物聯網框架標準

廖冠雄 助理教授
義守大學 資訊工程學系
e-mail：ghliaw@isu.edu.tw

大綱

- 前言
- Service Framework
- Architectural Framework

Requirements for IoT Architecture

- Interoperability
  - In the way in which software interacts with physical resources
  - Decouple IoT devices from the software that manages them
  - Discovery, Management and Reporting, Security, Authorization
- Scalability
  - Large number of devices, users, interactions, connections
  - Scale-less interaction

Requirements for IoT Architecture

- Technology Reuse and Modularity
  - Software, networks, protocols, data models
  - Across vendors in a vertical application segment
  - Across diverse vertical application segments
- Low Barrier to Innovation
  - Anyone can participate and innovate

IoT Service Frameworks

- The goal – providing same qualities as web services to IoT

- In the context of the Internet and the WWW
  - Any Web Browser works with any web service, more or less
  - Regardless of wire protocols used in the network
  - Regardless of the data models and content types
  - Across vertical application segments
  - The WWW scales to planetary size
  - Low barrier to Innovation

IoT Service Frameworks

- Internet and WWW Design Patterns
  - Narrow Waist, endpoint oriented
    - Innovation happens at the endpoints, enabled by common, openly available network protocols (the narrow waist)
  - Layered Protocols
    - Common set of IP protocols (TCP, UDP) abstract the lower communication layers
    - Common Application Protocols (HTTP, REST) abstract resources

IoT Service Frameworks

- Internet and WWW Design Patterns
  - Uniform Addressing
    - URIs and Hyperlinks point to resources
    - IP Addresses, DNS names are globally unique
  - Stateless Interaction
    - Client-Server pattern
    - Hypermedia As The Engine Of Application State

IoT Service Frameworks

- How Does it Apply to IoT?
  - Internet Protocol (IP) on Constrained Devices
  - Machine to Machine (M2M) Application Protocols
  - Standard Object Models and Data Models
  - Hypermedia for Machine APIs

IoT Architectural Frameworks

- Enabling cross-domain interaction and platform unification
  - system compatibility
  - interoperability
  - functional exchangeability
- Covering the architectural needs of the various IoT Application Domains

In this course, we briefly intro …

- Service Frameworks
  - Allseen AllJoyn
  - OIC Core Framework
  - OMA LWM2M
  - Thread
- Architectural Frameworks
  - oneM2M
  - ETSI M2M
  - IEEE P2413

Allseen AllJoyn (1)

- Allseen Alliance
  - Enable industry standard interoperability between products and brands with an open source framework for IoT
  - Main members
    - Qualcomm, Microsoft, Philips, LG, …

Allseen AllJoyn (2)

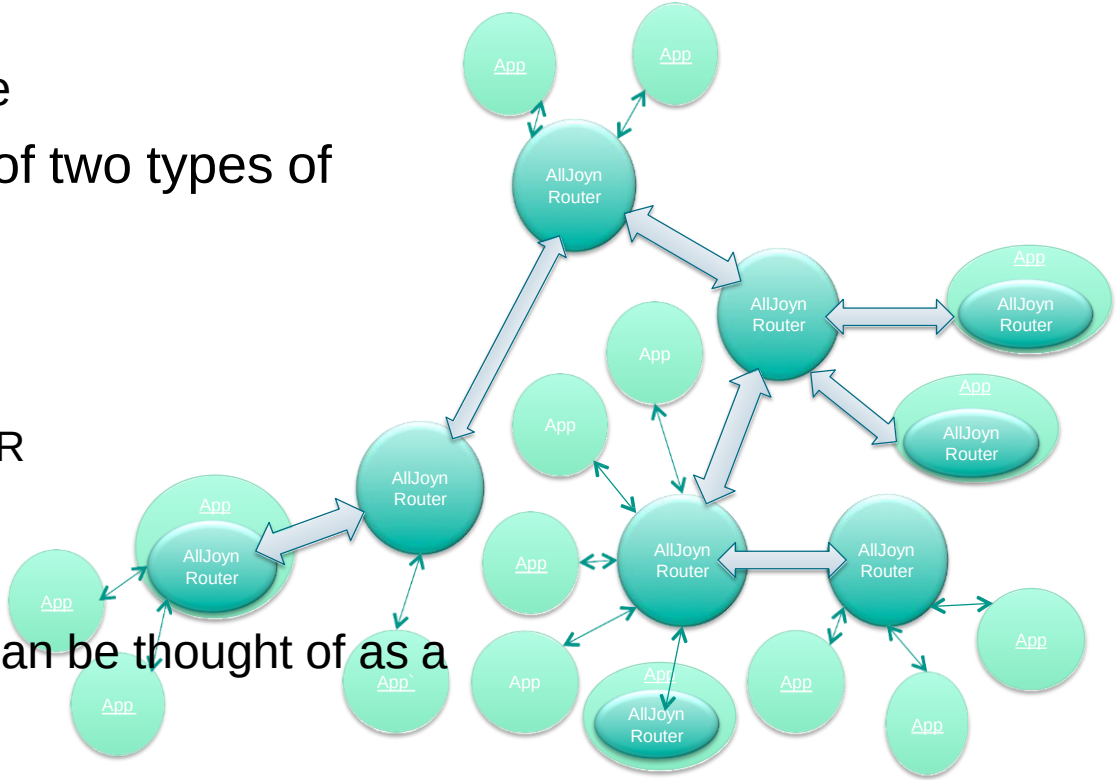
- What is the AllJoyn Framework?
  - An open source API framework for the Internet of Everything
  - A way devices and applications publish APIs over a network in a standard way
  - the “things” on the network expose to other “things”
  - provides application discovery and fine-grained discovery of the APIs supported by applications

Allseen AllJoyn (3)

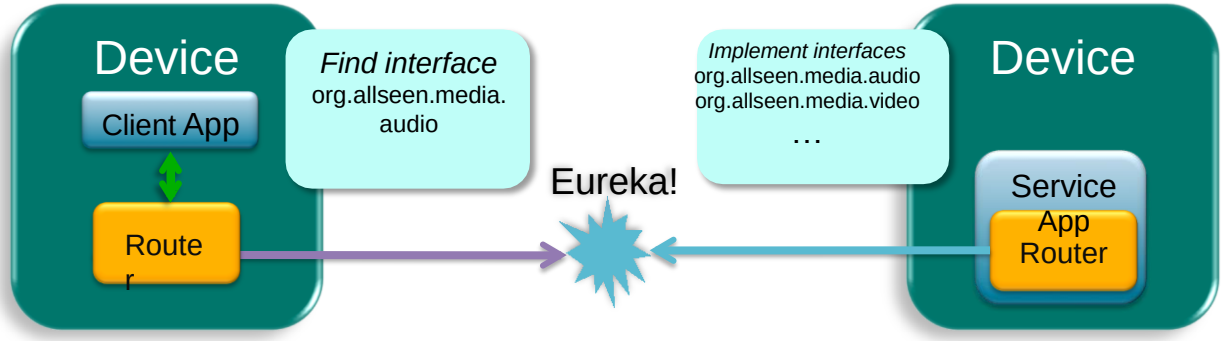
High-Level System Architecture

AllJoyn Bus is composed of two types of nodes:

- AllJoyn Router (R)
- Applications (App)
  - App can only connect to R
  - App can contain R
  - R can connect to other R
- The AllJoyn framework can be thought of as a mesh of stars



Ad Hoc Bus Formation: Discover



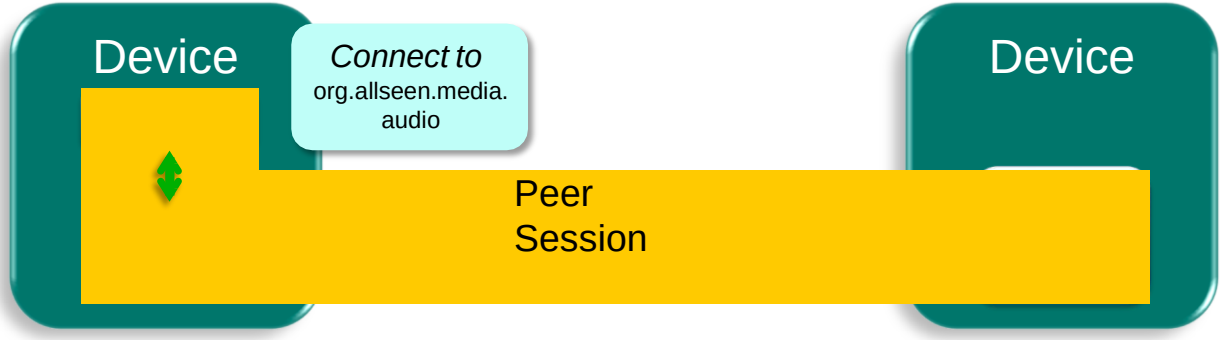
Interface descriptions contained in About message

- Services advertise, and Clients find, About messages
- Connect to advertisers supporting desired interfaces

Actual discovery mechanism is transport-dependent:

- On Wi-Fi, PLC, Ethernet: lightweight IP multicast protocol
- On Wi-Fi Direct: would use pre-association discovery

Ad Hoc Bus Formation: Session Creation



Session creation will cause AllJoyn Routers to connect and extend the bus

- Once connected, buses merge and have a single shared namespace
- Peers can discover when other peers join or leave the bus
- Peers can interact via their APIs
- Session reference counting keeps device-to-device connections alive
- Multicast events can be sent to all peers in the session

AllJoyn Software Framework: High-level architecture

A comprehensive software framework lets devices and applications communicate

AllJoyn App Layer

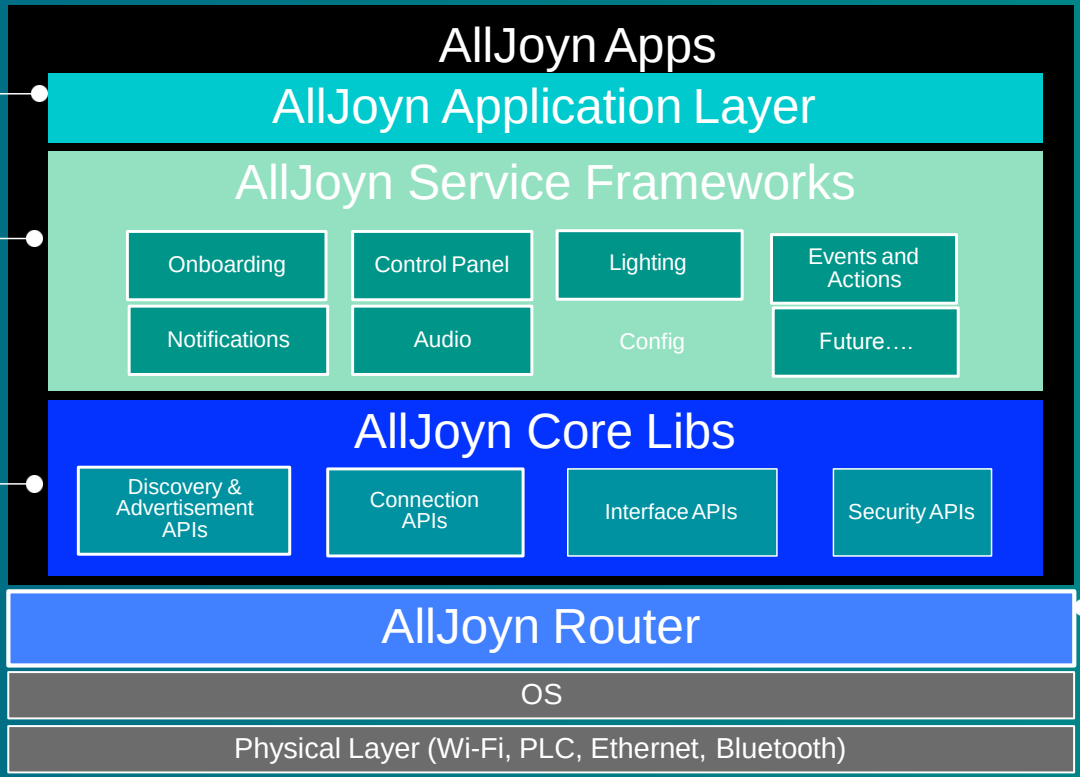
- Defines the User experience

AllJoyn Service Frameworks

- Interoperable, cross-platform modules for common IoE functionality
- Defines common interfaces between devices

AllJoyn Core Libs

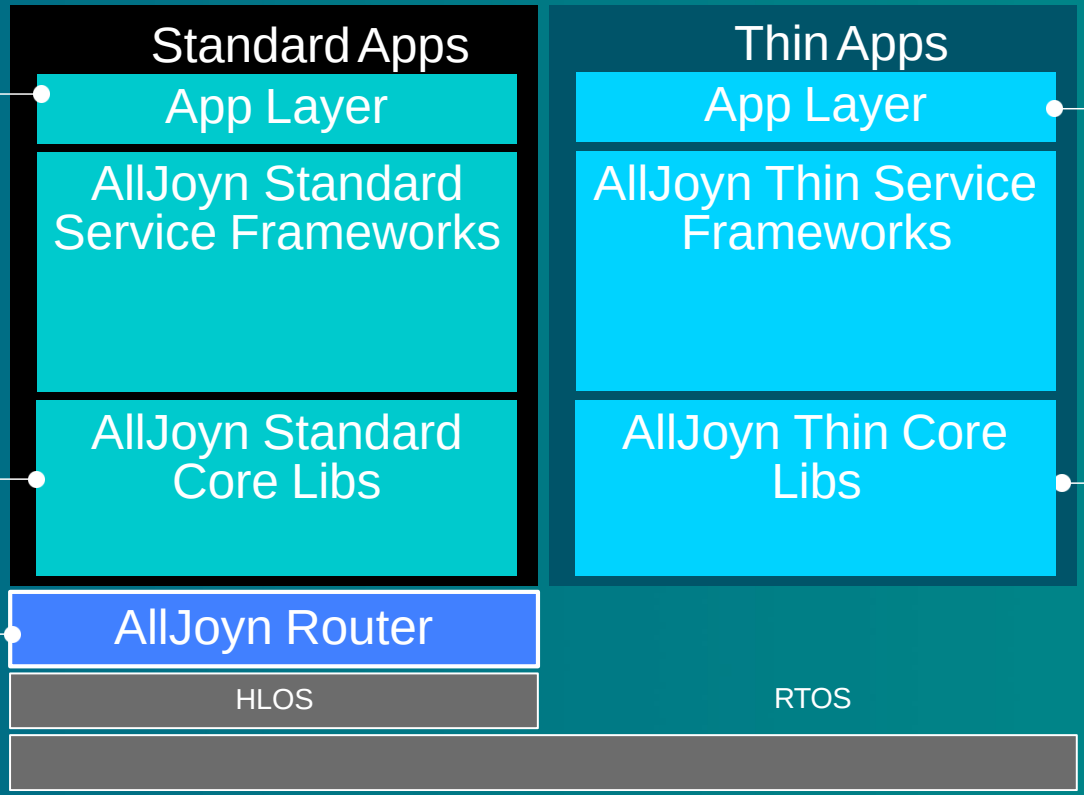
- Provides ability to find and connect to devices to do interesting things.
- Core libraries interact with the AllJoyn Router
- Provides access control and encryption



AllJoyn Router

- Manages communications between devices and apps
- Dynamic network management

Two Versions of the AllJoyn Framework To Choose



Standard App Layer

- App dev or OEM writes this

Standard Core Libraries

- Multiple bindings, runs on HLOS

AllJoyn Router

- AllJoyn Router can be bundled with a Standard App or run standalone

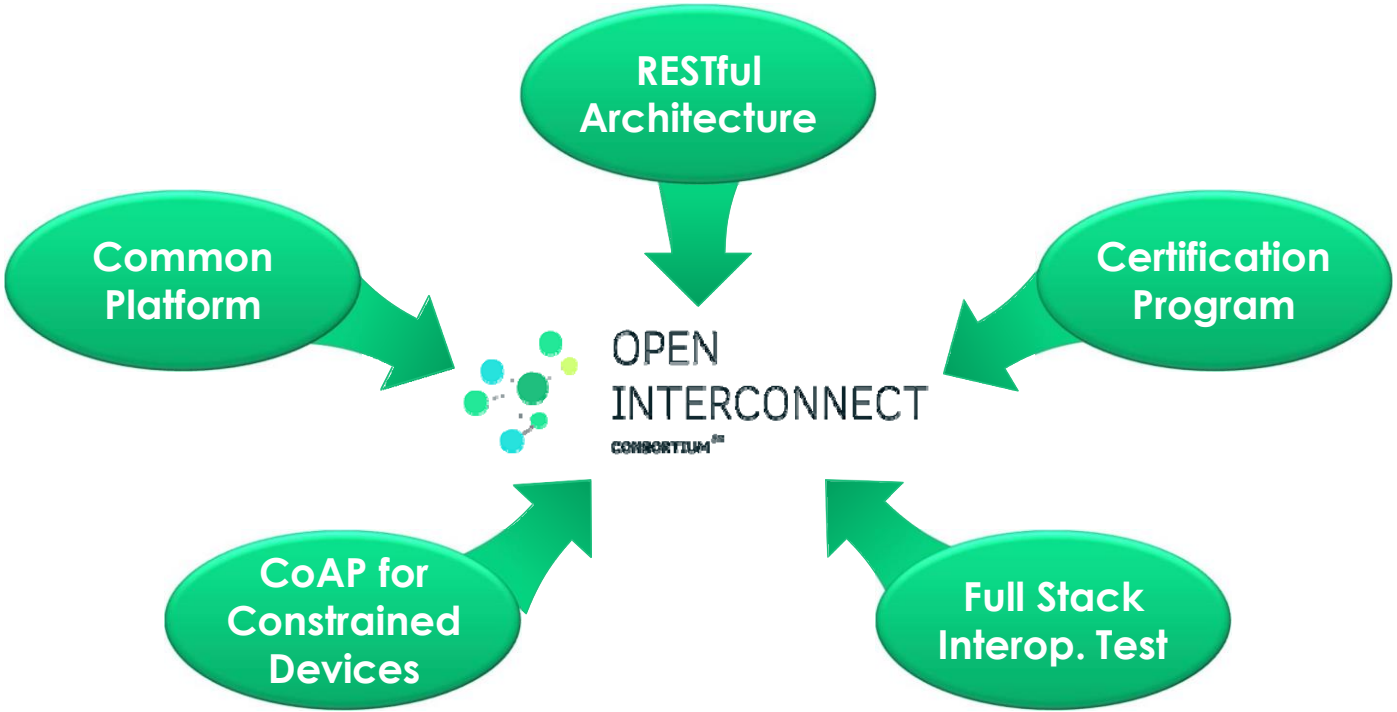
Thin App Layer

- OEM writes this

Thin Core Libraries

- C bindings, runs on RTOS
- Thin Apps using Thin Core requires an AllJoyn Router in the network

Introduction to OIC – Optimized for IoT



OIC Key Concepts (1/2)

- **Open-source Implementation – Iotivity project** (https://www.**iotivity**.org/)
- **Free IPR License** (Code: Apache 2.0 & Spec: RAND-Z)
  - License covers both code, standards and related IPR
  - License applies to members and affiliates of members
- **Dedicated and optimized protocols for IoT** (e.g. CoAP)
  - Specific considerations for constrained devices
  - Fully compliant towards RESTful architecture
  - Built-in discovery and subscription mechanisms
- **Standards and Open Source to allow flexibility creating solutions**
  - Able to address all types of devices, form-factors, companies and markets with the widest possibility of options
  - Open Source is just one implementation to solve a problem

OIC Key Concepts (2/2)

- **Full stack definition for maximum interoperability**
  - Connectivity, Platform and Vertical Services defined
  - License applies to members and affiliates of members
- **Certification and Logo program**
  - Guarantees all devices work together
  - Consistent user awareness for interoperability

Sample of Current Members

Diamond

ARRIS, CableLabs, cisco, GE Software, intel, SAMSUNG

Platinum

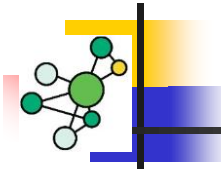
Atmel, WOX, CAICT 中国信息通信研究院, DELL, Honeywell, IBM, MEDIATEK, ZTE

Gold

accenture, acer, AEPONA, ALLION, alticast, Astoria networks, asurion, ASUS, avempace wireless, Axstone TECHNOLOGY, B&O, BigJungle, BONTOM, BUFFALO, cadence, ESI, cryptosoft everything secured, ConnectingYourThings, D+M GROUP, DigiOn, D-Link, oppo, DSP GROUP, Dr&C, DTT TECHNOLOGY GROUP, ECS, ENCORED, ETRI, FIH 富智康 FIH Mobile Limited, fine point technologies, IB firstbuild, fuse, GCR, GoPro Be a HERO, GRL GRANITE RIVER LABS, GrowSafe SYSTEMS, HANSONG, Hisense, hitron

Objectives

- Core Framework Specification Scope
  - Specifies the technical specification(s) comprising of the core architectural framework, messaging, interfaces and protocols based on approved use-case scenarios
  - Enables the development of vertical profiles (e.g. Smart Home) on top of the core
- Architect a core framework that is scalable from resource constrained devices to resource rich devices
- Evaluate technical specification(s) for maximum testability and interoperability
- Ensure alignment with OIC open source releases



OIC Roles

- OIC Client
  - i) Initiate an transaction (send a request) & ii) access an OIC Server to get a service
- OIC Server
  - i) host OIC Resource & ii) send a response & provide service

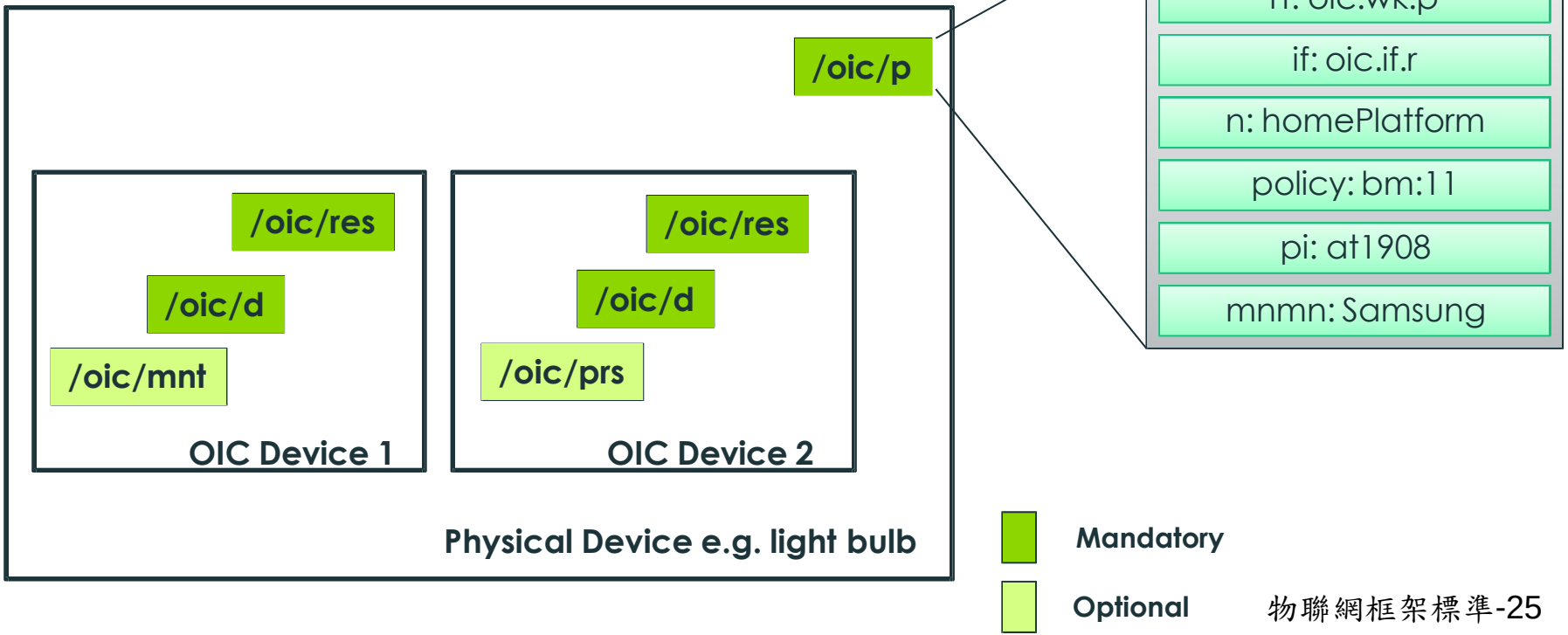
OIC Architecture

- OIC adopted RESTful Architecture
- Current OIC Architecture defines 2 logical roles that devices can take
  - OIC Server : A logical entity that exposes hosted resources
  - OIC Client : A logical entity that accesses resources on an OIC Server

Model 1

Organization of an OIC Device

- OIC Device concept

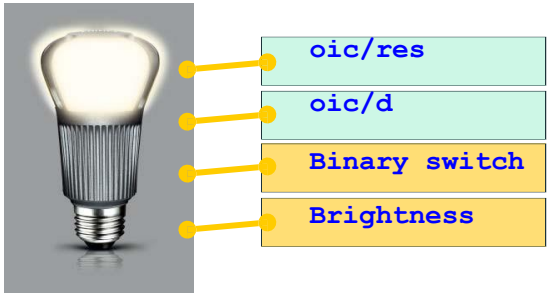


Device example: light device (oic.d.light)

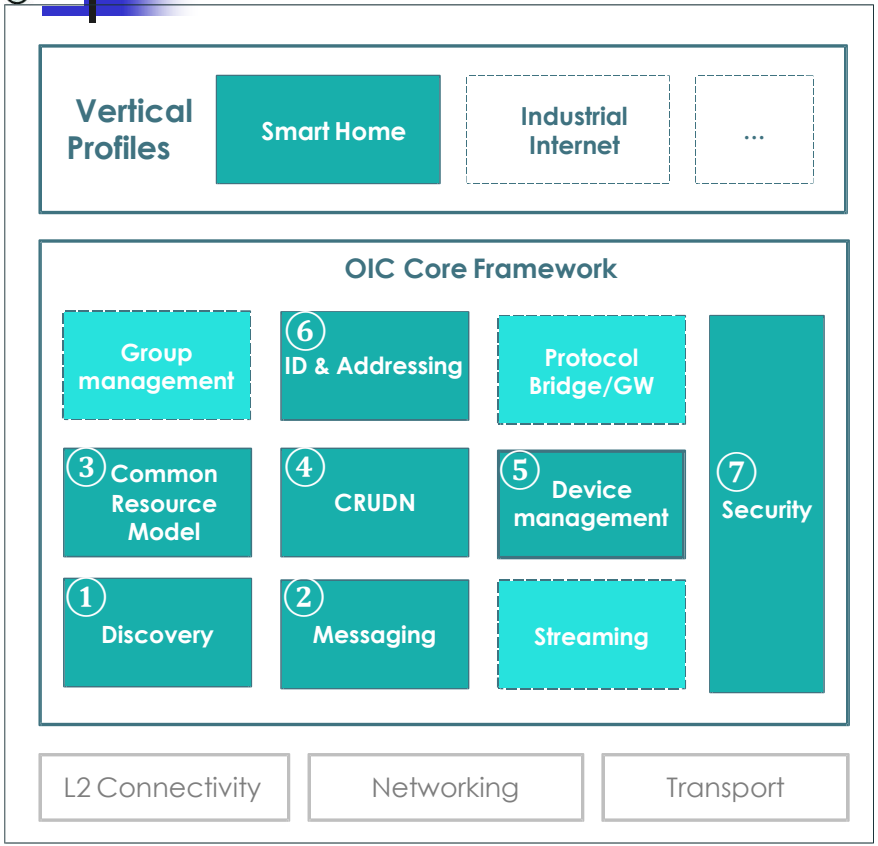
- Example overview
  - Smart light device with i) binary switch & ii) brightness resource
- Device type: Light device (oic.d.light) [Defined by the domain]
- Associated resources
  - Core resources: ① oic/res, ② oic/d
  - Device specific resources: ③ Binary switch (oic.r.switch.binary),
  - Other optional resources can be exposed, in this example ④ Brightness resource (oic.r.light.brightness)

Example: Smart light device with 4 resources

Device Title	Device Type	Associated Resource Type	M/O
Light	oic.d.light	oic/res (oic.wk.core)	M
		oic/d (**oic.d.light**)	M
		Binary switch (oic.r.switch.binary)	M
		Brightness (oic.r.light.brightness)	O



OIC Spec Features – Core Framework Spec



① **Discovery:** Common method for device discovery (IETF CoRE)

② **Messaging:** Constrained device support as default (IETF CoAP) as well as protocol translation via intermediaries

③ **Common Resource Model:** Real world entities defined as data models (resources)\

④ **CRUDN:** Simple Request/Response mechanism with Create, Retrieve, Update, Delete and Notify commands

⑤ **Device Management:** Network connection settings and remote monitoring/reset/reboot functions

⑥ **ID & Addressing:** OIC IDs and addressing for OIC entities (Devices, Clients, Servers, Resources)

⑦ **Security:** Basic security for network, access control based on resources, key management etc

OIC Core Framework Basic Operation

Discovery

- Discover access policies, device info and resources on the devices

Operation

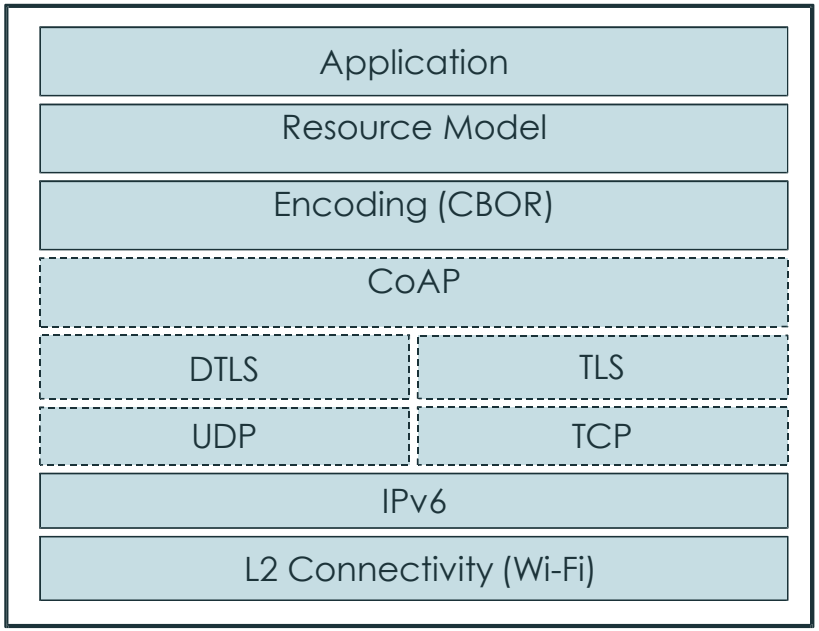
- Get device information by retrieving resources
- Control devices by changing resources
- Observe change on the properties of resources

Basic common services

- Device Monitoring
- Maintenance (e.g., reboot, factory reset, statistics collection, etc.)

Connectivity | Networking | Transport | Security

Protocol Stack



Project B OIC Stack

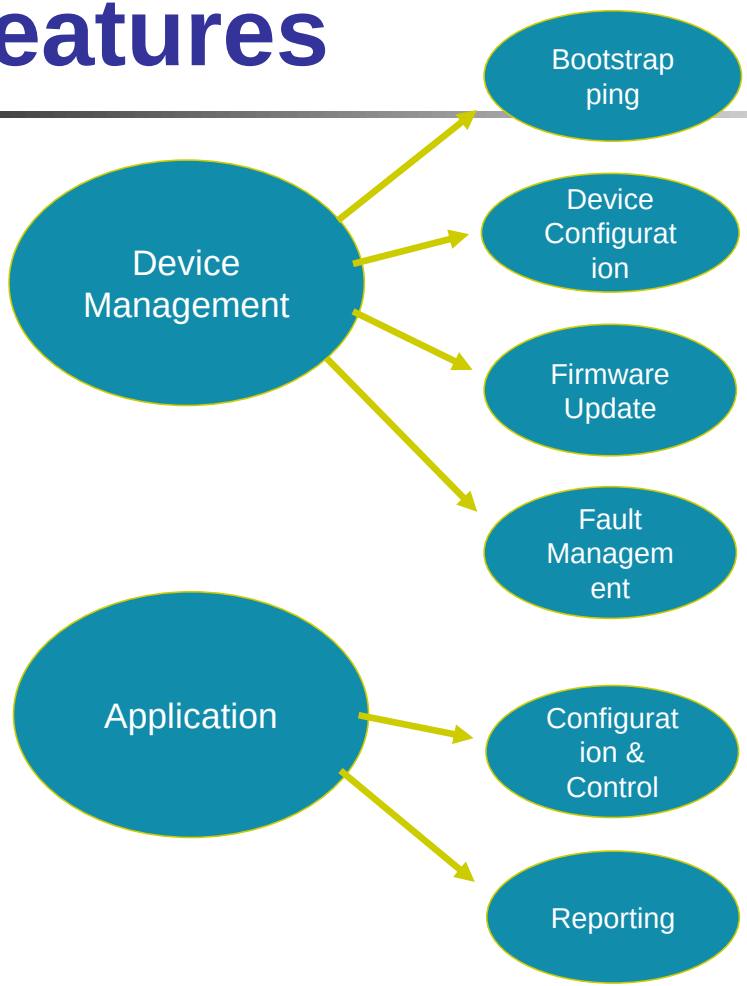
Alternatives

Encoding	JSON or XML/EXI can be negotiated
IP Version	v6 (v4 supported for legacy devices)

OMA LWM2M

- OMA – Open Mobile Alliance
- LWM2M – Light-Weight M2M
  - to serve the IoT market with a focus on lightweight nature

Features



- Key management
- Service provisioning
- Access Control

- Changes to settings
- Changes to parameters of the device

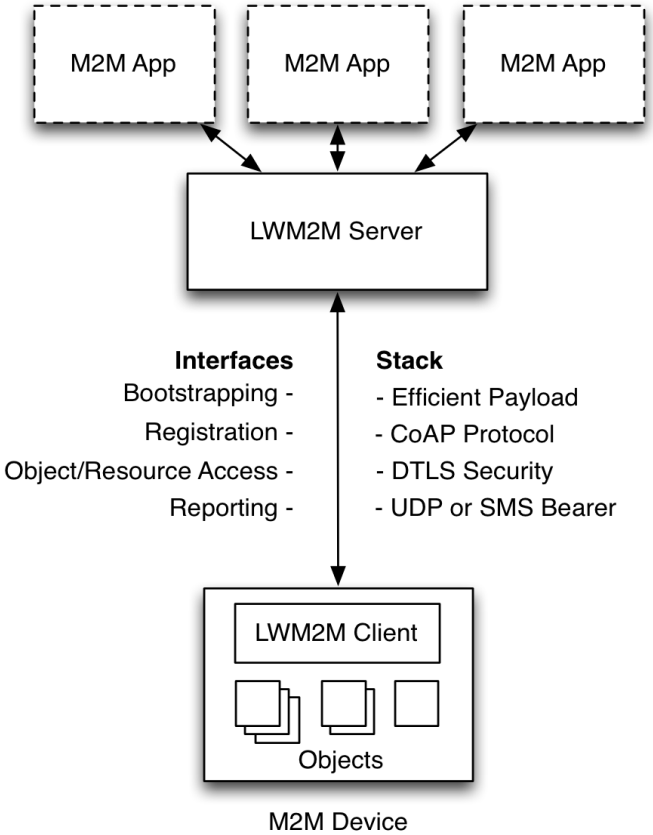
- Update application and system software
- Bug fixes

- Report Errors from devices
- Query about status of devices

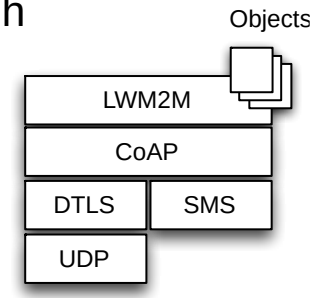
- Configure settings of the application
- Send control commands

- Notify changes in sensor values
- Notify alarms and events

OMA LWM2M Architecture



- M2M Applications
  - Application abstraction through REST API
  - Resource Discovery and Linking

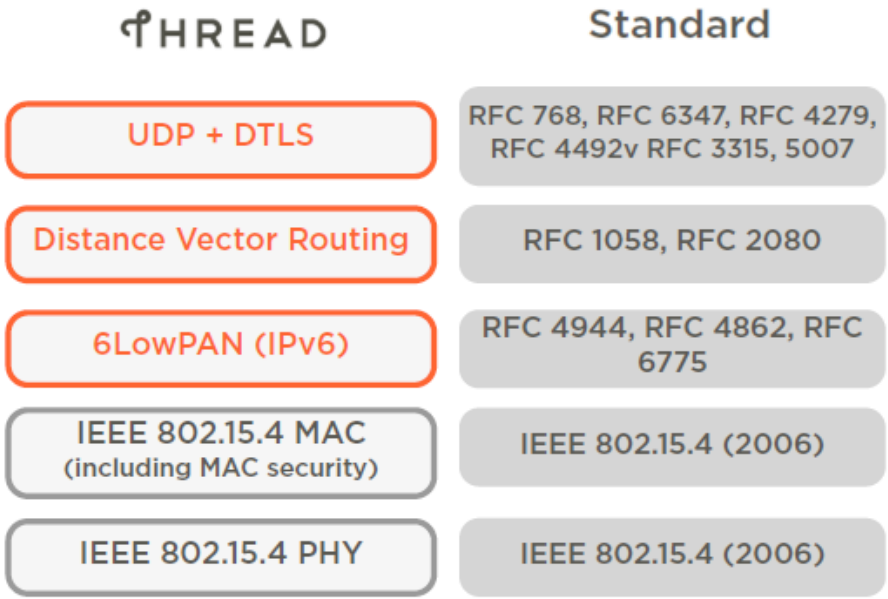


- LWM2M Server
  - Reuses IETF technologies, such as the CoAP protocol, DTLS, Resource Directory.
  - Deployable on gateways and in the cloud
- LWM2M Clients are Devices
  - Device abstraction through CoAP
  - LWM2M Clients are CoAP Servers
  - Any IP network connection

What is Thread?

A secure, wireless mesh networking protocol that:

- Supports IPv6 addresses and simple IP bridging
- Is built upon a foundation of existing standards
- Is optimized for low-power / battery-backed operation
- Is intended for control and automation (250kbps)
- Can support networks of 250 nodes or greater
- Supports low latency (less than 100 milliseconds)
- Offers simplified security and commissioning
- Runs on existing 802.15.4 wireless SoCs

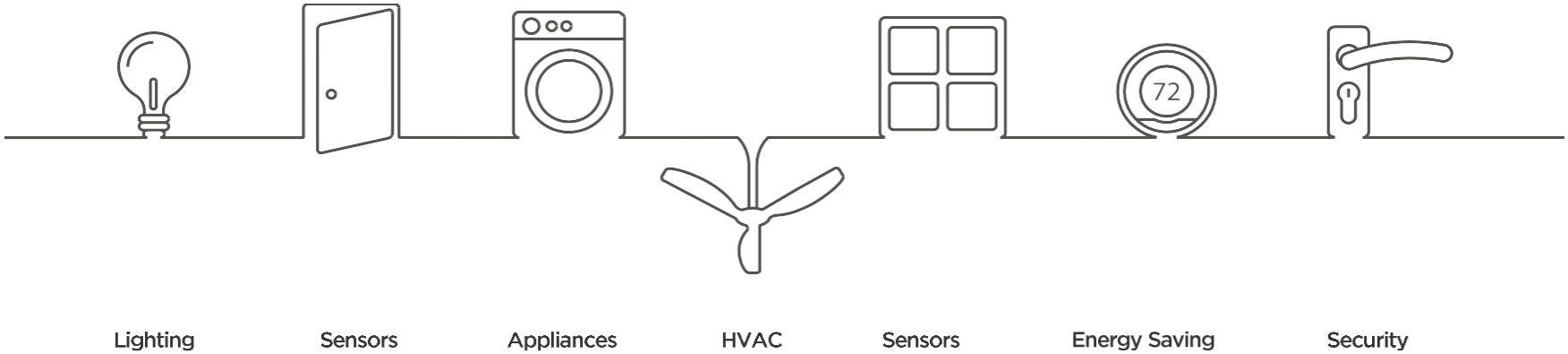


http://www.threadgroup.org/

Connected Home – Networking Requirements

- Direct addressability of devices – within the HAN and from smart phones or tablets
- Simplified forming and joining of network, limit special devices or customer knowledge of concepts like coordinator vs. router vs. end device
- Scalable to 200-300 devices in a home – with sufficient routers to provide coverage but remainder can be end devices
- Latency less than 100 milliseconds for typical interactions (user interaction concern)
- Allow the use of multiple gateways
- Seamless connectivity to user interaction on device of choice in the home (dedicated display, smart phone, tablet, etc.)
- Battery operated devices with years of expected life – door locks, security sensors etc.

Target Devices



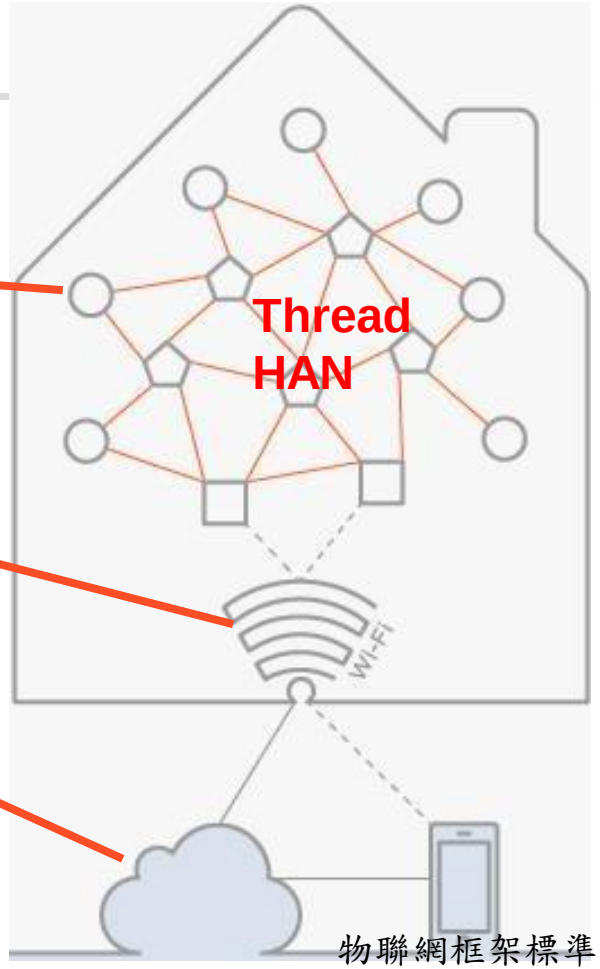
- Normally Powered
  - Gateway
  - Lighting
  - Appliances
  - Smart Meter
  - Garage door opener
  - HVAC equipment
  - Smart Plugs
  - Fans
- Powered or Battery
  - Thermostat
  - Light switches
  - Smoke detectors
  - In home display
  - Shades or blinds
  - Door bell
  - Glass break sensors
  - Robots/cleaners
- Normally Battery
  - Door sensors
  - Window sensors
  - Motion sensors
  - Door locks
  - Radiator valves
  - Body sensors

System Messaging Model

- Expect device to device communication within HAN for operations in the home

- Border Router forwards data to cloud
- Also provides WiFi connectivity to phone or tablet in the home

- Cloud connectivity for control when not at home
- When within the home, phone or tablet must go direct to gateway to reduce latency
- Has to be seamless to consumer using device



Key Features Overview

IP-based:

- Simplified bridging to other IP networks

Flexible Network:

- Simplified device types

Robust:

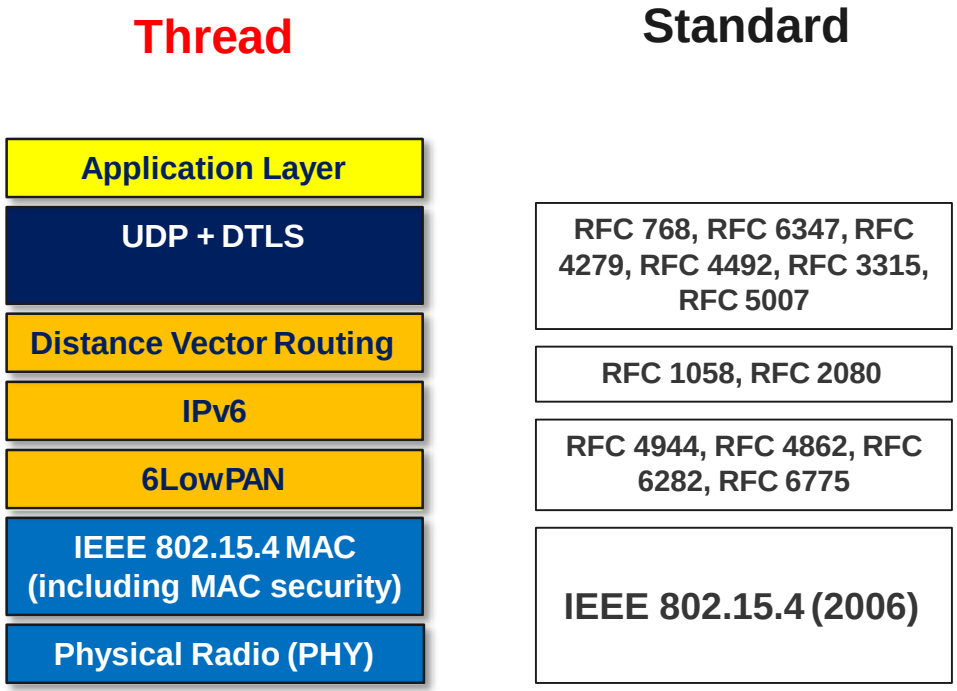
- No single point of failure

Secure:

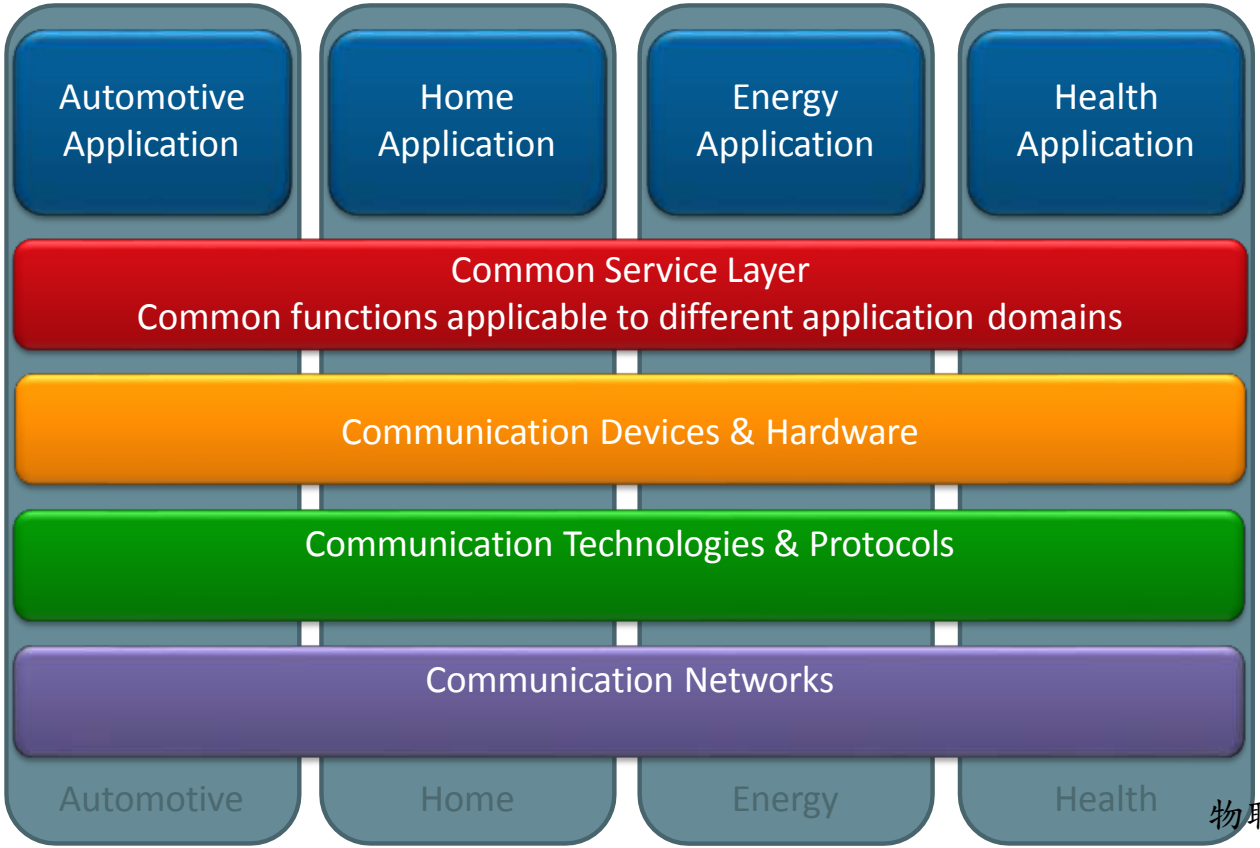
- Simple security and commissioning

Low Power Operation:

- Support for sleeping devices

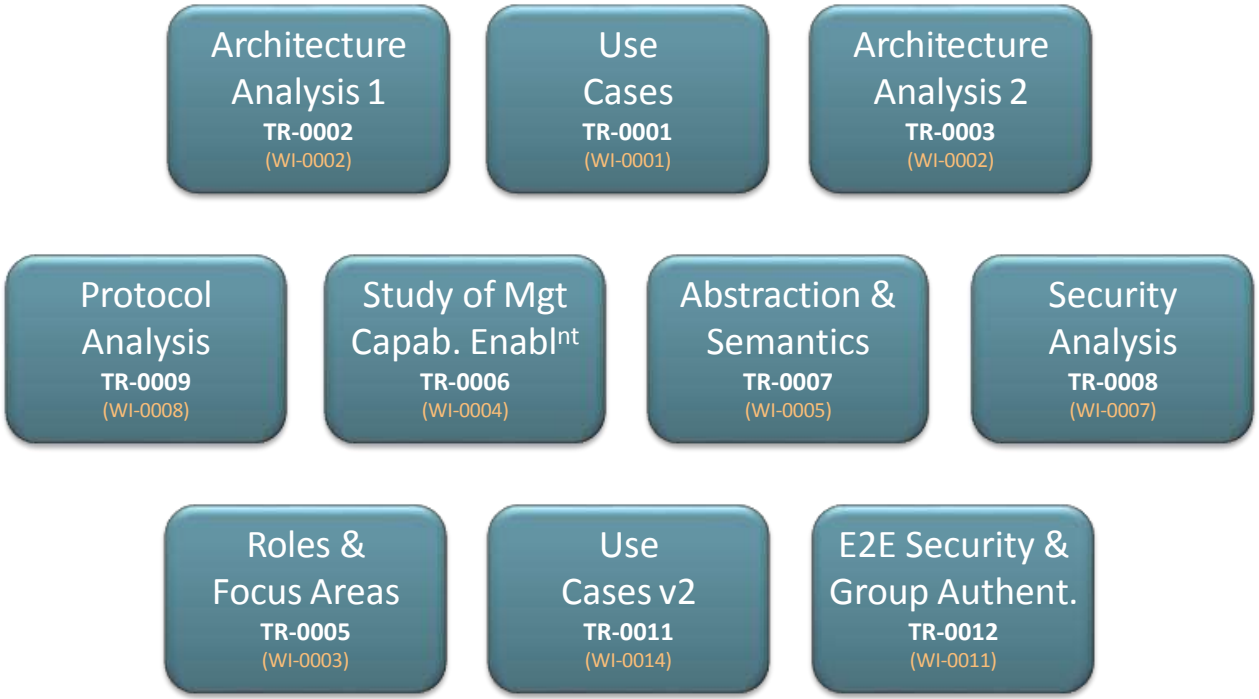


oneM2M – The Common Service Layer



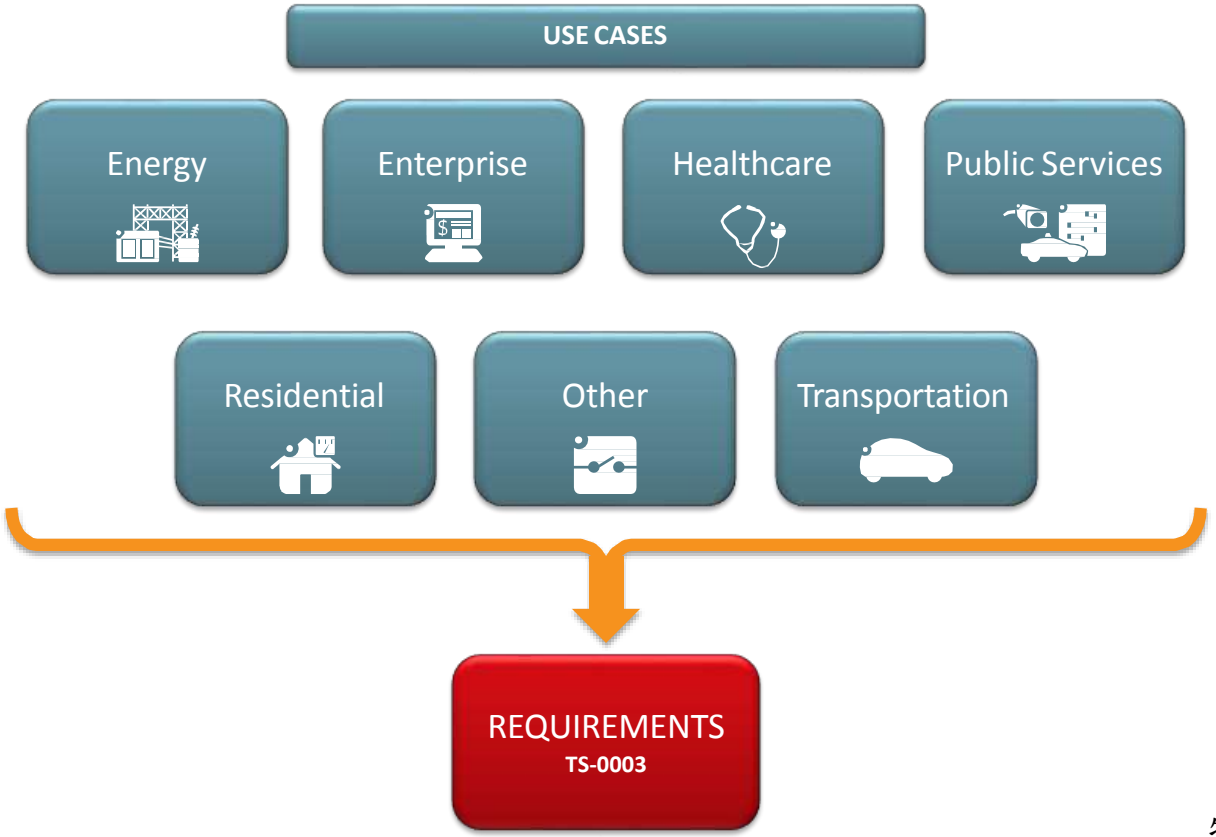
oneM2M – Common Service Functions

oneM2M – Technical Reports



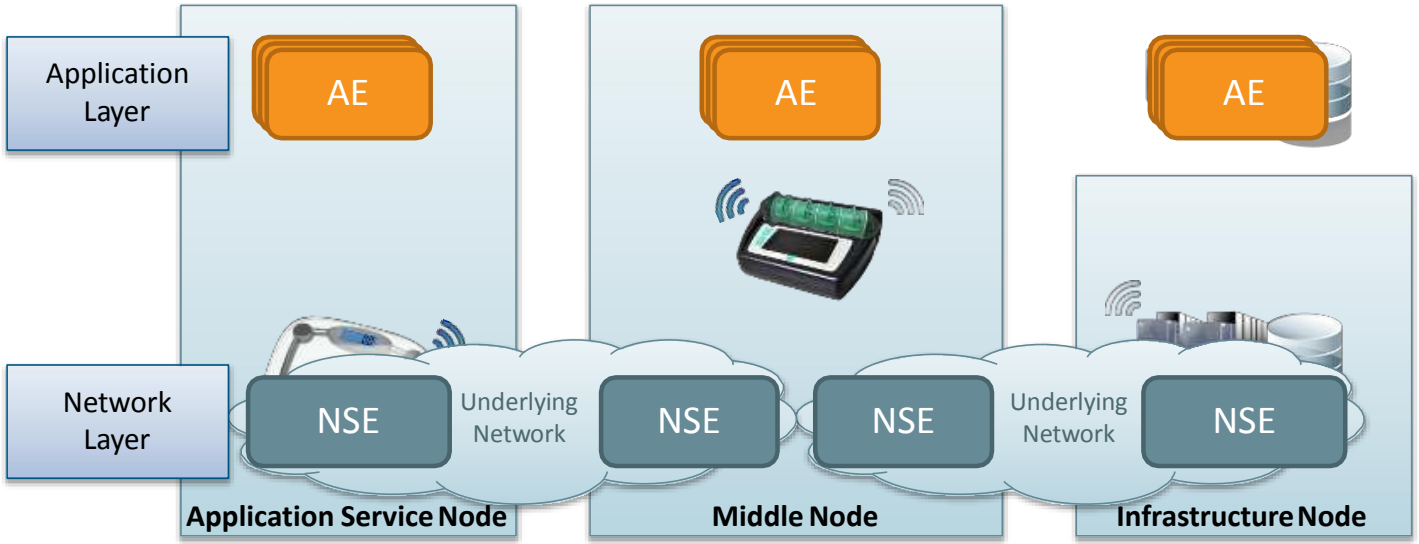
oneM2M – Technical Specifications

oneM2M – Use Cases & Requirements



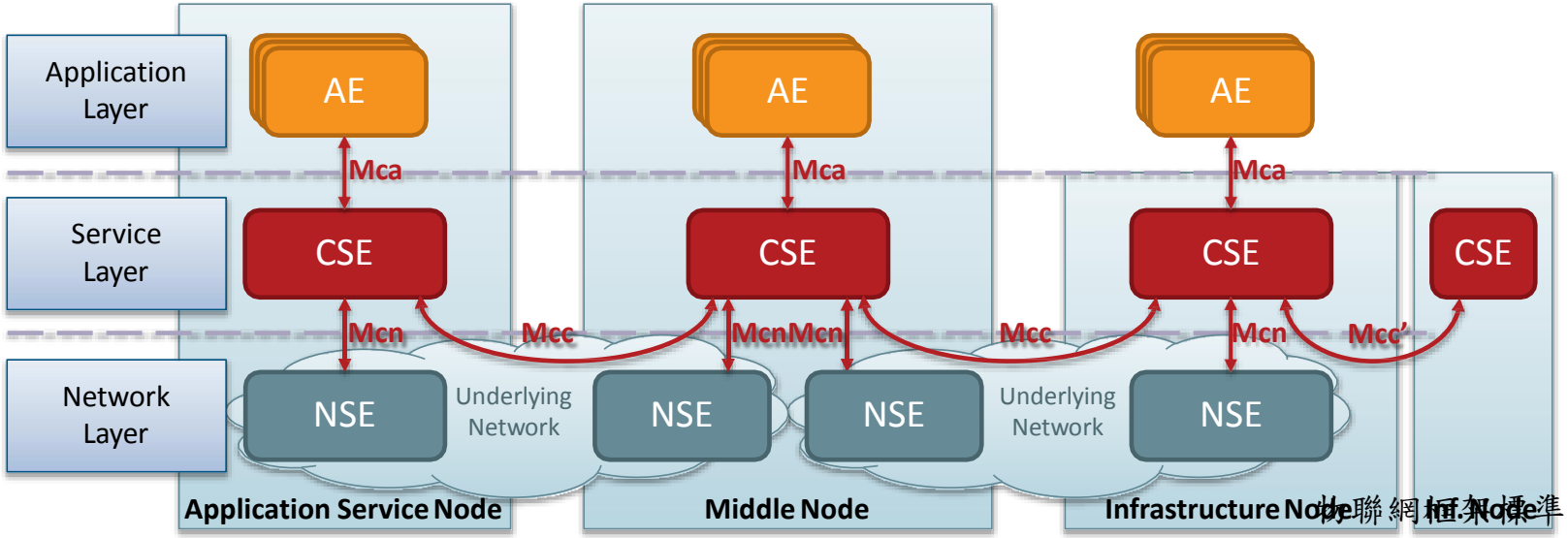
oneM2M - Architecture

Application Entity	Provides application logic for the end-to-end M2M solutions
Network Services Entity	Provides services to the CSEs besides the pure data transport
Node	Logical equivalent of a physical (or possibly virtualized, especially on the server side) device

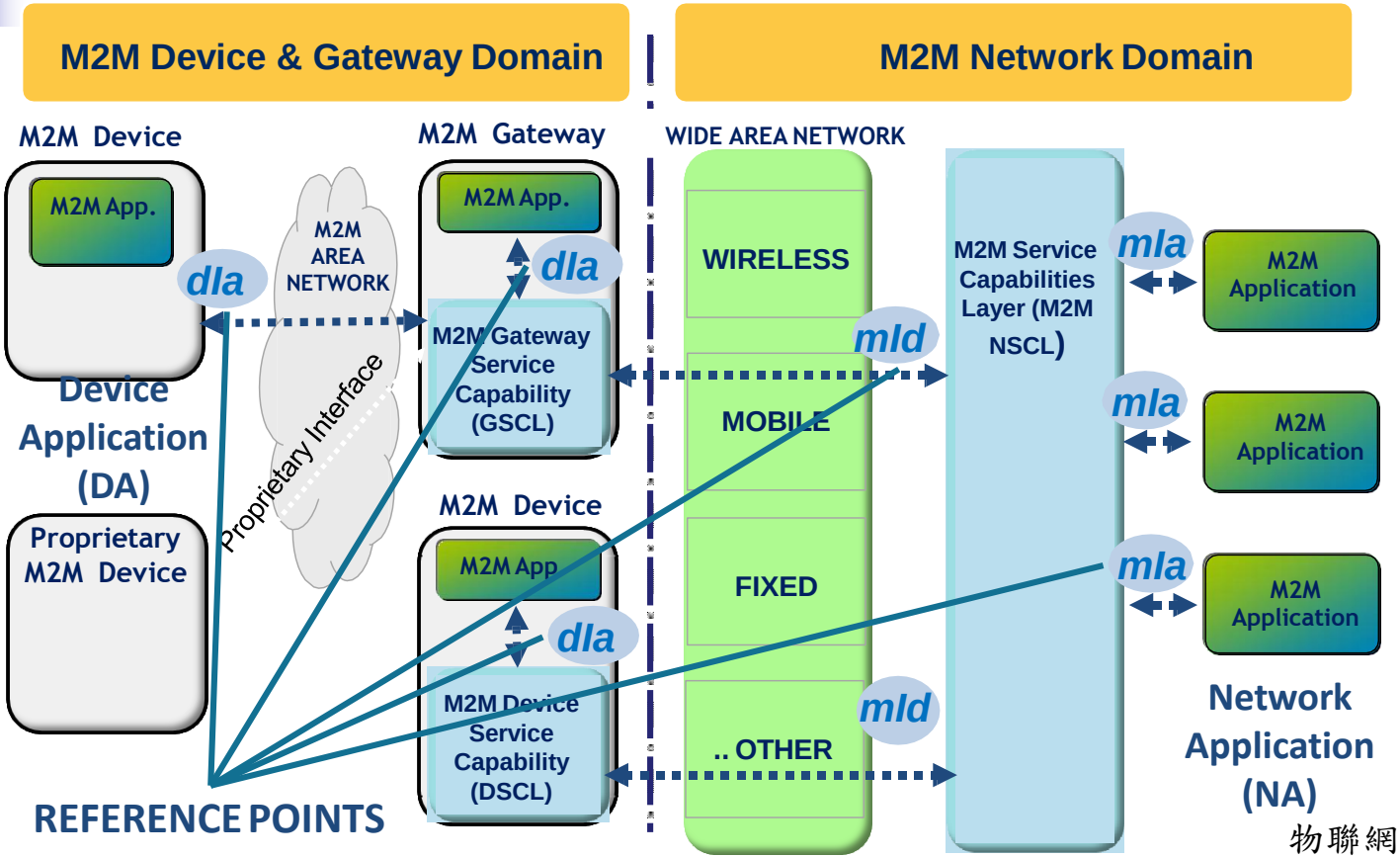


oneM2M - Architecture

Reference Point	One or more interfaces - Mca, Mcn, Mcc and Mcc' (between 2 service providers)
Common Services Entity	Provides the set of "service functions" that are common to the M2M environments
Application Entity	Provides application logic for the end-to-end M2M solutions
Network Services Entity	Provides services to the CSEs besides the pure data transport
Node	Logical equivalent of a physical (or possibly virtualized, especially on the server side) device



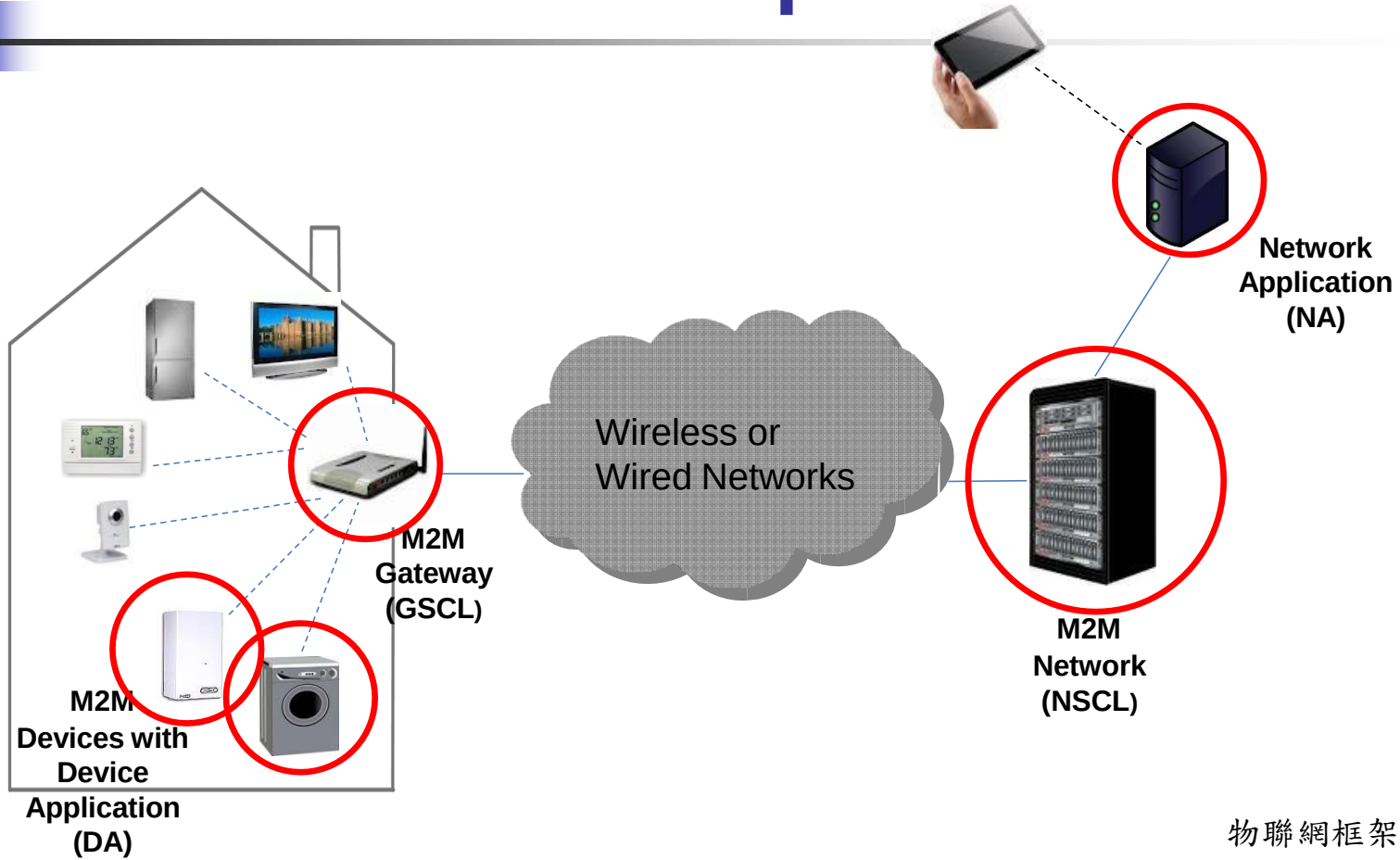
ETSI M2M - High Level Architecture



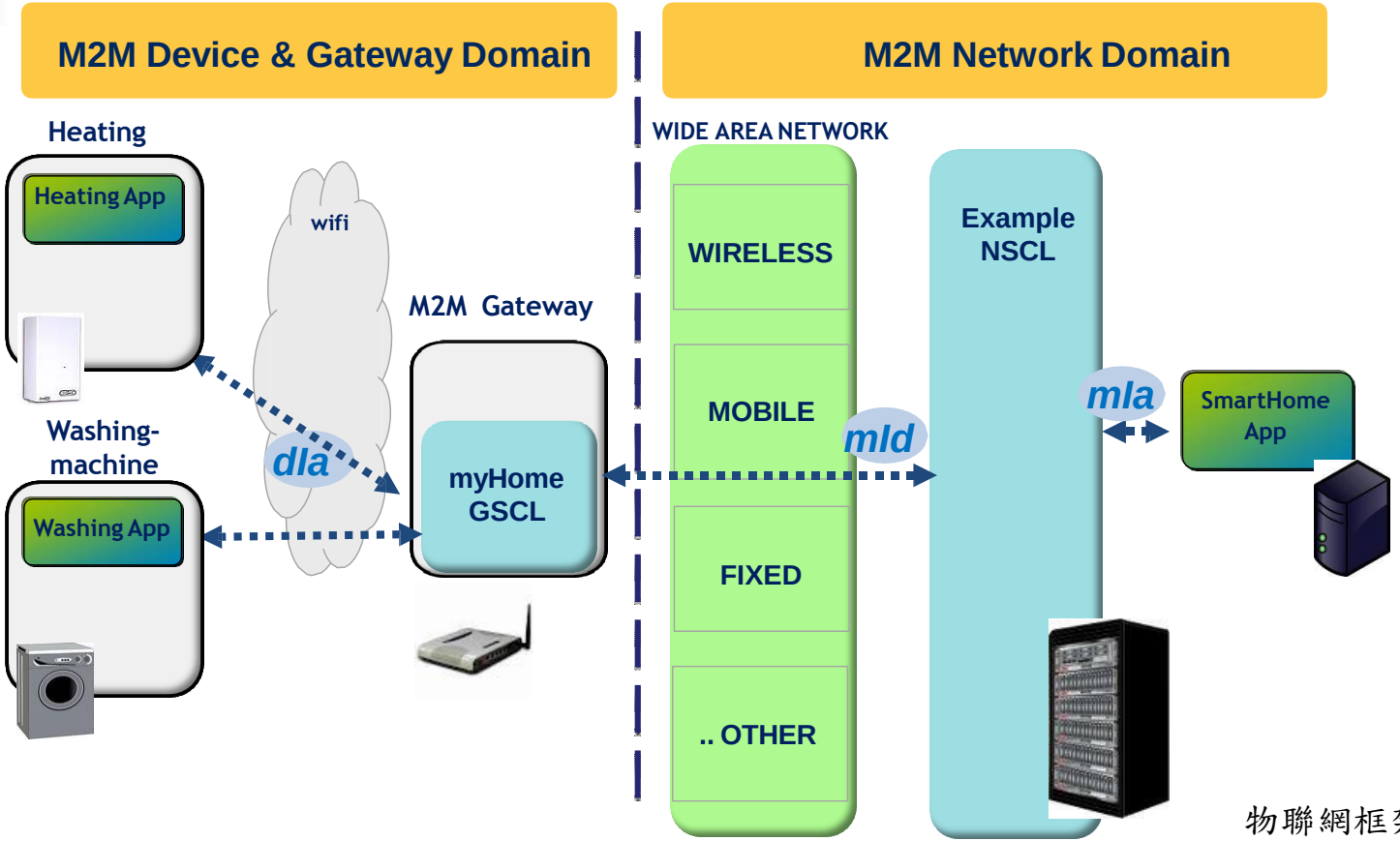
ETSI M2M - Architecture

- ETSI M2M adopted a RESTful architecture style
  - Information is represented by resources which are structured as a tree
- ETSI M2M standardizes the resource structure that resides on an M2M Service Capability Layer (SCL)
  - Each SCL contains a resource structure where the information is kept
- M2M Application and/or M2M Service Capability Layer exchange information by means of these resources over the defined reference points
- ETSI M2M standardizes the procedure for handling the resources

ETSI M2M - Example



ETSI M2M – High Level Deployment



IEEE P2413

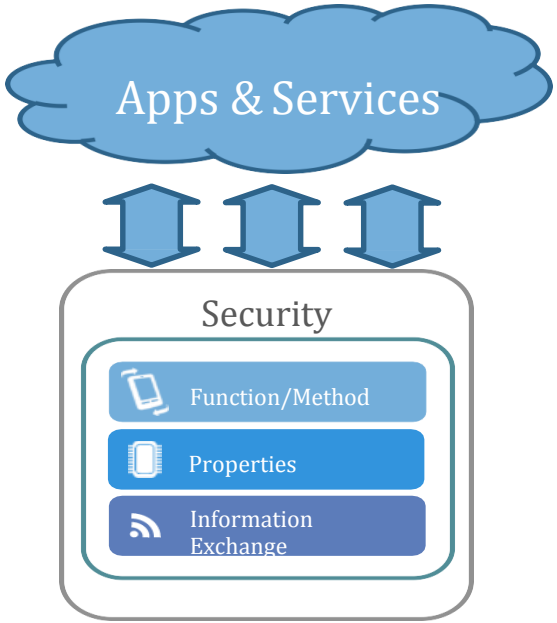
- Focus to integrate market needs with the developing IoT technology landscape
- Define an Architectural Framework for the IoT
  - descriptions of various IoT domains
  - definitions of IoT domain abstractions
  - identification of commonalities between different IoT domains

IEEE P2413

- The Architectural Framework for IoT provides
  - reference model
    - defines relationships among various IoT domains
  - reference architecture
    - builds upon the reference model
    - defines basic architectural building blocks and their ability to be integrated into multi-tiered systems
    - addresses how to document and mitigate architecture divergence.

IEEE P2413 Definitions

- The Group accepted the definition of the “Thing”:

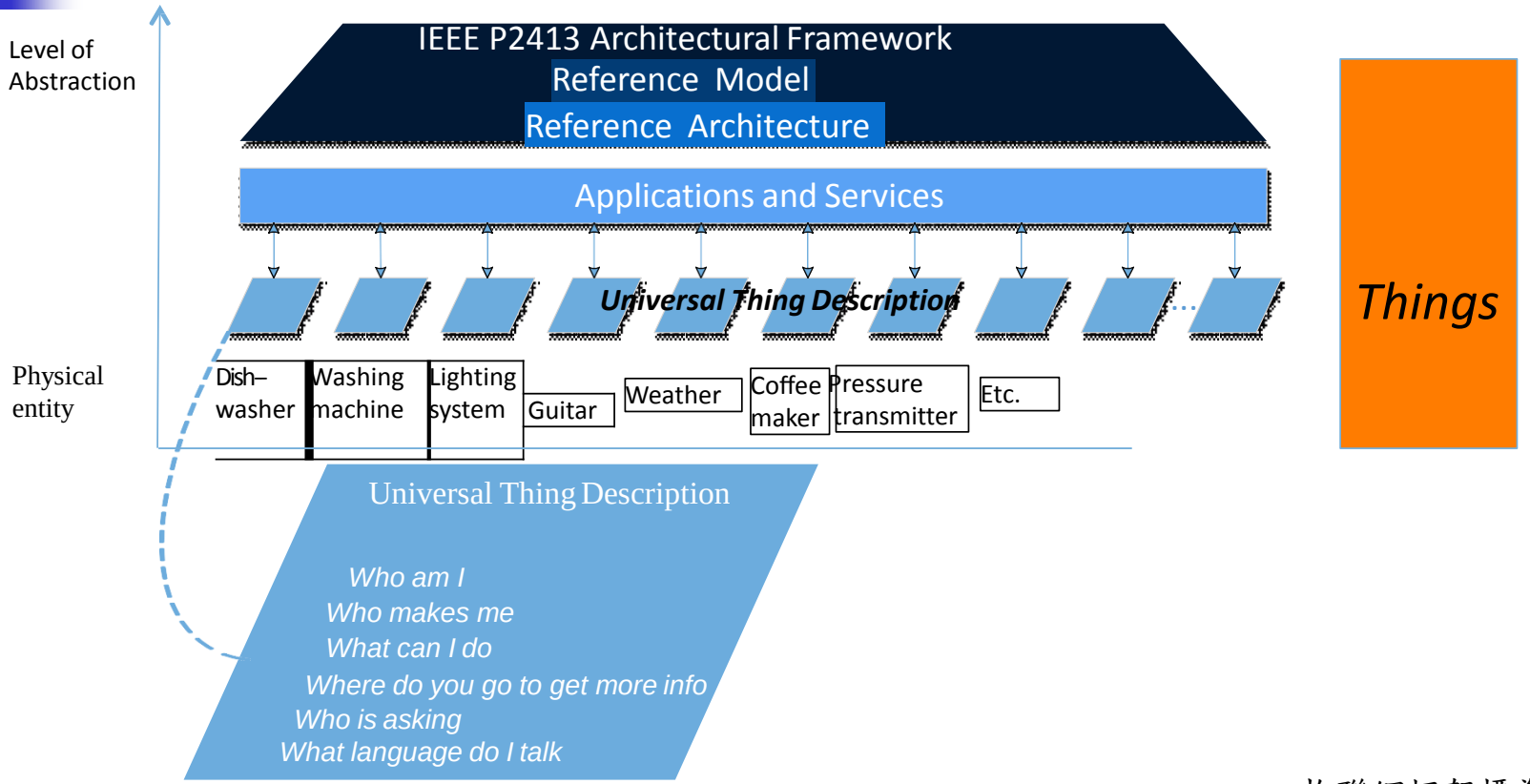


The “Thing”

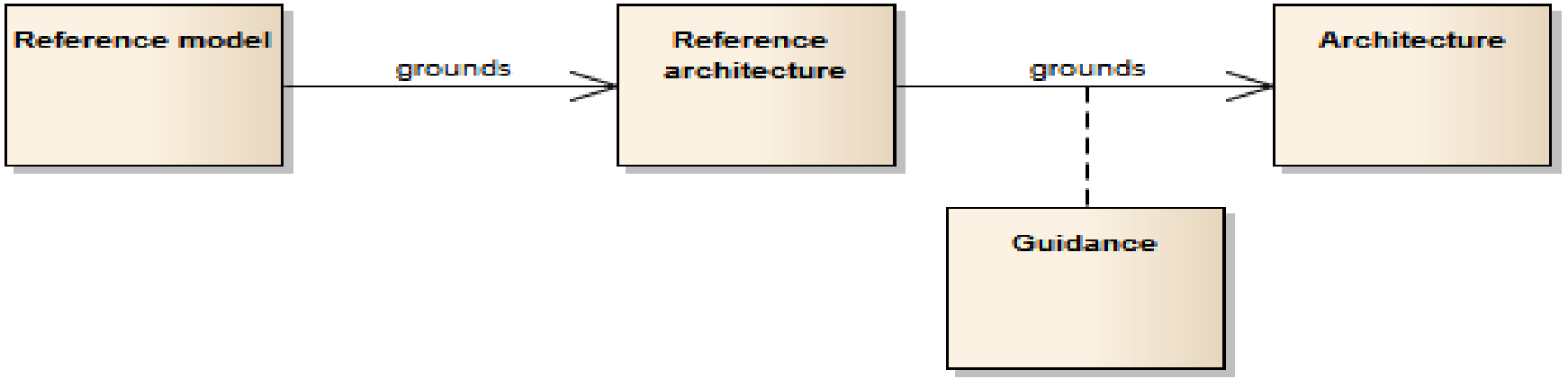
Notes:

- Things, Apps, and Services can be integrated into what would be abstracted as a “Thing”
- Information exchange could be “horizontal” (subscribe/ publish as an example) or vertical, or both
- Properties could be real or virtual

IEEE P2413 Levels of abstractions



IEEE P2413 Structure



IEEE P2413 Potential Profiles