

oneM2M

Giacomo Tanganelli
PostDoc @ University of Pisa
g.tanganelli@iet.unipi.it

oM2M



- oM2M is an OSGi implementation of oneM2M
- It implements the IN and the MN
- It exposes a REST interface to connect external Applications (http and CoAP)
- It implements a web interface to display the resource tree
- In your VM oM2M has been already deployed

Build oM2M

- Open a shell and issue:
 \$ cd \$HOME/git/org.eclipse.om2m
 \$ mvn clean install
- Two different projects will be generated:
 - IN-CSE

\$HOME/git/org.eclipse.om2m/org.eclipse.om2m.site.in-cse/target/products/in-cse/linux/gtk/x86

- MN-CSE

\$HOME/git/org.eclipse.om2m/org.eclipse.om2m.site.mn-cse/target/products/in-cse/linux/gtk/x86

Configure oM2M

- In the IN-CSE folder edit the file:

configuration/config.ini

– Edit:

- org.eclipse.om2m.dbReset=**true**
- org.eclipse.om2m.cseBaseId=**\$yourname-in-cse**
- org.eclipse.om2m.cseBaseName=**\$yourname-in-name**

- In the MN-CSE folder edit the file:

configuration/config.ini

– Edit:

- org.eclipse.om2m.dbReset=**true**
- org.eclipse.om2m.cseBaseId=**\$yourname-mn-cse**
- org.eclipse.om2m.cseBaseName=**\$yourname-mn-name**
- org.eclipse.om2m.remoteCseId=**\$yourname-in-cse**
- org.eclipse.om2m.remoteCseName=**\$yourname-in-name**

Start oM2M

- Start the IN-CSE:
 - In the IN-CSE folder execute:
`bash start.sh`
 - Verify the IN
<http://127.0.0.1:8080/webpage>

user:admin
password:admin

IN resource tree



Logout

OM2M CSE Resource Tree

<http://localhost:8080/~Tanganelli-in-cse>

- Tanganelli-in-name
 - └ acp_admin



Attribute	Value
ty	5
ri	/Tanganelli-in-cse
ct	20161201T102227
lt	20161201T102227
acpi	<div>AccessControlPolicyIDs</div> <div>/Tanganelli-in-cse/acp-592775263</div>
cst	1
csi	Tanganelli-in-cse
srt	1 2 3 4 5 9 14 15 16 17 23
poa	<div>Point Of Access</div> <div>http://127.0.0.1:8080/</div>

IN resource tree

- Start the MN-CSE:
 - In the MN-CSE folder execute:
bash start.sh
 - Verify the new resource tree

IN resource tree



Logout

OM2M CSE Resource Tree

<http://localhost:8080/~Tanganelli-in-cse/csr-618387865>

- Tanganelli-in-name
 - acp_admin
 - Tanganelli-mn-cse



Attribute	Value
ty	16
ri	/Tanganelli-in-cse/csr-618387865
pi	/Tanganelli-in-cse
ct	20161201T104150
lt	20161201T104150
acpi	<div>AccessControlPolicyIDs</div> <div>/Tanganelli-in-cse/acp-37819740</div>
poa	<div>Point Of Access</div> <div>http://127.0.0.1:8282/</div>
cb	//om2m.org/Tanganelli-mn-cse
csi	<div>/Tanganelli-mn-cse</div>
rr	true

What is the new resource?

Create an AE

- Create a new Maven project and include Californium as dependency
- Add the json library:
 - Open a browser and explore the mvn repository

```
<dependency>
```

```
  <groupId>org.json</groupId>
```

```
  <artifactId>json</artifactId>
```

```
  <version>20160810</version>
```

```
</dependency>
```

Create an AE (2)

- Create a CoAP client to interact with the IN node

- To create an AE:

- POST to 127.0.0.1:5683/~/\$yourname-in-cse
- Payload in json:

```
{  
  "m2m:ae":{  
    "api": "TempApp-ID",  
    "rn": "TempApp",  
    "rr": "true"  
  }  
}
```

api = Application ID
rn = ResourceName
rr = RequestReachability

Create an AE (3)

- Set oM2M specific options
 - ty = 2 (ResourceType for the AE is 2)
 - new Option(267, 2)
 - authorization (admin:admin)
 - new Option(256, "admin:admin")
 - Set Content Format to json
 - Set accept to json

Json Library

- To create an empty object:
 - `JSONObject obj = new JSONObject();`
- To add elements to object:
 - `obj.put(key, value)`
- To create an object from a json string:
 - `JSONObject obj = new JSONObject(string)`
- To get an element from an object:
 - `obj.get(key)`

Exercise 1

- Create a CoAP Client that creates an AE on the IN node. Exploit the json library to create the request payload and to parse the response payload.
- Open the AE in the resource tree.

Create a Container

- To create a Container:
 - POST to:
127.0.0.1:5683/~/\$yourname-in-cse/\$yourname-in-name/TempApp
 - Payload in json:

```
{  
  "m2m:cin":{  
    "rn": "DATA",  
  }  
}
```

Create a Container (2)

- Set oM2M specific options
 - ty =3 (ResourceType for the Container is 3)
 - new Option(267, 3)
 - authorization (admin:admin)
 - new Option(256, "admin:admin")
 - Set Content Format to json
 - Set accept to json

Exercise 2

- Add a container to the AE of the previous example. Parse the results to get the “la” (last data) path
- Open the Container in the resource tree.

Publish data

- Every data is a ContentInstance:

- POST to:

127.0.0.1:5683/~/\$yourname-in-cse/\$yourname-in-name/TempApp/DATA

- Payload in json:

```
{  
  "m2m:cnt":{  
    "cnf": "new Reading",  
    "con": "12 C°"  
  }  
}
```

cnf = ContentInfo
con = Content



Create a ContentInstance (2)

- Set oM2M specific options
 - ty =4 (ResourceType for the ContentInstance is 4)
 - new Option(267, 4)
 - authorization (admin:admin)
 - new Option(256, “admin:admin”)
 - Set Content Format to json
 - Set accept to json

Exercise 3

- Publish new data to the DATA container created before.
- Write a Client that:
 - Use a Thread to periodically read the “la” path to get the last value
 - Use another Thread to publish data.