Home SOA – Controlling Home Pervasive Devices

André Bottaro[1, 2], Anne Gérodolle[1], Levent Gurgen[1]

[1] *France Telecom – Orange Labs R&D*
28, chemin du vieux chêne,
38240 Meylan. FRANCE
{firstname.lastname}@orange-ftgroup.com

[2] *Grenoble University*
Laboratoire LIG-Adele
38041 Grenoble, Cedex 9, France
{firstname.lastname}@imag.fr

Abstract

The home network becomes a ground for pervasive computing design. Its openness to dynamic distributed and heterogeneous devices emphasizes the pervasive challenges in home application design. We present here a protocol-agnostic application inventorying available home devices and making the user able to configure and test device operations. First, proxying techniques are applied in order to mask distributed aspects in the device reification on the execution platform. Second, mediation techniques are used to mask the heterogeneity of home device protocols in uniform programming interfaces. Finally, device availability is dynamically mirrored by the application thanks to service discovery mechanisms reified on the platform and thanks to web 2.0 graphic interfaces.

1. Context

The Home network is open by essence. First, it is open to dynamic connections: devices enter and leave the network, providing context-dependent functionalities (e.g. according to users location and activity). Second, it is open to heterogeneous devices: protocols [5] (see Figure 1) and device types differ according to application domains and service providers. Moreover, devices are numerous and spread over the home space. Designers of innovative home applications therefore have to face the three main challenges of this pervasive environment: dynamicity, heterogeneity, distribution.

2. Use case

Mr. Techo, a technician working for an Internet Service Provider, can monitor the home network of all customers who, like Nathalie, subscribe to this option. Today, Mr. Techo is dealing with a message concerning Nathalie's network. The application has detected a device known for giving problems. With the customer's authorisation, Mr. Techo tests the state of the device and its compatibility with her other equipment. The Home SOA prototype could also be included in a system that processes such a case automatically, without human intervention, as soon as Nathalie gives her permission.

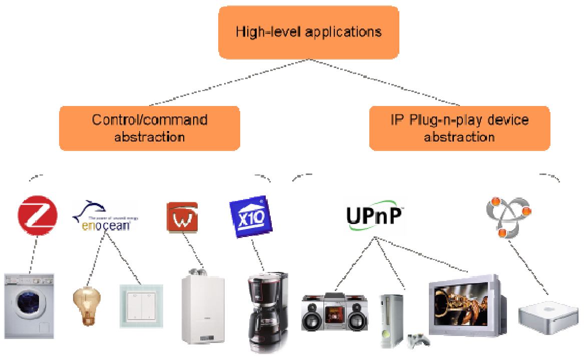
High-level applications
Control/command abstraction
IP Plug-n-play device abstraction
enocean
X10
UPnP

Figure 1 Supporting an extensible list of protocols

3. Architecture

The Home SOA prototype is developed with the OSGi technology [1]. OSGi technology provides the foundations for service-oriented programming [6]. Home SOA utilizes these foundations by recommending the use of advanced design patterns.

In Home SOA, service-oriented drivers represent each device as a Java object and uses OSGi mechanisms to transfer the home dynamics to the programming-language level. All the events on the home network are processed by registering, removing and modifying the objects in the OSGi service registry. The latter becomes the receptacle for all entities distributed on the network.

Also, while reflecting the dynamics of the home network in the programming language, the Home SOA techniques mask the protocol details of each supported technology (here UPnP/DLNA, DPWS, Apple Bonjour/iTunes, X10) in the object representation of the devices. The goal is to let the objects that are handled by the developer, be exempt from these technological aspects. Each protocol set is processed by a series of bundles called a Base Driver (see Figure 2). The list of supported protocols is thus extendable.

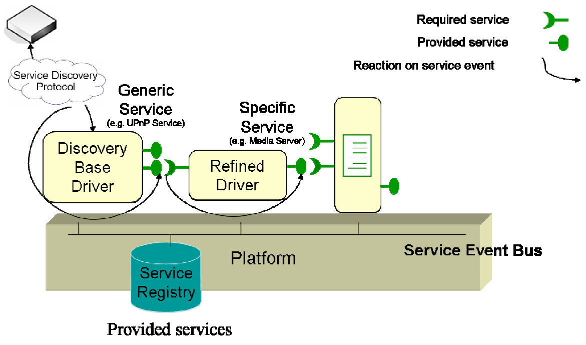

Figure 2 Dynamic mediation on a service platform

Finally, Home SOA deals with the heterogeneity of the technologies mentioned above. Uniform interfaces are obtained by mediating towards a set of programming interfaces that are close to a targeted application domain. Refined drivers (see Figure 2) are dynamically reacting to the registration, modification, unregistration of device proxies by discovery base drivers. On each event, this driver registers, modifies or unregisters a adaptor implementing an interface whose semantic is closer to the application logic above.

4. Use case implementation

For instance in the described scenario, the shared interfaces for device management comply with the JWSDL standard [7] – a Java representation of WSDL descriptions. The various objects that represent the UPnP, DPWS, iTunes, and X10 devices are transformed into objects that comply with the Java-standard interface set (see Figure 3), which is associated to the additional notion of device.

This language interface pivot for device control protocols is named Smart Device API in Figure 3. Refined drivers – the second range of drivers depicted on the figure – transform every registered device entity in the registry into a *SmartDevice* service. The application – named Smart Application in the figure – is then able to dynamically treat any available device independently of its original technology thanks to this uniform device interface.

The tool that is used by Mr Techo consists of web pages served on an OSGi platform embedded in a home device. The Java servlets are graphically representing the *SmartDevice* objects of the service registry. In order to dynamically inventory available devices, Ajax technology is used in order to refresh web page content without user manual intervention.

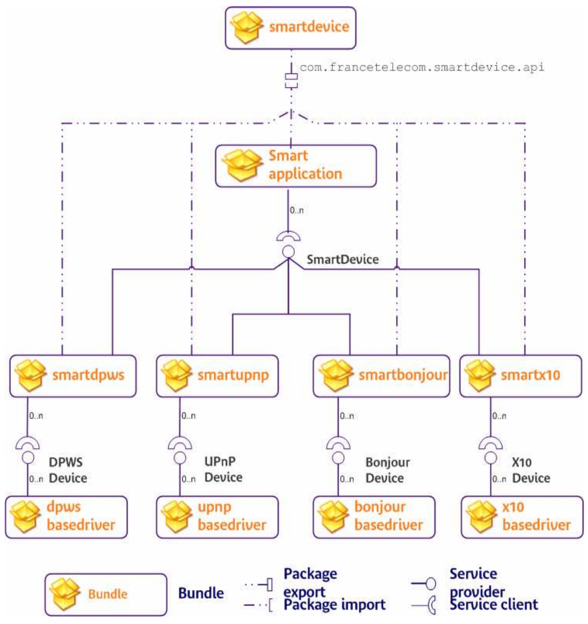

Figure 3 Smart device API – a language pivot

We implemented this architecture and tested applications above the OSGi R4 felix open source platform. Parts of the work are delivered open source in the IST Amigo project and some related actions are carried out into the OSGi standardization process [2][3][4].

5. References

1 OSGi Alliance, "OSGi Service Platform Core Specification and Service Compendium Release 4", October 2005.

2 André Bottaro, "RFP 72 Extended Mapping for UPnP Discovery Transparency", OSGi Alliance, April 2006.

3 André Bottaro, Anne Gérodolle, Sylvain Marié, Stéphane Seyvoz, Eric Simon, "RFP 86 DPWS Discovery Base Driver, OSGi Alliance", May 2007.

4 André Bottaro, Anne Gérodolle, Sylvain Marié, "Combining OSGi technology and Web Services to realize the plug-n-play dream in the home network", OSGi Community Event, Munich, Germany, June 2007.

5 Feng Zhu, Matt W. Mutka, and Lionel M. Ni, "Service Discovery in Pervasive Computing Environments", IEEE Pervasive Computing, vol. 4, pp. 81-90, 2005.

6 H. Cervantes, R. Hall, "Autonomous Adaptation to Dynamic Availability Using a Service-Oriented Component Model", 26th ACM International Conference on Software Engineering, Edinburgh, May 2004.

7 Java Community Process, "JSR 110: Java APIs for WSDL – Final Release 3", September 2006