IoT Middleware
Thursday June 11th, 15
Lionel Florit, Principal Engineer (CTAO),
lflorit@cisco.com
@LFlorit
Agenda

- Introduction to an IoT Middleware, technology overview
  - Where it fits, working principles
  - The oneM2M example
- Building a multiparty IoT Application
- From Theory to Code
  - Open Source implementation
- Examples and Demo
“The push to create an “Internet of Things” (IoT) will fail unless electronics firms collaborate more”

Samsung’s CEO, CES2015  BK Yoon.
Challenge #1: Many Devices, Many Applications, Many Protocols, So Little Time…
Challenge #2: IoT Solution Lock-in

Body Cameras Quick Facts:

- in the US, funding for 50,000 cameras, list price ~ $400
- 5 years archives
- Largest deployment to date: Oakland 600 cameras
The Goal

• Connect any device to any application

• Minimize the integration efforts
  • As little custom code as possible
  • As little interaction with device and software manufacturers as possible

• Get as close as possible to plug and play

• Maintain a flexible solution which fits the use case
Several Ways to Build IoT Solutions

• **Point Solutions**
  • Address a single problem well with custom made devices & software

• **Closed Systems**
  • Qualify a set of devices and integrate them into a solution
  • Single company

• Form an small team of vendors to address a set of problems

• Form an eco-system for your vertical
Internet of Things Reference Model

Levels

6 Application
(Reporting, Analytics, Control)

5 Data Abstraction
(Aggregation & Access)

4 Data Accumulation
(Storage)

3 Data Element
(Analysis & Transformation)

2 Connectivity
(Comm & Processing Units)

1 Physical Devices
(The “Things” in IoT)

Vertical Industry Solutions

Targeted Solutions

Focus Here

Data at Rest

Data in Motion

Sensors, Devices, Machines, Intelligent Edge Nodes of all types
Enters the IoT MiddleWare

Standard APIs
Protocol Plugins

Data Repository

Data Reasoning

Create, Retrieve, Update, Delete, Notify

IoT Apps

Standard and Published APIs

CRM

ERP

Devices
IoT Middleware(s)

IoT Applications

IoT Services

Transport

Link
Agenda

• What is IoT Middleware, technology overview
  • Where it fits, working principles
  • The oneM2M example
• Building a multiparty IoT Application
• From Theory to Code
  • Open Source implementation
• Examples and Demo
oneM2M Background

- Partnership project founded by 7 SDOs to minimize standards fragmentation
  - USA: TIA, ATSI
  - Japan: ARIB, TTC
  - China: CCSA
  - Europe: ETSI
  - Korea: TTA

- Focus on M2M horizontal common services

- Founders agreed to transfer & stop their own M2M service layer work
oneM2M

Scope

Common Services Entity (CSE)
- Data Exchange
- Remote Device Management
- Security & Access control
- Connectivity Handling

Underlying Network provides value added services to the CSEs. Such as QoS, device management, location services and device triggering.
oneM2M Common Service Functions

Application Entity (AE)

Mca Reference Point

Common Services Entity (CSE)

- Application and Service Layer Management
- Data Management & Repository
- Location
- Security
- Communication Management/Delivery Handling
- Registration
- Service Session Management
- Device Management
- Subscription Notification
- Service Charging & Accounting
- Discovery
- Network Service Exposure/Service Ex+Triggering
- Group Management

Mcc Reference Point

Mcn Reference Point

Underlying Network Service Entity (NSE)
oneM2M Functional Architecture

- **Application Service Nodes**: smart oneM2M devices
- **Common Services Entity (CSE)**: sets of "service functions"
- **Application Entity (AE)**: provides Application logic
- **Infrastructure nodes**: servers
- **Middle Nodes**: gateways
- **Application Dedicated Nodes**: dumb oneM2M devices
- **Non-oneM2M devices**
RESTful Architecture

- All entities represented in a tree as “Resources”:
  - Applications
  - Devices
  - Data
  - Groups
  - Access Rights
  - Billing Policies, etc.

- Resources are manipulated using CRUD verbs
Did you say “orange”?  
Instead of dictionaries, machines have online locations which disambiguate what they read

- Humans can usually tell which “orange” we are talking about

**orange** [awr-inj, or-]  
*Show IPA*  
noun
1. a globose, reddish-yellow, bitter or sweet, edible citrus fruit.
2. any white-flowered, evergreen citrus trees of the genus *Citrus*, bearing this fruit, as *C. aurantium* (*bitter orange, Seville orange, or sour orange*) and *C. sinensis* (*sweet orange*) cultivated in warm countries.
3. any of several other citrus trees, as the *trifoliate orange*.
4. any of several trees or fruits resembling an orange.
5. a color between yellow and red in the spectrum, an effect of light with a wavelength between 590 and 610 nm; reddish yellow.
Examples of Machine “Dictionaries”

<table>
<thead>
<tr>
<th>Name/Org</th>
<th>Topic</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>QUDT - Quantities, Units, Dimensions and Data Types Ontologies (NASA)</td>
<td>Anything that can be measured</td>
<td><a href="http://www.qudt.org">http://www.qudt.org</a></td>
</tr>
<tr>
<td>W3C Semantic Sensor Networks</td>
<td>Terms to describe sensors</td>
<td><a href="http://purl.oclc.org/NET/ssnx/ssn#">http://purl.oclc.org/NET/ssnx/ssn#</a></td>
</tr>
<tr>
<td>Open Biological and Biomedical Ontologies (OBO) Foundry</td>
<td>a wiki space for collecting science based ontologies</td>
<td><a href="http://www.obofoundry.org">http://www.obofoundry.org</a></td>
</tr>
<tr>
<td>CHEBI</td>
<td>provides a URI for every chemical it defines (20 000!)</td>
<td><a href="http://www.obofoundry.org">http://www.obofoundry.org</a></td>
</tr>
<tr>
<td></td>
<td>geographical data (city, state, country)</td>
<td>schema.org</td>
</tr>
</tbody>
</table>

http://www.qudt.org/qudt/owl/1.0.0/quantity/Instances.html#Area

**quantity:Area: Area**

Area is a quantity expressing the two-dimensional size of a defined part of a surface, typically a region bounded by a closed curve.
Mapping URIs to Data Types using Ontologies

URI Refers to this

How do we get this & from where?

SSN Ontology

Things

Data

IoTDM

Cisco Live!  
Ref: oneM2M

URI: http://purl.oclc.org/NET/ssnx/ssn#frequency

Label: Frequency

Subclass of: ssn:MeasurementProperty

Paraphrase: A ssn:Frequency is something that is a ssn:MeasurementProperty (experimental)
Security: Mechanisms Needed

• Getting **access** to the network
  • Authentication

• Communicate securely
  • **Secure communication** (TLS/DTLS, secure networks, VPNs…)

• **Authenticates** to the Middleware
  • Authentication of the devices (PSK, Certificates)

• **Authorization** to do CRUDs
  • Access control policies
What devices can do and how secure the solution should be?

<table>
<thead>
<tr>
<th>Class 0</th>
<th>Sec Level 1</th>
<th>Sec Level 2</th>
<th>Sec Level 3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pre-Shared Symmetric keys serialized at manufacturing time</td>
<td>Authentication based on Pre-shared keys (ACLs)</td>
<td>Pre-Shared Key Encryption</td>
</tr>
<tr>
<td>Class 1</td>
<td>Serialized Certificates at manufacturing time</td>
<td>Certificate Authentication enables secure token and Symmetric key</td>
<td>Encryption based on symmetric key, Authorization based on Token</td>
</tr>
<tr>
<td>Class 2</td>
<td>PKI Authentication</td>
<td>Certificate Authentication enables secure token and Symmetric key</td>
<td>Encryption based on symmetric key, Authorization based on Token</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>data size (e.g., RAM)</th>
<th>code size (e.g., Flash)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 0, C0</td>
<td>&lt;&lt; 10 KiB</td>
<td>&lt;= 100 KiB</td>
</tr>
<tr>
<td>Class 1, C1</td>
<td>~ 10 KiB</td>
<td>~ 100 KiB</td>
</tr>
<tr>
<td>Class 2, C2</td>
<td>~ 50 KiB</td>
<td>~ 250 KiB</td>
</tr>
</tbody>
</table>

*Table 1: Classes of Constrained Devices (KiB = 1024 bytes)*

RFC-7228
Authorization Architecture
Resource access is authorized upon satisfying at least one ACP rule in one of the linked ACPs.
oneM2M Middleware Operations

[Device Turns On]

IP + Mgt_Srv

Credentials (From Factory or onboard App)

Validation Reg

Token

Checks Credentials & Subscriptions

Key or Token + Config + DDM IP

Start [Id]

OK

DDM_Reg

DDM_DataStore
Agenda

• What is IoT Middleware, technology overview
  • Where it fits, working principles
  • The oneM2M example

• Building a multiparty IoT Application

• From Theory to Code
  • Open Source implementation

• Examples and Demo
Application Overview

Location data from multiple sources: GPS, IBeacon, WIFI
The Planning Phase

I know about backends

I know about WIFI
And CMX APIs

I know about smart phones

I know about IoT Middleware
Who Does What?

Client

oneM2M API

IoT MiddleWare Platform

CMX Plugin

oneM2M API

CMX API

Nothing to do here
If a client needs to locate a thing, created a resource here.

The location of all things are stored here.

Maps and other helper data are stored here.

- Root
- LocationA
- FinderOutBox
- Things
- MapRepo
- MAC
- X
- Y
- MapID
- MAC Add Of Phone 1
- MAC Add Of Phone 2
- BeaconID 1
- BeaconID 2
- MapID 1

If a client needs to locate a thing, created a resource here. The location of all things are stored here. Maps and other helper data are stored here.
IoT Client App
I want to know where IP Add X is

Create root/LocationAE/FinderOutBox/IP_X
IoT App needs to know 2 things:
1- Objects locations if known is stored in root/LocationAE/Things
2- If the location doesn’t exist post the request to root/LocationAE/FinderOutBox
BostonU Positioning Mobile App
Building on Top: the VM Follow-me App
The Planning Phase II

I know about IoT Middleware
I know about Wi-Fi and CMX APIs
I know about phones
I know about reasoning
I know about backends
I know about Networking
I know about virtualization
Who Does What?

Reasoner (decides if/when/where to move the VM)

IoT MiddleWare Platform

Client (Monitors Compute resources)

oneM2M API

SNMP Plugin

CMX Plugin

Report Network conditions

CMX API

oneM2M API

SNMP
VM Follow Me Resource Tree

- Root
  - VMFollowMe
  - LocationAE
    - Network
      - Path1
      - Path2
      - Path3
        - Latency
        - Packet Loss
        - Avg Util
      - Compute
        - Server1
        - Server2
        - Server3
          - CPU
          - Mem
          - Disk
          - Processes
    - Things
      - VM – Thing Pairing
Agenda

• What is IoT Middleware, technology overview
  • Where it fits, working principles
  • The oneM2M example
• Building a multiparty IoT Application
  • From Theory to Code
    • Open Source implementation
• Examples and Demo
Open Plugin Architecture

Enable Ecosystem of App Developers

Enable Ecosystem of Device Manufacturers

Enable 3rd Party Cloud as data Client

Enable 3rd Party Cloud as data Provider

Backward Compatibility (e.g. proprietary) & future-proofing
IoTDM at the Linux Foundation

- Project started in December 2014
- Goal:
  - open source IoT Middleware over OpenDaylight
  - Based on oneM2M
  - Basic set of resources supported
- Delivery: June 2015, Lithium release
- [https://wiki.opendaylight.org/view/IoTDM:Main](https://wiki.opendaylight.org/view/IoTDM:Main)
Overall Architecture

UCS

Data Collection (IoTDM)

Data Store

ODL APIs (ReST)

Service Abstraction Layer
- Plugin Manager
- Capability Abstractions
- Inventory

ODL Service Modules
- Data Collection (IoTDM)
- Other ODL Services

Legacy DB (Oracle, MSN, …)

Vertical IoT Apps (Integration if needed)

SQL

ReST HTTP

CoAP

HTTP

CoAP

Other IoT Protocols

Plugins to Cisco Apps

Cisco Devices & Services

IoT App

Existing IoT Mgmt Systems

Devices and Things

IoTDM-ODL Open Source

JSON

OAuth

PKI

CoAP

TLS/DTLS

HTTP

CoAP

Other IoT Protocols

Cisco Devices & Services

© 2015 Cisco and/or its affiliates. All rights reserved. Cisco Public

47
The Building Blocks: oneM2M aware Client

Client

- IoT App
  - Programing API (Python, Java, C, etc...)
- Device Plugin
- Protocol Binding
- CoAP/H TTP/…

Middleware: ODL-IoTDM

- Data Store
- Core Functions
- RPC Call/Resp
- Protocol Plugin
- Protocol Binding
- CoAP/H TTP/…

JSON Payload
{"cr":"jb", "mni":"1", "mbs":"3", "or":"foo.bar.org"}
Example of oneM2M JSON Message

POSTMAN

POST CSE-PROVISIONING

POST HTTP_CreateContainer

DELETE HTTP_Delete

GET HTTP_GET

POST HttpCreatAE

HTTP_CreateContainer

http://localhost:8282/inCSE1/BILL?ty=cnt&rcn=1

<table>
<thead>
<tr>
<th>ty</th>
<th>cnt</th>
</tr>
</thead>
<tbody>
<tr>
<td>rcn</td>
<td>1</td>
</tr>
</tbody>
</table>

URL Parameter Key | Value
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>form-data</td>
<td>x-www-form-urlencoded</td>
</tr>
</tbody>
</table>

```json
{"cr":"jb", "mnl":"1", "mbs":"3", "or":"http://hey/you"}
```
The Building Blocks: Client Unchanged

**Client**
- IoT App
- Programing API (Python, Java, C, etc…)
- Device Plugin
- Protocol Binding
- Any Protocol

**Middleware: ODL**
- Data Store
- Core Functions
- RPC Call/Resp
- Proprietary Plugin
- Protocol Binding
- Any Protocol

Any Payload
Open for contributions

Where can I Contribute?

Client

IoT App
Programing API
(Python, Java, C, etc.)
Device Plugin
Protocol Binding
CoAP/H TTP/…

Contributions are welcome!

Middleware: ODL

Core Functions
RPC Call/Resp
Protocol Plugin
Protocol Binding
CoAP/H TTP/…

JSON Payload
{"cr":"jb", "mni":"1", "mbs":"3", "or":"foo.bar.org"}
Where do I start?

- I just want to send CRUDs, build a tree, see how it works
  - Go to developer.cisco.com, open an account, navigate to the IoTDM sandbox

- I am a device developer, I want to test my device against the IoT Middleware
  - I don’t want to install the middleware: Go to developer.cisco.com
  - I want to install the middleware locally: Go to the IoTDM project page, https://wiki.opendaylight.org/view/IoTDM:Main find the instruction to download IoTDM

- I am a Java developer and I want to contribute to the development of the IoT middleware
  - Go to the IoTDM project page, https://wiki.opendaylight.org/view/IoTDM:Main find the developers instruction
Examples and Demo
VM Follow Me Demo
Challenge #2: IoT Solution Lock-in (revisited)
MiddleWare and… Quantum Service Bus

I need to transfer more than X bytes to IMSI_X, is now a good time?

QSB Services exposed
- Location
- Roaming Notification
- Mobility Notification
- Network conditions (congestion notifications)
- Short messages
- Bandwidth and QoS on demand

### SUL level | Congestion Status
--- | ---
5 | High Congestion
4 | Medium Congestion
0-3/No SUL | No Congestion
Summary

• Introduced the role of IoT Middleware.
• We talked about its operations using oneM2M as an example
• Looked at how Multiparty IoT Applications are created
• Reviewed a proof of concept implementation example
Conclusion

• Collecting IoT data for an application is not difficult, getting everybody to do it the same way is.

• To enable any device to communicate with any application, the architecture needs to be open so many vendors will join

• We started with oneM2M as a middleware platform

• Other middleware can be added in the future

• Open daylight as an application development platform
Participate in the “My Favorite Speaker” Contest
Promote Your Favorite Speaker and You Could Be a Winner

- Promote your favorite speaker through Twitter and you could win $200 of Cisco Press products (@CiscoPress)

- Send a tweet and include
  - Your favorite speaker’s Twitter handle @LFlorit
  - Two hashtags: #CLUS #MyFavoriteSpeaker

- You can submit an entry for more than one of your “favorite” speakers

- Don’t forget to follow @CiscoLive and @CiscoPress

Complete Your Online Session Evaluation

• Give us your feedback to be entered into a Daily Survey Drawing. A daily winner will receive a $750 Amazon gift card.

• Complete your session surveys though the Cisco Live mobile app or your computer on Cisco Live Connect.

Don’t forget: Cisco Live sessions will be available for viewing on-demand after the event at CiscoLive.com/Online
Continue Your Education

- Demos in the Cisco campus
- Walk-in Self-Paced Labs
- Table Topics
- Meet the Engineer 1:1 meetings
- Related sessions
Thank you
TOMORROW starts here.