基于 JAIN SIP的 SIP呼叫控制的实现

华 山, 王秋光
(哈尔滨理工大学 电气与电子工程学院,黑龙江 哈尔滨 150040)

摘 要:为了实现基于 JAIN SIP的 SIP呼叫控制,应用 JAIN SIP APIs研究了会话的建立和拆除过程,并根据生成的日志对会话过程进行了分析,对实现结果进行了模拟. 其结果显示对开发基于 JAIN SIP协议栈的业务具有参考作用.

关键词: SIP; JAIN SIP APIs; 会话建立; 会话拆除
中图分类号: TP393. 03 **文献标志码**: A **文章编号**: 1007- 2683(2009)04- 0054- 05

Implementation of SIP Session Control Based on Protocol of JAIN SIP

HUA Shan, WANG Qiu-guang
(School of Electrical and Electronic Engineering, Harbin University of Science and Technology, Harbin 150040, China)

Abstract: In order to achieve Implementation of SIP Session Control based on Protocol of JAIN SIP, in this paper, the session's establishing and terminating are implemented successfully with JAIN SIP, the process of the session is analyzed according to the log which has been created, and the results are simulated. This paper is valuable to references of the development of the business, which is based on the JAIN SIP stack protocol.

Key words: SIP; JAIN SIP APIs; session establishing; session terminating

0 引 言

SIP(Session Initiation Protocol,会话发起协议)是下一代网络中应用层的核心控制协议,正得到越来越广泛的关注. 2001年,SUN公司发布了基于 JAVA技术的规范 JAIN(Java Advanced Intelligent Network) SIP API,将 SIP协议规范为标准的 JAVA接口,开发者可以方便实现 SIP网络架构中的所有 SIP实体,并使其具有 JAVA网络编程语言方便移植的特点. 近年来,国内对 JAIN SIP的应用越来越广泛,上海交通大学[1]、华中科技大学[2]等高校科研机构都对 JAIN SIP技术进行了相关研究.

1 SIP

SIP是由 IETF(Internet工程任务组)提出的 IP电话信令协议. RFC 2543[3]中对 SIP的定义为: SIP是基于应用层的信令控制协议,用来在一个或多个参与者之间创建、修改或终止会话,这些会话类型包括 Internet上的多媒体会议、多媒体发布、电话呼叫等.

近年来,许多设备厂商都开始将其下一代网络的协议标准向 SIP方向倾斜, SIP凭借其简单,易于扩展、便于实现等诸多优点,逐步成为下一代网络和 IP多媒体子系统领域中的重要协议. SIP还是未来的 4G全 IP移动网的指定基础协议. 可以说, SIP协议是未来网络的核心协议[3].

SIP充分借鉴了 HTTP, SMTP(简单邮件传送协议)这两个互联网上最成功的应用层协议,其编码是文本方式,继承了互联网协议简单、开放、灵活的特点. 基于 SIP的网络不仅重新定义了 PSTN,包含了所需的所有 PSTN功能,而且为语音以外的媒体

收稿日期: 2008 - 03 - 18
作者简介: 华 山(1983—),女,硕士研究生.

© 1994-2009 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

通信提供了成长和扩展的空间[4].

SIP协议中有两个要素:SIP用户代理和 S1P网络服务器.用户代理是呼叫的终端元素,本身包含一个用户代理客户机(UAC:User Agent Client)和一个用户代理服务器(UAS:User Agent Server),其中UAC负责呼叫的发起,UAS负责呼叫的接收.SIP服务器是处理与多个呼叫相关联信令的网络设备,主要包括注册服务器、代理服务器和重定向服务器.

2　JAIN SIP

JAIN APIs是一套由 JCP组织推动开发的基于 Java API和面向对象技术的标准接口,应用 Java平台开发基于下一代网络的电信产品业务.JAIN APIs包含一系列的 API,其中应用于 SIP协议的 API有以下 3种:JAIN SIP,JAIN SIP Lite和 SIP Servlets其中 JAIN SIP是基于 SIP协议标准的 Java类库,提供了一系列接口,包括标准化的协议栈接口、信息接口、事件和事件语义,它主要用来开发客户端和服务器.

2.1　JAIN SIP的体系结构

JAIN SIP API实现的是事务层和事务用户层之间的接口,也就是 SIP协议栈底层到 SIP实体的接口.JAIN SIP利用 JAVA的面向对象特性,使用对象、消息以及事件来描述 SIP协议.在 JAIN SIP体系结构中为所有的 SIP报头和报文定义了专门的类,并借助提供者/监听者(SipProvider/SipListener)接口,将用于处理报文的体系结构的接口定义为事件.JAIN SIP的体系结构[4]如图 1所示:

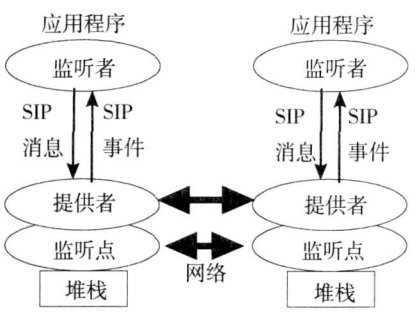

图 1　JAIN SIP的体系结构

堆栈与实现 SIP协议栈的数据结构相对应,所有的 SIP消息和状态码都经由堆栈通过网络收发,它没有面向对象功能,是 SIP的一个原始实现[5].

提供者是 JAIN SIP提供的一个接口,它通过监听点封装了一系列对堆栈进行操作的方法,提供给上层对象面向对象的 SIP方法,从而实现了面向对象的操作.提供者不解释 SIP消息在应用程序中的意义,它仅仅通过上、下层的接口处理 SIP消息的准确发送和接收.两个 SIP应用程序之间的交互在逻辑上就表现为它们之间的提供者进行交互.SIP协议本质上是一个请求/应答协议,是一个无状态的协议,这样既简化了各个代理服务器的功能,又提高了其处理用户请求的能力,同时也增强了 SIP应用程序的灵活性;但另一方面,无状态的协议不能很好地直接支持事务的概念,为了给上层应用提供有状态的请求应答消息,提供者提供了客户端事务(Client Transaction)和服务器端事务(Server Transaction)的创建方法.在 SIP的一个应用程序中可以有多个提供者,每个对应一个堆栈[6].

监听者是 JAIN SIP提供的管理一个或多个提供者的接口,一个应用程序只能有一个监听者.当一个提供者建立之后,将被注册到监听者应用程序.当某个提供者被访问时,访问者需要通过监听者向该提供者发送消息,当提供者收到了其他提供者的 SIP消息时,该提供者需要将其传递给相应的监听者应用程序进行解析,此时它将产生一个事件,提供者理解该事件的含义,并能根据不同的事件触发相应的响应.

2.2　JAIN SIP的对象初始化

从 JAIN SIP体系结构中我们可以看出,JAIN SIP的开发都是围绕着监听者和提供者进行的.在实际应用中还有其他对象共同参与实现其功能,在程序的开始阶段必须对其进行初始化,以创建一系列必须的对象.图 2描述了 JAIN SIP应用程序中各个对象的创建关系[7].

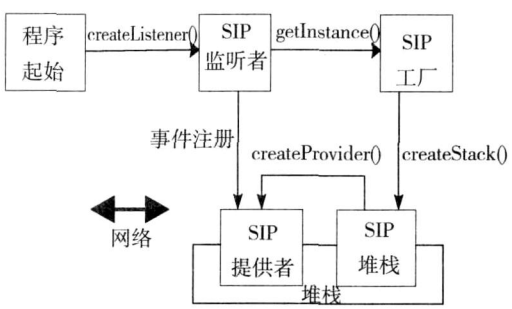

图 2　JAIN SIP的对象初始化过程

对象的创建过程为

1)JAIN SIP应用程序调用 createListener()方法创建出一个全局唯一的 SIP监听者对象;

2)SIP监听者通过 getInstance()方法创建 SipFactory对象,SipFactory就是 SipStack的创建工厂;

3)Sip工厂调用 createStack()方法创建 Sip堆

© 1994-2009 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

栈对象;

4) Sip堆栈对象调用 createProvider()创建提供者,并将其私有的 SIP堆栈由 SIP提供者管理;

5)一个提供者成功建立后,SIP监听者对其进行注册,建立起消息/事件机制,从而完成初始化.

3 JAIN SIP会话的实现

使用 JAIN SIP创建会话过程,实现 UAC和 UAS之间的呼叫建立、连接和呼叫拆除.

3.1 SIP呼叫建立和拆除的流程

SIP呼叫建立和拆除流程如图 3所示.

图 3 SIP呼叫建立和拆除的流程

1) UAC向 UAS发 INVITE请求;

2) UAS收到 INRITE请求后向 UAC发 100/TRYING响应;

3) UAS向 UAC发 200/OK响应;

4) UAC收到 200/OK响应后,发送 ACK请求给 UAS;

5) UAC和 UAS互发 REFER消息,双方进行通话;

6) UAC向 UAS发 BYE请求,请求通话结束;

7) UAS收到 BYE请求后向 UAC发 200/OK响应,通话结束.

3.2 SIP呼叫建立和拆除的实现

JAIN SIP程序的初始化.程序的运行主体放在 init类中:

```
public void init() {
…
//首先 UAC和 UAS分别生成 sip堆栈.
sipStack = sipFactory. createSipStack(properties);
…
//每个 sipStack都分别通过 sipFactory类的相//应方法创建报头、地址和报文的工厂.
headerFactory = sipFactory. createHeaderFactory();
addressFactory = sipFactory. createAddressFactory();
messageFactory = sipFactory. createMessageFactory();
…
//监听点的生成是通过 sipStack类的//createListeningPoint()方法来实现.
udpListeningPoint = sipStack. createListeningPoint("127.0.0.1", 50*0, "udp");
…
//生成监听点后创建 SIP提供者,并通过//sipProvider类的 addSipListener()方法对其//进行注册.
sipProvider = sipStack. createSipProvider(udpListeningPoint);
Shoot* listener = this;
sipProvider. addSipListener(listener);
…}
```

UAC部分的实现.在通信过程中,主要使用 INVITE请求、ACK请求、BYE请求、CANCEL请求.其中 INVITE请求是通过 init类实现的:

```
public void init() {
…
//UAC发送 INVITE请求发起会话.
Request request = messageFactory.
createRequest(requestURI, Request INVITE, callIdHeader, cSeqHeader, fromHeader, toHeader, viaHeaders, maxForwards);
…}
```

而 ACK请求、BYE请求和 CANCEL请求是在同一个类 processResponse中实现的,UAC通过 response类中的 getStatusCode()方法判断收到的响应类型,并通过 cseq类中的 getMethod(). equals()方法来判断前次请求消息类型:

```
public void processResponse(ResponseEvent responseReceivedEvent) {
…
if (response. getStatusCode() = Response. OK)
{ if (cseq. getMethod(). equals(Request INVITE))
{ //当收到的响应类型为 200/OK响应,前次//请求消息类型为 INVITE请求时,UAC通//过 dialog类中的 sendAck()方法发送 ACK//请求.
ackRequest = dialog. createRequest(Request ACK);
dialog. sendAck(ackRequest);
//创建会话连接
dialog. sendRequest(sipProvider. getNewClientTransaction(dialog. createRequest("REFER")) } else if (cseq. getMethod(). equals(Request CANCEL)) { if
```

© 1994-2009 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

```
(dialog.getState() == DialogState.CONFIRMED)
{ //通话结束时 UAC通过 dialog类中的 //sendRe-
quest()方法发送 BYE请求.
Request
byeRequest = dialog.createRequest(Request.BYE);
ClientTransaction ct = sipProvider.getNewClientTrans-
action(byeRequest);
dialog.sendRequest(ct); }}
…}
```

UAS部分的实现. requestEvent类为 UAS提供了一个 getRequest()方法用来接收请求,通过 getMethod().equals()方法来判断收到的请求消息类型,从而做出相应的响应.

```
public void processRequest(RequestEvent requestEvent) {
Request request = requestEvent.getRequest();
ServerTransaction
if(request.getMethod().equals(Request.INVITE))
{…}
else
if(request.getMethod().equals(Request.ACK))
{…}
else
if(request.getMethod().equals(Request.BYE))
{…}
else
if(request.getMethod().equals(Request.CANCEL))
{…}
…}
```

如果 UAS收到 INVITE请求,则通过 messageFactory类中的 createResponse()方法创建 100/TRYING响应:

```
public void processInvite() {
…
Response
response = messageFactory.createResponse(Response.
TRYING, request);
…}
```

如果通话准备就绪,通过 inviteTid类中的 sendResponse()方法发送 200/OK响应:

```
private void sendInviteOK() {
…
if(inviteTid.getState()! = TransactionState.COM-
PLETED)
inviteTid.sendResponse(okResponse); }
…}
```

进一步通过相应方法创建消息体.

如果 UAS收到 ACK请求,则通过 getSource()确认会话链接:

```
public void processAck() {
…
SipProvider
provider = (SipProvider) requestEvent.getSource();
…}
```

如果 UAS收到 BYE请求,则通过 createResponse()方法创建 200/OK响应,并通过 messageFacto()方法发送该响应,通话结束:

```
public void processBye() {
…
Response response = messageFactory.createResponse
(200, request);
serverTransactionId.sendResponse(response);
…}
```

如果收到 CANCEL请求,则通过 createResponse()方法创建 200/OK响应,通话结束:

```
public void processCancel() {
…
Response response = messageFactory.createResponse
(200, request);
…
inviteTid.sendResponse(response);
…}
```

3.3 呼叫建立和拆除日志

JAIN SIP提供了一个 SERVER_LOG方法来创建通话日志,通过它生成的文本文件,我们可以清晰的看到整个通话连接和结束过程. UAS的日志片段如下:

```
…
from = "127.0.0.1:5060"
//请求发起者的 IP地址及端口号
to = "127.0.0.1:5070"
//请求接收者的 IP地址及端口号
…
firstLine = "INVITE sip:LittleGuy@127.0.0.1:5070
SIP/2.0"
//接收 INVITE请求,并标识接收者
//的 SIP地址及 SIP版本号
…
```

© 1994-2009 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

firstLine = " SIP/2. 0 100 Trying"
/ /发送 100 Trying响应
…
firstLine = " SIP/2. 0 200 OK"
/ /发送 200/OK响应
…
firstLine = "ACK sip: 127. 0. 0. 1: 5070 SIP/2. 0"
/ /接收 ACK请求
…
firstLine = " REFER sip: LittleGuy@127. 0. 0. 1: 5070 SIP/2. 0"
/ /接收 REFER消息
…
firstLine = " REFER sip: BigGuy@127. 0. 0. 1: 5060 SIP/2. 0"
/ /发送 REFER消息
…
firstLine = "BYE sip: 127. 0. 0. 1: 5070 SIP/2. 0"
/ /接收 BYE请求
…
firstLine = " SIP/2. 0 200 OK"
/ /发送 200/OK响应
…

4 结 语

SIP协议作为下一代网络的核心协议,以其易于扩展、开放的业务生成环境、对移动性的支持等特点而具有广泛的应用前景,而基于 Java技术制定的 SIP协议栈规范 JAIN SIP将对 SIP消息的处理转化为对消息和事件的处理,极大地提高了开发的效率. 本文对基于 JAIN SIP的会话过程进行了设计分析,进行了相应程序的设计,对实现结果进行了模拟. 本文对开发基于 JAIN SIP协议栈的业务具有一定的参考作用.

参 考 文 献:

[1] 周 宇,管海兵,白英彩. 基于 JAIN SIP的 SIP实现技术[J]. 计算机应用与软件, 2005, 22(3): 54 - 56.
[2] 欧阳明星,程 剑. 使用 JAIN SIP开发基于 SIP协议的应用[J]. 计算机应用, 2005, 25(3): 493 - 495.
[3] NATHAN M R. Jain Sip Tutorial[EB/OL]. 2003 - 07. http: //java. sun. com/products Jain.
[4] JAIN SIP 1. 0 Specification[EB/OL]. 2003 - 07. http: //java. sun. com/products jain.
[5] ROSENBERG J, SCHULXZRNNE H, CAMARILLO G, et al RFC3261 SIP: Session Initiation Protocol[S]. IETF, 2002.
[6] SUN Microsystems JAIN SIP Tutorial[EB/OL]. 2004 - 06. http: //java. Sun. com/products/jain/.
[7] LEWIS J, LOFTUS W. Java程序设计基础[M]. 3版. 王锦全,译. 北京:清华大学出版社, 2004.
[8] 白建军,彭 晖,田 敏,等, CAMARILLO G. SIP揭密[M]. 译. 北京:人民邮电出版社, 2003.

(编辑:王 萍)

© 1994-2009 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net