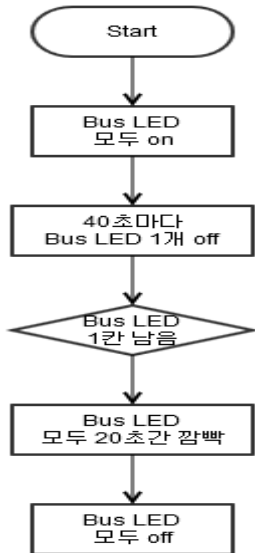


Figure 7 Fuel Flowchart

Fuel 기능

최초 로그인시 BUS LED 를 모두 켜준다. 특정 시간마다 BUS LED 를 우측부터 하나씩 끈다. 이번 프로젝트에서는 짧은 시간동안 데모를 하기 위하여 40 초로 설정하였다. BUS LED 가 1 칸 남았을 때 20 초간 BUS LED 를 모두 깜빡여서 사용자에게 연료가 얼마 남지 않았음을 경고한다. 그 후에 연료가 다 떨어지면 BUS LED 를 모두 꺼준다.

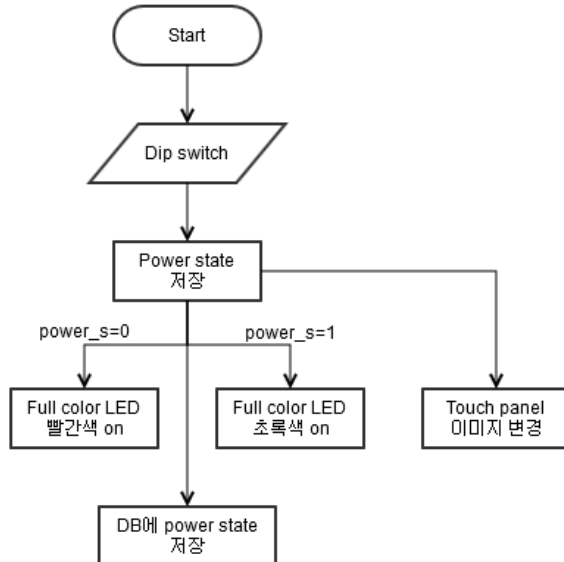


Figure 8 시동기능 Flowchart

시동기능

시동기능의 경우는 Embedded Device 의 Dip Switch 와 Fullcolor LED, Frame Buffer 를 이용하여 On/Off 상태를 조절한다. Dip Switch 의 상태를 1 로 만들게 되면 Fullcolor LED 의 색상을 Green 색상으로 변경하고, Database 에 해당 부분을 업데이트한다. 최초 시동을 걸게 되면 시작시간을 Set 해주고 종료시간을 Null 값을 넣어준다. Switch 의 상태를 2 로 만들게 되면 Fullcolor LED 의 색상을 Red 색상으로 변경하고, 마찬가지로 Database 에 해당 부분을 업데이트 한다. 시동을 끄게 되면 종료시간을 Set 하여, 해당 사용자가 얼마나 사용하였는 지 확인할 수 있도록 한다. 그리고 해당 상태 변화에 따라 Frame Buffer 를 이용, Tablet 화면을 변화 시킨다.

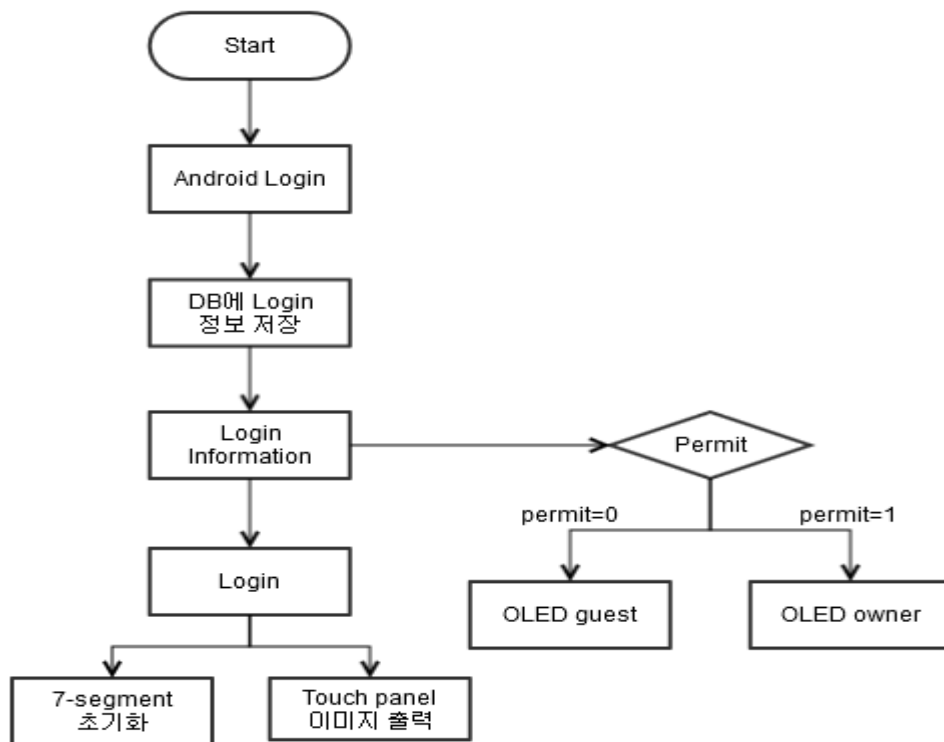


Figure 9 Login 기능 Flowchart

Login 기능

로그인의 경우는 Android로부터 로그인 정보를 Database에 저장하고, Embedded Device는 Database에서 해당 데이터를 변수로 저장받는다. 이 중 Permit의 정보를 가지고 Owner와 Guest를 구분할 수 있고 이를 OLED에 이미지를 출력하여 사용하는 User에게 알릴 수 있도록 조치하였다. 또한, 최초 로그인이 안되었는 상황에서는 Tablet에 이미지를 통하여 로그인이 안된 상태임을 User에게 알리고 로그인을 할 수 있도록 하였다.

3.2.4. Server

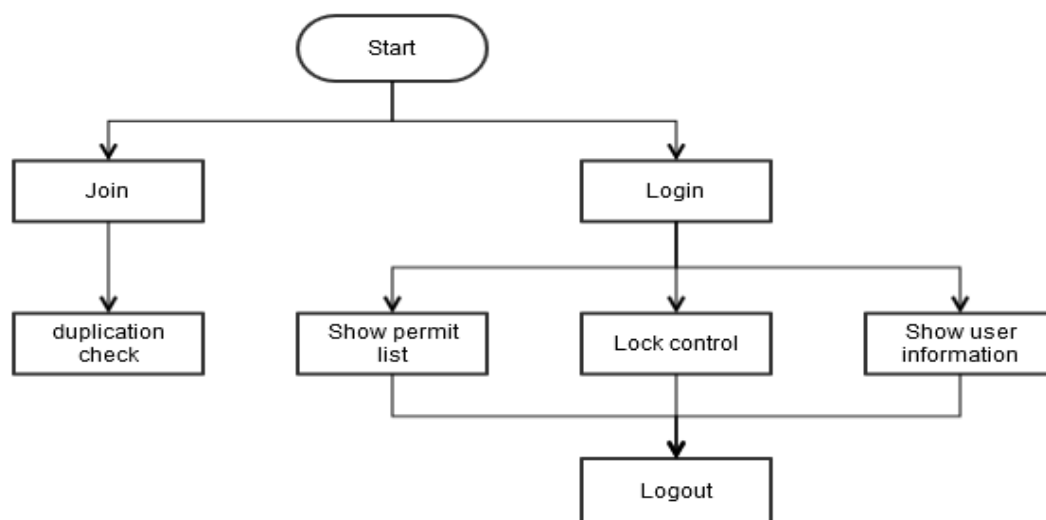


Figure 10 Server Flowchart

Server 는 Android 및 Embedded device 와의 communication 이 주된 역할이다. 각기 Device 들의 요청에 따라 적절한 data 를 전달해주거나, 데이터를 저장하거나 업데이트 한다. Server 가 android 와 embedded device 의 요청에 응답하는 기능들은 다음과 같다.

1. Join 및 User information 요청
2. Log in (회원가입)
3. 권한 부여
4. Door control
5. Emergency

첫째로 회원가입의 경우에는 Android 가 사용자로부터 입력 받은 데이터들을 서버에 전달한다. 입력받은 데이터들을 토대로 해당 사용자가 DB 에 존재하는 지 중복 여부를 체크한다. 중복된 사용자일 경우 가입에 실패했다는 메시지를 통해 사용자가 다시 가입요청을 할 수 있도록 한다. 중복되지 않은 사용자일 경우 가입이 정상적으로 완료되고 DB 에 사용자의 정보가 저장된다.

둘째로 로그인인 경우에는 Embedded device 와 Android 둘 모두가 Server 와 통신한다. 먼저 Android 에서 사용자가 서버에 로그인을 요청하고 서버는 로그인 요청에 응답하며 embedded device 도 사용자가 로그인 하였음을 알 수 있게 DB table 의 login column 을 update 하여 준다. DB connection 을 통해 DB table 을 일정 시간마다 체크하던 embedded device 는 login column 의 변화를 감지하고 사용자가 로그인 하였음을 알 수 있다. 그 후 로그인 요청에 대한 응답으로 Server 에서는 사용자의 정보들을 Android 에 보내주고 Android 는 이를 처리하여 사용자의 정보를 화면에 출력할 수 있게 된다.

셋째로 권한 부여의 경우에는 두번째와 마찬가지로 Embedded device 와 Android 모두가 Server 와 통신한다. 먼저 Android 에서 권한 부여 기능을 통해 다른 사용자에게 권한을 부여할 때 Server 에 새로 등록될 사용자의 정보를 보내준다. Server 는 받은 데이터를 DB 에 삽입한다. 이때 권한을 부여받은 새로운 사용자는 Owner 의 권한이 아닌 Guest 의 권한으로써 DB table 에 permit column 에 삽입된다. 부여가 완료되면 DB connection 을 통해 DB table 을 체크하던 embedded device 가 permit column 의 변화를 감지하고 그에 맞는 처리를 할 수 있게 된다.

넷째로 Door control 기능이다. 마찬가지로 Embedded device 와 Android 모두가 Server 와 통신한다. 먼저 Android 에서 Lock / Unlock 버튼을 누르게 되면 Server 에 요청이 전송된다. Server 는 요청받은 Lock 의 값에 따라 DB Table locstate column 을 update 하여 준다. 이에 Embedded device 는 DB connection 을 통해 table 의 변화를 감지하게 되고 변화된 값에 따라 처리를 진행한다.

마지막으로 Emergency 가 발생했을 경우에는 Embedded device 가 Server 와 통신한다. 먼저 Arduino 에서 충격 센서를 통해 들어온 데이터가 일정 수준을 넘을 경우, 충격의 sensing data 가 embedded device 로 넘겨진다. 넘겨받은 data 를 embedded device 는 즉시 Server 의 DB table 에 emergency 상태(주행 중 사고, 정지상태 사고) 와 해당 차량의 운전자의 이름을 삽입한다. 이때 우리가 유관기관으로 가정한 <http://152.149.43.194/emergency.php> 웹페이지는 DB Table 을 관찰하다가 새로운 사고 data 가 table 에 들어오면 즉시 웹 페이지에 운전자의 이름과 해당 사고의 종류를 출력한다.

3.3. 전체 알고리즘의 개요

Android 에서 사용자가 로그인을 하면 해당 사용자의 로그인 정보를 DB 에 저장을 해준다. Embedded 보드는 DB 를 체크하여 사용자가 로그인을 하면 Tablet 에 로그인된 화면을 출력해주고, permit 을 체크하여 사용자의 권한에 맞는 로고를 OLED 에 출력해준다. 또 Text LCD 에 Normal state 를 출력해주고, Bus LED 를 실행하고, 7-Segment 에 이용시간을

출력해준다. 새로운 사용자가 Android 에서 로그인을 하면 DB 에 새로운 로그인 정보를 저장하고 Embedded Board 는 새로운 사용자의 permit 을 체크하여 새로운 사용자의 권한에 맞는 로고를 OLED 에 출력해주면서 7-Segment 를 초기화해준다.

사용자가 Android 에서 Lock / Unlock 을 변경해주면 Touch panel 의 화면 이미지를 변경해주고, Dip switch 로 power on / off 를 변경해주면 Touch panel 의 화면 이미지 변경과 동시에 Full color LED 의 색상을 변경해준다. 또 Touch panel 이 touch 를 인식하면 Camera 를 실행하며 거울기능을 활성화하고 5 초후에 거울 기능을 종료하고 다시 touch panel 을 원래대로 출력해준다.

Arduino 에서 센싱값이 인식이 되면 Buzzer 가 울리고 Dot matrix 가 깜빡거리면서 경고를 알려주고, Text LCD 에 사고상황에 맞는 메시지를 출력해준다. 동시에 Server 에 사고상황을 알려주어서 자동으로 신고하는 기능을 수행한다. 이때 사고가 크지 않아서 신고를 할 필요가 없는 경우에는 Key matrix 버튼을 눌러서 Buzzer, Dot matrix 를 모두 취소하고 server 에 신고하는 것도 취소해준다. 이때 Text LCD 에 다시 Normal state 를 출력해준다.

시스템에서 Embedded Board 는 3 초마다 Query 문을 통하여 DB 에서 각종 데이터를 받고, DB 에 데이터를 준다.

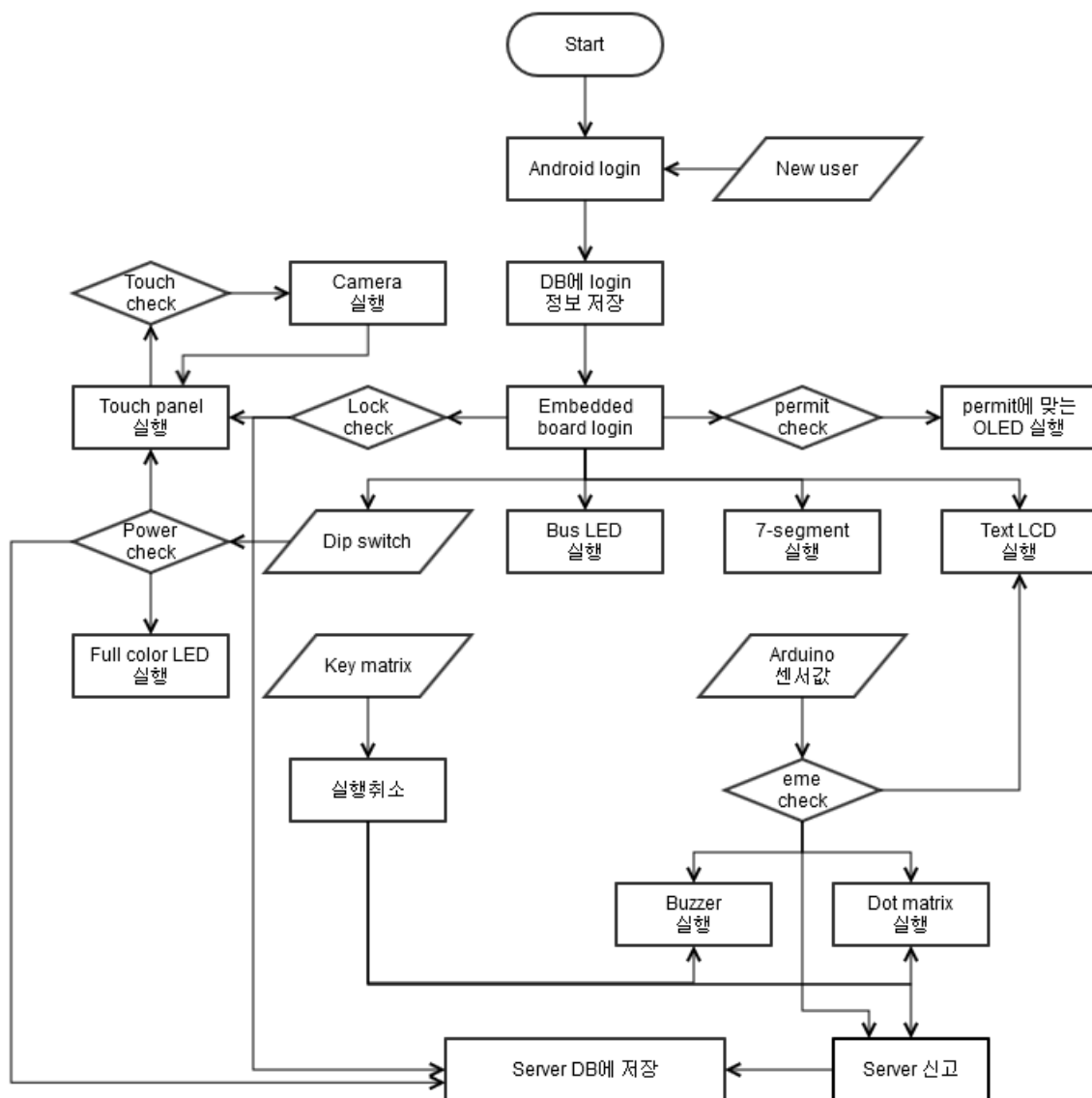


Figure 11 전체알고리즘 Flowchart

4. 변동사항

기존 제안서에서 언급했지만, 최종 발표 시 구현하지 않은 부분이 있다. 내용은 다음과 같다.

제안서에서의 내용	변동된 내용	이유
NFC 기능 스마트 폰의 NFC 를 사용하여, 차량의 Door control 을 실시한다.	온라인을 통한 Door control	NFC 를 구현하기 위해서는 추가적인 Arduino NFC 모듈이 필요하고, 단거리에서 Door 를 control 하는 것은 새롭게 개념이기 때문에.

Table 4 변동사항

또한, 제안서에서도 언급하지 않고, 최종 발표 시에도 구현하지 않았지만 기능 구현을 시도했던 것이 있다. 내용은 다음과 같다.

5. 시험평가

5.1. 프로그램 컴파일 환경 및 방법, 실행 방법

5.1.1. 프로그램 컴파일 환경 및 방법

본 프로젝트에는 여러 가지 프로그램이 Combination 을 이룬다. Embedded Device 만 가지고 본 프로젝트를 구현하여 실행할 수 없고, Android 도 마찬가지다. 각각의 다른 프로그램들로 이뤄진 프로젝트기 때문에, 컴파일 환경이나 방법도 각각 다르다. 각 프로그램 별 컴파일 및 방법은 다음과 같다.

구분	컴파일 환경	컴파일 방법
Arduino	개발 환경 : Sketch IDE (Integrated Development Environment) Arduino model : Arduino Mega ADK	Sketch IDE 를 통해 디버깅 및 컴파일 수행
Android	개발 환경 : Android Studio 1.2 Android version : Android kitkat 4.4	Emulator 를 이용하지 않고 실제 스마트폰(LG G3)를 이용, Android studio 내에서 compile 수행
Embedded Device	운영체제 : Linux Kernel 3.0.15 컴파일러 : arm-2009q3-67-arm-none-linux-gnu-eabi-i686	Terminal 에서 명령어를 이용하여 compile 수행
Server	운영체제 : Windows server 2008 Enterprise R2 서버스크립트 컴파일러 : PHP 5.2.12 웹서버 : Apache 2.2.14 데이터베이스 : MySQL 5.1.41 테스트 브라우저 : Chrome 43.0.2357.124m	PHP 를 이용하여 compile 수행

Table 5 프로그램 컴파일 환경 및 방법

5.1.2. 실행 방법

구분	실행 방법
Arduino	IDE 를 통해 program 을 Arduino 에 기록한다. 그 후, Arduino 는 기록되어 있는 code 를 전류가 흐르게 되면 실행하게 된다.
Android	스마트폰을 이용해 실행시킨다.
Embedded Device	호스트 컴퓨터에서 cross-compile 한 실행파일을 tftp 를 이용하여 device 로 전송, 전송 받은 실행 파일을 embedded device 의 terminal 에서 명령어로 실행
Server	PHP script 를 작성하여 httdocs 디렉토리에 올려 놓는다. 올려 놓은 php script 들은 android device 의 요청이 있을 시 이에 응답하여 요청받은 데이터를 전달함

Table 6 실행방법

5.2. 테스트 및 개선 사항

Embedded device 는 Android 에서 login 을 실시하면, server 로부터 login 상태를 인지하고, 준비 상태를 실시한다. 이때 여러 가지 Thread 를 실행시키는데, eCube 의 리소스 부족으로 인한 shutdown 이 되기도 한다. 리소스 부족의 이유는 실시간으로 rssi(received signal strength indicator)값을 받아와 모든 device 들을 계속 call 해주기 때문인 것으로 보인다. 그 과정에서 device driver 들이 필요한 공통 data 가 변조되는 현상이 있다. 이 사항을 개선하려면 세 가지 방법이 있겠다. 첫 번째, eCube 말고 다른 좋은 embedded 시스템을 사용한다. 두 번째, 쓰레드 스케줄링을 정밀하게 하여 최대한 각 device 에 대한 부담을 줄여준다. 하지만 메인 프로그램의 sleep()함수의 변수 값을 각 쓰레드 마다 바꿔주었지만 여전히 busy 에러는 발생했다. 세 번째, 실용성은 없겠지만 sleep 함수의 시간을 길게 늘려 cpu 의 부담을 줄여준다.

Android 에서는 한글을 출력하는 것에 어려움이 있겠다. 이는 인코딩 방식으로 인한 어려움인데, UTF-8 방식 인코딩을 이용하면 간단히 해결가능하다.

5.3. 미 구현 사항

구분	구현을 시도했던 내용	이유
1	Android GCM 기능 사고 발생 시 사용자의 스마트폰으로 내용을 전달해주는 팝업을 생각하다가, Android 에 Google 에서 제공하는 GCM 을 이용하여 구현 하려고 했다.	Embedded device 의 결함으로 인해 장비교체를 실시하였고, 이에 장비의 구현 정도가 미비하게 되어, 장비에 시간을 집중하였다.

2	<p>실제 차량 모형 최종 발표 시 구현하였던 장난감 자동차가 아니라, 유아들이 실제로 탈 수 있는 자동차를 이용해, 문이 열리고 닫히는 기능을 실제로 보여주고자 했다.</p>	<p>문이 열리고 닫히는 기능을 구현하기 위해서는 Arduino sensor 외에도 동력을 전달할 수 있는 추가적인 동력원의 구매가 필요했다.</p>
3	<p>블랙박스 기능 본 프로젝트에서 정차 시 사고를 인지하는 기능이 있다. 해당 기능을 구현할 때, 실제 차량의 블랙박스의 기능을 추가하여 좀 더 효과적인 기능구현을 실시하고자 했다.</p>	<p>Embedded device 의 카메라는 생각보다 다루기 어려웠다. Process 자체가 무거웠고, 캡처한 화면을 server 를 통해 사용자에게 전달하는 데 걸리는 시간도 무시할 수 없었다. 차량 사고라면 긴급 상황일수도 있는데, 전송 속도를 생각하지 않는다면 안 된다고 판단했다.</p>

Table 7 미 구현 사항

6. 첨부

6.1. 조원 별 역할 분담

강동현 - 팀 리더 (Server 개발)

송범진 - 개발 (Arduino 및 Embedded device 전담)

전현성 - 개발 (Server 및 Embedded device 전담)

박준영 - 개발 (Arduino 및 Android)

6.2. 회의록

참석자	강동현 박준영 송범진 전현성	일시	2015 년 03 월 17 일
주요내용	프로젝트 주제 선정		
회의내용 및 결과	많은 후보들 중 Connected car 를 주제로 선정함. 각자 Connected car 에 대해 좋은 IoT 관련 idea 를 생각해오기로 하고 헤어짐.		

참석자	강동현 박준영 송범진 전현성	일시	2015 년 03 월 24 일
주요내용	프로젝트 제안서 작성		
회의내용 및 결과	생각해 온 idea 들을 종합하고 이를 바탕으로 Connected car 에 대한 추가적인 idea 구상 회의를 진행. 최종적으로 정리된 내용들을 바탕으로 제안서를 작성함.		

참석자	강동현 박준영 송범진 전현성	일시	2015 년 04 월 7 일
주요내용	아두이노 구상		
회의내용 및 결과	제안서에 작성한 내용을 바탕으로 프로젝트 구현을 위해 아두이노를 정확히 어떻게 구현할 것인지 회의를 진행함. 추가적인 센서를 구입하기로 결정.		

참석자	강동현 박준영 송범진 전현성	일시	2015 년 04 월 14 일
주요내용	아두이노 및 임베디드 보드 연동		
회의내용 및 결과	임베디드 보드와의 유선 통신을 통하여 센싱된 데이터들을 주고받는 방법에 대해 회의. 주행 중 사고와 주차상태 사고를 구분하는 알고리즘을 어떻게 정확하게 구현할지 논의함.		

참석자	강동현 박준영 송범진 전현성	일시	2015 년 05 월 12 일
주요내용	임베디드 보드 디바이스 드라이버 활용		
회의내용 및 결과	임베디드 보드의 디바이스 드라이버를 어떻게 활용할 것인지 회의함		

참석자	강동현 박준영 송범진 전현성	일시	2015 년 05 월 22 일
주요내용	사고 상황 발생시 알림		
회의내용 및 결과	GCM 을 활용한 push 알림 구현 여부에 대해 토론. 구현하지 않는 것으로 결정		

참석자	강동현 박준영 송범진 전현성	일시	2015 년 6 월 2 일
주요내용	최종 발표 및 데모 영상		
회의내용 및 결과	통합을 앞두고 최종 발표는 누가 할 것인지, 또한 어떻게 공들인 프로젝트를 데모영상을 통해 효과적으로 보여줄 수 있을지에 대해 함께 고민해보고 논의함.		

6.3. 일정

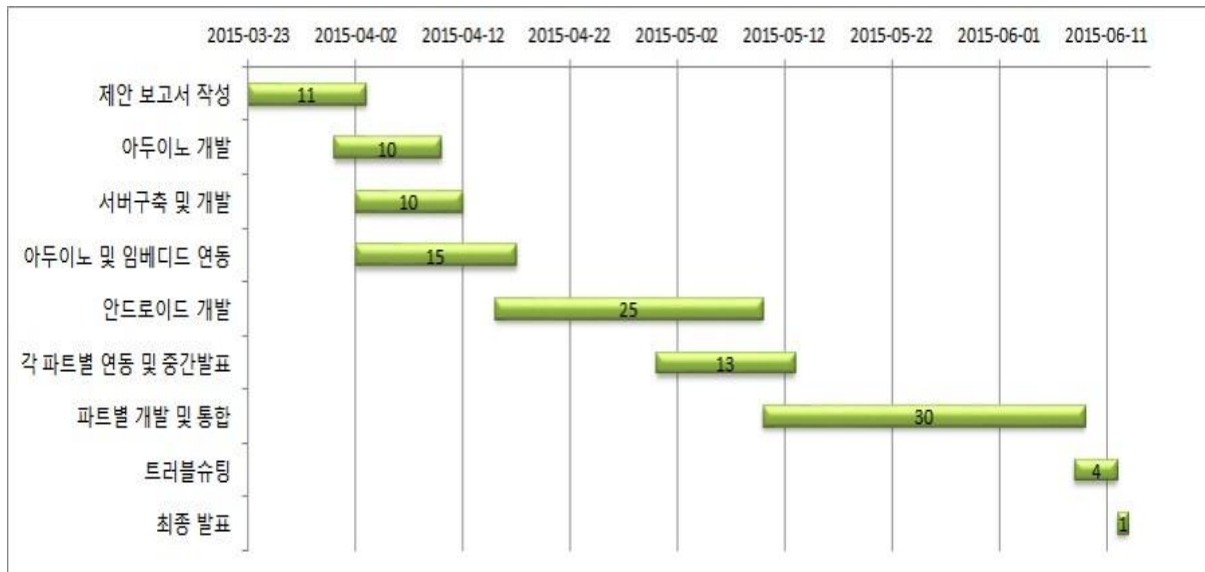


Figure 12 일정