Device Management for OSGi IoT Gateways

Luca Dazi @ Eurotech
Julien Vermillard @ Sierra Wireless
Agenda

- Introduction to IoT Gateways and Kura
- Device Management for OSGi
- Which Protocols?
  - MQTT
  - LwM2M/CoAP
- Demo
- Next Steps
IoT Gateways
Revolution: Towards Real-time Actionable Data
Eclipse Open IoT Stack for Java
Eclipse Kura

OSGi Application Container (Eclipse Equinox, Concierge)

Java SE 7 / 8 (OpenJDK)
Device Management

- Manage the OSGi application software
  - Remote OSGi Bundle Updates
  - Remote OSGi Service Configuration
  - Remote OSGi Control

- Management of the Device
  - Firmware Updates
  - Monitor and Diagnostics
  - Remote Access
Device Management

Which protocol?

- Usual suspects
  - TR-069
  - OMA-DM
  - LwM2M

- Can we have a Device Management over the telemetry protocol?
  - Single protocol
  - Single security
  - Simplified deployment
  - Simplified management

- Kura Case Study
  - MQTT
  - CoAP/LwM2M
M2M Integration Technologies
Message Queue Telemetry Transport (MQTT)

- M2M Messaging Protocol
- Low Bandwidth / Low Power
- 2-way Communication
- Publish and Subscribe
- Hierarchical Topic Namespaces
- Data Payload Agnostic
- Device Initiated Connection
- Firewall-friendly
- SSL and Authenticated
- Large ecosystem
Kura MQTT Topic Namespace

Introduction

- **Data Topics**
  - Used for push of sensor data, metrics, or other data
  - Example: [account_id]/[client_id]/...

- **Control Topics**
  - Used for dynamic request/response interactions between clients
  - Example: CTL/[account_id]/[client_id]/[app_id]/...

- **Separate data from control topics**
  - Provides increased granularity on ACLs
  - Allows for segregation of persistent and transient data
Kura MQTT Resource Management

REST like

- **Request**
  - CTL/[account_id]/[client_id]/[app_id]/GET/[resource_id]
  - CTL/[account_id]/[client_id]/[app_id]/POST/[resource_id]
  - CTL/[account_id]/[client_id]/[app_id]/PUT/[resource_id]
  - CTL/[account_id]/[client_id]/[app_id]/DEL/[resource_id]
  - CTL/[account_id]/[client_id]/[app_id]/EXEC/[resource_id]

- **Response**
  - CTL/[account_id]/[requester_client_id]/[app_id]/REPLY/[req_id]
MQTT Request/Response

Requester (manager1)

Responder (device1)

Responder
Subscribes on request topic:
CTL/kura/device1/CONF-V1/GET/configurations
**MQTT Request/Response**

**Requester** (manager1)

**Responder** (device1)

**Requester**

Generates request metadata (for request payload):
- request_id: 1363603920892
- request_client_id: manager1

Subscribes on reply topic:
CTL/kura/manager1/CONF-V1/REPLY/1363603920892
MQTT Request/Response

Requester (manager1) sends MQTT request to Responder (device1):

Sends MQTT request
CTL/kura/device1/CONF-V1/GET/configurations
MQTT Request/Response

Responder

Handles request and generates response using request specific parameters in the payload of the request
- [request_id] and [request_client_id] included in request payload
- Mandatory response code
- Optional application specific parameters
- Optional logs, exceptions, stacktraces, etc
MQTT Request/Response

Requester (manager1)

Responder (device1)

Responder

Sends response to requesting client
CTL/kura/manager1/CONF-V1/REPLY/1363603920892
MQTT Request / Response

Response Codes: HTTP Like

- response_code
  - 200 (RESPONSE_CODE_OK)
  - 400 (RESPONSE_CODE_BAD_REQUEST)
  - 404 (RESPONSE_CODE_NOTFOUND)
  - 500 (RESPONSE_CODE_ERROR)

- response_exception_message
  - Optional and contains error message if an error occurred

- response_stacktrace_message
  - Optional and contains a stacktrace
Kura OSGi Management via MQTT

- Bundle management
  - Retrieve Bundle information for installed bundles
  - Start/Stop currently installed bundles

- Deployment packages
  - Install new deployment packages (collections of OSGi Bundles)
  - Update/Uninstall existing deployment packages

- Configuration Management
  - Retrieve, review, update Service Configuration
  - Support OSGi MetaType service descriptors

- Remote Access
CoAP

- RESTful protocol designed from scratch IETF

- Transparent mapping to HTTP:
  - GET, POST, PUT, DELETE and URLs

- Features for M2M scenario (observe & sms)

- Based on UDP, draft for using TCP
OMA Lightweight M2M

- Open Mobile Alliance standard
- Device management on top of CoAP
- Object model: for D.M. or Applications (IPSO)
Standard Objects

- Security
- Device
- Location
- Firmware
- Connectivity monitoring & management
- Lock & wipe

You can provide your own object to OMA
URL Example

/\{object\}/\{instance\}/\{resource\}

Example:

“/6/0” : get the whole GPS position instance
“/6/0/2” : get only the altitude resource
### LwM2M in Kura

#### Draft

<table>
<thead>
<tr>
<th>Kura Service</th>
<th>LwM2M Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Service</td>
<td>Device</td>
</tr>
<tr>
<td>Position Service</td>
<td>Location</td>
</tr>
<tr>
<td>Kura</td>
<td>Firmware</td>
</tr>
<tr>
<td>Modem Monitor</td>
<td>Connectivity Monitoring</td>
</tr>
<tr>
<td>Cloud/Data/Transport Service (TBD)</td>
<td>Connectivity Statistics</td>
</tr>
<tr>
<td>Kura Deployment Package</td>
<td>Software Management</td>
</tr>
<tr>
<td>Kura Component Meta-Type</td>
<td>Custom: Software Configuration Definition</td>
</tr>
<tr>
<td>Kura Configurable Component</td>
<td>Custom: Software Configuration Values</td>
</tr>
</tbody>
</table>
LwM2M Software Configuration

Kura Data Service Component

LwM2M Objects

/0 – Custom Object for Kura Component

/90 – LwM2M Object Instance

/90/7 – String Resource (name)

Data Service

/90/7/3 – Multiple String Resource (Value list)

‘false’, ‘80’, ‘10’, ...

/90/7/4 – Multiple Integer Resource (AD list)

0x012, 0x013, 0x014, ...

/91 – Custom Object for Kura Metatype Definitions

/91/20 – LwM2M Object Instance

/91/20/0 – String Resource (id)

disconnect.quiesce-timeout

/91/18/2 – Integer Resource (type)

1
Demo

kura

leshan
Next Steps

- [https://github.com/eclipse/kura/tree/lwm2m](https://github.com/eclipse/kura/tree/lwm2m)
- CoAP as alternative DataTransport in Kura
- LwM2M over MQTT
- Join the discussion:
  - kura-dev mailing list
  - iot-dev mailing list