Open the Box

Customer journey in an open Smart Home

Orange Open the Box team
Andre Bottaro, Sébastien Bolle, Jacques Pulou, Antonin Chazalet, Gregory Bonnardel, Cyrille Barea, Fabrice Blache, Orange Labs

Eclipse IoT Day, March, 30th, 2015

http://openthebox.org
@openthebox
Summary

- A **new world of applications** is about to emerge at home thanks to the growing variety of available sensors and actuators.

- To unleash service delivery, a telecom infrastructure is to be **open to third party applications** through cloud and embedded APIs and **open to any device** through local area networks.

- Today technology status enables to open the infrastructure to a set of trusted partners. **Remaining challenges are addressed in the project.**

- This presentation introduces the results of the project related to
  - the customer journey with an open smart home system
  - the device abstraction layer needed for the Smart Home
Open the Box project results

**An open infrastructure with an ecosystem role model**
- Shared sensors
- Shared screens
- Smart Home box
- Gateway
- Application Store
- Smart Home Management Platform
- Remote Access
- Business service platform

**An end-to-end technical architecture**
- Open APIs to 3rd party applications*
- Uniform customer journey*

**Research issues solved on the embedded platform**
- Security attacks and testing platform
- Hardware resource management
- Stale reference resolution for Java components
- Conflict management and application collaboration
- 3 embedded targets: openness vs hardware costs

**Dissemination**
- products
- standards
- open source
- publications
Open APIs to 3rd party applications

- The Smart Home operator provides
  - APIs for partners to access/manage locally and remotely apps and devices
  - an end-to-end infrastructure for the client to select, deploy, install, configure, use applications and devices
- Partners provide base drivers for the share of devices
  - e.g., Delta Dore, Sagemcom, STM, Orange, Bouygues T.
- Partners use APIs to deliver multiprotocol applications, e.g., Delta Dore, Sagemcom, STM, LIG, Orange
Customer journey
Application selection in an application store

Presentation of eligible applications with respect to client home devices
Application store and device access rights

Device access rights are presented to be validated for the selected application
User dashboard

- Dashboard presenting deployed applications and connected devices
Management of device access rights

- Connected things visible by the ‘fire’ app.
- Configuration of device access rights for ‘fire’ app.
- Only devices of relevant types are presented.
Device abstraction layer
3 levels to represent local home devices

1. Business-level application communication
e.g., "decrease the overall energy consumption with 500kW"

2. (Abstract) device communication
e.g., "switch off the radiator with ID=00003"

3. Techno-specific device communication
e.g., "send the command <ZigBee Specific Command> to endpoint ID=<ZigBee specific ID>"
### 3 levels to represent home connected things

1. Business-level application communication
   e.g.,
   "home.setPowerConsumptionTarget(-300kW)"

2. (Abstract) device communication
   e.g., "radiator.setState(OFF)"

3. Techno-specific device communication
   e.g.,
   "ZigBeeEndPoint.getCluster(<clusterID>).invoke(<commandname>, <value>)"
Handling distribution
Service proxies populating a registry

![Diagram of service distribution]

- Required service
- Provided service
- Reaction on service event

**Service Event Bus**

- Platform
- Service Registry
- Provided services

**Device**

- Service Discovery
- *Base Driver*: UPnP, ZigBee, EnOcean, etc.

**Client**

- DeviceService
The role of the Base Driver (BD)
1. Making distributed computing transparent

- Networked devices represented as programmed proxies
- Network management and message parsing by the BD

**Import**
- Method calls are relayed into device control messages
- Listeners registration in the registry replaces event subscription
- Network events are relayed into programmed listener notification

**Export**
- Platform–based services accessible through BD network interfaces
- Event subscription of networked subscribers relayed into listener registration by the BD
- Event notification is relayed on the network by the BD.
The role of the Base Driver (BD)

2. Mirroring the network dynamicity

- **Import: Mirroring network dynamicity on the platform**
  - At starting time
    - The BD searches for available devices
    - Every found device is registered in the platform service registry
    - The platform service registration generates a platform service event
  - At runtime
    - The BD listens to device registration, modification, unregistration
    - The BD answers with platform service registration, modification, unregistration
    - Each platform service operation generates an event on the platform

- **Export: Mirroring platform dynamicity on the network**
  - Platform service registration, modification, unregistration relayed on the network
    - Discovery messages sent by the BD on the network
Example: The EnOcean Base Driver
with OSGi EnOcean Device service standard specification

An EnOcean RPC Descriptor

<<Interface>>
EnOceanRPC
DescriptionSet

1 has
0..n

<<Interface>>
EnOceanRPC
Description

0..n associated with
1

<<Interface>>
EnOceanRPC

invoked by

1

0..n

A listener implementing
EventHandler

An EnOcean device client

requests by
gets

An EnOcean device implementer

1

0..n

EnOceanBase Driver

0..n

1

An EnOcean Message Descriptor

<<Interface>>
EnOceanMessage
DescriptionSet

1 has
0..n

0..n

An EnOcean Message

gets

invoked by

0..n

1

EnOcean Base Driver

0..n

1

EnOceanHost

<<Interface>>
EnOceanChannel
DescriptionSet

1 has
0..n

0..n

An EnOcean Channel

receives messages with EventAdmin

sends messages with EventAdmin

0..n

1
Handling heterogeneity
Refined proxies populating a registry

Client

Domain-specific Service (e.g. Switch)

Base Driver

Protocol-specific Service (e.g. EnOceanDevice)

Refining Driver

Service Discovery

Device

Service Event Bus

Platform

Service Registry

Provided services

Reaction on service event
The role of the Refining Driver (RD)

1. Masking protocol heterogeneity
Mediating protocol specific proxies into domain specific objects
   - Service interfaces are domain-specific, e.g., Media Sources and Sinks
   - Events are domain-specific, e.g., playing status

2. Mirroring service dynamicity
   - At starting time
     - The RD searches for available protocol-specific services
     - Every found service is adapted and registered in the platform service registry
     - The platform service registration generates a platform service event
   - At runtime
     - The RD listens to service registration, modification, unregistration
     - The RD answers with platform service registration, modification, unregistration
     - Each platform service operation generates an event on the platform
Open source, standardization and research perspectives

- **Open source and freeware software**
  - [EnOcean] Orange EnOcean base driver, under transfer to eclipse Smart Home.
  - [Modus] TR-069 device management client, Apache license.
  - [SimpleBee] SimpleBee base driver and Arduino client for fablabs, Apache license.
  - [http://openthebox.org](http://openthebox.org) Orange freeware environment available with tutorials
  - Future work: integrate eclipse Smart Home rule engine and eclipse OM2M cloud APIs

- **Standardization**
  - Base driver layer now in HGI reference architecture and OSGi specifications
    - ZigBee, EnOcean, USB device service specification under finalization at OSGi Alliance
  - Remaining efforts: Standardize Home and IoT device abstraction layer

- **Research perspectives**
  - How to detect and handle device access conflicts between applications?
  - How to share data for the better aggregation, management and mining?
  - How to manage an IoT infrastructure as cloud resources with optimal deployment of application parts?
Thanks
Standardization,

open source results,

and scientific communications
Contributions to the standardization ecosystem

• Contributions to HGI reference architecture
  • technical requirements for resource management on an open gateway
  • a Base Driver sublayer in the full abstraction layer

• Specifications in the next OSGi Residential Specification release – March 2015
  • ZigBee Device Service (RFC 192)
  • EnOcean Device Service (RFC 199)
  • Resource Monitoring (RFC 200)
## Open source results

<table>
<thead>
<tr>
<th>Project Description</th>
<th>Owner</th>
</tr>
</thead>
<tbody>
<tr>
<td>EnOcean Base Driver</td>
<td>Orange</td>
</tr>
<tr>
<td>SimpleBee Base Driver</td>
<td>Orange, LIG</td>
</tr>
<tr>
<td>GreenNet Base Driver</td>
<td>STMicroelectronics</td>
</tr>
<tr>
<td>INCINERATOR – JVM for stale reference free OSGi platforms</td>
<td>Orange</td>
</tr>
<tr>
<td>Secure coding good practices (French)</td>
<td>Sogeti High Tech</td>
</tr>
<tr>
<td>Attacks defined and implemented on a testing platform targeting OSGi technology</td>
<td>Sogeti High Tech</td>
</tr>
<tr>
<td>(French – to be translated)</td>
<td></td>
</tr>
<tr>
<td>MODUS – TR-069 Device Management client</td>
<td>Orange</td>
</tr>
<tr>
<td>RoSE – Distributed OSGi framework and sensor interaction</td>
<td>LIG</td>
</tr>
<tr>
<td>APAM – Framework for applications collaboration</td>
<td>LIG</td>
</tr>
</tbody>
</table>
Technical and scientific communications
Two topics

- **Software Engineering of adaptive applications**
  Open the Box project has contributed to the state of the art the functionalities of an embedded service platform: dynamic dependency management, applications modeling, automatic deployment, autonomic application management.

- **Software engineering of multi-tenant service platforms**
  Open The Box project envisions a software platform shared by a dynamically extensible set of applications from distinct providers. This objective raises challenges to enable fair resource sharing and isolation, and to address security issues with respect to bugged and malicious applications.
Software engineering of adaptive applications

### Publications


### PhD theses

- J. Bardin (LIG) RoSe : a framework to design and execute dynamic distributed heterogeneous applications. Oct. 2012. (French)
- Diana Moreno-Garcia. Modèles, outils et plate-forme d'exécution pour les applications à service dynamiques. Grenoble University, Feb. 2013. (French)
Software engineering of multi-tenant service platforms

- **Publications**

- **PhD theses**
  - Koutheir Attouchi. Managing Resource Sharing Conflicts in an Open Embedded Software Environment, Pierre et Marie Curie University, July 2014. (English)