**졸업 프로젝트**

**프로젝트 설계서(2차)**

-아두이노와 웹서버를 이용한 스마트홈 오토메이션-

고지혜 201311193

김예찬 201311200

1. **개요**
   1. **프로젝트의 주요기능**

1) 안드로이드 기기 등록

처음으로 사용하는 안드로이드 기기의 경우 등록 과정을 거친다. 해당 기기의 아이디를 홈 오토메이션 시스템의 데이터베이스에 저장하여, 등록된 아이디를 가진 기기가 명령을 보낼 때만 서버에서 명령을 아두이노로 보내게 된다.

2) 집안의 사물 상태 확인

집 안의 전자제품과 조명의 On/Off상태, 커튼의 열림/닫힌 상태 등 집 안의 사물들의 상태를 안드로이드 앱에서 보여준다. 사물들의 상태는 변경이 있을 때 마다 서브 아두이노에서 웹 서버로 보내어 안드로이드 앱으로 보낸다.

3) 집안의 사물 제어

안드로이드 앱에서 LTE및 Wifi를 이용하여 get 방식을 통해 서버로 정보를 보내면 그 정보를 데이터 베이스에 저장한다. 중앙 아두이노에서는 이더넷 쉴드를 통해 인터넷에 연결하여 서버에서 php 파일을 요청하여 읽어와 디비에 저장되어 있는 명령 중 최근 명령을 가져온다. 명령을 분석하여 해당 되는 물체의 서브 아두이노에 신호를 보내어 제어한다.

집과 멀리 떨어진 외부에서도 와이파이나 핸드폰 데이터 통신을 이용하여 서버에 접근할 수 있으므로 아두이노로 명령을 보내는 것이 가능해진다.

4) 자동 제어

4-1) GPS 체크

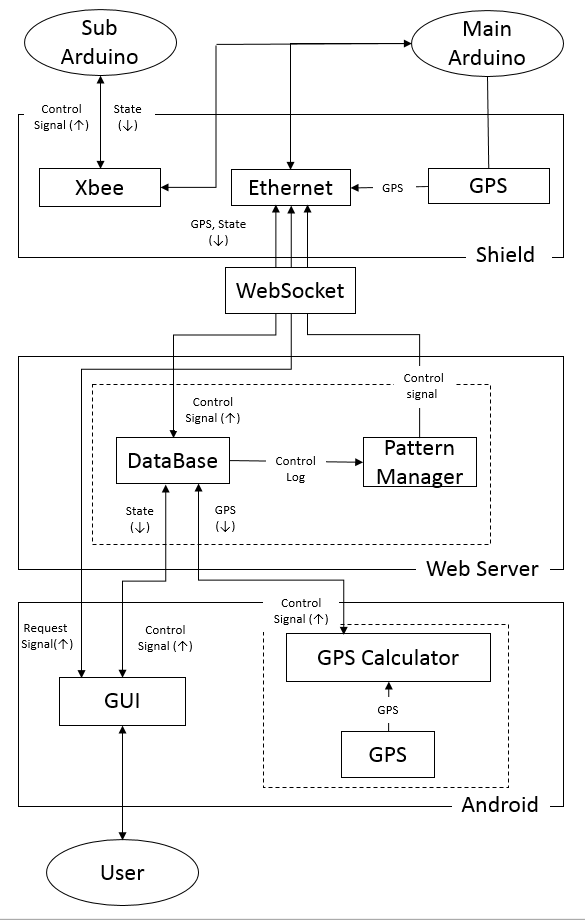
사용자가 외출한 후 집에 돌아올 때 에어컨, 보일러, 컴퓨터(부팅이 오래 걸리는) 등 사용자가 집에 도착하기 전에 미리 틀어놓으면 좋은 전자제품들은 사용자의 GPS의 위도 경도와 중앙 아두이노 의 GPS의 위도 경도의 거리를 계산하여 적정 값 이하일 때 에어컨을 틀도록 신호를 보낸다.

외출 시에 계속해서 서버로 GPS의 위도 경도를 보내게 되면 와이파이 환경이 아닌 이상 데이터 통신비가 많이 들게 되므로 GPS계산은 안드로이드 앱에서 한다. 외출 모드로 전환하면 서버의 데이터베이스에서 중앙 아두이노의 GPS값을 한 번 읽어온 후에 안드로이드 앱에서 현재 GPS위치와 중앙 아두이노의 GPS값을 비교하여 적정 값 이하일 때 서버로 명령을 보내도록 한다.

4-2) 행동 패턴 분석

데이터 베이스에 저장되어 있는 사용자가 보내온 명령들의 내용, 시간, 빈도 등을 바탕으로 행동 패턴을 분석하여 서버에서 적절한 시간에 적절한 명령을 내려 스스로 사물들을 제어할 수 있도록 한다.

* 1. **소프트웨어 Top-Level 구조**



1. **컴포넌트(모듈)별 기능 정의**

**Xbee (Shield)**

Sub Arduino(이하 서브)와 Main Arduino(이하 메인)의 보드에 탑재되어 있는 아두이노 간의 통신을 위한 쉴드이다. 서브에서는 이를 통해 메인으로 현재 제어중인 제품의 상태를 보낸다. 또한 메인으로부터 전달된 제어 명령에 따라 제품들을 제어하기 위한 신호를 서브로 보낸다.

**GPS (Shield)**

메인에 탑재되어 있는 쉴드이며, 메인과 User의 거리를 계산하기 위하여, 메인의 현재 위치를 얻기 위한 쉴드이다.

**Ethernet (Shield)**

서브의 Xbee로부터 메인의 Xbee로 전달된 데이터를 웹서버로 보내는, 아두이노와 서버 통신을 위한 쉴드이다. 메인에서는 이를 통해 서브로부터 온 데이터와 GPS 쉴드로부터 받은 메인의 현재 위치를 서버로 보낸다. 또한 서버로부터 전달된 제어 명령을 Xbee를 통해서 서브로 전달하는 역할을 한다. 전달된 데이터들은 서버의 Database에 저장이 된다.

**WebSocket (Shield&WebServer)**

중앙 아두이노와 웹 서버간의 통신 소켓이다. 웹 서버에서 중앙 아두이노로 제어 명령을 송신하고, 중앙 아두이노에서 웹서버로는 사물의 상태를 송신한다. 또한 웹서버에서 중앙 아두이노로 상태 요청 메시지도 송신한다.

**DataBase (WebServer)**

User와 Arduino 간의 통신 매체가 된다. 이를 통해 User는 제어 명령을 아두이노로 전달할 수 있으며, 아두이노의 상태를 원거리에서 확인할 수 있다. 또한 User의 제어 명령과 시간을 지속적으로 저장하여, User의 패턴을 분석할 수 있게 한다.

**Pattern Manager (WebServer)**

DB에 저장된 User의 명령과 시간, 위치를 이용하여, 사용자의 명령 패턴을 특정한 알고리즘을 이용하여 분석한다. 그리고 이를 통해 사용자 맞춤 서비스를 제공할 수 있다.

**GUI (Android)**

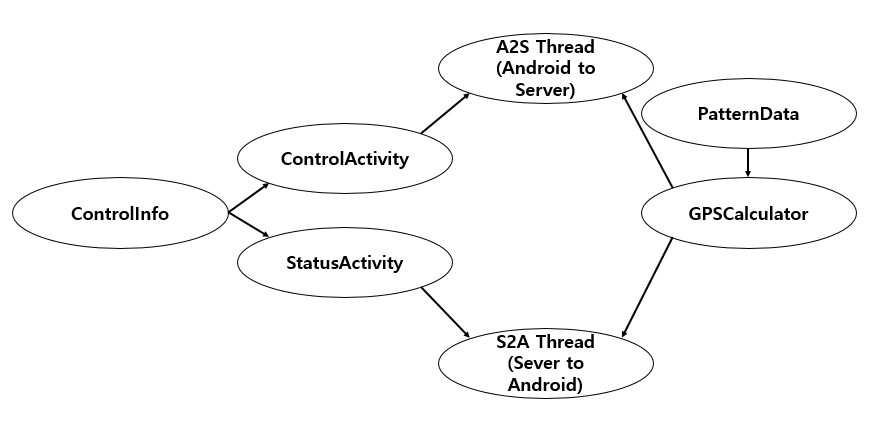
User에게 서버로부터 받은 아두이노의 상태를 보여주고, 단순한 인터페이스를 통해 아두이노가 설치된 제품을 제어할 수 있도록 한다.

**GPS Calculator (Android)**

서버로부터 전송된 메인의 위치정보를 SQLite에 저장을 하고, 이것과 현재 User의 위치를 주기적으로 계산을 한다. 그리고 메인으로부터 일정거리 이내에 접근하면 사용자가 설정한 제어를 서버로 자동으로 전송한다.

1 안드로이드(UI + GPS)설계

1.1 안드로이드 구성도



1.2 설계

1.2.1 안드로이드 data 설명

ControlActivity

|  |  |
| --- | --- |
| a2s | 서버로 제어 명령을 보내기 위하여 네트워크 작업을 하는 A2S\_Thread 클래스 객체 변수 |
| handler | 제어 명령 후 해당 사물의 상태를 보여주기 위해 UI작업을 하는 핸들러 |
| info | 제어하고 있는 사물의 정보를 담고 있는 ControlInfo 클래스 객체 변수 |

A2S\_Thread (Android to Server)

|  |  |
| --- | --- |
| userPosLat, userPosLng | 제어하고 있는 사용자의 위치(위도, 경도)를 담고 있는 Double형 타입의 변수 |
| thread | 네트워크 작업을 하는 HttpThread |
| siteURL | 명령을 제어하는 FLASK 서버의 라우트 주소를 담고 있는 URL 변수 |
| cname, corder | 제어할 기기와 제어 값이 저장되어 있는 String타입의 변수 |

S2A\_Thread (Server to Android)

|  |  |
| --- | --- |
| info | 제어하고 있는 사물의 정보를 담고 있는 ControlInfo 클래스 객체 변수 |
| xpp | 서버요청 결과로 얻을 xml 데이터를 파싱할 XMLPullParser 변수 |
| thread | 네트워크 작업을 할 HttpThread |
| siteURL | 사물의 상태를 받아오는 FLASK 서버의 라우트 주소를 담고 있는 URL 변수 |

PatterData

|  |  |
| --- | --- |
| PatternLat, Lng | 패턴으로 얻은 사용자의 위치를 담고 있는 변수 |
| info | 패턴 제어에 대한 사물의 정보를 담고 있는 변수 |

Calculator

|  |  |
| --- | --- |
| pdata | 패턴에 대한 정보를 담고 있는 변수 |
| userPosLat, userPosLng | 제어하고 있는 사용자의 위치(위도, 경도)를 담고 있는 Double형 타입의 변수 |
| a2s | 서버로 제어 명령을 보내기 위하여 네트워크 작업을 하는 A2S\_Thread 클래스 객체 변수 |

1.2.2 안드로이드 함수 설명

ControlActivity

|  |
| --- |
| void controlBtnClick() |
| 각 제어 버튼에 맞는 제어 기기의 명과 제어 값을 a2s 객체에 setter함수를 이용하여 세팅을 하고 thread run을 호출한다. |
| void initApp() |
| 제어 후, 상태를 보여주기 위한 핸들러의 정의와 xml에 정의된 뷰들을 객체화하기 위한 초기화 를 하는 함수이다. |

A2S\_Thread (Android to Server)

|  |
| --- |
| void setControl(String \_cname, String \_corder) |
| 사용자가 제어하기 위한 |
| void setUserPos() |
| 제어 후, 상태를 보여주기 위한 핸들러의 정의와 xml에 정의된 뷰들을 객체화하기 위한 초기화 를 하는 함수이다. |
| void transmitData() |
| Thread를 이용해서 네트워크 작업을 run하는 함수. 서버에 명령데이터를 url open을 이용해서 get방식으로 전달한다. |

S2A\_Thread (Server to Android)

|  |
| --- |
| String getData() |
| HttpThread를 이용해서 상태를 얻도록 route된 FLASK 서버에 사물 정보를 get방식으로 전달을 하고, 라우트에서 사물에 대한 정보를 XML형태로 반환하면 반환된 데이터를 가져온다. |
| void parseInfo() |
| getData() 함수를 통해 반환된 XML형태의 데이터를 파싱하기 위한 함수로, 파싱된 데이터를 ControlInfo 클래스의 객체에 setter함수를 통하여 저장을 한다. |

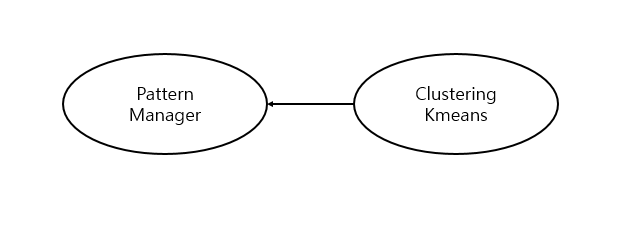
GPS Calculator

|  |
| --- |
| String getPattern() |
| HttpThread를 이용해서 서버에 저장되어 있는 패턴을 얻도록 route된 FLASK 서버에 사용자 안드로이드 디바이스 아이디를 get방식으로 전달을 하고, 라우트에서는 패턴에 대한 정보를 XML형태로 반환하면 반환된 데이터를 가져온다. |
| void ParsePattern() |
| getPattern() 함수를 통해 반환된 XML형태의 데이터를 파싱하기 위한 함수로, 파싱된 데이터를 PatternData클래스의 객체에 setter함수를 통하여 저장을 한다. |

※ PatternData와 ControlInfo 클래스의 함수는 데이터를 얻고 저장하기 위한 Setter함수와 Getter 함수로 구성되어 있다.

2 패턴 제어 설계

2.1 패턴 제어 구성도



2.2 설계

2.2.1 패턴 제어 data 설명

PatternManager

|  |  |
| --- | --- |
| Alg =ClusteringKmeans(…) | 클러스터링 작업을 할 ClusteringKmeans 클래스의 객체 변수이다. |
| Data = np.array(…) | 데이터베이스로부터 가져온 제어 기록이 저장되어 있는 numpy배열 변수이다. |
| transData = np.array(…) | 데이터베이스로부터 가져온 튜플데이터가 클러스터링에 필요한 2차원 벡터로 변환된 배열 변수이다. |
| Prior = tuple(…) | 데이터베이스로부터 가져온 사용자의 기존 정보가 저장되어 있는 튜플 변수이다. |

ClusteringKmeans

|  |  |
| --- | --- |
| Optimize\_k | elbowMethod로부터 얻은 optimize된 클러스터의 개수를 담고 있는 정수형 변수이다. |
| Centroid = np.array(…) | 각 클러스터들의 centroid(2차원 벡터)가 저장되어 있는 배열 변수이다. |
| Idx = np.array(…) | Data변수의 값들이 어디 클러스터에 속하는지 저장되어 있는 배열 변수이다. |
| Data = np.array(…) | 클러스터링의 입력데이터가 되는 2차원 벡터 데이터를 담고 있는 배열 변수이다. |

2.2.2 패턴 제어 함수 설명

PatternManager

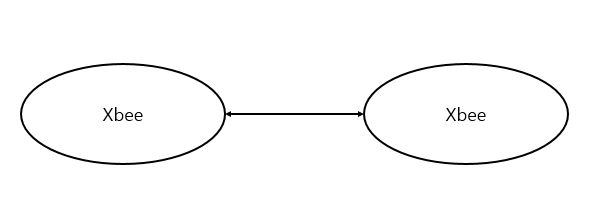
|  |
| --- |
| def getData(self, aid) |
| 사용자의 안드로이드 디바이스 아이디를 입력으로 받고, 데이터베이스로부터 해당 사용자에 대한 제어기록들을 모두 select하여 가져오고 이 데이터들을 data라는 numpy 배열 변수에 저장을 한다. |
| def transVector(self, data) |
| data 변수에 저장되어 있는, Select 쿼리를 통해 얻은 튜플 데이터(시간, 온도, 위치)들을 2차원 벡터들(온도에 따른 제어 횟수, 위치에 따른 제어 횟수)로 변환하여 새로운 배열 변수에 저장을 한다. |
| def clusteringRun(self) |
| 클러스터링 작업을 할 ClusteringKmeans의 run함수를 호출하여 클러스터링을 하는 함수이다. |

ClusteringKmeans

|  |
| --- |
| def elbowMethod(self) |
| k값을 최소의 클러스터 값부터 최대의 클러스터 값까지 증가시키면서 클러스터 k값에 해당하는 SSE를 계산한다. |
| def CalculateSSE(self, centroid, idx, cnum): |
| 클러스터의 개수와 클러스터 centroid값 그리고 데이터가 속한 클러스터 인덱스를 입력으로 받아 모든 클러스터에 대한 SSE(Sum of Squared Error)를 계산하여 반환한다. |
| def Run(self) |
| ElbowMethod를 호출하고, 여기서 얻은 클러스터의 개수 k를 이용하여 클러스터링을 하는 함수이다. |

3 Xbee 설계

3.1 Xbee 구성도



3.2 설계

3.2.1 Xbee data 설명

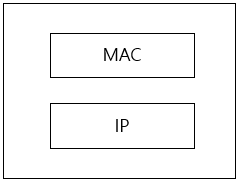
|  |  |
| --- | --- |
| controlValue | (서브 아두이노) 제어를 위한 아두이노의 보드에 맞는 입출력 포트 번호를 담고 있는 변수이다. |
| strData | 사물에 대한 명령, 상태를 저장하고 다른 xbee로 보내는 String 타입의 변수이다. 이 변수에는 cname과 corder가 조합되어 저장된다. |
| cname | 데이터베이스로부터 얻은 제어할 사물의 이름이 저장되어 있는 String 타입의 변수이다. |
| corder | 데이터베이스로부터 얻은 제어할 사물의 제어 값이 저장되어 있는 String 타입의 변수이다. |

3.2.2 Xbee 함수 설명

|  |
| --- |
| void setup() |
| Xbee 통신과 제어를 위해 Serial port를 설정하고, digital write 포트를 설정하는 함수이다. |
| void loop() |
| 데이터를 보낼 때 – cname과 corder를 조합하여 하나의 String 타입의 변수로 만들고, Serial3 포트에 print 하여 다른 xbee로 보낸다.  데이터를 받을 때 – Serial3.available()함수를 지속적으로 호출하여 데이터가 해당 xbee로 데이터가 들어오는지 확인한다. 데이터가 들어오면(하나의 String타입의 변수) cname과 corder로 sclice를 하고, 구분하여 각각의 변수로 할당을 한다. |

4 Ethernet 설계

4.1 Ethernet 구성도



4.2 설계

4.2.1 멤버 data 설명

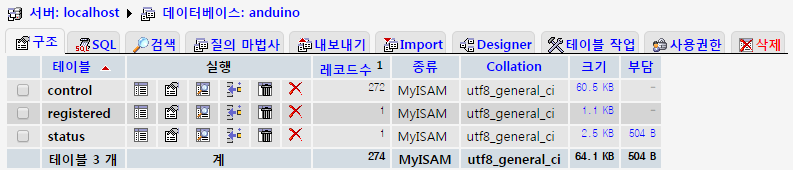
|  |  |
| --- | --- |
| byte mac[] | 이더넷 쉴드의 mac주소. 디폴트 값으로 설정한다. |
| IPAddress ip | 아두이노에 할당할 ip주소 |

4.2.2 멤버 함수 설명

|  |
| --- |
| int begin(uint8\_t \*mac\_address) |
| 이더넷 쉴드를 초기화하여 이더넷 연결을 시작한다. MAC주소를 통해 ip등 나머지 환경을 DHCP를 통해 설정한다. DHCP를 통한 환경 설정이 실패하면 0을, 성공하면 1을 반환한다. |
| void begin(uint8\_t \*mac\_address, IPAddress local\_ip) |
| MAC주소뿐만 아니라 ip를 할당하여 이더넷 연결을 시작한다. |

5 DataBase(server-side) 설계

5.1 DataBase 구성도



5.2 설계

5.2.1 각 Column 설명

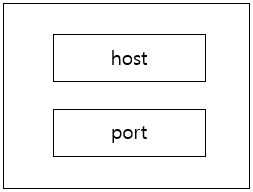
|  |  |  |
| --- | --- | --- |
| Control | num | Primary key, Auto increase |
| cname | 제어할 사물의 이름 (aircon/lamp/window) |
| corder | 제어 명령 (0/1) |
| date | 제어한 날짜/시간 |
| lat | 제어한 장소 (위도) |
| lng | 제어한 장소 (경도) |
| aid | 제어한 안드로이드 기기의 아이디 |
| status | num | Primary key, Auto increase |
| sname | 해당 상태를 갖는 사물 이름 |
| sstatus | 사물의 상태 |
| date | 해당 상태로 바뀐 날짜/시간 |
| registered | aid | 등록된 안드로이드 기기의 아이디 |

5.2.2 접근 함수 설명

|  |
| --- |
| def connect\_db(host, user, passwd, db): |
| 매개 변수로 받은 host, user, passwd, db를 바탕으로 MySQL에 연결하여 연결된 데이터베이스를 반환한다. |
| def set\_control(cname, corder, lat, lng, aid): |
| 데이터베이스의 control 테이블에 cname, corder, lat, lng, aid를 동일한 이름의 column에 맞추어 투플을 삽입한다. 이 때 date column에는 현재 시간을 넣는다. 결과로 “Control Saved”라는 문자열을 반환한다. |
| def get\_status(sname): |
| 데이터베이스의 status테이블에서 sname column이 sname인 투플을 검색한다. 검색 결과는 들어간 순서로 정렬되도록 한다. 가장 처음 결과, 즉 가장 최근의 상태를 얻는다. 얻은 투플의 sname, sstatus, date를 xml형태로 반환한다. |

6 WebSocket(server-side) 설계

6.1 WebSocket 구성도



6.2 설계

6.2.1 웹소켓 서버 data 설명

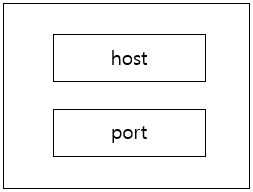
|  |  |
| --- | --- |
| websock | WebSocket타입의 변수. 전역변수로 서버와 중앙 아두이노가 연결된 웹소켓을 저장한다. |
| host | WebSocket서버의 host |
| port | WebSocket서버의 port |

6.2.2 웹소켓 서버 함수 설명

|  |
| --- |
| def wsgi\_app(environ, start\_response): |
| 클라이언트가 서버에 요청을 보내면 호출되는 함수이다. 매개변수로 들어오는 딕셔너리로, 여러 값들을 갖고 있다. 그 중 environ["PATH\_INFO"]을 통해 어떤 url을 통한 요청이 들어왔는지 알 수 있다. 따라서 정규식을 사용하여 url을 분리하여 그에 해당하는 동작을 수행한다.  environ["wsgi.websocket"]은 연결된 웹소켓이므로 이를 전역변수 websock에 저장한다. 그 후 그 웹소켓에 해당하는 이벤트들을 처리하기 위해 handle\_websocket을 호출한다. |
| def handle\_websocket(ws): |
| 매개변수로 들어온 웹소켓으로부터 들어오는 메시지를 받는다. 메시지 프로토콜에 따라 해당하는 동작을 수행한다. |

7 WebSocket(client-side) 설계

7.1 WebSocket 구성도



7.2 설계

7.2.1 웹소켓 클라이언트 data 설명

|  |  |
| --- | --- |
| webSocket | WebSocketsClient타입의 변수. 서버와 중앙 아두이노가 연결된 웹소켓을 저장한다. |
| host | WebSocket을 연결할 host (WebSocket서버의 ip) |
| port | WebSocket을 연결할 port |

7.2.2 웹소켓 클라이언트 함수 설명

|  |
| --- |
| void setup() |
| 아두이노 설정함수이다. 웹소켓 연결을 시작하고, 웹소켓 이벤트를 처리할 함수를 등록한다. |
| void loop() |
| 아두이노가 전원이 켜져있는 동안 반복하는 함수이다. 웹소켓을 유지한다. |
| void webSocketEvent(WStype\_t type, uint8\_t \* message, size\_t length) |
| 웹소켓 이벤트를 처리할 함수이다. type에 따라 어떤 이벤트인지 알 수 있고, message는 웹소켓을 통해 들어온 데이터이고 length는 데이터의 길이이다.  WStype\_CONNECTED는 웹소켓이 연결되었을 때, WStype\_DISCONNECTED는 웹소켓 연결이 끊겼을 때, WStype\_TEXT는 웹소켓을 통해 데이터가 들어왔을 때의 type이다  웹소켓에서 데이터가 들어왔을 때 들어온 메시지를 서브 아두이노로 송신한다. |