IoTivity – Connecting Things with IoT

Ashok Subash

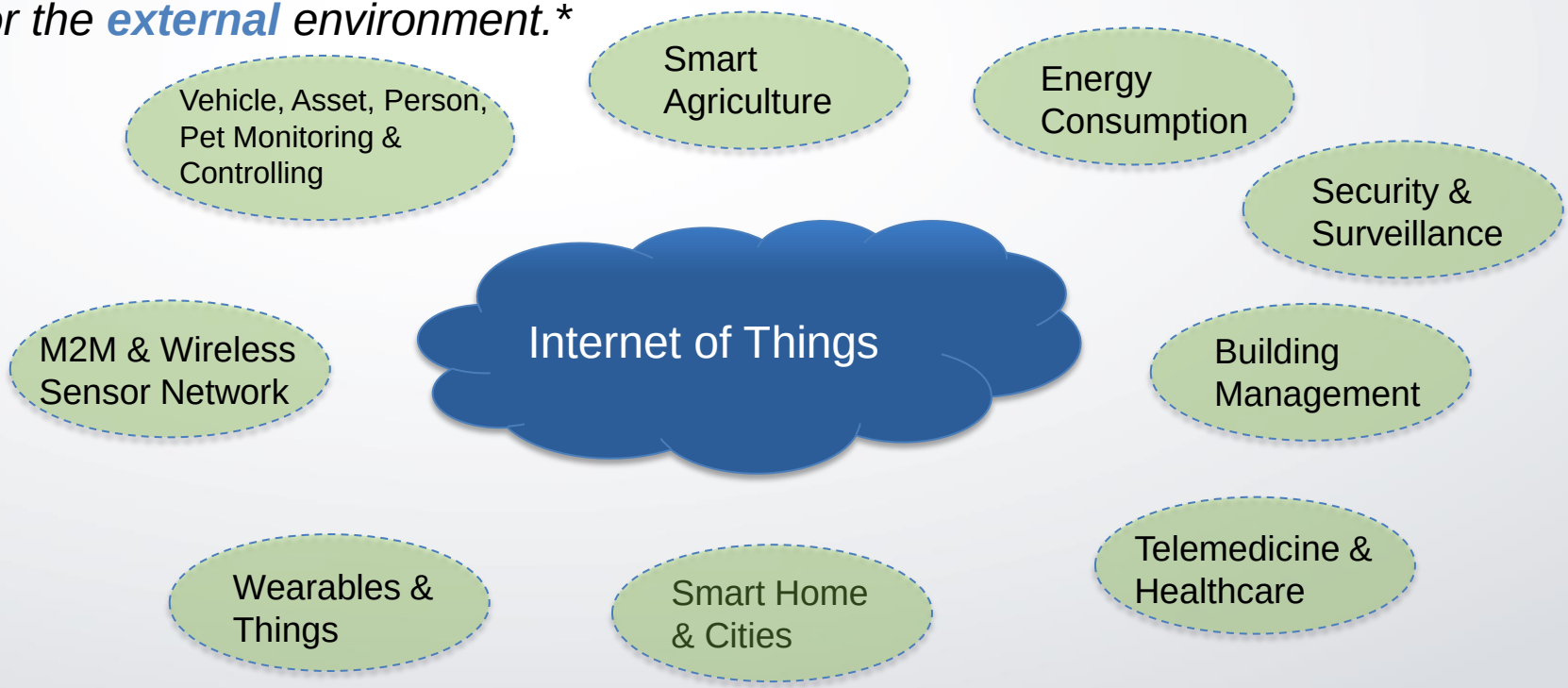
Samsung Electronics

Agenda

- Overview of IoT, OIC & IoTivity
- IoT Stack, Topologies & Protocol
- IoTivity Architecture
- Programming IoTivity Core APIs
- Programming IoTivity Service APIs
- Summary

Internet of Things – What is it?

*The Internet of Things is the **network of physical objects** that contains embedded technology to **communicate** and **sense or interact** with the objects' **internal** state or the **external** environment.**



*Gartner, July 2014

What makes IoT Possible?

- H/W Miniaturization & Lower BOM Cost
- Advancements in Sensor Technology
- Low Power Connectivity Technologies
- IP as key Interoperability Protocol
- Devices ability to run on battery for longer duration (> 10 years)

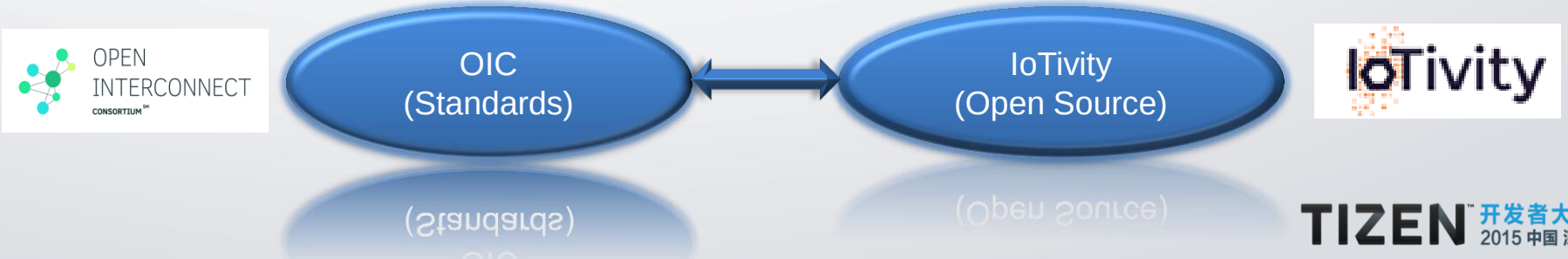
What's hindering IoT?

- Non IP based standards technology resulting in limited Interoperability
- Proprietary Protocols & Technologies
- Licensing issues
- Companies creating "Closed Ecosystem" (Zero or limited Interoperability with other Vendor devices)
- Low adoption of Open Standards by various Industry Consortiums
- Security & Privacy concerns
- Dilemma in "Ownership" of Data from variety of IoT devices
- Lack of Strong Certification for ensuring Protocol & Application Profile Interoperability

Need **Standards** & **Reference Implementation** which cater to these issues

OIC & IoTivity

- Open Interconnect Consortium (OIC)
  - Founded by Leading Technology Companies including Cisco, GE, Intel,MediaTek & Samsung
  - Defines standards for connectivity requirements
  - Ensures interoperability of billions of Internet of Things (IoT) devices.
- IoTivity
  - An open source software framework implementing OIC Standards
  - Ensures seamless device-to-device connectivity to address the emerging needs of the Internet of Things.



Key Focus of OIC

- OIC Standards addresses multiple vertical domains including Home Automation, Automotive, Enterprise, HealthCare, Industrial scenarios
- Initial focus on Smart Home & Office solutions
- Adopt Open Standards like IETF when applicable & standardize on areas, not addressed

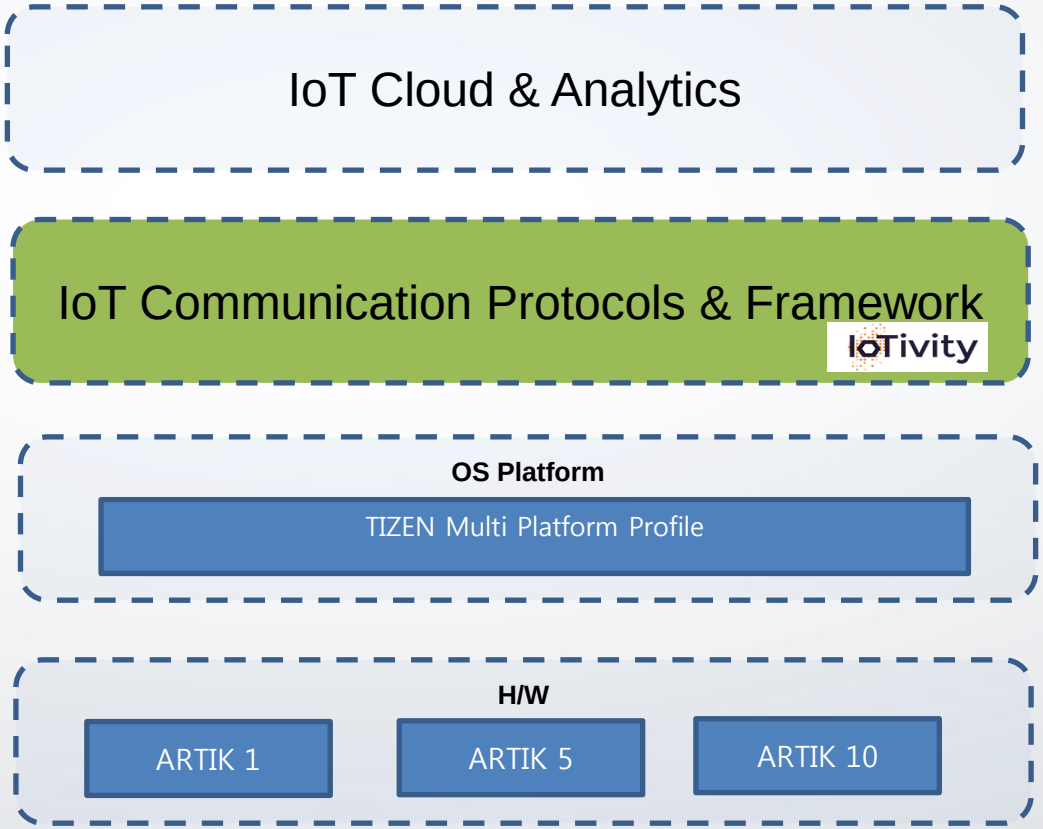
http://openinterconnect.org/

Key Focus of IoTivity

- Open Source Framework implementing OIC Standards
- Licensed under Apache License Version 2.0
- Available on TIZEN, Android, Arduino, Linux(Ubuntu) Platforms
- Provide APIs at 2 Levels
  - ✓ IoTivity Base
  - ✓ IoTivity Services

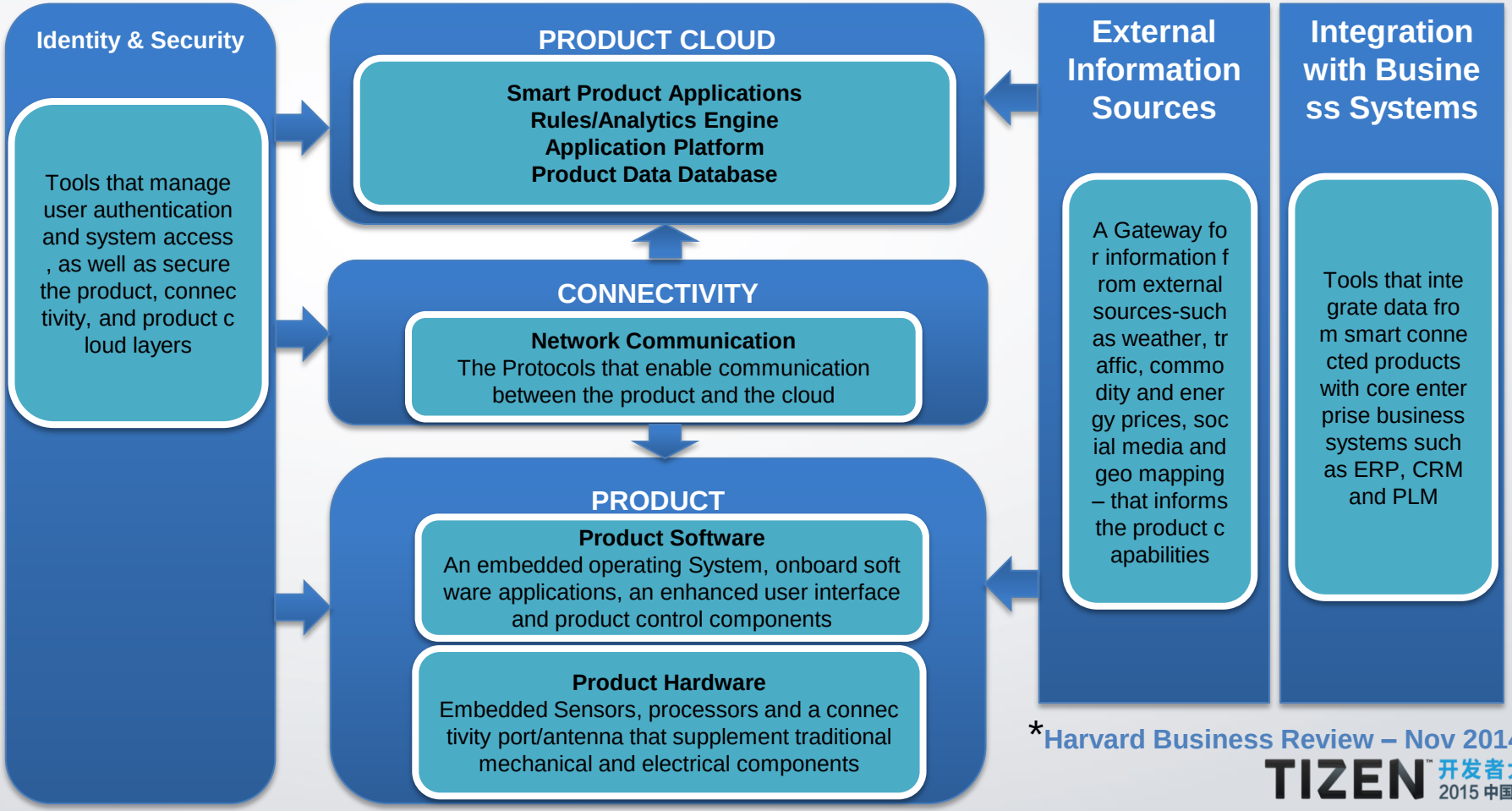
https://www.iotivity.org

TIZEN based IoT EcoSystem



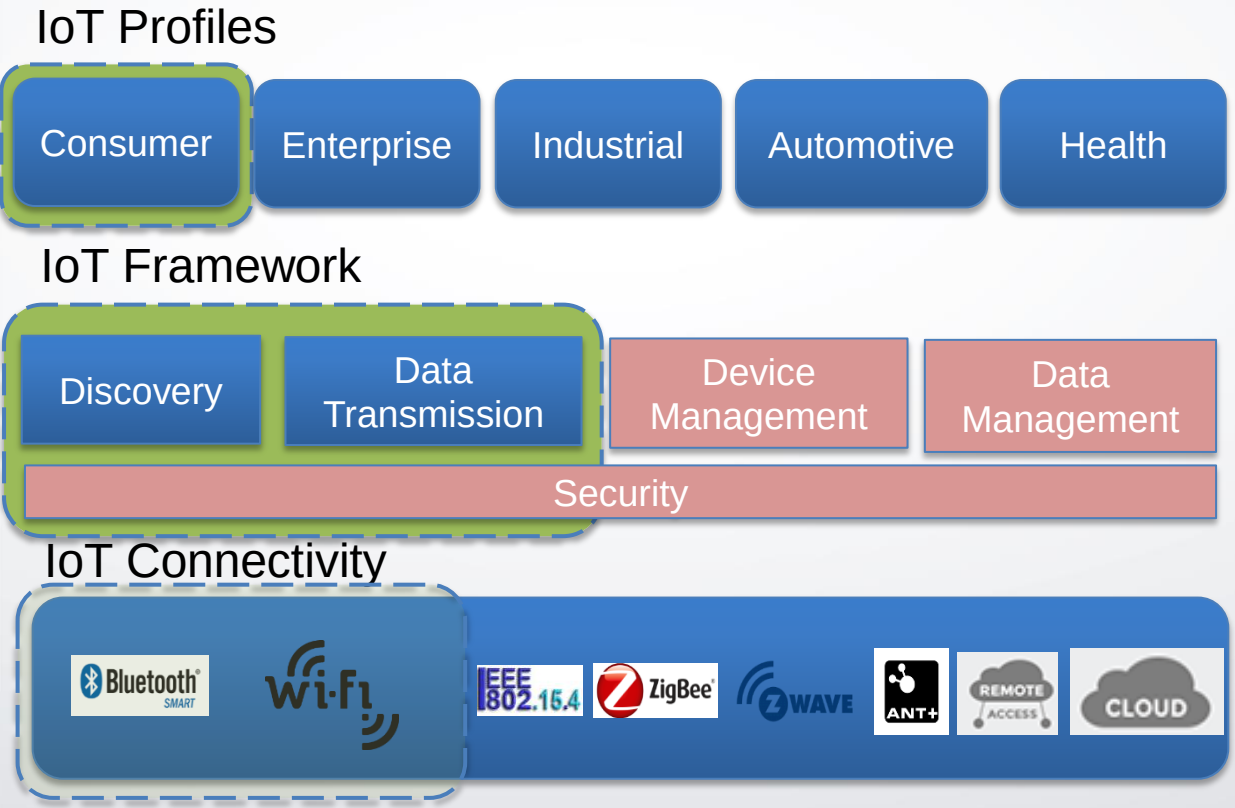
IoT Stack, Topologies & Protocol

IoT Technology Stack – End to End - Executive View*



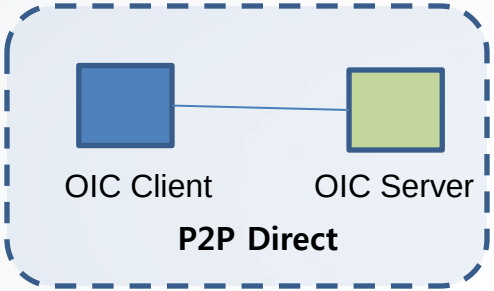
*Harvard Business Review – Nov 2014

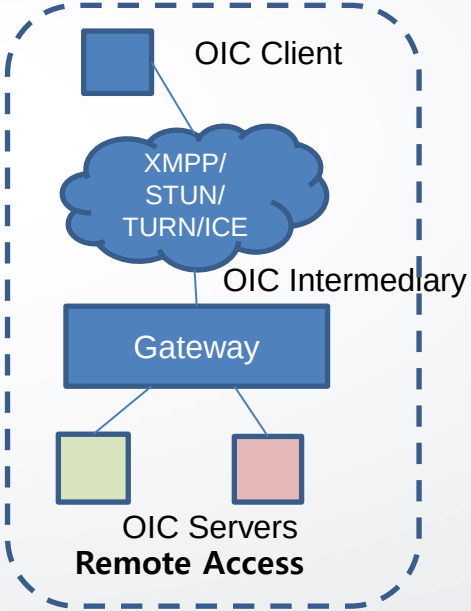
IoT Technology Stack – Technical View

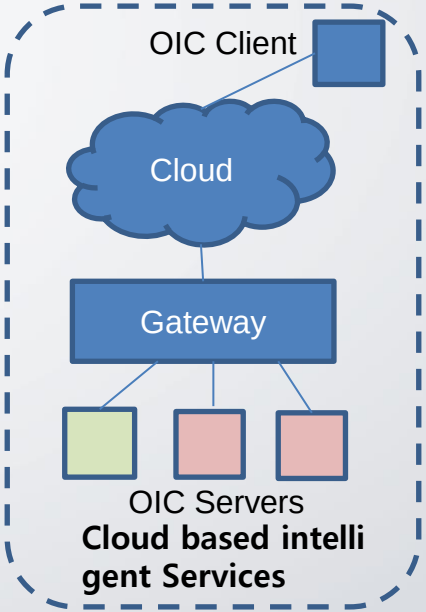


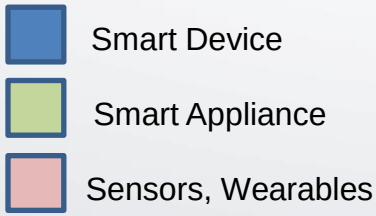
- Common Solution
- Established Protocols
- Security & Identity
- Standardized Profiles
- Interoperability
- Innovation Opportunities
- Necessary connectivity

OIC - Topologies

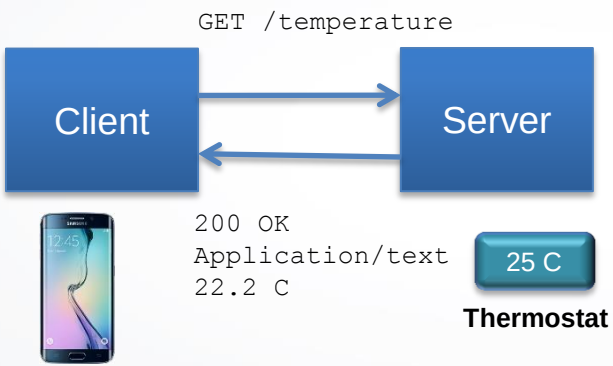








Concept of Resource & RESTful Architecture

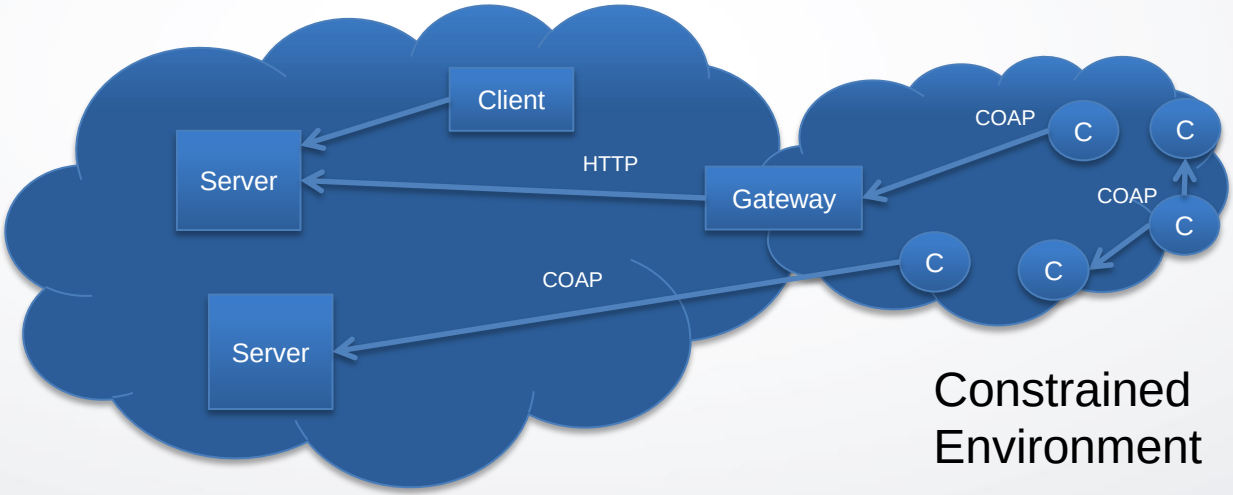


- Client-Server
- Stateless
- Cache
- Uniform Interface
- Layered System

REST Principles

Resource: Any information that can be named
e.g. Document, Image, a collection of other resources, non-virtual objects (Things)

Constrained Application Protocol (CoAP)



CoAP

- Open IETF Standard (RFC 7252)
- Compact 4 Byte Header
- UDP (Default), SMS, TCP Support
- Strong DTLS Security
- Asynchronous Subscription
- Built-In Discovery

Constrained Environments – What is it?

- Limitations on Code Size (ROM/Flash)
- Size of State & Buffers (RAM)
- Processing Power (CPU)
- Power Consumption
- User Interface & Accessibility in deployment

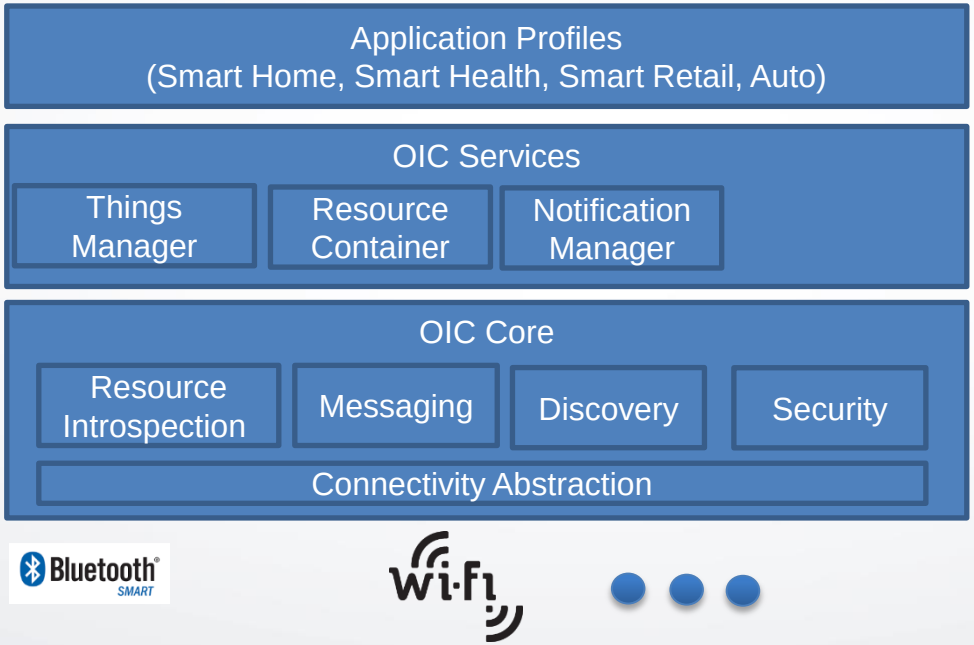
Constrained Nodes

- Low achievable bitrate/throughput
- High packet loss & high variability of packet loss (delivery rate)
- Highly asymmetric link characteristics
- Severe penalties for using larger packets
- Limits on reachability over time
- Lack of advanced services such as IP multicast

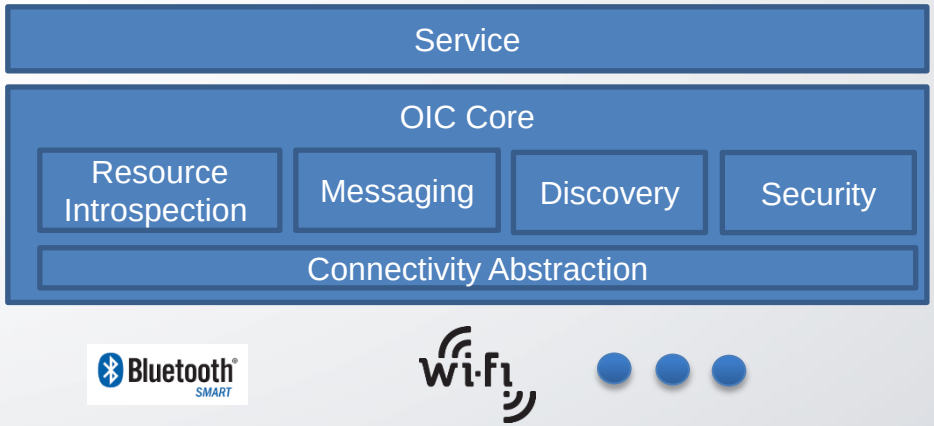
Constrained Networks

IoTivity Architecture

IoTivity – High Level Architecture



Smart Devices (OIC Clients)
(Smartphone, SmartTV, SmartHub etc)

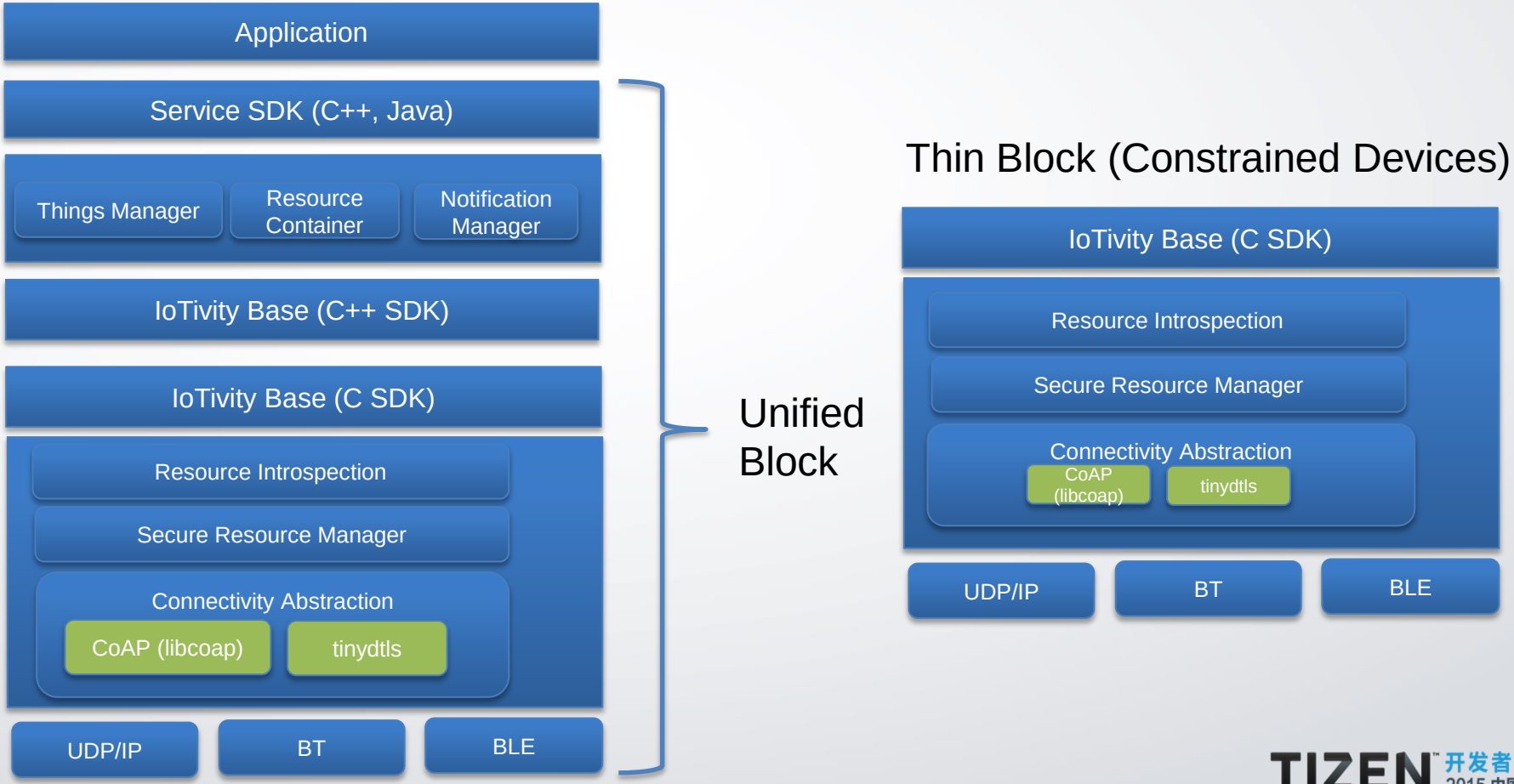


Thin Device (OIC Server)
(Thermostat, Motion Sensor etc)

IoTivity Framework – Key Functionality

Functionality	Description
Discovery	IoTivity discovery supports multiple discovery mechanisms for devices and resources in proximity and remotely
Data Transmission	IoTivity data transmission supports information exchange and control based on a messaging and streaming model
Data Management	IoTivity data management supports the collection, storage and analysis of data from various resources.
Device Management	IoTivity device management supports configuration, provisioning and diagnostics of devices.

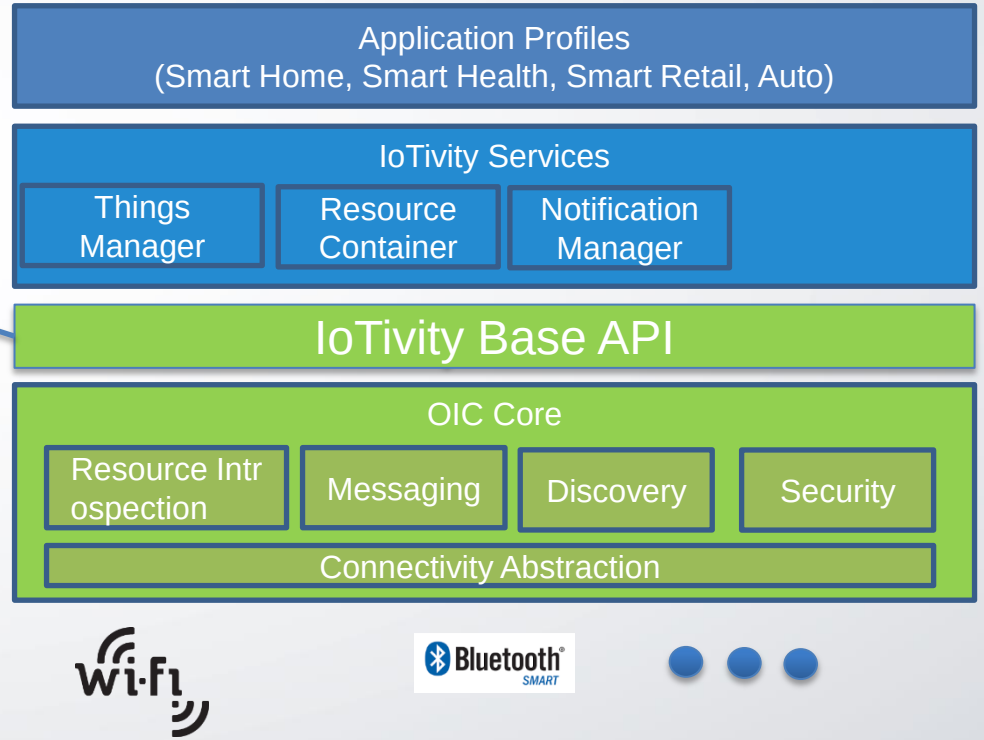
IoTivity Module View



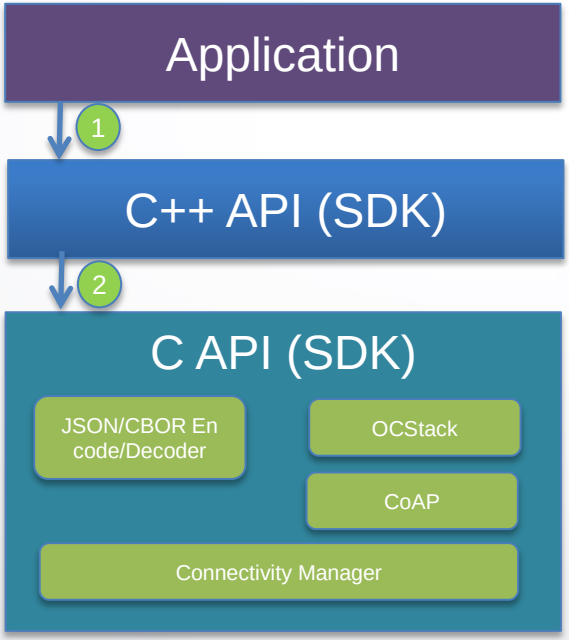
Programming IoTivity Core APIs

Steps involved in using IoTivity Core API

- Registering a Resource
- Finding a Resource
- Querying a Resource State
- Setting a Resource State
- Observing Resource State



Registering a Resource – Call Flow



Resource URI

Resource Type

```
OCPlatform::registerResource(resourceHandle, "/light/1", "core.light",
                             "oic.if.baseline", entityHandlerCb,
                             OC_DISCOVERABLE | OC_OBSERVABLE);
```

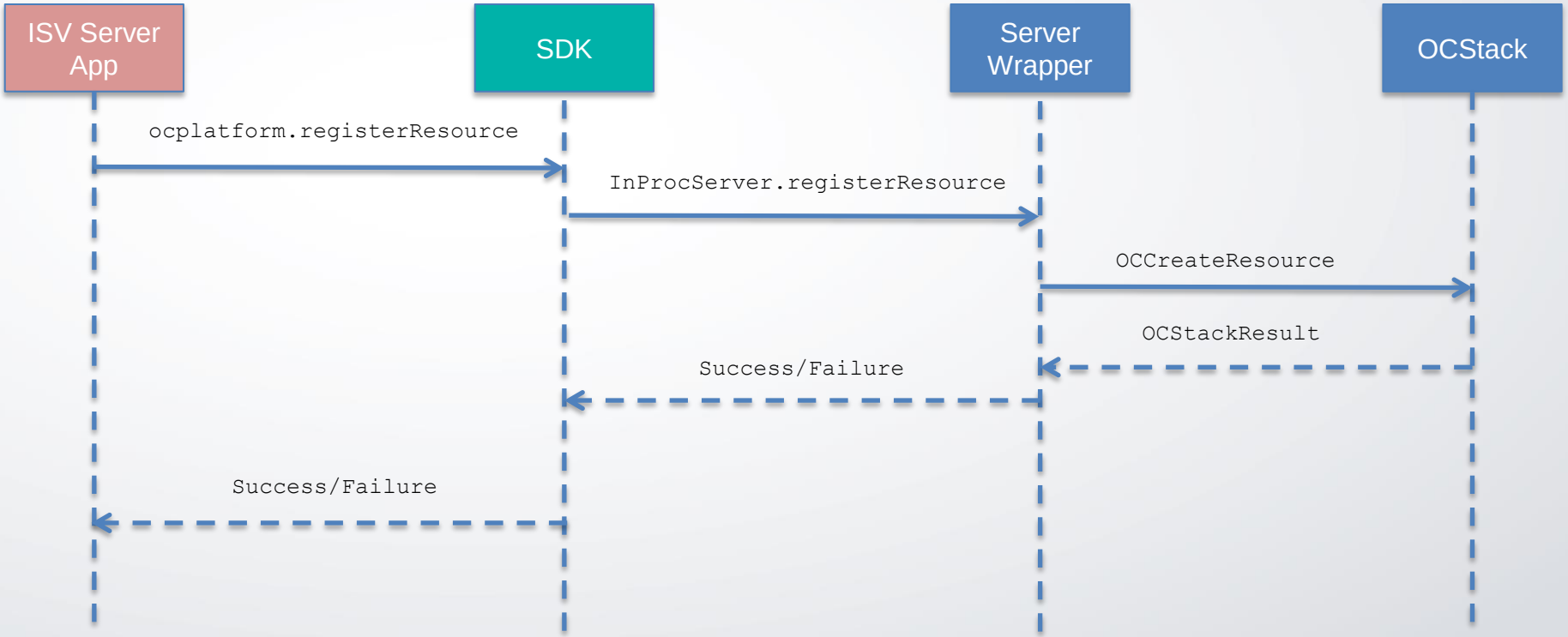
Resource Interface

Entity Handler

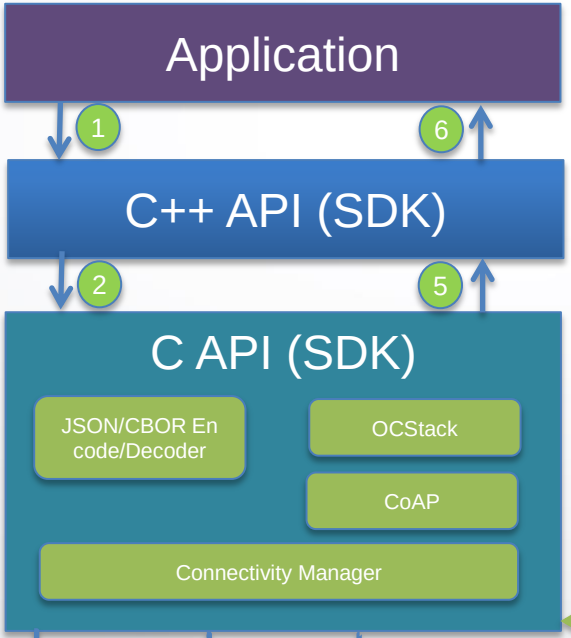
Resource Property

```
OCCreateResource(resourceHandle, resourceTypeName.c_str(),
                 resourceInterface.c_str(), resourceURI.c_str(),
                 EntityHandlerWrapper, NULL, resourceProperties);
```

Registering a Resource – Sequence Diagram



Finding a Resource – Call Flow



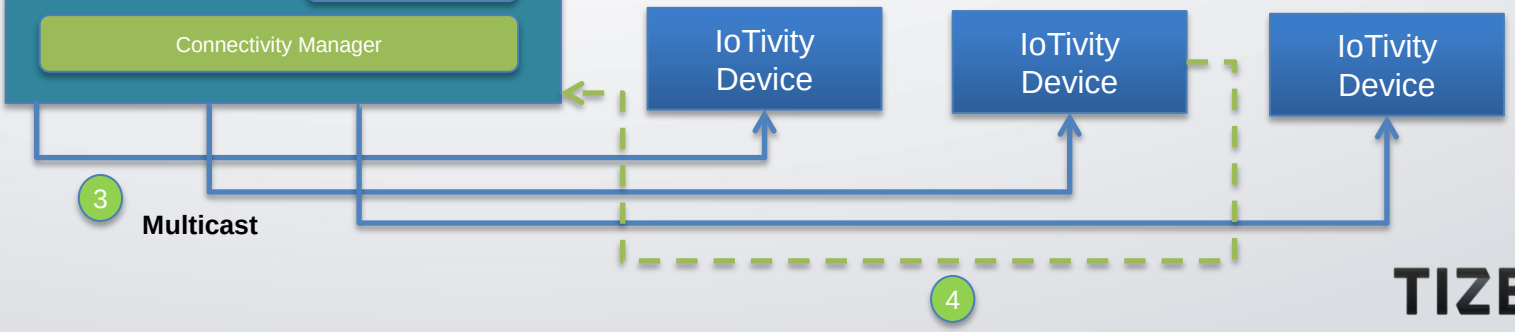

```
OCPlatform::findResource(host, "/light/1", connectivityType,
                         resourceHandlerCb);
```

```
OCDoResource(resourceHandle, OC_REST_GET, "/light/1", 0, payLoad,
connectivityType, qos, &cbData, headerOptions, numOptions);
```

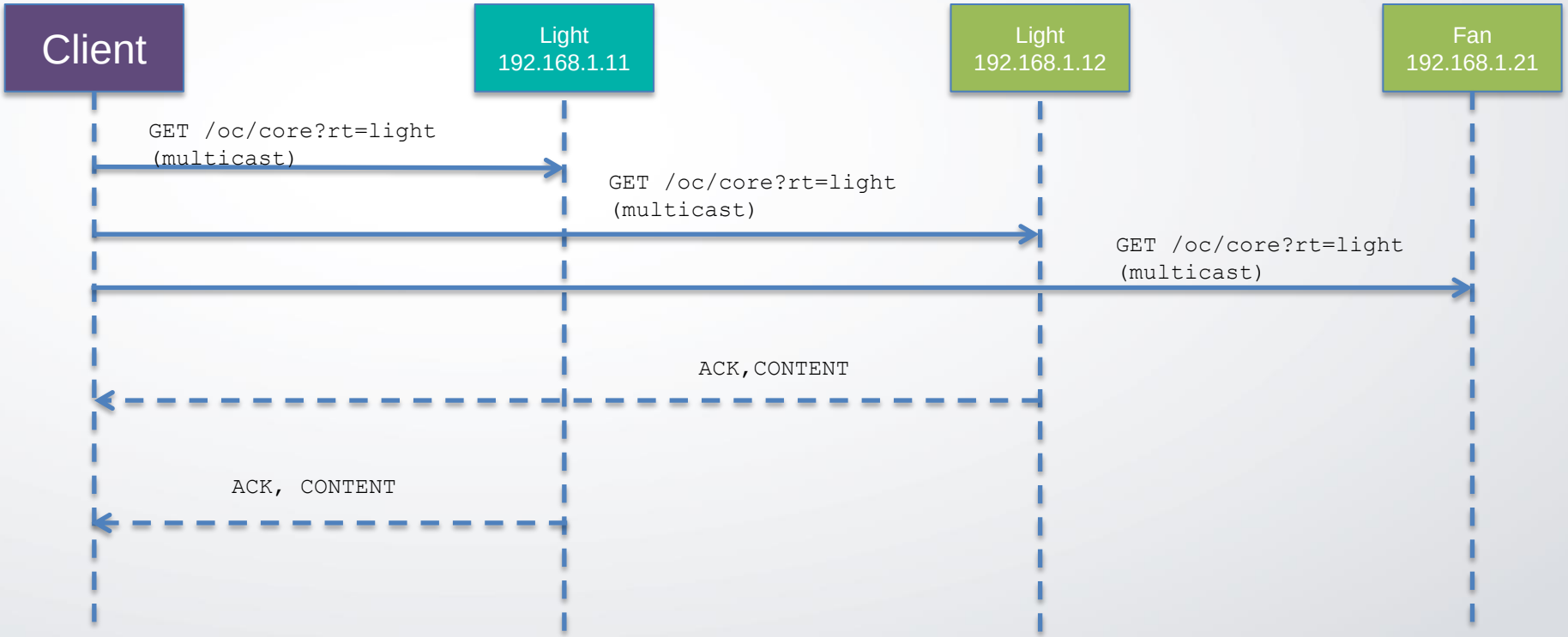
```
CASendRequest(endPoint, &requestInfo);
```

Sends a multicast query

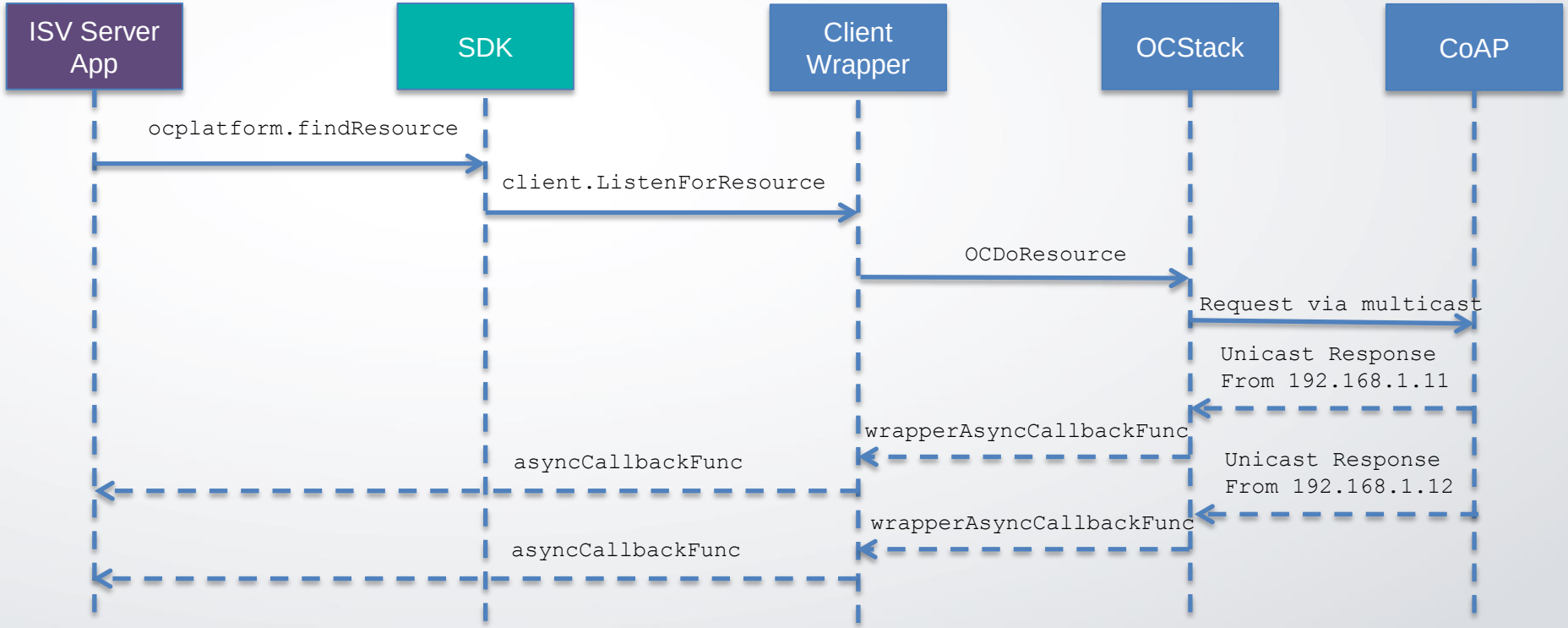
//Devices that matches the query answers as indicated below



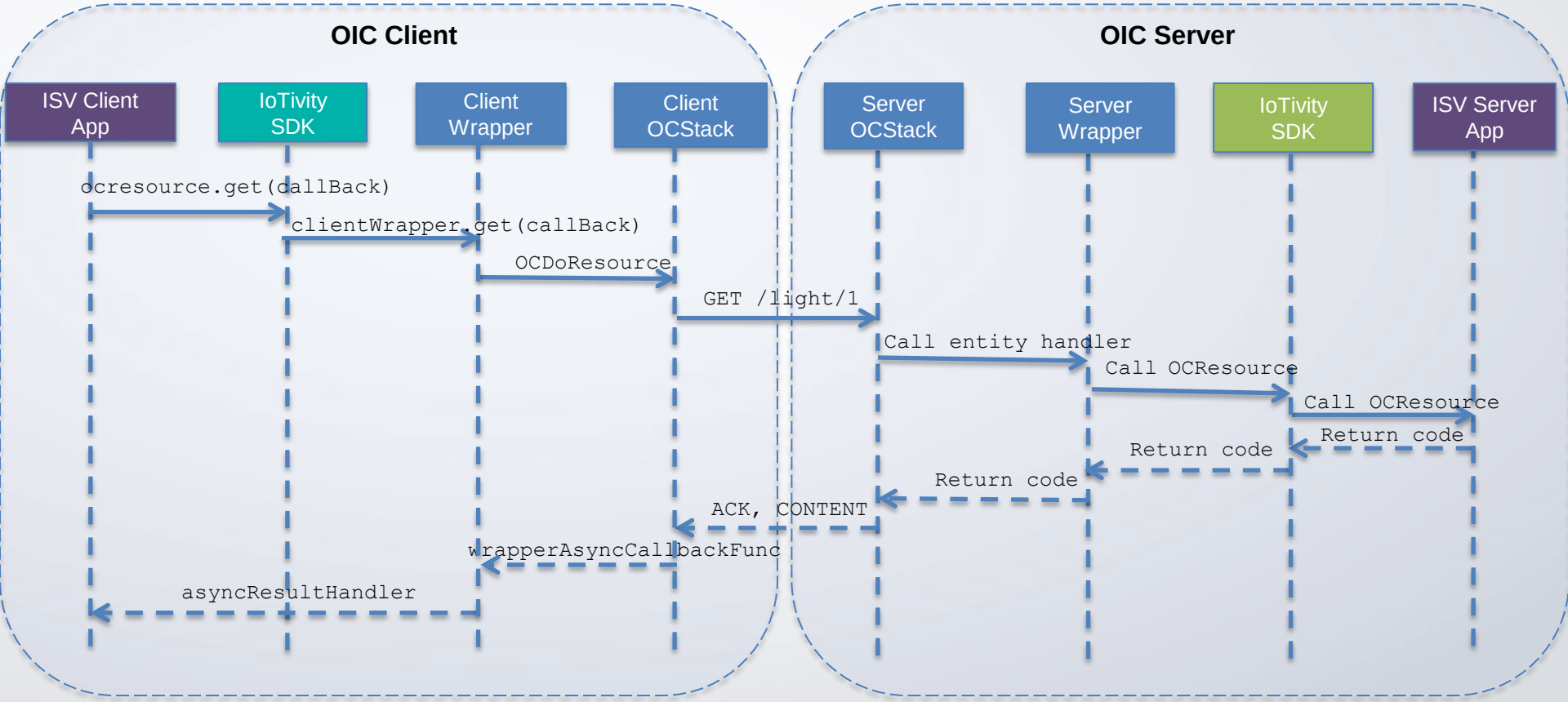
Finding a Resource – System Sequence Diagram



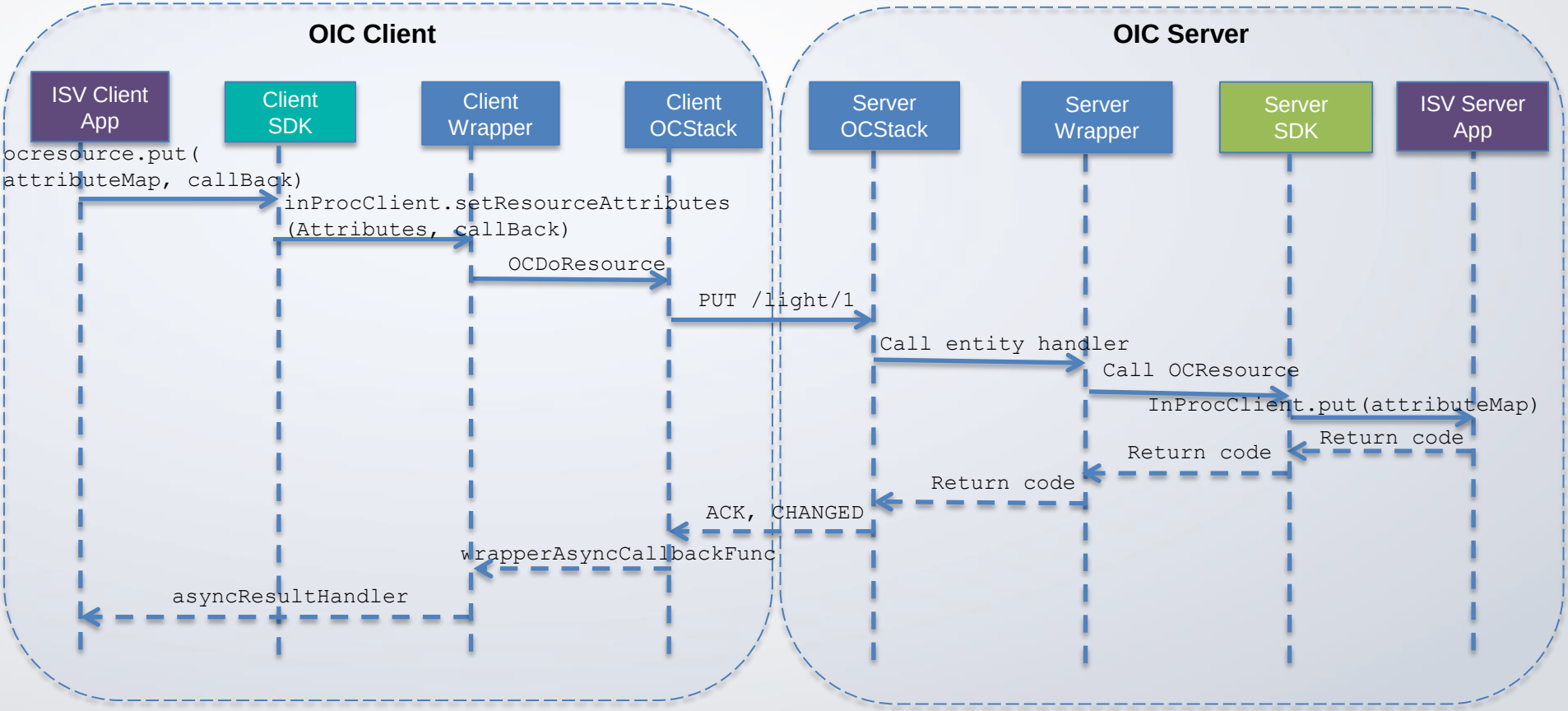
Finding a Resource – Sequence Diagram



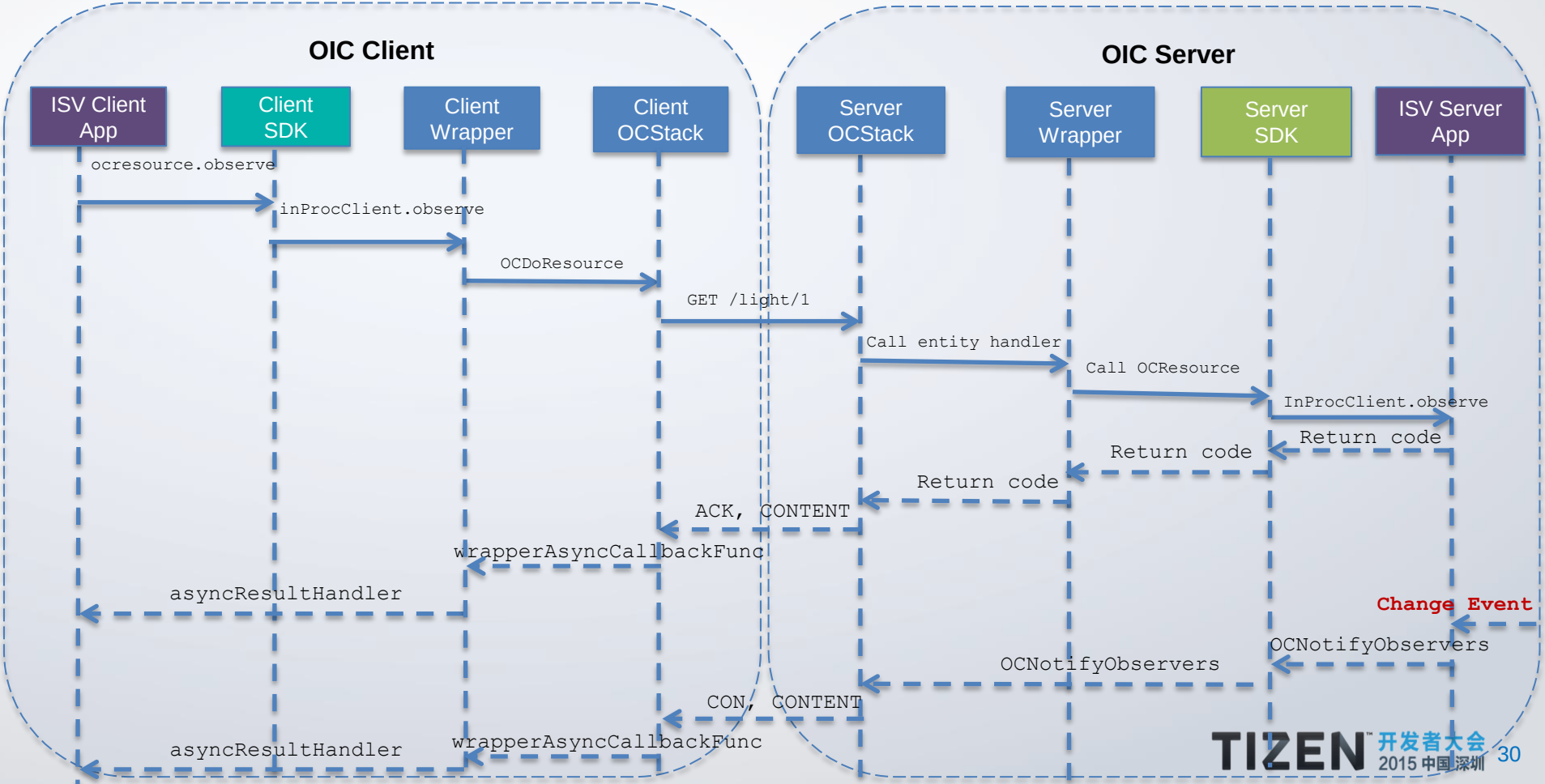
Querying Resource State [GET]



Setting a Resource State – Sequence Diagram

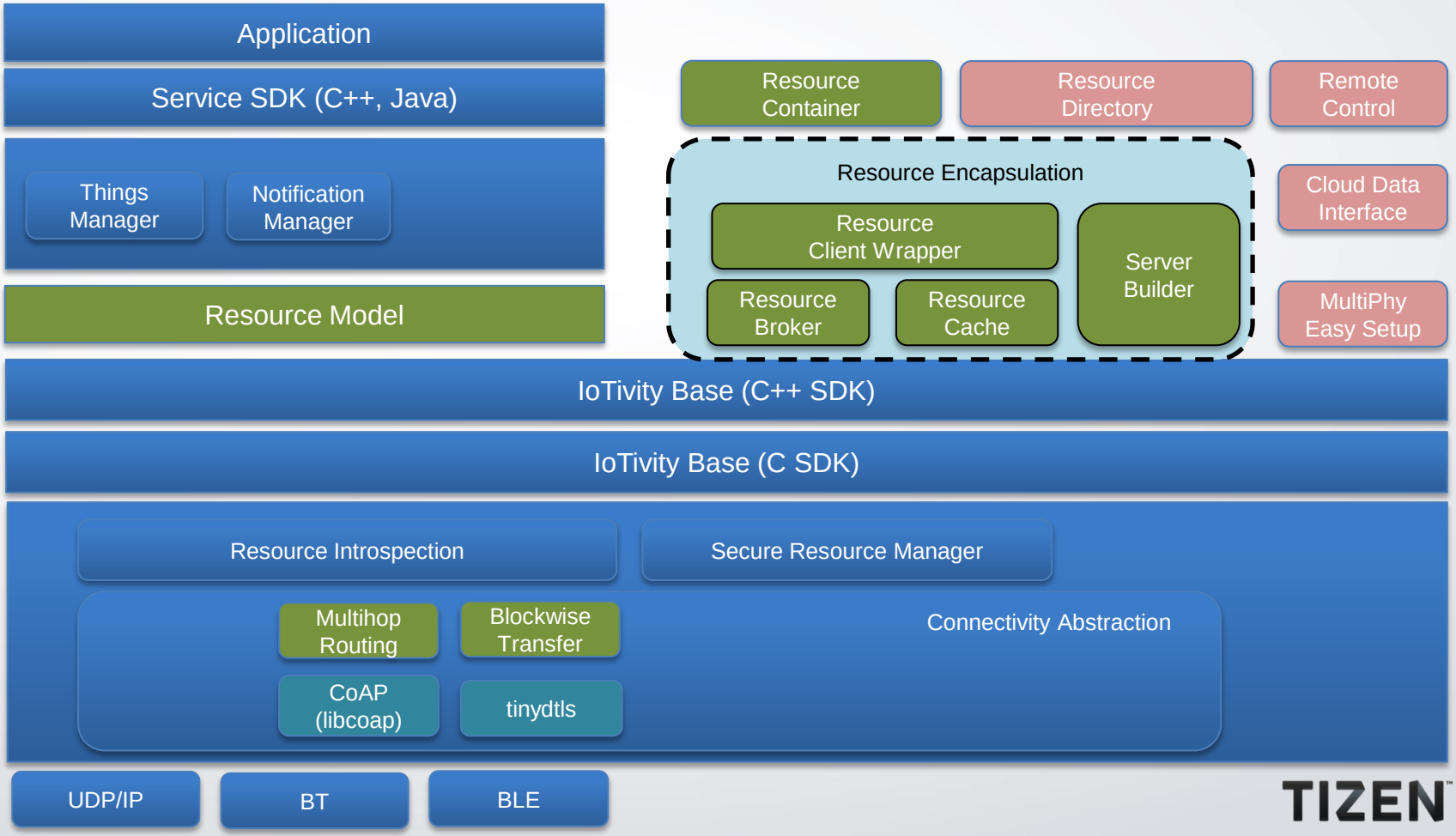


Observing Resource State



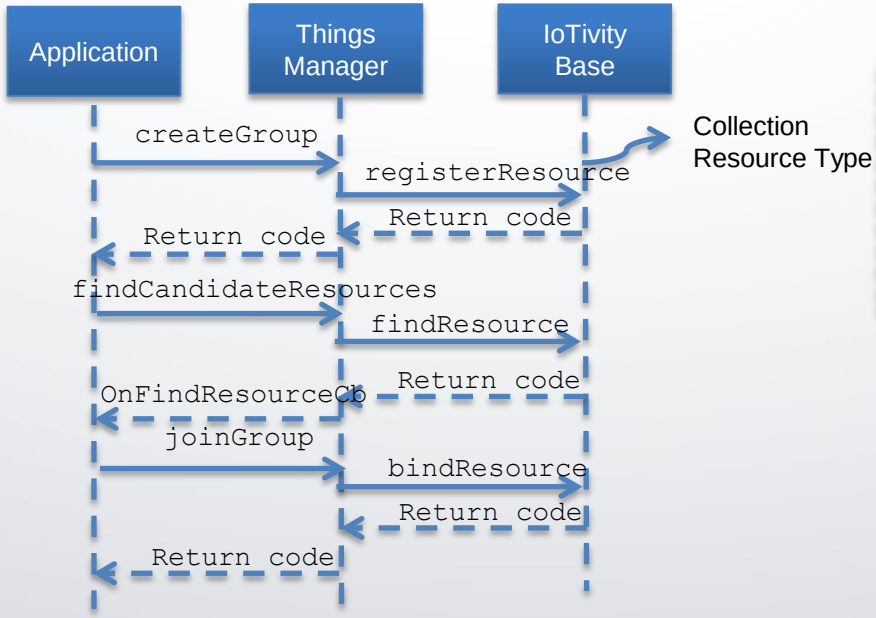
Programming IoTivity Service APIs

IoTivity Core & Primitive Services Update



Things Manager

- **Group Management**
  - ✓ Find candidate devices to form a group
  - ✓ Create a group of found devices
  - ✓ Create a group action for the group
  - ✓ Execute the group action



Collection Resource Type

Action Set

Action Set Name

Delay/Step of Time Delay

Type of Group Action (normal, scheduled, recursive)

```
OutofHome*10 1 *uri=coap//10.251.44.228:49858/a/light|power=on*
uri=coap://10.252.44.221:49458/b/aircon|power=off
```

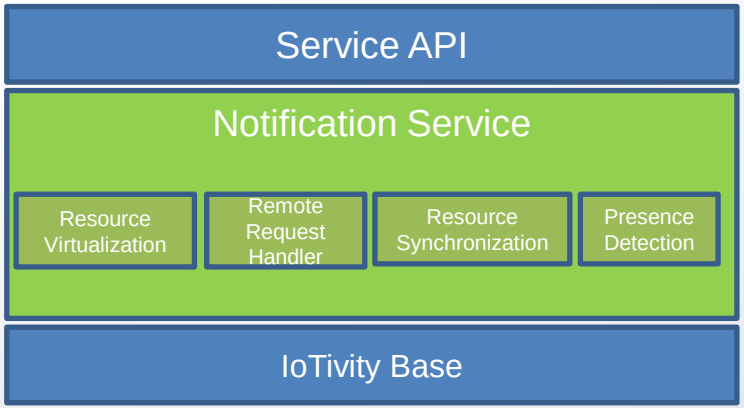
Things Manager – Configuration & Diagnostics

- **Things Configuration**
  - ✓ Server Side - Bootstrapping requisite information from a bootstrap server to access other IoT services
  - ✓ Client Side – Getting/Updating system configuration parameters from/to multiple remote things

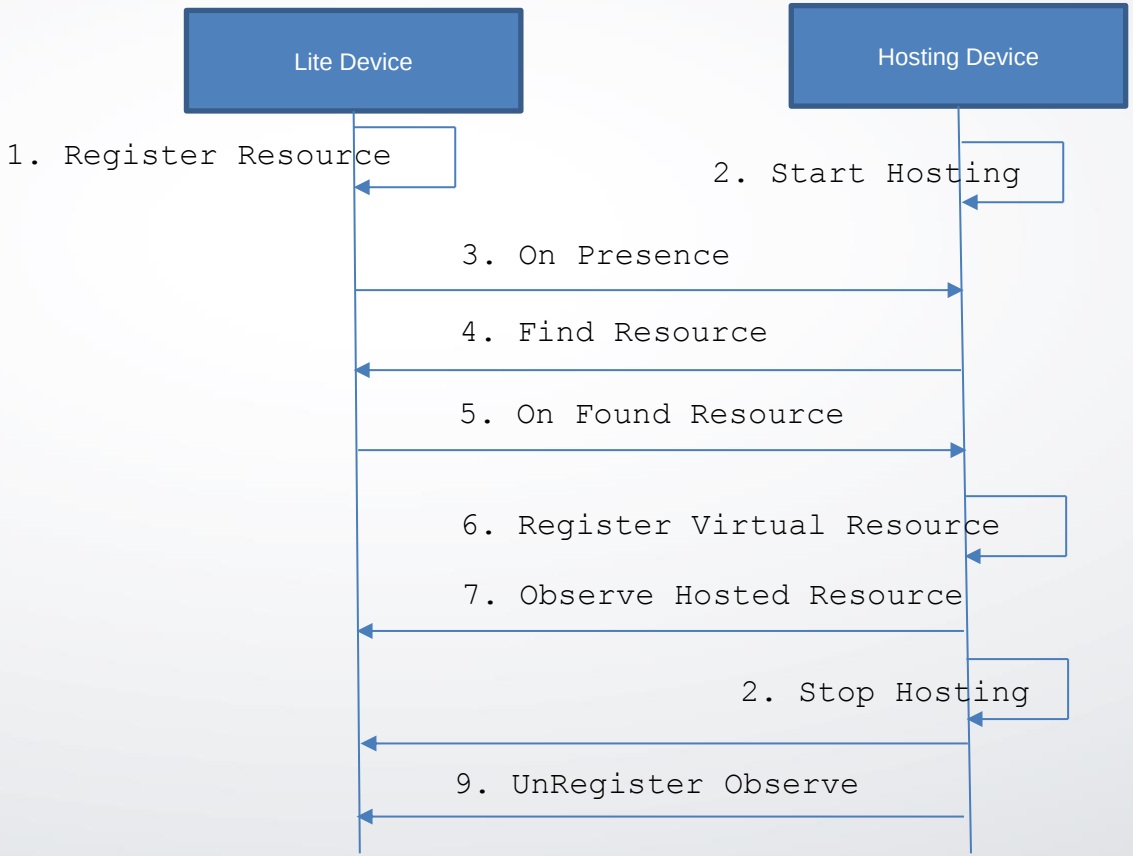
- **Things Diagnostics**
  - ✓ Factory reset to restore all configuration parameters to default one
  - ✓ Reboot to request a system rebooting

Notification Manager

- Service on Unconstrained device host resources for other Lite/Thin devices
- Hosting device mirrors resources from other Lite devices by Observing the pre sence & changes in other sources

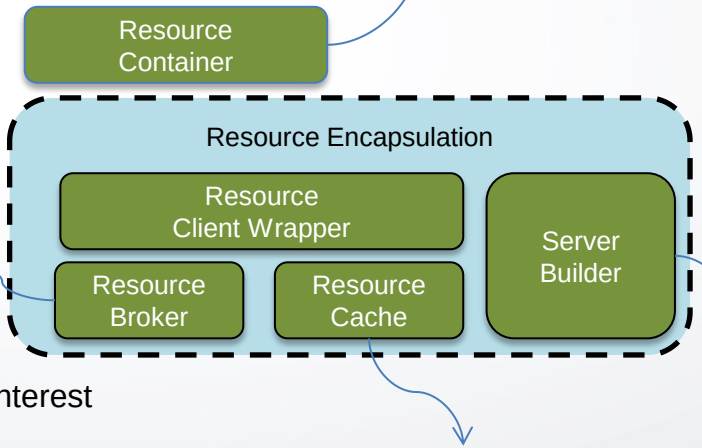


Notification Manager – System Sequence Diagram



Resource Encapsulation

- Integrates non-OIC protocols as resources
- Handles dynamic loading of resource bundles & dynamic creation of resources

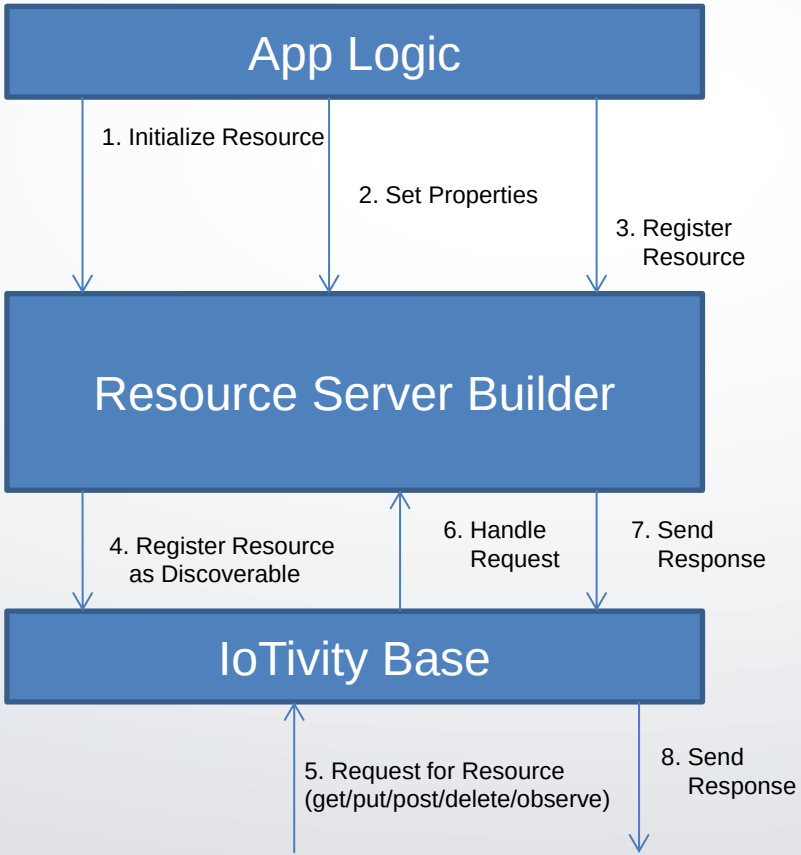


- Remote resource discovery & presence check
- Provide consistent reachability mgmt for discovered resource of interest

- Provides easier way to create resource Servers
- Monitors value of attributes & automatically transmits a message

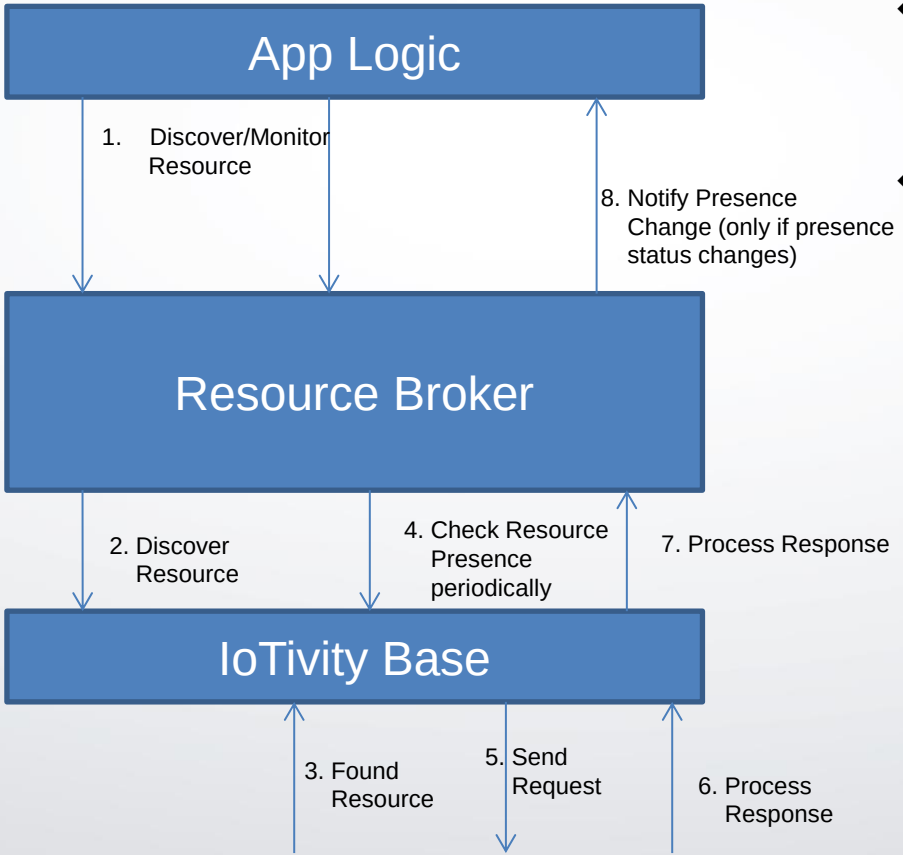
- Maintains last information of remote resource
- Data centric APIs (send/receive message → getter/setter, data cache)

Resource Server Builder



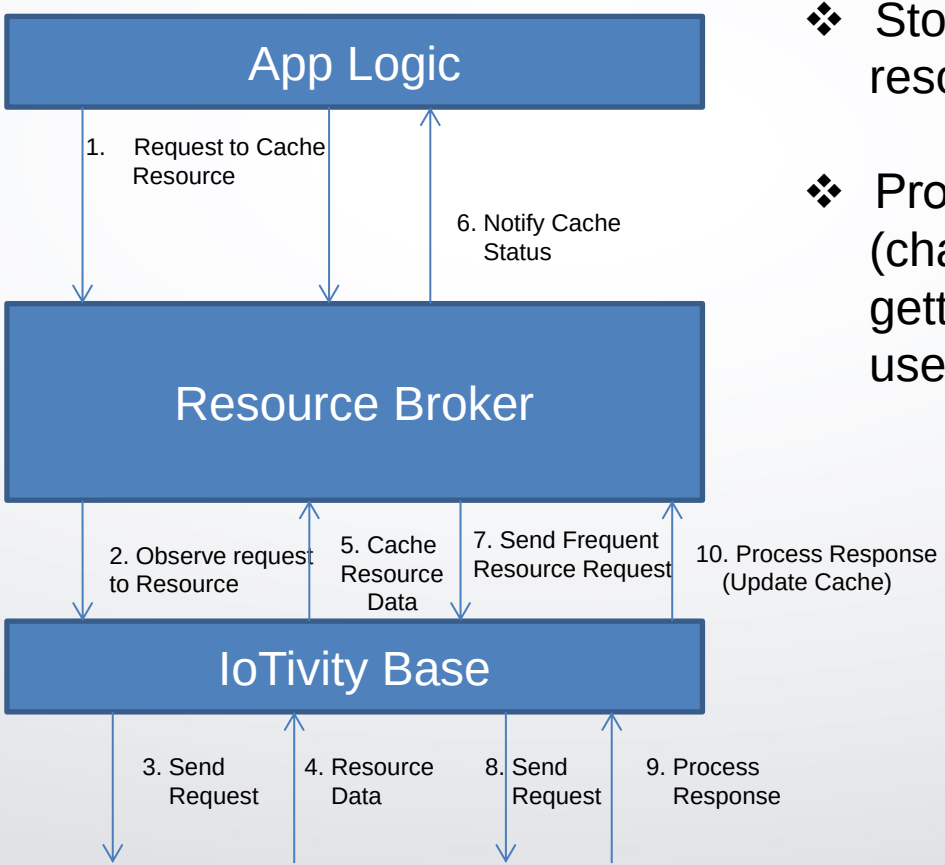
- Provide “data centric” API for users to set/create attributes of a resource server
- Notification for observers
- Developer does not need to deal with low-level details of CoAP communication
- Resources are defined based on properties & developer has to provide getter/setter methods

Resource Broker



- Provides presence check for resource of interest
- Consistent reachability management for the resources of interest(resource duplication detection and rediscover when temporally unreachable)

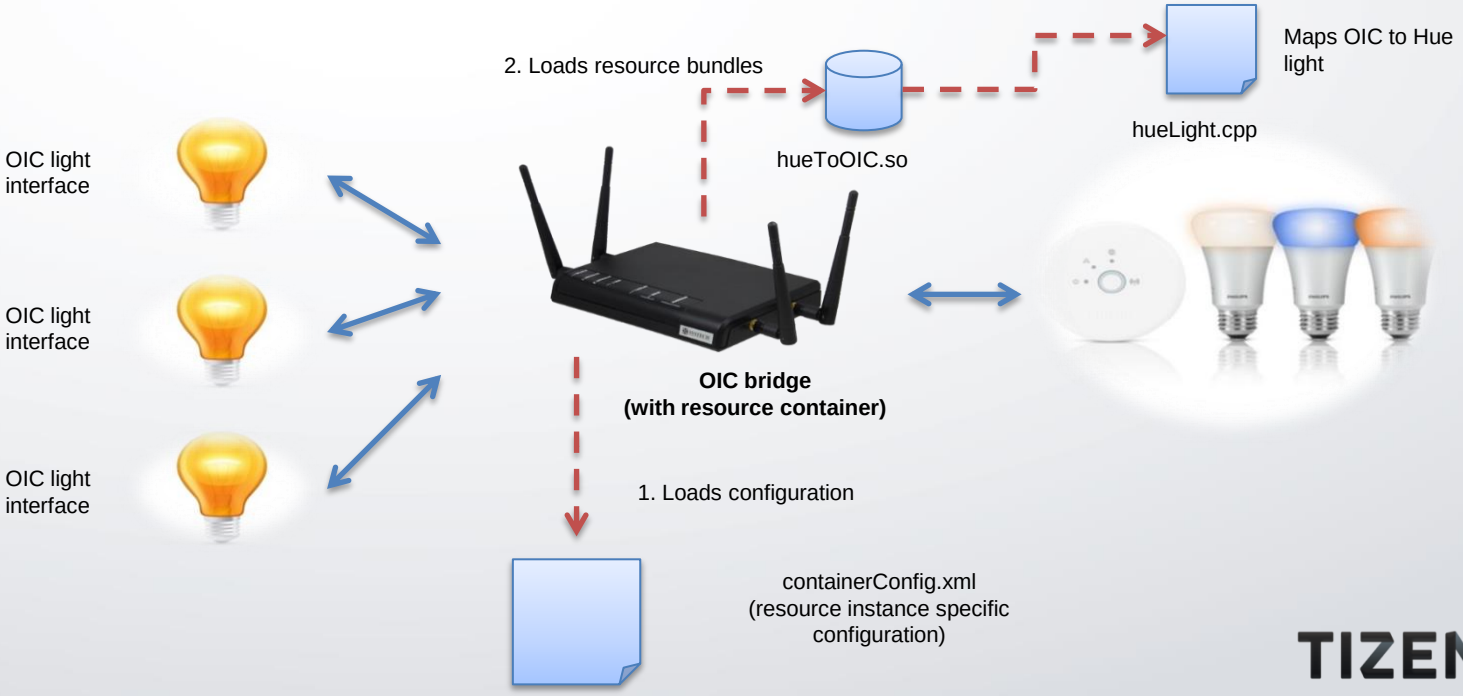
Resource Cache



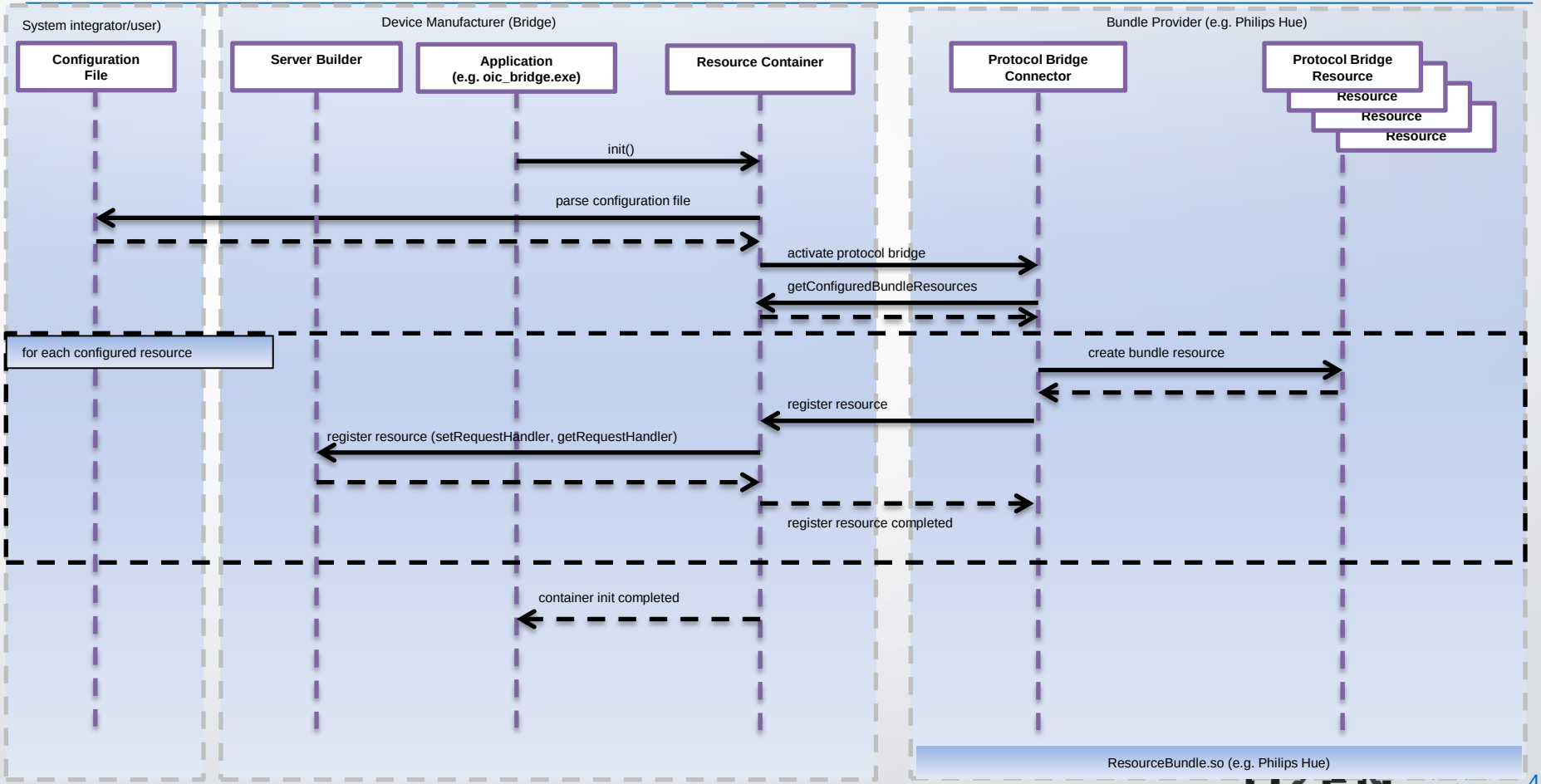
- Stores and updates latest resource data from remote resource
- Provides “data centric” interfaces (changes from messaging API to data getter-like interface) to resource client users

Resource Container

- Feature
  - Integrates non-OIC resources into the OIC ecosystem
  - Handles dynamic loading of resource bundles and dynamic creation of resources
  - Supports C++ .so files and Java .jar files
  - Common configuration for bundles and configured resources



Resource Container Interaction Flow



OIC & IoTivity – Road Ahead

IoTivity 0.9.0 (Dec 2014)

- Device & Resource Discovery IPv4
- OIC Resource Model
- ID/Addressing
- Messaging (CoAP)
- Payload encoding JSON

IoTivity 0.9.2 (August 2015)

- Initial IPv6 Support
- BLE Support
- Initial support for Remote Access
- Client-driven credential & ACL Provisioning
- Subject-based Access Control
- Resource Encapsulation
- Payload encoding CBOR

IoTivity 1.0 (October 2015)

- Multi-Phy Easy Setup
- Cloud Data Interface (CoAP over TCP)
- Resource Directory
- Simulator
- Security Updates
- Block wise Transfer
- Multi-Hop Routing Manager

OIC & IoTivity – Road Ahead

Feature	Description
Multi-Phy Easy Setup	• Connect Out-of-box device without UI onto network & provisioning
Data Interface to Cloud	• Actuation of devices from Cloud Apps, Collection of Sensor Data in Cloud
Resource Directory	• Constrained device that needs to sleep and cannot respond to multicast discovery queries • Power constrained device that cannot keep responding to multicast queries
Simulator	• Developers can test implementations without having real hardware • Manufactures can provide reference profiles using simulator • Enables users/developers to test the functionality of the device/profile first before purchasing the real device • Easy for manufacturers to test the profiles with the test suite before releasing the profiles.

OIC & IoTivity – Road Ahead

Feature	Description
Security Updates	• Filter Resource requests • Access control of resources • Secure Transmission of data across variety of IoT devices • Certificate based Key Mgmt
Blockwise Transfer	• Send/Receive of Larger data over IoTivity Stack
Multi Hop Routing Manager	• Routing of packets across variety of connectivities
AV Streaming	• Audio Video Streaming
IPv6 & 6LowPAN support	• Supporting IPv6 and 6LoWPAN as part of IoTivity Connectivity Abstraction

Summary

- IoT Landscape, Roles of OIC & IoTivity
- Understanding the big picture in IoT including various topologies
- Architectural Principles & Key Protocols adopted by OIC & IoTivity
- High Level Architecture of IoTivity Stack & types of Deployment
- Programming using IoTivity Base APIs
- Programming using IoTivity Service APIs
- Ongoing & Future work

Thank you