Web Call Example Application

A dissertation submitted to the Royal Institute of Technology (KTH) in partial fulfillment of the requirements for the degree of Master of Science

SHANBO LI

Master's Thesis at ERICSSON AB
Supervisor at Ericsson: Peter Yeung
Supervisor at KTH: Professor Mihhail Matskin, PhD
Examiner: Professor Mihhail Matskin, PhD

TRITA xxx yyyy-nn

Abstract

Web Call Example Application from Ericsson Developer Connection is an application that hosted at a web server and supplies functionality of VoIP phone calls. Users can access the service from desktop browser, mobile phone browser or Java ME Client. Users can also manage their contact books. Each user can have more than one VoIP service accounts, so they can choose the cheapest on when they make phone call. The Web Call Example Application supports two kinds of VoIP phone call connection: Relay Call and Third Party Call. For the Third Party Call there are four different implementations: Call Transfer, SDP Swap, Re-invite and Web Client. This master thesis introduces the theory of third party call and the implementation of Web Call Example Application.

To my parents, LI Chongzhi and WU Wei, who have guided me through life and encouraged me to follow my own path, and to my wife, MEI Dan, for being waiting for me and keeping faith in me.

Contents

Acknowledgments

My first acknowledgment goes to Peter Yeung, my supervisor at Ericsson Developer Connection, who shows me the standard of enterprise level programming, the magic of design pattern and the way of working at a world-leading company. Web Call Example Application was Peter's idea, and it is my pleasure to make it come true. I also give my acknowledgment to Pär and Erik, my team members at Ericsson Developer Connection, thank you for discussing technical problems with me, I learned a lot from you. I thank all my colleagues, Vera, Yanling, Lena, Olle, Jean-Louis, Marc, Marcelo and others for your kindly help during my life at Ericsson. I thank my manager at EDC, Jörgen, for his support and continuous encouragement.

I thank Professor Mihhail Matskin, who is my superviser at KTH. He reviewed my thesis carefully and gave me a lot of useful comments. I thank Professor Gerald Q. "Chip" Maguire Jr., who gave me many good advices on solving Third Party Call. I thank Abdul Haseeb, who helped me to coordinate my study and thesis work.

At last, I give special thanks to Sike Huang, my unique teammate at KTH. I want to say, you are the one who teach me Java and it is so lucky and happy to ever work with you.

Chapter 1

Background

This chapter contains the background of Web Call Example Application as well as the task of the project. This chapter also explains the relationship between Web Call Example Application and its predecessor project Web Call SDK. Terminologies are also covered in this chapter.

1.1 Background

The technology of Voice over Internet Protocol (VoIP) enables communications services go through Internet. And it is on its way of taking the place of the traditional PSTN telephone network, especially in the domain of long distance or international phone calls. The market of VoIP is getting larger and larger. As the world-leading supplier in telecommunications, Ericsson realizes that the position of VoIP technology in the future of telecom industry is absolutely outstanding. To get the potential huge amount of customers, a project named Web Call Example Application(former *Web Call SDK*) is planned by the department of Ericsson Developer Connection. The design goal is to create a web application that makes phone to phone VoIP calls.

1.2 Task

The task is to take all the benefits of VoIP and create a powerful application for phone calls. The application should be simple, stable, reusable, extendable, easy to access and user friendly.

It should be a splendid application that supplies the function for users to make a phone call anytime, anywhere and to anyone in this world.

1.3 Terminology

Java

The Java™ programming language is a popular high-level language which provides a portable feature and can be used on many different operating systems.

The source code of Java is first written in plain text files which end with the `.java`. Then the Java source files are compiled into bytecodes which ends with `.class` by Java compiler (javac). A `.class` file is platform independent. It will be executed by the Java Virtual Machine*. [3]

A Java application can be distributed in the format of Java ARchive (JAR) file.

Java EE

Java Platform, Enterprise Edition (Java EE) is a set of coordinated technologies that significantly reduces the cost and complexity of developing, deploying, and managing multitier, server-centric applications. [19]

Java ME

Java Platform, Micro Edition (Java ME) is a collection of technologies and specifications to create a platform that fits the requirements for mobile devices such as consumer products, embedded devices, and advanced mobile devices. [20]

MIDlet

A MIDlet is a Java application framework for the Mobile Information Device Profile (MIDP) that is typically carried out on a Java-enabled cell phone or other embedded device or emulator.

RMS

The Java ME Record Management System (RMS) provides a mechanism through which MIDlets can persistently store data and retrieve it later. [10]

JAD

The Java Application Descriptor (JAD) file, as the name implies, describes a *MIDlet* suite. The description includes the name of the MIDlet suite, the location and size of the JAR file, and the configuration and profile requirements. The file may also contain other attributes, defined by the Mobile Information Device Profile (MIDP), by the developer, or both. [18]

*The terms "Java Virtual Machine" and "JVM" mean a Virtual Machine for the Java platform. [3]

SSL

Secure Sockets Layer (SSL), is a cryptographic protocol which provides security and data integrity for communications over networks such as the Internet. [35]

SIP

Session Initiation Protocol (SIP) is an application-layer control (signaling) protocol for creating, modifying, and terminating sessions with one or more participants. These sessions include Internet telephone calls, multimedia distribution, and multimedia conferences. [29]

SIP is the format of control signal in VoIP. It describes the sender, receiver. A agent use SIP messages to register on a proxy, establish session or close session.

VoIP

"Internet telephony refers to communications services—voice, facsimile, and/or voice-messaging applications—that are transported via the Internet, rather than the public switched telephone network (PSTN). The basic steps involved in originating an Internet telephone call are conversion of the analog voice signal to digital format and compression/translation of the signal into Internet protocol (IP) packets for transmission over the Internet; the process is reversed at the receiving end." [6]

IMS

The IP multimedia Subsystem (IMS) is a network functional architecture that is seen as a promising solution for facilitating multimedia service creation and deployment, as well as supporting interoperability and network convergence. [2]

SDP

SDP is short for Session Description Protocol. It is intended for describing multimedia sessions for the purposes of session announcement, session invitation, and other forms of multimedia session initiation. [13]

A SIP message may carry a SDP message. The SDP message contains protocol version, session name, information, and most important, the connection data and media descriptions. This supplies a way to manipulate the connection of media flow. [13]

RTP

RTP, the real-time transport protocol, provides end-to-end network transport functions suitable for applications transmitting real-time data, such as audio, video or simulation data, over multicast or unicast network services. [30]

Web Service

A web service is defined by the W3C as *"A software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP-messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards."*. [11]

SOAP

"SOAP is a lightweight protocol intended for exchanging structured information in a decentralized, distributed environment. It uses XML technologies, an extensible messaging framework containing a message construct that can be exchanged over a variety of underlying protocols." [12]

WSDL

"WSDL is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information." [5]

PSTN

PSTN is short for Public Switched Telephone Network. It is the network of the world's public circuit-switched telephone networks. In another word, it is just the traditional phone network.

ISDN

Integrated Services Digital Network (ISDN) is a telephone system network.

ISUP

The ISDN User Part or ISUP is control signal of Public Switched Telephone Networks (PSTN).

GSM

"GSM (Global System for Mobile communications) is an open, digital cellular technology used for transmitting mobile voice and data services." [1] More about GSM could be found at the web site of GSM world (http://www.gsmworld.com/).

1.4 About

Web Call Example Application is an open source project at Ericsson Developer Connection[†] (EDC), Ericsson[‡]. It is the successor project of **Web Call SDK** [4] which is developed by Yuening Chen at Ericsson AB during the year 2007 and 2008. After Yuening finished Web Call SDK, the people at EDC found it is not stable enough and many of the functions are not usable. So they decided to start a new project follow Web Call SDK to make the it stable and add some new features to it. The new project is called **Web Call Example Application**. As the main developer, I took over the new project at March 2008 and finished it at February 2009. I rewrite most code of the old project to make it stable and runnable and added some new features to it. As a result, the **Web Call Example Application** is used as the base library of Ericsson's demo of **Using REST and Web Services to Mash Up Communications Capabilities** [36] [27] at JavaOne™[§] 2009.

[†]Ericsson Developer Connection (former Ericsson Mobility World Developer Porgram) is a department of Ericsson. It helps developers to create applications that incorporate telecommunication network capabilities, such as location-based services, charging, messaging and presence, with sustainability in mind.

[‡]Ericsson is a world-leading provider of telecommunications equipment and related services to mobile and fixed network operators globally.

[§]JavaOne is an annual conference (since 1996) put on by Sun Microsystems to discuss Java technologies

Chapter 2

Requirement

THIS chapter contains the requirements for Web Call Example Application, from the aspect of programming language, stability, reusability, extendibility and integration.

2.1 Programming Language

To make the application potable, **Java** is chosen as the programming language of Web Call Example Application. The Java programming language is a popular high-level language which provides a portable feature and can be used on many different operating systems. For the introduction and more detail of Java please refer to 1.3.

2.2 Stability

The application should be stable and has as less bugs as possible. The application should be designed for deploying on a server for long term use. The concurrent request users may more than one hundred.

2.3 Reusability

The code should be made as generic and reusable. The interface should not constrain on any specific network or service provider. It should follow a common accepted standard. Session Initiation Protocol (SIP) is a signaling protocol, which defined in RFC 3261 SIP: Session Initiation Protocol [29], widely used for multimedia communication sessions such as voice and video calls over the Internet. It should be used as the main signal protocol of Web Call Example Application.

2.4 Extendibility

The application should be able to add new feature according to customer's requirement, e.g., add video call and instant message.

2.5 Integration

The Web Call Example Application should supply a web service API that can be used by other applications. This interface should contain most of the functions of Web Call Example Application.

Chapter 3

Study

This chapter contains the situation of VoIP market. Both service provider and clients are introduced. It also covered an introduction of Third Party Call Control.The unique sale point of Web Call Example Application is also introduced at the end of this chapter.

3.1 VoIP Market

In the telecom market, VoIP technology has gained more and more customers. The advantage of VoIP is obviously, much cheaper fee and almost same quality as traditional telephone. According to *State of the VoIP Market 2008* [16], report from Infonetics shows that, in the year 2007, the subscribers for VoIP are fewer than 80 million all around world. Most of them are in the Asia Pacific region. However by the year of 2011 the user will be 135 million, predicted by MarketResearch.com. And a UK research company Disruptive Analysis Ltd. predicts the users of mobile-VoIP will be 250 million by the year of 2012.

The analyses and figures above draw a brilliant future of VoIP market.

3.1.1 VoIP Service Provider

VoIP service provider is the company which supplies the products of VoIP/PSTN gateway. Or the ones who supply a service which helps customers to call a PSTN phone by a VoIP phone via their service/network. There are hundreds of such companies in the world.

3.1.2 VoIP Client

A VoIP client is a common SIP client software or a IMS client software. This kind of software runs on a computer or mobile device and implements the SIP or/and IMS standard. It works as a phone to dial or answer VoIP calls.

3.1.3 Solution Provider

A solution provider is a company that supplies both VoIP service and software client, such as Skype™*, Nonoh† and JAJAH‡. Among them Skype is the most famous one. It has a very good quality of voice and functionality client. However, the Skype is not following the standard of SIP. So it means, only the Skype client itself can use the service of Skype. Nonoh and JAJAH supply relevant lower fee and less quality of audio.

3.2 Third Party Call Control

According to *Best Current Practices for Third Party Call Control (3pcc) in the Session Initiation Protocol (SIP) (RFC 3725)* [28], in the traditional telephony context, third party call control allows one entity (which we call the controller) to set up and manage a communications relationship among two or more other parties. Third party call control (referred to as **3pcc**) is often used for operator services (where an operator creates a call that connects two participants together) and conferencing.

A general work flow of 3pcc is shown in Figure 3.1. The initial side of the phone is the *controller*. The *controller* sends a signal "connect *client A* and *client B*" to the server ①. And the server establishes a call between *client A* and *B* ②.

3.3 Why Web Call Example Application?

"Based on IMS/SIP network technology, the Web Call SDK integrates SIP call control functionality into web containers. This presents an efficient way of implementing the communication convergence of Web, IMS/SIP network, and CS networks. It does not require the installation of client-side browser plug-ins or special client software as most other VoIP services require." [4]

The Web Call Example Application is neither a service provider nor a client that described above. It is more like a controller which acts as an initial side in third party call control. That is, it supports all standard client and service provider. Another advantage of Web Call Example Application is that desktop browser view, mobile browser view and Java ME client all share a same database. The user can

*http://www.skype.com
†http://www.nonoh.net
‡http://www.jajah.com

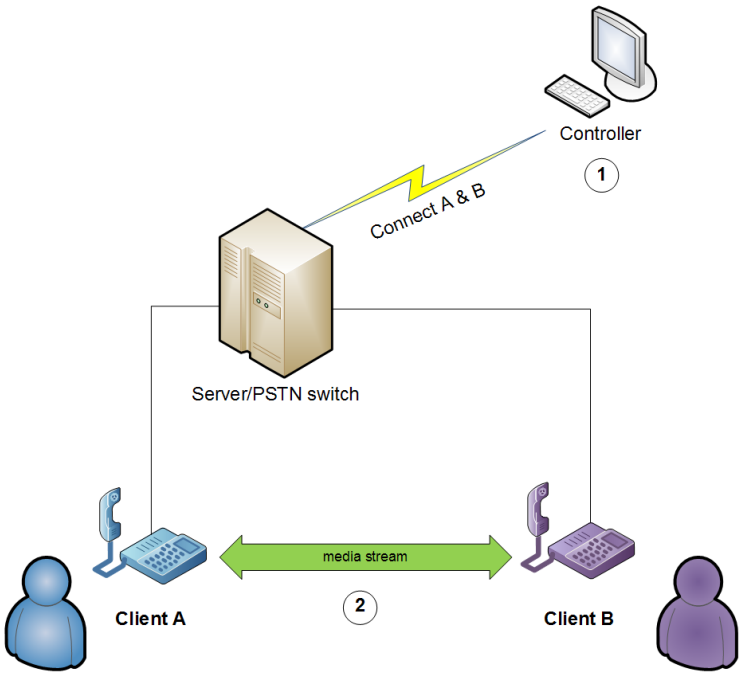

Figure 3.1. Third party call control

access a same contact book and use a same service account from different platform. None of the solution provider or client has the same function.

Chapter 4

Solution

The Web Call Example Application should supplies a functionality of connect VoIP phone calls. The client could be a GSM based mobile phone, a fixed telephone in a PSTN network, or even a VoIP software phone.

There are two kind of connection solution of the core of Web Call, the Relay Call and Third Party Call. The different between them is the way they handle media stream.

The Relay Call Controller works as a back-to-back agent and forwards media streams, while Third Party Call Controller only establishes connections by sending out SIP messages and it does not handle any streams itself.

4.1 Relay Call

The signal and media flow of Relay Call is shown in Figure 4.1. According to *Web Call SDK* [4], in this scenario, the Web Call Example Application acts as a back-to-back user agent. It sets up the connection and forwards the media stream. It can be seen from the picture that both signal and media are handled by Web Call Server. When it starts, it try to call client A. After it establishes a session with client A, it will try to call client B and also establish a session with client B. After that, it will work as a media stream bridge and forward media stream from client A to B, as well as from client B to A.

For a detail description and mechanism of Relay Call, please refer to the master thesis of **Web Call SDK** by *Yuening Chen* [4].

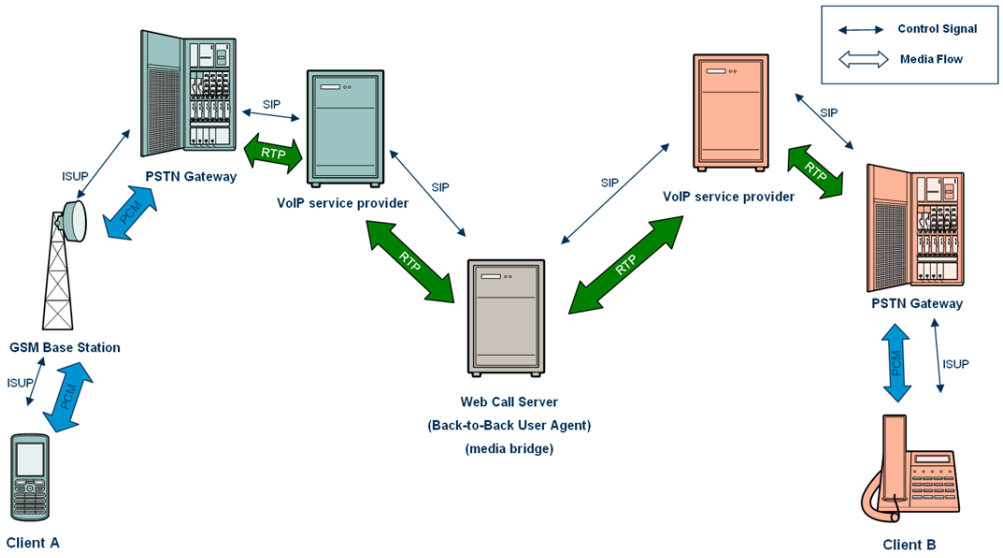

Figure 4.1. The signal and media flow of Relay Call

4.2 Problem of Relay Call

4.2.1 The Load on Controller

The controller here acts as a back to back user agent (B2BUA). It receives the RTP flow from one client and transfers it to another client. It does the same in the opposite direction. That means all the RTP traffic will go through the controller. So the load on controller will be heavier as with the number of concurrent users increases. A powerful host is needed to transfer the RTP flows. However the design goal of Web Call SDK was simply a tool kit that can easily be integrated into a web site. Unfortunately, this current mechanism of session control will decrease the performance of whole web site.

4.2.2 The Latency of Audio

The session flow shows that, the RTP goes from one client to the controller via a SIP provider and then the controller transfers the RTP to another client via the SIP provider again. This means that each direction of RTP will go through SIP provider twice. So the latency of the phone call via the SIP provider will be double that of normal calls. The latency of phone-to-phone call via Web Call SDK test turned to be more than 2 seconds. It is quite unacceptable.

4.2.3 Lack of Reliability

Since all RTP flows go through the controller, thus if the controller crashes, all calls will be immediately interrupted.

4.3 Third Party Call

"In the traditional telephony context, third party call control allows one entity (which we call the controller) to set up and manage a communications relationship between two or more other parties. Third Party call control (referred as 3pcc) is often used for operator services (where an operator creates a call that connects two participants together) and for conferencing." [28] The signal and media flow in third party call is show in Figure 4.2. The advantage of third party call in Web Call is that the controller only need to handle message transfer and leaves the media flow for the ISP.

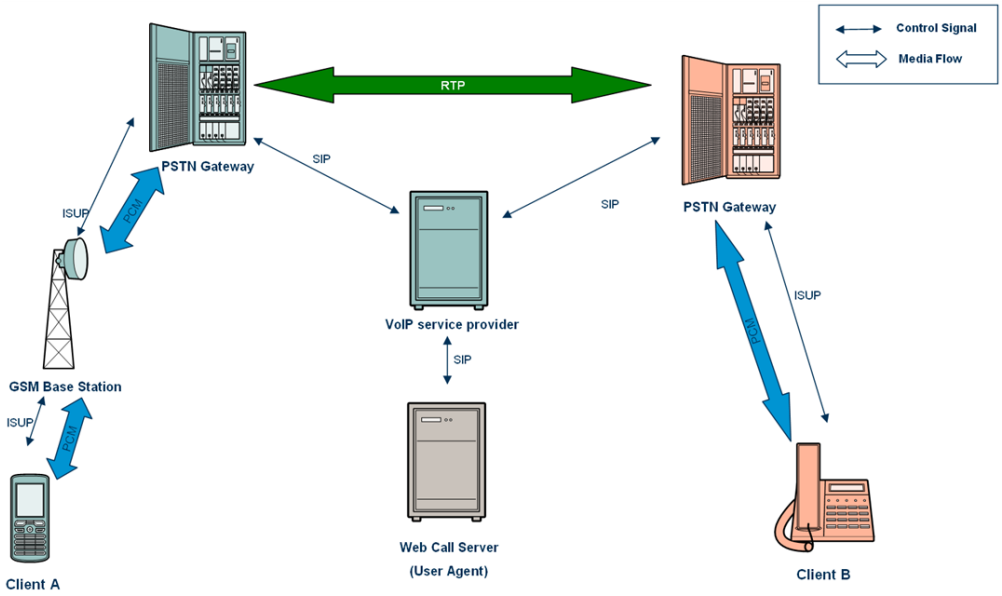

Figure 4.2. The signal and media flow of Third Party Call

4.3.1 Call Transfer

The call transfer implementation use a `REFER` method which defined in *The Session Initiation Protocol (SIP) Refer Method* (RFC 3515) [33]. *"The `REFER` method indicates that the recipient (identified by the Request-URI) should contact a third party using the contact information provided in the request"* [33].

The Call flow is shown in Figure 4.3. The controller first sends an `INVITE` to client A (1). This invite is just a normal invite. A's phone rings and answers. This results in a `200 OK` (2). The controller then answer client A an `ACK` (3). Follow that, the controller send out a `REFER refer-to`: Client B (4), according to RFC 3515 [33], which means controller wish client A to make a phone call to client B. The client A understands that and returns a `200 OK` to controller (5). Then client A sends an `INVITE referred-By`: C to client B (6). Client A and Client B can establish a session according the `INVITE` from A to B. The `BYE` (8) and `200 OK` (9) means the controller cut the media stream between itself and client A.

To accomplish the whole call flow, client A must support RFC3515.

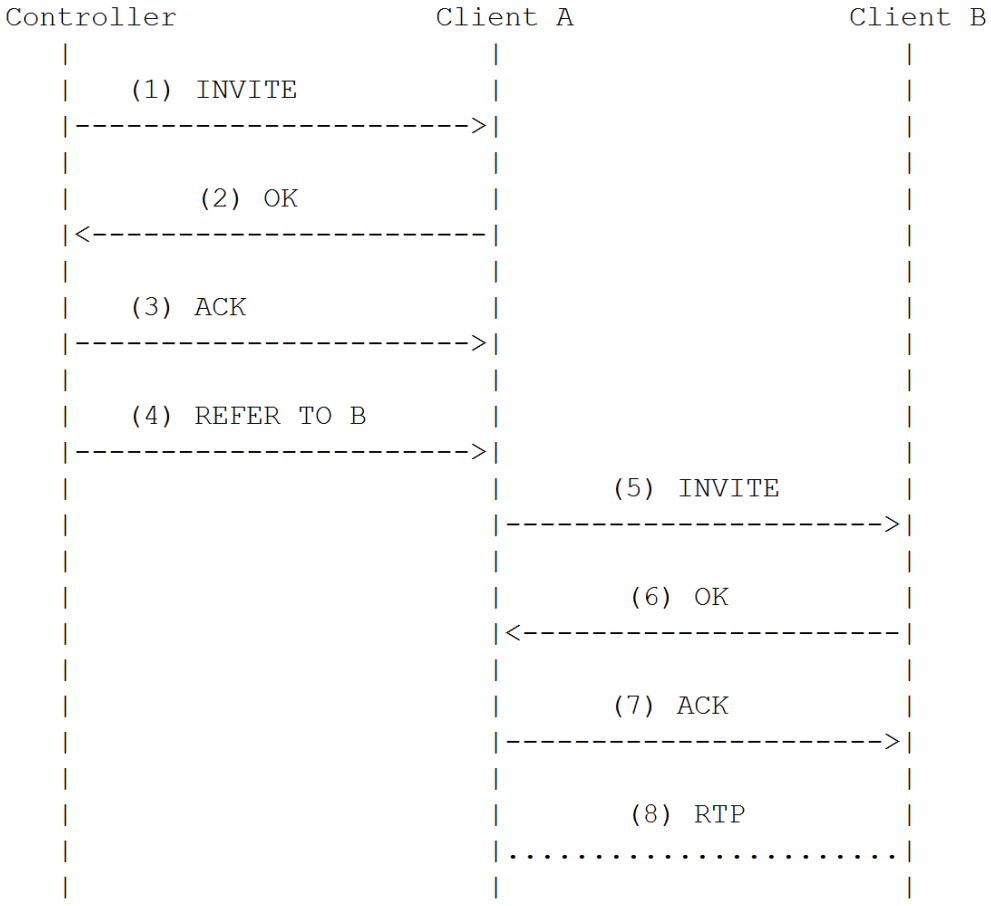

Figure 4.3. The signal and media flow of Call Transfer(Figure according to RFC 3515[33])

4.3.2 SDP Swap

This implementation of third party call control swaps SDP from two clients. The prototype of SDP swap comes from *Best Current Practices for Third Party Call Control (3pcc) in the Session Initiation Protocol (SIP)* (RFC 3725) [28]. However, the call flow in RFC 3725 is rather complicated. The concept of **SDP Swap** is that controller calls to client and swaps two SDP which got from clients. These two clients seem to call controller but the media stream is connected directly between them.

The call flow of SDP swap implementation of third party call control is shown in Figure 4.4. Controller first call client A. This `INVITE` has no session description. Client A's phone rings, and client A answers. It results a `200 OK` (3) that contains an offer which contains client A's SDP [29]. Meanwhile, the controller does the same to client B (2) and gets client B's offer (4) which contains client B's SDP. So far, controller has both client A and client B's SDP. It just simply swaps these two SDPs, then `ACK` client B with client A's SDP (5) and `ACK` client A with client B's SDP (6). So client A gets client B's SDP by (6) and client B gets client A's SDP by (5). Therefore, media flows between client A and client B (7). The SDP Swap method is similar as Flow III of RFC 3725 [28], but not identical. The main differences between them are that SDP Swap send `INVITE` to Client A and B concurrently, while in Flow III of RFC 3725, the controller first send `INVITE` to Client A, then Client B.

4.3.3 Re-invite

According to *SIP: Session Initiation Protocol (RFC 3261)* [29], a `Re-INVITE` is used to change the session parameters of an existing or pending call. It uses the same Call-ID, but the CSeq is incremented because it is a new request. The Re-invite implementation send two client's SDP to each other in the Re-invite process.

The call flow is shown in Figure 4.5. From the SIP message (1) to (6), the controller uses a three way handshake to establish connections with client A and client B. In (3) and (4) the controller gets client A and client B's SDP which are going to be used in the Re-invite phase in (7) and (8). The controller sends a `Re-INVITE` to A with B's SDP (7), which indicates the controller changes its media port to B's. At the moment, controller also sends a `Re-INVITE` to client B with client A's SDP, which indicates the controller change its media port to A's. Both client A and client B gets each other's SDP, thus, a media stream could be established between A and B (13).

4.3.4 Web Client

Most of the VoIP service providers supply a way of making phone to phone call via their web sites. To establish the call, users have to login to the web site and fill in the caller number and callee number. The web client way of third party call use

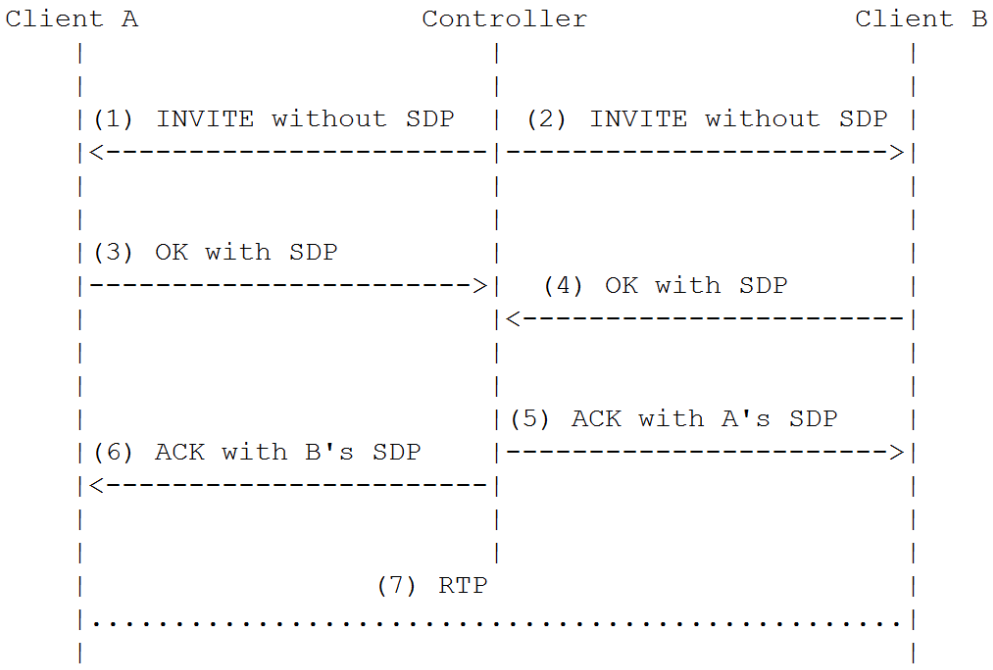

Figure 4.4. The signal and media flow of SDP Swap

a project called **scallope** [17]. It is an application which acts as a web client and login to sip provider's web page and to make VoIP calls.

Scallope is a Java API which based on `apache common http` API. All the user needs to do is just pass the user name, password, caller number and callee number to the API. The Scallope will access VoIP service provider's web page via http protocol and establish a call session for user.

This kind of 3pcc makes it possible to start a phone to phone call without a web browser. For the aspect of control signal and media flow, everything is within provider's network. So the quality of audio is quite close to PSTN and very much acceptable.

4.4 Conclusion

This section introduced Relay Call, Call Transfer, SDP swap, Re-invite and Web Client, five different method of VoIP calls. SIP signal flows are shown for Call Transfer, SDP swap, Re-invite. All VoIP call methods have their cons and pros. The comparison and analysis of them can be found at chapter 10.

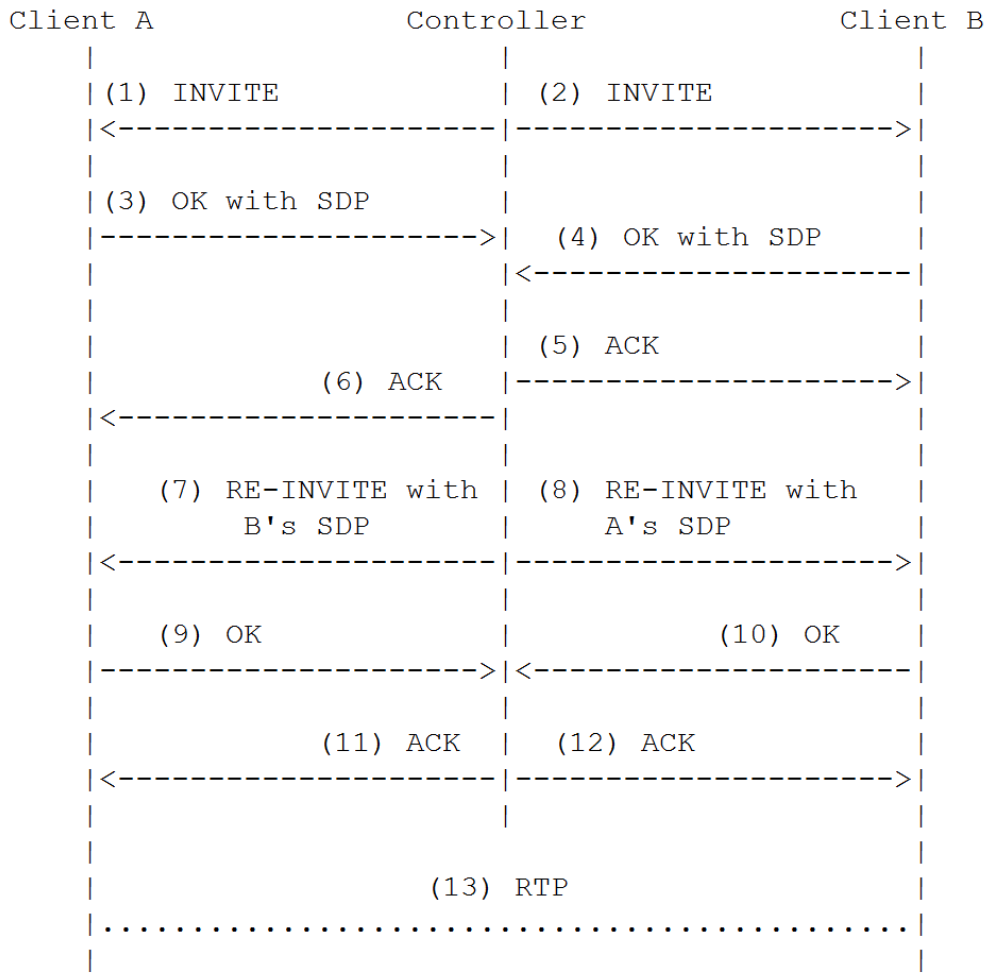

Figure 4.5. The signal and media flow of Re-invite

Chapter 5

Application Overview

CHAPTER 5 shows the brief use cases of Web Call Example Application. It also demonstrates the design architecture of the application. The components are also explained in general. In follow chapters, that is from chapter 6 to chapter 9, SIP Call Component, Web Application, Web Service Interface and Java ME client will be introduced separately in detail.

5.1 Use Cases

Web Call Example Application is an application that resides at a web server and supplies functionality of VoIP phone calls.

As shown in Figure 5.1, there are three use cases. They are Desktop Browser, Mobile Browser and Java ME Client. This makes it possible for users to make VoIP calls anytime anywhere, to anyone. The use case of Desktop Browser is used when user have a computer connected to Internet. He can use his desktop browser, e.g., Firefox or Chrome, to reach the web call site by inputting URL to address bar of browser. Users can use this web site to make VoIP phone calls as well as manage his account. The use case of Mobile Browser is just a web site for mobile browsers. The function is the same as Desktop Browser. The user can also manage his account from the Mobile Browser view. Compare with Mobile Browser, the use case of Java ME Client gives the user an even easier way to make VoIP phone calls. There is no need to sign in and input any information. As long as user pre-input everything on web site, the Java ME Client will retrieve account information from the server. All user need to do is just click the "call" button and enjoy the cheap VoIP phone calls. The Java ME Client can also load mobile phone's native contact book and call the people in the list. It can also synchronize the contact book of mobile phone to server. So user can use the contact book later when he sits in

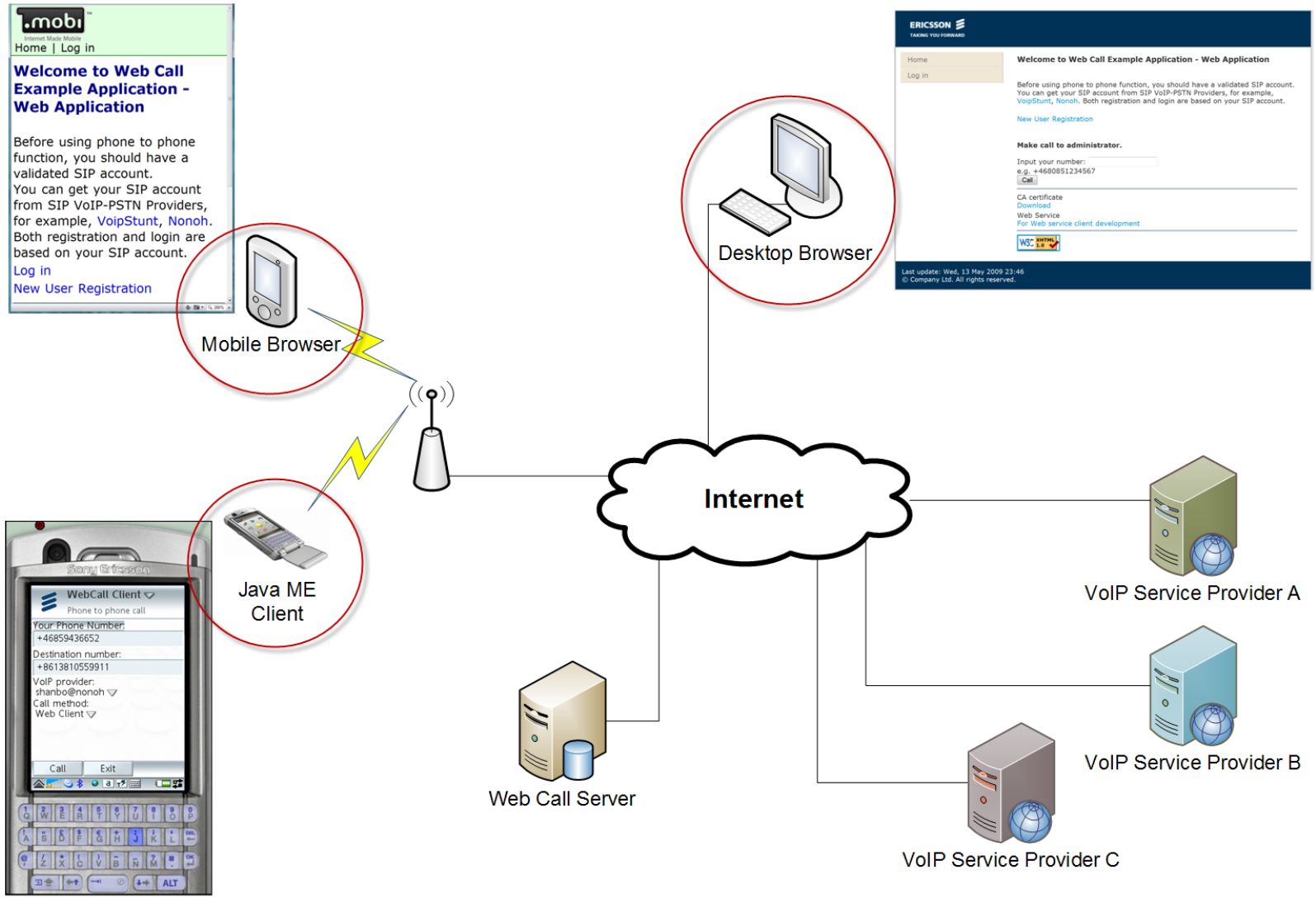

Figure 5.1. Use Cases of Web Call Example Application

front of a computer. A user could have multi-account for different VoIP service providers. So when the user wants to make an international call, he could choose a cheap cost service provider according different rates among providers.

5.2 Architecture

This section presents a brief solution description of Web Call Example Application. The design architecture is shown in Figure 5.2.

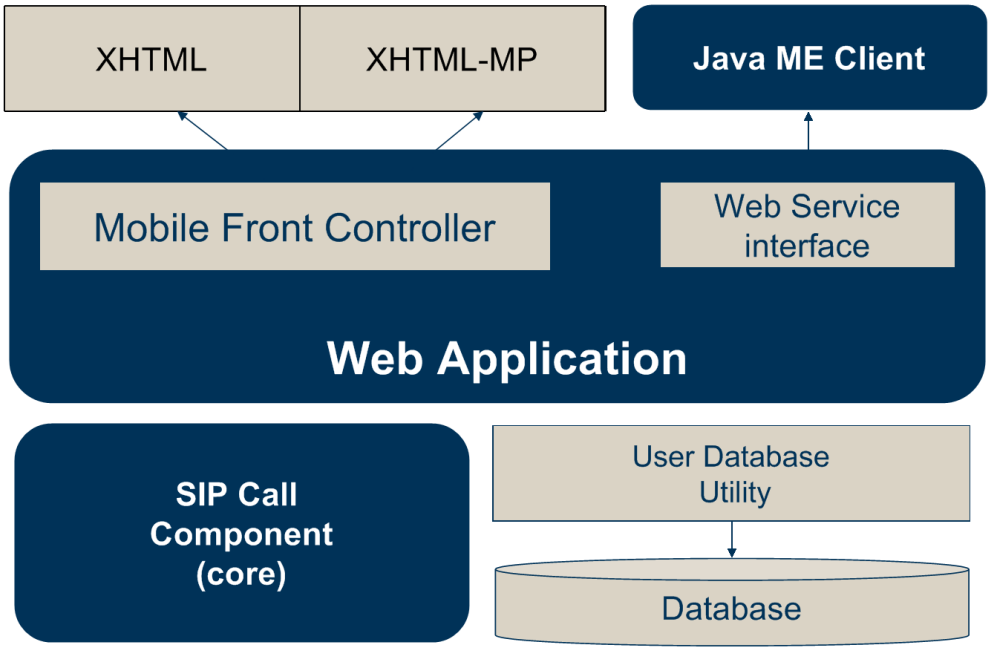

Figure 5.2. The Architecture of Web Call Example Application

Web Call Example Application contains three components which are SIP Call Component, Web Application (include web service interface) and Java ME client. The three components are shown with dark blue back ground in Figure 5.2.

5.2.1 SIP Call Component

The SIP Call Component is the core of Web Call Example Application. It offers one kind of relay call and four kinds of third party call. It can be also used as a standalone VoIP high-level API.

The details about SIP Call Component will be described in Chapter 6.

5.2.2 Web Application

The Web Application is built on the architecture of Mobile Front Controller (MFC). SIP Call Component integrated into the web server as Java EE components: servlet, and web service. The Mobile Front Controller is used for detecting and selecting views, i.e., applications with views for desktop and mobile browsers. So Web Call Example Application supplies both XHTML view which used by desktop browser and XHTML-MP view which used by mobile browser.

The details about Web Application will be described in Chapter 7.

5.2.3 Web Service Interface

The web service interface in web application supplies a common interface for using sip call function. It uses a same database as MFC based Web Application.

The details about Web Service Interface will be described in Chapter 8.

5.2.4 Java ME client

Java ME client is a client of web service interface. It not only implements all the client side functions of web service interface, but also includes some convenient functions such as read phone contact book and synchronize the contact book with server.

The details about Java ME client will be described in Chapter 9.

Chapter 6

SIP Call Component

This chapter focuses on the SIP Call Component of Web Call Example Application. It introduces a five layers software architecture and explains how Web Call Example Application follows this architecture. It also demonstrates the design and class hierarchy of Call Controller.

6.1 Five layers architecture

A five layers architecture has to be introduced before illustrate the Third Party Call Controller. The five layers architecture is shown in Figure 6.1. From bottom to top, the five layers are Protocol stack, Abstraction layer, Implementation layer, OOP (Object-oriented programming) use-case based API layer and JavaBean for synchronized communication. This five layers architecture is developed by *Peter Yeung* from Ericsson Developer Connection.

6.1.1 Protocol stack

The protocol stack is the low level API for the project. There could be several implementations of the protocol stack.

6.1.2 Abstraction layer

This layer is used for abstracting low level protocol stack. So the project can be easily migrated from one protocol stack to another without modifying the logic layer.

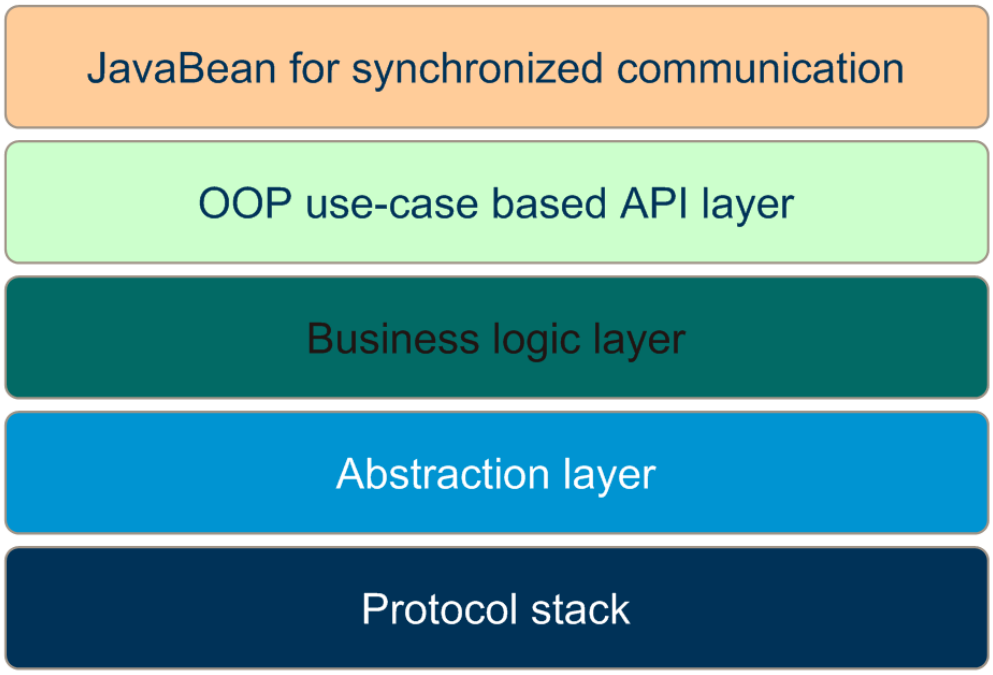

Figure 6.1. Five layers architecture

6.1.3 Business logic layer

This layer contains the main logic of the project.

6.1.4 OOP use-case Based API Layer

This layer is used for exposing public API for end user. Design Patterns such as Abstract Factory Pattern can be used here, especially for java SE.

6.1.5 JavaBean for synchronized communication

Components in Java are called beans. This layer contains the JavaBean which special designed for synchronized communication has two pairs of constructor and business logic group. The constructor which has the system parameters works with business logic which doesn't have system parameters. The constructor which doesn't have system parameters works with business logic which has the system parameters. In addition, this kind of JavaBean only has the method of `getState()` and no method of `setState()`.

6.2 Five layers architecture in Web Call Example Application

Web Call Example Application follows the concept of five layers architecture.

In SIP Call Component, there is abstraction layer and protocol stacks. As is shown in Figure 6.3, among the four implementations of Third Party Call Controller, the three implementations on the left are based on MjSIP and Web Client is based on Apache Common HttpClient. An advantage of separate logic and Protocol stack is that users can choose to use different stacks according to their requirements. Another advantage is that developers can focus on the business logic part and no need to touch lower level stacks. In SIP Call Component, the abstraction layer defined a general interface for normal phone call behaviors, such as `start()` and `terminate()`. The interface of abstraction layer in SIP Call Component is shown in Figure 6.2. As the business logic of SIP Call Component, Call Controller only needs to take care of the call flow, e.g., set properties, register, make phone call and terminate call. And it will not handle any details about SIP message or media flow from the API level. The public API that opens to developers in SIP Call Component is the "OOP use-case Based API Layer".

In Web Call Example Application, SIP Call Component is used as the base library of Web Application. However, it is designed to use as a standalone SIP call high level API. It is independent with other parts of the application. Any developers can take it and integrate it to their own applications.

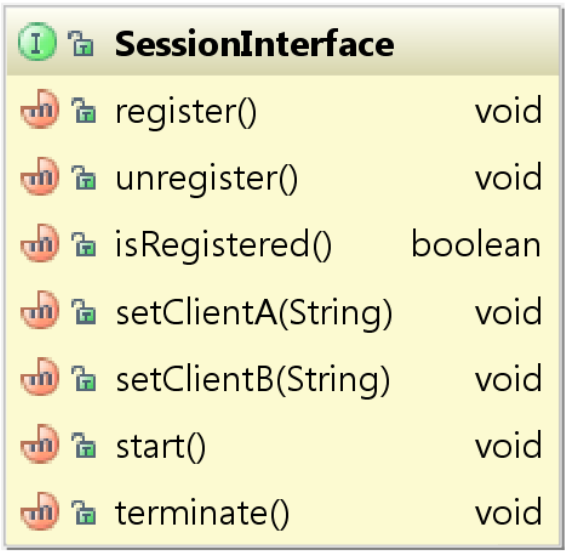

Figure 6.2. Class diagram of `SessionInterface`

6.3 Architecture of SIP Call Component

The SIP Call Component is the core of Web Call Example Application. The design architecture is shown in Figure 6.3.

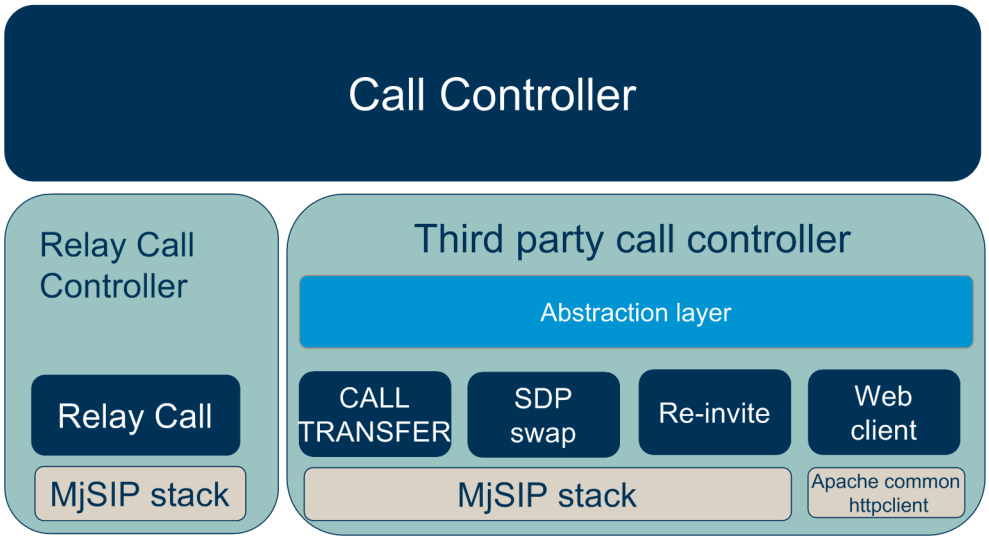

Figure 6.3. The Architecture of SIP Call Component

It can be seen from Figure 6.3, there is a interface `CallController` on the top of all kinds of call controllers. There are two subtype of Call Controller, one is Relay Call Controller and another is Third Party Call Controller. The Relay Call Controller is the legacy code of Web Call SDK which developed by Yuening Chen. In the old project, the Relay Call Controller is the core of all phone calls [4]. However, in this version, four other implements are added to the SIP Call Component. They are Call Transfer, SDP Swap, Re-invite and Web Client. From a outside view, all the four methods are doing the same thing. They just try to establish a phone call between caller and callee. Chapter 4 describes the different among the four methods in detail. Chapter will focus on the architecture and design part. All four third party call methods are implemented the same interface. This means the user can easily switch call method from one to another. And other call method, if there is any, as long as it implements the interface at abstraction layer. It can be used in the SIP Call Component.

6.4 Class Hierarchy

The hierarchy of call controller is shown in Figure 6.4. All of call controllers are implemented the first interface `CallController`. This kind of design enables easily

switch of call controllers. `SessionInterface` is an interface for third party call. `WebClientImpl`, `ReinviteImpl`, `SdpSwapImpl` and `MjSipCallTransferImpl` are implementations for `SessionInterface`.

There are different call controller factories to create call controllers. The call controller factory follows the design pattern of Abstract Factory. An Abstract Factory is a class that exists to create instances of another class [8]. The design of SIP Call Component considered that a call controller cannot only establish audio call, but also video call and message channels. This can be seen from the Figure 6.5. The very general difference between Relay Call and Third Party Call is the way to handle media flow. But both of them should have the competence to establish all kind of sessions. In current version of Web Call Example Application, only audio call is implemented.

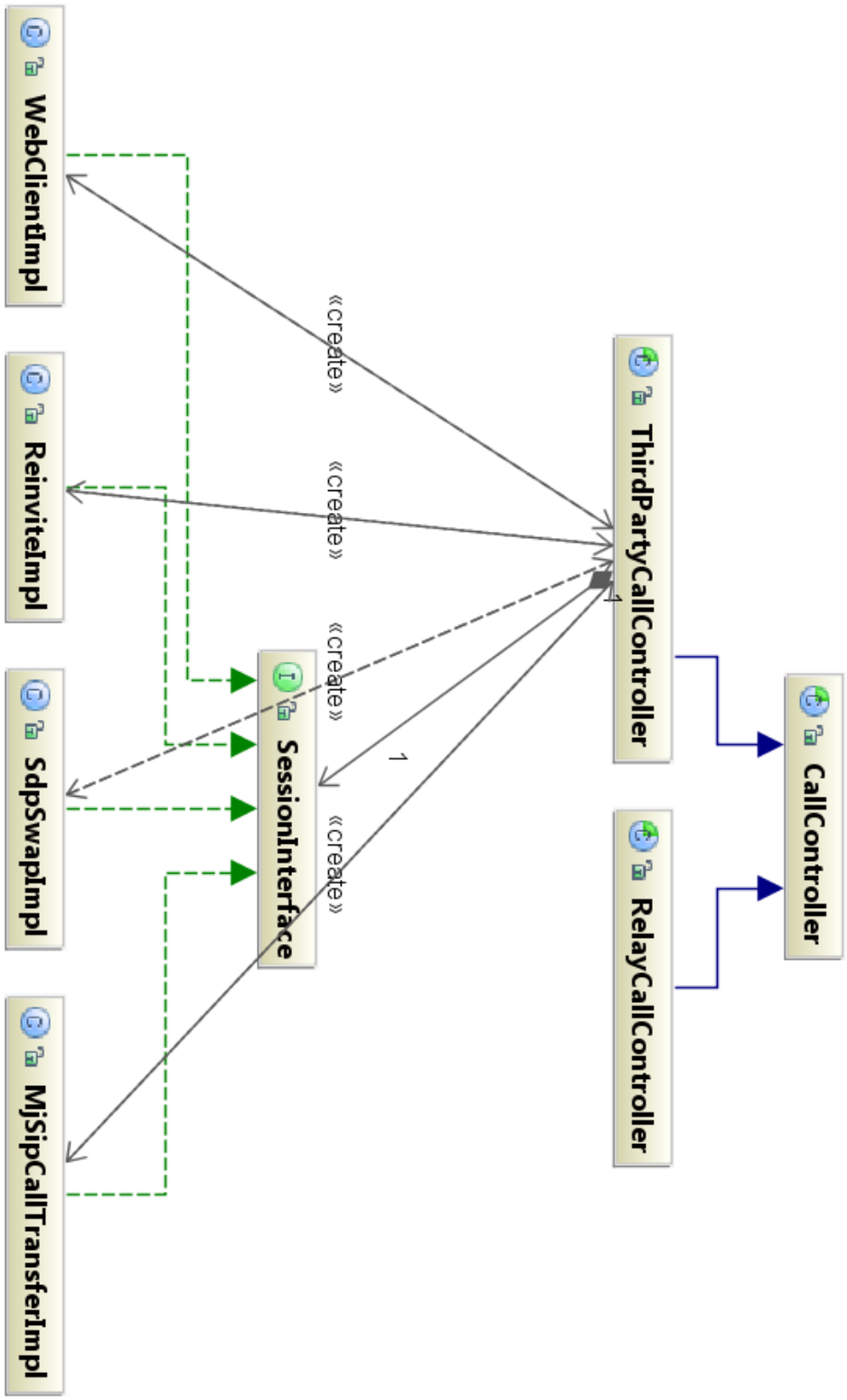

Figure 6.4. Call Controller class hierarchy diagram

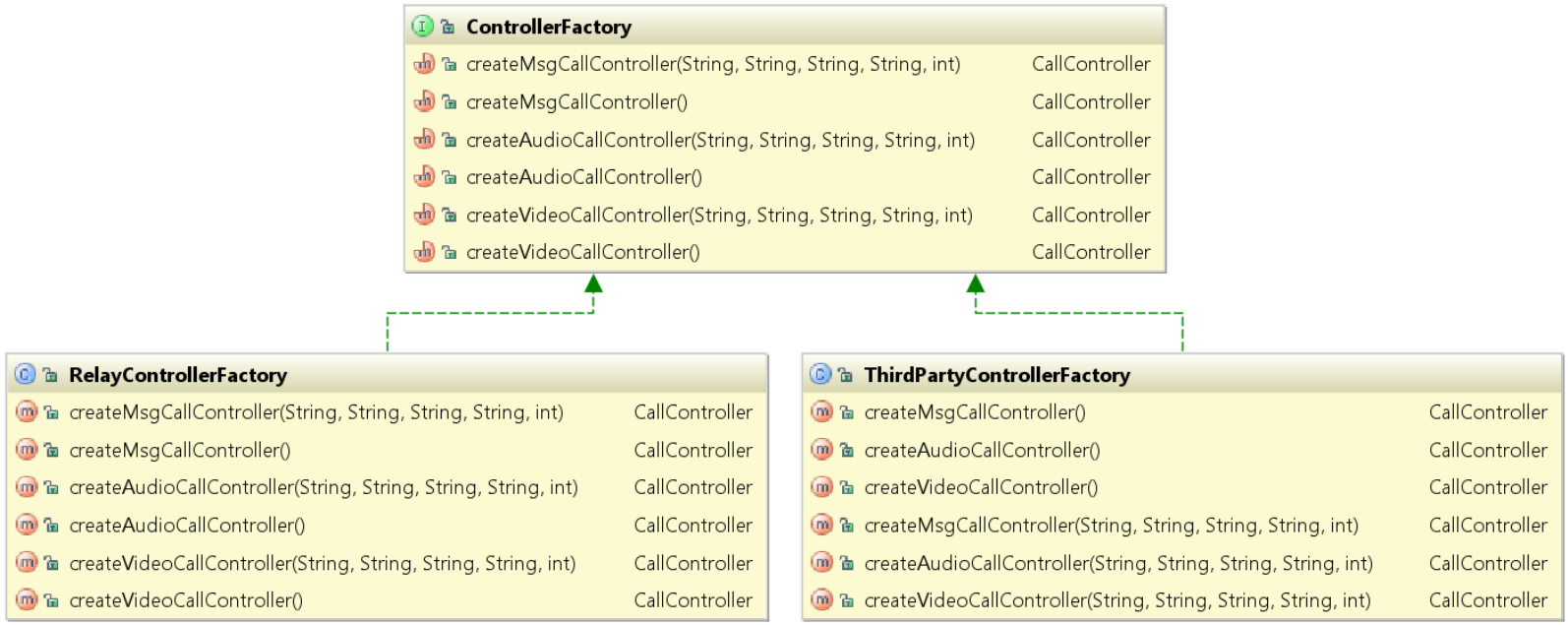

Figure 6.5. Controller Factory class hierarchy diagram

Chapter 7

Web Application

The web application is the main portal of the whole application. It is based on Mobile Front Controller. The role of web application in the whole application is shown in Figure 5.2. Users can use the web application to manage their accounts and VoIP calls. Administrators can use web applications to manage users. The web application can be packaged into a war file and easily to deploy. It only needs a sevlet container, like Tomcat, to run. It supplies two kind of views one is for desktop browser and another is for mobile browser. Besides these two views, the web application also contains a web service interface. This chapter will focus on two browser view. The web service interface will be introduced with detail in chapter 8.

A welcome screen of desktop browser view of Web Application is shown in Figure 7.1.

7.1 Architecture of Mobile Front Controller 3

"Mobile Front Controller (MFC) is a light-weight Java EE web application framework for creating web applications for web browsing and mobile browsing." [37] It was developed by Peter Yeung and Pär Johansson from Ericsson Developer Connection, Ericsson [37]. The mobile front controller uses a sevlet to handle http request, and redirect request to different kind of view. All views share a same logic. An overview of Mobile Front Controller used by a web application on a Java EE web container is shown in Figure 7.2.

"MFC is used on top of a Java EE web container, and does not require any other framework, such as JSF, Struts, etc. MFC does the following:

- *Detects and selects appropriate views based on HTTP request headers. A view is a subdirectory that, for example, corresponds to a markup language such as*

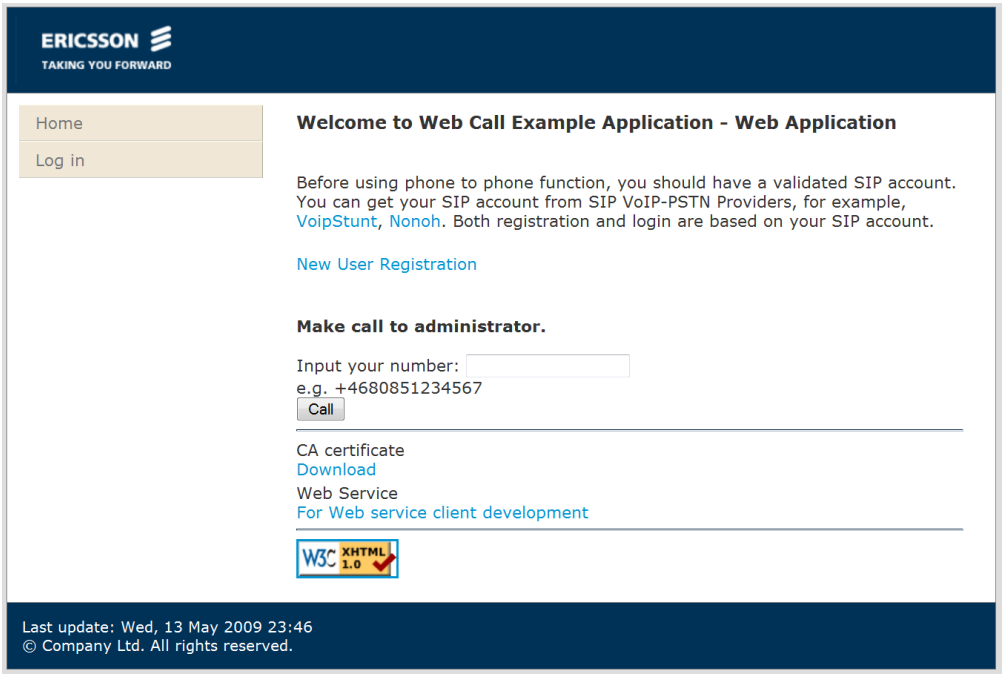

Figure 7.1. Welcome page of desktop browser view of web application

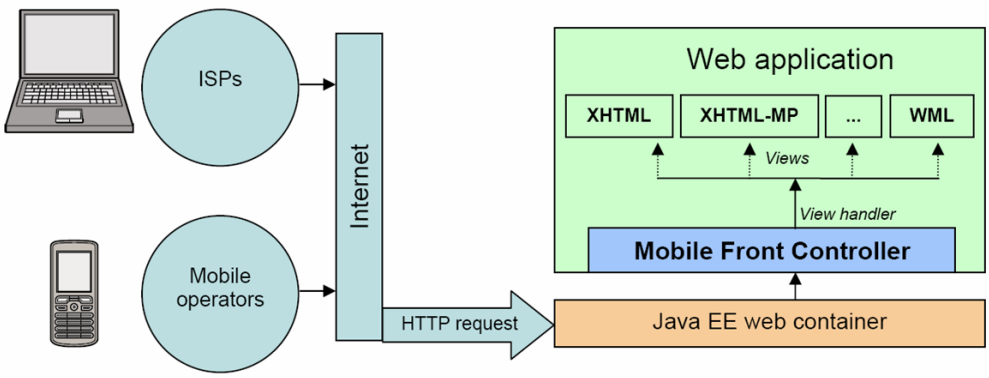

Figure 7.2. An overview of Mobile Front Controller used by a web application on a Java EE web container. (Figure taken from *Mobile Front Controller Developer's guide for software version 3.1*[15])

XHTML, XHTML Mobile Profile (MP) and WML (see Figure 7.3). The way MFC detects and selects views is customizable using view handlers.

- *Shares UI logic among different views, for example, web and mobile browsing. This is done using action commands, which are classes that contain an execute method that is executed when a URL with, for example, the URL pattern *.do is called (for example http://localhost:8080/mfc-basic-demo/xhtml/Print.do).”* [15]

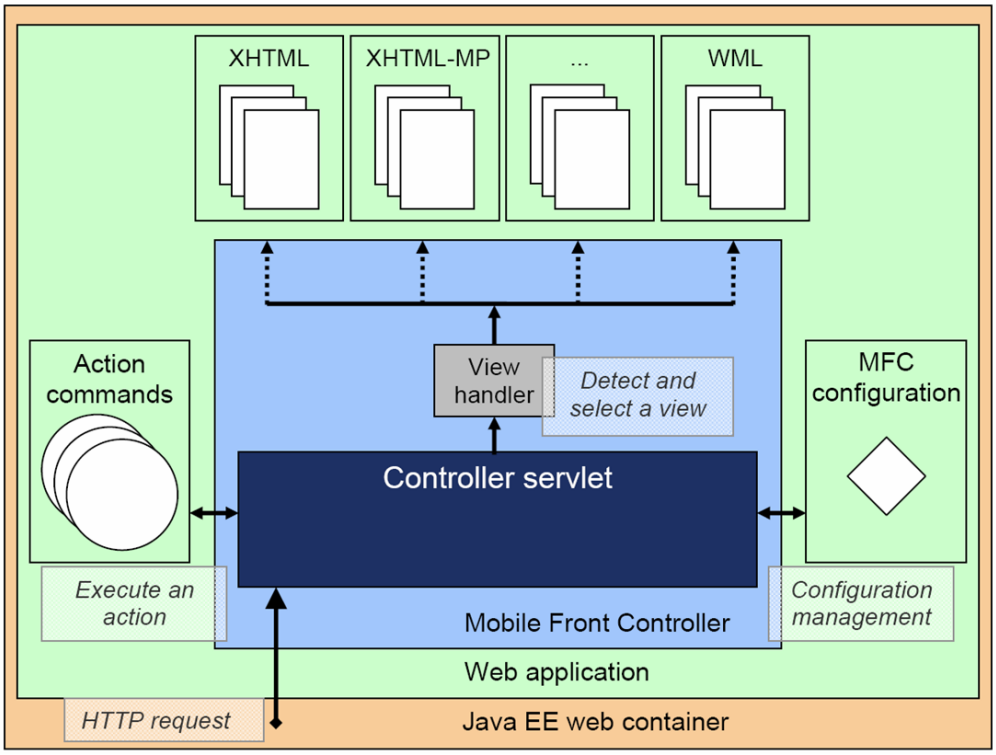

Figure 7.3. The architecture of Mobile Front Controller (Figure taken from *Mobile Front Controller Developer's guide for software version 3.1* [15])

The version of Mobile Front Controller used in Web Call Example Application is the latest version, v3.1.

7.2 Dual-view

As described in the section 7.1, Mobile Front Controller supplies multi-view function. Web Call Example Application is developed with two views, one is desktop

browser view and the other is mobile browser view. The will be illustrated separately in subsection 7.2.1 and 7.2.2.

7.2.1 Desktop browser view

The Mobile Front Controller supplies a function of select view. According to *Mobile Front Controller Developer's guide* [15], if the user accesses web application via a desktop browser, the desktop browser view (XHTML view) will return to user. The desktop browser view is a normal web site that follows the standard of The XHTML 1.0. Extensible HyperText Markup Language (XHTML) is a markup language for web pages which developed by W3C HTML Working Group. It is widely used as a standard language of web page on Internet. An example page (welcome page) of desktop browser view is shown in Figure 7.1.

7.2.2 Mobile browser view

According to *Mobile Front Controller Developer's guide*, if the user accesses web application via a mobile browser, the mobile browser view (XHTML MP view) will return to user. The mobile browser view is a normal web site that follows the standard of XHTML MP 1.1. XHTML Mobile Profile (XHTML MP) is a computer language standard designed specifically for mobile phones. It is developed by an example page (welcome page) of mobile browser view is shown in Figure 7.4.

7.3 Site structure

The web set structure contains three levels. `Base` → `Protected` → `admin`. Both desktop browser view and mobile browser view use the exactly same structure.

Under base directory, there is welcome page, user register page and login page. The pages on this level are public opened. There are no restrictions on requesting these pages.

All pages in protected directory are authenticated resources. To visit these pages, the user has to login first. Both user and administrator can request this kind of resources. Under user panel, user can freely edit information, e.g., user phone number, contact book, change password, VoIP provider account and recent calls. Users can make VoIP calls by clicking the link of Phone to Phone Call under user panel. The pages under admin directory are used for managing user information. Only user who has a role of administrator can access pages under admin directory. Administrator can modify user information or even delete users. For more details about security, please refer section 7.11.

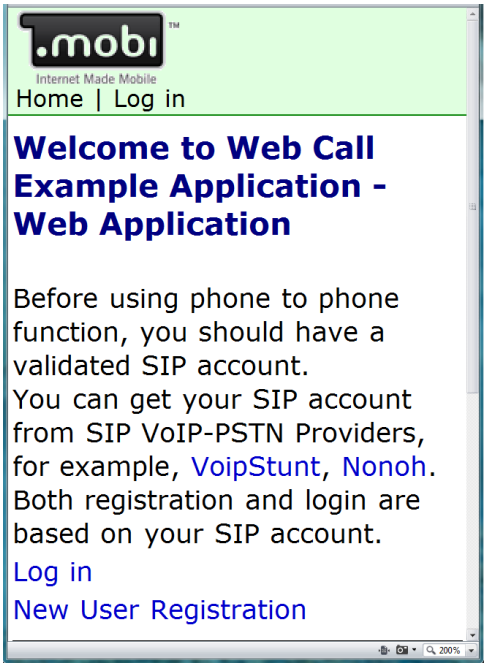

Figure 7.4. Welcome page of mobile browser view

7.4 User action

7.4.1 User Registration

To use the Web Call Example Application, a user must register him on the web site. It is quite easy and simple to register and there are only three steps:

1. Navigate to the web site URL and open the welcome page. There are user registration functions on both desktop and mobile views. So users can register either from a computer or a mobile phone.

2. At the welcome page. Click the link to New User Registration.

3. Fill the form, include user name (in the format of email address), password and confirm password.

7.4.2 User panel

The user profile information menu is shown in Figure 7.5. User can edit his phone number, contact book, password, VoIP provider accounts and recent calls log.

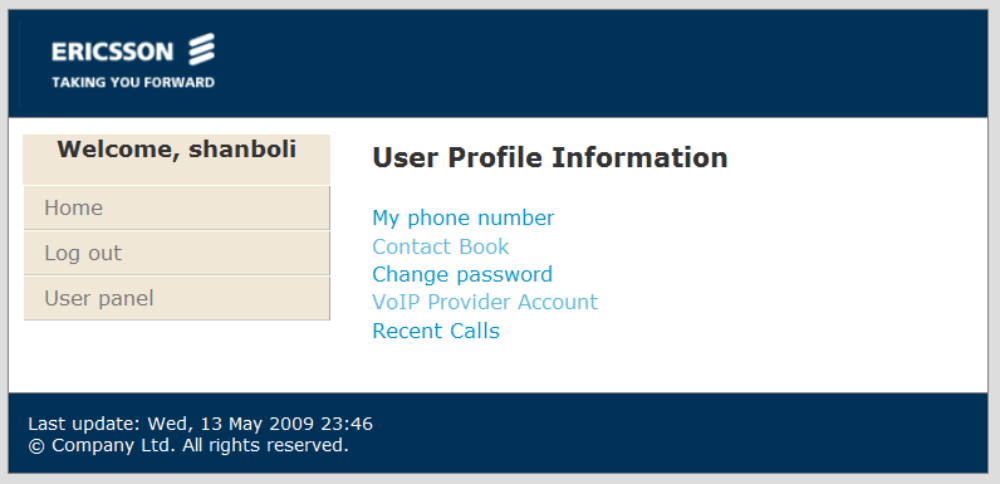

Figure 7.5. User profile information page of desktop browser view

7.4.3 VoIP service provider account

To make a VoIP call, the user must have a VoIP Provider Account first. Figure 7.6 shows how to register a VoIP provider account. To get the view of adding VoIP provider account, user should go to user panel → User Profile Information → VoIP Provider Account → add new VoIP provider account. Follow the example in Figure 7.6 and input the VoIP account information. The outbound proxy should in a format of SIP_PROVIDER_URL:PORT_NUMBER. If no port number specified, the port `5060` will be used as defined in RFC3261 [29].

Web Call Example Application supports multi-account. This means as a user, you can have more than one account and choose a cheaper one when you make VoIP calls.

7.4.4 Phone-to-Phone call

The integrated web application offers phone-to-phone function for registered users.

Figure 7.7 is the user panel of desktop browser view. Users can either go to user profile Information page or the phone-to-phone dialing page. To make VoIP calls, the user just need click the link of Phone to phone call.

In the phone-to-phone call page, which is shown in Figure 7.8, users can input their own number, or use the default numbers, which is set in “my phone number” of user profiles information. And input the destination phone number, or select from recent calls or the contact list. The client number could either be SIP address or a complete phone number. If users have more than one VoIP provider account, they can use the drop down list of VoIP provider account and choose an account

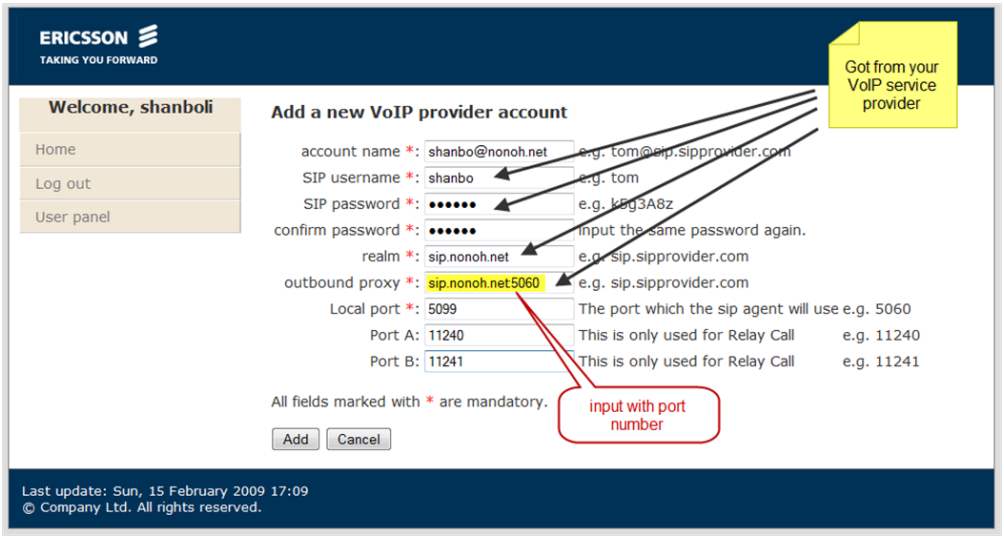

Figure 7.6. Register VoIP service provider account

Figure 7.7. User panel desktop browser view

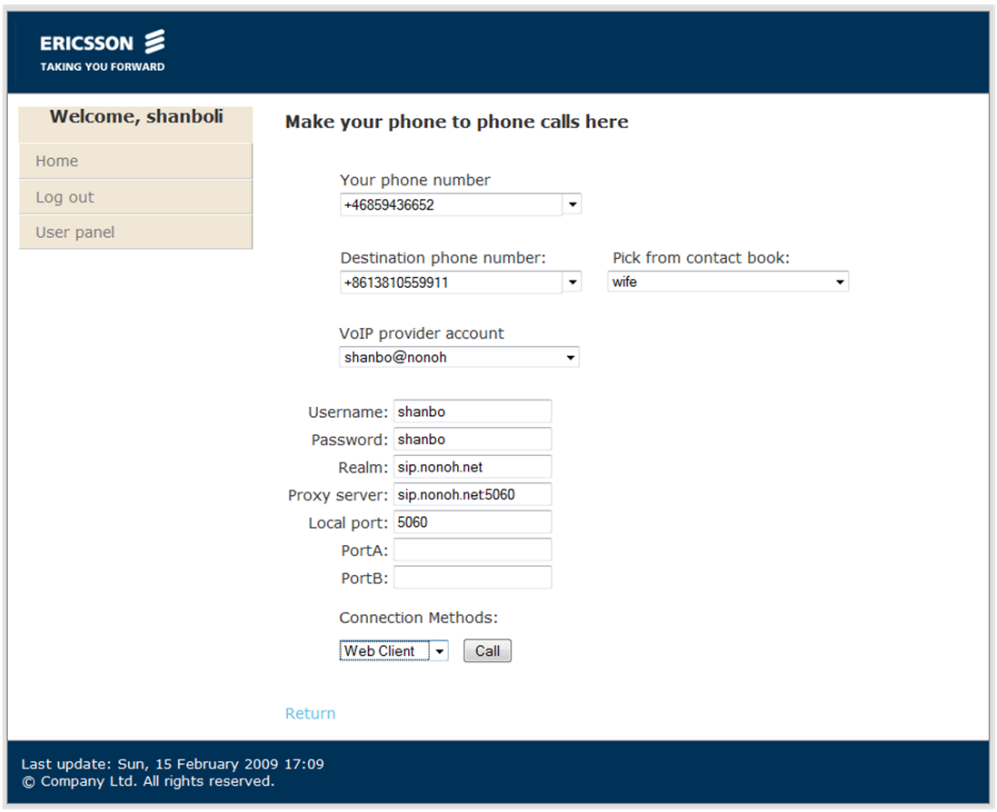

Figure 7.8. Phone to phone call

as their wish. Different VoIP service provider may supply different rate for different destination. So the multi-account function can help user on saving money by choosing a cheapest provider. This page uses the technology of Ajax to dynamically show the information of VoIP account. More Ajax technologies will be introduced in section 7.8.

There is a drop down list of call methods. Five different implementations of connection methods will be illustrated at that list. They are Relay Call, Call transfer, Re-invite, SDP swap and Web Client. The differences of connection methods is discussed in chapter 4 and chapter 10. It is recommend that by default user uses the method of Web client.

After user clicked "Call" button, the page will be directed to call state page. On call state page, users can see the phone numbers of two sides, call state, or even cancel the call.

7.5 Administrator action

The administrator actions include user profile management as well as all of user relevant information. Figure 7.9 is the user list page. The administrator can use the button of edit to change the user information as well as user roles.

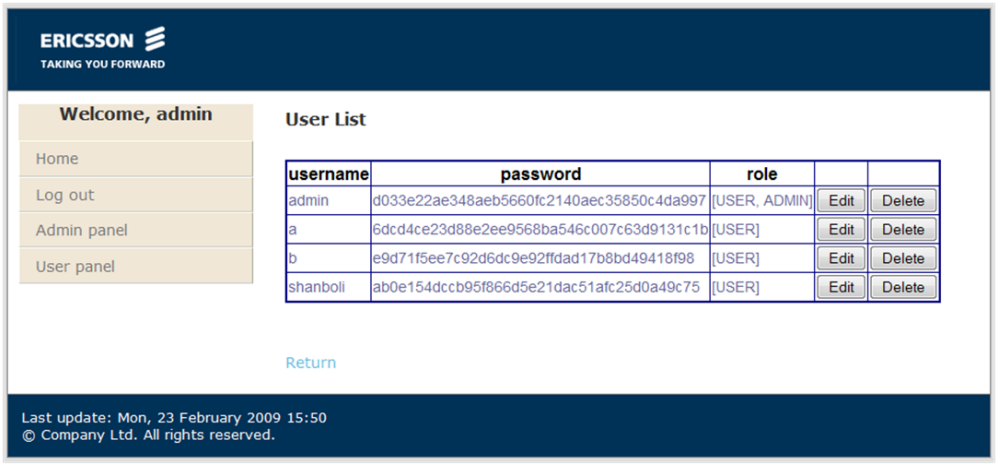

Figure 7.9. User list of administrator panel

Figure 7.10 shows an administrator view of user info. The administrator can update user's basic info, e.g., password and roles. Administrators can change SIP relevant information, such as user's phone number, user's contact list and user's VoIP account info by clicking "show" button in the SIP relevant info section.

The administrator cannot delete the last administrator; otherwise no one can manage the application.

7.6 Validation mechanism

The validation of web application is implemented both at presentation tier and logic tier. On presentation tier, all forms are validated by `javascript`.

7.6.1 Page level

The validation on presentation layer can save the time of communication between browser and server. Web Call Example Application uses `JavaScript` to validate user input. All pages share the same generic validation methods. An example code of "checking if a field is empty" is shown in Listing 7.1.

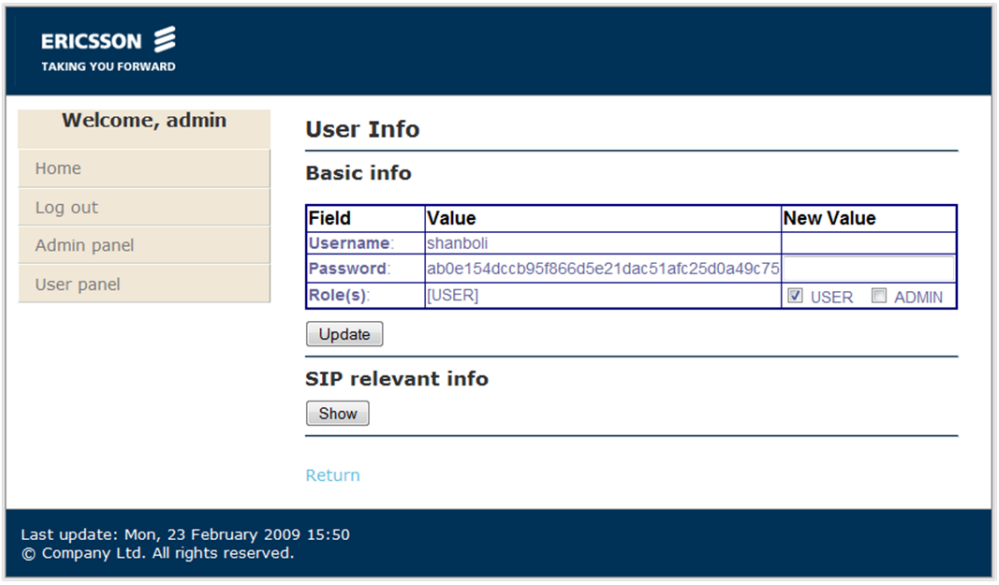

Figure 7.10. Administrator view of user information

7.6.2 Server level

However, to prevent user disables `javascript`, there is also a validation mechanism on server side, which is called server level validation. To make the server response faster, only important or sensitive actions use a server level validation. E.g., the action of changing password has a server level validation of checking if password matched. The action of cancel call is another sensitive function. In addition to the session id, server will also check if this session belongs to the user who wants to cancel it. This will prevent from arbitrary interrupt calls.

Listing 7.1. JavaScript validation code

```
function has_value(field, alert_message) {
    if (field.value == null || field.value == "") {
        alert(alert_message);
        return false;
    }
    return true;
}
```

7.7 Session control

Web Call Example Application supports multi thread actions, that is, many users can make phone calls concurrently. An `enum` class `CallSessions` works as a singleton* and manages call sessions. A class diagram of `CallSessions` is shown in Figure 7.11. It can be seen from picture, the four methods `add`, `getCallInfo`, `remove`, `contains` supply a very convenient way to manipulate sessions. An instance of `CallInfo` contains all information of a phone call, it can be treated as a property pack.

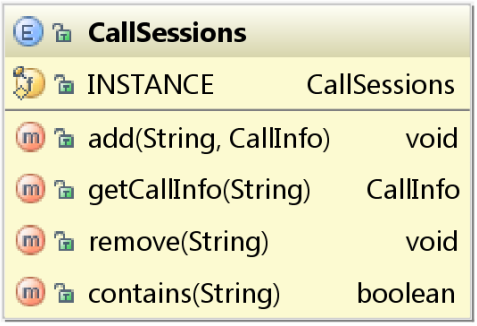

Figure 7.11. Class diagram of CallSessions

7.8 Ajax in web application

Ajax is short for Asynchronous JavaScript and XML. *"Ajax is a group of interrelated web development techniques used on the client-side to create interactive web applications or rich Internet applications."* [34] For more detail about Ajax, please refer to *Ajax : A New Approach to Web Applications* by Jesse James Garrett [9].

There are two places use Ajax in Web Call Example Application. One is at phone to phone call page, which is shown in Figure 7.8. When user chooses a different VoIP service provider account, an Ajax request will be used to fetch account information from server. The fields will be updated automatically after browser gets new account info. And there is no refresh of the page.

Another place that uses Ajax is the call state page. The call state page shows a real time state of a call session. There is a `JavaScript` function named `updateState` sends ajaxGet request to server to check the session state in every minute. An action named `LoadCallState` at server side will communicate with `CallSessions` which described in section 7.7, and response browser with the latest state of that session.

*According to *Design Patterns* [8], Singleton is a design pattern that used to restrict instantiation of a class to one object.

7.9 Java ME helper

Java ME helper is a servlet which responses a content of JAD file to mobile devices. It is not simply load a JAD file and send content to client. Instead, it automatically generates the content of JAD file on demand. Normally, a JAD file contains the description of JAR and MIDlets. Besides the standard properties, the Java ME helper will also collect the user name and hashed password and write it into JAD. The username and hashed password will be used in communication of Java ME Client which will be introduced in chapter 9.

7.10 Database

The database of Web Call Example Application stores all user information. When users login to the web page they will always be asked to input their user name and password. The inputted user name and password will be compared with the ones in database.

7.10.1 Design of database

An EER (Extended Entity-Relationship) diagram of database is shown in Figure 7.12. There are seven tables in the database. They are `user`, `role`, `user_role`, `contact`, `user_uri`, `sip_account` and `recent_call`. Table `user` stores user name and hashed password. Table `role` stores typical roles in the application, which are USER and ADMIN. Table `user_role` is a bridge table that acts as a many to many relationship table of `user` and `role`. Table `contact` stores the contacts informations. Table `user_uri` stores primary phone numbers of user. Table `sip_account` stores VoIP service provider account that is used to register to a proxy and make phone calls. Table `recent_call` stores recent call history.

7.10.2 User database utility

User database utility is originally a part of Web Call Example Application. It was separated from Web Call Example Application and distributed as an example of Mobile Front Controller. The User database utility supplies a light weight java API for manipulation database. The user database utility uses a Strategy Pattern[†] to make user easy to interchange database connection from data source to direct connection.

As long as user creates a database, he can use Web Call Example Application. When the first time deploys, it will check if the database has the tables which it needs. If it is an empty database, Web Call Example Application will automatically

[†]According to *Design Patterns* [8], the intent of the Strategy pattern is: Define a family of algorithms, encapsulate each one, and make them interchangeable. Strategy lets the algorithm vary independently from clients that use it.

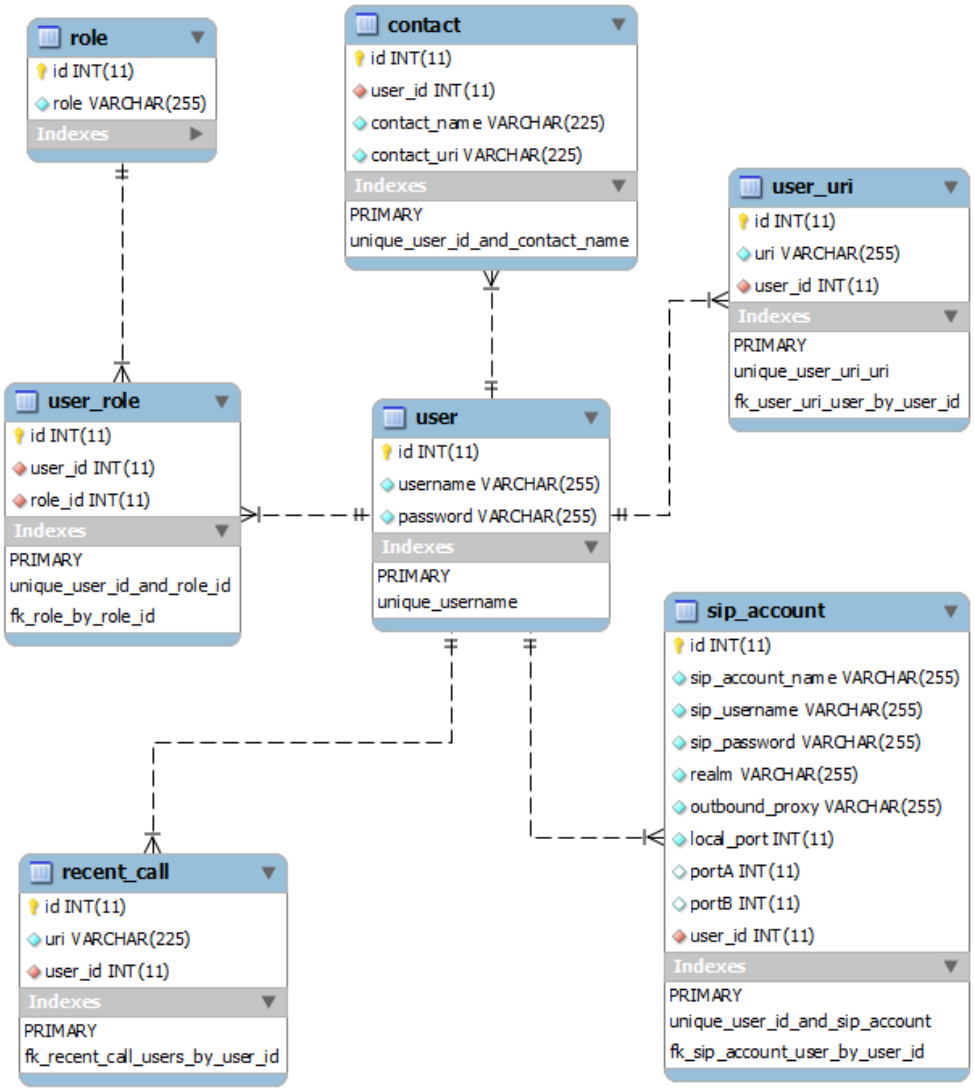

Figure 7.12. EER diagram of database(Diagram generated by MySQL Workbench)

generate essential tables. And the first user of this application will be automatically registered as administrator, which is shown in Figure 7.13

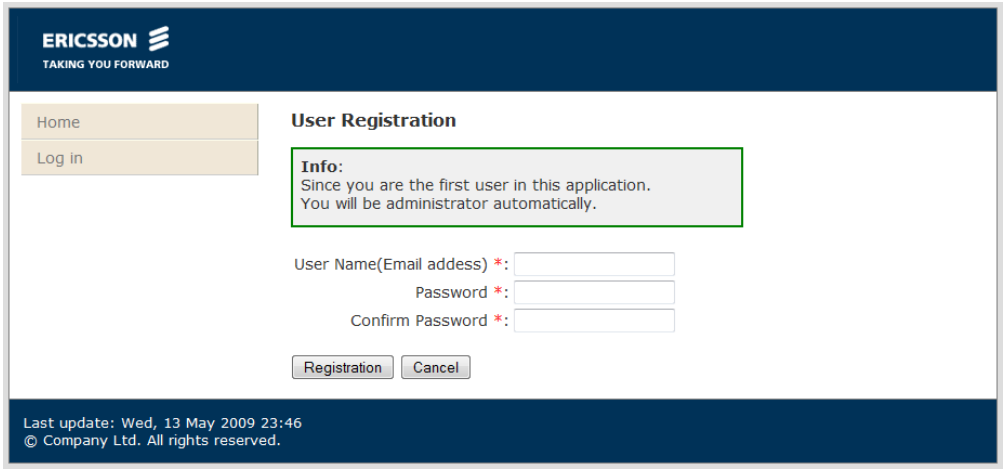

Figure 7.13. First user will be administrator

7.11 Security

There are two parts of resources are in the security constraint domain. One is the user resource and another is administrator resources. They are both in the CONFIDENTIAL level. *"A user data constraint (<user-data-constraint> in the deployment descriptor) requires that all constrained URL patterns and HTTP methods specified in the security constraint are received over a protected transport layer connection such as HTTPS (HTTP over SSL)."* from Java EE 5 Tutorial [14].

Web Call Example Application supports six different ways of password digest. They are MD2, MD5, SHA-1, SHA-256, SHA-384 and SHA-512. The method of digest can be specified in `web.xml`. They are fully compatible with standard servlet containers such as Apache Tomcat. All users' passwords in database are hashed password, even site/database managers cannot get the plain text of users' passwords. The hashed password is also used in operations of web service interface.

Chapter 8

Web Service Interface

THIS chapter focuses on the web service interface of Web Call Example Application. It introduces web service technology at the first section. Then it explains the web service methods that defined in the web service interface. This chapter also mentions how to deploy this web service at the last section.

8.1 Introduce web service and metro

> *"A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP-messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards."* [11]
>
> –W3C

A web service interface of Web Call Example Application gives a most broad way for clients. Chapter 9 describes a Java ME Client of this web service interface. However, it is definitely not the only client for the web service. As long as a client implements the interface, it can use the web call service and database.

The web service interface of Web Call Example Application is based on Metro. According to *The homepage of Metro* [22], Metro is a *high-performance*, *extensible*, *easy-to-use* web service stack. It is supported by Sun Microsystems. For more detail about Metro, please refer to Metro Homepage at https://metro.dev.java.net/.

8.2 SOAP web service

The web service source file is
`sip.components.webapp.webserviceg.WebCallImpl`.

List 8.1 shows a fragment of the source code that calls operation in Sip Call Component.

Listing 8.1. Web Service implementation (fragment)

```
...
import sip.components.core.controller.CallController;
...

@WebService(name = "WebCall")
public class WebCallImpl {

...
@WebMethod
@WebResult(name = "callID")
public String call(@WebParam(name = "clientA") String clientA,
                   @WebParam(name = "clientB") String clientB,
                   @WebParam(name = "sipAccountName")
                             String sipAccountName,
                   @WebParam(name = "implType") String implType,
                   @WebParam(name = "username") String username,
                   @WebParam(name = "password") String password)
                             Throws PersistenceLayerException,
                             AuthenticationFailedException {

...
...
CallController controller = cf.createAudioCallController();
controller.setClientA(clientA);
controller.setClientB(clientB);
controller.register();
controller.start();
...
}
}
```

The class of web service uses the `@WebService` annotation to set the port name, service name and the target namespace. The web service methods use the `WebMethod` to annotate the operation name and actions. The parameter of the method is annotated with `@WebParam`. There are nine web service methods in the implementation Web Service:

- `getUserURIs`: Get a list of user's URIs. The URI (Uniform Resource Identifiers) can be a phone number with international dialing codes or in the form of `sip:USERNAME@HOST:PORT`. Username is the user name registered at host. Host is the domain name or IP address of host. The URIs in this list will be treated as caller/clientA.

- `getContacts`: Get a contact list. The items in the list are instances of `ContactBean`. A `ContactBean` has a `contactName` and a `contactURI`.

- `getRecentCalls`: Get a recent calls list. The recent call is just like a call history with limited numbers. In Web Call Example Application, the default size of recent call list is 5. It could be specified in `web.xml` file.

- `getSipAccounts`: Get accounts for sip providers. The registration of sip provider's account can be done at both desktop and mobile browser views.

- `getImplTypes`: Get the implementation types list of call method. As mentioned in chapter 4, there are five types of implementation. They are Relay Call, Call transfer, Re-invite, SDP swap and Web client.

- `call`: Establish phone calls. In this web service method, the user's phone number and destination phone number will also be stored in database. A Ring Buffer is used to store the recent calls.

- `cancelCall`: Cancel a phone call. To cancel the call, client has to pass the username, password and call ID as the parameter. Server will check if this call session belongs to the user, before cancel the call.

- `getState`: Get a phone call's state. The state could be `INITIALIZED`, `UNREGISTERED`, `REGISTERED`, `DIALING`, `OUTGOING_A`, `OUTGOING_B`, `REGISTER_FAILED`, `ON_CALL`, `CALL_FAILED`, `IDLE`, `ERROR`, `CLOSED` or `REFUSED`.

- `syncContacts`: Synchronize contacts. The method keeps contacts list synchronized on clients and server. If same name with different phone number happens, the number in client will be kept. This method can be used after load/import contact from a third party contact book. E.g., the native contact book in mobile phone. The usage of synchronize contacts will also be explained in chapter 9.

All methods except `getState` and `getImplTypes` require a username and password. The password is in the format of digest. So even if the communication package is intercepted, the third party will not know the plain text of user password. For details about the method, please refer the source code in `WebCallImpl`.

8.3 Deploy of Web Service

To deploy web service, the service needs portable artifacts. In Web Call Example Application, the portable artifacts are generated by art (annotation processing tool). According to *Metro 1.5 FCS - APT* [21], the apt tool is a tool which distributed with Metro. It generates portable artifacts used in JAX-WS services. The execution of apt is integrated into ant target in `build.xml`.

The web service listener is setted to `WSServletContextListener` in `web.xml`. The url-pattern and implementation is specified at `sun-jaxws.xm`. The two configuration files are located at folder of `WEB-INF` in distributed package(WAR package).

Chapter 9

Java ME Client

The Java ME Client is a client of web service interface which described in chapter 8. It implemented all the interfaces of web service of Web Call Example Application. Besides the web service interface, Java ME Client also provide some convenient function such as load contact book from mobile phone and synchronize it with server.

9.1 Architecture

The architecture of Java ME web service client is shown in Figure 9.1. It can be seen from the diagram that there are three layers in the Java ME client. They are, from bottom to top, Java ME API, Business Logic and User Interface.

The Java ME API layer is the standard API set for Java ME platform. To meet the requirement of installation, the mobile device mast have a support of `JSR 172` (J2ME™ Web Services Specification) [24] and an optional support of `JSR 75` (PDA Optional Packages for the J2ME Platform) [25].

In business logic layer, there are several components that control the logic. A very important one, which is also the core of Java ME client, is the `Web Call Client`. The Web Call Client bases on `Web Call Client Stub` which is the client stub of web service interface of Web Call Example Application. The detail of Web Call Client and Web Service Client Stub will be discussed separately in section 9.5 and 9.2. The Web Call Client also uses a component named `Record Store Manager` which will be described in detail in section 9.3. The last component is `PIM Contact Helper`. It is a utility which helps to load contact book from mobile device.

In user interface (UI) layer, there are two implementations of UIs. The details of user interface will be shown in section 9.6.

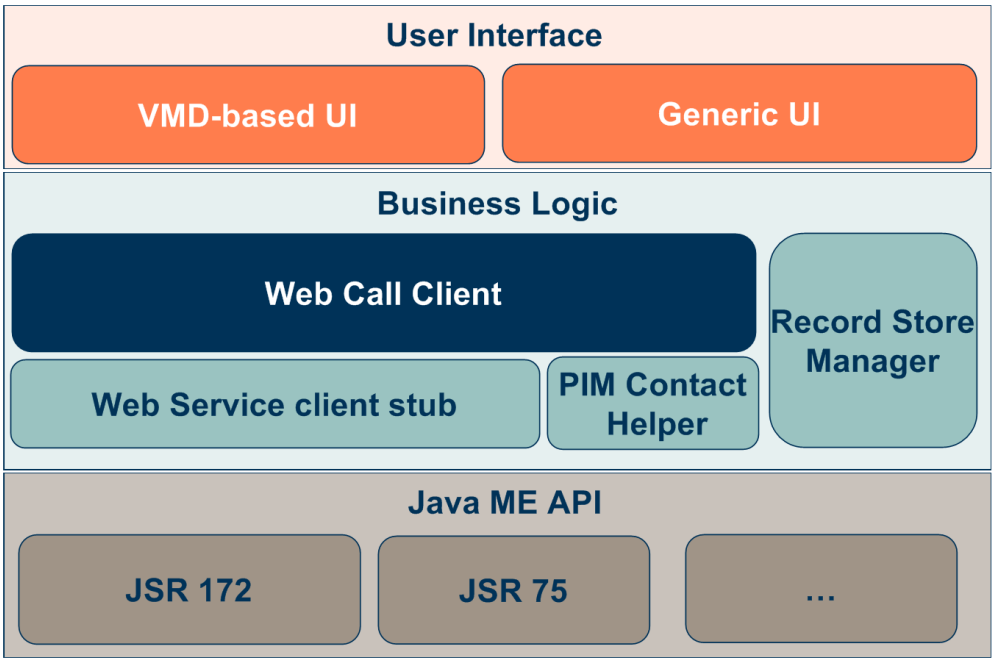

Figure 9.1. The Architecture of Java ME Client

9.2 Web Service Client Stub

The web service client stub is a stub of web service interface which described in chapter 8. It is generated by a wizard from NetBeans IDE. NetBeans is an open-source and free IDE sponsored by Sun Microsystems. The generation wizard is shown in Figure 9.2. The client stub is based on JSR 172, so only the handset which supports `JSR 172` can run the Java ME web service client.

Once the wizard is finished, NetBeans will automatically generate some classes that wrap the stub and make the operation of web service easily. And it communicates with web service server via SOAP protocol. The web service client stub is designed to be separately from the core logic of Java ME client (Web Service Client). This makes the lower stack of web service exchangeable. As just mentioned above, the client must be installed in a `JSR 172` enabler. The project team is planning to define a RESTful web service and a RESTful web service client to replace current SOAP based web service client stub. Since The RESTful web service only need a HTTP connection which is widely supported by modern mobile phones, the Java ME client will be installed on more devices by then.

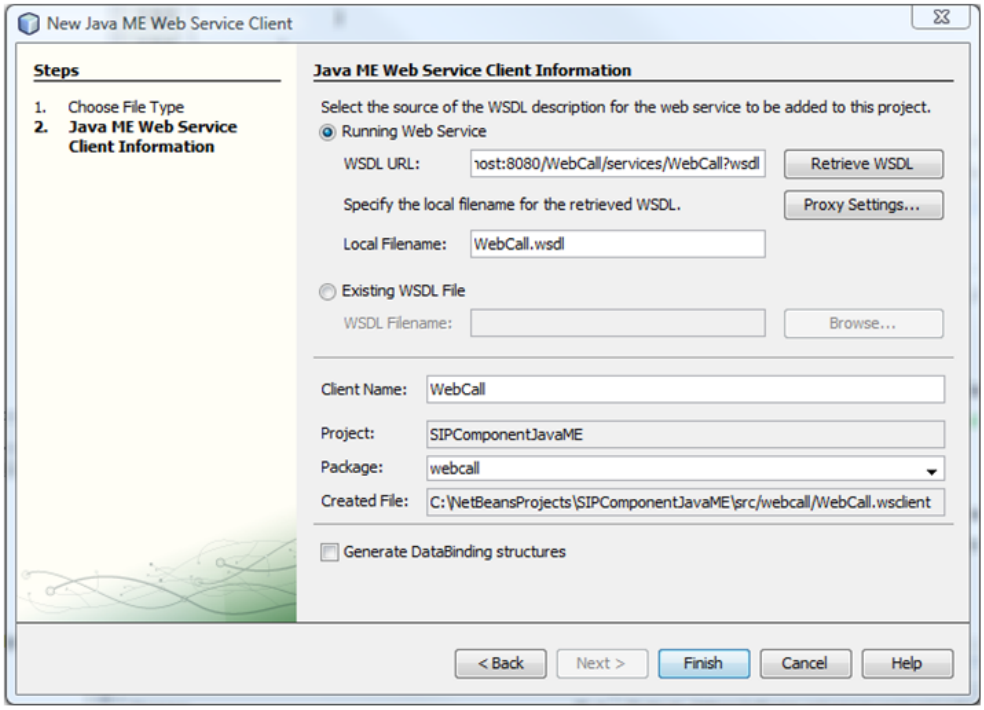

Figure 9.2. Netbeans Java ME Web Service Client Wizard

9.3 Record Store Manager

Record Store Manager is used for reading and writing the data from/to Java ME *record management system* (RMS). It extends an abstract class which named `AbstractRecordStoreManager` which is a standalone component which supplies a very convenient way to interactive with RMS. The `AbstractRecordStoreManager` wraps the action of saving and loading data to/from RMS. Two methods `setData()` and `getData()` handles everything. The `AbstractRecordStoreManager` can be not only used in Java ME Client of Web Call Example Application, but also other Java ME applications. A class hierarchy diagram of `RecordStoreManager` and `AbstractRecordStoreManager` is shown in Figure 9.3.

9.4 PIM Contact Helper

PIM Contact Helper provides access to contact list of *Personal Information Management* (PIM) data on J2ME devices. The PIM API is designed for a widely use and try to compatible with most of Java ME enabler. But it is lack of user friendly. The

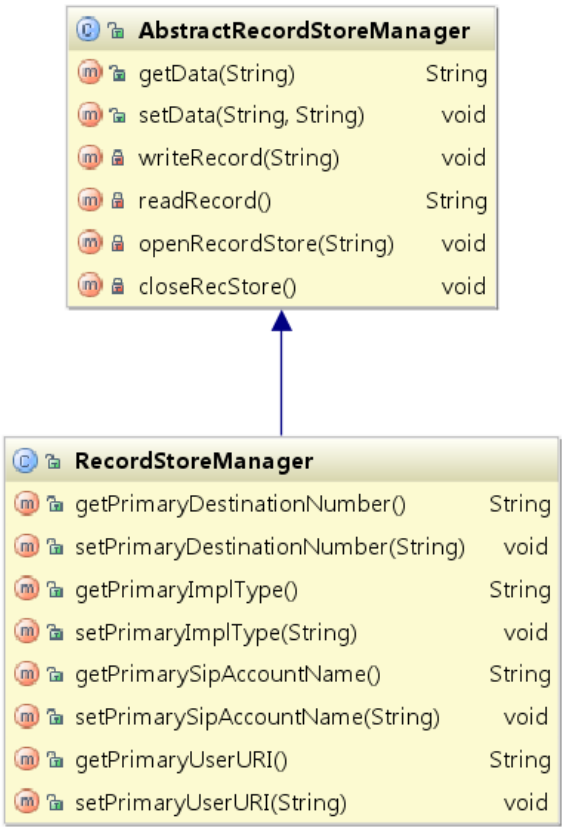

Figure 9.3. The Class Diagram of Record Store Manager

PIM Contact Helper wraps some APIs of JSR 75 and makes it very easy to load contact lists from mobile phone. Like Record Store Manager, PIM Contact Helper is also a standalone component. It can also be used in other Java ME applications. To use it, the mobile device must support `JSR 75` [25]. When first the time use it, there will be pop up option window to ask if user want this application to read contacts list as shown in Figure 9.4. As long as user approved, a phone contacts list will be shown on screen as shown in Figure 9.5.

9.5 Web Call Client

The Web Call Client component bases on the Web Service Client Stub. As described in section 9.2, The Web Service Client Stub implements only the client interface. And the `Web Call Client` manages the business logics, e.g., the load of last used

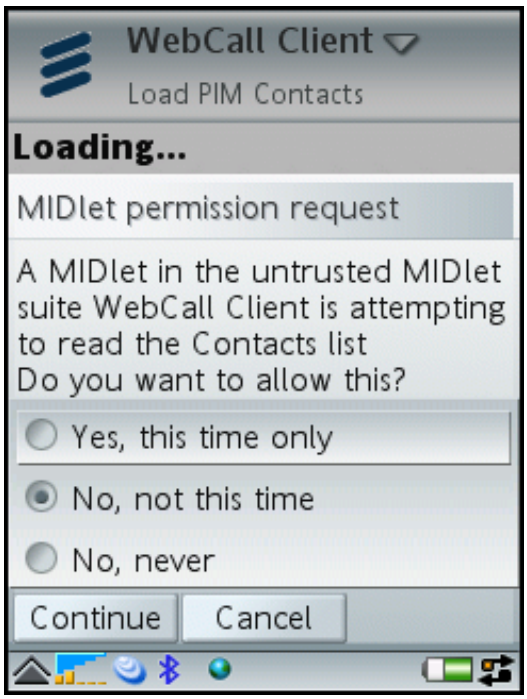

Figure 9.4. Load PIM Contacts

configuration, the recent call list and logic of synchronization. The Web Call Client can be treated like an adapter between UI and Web Service Client Stub. It collects phone call information from UI and set them as parameters of Web Service Client Stub.

9.6 User Interface

9.6.1 VoIP Call Form

A VoIP call form is shown in Figure 9.6. On top of that form, there are two input fields, *your phone number* and *destination phone number*. Under that, where are two drop down lists. One is *VoIP provider account*, and another is *call method*. The contents of these two lists are fetched from web service interface. The VoIP provider is used to choose a VoIP account that user wishes to use. Users can also change the content of VoIP provider drop down lists via the desktop browser view or the mobile browser view. See how to configure the VoIP provider account, please refer to section 7.4.3. Below that, there is a drop down list to choose call methods. There are five different methods to use. They have similar effects but not the same

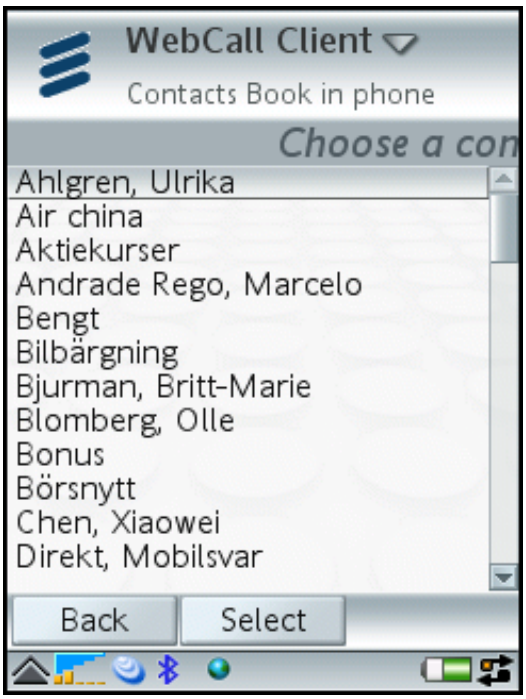

Figure 9.5. PIM Contacts List

way of establishing connection. The differences of call method are introduced in chapter 4. A screen shot of call method drop down list is shown in Figure 9.7. The left bottom of this page is the "Call" button which will connect the phone call when user click it. Next to the call button is the "Exit" button.

The Java ME client uses the RMS (Record Management System) to store the last call information include caller, callee, VoIP provider and call method. When the Java ME client starts again, it will read the record from RMS and automatically set call method as the same as last time used. The interaction with RMS is accomplished by Record Store Manager which is described in section 9.3.

On the VoIP call form menu, there are five screen menu items, PIM Contacts, My phone number, Contacts, Recent Call and Sync which is shown in Figure 9.8. The functionality of five menu items as well as phone call will be introduced within follow subsections.

9.6.2 PIM Contacts

PIM contacts supplies feature of reading contacts from Personal Information Management (PIM) data from cell phone. This function is implemented by the com-

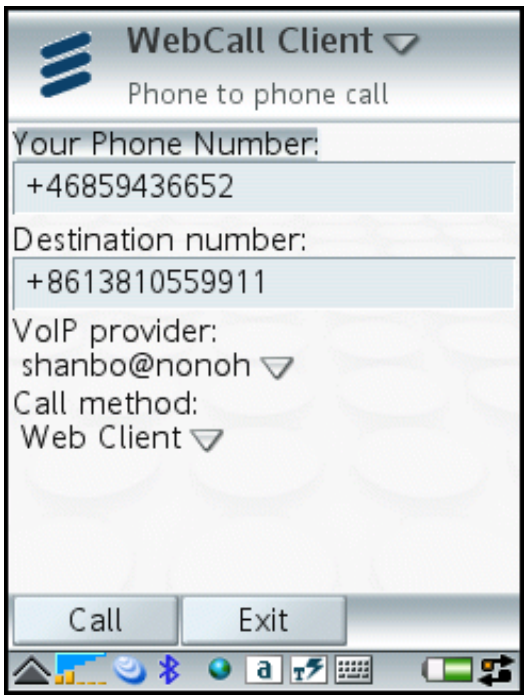

Figure 9.6. VoIP call form of Java ME Client

ponent of PIM Contact Helper which is described in section 9.4. In the view of "Contacts Book in phone" there is also another command to save your contacts in contact book of Java ME client. As long as user uses the synchronize function, the new contact will also be stored in the remote database of web application.

9.6.3 My Phone Number

My phone number is the phone number you wish to call from. It could your current phone or a fixed telephone. You can have more than one phone number saved in the database of server side. Click one of "my phone number", the number will be set as your phone number in the phone to phone call form.

9.6.4 Contacts

The contacts here refer to the contacts book in the web application. It is not same as the PIM contact list in phone memory. However, you can save contacts from contacts book in phone to contacts book in web application by the function mentioned in section 9.6.2. The number of selected contact will be set as destination number. The view of Contact list is shown in Figure 9.9.

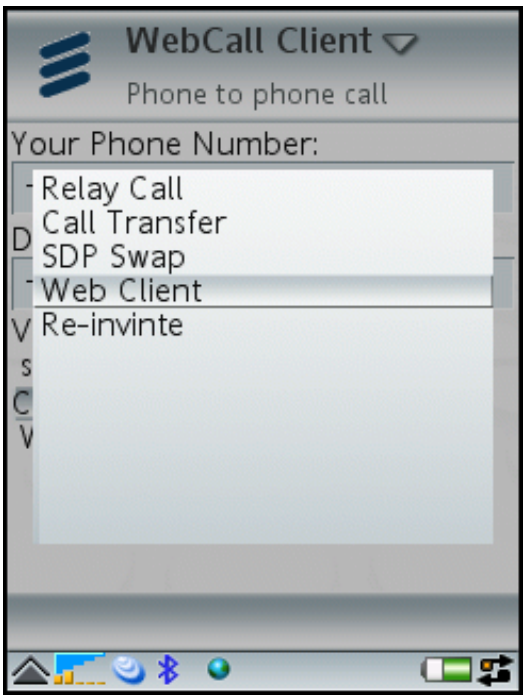

Figure 9.7. Call method drop down list

9.6.5 Synchronize

The synchronize function in phone call form is used to synchronize VoIP provider account and contacts. If same name with different phone number happens, the number in client will be kept. The synchronize function is shown in Figure 9.10.

9.6.6 Phone Call

After the user setup the phone number, destination number, VoIP provider account and Call method, just simply click "Call" button then a session will be established between the user and his friend.

The phone call function in Java ME client is easy and convenient. Every time it restarts, the configuration will be the same as last time he used. It is especially useful when the user always calls the same number. The phone call function is shown in Figure 9.11.

Figure 9.8. Screen menu items of VoIP call form

9.7 Two implementations of UI

9.7.1 VMD-based UI

This Java ME client is first developed with Visual Mobile Designer. The Visual Mobile Designer (VMD) is a graphical interface within NetBeans Mobility that enables you to design mobile applications using drag and drop components. When compile and deploy the Java ME application, The application must include a NetBeans VMD library.

The VMD-based working flow is shown in Figure 9.12. For a larger image, please refer to the *original version** of working flow on web.

9.7.2 Generic UI

To avoid using the VMD library, a generic UI is introduced. It doesn't use any particular class from NetBeans. Change `WaitScreen` to `Form`, change

*http://shanbohomepage.googlecode.com/svn/trunk/master_thesis/chap09/resources/java_me_working_flow.png

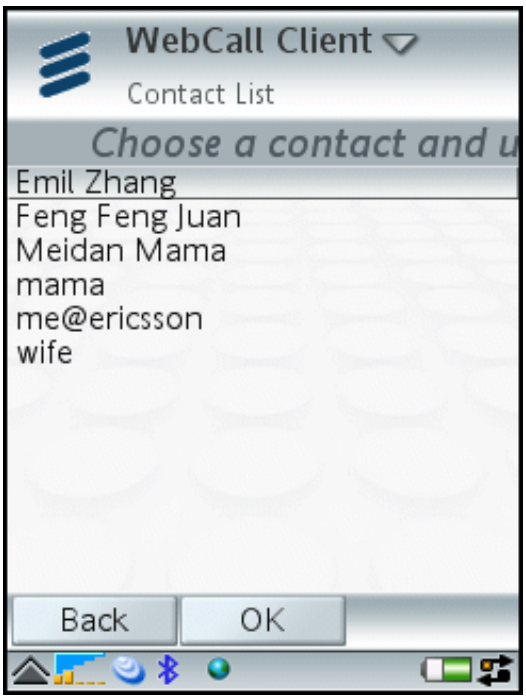

Figure 9.9. Contact list of VoIP call form

`SimpleCancellableTask` to normal method. There are three sub packages in package `sip.components.me.ui`, `nb`, `nb2` and `general`. The package `nb` contains the VMD-based UI. The package `nb2` is a UI which doesn't contain any NetBeans library. The UI in package `general` is the code copied from nb2 and refactored according the package name and Midlet name. All three UIs can be treated as Midlets. They invoke the same logic in Web Call Client and PIM Contact Helper. By default only the UI in general will be compiled.

9.8 Installation of Java ME Client

The installation of Java ME Client could be down via mobile browser. After login to mobile view of web application, go to the user profile page, users will see "Download your JAD file", as shown in Figure 9.13. Click the link to download and install the Java ME client.

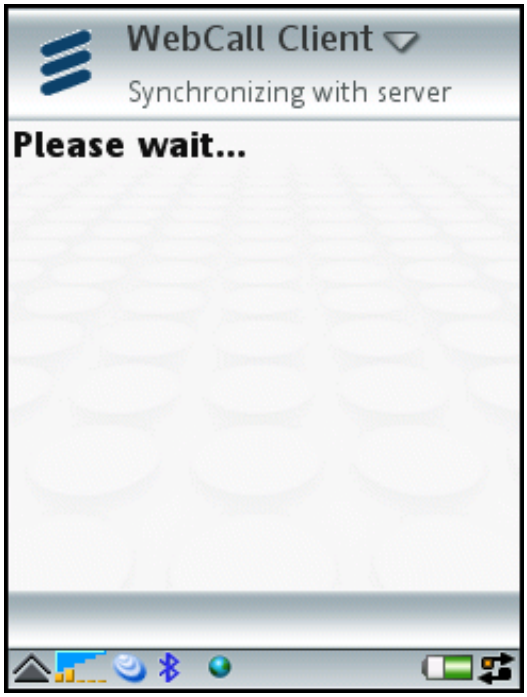

Figure 9.10. Synchronize with server

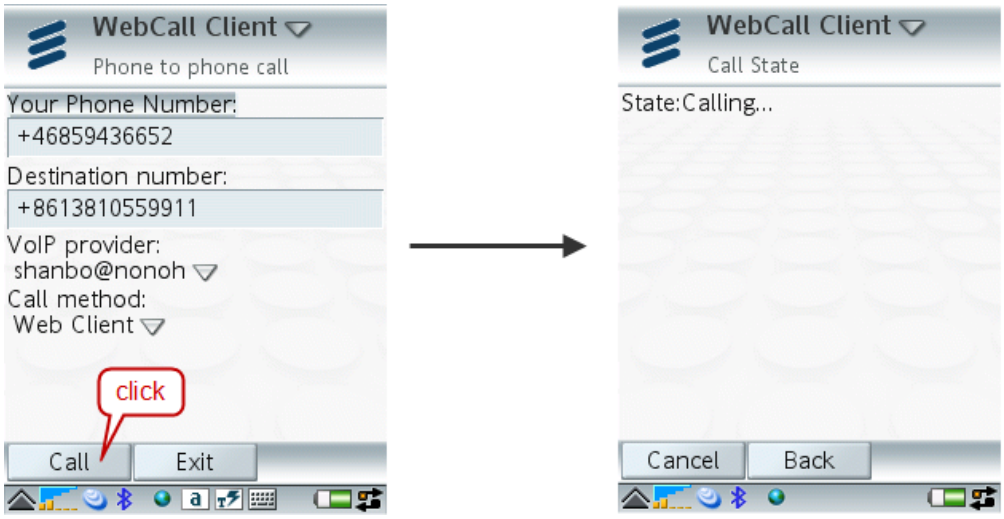

Figure 9.11. Phone call (via VoIP technology) of Java ME client

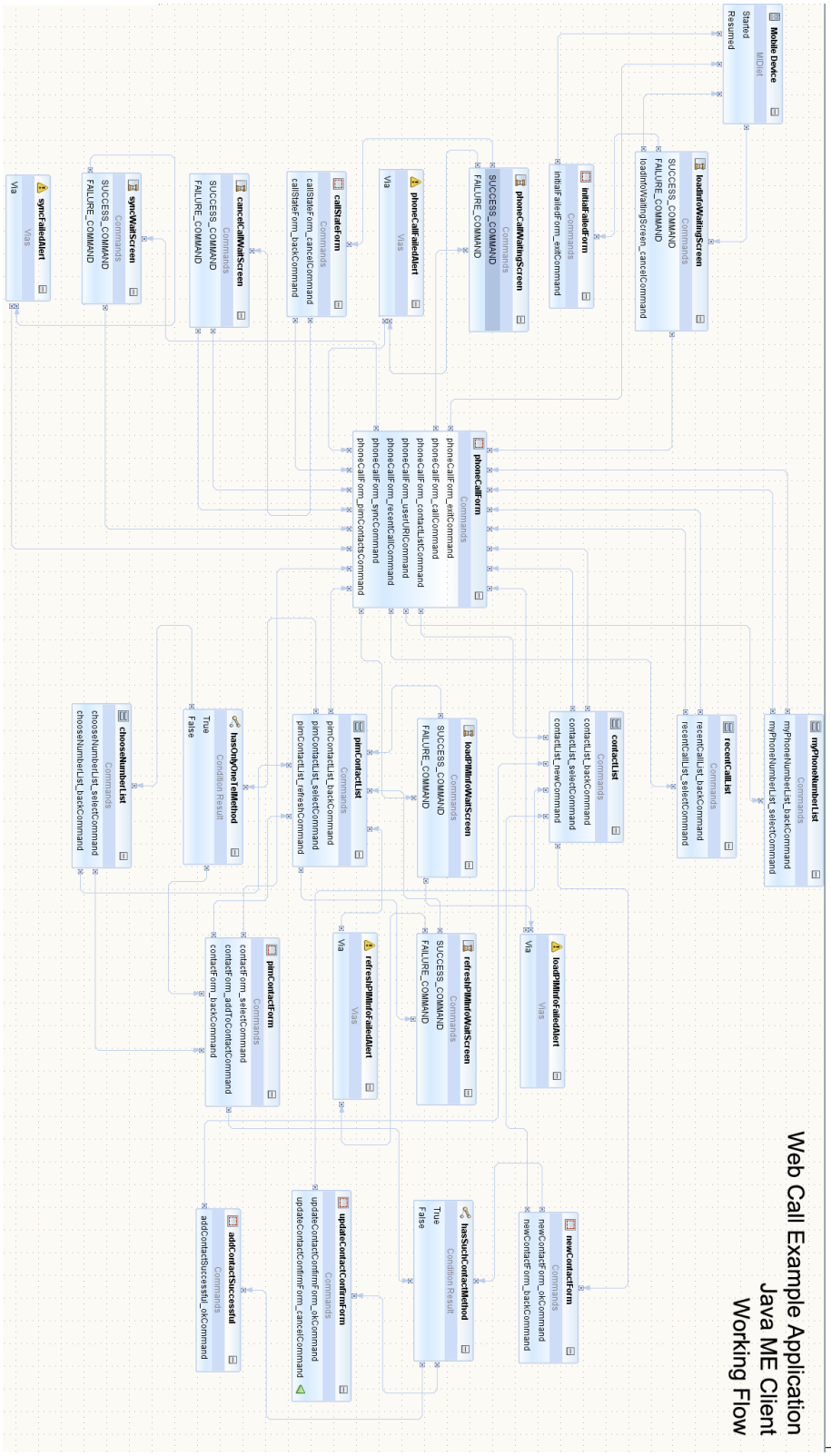

Figure 9.12. Java ME Client Working Flow

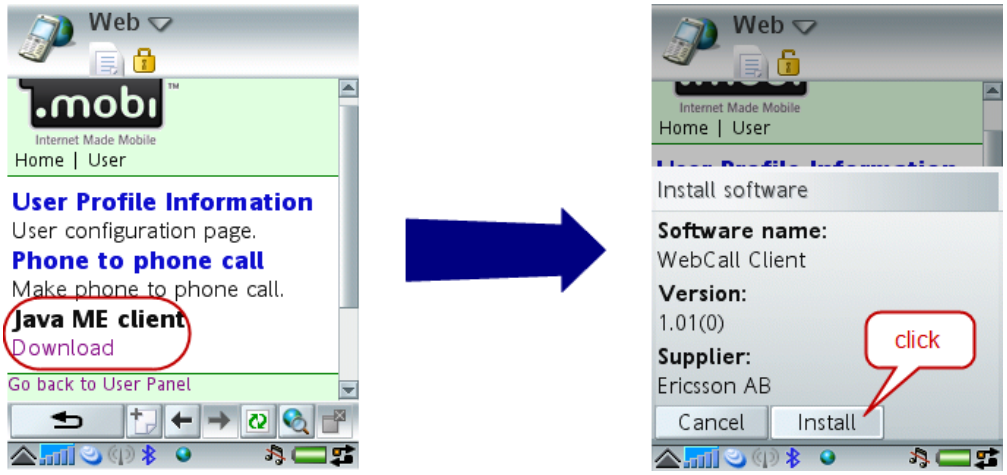

Figure 9.13. Download and install Java ME Client

9.9 Security of Java ME Client

All the operations with web service interface need a user name and a hashed password. Before download the Java ME Client from web application, the user has to login the web site. The user name and password are stored in JAD file as properties. The JAD file is automatically generated by the web application. The functionality of automatically generate JAD file is introduced in section 7.9. The JAD file will be saved in the memory of mobile phone, so users will not need to repeatedly input password when use the Java ME Client.

Chapter 10

Analysis

This chapter will analysis the performance, usability and reliability of Web Call Example Application. A comparison table of Call Method will also be illustrated at the end of this chapter.

10.1 Performance

Since all portal of Web Call Example Application uses the same core that is SIP call component. The evaluation and analysis of performance will focus on the different call method of SIP call component.

10.1.1 Evaluation Environment

Ericsson Service Development Studio (SDS) 4.1 [7] is used as the SIP service provider. *"SDS is the only fully comprehensive tool for development and end-to-end testing of both the client and server side of new convergent all-IP (IMS) applications."* [7] Two clients are Express Talk [31] and PhonerLite [32]. They are both widely used SIP clients. Both clients registered with Ericsson SDS. Express Talk registers as A, which means it is client A and PhonerLite registers as B, which means it is client B.

In this evaluation, we tried several use cases, direct client to client call, Relay Call, Call Transfer, SDP Swap and Re-invite. The Web Client is not test in this environment because it is special designed for real VoIP service provider.

10.1.2 Analysis

Different call methods are compared with direct call. The establish time for direct call is less than one second. SDP swap and Re-invite are less than two seconds. Call

Transfer is a bit longer than them. And Relay Call takes about three seconds to establish the connection.

The result shows that Relay Call gives the longest time of setup the connection. This is because the mechanism of Relay Call. In the process of establishing phone call, according to *Web Call SDK* [4], it needs two phrase. One is setup the connection with Client A and the other is setup the connection with Client B. So the continuously two hand shake take longer time than others. And the method of Call Transfer uses a similar mechanism (explained in section 4.3.1). So it also spends longer time than others. SDP Swap sends `INVITE` concurrently to two clients (explained in section 4.3.2). This makes it finish the connection in one time unit of hand shake. From the point of signal flow (explained in section 4.3.3), the connection time of Re-invite should be longer than SDP Swap. However, the `Re-INVITE` does not need any manually action from user. So it is also fast.

For the latency of phone call, Call Transfer, SDP swap and Re-invite almost have no latency. This is the same as direct call. On the other hand, Relay Call has a latency of about 2 seconds.

The reason of different level of latency is obviously. Relay Call connects two stand alone phone calls. So the latency should be two times of normal latency plus the process time of controller. And all others are direct phone to phone audio flows.

10.2 Usability

This section will separately explain the usability of call method and portal.

10.2.1 Usability of Call Method

- Relay Call According to *Web Call SDK* [4], Relay Call connects two VoIP calls. So there is no limitation at the VoIP service provider. It can be used at all standard SIP network. However, the drawback of Relay Call is that it has a relevant long latency and it need more time to establish connection. This is because of its "Bridge" mechanism.

- Call Transfer The limitation of Call Transfer is that it needs a support of *The Session Initiation Protocol (SIP) Refer Method (RFC 3515)*[33] at client side. Also the VoIP call cannot be terminated from controller side. Except the limitation above the latency and establish time is fine.

- SDP Swap The limitation of SDP Swap is that some VoIP service provider, e.g., Nonoh, may treat SIP message which do not contain a SDP as an invalid message. So for such service providers, SDP Swap method will not work. So it is recommended to test it before use it in a commercial environment. The latency and establish time of SDP Swap method is fine.

- Re-invite The limitation of SDP Swap is that some VoIP service provider, e.g., Nonoh, may treat `RE-INVITE` SIP message as an invalid message. So for such

service providers, Re-invite method will not work. So it is recommended to test it before use it in a commercial environment. The latency and establish time of Re-invite method is fine.

- Web Client As explained in section 4.3.4, Web Client method is special designed for some existing VoIP service provider, e.g., Nonoh. It is recommended to use it when Web Client works with the service provider. Because all signal and media flow are handled inside the network of service provider. It is the most efficiency method. The limitation is that it does not follow the standard of SIP. And it is special designed for some specified service provider. Each new service provider need a whole new design and implementation.

A comparison table of call method can be found in section 10.4.

10.2.2 Usability of Portal

- Desktop Browser View The desktop browser view is special designed for desktop browser of a compute. It could be either a laptop or a workstation. Users can access all functions via it. The limitation is that to use desktop browser view, user must have a computer with him.

- Mobile Browser View Like desktop browser view, mobile browser view also supplies all function of Web Call Example Application. As long as user has a mobile phone which connects to Internet and has a HTTP browser, user can enjoy it. However, due to the size of screen of mobile device, the operation is not as convenient as desktop browser.

- Java ME Client The Java ME Client is special designed for fast dial. So it has limitation of the function. It do not has a function of VoIP service provider account management. But it can load the native contact book from a mobile phone and synchronize it with the contact book at Web Call server. A limitation of Java ME Client is that it needs JSR 172 [24] to communicate with server. So only the mobile device which supports JSR 172 can use it.

10.3 Reliability

For the point of connection, Web Call Example Application contains five different call methods. From the best choice (see section 10.4) to the must work choice (Relay Call, see section 10.2.1) there must be a method suit the specified VoIP network.

Since the VoIP call uses the network of VoIP service provider, the reliability of VoIP calls depend on the reliability of service provider.

<table>
<tr><td rowspan="3"></td><td rowspan="3">PSTN</td><td colspan="4">VoIP</td></tr>
<tr><td rowspan="2">Relay Call</td><td colspan="3">Third Party Call</td></tr>
<tr><td>Call Transfer</td><td>SDP Swap</td><td>Re-Invite</td><td>Web Client</td></tr>
<tr><td rowspan="2">service provider</td><td rowspan="2">All PSTN switch</td><td rowspan="2">support by all VoIP service providers</td><td colspan="4">Not support by all VoIP service providers</td></tr>
<tr><td>The ones who support REFER method</td><td>The ones who do not filter SIP message which doesn't carry SDP</td><td>The ones who support Re-Invite message</td><td>The ones who supply web site call</td></tr>
<tr><td rowspan="2">Client</td><td rowspan="2">Traditional telephone or mobile phone</td><td rowspan="2">All clients (traditional phone, mobile phone, software-client)</td><td colspan="3">Traditional phone, mobile phone, software-client, under particular requirements of software-client or PSTN gateway</td><td rowspan="2">All clients (traditional phone, mobile phone, software-client)</td></tr>
<tr><td>Client need support REFER method</td><td>Client need implement RFC 3725</td><td>Client need support Re-Invite</td></tr>
<tr><td>Media Stream</td><td>In PSTN network</td><td>Internet and/or PSTN, need to be handled on controller</td><td colspan="4">Internet and/or PSTN</td></tr>
<tr><td>Latency</td><td>Very Short / QoS guarantee</td><td>long</td><td colspan="4">Short, acceptable. But no QoS guarantee</td></tr>
<tr><td>Cost</td><td>Expensive (especially for international call)</td><td colspan="5">cheap</td></tr>
</table>

Table 10.1. Comparison of Call Method

10.4 Comparison of Call Method

A comparison of different call method is shown in Table 10.1. It can be seen from the table, when talk about latency, the best call method is the PSTN (the traditional way). It is the only method that supports QoS (Quality Of Service). So there is a guarantee of quality of audio. But it costs a lot for international sessions. The Relay call can be used on any server and client with a low fee. But the latency is long and sometimes unacceptable. The relay call method brings media traffic to the controller so it not possible to have it on a small or personal server. The third party call is not support by all the services. We have developed four different solutions on third party call. Each of them has its own Cons and Pros. It is recommended that when make an international or long distance call, the web client method should be choose. If it is not possible to use web client method, users can try the other third party call implementations. If none of them works, user can choose either the PSTN which supplies a good quality and high cost or the Relay Call which supplies a poor quality but with low price.

Chapter 11

Conclusion

THIS dissertation has focused on the solution and implementation of Web Call Example Application. In the solution part, it introduces the theory of manipulating SIP message to achieve third party call control. And in implementation part, it covers all the components of Web Call Example Application.

11.1 Conclusion of solution

Two kind of connection solution of the core of Web Call are introduced, they are Relay Call and Third Party Call. This article explained the drawback of Relay Call and how third party call could cover these facts. Four different implementations of third party call are introduced in detail with SIP message flow.

Relay Call is the legacy solution of previous version of Web Call Example Application. According to *Web Call SDK* [4], the Relay Call controller works as an User Agent Client, connect two sessions and transfer media flow from one session to another. It can be used in any standard SIP network with the drawback of relevant long latency and long connection time.

Call Transfer is a solution of third party call that uses *The Session Initiation Protocol (SIP) Refer Method (RFC 3515)* [33]. When connect with the Client A, controller sends a `REFER` signal which force Client A calls Client B. This solution needs a support of RFC 3515 at client side.

SDP Swap is a solution of third party call that manipulates SIP messages. While the hand shake with Client A and Client B, controller swaps SDP message from A and B. This makes Client A and Client B's media flow go direct to each other. It is a very efficient solution with limitation. SIP message without SDP will be used in this solution. Though it is validate according to *SIP: Session Initiation Protocol (RFC 3261)* [29], some service provider may treat such kind of message as illegal.

Re-invite is a solution of third party call that use a `RE-INVITE` message to re-

establish media flow. At the first phase of hand shake, controller collects SDP from Client A and Client B. At the second phase, controller swaps the SDPs via a `RE-INVITE` to Client A and Client B. It is also a very efficient solution with limitation. `RE-INVITE` message will be treated as invalid message by some service provider.

Web Client is a solution of third party call that works as a HTTP client to communicate with the provider's web portal. It does not manipulate any SIP message at all. Instead, it send HTTP request to the web site, and the VoIP connection will be established by the native method of VoIP service provider.

11.2 Conclusion of application

Web Call Example Application is an application that hosted by a web server and supplies functionality of VoIP phone calls. The thesis covered all components of the application. SIP Call Component, Web Application, Web Service Interface and Java ME client are all introduced in the article.

SIP call component is a high level API that used for making VoIP phone calls. It integrated five different methods. It is the core stack of Web Call Example Application. It is designed to work as a standalone library and can be used in other applications. The abstract factory in SIP call component gives a very convenient way to initialize different call controllers.

Web Application is a component that supplies variety functions of VoIP call, and furthermore, it supplies a way of managing SIP account and contact books. Based on Mobile Front Controller, Web Application provides two view that is desktop browser view and mobile browser view. The two views are special optimized for different type of web browsers.

Web Service interface is a common interface of Web Call Example Application. It bases on SOAP Web Service technology, implements most of the functions of Web Application. Besides that, it also contains a feature of synchronize contact book. Java ME client is one of the possible clients of Web Service interface.

Java ME client is a client of Web Service interface. The main purpose of it is to give a way of fast dial. It can record all the latest operations include the number of caller and callee. The last used call method will also be stored in the Java ME client. It can also load the native contact book of mobile phone and store it into contact book of Web Application.

11.3 Future Work

There are many other tasks can be counted into requirements of Web Call Example Application. For the limitation of time and resources, they are planned to be implemented in the next version of release.

11.3.1 Switch SIP stack to JSR 32

Most of our implementations are based on MjSIP, which is already stopped maintenance in year 2006. [23] So it is not a good idea to continue working with this SIP stack.

JSR 32 JAIN™ SIP API is the best candidate for base SIP stack. *"The JAIN™ SIP API specification provides a standard portable interface to share information between SIP Clients and SIP Servers, providing call control elements enabling converged-network applications."* [26] It is a java standard and can be use for free. And furthermore, it is still been well maintained. The latest version is 1.2.

Thanks to the five layers architecture, it will not be a tough work to switch SIP stack API.

11.3.2 RESTful web service interface

Though Web Call Example Application already has a web service as described in chapter 8, it cannot be used for all kinds of clients. Current web service interface in Web Call Example Application is based on SOAP. The SOAP based Java ME Client requires a JSR 172 supported device. However, not all the devices are designed to have such API. e.g., most windows mobile based cell phones do not supply JSR 172. On the other side, almost all mobile phones have a HTTP connection support. And it is enough to have a RESTful web service client. RESTful web service can also be used for web 2.0 mushup.

There is an application called RESTful Web Call Gateway which is developed by *Peter Yeung* from Ericsson Developer Connection. It is a RESTful web service gateway based on Web Call Example Application. It is like a gateway convert SOAP web service to RESTful web service. The architecture of RESTful Web Call Gateway and use case is shown in Figure 11.1.

However, the RESTful Web Call Gateway is not a native interface of Web Call Example Application. So it is not that efficiency. A good design should be that have both SOAP web service and RESTful web service on top of SIP Call Component.

11.3.3 Gadgets

There could be many gadgets client for Web Call Example Application. Most of them will use RESTful web service interface. The gadgets could be Facebook gadget, igoogle gadget, or Vista sidebar gadget. A vista gadget Call You Now was developed by Peter Yeung. It uses the RESTful web service supplied by RESTful Web Call Gateway.

11.3.4 Call history

If an administrator wants to know the call histories, the only way is to see the server log. However, it is not convenient. And it is also hard to manage the call logs. There

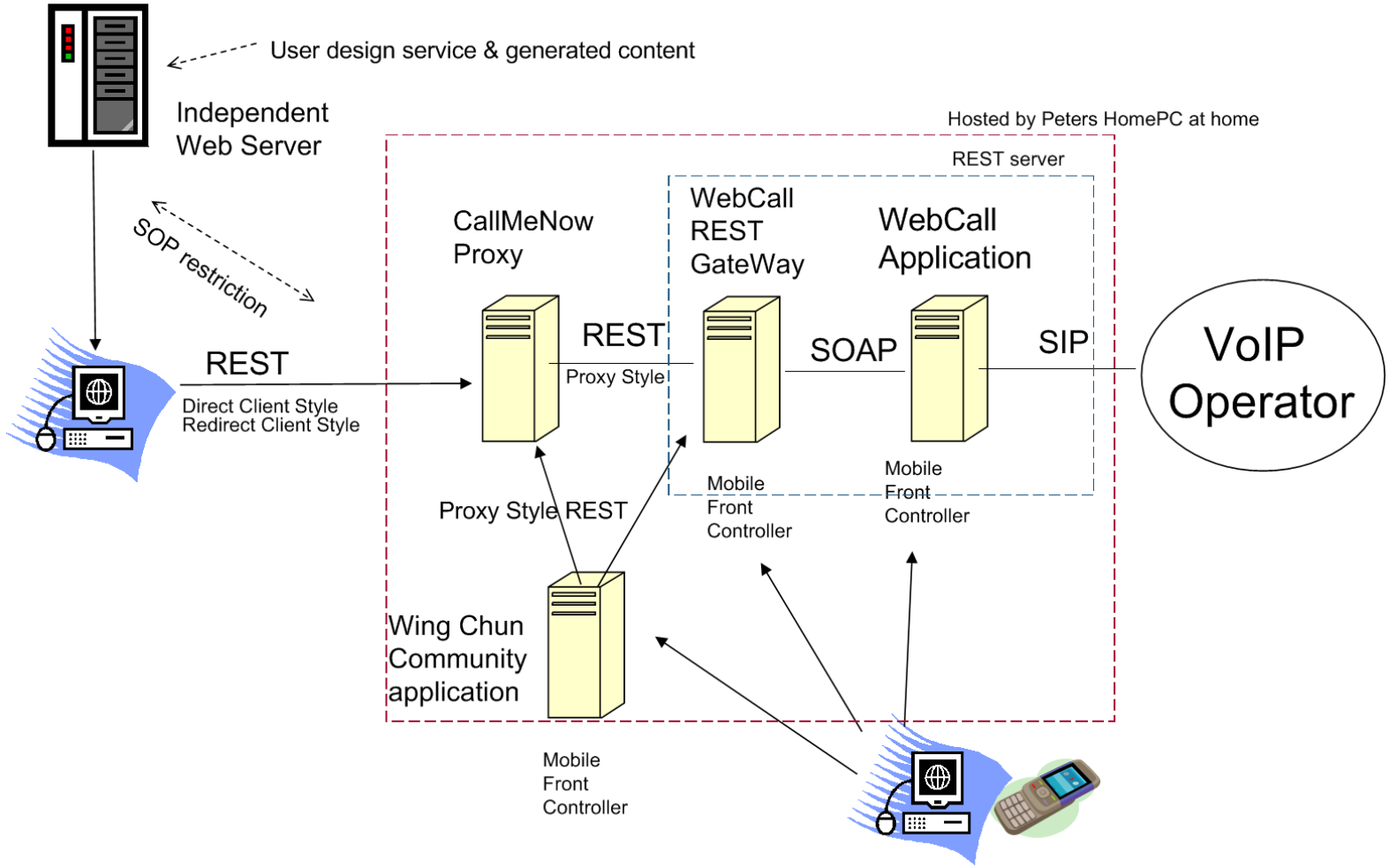

Figure 11.1. REST web call gateway (Figure taken from *Using REST and Web Services to Mash Up Communications Capabilities* by Elena Fersman and Peter Yeung [36])

is also a plan for next release of web call to add call histories to database. So it is easy to track and manage.

Bibliography

[1] GSM Association. GSM.
http://www.gsmworld.com/technology/gsm/index.htm, July 2009. [cited at p. 6]

[2] Gilles Bertrand. The IP Multimedia Subsystem in Next Generation Networks.
http://www.rennes.telecom-bretagne.eu/~gbertran/files/IMS_an_overview.pdf, May 2007. [cited at p. 5]

[3] Mary Campione and Kathy Walrath. The java programming language.
http://java.sun.com/docs/books/tutorial/getStarted/intro/definition.html, February 2008. [cited at p. 4]

[4] Yuening Chen. *Web Call SDK*. Uppsala Universitet, March 2003. [cited at p. 7, 12, 15, 30, 68, 73]

[5] Erik Christensen, Francisco Curbera, Greg Meredith, and Sanjiva Weerawarana. Web Services Description Language (WSDL) 1.1.
http://www.w3.org/TR/wsdl, March 2001. [cited at p. 6]

[6] International Engineering Consortium. Voice over Internet Protocol.
http://www.iec.org/online/tutorials/int_tele/index.asp, 2007. [cited at p. 5]

[7] Ericsson. Ericsson Service Development Studio (SDS) - 4.1.
http://www.ericsson.com/developer/sub/open/technologies/ims_poc/tools/sds_40, February 2009. [cited at p. 67]

[8] Erich Gamma, Richard Helm, Ralph Johnson, and John M. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional, November 1994. [cited at p. 31, 45, 46]

[9] Jesse James Garrett. Ajax: A New Approach to Web Applications.
http://www.adaptivepath.com/ideas/essays/archives/000385.php, February 2005. [cited at p. 45]

[10] Soma Ghosh. J2ME record management store.
http://www.ibm.com/developerworks/library/wi-rms/, May 2002. [cited at p. 4]

[11] Web Services Architecture Working Group. Web Services Glossary.
http://www.w3.org/TR/2004/NOTE-ws-gloss-20040211/, February 2004. [cited at p. 6, 49]

[12] Martin Gudgin, Marc Hadley, Noah Mendelsohn, Jean-Jacques Moreau, Henrik Frystyk Nielsen, Anish Karmarkar, and Yves Lafon. Soap Version 1.2. http://www.w3.org/TR/soap12, April 2007. [cited at p. 6]

[13] M. Handley, V. Jacobson, and C. Perkins. SDP: Session Description Protocol , RFC 4566. http://www.ietf.org/rfc/rfc4566.txt, July 2006. [cited at p. 5]

[14] Eric Jendrock, Jennifer Ball, Debbie Carson, Ian Evans, Scott Fordin, and Kim Haase. *Java(TM) EE 5 Tutorial, The 3rd Edition*. Prentice Hall PTR, February 2005. [cited at p. 48]

[15] Pär Johansson and Peter Yeung. *Mobile Front Controller Developer's guide for software version 3.1*. Ericsson Developer Connection, Ericsson, October 2008. [cited at p. 36, 37, 38, 84]

[16] Paul D. Kretkowski. State of the voip market 2008. http://www.voip-news.com/feature/state-voip-market-2008-031008/, March 2008. [cited at p. 11]

[17] Shanbo Li. Scallope. http://scallope.googlecode.com/, February 2009. [cited at p. 20]

[18] Richard Marejka. Learning Path: MIDlet Life Cycle. http://developers.sun.com/mobility/learn/midp/lifecycle/, February 2005. [cited at p. 4]

[19] Sun Microsystems. Java EE Technology. http://java.sun.com/javaee/technologies/, June 2009. [cited at p. 4]

[20] Sun Microsystems. Java ME Technology. http://java.sun.com/javame/technology/, June 2009. [cited at p. 4]

[21] Sun Microsystems. Metro 1.5 FCS – APT. https://metro.dev.java.net/1.5/docs/apt.html, July 2009. [cited at p. 52]

[22] Sun Microsystems. metro: Home. https://metro.dev.java.net/, July 2009. [cited at p. 49]

[23] MjSip. Mjsip. http://www.mjsip.org/, December 2006. [cited at p. 75]

[24] Java Community Process. JSR-00172 J2ME Web Services Specification. http://jcp.org/en/jsr/detail?id=172, October 2003. [cited at p. 53, 69]

[25] Java Community Process. JSR 75: PDA Optional Packages for the J2METM Platform. http://jcp.org/en/jsr/detail?id=75, June 2004. [cited at p. 53, 56]

[26] Java Community Process. JSR 32: JAINTM SIP API Specification. http://jcp.org/en/jsr/summary?id=32, November 2006. [cited at p. 75]

[27] Benny Ritzén. JavaOne: Come to Ericsson's BOFs and technical session and be inspired . http://www.ericsson.com/developer/sub/articles/other_articles/090514_javaone_bof, May 2009. [cited at p. 7]

[28] J. Rosenberg, J. Peterson, H. Schulzrinne, and G. Camarillo. Best Current Practices for Third Party Call Control (3pcc) in the Session Initiation Protocol (sip), RFC 3725.
http://www.ietf.org/rfc/rfc3725.txt, April 2004. [cited at p. 12, 17, 19]

[29] J. Rosenberg., H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol, RFC 3261.
http://www.ietf.org/rfc/rfc3261.txt, June 2002. [cited at p. 5, 9, 19, 40, 73]

[30] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A Transport Protocol for Real-Time Applications, RFC 3550.
http://www.ietf.org/rfc/rfc3550.txt, July 2003. [cited at p. 5]

[31] NCH Software. Express Talk.
http://www.nch.com.au/talk/, 2009. [cited at p. 67]

[32] Heiko Sommerfeldt. PhonerLite.
http://www.phonerlite.de/index_en.htm, 2009. [cited at p. 67]

[33] R. Sparks. The Session Initiation Protocol (SIP) Refer Method.
http://www.ietf.org/rfc/rfc3515.txt, April 2003. [cited at p. 17, 18, 68, 73, 84]

[34] Wikipedia. Ajax (programming).
http://en.wikipedia.org/wiki/Ajax_(28programming), 2009. [cited at p. 45]

[35] Wikipedia. Transport layer security.
http://en.wikipedia.org/wiki/Secure_Sockets_Layer, June 2009. [cited at p. 5]

[36] Peter Yeung and Elena Fersman. Using REST and Web Services to Mash Up Communications Capabilities. In *JavaOne 2009*.
http://www.ericsson.com/developer/developerszonedown/downloads/open/top/JavaOne_09_Ericsson_BOF.ppt, June 2009. [cited at p. 7, 76, 85]

[37] Peter Yeung and Pär Johansson. Mobile Front Controller.
http://www.ericsson.com/developer/sub/open/technologies/mobile_browsing/tools/mobile_front_controller, March 2009. [cited at p. 35]

List of Abbreviations

Abbreviation	Description	Definition
3pcc	Third Party Call Control	page 12
Ajax	Asynchronous JavaScript and XML	page 45
GSM	Global System for Mobile communications	page 6
IMS	IP Multimedia Subsystem	page 5
ISDN	Integrated Services Digital Network	page 6
ISUP	ISDN User Part	page 6
JVM	Java Virtual Machine	page 4
PSTN	Public Switched Telephone Network	page 6
QOS	Quality Of Service	page 71
RMS	Record Management System	page 4
SDP	Session Description Protocol	page 5
SIP	Session Initiation Protocol	page 5
VoIP	Voice over Internet Protocol	page 5
XHTML	Extensible Hypertext Markup Language	page 38

List of Figures

List of Tables