

國立交通大學

資訊科學與工程研究所

碩士論文



使用者界面標準化之萬用家電控制平台

Standards-based User Interface Technology

for Universal Home Domination

研究生：林良彥

指導教授：袁賢銘 教授

使用者界面標準化之萬用家電控制平台
Standards-based User Interface Technology for Universal Home domination

研究生：林良彥

Student : Liang-Yen Lin

指導教授：袁賢銘

Advisor : Shyan-Ming Yuan

國立交通大學

資訊科學與工程研究所

碩士論文



Submitted to Institute of Computer Science and Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

June 2006

Hsinchu, Taiwan, Republic of China

中華民國九十五年六月

國立交通大學

研究所碩士班

論文口試委員會審定書

本校 資訊科學與工程 研究所 林良彥 君

所提論文：

使用者界面標準化之萬用家電控制平台
Standards-based User Interface Technology for Universal Home Domination

合於碩士資格水準、業經本委員會評審認可。

口試委員：

招興 _____

鄭家文 陳穎年

指導教授：

招興

所長：

翁之貴

中華民國九十五年 月 日

國立交通大學

博碩士論文全文電子檔著作權授權書

(提供授權人裝訂於紙本論文书名頁之次頁用)

本授權書所授權之學位論文，為本人於國立交通大學資訊科學與工程研究所 資訊系統 組，94 學年度第 二 學期取得碩士學位之論文。

論文題目：使用者界面標準化之萬用家電控制平臺
指導教授：袁賢銘

■ 同意

本人茲將本著作，以非專屬、無償授權國立交通大學與台灣聯合大學系統圖書館：基於推動讀者間「資源共享、互惠合作」之理念，與回饋社會與學術研究之目的，國立交通大學及台灣聯合大學系統圖書館得不限地域、時間與次數，以紙本、光碟或數位化等各種方法收錄、重製與利用；於著作權法合理使用範圍內，讀者得進行線上檢索、閱覽、下載或列印。

論文全文上載網路公開之範圍及時間：

本校及台灣聯合大學系統區域網路	<input checked="" type="checkbox"/> 立即公開
校外網際網路	<input checked="" type="checkbox"/> 立即公開

■ 全文電子檔送交國家圖書館

授權人：林良彥

親筆簽名：林良彥

中華民國 95 年 7 月 1 日

國立交通大學

博碩士紙本論文著作權授權書

(提供授權人裝訂於全文電子檔授權書之次頁用)

本授權書所授權之學位論文，為本人於國立交通大學資訊科學與工程研究所 資訊系統 組，94 學年度第 二 學期取得碩士學位之論文。

論文題目：使用者界面標準化之萬用家電控制平臺
指導教授：袁賢銘

■ 同意

本人茲將本著作，以非專屬、無償授權國立交通大學，基於推動讀者間「資源共享、互惠合作」之理念，與回饋社會與學術研究之目的，國立交通大學圖書館得以紙本收錄、重製與利用；於著作權法合理使用範圍內，讀者得進行閱覽或列印。

本論文為本人向經濟部智慧局申請專利(未申請者本條款請不予理會)的附件之一，申請文號為：_____，請將論文延至____年____月____日再公開。

授權人：林良彥

親筆簽名： 林良彥

中華民國 95 年 7 月 1 日

國家圖書館 博碩士論文電子檔案上網授權書

(提供授權人裝訂於紙本論文本校授權書之後)

ID:GT009323526

本授權書所授權之論文為授權人在國立交通大學資訊科學與工程研究所 94 學年度第二學期取得碩士學位之論文。

論文題目：使用者界面標準化之萬用家電控制平臺
指導教授：袁賢銘

茲同意將授權人擁有著作權之上列論文全文（含摘要），非專屬、無償授權國家圖書館，不限地域、時間與次數，以微縮、光碟或其他各種數位化方式將上列論文重製，並得將數位化之上列論文及論文電子檔以上載網路方式，提供讀者基於個人非營利性質之線上檢索、閱覽、下載或列印。

※ 讀者基於非營利性質之線上檢索、閱覽、下載或列印上列論文，應依著作權法相關規定辦理。

授權人：林良彥

親筆簽名：林良彥

民國 95 年 7 月 1 日

中華民國 九十五年六月

使用者界面標準化之萬用家電控制平台

研究生: 林良彥

指導教授: 袁賢銘

國立交通大學資訊科學與工程研究所

摘要

家電控制的技術在資訊產業中所佔有的份量日漸提高，但是一切都還處於未標準化的狀態，每一個家電廠商都有自己選擇的一套家電控制協定，再加上早就存在的傳統家電，家電控制這個領域的標準統一也就遙遙無期。目前許多研究都著重在底層的異質控制協定的整合，卻往往忽略了家電遙控器也是很重要的議題，因為遙控器是整個家電控制系統和人類接觸的最前線，要將之考慮進去才是完整的設計。

目前存在的資訊家電控制協定有 UPnP、Jini、Lonworks、X10、Insteon…等等，而底層的傳送媒介包括有線網路、無線網路、GPRS/3G 網路、藍芽、電源線、紅外線…等等。在這樣的混沌未明情形下，比家電廠商更痛苦的就是遙控器廠商，若要做出萬用的遙控器，就必須支援各種底層網路和控制的協定，並且為每一種家電裝置設計使用者界面以供操作，完全不敷成本。

為了解決上面的問題，我們設計了一個萬用的家電控制平台，可以同時運行資訊家電以及傳統家電，並且讓家電廠商和遙控器廠商都可以降低開發的痛苦指數。我們在 OSGi 這個以服務為導向的架構上面，將不同的家電控制協定製作成服務包，服務包之間便可以透過統一的介面提供服務。底層的異質控制協定的整合後，我們再依循 AIAP-URC 國際標準，加上使用者界面抽象化描述。在這個平台實現了「每一種家電都可以被每一種遙控器控制」和「使用者界面只需設計一次即可到處使用」的目的。

Standards-based User Interface Technology for Universal Home Domination

Student: Liang-Yen Lin

Advisor: Shyan-Ming Yuan

Institute of Computer Science and Engineering

National Chiao Tung University

Abstract

There are two roles in home control scenario, they are man and machine, and a complete home control system must concern both of them. Based on AIAP-URC and OSGi standards, we developed a SUIT home control platform which is suitable for any kind of home appliances and remote consoles. The SUIT platform provides an easy but standardized environment for both appliance and remote console manufacturers to develop their products. In addition, manufacturers are able to update software on the home gateway remotely and dynamically, and that is how we realized “Universal home Domination”. In response, we pretend ourselves as the manufacturers of IM remote consoles. I am also home appliance manufacturers of InsteOn, X10, UPnP, IR Robot. These products are developed successfully on the SUIT gateway.

Acknowledgement

首先我要感謝袁賢銘教授這兩年來的創意啟發和指導，讓我可以充滿樂趣的環境中做學問；也感謝博士班學長們的諄諄教誨，葉秉哲、蕭存喻、吳瑞祥、邱繼弘、鄭明俊以及高子漢學長，感謝你們這兩年給我的意見及幫助，讓我可以順利完成這篇論文。另外也要謝謝實驗室的其他同仁們，碩二的同學們、碩一的學弟妹的關心。

大學的同學們，資科 93 級的朋友們，大家在學業和生活上面的互相照顧以及煩惱的分擔，讓我覺得研究生活感覺有依靠，謝謝你們。還有陪著我已經有七年的惠尹，謝謝你一直陪著我，照顧我，和我一起煩惱、一起開心，在背後默默的付出。

最後要感謝的是我的父母，對於栽培我都是無怨無悔的付出，以及許多社會經驗的教導，讓我可以順利完成碩士學位。對於未來，我會更加努力不辜負你們的期望。

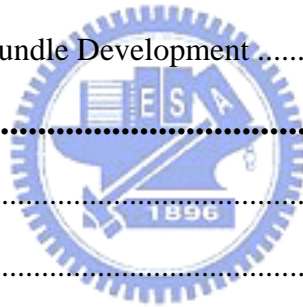


Table of Contents

Acknowledgement	ix
Table of Contents.....	x
List of Figures	xiii
List of Tables	xv
List of Codes	xvi
Chapter 1 : Introduction.....	1
1.1. Preface	1
1.2. Motivation	1
1.3. Objectives	2
1.3.1. Ease the pain for home appliance manufacturer.....	3
1.3.2. Ease the pain for remote console manufacturer.....	3
1.3.3. Heterogeneous Networks Integration	4
1.3.4. User Interface Abstraction.....	4
1.3.5. Access Control and Dependency Policy.....	5
1.3.6. Synchronization.....	5
1.4. Organization	6
Chapter 2 : Background	7
2.1. Home Appliances	7
2.1.1. IR Technology	7
2.1.2. Powerline Technology	7
2.1.3. X10 and INSTEON Technology.....	8
2.1.4. Lonworks Technology	9
2.1.5. Jini Network Technology.....	9

2.1.6.	UPnP Technology	10
2.2.	Remote Consoles	11
2.2.1.	Traditional Remote Control.....	11
2.2.2.	Information Appliance Remote Console	11
2.2.3.	3G/GPRS/Blurtooth Mobile Device.....	12
2.2.4.	Instant Massaging.....	12
2.3.	User Interface Technology	13
2.3.1.	AIAP-URC Standards.....	13
2.4.	Service-oriented Infrastructure	15
2.4.1.	OSGi Service Platform	15
2.5.	Related Works	17
Chapter 3 : System Architecture.....		18
3.1.	Overview	18
3.2.	Roles	21
3.3.	SUIT Platform	23
3.3.1.	SUIT Core Part	25
3.3.2.	SUIT Remote Console Part	27
3.3.3.	SUIT Home Network Part	30
3.4.	User Interface Generation.....	33
3.5.	Control Invocation.....	34
3.6.	Summary.....	35
Chapter 4 Design Issues.....		36
4.1.	Integration of Heterogeneous Home Networks.....	36
4.2.	AIAP-URC View in SUIT System.....	37
4.3.	OSGi View in SUIT System.....	40
4.4.	Abstract User Interface Description	41

4.4.1. Case Study: UPnP Bulb	43
4.5. Discovery Management	47
4.6. Synchronization and Notification	48
4.7. Bundle Management	49
Chapter 5 Implementation	51
5.1. OSGi Environment Construction	51
5.2. Bundle Implementation	52
5.3. SUIT Core Part Implementation	53
5.3.1. SUIT URC Engine	53
5.3.2. SUIT Controller	56
5.4. Home Appliance Bundle Development	56
5.5. Remote Console Bundle Development	59
Chapter 6 Conclusion.....	63
6.1. Conclusion	63
6.2. Comparisons	63
6.3. Future Works	65
Chapter 7 Bibliography	66



List of Figures

Figure 2-1 The Architecture of AIAP-URC	14
Figure 2-2 The Architecture of OSGi Service Platform	16
Figure 3-1 General Home Control Architecture	18
Figure 3-2 Give Appliances Costumes	19
Figure 3-3 Roles of SUIT Platform	21
Figure 3-4 Using MSN to control IR Robot	22
Figure 3-5 SUIT Platform Architecture	24
Figure 3-6 SUIT Core Part	25
Figure 3-7 SUIT Remote Console Part	27
Figure 3-8 MSN Remote Console	29
Figure 3-9 SUIT Home Network Part	31
Figure 3-10 UPnP Protocol Integration	32
Figure 3-11 IR transmitter for Traditional Appliances	33
Figure 3-12 User Interface Generation Procedure	34
Figure 3-13 Control Invocation Procedure	34
Figure 4-1 OSGi Execution Environment	37
Figure 4-2 AIAP-URC Architecture	38
Figure 4-3 OSGi Solution for AIAP URC Underlying Network	39
Figure 4-4 OSGi Adaptation with AIAP-URC	40
Figure 4-5 Home Appliance List	43
Figure 4-6 UPnP Bulb Required URC Documents	44
Figure 4-7 UPnP Bulb Target Description	44
Figure 4-8 UPnP Bulb Socket Description	45

Figure 4-9 UPnP Bulb Presentation Template..... 46

Figure 4-10 UPnP Bulb Resource Sheet..... 47

Figure 4-11 Synchronization and Notification 48

Figure 4-12 Bundle Installation with URL..... 50

Figure 5-1 Knopflerfish 2 - An OSGi R4 Implementation..... 52

Figure 5-2 RoboSapien IR Control codes 57

Figure 5-3 MSN Remote Console Screen Shot..... 61



List of Tables

Table 4-1 Elements in a Socket42

Table 6-1 Comparison with Other Systems64



List of Codes

Listing 5-1 BundleActicator	53
Listing 5-2 URC Engine Element Data Structure	54
Listing 5-3 Methods Provided By URC Engine	55
Listing 5-4 SUIT Controller - Action Invocation	56
Listing 5-5 Wrapping Robot Control Point as OSGi Bundle	58
Listing 5-6 MSN Robot Remote Console Code	60



Chapter 1: Introduction

1.1. Preface

Home Domination is how a man dominates the home environment. Nowadays, we have different remote controls or consoles for every kind of home appliances. In other words, we need specific remote control for specific home appliance. What is universal home domination? Universal home domination means, every remote control is capable of operating every home appliance, and also any kind of home appliance is operable by any kind of remote control.

Moreover, what makes it universal is the abstraction of user interface. With the user interface abstraction technology, home appliance manufacturer is able to design presentation suite for each appliance product. User interface suite is designed once and used on various remote controls or consoles.



In order to reach the goal of universal home domination, we designed “SUIT” platform (Standards-based User Interface Technology) which means suitable for every home appliance and remote console. We will introduce the platform as following.

1.2. Motivation

Home control is more and more important in IT industry since information appliance appears. Information appliance is the network capable appliance which is different from traditional appliance. Various standards or protocols for home control explode over the past years. It turns out that we are going live in a maze home environment for several years until the protocols are standardized, and traditional home appliances go out of use. How maze is our home environment? We have different kinds of traditional home appliances and

information appliances, and we also hold numerous types of remote controls and consoles in our hands to dominate home devices.

For Example, when we want to watch a DVD movie on the television, we have to access more than 3 remote controls for DVD player, TV and some other machines. The situation is quite not a good home environment for human, every remote consoles need to be taken cared. Once one of the remote controls is broken, people have to buy a new control or live without remote control inconveniently, you have to stand up, walk toward the appliance then press buttons for operations.

Second scenario is that some appliances even don't have remote control when you buy it. It is not cost effective for the light manufacturer to provide you a remote control just for a light. Even the remote control is available in the stock, people won't buy it just because they won't spend extra money for a remote control which is much expensive than the light it self. This is not a good business model for low price product manufacturers.

The last example, when tourists in some hotels find a remote control on the desk, they will feel uncomfortable to see it in languages they don't understand. The problem is obviously the user interface, one solution the hotel takes is to put several possible languages mapping labels on the remote control, but it is not the best solution.

It seems like we are living in a difficult home environment, so I design a universal home domination system which will be introduced in next sections and chapters. The architecture is a solution for user interface abstraction and integration of heterogeneous home network.

1.3. Objectives

To Design a universal home control system, there are a lot of issues to concern. I simply divide the system into 6 main objectives. The most important objective is to ease the

sufferings of both home appliance manufacturers and remote console manufacturers. Integration of heterogeneous home control network on the network layer, then add user interface abstraction on those integrated services for the presentation layer. Access control and dependency policy is required. The final is synchronization and notification since there should be different numbers and various kinds of remote consoles on the SUIT system.

1.3.1. Ease the pain for home appliance manufacturer

There are a numbers of network control platform available now. As those manufacturers need to decide which protocols or platforms to use, they are confused at this point because they can't enlarge the customer populations once they made wrong decision. For example, it doesn't make any sense to support multiple network protocols or standards for a cheap light bulb. The company will go into bankruptcy if the boss makes the very wrong and stupid decision. SUIT technology helps to face the dilemma.



1.3.2. Ease the pain for remote console manufacturer

SUIT provides the solutions for problems listed below.

a. Various network control platforms.

This point is just the same as in 1.3.1.

b. Specific user interface for specific appliance.

Remote console developers construct user interface for appliances one by one. However it will be a terrible nightmare when a big number of devices appear in a short period. Those programmers have to work day and night for this situation.

On the SUIT platform, developer's job is the design of pluggable presentation themes, the same idea with desktop themes in Windows XP. In Windows XP you can change themes yourself or even design it. Developers don't have to worry anything about context because specific user interface are automatically generated.

c. Expected to be standardized but versatile and custom-tailored.

Sometimes standardized and custom-tailored may be a conflict, you have to give up one of the requirement when you accommodate another. If there is a widely accepted alternative interface access protocol (AIAP) or user interface standard which is custom-tailored capable, then we reach the goal to be standardized and also versatile. SUIT platform follows some INCITS standards as the user interface abstraction description and reach the goal.

1.3.3. Heterogeneous Networks Integration

There are several home control network protocols in the world, including UPnP, Jini, X10, Insteon, ZigBee, Lonworks and so on. Besides those information appliances, what we can't ignore are traditional appliances, including TV, DVD player, Air conditioning...etc. These types of traditional home appliance has built-in infrared receiver, and of course, a remote control that transmits infrared.

On the SUIT platform, different home control networks are wrapped as standard bundles and can be recognized by each other, this is the integration of heterogeneous networks. With this property, the following design becomes quite easy.

1.3.4. User Interface Abstraction

As the heterogeneous network device services are standardized, you can find those home control services in the service repository. Only service is not so good enough, we add user interface abstraction on each of the home appliances. Those user interface descriptions can be thought as some suites for a home appliance, we dress them up on the SUIT platform.

With those suites, remote consoles are able to gather information about the home appliance, functions or operations on it, manuals of those operations, presentation hints for construction...etc. Then the remote console automatically generates the user interface with resources instructed in the documents, resources are labels, images, sounds and so on. The

advantage of constructing the user interface with resource indicated in the documents is flexible. You can design multi-language labels for an appliance, so that tourist can simply replace those documents and make the device friendly, which shows the language he speaks.

This objective is really important on the SUIT platform, developers or usual users can do a lot of applications on it, not just multi-language.

1.3.5. Access Control and Dependency Policy

Universal home domination is really helpful and powerful when you want to control some thing in your house. But universal power leads to disaster when you don't mean to control something, or someone you don't trust dominates your home. Access control achieves authentication and authorization. For authentication, users need to enter user id and password when they login the SUIT gateway. For authorization, during the process of generating user interface to people, the system will check user's profile then generates the proper user interface for him, for those operations the user is not allowed to do, he can't find the operation through the generated user interface.

Dependency policy defines execution conditions for certain operation. For example, when you want to reset a device, a notification pops up, and waits for you to press "OK" as an acknowledgement, the sequence of operations have to be defined by dependency policy. The purpose of this objective is to avoid SUIT gateway going into a maze state.

1.3.6. Synchronization

Information about current appliance status must be synchronized on each of remote consoles. For example, a family is watching TV together. In this scenario, every one in the family holds some types of universal remote console. Once the father switches the channel, status of the TV on each remote console should be changed, this is synchronization.

1.4. Organization

In Chapter 2, we introduce the background and technologies related to the SUIT platform. In Chapter 3, we present the system architecture of SUIT platform. In Chapter 4, we discuss the design issues of the platform, and the reasons for those designs. In Chapter 5, we talk about the implementation of the system and how both home appliance manufacturers and remote console manufacturers develop their product on SUIT platform. In Chapter 6, we make the conclusions and point out future works. In Chapter 7, we list references.



Chapter 2: Background

In this chapter, we will introduce the technologies of home appliance and remote console nowadays. After that, we are going to talk about AIAP URC standards and OSGi infrastructure. These two technologies are main spirit of SUIT home platform. At last, we introduce some related works and give summary of this chapter.

2.1. Home Appliances

In this section, we will introduce the existing home appliances technologies in our daily live, including both information appliances and traditional appliances. Eventually, you can figure out how a maze environment we are living, there are so many networks and protocols.

2.1.1. IR Technology



What is the most frequent action we do in living room? Maybe you will say watching TV. Yes, this is a good answer. But what do we need to do for watching TV? The answer is obviously to press the button on the TV remote control. TV is embedded with an IR receiver and the remote control is the IR transmitter. Once the IR transmits a correct IR code into the air, TV will act correctly as the remote control says.

The examples of same infrastructure are Air conditioning, surveillance for home or for car, rolling door...etc. Even a robot toy for child uses IR technology. So it seems IR technology is every where in our live.

2.1.2. Powerline Technology

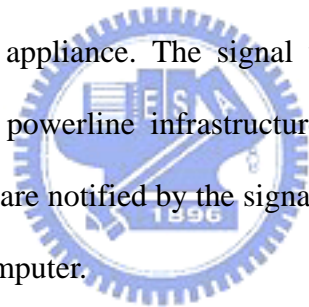
Powerline technology is the basic mechanism for home control, and has been used from nineteen centuries when electricity was invented. We put button on the wall for controlling

any kind of lights in our houses. Apparently, we don't expect to have any remote control or console according to our habits, what we need to do is, stand up, walk to the button, press the button and so on.

2.1.3. X10 and INSTEON Technology

For X0 home control scenario two components are required, they are X10 module and X10 control. Traditionally, we connect the power plug of an electronic device to the outlet some where in the house.

Now with X10 technology, instead of directly connecting straightly to the outlet, we connect it with X10 module and X10 module to outlet. At the control part, we have X10 control connected to computer. The computer is able to send signals through serial port or USB port for controlling the appliance. The signal will start from X10 control with the computer, then goes into the powerline infrastructure of the building. Finally some X10 modules with some appliances are notified by the signal, then the appliances are under control of the software hosts on the computer.



Actually X10 is an old technology which already exists for 30 years. It is the time for new technology to beat X10. INSTEON is an integrated dual-mesh network that combines wireless radio frequency (RF) with the powerline. INSTEON improves reliability by providing a backup system by wireless communication technology. One thing very different from X10 is that, each device acts as a two way repeater.

INSTEON is also compatible with X10 devices. Homeowners with existing X10 networks can migrate to an INSTEON network without throwing away the old X10 device. This must be a big inducement to people who have dreams to realize smart home.

2.1.4. Lonworks Technology

Lonworks is a home control protocol built on low bandwidth which is created by Echelon. The platform is for networking devices over media such as twisted pair, powerlines, fiber optics, and RF...etc. It is popular now very popular in the industry.

Thousands of companies nowadays have built products or applications on Lonworks platform, performing functions as embedded machine control, highway street lighting, heating and air conditioning systems, intelligent electricity metering, subway train control, stadium lighting and speaker control, security systems, fire detection and suppression, and newborn location monitoring and alarming.

2.1.5. Jini Network Technology



Jini is a network architecture for the construction of distributed systems where scale, rate of change and complexity of interactions within and between networks are extremely important and cannot be satisfactorily addressed by existing technologies. Jini technology provides a flexible infrastructure for delivering services in a network and for creating spontaneous interactions between clients that use these services regardless of their hardware or software implementations.

Jini network technology is an open architecture that enables developers to create network-centric services -- whether implemented in hardware or software -- that are highly adaptive to change. Jini technology can be used to build adaptive networks that are scalable, evolvable and flexible as typically required in dynamic computing environments. There are three main features of Jini technology, it supports Plug-and-Play mechanism for network service, provides toolkit for developers to build strong distributed systems and the architecture which is based on services. Roles in Jini network may have 3 protocols to follow:

discovery, join and lookup. For more details about related protocols, please reference to Jini.org, a central place and resource for the Jini Community.

2.1.6. UPnP Technology

The UPnP architecture created by Microsoft offers pervasive peer-to-peer network connectivity of PCs, intelligent appliances, and wireless devices. The UPnP architecture is a distributed, open networking architecture that uses TCP/IP and http to enable seamless proximity networking in addition to control and data transfer among networked devices in the home, office, and everywhere in between.

The most significant difference between UPnP and JINI is that instead of Lookup Service, every client communicates with the service provider directly. Additionally, UPnP technology uses services with remote procedure call, while clients on JINI platform execute services through proxy objects.

Start with Discovery phase, the service provider (device) broadcasts messages, which include the information of the services, such as device name, manufacturer, and etc, just like WSDL in web services. The Description is expressed in standard XML language. Another way for Discovery is that the client sends message for searching services and wait for service provider to response with information of the services.

The next step is controlling the UPnP device. Based on Description, control messages are used to execute actions. Control messages are also expressed in XML using the Simple Object Access Protocol (SOAP). In response to the control message, the service returns any action-specific values. These two steps form complete function calls.

The rest two UPnP features are Notification and Presentation. Notification is event-driven mechanism. Service provider publishes updates and control point subscribes the updates. For Presentation, if a device has a URL for presentation, then the control point can retrieve a page from this URL and present it to users.

2.2. Remote Consoles

What's the difference between remote control and remote console? Remote control is one-way mechanism. However remote console is two-way mechanism. With remote control, people can only operate appliance. With remote console, not only controlling appliance, people can query the status of the appliance or even they can be notified by appliance, this is what so-called two-way. In this chapter, we will introduce remote consoles nowadays.

2.2.1. Traditional Remote Control

A remote control is an electronic device used for the remote operation of a machine, like televisions or other consumer electronics such as stereo systems and DVD players. Remote controls for these devices are usually small handheld objects with an array of buttons for adjusting various settings such as television channel, track number, and volume. In fact, for the majority of modern devices with this kind of control, the remote contains all the function controls while the controlled device itself only has essential primary controls. Most of these remotes communicate to their devices via infrared (IR) signals and a few via radio signals.

2.2.2. Information Appliance Remote Console

Most of information remote consoles are software that hosts on certain device, performs proprietary networking protocol, alias name for that is device control point. The device control point may be independent from underlying network layer. The underlying network may be wired internet, wireless internet, 3G/GPRS cellular or Bluetooth...etc. For example, a "UPnP on Bluetooth", a "Jini on 802.11b", or a "Jini on Bluetooth" and so on.

Since the information appliance remote console is simply software, developers may write a program that performs the proprietary protocol for certain home control network, including sending packets and receiving packets.

2.2.3. 3G/GPRS/Bluetooth Mobile Device

Mobile device is potentially the most popular remote console in the future. Scientists and engineers put their efforts to figure out mobile services which tend to change human's life. What you have to do is to install control point software, and your mobile devices turn to become remote console of home appliances, isn't that great?

You can find some clues to prove that mobile devices must be remote consoles in the future. Nokia N series mobile devices support UPnP. The N series allows you to play music on your phone over your home UPnP-enabled HiFi system, download music and content from your PC, and display images and video from your phone on your UPnP TV. This is done over a Wi-Fi connection, using the industry standard UPnP protocols.

2.2.4. Instant Messaging

Using instant messaging mechanism as the remote console is our innovative idea. About the interaction service for a service-oriented infrastructure is always web pages, we just try to be different to see if it is better. The further discussion about using IM as the remote console will be in chapter 5, implementation portion. Here we just introduce what IM is.

Instant Messaging is the act of instantly communicating between two or more people over a network such as network. Instant Messaging requires the use of a client program that hooks up an instant messaging service and differs from e-mail in that conversations are then able to happen in realtime. Most services offer a presence information feature, indicating whether people on one's list of contacts are currently online and available to chat. This may be called a 'Buddy List'.

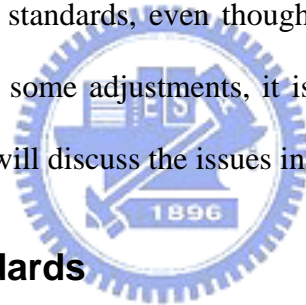
Popular instant messaging services on the public Internet include Qnext, MSN Messenger, AOL Instant Messenger, Yahoo! Messenger, Skype, Google Talk, .NET

Messenger Service, Jabber, QQ, iChat and ICQ. These services owe many ideas to an older (and still popular) online chat medium known as Internet Relay Chat (IRC).

2.3. User Interface Technology

Besides integration of heterogeneous home control network which refers to the interface between SUIT gateway and home appliances, the interface between human and SUIT gateway is also very important. Why is that so important on SUIT platform? The answer might be “we have different types and a large numbers of device remote console”.

If every remote console manufacturer always spends time to construct user interface as new appliances appear, it is really not cost effective. So we need standards for user interface. We choose AIAP URC as the standards, even though AIAP URC is not 100% suitable for SUIT platform. However with some adjustments, it is nice in SUIT platform, and it is still standards, not proprietary. We will discuss the issues in chapter 4, design issues.



2.3.1. AIAP-URC Standards

AIAP-URC is Alternative Interface Access Protocol for Universal Remote Console, the Development work conducted in V2 - Information Technology Access Interfaces, operating in accordance with INCITS (International Committee for Information Technology Standards). There are five standards in the URC standards set, below are some simple descriptions about those five standards:

- Universal Remote Console Standard (ANSI/INCITS 389)

This standard specifies the URC framework and technical components of the standard. The document is for those who is unfamiliar with the URC standard and its scope. It provides requirements for Targets, URCs and the network that connects them. There are four additional standards which complement the URC standard, yielding a set of 5 standards.

➤ User Interface Socket Description Standard (ANSI/INCITS 390)

This standard describes the User Interface Socket Description component of the URC Portal. A user interface socket description represents the functionality and state of a Target in a machine interpretable manner.

➤ Presentation Template Standard (ANSI/INCITS 391)

This standard specifies a language for providing a Presentation Template. A Presentation Template is a modality-independent scaffold for a user interface that provides the necessary structure and hints to build a user interface.

➤ Target Properties Sheet Standard (ANSI/INCITS 392)

This standard describes the Target Properties Sheet component of the Target. A Target Properties Sheet (TPS) is an XML document that provides the information needed by a URC to connect to the Target for a control session.

➤ Resource Description Standard (ANSI/INCITS 393)

This standard specifies resources and resource descriptions for user interface components that are used in building concrete user interfaces, including labels, help texts...etc.

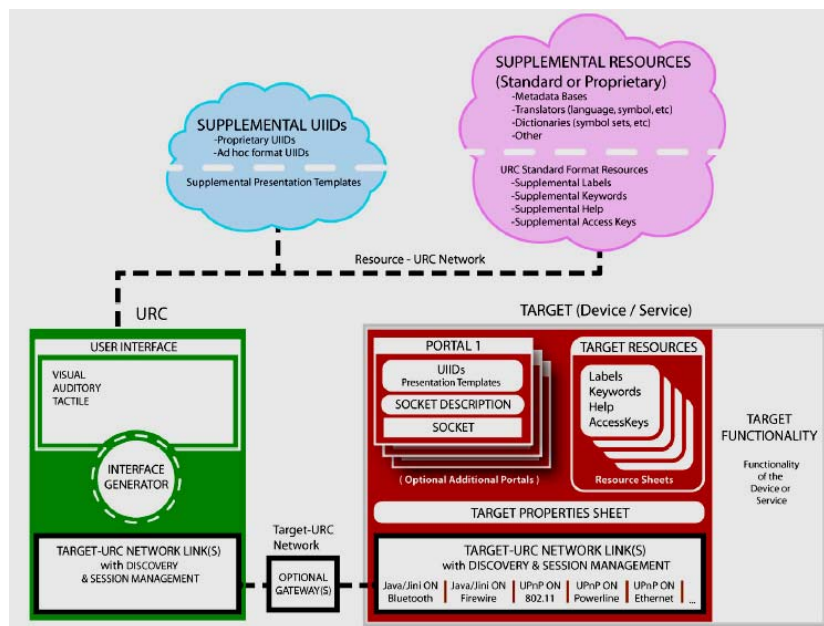


Figure 2-1 The Architecture of AIAP-URC

Figure 2-1 illustrates the architecture of AIAP-URC, This architecture can be thought of as comprising four major components and two networks.

- ✓ The Universal Remote Console (URC).
- ✓ The Target.
- ✓ Supplemental User Interface Implementation Descriptions (UIIDs).
- ✓ Supplemental Resources.
- ✧ Target-URC Network (TUN): between the URC and the Target (required).
- ✧ Resource-URC Network (RUN): between the URC and additional resources.

The AIAP-URC architecture expects the target devices and the URC devices are capable of those functionalities listed above. The interactions and communications between URCs and targets must follow what standards definitions. We are not going to introduce the details of AIAP-URC now, but the majority ideas are just mentioned. Later in chapter 4, Design Issues, we will discuss the role of AIAP-URC in SUI platform and comparisons of both two.

2.4. Service-oriented Infrastructure

For heterogeneous network integration, a service-oriented infrastructure is necessary. Unlike traditional point-to-point architectures, service-oriented infrastructure comprises loosely coupled, highly interoperable services. Every service exposes interface to service repository. Then others are able to get services from the service repository and use the service. Implementations are hidden behind, so the architecture is dynamic and flexible.

2.4.1. OSGi Service Platform

We chose OSGi as the service-oriented infrastructure of SUI platform. Not like J2EE which is large scale infrastructure and also proposed by SUN, OSGi is dynamic infrastructure that is suitable for home environment, compared with enterprise environment.

The OSGi Alliance (formerly known as the Open Services Gateway initiative) is an open

standards organization. Their mission is to specify, create, advance, and promote an open service platform for the delivery and management of multiple applications and services to all types of networked devices in home, vehicle, mobile and other environments.

Over the past few years it has specified a Java-based service platform that can be remotely managed. The core part of the specifications is a framework that defines an application life cycle model and a service registry. Based on this framework, a large number of OSGi Services have been defined: Log, Configuration management, Preferences, Http Service (runs servlets), XML parsing, Device Access, Package Admin, Permission Admin, Start Level, User Admin, IO Connector, Wire Admin, Jini, UPnP Exporter, Application Tracking, Signed Bundles, Declarative Services, Power Management, Device Management, Security Policies, Diagnostic/Monitoring and Framework Layering..

Started from 2000, R1 release. The OSGi standard is now on the stage of R4 Release, 2005. It is widely accepted by the IT industry.

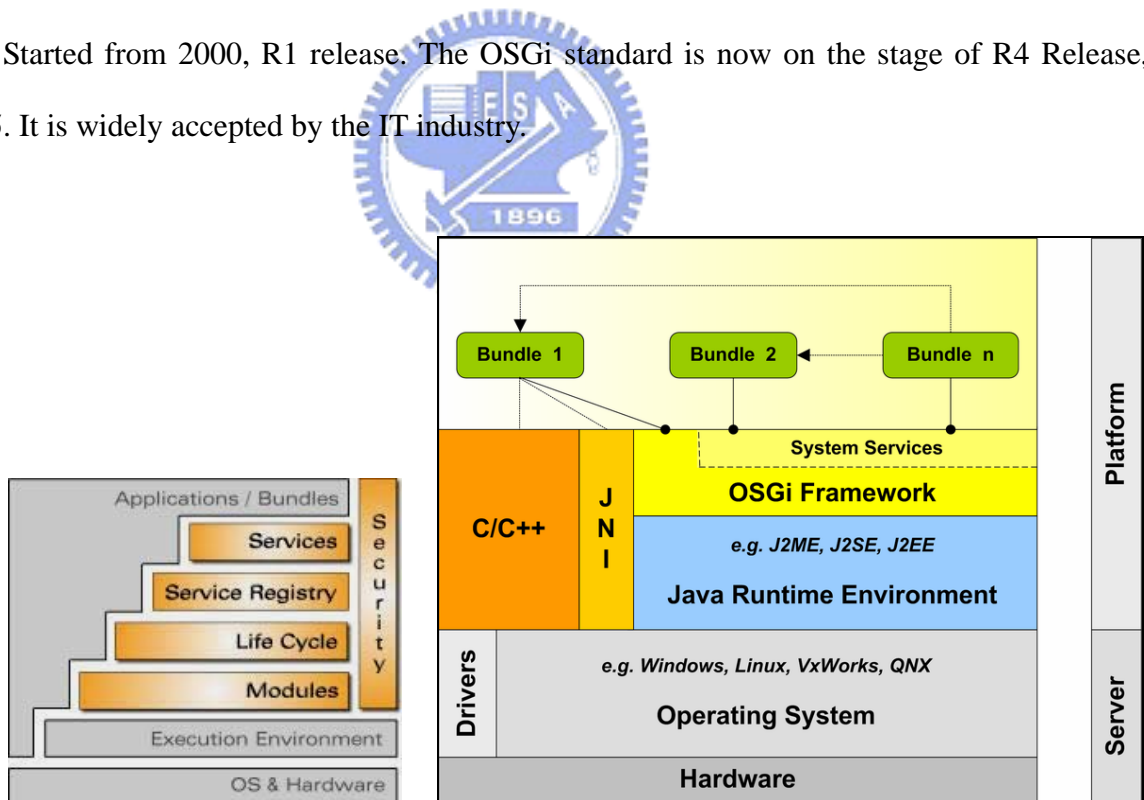


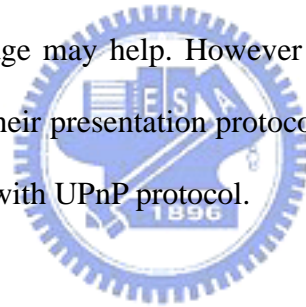
Figure 2-2 The Architecture of OSGi Service Platform

2.5. Related Works

Related works may be considered as two parts: Integration of heterogeneous network and abstraction user interface description.

There are so many researches about the integrations of home networks. Most solutions provide intermediate proxies to connect two different home networks. The solution is not so good because of the flexibility is very bad when new protocols come out. Service-oriented approach may be the direction of the research field.

About the appliances presentation to human, there are not so many researches in this field. Researchers put their efforts on the underlying protocol integration, but they develop user interface one by one, or automatically generated by their proprietary descriptions that are not standardized. XML language may help. However still there is not a XML language for home appliances. UPnP have their presentation protocols for UPnP control point applications, but the protocol must conduct with UPnP protocol.



Chapter 3: System Architecture

3.1. Overview

Before we talk about details of SUIT platform, we would like to point out some scenario. As Figure 3-1 shows, the most proper way to build a smart home is to place a home gateway as the head of the whole environment. Home appliances are connected wired or wireless to the home gateway, and they can be controlled by it, here the home gateway is like a commander. Remote consoles are connected to the home gateway too, so that they can ask the home gateway to do actions for them.

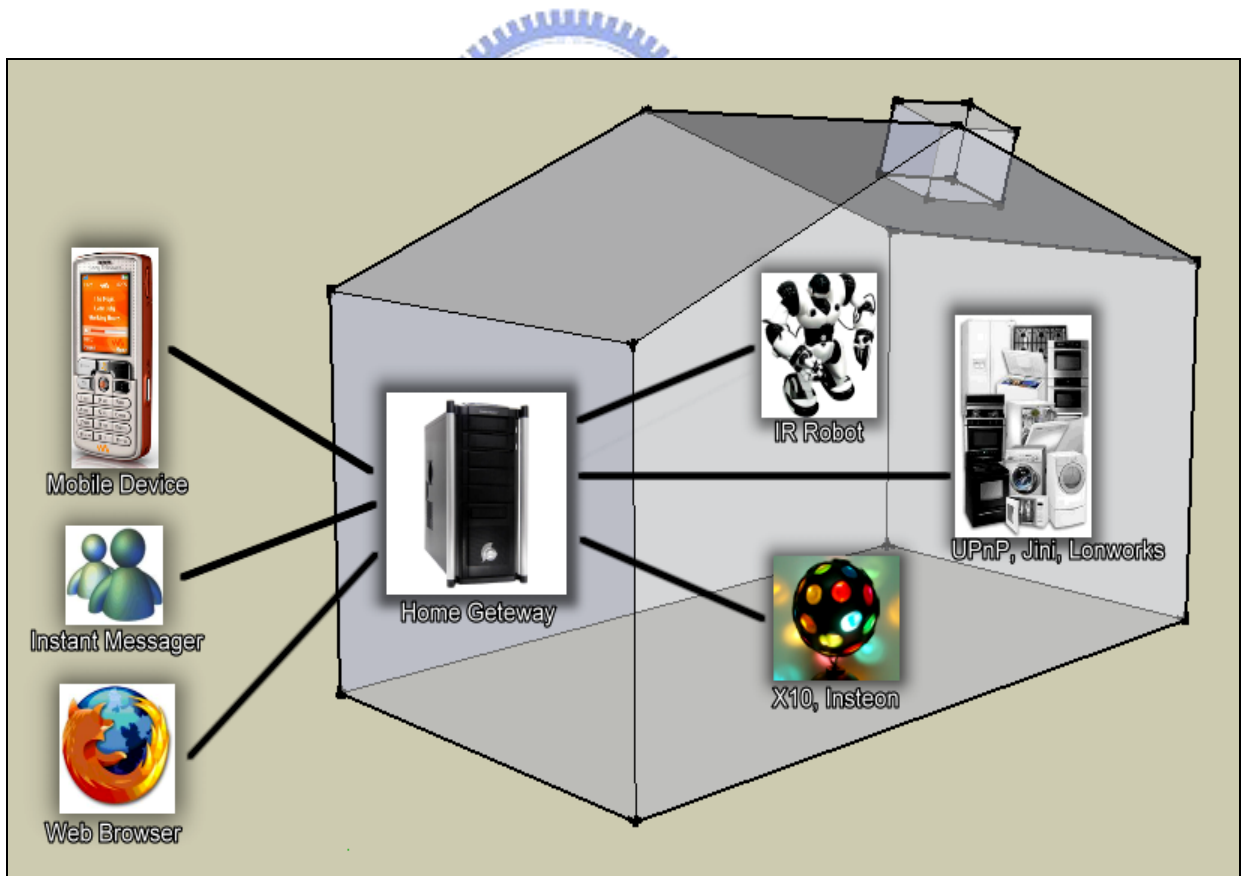


Figure 3-1 General Home Control Architecture

There are several home control protocols on the right hand sides of the architecture, those heterogeneous networks are integrated at home gateway. The left hand side shows several remote consoles which are able to control any home appliances on the system.

Home appliances can be dressed up with costumes, by manufacturers or anyone who knows the rule of producing the costumes, here the costumes or suits refer to some kind of descriptions of how appliances want to be seen by human beings. As the Figure 3-2 shows, we can give the IR Robot a cool jacket look, UPnP device a grace suit and cute cloth for X10 shining light. This functionality is like theme selection in windows XP, you are able to download or design by yourself, then replace it as you want.



Figure 3-2 Give Appliances Costumes

◆ Remote Console

Jenny uses web browser on her desktop computer to control the Hi-Fi in her room. Every action she takes which will change the status of the Hi-Fi device will also synchronously show the correct information on the remote console. This is console, bi-direction communication, not like remote controller which is only one-way communication.

◆ **Easy to Operate**

Mary is a house keeper who is responsible to manage every home device. With the SUIT platform, she doesn't have to learn anything about UPnP, Jini, X10, Lonworks...etc. All she needs to do is to pick up the remote console, ask SUIT gateway how many appliances are there available? Then she checks up operations for those appliances and does some actions according to the instructions given from SUIT gateway. It is so easy to operate through SUIT standardized interface, every thing looks just the same.

◆ **User Interface Theme**

John had received a secondhand coffee maker from his elder sister. The user interface of the coffee maker presented by his mobile device is quite women-like which John doesn't like it at all. He connected to the coffee maker manufacturer's web page through internet and downloaded other costumes. After installation, the user interface of the coffee make is totally different from the previous one. It is a man style presentation now.

◆ **Notification**

David and his father are watching TV together. His father holds a handheld device with SUIT remote console software on it, and David uses his MSN client as the remote console on his notebook. Once David switches channels, his father's handheld device will change the status information on the screen of the handheld device. When it is time for David to study, his father can have an eye on the TV or other entertainments which bother David from study well. Once any notification appears on the screen of the remote console held by his father, David will be in trouble soon.

◆ **Multiple Language Support**

Ichiro is a Japanese business man who is doing business now in Italy. At the first day he arrived the hotel, he turned his universal remote console on for the purpose to watch TV in the room. Unfortunately he found the language presented by the console is Italian which he doesn't speak. By supplemental resource services, he upgraded the Italian labels by Japanese

labels that are used to construct user interface for the TV. Now he is satisfied with the hotel.

3.2. Roles

In this section, we will list roles related to SUIT platform, Figure 3-3 shows the relationship of each role of the SUIT platform.

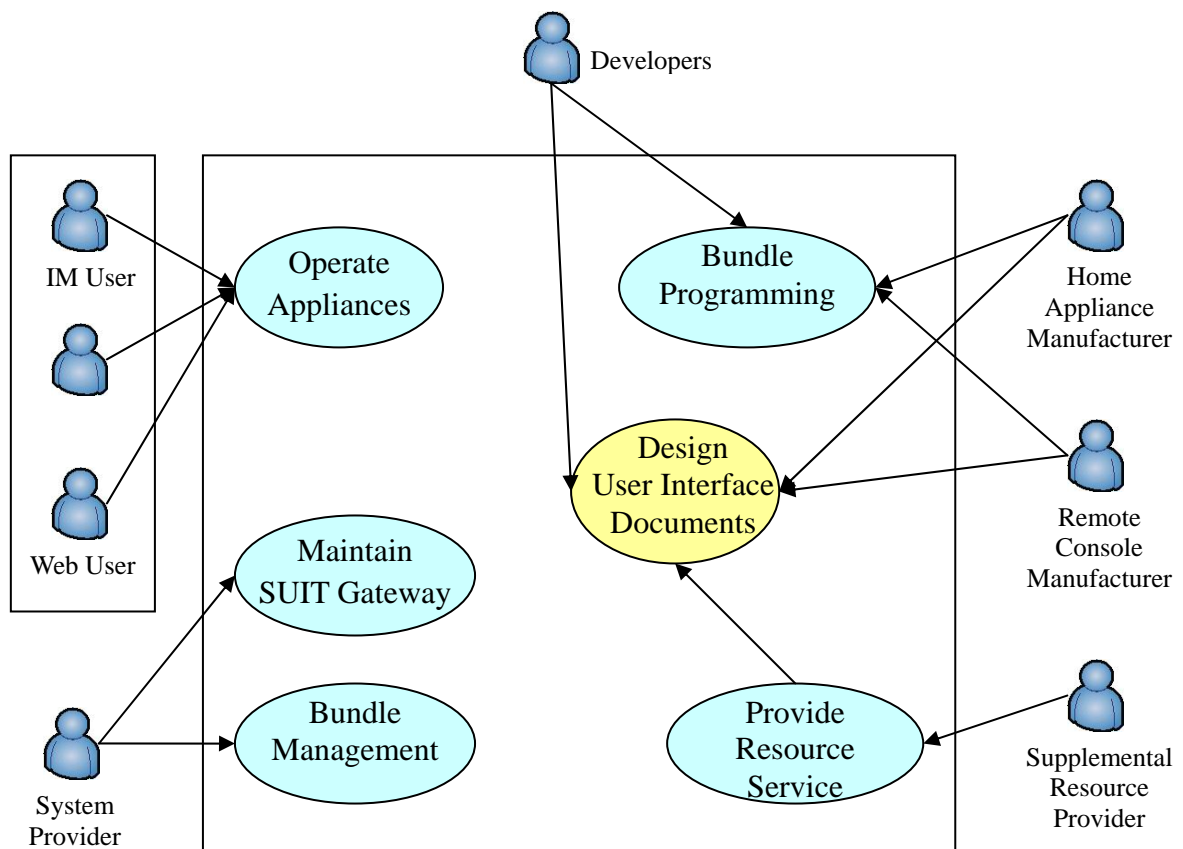


Figure 3-3 Roles of SUIT Platform

IM user, Mobile user and Web user are typical type of users in SUIT system. Human can operate home appliances through instant messaging interface, mobile device, web page or some other remote console technologies that can be easily deployed on SUIT platform. Besides controlling home devices, they can a design user interface for themselves, or

download user interface documents form internet and replace the old one.

Home Appliance Manufacturer is responsible to provide control point bundles for their products, and they have to follow with the standard bundle programming procedure defined by OSGi Alliance. Besides providing those control point software or drivers, the manufacturer needs to give a default AIAP-URC description document which is used to present the default user interface to human beings.

Remote Console Manufacturer is responsible to provide control point bundles for their products, and they have to follow with the standard bundle programming procedure defined by OSGi Alliance. For example, if I'm a MSN remote console manufacturer, then I need to write a MSN agent, wrap it into OSGi bundle and install it on SUIT gateway. So now we have a MSN agent hosts on the home gateway, anyone who adds the agent to his buddy list will have the ability to control the appliances around. Besides, they are able to provide AIAP-URC description document for certain device.

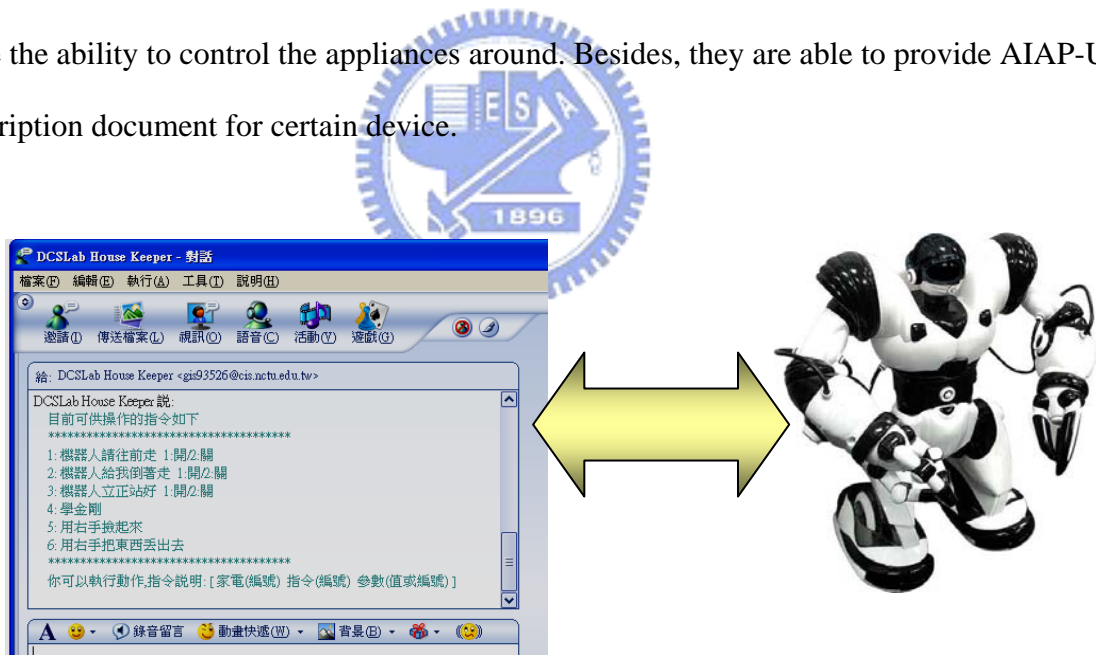


Figure 3-4 Using MSN to control IR Robot

Developers refer to those people who have computer science background and programming skill, but they don't belong to any part of appliance manufacturer or console manufacturer. They are allowed to develop bundles and AIAP-URC documents. For example,

as Figure 3-4 shows. I'm not an employee of IR Robot manufacturer, but I hope to integrate the IR Robot on SUIT gateway. I can develop a control point which sends correct IR code for Robot through an IR transmitter connected to SUIT gateway with console line. And also I write some action documents according to AIAP-URC definition. Besides that, I develop an MSN remote console bundle, so that I can control the robot with MSN.

System Provider have to setup and manage the whole SUIT environment, the majority job is taking good care of those bundles from both home appliance manufacturer and remote console manufacturer, including bundle installation, bundle upgrade and removing bundles.

Supplemental Resource Provider provides external resource services for SUIT platform. Resources include images, sounds, labels...etc. For example, we can get multi-language description packs from supplemental resource services.

3.3. SUIT Platform



SUIT means “Standards-based User Interface Technology” which is suitable for any kind of home appliances. The objective is to prepare an open service-oriented environment for both home appliance manufacturer and remote console manufacturer, and SUIT eases their sufferings of deciding which protocol to follow.

On the SUIT platform, every home appliance can be controlled by any remote console, and any remote console can dominate any home appliance. Each manufacturer only has to follow OSGi and AIAP-URC international standards and their proprietary protocols. They don't have to study other manufacturers' proprietary protocols, but others' product can be accessible on SUIT platform.

Next page is the overall architecture of SUIT platform. In this section, we separate the SUIT platform into three parts for discussion. The First is core part of SUIT platform, the second is remote console part (including console bundles) and the last is the home network part (including appliance bundles).

STANDARDS-BASED UI TECHNOLOGY
FOR UNIVERSAL HOME DOMINATION

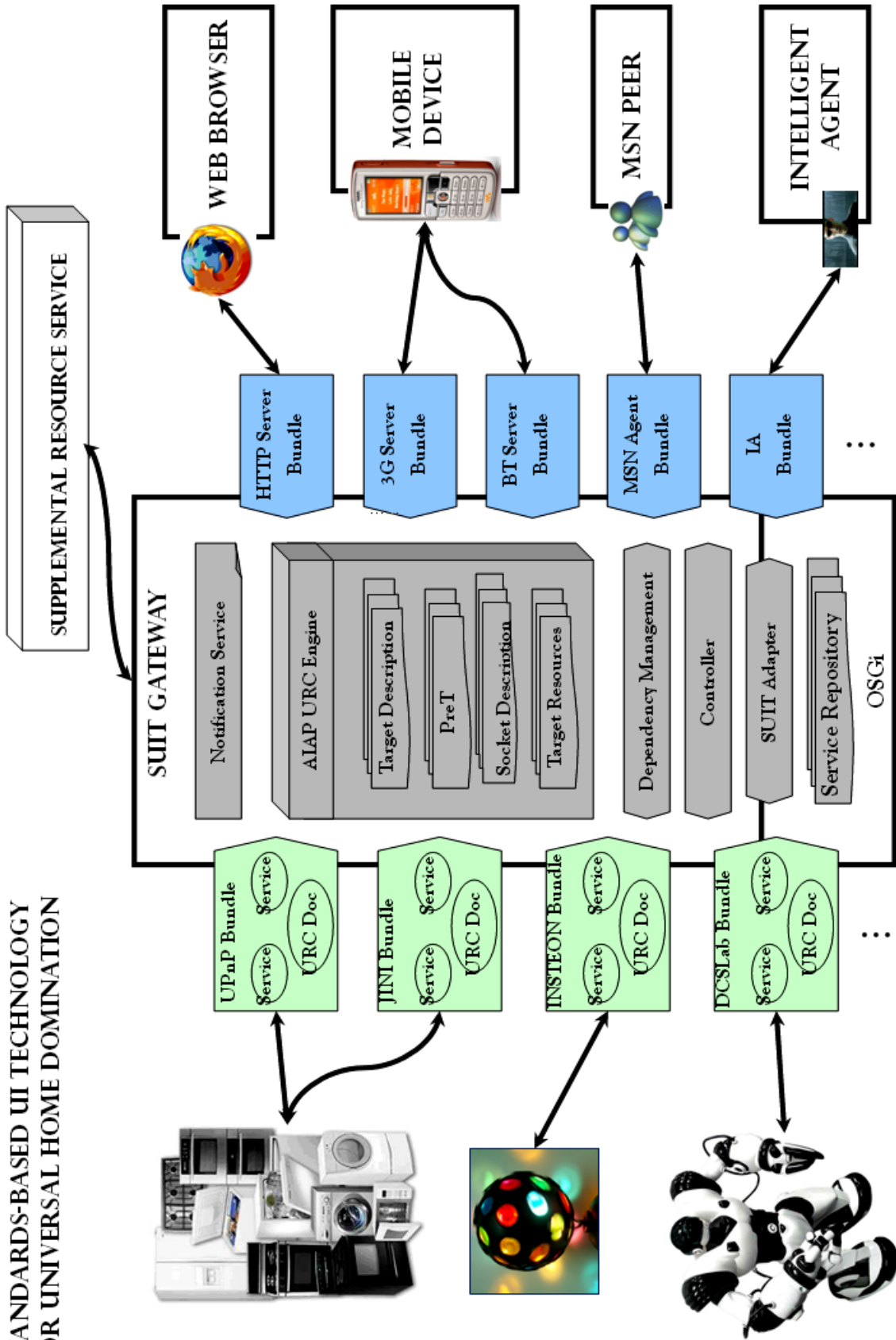


Figure 3-5 SUIT Platform Architecture

3.3.1. SUIT Core Part

Figure 3-6 shows the core portion of SUIT platform. In this section, we will introduce each component on the SUIT core part.

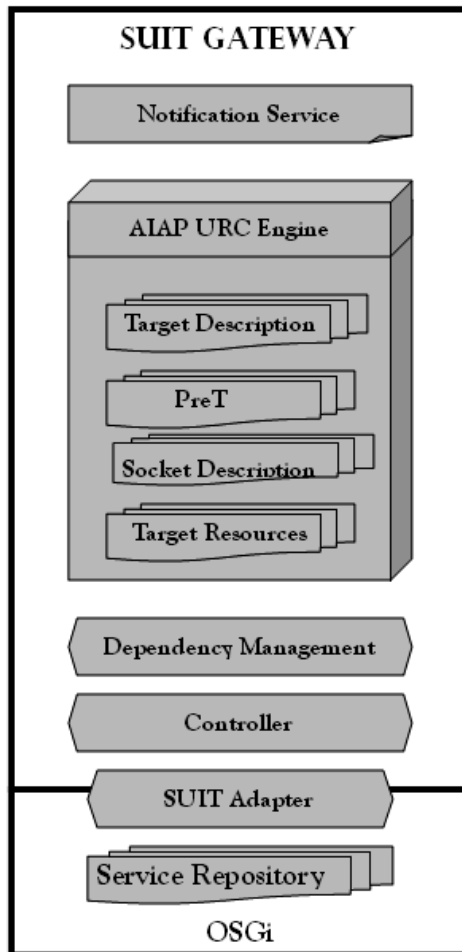


Figure 3-6 SUIT Core Part

Service repository is like a service pool that contains different services. Every installed bundle is able to register services into the pool. It creates objects and throws them into the service repository to accomplish the registration procedure. And also everyone on the gateway is able to use services provided by other bundles, it looks up services in the pool and picks up the one it wants then uses the service. Service repository is the base to reach the goal of service-oriented architecture, the approach to the universal home domination.

SUIT adapter is the bridge between SUIT platform and the underlying OSGi technology. It is responsible for the communication of both two. When there is any talk or invocation between two bundles, or say, one remote console and one home appliance, the action needs to go through SUIT adapter forward and backward. In other words, SUIT adapter is a bundle who is in charge of looking up all of related services from OSGi service repository and also responsible to picking up correct service objects for SUIT gateway.

Controller is actually the commander in SUIT platform. Requests are passed to the controller. Then controller will perform related actions in response, and rely to the party who sent the request. For example, when one of the consoles requests to turn off TV, the controller will pick up the TV service object through SUIT adapter from the service repository. Then the controller will check up the dependency relationship of the action which is going to be performed on the TV through dependency management. If the access is permitted, controller will invoke the corresponding method “Turn Off” in the “TV” service object. After the invocation, the object may return some results message or null, the controller takes care of the message and passes a proper result message to the remote console.

Notification Service needs to pass messages to every remote console when any appliance’s status information is changed. It is implemented by two kinds of standard message passing approaches. The first one is JMS Topic, every bundle that implements standard JMS client is able to be notified. We have JMS Topic for each appliance. The remote consoles may choose the Topic to subscribe and will be notified once any changes happen. The second method to implement notification service is OSGi service listener standard. Every bundle listens to specific services in the appliance bundle, once the services are changed, notification happen. Notification services help to send messages to dynamic and heterogeneous remote console bundles.

AIAP-URC engine is a preprocessor for AIAP-URC standards, the purpose of this engine is processing those standard documents provided by home appliance manufacturers or

others. After the processing of those URC documents, the URC engine provides two kinds of methods for remote console bundles to get those metadata. The first one is for those remote console bundle which not only follows OGSi standard but also AIAP-URC standards. The engine just passes those original documents as metadata to the bundle because the bundle is capable of AIAP-URC. It has own URC processing engine. The second way to pass metadata is by function calls. The remote console may ask for appliance list then URC engine returns the available appliances list. For action list and descriptions of specific home appliance, the URC engine replies the information to the bundle and so on.

3.3.2. SUIT Remote Console Part

The remote console manufacturer provides control point bundle in order to be a part of SUIT gateway. We introduce the SUIT remote console part in this section.

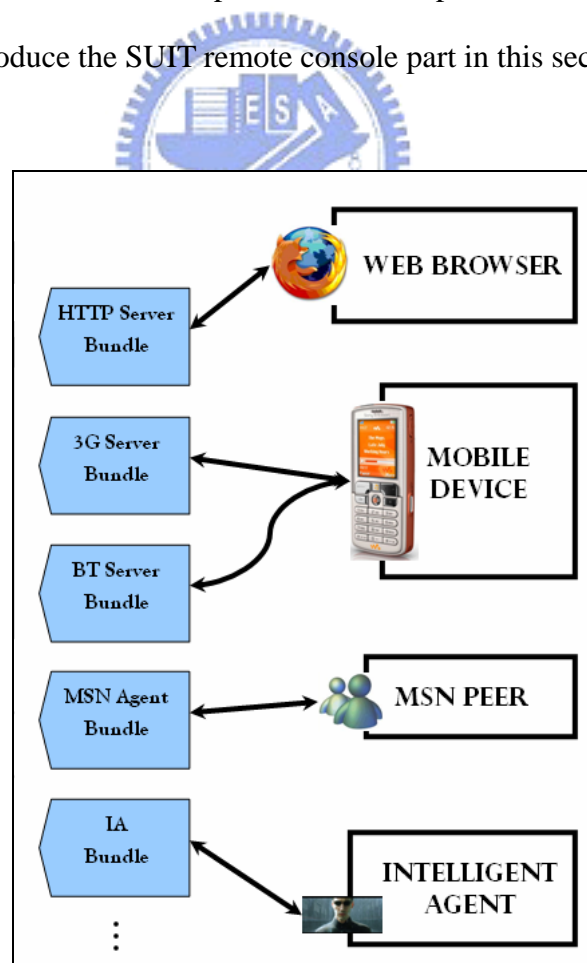


Figure 3-7 SUIT Remote Console Part

In the architecture of SUIT remote console part as shown in Figure 3-7, the protocols or communication mechanisms between the bundle and users are proprietary which is great for remote console manufacturer. After service transformation by constructing bundle software to the SUIT gateway, the proprietary portion becomes standardized on SUIT gateway.

The bundle will first get AIAP-URC documents or metadata from the system, process the data and create user interface for presentation. Then it will work with the controller in the SUIT core part to fulfill remote consoles request. Following we'll give some real cases.

The most general interaction service of a service-oriented infrastructure is web pages. If we want to construct a web remote console for our smart home, what does the bundle look like? The answer is pretty trivial. Http server is obviously the best choice. What we have to do is wrapping the http server in a bundle, and deploy the bundle successfully. Here the user is web browser, people operate the web pages from http server and they are able to operate every device connected to the SUIT system. About the user interface technology, the http server bundle retrieves the URC descriptions from SUIT core and generates the html source code for web client's request of web pages. This is how we make web pages as remote consoles.

The population of mobile device owners is growing day by day. Almost every owns one or more handsets, so this may be a good choice to hold remote console software. What does the bundle for mobile device looks like? It may be Bluetooth server or 3G/GPRS server which depends on what kind of network protocols the mobile device supports. Take Bluetooth for example, we have to place a Bluetooth dongle at the SUIT server side. The Bluetooth will get local address and publish the Bluetooth service as it is the server. As the Bluetooth client, mobile device searches devices then searches services on each device to see if there is SUIT universal home domination service. Finally the mobile device gets the connection from the service recode it finds and starts to be the remote console. For 3G/GPRS handset without Bluetooth module, the client is likely to use socket program that gets connection from the

socket server bundle through the cellular mobile network. About the user interface technology, the server bundle retrieves the URC descriptions from SUIT core and generates user interface components for mobile client such as Forms, Lists, CheckBox...etc. This is how we make mobile phone as remote consoles.

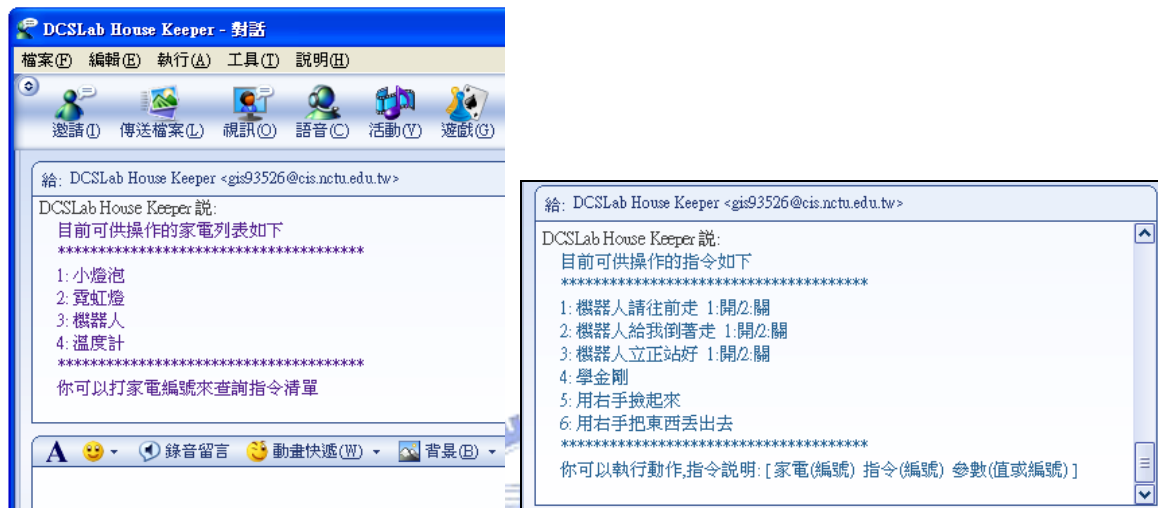


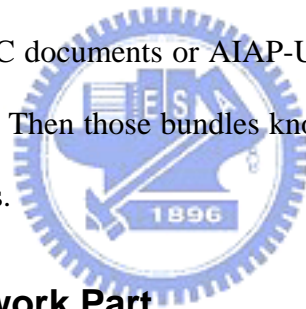
Figure 3-8 MSN Remote Console

Using instant messenger as a remote console is our brilliant innovative idea. IM remote console manufacturer develops an instant messenger robot as a bundle hosts on the SUIT gateway. The job of IM robot is reporting appliances status to other IM client, accepting requests from IM client and invoking the actions. Figure 3-8 displays an example of MSN remote console. We wrote a MSN robot names “DCSLab House Keeper”, wrapped it as a bundle, and installed it to SUIT gateway. I added the MSN robot to the buddy list of my personal MSN client which I used to use. After this procedure, I can always see the Robot on my buddy list. Then I am able to open a dialog and talk to the MSN robot on the SUIT gateway. On the right hand side of the screen capture as you see, when I ask for home appliance list, the Robot shows me the available and operable appliance list. IR robot is listed at the third place of the list MSN robot just gave me. If I want to know more details about the operations of the IR Robot, I make a request, and the MSN robot tells me the details as shown

on the left hand side of the screen capture, now I can easily operate the IR robot. It seems quite convenient for end users to use IM remote console with any extra installation if they already an IM client, all they need to do is to add some robots as their friends.

The last one to introduce is IA remote console, IA here means intelligent agent. Intelligent devices will be a big research issue in the future, and this may conduct into SUIT platform easily. Imagine that when you are driving, the device generates voice user interface to prevent you from not paying attention on driving. When you are in the meeting place which requires extreme silent then the intelligent agent generates visual user interface for you. All these features cab be down in out platform regarding the abstraction of user interface layer.

We can conclude this section that besides the integration of underlying home control, the user interface generation mechanism is more important in SUIT platform. Each remote console bundle gets AIAP-URC documents or AIAP-URC metadata from the URC engine in the core part of SUIT gateway. Then those bundles know how to present the user interface of the target appliance to the users.



3.3.3. SUIT Home Network Part

SUIT platform provides an open environment for home appliance manufacturer to develop bundles for their electronic products as Figure 3-9 shows. In this section we will introduce more details about this part, and will give some examples.

Protocols between home appliances and the control point bundles are proprietary to each individual manufacturer. The manufacturer doesn't need to worry if their proprietary protocols are compatible with other manufacturers' protocols because after installation of bundles, various protocols are integrated with standard interfaces and are registered to the service repository. Besides exposing the services, AIAP-URC documents are required as the default user interfaces for remote consoles. The documents include service method name, arguments, return values and also resources used to compose the user interface, and other necessary

information. In the next paragraphs, we will give two examples about the home appliance bundles for UPnP appliance and IR Robot. UPnP represents the future information appliance which is two-way network capable, and IR Robot is for one-way traditional home appliance. We are going to show how to integrate these two different types of home appliances.

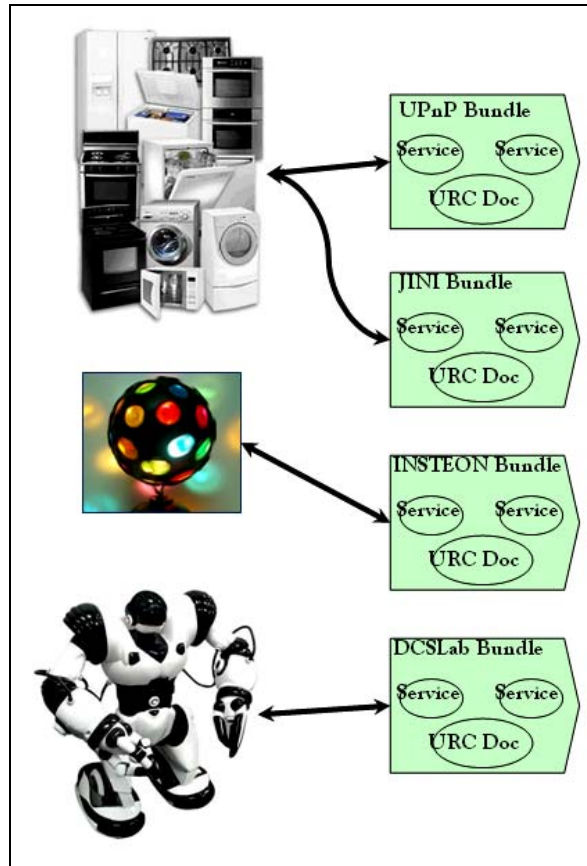


Figure 3-9 SUIT Home Network Part

For UPnP, each device broadcasts its information to the local area network. In order to receive the broadcasted information that is formed in XML descriptions, and take next actions, we have to have an UPnP control point software listening to the local are network, and the control point is also responsible for invoking the actions with parameters in the UPnP devices. In SUIT platform, the UPnP manufacturer develops the UPnP control point as a bundle, and exposes services to the service repository for remote console bundles to use. For the exposed

services in the service repository, the manufacturer writes AIAP-URC documents that describes the standardized services. Eventually, the UPnP environment is now transformed into SUIT platform, and working very well with it. Figure 3-10 illustrates the construction.

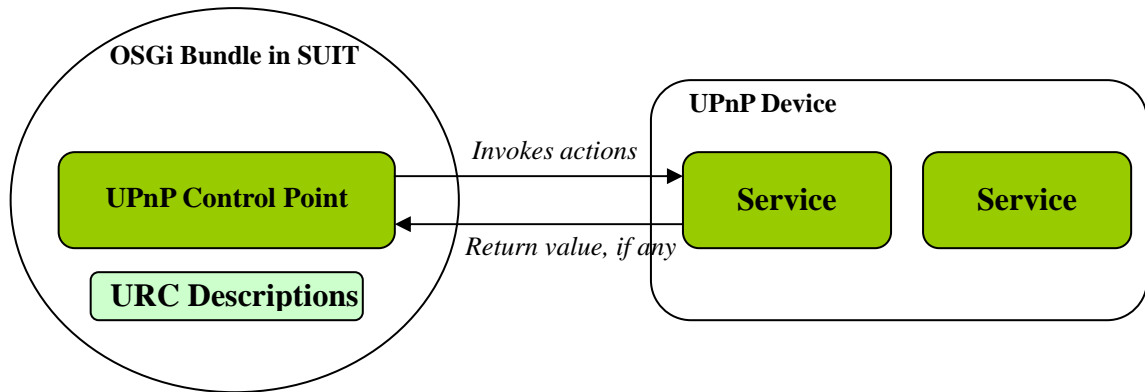


Figure 3-10 UPnP Protocol Integration

Traditional home appliances like TV, air conditioning are controlled by IR signal, and will not feedback any information because it is one-way communication. We take IR Robot as an interesting example to see how to integrate the IR Robot into SUIT platform. Actually it is just a toy that carries with an IR receiver, listening for IR signal in the air and then responds actions for correct IR code.

As Figure 3-11 shows, the first step to build the SUIT home appliance part is to find an IR transmitter like what kids hold in their hands. The IR transmitter is connected to the SUIT home gateway, and the IR Robot bundle transmits correct IR codes to the Robot through serial port. Sending the correct IR codes from the SUIT gateway is no different from pressing buttons on the kid's remote controller, because the IR Robot doesn't care who sends the IR message, however it cares if there is any action corresponds to the IR code for it to perform gorgeous movements. The IR Robot bundle wraps those action commands like functions or methods, and registers the service objects to the service repository. So that now remote

console bundles can see the services, get them and use them.

Besides developing IR Robot bundle, we also design AIAP-URC documents for the remote consoles. In Figure 3-8, you can see now we have opened 6 actions of IR Robot (Actually, it has 60 actions). With the URC descriptions, we present lively action instructions to the end users, instead of expecting them to input correct IR code directly. It is also very easy to support multi-language choice in this architecture.

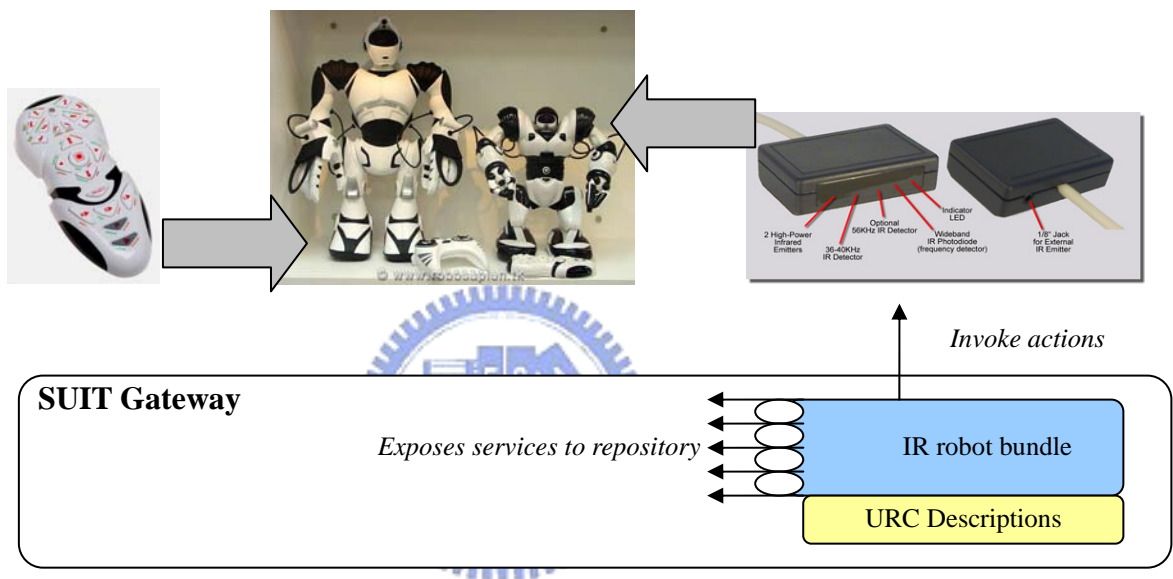


Figure 3-11 IR transmitter for Traditional Appliances

3.4. User Interface Generation

Figure 3-12 illustrates the user interface generation procedure for MSN remote console. First URC descriptions are installed with the home appliance bundles and can be accessed by the AIAP-URC Engine. When a MSN peer asks for user interface with device ID, the MSN bundle (agent) will ask the AIAP-URC Engine for the user interface through the exposed service interfaces, and process the user interface if necessary. Then the MSN bundle returns the target user interface in proprietary MSN format to the MSN peer who requests at the very

first step. Here the user interface for MSN peer is just an example of the procedure.

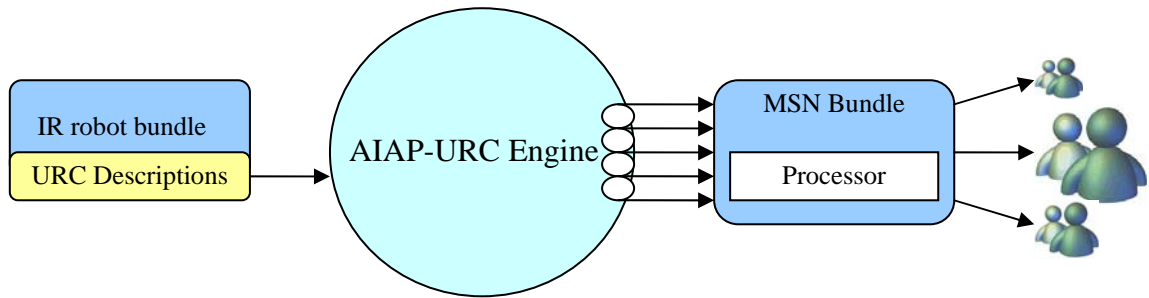


Figure 3-12 User Interface Generation Procedure

3.5. Control Invocation

Figure 3-13 shows the steps of control invocation procedure. At the prepare time, the installed bundle must register objects to the service repository, and the remote console gets the user interface as we mentioned in section 3-4. The MSN peer sends a control request to the MSN bundle, and the MSN bundle requests SUI core part. The SUI core part will lookup the object in the service repository and get the object back, then it performs the request. After IR robot bundle finishes its job, it acknowledges SUI core, and the SUI core replies the message the MSN bundle, and finally to the MSN peer.

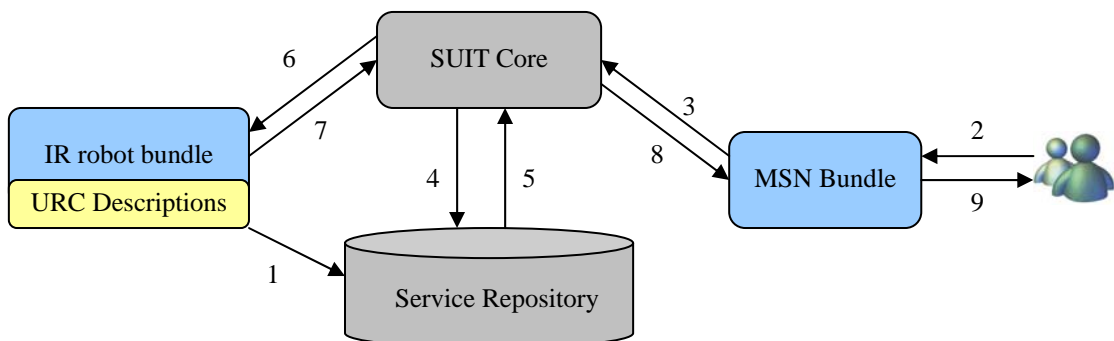


Figure 3-13 Control Invocation Procedure

3.6. Summary

In this chapter, we first gave an overview about SUIT platform, and then pointed out what kind of roles or users are related to the system. Then we talked about the system architecture, and separated it into three parts for more detail discussion. Three parts are SUIT core part, SUIT remote console part and SUIT home appliance part. We also discussed the user interface generation procedure and control invocation procedure. In next chapter, we will talk about design issues of SUIT platform.



Chapter 4 Design Issues

In this chapter, we will discuss about design issues of SUIT platform, why and how we decided to design it in this way, and also point out the advantages of the system. In section 4.1, we talk about the issues of integration of heterogeneous home control network. In section 4.2, we will make some discussion of adapting AIAP-URC standards into SUIT platform. In section 4.3 to 4.5, we discuss three important phases of SUIT platform for home appliances and remote consoles. They are discovery phase, control phase and notification phase. In section 4.6, the design of UCR engine in the core part of SUIT platform. In section 4.7, we talk about the management of control point software, and the last, section 4.8, the debate the issue about supplemental resource services.

4.1. Integration of Heterogeneous Home Networks

To integrate heterogeneous network, we have to find some service-oriented architecture as the base of our system. First we have three candidates in our pockets, J2EE、Web services and OSGi, finally we choose OSGi as the base of SUIT platform. In the following paragraph we will give discussions about this.

JMX、JCA、EJB and JNDI are JCP standards (Java Community Process), which belong to J2EE family. We can see each EJB as a service in the environment, in order to reach the concept of service-oriented infrastructure. But J2EE is born for enterprise usage which is built on large scale purpose. It is too big and expensive for a home environment, just like taking a musket to kill a butterfly. In stead of being built on large scale purpose, OSGi is born to be used on dynamic scale, which is more extensible and it is cheaper to have one. The extensibility property is suitable for SUIT platform because we always need to update bundle and documents in a short period. OSGi won the game with J2EE.

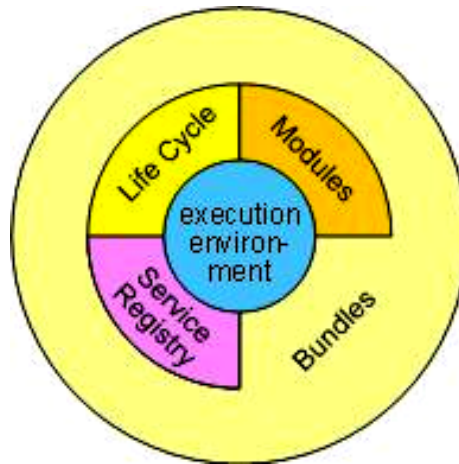


Figure 4-1 OSGi Execution Environment

Web services and WMX (Web Service Management Extension) are also considered at the design phase of SUIT platform. However after evaluation, we think it is also not so good for a home environment. The performance will be a big problem. It is used in the purpose of composing functions in different machine, and constructed on HTTP protocols. Performance is not its first consideration but to build a distributed system is. The problem here is that we just have one machine which we call it home gateway, so we don't have to pay the cost of performance but benefit nothing from it. OSGi won again.

4.2. AIAP-URC View in SUIT System

In this chapter, we will introduce the design of AIAP-URC adaptation to SUIT platform. We add some restrictions to the AIAP-URC standards instead of modifying them. So they are still standards, they are not proprietary after adaptation.

The biggest problem that needs us to provide solution is that, in the AIAP-URC standards architecture as shown in Figure 4-2. The target is required to be manufactured following the AIAP URC standard. For example, if you are UPnP appliance manufacturer, the proprietary network is UPnP on Ethernet, or on wireless. With the underlying network layer, you have an abstraction user interface layer that follows the AIAP-URC standards. If you are remote console manufacturer, you have to build AIAP-URC engine on the UPnP network

layer. This is the problem, not cost effective to both manufacturers. Another topic here is that AIAP-URC provides the universal solution for user interface, but we think real and complete universal home domination system must consider every related issue.

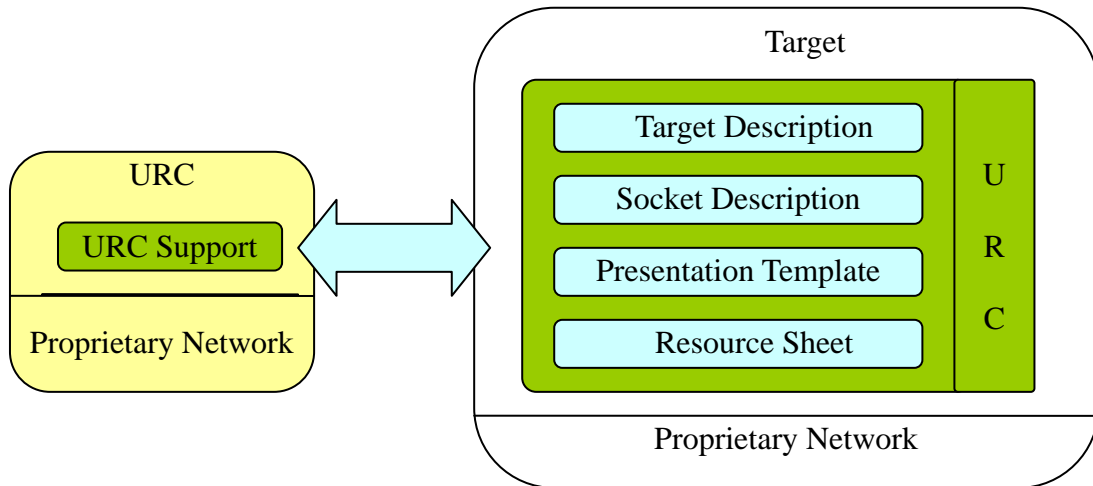


Figure 4-2 AIAP-URC Architecture

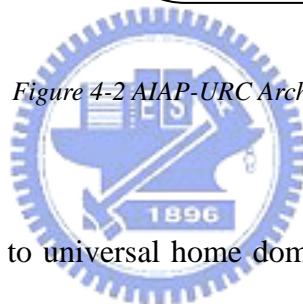


Figure 4-3 is our solution to universal home domination, which we would like to build OSGi concept to solve the problem about the proprietary network mentioned above. As section 4.1 says, OSGi is a great choice for the integration of heterogeneous network in a home environment. The solution is that both home appliance developers and remote console developers need to learn how to wrap their programs into OSGi bundles and also how to write AIAP-URC profiles.

We transform each proprietary network into OSGi services, so that different venders' networks are able to communicate with each other now. Between the bundle that is installed on the SUIT gateway and the existing home appliance, the communication is over their proprietary networks. For example, UPnP on 802.11 、Jini on Ethernet, etc. Between different venders' bundles, the communication is over OSGi services definition.

The AIAP-URC documents information are installed with the bundles and can be

changed or replaced by others. Every bundle should provide one set of URC documents for each target. The URC document set includes Target Description、User Interface Socket description、Presentation Template and Resource Sheet.

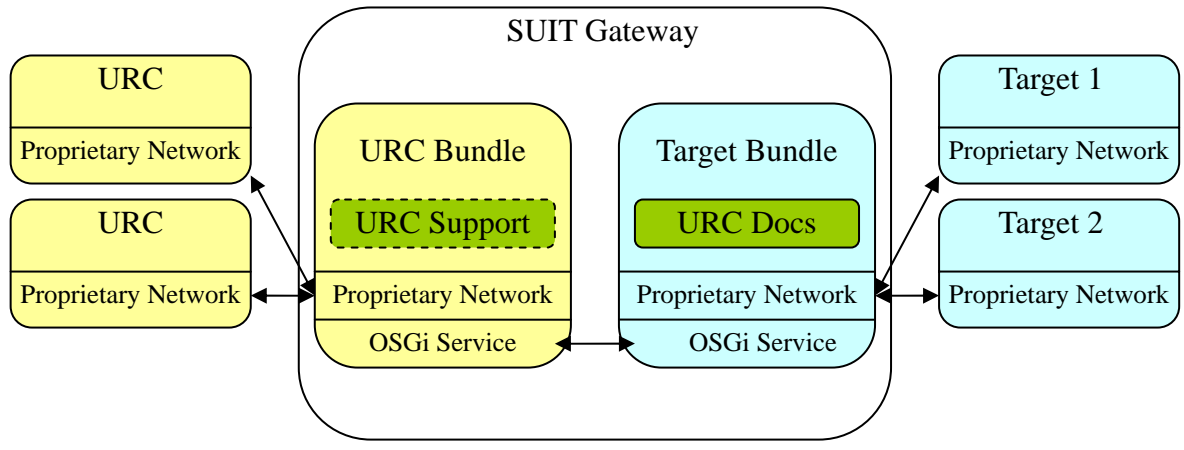


Figure 4-3 OSGi Solution for AIAP URC Underlying Network

The URC support is optional for remote console manufacturers. If the remote console bundle supports URC standards, then SUIT gateway is asked to pass those URC documents to the bundles and leaves the job to the bundles itself. If there is no URC support for the remote console bundle, which means the remote console manufacturer doesn't even know what the AIAP-URC standards are. Then SUIT gateway will read URC documents for the remote console bundle. The SUIT URC engine will parse and organize those documents, and expose methods for remote console bundles to call. When developers are working for bundles, they may read developers' guide for useful methods that are provided by the URC engine. For example, the remote console bundles can ask for all available device list、available device list by certain proprietary network、actions for specific device, etc. Those methods are contained in an object which registered to the service repository by the SUIT system, and remote console bundles can pick up objects from the service repository and use the methods as just mentioned which are carried with them.

4.3. OSGi View in SUIT System

Using services on the OSGi platform is not very difficult. We just have to look up the services in the service repository, generate the objects and use them. This is a great contribution from OSGi. How about the world after integration? Some developers design user interfaces for those integrated services one by one, and some just design their own presentation languages for user interface auto generation. It is a big pity to construct user interface in non-standardized way after the standardized services.

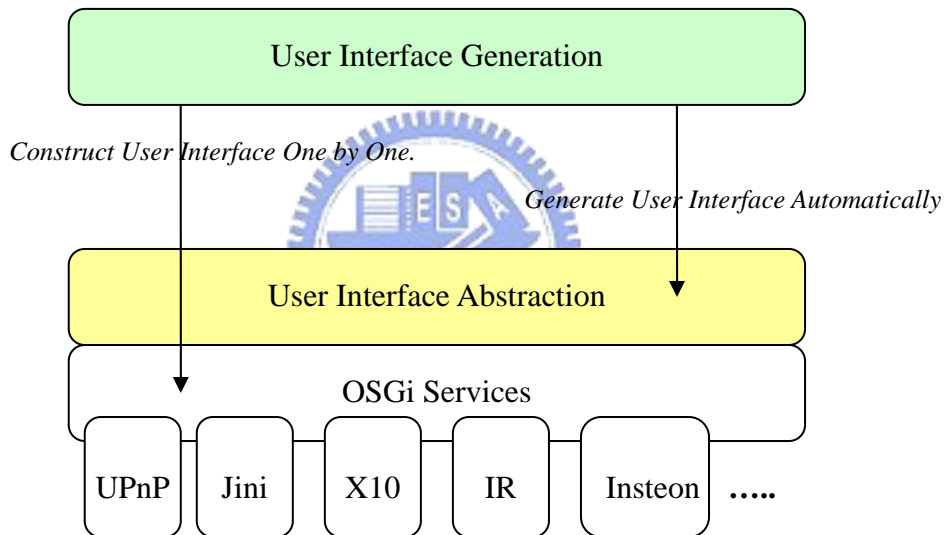


Figure 4-4 OSGi Adaptation with AIAP-URC

AIAP-URC is basically user interface protocol by INCITS V2 group which is very possibly becoming the international standards in the future. The latest drafts of the ISO/IEC standard on a universal remote console have been recently distributed to the members of the pertaining subcommittee, ISO/IEC JTC1 SC35 user interfaces, for final committee draft voting. If successful, this vote (closing in August 2006), will likely be the last vote on the subcommittee level. At that point, the draft standard will be passed on to JTC1 for final voting

and publication as an ISO/IEC International Standard.

We choose this future star to construct our abstraction user interface layer. People are able to write descriptions for integrated services, and remote consoles on the SUI system will be aware of the user interface information, which mean they will generate user interface automatically. Themes for the appliances can be downloaded from somewhere and replace the old ones to keep them in fashion shapes.

Now the SUI platform is not only universal for machine, it is also universal for human. We think real universal home domination is both for machine and human beings.

4.4. Abstract User Interface Description

Before we discuss abstraction user interface description, now first we would like to review overall documents concept about AIAP-URC defined in ANSI/INCITS 389、ANSI/INCITS 390、ANSI/INCITS 391、ANSI/INCITS 392 and ANSI/INCITS 393. There are four kinds of documents in the AIAP-URC, Target Description、Socket Description、Presentation Template and Resource Description. After that, we will talk about the design of URC documents in SUI platform.

Target Description provides information on a Target that is necessary for access to the Target and its Portals (Sockets). Each target shall provide exactly which is an XML document. It provides all information and reference required by a URC in the Discovery Phase and allows URC to discover, identify and understand a Target and its Portal(s). Finally, it contains references to XML documents, such as a User Interface Socket Description, Resource Sheet Collections and User Interface Implementation Descriptions.

User Interface Socket Description is an abstract concept that exposes the functionality and state of a target in a machine-interpretable manner. Every Portal shall contain exactly one Socket. A Socket contains statics, variables, commands and notifications as following. These four elements are all optional which depends on the requirements.

➤ Statics : represent fixed or constant information. Example of statics is the model and serial number of a digital projector.
➤ Variables : all of the dynamic data. Example of variables include the current channel showing on a television set
➤ Command : function that cannot be represented by a variable. Example of command is the seek button on a CD player.
➤ Notifications : special states where normal operation is suspended. Examples of notifications include a clock alarm, or a response to invalid input for a field of a form.

Table 4-1 Elements in a Socket

Presentation Template is modality-independent presentation information for a user interface socket description Presentation Templates associated with a User Interface Socket Description. The purpose of a Presentation Template provides hints for building a usable and consistent user interface for a Target device or service that is described in a User Interface Socket Description.

A Resource Sheet is a file that contains Resource Descriptions. A Resource Description specifies properties of a Resource. Properties include its type, its use context (the usage location, usage role, and language context), and the Resource’s storage location and format. A Resource is defined as an identifiable object of a user interface that is used as an entity in the construction of a concrete user interface.

We have added more conditions to the URC standards in order to compromise with OSGi. The major difference is that we are describing a control point service bundle, instead of a unique device. In next section, we will introduce a real case about abstraction user interface on an UPnP control point bundle, and an UPnP Bulb device.

4.4.1. Case Study: UPnP Bulb

In the SUIT platform, we manage those URC standard profiles from manufacturers' bundles in the file system. Every directory represents a document set for a home appliance. The naming convention of the directory is "Manufacturer-DeviceName" as shown in Figure 4-5. Now we have four appliances on our SUIT gateway, they are thermometer produced by AIAP Manufacturer、IR Robot produced by DCSLab Manufacturer、bulb produced by UPnP manufacturer and neon produced by X10 manufacturer. Following we will look into UPnP bulb and show the design of AIAP-URC document set for this electronic product on SUIT platform and the usages of those documents.

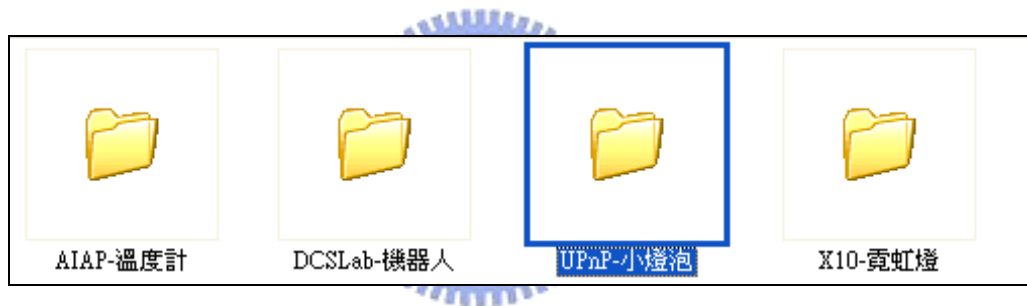


Figure 4-5 Home Appliance List

Inside the directory of UPnP bulb, there are four URC documents for this device. The naming convention is "Manufacturer-DeviceName.td.xml" for target description which informs the other three documents. "Manufacturer-DeviceName.uisocket.xml" is for User Interface Socket Description which indicates the functionalities enabled on the appliance. "Manufacturer-DeviceName.pret.xml" is for Presentation Template that gives hints for remote consoles to build user interface. Finally "Manufacturer-DeviceName.rsheets.xml" is for Resource Sheet that remote consoles use to construct the interactive user interface. Another way to name these documents besides Target Description references them in the Target

Description, in this case, you can name them freely. These four documents should be maintained very well by the SUIT gateway.



Figure 4-6 UPnP Bulb Required URN Documents

Figure 4-7 is the example of UPnP bulb Target Description, we retrieve three important parts in this document. The first part tells the name of User Interface Socket Description. The second part points out the Resource Sheet, and here in this case, the resource is of the Text type, it maybe image type or audio type for other appliances. The last part says the name of the Presentation Template which is used by the UPnP bulb.

```

<socket id="socket" name="http://mick.no-ip.org/UPnP-小燈泡/socket">
<socketDescriptionLocalAt>socket/UPnP-小燈泡.uisocket.xml</socketDescriptionLocalAt>
<category taxonomyName="http://www.unspsc.org">41.11.22.00</category></socket>

<resourceSheet>
<ResourceSheet rdf:about="http://mick.no-ip.org/UPnP-小燈泡/UPnP-小燈泡.rsheet.xml">
<dcterms:conformsTo rdf:resource="http://www.incits.org/incits393-2005"/>
<localAt>socket/UPnP-小燈泡.rsheet.xml</localAt>
<containsType>Text</containsType>
<containsDomain rdf:resource="http://mick.no-ip.org/UPnP-小燈泡/socket"/>
<containsDomain rdf:resource="http://mick.no-ip.org/UPnP-小燈泡/cpt"/>
<containsRole rdf:resource="http://www.incits.org/incits393-2005#label"/>
<containsLanguageContext>cht</containsLanguageContext>
</ResourceSheet></resourceSheet>

<uiidDescription>
<UiidDescription rdf:about="http://mick.no-ip.org/UPnP-小燈泡/UPnP-小燈泡.corepret.xml">
<dcterms:conformsTo rdf:resource="http://www.incits.org/incits391-2005"/>
<coreForSocket rdf:resource="http://mick.no-ip.org/UPnP-小燈泡/socket"/>
<localAt>socket/UPnP-小燈泡.pret.xml</localAt>
<dc:publisher>mick.no-ip.org</dc:publisher> </UiidDescription> </uiidDescription>

```

Figure 4-7 UPnP Bulb Target Description

Figure 4-8 presents the real case for User Interface Socket Description for UPnP bulb. Although elements of a socket are all optional which depends on the functionalities provided target device, SUIT requires each Socket Description to provide a “deviceID” of static type. The reason for that is we are writing descriptions for control point bundle which is able to control multiple products in the same proprietary network. It is not the situation in the typical AIAP-URC architecture which the description is for unique device. In SUIT platform, manufacturers have to specify the device identifier, for control point to recognize which device is the real target. “it.cnr.isti.niche.osgi.sample.light.LightDevice” is the UPnP device identifier that is recognizable for UPnP control point in over UPnP network. “CS703A” is the model number for the UPnP bulb that is unchangeable, so we choose the static type. The UPnP bulb is enabled with a function named “GetStatus” for remote consoles to query the current status of the UPnP bulb. The “GetStatus” action is not representable of a variable, it is a action which will be done after several steps, so we use command type. The last element is “SetTarget”, which remote consoles are able to set it true or false. The dependency for “SetTarget” action is if other remote console is also trying to change the state of variable.

```
<uiSocket about="http://mick.no-ip.org/UPnP-小燈泡/socket"
id="socket" xmlns="http://www.incits.org/incits390-2005"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">

<static id="deviceID" type="xsd:string">
it.cnr.isti.niche.osgi.sample.light.LightDevice
</static>

<static id="modelName" type="xsd:string">
CS703A
</static>

<command id="GetStatus" />

<variable id="SetTarget" type="xsd:boolean">
<dependency write="false()"/>
</variable>

</uiSocket>
```

Figure 4-8 UPnP Bulb Socket Description

Figure 4-9 is the example of UPnP bulb Presentation Template. Presentation gives hints for universal remote consoles to build universal user interface. In this case, we output the model number which is static type to the user, the URI “http://mick.no-ip.org/UPnP-小燈泡/socket#modelName” is the mapping of presentation element to the socket element, here is the “modelName” element in Figure 4-8. For the command action “GetStatus” in Figure 4-8, we have the mapping element with “trigger” type as the hint to the remote console. For “SetStatus” function, the Presentation hint is “input” type for remote console to set status.

```
<pret name="http://mick.no-ip.org/UPnP-小燈泡/corepret" id="pret" xmlns="http://www.incits.org/incits391-2005">
<output id="modelName" ref="http://mick.no-ip.org/UPnP-小燈泡/socket#modelName"/>
<trigger id="GetStatus" ref="http://mick.no-ip.org/UPnP-小燈泡/socket#GetStatus"/>
<input id="SetStatus" ref="http://mick.no-ip.org/UPnP-小燈泡/socket#SetStatus"/>
</pret>
```

Figure 4-9 UPnP Bulb Presentation Template

The last document is Resource Sheet, used to construct concrete user interface. According to the Presentation Template, there are two functions which need supports from Resource Sheet to help them complete the concrete user interface construction. In this example, we give two Text type resources for remote consoles. These two Text are in Chinese, “看看現在燈的狀況” is for presenting “GetStatus” to Chinese people, and the Text type resource “控制小燈泡的開關” is for “SetTarget”. We can do multi-language support by substitution of Resource Sheet.

One worth discussing is that the AIAP-URC specifications do not define the presentation of arguments for functions. For example, when we want to open the UPnP bulb, we try “SetTarget On”, and turning it off by “SetTarget Off”. But in the Socket Description, the

machine reads Boolean type, not “On” or “Off”. So the mapping job is thrown to the remote console bundle, the remote console manufacturer must pay attention to this. Another possible design is to use “select1” instead of input. The remote console has to provide a user interface structure that only allows users to choose one option from a set of choices, and the set of choices can be defined with <selection> tag. More detail information about the Presentation Template can be found at the specification of INCITS 391: Presentation Template.

```

<aResourceDescription>
<AResourceDescription rdf:about="http://mick.no-ip.org/UPnP-小燈泡/UPnP-小燈泡.rsheet.xml#GetStatus_label">
<content rdf:parseType="Literal"> <span xml:lang="en">看看現在燈的狀況</span> </content>
<dcterms:conformsTo rdf:resource="http://www.incits.org/incits393-2005" />
<useContext><UseContext> <elementRef rdf:resource="http://mick.no-ip.org/UPnP-小燈泡/socket#GetStatus" />
<role rdf:resource="http://www.incits.org/incits393-2005#label" />
<languageContext>cht</languageContext>
</UseContext></useContext>
<dc:publisher>mick.no-ip.org</dc:publisher><dc:type>Text</dc:type></AResourceDescription></aResourceDescription>

<aResourceDescription>
<AResourceDescription rdf:about="http://mick.no-ip.org/UPnP-小燈泡/UPnP-小燈泡.rsheet.xml#SetTarget_label">
<content rdf:parseType="Literal"> <span xml:lang="en">控制小燈泡的開關</span> </content>
<dcterms:conformsTo rdf:resource="http://www.incits.org/incits393-2005" />
<useContext><UseContext> <elementRef rdf:resource="http://mick.no-ip.org/UPnP-小燈泡/socket#SetTarget" />
<role rdf:resource="http://www.incits.org/incits393-2005#label" />
<languageContext>cht</languageContext></UseContext></useContext>
<dc:publisher>mick.no-ip.org</dc:publisher><dc:type>Text</dc:type></AResourceDescription> </aResourceDescription>

```

Figure 4-10 UPnP Bulb Resource Sheet

4.5. Discovery Management

Discovery management is in charge of reporting the available appliance list to the remote consoles. We separate this discovery issue into two categories, information appliances and traditional appliances. We would like to discuss it in next two paragraphs.

For information appliances, there exists some mechanism for discovery management. The information appliance is two-way network capable, so that it is able to report some messages to show the availability which is quite helpful for SUIT platform. Once a remote

console requests the appliance list, the SUIT gateway asks each control point bundles for lists, and finally merges them into one list.

For traditional appliances, SUIT platform can only guarantee that the control point bundle sends the signal, but it can't assure the traditional appliance receives the signal, since the traditional appliance is in one-way network. For example, we have a control point bundle that drives an IR transmitter to send IR codes into the air, but the bundle doesn't guarantee that the IR robot will receive the IR code and act as the user wants.

To merge these two lists, we will have annotations after each traditional home appliance that we don't guarantee the control to this device. About the information appliance list, the SUIT system will always track the status with the acknowledgments from the device, and keeps a fresh list for remote consoles.

4.6. Synchronization and Notification

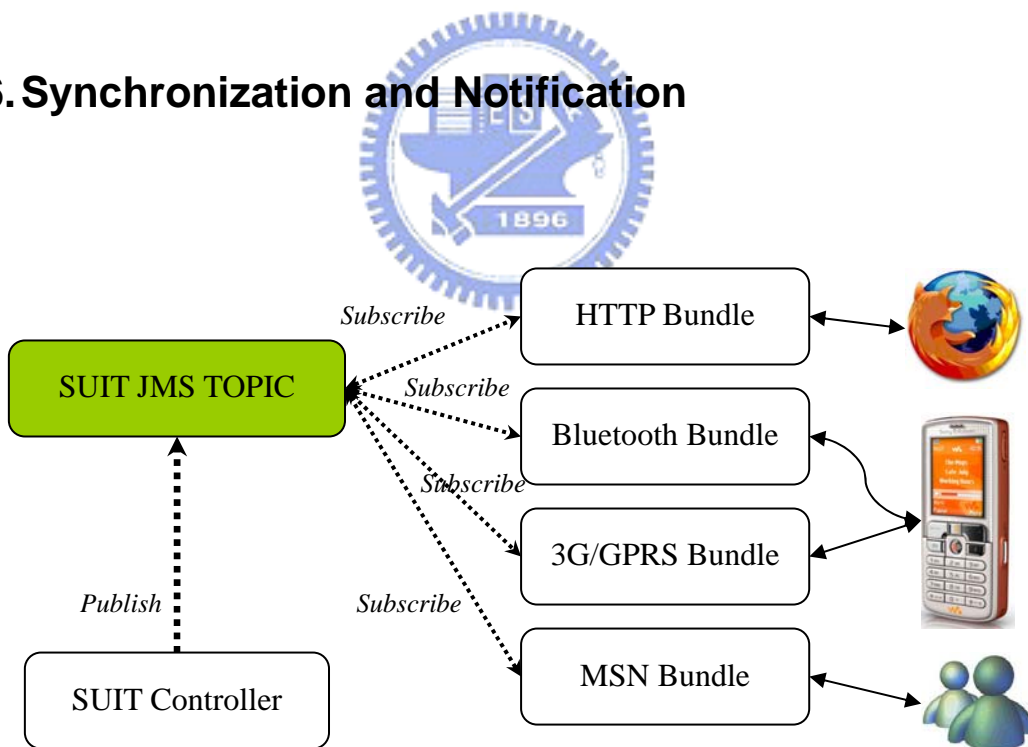


Figure 4-11 Synchronization and Notification

With various types of remote consoles, it is not easy to notify all of the remote console bundles. The purpose of notification is to synchronize the status information of each appliance

on each remote console. The design of this issue on SUIT platform introduces JMS standards into the architecture. As figure 4-11 illustrates, each remote console implements the JMS subscriber if it needs to do synchronization. The SUIT core part is responsible to establish a JMS TOPIC there, and pastes update information on it. SUIT controller whose job is to invoke actins will publish message to the TOPIC once any of the remote consoles successfully completes an action invocation.

After the remote console bundle receives the update message from the JMS TOPIC, it is the proprietary network world. So what will be the next depends on the remote console venders who develop the bundle. Take MSN messenger as an example, when the MSN agent that is wrapped as a bundle receives messages, it will broadcast this message to remote consoles or you can say, friends, in its buddy list. Then users who have added this MSN agent in their buddy list will be notified, and the information for them is synchronized.

4.7. Bundle Management

SUIT is a dynamic scale home domination system, software on the SUIT gateway can be refreshed when the home gateway is restarted or by administrator's refresh command to specific bundle. The run time for bundles is maintained by OSGi implementation which is widely accepted by industry.

In Figure 4-12, we can see that bundles are downloaded from and URL, transferred by HTTP protocol, and installed correctly on the gateway. Every time we try to refresh a bundle that is added by an URL, the procedure goes through again. For example, the IR Robot is embedded with more than 60 actions, however only 20 actions have been tested OK for kids to play with it. The manufacturer may provide the bundle or the documents with bundle that only support 20 actions for that IR robot. Once there are a batch of actions have been tested OK, parents can refresh the bundle or documents, and the remote consoles on the system will show more actions for kids.

This is a fine approach to update software on the home gateway. It keeps the home gateway fresh, and always has the newest version bundles for both home appliance and universal remote consoles.

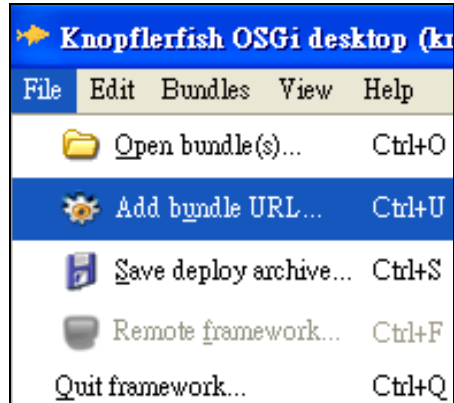


Figure 4-12 Bundle Installation with URL



Chapter 5 Implementation

After the discussion of SUIT system architecture and design issues, we will go into more practical about the implementation. In this chapter we will introduce the OSGi environment construction and SUIT gateway implementation. Then we will present a real case of using MSN remote console to control IR Robot, for the purpose to show how manufacturers of home appliances and remote consoles develop their products on SUIT platform. At last, we will give a summary about this chapter.

5.1. OSGi Environment Construction

The first jog to build our runtime environment is to find an implementation of OSGi specifications. There are three major OSGi specifications implementations available for open source now. They are Oscar, Knopflerfish and Physalis. Oscar and Knopflerfish are implemented by Java and Physalis is for .NET compact framework. We choose Knopflerfish as the base framework of our system. It is the only one that is OSGi R4 compliant and implemented by Java.

After installation, we start the Knopflerfish and it comes with a GUI window. In the upper left frame, there are many icons around. Each icon represents a bundle application, for example, “AIAP-MSNRobot” is a MSN remote console bundle, “RobotActorCP” is the bundle of IR robot home appliance. We will introduce implementation of these two bundles to show how manufacturers develop their bundles in order to join the SUIT system.

We can choose “File” option in the top toolbar, and choose “Open bundles(s)” or “Add bundle URL” to install a bundle, the format of a bundle is *.jar. On the upper right frame, there are descriptions for bundles when you select any bundle on the left. Bottom frame is the console screen for bundle applications.

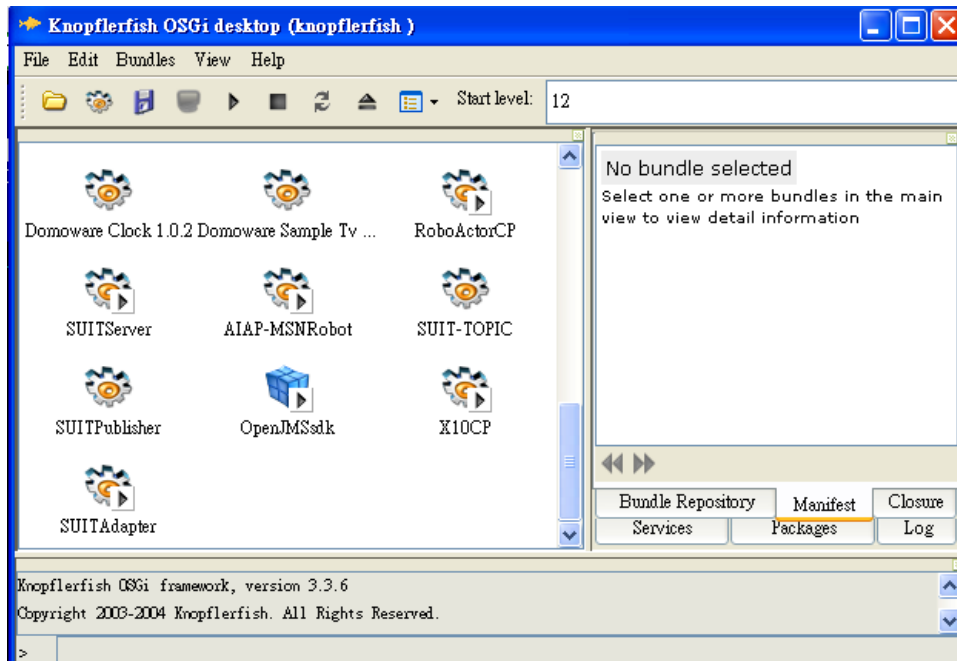


Figure 5-1 Knopflerfish 2 - An OSGi R4 Implementation

5.2. Bundle Implementation

Knopflerfish 2 has published a plugin for Eclipse, the IDE for development. After the installation of plugin to Eclipse, we are able to create a Bundle Project easily.

The BundleActivator is an interface that every bundle must implement it. It can be considered your application's main class or entry. A bundle which is expected to start, register or use other service must have a BundleActivator implementation and the BundleActivator's class name must be referenced in its manifest file. The Listing 5-1 gives a sample code.

The bundle specifies what to do when it starts and ends by implementing the start(BundleContext bc) method and stop(BundleContext bc) method of the BundleActivator interface. The static BundleContext object is used to do jobs related to OSGi runtime, for example, exposing services, using services and looking up services, etc. For example, when we want to register services to the service repository, we call BundleContext.registerService

to expose services for other bundles to lookup and consume the services.

```
// Static bundle context
public class Activator implements BundleActivator {
    static BundleContext bc;
    public void start(BundleContext bc) {
        Activator.bc = bc;
        ...
    }
    public void stop(BundleContext bc) {
        Activator.bc = null; // allow for garbage collection
        ...
    }
}
```

Listing 5-1 BundleActivator



5.3. SUIT Core Part Implementation

The superior parts in SUIT system are URC engine and controller. URC engine is responsible to extract necessary information from raw URC documents and it provides methods for remote consoles to access. The second superior part is SUIT controller which acts like a commander to invoke service bundles for heterogeneous home control network. In next sections, we will discuss the implementation of these two superior parts.

5.3.1. SUIT URC Engine

First in URC engine, we defined three classes for Socket Description Element, Presentation Template Element and Resource Sheet Element, as the code in Listing 5-2. For SocketElement class, we have four class constructors for static, variable, command and notify which are possible elements defined in AIAP-URC specifications. Each of constructors is


designed to have different types and numbers of arguments which depends on the socket element type. For PreElement class, p_ref specifies the related socket element of the Presentation Template Element. p_type is the argument that hints the remote console how to construct the concrete user interface. For ResourceElement class, r_content is the resource used to build up user interface, and e_ref references the related Socket Description Element. r_type here can be “Text” 、”audio” or “image”.

```
// Socket Description Element Class
public class SocketElement {
    //static
    public SocketElement(String s_type, String id, String d_type, String s_content,
        String NULL){ ... }
    //variable
    public SocketElement(String s_type, String id, String d_type){ ... }
    //command
    public SocketElement(String s_type, String id){ ... }
    //notify
    public SocketElement(String s_type, String id, String e_ACK, String c){ ... }
    ...
}
// Presentation Template Element Class
public class PreElement {
    public PreElement(String p_type, String id, String p_ref){ ... }
    ...
}
// Resource Description Element Class
public class ResourceElement {
    public ResourceElement(String r_ID, String r_type, String r_content,
        String e_ref){ ... }
    ...
}
```

Listing 5-2 URC Engine Element Data Structure

URC engine provides services for remote console bundles to use, the Listing 5-3 shows methods provided by the URC engine. `getDeviceList` returns the available home appliance list. After the remote console gets the device list, it picks up one target device and ask Socket Description Elements 、 Presentation Template Elements and Resource Sheet Elements by `GetSocketElementList` 、 `GetPretElementList` and `GetResourceElementList` methods . Each method returns an array of our specified class object types in Listing 5-2. So the URC engine provides sufficient information for remote console bundles.

The method `creatDocument` is for creating `org.w3c.dom.Document` Object which is defined by `w3c`. After the document creation, we can treat the xml file as a DOM tree, and extract the necessary information from the data structure by traversing the data structure. This method is used by URC engine itself, it is not exposed to service repository.



```
// Get available appliance list
public String[] getDeviceList(){ ... }

// Get socket description elements by deive name
public SocketElement[] GetSocketElementList(String device){ ... }

// Get presentation template elements by deive name
public PretElement[] GetPretElementList(String device){ ... }

// Get resource sheet elements by deive name
public ResourceElement[] GetResourceElementList(String device){ ... }

// Create org.w3c.dom.Document object by file name
private Document creatDocument(String docFileName){ ... }
```

Listing 5-3 Methods Provided By URC Engine

5.3.2. SUIT Controller

The main jobs of the SUIT controller are to invoke actions requested from the remote consoles, and return results to them. In Listing 5-4, the arguments part, deviceID is the static element in Socket Description that helps a control point to recognize the device, commandID is the specify action that is enable on the device, args is an array of arguments for the action. The invokeAction method returns a String array which home appliance bundles return as the result of action invocation.

```
// Action Invokation
public String[] invokeAction(int deviceID, int commandID, String[] args){
    ...
}
```

Listing 5-4 SUIT Controller - Action Invocation

5.4. Home Appliance Bundle Development

SUIT platform allows home appliance manufacturers to develop OSGi bundles for their products in order to conduct with our system. In this section, we pretend ourselves as an IR Robot manufacturer who wishes to setup its robot product on the SUIT gateway.

First we find an 8051 IR transmitter that can be connected to computer through serial port. The carrier for this Robot is 39.2 kHz, and it is different from usual home appliance like television which is around 38 kHz. Secondly we have to know the correct IR control codes for the product as Figure 5-2 lists. After the preparation, we start with a program that sends IR control codes through serial port to the 8051 IR transmitter that transmits the codes into the air.

The IR Robot with IR receiver will receive the IR control codes and act as the codes instruct. So now we have a standalone application that is capable to control Robot. We call the program “RoboActorCP” from now on.

Robosapien USBUIRT IR Control Codes				
Group	Action	Command	USBUIRT Code	
Arms	R arm up	rarmup	R08008109808520212122212221212122212121808720	
	R arm down	rarmdown	R08008101808321212122212121212180862022212121	
	R arm out	rarmout	R08008109808321212122212121212122218086202121	
	R arm in	rarmin	R2D9980FE80852021212221212123218086202021808720	
	L arm up	larmup	R0800810E80832121212221212180852120212221808520	
	L arm down	larmdown	R080080F780852021212221212180872080862022212121	
	L arm out	larmout	R0800810B80842021212221212180872021218085212121	
	L arm in	larmin	R080080F48086202121222122218086208086202221808620	
	R arm all up	rarmallup	R14D18109808520212122212221212122212121808720	
	R arm all down	rarmalldown	R12D08101808321212122212121212180862022212121	
	R arm all out	rarmallout	R0FD38109808321212122212121212122218086202121	
	R arm all in	rarmallin	R105180FE80852021212221212123218086202021808720	
	L arm all up	larmallup	R1116810E80832121212221212180852120212221808520	
	L arm all down	larmalldown	R128980F780852021212221212180872080862022212121	
	L arm all out	larmallout	R0FE6810B80862021212221212180872021218085212121	
	L arm all in	larmallin	R0DC680F48085202121222122218086208086202221808620	
	Movement	walk forward	walkf	R239B80F980842121212221212121218086208086202121
		walk backward	walkb	R153480EF8085202121222121212321808620808421808520
step forward		stepf	R080080F38085202121808620222121218086208086202121	
step backward		stepb	R08008108808520212180862022212121808620808620808520	
stop		stop	R165580FC8085202121222122218085208086208086202121	
lean left		leanl	R149C80FC8085202121222122218085202121808620808520	
lean right		leanr	R10AD810680852021212221212123212021808620808720	
turn step left		turnstepl	R080080FD80852022218086202021808720212121212121	
turn step right		turnstepr	R08008103808420212180862121212121222122212121	
left		left	R2AAB8104808520212122212221808620212122212121	
right		right	R1FC381098085202121222122212121222121212121	

Figure 5-2 RoboSapien IR Control codes

To wrap the standalone application as control point software on SUIT gateway, we give it a BundleActivator implementation as the entry of the program. In Listing 5-5, we create a RoboActorCP object in the start() method, and register the RoboActorCP object by BundleContext.registerService. Then the registration procedure is done after the installation

of the Robot control point bundle. SUIT controller will be able to lookup this service in the service repository, create the object, and use the methods with the object. And every remote console bundle sends invocation requests to the SUIT controller to control IR robot.

```
public class Activator implements BundleActivator {
    public void start(BundleContext context) throws Exception {
        RoboActorCP RBCP=new RoboActorCP();

        Hashtable props = new Hashtable();

        props.put("description", "This is RoboActor Control Point");

        context.registerService(RoboActorCP.class.getName(), RBCP, props);
    }

    public void stop(BundleContext context) throws Exception { ... }
}
```

Listing 5-5 Wrapping Robot Control Point as OSGi Bundle



Besides, the home appliance manufacturers have to provide xml documents which obey the AIAP-URC specifications, the documents describes the exposed services. For example, “RoboActorCP” provides services to the home domination system, and the IR robot manufacturer provides the AIAP-URC documents to describe the service. For example, the variable type element with the id “12” described in Socket Description is the argument passed to the invocation service. The developer also writes a Presentation Template element with select type references to the “12” element in Socket Description, the select type gives remote consoles hint to construct the user interface. At last, they don’t forget the most important part which is human readable resource. In the Resource Sheet, there is an element with the context “Go ahead, Robot!” references to “12” element in Socket Description. So now human have lively words for their user interfaces instead of “12”.

Above is a practical example of home appliance bundle development. Manufacturers may follow almost the same steps to produce their bundles, UPnP·Java Jini·X10·Insteon and so on. Once the manufacturers have the technology to control their device, it is easy to conduct their product by wrapping up control software as OSGi bundles and give xml documents that obey the AIAP-URC specifications.

5.5. Remote Console Bundle Development

SUIT platform allows remote console manufacturers to develop OSGi bundles for their products in order to conduct with our system. In this section, we pretend ourselves as a MSN remote console manufacturer who wishes to setup its robot product on the SUIT gateway.

Using IM agent as portal of a service-oriented infrastructure is our innovation. IM agent is varied from traditional portal web pages, IM agent notifies people when web pages expect you to query passively. You can think it as a friend in your buddy list, this friend works for you, listens to you, and does what you want.

To write a MSN robot/agent, it is complicated to finish it from bottom up. In other words, it is difficult to implement the MSN protocol stack ourselves. So we found a open source MSN library called MSNM from the sourceforge.org, traced the code and learned how to use the library. Listing 5-6 is an example of MSN agent that waits for its friends to request for “device list”, and the agent tries to find out the available device list form the SUIT system, then the MSN agent sends device list back to the friends in its buddy list.

After remote consoles get the device list, they can pick up one device form the list and use it as parameter to call other methods that are exposed by the URC engine in the service repository as listed in Listing 5-3. For example, GetSocketElementList(String device) returns the array of SocketElement related to the target device.

```

// chatroomAdapter.java
import rath.msnm.event.MsnAdapter;
import rath.msnm.MSNMessenger;
import rath.msnm.SwitchboardSession;
import rath.msnm.entity.MsnFriend;
import rath.msnm.msg.MimeMessage;

public class ChatroomAdaptor extends MsnAdapter{
    CP controlPoint=new CP();
    String msg="";
    ...
    public void instantMessageReceived( SwitchboardSession ss,
        MsnFriend friend, MimeMessage mime ){
        if(mime.getMessage().contains("Device List")){
            msg= controlPoint.getDiviceList(bc);
            sendMessage(ss,msg);
            break;
        }
        ...
    }
    ...
}

// CP.java
public class CP {
    ...
    public String[] getDiviceList(BundleContext context)throws Exception {
        this.bc=context;
        ServiceReferenc sr=bc.getServiceReference(Adapter.class.getName());
        Adapter aiapAdapter=(Adapter)bc.getService(sr);
        String deviceList[]=aiapAdapter.getDeviceList();
        return deviceList;
    }
    ...
}

```

Listing 5-6 MSN Robot Remote Console Code

After successfully installation of MSN remote console bundle, we can add the MSN account to our buddy list. Once the bundle starts, as Figure 5-3 shows, present message of the MSN agent appears. When we open a dialog with the MSN agent, it gives instructions about how to talk with it. I type “Give me device list” (in the figure is in Chinese), it tells me there are four device available on the gateway, and I am able to ask more details about certain appliance, and more important, operate it! You can see that six actions for the IR robot which are accessible for remote consoles now.

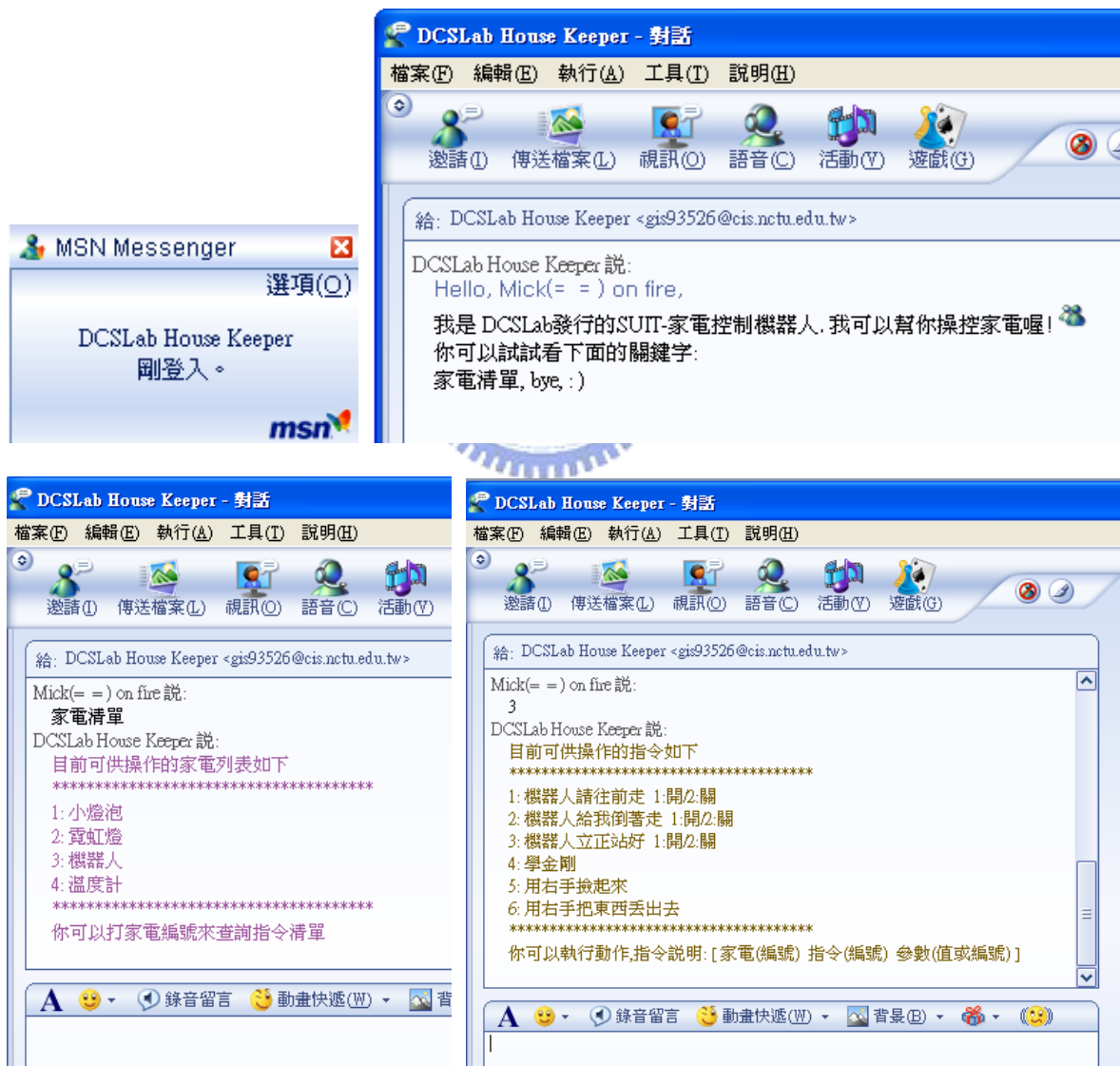


Figure 5-3 MSN Remote Console Screen Shot

In this real case implementation, it shows that as a consumer or say, human in the SUIT system, I don't have to know anything about UPnP, Jini, X10, etc. They are the same to me, they are just home appliances, and easy to operate. To control the IR robot, by user interface abstraction technology, I don't have to memorize the IR control codes. Instead I see friendly and interesting operation instruction which is for man, not cold machine.

Above is a practical example of remote console bundle development. Manufacturers may follow almost the same steps to produce their bundles, Bluetooth server 、 3G server 、 http server or intelligent agent service, etc. Every remote console is able to control every home appliance once remote console bundle is developed well and installed successfully.



Chapter 6 Conclusion

6.1. Conclusion

We have introduced our solution to realize the universal home domination, the SUIT platform which is suitable for every appliance and remote console. SUIT platform eases the sufferings for both home appliance manufacturer and remote console manufacturer. They don't have to worry about what kind of network they must support, which one will be more competitive in the future. Proprietary technology plus OSGi bundle and AIAP-URC documents will be perfectly integrated with other proprietary networks, and also different kinds of remote consoles.

More importantly, customers who use the SUIT system to manage their homes are likely to love it after trying it. They don't have to care numerous types of home control technologies, what they see are just simply appliances, with beautiful costumes, or say, with friendly and kindly user interface, not just cold machine instructions. In another point of view, customers are free to install the remote console they're used to, and any kind of remote consoles are capable to control anything on the SUIT gateway. SUIT platform is designed to serve people, instead of expecting people to compromise with it.

In order to appraise the SUIT platform from manufacturers' standpoints, we pretended ourselves as home appliances manufacturers and remote console manufacturers. After real cases implementation and evaluation, we prove that SUIT platform is quite satisfying for those manufacturers who wish to conduct their products into SUIT system.

6.2. Comparisons

Two majority spirits of universal home domination is the integration of the underlying home control network and user interface abstraction. So we would like to compare four types

of home control system as in Table 6-1.

Home Network Integration	User Interface Abstraction	Home Control System
No	No	<p>This is the widely used system type.</p> <p><u>Integration</u>: Use specific proxy to do the transformation of two networks. Hard to support new networks.</p> <p><u>UI Abstraction</u>: Developers design user interfaces one by one based on the transformed functions, and they have to design for every kind of remote console devices.</p>
Yes	No	<p>Service-oriented system type</p> <p><u>Integration</u>: Service-oriented model, home networks are transformed as services. Easy to have new networks.</p> <p><u>UI Abstraction</u>: Although services are transformed and managed well, developers still have to design user interfaces one by one based on the services, for every kind of remote console device.</p>
No	Yes	<p>URC based system type</p> <p><u>Integration</u>: URC standards expect vendors build the standards on their proprietary networks. There is no integration solution provided in URC specifications. To support different networks, you need to develop for each.</p> <p><u>UI Abstraction</u>: Complete user interface abstraction layer, no need to design user interface one by one on every remote console for one specific appliance.</p>
Yes	Yes	<p>SUIT system</p> <p><u>Integration</u>: OSGi based service oriented infrastructure. Various home networks are transformed as services in the service repository. Easy to have new ones. Every appliance can be controlled by any remote console.</p> <p><u>UI Abstraction</u>: With URC adaptation to OSGi services, user interfaces are wrote once and used everywhere. Developers don't have to design user interfaces one by one for each remote consoles. URC is and international standards, not proprietary defined, which every one knows how to design documents.</p>

Table 6-1 Comparison with Other Systems

6.3. Future Works

So far we just gave a start to realize the universal home domination concept. There are still things to do to make the SUIT platform better and more complete. We list some future works and enhancements.

- Complete support of AIAP-URC

We propose the idea to adapt URC standards to OSGi platform. In our implementation, we have extracted the key points of URC as our bases. It is easy to propose the idea, but hard to completely support the specifications in the implementation.

- Toolkits to design URC documents

Now in our system, manufacturers or third-parties design URC documents by opening text editors and key in their designs in XML format. It will be really nice to have GUI toolkit when designing costumes for home appliances.

- Supplemental resource services model

Supplemental resources are quite important on the SUIT platform. Web service is he suggested mechanism that provides supplemental resource services. This issue is worth of discussion in the future.

Chapter 7 Bibliography

- [1] V2 Technical Committee of the InterNational Committee for Information Technology Standards, <http://www.myurc.com>
- [2] Universal Remote Console Standard–ANSI/INCITS 389-2005, Protocol to Facilitate Operation of Information and Electronic Products through Remote and Alternative Interfaces and Intelligent Agents: Universal Remote Console.
- [3] User Interface Socket Description Standard–ANSI/INCITS 390-2005, Protocol to Facilitate Operation of Information and Electronic Products through Remote and Alternative Interfaces and Intelligent Agents: User Interface Socket Description.
- [4] Presentation Template Standard–ANSI/INCITS 391-2005, Protocol to Facilitate Operation of Information and Electronic Products through Remote and Alternative Interfaces and Intelligent Agents: Presentation Template.
- [5] Target Properties Sheet Standard 392-2005, Protocol to Facilitate Operation of Information and Electronic Products through Remote and Alternative Interfaces and Intelligent Agents: Target Description.
- [6] Resource Description Standard–ANSI/INCITS 393-2005, Protocol to Facilitate Operation of Information and Electronic Products through Remote and Alternative Interfaces and Intelligent Agents: Resource Description.
- [7] Gregg C Vanderheiden Ph.D., Gottfried Zimmermann Ph.D., “Use of User Interface Sockets to Create Naturally Evolving Intelligent Environments”. 2005
- [8] OSGi Alliance, <http://www.osgi.org/>
- [9] Knopflerfish - Open Source OSGi, <http://www.knopflerfish.org/>
- [10] Oscar Project, <http://oscar.objectweb.org/>
- [11] Prosys, <http://www.prosys.com/exec/osgi>

- [12] Domotics Software, <http://domoware.isti.cnr.it/>
- [13] Pebbles Project, <http://www.pebbles.hcii.cmu.edu/>
- [14] Gregg Vanderheiden Ph.D., Gottfried Zimmermann Ph.D., Shari Trewin Ph.D.,
“Interface Sockets, Remote Consoles, and Natural Language Agents A V2 URC
Standards Whitepaper”. 2005
- [15] Gottfried Zimmermann, “Universal Control Hub & Pluggable User Interfaces”. 2005
- [16] Trewin, S., Zimmermann, G., and Vanderheiden, G., “Abstract User Interface
Representations: How well do they Support Universal Access?” 2nd ACM Conference
on Universal Usability, Nov. 10-11, 2003 , Vancouver, BC, Canada.
- [17] Gottfried Zimmermann, Gregg Vanderheiden, Matthew Ma, Maribeth Gandy, Shari
Trewin, Sharon Laskowski , Mark Walker , “Universal Remote Console Standard -
Toward Natural User Interaction in Ambient Intelligence”, 2005
- [18] Zimmermann, G. “Toward a Unified Universal Remote Console Standard”, CHI 2003
Extended Abstracts, pp. 874f.
- [19] 林慧雯, “A High Extensible Home Appliance Control Platform”,國立交通大學電資學
院碩士班論文，民 94.
- [20] Latvakoski, E.J.; Paakkonen, P., ”Remote interaction with networked appliances
attached in a mobile personal area network,” Communications, 2003. ICC '03. IEEE
International Conference, May 2003.
- [21] Mariana Nikolova, Frans Meijs and Peter Voorwinden,” Remote Mobile Control of
Home Appliances,” IEEE Transactions on Consumer Electronics, FEBRUARY 2003.
- [22] Noriyuki Kushiro, Shigeki Suzuki, Masanori Nakata, Hideki Takahara and Masahiro
Inoue,” Integrated Residential Gateway Controller for Home Energy Management
System engineering,” IEEE Transactions on Consumer Electronics, AUGUST 2003.
- [23] Zhaohui Ye, Yindong Ji, Shiyuan Yang,” Home Automation Network Supporting
Plug-and-Play,” IEEE Transactions on Consumer Electronics, Vol. 50, No. 1,

FEBRUARY 2004.

- [24] Song Yean Cho, Dae Young Seo, Tai Yun Kim, "Gateway Framework for Home Appliance's Interoperability Based on Heterogeneous Middleware in Residential Networks," Consumer Electronics, 2002. ICCE. 2002 Digest of Technical Papers. International Conference on 18-20 June 2002
- [25] URC Simulation Environment - Trace Center Version 2.0 (Aug. 2005),
<http://www.myurc.com/UrcSimEnv/applet.html>
- [26] Smarthome.com - World's Largest Home Automation Retailer, www.smarthome.com/
- [27] UPnP™ Forum, <http://www.upnp.org/>
- [28] Jini.org, <http://www.jini.org/>
- [29] JMSN, a MSN library, <http://sourceforge.net/projects/jmsn>
- [30] Robosapien.tk, Unofficial Robosapien Hacks and Mods Site, <http://robosapien.tk/>

