oneM2M Federation: Multi-Vendor Internet of Things

Team 30: Adib Pournazari, Altay Adademir, Dennis Parchkov, Edmond Ipindamitan, and Matthew Consterdine.

The Problem

This project is about the Internet of Things.

- Vendors typically isolate their platforms.
- Restricted interoperability leads to restricted functionality.
- This limits what the end user can achieve.

A global Machine to Machine communication standard is needed.

oneM2M Standard

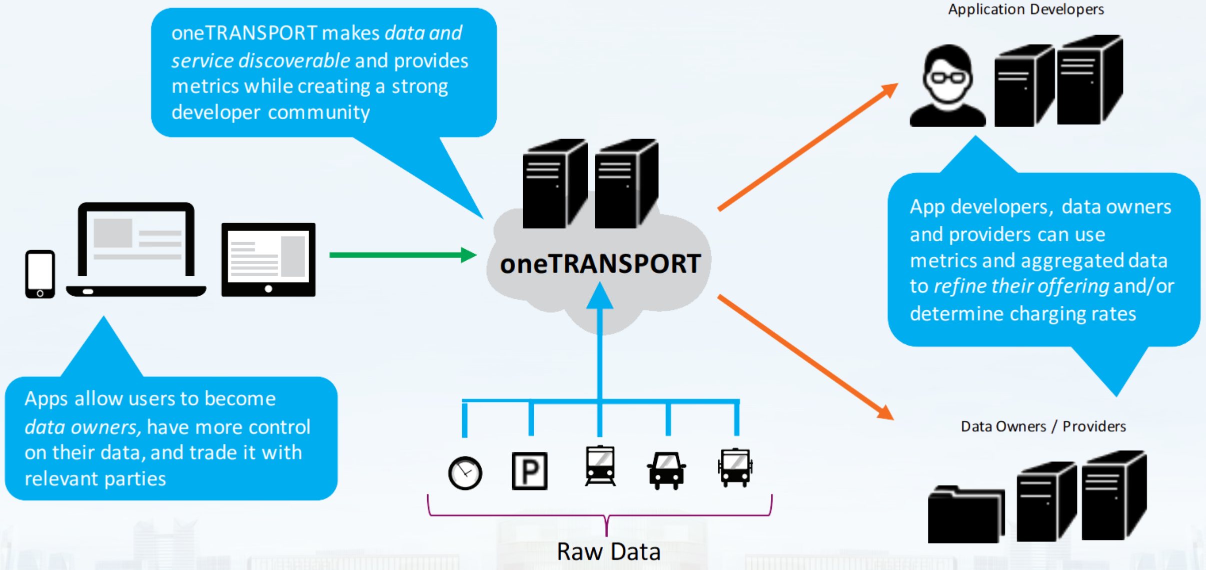
Introducing the oneM2M standard.

- An open, global Machine to Machine communication standard.
- Created by eight leading standard organisations.
- Allows vendors to federate different data sources.

Adopted into many Internet of Things platforms, including…

InterDigital’s oneTRANSPORT

- A smart transport system by the projects client, INTERDIGITAL.
- Currently interoperates with external service providers.



oneTRANSPORT makes *data and service discoverable* and provides metrics while creating a strong developer community

Apps allow users to become *data owners,* have more control on their data, and trade it with relevant parties

App developers, data owners and providers can use metrics and aggregated data to *refine their offering* and/or determine charging rates

Primary Goals

Explore the practical applications of oneM2M federation:

1. Investigate open source oneM2M implementations.
2. Evaluate various hardware platforms.
3. Develop a plug-in integrating sensor data.
4. Deploy to hardware and the cloud.
5. Federation with our clients system.

Secondary Goals

Less critical, but still important:

- Increasing data source complexity.
- Adding secure connections.
- Client authentication and access levels.

- ~~Interactive visualisation dashboard.~~

Project Scope

It is useful to define project scope. For this project:

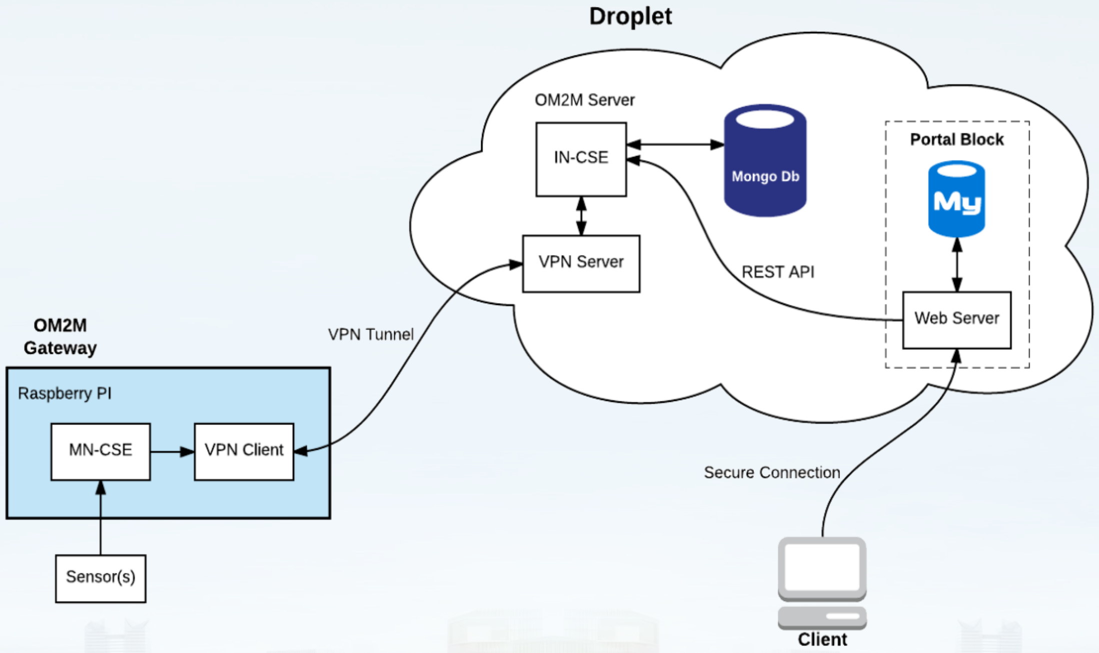
- A rich computing and communication environment is assumed:
  → HTTP is used instead of MQTT or CoAP.

- Performance, efficiency or scalability aren't a primary concern:
  → While they will be considered, this project is on a small scale.

This is primarily a research focused project.

Initial Design

Designs change, but it never hurts to plan ahead.

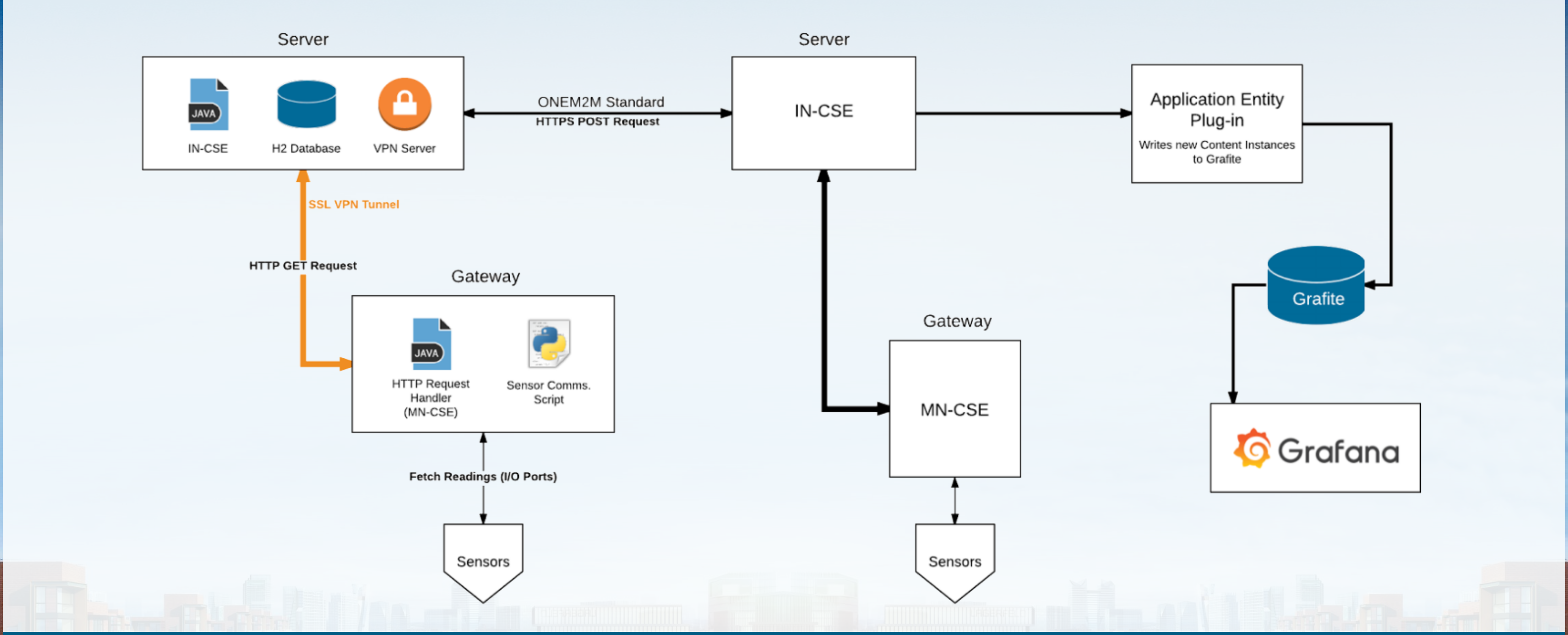


Client Feedback

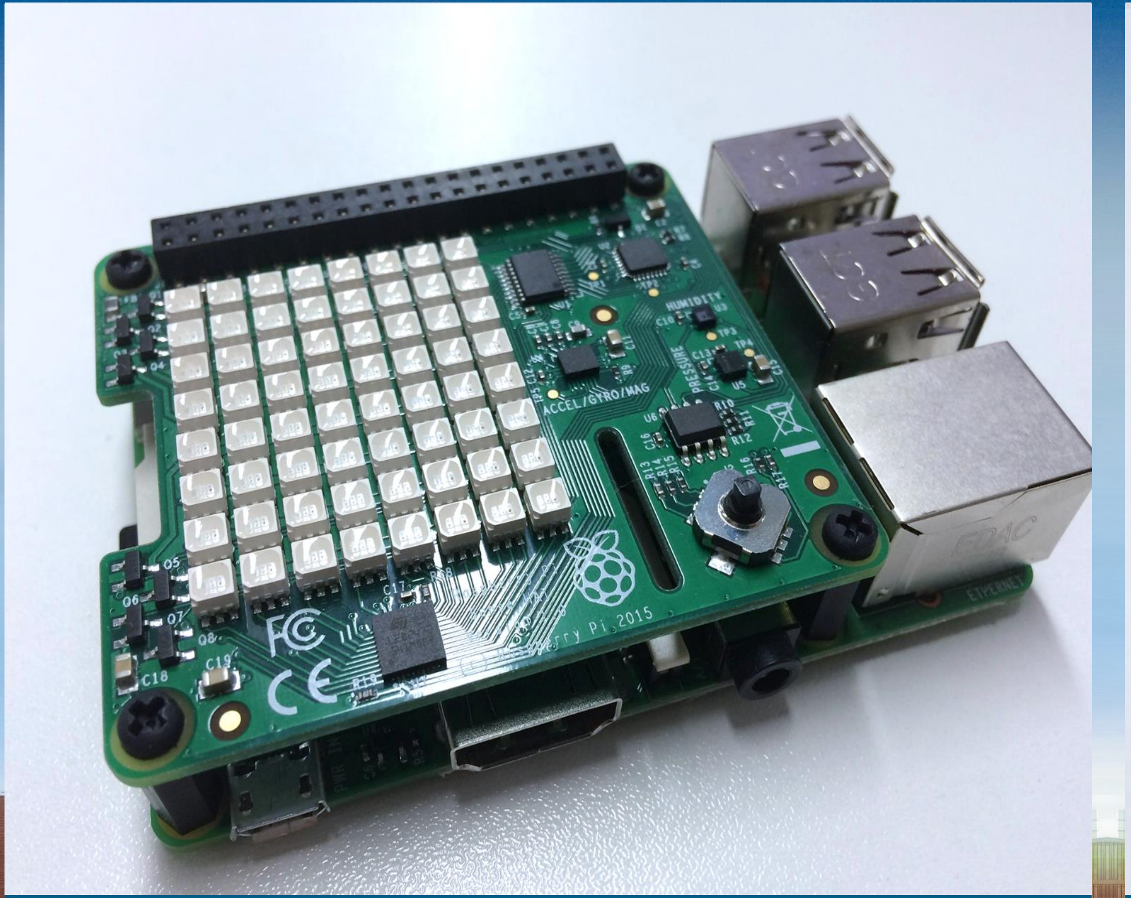
Taking into account feedback, the project’s aims changed:

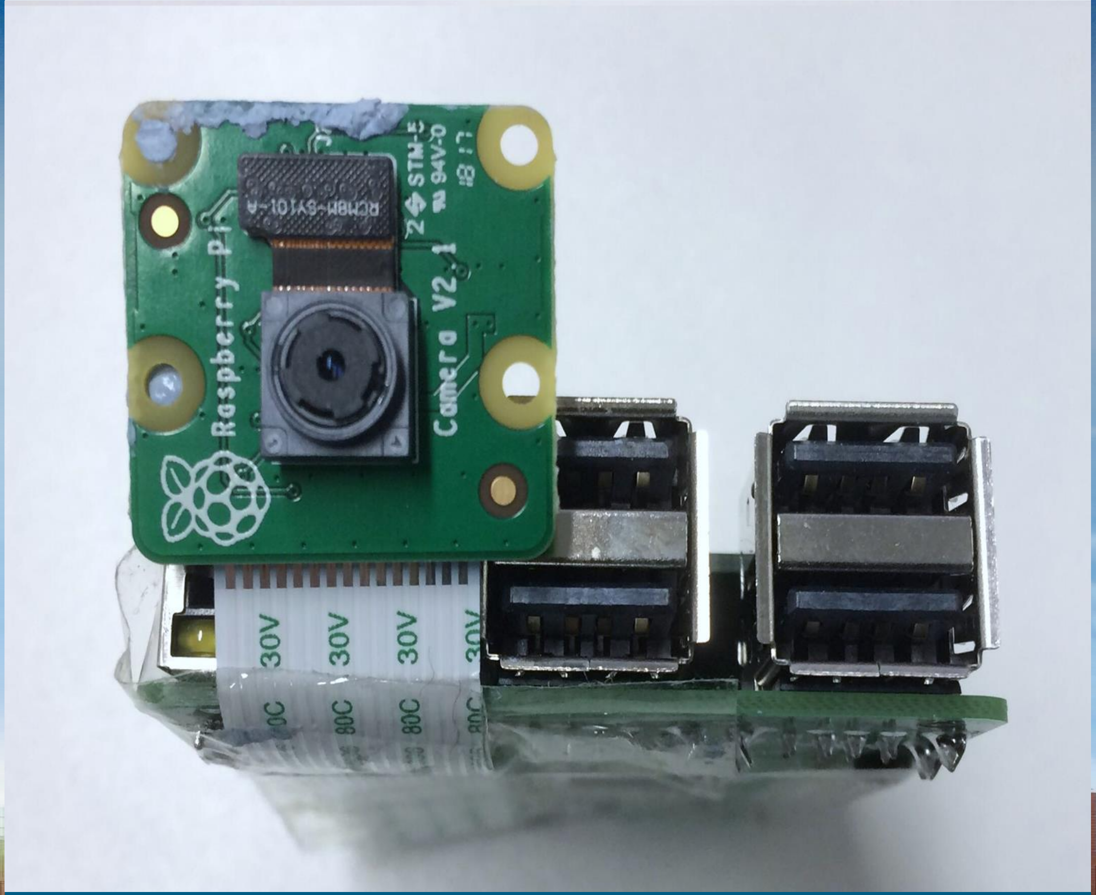
- Scrapping unnecessary features.
- Removing the dashboard.
- Produce a simple indoor demo.
- Ensure ease of use.
- Focus on federation.

Final Design



Raspberry Pi and Camera Streaming





Plug-in Implementation

Eclipse OM2M Plug-In

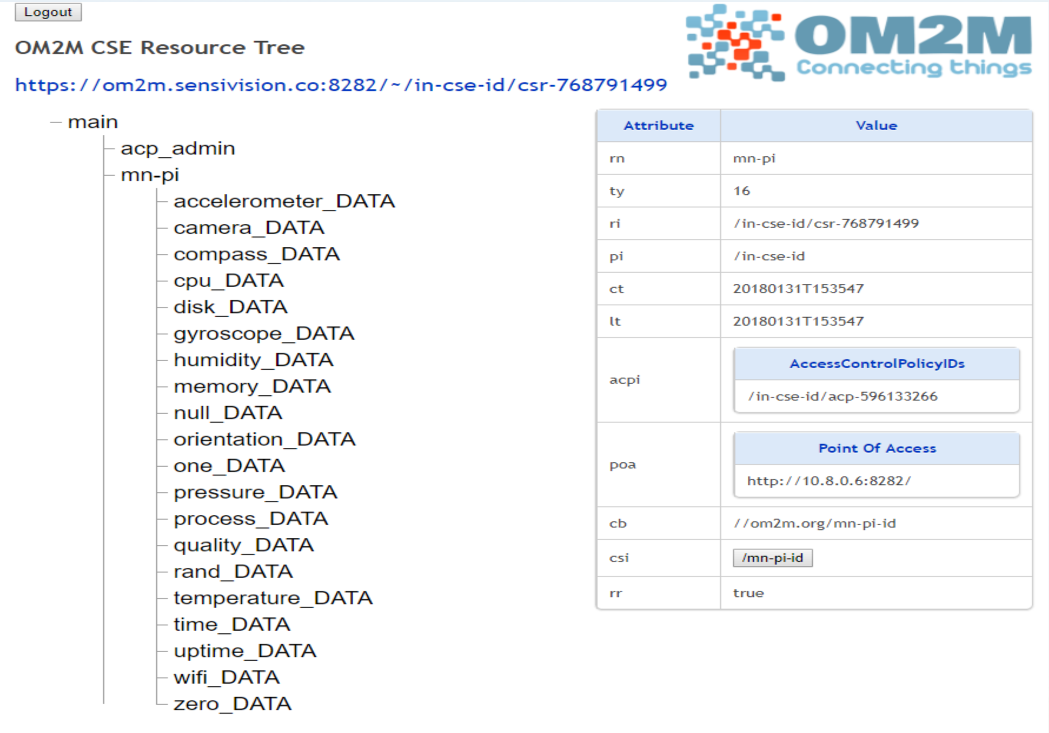
- Provides an easy to use, web Interface.
- Requires a single IDE leading to IDE configuration issues.
- Little documentation, but plenty of guides and wiki articles.

OpenMTC Plug-in

- No dashboard, so the Insomnia REST Client was used in place.
- Simpler design lead to improved video streaming.

Development Tools

Eclipse OM2M CSE Resource Tree



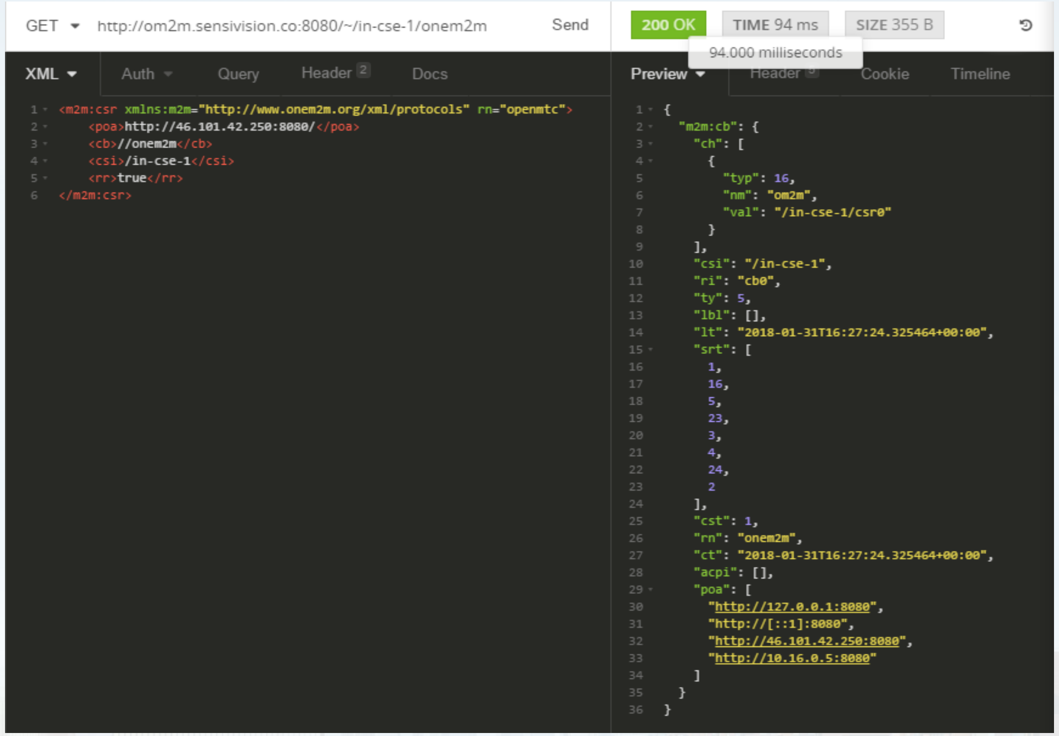
OM2M CSE Resource Tree

https://om2m.sensivision.co:8282/~/in-cse-id/csr-768791499

- main
  - acp_admin
  - mn-pi
    - accelerometer_DATA
    - camera_DATA
    - compass_DATA
    - cpu_DATA
    - disk_DATA
    - gyroscope_DATA
    - humidity_DATA
    - memory_DATA
    - null_DATA
    - orientation_DATA
    - one_DATA
    - pressure_DATA
    - process_DATA
    - quality_DATA
    - rand_DATA
    - temperature_DATA
    - time_DATA
    - uptime_DATA
    - wifi_DATA
    - zero_DATA

Attribute	Value
rn	mn-pi
ty	16
ri	/in-cse-id/csr-768791499
pi	/in-cse-id
ct	20180131T153547
lt	20180131T153547
acpi	AccessControlPolicyIDs /in-cse-id/acp-596133266
poa	Point Of Access http://10.8.0.6:8282/
cb	//om2m.org/mn-pi-id
csi	/mn-pi-id
rr	true

The Insomnia REST client



GET http://om2m.sensivision.co:8080/~/in-cse-1/onem2m

200 OK TIME 94 ms SIZE 355 B

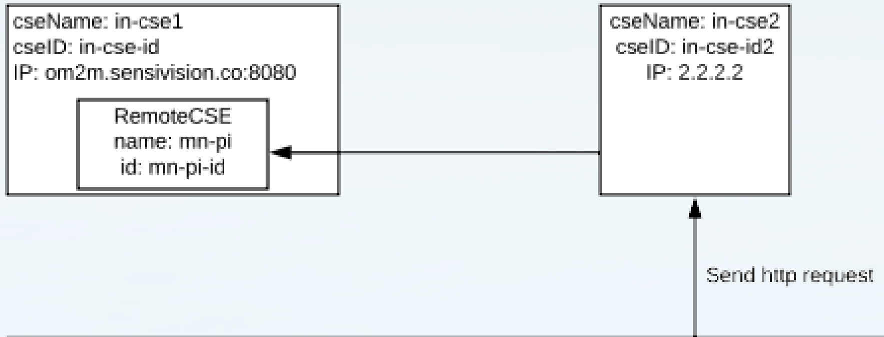
94.000 milliseconds

```xml
<m2m:csr xmlns:m2m="http://www.onem2m.org/xml/protocols" rn="openmtc">
    <poa>http://46.101.42.250:8080/</poa>
    <cb>//onem2m</cb>
    <csi>/in-cse-1</csi>
    <rr>true</rr>
</m2m:csr>
```

```json
{
  "m2m:cb": {
    "ch": [
      {
        "typ": 16,
        "nm": "om2m",
        "val": "/in-cse-1/csr0"
      }
    ],
    "csi": "/in-cse-1",
    "ri": "cb0",
    "ty": 5,
    "lbl": [],
    "lt": "2018-01-31T16:27:24.325464+00:00",
    "srt": [
      1,
      16,
      5,
      23,
      3,
      4,
      24,
      2
    ],
    "cst": 1,
    "rn": "onem2m",
    "ct": "2018-01-31T16:27:24.325464+00:00",
    "acpi": [],
    "poa": [
      "http://127.0.0.1:8080",
      "http://[::1]:8080",
      "http://46.101.42.250:8080",
      "http://10.16.0.5:8080"
    ]
  }
}
```

Achieving Federation

Federation was achieved by sending an XML REST request:



HTTP request	POST http://2.2.2.2/~/in-cse-id2/
x-m2m-origin	admin:admin
content-type	application/xml;ty=16

Body	<m2m:csr xmlns:m2m="http://www.onem2m.org/xml/protocols" xmlns:hd="http://www.onem2m.org/xml/protocols/homedomain" rn="pi">
	    <poa>http://om2m.sensivision.co:8080/</poa>
	    <cb>//om2m.org/mn-pi</cb>
	    <csi>/mn-pi-id</csi>
	    <rr>true</rr>
	</m2m:csr>

Testing

Untested code is untrusted code, so the team used many techniques to validate the different plug-ins:

- Pair Programming.
- Integration Testing.
- Unit Testing.
- Regression Testing.

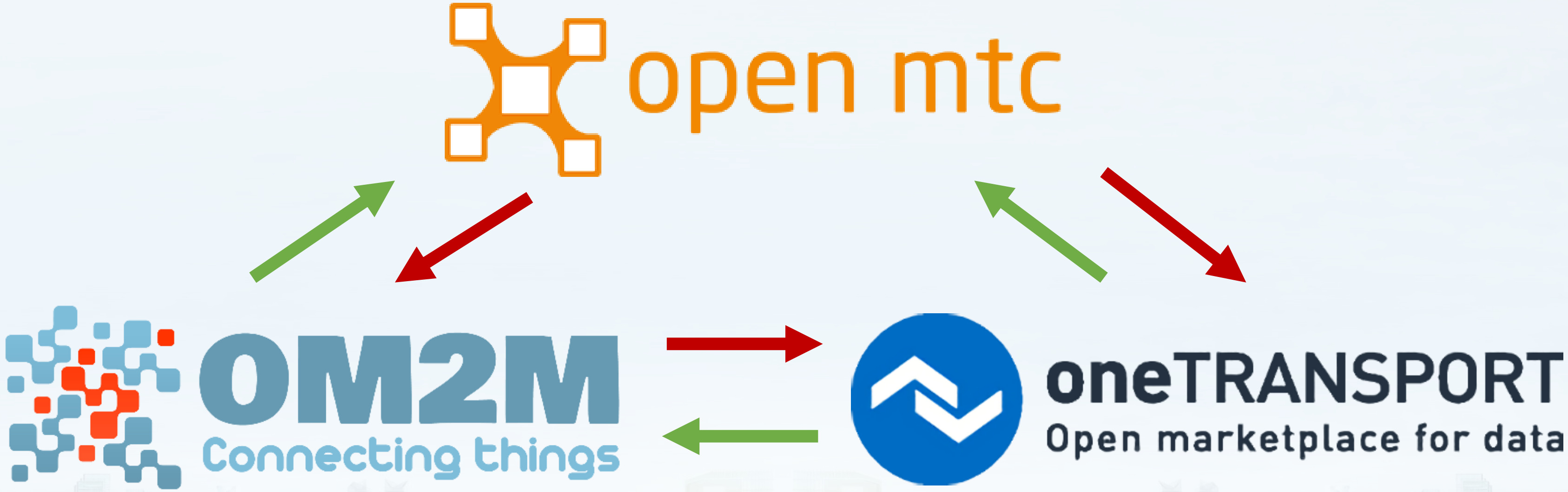
Eclipse OM2M and OpenMTC Evaluation

The two plug-ins were evaluated, and certain key points emerged:

- Development usability.
- Interfacing with the different implementations.
- Issues with adding HTTPS.
- Video streaming performance.
- Intra/Inter Federation.

Federation Evaluation

- Between Eclipse OM2M and OpenMTC.
- Between Eclipse OM2M/OpenMTC and oneTRANSPORT.

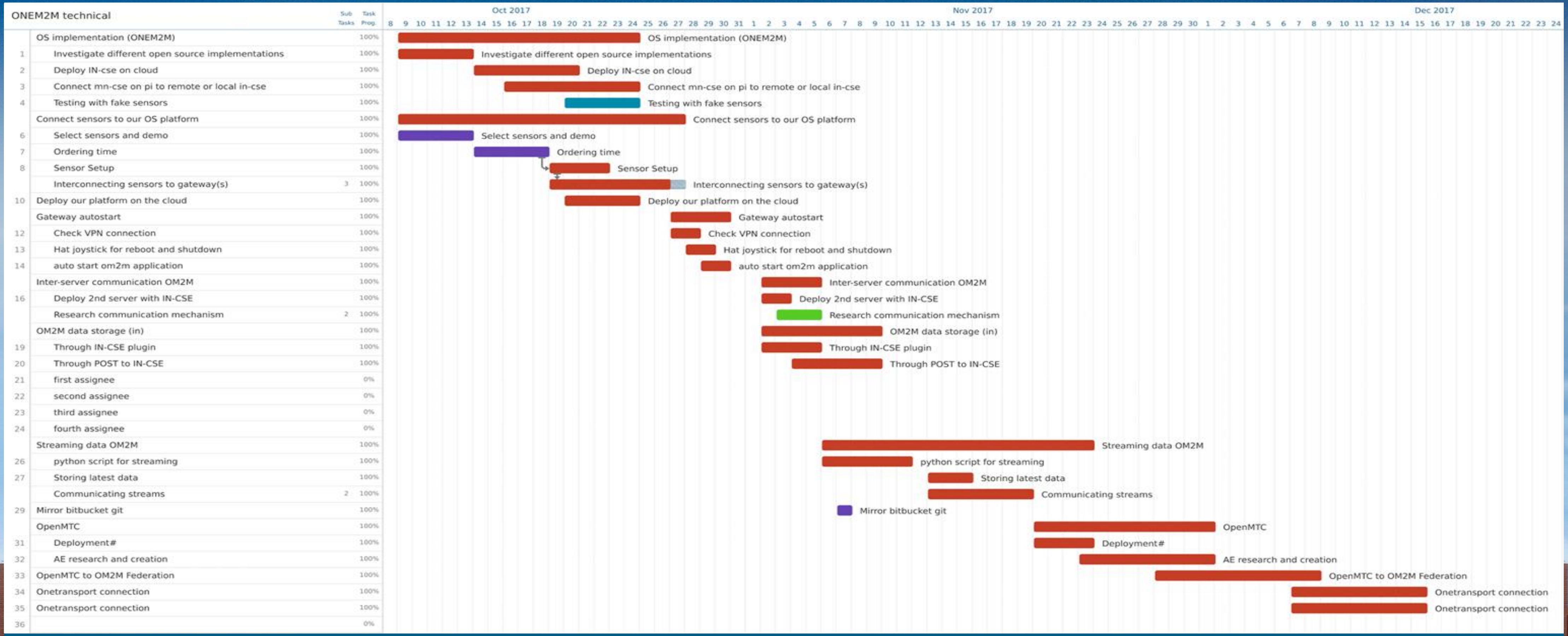


Future Work

As with any major project, there is always more to be done:

- Federation mechanisms.
- Increasing complexity.
- CoAP and MQTT communication.
- Alternate video streaming.
- Dashboard visualisation.

Project Planning



	ONEM2M technical	Sub Tasks	Task Prog.
	OS implementation (ONEM2M)		100%
1	Investigate different open source implementations		100%
2	Deploy IN-cse on cloud		100%
3	Connect mn-cse on pi to remote or local in-cse		100%
4	Testing with fake sensors		100%
	Connect sensors to our OS platform		100%
6	Select sensors and demo		100%
7	Ordering time		100%
8	Sensor Setup		100%
	Interconnecting sensors to gateway(s)	3	100%
10	Deploy our platform on the cloud		100%
	Gateway autostart		100%
12	Check VPN connection		100%
13	Hat joystick for reboot and shutdown		100%
14	auto start om2m application		100%
	Inter-server communication OM2M		100%
16	Deploy 2nd server with IN-CSE		100%
	Research communication mechanism	2	100%
	OM2M data storage (in)		100%
19	Through IN-CSE plugin		100%
20	Through POST to IN-CSE		100%
21	first assignee		0%
22	second assignee		0%
23	third assignee		0%
24	fourth assignee		0%
	Streaming data OM2M		100%
26	python script for streaming		100%
27	Storing latest data		100%
	Communicating streams	2	100%
29	Mirror bitbucket git		100%
	OpenMTC		100%
31	Deployment#		100%
32	AE research and creation		100%
33	OpenMTC to OM2M Federation		100%
34	Onetransport connection		100%
35	Onetransport connection		100%
36			0%

Thank you for listening

Any Questions?

Image Credits

- City Render by Regiars, CC BYSA 3.0: https://commons.wikimedia.org/wiki/File:Территория "СМАРТ Сити Казань".jpg
- Cloudy Sky by Matteo Fusco, Unsplash: https://unsplash.com/photos/m94kn8Rp61Q
- Creating the Living Network by oneTRANSPORT: http://interdigital.com/download/56d5c75ae26228c7b700189c