OM2M 오픈 소스 설치 가이드

2014년 10월

경북대학교 통신프로토콜연구실

강형우 (hwkang0621@gmail.com)

요 약

최근 사물 인터넷 (Internet of Things - IoT)이 주요 이슈가 되고 있다. 기존 인간 중심의 통신 패러다임에서 사물이 통신의 주체로 참여하는 IoT에 대한 시대가 도래될 것으로 전망되는 지금 전 세계적으로 다양한 오픈 플랫폼을 통하여 IoT 서비스들을 제공하기 위한 노력이 계속되고 있다. 본 문서에서는 오픈 플랫폼 중 eclipse에서 구현한 OM2M 오픈 소스를 리눅스 환경에서 설치 및 실행한 방법에 대하여 설명하도록 하겠다.

목 차

1. 서론

최근 많은 곳에서 사물인터넷에 대한 이야기를 들을 수 있다. 기존의 인간 중심의 통신 패러다임에서 사물이 통신의 주체가 되는 사물인터넷에 대한 관심이 많아지면서 이에 대한 전 세계적인 투자 및 연구가 활발하게 진행되고 있다. Google의 안경 및 삼성의 갤럭시 기어 등 현재 시판되고 있는 제품도 있으며, 국내에서도 SK Telecom, KT, LG U+ 등의 국내 통신사들도 사물인터넷에 대한 투자를 늘리고 있다.

본 문서에서는 사물인터넷 서비스를 제공하기 위해 개발 중인 많은 Open Source Platform 중에서 오픈소스 기반의 통합 개발환경인 이클립스(Eclipse)로 많이 알려진 이클립스 재단에서 개발 중인 OM2M을 Ubuntu 14.04 기반의 리눅스 환경에서 설치하고 실행하는 방법에 대하여 설명하겠다.

2. OM2M

LAAS-CNRS에 의해 시작된 OM2M 프로젝트는 Eclipse Technology Project에서 제안된 Open Source Project이다. OM2M은 M2M (Machine to Machine) 리소스를 생성하고 관리할 수 있는 RESTful API를 제공하며, Plugin을 통해 확장하게 만들어진 OSGi (Open Service Gateway initiative) layer의 상부에서 실행되는 모듈 식 구조를 제안한다. 또한 HTTP나 CoAP과 같은 multiple protocol binding을 지원한다. OM2M 아키텍처는 ETSI M2M 표준을 따른다. OM2M은 그림 1과 같은 요소들로 구성되어 있다. 자세한 내용은 OM2M project 웹 사이트에서 확인할 수 있다.

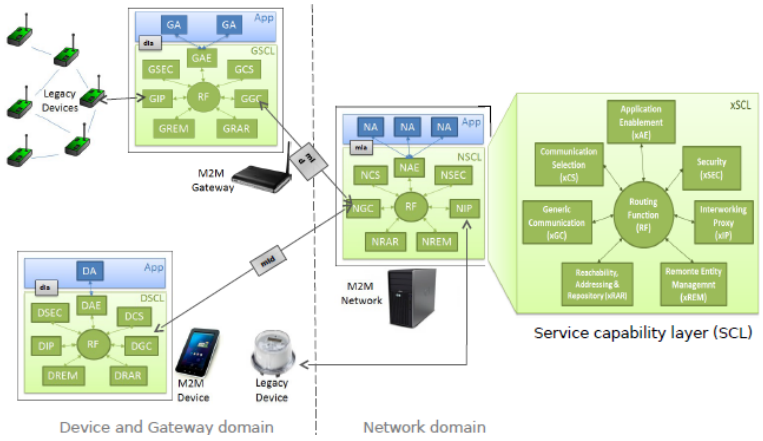

Figure 1 OM2M architecture

2.1 OM2M Download

OM2M 플랫폼을 실행하기 위해서 우리는 eclipse.org/om2m 사이트에 접속하여 OM2M 파일을 다운 받을 수 있다. 해당 사이트에서 OM2M 플랫폼을 다운 받고 실행하기 위해서 두 가지 방법을 설명하고 있다. 바로 OM2M binary 파일을 사용하여 실행하는 방법과 source code를 빌드하여 사용하는 방법이다.

2.1.1 Get OM2M binaries

첫 번째 방법으로, OM2M binary (웹 사이트에서의 OM2M.rar)를 다운 받아서 실행하는 방법이다. 해당 파일을 실행하기 위해서 우선 해당 파일을 다운 받고, 압축을 해제한다. 그 후 NSCL과 GSCL products를 찾을 수 있을 것이다. NSCL과 GSCL을 실행하기 위해서 windows에서는 start.bat, linux에서는 start.sh을 실행하면 된다. OM2M을 실행하기 위해서는 JAVA 1.7의 설치가 필요하다. JAVA 1.7 설치는 내용을 쉽게 검색할 수 있으므로, 생략하도록 하겠다.

2.1.2 Build OM2M from source code

두 번째 방식으로는, source 코드로 OM2M을 빌드하여 사용할 수 있다. OM2M 웹 사이트에서 제공하는 source code의 repository는 다음과 같다.

- http://git.eclipse.org/gitroot/om2m/org.eclipse.om2m.git.

OM2M source code를 사용하여 빌드하기 위해서 우선 OM2M에서 제공하는 git을 활용하여 repository에 저장된 source code를 가져온다. (다음의 command를 입력하여 진행)

- git clone http://git.eclipse.org/gitroot/om2m/org.eclipse.om2m.git

그 다음 과정으로는 library들을 추가해야 한다. OM2M 웹 사이트에서 om2m-libs.rar파일을 다운 받은 후 다음과 같은 과정을 수행한다.

- “db4o-core-java5-8.1-SNAPSHOT.jar” library를 “org.eclipse.om2m.core\libs” 폴더에 복사한다.
- “obix.jar” library파일은 “org.eclipse.om2m.commons\libs” 폴더에 복사한다.
- “xsd”폴더를 “org.eclipse.om2m.commons\src\main\resources” 폴더에 복사한다.

마지막으로 OM2M을 빌드하기 위한 Apache Maven 3를 설치한다. Maven 3의 설치과정은 다음과 같다.

1. Remove old version
   - sudo apt-get remove maven2

2. Add following lines to sources.list
   - sudo add-apt-repository “deb http://ppa.launchpad.net/natecarlson/maven3/ubuntu precise main”
3. Update Repository and Install
   - sudo apt-get update
   - sudo apt-get install maven3
4. Add Symbolic Link
   - sudo ln –s /usr/share/maven3/bin/mvn /usr/bin/mvn
5. Remove PPA from sources.list
   - Remove Repository via Software Center and Update

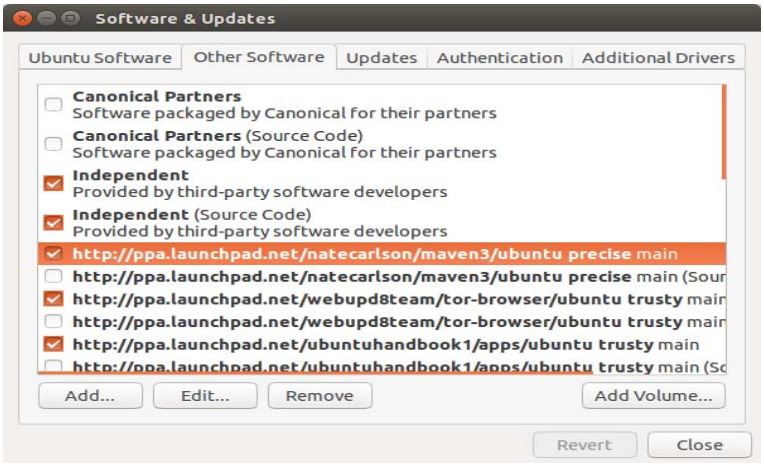

Figure 2. Software & Updates에서 ppa를 제거

   - sudo apt-get update

Maven3의 설치가 끝나면 org.eclipse.om2m 폴더로 이동한다. 그리고 다음의 command를 실행하여 OM2M을 빌드한다.

- mvn clean install

빌드가 성공적으로 실행되었을 시, 다음과 같은 products가 생성됐을 것이다.

“om2m/org.eclipse.om2m/org.eclipse.om2m.site.nscl/target/products/nscl/<os>/<ws>/<arch>” 폴더 안에 NSCL product가 생성되었을 것이다.

“om2m/org.eclipse.om2m/org.eclipse.om2m.site.nscl/target/products/gscl/<os>/<ws>/<arch>” 폴더 안에 GSCL product가 생성되었을 것이다.

2.2 OM2M Configuration

2.2.1 NSCL configure

NSCL product가 있는 폴더로 이동하여 configuration/config.ini 파일을 수정하여 NSCL에 대한 환경설정을 한다. parameter에 대한 정보는 다음과 같다. (OM2M 웹 페이지에서 확인 할 수 있다.)

Parameter	Description	Example
org.eclipse.om2m.sclType	SCL type	NSCL
org.eclipse.om2m.sclBaseAddress	NSCL ip address	127.0.0.1
org.eclipse.equinox.http.jetty.http.port	NSCL listening port	8080
org.eclipse.om2m.sclBaseContext	NSCL listening context	/om2m
org.eclipse.om2m.dbFile	NSCL Embedded database file name	nscl.db4o
org.eclipse.om2m.sclBaseId	Network SclBase resource id	nscl
org.eclipse.om2m.reset	Reset NSCL database after each restart	true
org.eclipse.om2m.sclBaseProtocol.default	NSCL default communication protocol	http
org.eclipse.om2m.maxNrOfInstances	Maximum number of instances stored in a ContentInstances resource	100
org.eclipse.om2m.adminRequestingEntity	Default NSCL admin requesting entity. (username / password)	admin:admin
org.eclipse.om2m.guestRequestingEntity	Default NSCL guest requesting entity. (username / password)	guest:guest

Figure 3. NSCL configuration

2.2.2 GSCL configure

GSCL product가 있는 폴더로 이동하여 configuration/config.ini 파일을 수정하여 NSCL에 대한 환경설정을 한다. parameter에 대한 정보는 다음과 같다.

Parameter	Description	Example
org.eclipse.om2m.sclType	SCL type	GSCL
org.eclipse.om2m.sclBaseAddress	GSCL ip address	127.0.0.1
org.eclipse.equinox.http.jetty.http.port	GSCL listening port	8181
org.eclipse.om2m.sclBaseContext	GSCL listening context	/om2m
org.eclipse.om2m.dbFile	GSCL Embedded database file name	gscl.db4o
org.eclipse.om2m.sclBaseId	Gateway SclBase resource id	gscl
org.eclipse.om2m.reset	Reset GSCL database after each restart	true
org.eclipse.om2m.sclBaseProtocol.default	GSCL default communication protocol	http
org.eclipse.om2m.maxNrOfInstances	Maximum number of instances stored in a ContentInstances resource	100
org.eclipse.om2m.adminRequestingEntity	Default GSCL admin requesting entity. (username / password)	admin:admin
org.eclipse.om2m.guestRequestingEntity	Default GSCL guest requesting entity. (username / password)	guest:guest

Figure 4. GSCL configuration

GSCL configuration 파일은 GSCL이 인증되어야 하는 remote NSCL을 명시하기 위한 4개의 추가적인 parameter를 포함하고 있다. 해당 parameter의 정보는 다음과 같다.

Parameter	Description	Example
org.eclipse.om2m.remoteNsclId	Remote Network SCL Id	nscl
org.eclipse.om2m.remoteNsclAddress	Remote Network SCL ip address	127.0.0.1
org.eclipse.om2m.remoteNsclPort	Remote Network SCL listening port	8080
org.eclipse.om2m.remoteNsclContext	Remote Network SCL listening context	/om2m

Figure 5. Additional parameters in GSCL

환경설정을 통해서 다른 machine에서 GSCL과 NSCL을 동작시킬 수 있으며, 또한 다른 identifier를 사용하여 여러 개의 GSCL을 설정할 수 있다.

2.3 OM2M Starting

2.3.1 NSCL startup

앞서 언급했듯이, OM2M을 실행하기 위해서는 Java 1.7이 필요하다. NSCL을 실행하기 위해서 NSCL product가 있는 폴더로 이동한다. 그 후 다음과 같이 JVM을 호출하여 NSCL을 바로 시작할 수 있다.

- java –jar –ea –Declipse.ignoreApp=true –Dosgi.clean=true –Ddebug=true plugins/org.eclipse.equinox.launcher_1.3.0.v20140415-2008.jar –console –noExit

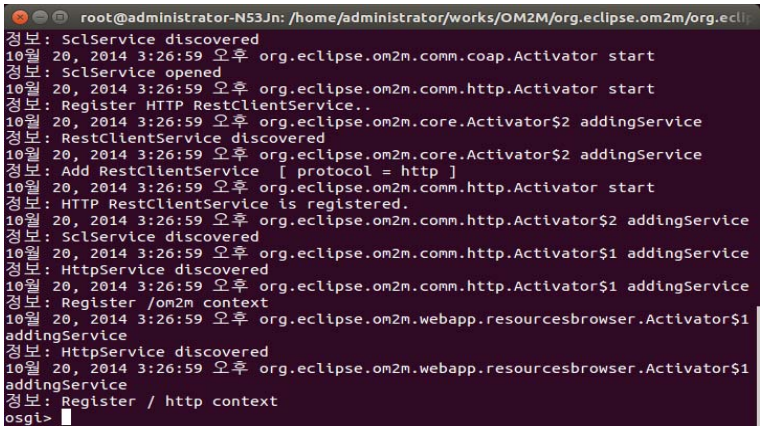

Figure 6. NSCL 실행

NSCL이 실행되면, console에서 OSGi를 볼 수 있을 것이다. “ss”를 입력하면 모든 설치된 번들에 대한 상태를 보여주게 된다. Exit를 입력하면 NSCL이 종료된다.

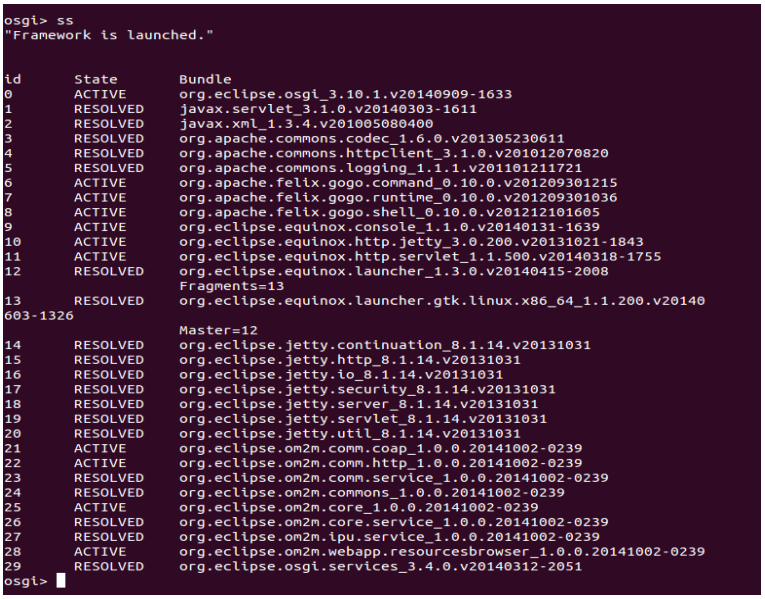

Figure 7. NSCL OSGi console

또한 그림 8와 같이 127.0.0.1:8080으로 접속하여 NSCL web interface에 연결할 수 있다. Web interface에는 User name과 password에는 “admin”을 입력하여 접속 할 수 있다.

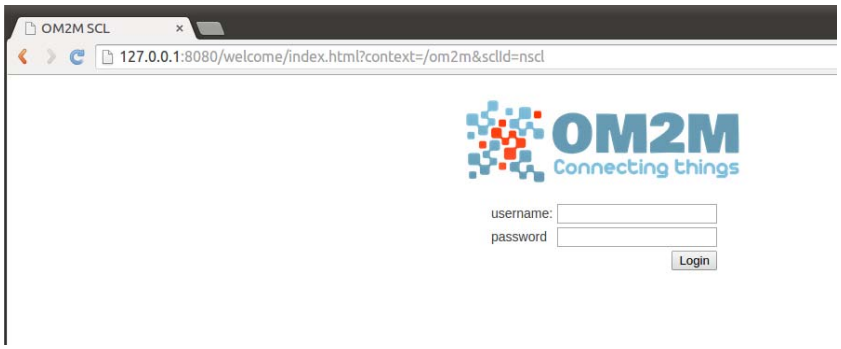

Figure 8. NSCL web interface: authentication

정상적인 인증을 거친 후에 그림 9와 같이 NSCL web interface에 접속할 수 있다. Web interface에서는 NSCL Resource Tree를 확인할 수 있다.

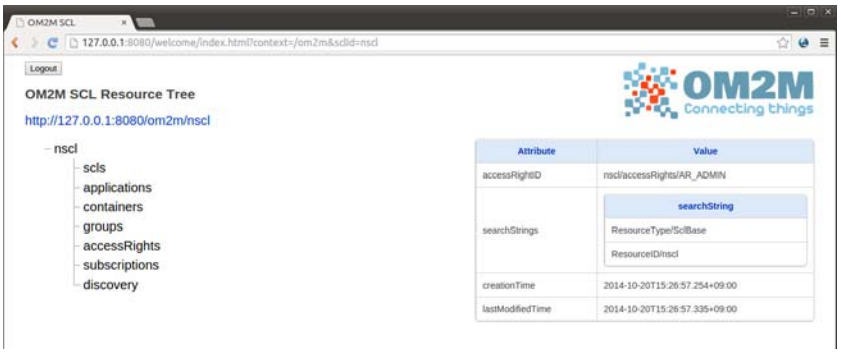

Figure 9. NSCL web interface: NSCL sclBase resource

2.3.2 GSCL startup

GSCL 역시 NSCL처럼 GSCL product가 있는 폴더로 이동하여 NSCL과 같은 command를 사용하여 실행한다. GSCL을 실행하면 그림 10과 같은 화면을 볼 수 있다.

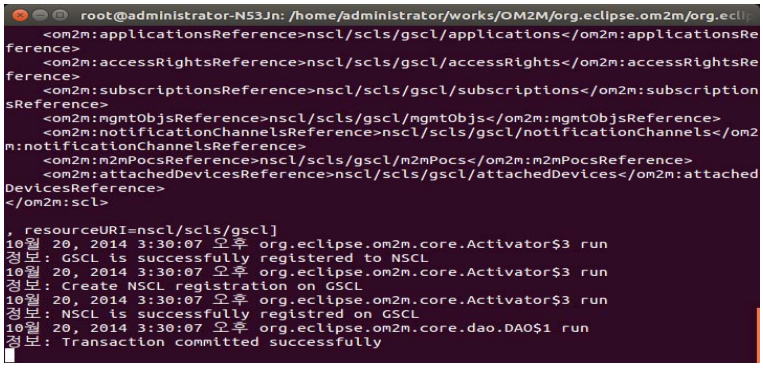

Figure 10. GSCL 실행

GSCL 역시 NSCL과 마찬가지로 “ss”를 입력하여 설치된 번들의 상태를 볼 수 있다. GSCL은 gateway configuration 파일에 명시된 remote NSCL과 자동적으로 인증을 하게 된다. 만약 NSCL이 실행되지 않은 상태라면, GSCL은 계속 인증 요청 메시지를 보내게 된다 (요청 메시지를 각 10초의 주기를 갖는다).

성공적으로 인증과정을 거친 후, gscl resource가 nscl/scls의 collection에 추가되게 되고, 또한 nscl recource가 gscl/scls의 collection에 추가되게 된다. NSCL resource를 이용하여 GSCL resource에 접속하기 위해서는 NSCL web interface의 nscl/scls/gsci uri로 접속할 수 있다 (그림 11 참조).

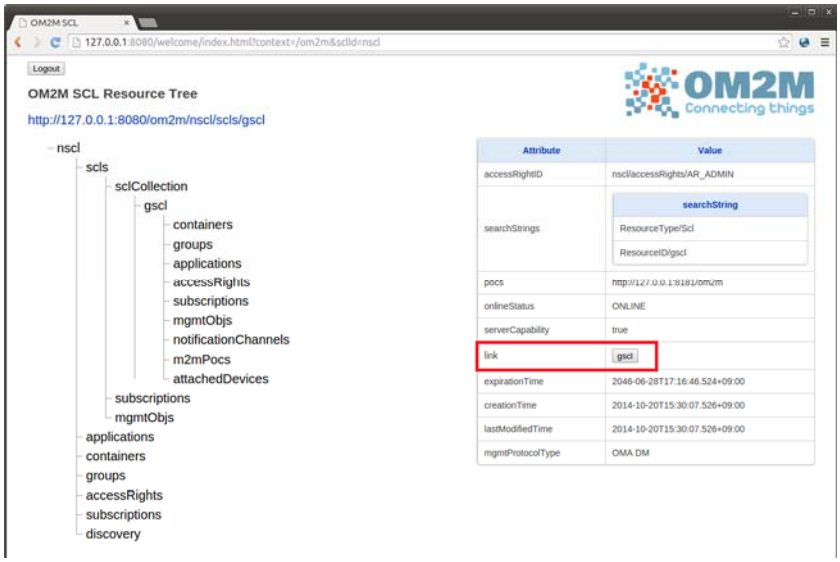

Figure 11. NSCL web interface: remote gscl resource

그림 12는 GSCL의 web interface를 보여준다. 그림 13에서는 GSCL의 resource tree를 확인할 수 있다.

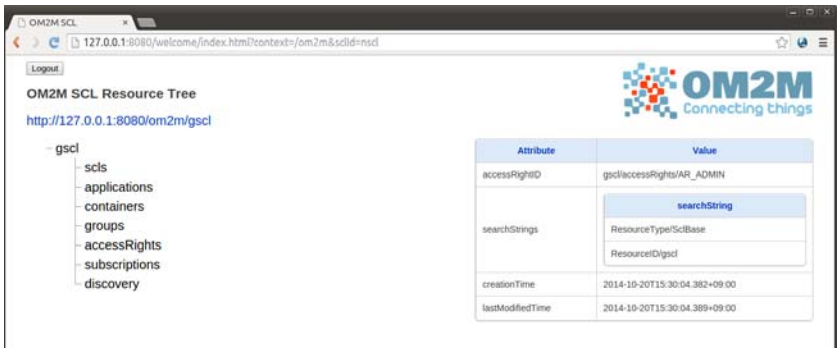

Figure 12.NSCL web interface: gscl sclBase resource

2.4 OM2M Web Interface

2.4.1 Start the sample plugin

GSCL OSGi console에서 “ss”를 입력하여 그림 13과 같이 plugin들을 확인할 수 있다. 그림 13에서 “org.eclipse.om2m.ipu.sample” plugin (id = 26 in figure 13)을 확인할 수 있다.

```
ss
"Framework is launched."


id      State       Bundle
0       ACTIVE      org.eclipse.osgi_3.10.1.v20140909-1633
1       RESOLVED    javax.servlet_3.1.0.v20140303-1611
2       RESOLVED    org.apache.commons.codec_1.6.0.v201305230611
3       RESOLVED    org.apache.commons.httpclient_3.1.0.v201012070820
4       RESOLVED    org.apache.commons.logging_1.1.1.v20110211721
5       ACTIVE      org.apache.felix.gogo.command_0.10.0.v201209301215
6       ACTIVE      org.apache.felix.gogo.runtime_0.10.0.v201209301036
7       ACTIVE      org.apache.felix.gogo.shell_0.10.0.v201212101605
8       ACTIVE      org.eclipse.equinox.console_1.1.0.v20140131-1639
9       ACTIVE      org.eclipse.equinox.http.jetty_3.0.200.v20131021-1843
10      ACTIVE      org.eclipse.equinox.http.servlet_1.1.500.v20140318-1755
11      RESOLVED    org.eclipse.equinox.launcher_1.3.0.v20140415-2008
                    Fragments=12
12      RESOLVED    org.eclipse.equinox.launcher.gtk.linux.x86_64_1.1.200.v20140603-1326
                    Master=11
13      RESOLVED    org.eclipse.jetty.continuation_8.1.14.v20131031
14      RESOLVED    org.eclipse.jetty.http_8.1.14.v20131031
15      RESOLVED    org.eclipse.jetty.io_8.1.14.v20131031
16      RESOLVED    org.eclipse.jetty.security_8.1.14.v20131031
17      RESOLVED    org.eclipse.jetty.server_8.1.14.v20131031
18      RESOLVED    org.eclipse.jetty.servlet_8.1.14.v20131031
19      RESOLVED    org.eclipse.jetty.util_8.1.14.v20131031
20      ACTIVE      org.eclipse.om2m.comm.coap_1.0.0.20141002-0239
21      ACTIVE      org.eclipse.om2m.comm.http_1.0.0.20141002-0239
22      RESOLVED    org.eclipse.om2m.comm.service_1.0.0.20141002-0239
23      RESOLVED    org.eclipse.om2m.commons_1.0.0.20141002-0239
24      ACTIVE      org.eclipse.om2m.core_1.0.0.20141002-0239
25      RESOLVED    org.eclipse.om2m.core.service_1.0.0.20141002-0239
26      RESOLVED    org.eclipse.om2m.ipu.sample_1.0.0.20141002-0239
27      RESOLVED    org.eclipse.om2m.ipu.service_1.0.0.20141002-0239
28      ACTIVE      org.eclipse.om2m.webapp.resourcesbrowser_1.0.0.20141002-0239
29      RESOLVED    org.eclipse.osgi.services_3.4.0.v20140312-2051
osgi>
```

Figure 13. OSGi consolem: Start the lamps sample IPU

그림 14와 같이 “start 26” command를 실행하여 sample plugin을 실행한다. 그림 15는 lamps sample을 보여준다.

```
osgi> start 26
10월 20, 2014 3:39:19 오후 org.eclipse.om2m.ipu.sample.Activator start
정보: Register IpuService..
10월 20, 2014 3:39:19 오후 org.eclipse.om2m.core.Activator$1 addingService
정보: IpuService discovered
10월 20, 2014 3:39:19 오후 org.eclipse.om2m.core.Activator$1 addingService
정보: Add IPU [ path = lamps ]
10월 20, 2014 3:39:19 오후 org.eclipse.om2m.ipu.sample.Activator start
정보: IpuService is registered.
10월 20, 2014 3:39:19 오후 org.eclipse.om2m.ipu.sample.Activator$1 addingService
정보: SclService discovered
osgi> 10월 20, 2014 3:39:19 오후 org.eclipse.om2m.ipu.sample.SampleMonitor start
정보: Lamps waiting for attachement..
10월 20, 2014 3:39:19 오후 org.eclipse.om2m.core.router.Router doRequest
정보: RequestIndication [method=CREATE, base=null, targetID=gscl/applications, representation=<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<om2m:application xmlns:om2m="http://uri.etsi.org/m2m" xmlns:xmime="http://www.w3.org/2005/05/xmlmime" appId="LAMP_0">
    <om2m:aPoCPaths>
        <om2m:aPoCPath>
            <om2m:path>lamps</om2m:path>
        </om2m:aPoCPath>
    </om2m:aPoCPaths>
</om2m:application>
, requestingEntity=admin:admin, protocol=null]
10월 20, 2014 3:39:19 오후 org.eclipse.om2m.core.router.Router doRequest
정보: ResourceController [ApplicationController]
```

Figure 14. Sample Plugin Execution

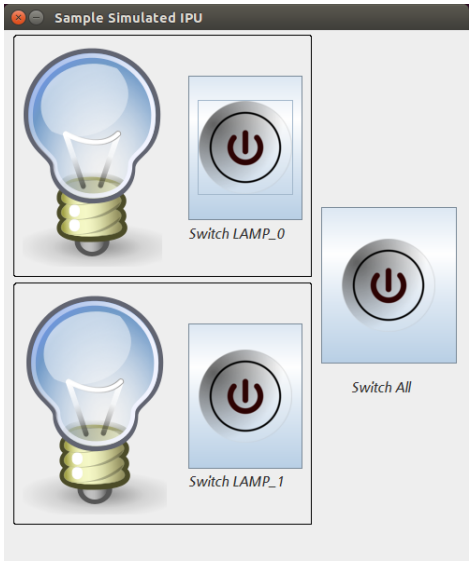

Figure 15. GUI: Lamps Sample

Sample Plugin에서 Switch 버튼을 클릭하여 전구의 불을 키고 끌 수 있다.

2.4.2 Application resources

NSCL web interface의 GSCL resource tree의 applications resource를 클릭하면 그림 16과 같이 등록된 application을 확인할 수 있다. LAMP_0, LAMP_1은 lamp 0, 1을 컨트롤 할 수 있는 application resource이다. LAMP_ALL은 lamp 0, 1을 동시에 컨트롤 할 수 있는 application resource이다.

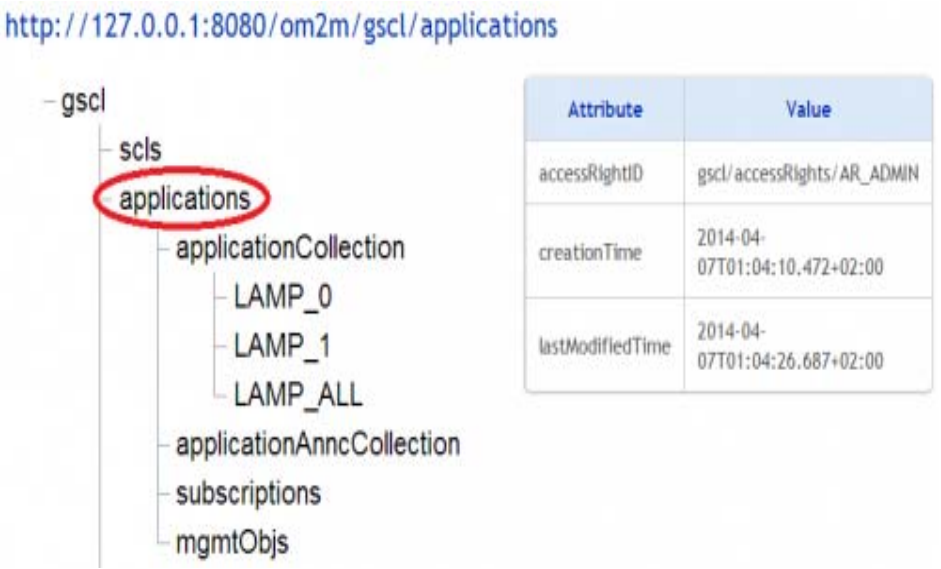

Figure 16. Web interface: The "applications" resource

2.4.3 Container resources

LAMP_0을 클릭하고 container를 클릭하면 그림 17과 같은 화면을 볼 수 있다. DESCRIPTOR는 lamp description을 저장한다. Data는 lamp의 data를 저장한다.

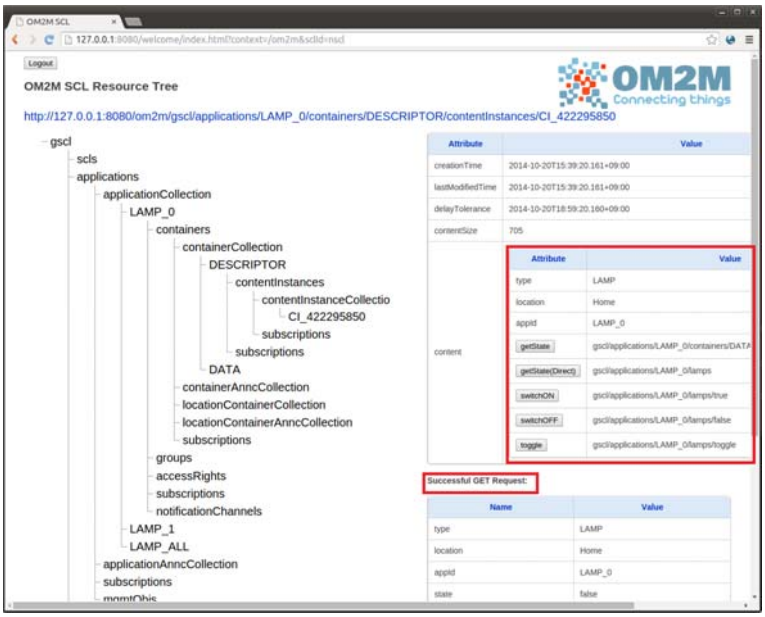

Figure 17. Web interface: The LAMP_0 description resource

getState를 클릭하면 가장 최근 lamp 상태를 GSCL database에서 읽어온다. getState(Direct) 버튼은 현재 lamp의 상태를 직접 읽어온다. 또한 switchON, switchOFF, toggle버튼 클릭을 통해서 lamp의 불을 키고 끌 수 있다.

2.4.4 Remote control lamps

GUI 또는 LSCL의 web interface를 통해서 그림 18과 같이 lamp의 상태를 바꿀 수 있다.

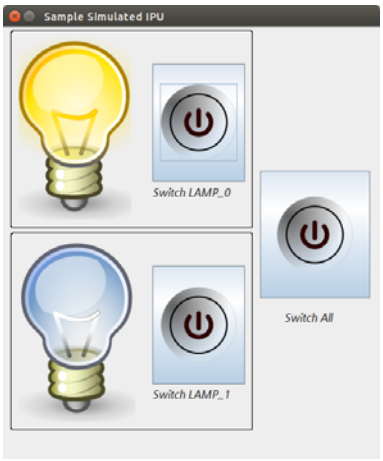

Figure 18. GUI: LAMP_0 switched ON

2.4.5 Groups resources

Group broadcast 동작을 확인하기 위해서 gscl base resource의 “groups” resource를 확인할 수 있다. “ON_ALL“ group은 모든 lamp들을 ON으로 전환하기 위한 요청을 처리한다. “OFF_ALL” group은 모든 lamp들을 OFF로 전환하는 요청을 처리한다. 그림 17은 groups의 “ON_ALL” resource를 클릭했을 때 화면을 보여준다. 그리고 LAMP_0, LAMP_1을 ON 시키기 위한 member attribute의 request URI value를 확인할 수 있다.

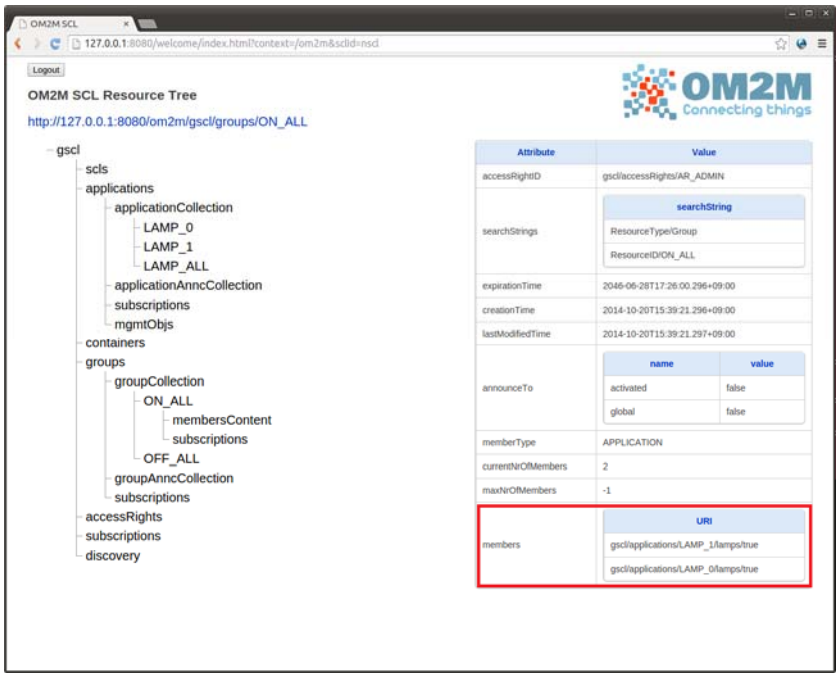

Figure 19. Web interface: The ON_ALL “group” resource

2.4.6 Remote control group of lamps

만약 “membersContent” resource에 request를 보내면, GSCL은 모든 group member의 lamp들에 broadcast를 보낸다. 그림 20의 switchAllON, switchAllOFF를 클릭하면 LAMP_0, LAMP_1을 동시에 ON/OFF 할 수 있다. 또한 그림 21처럼 java GUI의 Switch ALL을 클릭하여 group control을 할 수 있다.

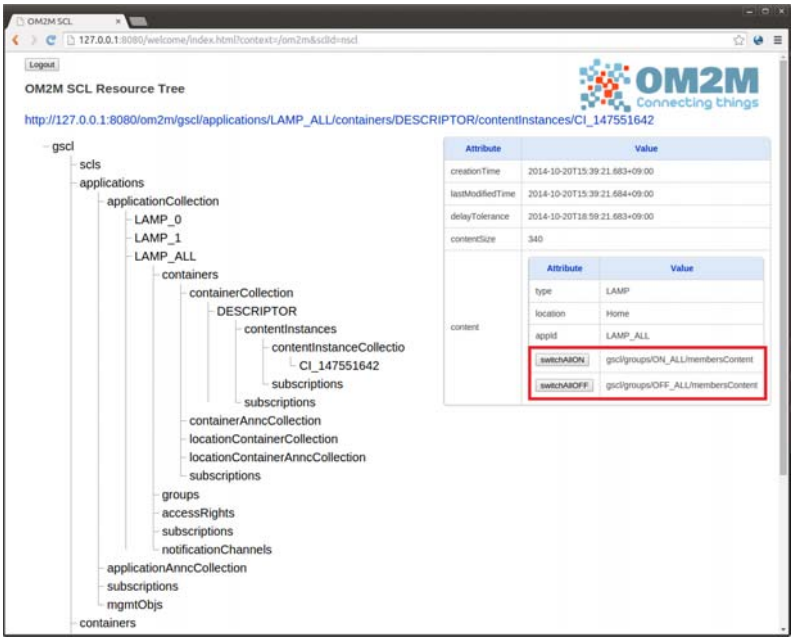

Figure 20. Web interface: The LAMP_ALL description resource

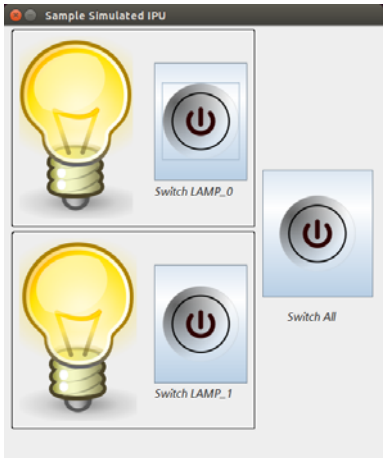

Figure 21. GUI: All lamps are switched ON

2.5 REST API

2.5.1 Install a rest client

이번 2.5장에서는 REST client를 활용하여 HTTP request (GET, POST, UPDATE, DELETE)를 OM2M platform으로 보내는 작업이 필요하다. 따라서 Chrome에서 제공하는 Client REST Simple이라는 app을 사용하겠다. Chrome의 Client REST Simple은 다음 URL에 접속하여 설치할 수 있다.

- https://chrome.google.com/webstore/detail/simple-rest-client/fhjcajmcbmldlhcimfajhfbgofnpcjmb

OM2M은 인증을 위해서 username/password를 반드시 base64로 encode하여 Basic Authorization 을 header에 적어주어야 한다. www.base64encode.org사이트를 활용하여 base64 암호화를 할 수 있다. (ex. Base64(admin:admin) = YWRtaW46YWRtaW4=)
그림 26은 chrome에서 실행한 Simple REST Client를 보여준다.

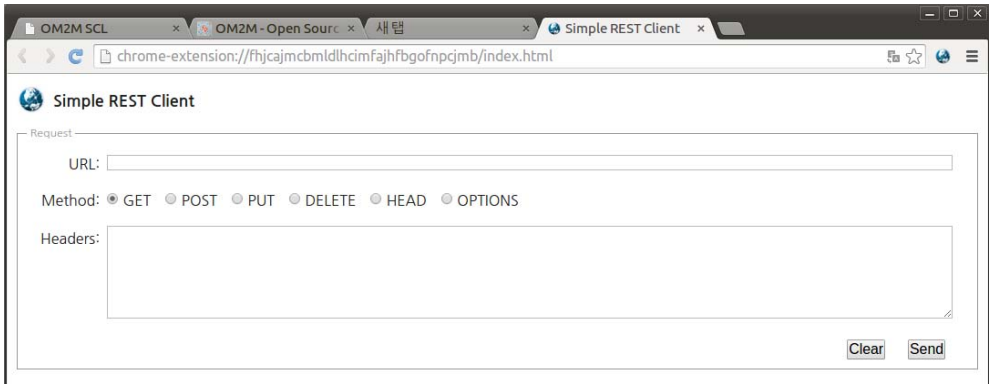

Figure 22. Simple REST Client

2.5.2 Retrieve a resource

그림 23과 같이 HTTP request를 활용하여 Resource를 검색할 수 있다. 그리고 그에 대한 응답은 그림 24에서 볼 수 있다.

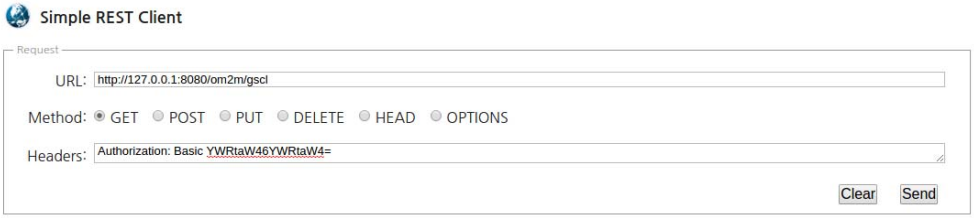

Figure 23. Retrieve a resource: HTTP request

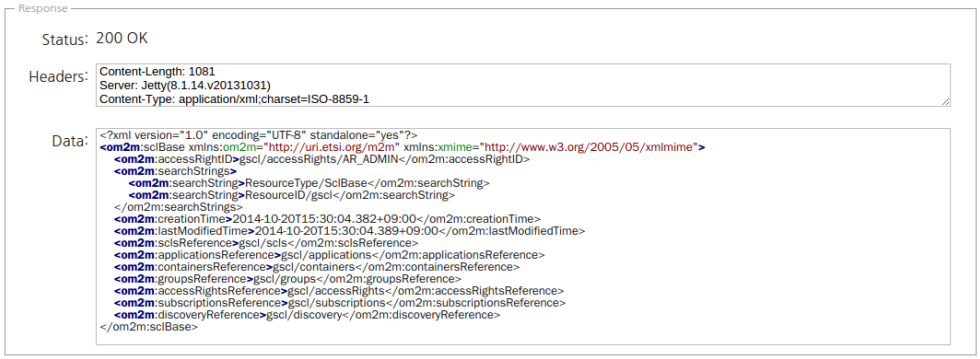

Figure 24.Retrieve a resource: HTTP response

2.5.3 Discover resources based on their search strings

Search string을 활용한 resource discovery를 활용하여 사용할 수 있는 resource를 검색할 수 있다.

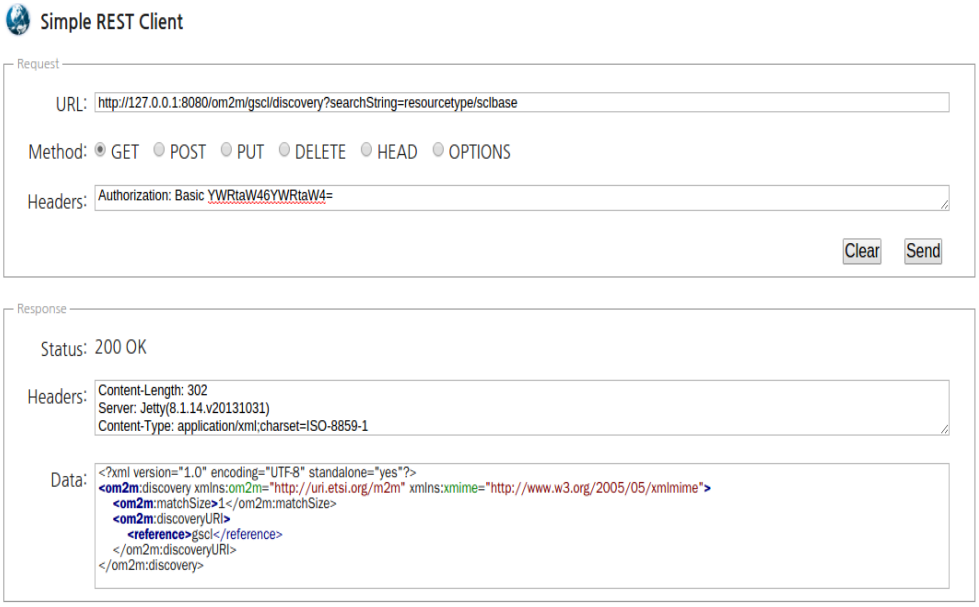

Figure 25. Discover resource based on search strings

2.5.4 Create a "MY_SENSOR" application

그림 26과 같이 HTTP 요청 메시지를 사용하여 gateway의 application에 "MY_SENSOR" parameter를 생성할 수 있다. 그림 27에서 생성된 "MY_SENSOR" 어플리케이션을 Web interface를 통해서 확인할 수 있다.

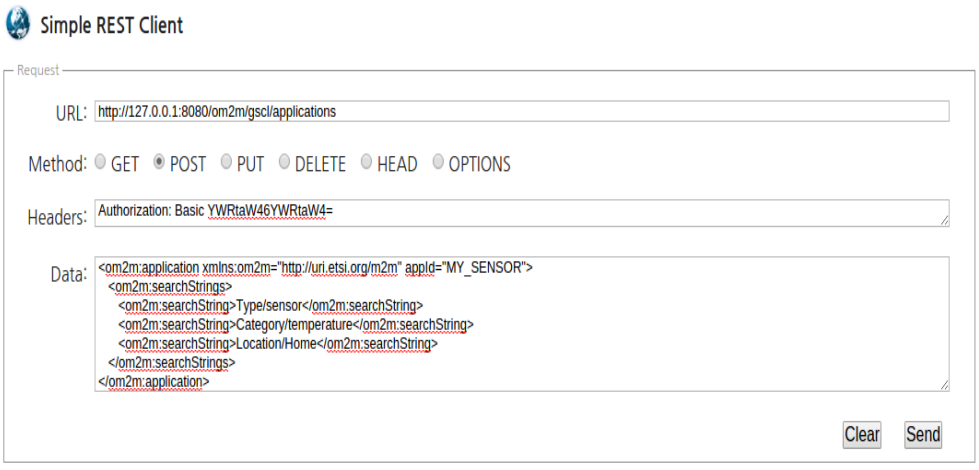

Figure 26. Create a MY_SENSOR application

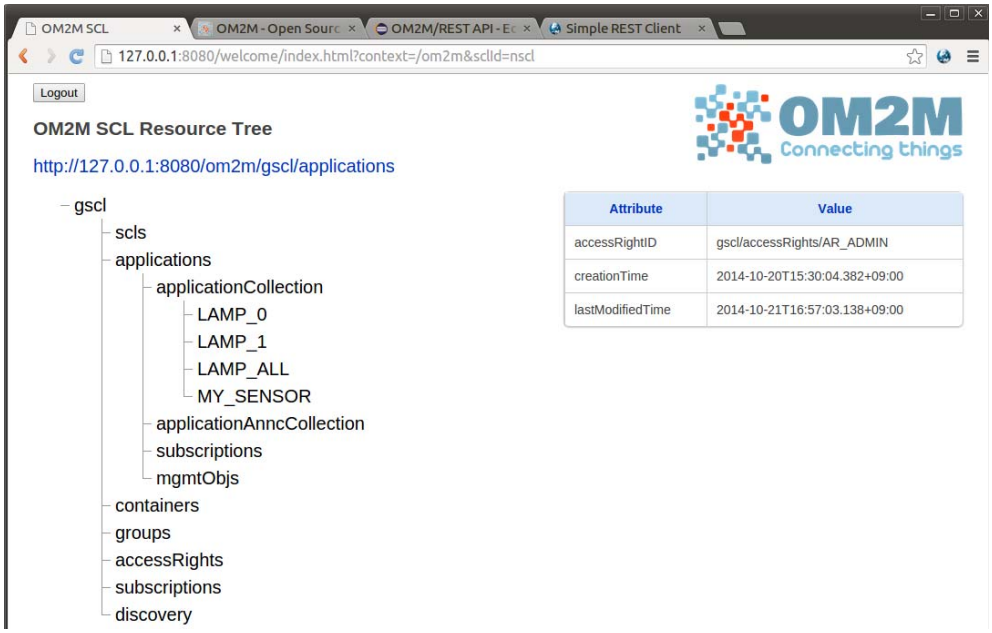

Figure 27. Create a MY_SENSOR: Web interface

2.5.5 Create a DESCRIPTOR container

그림 28과 같이 HTTP 요청 메시지를 보내서 “DESCRIPTOR” container resource parameter 를 “MY_SENSOR” application 밑에 생성할 수 있다.

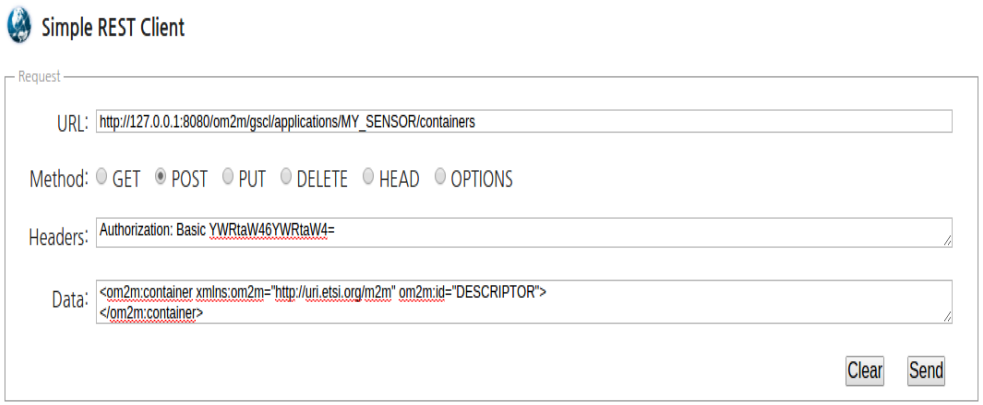

Figure 28. Create a DESCRIPTOR container

그림 29와 같이 web interface를 통해서 결과를 확인할 수 있다.

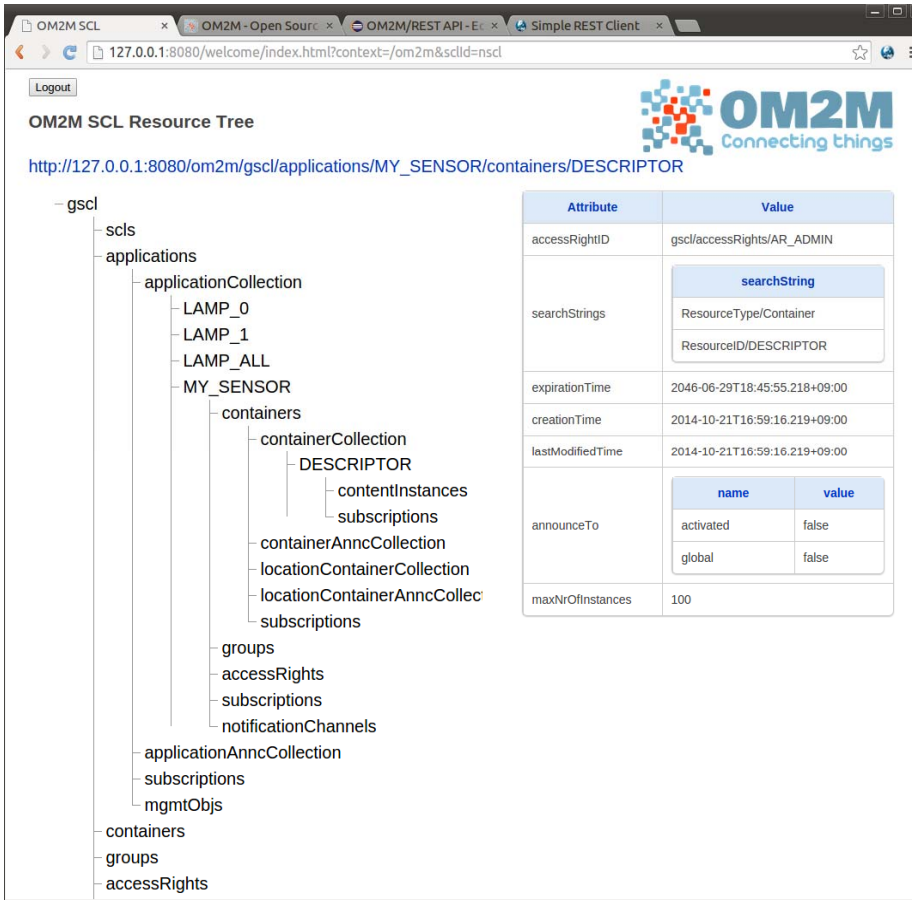

Figure 29. Create a DESCRIPTOR: Web interface

2.5.6 Create a description contentInstance

그림 30과 같이 HTTP 요청 메시지를 보내서 "DESCRIPTOR" container 밑에 description content instance resource parameter를 생성할 수 있다.

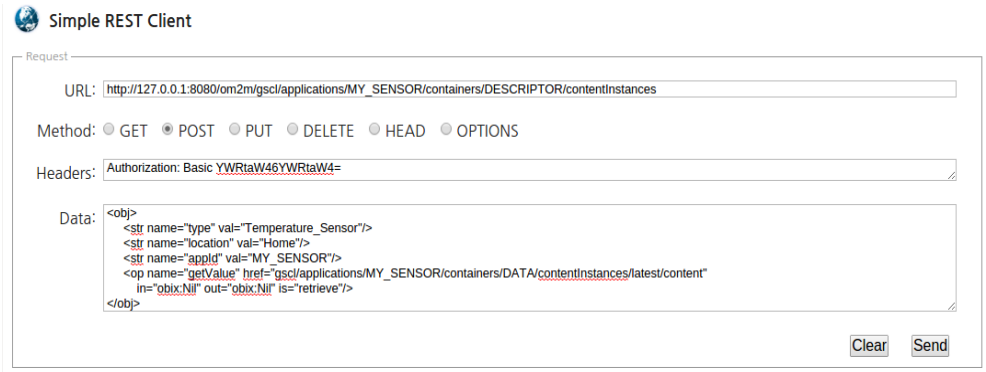

Figure 30. Create a description contentInstance

그림 31과 같이 생성된 contentInstance를 Web interface를 통해서 확인할 수 있다.

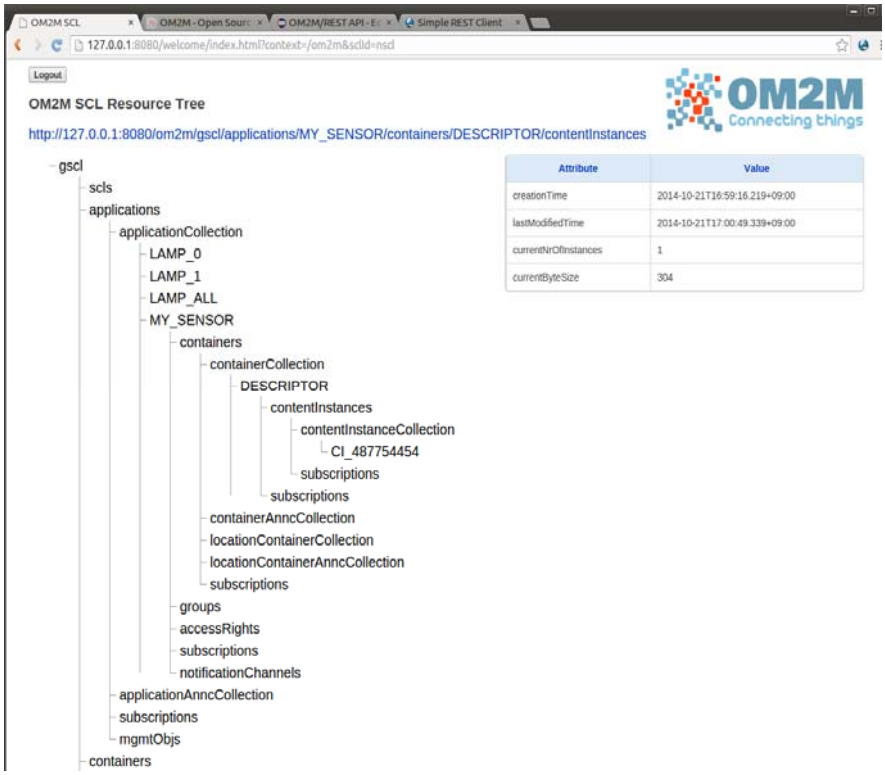

Figure 31. Create a descriptor contentInstance: Web interface

2.5.7 Create a DATA container

그림 32와 같이 HTTP 요청 메시지를 보내서 “MY_SENSOR” application 밑에 “DATA” container resource parameter를 생성할 수 있다.

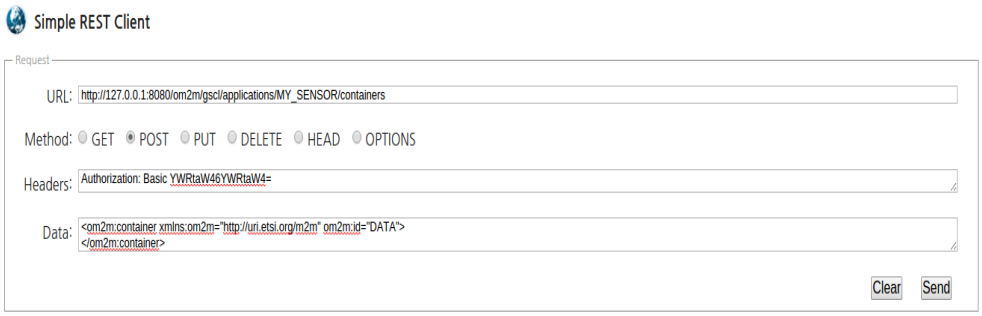

Figure 32. Create a DATA container

역시 그림 33과 같이 web interface를 통해서 생성된 DATA container를 확인할 수 있다.

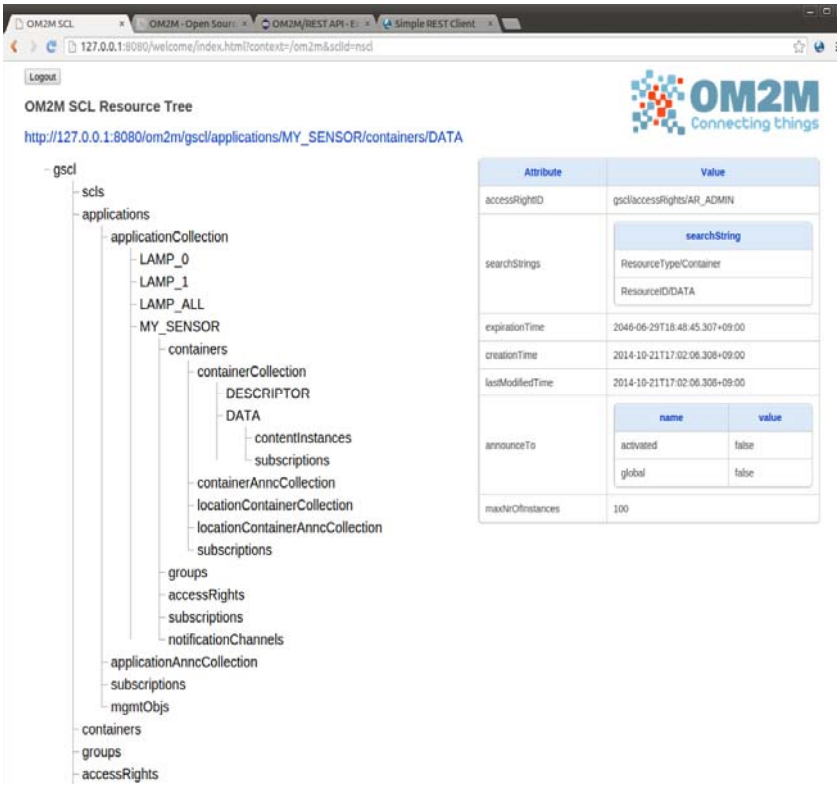

Figure 33. Create a DATA container: Web interface

2.5.8 Create a data contentInstance

그림 34와 같이 HTTP 요청 메시지를 보내서 “DATA” application 밑에 data content instance resource parameter를 생성할 수 있다. 그림 35와 같이 생성된 content instance를 web interface에서 확인할 수 있다. 또한 getValue를 클릭하여 생성된 sensor data를 확인할 수 있다.

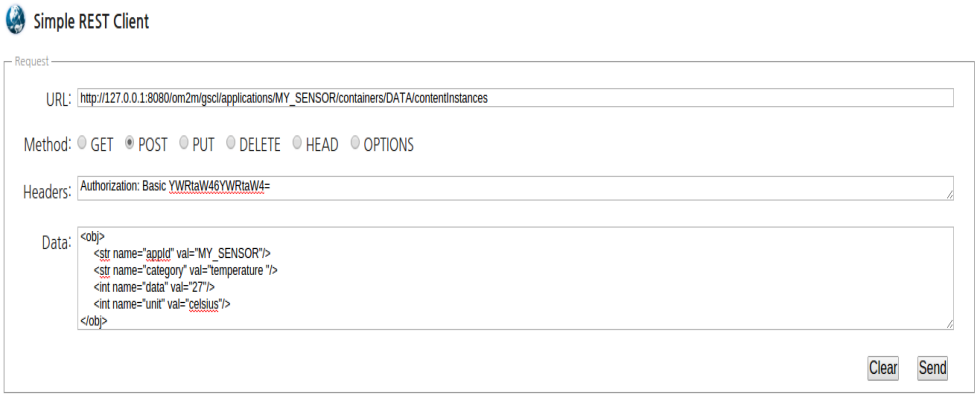

Figure 34. Create a data contentInstance

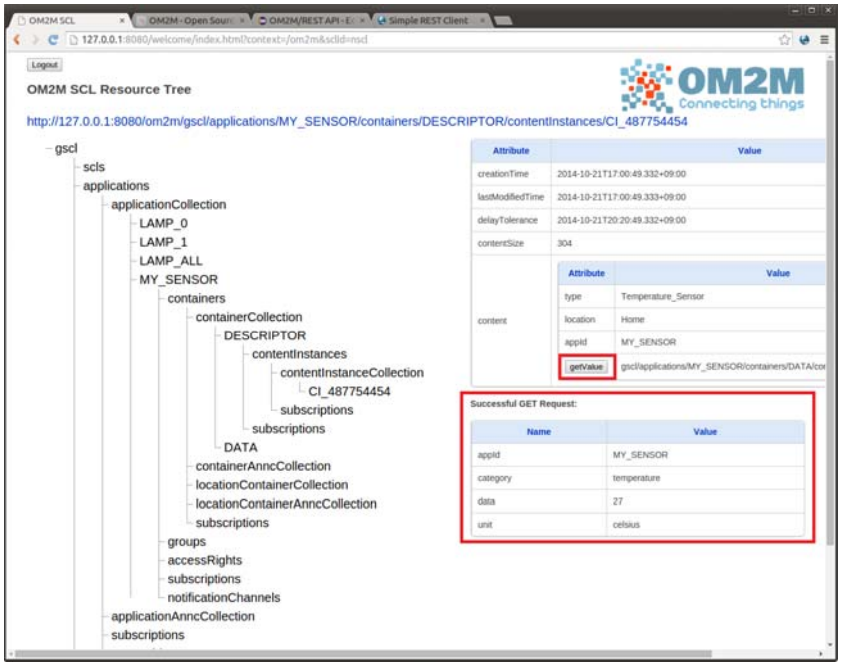

Figure 35. Create a data contentInstance and getValue

2.5.9 Subscribe to MY_SENSOR data

OM2M project 웹 사이트에서 제공하는 Monitor server sample을 다운 받아서 사용을 한다. 해당 파일의 압축을 풀고 다음의 command를 사용하여 실행한다.

- java –jar monitor.jar

Monitor의 listening port 및 context를 변경하기 위해서는 “config.properties”를 확인한다. Default port = 1400, context = /monitor이다. 그림 36은 monitor를 실행한 화면이다.

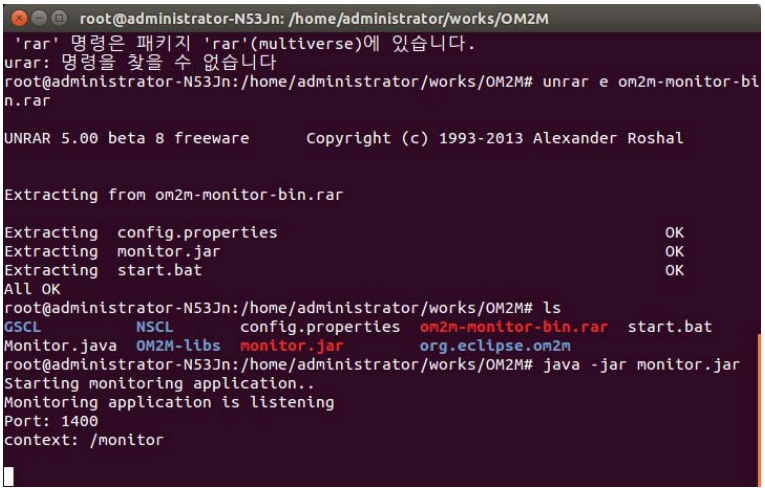

Figure 36. monitor 실행화면

그림 37과 같이 HTTP request 메시지를 보내서 subscription resource parameter를 MY_SENSOR application 밑에 생성할 수 있다.

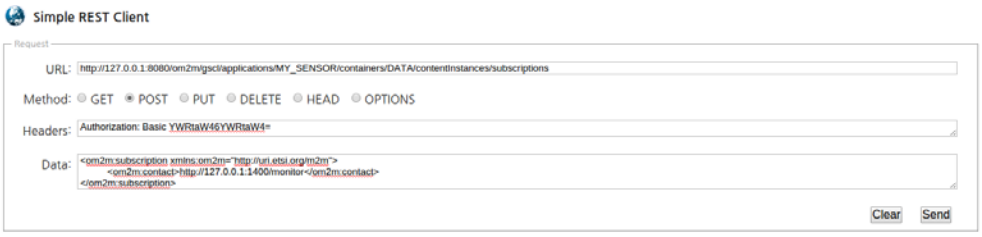

Figure 37. Create a subscription resource

또한 생성된 subscription resource를 web interface에서 확인할 수 있다.

Figure 38. Create a subscription resource: Web interface

3. 결론

지금까지 본 고에서는 Eclipse의 Open Source Platform인 OM2M을 Ubuntu 14.04기반의 리눅스 환경에서 설치하고 예제를 실행하는 방법에 대하여 살펴보았다. OM2M은 M2M 리소스를 생성하고 관리할 수 있는 RESTful API를 제공하며, 시스템 인증, 리소스 검색, 어플리케이션 등록, 컨테이너 (Container) 관리, 동기 및 비동기 통신, 접근 권한 인증 및 그룹 관리 등 다양한 서비스를 제공하고 있다. 또한 HTTP 및 CoAP 등 다양한 프로토콜 바인딩을 지원하고 Zigbee나 Phidgets 등의 장치와 원할한 통신을 제공하므로써, 앞으로 진행될 사물인터넷 중심의 시대에 제공 될 다양한 IoT 서비스들을 개발에 활용될 것으로 생각된다.

참고 문헌

[1] OM2M project, http://eclipse.org/om2m/