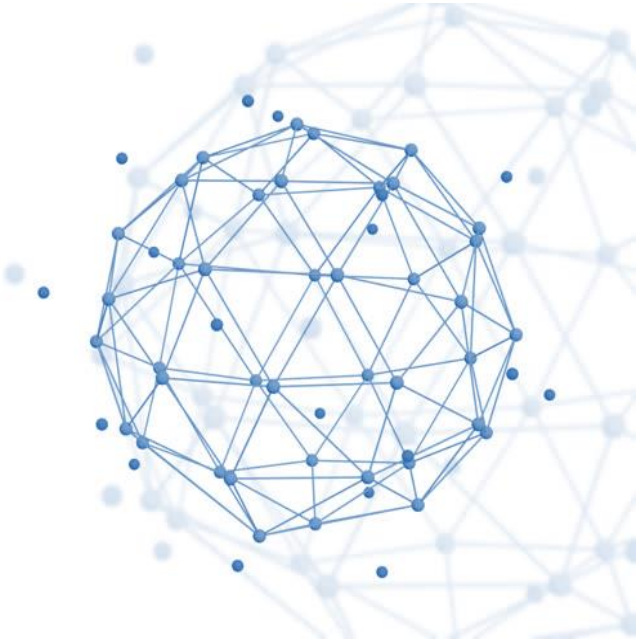
HALLYM UNIVERSITY

CoAP Programming 1

2020.03.18

Sang-woo Lee

glutton.leesw@gmail.com



CIC Convergence Information & Communications

Contents

- Open source software (OSS)
- WS4D-jCoAP (Java)
- CoAP introduction
- Import jCoAP project
- jCoAP Server
- jCoAP Resource
- jCoAP GUI client

Open source software (OSS)

- Open source software (OSS)
  - 소스 코드를 공개해 누구나 특별한 제한 없이 그 코드를 보고 사용할 수 있는 오픈 소스 라이선스를 만족하는 소프트웨어를 의미함
  - 즉, 공개적으로 접근하여 사용할 수 있게 설계되어 누구나 자유롭게 확인, 수정, 배포할 수 있는 코드로, 통상적으로 오픈 소스라 함
  - 오픈소스는 단일 작성자 또는 기업이 아닌 커뮤니티가 개발하므로 유연성 및 지속성을 특징으로 함
  - 소프트웨어의 소스 코드를 자유롭게 읽고, 재배포 및 개조를 가능하게 함으로써 소프트웨어가 향상되고, 한 사람이 느린 속도로 소프트웨어를 개발하는 것보다 여러 사람들이 고치고 쓰고 버그를 개선하는 것이 보다 빠를 수 있다는 것이 오픈 소스의 기본 이념임

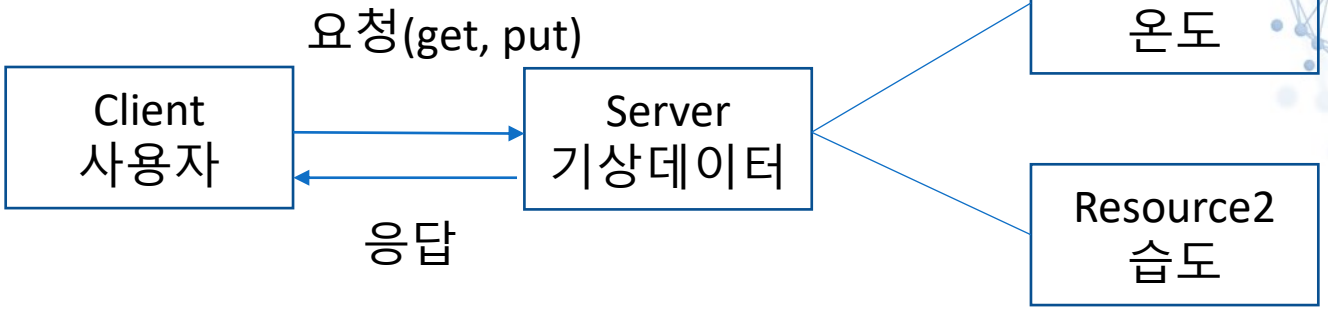
WS4D-jCoAP (Java)

- Web Services for Devices (WS4D)
  - SOA (Service-Oriented Architecture) 및 웹 서비스 기술을 산업 자동화, 홈 엔터테인먼트, 자동차 시스템 및 통신 시스템의 애플리케이션 영역으로 가져 오기 위한 계획
  - XML, HTTP 및 웹 서비스와 같은 인터넷 기술을 사용하여 ad hoc 네트워크에서 resource-constrained 디바이스를 연결하고 W3C에서 지정한 웹 서비스와의 상호 운용성 유지와 관련됨
  - WS4D를 통해 low level의 분산적인 임베디드 시스템에서도 웹 서비스에 대한 high level의 개념을 사용할 수 있음
  - 따라서 WS4D는 분산적인 임베디드 시스템에서 네트워크로 연결된 장치를 쉽게 설정하고 관리할 수 있는 기술을 제공함
- WS4D-jCoAP
  - Java를 위한 CoAP의 구현
  - CoAP은 IETF CoRE WG에 의해 정의된 유망한 프로토콜 (RFC7252)
  - CoAP은 매우 리소스 제약적인 디바이스에서 RESTful APIs를 위한 HTTP의 대안임
  - HTTP는 대개 TCP를 기반으로 하는 반면, CoAP은 가장 기초적인 신뢰성 기능을 정의하므로, UDP에서 실행 가능함
  - 이에 따라 TCP와 함께 제공되는 많은 프로토콜 오버헤드를 줄일 수 있음
  - Gitlab repository: https://gitlab.amd.e-technik.uni-rostock.de/ws4d/jcoap

CoAP introduction

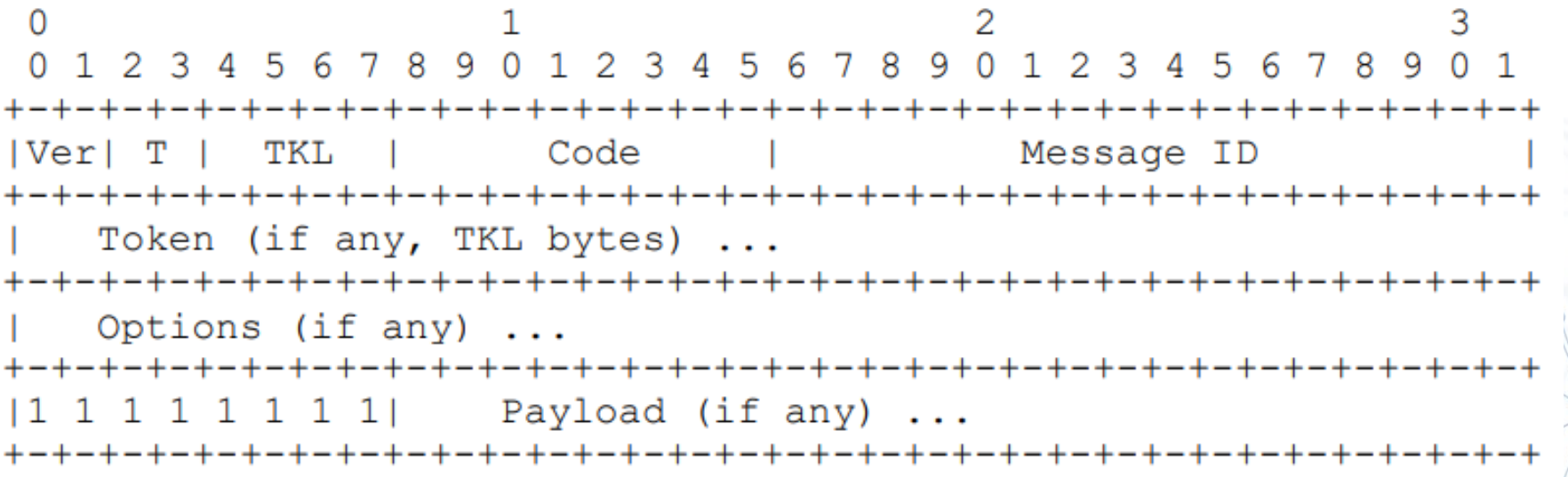
- CoAP Messages
  - Confirmable (CON): 신뢰성을 보장하는 전달을 위한 메시지
  - Non-confirmable (NON): 신뢰성을 보장하지 않는 전달을 위한 메시지
  - Acknowledgement (ACK): CON 메시지에 대한 응답을 위한 메시지
  - Reset (RST): 메시지의 처리가 불가함을 알리기 위한 메시지
- CoAP Method
  - Get: 리소스 값 읽기
  - Put: 리소스 값 쓰기 (변경)
  - Post: 새로운 리소스 생성
  - Delete: 리소스 삭제

- CoAP 서비스의 기본적인 구조



CoAP introduction

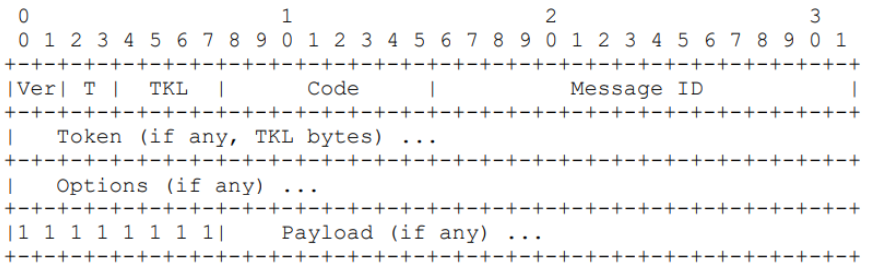
- CoAP message format
  - CoAP message는 4 bytes 고정 header를 포함함
  - Token과 option이 포함될 수 있음 (optional)
  - Payload가 존재할 경우, 데이터그램의 끝까지 배치됨



CoAP message format

CoAP introduction

- Version (Ver)
  - CoAP의 버전을 나타냄 (현재 버전은 1로 이진수 01이어야 함)
- Type (T)
  - 메시지 타입을 의미함 (0: CON, 1: NON, 2: ACK, 3: RST)
- Token Length (TKL)
  - Token field의 길이를 나타냄 (0~8)
- Code
  - 3 bits는 클래스를 의미하고, 5 bits는 상세 내용을 의미함 (Ex. c.dd)
- Message ID
  - 중복 확인 및 CON/NON 메시지에 대한 짝으로서 ACK/RST 메시지에 사용됨
- Token
  - TKL에 따라 메시지 헤더 다음에 Token 값이 존재함
- Options
  - Token에 이어 옵션을 나타냄
- Payload Marker
  - 0xFF의 값을 가지며, 더 이상의 옵션이 없음을 알 수 있음



CoAP message format

Import jCoAP project

- Import jCoAP project
  1. Smart campus에 업로드된 jCoAPExample 압축파일을 자신의 PC에 내려 받고 압축을 해제함
  2. Eclipse를 실행함
  3. File – Open project from file system
  4. Directory -> ws4d-jcoap 추가
  5. Browse… – jCoAPExample 폴더 선택 – Finish
  6. 프로젝트 우클릭 -> Build path -> Configure build path
  7. Project “ws4d-jcoap” 추가

jCoAP Server

```
public class CoAP_Server {
    private static CoAP_Server coapServer;
    private CoapResourceServer resourceServer;

    public static void main(String[] args) {
        coapServer = new CoAP_Server();
        coapServer.start();
    }

    public void start() {
        System.out.println("===Run Test Server ===");

        // create server
        if (this.resourceServer != null)    this.resourceServer.stop();
        this.resourceServer = new CoapResourceServer();


        // initialize resource
        LED led = new LED();


        // add resource to server
        this.resourceServer.createResource(led);


        // run the server
        try {
            this.resourceServer.start();
        } catch (Exception e) {
            e.printStackTrace();
        }

    }
}
```

서버 생성

Resource 초기화

서버에 리소스 추가

jCoAP Resource

```
public class LED extends BasicCoapResource{
    private String state = "off";


    private LED(String path, String value, CoapMediaType mediaType) {
        super(path, value, mediaType);
    }

    public LED() {
        this("/led", "off", CoapMediaType.text_plain);
    }

    @Override
    public synchronized CoapData get(List<String> query, List<CoapMediaType> mediaTypesAccepted) {
        return get(mediaTypesAccepted);
    }

    @Override
    public synchronized CoapData get(List<CoapMediaType> mediaTypesAccepted) {
        return new CoapData(Encoder.StringToByte(this.state), CoapMediaType.text_plain);
    }

    @Override
    public synchronized boolean setValue(byte[] value) {
        this.state = Encoder.ByteToString(value);
        return true;
    }

    @Override
    public synchronized boolean post(byte[] data, CoapMediaType type) {
        return this.setValue(data);
    }

    @Override
    public synchronized boolean put(byte[] data, CoapMediaType type) {
        return this.setValue(data);
    }

    @Override
    public synchronized String getResourceType() {
        return "LED";
    }

}
```

jCoAP client

```
btn_get.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        // TODO Auto-generated method stub
        String path = path_text.getText();
        String payload = payload_text.getText();

        CoapRequest request = clientChannel.createRequest(CoapRequestCode.GET, path, true);
        displayRequest(request);
        clientChannel.sendMessage(request);
    }
});
btn_put.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        // TODO Auto-generated method stub
        String path = path_text.getText();
        String payload = payload_text.getText();
        CoapRequest request = clientChannel.createRequest(CoapRequestCode.PUT, path, true);
        request.setPayload(new CoapData(payload, CoapMediaType.text_plain));
        displayRequest(request);
        clientChannel.sendMessage(request);
    }
});
 btn_post.addActionListener(new ActionListener() {
     @Override
     public void actionPerformed(ActionEvent e) {
         // TODO Auto-generated method stub
         String path = path_text.getText();
         String payload = payload_text.getText();
         CoapRequest request = clientChannel.createRequest(CoapRequestCode.POST, path, true);
         request.setPayload(new CoapData(payload, CoapMediaType.text_plain));
         displayRequest(request);
         clientChannel.sendMessage(request);
     }
 });
 btn_delete.addActionListener(new ActionListener() {
     @Override
     public void actionPerformed(ActionEvent e) {
         // TODO Auto-generated method stub
         String path = path_text.getText();
         String payload = payload_text.getText();
         CoapRequest request = clientChannel.createRequest(CoapRequestCode.DELETE, path, true);
         displayRequest(request);
         clientChannel.sendMessage(request);
     }
 });
```

차주 수업 내용

- CoAP Programming #2

HALLYM UNIVERSITY

Thank you

CIC Convergence Information & Communications