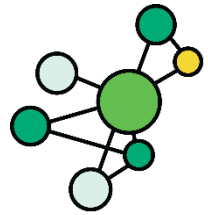
Open Internet Consortium (OIC) – Ecosystem, Specifications and Framework

March, 2016

OIC Presenters: Ravi Subramaniam

Standard Working Group
Open Interconnect Consortium

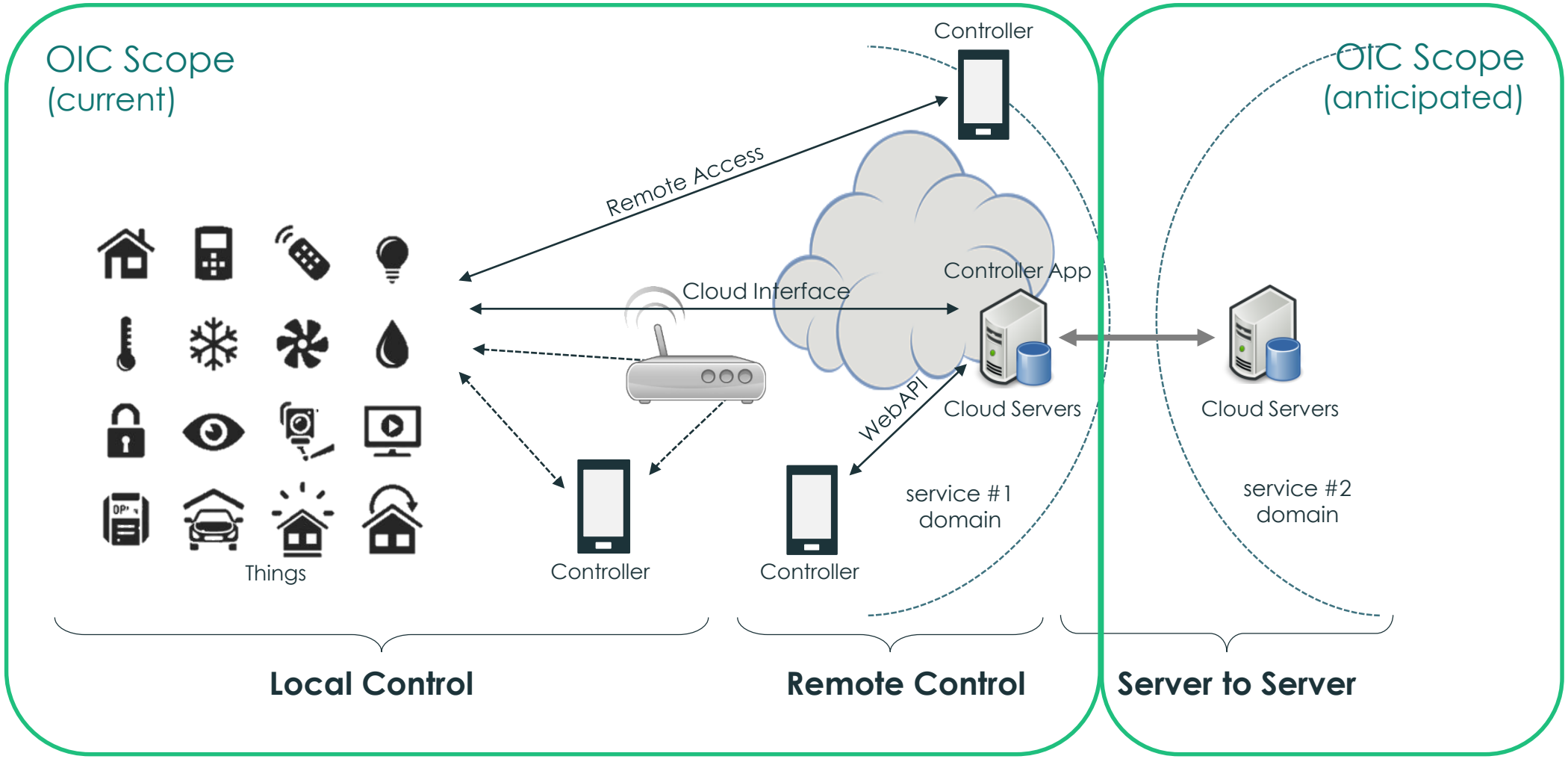
Table of Contents



OPEN INTERCONNECT CONSORTIUM℠

Technical Principles for an Internet of Things Ecosystem

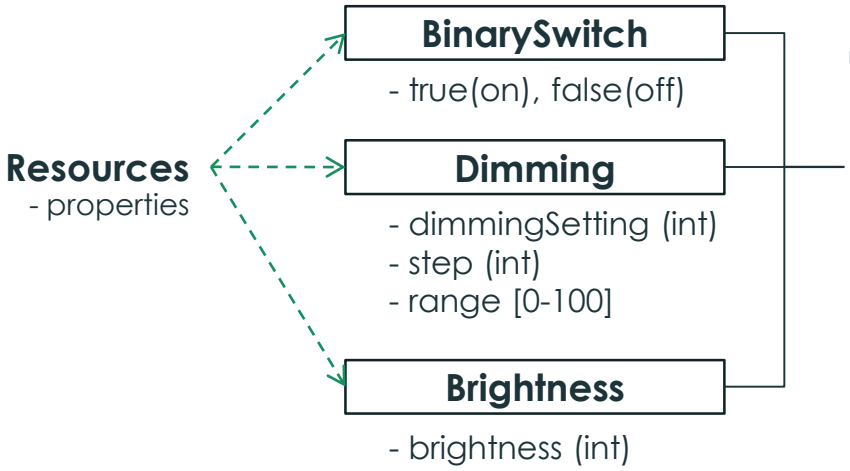
Scope of IoT


OIC Scope
(current)
OIC Scope
(anticipated)
Controller
Remote Access
Cloud Interface
Controller App
Cloud Servers
WebAPI
Cloud Servers
Things
Controller
Controller
service #1
domain
service #2
domain
Local Control
Remote Control
Server to Server

Approaches for defining and interacting with Things

OIC Scope

- **Declarative**: By defining things as resources and its properties

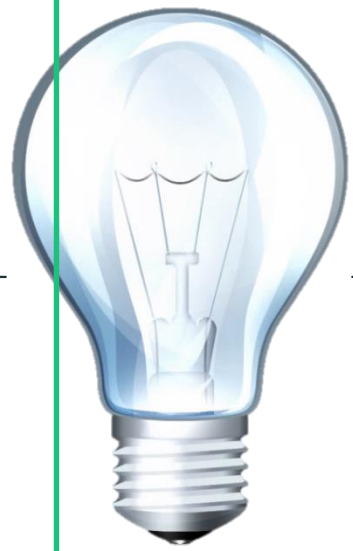


- (no Verbs) + Objects

*Fixed set of verbs (CRUDN) from transport layer will be used

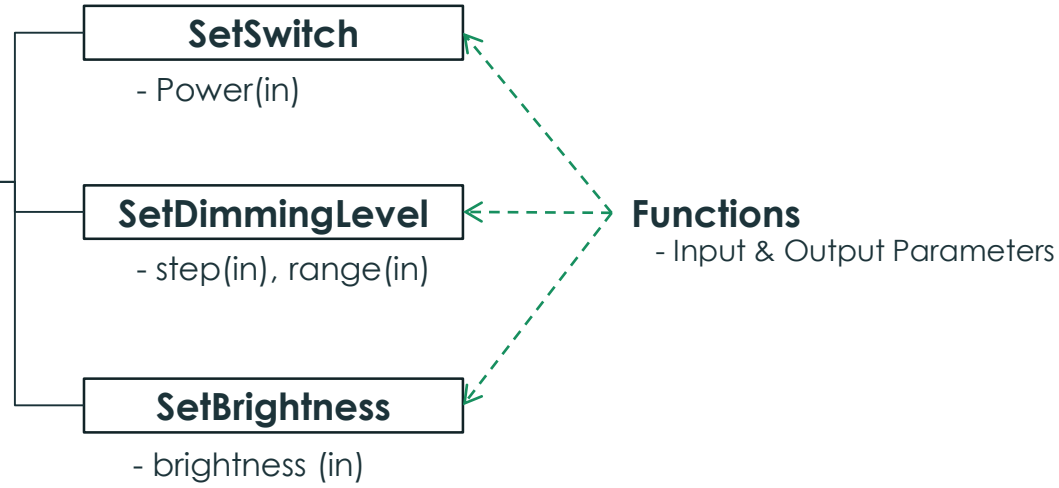
- Resource model in RESTful Architecture

(e.g., W3C, CSEP, etc.)



e.g., Light bulb

- **Imperative**: By defining functions of and operations on things

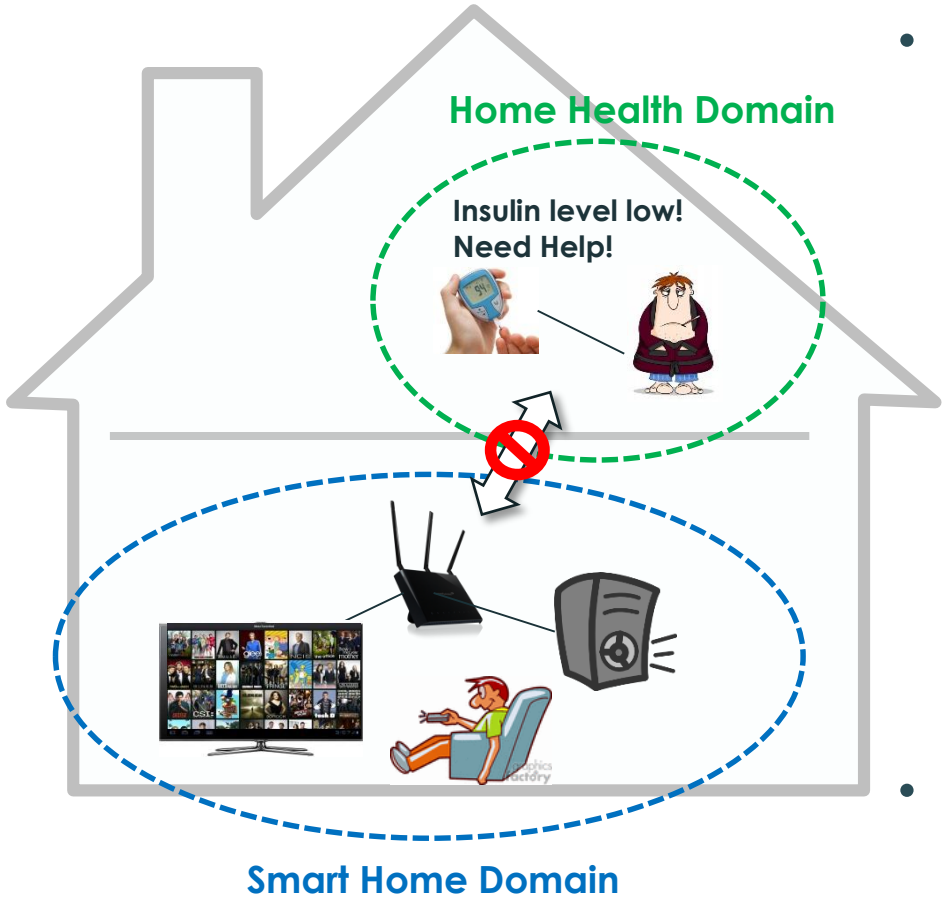


- (Verbs + Objects)
- RPC model

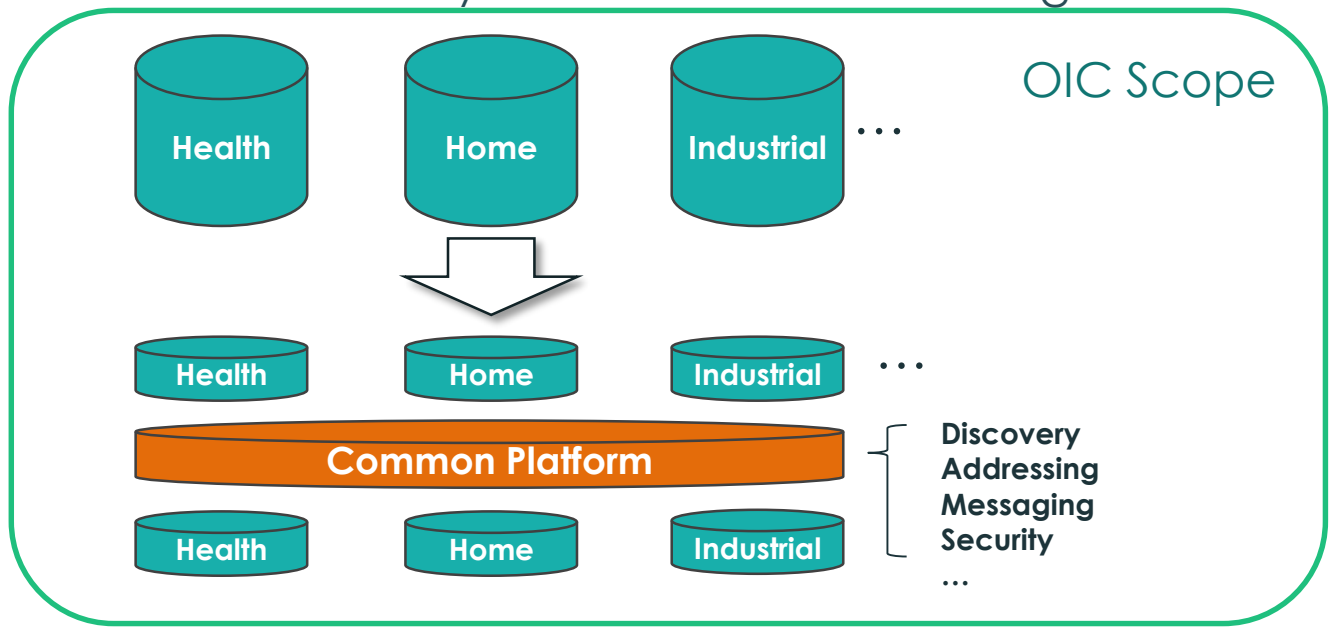
Additional Considerations*

- Late Binding (Reliability, Resilience, Conformance)
- Multi-Protocol
- Peer – Peer (Gateway and Cloud are not first class citizens but peers)
- *Framework as middleware*
  - *what defines the ecosystem (is it only spec?)*
  - *Need consistent behavior*
  - *Adaptability*
  - *Optimization (e.g. sensors, connectivity)*

Support of Multiple Verticals



- Legacy vertical services usually designed as silos
  → No common way to communicate among them

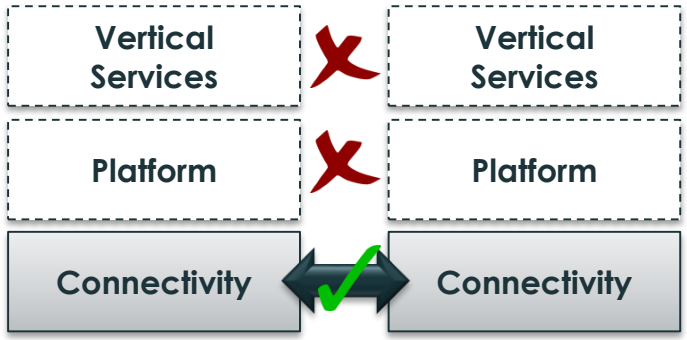


- A common platform provides a foundation for vertical services to collaborate and interwork by providing common services and data models

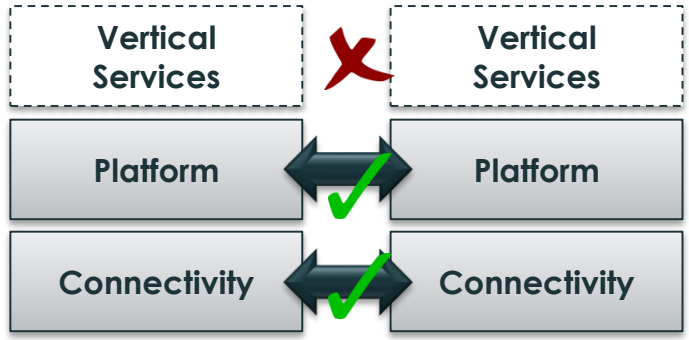
Interoperability

- **Full interoperability** from the connectivity layer up to the service layer is the only way to truly guarantee a satisfactory UX
- Interoperability at the Connectivity and/or Platform layer only provides partial interoperability which can ultimately lead to fragmentation

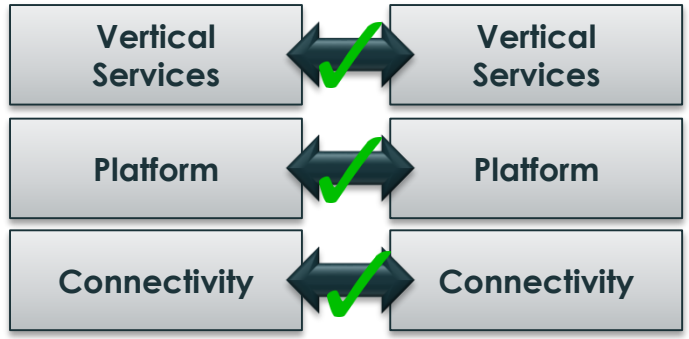
① **Connectivity Level Interoperability**



② **Platform Level Interoperability**



③ **Service Level Interoperability**

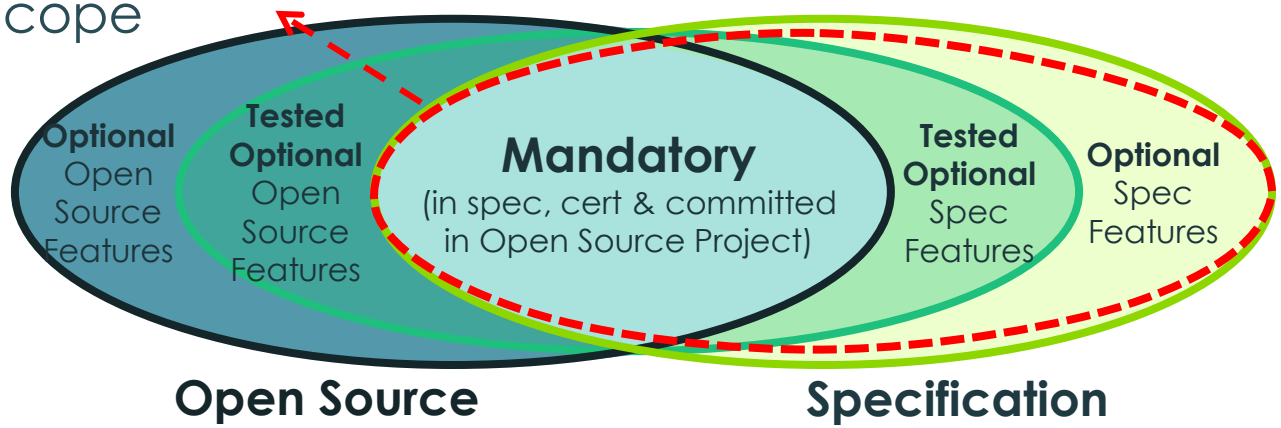


OIC Scope

Interoperability & Certification

- Conformance test - Each device proves conformance to specifications
- Interoperability test - Each device proves interoperability with other devices

- Certification Scope



Licensing

OIC Scope

- For IPR Policy : e.g. RAND-Z, RAND, etc…
- For Open Source : e.g. Apache2, Internet Software Consortium (ISC), etc…

- Due to the common nature of IoT connecting everything over the Internet, it's most critical for manufacturers to avoid a licensing risk
  - Everything connected could be at potential risk

- Offering manufacturer-friendly Licensing and IPR Policy enables growth of market by attracting both start-ups and large enterprises; such an IPR policy must be clear and readily understandable ensuring that the terms are offered by all IP holders.

Introduction of Open Interconnect Consortium

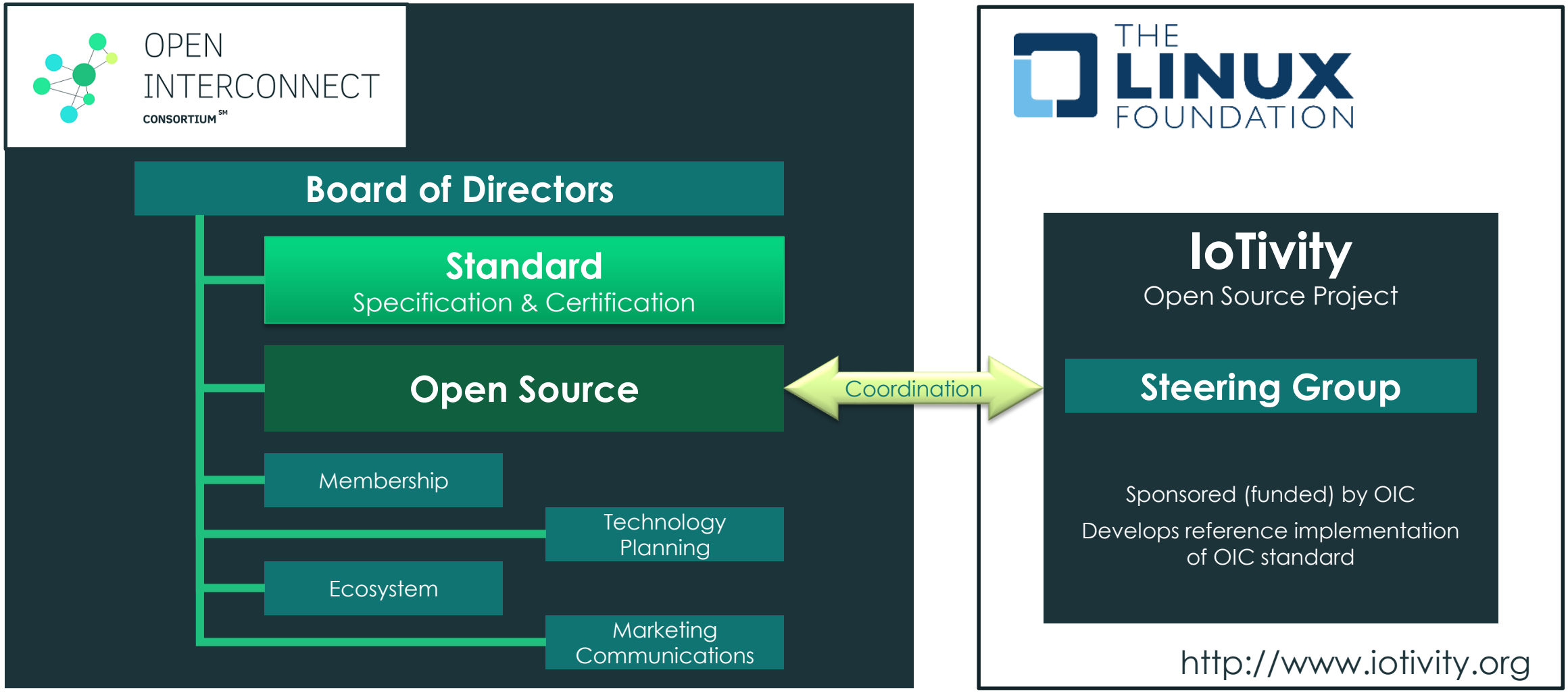
Growing Membership

Diamond

Platinum

Gold


OIC Organizational Structure



OIC Key Concepts

- **Dedicated and optimized protocols for IoT**
- **Standards and Open Source to allow flexibility creating solutions**
- **Full stack definition for maximum interoperability**
  - Connectivity, Platform and Vertical Services defined
  - License applies to members and affiliates of members
- **Certification and Logo program**
- **Free IPR License**
  - Code: Apache 2.0
  - Specification: RAND-Z

OIC Specification overview

Specification Structure

Infrastructure

- Core Framework
- Security
- Remote Access
- Certification Test Plans and Test Cases

Resource Model

- Resource Specification (Domain agnostic)

Per Application Vertical

- Device Specification
- Vertical Specific Resource Specification

Core Framework Specification

Overview

Objectives

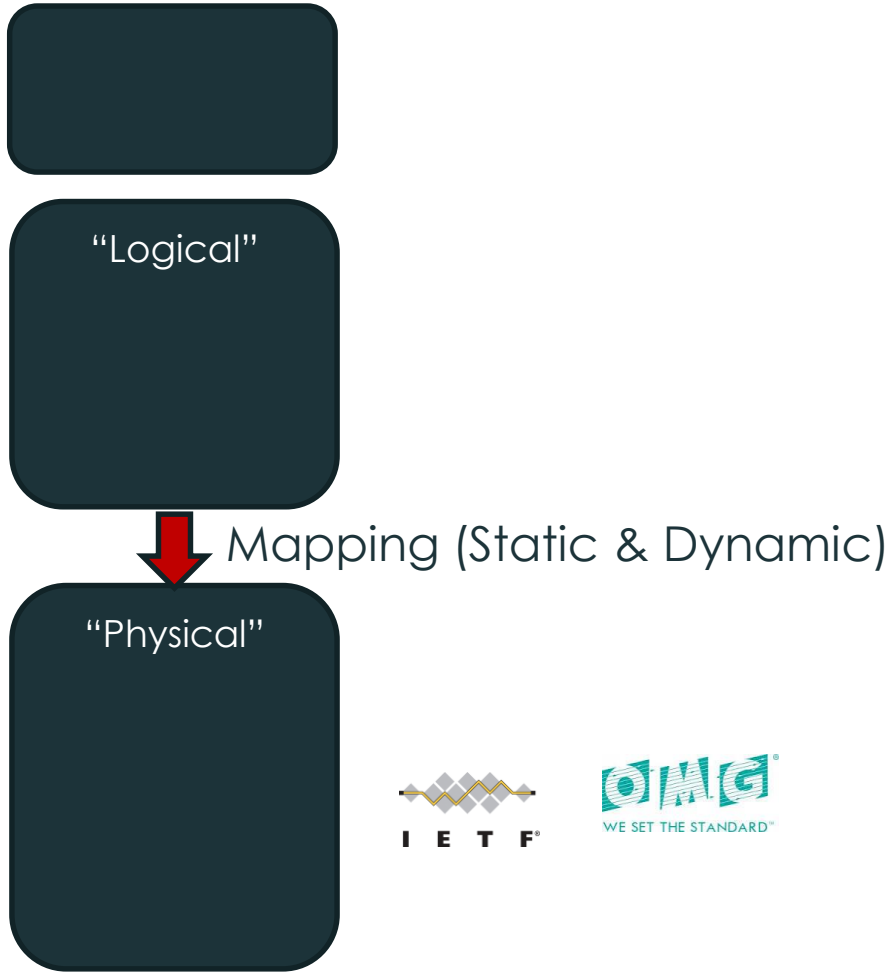
- Core Framework Specification Scope
  - Specifies the technical specification(s) comprising of the core architectural framework, messaging, interfaces and protocols based on approved use-case scenarios
  - Enables the development of vertical profiles (e.g. Smart Home) on top of the core
- Architect 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

Separation of Concern

Data Model

Information Model

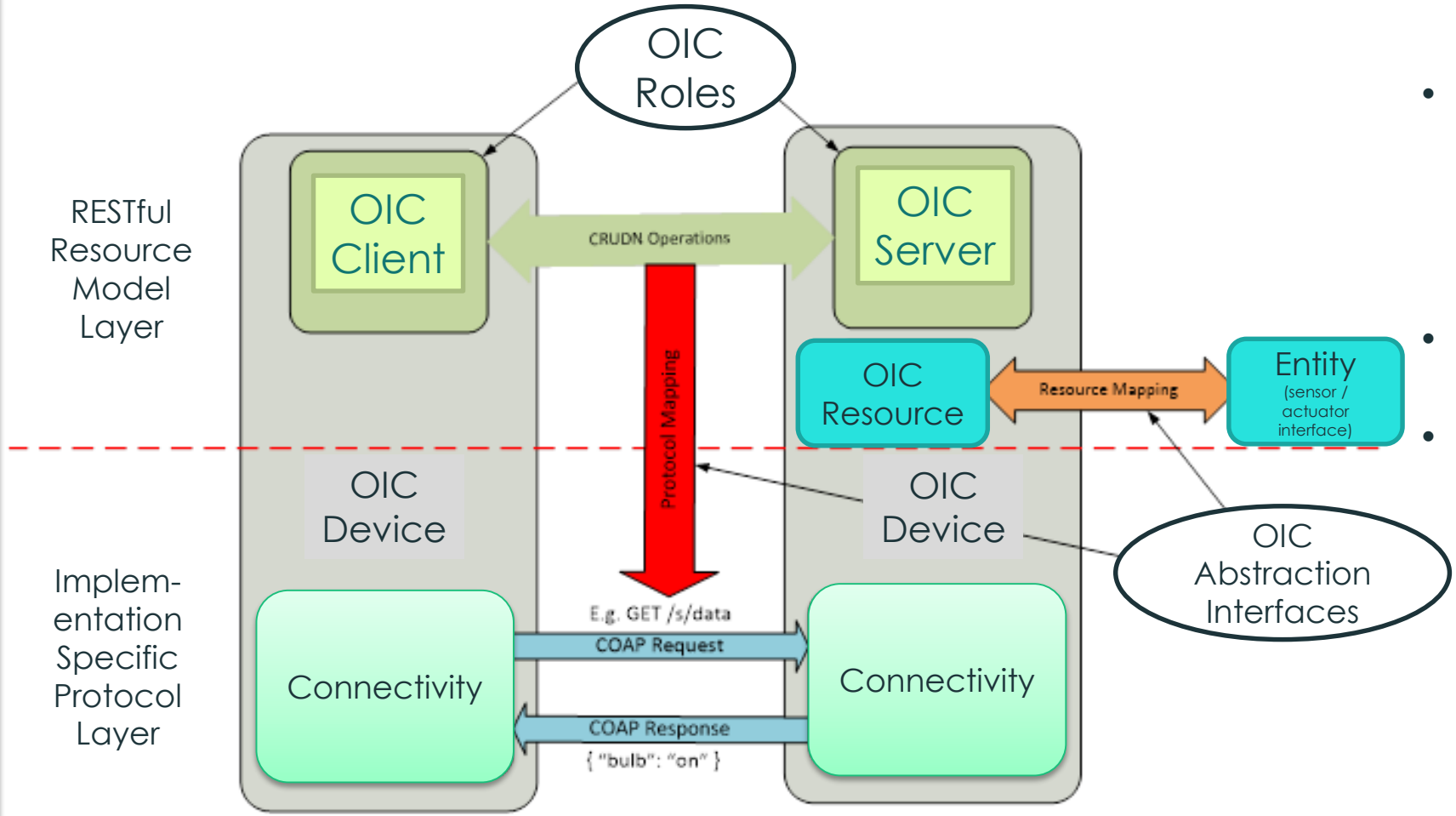
Connectivity Model



IETF

OMG
WE SET THE STANDARD

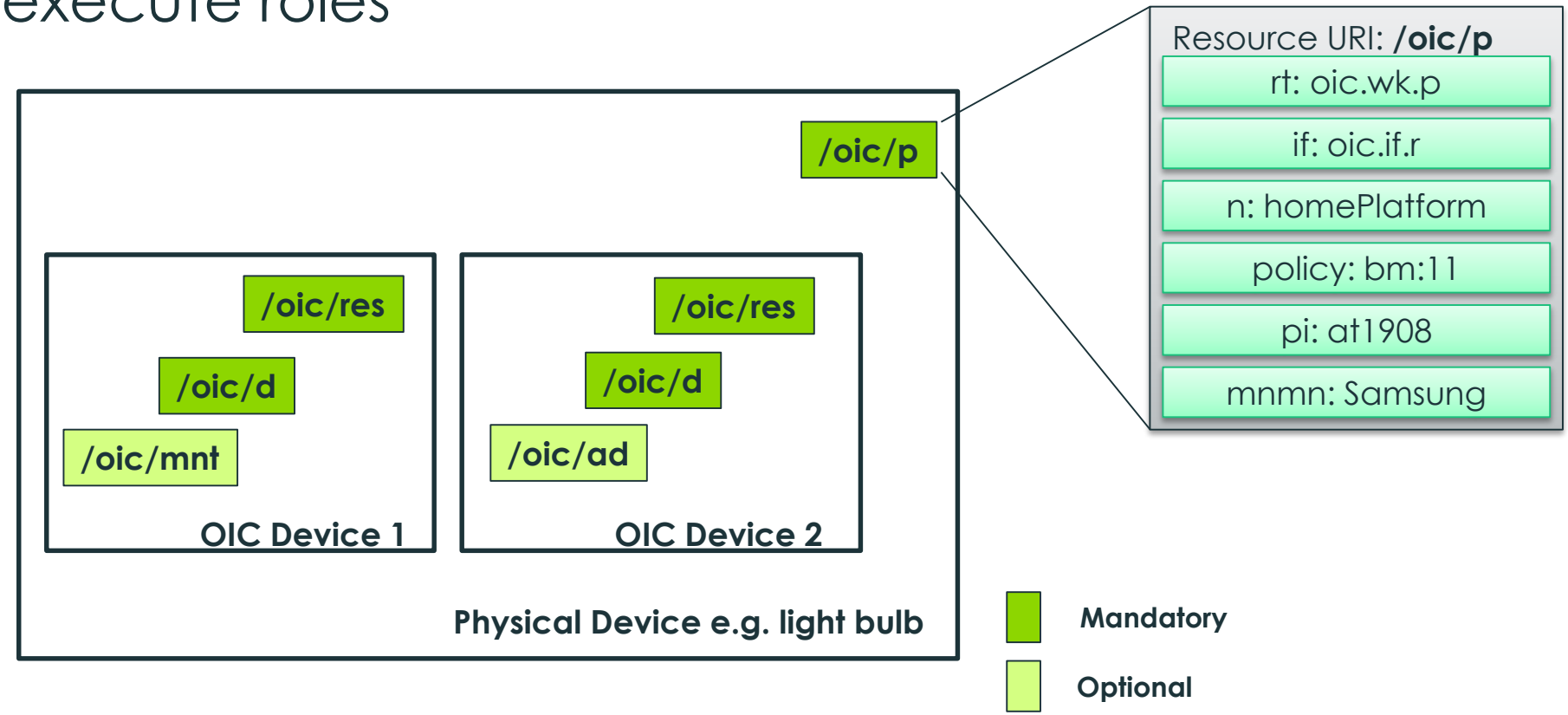
OIC Conceptual Architecture



- **Information Model**
  - Resource oriented
  - RESTful architecture
  - Semantics
  - Physical abstraction
- **Data Model**
  - For vertical and device
- **Data connectivity abstraction**
  - Protocol and layer agnostic
  - Dynamic and late binding

Organization of an OIC Device

- OIC Device – abstraction on a platform to host resources and execute roles

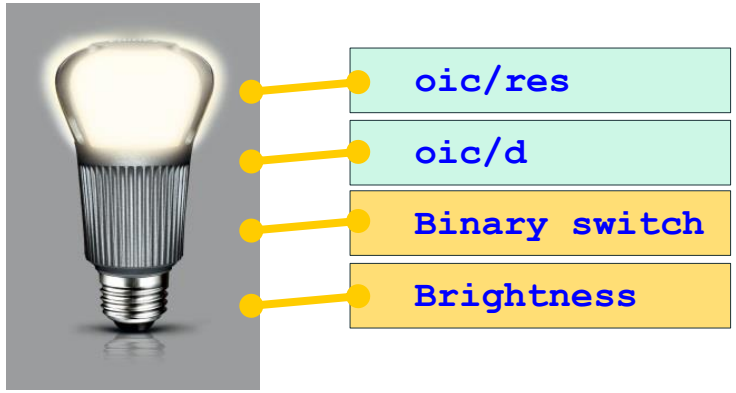


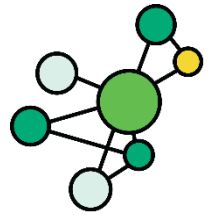
Device example: light device (oic.d.light)

- 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



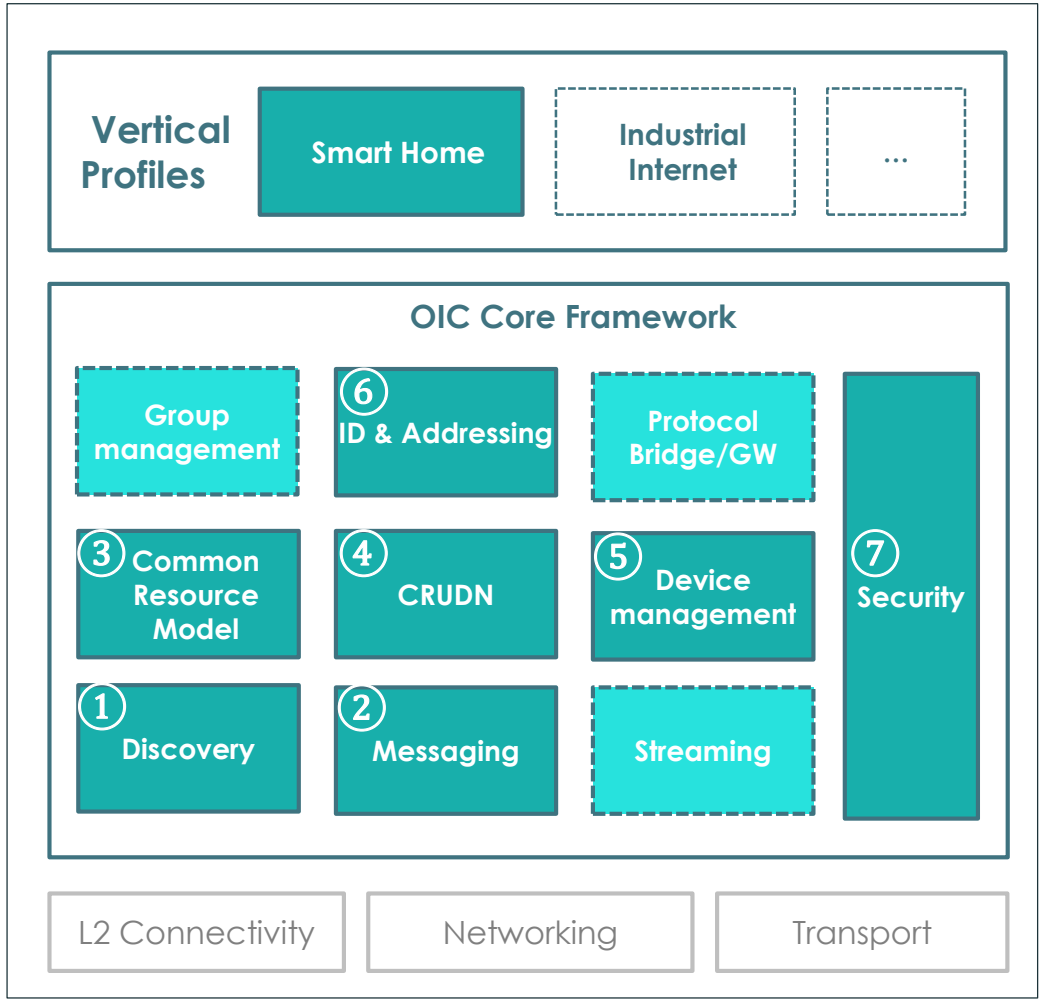


OPEN INTERCONNECT CONSORTIUM℠

Core Framework Specification

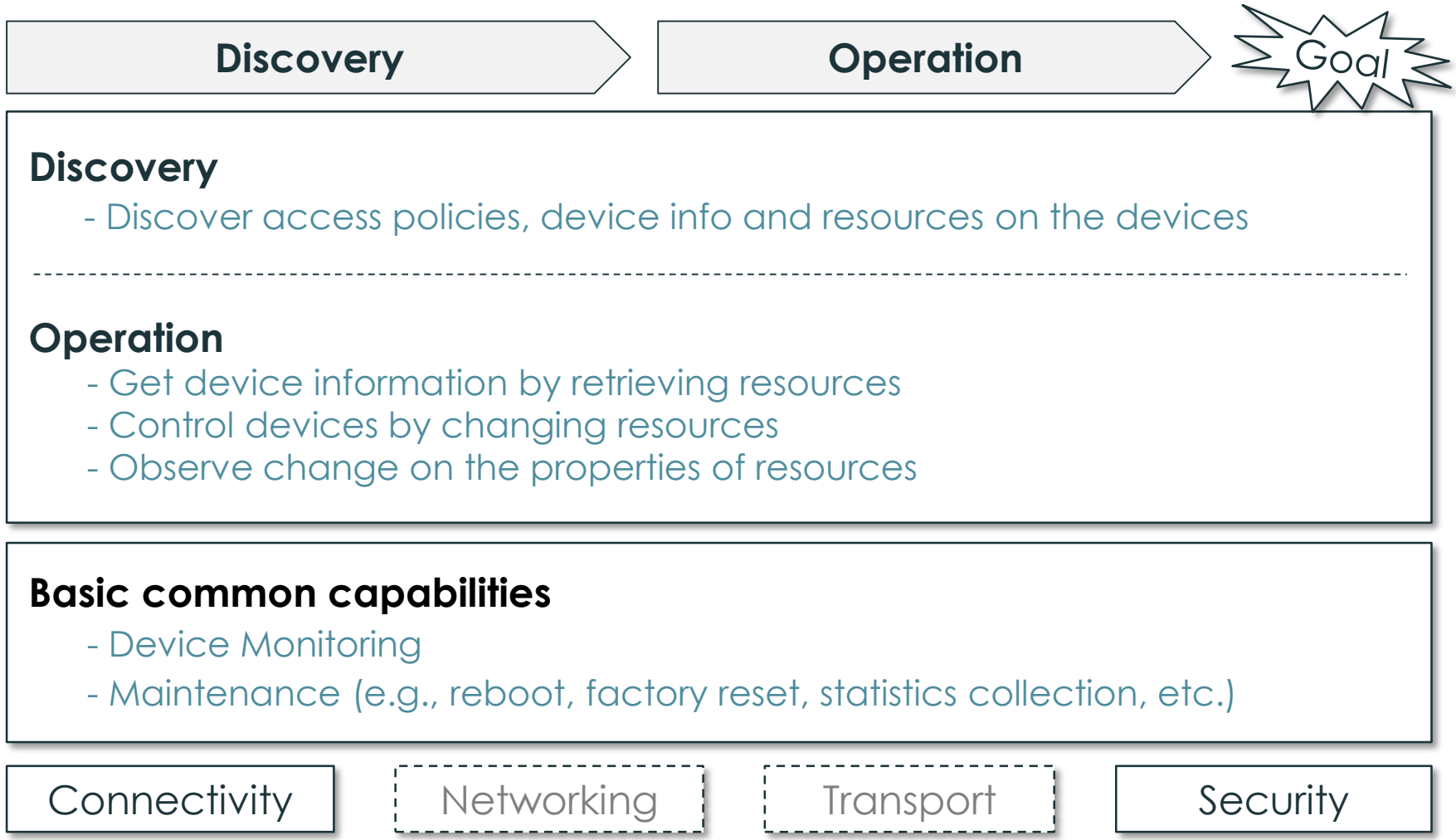
Key Features

OIC Spec Features – Core Framework Spec

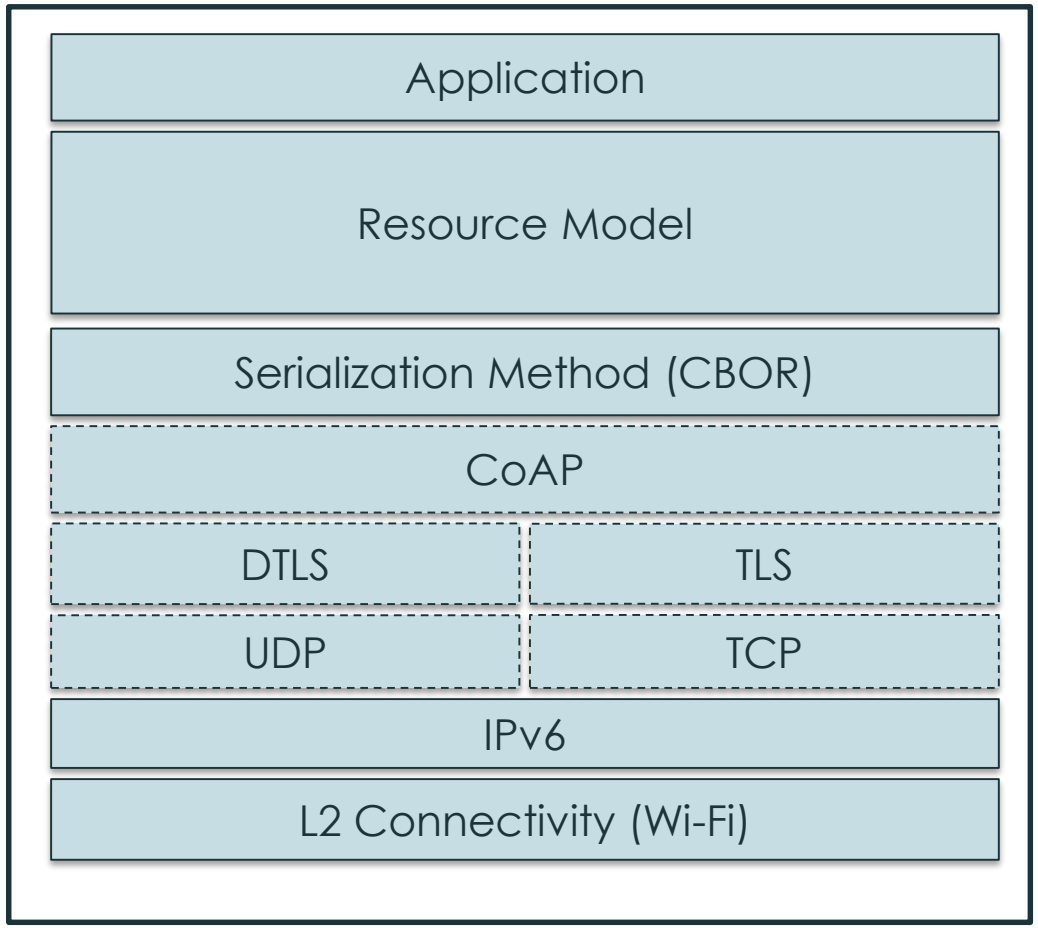


1. **Discovery:** Common method for device discovery (IETF CoRE)
2. **Messaging:** Constrained device support as default (IETF CoAP) as well as protocol translation via intermediaries
3. **Common Resource Model:** Real world entities defined as data models (resources)\
4. **CRUDN:** Simple Request/Response 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



Protocol Stack



OIC Stack Layering
(may change over time)

Alternative Options for Interoperability

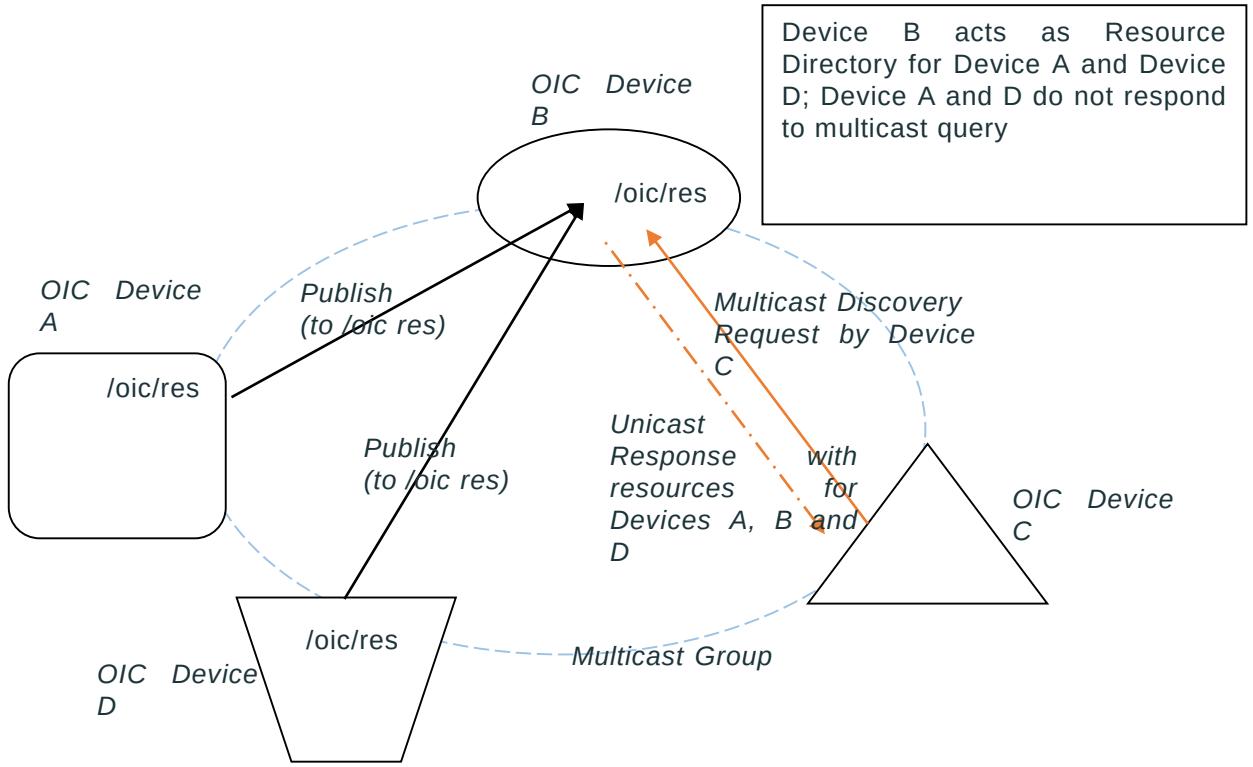
Serialization Method	JSON or XML/EXI can be negotiated
IP Version	v4 supported for legacy devices

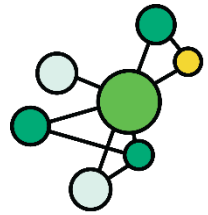
Resource Model Building Blocks

- Resource
- Link
  - Establishes relationship between a context resource and a target resource
- Collection
  - Contains one or more references (i.e. Links) to other resources
- Scene
  - A set of pre-defined resource property values that may be used to initialize a collection
- Rules
  - A logical “if then” statement (i.e. Links)
- Scripts
  - A programmatic element that can be used to incorporate conditionals, delays, loops and other programmatic elements, including reading and writing scenes

Resource Discovery

- Peer-peer
  - multicast
- Resource Directory
  - Offloads handling of discovery (response to multicast messages) to devices that are capable of doing so
  - Key enabler for sleepy end nodes, enhances battery life.





OPEN INTERCONNECT CONSORTIUM℠

Security Specification

Key Features

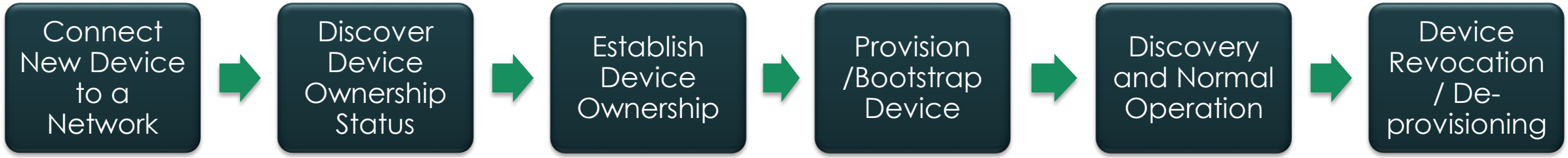
Security Goals

- Protect OIC networks
  - Manage devices entering user's IoT network(s)
- Protect OIC devices
  - Manage device identities for security and privacy
  - Prevent device impersonation
- Protect OIC resources
  - Manage authorized access
  - Prevent data theft
  - Consider safety and resiliency impact to DoS attacks

OIC Security Meta Objective

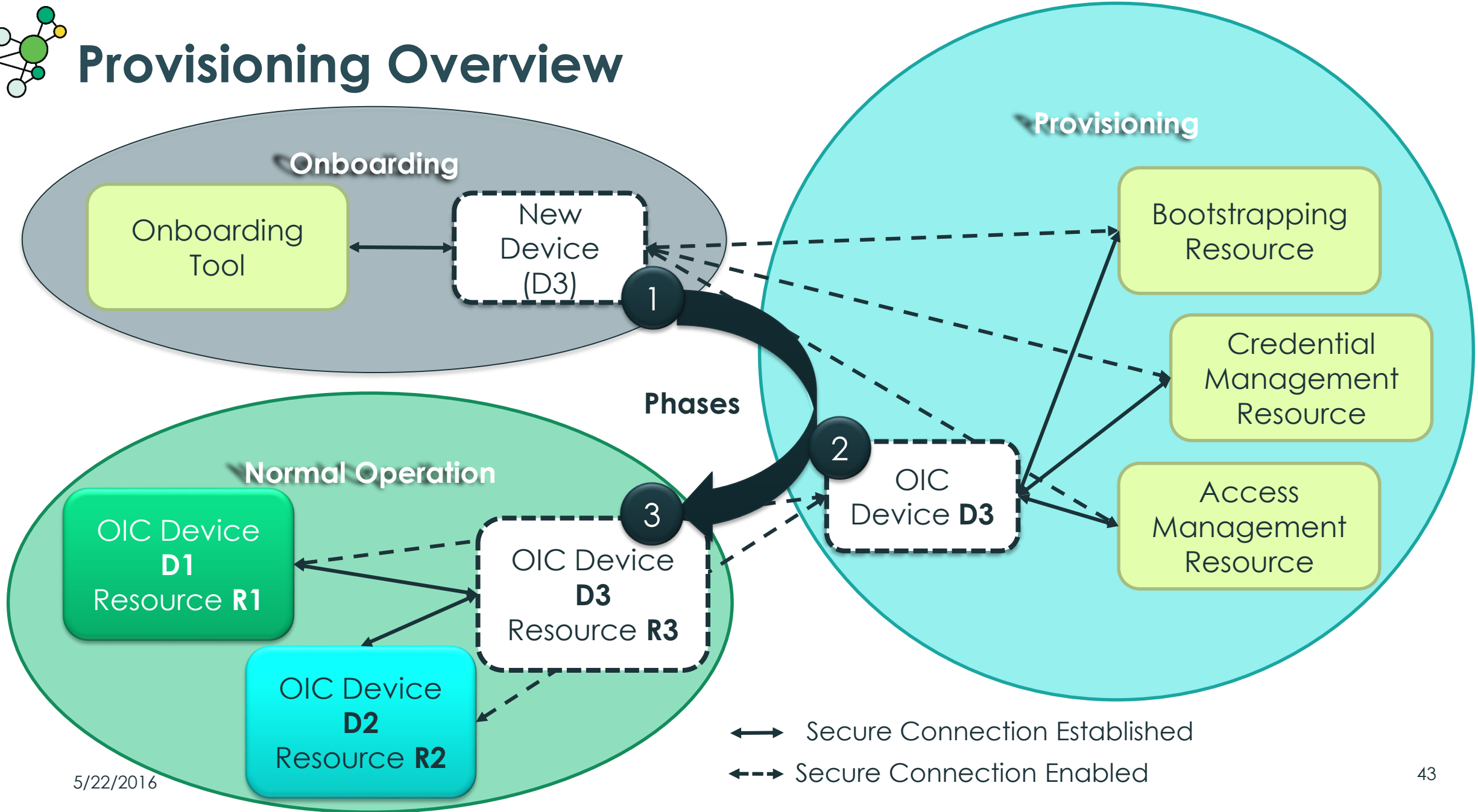
- Apply the **OIC Declarative Model** to security design
  - Security objects are *Resources*
  - Secure interactions are *Restful*
  - Built on OIC core communication stack
  - Data representations and serializations are the same as OIC core
    - (e.g. CBOR)

OIC Device Lifecycle



- OIC platform ships from manufacturer in "un-owned", operationally limited state
- A user connects the platform allowing discovery by an Onboarding Tool (OBT)
  - An "ownership transfer" allows secure provisioning by the OBT
- OBT provisions:
  Device identity / credentials
  Services – Credential Management, Access Management, Bootstrap.
- The platform reconnects as an OIC device
- The OIC device is discoverable by other OIC devices

Provisioning Overview



Ownership Transfer Methods

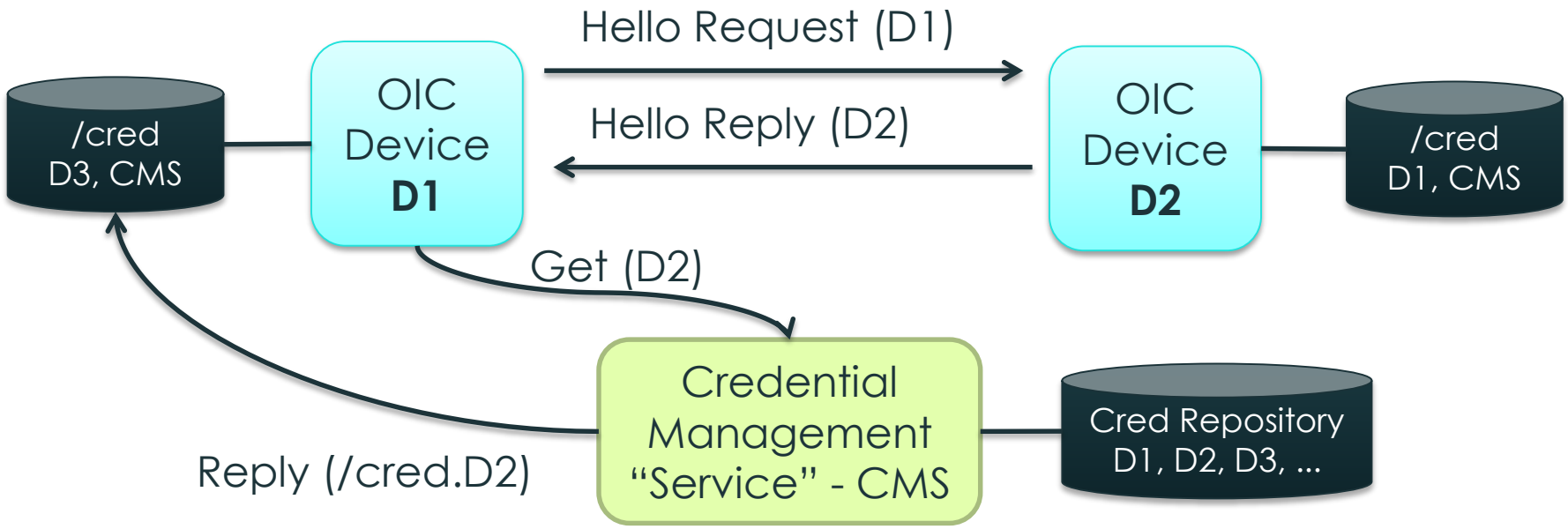
- Several Ownership Transfer Methods are defined to support a variety of manufacturing processes
  - *Just-Works*, Random-PIN, Manufacturer Certificates, Decentralized Public Key
- All are optional, but it is mandatory to implement at least one.
  - An OBT should implement all methods
  - Vendor specific methods are also supported
- OTMs may differ in terms of:
  - How a device establishes trust
  - How the physical owner's "intent" is proved
  - What cipher suites are used
- OTMs bring the device to a well-defined state
  - Device ID is known within the user's network
  - Device can be managed by user's network

Secure Communication

- CoAP over DTLS
- OIC device authentication
  - Pre-shared key
  - Certificates
- Secure session using TLS ciphersuites
  - TLS_PSK_ECDHE, TLS_ECDSA
- Credential Management
  - OBT may provision at device introduction.
  - CMS may provision proactively.
  - Device may request CMS provisioning dynamically.

Credential Management

- OIC devices may support symmetric or asymmetric keys
- Missing credentials could be provisioned dynamically



Credential Resource

Resource

/oic/sec/cred

Properties:

- **CredID:** Local credential reference
- **SubjectID**: OIC device
- **RoleID(s):** roles the subject may assert
- **CredType:** sym/asym/cert/…
- **PublicData, PrivateData, OptionalData**
- **Period:** Expiration period
- **Credential Refresh Method:** Used if nearing expiration
- **Rowner:** service that can modify this resource

Sample JSON

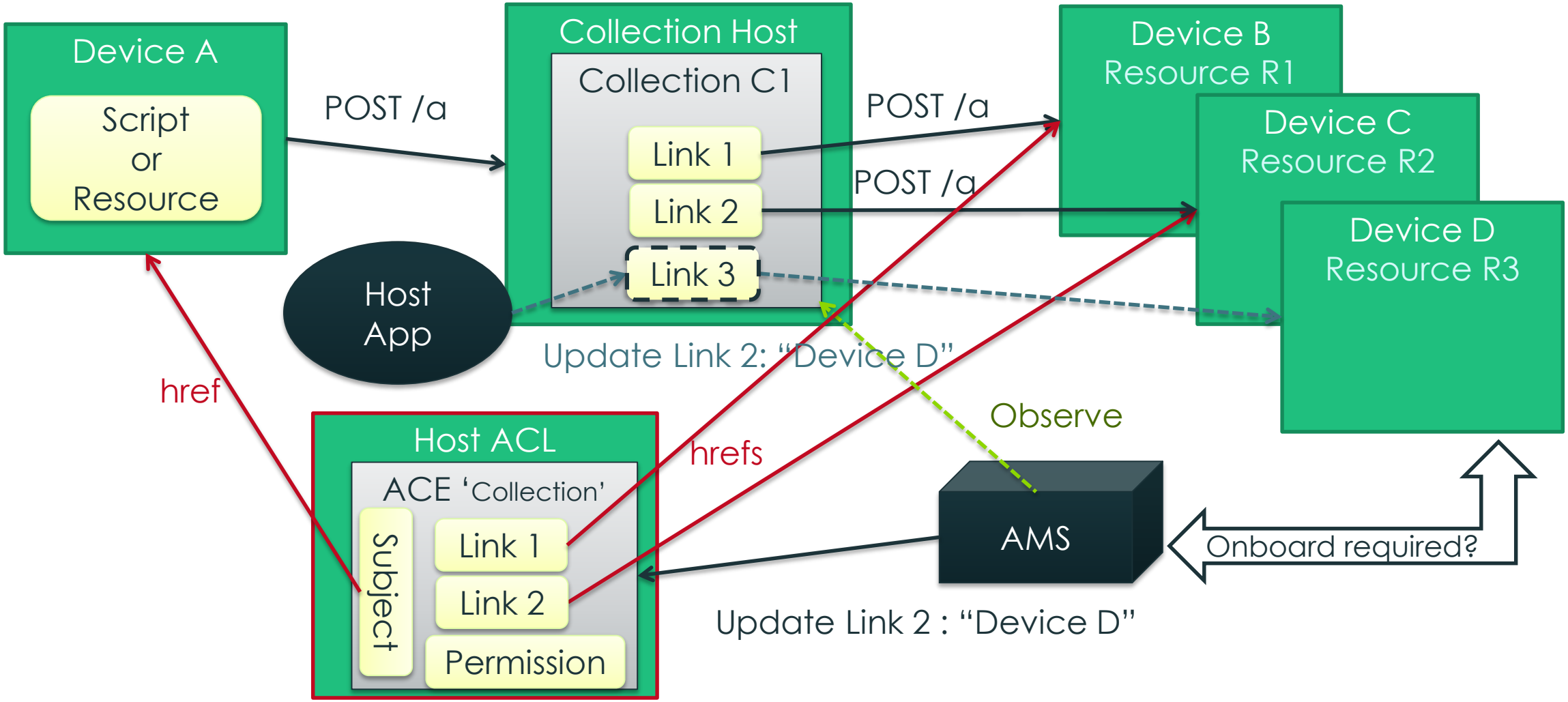
```
{
   ”CredID": ”1”,
   "SubjectID": ”device1”,
   ”RoleID”: ” ”,
   ”CredType": "1”, <symmetric pair-wise>
   ”PublicData”: “”,
   “PrivateData”: “ABCDEFGHIJKLMNP”,
   "Period": ”20150101T180000Z/20150102T070000Z",
   “Refresh: “oic.sec.crm.pro”,
   "Rowner": "oic.sec.ams"
}
```

Access Control

- Resources on the secured interface (that should be almost everything) are **only** accessible if there is an entry in the Access Control List resource
  - No ACL entry means no access
- An ACL says "X can do Y on resource Z"
  - X can be a device ID, a role, or a group (in the future)
  - Y can be any combination of CRUDN
  - Z can be any host resources or '*' wildcard
- If no ACL is present, and the device has an AMS configured, it can ask the AMS what authorization X has on Z.

Access Control with Collections

- AMS 'observes' Collection to proactively respond to change



Access Control Resource

Resource

/oic/sec/acl

Properties:

- **Subject:** *device, role or group*
- **Resource(s):** one or more URN
- **Permission:** bitmask of CRUDN
- **Period(s):** validity periods
- **Recurrence(s):** recurrence rule(s)
- **Rowner:** the service that owns this acl

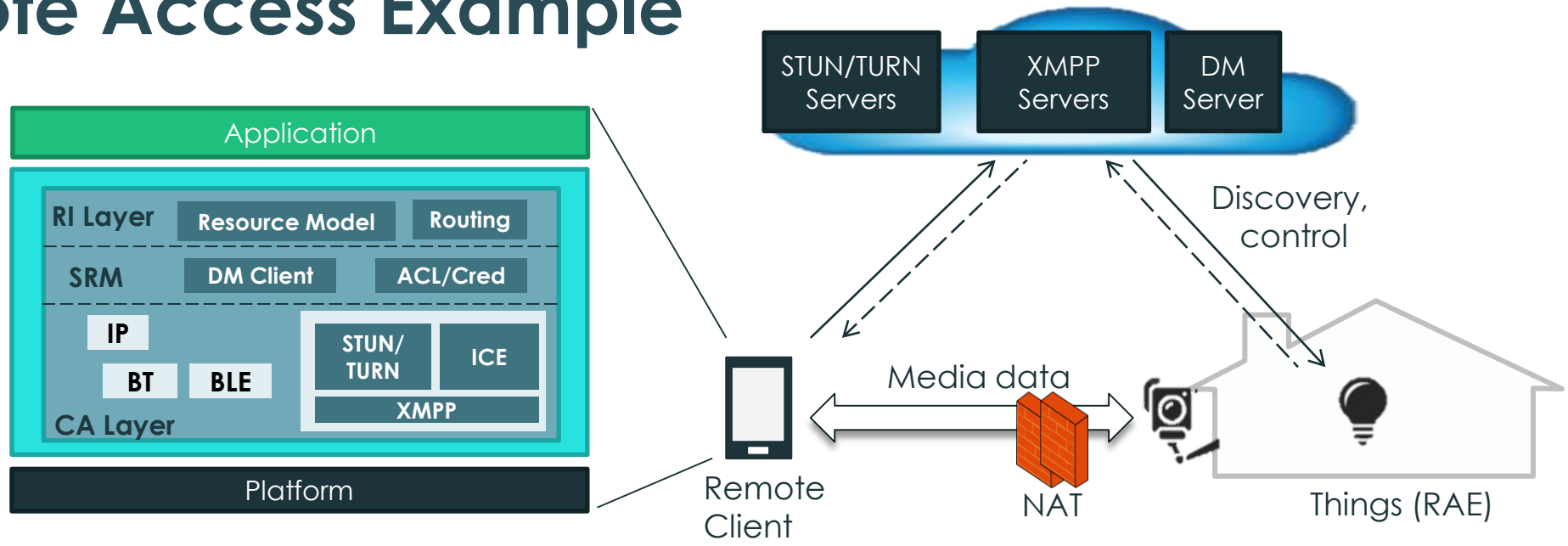
Sample JSON

```
{
   "Subject": "de305d54-75b4-431b-adb2-eb6b9e546014",
   "Resource": "/light",
   "Permission": "00000100", <i.e. CRUDN>
   "Period": "20150101T180000Z/20150102T070000Z",
   "Recurrence": "RRULE:FREQ=WEEKLY;UNTIL=20150131TO70000Z",
   "Rowner": "oic.sec.ams"
}
```

Remote Access Specification

Key Features

Remote Access Example



- Server Components:
  - Device Management Server: Device/Capability Registration and Authorization
  - STUN/TURN Server: Finding candidate address (reflexive and relay transport address)
  - Signaling Server: Delivering candidate address to recipient, discovery, presence, low BW data, SDP control
- Client Components: RA Endpoint (RAE) & RA-Proxy
  - XMPP Client
  - ICE Agent (optional
  - STUN/TURN Client
  - DM Client

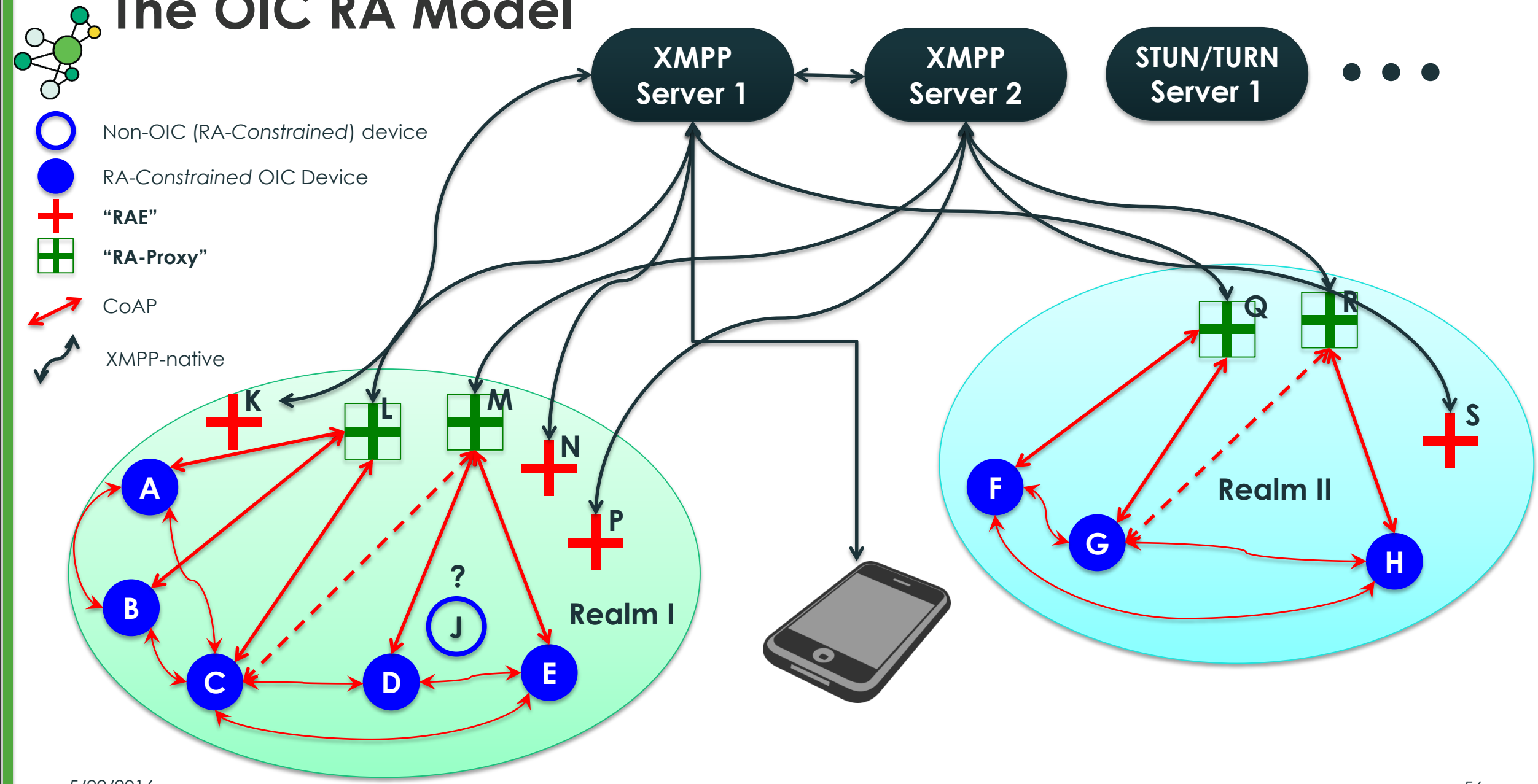
Remote Access ("RA") in OIC – Terminology

- Remote Access endpoint Devices:
  - Remote Access Endpoints ("RAE"):
    - OIC Servers *also* capable of XMPP, optionally capable of ICE-client
  - Remote Access Proxies ("RA-Proxy"):
    - Superset of RAE – Capable of 'representing' "RA-constrained devices"
      - "RA-Constrained": Devices incapable of *natively* supporting RA tech
- Cloud Components:
  - XMPP Server(s)
  - STUN/TURN Server(s)
- Security objective:
  - Protect RAE / RA-proxy connection to XMPP server

Remote Access using XMPP

- Format for bare-JIDs (owner) and full-JIDs (for RAEs)
  - Includes JID-Resource overloading for:
    - OIC Spec version
    - Device-type
    - UUID
- Mapping from Core/Smart-Home Resources to full-JID format
  - Allows for Presence, Remote Discovery, XMPP-Roster-based access, more

The OIC RA Model



OIC Specification overview

Smart Home Device and Resource Specification

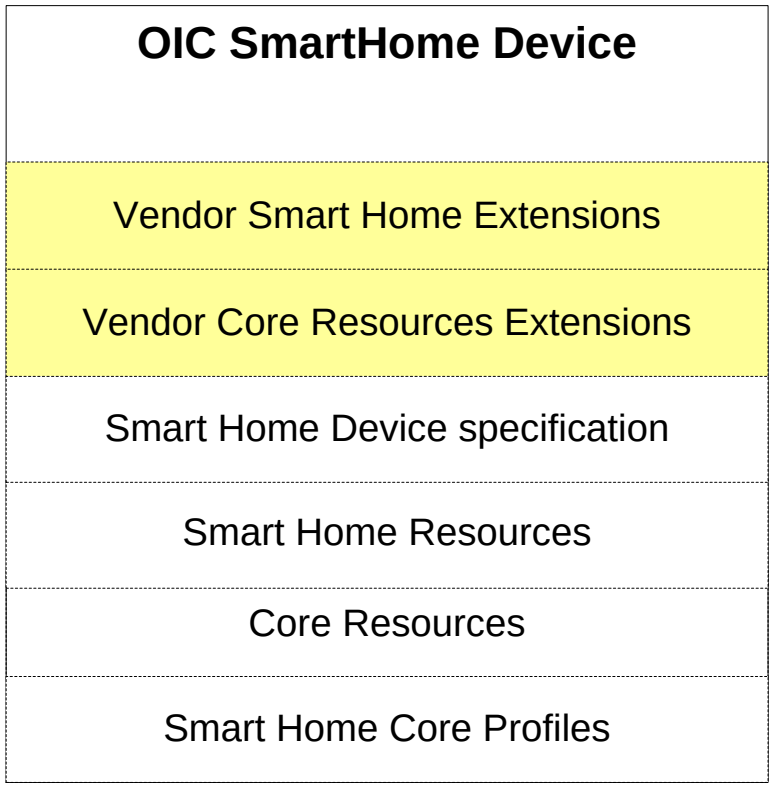
Smart-home Specifications

- Specifications are split in 2 documents:
  - Device specification
  - Resource specification

The Device specification uses the resources defined in the resource specification

Smart-home Specific Device Specification

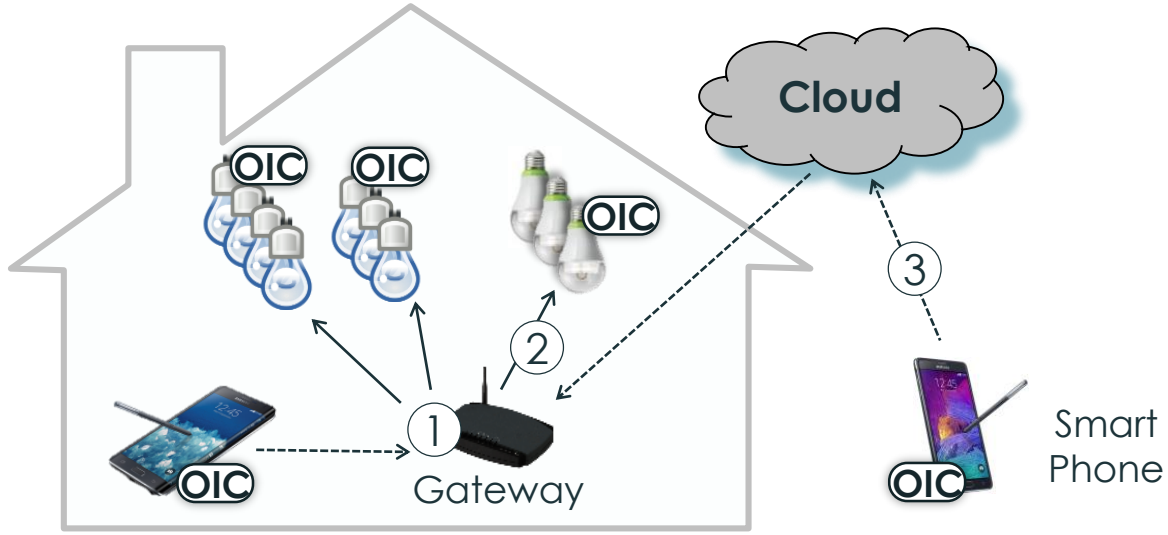
- Contains profiles of
  - OIC Core Framework specification
  - OIC Security specification
- Contains list of smart home devices
  - RAML & JSON Schema
- Each Smart home device definition contains:
  - type identifier (rt)
  - a list of mandatory resources



Smart Home Use Cases

- Selected key enabling use cases to scope activity

Use Case	Priority
Indoor Environment Control	1
Lighting control	
Energy Saving Washer/Dryer	
Energy Management	
Remote Access for Device Control	
Smart watch notify and control	6
Smart Video Environment	3
Smart Home Office	
Smart Garage	
Device Grouping and Control	
Multi player gaming	7
Smart watch gaming on TV	
Fire safety monitor and Notify	4
Keyless Entry	2
Home Security	
Health Monitor and Notify	5



① Control proximal OIC Devices
② On board new Devices
③ Control remotely with an OIC Client

Example Smart Home Device: IPCamera Resource – pan, tilt, zoom

- Resource Types - rt:
  - Physical device: 'rt'=oic.r.movement.ptz
  - Digital image: 'rt'=oic.r.image.ptz
- Supported Interfaces/CRUDN:
  - oic.if.a (actuator)

Property	Value/Type	Read/ Write	Mandatory	Comments
pan	Number	rw	M	[-180,180], where 0 is default position. Integer by default, float if range indicates as such.
tilt	Number	rw	M	[-180,180], where 0 is default position, , float if range indicates as such.
panRange	CSV	r	O	Min, max range (If includes decimal point accuracy then float)
tiltRange	CSV	r	O	Min, max range (If includes decimal point accuracy then float)
zoomFactor	string	rw	M	Value determined by allowed range
zoomFactorRange	Enum	r	O	Enum Values: {linear, 1x, 2x, 4x, 8x, 16x, 32x} 'linear' applies to optical zoom and equates to a range of 1-100.
				Note that this resource can be reused as offset

Other Resources – Camera Settings Controls

Resource	Properties	Value/Type	Comments
Auto White Balance (oic.r.colour.autoWhiteBalance)	autoWhiteBalance	boolean	True= auto white balance is on, False = auto white balance is off
Colour Saturation (oic.r.colour.saturation)	colourSaturation	integer	Range 0-100; 0 = black and white images; 50 = device specific normal colour; 100 = very full colour images
Night Mode (oic.r.nightMode)	nightMode	boolean	True – night mode on, False – night mode off.
Auto Focus (oic.r.autoFocus)	autoFocus	boolean	True – auto focus on, False – auto focus off.

Thank you!