Implementing SIP and H.323 Signalling as Web Services

Ge Zhang, Markus Hillenbrand
University of Kaiserslautern, Department of Computer Science, Postfach 3049, 67653 Kaiserslautern, Germany
{gezhang, hillenbr}@informatik.uni-kl.de
http://www.icsy.de

Abstract

Nowadays, existing Voice over IP (VoIP) clients are signalling protocol dependant. This means that they can only support one signalling protocol. One disadvantage of these VoIP clients is that they are rich clients, and the other is that the problem of interoperability between two clients supporting different signalling protocols must be solved by a signalling protocol translator. We propose a VoIP Web Services architecture, taking advantage of the Web Services technology to solve the above problems. This paper explores the design and implementation of a VoIP Web Services system architecture. We first propose a VoIP Web Service architecture, and then we focus mainly on designing the signalling protocol Web Services, and illustrate the call procedures between the Simple Client and the normal H.323/SIP client under this architecture. We also present the experience in implementing the prototype for the H.323/SIP Web Services, and finally give some considerations about the implementation.

1. Introduction

Voice over IP (VoIP) is the technology to make phone calls over the IP based network. The introduction of the VoIP technology has brought great changes in the traditional telecommunication field. With the VoIP technology, the high cost of the long distance and international voice calls transported with the circuit switch network can be reduced by transporting voice calls over the low cost and flat pricing packet switch network. The economic advantages of VoIP have attracted more and more customers and corporations from the circuit switch network to the packet switch network.

H.323 [1] and Session Initiation Protocol (SIP) [2] are two standard protocols for realizing of the VoIP technology. The H.323 protocol is a recommendation suggested by ITU-T for multimedia communication over packet switched network. H.323 is an umbrella specification which consists of a set of protocols for audio/video encoding, decoding, and call signalling and call control, and capability exchange etc. SIP is also a multimedia communication protocol which is suggested by IETF. SIP handles mainly call signalling and control. In both H.323 and SIP architecture the media flows are carried by RTP [3] protocol.

At present there are many kinds of VoIP products supporting H.323 or SIP. But one problem is that, a H.323 client can not directly connect to a SIP client. In order to connect a H.323 client with a SIP client, some mechanisms, such as H.323-SIP translator, should be used to map the H.323 signalling to the corresponding SIP signalling, or vice versa.

Now existing VoIP solutions implement the client as standalone software. The client software needs to be installed on the terminal devices. The client is usually a so called rich client, since it integrates all VoIP client functions like signalling, media processing and transmission functions. The disadvantage of this kind of rich client is that, whenever the protocols need to be changed or new features should be added, the client software in the terminal device must also be updated.

Web Services are distributed software components which can be described, published, discovered and invoked with standard protocols. Web Services represent a new form of middleware based on XML and the Web. Web Services are platform and language independent. Web Services can be developed using any language, and can be deployed on any platform, from the tiniest device to the largest supercomputer. Furthermore any Web Service can be accessed by any other application, regardless of either's language or platform. Web Services communicate using XML and Web protocols, which are pervasive, work both
internally and across the Internet, and support heterogeneous interoperability.

To solve the problems described above with the currently existing VoIP solutions, we introduce the Venice project which utilizes Web Services technology to implement the VoIP functions as Web Services. In this paper we present the VoIP Web Services architecture of the Venice project, and then focus mainly on the implementation of the H.323/SIP Web Services. This paper is organized as follows: section 2 discusses some related work, section 3 illustrates the VoIP Web Services architecture, section 4 describes the call procedures under the VoIP Web Services architecture, and section 5 explains the implementation of the H.323/SIP Web Service and discusses some considerations while section 6 makes some conclusions.

2. Related work

The interoperability among different signalling protocols has been extensively researched over the years [4][5]. [4] suggests a signalling gateway to allow SIP user agents to call H.323 terminals and vice versa, and [5] uses translators to provide the call translation between an H.323 endpoint and a SIP server or call translation between an MGCP endpoint and a SIP server. These researches all utilize the signalling protocol translator mechanism to translate one signalling protocol into another and vice versa. With this translator mechanism, the performance cost for the translation between different signalling protocols should be considered.

VoIP solutions, such as Net2Phone Dialer [6], MediaRing [7] etc., provide rich clients to make VoIP calls. These clients need to be updated to accommodate newly integrated functions. Although some providers offer their VoIP clients as Web applications which utilize the Applet technique to avoid the problem of client updates, the loading of the rich client results long waiting period, especially for users with low speed internet connections.

3. The VoIP Web Services architecture

Figure 1 illustrates the VoIP Web Services architecture. There are three types of entities in this architecture.

**Simple Client**: In figure 1 a Simple Client (SC) is defined in the VoIP Web Services architecture. Comparing with the traditional rich client wrapping all functions such as signalling, media processing and media flow transmitting, the Simple Client wraps only the media processing and media flow transmitting functions. The signalling protocol function will be realized in the VoIP protocol Web Services server.

![Figure 1 VoIP Web Services architecture](image)

With this architecture the Simple Client can communicate with different types of VoIP clients as long as the VoIP protocols are supported by the VoIP protocol Web Services server. That means that the Simple Client can be used as a multi VoIP signalling protocol client. What kind of signalling protocol can be supported by the Simple Client depends on how many signalling protocol Web Services can be provided by the VoIP protocol Web Services server. With this architecture the Simple Client can be extended to support new protocols without any modification.

**VoIP Web Service manager server**: The VoIP Web Service manager server connects both the Simple Clients and the signalling protocol Web Services server. It is responsible for the authentication of the Simple Clients, the management of sessions, and the forwarding of messages. It can also provide the accounting and billing functions.

**VoIP signalling protocol Web Services server**: It realizes different VoIP protocols. Each VoIP protocol will be implemented as one protocol Web Service component. These components mainly perform the signalling processing. In H.323 and SIP, RTP is the suggested protocol to be used to transmit the media flows. In our VoIP Web Services architecture the RTP protocol will be implemented in the Simple Clients.

With this architecture the VoIP signalling processing and the media processing and transmission will be handled separately. Figure 2 illustrates H.323 protocol stack in the VoIP Web Service architecture.
Figure 2 shows that the H.225 RAS, H.225 call signalling and the H.245 protocols will be processed by the H.323 Web Service component. The media processing and transmission related protocols such as audio codecs and RTP protocol will be realized in the Simple Clients.

The signalling processing Web Services are responsible for call setup and call control. The Simple Clients are responsible for media processing such as encode, decode, and media flow transmission using the RTP protocol. The advantages of the separated signalling protocol processing and media processing are:

- A Simple Client can be used as a multi protocol VoIP client. For example, a Simple Client can connect to either a H.323 client or a SIP client.
- With the extension of the VoIP Web Services to support new signalling protocols, the Simple Clients can also support new signalling protocols without any modification or update. This process is transparent to the Simple Clients.
- Using the VoIP Web Service architecture, different VoIP providers can build their own VoIP service easily. Since they need not care about the signalling processing, the signalling protocol processing can be simply added into their own business logic as a Web Service. It is also very attractive for the company to build their intranet VoIP system. This is one advantage brought by the Web Service technology.
- If the signalling Web Services should be changed or updated, the Simple Clients need not be updated or modified. It is transparent to the Simple Clients.
- The Simple Clients are thin clients compared to the traditional rich VoIP clients. This will facilitate the Simple Clients to be embedded into different hardware platforms.

4. Call procedures between Simple Client and H.323/SIP client under the VoIP Web Services architecture

According to the above description the calls between Simple Clients and the H.323 clients or the SIP clients will be performed with the help of the VoIP signalling protocol Web Services. In this section we use call setup examples to describe the call procedures between Simple Clients and H.323/SIP clients.

4.1 Simple Client calls H.323/SIP client

In this section, we use the example in Figure 3 to illustrate the procedure of a Simple Client calling a H.323 client. This principle can be also applied to the procedure of a Simple Client calling a SIP client.

Before a Simple Client can make a call, the user of a Simple Client should first register to the VoIP Web Service manager. Then the user can login to the manager to begin VoIP service.

If a user of a Simple Client wants to call a H.323 client, the Simple Client will send a MakeCall request to the VoIP Web Service manager. The MakeCall request includes parameters such as the callee's ID, caller's ID, RTP IP address and port number used by the Simple Client to transmit media flow, and the codec list supported by the Simple Client etc.

The VoIP Web Service manager will at first authenticate the user. If the user passes the authentication procedure, the manager will forward the MakeCall request to the H.323 Web Service server. Otherwise an ENDCALL message will be sent to the Simple Client.

The H.323 Web Service server will generate a H.323 Web Service instance to represent the Simple Client to handle the call setup processing with the remote H.323 client.

The H.323 Web Service server will generate a H.323 Web Service instance to represent the Simple Client to handle the call setup processing with the remote H.323 client.

Now the H.323 Web Service instance starts the normal H.323 call setup process with the remote H.323 client.

If the connection establishment fails, the call setup signalling process between the H.323 Web Service instance and the H.323 client will be ended, and an ENDCALL message will be sent back to the Simple Client. The ENDCALL message includes the call setup

Figure 2 H.323 protocol stack

The signalling processing Web Services are responsible for call setup and call control. The Simple Clients are responsible for media processing such as encode, decode, and media flow transmission using the RTP protocol. The advantages of the separated signalling protocol processing and media processing are:

- A Simple Client can be used as a multi protocol VoIP client. For example, a Simple Client can connect to either a H.323 client or a SIP client.
- With the extension of the VoIP Web Services to support new signalling protocols, the Simple Clients can also support new signalling protocols without any modification or update. This process is transparent to the Simple Clients.
- Using the VoIP Web Service architecture, different VoIP providers can build their own VoIP service easily. Since they need not care about the signalling processing, the signalling protocol processing can be simply added into their own business logic as a Web Service. It is also very attractive for the company to build their intranet VoIP system. This is one advantage brought by the Web Service technology.
- If the signalling Web Services should be changed or updated, the Simple Clients need not be updated or modified. It is transparent to the Simple Clients.
- The Simple Clients are thin clients compared to the traditional rich VoIP clients. This will facilitate the Simple Clients to be embedded into different hardware platforms.

4. Call procedures between Simple Client and H.323/SIP client under the VoIP Web Services architecture

According to the above description the calls between Simple Clients and the H.323 clients or the SIP clients will be performed with the help of the VoIP signalling protocol Web Services. In this section we use call setup examples to describe the call procedures between Simple Clients and H.323/SIP clients.

4.1 Simple Client calls H.323/SIP client

In this section, we use the example in Figure 3 to illustrate the procedure of a Simple Client calling a H.323 client. This principle can be also applied to the procedure of a Simple Client calling a SIP client.

Before a Simple Client can make a call, the user of a Simple Client should first register to the VoIP Web Service manager. Then the user can login to the manager to begin VoIP service.

If a user of a Simple Client wants to call a H.323 client, the Simple Client will send a MakeCall request to the VoIP Web Service manager. The MakeCall request includes parameters such as the callee’s ID, caller’s ID, RTP IP address and port number used by the Simple Client to transmit media flow, and the codec list supported by the Simple Client etc.

The VoIP Web Service manager will at first authenticate the user. If the user passes the authentication procedure, the manager will forward the MakeCall request to the H.323 Web Service server. Otherwise an ENDCALL message will be sent to the Simple Client.

The H.323 Web Service server will generate a H.323 Web Service instance to represent the Simple Client to handle the call setup processing with the remote H.323 client.

The H.323 Web Service server will generate a H.323 Web Service instance to represent the Simple Client to handle the call setup processing with the remote H.323 client.

Now the H.323 Web Service instance starts the normal H.323 call setup process with the remote H.323 client.

If the connection establishment fails, the call setup signalling process between the H.323 Web Service instance and the H.323 client will be ended, and an ENDCALL message will be sent back to the Simple Client. The ENDCALL message includes the call setup
failure reason. The corresponding H.323 Web Service instance will also be destroyed.

If the call setup succeeds, the H.323 Web Service instance will receive a CONNECT message from the H.323 client. The former sends to the VoIP Web Service manager a CONNECT message which includes the RTP IP address and port number of the H.323 client and the selected codec decided during the call setup process.

The VoIP Web Service manager logs the CONNECT information for the purpose of management and forwards the CONNECT message to the Simple Client.

The Simple Client uses the codec information in the CONNECT message to decode and encode the media flow, and uses the RTP IP address and port to transmit the media flow to the H.323 client. And now the dialog begins.

If the Simple Client wants to end the call, it sends an EndCall request to the VoIP Web Services manager. The VoIP Web Services manager first logs the management related information and then forwards the EndCall request to the H.323 Web Service instance. The H.323 Web Service instance sends a Release Complete message to the H.323 client to teardown the established call connection. Then the H.323 Web Service instance will be destroyed.

4.2 H.323/SIP client calls Simple Client

In this section, we use the example in Figure 4 to illustrate the procedure of a SIP client calling a Simple Client. The principle can be also applied to the procedure of a H.323 client calling a Simple Client.

Before a Simple Client can accept a call from a SIP client, the user of the Simple Client first needs to register to the VoIP Web Service manager. After the registration the SIP Web Service server can represent the Simple Client to handle the SIP signalling protocol.

When the user of a SIP client calls a Simple Client, an INVITE message will be sent to the SIP Web Service server. The server generates a SIP Web Service instance which stands for the Simple Client to perform the call setup signalling processing.

The SIP Web Service instance sends an INCOMINGCALL message to the VoIP Web Service manager.

The VoIP Web Service manager checks the Simple Client’s status information to decide if the Simple Client can accept the call. The status information includes AAA (Authentication, Authorization and Accounting), online status, busy status etc. If the Simple Client can accept the call, the VoIP Web Services manager sends a CallProceeding message to the SIP Web Service instance and it also forwards the INCOMINGCALL message to the corresponding Simple Client. Otherwise a DenyCall method of the SIP Web Service will be invoked.

The CallProceeding method of the SIP Web Service instance sends the 180 (Ringing) message to the SIP client. The DenyCall method rejects the call setup.
request from the SIP client, and then the SIP Web Service instance will be destroyed.

When the Simple Client receives the INCOMINGCALL message, the user can decide to accept the call or not. If the user accepts the call, an AcceptCall message which includes the Simple Client’s RTP IP address and port number information and also the supported codec list or otherwise a DenyCall message will be sent to the VoIP Web Service manager. The VoIP Web Service manager forwards the response message to the SIP Web Service instance.

The SIP Web Service instance checks the Simple Client’s response message. If the Simple Client rejects the answer, the SIP Web Service instance performs the DenyCall processing to reject the call, and the SIP Web Service instance will be destroyed. Otherwise it performs the AcceptCall processing by sending a 200 (OK) message to the remote SIP client.

When the remote SIP client receives the 200 (OK) message, it sends an ACK message to the SIP Web Service instance to confirm that the client has received a final response to an INVITE request.

The SIP Web Service instance converts the ACK message to a CONNECT message and sends it to the VoIP Web Service manager to inform the successful establishment of the call connection. The information about the RTP IP address and port number of the remote side and the selected codec will also be included in the Connect message.

The VoIP Web Service manager can use the CONNECT message to log the related information about this call for the management purposes such as billing etc. And it forwards the CONNECT message to the Simple Client.

The Simple Client uses the RTP IP address and port number of the remote side and the selected codec to begin the media flow transmitting.

If the remote SIP client wants to end the call, it sends a BYE message to the SIP Web Service instance. The SIP Web Service instance will perform the signalling processing for the BYE message and sends back a 200 (OK) message to the remote SIP client to confirm the end call request. And it also sends a ENDCALL message to the VoIP Web Service manager. Then the SIP Web Service instance will be destroyed.

The VoIP Web Services manager will first log the management related information, and then forwards the ENDCALL message to the Simple Client.

After the Simple Client receives the ENDCALL request, it will stop the RTP media flow transmitting to end this call.

5. H.323/SIP Web Service

For the Venice project, which aims at building a VoIP Web Services system, a prototype of H.323 and SIP Web Services are being developed. The H.323/SIP
Web Services are part of the VoIP Web Services system architecture, and as above described the H.323/SIP Web Services provide the abilities to handle the H.323 or SIP signalling protocol processing. In this section we present the key issues as well as our considerations in implementing the prototype.

5.1. Implementing the signalling Web Services

The signalling Web Services provide services to handle different signalling protocols. Now in the prototype system of our Venice project, H.323 and SIP signalling protocols are being considered. Since the VoIP Web Services architecture (see Figure 1) is an open architecture, other protocols can also be easily integrated into the system. The signalling Web Services will provide uniform service interfaces no matter it is H.323 Web Service or SIP Web Service. We chose the Java 2 SDK (J2SE) and the Java Web Services Developer Pack (JWSDP) to develop the Web Services.

For the H.323 protocol we chose OpenH323, which is an open source H.323 protocol implementation developed by the OpenH323 project [8], to handle the H.323 signalling processing. The OpenH323 project implements the H.323 protocol in C++. As for the SIP protocol we chose JAIN SIP, which is also an open source SIP protocol implementation developed by NIST [9], to handle the SIP signalling processing. The JAIN SIP implements the SIP protocol in Java.

We wrap these two implementations to provide signalling protocol Web Services. The H.323 and SIP signalling protocol Web Services provide the same methods described below:

**InitCall Method**: this method is used to initialize the H.323 or SIP Web Service. It starts the signalling protocol listening server. When the server is started, it will listen to the incoming call setup request. If a call setup request from the remote H.323 or SIP client is received, a corresponding signalling processing Web Service instance will be created to process the signalling of this call, and this instance will exist until the end of the call. Parameters for this method:

- **CallBackHandler**: it specifies the interface which is responsible for receiving the response or notifications from the H.323/SIP Web Service instance.

**MakeCall method**: this method is invoked when a Simple Client wants to setup a call to a H.323 or SIP client. When this method is called, a corresponding signalling processing Web Service instance will be created to process the signalling of this call, and this instance will exist until the end of the call. The instance will then use the following parameters to process the call setup signalling.

- **Caller (Simple Client) ID**: to identify the Simple Client. It is the Simple Client’s IP address or party number or alias which is in the format corresponding to definition of the specified signalling protocol.
- **Callee (H.323/SIP client) ID**: to identify the remote H.323 or SIP client. It is also the H.323 or SIP client’s IP address or party number or alias which is in the format corresponding to definition of the specified signalling protocol.
- **Simple Client’s RTP IP address and port number**
- **Simple Client’s codec list**: list of the codecs supported by the Simple Client. This will be used during the call setup to coordinate the codec which can be supported by both sides.

**AcceptCall method**: this is a response of the Simple Client to the notification INCOMINGCALL. This method will use the following parameters to setup the call connection.

- **Simple Client’s RTP IP address and port number**
- **Simple Client’s codec list**

**DenyCall method**: this is a response to the notification INCOMINGCALL. This response may come from the Simple Client if the use rejects the call, or from the VoIP Web Service manager if it detects the Simple Client can not accept the call. This method will use the following parameter as the reason to reject the call request from the remote H.323/SIP client.

- **Reason**

**CallProceeding method**: this is a response to the notification INCOMINGCALL. It is invoked when the VoIP Web Service manager decides the Simple Client can accept the call and forwards the INCOMINGCALL request to the Simple Client. This will cause the H.323/SIP Web Service instance to send a corresponding signalling message, e.g. CallProceeding in H.323 or 180 (Ring) in SIP, to the H.323/SIP client.

**EndCall method**: this is the end call request from the Simple Client. The H.323/SIP Web Service instance will perform the end call signalling and then the instance will be destroyed.

- **Reason**
The following notification messages will be sent to the CallBackHandler specified in the InitCall method by the Web Service instance:

**INCOMINGCALL**: this message informs the Simple Client that a remote call setup request has been received. Following parameters will be sent together with this message:
- Caller (H.323/SIP client) ID
- Callee (Simple Client) ID

**CONNECT**: this message informs the Simple Client that the call setup was successful and a connection has been established. Parameters included in this message are:
- H.323/SIP client’s RTP IP address and port number
- Selected Codec

**ENDCALL**: this message informs the Simple Client that the remote H.323/SIP client wants to teardown the call.
- Reason

### 5.2. Considerations about implementing the signalling protocol Web Services

In the implementation of the VoIP Web Service system, the VoIP performance should be carefully considered. In our suggested VoIP Web Services architecture the VoIP Web Services server is added as an intermediate layer between the Simple Client and the H.323/SIP client; this will certainly affect the call performance, e.g. by increasing the call setup delay. In our implementation we use RPC to decrease the Web Service call delay, and we carefully design the messages mechanism to reduce the message transmission round trip time between the Web Service instance and the Simple client to improve the overall system performance.

Another consideration is security. Our suggested VoIP Web Services system architecture uses the VoIP Web Service manager to authenticate the Simple Clients and it will control the access to the H.323/SIP signalling protocol Web Services. This centralized management will enhance the system security. The manager can also provide service such as session management, accounting and billing etc.

### 6. Conclusion

In this paper, we present a VoIP Web Service system architecture. The call procedures between the Simple Client and the H.323/SIP client under the VoIP Web Service architecture are analyzed. We also explain the implementation of the signalling protocol Web Service and some considerations.

With the suggested architecture, a more general thin VoIP client can easily be developed for heterogeneous platforms benefiting from the Web Service technology. The Simple Client can also be extended and improved transparently to support more signalling protocols by integrating new signalling protocol Web Services into the VoIP signalling protocol Web Services server. This architecture provides flexible features in implementing VoIP systems.

### 7. References


