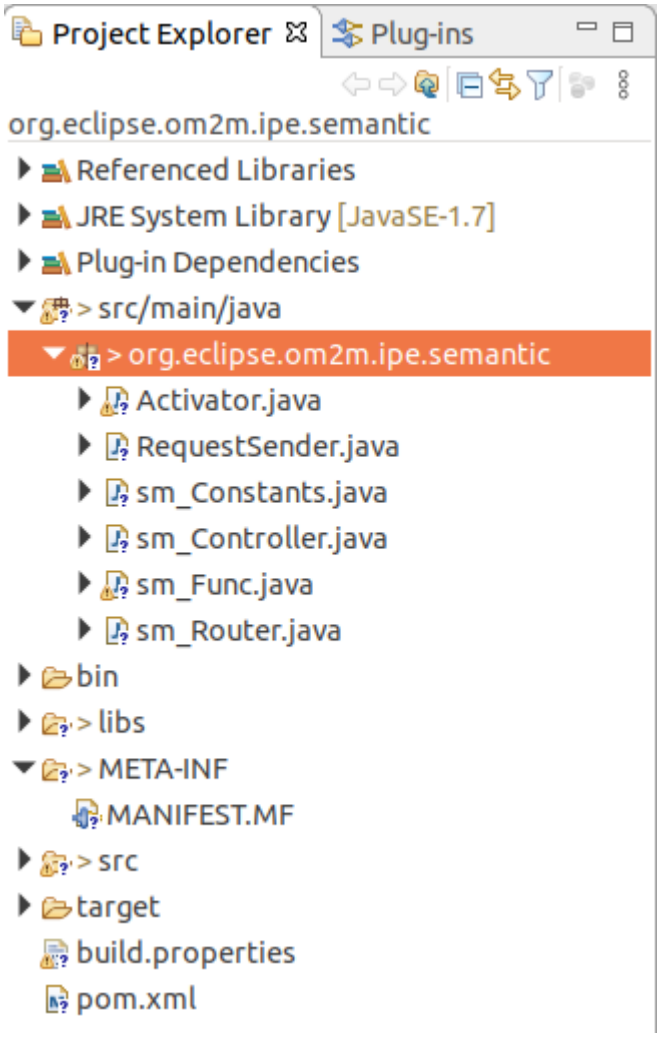
物聯網核心網路技術 **LAB6**

姓名：陳維倫　　學號：**N16094409**　系所:機械碩一

目標:在 mn-cse 建立一個 IPE，可以自動訂閱新的 CSE、sm_AE 及 sm_CNT，並支援 ttl RDF 格式 read、query、update，在使用 query 模式下，提供 POST 功能做動作。

步驟解釋:

先使用 Eclipse 建立 IPE Bundle，並建立 Packages 及 Files 如下圖



Activate.java

```
package org.eclipse.om2m.ipe.semantic;

import java.io.IOException;

import org.apache.commons.logging.Log;
import org.apache.commons.logging.LogFactory;
import org.eclipse.om2m.core.service.CseService;
import org.eclipse.om2m.interworking.service.InterworkingService;
import org.osgi.framework.BundleActivator;
import org.osgi.framework.BundleContext;
```

```
import org.osgi.framework.ServiceReference;
import org.osgi.util.tracker.ServiceTracker;

public class Activator implements BundleActivator {
    /** Logger */
    private static Log logger = LogFactory.getLog(Activator.class);
    /** SCL service tracker */
    private ServiceTracker<Object, Object> cseServiceTracker;

    public void start(BundleContext bundleContext) throws Exception {
        logger.info("Register IpeService..");

    bundleContext.registerService(InterworkingService.class.getName(),
new sm_Router(), null);
        logger.info("IpeService is registered.");
        cseServiceTracker = new ServiceTracker<Object,
Object>(bundleContext, CseService.class.getName(), null) {
            public void removedService(ServiceReference<Object>
reference, Object service) {
                logger.info("CseService removed");
            }

            public Object addingService(ServiceReference<Object>
reference) {
                logger.info("CseService discovered");
                CseService cseService = (CseService)
this.context.getService(reference);
                sm_Controller.setCse(cseService);
                // create Resource in in-cse
                sm_Func.createResources(sm_Constants.AE_NAME,
sm_Constants.POA);
                RequestSender.createSubscribe("SEMANTIC_SUB", "/in-
cse/in-name");
                return cseService;
            }
        };
        cseServiceTracker.open();
    }
```

```
    public void stop(BundleContext bundleContext) throws Exception {
        System.out.println("Stopping Sample Ipe");
    }
}
```

RequestSender.java

```
package org.eclipse.om2m.ipe.semantic;

import java.math.BigInteger;
import java.util.List;

import org.eclipse.om2m.commons.constants.Constants;
import org.eclipse.om2m.commons.constants.FilterUsage;
import org.eclipse.om2m.commons.constants.MimeMediaType;
import org.eclipse.om2m.commons.constants.NotificationContentType;
import org.eclipse.om2m.commons.constants.Operation;
```

```
import org.eclipse.om2m.commons.constants.ResourceType;
import org.eclipse.om2m.commons.constants.ResponseStatusCode;
import org.eclipse.om2m.commons.resource.AE;
import org.eclipse.om2m.commons.resource.Container;
import org.eclipse.om2m.commons.resource.FilterCriteria;
import org.eclipse.om2m.commons.resource.RequestPrimitive;
import org.eclipse.om2m.commons.resource.Resource;
import org.eclipse.om2m.commons.resource.ResponsePrimitive;
import org.eclipse.om2m.commons.resource.Subscription;
import org.eclipse.om2m.commons.resource.URIList;
import org.eclipse.om2m.datamapping.jaxb.Mapper;

public class RequestSender {
    private static Mapper mapper;

    public RequestPrimitive createRP(String targetId, Object content,
BigInteger op, String requestContentType,
            Integer rt) {
        RequestPrimitive rp = new RequestPrimitive();
        rp.setFrom(Constants.ADMIN_REQUESTING_ENTITY);
        rp.setTo(targetId);
        rp.setContent(content);
        rp.setOperation(op);
        rp.setRequestContentType(requestContentType);
        if (rt != null) {
            rp.setResourceType(rt);
        }
        return rp;
    }

    public static ResponsePrimitive createResource(String targetId,
Resource resource, int resourceType) {
        RequestPrimitive request = new RequestPrimitive();
        request.setFrom(Constants.ADMIN_REQUESTING_ENTITY);
        request.setTo(targetId);
        request.setResourceType(BigInteger.valueOf(resourceType));
        request.setRequestContentType(MimeMediaType.OBJ);
        request.setReturnContentType(MimeMediaType.XML);
```

```
        request.setContent(resource);
        request.setOperation(Operation.CREATE);
        System.out.println(request);
        return sm_Constants.CSE.doRequest(request);
    }

    public static ResponsePrimitive createAE(AE resource) {
        // TODO Auto-generated method stub
        return createResource("/" + Constants.CSE_ID, resource,
ResourceType.AE);
    }

    public static ResponsePrimitive createContainer(String targetId,
Container resource) {
        return createResource(targetId, resource,
ResourceType.CONTAINER);
    }

    public static ResponsePrimitive createSubscribe(String subName,
String targetId) {
        Subscription sub = new Subscription();
        sub.setName(subName);
        sub.getNotificationURI().add("/in-cse/in-name/" +
sm_Constants.AE_NAME);

    sub.setNotificationContentType(NotificationContentType.MODIFIED_ATTR
IBUTES);
        return createResource(targetId, sub,
ResourceType.SUBSCRIPTION);
    }

    public static List<String> Type_Discovery(String targetId, int ty) {
        FilterCriteria fc = new FilterCriteria();
        fc.setFilterUsage(FilterUsage.DISCOVERY_CRITERIA);
        fc.setResourceType(BigInteger.valueOf(ty));
        return do_Discovery(targetId, fc);
    }
```

```
    public static List<String> do_Discovery(String targetId,
FilterCriteria fc) {
        RequestPrimitive rep = new RequestPrimitive();
        rep.setFrom(Constants.ADMIN_REQUESTING_ENTITY);
        rep.setTo(targetId);
        rep.setOperation(Operation.RETRIEVE);
        ResponsePrimitive resp = sm_Constants.CSE.doRequest(rep);
        if (resp.getResponseStatusCode().equals(ResponseStatusCode.OK))
{
            return ((URIList) mapper.stringToObj((String)
resp.getContent())).getListOfUri();
        } else {
            return null;
        }
    }
}
```

Sm_Router.java

```
package org.eclipse.om2m.ipe.semantic;

import java.io.IOException;
import java.io.StringReader;
import java.io.UnsupportedEncodingException;
import java.util.Base64;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
```

```java
import org.eclipse.om2m.commons.constants.ResponseStatusCode;
import org.eclipse.om2m.commons.resource.RequestPrimitive;
import org.eclipse.om2m.commons.resource.ResponsePrimitive;
import org.eclipse.om2m.interworking.service.InterworkingService;
import org.w3c.dom.Document;
import org.xml.sax.InputSource;
import org.xml.sax.SAXException;

public class sm_Router implements InterworkingService {
    final Base64.Decoder decoder = Base64.getDecoder();
    final Base64.Encoder encoder = Base64.getEncoder();

    @Override
    public ResponsePrimitive doExecute(RequestPrimitive request) {
        // TODO Auto-generated method stub
        ResponsePrimitive response = new ResponsePrimitive(request);
        System.out.println("sm_Router1: " + request);
        DocumentBuilder builder;
        InputSource src;
        Document doc = null;
        String ty = null;
        try {
            builder =
DocumentBuilderFactory.newInstance().newDocumentBuilder();
            src = new InputSource();
            src.setCharacterStream(new
StringReader(request.getContent().toString()));

            doc = builder.parse(src);
            ty =
doc.getElementsByTagName("ty").item(0).getTextContent();

            response.setResponseStatusCode(ResponseStatusCode.OK);
        } catch (ParserConfigurationException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } catch (SAXException e) {
```

```
                // TODO Auto-generated catch block
                e.printStackTrace();
            } catch (IOException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
            if (ty.equals("16")) {
                System.out.println("sm_Router2: " + ty);
                String csi =
doc.getElementsByTagName("csi").item(0).getTextContent();
                System.out.println("sm_Router2_1: " + csi);
                RequestSender.createSubscribe("SEMANTIC_SUB_MN", csi);
            } else if (ty.equals("2")) {
                System.out.println("sm_Router3: " + ty);
                String lbl =
doc.getElementsByTagName("lbl").item(0).getTextContent();
                System.out.println("sm_Router3_1: " + lbl);
                if (lbl.contains("semantic") == true) {
                    System.out.println("semantic");
                    String ri =
doc.getElementsByTagName("ri").item(0).getTextContent();
                    RequestSender.createSubscribe("SEMANTIC_SUB_AE", ri);
                }
            } else if (ty.equals("3")) {
                System.out.println("sm_Router4: " + ty);
                String ol =
doc.getElementsByTagName("ol").item(0).getTextContent();
                System.out.println("sm_Router4_1: " + ol);
                if (ol.contains("sm_DATA") == true) {
                    System.out.println("sm_DATA");
                    String ri =
doc.getElementsByTagName("ri").item(0).getTextContent();
                    RequestSender.createSubscribe("SEMANTIC_SUB_smDATA",
ri);
                }
            } else if (ty.equals("4")) {
                System.out.println("sm_Router5: " + ty);
```

```
            String con =
doc.getElementsByTagName("con").item(0).getTextContent();
            String cnf =
doc.getElementsByTagName("cnf").item(0).getTextContent();
            try {
                System.out.println("sm_Router5_1: " + new
String(decoder.decode(con), "UTF-8"));
                switch (cnf) {
                case "read":
                    sm_Func.handleCI_read(new
String(decoder.decode(con), "UTF-8"));
                    break;
                case "query":
                    sm_Func.handleCI_query(new
String(decoder.decode(con), "UTF-8"));
                    break;
                case "update":
                    sm_Func.handleCI_update(new
String(decoder.decode(con), "UTF-8"));
                    break;
                default:
                    response.setContent("This cnf is not supported");
                    break;
                }
            } catch (UnsupportedEncodingException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        } else {
            System.out.println("sm_Router6: " + ty);
        }
        return response;
    }

    @Override
    public String getAPOCPath() {
        // TODO Auto-generated method stub
        return sm_Constants.POA;
```

```
    }
}
```

Sm_Constants.java

```
package org.eclipse.om2m.ipe.semantic;

import org.apache.jena.rdf.model.Model;
import org.apache.jena.rdf.model.ModelFactory;
import org.eclipse.om2m.core.service.CseService;

public class sm_Constants {
    public static final String POA = "ipe_semantic";
    protected static final String AE_NAME = "SEMANTIC";
    public static final String DATA = null;
    public static CseService CSE = null;
```

```
    public static Model model = ModelFactory.createDefaultModel();
}
```

Sm_Func.java

```
package org.eclipse.om2m.ipe.semantic;

import java.io.ByteArrayInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.math.BigInteger;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
```

```
import org.apache.commons.logging.Log;
import org.apache.commons.logging.LogFactory;
import org.apache.jena.query.Query;
import org.apache.jena.query.QueryExecution;
import org.apache.jena.query.QueryExecutionFactory;
import org.apache.jena.query.QueryFactory;
import org.apache.jena.query.QuerySolution;
import org.apache.jena.query.ResultSetFactory;
import org.apache.jena.query.ResultSetRewindable;
import org.apache.jena.rdf.model.Literal;
import org.apache.jena.update.UpdateAction;
import org.apache.jena.update.UpdateFactory;
import org.apache.jena.update.UpdateRequest;
import org.eclipse.om2m.commons.constants.ResponseStatusCode;
import org.eclipse.om2m.commons.resource.AE;
import org.eclipse.om2m.commons.resource.Container;
import org.eclipse.om2m.commons.resource.ResponsePrimitive;

public class sm_Func {

    private static Log LOGGER = LogFactory.getLog(sm_Func.class);

    public static void createResources(String appId, String poa) {
        // TODO Auto-generated method stub
        // create the application resource

        AE ae = new AE();
        ae.setRequestReachability(true);
        ae.getPointOfAccess().add(poa);
        ae.setAppID(appId);
        ae.setName(appId);

        ResponsePrimitive response = RequestSender.createAE(ae);
        // Create Application sub-resources only if application not yet
created
        if
(response.getResponseStatusCode().equals(ResponseStatusCode.CREATED)) {
            Container cnt = new Container();
```

```
            cnt.getLabels().add("semantic");
            cnt.setMaxNrOfInstances(BigInteger.valueOf(0));
            cnt = new Container();
            cnt.setMaxNrOfInstances(BigInteger.valueOf(10));
            // Create STATE container sub-resource
            cnt.setName(sm_Constants.DATA);

    LOGGER.info(RequestSender.createContainer(response.getLocation(),
cnt));
        }
    }

    public static void handleCI_read(String con) {
        InputStream is = new ByteArrayInputStream(con.getBytes());
        sm_Constants.model.read(is, null, "TURTLE");
        sm_Constants.model.write(System.out, "TURTLE");
    }

    public static void handleCI_query(String con) {
        Query query = QueryFactory.create(con);
        QueryExecution qexec = QueryExecutionFactory.create(query,
sm_Constants.model);
        try {
            ResultSetRewindable results =
ResultSetFactory.makeRewindable(qexec.execSelect());
            while (results.hasNext()) {
                QuerySolution soln = results.nextSolution();
                System.out.println(soln);
                Literal name = soln.getLiteral("result");
                System.out.println(name);
                try {
                    URL url = new URL(name.toString());
                    HttpURLConnection conn = (HttpURLConnection)
url.openConnection();
//                conn.setDoOutput(false);
                    conn.setRequestMethod("POST");
                    conn.setRequestProperty("X-M2M-Origin",
"admin:admin");
```

```
                    System.out.println(conn.getResponseCode());
                    conn.disconnect();
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        } finally {
            qexec.close();
        }
    }

    public static void handleCI_update(String con) {
        UpdateRequest updateRequest = UpdateFactory.create(con);
        UpdateAction.execute(updateRequest, sm_Constants.model);
        sm_Constants.model.write(System.out, "TURTLE");
    }
}
```

Sm_Controller.java

```
package org.eclipse.om2m.ipe.semantic;

import org.eclipse.om2m.core.service.CseService;

public class sm_Controller {

    public static void setCse(CseService cseService) {
        // TODO Auto-generated method stub
        System.out.println("setCSE: " + sm_Constants.CSE);
        sm_Constants.CSE = cseService;
    }
```

```
}
```

建立一個 MN-AE(名稱: **org.eclipse.om2m.semantic**)在 MN-CSE 底下，其中 sm_Router.java 及 sm_Controller.java 各別將 request 的內容處理後儲存到 MN-CSE 中且建立新的 Content Instance，其中 sm_Router.java 會根據 `ty = doc.getElementsByTagName("ty").item(0).getTextContent();` 的 ty equals 不同數值來做出不同的處理，例如 ty.equals(“4”)時會根據 cnf 為 read、query 或是 update 這三種請求來做出不同的 sm_Func.handleCI_read 、

sm_Func.handleCI_query 或是 sm_Func.handleCI_update 處理，在 sm_Func.java 中 Query 模式下(sm_Func.handleCI_query)可以進行 POST 的功能。

Jupyter notebook 程式碼:

(1) Ipynb code:

```
import requests, base64, time
OM2M_URL = "http://127.0.0.1:8282/~"
CSE_ID = "/mn-cse/"
CSE_NAME = "mn-name"
LOGIN="admin"
PSWD="admin"
OM2M_BASE = OM2M_URL+CSE_ID
auth_headers = {"X-M2M-ORIGIN":LOGIN+":"+PSWD}
# The other accepted value is application/xml
common_headers = {"Accept": "application/json"}

def create_AE(name):
    header_ae = {"Content-Type":"application/xml;ty=2"}
    name_ae = name
    body_ae = """
    <m2m:ae xmlns:m2m="http://www.onem2m.org/xml/protocols" rn="{0}">
        <api>semantic</api>
        <lbl>Type/semantic Category/temperature Location/home</lbl>
        <rr>false</rr>
    </m2m:ae>
    """.format(name_ae)
    response = requests.delete(OM2M_BASE+CSE_NAME+"/"+name_ae,
headers={**auth_headers, **common_headers})
    print(response)
    response = requests.post(OM2M_BASE, data=body_ae,
headers={**auth_headers, **common_headers, **header_ae})
```

```
    print(response)
def create_CNT(name_ae, name_cnt):
    header_cnt = {"Content-Type":"application/xml;ty=3"}
    body_cnt = """<m2m:cnt
xmlns:m2m="http://www.onem2m.org/xml/protocols"
rn="{0}"></m2m:cnt>""".format(name_cnt)
    print(body_cnt)
    response = requests.post(OM2M_BASE+CSE_NAME+"/"+name_ae,
data=body_cnt, headers={**auth_headers, **common_headers, **header_cnt})
    print(response)

def create_CIN(name_ae, name_cnt, mode, cin):
    header_cin = {"Content-Type":"application/xml;ty=4"}
    body_cin = """<m2m:cin
xmlns:m2m="http://www.onem2m.org/xml/protocols"><cnf>{0}</cnf><con>{1}</
con></m2m:cin>""".format(mode, base64.b64encode(cin).decode("utf-8"))
    print(body_cin)
    response =
requests.post(OM2M_BASE+CSE_NAME+"/"+name_ae+"/"+name_cnt,
data=body_cin, headers={**auth_headers, **common_headers, **header_cin})
     print(response)
```

(2) 建立 **AE** 及 **CONTAINER:**

```
    create_AE("sm_sensor_1")
    time.sleep(2)
    create_CNT("sm_sensor_1", "sm_DATA")
```

(3) 建立 AE 及 CONTAINER:

```
# 新增觀測值 (read)
read1 = b"""
@prefix exq: <http://example.org/ns#> .
@prefix sosa: <http://www.w3.org/ns/sosa/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

exq:obs005 a sosa:Observation ;
    sosa:hasResult 20 ;
    sosa:madeBySensor exq:thermometer ;
    sosa:observedProperty exq:Temperature ;
    sosa:resultTime "2020-06-20T11:30:12+00:00"^^xsd:dateTimeStamp .

exq:obs006 a sosa:Observation ;
    sosa:hasResult 32 ;
    sosa:madeBySensor exq:thermometer ;
    sosa:observedProperty exq:Temperature ;
    sosa:resultTime "2020-06-20T23:30:12+00:00"^^xsd:dateTimeStamp .
"""
    create_CIN("sm_sensor_1", "sm_DATA", "read", read1)
```

(4) QUERY:

obs006 溫度 >= 30 POST 指令 (將燈泡 1 打開)
obs006 溫度 <= 25 POST 指令 (將燈泡 1 關閉)

```
query1 = b"""
PREFIX sosa: <http://www.w3.org/ns/sosa/>
PREFIX exq: <http://example.org/ns#>
SELECT ?result
WHERE {
    exq:obs006 a sosa:Observation;
```

```
        sosa:hasResult ?val
    BIND (
      COALESCE(
        IF(?val >= 30, "http://127.0.0.1:8080/~/mn-cse/mn-name/LAMP_1?op=setOn&lampid=LAMP_1", 1/0),
        IF(?val <= 25, "http://127.0.0.1:8080/~/mn-cse/mn-name/LAMP_1?op=setOff&lampid=LAMP_1", 1/0),
        "ERROR"
      ) AS ?result
    )
}
"""
create_CIN("sm_sensor_1", "sm_DATA", "query", query1)
```

(5) 刪除 **obs006** 觀測值**:**

```
# 更新 (刪除) exq:obs006 觀測值
update2 = b"""
PREFIX sosa: <http://www.w3.org/ns/sosa/>
PREFIX exq: <http://example.org/ns#>

DELETE { exq:obs006 ?property ?value }
WHERE { exq:obs006 ?property ?value }
"""
create_CIN("sm_sensor_1", "sm_DATA", "update", update2)
```

(6) 新增 **obs006** 觀測值**:**

```
# 新增 exq:obs006 觀測值
read1 = b"""
@prefix exq: <http://example.org/ns#> .
@prefix sosa: <http://www.w3.org/ns/sosa/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

exq:obs006 a sosa:Observation ;
    sosa:hasResult 23 ;
    sosa:madeBySensor exq:thermometer ;
    sosa:observedProperty exq:Temperature ;
    sosa:resultTime "2020-06-20T23:30:12+00:00"^^xsd:dateTimeStamp .
```

```
"""
create_CIN("sm_sensor_1", "sm_DATA", "read", read1)
```

(7) 再做一次 **QUERY:**

```
# query
query1 = b"""
PREFIX sosa: <http://www.w3.org/ns/sosa/>
PREFIX exq: <http://example.org/ns#>
SELECT ?result
WHERE {
    exq:obs006 a sosa:Observation;
        sosa:hasResult ?val
    BIND (
      COALESCE(
        IF(?val >= 30, "http://127.0.0.1:8080/~/mn-cse/mn-name/LAMP_1?op=setOn&lampid=LAMP_1", 1/0),
        IF(?val <= 25, "http://127.0.0.1:8080/~/mn-cse/mn-name/LAMP_1?op=setOff&lampid=LAMP_1", 1/0),
        "ERROR"
      ) AS ?result
    )
}
"""
   create_CIN("sm_sensor_1", "sm_DATA", "query", query1)
```