6th UCAAT User Conference on Advanced Automated Testing

Paris, 16-18 October 2018

Organizer: Testing Solutions & Services

Continuous testing of oneM2M IoT products with Docker and Jenkins

Presented by Bogdan Stanca-Kaposta (Spirent)

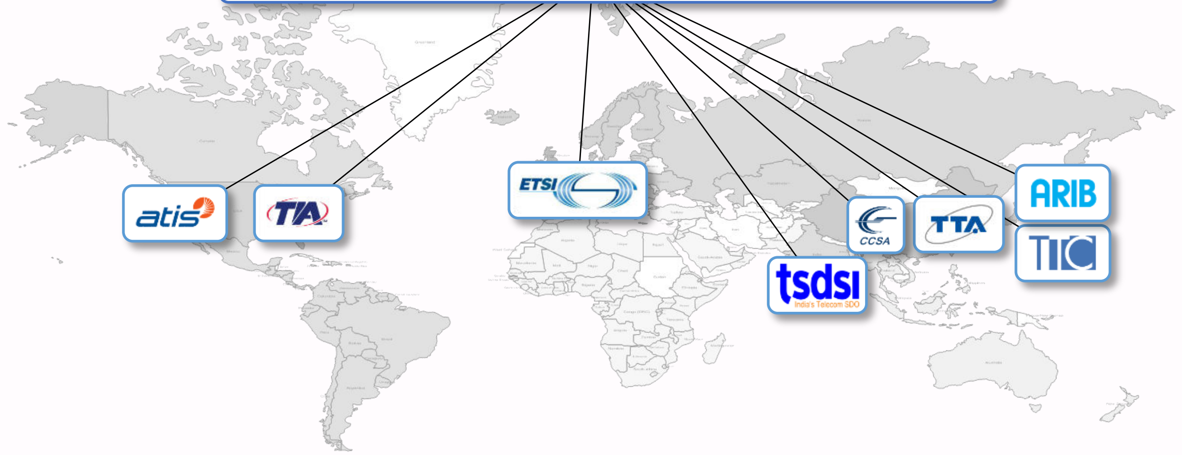
Dale Seed, Bob Flynn (InterDigital)

spirent Promise. Assured.

What is oneM2M ?

oneM2M Partnership Project

Over 200 member organizations in oneM2M



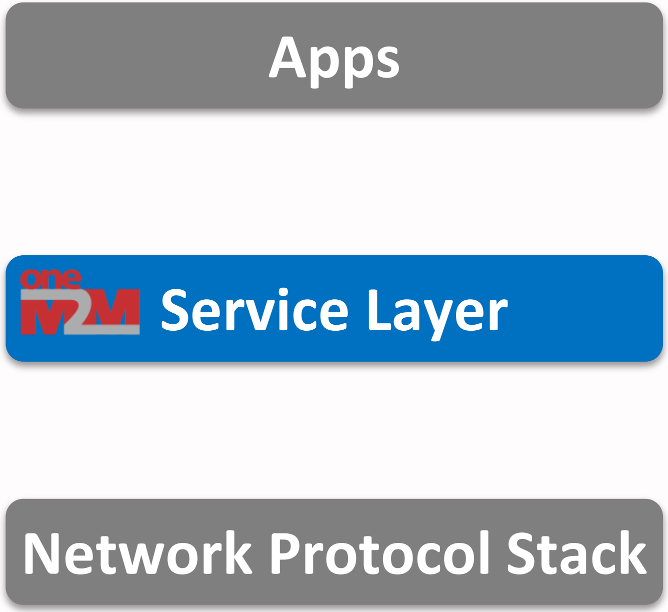
cen CENELEC | broadband forum | oma Open Mobile Alliance | New Generation M2M Consortium | GlobalPlatform

www.oneM2M.org

All document are publically available

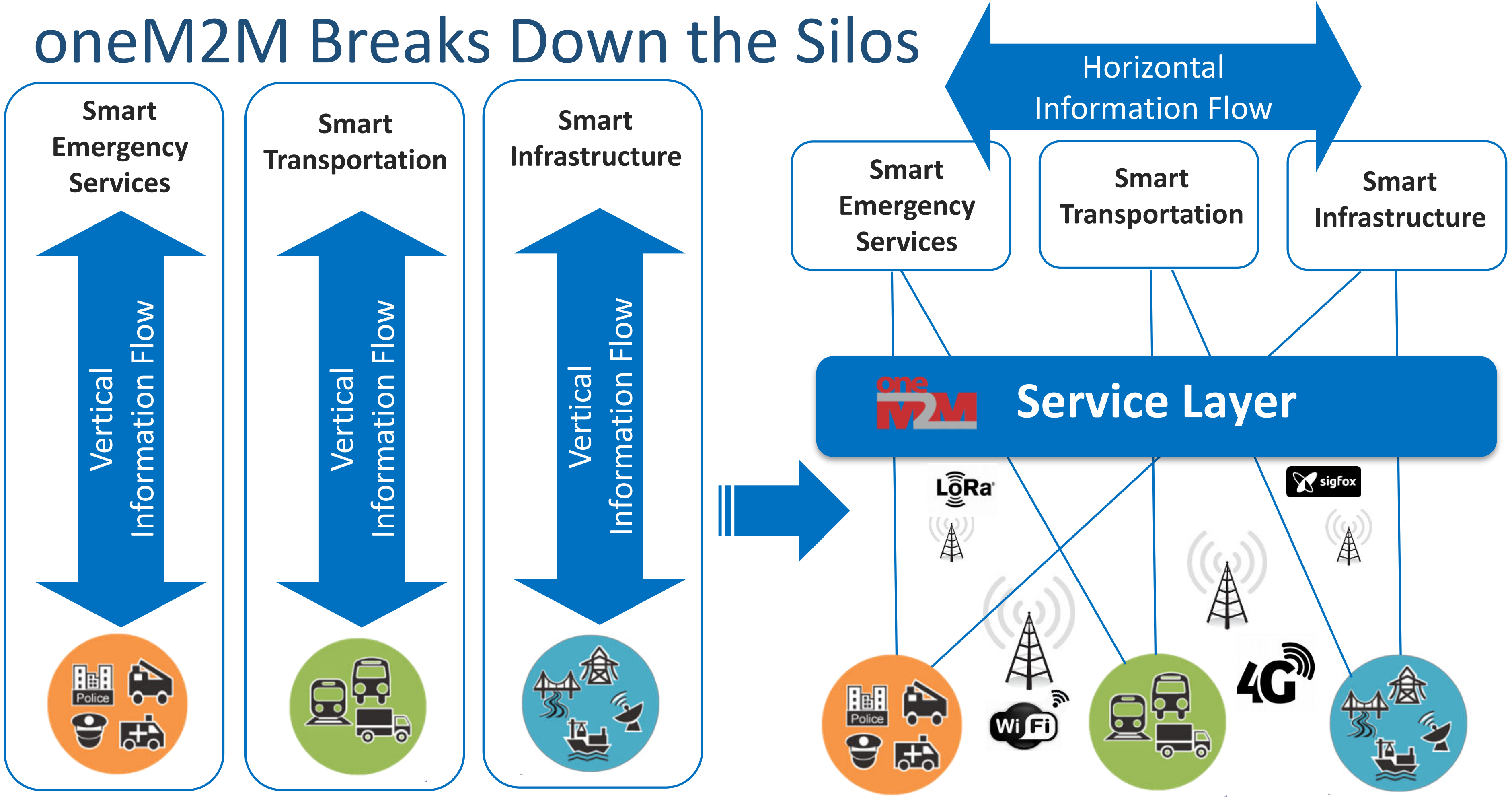
oneM2M is a trademark of the Partners Type 1 of oneM2M

oneM2M Service Layer

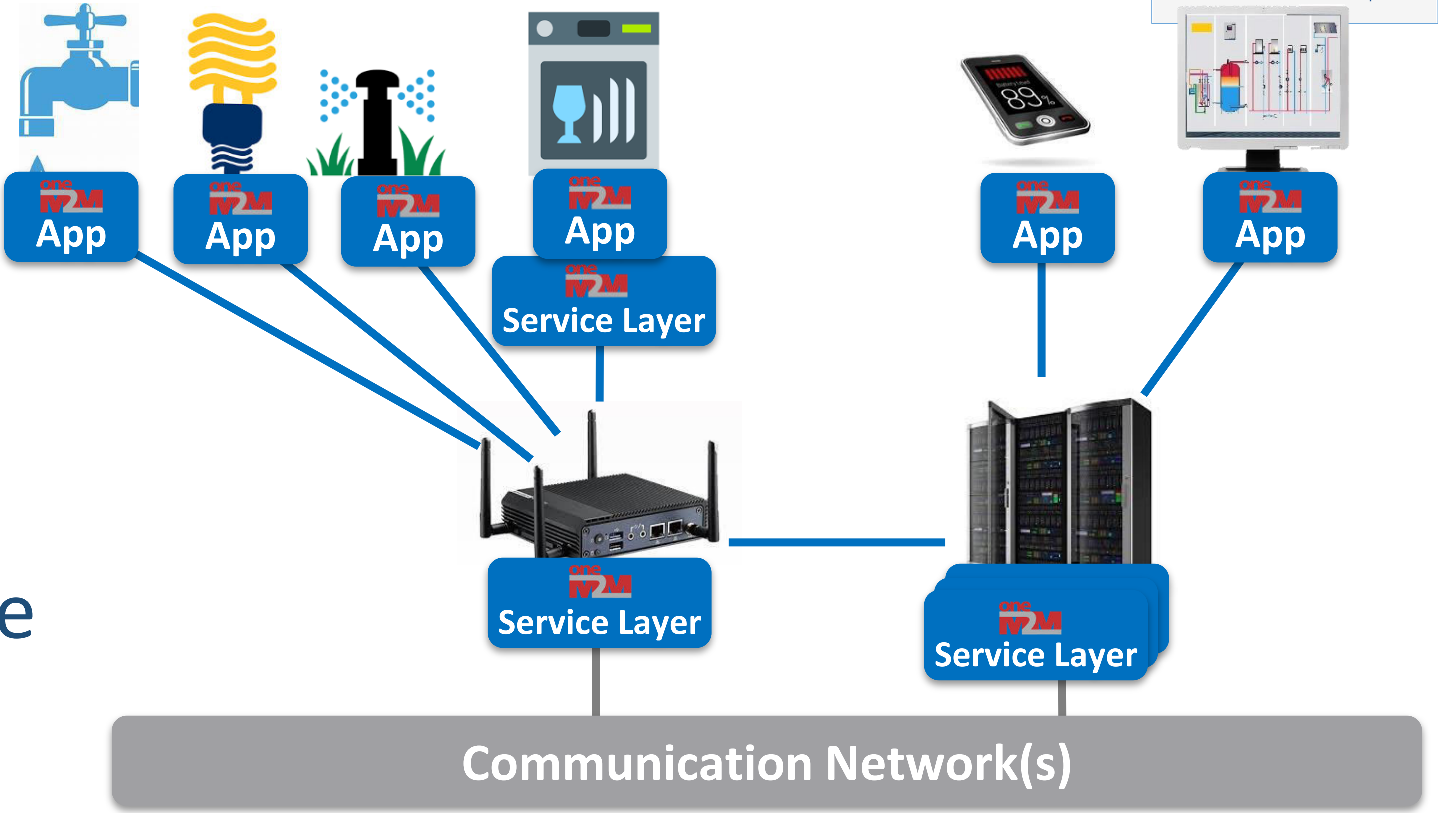


- A software “framework” that sits between IoT apps and underlying network protocol and communication stack
- Supports a common set of horizontal services that IoT devices and apps across different industry verticals commonly require
- Can be deployed on devices, gateways and servers, highly distributed and scalable

oneM2M Breaks Down the Silos



oneM2M is Distributive and Scalable



The Problem

- Assure the quality of the development process of oneM2M components and their tests
- All components were under development
  - TTCN-3 Test Suite
  - Test Adaptation
  - System Under Test (SUT)
- Multiple configurations possible

oneM2M

- Provides interoperability for Machine-to-Machine and IoT technologies
- TTCN-3 Test Cases under development
  - 700+ Test Cases
  - 4 Bindings (HTTP, MQTT, CoAP, WebSockets)
  - 3 Encodings (JSON, XML, CBOR)
  - 3 Standard Releases (4th release is being currently developed)
  - 7 Profiles

SUT issues

- SUT still under development
  - Software SUT regularly updated
- How to make sure that
  - The developers have all the same SUT configuration
  - All machines run the same OS version and libraries
  - The build servers can handle multiple SUTs

Docker

- Containers are portable
- Uses 50% less resources comparing to VMs
- Ideal for
  - Micro services
  - Continuous integration and continuous delivery

Example Dockerfile

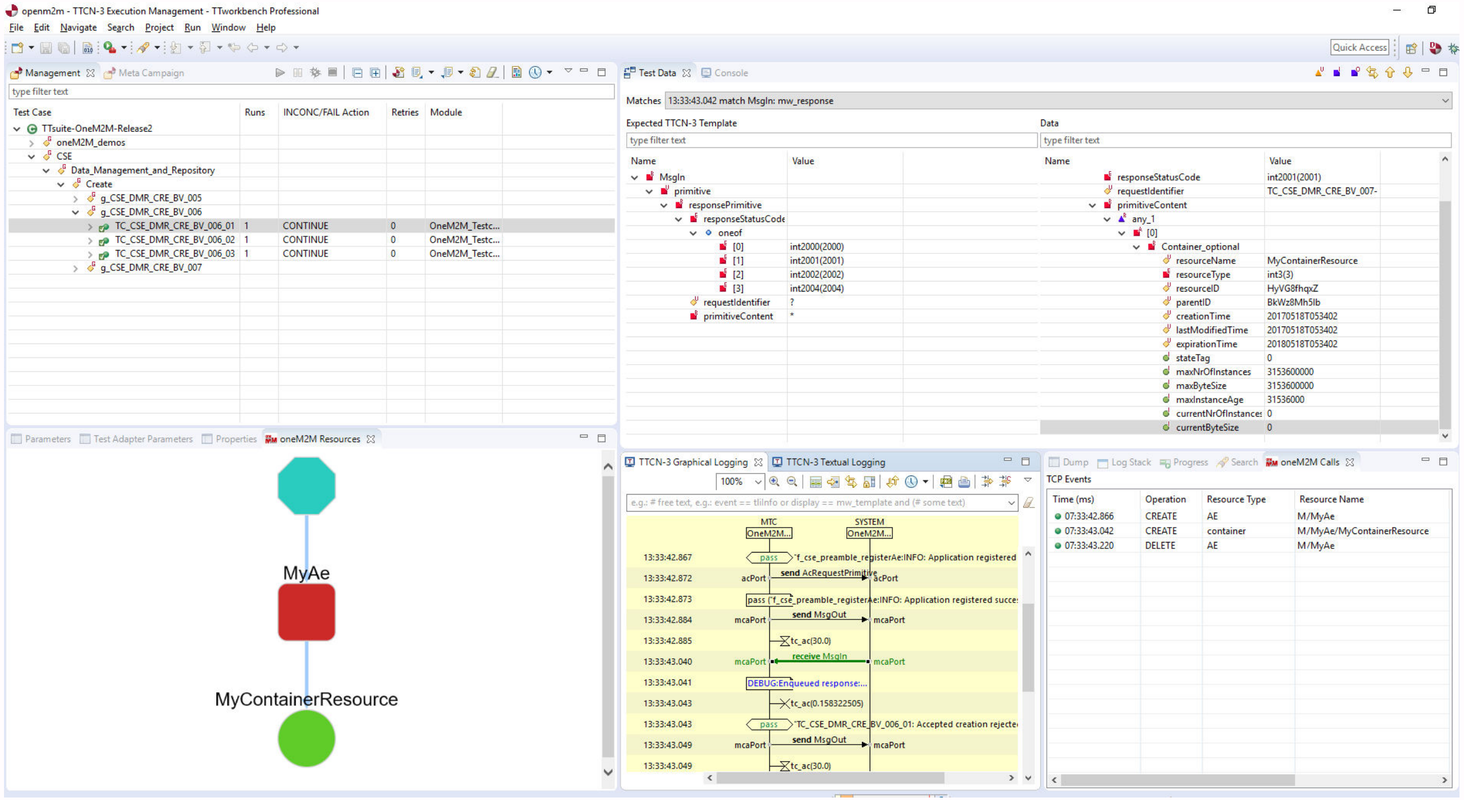
- Simple configuration
- Reusable

```
# Use official node as base image.
FROM node:carbon

# Install the latest app
WORKDIR /root/app
COPY app/* ./

RUN npm install

# Expose the ports used by application
# 3000 HTTP
EXPOSE 3000

ENTRYPOINT [ "npm", "start" ]
CMD [ "127.0.0.1:4141" ]
```

Docker features match our needs

- Portable
- Hide the configuration from users
- Fast container start/restart
- Consistent SUT configuration, host OS and libraries
- Multiple instances possible

Manual Test Execution


openm2m - TTCN-3 Execution Management - TTworkbench Professional

File Edit Navigate Search Project Run Window Help

Management | Meta Campaign

Test Case	Runs	INCONC/FAIL Action	Retries	Module
TTsuite-OneM2M-Release2				
oneM2M_demos				
CSE				
Data_Management_and_Repository				
Create				
g_CSE_DMR_CRE_BV_005				
g_CSE_DMR_CRE_BV_006				
TC_CSE_DMR_CRE_BV_006_01	1	CONTINUE	0	OneM2M_Testc...
TC_CSE_DMR_CRE_BV_006_02	1	CONTINUE	0	OneM2M_Testc...
TC_CSE_DMR_CRE_BV_006_03	1	CONTINUE	0	OneM2M_Testc...
g_CSE_DMR_CRE_BV_007				

Parameters | Test Adapter Parameters | Properties | oneM2M Resources

Test Data | Console

Matches 13:33:43.042 match MsgIn: mw_response

Expected TTCN-3 Template

Name	Value
MsgIn	
primitive	
responsePrimitive	
responseStatusCode	
oneof	
[0]	int2000(2000)
[1]	int2001(2001)
[2]	int2002(2002)
[3]	int2004(2004)
requestIdentifier	?
primitiveContent	*

Data

Name	Value
responseStatusCode	int2001(2001)
requestIdentifier	TC_CSE_DMR_CRE_BV_007-
primitiveContent	
any_1	
[0]	
Container_optional	
resourceName	MyContainerResource
resourceType	int3(3)
resourceID	HyVG8fhqxZ
parentID	BkWz8Mh5lb
creationTime	20170518T053402
lastModifiedTime	20170518T053402
expirationTime	20180518T053402
stateTag	0
maxNrOfInstances	3153600000
maxByteSize	3153600000
maxInstanceAge	31536000
currentNrOfInstances	0
currentByteSize	0

TTCN-3 Graphical Logging | TTCN-3 Textual Logging

```
                 MTC                 SYSTEM
                 OneM2M...           OneM2M...
13:33:42.867             pass  'f_cse_preamble_registerAe:INFO: Application registered
13:33:42.872     acPort  send AcRequestPrimitive  acPort
13:33:42.873             pass ('f_cse_preamble_registerAe:INFO: Application registered succe
13:33:42.884     mcaPort send MsgOut      mcaPort
13:33:42.885             tc_ac(30.0)
13:33:43.040     mcaPort receive MsgIn    mcaPort
13:33:43.041             DEBUG:Enqueued response:...
13:33:43.043             tc_ac(0.158322505)
13:33:43.043             pass  'TC_CSE_DMR_CRE_BV_006_01: Accepted creation rejecte
13:33:43.049     mcaPort send MsgOut      mcaPort
13:33:43.049             tc_ac(30.0)
```

Dump | Log Stack | Progress | Search | oneM2M Calls

TCP Events

Time (ms)	Operation	Resource Type	Resource Name
07:33:42.866	CREATE	AE	M/MyAe
07:33:43.042	CREATE	container	M/MyAe/MyContainerResource
07:33:43.220	DELETE	AE	M/MyAe

Jenkins

- Automation server
  - Used for Continuous Integration and Continuous Delivery
  - Distributed
  - Extensible
  - Huge community

Automated Test Execution using Jenkins

- Execute regression tests
- Start multiple configurations in parallel
- Analyze the test results

Visualize the results

Automation is great but how to analyze this?

700 tests *

12 configurations *

3 Releases

TC	HTTP JSON	CoAP JSON	MQTT JSON	WS JSON	HTTP XML
TC_1	✓	✓	✓	✓	✓
TC_2	✓	✓	✓	✗	✓
TC_3	✓	✓	✓	✓	✓
TC_4	✗	✓	✓	✓	✓
TC_5	✓	✗	✓	✗	✗
TC_6	✗	✓	✓	✓	✓
TC_7	✗	✓	✓	✓	✗

Interpreting the results

What do we identify here?

Special problem spots or even single failing tests are identified

TC	HTTP JSON	CoAP JSON	MQTT JSON	WS JSON	HTTP XML
TC_1	✓	✓	✓	✓	✓
TC_2	✓	✓	✓	✗	✓
TC_3	✓	✓	✓	✓	✓
TC_4	✗	✓	✓	✓	✓
TC_5	✓	✗	✓	✗	✗
TC_6	✗	✓	✓	✓	✓
TC_7	✗	✓	✓	✓	✗

Interpreting the results

- Where is the issue?
  - Configuration
  - Adaptation
  - DUT

TC	HTTP JSON	CoAP JSON	MQTT JSON	WS JSON	HTTP XML
TC_1	✗	✓	✓	✓	✓
TC_2	✗	✓	✓	✓	✓
TC_3	✗	✓	✓	✓	✓
TC_4	✗	✓	✓	✓	✓
TC_5	✗	✓	✓	✓	✓
TC_6	✗	✓	✓	✓	✓
TC_7	✗	✓	✓	✓	✓

Interpreting the results

- Where is the issue?
  - Test case

TC	HTTP JSON	CoAP JSON	MQTT JSON	WS JSON	HTTP XML
TC_1	✓	✓	✓	✓	✓
TC_2	✓	✓	✓	✓	✓
TC_3	✓	✓	✓	✓	✓
TC_4	✓	✓	✓	✓	✓
TC_5	✗	✗	✗	✗	✗
TC_6	✓	✓	✓	✓	✓
TC_7	✓	✓	✓	✓	✓

Problems solved?

- Reproduceable setup on all machines
- Faster execution due automated parallel execution
- Visual analysis of the results highlight hotspots
- Faster feedback to the development teams

Future work

- Stress tests
- Testing the oneM2M application in the cloud
- Complex scenarios

6th UCAAT User Conference on Advanced Automated Testing

Paris, 16-18 October 2018

Organizer: TESTING SOLUTIONS & SERVICES

Thank you!

Questions?