Service-oriented architecture for home networks

Long Ngo
Helsinki University of Technology
lngo@cc.hut.fi

Abstract

The architecture of traditional smart home, Server Centralized Architecture, has limitation on interoperability, scalability and extensibility. The dynamic in the home environment also makes the control of appliances difficult. This paper presents Service Oriented Architecture as a new architecture to solve the problems. The paper also mentions the current problems, which need to be researched more, with this architecture.

KEYWORDS: service-oriented architecture, home appliance, home automation, networked appliance, smart home

1 Introduction

Nowadays home appliances are not stand-alone things anymore. They are becoming information appliances, or they can be networked to exchange their information. This technology brings a new future for how a house can serve users. Every thing in your house will be connected and work automatically. For example, when you receive a land-line call while watching a movie in TV, the TV will automatically reduce the volume. When you finish, the sound volume will be turned back to normal level. A house with this kind of appliances is called a smart house, smart home or house with home automation.

This technology can also assist disabled or elderly, or even vehicular networks[11]. For instance, you use the toilet, it will analyze your urine for information about your health condition. If the information indicates that you are not so well today, the toilet will send a notice to other smart equipments, for example to your car. Knowing that news, the car will limit its speed to 30 kms/hour when you are driving. In that way, the technology brings safety to people.

However, today is just the dawn of home automation so that we have not had any standard technology yet. In the current picture of home automation, different vendors have been developing different technologies for different purposes. These technologies can not work together without any *"bridge"* between them. In most cases the system requires a central server, which is not only to control all other appliances but also to interconnect sub systems of different technologies. This architecture is called *Server Centralized Architecture* and it has been known to have limited interoperability, scalability and extensibility. For example if a new appliance has a new technology that is not supported by the server, then the new appliance can not join the system. In addition, because of tight coupling, changing in client side also requests for changing in server side. For these reasons, it is necessary to have a better architecture. *Service Oriented Architecture* (SOA), a software architecture evolved from *Distributed System*, is proposed to be applied in this area. SOA can solve the problems we have with SCA, promising a suitable architecture for the future home networks. In this paper, we describe how SOA can be applied for home networks to solve the problems that SCA have. The paper also mentions the issues, which remain to be researched more, with SOA.

The rest of the paper is organized as follows. Section 2 gives an overview of technologies used for networking appliances, the current architecture and its problems. After that the section 3 presents SOA as an architecture that can solve the problems. Section 4 discusses about implementation of the architecture and potential future research. The last section concludes the paper.

2 Enabling technologies and state-of-the-art

In this section we review the current situation of networked appliances. By having an overview of the technologies and their architecture, we can understand why the architecture may not be suitable in the future.

2.1 Networking technologies and middleware

To network appliances, we need technologies for physical network layer and middleware that provide control of hardware for application layer.

Physical media for home networks are divided into three broad groups:

- Structured wiring: New cable is installed for appliance networks. Some typical technologies are Ethernet, Universal Serial Bus (USB), IEEE 1394

- Existing wiring: Use existing home infrastructure such as electrical, telephone and so on. Some technologies of this group are X10, HomePlug, LonWorks, CEBus, HomePNA

- Wireless: Wireless connectivity is a great way for networking appliances, because it brings mobility and convenience. Prominent candidates are IEEE 802.11, Bluetooth, Infrared, IEEE 802.15.3, IEEE 802.15.4. It is hard to say which one will be the best, because they are designed for different situations[11].

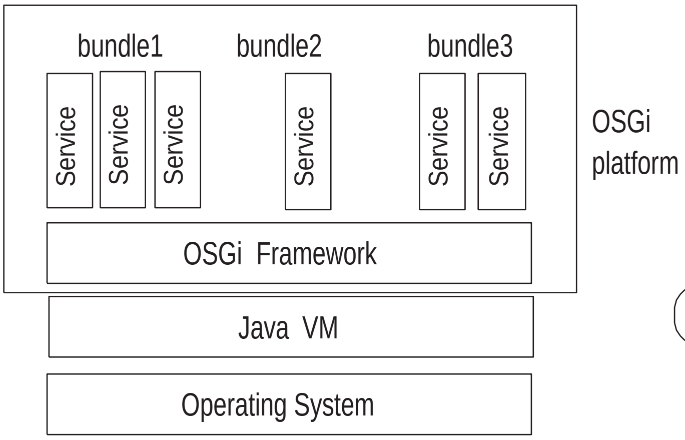

Figure 1: OSGi layers [8]

Middleware is a software component that sits on the top of a home device's operating system. Middleware is used to isolate application from hardware. It provides abilities to discovery other appliances' service and communicates with each other. There are several middleware developed for networked appliances, but they serve for different purposes and are not interoperable with each other. For example, UPnP is a set of interfaces that allow you to plug any appliance with "zero configuration", Jini is a Java-based distributed platform for information appliances, HAVi is a protocol for transporting real-time audio/video stream, and so on. We will analyze how this abundance of technologies brings difficulties to home network development in Sec. 2.3. However, OSGi, a middleware technology with its original purpose is to build *service gateways* in the server-centric architecture [2], that provides the ability to interconnect networks with different technology. We discuss more in details about OSGi because now it is adopted to use in many SOA implementations.

The Open Services Gateway Initiative (OSGi) is a managed, extensible framework to connect various devices in home and vehicular networks [9]. OSGi framework supports building extensible and downloadble Java-based service application, or *bundles*. An OSGi platform includes an instance of Java virtual machine, the OSGi framework and a set of bundle[9]. It now is becoming a very important technology today and some service-oriented architecture have been implemented based on this framework (see Section 4). Bundles collaborate by registering its services with the framework's service registry. It should be noticed that the framework provides only interfaces between the framework and services and leave the real implementation to bundle developers. Fig. 1 depicts the OSGi framework architecture. OSGi also provides specification for *service gateways*, which coordinate different device technologies support them to communicate through different networking technologies.

While other technologies such as Jini, UPnp or HAVi concentrate on interoparability, OSGi plays an complementary role by concentrate on delivery of services[6]. Fig. 2 explains how different networking technologies can be combined in an OSGi home network. We will see why OSGi is preferable for implementing SOA in Sec. 4.

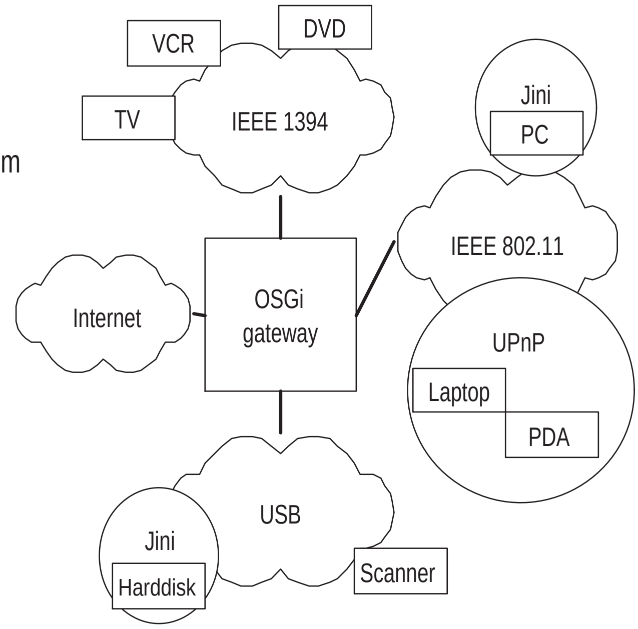

Figure 2: OSGi home network architecture [4]

We just have a brief look at current technologies for networked equipments. Discussing them in details is out of the scope of this paper.

2.2 The traditional architecture

In the traditional architecture, all components in a homenetwork are controlled by a home gateway, which acts as service provider for users[2]. Users control all other appliances through this server. Users must define scenarios about how home equipments work together from this server. From the defined scenarios, the server knows to it will control other equipments. And to be able to control and interconnect all other devices, the server must be able to work with appliances that have different technologies and protocols. We can see in Fig. 3 that the home gateway controls other home appliances and also connects with other user devices through the Internet.

In some cases,appliances can work in a peer-to-peer network, but this feature is applicable to appliances that use the same protocol. The reason is they can not understand the others that use different protocols. That is why the home gateway is needed to act as a service gateway, translating between different protocols for appliances.

2.3 Current issues

This section discusses some issues with current technologies, which lead to the need of better architectures, such as service-oriented architecture.

One problem with most of current solutions is interoperability. Today appliances are restricted to a specific kind of services or technologies [11]. This makes appliances with different technologies incompatible. For example, an oven with Jini technology can not talk with a TV with HAVi technology. We can use a service gateway to control and inter-

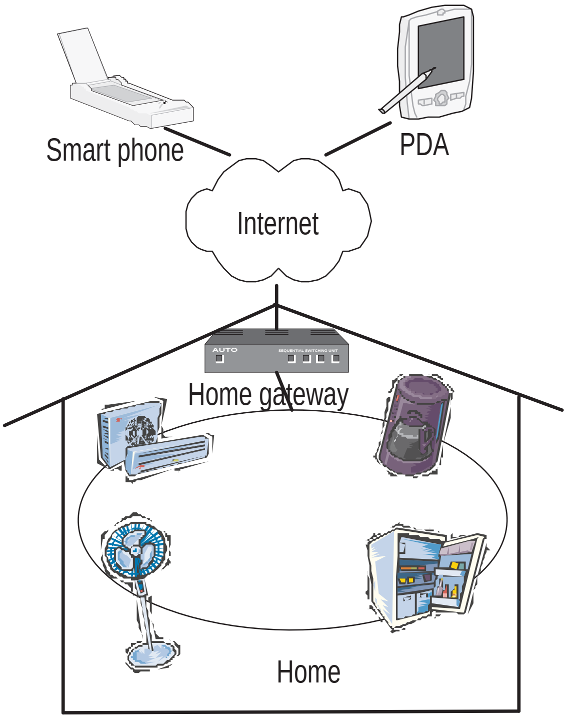

Figure 3: The traditional architecture for smart home

connect different appliance systems, but it brings difficulty of usage and maintenance. Users demand all of their home equipments communicate and work together seamlessly, regardless of what service discovery and communication technologies they use. Otherwise the benefits from networking appliances would be limited.

Another problem is scalability. Because we use a service gateway to interconnect different systems from different technologies, there might be a problem with a new device whose communication technologies are unknown for the gateway. In this situation, users have to update their server, that is always complicated, or they can not use the new device.

The tight-coupling between appliances leads to the problem of extendibility. In most cases, a change in a certain appliance makes it can not provide its service to others, because now they can not understand the appliance anymore. From user's point of view, every appliance should work even when another one changes its interface or when a new appliance is installed into the network.

The mobility of appliances and the dynamic of appliances' configuration also cause to a problem of controlling. In this case, it is better to let appliances work autonomously, rather than to have one server to control every thing [2]. A server that can work with different protocols can be too complicated for a normal user to know how it works.

All problems mentioned above urge us to have a better architecture, which can solve the problems, for home networks in future.

3 Service-oriented architecture

This section talks about a new architecture, SOA, and explains why it could be a good choice for home networks.

3.1 Service Oriented Architecture in general[5]

SOA is a software architecture, that is not a revolution but an evolution from *Component Based Architecture*, *Object Oriented Architecture* and *Distributed Systems*. Some remarkable technologies today such as *Web Services*, *.NET*, *J2EE*, *CORBA* or *ebXML* are implementations of SOA. The idea of SOA is that services with defined interfaces can be called to perform their tasks, without knowing the calling applications and the application also do not need to know how the services work. Every component in a system advertises its services, meanwhile discovering the services from others. Components negotiate to use the services from each other without any central controller. The followings are some main SOA concepts, which are used in all of SOA implementations.

- **A service** is a defined behavior that can be implemented and provided by a component.
- **A service description** contains information in a standardized format about the service, including technical parameters, constraints and policies
- **Advertising** is the way a service lets potential consumers know about it. There are two methods.*Pull* is used when a service reply the request from a potential consumer. *Push* is the method when a service sends its service description to potential consumers, not necessarily having a request before.
- **Discovery** is a mechanism for a potential consumer to get information about the existence of a service and its description.
- **A registry/repository** is a component where users store and manage their artifacts, which are needed for their enterprise to function
- **A directory** is an interface about the artifacts and how to bind to them.

The reasons why SOA has been adopted widely are linking computational resources and promoting their reuse. It is worth to notice that reuse in SOA is in a macro level (services), not a micro level (objects).

3.2 Home appliance networks in SOA[10]

3.2.1 How appliances communicate

SOA can be applied in networks of home appliance to link different appliance types, making them able to communicate with each other in a standardized manner. Each appliance system shows its services and their interfaces in a form of *exported methods*. A standardized platform-independent framework provides remote procedure call for appliance systems. Fig. 4 depicts the way appliances call others' services.

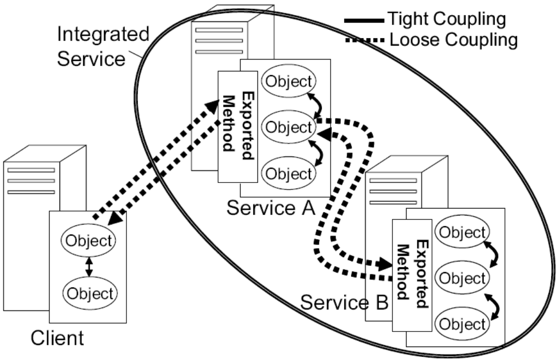

Figure 4: Home appliance network in SOA [4]

Each client A and B has its internal objects, but they show only exported methods.

The way appliances work together must meet the following requirements:

- **Standard communication and loose coupling:** It guarantees that any appliance can interoperate with others and changes inside each appliance do not affect the network.
- **Autonomous interoperation:** This requirement allows appliances to function without any centralized component or foreknowledge about others.

Of course for application layer to export the methods, it is necessary that this layer does not have to care about what protocols and physical media underneath. We will focus more about this issue in the next section.

3.2.2 Appliance structure

For appliances that can be networked, each one must have hardware, including a processor, a network interface and application storage. Applications control the hardware by a set of APIs. To meet the two requirements in Sec. 3.2.1 each appliance must have the following features:

- **Exporting Self-Feature:**Device interfaces are encapsulated and exported to the network in a standardized manner.
- **Controlling Other Appliances:** Invoke other exported methods according to given scenarios.

From the viewpoint that appliances are components that provide services to others, an appliance contains two layers, a device layer and a service layer, as shown in Fig. 5

The device layer is the hardware of an appliance, including the middleware. The control method of each appliance depends on what technologies are used. For example, ECHONET[3] for sensors and lights, HAVi[7] for set-top boxes, etc. This layer provides the set of APIs for the service layer to communicate with other devices.

The service layer wraps the device interfaces and exports them to the network as understandable services for others. In an appliance, basically the service layer is implemented as an application, which uses the APIs of the device layer. From the view of this layer, there is no difference between appliaces. Every device exports and discovers services in a standardized way, although how each appliance works inside could be different. In each export method there is a mechanism to trigger other methods of other appliances. Any SOA framework can be used for method exportation, for example Web Services (with SOAP/XML and WDSL). We will discuss more about implementation in the Sec. 4

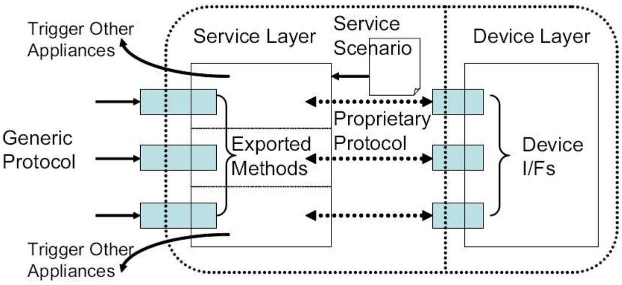

Figure 5: Appliance architecture [10]

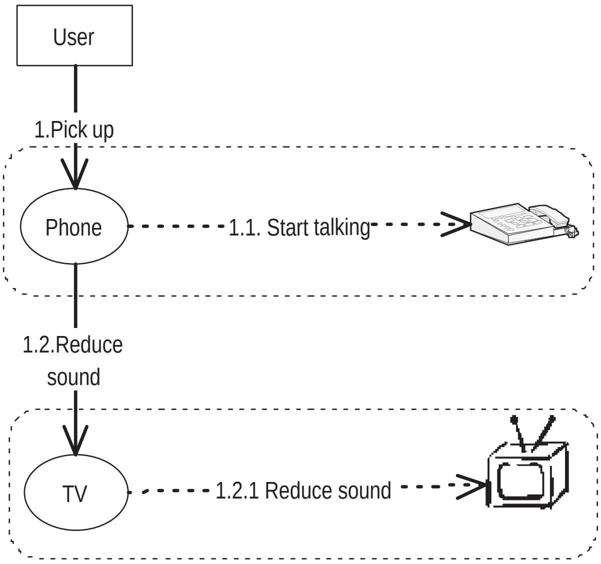

Figure 6: Scenario example

3.2.3 Service scenario example

We continue with the example in the Sec. 1. This is one of many possible scenarios in a home network. Now we examine how the scenario is designed to make the components in the network work autonoumously.

In the Fig. 6, the scenario is triggered when the user picks up the phone. The *Phone* service will call an internal method *1.1* to let user connected with the caller. After that, it calls the *Reduce sound* method from *TV* service. The *TV* service calls its internal method to reduce the sound volume.

The scenarios that appliances follow are defined clearly for each home network.

3.3 Why SOA

We can see that SOA can solve the mentioned issues in Sec. 2.3. In SOA, appliances advertise their services with the description in a standardized format. As a result, if two different devices use different technologies, they can still understand how to call each other's services. And because the architecture produces loose coupling, no appliance depends on others' work, so it provides extendability and scalability.In addition, each appliance can discover what services others offer and also advertise its services, they can work autonomously with minimum user's control. Changing is not a problem anymore.

4 Implementation Issues and Future Research

In general, we can use any implementation of SOA, such as *Web Services*, *.NET*, *J2EE*, as a framework to build applications for the service layer. However, the framework must be supported by the middleware. So far OSGi framework, a part of OSGi platform, is the best choice for developing smart home. The reason is that the original goal of OSGi is to provide a framework for building central service gateway. Because of this goal, OSGi provide bridges to different protocols, for example OSGi-to-UPnP or OSGi-to-Jini. By providing such features, OSGi allows a service gateway to work with a various types of appliances with different technologies. However, up to now OSGi still supports only few technologies and more work should be done. In [2] a SOA based on OSGi and Mobile-Agent technology was proposed. Mobile-Agent technology is used to make the system more dynamic and process data remotely. [10] uses *Web Services* to prototype a system.

The way users control a SOA network is different from a SCA one. Users will define scenarios in each appliance. Each appliance discovers others' services then displays a list of available ones. This work requires each appliance to be able to discover the services in the network. The only thing users need to do is to define how each appliance behaves towards others. Basically this job does not require much effort or knowledge. Normally vendors make built-in default scenarios for their products, so that the products can work automatically together without any initial set up. [10] proposes a way for users to define scenarios for appliances.

Although SOA promises a better solution for home network, it still has the following issues that we should research more. The first issue is adaptation to failure [11]. We can not guarantee that network connectivity is OK all of the time or all of appliances work correctly. The second issue is conflict. What happens if the DVD theater requests the curtain to close, but the illuminometer asks it to open at the same time?. Or there may be problems when appliances receives message in a wrong order. In SCA architecture these problems can be solved easily with the centralized server, which acts as the controller for all actions. The third is performance of the network. When the network grows, communication overhead could be a problem [10][11]. The fourth issue is resource constraint. In SCA all of workload is on the server, but in SOA we need more intelligent appliances. However, many appliances have only limited memory and processing resource. If we make appliances with more memory and faster processors, we will face the problem of cost. Last but not least, a very important issue is security. Because the environment of networked appliances is different from traditional computer networks, so we may have some new kinds of threats here. A compromised appliance can give wrong request that can cause damage. Information sent between appliances can also be changed for malicious purpose. Denial-Of-Service can be carried out by sending messages to an appliance until it runs out of battery. One basic model of security of confidentiality, integrity and availability is analyzed in [11]. Security for smart home has been considered for communication [1] or hardware-supported security [12], but not yet specifically for smart home in SOA.

5 Conclusion

In this paper we have analyzed the problems with the current architecture for smart home. We also discussed about how SOA can be applied for smart home and how it can solve the existing problems. Some implementations of SOA for smart home have been developed but they are still for research in laboratories. There are still many issues to be solved before SOA can be applied for commercial products. However, SOA is a promising architecture and good candidate architecture for the future smart home.

References

[1] P. Bergstrom, K. Driscoll, and J. Kimball. Making home automation communications secure. *Computer*, 34(10):50–56, 2001.

[2] L.-C. F. Chao-Lin Wu, Chun-Feng Liao. Service-Oriented Smart-Home Architecture Based on OSGi and Mobile-Agent Technology. *Systems, Man and Cybernetics, Part C: Applications and Reviews, IEEE Transactions*, pages 193–205, 2007.

[3] E. Consortium. ECHONET Overview. *URL http://www.echonet.gr.jp/english/*, December 2006.

[4] P. Dobrev, D. Famolari, C. Kurzke, and B. Miller. Device and service discovery in home networks with OSGi. *IEEE Communications Magazine*, 40(8):86–93, 2002.

[5] T. Erl. *Service-oriented architecture*. Prentice Hall PTR Upper Saddle River, NJ, 2004.

[6] R. Hall and H. Cervantes. Challenges in building service-oriented applications for OSGi. *Communications Magazine, IEEE*, 42(5):144–149, 2004.

[7] HAVi. The HAVi Specification: Specification of the Home Audio/Video Interoperability (HAVi) Architecture. *URL http://www.havi.org*, May 2001.

[8] T. Leader, B. Hillen, and I. Passchier. OSGi in Residential Gateways. *Residential Gateway Environment project*, March 2004.

[9] C. Lee, D. Nordstedt, and S. Helal. Enabling smart spaces with OSGi. *Pervasive Computing, IEEE*, 2(3):89–94, 2003.

[10] M. Nakamura, H. Igaki, H. Tamada, and K. Matsumoto. Implementing integrated services of networked home appliances using service oriented architecture. *Proceedings of the 2nd international conference on Service oriented computing*, pages 269–278, 2004.

[11] D. O'Sullivan. An Advanced Appliance Interaction Architecture. *Master thesis, University of Dublin, Trinity College*, 2005.

[12] J. Seigneur, C. Jensen, S. Farrell, E. Gray, and Y. Chen. Towards Security Auto-configuration for Smart Appliances. *Proceedings of the Smart Objects Conference*, 2003, 2003.