IoTivity Core Framework: Features & Opportunities

Kishen Maloor
Intel Open Source Technology Center
Outline

• What is IoTivity and why is it useful?
• IoTivity stack architecture
• IoTivity resource model and request-response flow
• Role in the IoT ecosystem
• Cross-platform support
What is IoTivity?

- Open source framework and SDK for building IoT applications
- Independently governed
Why is it useful?

- **Turn Lights ON**
  - Smartphone
  - Light bulbs with BLE radios

- **Regulate Temperature**
  - Smart TV
  - Tablet

- **Notify Current Setting**
  - Digital Thermostat
  - 75F
Why is it useful?

- Cross-platform support
- Uniform and easy-to-use APIs
- Based on open standards
- Support for multiple connectivity types
- Extensible to support proprietary protocols
IoTivity stack architecture

- Smart Home
- Health
- Industrial
- Automotive
- Enterprise

Services Layer

Resource Model
- JNI Glue Layer (Java API)
- Object Model (C++ API)
- Base Functionality (C API)

Secure Resource Manager

Connectivity Abstraction
OIC protocol & connectivity types

- Messaging protocol
  - Currently based on CoAP (RFC 7252)
- OIC payloads encoded using CBOR (RFC 7049)
- Adapter abstraction
  - Handle multiple connectivity types
    - Dual-stack IPv4 / IPv6
    - Bluetooth Low Energy using GATT
    - Bluetooth EDR using RFCOMM
IoTivity resource model

- RESTful design -> Things modeled as resources
- Server role: Exposes hosted resources
- Client role: Accesses resources on a server
- Intermediary role: Bridges messaging between client and server
Resource URI : /a/light1
- rt: oic.ex.light
- if: oic.if.rw
- prop: discoverable, observable
- n: myHallWayLight
  - State: 0 (OFF)
  - Dim Level: 0

Resource URI : /a/fan1
- rt: oic.ex.fan
- if: oic.if.rw
- prop: discoverable
- n: myKitchenFan
  - State: 1 (ON)
  - Speed: 10

Identifies the type of resource
List of interfaces associated with the resource
Policy associated with resource: discoverable, observable, secure, etc
Friendly name

Common Properties
- Resource URI
- rt: Resource Type
- if: Resource Interface
- prop: Policy
- n: Resource Name

Resource Specific
- Resource URI : /a/light1
- rt: oic.ex.light
- if: oic.if.rw
- prop: discoverable, observable
- n: myHallWayLight
  - State: 0 (OFF)
  - Dim Level: 0

Resource URI : /a/fan1
- rt: oic.ex.fan
- if: oic.if.rw
- prop: discoverable
- n: myKitchenFan
  - State: 1 (ON)
  - Speed: 10
### “Well-Known” resources

<table>
<thead>
<tr>
<th>Functionality</th>
<th>Fixed URI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discovery</td>
<td>/oic/res</td>
</tr>
<tr>
<td>Device</td>
<td>/oic/d</td>
</tr>
<tr>
<td>Platform</td>
<td>/oic/p</td>
</tr>
<tr>
<td>Presence</td>
<td>/oic/ad</td>
</tr>
<tr>
<td>Security</td>
<td>…</td>
</tr>
</tbody>
</table>

![Diagram showing OIC devices and fixed URIs]

- Resources are associated with “Entity Handlers”
- Execute OIC methods on resources
Resource collections

- Links to other resources (RFC 5988)
- Express hierarchy, groups, indexes

Collection URI

- rt: Resource Type
- if: Resource Interface
- prop: Resource Properties
- n: Resource Name
- links: [Other resource URI]
IoTivity request-response flow

URI: /a/light; rt = ‘oic.ex.light’, if = ‘oic.ex_rw’, prop = discoverable, observable
Resource discovery

Multicast GET coap://224.0.1.187:5683/oic/res

Unicast response

[URI: /a/light; rt = ‘oic.ex.light’, if = ‘oic.ex.rw’, prop = discoverable, observable]

Client 192.168.1.2

Server 192.168.1.1

<table>
<thead>
<tr>
<th></th>
<th>IPv4</th>
<th>IPv6</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>224.0.1.187: 5683</td>
<td>FF0X::FD: 5683</td>
</tr>
</tbody>
</table>
GET operation

Unicast GET `coap://192.168.1.1:9000/a/light`

Unicast response

[URI: /a/light; state = 0, dim = 0]
PUT operation

Client
192.168.1.2

Server
192.168.1.1

Unicast PUT coap://192.168.1.1:9000/a/light
PayLoad: [state=1;dim=50]

Unicast response

Status = Success
**OBSERVE operation**

Unicast GET `coap://192.168.1.1:9000/a/light; ObserveFlag = 1`

Unicast response

```
[URI: /a/light; state = 1, dim = 50]
```

Client

192.168.1.2

Application

Resource Model

CoAP over UDP

L2 Connectivity + IP

Server

192.168.1.1

Application

Resource Model

CoAP over UDP

L2 Connectivity + IP
OBSERVE notification

Notify Observers

[URI: /a/light; state = 0, dim = 0, sequence #: 1]

Client
192.168.1.2

Server
192.168.1.1

Application

Resource Model

CoAP over UDP

L2 Connectivity + IP
PRESENCE: “Active discovery”

- Servers can advertise themselves to clients
- Clients can request unicast or multicast notifications
  - Server coming online
  - Server going off-line
  - Changes to resources
- Clients may indicate interest in specific resource types
Notable IoTivity features

- Discovery
- Messaging and data model
- Message switching
- Remote access
- Services
  - Protocol plug-ins
  - Group management
- Security
Role in the IoT ecosystem

- **IoTivity Application**
  - **Vertical Profiles**
    - Healthcare
    - Industrial
    - Smart Home
    - ...
  - **IoTivity Base & Services**
  - **Connectivity Abstraction**
- Connectivity
  - Things
  - Controller
  - Cloud Servers
  - LE
- **Local Control**
- **Remote Control**
  - service #1 domain
- **Server to Server**
  - service #2 domain
Cross-platform support

- Linux (Ubuntu 12.04)
- Arduino: Due, ATMega 2560
- Android
- Tizen
Embedded support: Yocto Project

  - Hosted at the Linux Foundation
  - Create customized OS images for embedded targets
  - Ready-to-use BSPs for multiple platforms
  - Supports major CPU architectures
- Layers and recipes
meta-oic software layer for Yocto

- git://git.yoctoproject.org/meta-oic

**Samples**
- Resource and Service layer samples

**IoTivity**
- APIs
- Service layer
- Resource Model
- Base Framework

**Dependencies**
- Kernel Configuration
- Protocols
- Middleware
Constrained peripherals

- Storage and memory constraints
- Lightweight IoTivity server stack
  - Base framework, resource model, messaging
- Work in progress…
How can you participate?

- Adopt IoTivity as the framework of choice for IoT projects
- Contribute to the IoTivity project: www.iotivity.org
- IoTivity mailing list: iotivity-dev@lists.iotivity.org
Thanks for your time!

Q&A