Internet of Things
2016/2017

LESHAN
(pictures from standards docs & software descriptions in presentations)

Johan Lukkien
Leila Rahman

John Carpenter, 1982
Guiding questions

• How does LESHAN support the development of systems using LWM2M?
Framework for discussing frameworks ;-)  

- What is the goal of the framework? Where is it useful for?  
- What are its elements?  

- Describe the static part (before execution):  
  - programming model  
  - life cycle model  
  - relevant development tooling  

- Describe the dynamic part (during execution):  
  - run-time system  
  - support services  
  - a process model
Leshan goal

- Let developers develop LWM2M servers, clients and IoT applications
  - provide Java libraries (CoAP, LWM2M) and APIs
  - support the programming of the client objects
  - implement the server functionality and make it available to applications
- Monitor an installed system
  - through a Web UI
Layering provides two interfaces

- **Two interfaces:**
  - an API to develop applications, typically as a library
  - the protocol between library instances
    - to be implemented by the library
- Libraries on both sides may be different

- The library can be limited to just a means for generating protocol messages

- **Example:**
  - CoAP protocol
  - Californium library
Leshan programming model

- A programmer develops client and server as separate executables,
  - using Java
  - using an API delivered by Leshan

- The API represents basic operations, e.g.,
  - creating resources, objects
  - sending messages in the correct format

- The real behavior of both client and server are up to the programmer

Example Server (in Java)

// Build LWM2M server
lwServer = new LeshanServerBuilder().build();

// Listen to registrations/deregistration
lwServer.getClientRegistry().addClientRegistryListener(
    new ClientRegistryListener() {
        @Override
        public void registered(Client client) {
            System.out.println("New registered client with endpoint: " + client.getEndpoint());
        }

        @Override
        public void updated(Client clientUpdated) {
            System.out.println("Registration updated");
        }

        @Override
        public void unregistered(Client client) {
            System.out.println("Registration deleted");
        }
    });

// Start
lwServer.start();
System.out.println("Demo server started");
Life cycle of client and server

- Similar in development
- Deployment client: put code on device
  - through direct interaction
  - through a server
- Client execution and termination under control of server
- Application logic via web interface
Tooling

• Test Server Sandbox
  – The lwm2m server at http://leshan.eclipse.org/
  – The bootstrap server at http://leshan.eclipse.org/bs/

• Demo projects and integration tests project

  Leshan-client-demo : a simple demo client.
  Leshan-server-demo : a lwm2m demo server with a web UI.
  Leshan-bsserver-demo : a bootstrap demo server with a web UI.
  Leshan-integration-tests : integration automatic tests.
At run-time

- The Java JRE is the runtime environment for client and server

- The Leshan LWM2M API provides the following functionality
  - Client initiated bootstrap
  - Registration/Deregistration
  - Create objects
  - Read, Write, Execute Resources
  - Resource Observation
  - TLV encoding/decoding

- The process model addresses
  - ‘how it runs’ (not worked out here)
  - concurrency in client
    - responsibility of programmer to use Java concurrency control to avoid blocking waits in client
  - concurrent operation of run-time services in server
At run-time: server

- Services, registries and data stores supporting the LWM2M operations
- Web interface and web API for applications that use the devices

Johan J. Lukkien, j.j.lukkien@tue.nl
TU/e Informatica, System Architecture and Networking

7-Dec-16
Example of service APIs

- HTTP API for requesting list of registered clients

```json
[{
  "endpoint": "TuEIOT1611wGUI",
  "registrationId": "ptP2XQCPu2",
  "registrationDate": "2016-11-23T00:32:53+01:00",
  "lastUpdate": "2016-11-23T00:35+01:00",
  "address": "192.168.0.22:60684",
  "lwm2mVersion": "1.0",
  "lifetime": 30,
  "bindingMode": "U",
  "rootPath": "/",
  "objectLinks": [{
    "url": "/",
    "attributes": {
      "rt": "oma.lwm2m",
      "url": "/0",
      "attributes": { },
      "url": "/3/0",
      "attributes": { },
      "url": "/6/0",
      "attributes": { },
      "url": "/3303/0",
      "attributes": { },
      "secure": false,
      "additionalRegistrationAttributes": {}
    }
  }]
}
```

- HTTP API for requesting the value of a certain resource

```json
{"status": "CONTENT", "content": {"id": 5700, "value": 11.9}}
```
At run-time: client

- Client side is mainly API on top of Californium
  - Californium: CoAP messaging
  - run-time support for common functions

- API support the creation of objects and their logic and to initialize them
  - the objects are approached using the LWM2M messaging
Concluding

• Leshan is mainly a library with application support services inside a single executable
  – to ease the construction and transmission of LWM2M messages
  – to implement standard functionality of LWM2M
  – to increase the abstraction level of development
    • a layer say, that gives functions rather than just messaging

• Provisioning for distributed application is through browser access and the client-server relations

• The respective components are started as executables on the respective devices
Guiding questions

• How does LESHAN support the development of systems using LWM2M?
Framework for discussing frameworks ;-) 

- What is the goal of the framework? Where is it useful for?
- What are its elements?

- Describe the static part (before execution):
  - programming model
  - life cycle model
  - relevant development tooling

- Describe the dynamic part (during execution):
  - run-time system
  - support services
  - a process model