IP Based Architecture for the Internet of Things

IPV6 and Related Standards for IoT Interoperability

November 20, 2014
IoT Architecture – Problems to Solve

- **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

- **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
These Problems are Already Solved

- 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

- We want the same qualities for IoT
- What are the important design patterns in Internet and WWW Architecture?
- How can these design patterns apply to IoT?
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

- 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
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
IP on Constrained Devices

- CoAP
- HTTP
- 6LowPAN
- IPV4/IPV6
- 802.15.4 WiFi, Ethernet
- MCU – 16KiB RAM
- MPU

- Application Protocol
- Routing
- HW Network
- Hardware
Machine to Machine Application Protocols

- CoAP and Related IETF Standards
  - Machine to Machine (M2M) protocol modeled after HTTP
  - Compressed Binary mapping of REST API protocol
  - Asynchronous Notifications to support M2M use cases
  - Format for Machine Hyperlinks, CoRE Link-Format

- HTTP
  - Useful for less resource constrained environments
  - Works with existing libraries and servers
  - Well known extensions for asynchronous notification
Object Models and Data Models

- IPSO Smart Objects
  - Object/Resource URI template for M2M REST API
  - Defines Structure and Data Types for functionally specialized objects
  - E.g. Temperature Sensor, Light Controller, Load Controller
  - Compatible with CoAP, HTTP, and other underlying protocols

- Others being considered by various IoT Interest Groups (IOTWF, IIC, OIC)
- W3C Community group on Web of Things considering work on data models
Hypermedia for Machine APIs

- **CoRE Link-Format**
  - Format for Semantic Hyperlinks that describe resources
  - Mechanism for embedding links within data sources

- **CoRE Resource Directory**
  - Centralized directory to store Semantic Hyperlinks registered by devices
  - Mechanism for applications to discover resources by link attribute

- **IPSO Smart Objects**
  - Self-contained description in URI schema

- **Hypercat**
  - Catalog format for general purpose Semantic Hyperlinks
IPSO Smart Objects

- A simple Data Model for
- Semantic Interoperability across IoT Devices
- Requires only simple URI addressing and a few data types
- Mapping to internet content-types
- Usable on many different transport protocols (CoAP, HTTP, MQTT)

This presentation discusses the architecture and features of a feature rich implementation of IPSO Smart Objects on constrained devices and networks using CoAP and OMA LWM2M
IPSO Smart Object Architecture Use Cases

- Web Application
  - Web Applications running on Servers, Browsers, Smartphones, Tablets

- Service e.g. LWM2M
  - Smart Object Registration, Discovery and Data Layer
    - Service, Device Proxy and Cache

- Managed Device, e.g. 16KB RAM, 128KB Flash

- Client

- Server

- Peer-Peer

- Sensor/Actuator Device

- Devices with Embedded Applications

  - Applications can Discover and Interact with devices using Peer-Peer networking or through Services, using the Same Semantics
# IPSO Smart Objects And Related IoT Standards

<table>
<thead>
<tr>
<th>Application Software</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPSO Objects</td>
<td>Data Models</td>
</tr>
<tr>
<td>OMA LWM2M</td>
<td>API and Services</td>
</tr>
<tr>
<td>CoAP</td>
<td>Application Protocol</td>
</tr>
<tr>
<td>6LowPAN</td>
<td>Routing</td>
</tr>
<tr>
<td>IPV4/IPV6</td>
<td>HW Network</td>
</tr>
<tr>
<td>WiFi, Ethernet</td>
<td></td>
</tr>
<tr>
<td>MCU – 16KiB RAM</td>
<td>Hardware</td>
</tr>
<tr>
<td>MPU</td>
<td></td>
</tr>
</tbody>
</table>
IoT Standards That Build On Each Other

- CoAP and related standards from IETF
  - REST API for constrained networks and devices
  - HTTP Proxy provides abstraction through standard web APIs
  - Core-link-format (RFC 6690) provides semantic descriptors in the form of web links
  - Resource Directory provides an API for scalable discovery and linking using core link-format mediatype

- OMA LWM2M is based on CoAP
  - Provides a server profile for IoT middleware
  - Defines a simple reusable object model
  - Defines management objects and reuses REST API for onboarding and device life cycle management

- IPSO Smart Objects are based on OMA LWM2M
  - Defines application objects using the LWM2M Object Model
  - Complex objects can be composed from simple objects
  - Easy to add new resource and object types as needed
CoAP
CoAP Protocol

- Makes each device a lightweight server that exposes a REST API
- A CoAP endpoint can be both client and server
- Roles can be reversed and the sensor, as a client, can also interact with a REST API at another endpoint or server node
- Peer to Peer interaction is based on a duplex client-server pattern
CoAP Use Case Requirements

REST API

Less Constrained

Resource Constrained

REQ2: Constrained networks
REQ5: Resource manipulation
REQ8: Resource discovery
REQ3: Sleeping nodes
REQ10: UDP Transport
REQ6: Sub/Notify
REQ11: Reliability
REQ12: Low latency
REQ1: Limited Flash/RAM
REQ9: Multicast
CoAP Scope

• Transport
  – CoAP currently defines:
  – UDP binding with DTLS security
  – CoAP over SMS or TCP possible

• Base Messaging
  – Simple message exchange between endpoints
  – Confirmable or Non-Confirmable Message
  – Answered by Acknowledgement or Reset Message

• REST Semantics
  – REST Request/Response mapped onto CoAP Messages
  – Method, Response Code and Options (URI, content-type etc.) define
    REST exchanges, very similar to HTTP (HTTP 404 response
    semantics (not found) mapped to CoAP 4.04 response code)

• Asynchronous Notifications
  – Observer option for GET allows asynchronous state update
    responses from a single request – actually a separate IETF Draft
CoAP Maps HTTP-like protocol to a binary format

Ver - Version (1)
T - Message Type (Confirmable, Non-Confirmable, Acknowledgement, Reset)
TKL - Token Length, if any, the number of Token bytes after this header
Code - Request Method (1-10) or Response Code (40-255)
Message ID - 16-bit identifier for matching responses
Token - Optional response matching token
CoAP Example: GET Transaction

CoAP Client

CON [0xaf5] GET /light

ACK [0xaf5] 2.05 Content "<light>..."

CoAP Server

Confirmable Request

Response with Data Payload
CoAP Proxy Caching
CoAP Asynchronous Notification

- **CON GET /light** Observe: 0 Token: 0x3f
- **ACK 2.05** Observe: 27 Token: 0x3f "<light>..."
- **CON 2.05** Observe: 28 Token: 0x3f "<light>..."
- **ACK Token: 0x3f**
- **CON 2.05** Observe: 29 Token: 0x3f "<light>..."
- **ACK Token: 0x3f**

Additional Acknowledgements (...)

/light changes
Publish-Subscribe Over CoAP (IETF Draft)

EP1

PUT /pubsub/A
2.04 Changed

Publish

PUT /pubsub/A
2.04 Changed

SUBSCRIBE
GET obs:0 /pubsub/A

EP2

core.publisher service

SUBSCRIBE
GET obs:0 /pubsub/A

EP3

2.05 Content obs: 1

Publish

2.05 Content obs: 1

EP4

2.05 Content obs: 2

Publish

2.05 Content obs: 2

Publish

2.05 Content obs: 2
CoRE Link-Format (RFC 6690) - Web Linking for Machines

- RFC6690 is aimed at Resource Discovery and Linking for M2M
  - Defines semantic link serialization and content-types suitable for M2M
  - Defines a well-known resource where links are stored
  - Enables query string parameters for discovery by attribute and relation
  - Can be used with unicast or multicast (CoAP)

- Resource Discovery with RFC6690
  - Discovering the links hosted by CoAP (or HTTP) servers
  - GET /.well-known/core?optional_query_string
  - Returns a link-format document
  - URL, resource type, interface type, content-type, size are some basic relations
RFC 6690 CoRE Link-Format Example

<4001/0/9002>;rt="oma.lwm2m";ct=50;obs=1

- Resource Type
- Content Type
- Observable
Local Network Discovery

Could use multicast

Could return a link-format document:

```xml
</3//9>;obs;rt="urn:X-ipso:batt-level";ct="50",
</3//0>;rt="urn:X-ipso:dev-mdl";ct="50",
</3//1>;rt="urn:X-ipso:dev-mfg";ct="50",
</3305/0/5800>;obs;rt="urn:X-ipso:pwr-w";ct="50",
</3305/0/5805>;obs;rt="urn:X-ipso:pwr-accum-wh";ct="50",
</3303/0/5700>;obs;rt="urn:X-ipso:temp-C";ct="50"
```
Resource Discovery

- RFC 6690 CoRE Link Format defines
  - The link format media type
  - Peer-to-peer discovery
- A directory approach is also useful
  - Supports sleeping nodes
  - No multicast traffic, longer battery life
  - Remote lookup, hierarchical and federated distribution
- CoRE Link Format is used in Resource Directories
  - Nodes register their resource links to an RD
  - Nodes refresh the RD periodically
  - Nodes may unregister (remove) their RD entry

See draft-ietf-core-resource-directory
Resource Discovery Example Flow

REGISTRATION
POST /rd?ep="235598376"&lt=19999
</3303/0/5700>;rt="urn:X-ipso:temp-C"

2.01 Created Location:/rd/235598376

DISCOVERY
GET /rd-lookup?ep&rt="urn:X-ipso:temp-C"

2.05 Content
</235598376/3303/0/5700>;rt="urn:X-ipso:temp-C"

See draft-ietf-core-resource-directory
OMA LWM2M
OMA LWM2M Reference Architecture

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

- LWM2M Server
  - CoAP Protocol
  - Supports HTTP Caching Proxy
  - Resource Directory
  - Gateway and Cloud deployable

- LWM2M Clients are Devices
  - Device abstraction through CoAP
  - LWM2M Clients are CoAP Servers
  - Any IP network connection
LWM2M Interfaces

- **Bootstrap Interface**
  - Configure Servers & Keying
  - Pre-Configured, Smart Card, or Server Initiated Bootstrap
  - CoAP REST API

- **Registration Interface**
  - RFC6690 and Resource Directory

- **Management Interface Using Objects**
  - Management Objects and Resources
  - CoAP REST API

- **Reporting Interface**
  - Object Instances and Resources Report
  - Asynchronous notification using CoAP Observe
LWM2M Object Model

- A Client has one or more Object Instances
- An Object is a collection of Resources
- A Resource is an atomic piece of information that can be
  - Read, Written or Executed
- Objects can have multiple instances
- Objects and Resources are identified by a 16-bit Integer, Instances by an 8-bit Integer
- Objects/Resources are accessed with simple URIs: //{Object ID}/{Object Instance}/{Resource ID}

Example:
/3/0/1 - Object Type=3 (Device), Instance=0, Resource Type = 1 (Device Mfg.)
# LWM2M Management Objects

<table>
<thead>
<tr>
<th>Object</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>LWM2M Security</td>
<td>0</td>
</tr>
<tr>
<td>LWM2M Server</td>
<td>1</td>
</tr>
<tr>
<td>Access Control</td>
<td>2</td>
</tr>
<tr>
<td>Device</td>
<td>3</td>
</tr>
<tr>
<td>Connectivity Monitoring</td>
<td>4</td>
</tr>
<tr>
<td>Firmware</td>
<td>5</td>
</tr>
<tr>
<td>Location</td>
<td>6</td>
</tr>
<tr>
<td>Connectivity Statistics</td>
<td>7</td>
</tr>
</tbody>
</table>
# LWM2M Position Object Example, OMA Template

<table>
<thead>
<tr>
<th>Resource Name</th>
<th>ID</th>
<th>Access Type</th>
<th>Multiple Instances?</th>
<th>Type</th>
<th>Range</th>
<th>Units</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Latitude</td>
<td>0</td>
<td>R</td>
<td>No</td>
<td>Decimal</td>
<td>Deg</td>
<td></td>
<td>The decimal notation of latitude, e.g. -43.5723 [World Geodetic System 1984]</td>
</tr>
<tr>
<td>Longitude</td>
<td>1</td>
<td>R</td>
<td>No</td>
<td>Decimal</td>
<td>Deg</td>
<td></td>
<td>The decimal notation of longitude, e.g. 153.21760 [World Geodetic System 1984]</td>
</tr>
<tr>
<td>Altitude</td>
<td>2</td>
<td>R</td>
<td>No</td>
<td>Decimal</td>
<td>m</td>
<td></td>
<td>The decimal notation of altitude in meters above sea level.</td>
</tr>
<tr>
<td>Uncertainty</td>
<td>3</td>
<td>R</td>
<td>No</td>
<td>Decimal</td>
<td>m</td>
<td></td>
<td>The accuracy of the position in meters.</td>
</tr>
<tr>
<td>Velocity</td>
<td>4</td>
<td>R</td>
<td>No</td>
<td>Refers to 3GPP GAD specs</td>
<td>Refers to 3GPP GAD specs</td>
<td></td>
<td>The velocity of the device as defined in 3GPP 23,032 GAD specification. This set of values may not be available if the device is static.</td>
</tr>
<tr>
<td>Timestamp</td>
<td>5</td>
<td>R</td>
<td>No</td>
<td>Time</td>
<td></td>
<td></td>
<td>The timestamp of when the location measurement was performed.</td>
</tr>
</tbody>
</table>
LWM2M Application Server

Web App

Soft Endpoints

/domain/endpoints/3303/0/5700

LWM2M Clients

_REGISTER_ DISCOVER

IP Device IP Device

LWM2M Server

/3303/0/5700
LWM2M Application Server

Web App

LWM2M Server

LWM2M Clients

1. NOTIFY

2. NOTIFY
LWM2M Supports Sleeping Endpoints “b=uq”

- Client uses the registration refresh to inform LWM2M server that it is awake, and listens for any queued operations
LWM2M Communication Through NAT Router

Web App

http/REST

Mbed Device Server

CoAP

NAT Router

REG UPDATE

GET/PUT

HW Device Endpoints

IP Device

IP Device
LWM2M Observe Parameters

- LWM2M provides a mechanism to control Observation
- “Write Attributes” Interface using query parameters to set observe attributes:
  - Pmin – minimum observation quiet period, to limit notification frequency
  - Pmax – maximum observation quiet period, to guarantee notifications
  - Lt – low limit measurement notification, like low alarm, engineering units
  - Gt – high limit measurement notification, like a high alarm, engineering units
  - Step – Minimum delta change required to notify, in engineering units
LWM2M Bulk Read

- Returns TLV or JSON based on requested content-format
- CBOR needs to be added
- Linked Objects are supported

```json
{"e": [
  {"n": "0", "sv": "Open Mobile Alliance"},
  {"n": "1", "sv": "Lightweight M2M Client"},
  {"n": "2", "sv": "345000123"},
  {"n": "3", "sv": "1.0"},
  {"n": "6/0", "v": "1"},
  {"n": "6/1", "v": "5"},
  {"n": "7/0", "v": "3800"},
  {"n": "7/1", "v": "5000"},
  {"n": "8/0", "v": "125"},
  {"n": "8/1", "v": "900"},
  {"n": "9", "v": "100"},
  {"n": "10", "v": "15"},
  {"n": "11/0", "v": "0"},
  {"n": "13", "v": "1367491215"},
  {"n": "14", "sv": "+02:00"},
  {"n": "15", "sv": "U"}
]}
```
LWM2M Discovery Returns RFC 6690 Links

- Links are uploaded during registration to inform the LWM2M server about resources on the endpoint.
- Links are discovered using GET with content type “application/link-format”.
- JSON representation using content type “application/link-format+json”.

\[<4001/0/9002>;rt="oma.lwm2m";ct=50;obs=1\]
IPSO Smart Objects
Smart Objects Use the LWM2M Object Model

- REST API with a URI template
  - Objects
  - Object Instances
  - Resources
  - (Resource Instances)

- Reusable resource and object IDs
  - Common definitions for concepts
  - Map to semantic terms e.g. temperature, currentValue
  - IDs are registered with the OMNA

- Can be embedded in a path hierarchy on the server
  - /home/weather/3303/0/5700
IPSO Smart Object Example

Object info

<table>
<thead>
<tr>
<th>Object</th>
<th>Object ID</th>
<th>Object URN</th>
<th>Multiple Instances?</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPSO Temperature</td>
<td>3303</td>
<td>urn:oma:lwm2m:ext:3303</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Resource Info

<table>
<thead>
<tr>
<th>Resource Name</th>
<th>Resource ID</th>
<th>Access Type</th>
<th>Multiple Instances?</th>
<th>Type</th>
<th>Units</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor Value</td>
<td>5700</td>
<td>R</td>
<td>No</td>
<td>Decimal</td>
<td>Cel</td>
<td>This resource type returns the Temperature Value in °C</td>
</tr>
<tr>
<td>Min Measured Value</td>
<td>5601</td>
<td>R</td>
<td>No</td>
<td>Decimal</td>
<td>Cel</td>
<td>The minimum value measured by the sensor since it is ON</td>
</tr>
<tr>
<td>Max Measured Value</td>
<td>5602</td>
<td>R</td>
<td>No</td>
<td>Decimal</td>
<td>Cel</td>
<td>The maximum value measured by the sensor since it is ON</td>
</tr>
</tbody>
</table>

Accessing the Resources

- Temperature Value  /3303/0/5700
- Min Measured Value /3303/0/5601
- Max Measured Value /3303/0/5602

LWM2M Client

Object with Internal Resources
### Table 1  Smart Objects defined by this Technical Guideline

<table>
<thead>
<tr>
<th>Object</th>
<th>Object ID</th>
<th>Multiple Instances?</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPSO Digital Input</td>
<td>3200</td>
<td>Yes</td>
</tr>
<tr>
<td>IPSO Digital Output</td>
<td>3201</td>
<td>Yes</td>
</tr>
<tr>
<td>IPSO Analogue Input</td>
<td>3202</td>
<td>Yes</td>
</tr>
<tr>
<td>IPSO Analogue Output</td>
<td>3203</td>
<td>Yes</td>
</tr>
<tr>
<td>IPSO Generic Sensor</td>
<td>3300</td>
<td>Yes</td>
</tr>
<tr>
<td>IPSO Illuminance Sensor</td>
<td>3301</td>
<td>Yes</td>
</tr>
<tr>
<td>IPSO Presence Sensor</td>
<td>3302</td>
<td>Yes</td>
</tr>
<tr>
<td>IPSO Temperature Sensor</td>
<td>3303</td>
<td>Yes</td>
</tr>
<tr>
<td>IPSO Humidity Sensor</td>
<td>3304</td>
<td>Yes</td>
</tr>
<tr>
<td>IPSO Power Measurement</td>
<td>3305</td>
<td>Yes</td>
</tr>
<tr>
<td>IPSO Actuation</td>
<td>3306</td>
<td>Yes</td>
</tr>
<tr>
<td>IPSO Set Point</td>
<td>3308</td>
<td>Yes</td>
</tr>
<tr>
<td>IPSO Load Control</td>
<td>3310</td>
<td>Yes</td>
</tr>
<tr>
<td>IPSO Light Control</td>
<td>3311</td>
<td>Yes</td>
</tr>
<tr>
<td>IPSO Power Control</td>
<td>3312</td>
<td>Yes</td>
</tr>
<tr>
<td>IPSO Accelerometer</td>
<td>3313</td>
<td>Yes</td>
</tr>
<tr>
<td>IPSO Magnetometer</td>
<td>3314</td>
<td>Yes</td>
</tr>
<tr>
<td>IPSO Barometer</td>
<td>3315</td>
<td>Yes</td>
</tr>
</tbody>
</table>
# Ad-Hoc IPSO Smart Object – Smart Thermostat

## Object info:

<table>
<thead>
<tr>
<th>Object</th>
<th>Object ID</th>
<th>Object URN</th>
<th>Multiple Instances?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smart Thermostat</td>
<td>12300</td>
<td>urn:oma:hwcm:x:12300</td>
<td>Yes</td>
<td>Smart Thermostat with multiple settings</td>
</tr>
</tbody>
</table>

## Resource Info:

<table>
<thead>
<tr>
<th>Resource Name</th>
<th>Resource ID</th>
<th>Access Type</th>
<th>Multiple Instances?</th>
<th>Mandatory</th>
<th>Type</th>
<th>Range or Enumeration</th>
<th>Units Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor Value</td>
<td>5700</td>
<td>R</td>
<td>No</td>
<td>Mandatory</td>
<td>Float</td>
<td>Per Units resource</td>
<td>Temperature measurement</td>
<td></td>
</tr>
<tr>
<td>Units</td>
<td>5500</td>
<td>R,W</td>
<td>No</td>
<td>Mandatory</td>
<td>String</td>
<td>ucum:degF, ucum:degC</td>
<td>Units for 5700</td>
<td></td>
</tr>
<tr>
<td>Application Type</td>
<td>5750</td>
<td>R,W</td>
<td>No</td>
<td>Optional</td>
<td>String</td>
<td>Name, e.g. “Hall Thermostat”</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cooling</td>
<td>5200</td>
<td>R</td>
<td>No</td>
<td>Optional</td>
<td>Boolean</td>
<td>1=cooling</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Heating</td>
<td>5201</td>
<td>R</td>
<td>No</td>
<td>Optional</td>
<td>Boolean</td>
<td>1=heating</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Heat Source</td>
<td>5203</td>
<td>R</td>
<td>No</td>
<td>Optional</td>
<td>String</td>
<td>“Emergency”, “Normal”</td>
<td>Indicates heat source</td>
<td></td>
</tr>
</tbody>
</table>

| Fan Timer Active | 5204 | R,W | No | Optional | Boolean | 1=running |
| Fan Timeout      | 5205 | R,W | No | Optional | String  | UTS       |
| Energy Save Mode | 5206 | R,W | No | Optional | Boolean | 1= Energy Save mode |
| Away Mode        | 5207 | R,W | No | Optional | Boolean | 0=Home, 1=Away |
| Setpoint         | 5208 | R   | No | Optional | Float   | Desired Temperature |
| High Setpoint    | 5210 | R,W | No | Optional | Float   | Highest desired temperature |
| Low Setpoint     | 5211 | R,W | No | Optional | Float   | Lowest desired temperature |
| High Away Setpoint | 5212 | R,W | No | Optional | Float   | Highest away mode temperature |
| Low Away Setpoint | 5213 | R,W | No | Optional | Float   | Lowest away mode temperature |
Composite IPSO Smart Objects

Energy Meter Module
- Actuation
  - On/Off
  - MultiState Output
- French TIC Info
  - Last TIC Sample
  - TIC Meter Type
- Digital Input
  - Digital Input State
  - Digital Input Counter
  - Digital Input Counter Reset
  - Application Type
  - Sensor Type

Fridge
- Actuation
  - On/Off
- SetPoint
  - SetPoint Value
  - SetPoint Unit

HVAC Controller
- Actuation
  - On/Off
  - MultiState Output
- Temperature Sensor
  - Sensor Value

Physical Object | OMA Item Object | Resource
IPSO Smart Object Development

- Smart Objects are Easy to Modify and Customize
  - Based on Consistent Design Patterns and Reusable Resource Definitions
  - Object Sets can be Forked and Modified
  - Expecting Domain-Specific Object Sets to be Created by Collaborative Vertical Working Groups
  - New Object Sets can be Released as new Smart Object Guidelines
  - Objects in Released Smart Object Guidelines are Registered with the OMA, Use Standard OMA DDF (XML) File Format Object Descriptors
IPSO Smart Objects Roadmap

- More Objects, Domain Specific
  - Gateway Management Objects – TR-069
  - Media Objects
  - HVAC
  - Smart Appliances
  - Mapping and Binding of Smart Objects to Zigbee Application Clusters
  - Mapping and Binding of Smart Objects to Bluetooth Application Profiles
  - Advanced Lighting Objects
  - Behavioral Objects – Timers, Controllers, Rules and Bindings

- Object Model
  - Linked Composite Objects
  - Semantic Annotation and Application Templates
# Ad-Hoc IPSO Smart Object – BLE Heart Rate Sensor Profile

## Object info:

<table>
<thead>
<tr>
<th>Object</th>
<th>Object ID</th>
<th>Object URN</th>
<th>Multiple Instances?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heart Rate</td>
<td>12200</td>
<td>urn:oma:lwm2m:x:12200</td>
<td>Yes</td>
<td>Heart Rate Monitor</td>
</tr>
</tbody>
</table>

## Resource Info:

<table>
<thead>
<tr>
<th>Resource Name</th>
<th>Resource ID</th>
<th>Access Type</th>
<th>Multiple Instances?</th>
<th>Mandatory</th>
<th>Type</th>
<th>Range or Enumeration</th>
<th>Units</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor Value</td>
<td>5700</td>
<td>R</td>
<td>No</td>
<td>Mandatory</td>
<td>Float</td>
<td></td>
<td>BPM</td>
<td>Heart Rate Measurement Value</td>
</tr>
<tr>
<td>Digital Input State</td>
<td>5500</td>
<td>R</td>
<td>No</td>
<td>Optional</td>
<td>Boolean</td>
<td></td>
<td></td>
<td>Sensor contact status 0=no contact, 1= contact</td>
</tr>
<tr>
<td>Total Energy</td>
<td>5950</td>
<td>R</td>
<td>No</td>
<td>Optional</td>
<td>Float</td>
<td></td>
<td>kJ</td>
<td>Energy Expended</td>
</tr>
<tr>
<td>Reset Cumulative Energy</td>
<td>5822</td>
<td>E</td>
<td>No</td>
<td>Optional</td>
<td>Opaque</td>
<td></td>
<td></td>
<td>Reset 5950 Energy Expended to zero</td>
</tr>
<tr>
<td>Body Sensor Location</td>
<td>5951</td>
<td>R,W</td>
<td>No</td>
<td>Optional</td>
<td>String</td>
<td></td>
<td></td>
<td>Intended sensing location on the body</td>
</tr>
<tr>
<td>R-R Interval</td>
<td>5952</td>
<td>R</td>
<td>No</td>
<td>Optional</td>
<td>String</td>
<td></td>
<td></td>
<td>Sequence of R-wave intervals</td>
</tr>
</tbody>
</table>
Semantic Annotation of Smart Objects

- Object annotation could use RFC 6690 for associating additional semantic descriptions with Smart Objects and Resources
- Can be used to add contextual metadata and dynamic link relations
- Described in IETF CoRE Interfaces document (http://datatracker.ietf.org/doc/draft-ietf-core-interfaces), enables Discovery by relation and attribute
  - For example, GET /rd-lookup?ep&rt="urn:X-ipso:temperature"
  Returns: </sensors/3303/0/5700>;obs;if="urn:X-ipso:sensor";rt="urn:X-ipso:temperature";ct=50;u="ucum:degC"

- Refers to qualified, resolvable namespaces and concepts
- Local discovery in .well-known/core or using Resource Directories
- GET/PUT can use semantic query GET <URL>?rt="urn:X-ipso:temperature"
Application Templates

- JSON templates for instance constructor and application schema
- Interface to high level semantic models – not part of the wire protocol
- Example template fragment for OMA LWM2M Application
- Can carry Semantic Annotation as link attributes

```
"objects":{
  3303:{
    "description": "ipso temperature sensor",
    "attributes": {"pmin":60, "pmax":300, "max-age":360},
    "link-attributes": {"rt": ["oma.lwm2m", "urn:X-ipso:temperature"]},
    "instances": {
      0: {
        "attributes": {},
        "link-attributes": {"rt": "urn:oma:lwm2m:ext:3303"},
        "resources": {
          5700: {
            "description": "Current Measured Value",
            "attributes": {"pmin":10, "step":0.5},
            "link-attributes": {"rt": "ucum:temperature", "obs", "ct":50}
          },
          5701: {
            "description": "units",
            "value": "ucum:Cel",
            "operations": ["r"]
          },
          5601: {"description": "Min Measured Value", "value":100},
          5602: {"description": "Max Measured Value", "value":0},
          5603: {"description": "Min Range Value", "value":0},
          5604: {"description": "Max Range Value", "value":100},
          5605: {"description": "Reset Min/Max"}
        }
      }
    }
  }
```
Summary

- **Application Software**
  - Not tied to specific device or protocol
  - Any Programming Language
  - Runs on devices, gateways, and services

- **IPSO Smart Objects**
  - Application Level Interoperability
  - Reusable Device to Application API
  - Not tied to any specific protocol

- **OMA LWM2M**
  - Service Layer Specification
  - Device Management over CoAP
  - Object Model for DM and Applications

- **CoAP**
  - REST protocol for constrained devices
  - IETF Standard (RFC 7252)
  - Uses TCP or UDP, any IP connection
  - Discovery using IP Multicast or Directory

---

53
References

IPSO Smart Object Guideline
http://www.ipso-alliance.org/smart-object-guidelines

OMA LWM2M Specification
http://openmobilealliance.hs-sites.com/lightweight-m2m-specification-from-oma

IETF CoAP and Related Specifications
CoAP (RFC 7252):
CoRE Link-Format (RFC 6690):
CoRE Resource Directory:

CoAP Community Site
http://coap.technology/