Analyze and Implementation of third party call control in Web Call SDK

Shanbo Li[1]

KTH - Royal Institute of Technology

SE-100 44 Stockholm

Sweden

shanbo@kth.se

Abstract

Session initiation Protocol (SIP) is widely used in VoIP commutation. Web Call SDK is a project from Developer Program, Ericsson Mobility World. It integrates call functions into Web containers. This paper firstly illustrates the current problem of Web Call SDK. Then it shows how Third Party Call Control (3PCC) mechanism is used in the new solution.

Categories and Subject Descriptors

J.m [Computer Applications]: Miscellaneous

General Terms

Performance, Design, Experimentation

Keywords

Session initiation Protocol, Voice over IP, Third Party Call Control

---

[1] Shanbo Li is a Java Developer in Ericsson Developer Program, Mobility World. He is in charge of Web Call SDK project.

1   Introduction

1.1   Background

IP telephone (VoIP) technology enables the communications between Internet users and the endpoints in PSTN circuit switched (CS) networks. As we know, Session Initiation Protocol (SIP) is widely adopted as a signaling protocol for VoIP communications. Because of its simplicity, power and extensibility, it has also been selected by the Third Generation Partnership Project (3GPP) as a major component of IP Multimedia Subsystem (IMS) for the evolved UMTS core network.

Web Call SDK from Ericsson includes a high-level API that can be used to implement SIP call control. Based on IMS/SIP network technology, Web Call SDK integrated call functions into Web containers. This presents a simple way to implement the communication convergence of Web, IMS/SIP network, and CS networks. It does not require the installation of the plug-in clients on the browser or other special client software as most VoIP services.

Compare to other customer-oriented VoIP products, Web Call SDK offers solutions for developers. It integrated the SIP call function into Web container as Java EE components, providing the example of the convergences between SIP/IMS and the Web with existing Internet Web technology on Java EE Web container.

1.2   Overview

The Web Call Kit is a software development kit (SDK) consists of SIP Control Component Library, an Integrated Web Application and a Java ME - Web Service Client application. The purpose of Web Call Controller Kit is to provide high level API to implement SIP call control, and demonstrate how to integrate SIP call component in web container as Java Beans and web services. It also includes a java ME application examples to access web service making phone-to-phone SIP calls. This SDK can help Java application developers to understand how SIP call component Library works and how Java EE and Java ME applications can be developed upon it. This SDK can be used for connecting to a SIP based network such as IMS core network.

1.2.1 Network Overview

Figure 1 Network overview shows the general network architecture for a service provider with the integrated web application. The application runs on a Java EE web container and resides outside the operator's mobile network.
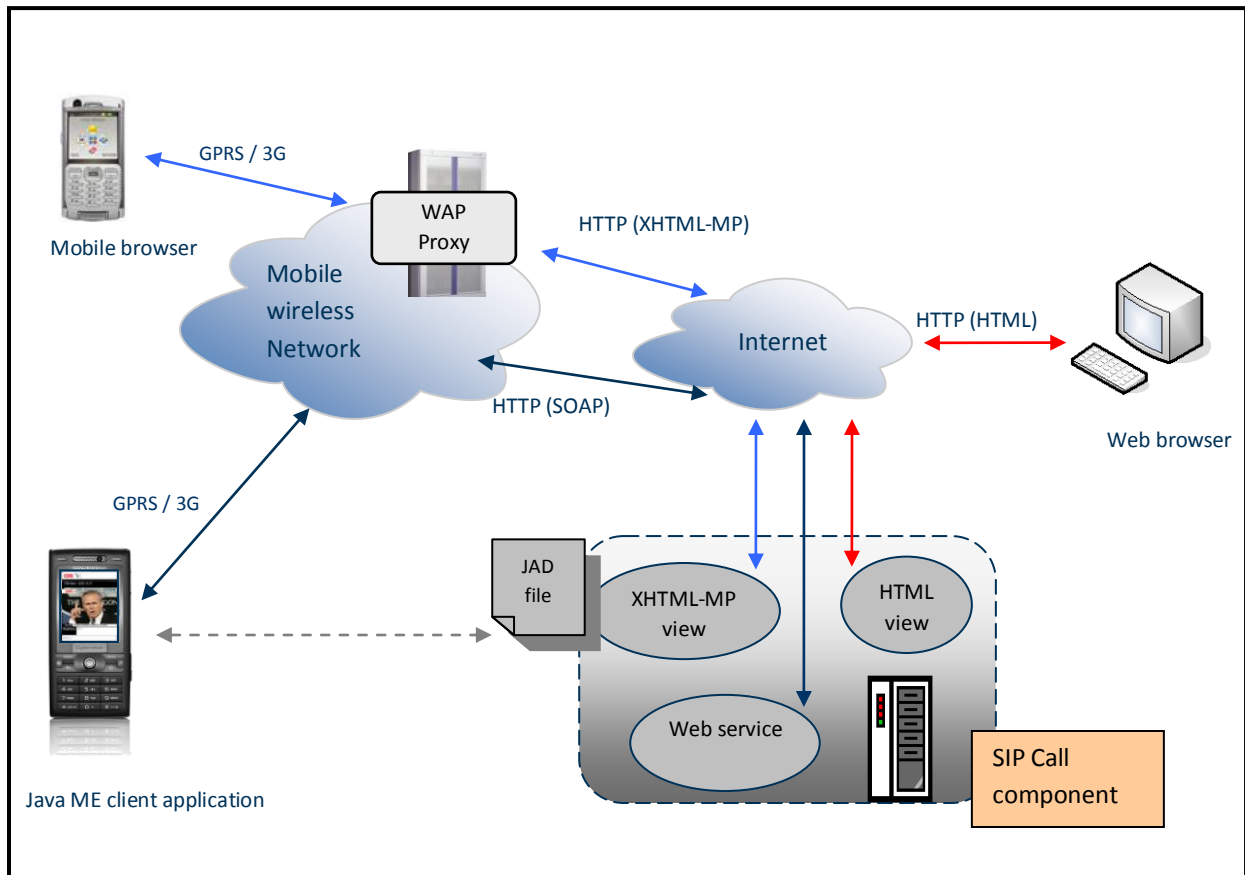
Figure 1 Network overview

In the front-end of the application, there are three types of access.

- Web users can use web browsers to visit the web site via the ISP on the internet. The web application provides HTML version for web users.

- Mobile users can use WAP browsers to visit the web site via the mobile operator's network. The web application provides XHTML-MP versions for mobile user having either bigger or smaller screens.

- More advanced mobile user, with JSR 172 support, can download and install the Java ME client application, and access the web services via mobile operator's network.

The interaction between the second access and the third one is that: the integrated web application dynamically generates the Java ME client application profile after the WAP users log in.

2 Problem

2.1 Current Web Call SDK

Currently, Web Call SDK manually setups and manages sessions, and meanwhile, act as media communication proxy.

The session control module handles the SIP session initiation and the coordination between sessions. SIP control component only implemented the phone-to-phone SIP call function. Thus, the primary primitive operation of Sip control component is establishment of session between endpoints A and B. Establishment of this session is orchestrated by a third party, SIP controller.
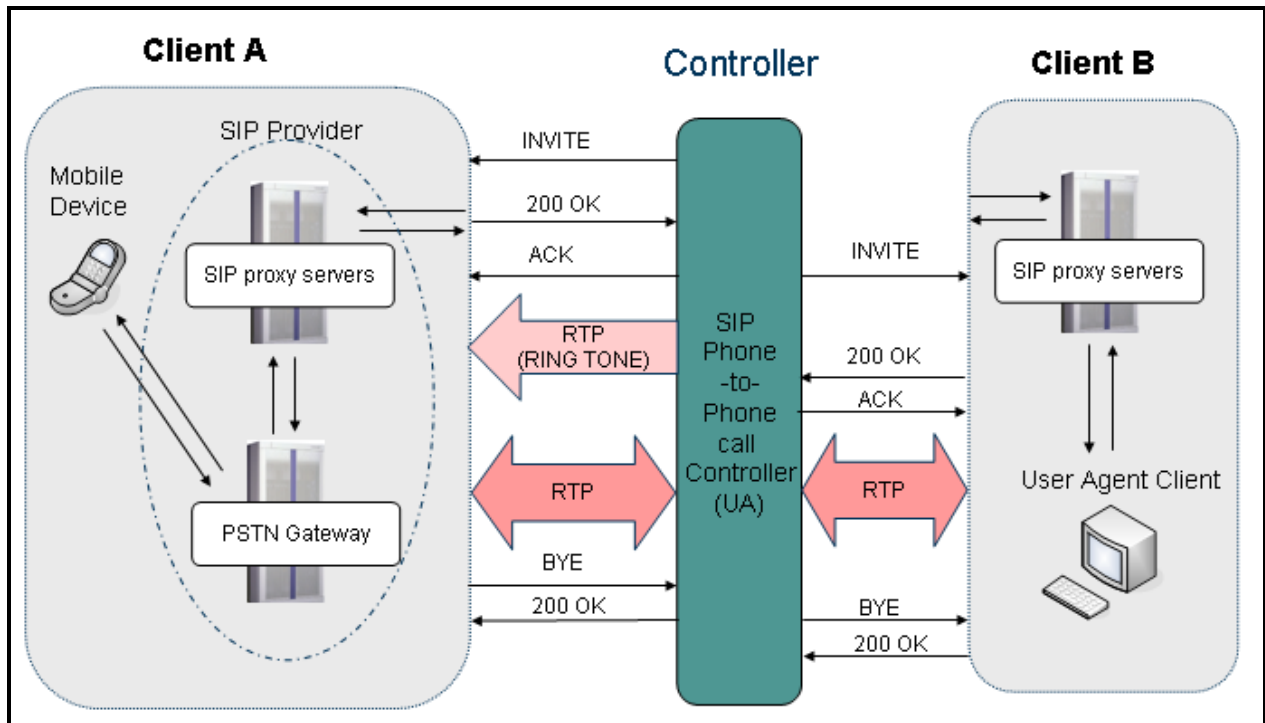

Figure 2 Use case of Web Call SDK (Relay Call)

The basic case flow is shown in Figure 2. The controller first sends an INVITE to A. A's phone rings, and A answers. This results in a 200 OK that is sent to session initiator, controller. The controller needs to send the ACK. This part is the typical SIP session setup steps.

After the session between controller and client A has been established, controller starts to send ring tone to client A, indicating that it is waiting for B to answer the phone.

Once the controller receives a 200 OK from client A, it start another session initiation with client B by sending INVITE to B.

After B answers the phone, controller receives 200 OK from client B. It will stop sending ring tone to A, and start RTP packets proxy between A and B.

2.2 Problems of current solution

2.2.1 The load of controller

The controller here acts as a back to back user agent (B2BUA). It receives the RTP flow from one client and transfers to another client. It is the same as the opposite direction. That means all of the RTP will go through the controller. So the load of controller will become much heavier with the number increasing of concurrent users. A powerful host is needed to handle the RTP flows. But the design goal of Web Call SDK is just a tool kit that can easily integrate to a web site. And current mechanism of session control will decrease the performance of whole web site.

2.2.2 The latency of phone call

As can be seen from the session flow, the RTP goes from one client to controller via a SIP provider and then the controller transfer the RTP to another client via the SIP provider again. It means that one direction of RTP will go through SIP provider twice. So the latency of phone call via SIP provider will be double of normal calls. Furthermore, the performance of single line transfer will decrease as mentioned in section 2.2.1. The latency of phone-to-phone call via Web Call SDK test turned to be more then 3 seconds. It is quite unacceptable.

2.2.3 Reliability

All of RTP flows go through the controller. So if the controller crashes, all of calls will be interrupted immediately.

3 Solution

A solution is needed urgently to solve current problem. The main reason of two problems in section 2 is the RTP flows have to go via controller. So the solution should find a way to avoid the RTP and establish the phone call inside SIP network.

3.1 Third Party Call Control

3.1.1 Introduce of third party call control

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 to as 3pcc) is often used for operator services (where an operator creates a call that connects two participants together) and conferencing. [1]

3.1.2 Third party call control in Web Call SDK

Third party call control is possible using only the mechanisms specified within RFC 3261 SIP: Session Initiation Protocol. [1, 2] And the most significant advantage of third party call control is

the controller only need to handle message transfer and leaves the RTP flow for SIP service provider.

The message transfer only happens at the initial of call or at the end of call. All the controller need to do is just send several messages between two clients. As the RTP flow is established directly between clients, on direction of communication only go through the SIP provider once. Just like a normal SIP call. And the latency just depends on the service quality of SIP provider.
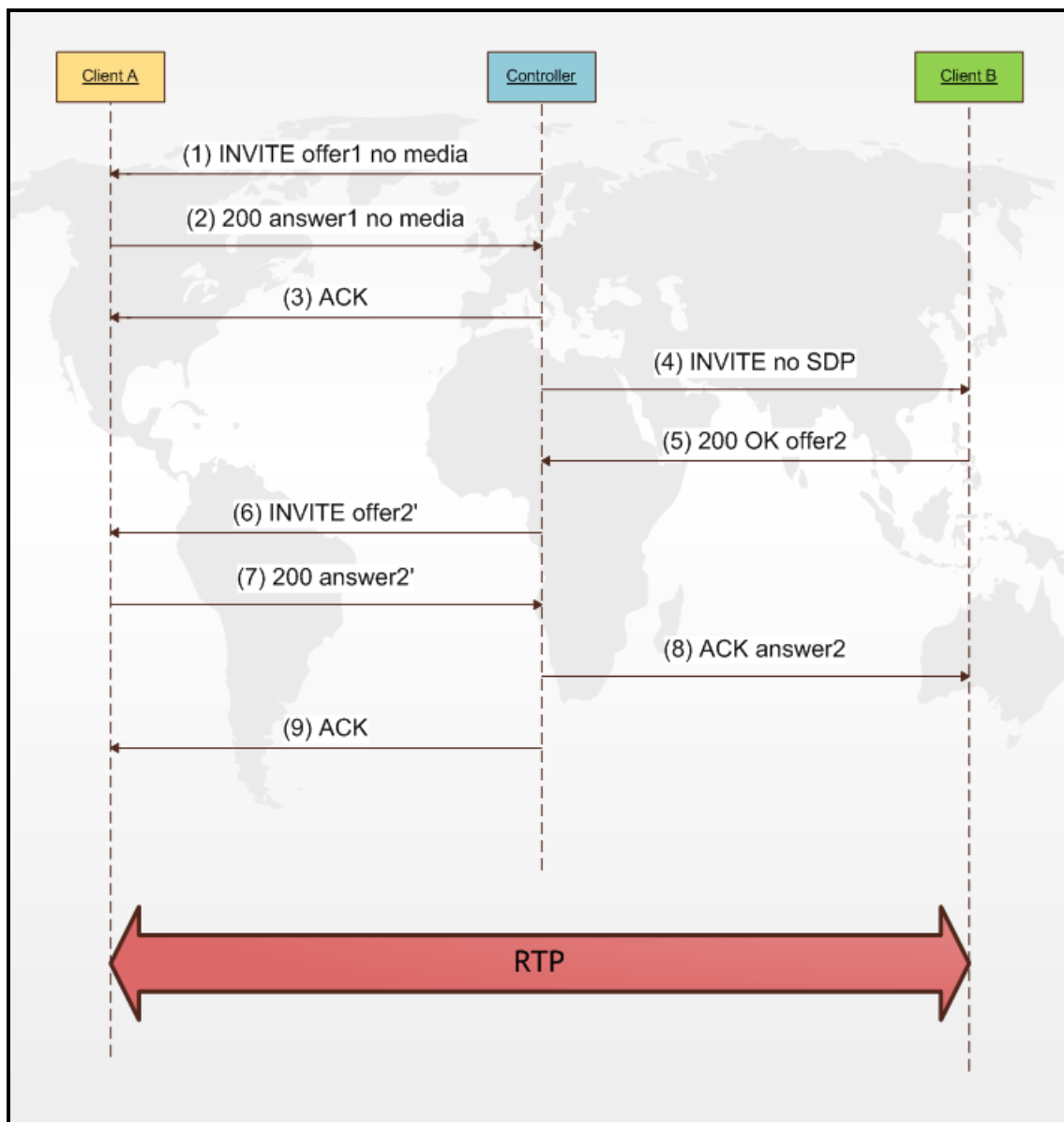

Figure 3 Flow of Third Party Call Control

3.2   Design

3.2.1 3PCC Call Establishment

As shown in Figure 3, our 3PCC call establishment follows the Flow IV in RFC 3725[1]. First, an INVITE(1) is send out from controller to client A. It's SDP has no media field. It means the RTP will be established later by a re-INVITE. After the INVITE(1), client A will return a 200 ok with no m field in SDP. Controller will acknowledge this by ACK(3). Then, controller will send an INVENT(4) without SDP to client B. And client B will answer this INVENT(4) by sending out a 200 OK (5). After controller gets the 200 OK (5), it will construct a SDP according the SDP in 200 OK(5) and send a re-INVENT(6) to client A. Client A now can reply a 200 OK(7) with SDP and controller will send an ACK(8) to client B according the 200 OK(7) from client. And finally, send an ACK(9) to client A. The RTP can start now.

3.2.2 Architecture

Figure 4 shows the new architecture of SIP Component of Web Call SDK. It uses an Abstract Factory Pattern to produce call controller. So developers can easily switch from one kind of controller to another.
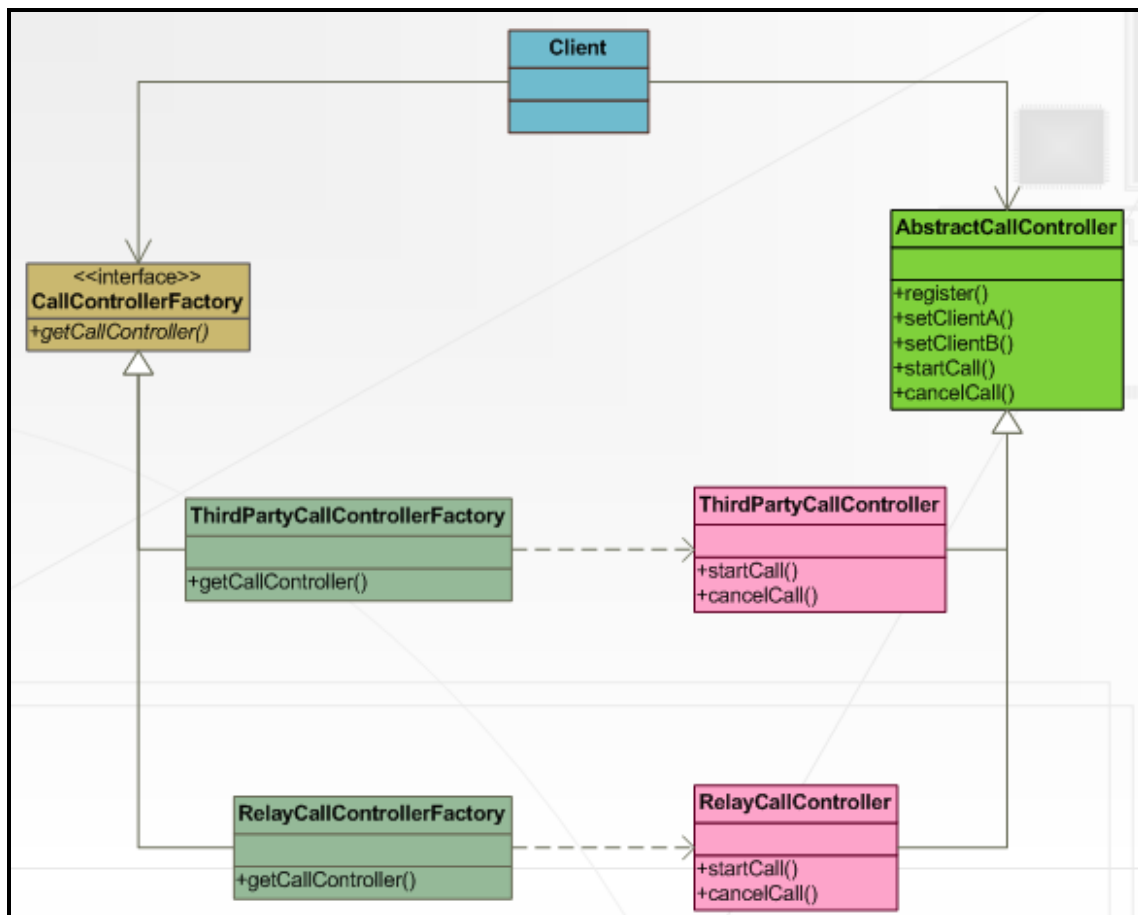

Figure 4   Architecture of SIP component of Web Call SDK

7

This architecture separates different implements of call controller so it is easy to manage and update the code. Since ThirdPartyCallController and RelayCallController both extend AbstractCallController. Some of public method can be defined in AbstractCallController, such as register(), setClientA() and setClientB(). And the two implements can establish the SIP call session with their own way. The information of controller as well as client A and client B are pre-configured when construct of CallControllerFactory. So developers don't need to input the clients twice when swich from one controller to another.

Web Call SDK uses MjSip v1.6[3] toolkit as the lower level SIP stack and implemented phone-to-phone session control.

In the implementation of Third Party Call Controller, the trickiest part is the flow control. In a general INVENT of SIP, a Three-Way Handshake is enough.[8] But in the 3PCC implantation of Web Call SDK, since the initial of session is controller and it aims to establish a phone-to-phone call between two users. It needs to send and handle 9 SIP messages. According to RFC3725 we customize the origin field from 200 OK offer2(9) and send new SDP within INVENT offer2'.

4   Evaluation

This section presents results of experiments of latency of SIP phone-to-phone call which established by Web Call SDK. We ran Web Call SDK both in simulation environment and live experiments. The result appears that the third party call significantly decrease the latency of phone-to-phone call.

4.1   Simulation

We use Ericsson Service Development Studio (SDS) 4.0 as the SIP service provider. [4] It 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.

We use two clients, Express Talk [5] and PhonerLite [6]. They are both very good SIP client. Express Talk register with A which means client A at Ericsson SDS, and PhonerLite register with B which indicates client B.

In this simulation environment evaluation, we tried three use cases. The first one is directly client to client call via Ericsson SDS. The second test use controller as a Back to Back User Agent (Relay Call) which is the original solution of Web Call SDK. The last is third party call control.

The test result and cooperation can be seen from Table 1. We do not have a professional test tool of duration in hand and we use a stop watch to count the latency. So the figures in table are not accuracy enough but it is enough to see the differences.

From the result, we can easily conclude that compare with the directly endpoints' call. Third party call control is much better than the relay call.

	Directly Phone-to-phone Call	Relay Call	Third Party Call control
Establish	1s	6s	5s
RTP Flow	Directly	Transfer at controller	Directly
Latency	Less than 1s	More than 2s	Less than 1s

Table 1 Test Result of SIP Phone Call in Ericsson SDS Simulation Environment.

4.2   Real IMS network

We plan to test and verify our Third Party Call Control solution remotely on a real Ericsson IMS core and application servers at our IMS Expert Center. Ericsson Mobility World's IMS Expert Center lab [7] in Montreal, Canada opens for the developers all over the world and it is the best way to prove our solution.

Unfortunately, due to the full schedule of IMS Expert Center, we can only test it in the end of May, 2008

5   Conclusions

After we implement the Third Party Call Control in Web Call SDK, the load on the controller server side is greatly reduced. Once the session is set up between end points, the controller only needs to manage the SIP session state. The RTP transferring would then be peer-to-peer between endpoints. Even the controller crash, the call will not be interrupted, because they are using directly RTP.

However, since the third party call control in session initiation protocol is just a specification. And most of SIP servicers and gateways don't support the 3PCC specification yet. That is way we keep the back to back user agent solution. This solution follow the most common protocol in SIP, so Web Call SDK can be used anywhere.

6   Future Work

There are some tasks and features in Web Call SDK that are left for future development because of time limitations.

This is a list of future task on third party call control and much of them will follow the instruction of RFC3725.

Error Handling

While establish the call, errors may happen both from client A or client B. Current solution haven't consider the error. So it may lead an establishment failed but without correct error message. After implement the error handling, one can easily found where is the exact place which error accord as well as why there is error.

Cancel Call

Due to the limitation of time, we haven't implemented the call cancel or test it. This will not be hard, because the controller can just send a simple BYE to both side of client.

7   Reference:

[1]. J. Rosenberg, J. Peterson, H. Schulzrinne,G. Camarillo. *RFC 3725 Best Current Practices for Third Party Call Control (3pcc) in the Session Initiation Protocol (SIP).* April 2004.

[2]. J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, E. Schooler. *RFC 3261 SIP: Session Initiation Protocol.* June 2002.

[3]. CreaLab. MjSIP. http://www.mjsip.org/

[4]. Ericsson. Ericsson Service Development Studio 4.0 . http://www.ericsson.com/mobilityworld/sub/open/technologies/ims_poc/tools/sds_40

[5]. NCH Software. Express Talk. http://www.nch.com.au/talk/

[6]. Heiko Sommerfeldt. PhonerLite. http://www.phonerlite.de/

[7]. Ericsson. Ericsson Mobility World's IMS Expert Center lab. http://www.ericsson.com/mobilityworld/sub/open/technologies/ims_poc/testarea.html

[8]. Gonzalo Camarillo. SIP Demystified. McGraw-Hill. 2001