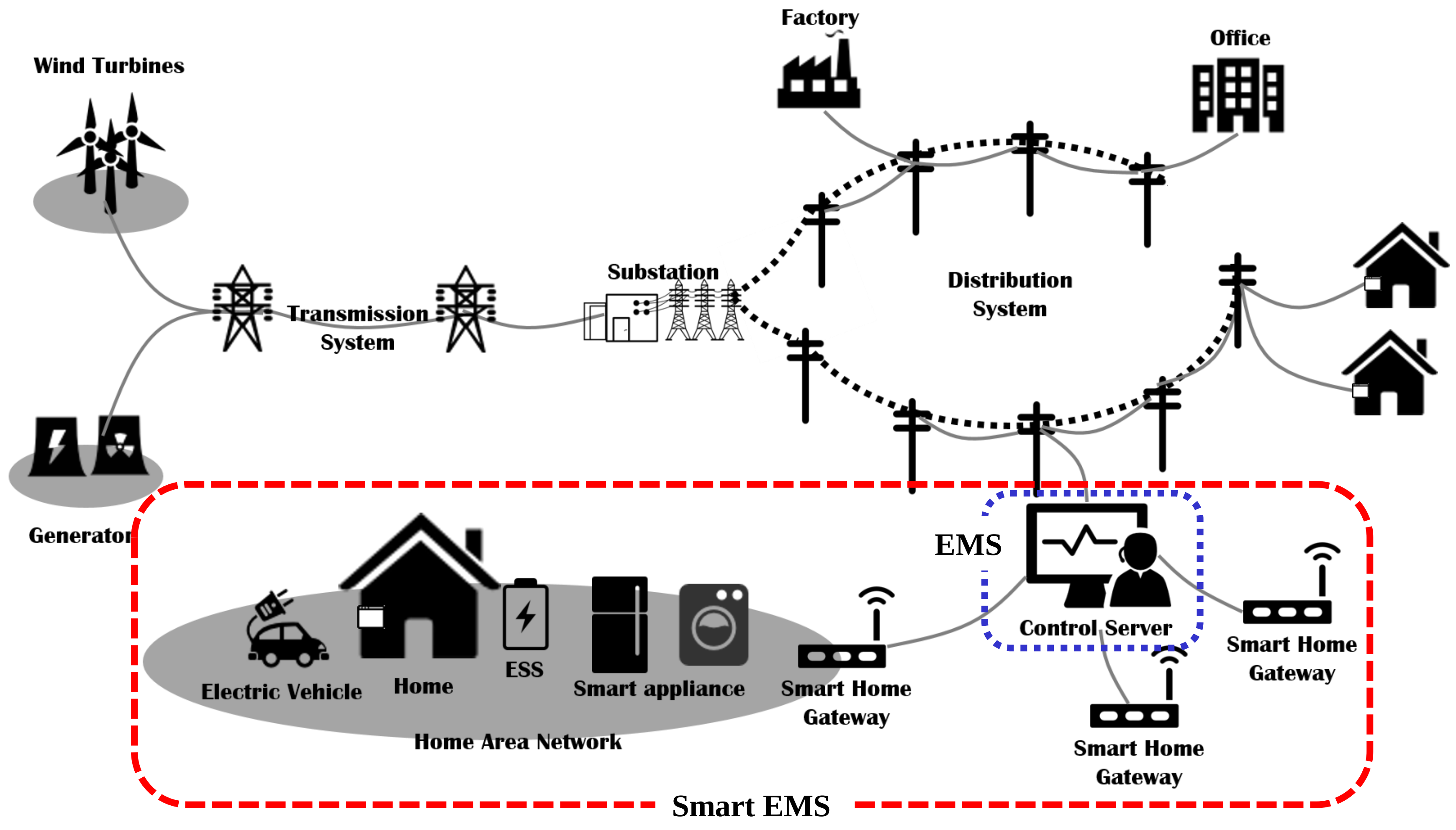
Energy Management System

MIR Lab
http://mir.hanyang.ac.kr

Lecture Index

3. EMS Overview

3-1. System Architecture



3-1. 스마트 에너지 홈 환경

스마트 에너지홈 환경

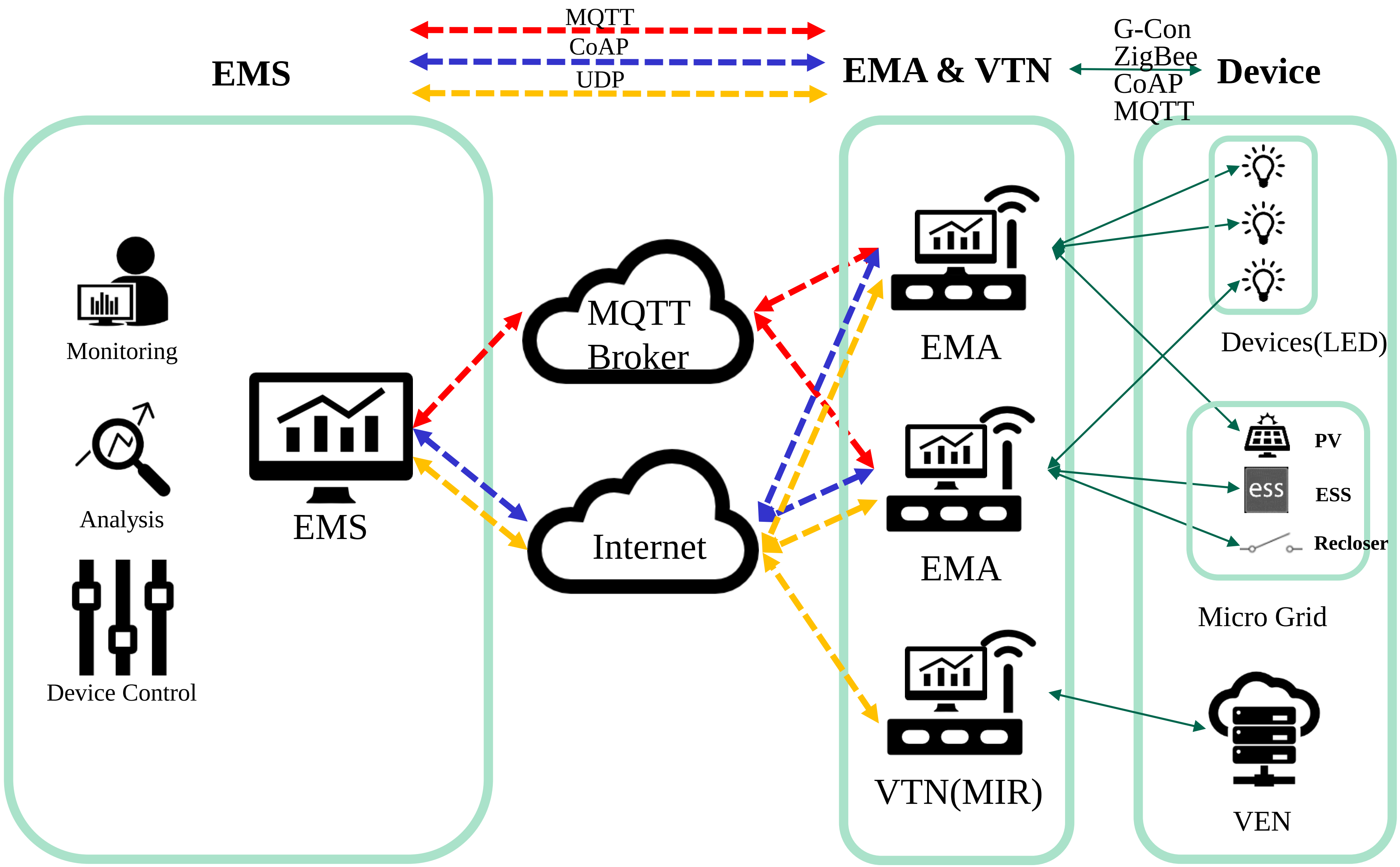
- 에너지 그리드 서비스를 제공하기 위한 스마트 에너지 홈 환경은 EMA가 관리하는 디바이스의 그룹과 상위의 서비스를 제공해 주는 서비스 제공자로 구분된다.
- EMA 하위에는 G-con, MQTT(MQ Telemetry Transport), CoAP(Constained Application Protocol)등과 같이 여러 가지 프로토콜로 구성된 디바이스의 그룹이 있고 이는 각각 아두이노와 라즈베리파이 등으로 구현되어 있다.
- 상위 서비스 제공자는 VTN, EMS, Utility등으로 구성되어 있으며 이들은 각각 에너지 소비에 대한 전략을 가지고 EMA에게 서비스를 제공해주는 역할을 한다.

3-1. 스마트 에너지 홈

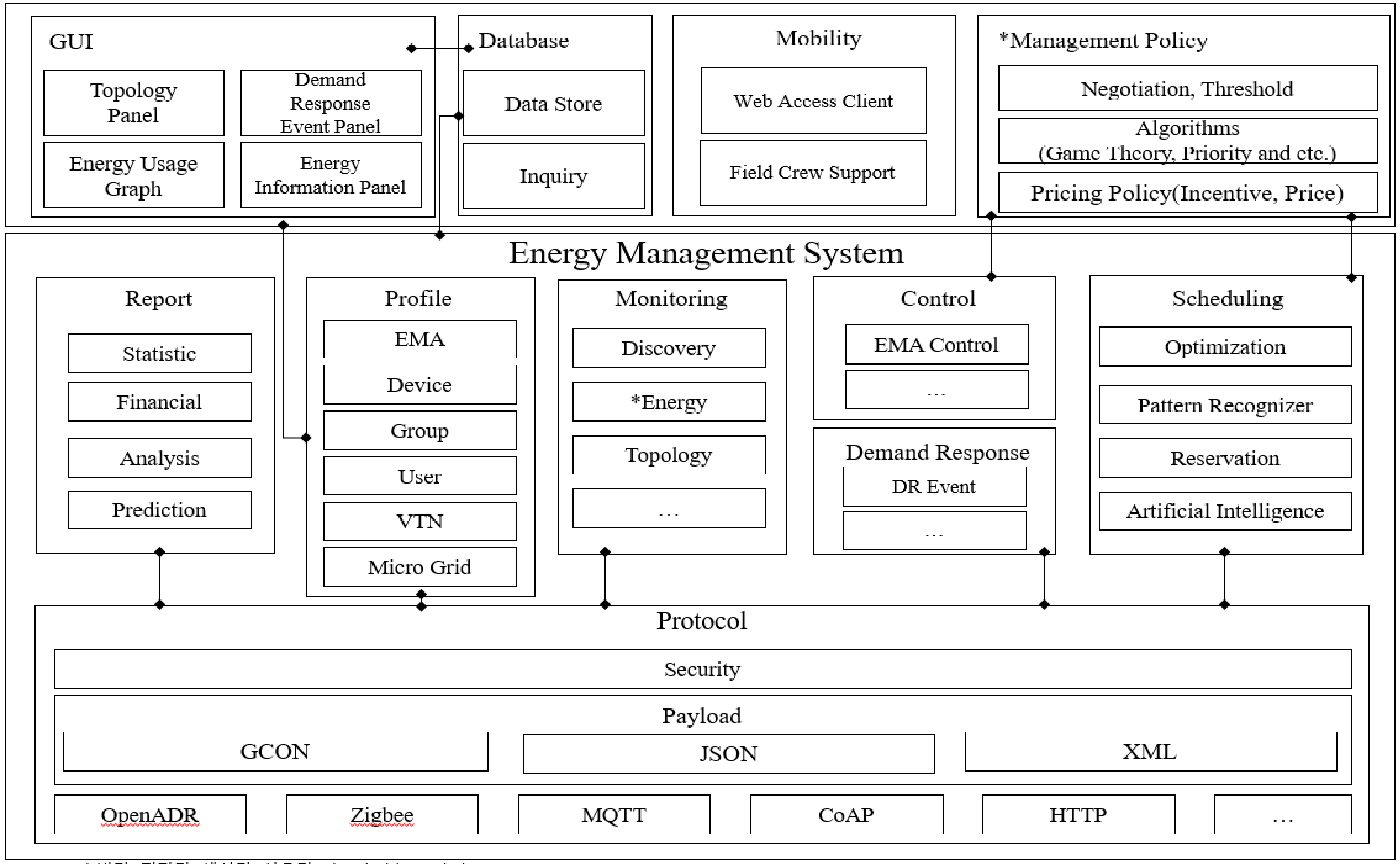
스마트 에너지 홈

– 스마트 에너지 홈란 일반적인 스마트 홈 환경에서 디바이스의 On/Off 제어 및 상태 보고 등과 같은 기능 외에도 자동적으로 지능화 된 수요반응 기능을 제공할 수 있는 환경을 의미한다.
– 앞서 이야기한 OpenADR 프로토콜을 통하여 전력 사업자와 사용자 간의 DR서비스 환경을 만들어서 지능적인 에너지 소모를 할 수 있도록 해야 한다.
– EMS는 EMA로부터 주기적으로 에너지 사용량을 보고 받아 모니터링 하게 되고, 디바이스를 직접 컨트롤 할 수 있도록 EMA에 명령을 전달 할 수 있도록 구성되어 있다.
– 다양한 모듈의 디바이스와 WiFi, Zigbee, IEEE 802.15.4통신을 위하여 해당 라이브러리에서 제공해주는 규격을 사용한다.

3-1. System Architecture with Protocol



3-2. EMS Overview



*Energy: 소비량, 저장량, 생산량, 사용량, Threshold, Load Flow
*Management Policy: 에너지 사용량 관리 정책 및 사용량에 따른 가격 정책

3-2. EMS Overview

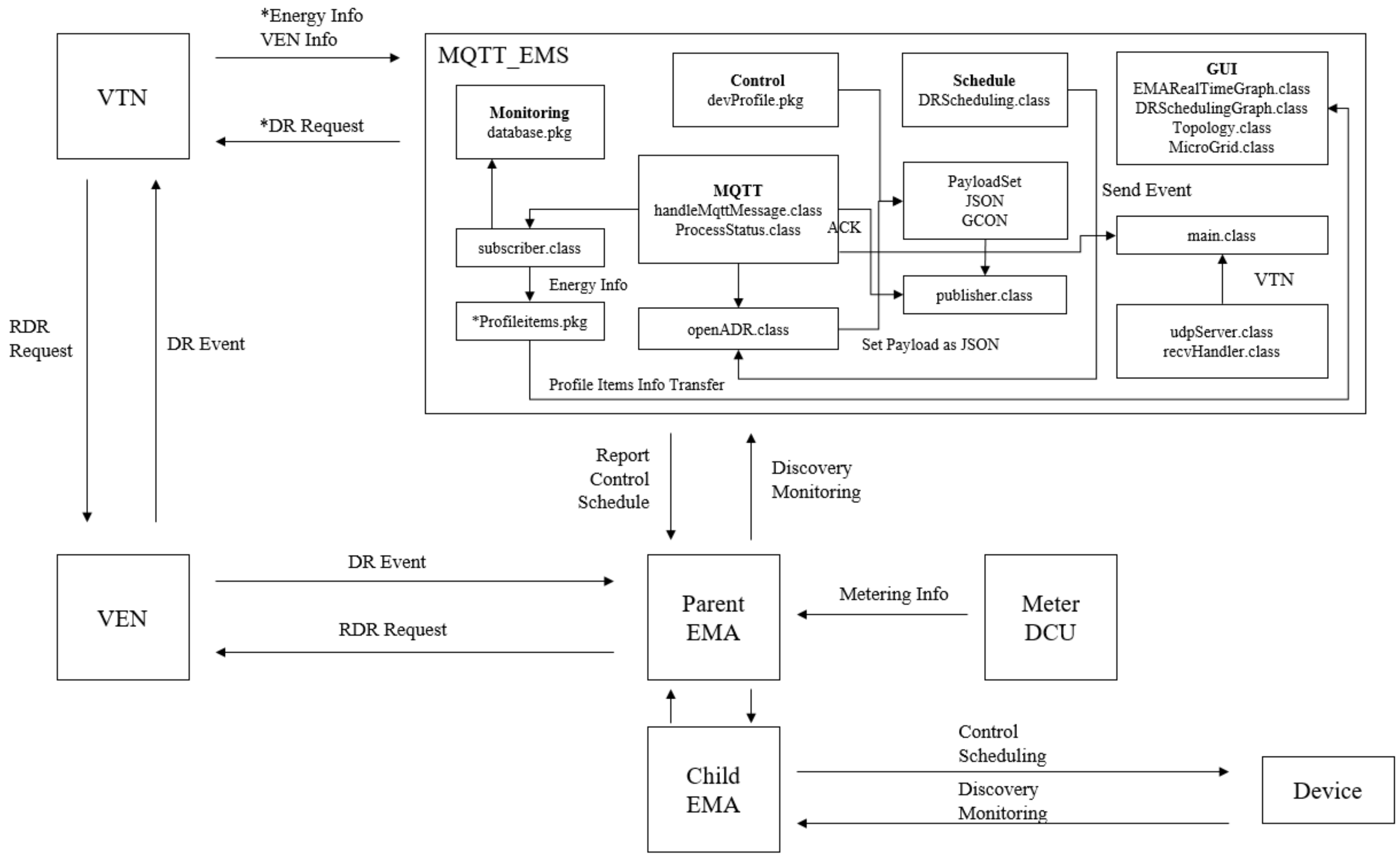
Application

•*GUI Subsystem* – To show the status and information.
•*Mobility Subsystem* – To be available for approaching anywhere.
•*Management Policy Subsystem* – Manage Policy

Energy Management System defines several primary services:

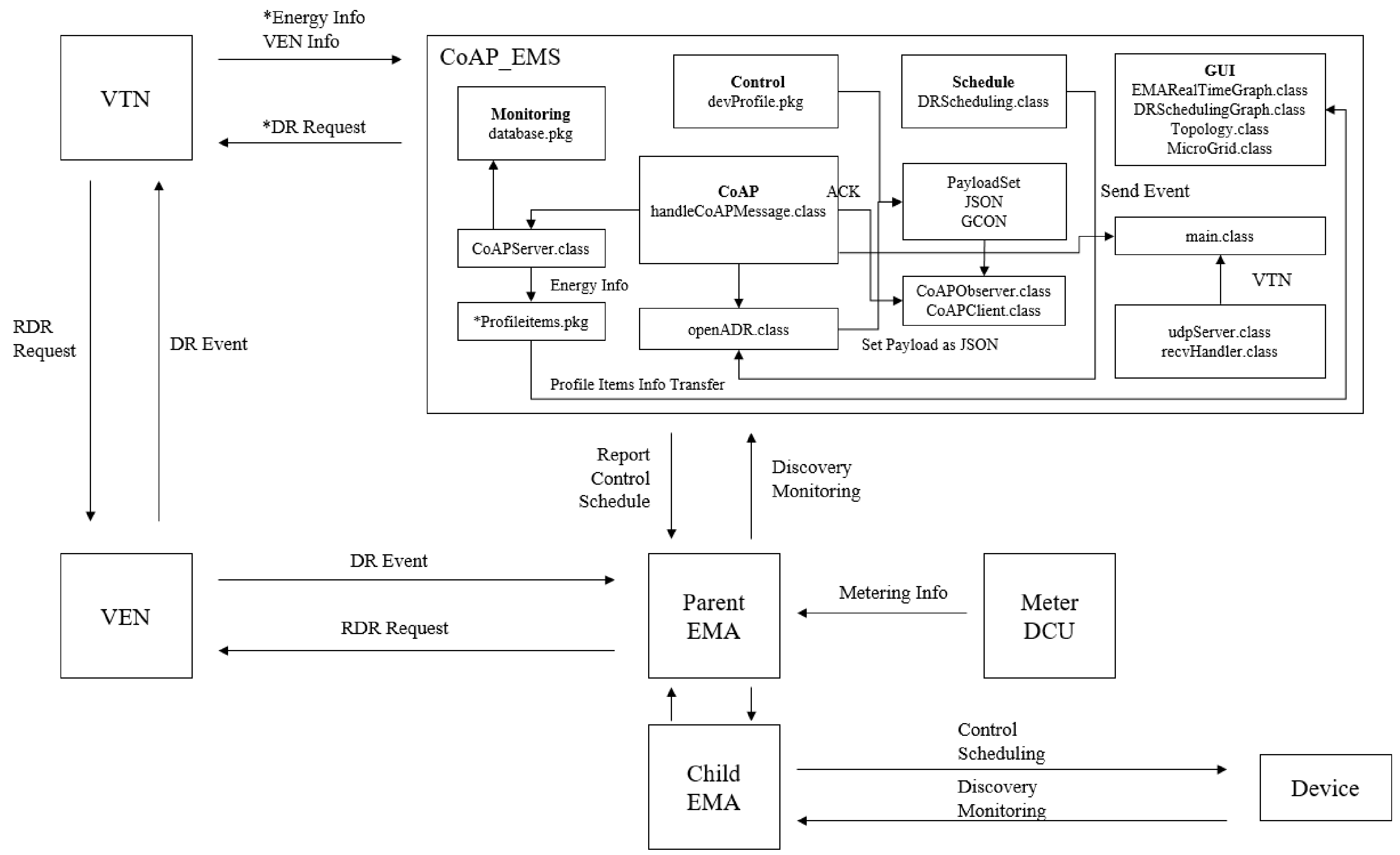
•*Scheduling Service* – Controls and manages EMA, Devices for optimization.
•*Report Subsystem* – Manage energy data to give processed report to energy customer.
•*Monitoring Subsystem* – Listen for EMA,VTN messages for manage and to make a topology and path
•*Profile Subsystem* - Manages inventory of sort of devices information and group
•*Control Subsystem* – To control child systems(EMA, OpenFMB and etc.) manually based on price policy(Incentive, Priced)

3-2. EMS Program Architecture



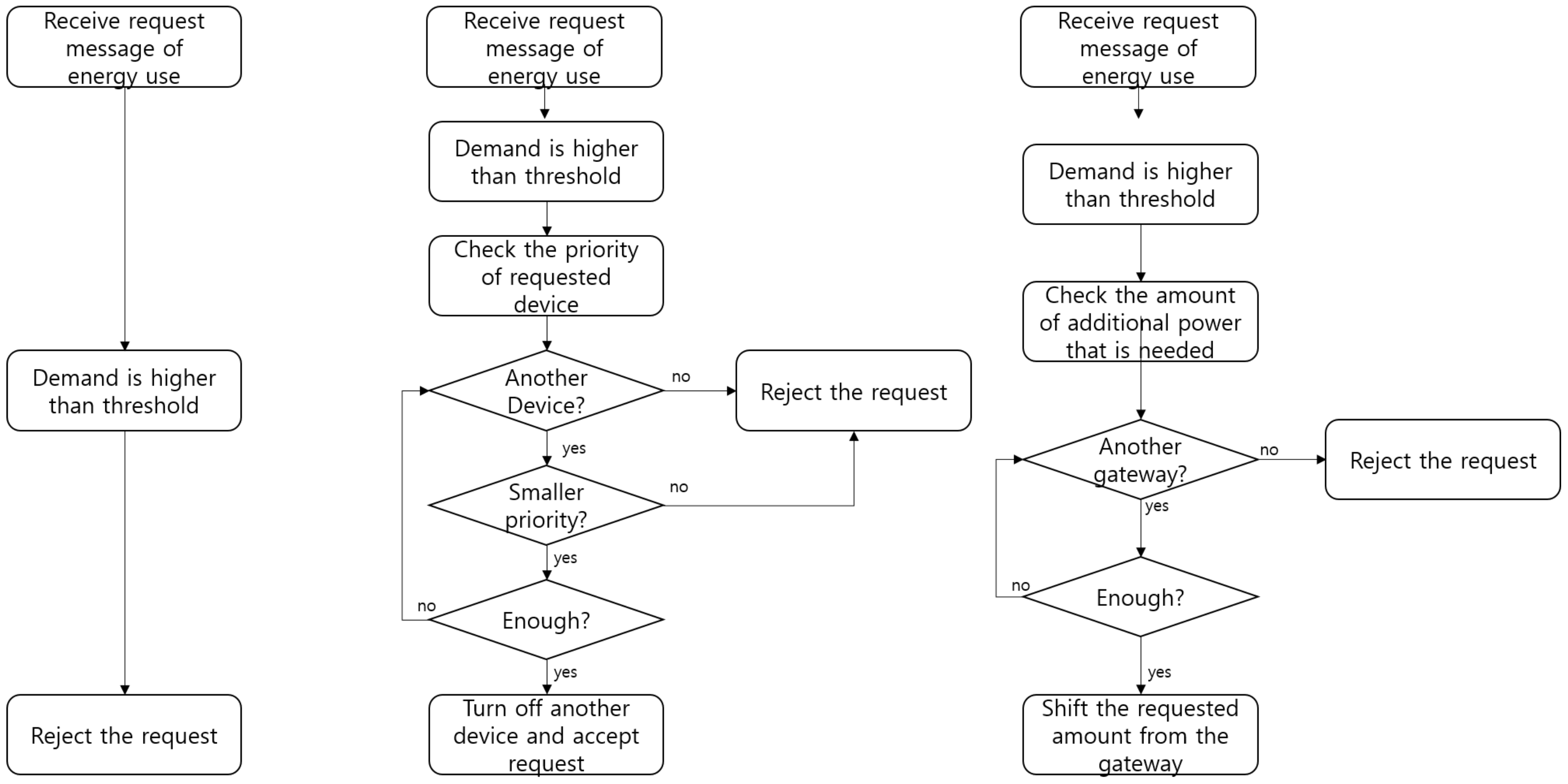
*DR Request: EMS가 VTN의 부하감축을 직접적으로 할 수는 없다. VTN Energy 정보를 바탕으로 최대 수요의 억제, 최대 부하의 이전, 가
*Energy: 소비량, 저장량, 생산량, 사용량, Threshold, Load Flow
*Profileitems: VTN, EMA, Device, Osc, EMB SmartMeter 정보

3-2. EMS Program Architecture



*DR Request: EMS가 VTN의 부하감축을 직접적으로 할 수는 없다. VTN Energy 정보를 바탕으로 최대 수요의 억제, 최대 부하의 이전,

3-3. EMS Optimization Overview


Receive request message of energy use
Demand is higher than threshold
Reject the request
Receive request message of energy use
Demand is higher than threshold
Check the priority of requested device
Another Device?
no
Reject the request
yes
Smaller priority?
no
yes
no
Enough?
yes
Turn off another device and accept request
Receive request message of energy use
Demand is higher than threshold
Check the amount of additional power that is needed
Another gateway?
no
Reject the request
yes
no
Enough?
yes
Shift the requested amount from the gateway

(a) Threshold (b) Threshold with priority (c) Negotiation

Appendix. EMA Overview

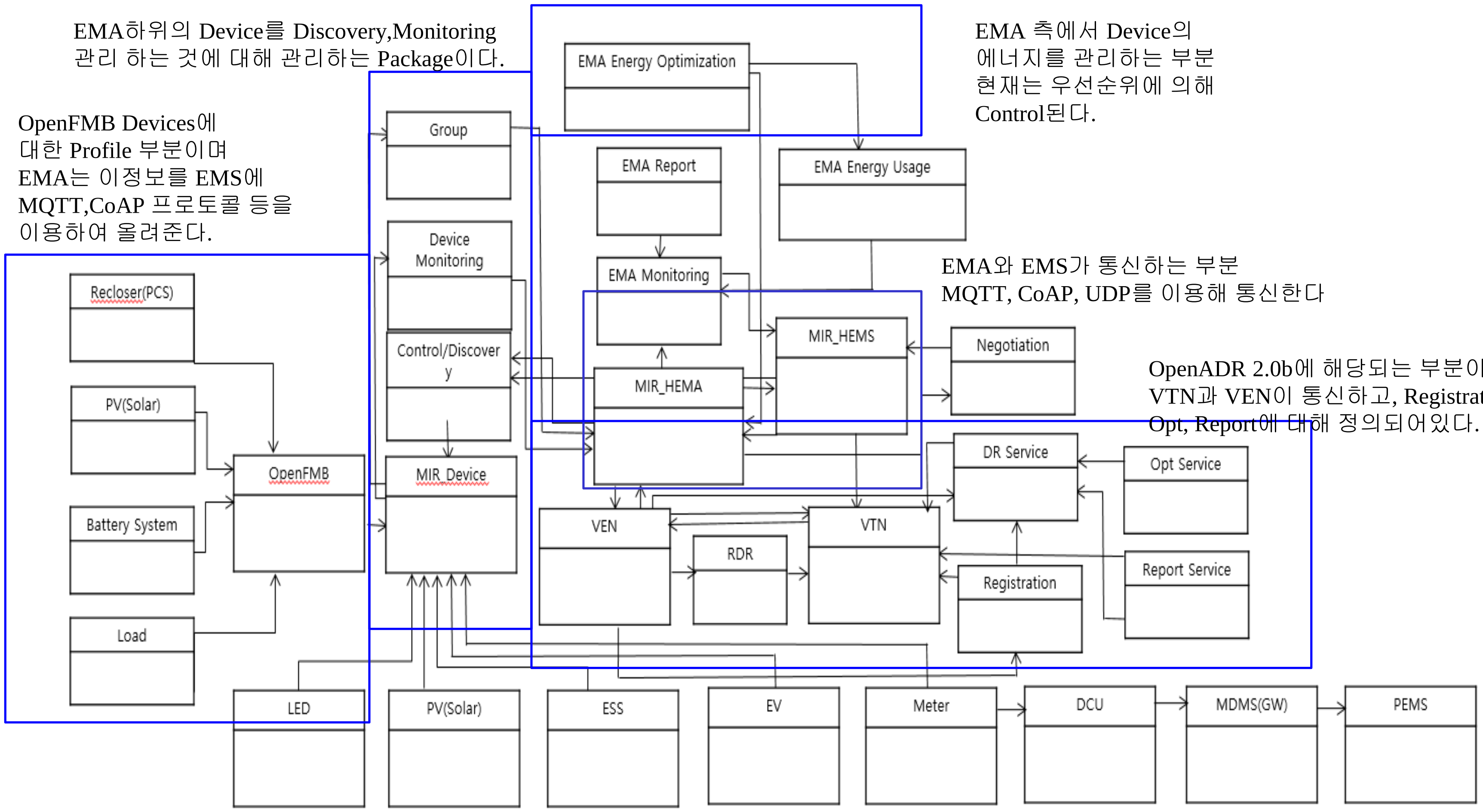
EMA하위의 Device를 Discovery,Monitoring
관리 하는 것에 대해 관리하는 Package이다.

EMA 측에서 Device의
에너지를 관리하는 부분
현재는 우선순위에 의해
Control된다.

OpenFMB Devices에
대한 Profile 부분이며
EMA는 이정보를 EMS에
MQTT,CoAP 프로토콜 등을
이용하여 올려준다.

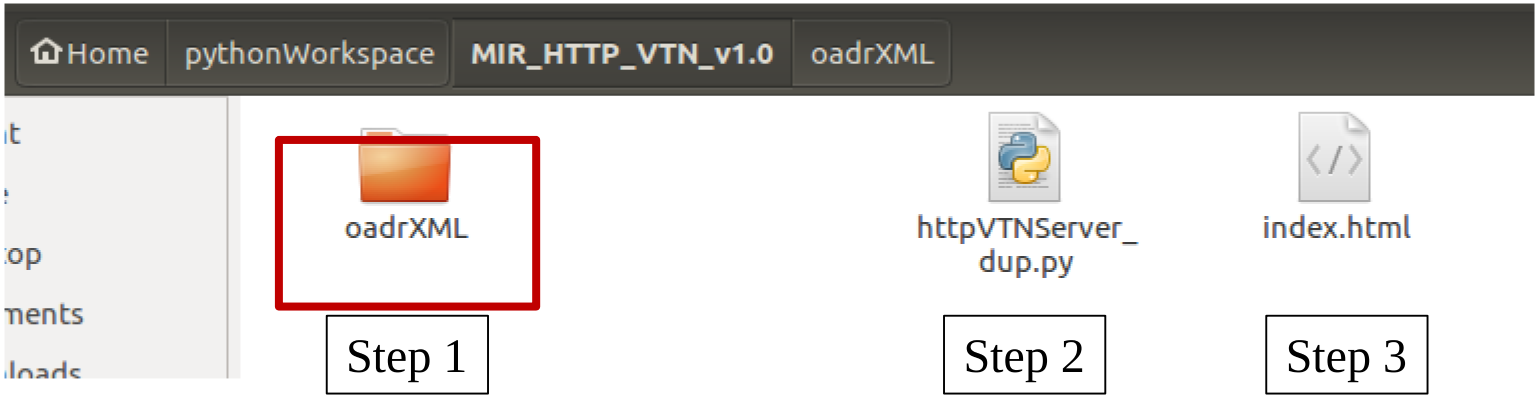
EMA와 EMS가 통신하는 부분
MQTT, CoAP, UDP를 이용해 통신한다

OpenADR 2.0b에 해당되는 부분이다.
VTN과 VEN이 통신하고, Registration, DR,
Opt, Report에 대해 정의되어있다.



Smart Meter -> DCU
-> MDMS(Future) -> EMS

MIR VTN – OpenAdr2.0b



Block - Set Payload

Step 1: xml format for file input and output, it is called when send response message to VEN
which means it sets payload values as OADR format

Block - Service

Step 2: VTN main class, call http Server , send response and Parse receive xml Message

and send EiEvent

Block - Web User Interface

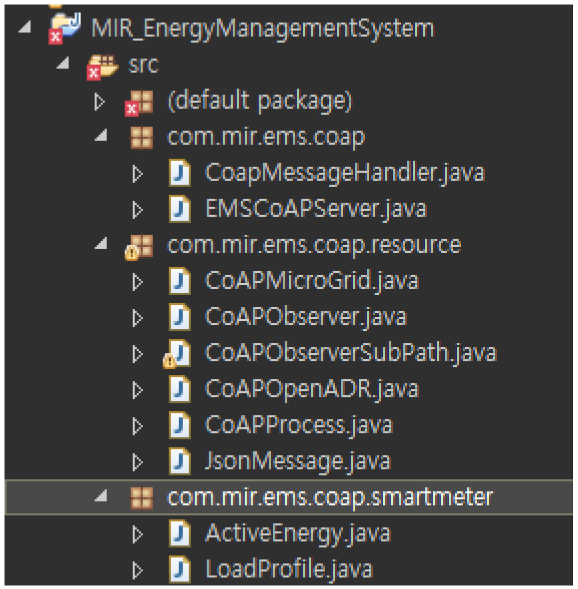
Step 3: Index VTN Page
set EiEvent options(e.g. set Target, Duration, Start Time and etc)

4. EMS : Package Explanation OpenADR

각ㅍ
무우

In MIR Lab, We are using californium CoAP library

CoAP Package



```
MIR_EnergyManagementSystem
  src
    (default package)
    com.mir.ems.coap
      CoapMessageHandler.java
      EMSCoAPServer.java
    com.mir.ems.coap.resource
      CoAPMicroGrid.java
      CoAPObserver.java
      CoAPObserverSubPath.java
      CoAPOpenADR.java
      CoAPProcess.java
      JsonMessage.java
    com.mir.ems.coap.smartmeter
      ActiveEnergy.java
      LoadProfile.java
```

com.mir.ems.coap

- EMSCoAPServer :
CoAP Server
- CoAPMessageHanlder :
Handling the message that receive from CoAP Client

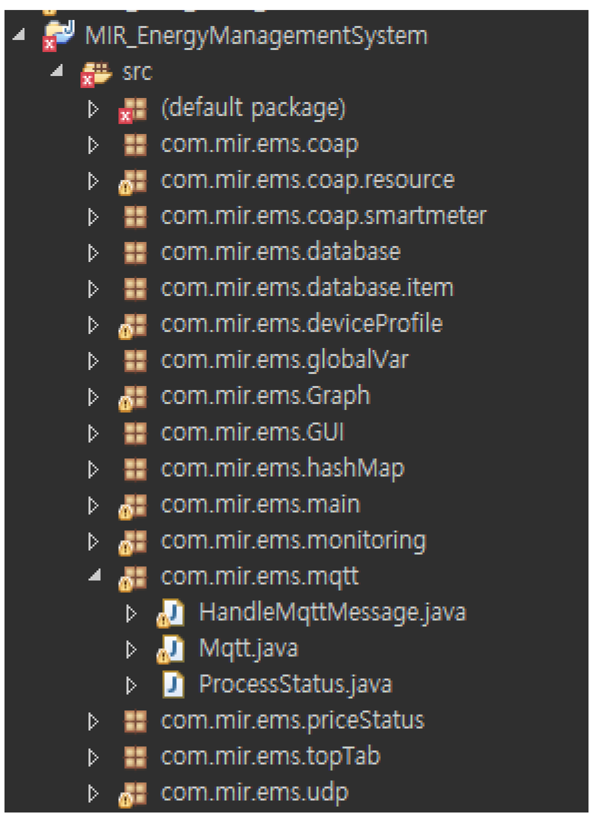
com.mir.ems.coap.resource

-CoAPMicroGrid:
Restful API (Only use PUT Method)

-CoAPObserver:
It is super class of CoAPObserverSubPath
-CoAPObserver:
Send Push Message when Event occur
-CoAPOpenADR:
Process of OpenADR2.0b(e.g. queryRegistration)
-JsonMessage:
Parsing JSON Type Message that receive from client

4. EMS : Package Explanation OpenADR

MQTT Package


```
◢ MIR_EnergyManagementSystem
  ◢ src
    ▷ (default package)
    ▷ com.mir.ems.coap
    ▷ com.mir.ems.coap.resource
    ▷ com.mir.ems.coap.smartmeter
    ▷ com.mir.ems.database
    ▷ com.mir.ems.database.item
    ▷ com.mir.ems.deviceProfile
    ▷ com.mir.ems.globalVar
    ▷ com.mir.ems.Graph
    ▷ com.mir.ems.GUI
    ▷ com.mir.ems.hashMap
    ▷ com.mir.ems.main
    ▷ com.mir.ems.monitoring
    ◢ com.mir.ems.mqtt
      ▷ HandleMqttMessage.java
      ▷ Mqtt.java
      ▷ ProcessStatus.java
    ▷ com.mir.ems.priceStatus
    ▷ com.mir.ems.topTab
    ▷ com.mir.ems.udp
```

In MIR Lab, We are using paho MQTT library

com.mir.ems.mqtt

-Mqtt:

Start Publish, Subscribe and MQTT Client

-HandleMqttMessage:

Handling the JSON and Text message and publish Message to MQTT Broker

-ProcessStatus:

Save the Process value (Poll or Event)

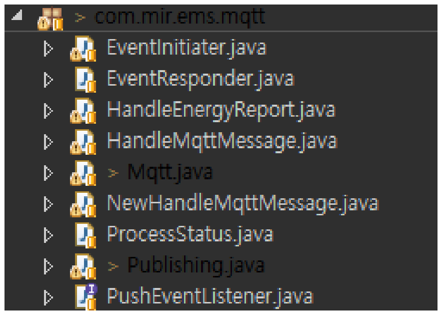
4. EMS : Package Explanation OpenADR

Protocol

Service

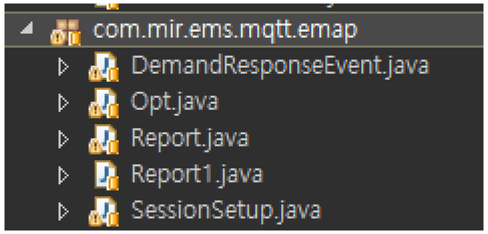
OpenADR Model Package

MQTT

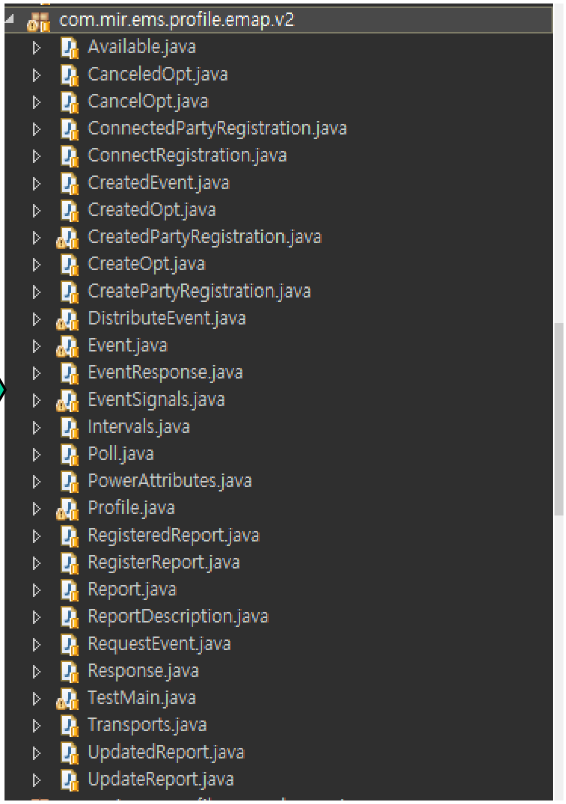


```
com.mir.ems.mqtt
    EventInitiater.java
    EventResponder.java
    HandleEnergyReport.java
    HandleMqttMessage.java
    Mqtt.java
    NewHandleMqttMessage.java
    ProcessStatus.java
    Publishing.java
    PushEventListener.java
```

프로토콜 서비스 참조

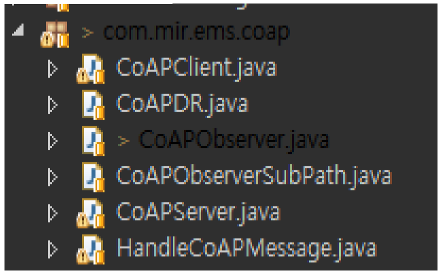


```
com.mir.ems.mqtt.emap
    DemandResponseEvent.java
    Opt.java
    Report.java
    Report1.java
    SessionSetup.java
```



```
com.mir.ems.profile.emap.v2
    Available.java
    CanceledOpt.java
    CancelOpt.java
    ConnectedPartyRegistration.java
    ConnectRegistration.java
    CreatedEvent.java
    CreatedOpt.java
    CreatedPartyRegistration.java
    CreateOpt.java
    CreatePartyRegistration.java
    DistributeEvent.java
    Event.java
    EventResponse.java
    EventSignals.java
    Intervals.java
    Poll.java
    PowerAttributes.java
    Profile.java
    RegisteredReport.java
    RegisterReport.java
    Report.java
    ReportDescription.java
    RequestEvent.java
    Response.java
    TestMain.java
    Transports.java
    UpdatedReport.java
    UpdateReport.java
```

COAP



```
com.mir.ems.coap
    CoAPClient.java
    CoAPDR.java
    CoAPObserver.java
    CoAPObserverSubPath.java
    CoAPServer.java
    HandleCoAPMessage.java
```

각 서비스에 맞는 모델링 참조

4. EMS : Package Explanation OpenADR

CoAP Server Class : 수/발신 메시지 서비스에 따른 분류

… [Line 952]

```
                        server.add(new Emap("EMAP"));

                        server.add(new OpenADR("OpenADR"));

                        // Observe
                        server.add(new CoAPObserver("OpenADR2.0b"));
                        server.add(new CoAPObserver("EMAP1.0b"));

public class Emap extends CoapResource {

            public Emap(String name) {
                        // TODO Auto-generated constructor stub
                        super(name);

                        add(new SystemID(global.SYSTEM_ID, name));

            }
}
….
                        // OpenADR
                        add(new SessionSetup("SessionSetup"));
                        add(new Report("Report"));
                        add(new Opt("Opt"));
                        add(new DemandResponseEvent("Event"));
                        add(new DemandResponseEvent("Poll"));
```

4. EMS : Package Explanation OpenADR

CoAP Observe Class[Push] 별도 Class : COAP Client Observe function for Push

```
public CoAPObserverSubPath(String name, String parentPath) {
        super(name);
        this.name = name;
        setObservable(true);                              // Observe 활성화
        setObserveType(Type.NON);
        getAttributes().setObservable();

        setParentPath(parentPath);

        Timer timer = new Timer();
        timer.schedule(new UpdateTask(), 0, 1);
}
```

[Line 44] : Observe 상태 체크

```
private class UpdateTask extends TimerTask {
        public void run() {

                if (global.getObs_emaProtocolCoAP_EventFlag().containsKey(name)) {
                        if (global.getObs_emaProtocolCoAP_EventFlag().get(name).isEventFlag()) {
                                changed();

                        }
                }
        }
}
```

4. EMS : Package Explanation OpenADR

CoAP Observe Class[Push] 별도 Class : COAP Client Observe function for Push

… [Line 63]Observe Initial

```
Response response = new Response(ResponseCode.CONTENT);
if ((!global.getObs_emaProtocolCoAP_EventFlag().containsKey(name))
                        || (!global.getObs_emaProtocolCoAP_EventFlag().get(name).isEventFlag())){

            response.setPayload("Initial_Success");
            exchange.respond(response);
            new Thread(new Runnable() {

                        public void run() {
                                    global.obs_emaProtocolCoAP_EventFlag.put(name, new EMAP_CoAP_EMA_DR());
            }}).start();

}
```

[Line 152] Event Send

```
if (getParentPath().contains("EMAP")) {

            new Thread(new Runnable() {
                        public void run() {
                        global.obs_emaProtocolCoAP_EventFlag.replace(name,
                        new EMAP_CoAP_EMA_DR().setEventFlag(false));
                        }
            }).start();

            …. Event Send
}
```

4. EMS : Package Explanation OpenADR

MQTT Class : 수/발신 메시지에 따라 서비스 분류 (Session Setup/Report/Event/Opt)

… [Line 183]

```
if (topicParse[1].equals("OpenADR")) {

            String profileVersion = "OpenADR2.0b";
            String service = msg_json.getString("service");
            service = service.replaceAll("oadr", "");
            // Session Setup
            if (topicParse[4].equals("SessionSetup")) {
                        if (service.matches("QueryRegistration|oadrQueryRegistration"))
                        service = "CONNECTREGISTRATION";
                        new SessionSetup(client, service, msg_json, profileVersion).start();
            }

            // Report
            else if (topicParse[4].equals("Report")) {
                        new Report(client, service, msg_json, profileVersion).start();
            }
            // Event
            else if (topicParse[4].matches("Event|Poll")) {
                        new DemandResponseEvent(client, service, msg_json, profileVersion).start();
            }
            // Opt
                        else if (topicParse[4].matches("Opt")) {
                        new Opt(client, service, msg_json, profileVersion).start();
            }


}
```

4. EMS : Package Explanation OpenADR

Session Setup/Report/Event/Opt Class : 상세 서비스 분류 ConnectRegistration, Poll…

… [Line 182]

```
switch (type) {
        case CONNECTREGISTRATION:
                this.setPayload = acknowledgeCONNECTREGISTRATION(payload);
                break;
        case CREATEPARTYREGISTRATION:
                this.setPayload = acknowledgeCREATEPARTYREGISTRATION(payload);
                break;
        case REGISTERREPORT:
                this.setPayload = acknowledgeREGISTERREPORT(payload);
                break;
        case POLL:
                this.setPayload = acknowledgePOLL(payload);
                break;
        case REGISTEREDREPORT:
                this.setPayload = acknowledgeREGISTEREDREPORT(payload);
                break;
        case REQUESTEVENT:
                this.setPayload = acknowledgeREQUESTEVENT(payload);
                break;
        case CANCELPARTYREGISTRATION:
                this.setPayload = acknowledgeCANCELPARTYREGISTRATION(payload);

                break;
```

4. EMS : Package Explanation OpenADR

각 CreatedPartyRegistration Class : 상세 서비스 CreatedPartyRegistration Message Build up

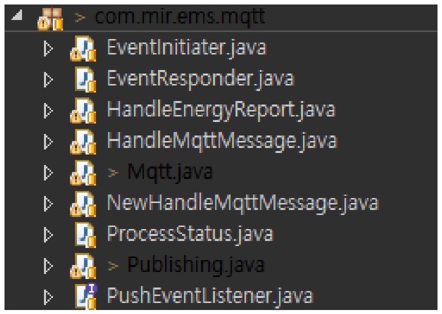
… [Line 101] JSON Message build up

```
public class CreatedPartyRegistration {

        private String srcEMA, destEMA, responseDescription, requestID, duration, service, registrationID;
        private int responseCode;
        private String profile;

        public CreatedPartyRegistration() {

        }
        @Override
        public String toString() {


                return "{\"vtnID" + "\":" + "\"" + getSrcEMA() + "\"" + ", "
                + "\"venID" + "\":" + "\"" + getDestEMA() + "\"" + ", "
                +"\"responseCode" + "\":" + "\"" + getResponseCode() + "\"" + ", "
                +"\"responseDescription" + "\":" + "\"" + getResponseDescription() + "\"" + ", "
                +"\"requestID" + "\":" + "\"" + getRequestID() + "\"" + ", "
                +"\"duration" + "\":" + "\"" + getDuration() + "\"" + ", "
                +"\"service" + "\":" + "\"" + getService() + "\"" + ", "
                +"\"registrationID" + "\":" + "\"" + getRegistrationID() + "\"" + ", "
                + "\"oadrProfile" + "\": "+ getProfile() + "}";

        }
```

4. EMS : Package Explanation EMAP

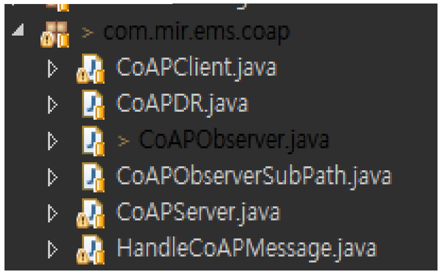
Protocol

MQTT



com.mir.ems.mqtt
- EventInitiater.java
- EventResponder.java
- HandleEnergyReport.java
- HandleMqttMessage.java
- Mqtt.java
- NewHandleMqttMessage.java
- ProcessStatus.java
- Publishing.java
- PushEventListener.java

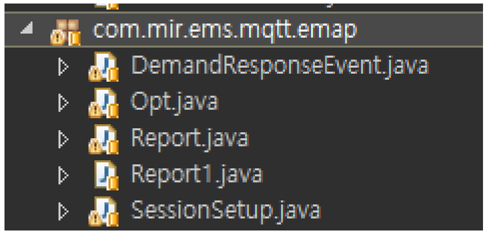
COAP



com.mir.ems.coap
- CoAPClient.java
- CoAPDR.java
- CoAPObserver.java
- CoAPObserverSubPath.java
- CoAPServer.java
- HandleCoAPMessage.java

Service

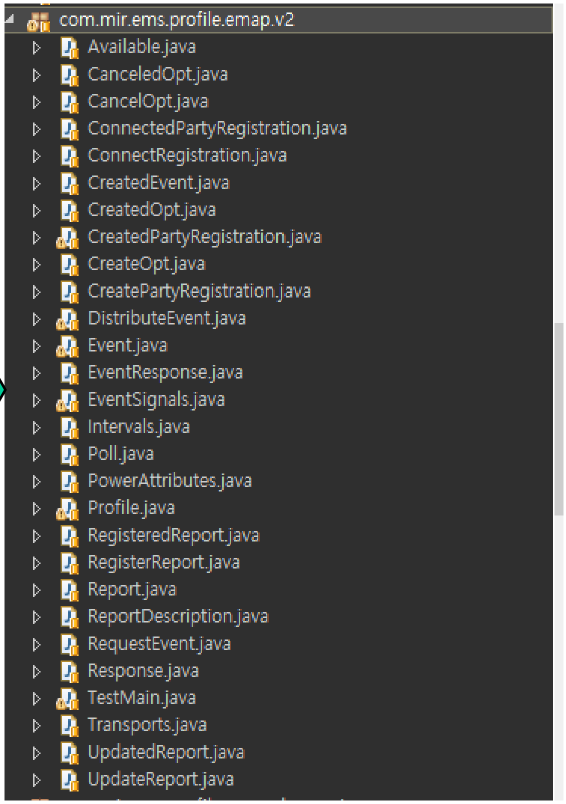
프로토콜 서비스 참조



com.mir.ems.mqtt.emap
- DemandResponseEvent.java
- Opt.java
- Report.java
- Report1.java
- SessionSetup.java

각 서비스에 맞는 모델링 참조

EMAP Model Package



com.mir.ems.profile.emap.v2
- Available.java
- CanceledOpt.java
- CancelOpt.java
- ConnectedPartyRegistration.java
- ConnectRegistration.java
- CreatedEvent.java
- CreatedOpt.java
- CreatedPartyRegistration.java
- CreateOpt.java
- CreatePartyRegistration.java
- DistributeEvent.java
- Event.java
- EventResponse.java
- EventSignals.java
- Intervals.java
- Poll.java
- PowerAttributes.java
- Profile.java
- RegisteredReport.java
- RegisterReport.java
- Report.java
- ReportDescription.java
- RequestEvent.java
- Response.java
- TestMain.java
- Transports.java
- UpdatedReport.java
- UpdateReport.java

4. EMS : Package Explanation EMAP

CoAP Server Class : 수/발신 메시지 서비스에 따른 분류

… [Line 952]

```
                    server.add(new Emap("EMAP"));

                    server.add(new OpenADR("OpenADR"));

                    // Observe
                    server.add(new CoAPObserver("OpenADR2.0b"));
                    server.add(new CoAPObserver("EMAP1.0b"));

public class Emap extends CoapResource {

          public Emap(String name) {
                    // TODO Auto-generated constructor stub
                    super(name);

                    add(new SystemID(global.SYSTEM_ID, name));

          }
}
….
                    // EMAP
                    add(new SessionSetup("SessionSetup"));
                    add(new Report("Report"));
                    add(new Opt("Opt"));
                    add(new DemandResponseEvent("Event"));
                    add(new DemandResponseEvent("Poll"));
```

4. EMS : Package Explanation EMAP

CoAP Observe Class[Push] 별도 Class : COAP Client Observe function for Push

```
public CoAPObserverSubPath(String name, String parentPath) {
        super(name);
        this.name = name;
        setObservable(true);                                    // Observe 활성화
        setObserveType(Type.NON);
        getAttributes().setObservable();

        setParentPath(parentPath);

        Timer timer = new Timer();
        timer.schedule(new UpdateTask(), 0, 1);
}
```

[Line 44] : Observe 상태 체크

```
private class UpdateTask extends TimerTask {
        public void run() {

                if (global.getObs_emaProtocolCoAP_EventFlag().containsKey(name)) {
                        if (global.getObs_emaProtocolCoAP_EventFlag().get(name).isEventFlag()) {
                                changed();

                        }
                }
        }
}
```

4. EMS : Package Explanation EMAP

CoAP Observe Class[Push] 별도 Class : COAP Client Observe function for Push

… [Line 63]Observe Initial

```
Response response = new Response(ResponseCode.CONTENT);
if ((!global.getObs_emaProtocolCoAP_EventFlag().containsKey(name))
                    || (!global.getObs_emaProtocolCoAP_EventFlag().get(name).isEventFlag())){

          response.setPayload("Initial_Success");
          exchange.respond(response);
          new Thread(new Runnable() {

                    public void run() {
                              global.obs_emaProtocolCoAP_EventFlag.put(name, new EMAP_CoAP_EMA_DR());
          }}).start();

}
```

[Line 152] Event Send

```
if (getParentPath().contains("EMAP")) {

          new Thread(new Runnable() {
                    public void run() {
                    global.obs_emaProtocolCoAP_EventFlag.replace(name,
                    new EMAP_CoAP_EMA_DR().setEventFlag(false));
                    }
          }).start();

          …. Event Send
}
```

4. EMS : Package Explanation EMAP

MQTT Class : 수/발신 메시지에 따라 서비스 분류 (Session Setup/Report/Event/Opt)

… [Line 147]

```
if (topicParse[1].equals("EMAP")) {

String profileVersion = "EMAP1.0b";

if (msg_json.getString("DestEMA").equals(global.getSYSTEM_ID())) {

String service = msg_json.getString("service");

// Session Setup
if (topicParse[4].equals("SessionSetup")) {
                new SessionSetup(client, service, msg_json, profileVersion).start();
}
// Report
else if (topicParse[4].equals("Report")) {
                new Report(client, service, msg_json, profileVersion).start();
}
// Event
else if (topicParse[4].matches("Event|Poll")) {
                new DemandResponseEvent(client, service, msg_json, profileVersion).start();
}
// Opt
else if (topicParse[4].matches("Opt")) {
                new Opt(client, service, msg_json, profileVersion).start();
}
}
}
```

4. EMS : Package Explanation EMAP

Session Setup/Report/Event/Opt Class : 상세 서비스 분류 ConnectRegistration, Poll…

… [Line 182]

```
switch (type) {
                case CONNECTREGISTRATION:
                        this.setPayload = acknowledgeCONNECTREGISTRATION(payload);
                        break;
                case CREATEPARTYREGISTRATION:
                        this.setPayload = acknowledgeCREATEPARTYREGISTRATION(payload);
                        break;
                case REGISTERREPORT:
                        this.setPayload = acknowledgeREGISTERREPORT(payload);
                        break;
                case POLL:
                        this.setPayload = acknowledgePOLL(payload);
                        break;
                case REGISTEREDREPORT:
                        this.setPayload = acknowledgeREGISTEREDREPORT(payload);
                        break;
                case REQUESTEVENT:
                        this.setPayload = acknowledgeREQUESTEVENT(payload);
                        break;
                case CANCELPARTYREGISTRATION:
                        this.setPayload = acknowledgeCANCELPARTYREGISTRATION(payload);

                        break;
```

4. EMS : Package Explanation EMAP

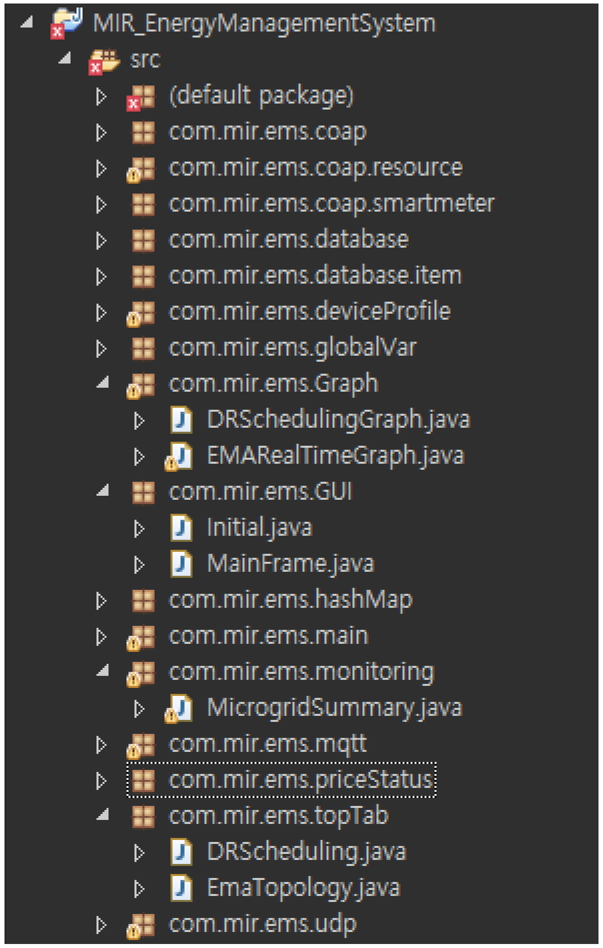
각 ConnectedPartyRegistration Class : 상세 서비스 ConnectedPartyRegistration Message Build up

… [Line 101] JSON Message build up

```
public class ConnectedPartyRegistration {

        private String srcEMA, destEMA, responseDescription, requestID, duration, service, version, time;
        private int responseCode;
        private String profile;

        @Override
        public String toString() {

                return "{\"SrcEMA" + "\":" + "\"" + getSrcEMA() + "\"" + ", "
                                + "\"DestEMA" + "\":" + "\"" + getDestEMA() + "\"" + ", "
                                +"\"responseCode" + "\":" + "\"" + getResponseCode() + "\"" + ", "
                                +"\"responseDescription" + "\":" + "\"" + getResponseDescription() + "\"" + ",
"

                                +"\"requestID" + "\":" + "\"" + getRequestID() + "\"" + ", "
                                +"\"duration" + "\":" + "\"" + getDuration() + "\"" + ", "
                                +"\"service" + "\":" + "\"" + getService() + "\"" + ", "
                                +"\"version" + "\":" + "\"" + getVersion() + "\"" + ", "
                                +"\"time" + "\":" + "\"" + getTime() + "\"" + ", "
                                +"\"registrationID" + "\":" + "\"" + "" + "\"" + ", "
                                + "\"profile" + "\": "+ getProfile() + "}";
        }
```

4. EMS : Package Explanation Monitoring

Monitoring



```
◢ MIR_EnergyManagementSystem
  ◢ src
    ▷ (default package)
    ▷ com.mir.ems.coap
    ▷ com.mir.ems.coap.resource
    ▷ com.mir.ems.coap.smartmeter
    ▷ com.mir.ems.database
    ▷ com.mir.ems.database.item
    ▷ com.mir.ems.deviceProfile
    ▷ com.mir.ems.globalVar
    ◢ com.mir.ems.Graph
      ▷ DRSchedulingGraph.java
      ▷ EMARealTimeGraph.java
    ◢ com.mir.ems.GUI
      ▷ Initial.java
      ▷ MainFrame.java
    ▷ com.mir.ems.hashMap
    ▷ com.mir.ems.main
    ◢ com.mir.ems.monitoring
      ▷ MicrogridSummary.java
    ▷ com.mir.ems.mqtt
    ▷ com.mir.ems.priceStatus
    ◢ com.mir.ems.topTab
      ▷ DRScheduling.java
      ▷ EmaTopology.java
    ▷ com.mir.ems.udp
```

com.mir.ems.monitoring

-MicrogridSummary:

To show the microgrid status (ess, pv, resource)

com.mir.ems.Graph

- EMARealTimeGraph:

To show EMA'S Energy USE on Real Time Graph

- EnergyGraph:

To show total energy use on Real Time Graph-

-EMATopology:

To show EMA Topology

- DRSchedulingGraph:

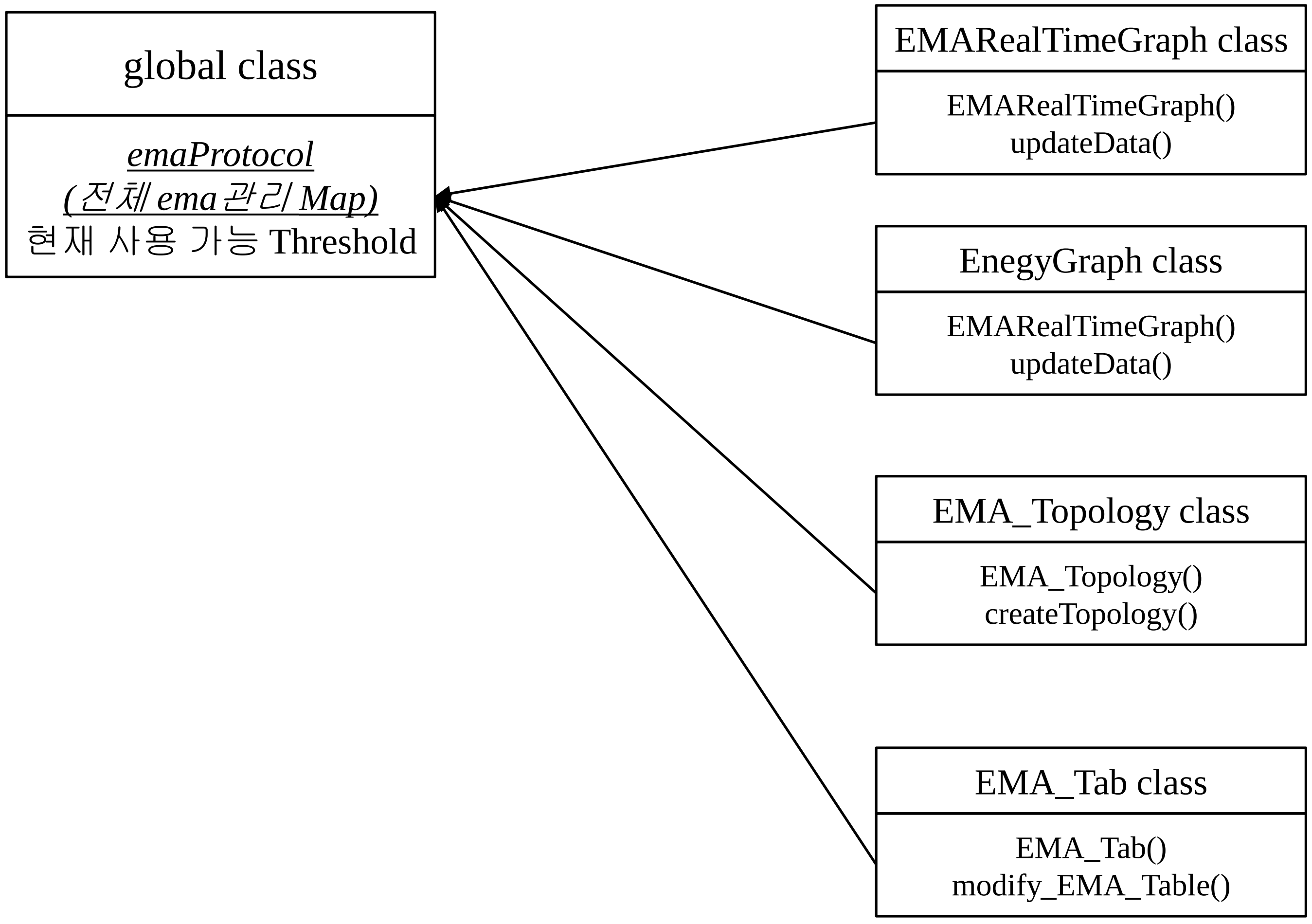
To show when you give a event to EMA

com.mir.ems.deviceProfile

: In order to store Devices profile in JVM

- **EMA_TAB**
- **LED_TAB**

4. EMS : Package Explanation Monitoring



현재 메모리에 저장되어있는
EMA 정보를 참조하여
에너지 사용량 업데이트

2Line Graph
(1) 현재 메모리에 저장되어있는
EMA 정보를 참조하여
총 더 한 후 에너지 사용량 업데이

(2) 현재 사용 가능한 Threshold 참

현재 메모리에 저장되어있는
EMA 정보를 참조하여
Topology 업데이트
(주기 5초)

현재 메모리에 저장되어있는
EMA 정보를 참조하여
Table 정보 업데이트
(주기 5초)

LED Tab 동일한 구조

4. EMS : Package Explanation
Monitoring : EMARealTimeGraph

EMARealTimeGraph Class : 각각의 EMA에 대한 실시간 그래프

… [Line 49] 그래프 생성 및 그래프 업데이트 주기 설정

```
public EMARealTimeGraph() {

        setBounds(14, 60, 1467, 700);
        final XYChart chart = getChart();
        setLayout(null);
        XChartPanel chartPanel = new XChartPanel(chart);
        chartPanel.setBackground(Color.WHITE);
        chartPanel.setBounds(0, 0, 1467, 700);
        add(chartPanel);

        TimerTask chartUpdaterTask = new TimerTask() {
                @Override
                public void run() {

                        updateData();

                        javax.swing.SwingUtilities.invokeLater(new Runnable() {
                                public void run() {
                                        repaint();
                                }
                        });
                }
        };
        Timer timer = new Timer();
        timer.scheduleAtFixedRate(chartUpdaterTask, 2000, 2000); (2초 주기로 그래프 업데이트)
}
```

4. EMS : Package Explanation
Monitoring : EMARealTimeGraph

EMARealTimeGraph Class : 각각의 EMA에 대한 실시간 그래프

… [Line 109] 그래프 업데이트 함수, X축 Y축 업데이트

```
@SuppressWarnings("unchecked")
public void updateData() {
		List<Date> newXdata = getCurrentTime();
		xData.addAll(newXdata);
		int emaListSize;
		emaListSize = global.emaProtocolCoAP.size();

		for (int i = 0; i < emaListSize; i++) {
				arr[i][1] = getRandomData((List<Double>) arr[i][1], i);
		}

		for(int i=emaListSize;i<20; i++){
				arr[i][1] = getRandomData((List<Double>) arr[i][1], i);
		}

		for(int i=0; i<20; i++){
				xyChart.updateXYSeries((String)arr[i][0], xData, (List<Double>)arr[i][1], null);
		}

}
```

4. EMS : Package Explanation Monitoring : EnergyGraph

EnergyGraph Class : 각각의 EMA에 대한 실시간 그래프

… [Line 52] 그래프 생성 및 그래프 업데이트 주기 설정

```
public EnergyGraph() {

        final XYChart chart = getChart();

        setBounds(14, 60, 1467, 700);
        setLayout(null);

        @SuppressWarnings({ "rawtypes", "unchecked" })
        XChartPanel chartPanel = new XChartPanel(chart);
        chartPanel.setBackground(Color.WHITE);
        chartPanel.setBounds(0, 0, 1467, 700);
        add(chartPanel);

        TimerTask chartUpdaterTask = new TimerTask() {          Timer timer = new Timer();
        timer.scheduleAtFixedRate(chartUpdaterTask, 2000, 2000); (2초 주기로 그래프 업데이트)
}
```

4. EMS : Package Explanation
Monitoring : EnergyGraph

EnergyGraph Class : 각각의 EMA에 대한 실시간 그래프

… [Line 196] Y축 Sin Graph, X축 현재 시간 업데이트 함수

```
        private List<Double> getYAXIS() {

                double radians = phase + (2 * Math.PI / 100 * val);
                val += 1;
                phase += ((2 * Math.PI * 2) / 20.0) / PERIOD;

                //Sin graph
                global.THRESHOLD = (YAXIS_TRANSFERENCE * Math.sin(radians) + BASEWATT) * 1000;
                global.AVAILABLE_THRESHOLD = (global.THRESHOLD - (global.THRESHOLD / 
global.RESERVE_THRESHOLD_PERCENTAGE));
                global.RESERVE_THRESHOLD = global.THRESHOLD - global.AVAILABLE_THRESHOLD;

                yData.add(YAXIS_TRANSFERENCE * Math.sin(radians) + BASEWATT);

                return yData;
        }

        private List<Date> getEMAAXIS() {

                //현재 시간
                long now = System.currentTimeMillis();
                Date date = new Date(now);
                totalEMAxData.add(date);

                return totalEMAxData;
        }
```

4. EMS : Package Explanation Monitoring : EMATopology

EMATopology Class : 각각의 EMA에 대한 토폴로지 그래프

… [Line 34] 토폴로지 그래프 생성

```
public EmaTopology() {
			java.net.URL emsUrl = EmaTopology.class.getResource("/IMAGE/dddd.png");
			System.setProperty("gs.ui.renderer", "org.graphstream.ui.j2dviewer.J2DGraphRenderer");
			Viewer viewer = new Viewer(graph, Viewer.ThreadingModel.GRAPH_IN_GUI_THREAD);
			viewer.disableAutoLayout();

			ViewPanel topologyPanel = (ViewPanel) viewer.addDefaultView(false);
			topologyPanel.setSize(1467, 700);
			add(topologyPanel);

			setBounds(14, 60, 1467, 700);
			setLayout(new BorderLayout(0, 0));
			setVisible(true);

			Node a = graph.addNode("EMS");
			a.addAttribute("ui.label", a.getId());

			int sum = 0;
			for (int i = 0; i < 20; i++) {
					sum += (i * 20);
			}
			a.setAttribute("x", (sum / 40));
			a.setAttribute("y", 10);

			a.addAttribute("ui.style","text-alignment: above; size: 65px, 65px; shape: rounded-box; size-mode: fit; fill-mode: image-scaled; fill-image: url('"+ emsUrl + "');");
			createTopology();
```

4. EMS : Package Explanation
Monitoring : EMATopology

EMATopology Class : 각각의 EMA에 대한 토폴로지 정보

… [Line 34] 그래프 생성 및 그래프 업데이트 주기 설정

```
TimerTask chartUpdaterTask = new TimerTask() {
            Node emaGroup = null;
            Node deviceGroup = null;
            @Override
            public void run() {
                        int cnt = 0;
                        int devCnt = 0;
                        for (int i = 0; i < emaList.length; i++) {
                        String key = emaList[i];
                        if (!strSet.contains(key.toString())) {
                                    cnt += 1;
                        try {
                                    // NODE 추가
                                    emaGroup = graph.addNode(key);
                                    // NODE 생성 위치
                                    emaGroup.setAttribute(" x " , (cnt * 10));
                                    // NODE ID 설정
                                    emaGroup.addAttribute("ui.label", emaGroup.getId());
                                    emaGroup.setAttribute("y", 0);

                                    //NODE 사이즈 설정
                                                emaGroup.addAttribute( " ui.style " ,
                                     " text-alignment: under; size: 65px, 65px; shape: rounded-box; size-mode: fit; fill-mode:
image-scaled; fill-image: url( '  " + gatewayUrl +  "  ' ); " );
                                    // EDGE 설정
                                    graph.addEdge(emsEdge + key, emsEdge, key);

            …..
```

4. EMS : Package Explanation Monitoring : EMA_Tab

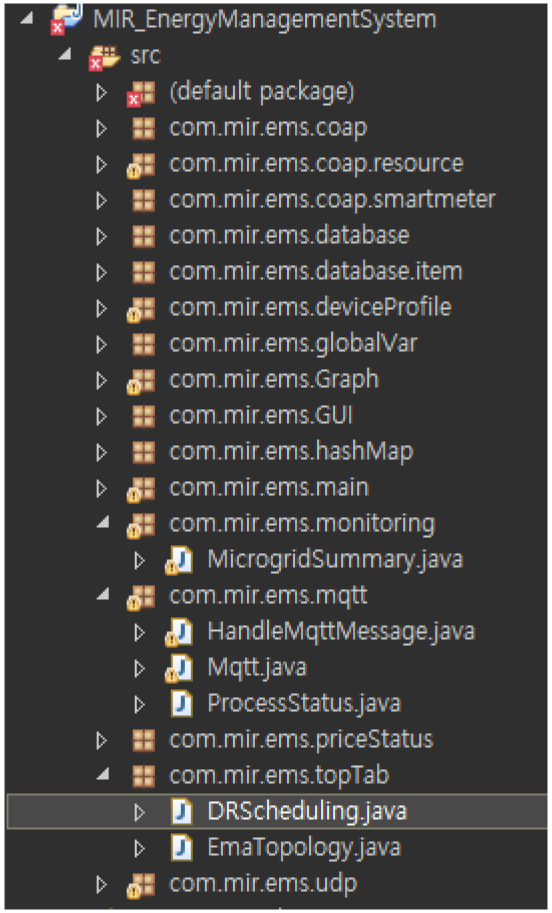
EMA_tab Class : 각각의 EMA에 대한 상세 정보 테이블

… [Line 23] 테이블 생성 및 테이블 업데이트 주기 설정

```
public void modify_EMA_table() {

                    int ema_rows_num = EMA_tab_temp.ema_table_model.getRowCount();
                    for (int i = ema_rows_num - 1; i >= 0; i--) {
                              EMA_tab_temp.ema_table_model.removeRow(i);
                    }

                    Iterator<String> keys = global.emaProtocolCoAP.keySet().iterator();

                    while (keys.hasNext()) {
                              String key = keys.next();

                              // 업데이트 항목을 global ema관리 Map에서 참조하여 업데이트
                              EMA_tab_temp.ema_table_model
                                                  .addRow(new Object[] { false, key,
global.emaProtocolCoAP.get(key).getProtocol(),

          global.emaProtocolCoAP.get(key).getqOs(), global.emaProtocolCoAP.get(key).getEmaCNT(),

          global.emaProtocolCoAP.get(key).getPower(), global.emaProtocolCoAP.get(key).getMaxValue(),

          global.emaProtocolCoAP.get(key).getMinValue(), global.emaProtocolCoAP.get(key).getMargin(),

          global.emaProtocolCoAP.get(key).getCustomerPriority() });

          }
```

4. EMS : Package Explanation Control

Control



```
◢ MIR_EnergyManagementSystem
    ◢ src
        ▷ (default package)
        ▷ com.mir.ems.coap
        ▷ com.mir.ems.coap.resource
        ▷ com.mir.ems.coap.smartmeter
        ▷ com.mir.ems.database
        ▷ com.mir.ems.database.item
        ▷ com.mir.ems.deviceProfile
        ▷ com.mir.ems.globalVar
        ▷ com.mir.ems.Graph
        ▷ com.mir.ems.GUI
        ▷ com.mir.ems.hashMap
        ▷ com.mir.ems.main
        ◢ com.mir.ems.monitoring
            ▷ MicrogridSummary.java
        ◢ com.mir.ems.mqtt
            ▷ HandleMqttMessage.java
            ▷ Mqtt.java
            ▷ ProcessStatus.java
        ▷ com.mir.ems.priceStatus
        ◢ com.mir.ems.topTab
            ▷ DRScheduling.java
            ▷ EmaTopology.java
        ▷ com.mir.ems.udp
```

com.mir.ems.topTab

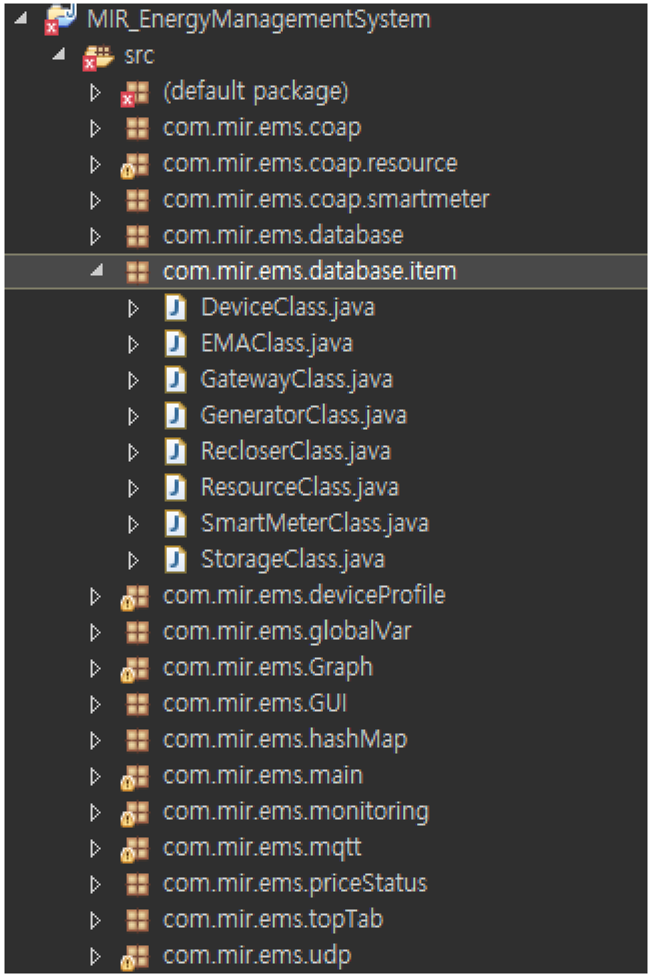
- DRScheduling:

Send DR Message to EMA

It is possible to send Push and Multicast Message here

4. EMS : Package Explanation Profile

Device Profile



```
◢ MIR_EnergyManagementSystem
  ◢ src
    ▷ (default package)
    ▷ com.mir.ems.coap
    ▷ com.mir.ems.coap.resource
    ▷ com.mir.ems.coap.smartmeter
    ▷ com.mir.ems.database
    ◢ com.mir.ems.database.item
      ▷ DeviceClass.java
      ▷ EMAClass.java
      ▷ GatewayClass.java
      ▷ GeneratorClass.java
      ▷ RecloserClass.java
      ▷ ResourceClass.java
      ▷ SmartMeterClass.java
      ▷ StorageClass.java
    ▷ com.mir.ems.deviceProfile
    ▷ com.mir.ems.globalVar
    ▷ com.mir.ems.Graph
    ▷ com.mir.ems.GUI
    ▷ com.mir.ems.hashMap
    ▷ com.mir.ems.main
    ▷ com.mir.ems.monitoring
    ▷ com.mir.ems.mqtt
    ▷ com.mir.ems.priceStatus
    ▷ com.mir.ems.topTab
    ▷ com.mir.ems.udp
```

com.mir.ems.database.item

<u>**Below all classes are Generic Class**</u>

:To make object type

DeviceClass

EMAClass

SmartMeterClass

…

com.mir.ems.hashMap

:Give key value each devices for easy to handle and search

ESS_values

PV_values

Recloser_values

Resource_values

VTN_values

4. EMS : Package Explanation Monitoring : global

gloal Class : 각각의 EMA에 대한 정보를 저장하는 Map

… [Line 23] 테이블 생성 및 테이블 업데이트 주기 설정

```
public static ConcurrentHashMap<String, EMA> emaProtocol = new ConcurrentHashMap<String, EMA>();

// EMA 정보 저장
public static void putEmaProtocol (String emaID, EMA) {
                try {
                                Thread.sleep(20);
                                emaProtocol.put(emaID, EMA);

                } catch (InterruptedException e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
                }

}

// Set EMA 정보
public static void setEmaProtocol (ConcurrentHashMap<String, EMA> emaProtocol) {
                global.emaProtocol = emaProtocol;
}
```

*** TIP: ConcurrentHashMap 은 비 동기 방식의 HashMap으로 빠른 응답이 필요하거나 Map내에 Sorting이 필요하지 않은 경우 사용한다. 멀티 스레딩 방식에서 주로 사용하는 방식이다.**

4. EMS : Package Explanation Profile : EMA

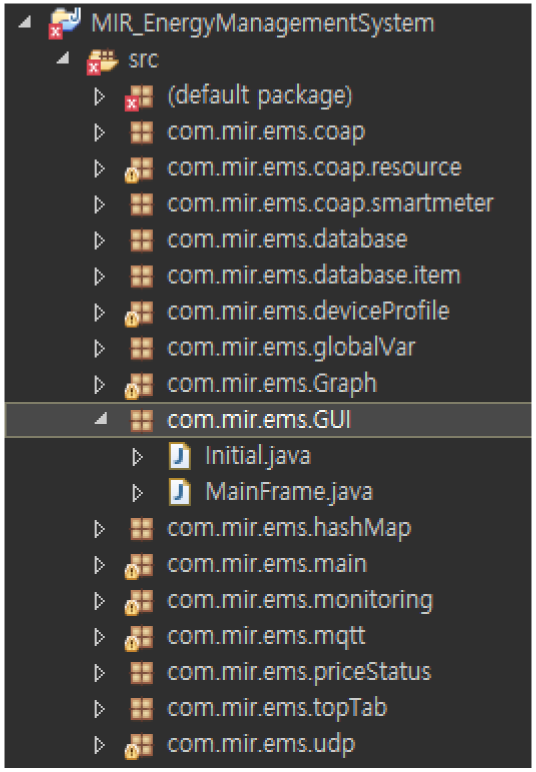
EMA Class : 각각의 EMA에 대한 상세 정보 테이블

… [Line 23] 테이블 생성 및 테이블 업데이트 주기 설정

```
public class EMA {

        private String emaID, qOs, type, registrationID, transportName, transportAddress, reportName, reportType, state,
                        profileName, requestID, version;
        private String time, maxTime, minTime, connect, protocol;
        private int customerPriority, reportOnly, httpPullModel, xmlSignature, emaCNT, priority, dimming;
        private double margin, minValue, maxValue, avgValue, power, generate, storage;
        private boolean pullModel, tableChanged, realTimetableChanged;
        private JSONObject EMARegisteredInfo, EMARegisteredMgnInfo;

        // EMA 정보를 저장하는 구조체
        public Emap_Cema_Profile(String protocol, String emaID, String registrationID, String qos, String state, double power, int
dimming, double margin, double generate, double storage, double maxValue, double minValue, double avgValue, String maxTime, String
minTime, int priority, boolean pullModel, boolean tableChanged, boolean realTimetableChanged, String connect) {

                setProtocol(protocol);
                setRegistrationID(registrationID);
                setEmaID(emaID);
                setqOs(qos);
                setState(state);
                setPower(power);
                …..
        }
```

4. EMS : Package Explanation GUI

GUI



```
MIR_EnergyManagementSystem
  src
    (default package)
    com.mir.ems.coap
    com.mir.ems.coap.resource
    com.mir.ems.coap.smartmeter
    com.mir.ems.database
    com.mir.ems.database.item
    com.mir.ems.deviceProfile
    com.mir.ems.globalVar
    com.mir.ems.Graph
    com.mir.ems.GUI
      Initial.java
      MainFrame.java
    com.mir.ems.hashMap
    com.mir.ems.main
    com.mir.ems.monitoring
    com.mir.ems.mqtt
    com.mir.ems.priceStatus
    com.mir.ems.topTab
    com.mir.ems.udp
```

com.mir.ems.GUI

Initial	**- First Page**
MainFrame	**- Main Frame that will be shown after first**

4. EMS : Package Explanation
GUI : Initial

Initial Class : IP 설정 및 프로토콜, 프로파일 설정 GUI

GUI 빌드 함수

```
public Initial() {
                // setting
                setTitle("MIREnergy Management System");
                setSize(326, 614);
                setResizable(false);
                setLocation(800, 450);
                setDefaultCloseOperation(EXIT_ON_CLOSE);
                setLocationRelativeTo(null);
                // panel
                JPanel panel = new JPanel();
                placeLoginPanel(panel);

                // add
                getContentPane().add(panel);

                JPanel panel_1 = new JPanel();
                panel_1.setBorder(new EtchedBorder(EtchedBorder.LOWERED, null, null));
                panel_1.setBounds(12, 66, 298, 118);
                panel.add(panel_1);
                panel_1.setLayout(null);
                ..
}
```

4. EMS : Package Explanation GUI : Initial

Initial Class : IP 설정 및 프로토콜 설정 GUI

GUI 빌드 함수

```
public Initial() {

        //IP 설정
        final Jlabel lblNewLabel = new Jlabel(" IP Address " );
        lblNewLabel.setFont(new Font( " Arial " , Font.BOLD, 13));
        lblNewLabel.setBounds(39, 75, 76, 25);
        panel_2.add(lblNewLabel);
        //PORT 설정
        final JLabel lblNewLabel_1 = new JLabel("Port");
        lblNewLabel_1.setFont(new Font("Arial", Font.BOLD, 13));
        lblNewLabel_1.setBounds(39, 110, 76, 25);
        panel_2.add(lblNewLabel_1);

        //Protocol 설정
        final JComboBox<String> comboBox = new JComboBox<String>();
        …
        comboBox.addItem("MQTT");
        comboBox.addItem("CoAP");
        comboBox.addItem("UDP");
        comboBox.addItem("BOTH");                ..
}
```

4. EMS : Package Explanation GUI : MainFrame

MainFrame Class : 가격 정보, 등록된 EMA 정보 CFG 파일을 가져오는 함수

```
public MainFrame() {

        // 가격정보를 가져오는 함수
        rdbtnmntmNewRadioItem_1.addActionListener(new ActionListener() {

        @Override
        public void actionPerformed(ActionEvent arg0) {
                // TODO Auto-generated method stub
                if (rdbtnmntmNewRadioItem_1.isSelected()) {
                        rdbtnmntmNewRadioItem.setSelected(false);

                        JFileChooser jfc = new
JFileChooser(FileSystemView.getFileSystemView().getHomeDirectory());
                        jfc.setDialogTitle("Select an configuration file");
                        jfc.setAcceptAllFileFilterUsed(false);
                        FileNameExtensionFilter filter = new FileNameExtensionFilter(".cfg files", "cfg",
"CFG");
                        jfc.addChoosableFileFilter(filter);

                        int returnValue = jfc.showOpenDialog(null);

                        if (returnValue == JFileChooser.APPROVE_OPTION)
                        new RealTimePriceFileReader(jfc.getSelectedFile().getPath());

                                }
                        }
                });
                ..
}
```

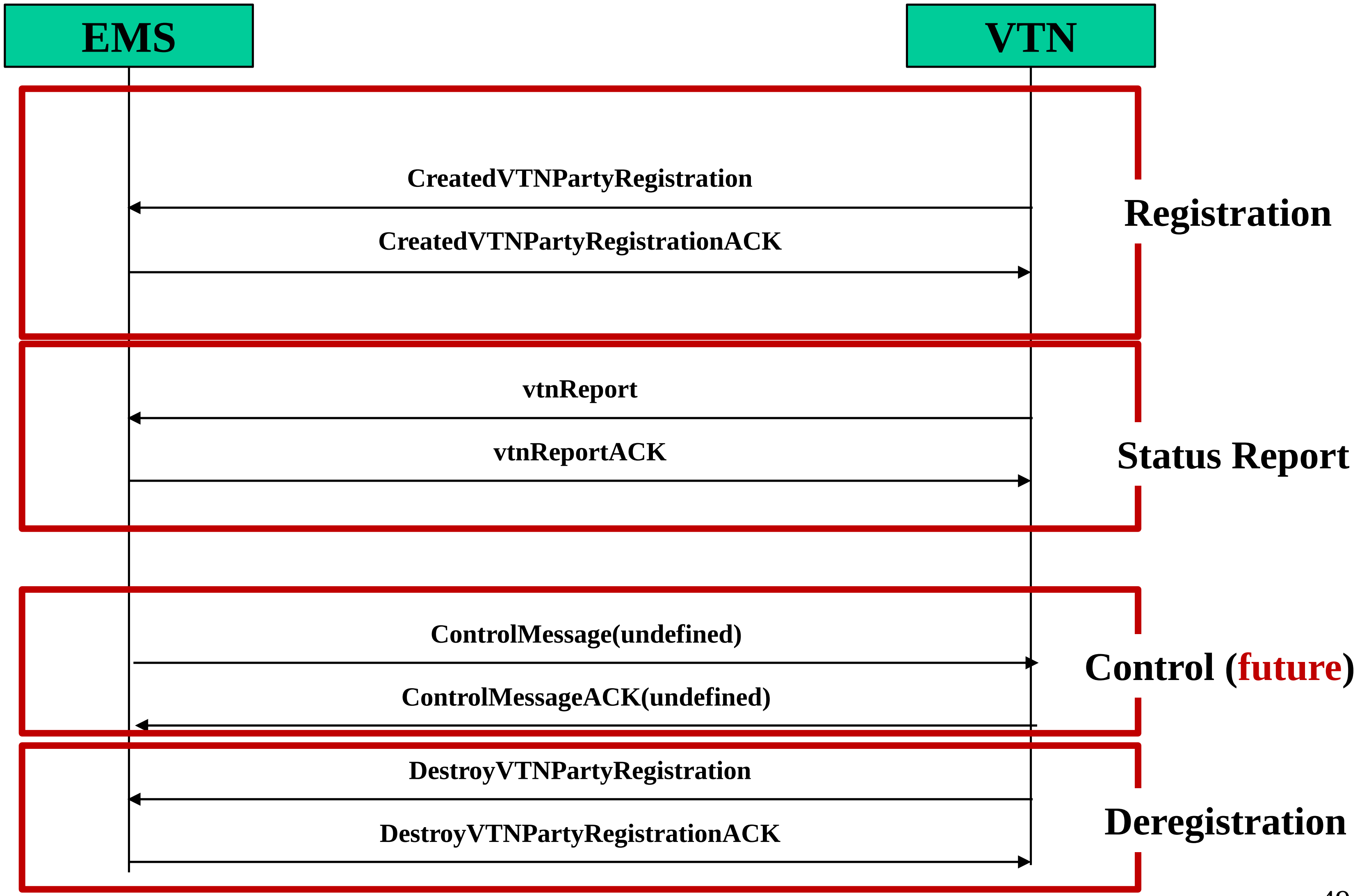
4. EMS : Package Explanation
GUI : MainFrame

Initial Class : IP 설정 및 프로토콜, 프로파일 설정 GUI

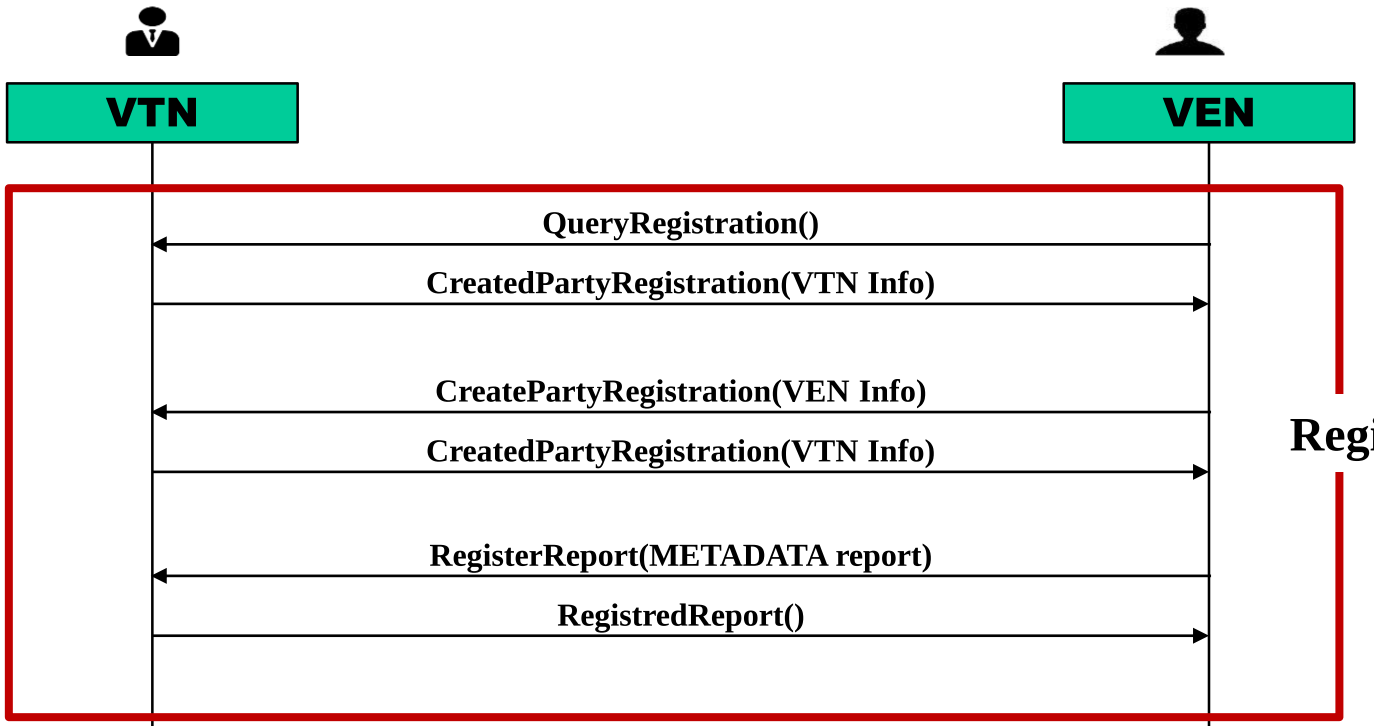
GUI 빌드 함수

```
public Initial() {
                // setting
                setTitle("MIREnergy Management System");
                setSize(326, 614);
                setResizable(false);
                setLocation(800, 450);
                setDefaultCloseOperation(EXIT_ON_CLOSE);
                setLocationRelativeTo(null);
                // panel
                JPanel panel = new JPanel();
                placeLoginPanel(panel);

                // add
                getContentPane().add(panel);

                JPanel panel_1 = new JPanel();
                panel_1.setBorder(new EtchedBorder(EtchedBorder.LOWERED, null, null));
                panel_1.setBounds(12, 66, 298, 118);
                panel.add(panel_1);
                panel_1.setLayout(null);
                ..
}
```

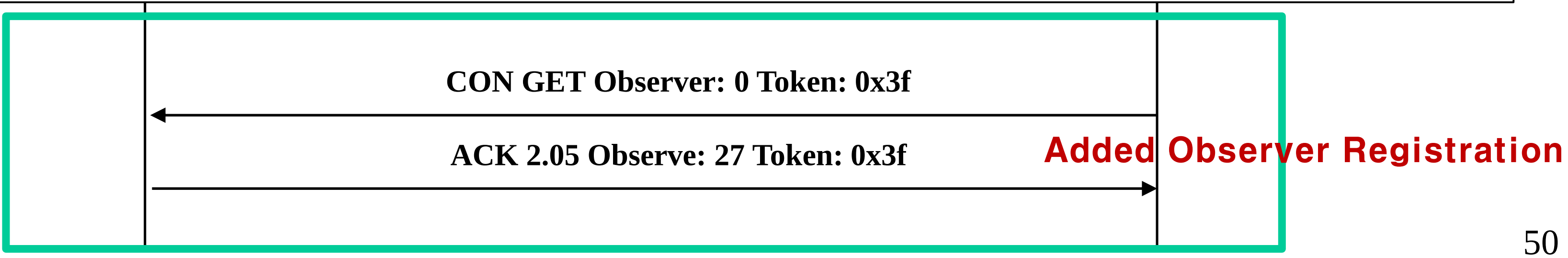
5. Message Flow of EMS-VTN(UDP)


EMS
VTN
CreatedVTNPartyRegistration
CreatedVTNPartyRegistrationACK
Registration
vtnReport
vtnReportACK
Status Report
ControlMessage(undefined)
ControlMessageACK(undefined)
Control (future)
DestroyVTNPartyRegistration
DestroyVTNPartyRegistrationACK
Deregistration

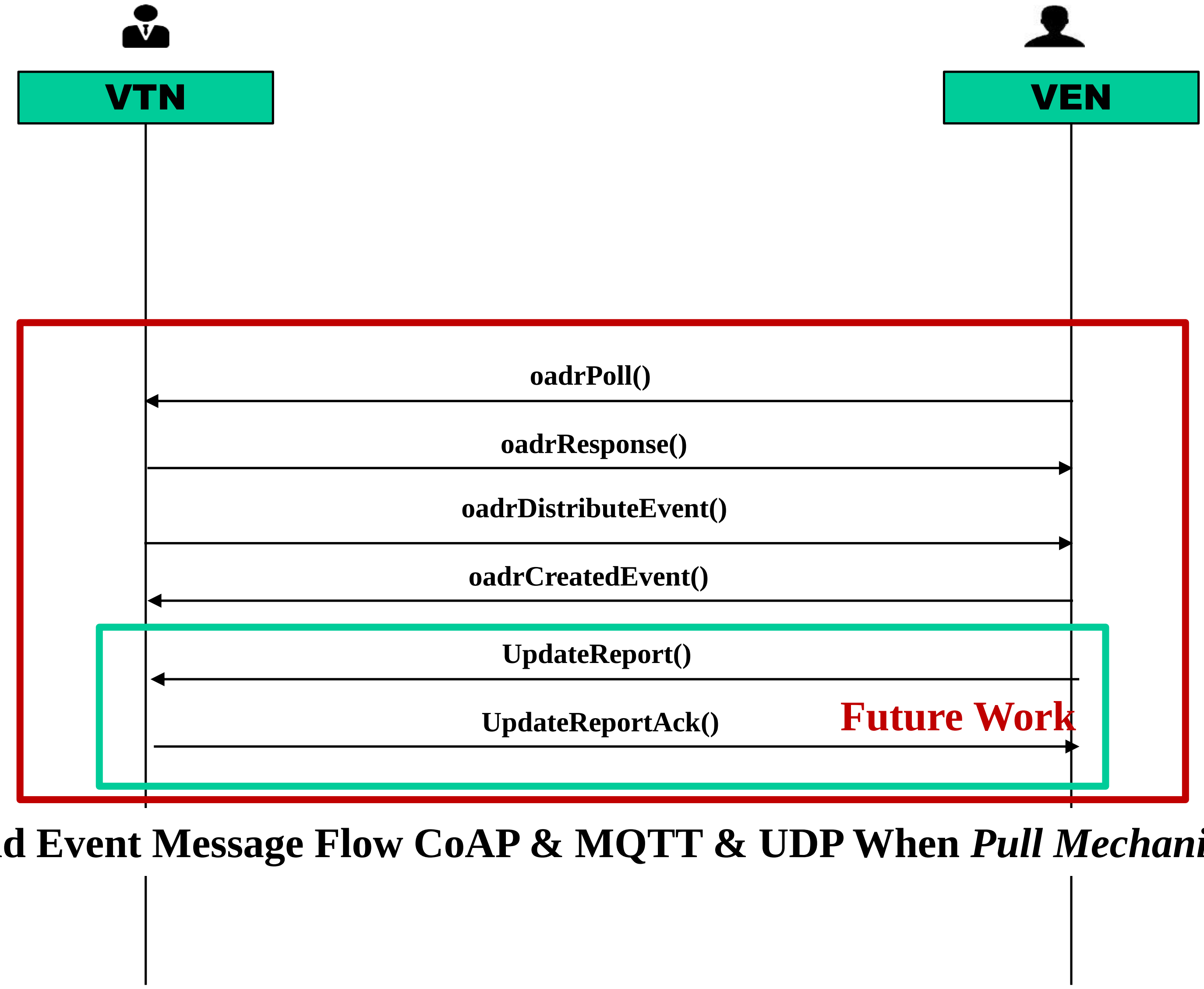
5. Message Flow of VTN-VEN(Registration)


VTN
VEN
QueryRegistration()
CreatedPartyRegistration(VTN Info)
CreatePartyRegistration(VEN Info)
CreatedPartyRegistration(VTN Info)
RegisterReport(METADATA report)
RegistredReport()
Registration

Above Registration Message Flow is Push(MQTT, UDP) and Pull(MQTT, CoAP, UDP) Mechanism;
However in case of CoAP Push mechanism, you should add Observer Registration

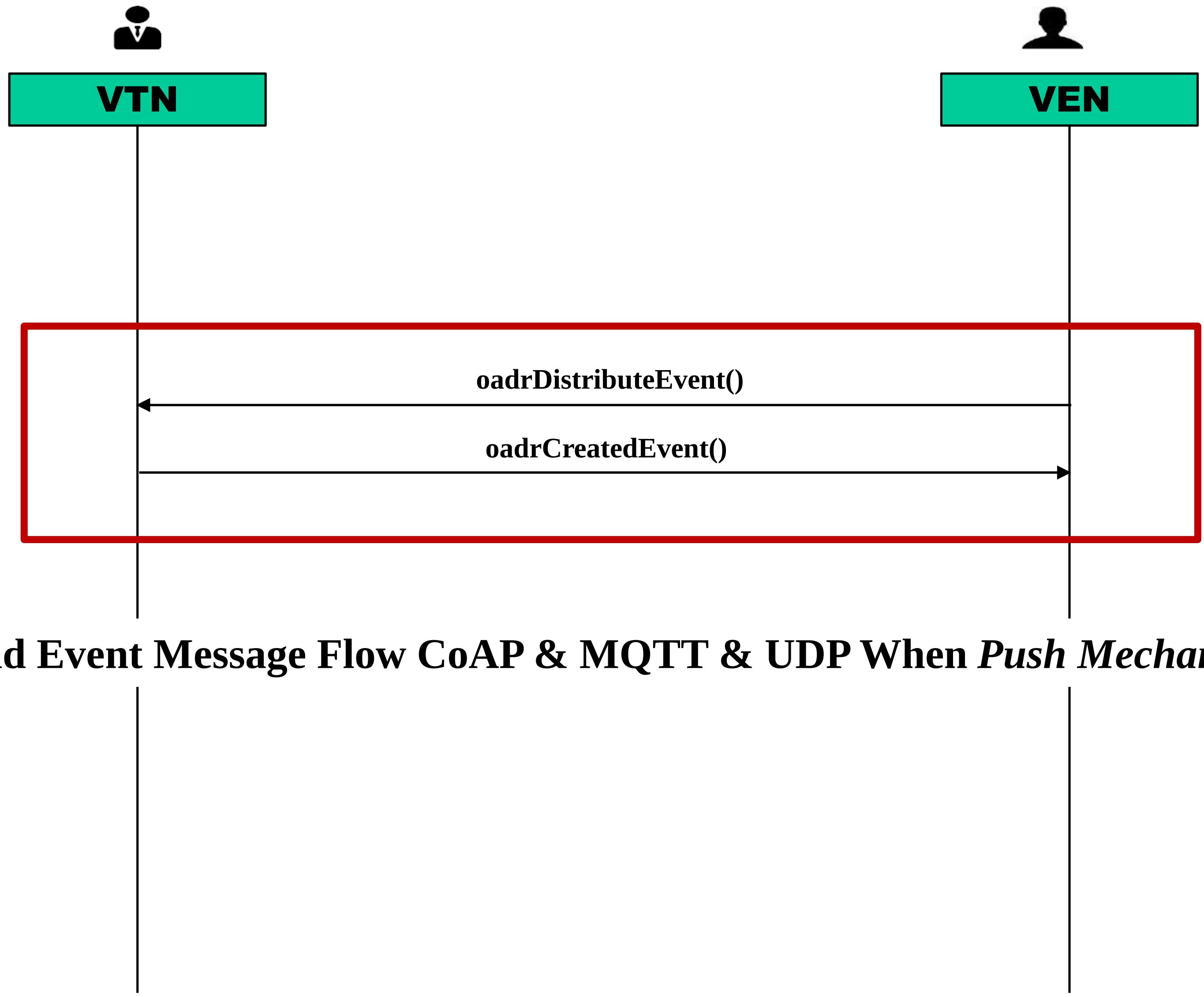

CON GET Observer: 0 Token: 0x3f
ACK 2.05 Observe: 27 Token: 0x3f
Added Observer Registration

5. Message Flow of VTN-VEN(Polling)



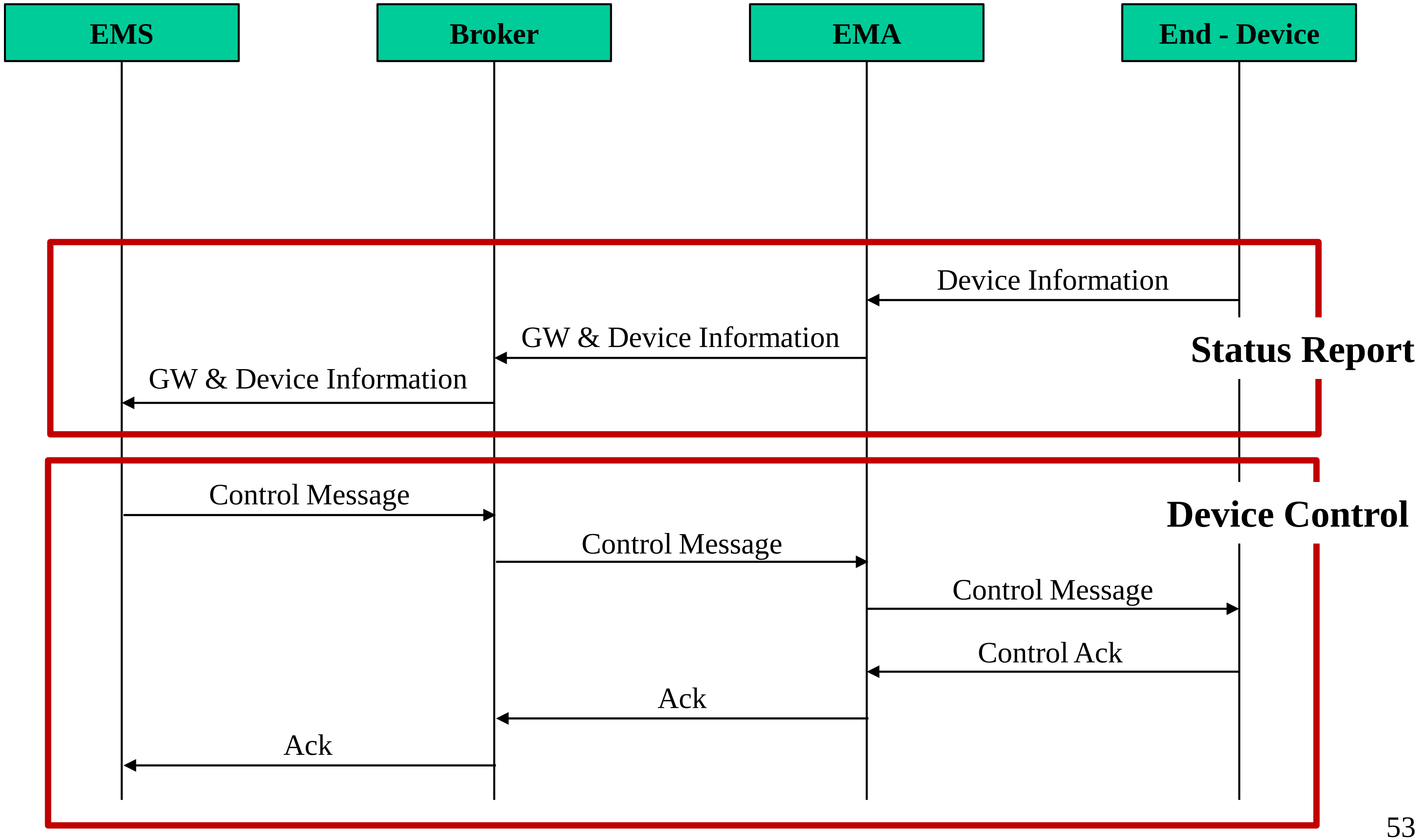
Send Event Message Flow CoAP & MQTT & UDP When *Pull Mechanism*

5. Message Flow of VTN-VEN(Push)

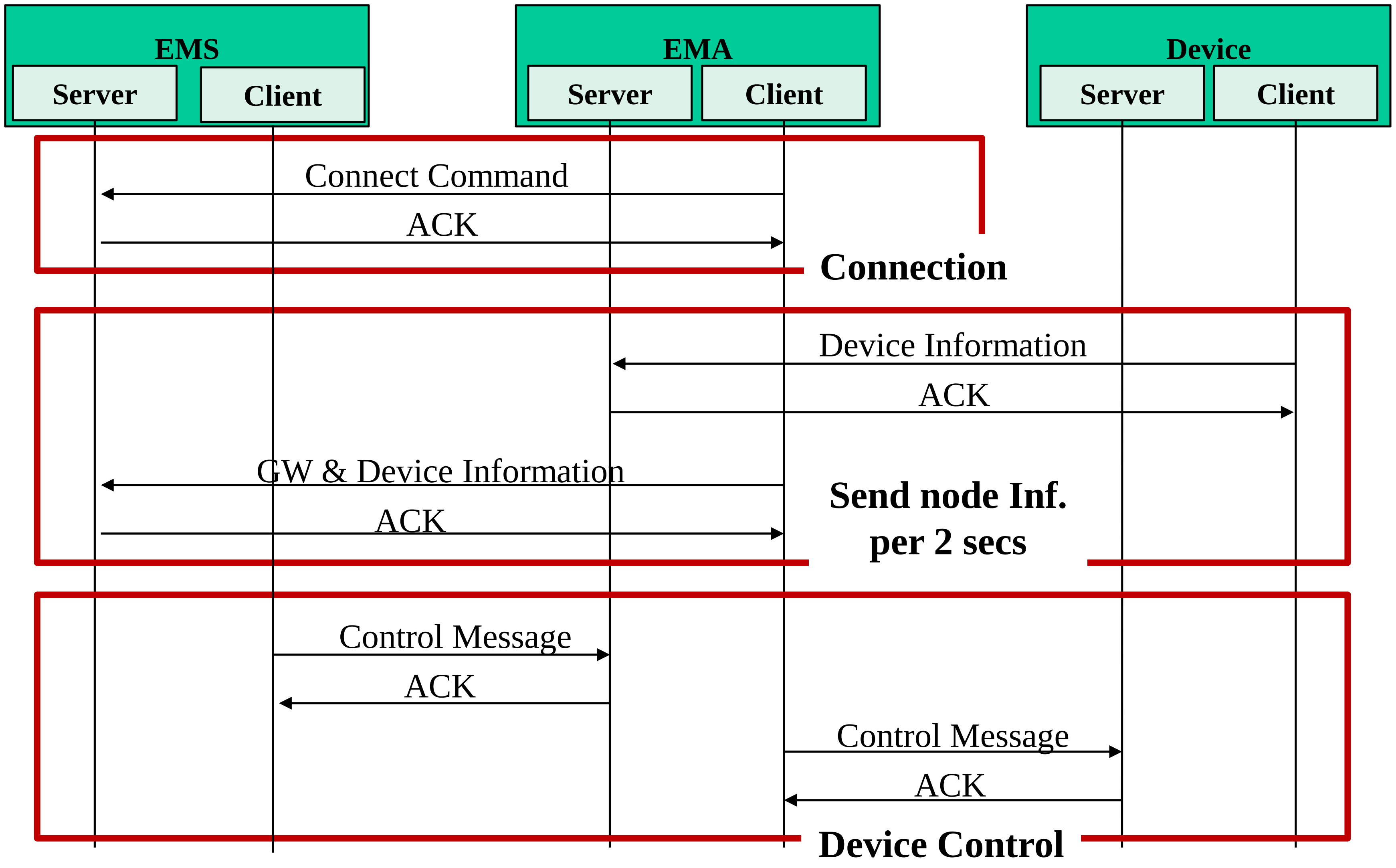


Send Event Message Flow CoAP & MQTT & UDP When *Push Mechanism*

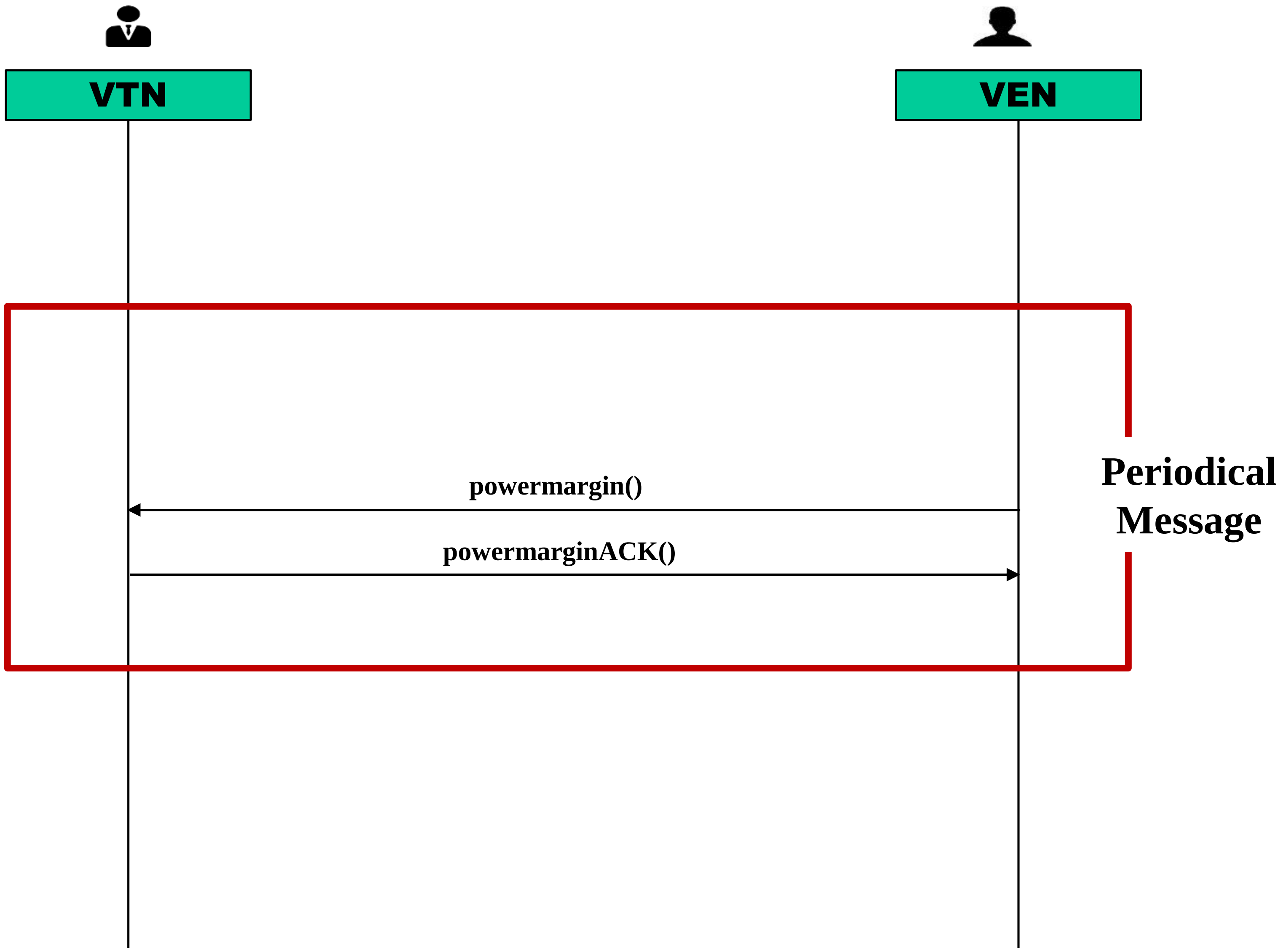
5. Message Flow of MQTT(EMS-EMA-Dev)



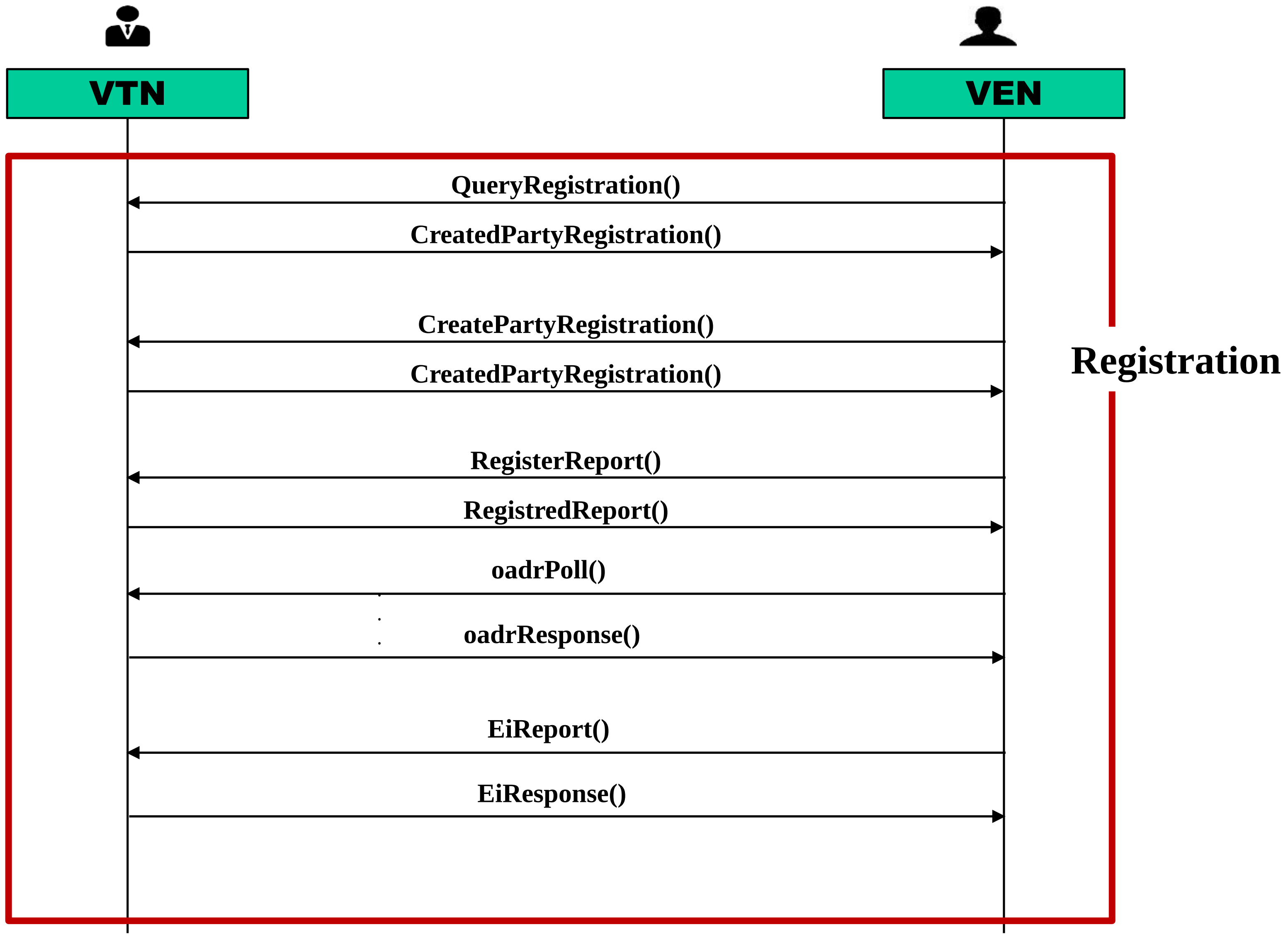
5. Message Flow of CoAP(EMS-EMA-Dev)



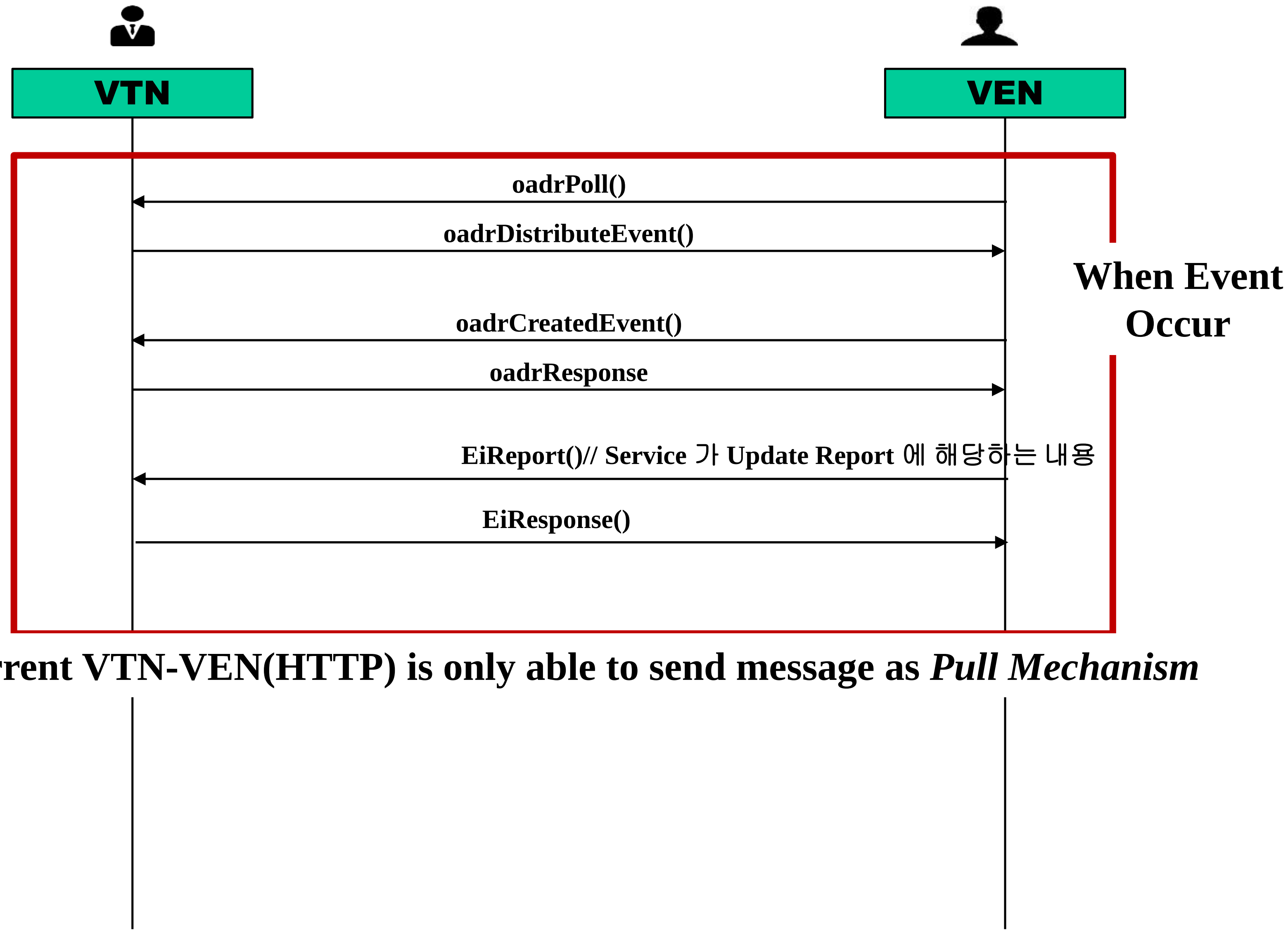
5. Message Flow of EMS-EMA



5. Message Flow of VTN-VEN(HTTP)

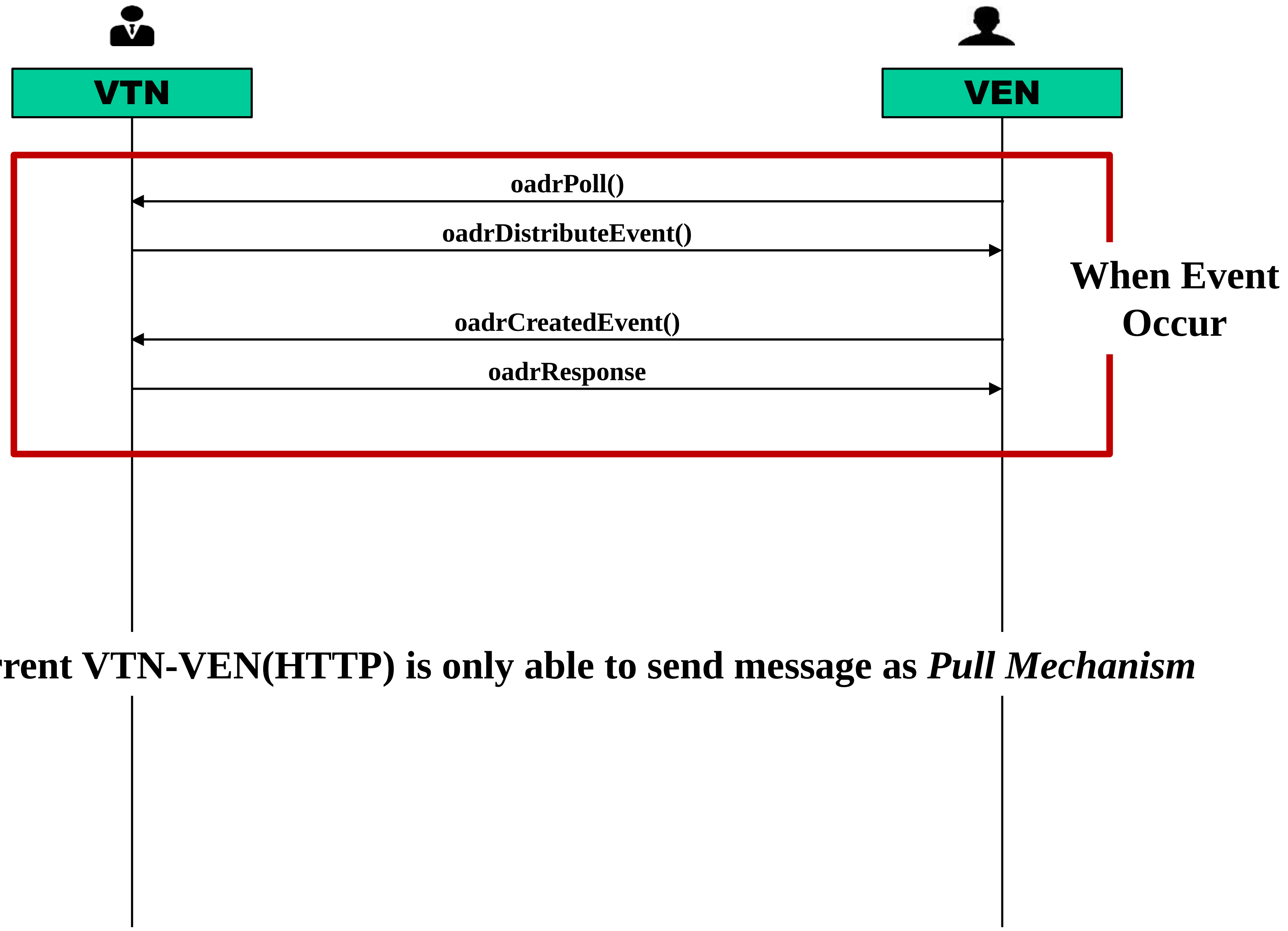


5. Message Flow of VTN-VEN(HTTP)



Current VTN-VEN(HTTP) is only able to send message as *Pull Mechanism*

5. Message Flow of VTN-VEN(HTTP)



Current VTN-VEN(HTTP) is only able to send message as *Pull Mechanism*

Appendix

- Java Thread Management
  - Smart Meter : Periodical On Demand Request

Java Thread Management

```
JButton btnNewButton = new JButton("SET");
btnNewButton.setBounds(556, 242, 62, 23);
add(btnNewButton);

btnNewButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                        if (global.onDemandCNT > 0) {
                                    Thread[] ee = new Thread[10000];
                                    Thread.enumerate(ee);

                                    for (int i = 0; i < ee.length; i++) {
                                                if (ee[i].getName().equals("onDemand")) {
                                                            ee[i].interrupt();
                                                            break;
                                                } else {

                                                }
                                    }
                        }
                        int sec = Integer.parseInt(textField.getText()) * 1000;
                        RunnableJob onDemandInterval = new RunnableJob(sec);
                        Thread onDemandRequest = new Thread(onDemandInterval);
                        onDemandRequest.setName("onDemand");
                        onDemandRequest.start();
                        global.onDemandCNT += 1;

            }
});
```

현재 실행되고 있는 Thread List를 Thread 배열에 저장한다

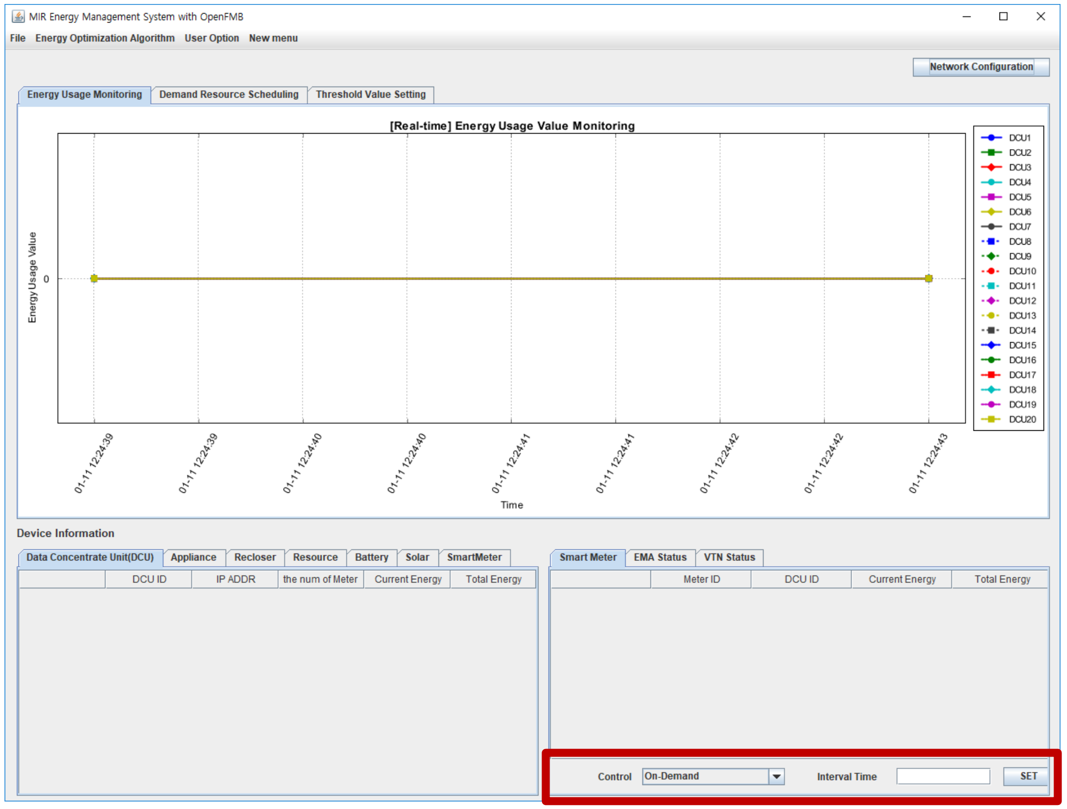
*Thread 배열에 **'onDemand'** 라는 이름인 **Thread**를 검색한다.*
*해당 **Thread**가 있을 경우 **Interrupt** 명령어를 통해 종료한다.*
*검색 시간을 줄이기 위해 **break;***
***Stop** 이라는 명령어를 쓰는 것 보다 수행하는 **Thread Class**에*
***Interrupt Option** 을 걸어 주는 것이 효율 적인 **Thread** 관리 방법*

Java Thread management

```
public class RunnableJob implements Runnable {
        public int intervalTime = 1000;
        public RunnableJob(int interval) {
                this.intervalTime = interval;
        }
        @Override
        public void run() {
                try {
                        while (!Thread.currentThread().isInterrupted()) {
                                long sTime = System.currentTimeMillis();
                                Thread.sleep(intervalTime);
                                long cTime = System.currentTimeMillis();
                                long rTime = cTime - sTime;
                                System.out.println(rTime / 1000 + "초");
                                Object[] dcuList = global.dcuHashMap.keySet().toArray();
                                for (int i = 0; i < dcuList.length; i++) {
                                        System.out.println(global.dcuHashMap.get(dcuList[i]).toString());
                                        String[] parseRemoteIp = global.dcuHashMap.get(dcuList[i].toString()).toString().split("/");
                                        String remoteIp = parseRemoteIp[0];
                                        String dcuId = parseRemoteIp[1];
                                        String meterId = parseRemoteIp[4];
                                        JSONArray meterList = global.dcuHashMap.get(dcuList[i]).getMeterInfo();

                                        for (int j = 0; j < meterList.size(); j++) {
                                                JSONArray meterIDarr = new JSONArray();
                                                meterIDarr.add(meterList.get(j));
                                                TcpClient tcpClient = new TcpClient(remoteIp, dcuId, meterIDarr, "ondemand");
                                                tcpClient.start();
                                        }
                                }
                        }
                } catch (InterruptedException e) {
                        e.printStackTrace();
                } finally {
                        System.out.println("OnDemand Thread is Dead");
                }
        }

}
```

***Thread.isInterrupt()** 이 아니라 현재 수행 중인 **Thread**를 종료하는 코드*

수정 된 사항



	DCU ID	IP ADDR	the num of Meter	Current Energy	Total Energy

	Meter ID	DCU ID	Current Energy	Total Energy

Control: On-Demand
Interval Time:

목적:
Data Traffic & Event Response 실험을 위해
On-Demand Interval Time을 설정 할 수 있다.

기존 문제점:
기존에는 'SET' 버튼을 누를 때 마다
Thread가 추가적으로 생성된다.

해결방법:
Thread Interrupt 함수를 이용하여
보다 효과적으로 Thread를 관리한다.