

基于 JTAPI 的调度服务器基本功能的设计与开发

田文锋, 卜宪德, 郭云飞, 朱 亮

(中国电力科学研究院, 江苏 南京 210000)

摘 要: 随着 IP 化技术的发展, 调度设备的更新, 多媒体调度系统比传统调度系统更加直观性与人性化。因此电力调度系统也引入了软交换来开发多媒体调度系统。在深入研究 CTI 技术及其 JTAPI 标准协议接口的基础上, 结合电力软交换调度系统的需求, 提出了基于思科统一通信平台的多媒体调度系统架构, 对框架中的调度服务器的基本外呼、多方会议、强插、强拆等功能, 给出了在 JAVA 平台上基于 JTAPI 标准接口开发的具体实现方法。

关键词: JTAPI; 多媒体调度; 会议; 强插; 强拆

1 概述

在电力系统, 指挥调度的作用非常重要。调度员需要跟电厂、变电站、供电所等上下级单位进行联络, 随时根据生产情况进行作业调整, 指令都要求尽可能快速准确地下达到下级各单位。目前, 随着电力专网的不断健全和完善, IP 化技术的不断发展, 调度交换机的不断更新换代, 软交换在电力通信网中的应用越来越广泛, 在传统调度系统工作的同时, 基于软交换的多媒体调度系统可以成为系统扩展和相互备份的最佳选择^[1]。

本文在研究思科统一通信平台的基础上, 提出了基于软交换的多媒体调度系统框架, 重点研究框架中的调度服务器基本电话功能实现问题, 利用 CTI 技术中的 JAVA 语言电话应用程序接口 (JTAPI), 编程实现调度服务器的基本功能, 给出了具体的实现步骤。这种基于软交换的调度服务器无论是在功能方面还是扩容性方面具有传统调度系统无法相比的先进性, 整个系统还可以向统一通信和 NGN 网络进行平滑过渡。

2 JTAPI 呼叫模型

2.1 CTI 与软交换的关系

CTI 是计算机电信集成 (Computer Telecommunication Integration) 技术^[2], 涵盖了数据通信网络和传统语音通信网络的内容, 是由传统的计算机电话集成 (Computer Telephony Integration) 技术演变而来的。CTI 技术是随着电信技术计算机技术的发展而产生的, 它可以把电话的通信功能和计算机的数据处理功能很好的结合在一起, 不仅可以实现传统的电话语音业务, 还可以实现各种多媒体增值业务。CTI 技术的主要应用包括有交互式语音应答 (IVR)、呼叫中心、统一消息处理、IP 电话等。目前, 多采用 CTI 中间件的方法来满足 CTI 应用的需要, 即在交换机和应用程序间建立了一个中间层, 屏蔽了交换机。这种方法具有松耦合的优点, 应用程序的实现与硬件无关, 具有很强的可移植性。

软交换^[3]的基本含义就是把呼叫控制功能从媒体网关中分离出来, 通过服务器上的软件实现基本呼叫控制功能。软交换的功能是以媒体网关控制器为基础, 采用开放式应用程序接口, 允许在交换机制中灵活引入新业务。呼叫中心和 IP 电话作为 CTI 技术的两种典型应用, 都与软交换具有紧密的联系, IP 电话构成了软交换的底层通话技术的媒体基础, 而呼叫中心构成了软交换中业务节点的技术基础, 并为软交换提供了一个具体的业务应用切入点。另一方面, 软交换不仅是为了提供媒体传输方面的透明性, 更重要的是能够为第三方灵活地提供各种增值业务奠定基础, 这两点与 CTI 技术的发展目标完全是吻合的。因此, 通过与 CTI 技术进行组合, 可以完成软交换所要求的部分功能, 两种技术可以完美融合。

2.2 JTAPI 呼叫模型

JTAPI (Java 电话应用程序接口)^[4] 就是为 CTI 中间件服务的, 它定义了一组跨平台、跨厂家的电话应用程序对象模型, 帮助应用程序与交换机交互信息。JTAPI 接口标准是 CTI 技术的三种接口标准 (TAPI、

TSAPI、JTAPI) 的其中一种, JTAPI本质上是一套可重用的语音呼叫控制对象, 使应用程序能够运行在任何带有Java虚拟机和JTAPI子系统的计算机上, 独立于任何操作系统和硬件平台, 从而支持跨平台的应用。

JTAPI呼叫模型^[5]共有六个Java对象组成, 这些对象以Java接口的形式定义在核心包中, 每个呼叫对象都代表电话系统中的一个物理上或逻辑上的实体, 这些对象之间通过一些方法联系在一起, 共同描述电话呼叫。图1是JTAPI最基本的电话呼叫模型, 此模型是搭建CTI程序的基础, 它可以以树状结构表现出来, 根据这幅树状电话模型图, 可以很方便的实现调度服务器的一些基本功能。

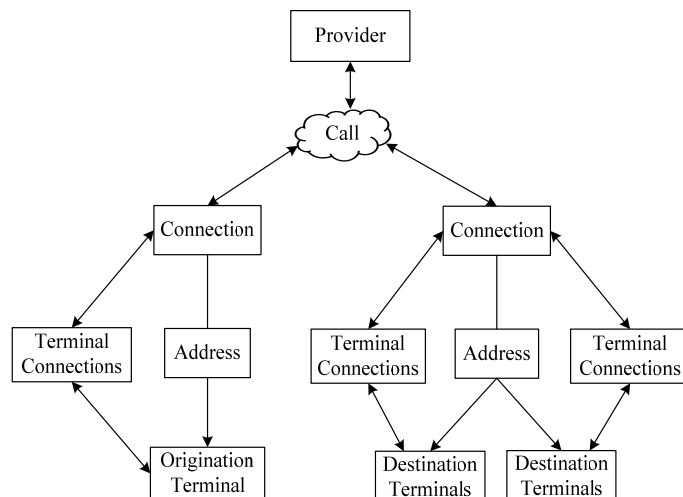


图1 JTAPI基本电话模型

下面对电话模型中涉及的最主要的几个实体进行简单的介绍:

(1) Provider 对象

Provider对象是对和CTI技术相关的硬件设备的一个抽象, 它把具体的硬件设备和程序员隔离开来, 隐藏了电话系统特定服务的细节, 为Java程序员提供了独立于设备系统的电话服务, 使系统简单化。利用Provider, 可以轻易的实现CTI程序和硬件的无关性, 即程序不变, 底层硬件可以从一种型号换成另一种型号。

(2) Call对象:

Call对象代表一个电话呼叫, 是对一次呼叫过程进行抽象, 它的属性记录了Provider和呼叫参与者之间的信息交流。

(3) Address对象:

Address 对象是对电话号码的抽象。它是对一部逻辑电话的描述, 所以一个Address 对象可能对应了多个物理话机。

(4) Terminal对象:

Terminal对象描述了一部具体的物理话机和其相关的属性。每个Terminal 对象可以有一个或者多个与之相关联的Address, 如一机多线、多机一线的电话。

(5) Connection对象:

Connection 对象是对Call和Address 对象之间的通讯连接关系的抽象。Connection对象有多种连接状态, 每一种状态都表明了Call与Address对象之间的不同关系, 通过对Connection 对象的属性查看, 可以分析出当前Call 对象与Address 对象的连接状态信息。

(6) TerminalConnection对象:

相对于Connection来说, TerminalConnection代表了当前Connection与一个特定Terminal之间的关系, TerminalConnection又称为物理连接, Connection相应的称为逻辑连接。

Provider、Call、Connection、TerminalConnection等核心对象都有状态, 这些状态从不同的角度描述电话系统的状态, JTAPI提供了对象观察者(Object Observer)的事件通知机制将各种呼叫对象的状态进行实时

发布，在呼叫过程中可以随时得到呼叫模型对象的当前状态及变化，以通过调用相应的函数做出反应。

3 多媒体调度系统方案

思科统一通信^[6]是基于IP的通信系统，它集成了语音、视频、数据以及移动产品和应用，思科统一通信的部署环境需要依赖CUCM为其提供呼叫处理和呼叫路由功能。CUCM（思科统一通信管理器）是思科的软交换产品，作为思科统一通信系统中的呼叫处理组件，它将企业电话通信特性及功能扩展到数据报电话通信网络设备中。这里所说的数据包电话通信网络设备包括IP电话、媒体处理器、VoIP网关以及多媒体应用。其他的数据、语音和视频服务可以通过CUCM API来与IP电话通信解决方案相互作用。

CUCM 服务器会使用 SIP 来与思科 IP 电话进行通信，并且通过这种方式来建立和拆除呼叫，以及提供补充服务。在呼叫建立之后，媒体流的交互就直接发生在思科 IP 电话之间，CUCM 就不在参与到呼叫中了，只有在呼叫建立、拆除，以及使用一些特性的时候，CUCM 才会参与进来。在交互过程中，电话会使用 RTP（实时传输协议）来承载音频流。

本研究提出的多媒体调度系统^[7]就是在思科统一通信平台基础上，通过开发多媒体调度服务器和调度台，提供电力多媒体调度通信服务。系统由CUCM、MCU（多点控制单元）、调度服务器、调度台、数据服务器、录音录像服务器、IP电话、软件电话等组成，系统架构如图 2 所示。

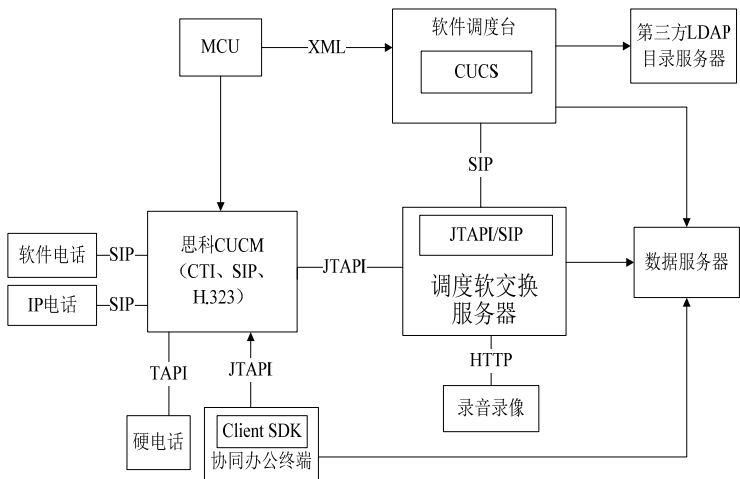


图 2 多媒体调度系统架构

考虑到软交换调度服务器^[8]以及调度台的可扩展性以及对平台的依赖性，呼叫相关的通信协议采用 SIP 协议，将来可以同其他厂家的通信平台进行互联互通，而 JTAPI 协议作为内部协议，思科统一通信的相关产品都可以互相通信。

调度服务器作为网关，将接收调度台发送的调度指令并进行解析，根据要求将转换成 JTAPI 接口，同 CUCM 进行通信，配合调度台实现调度通信的基本功能。调度服务器软件底层采用 JTAPI 开发接口，其中 JTAPI 呼叫模型共有六个 Java 对象组成，应用程序通过分析来自终端的 SIP 数据包，将调用这些 Java 对象来实现呼叫。

4 调度服务器基本功能实现

调度服务器是调度系统的服务器端，是整个系统的核心。调度台或调度电话信息最终将集中到调度服务器上进行分析处理。调度服务器通过 JTAPI 标准接口来控制并处理调度和电话信息，不仅可以处理如接听、外拨等基本通话功能，还能处理强插、强拆、会议等复杂通话功能，还可以将电话及系统的状态信息通过广播方式对所有相关的调度台进行动态更新，使调度员实时掌握最新数据，并保证数据信息的一致性。

调度服务器是基于 JAVA 平台上编程实现，可以独立于任何操作系统和硬件平台^[9]，支持跨平台应用，

系统具有很高的可移植性，并且很容易对功能进行扩充。调度服务器通过JTAPI标准接口和CUCM一起完成呼叫处理功能，并通过与调度台和后台数据库的交互，实时处理调度台的需求信息并将相关数据存入数据库中。调度服务器收到调度台发送的某种通讯需求的数据包后，首先按照约定要求对其进行解析，得到具体需求和相关参数，然后通过调用JTAPI接口与CUCM进行交互，实现所需的通讯功能。这些基本通讯功能是调度服务器的核心功能，下面对这些基本通信功能的实现进行说明。

4.1 两方呼叫功能

呼叫功能也就是基本的外呼功能，用于建立一方与另一方的通话。这一步也是实现呼叫强插、强插和组成会议等功能的第一步，先建立一个两方呼叫，然后通过调用 JTAPI 方法实现对该呼叫的强插、强拆或添加会议成员组成会议等。两方呼叫功能的实现是通过标准 JTAPI 接口中的 Provider.createCall()和 Call.connect()方法实现。图 3 为呼叫功能实现的流程图。

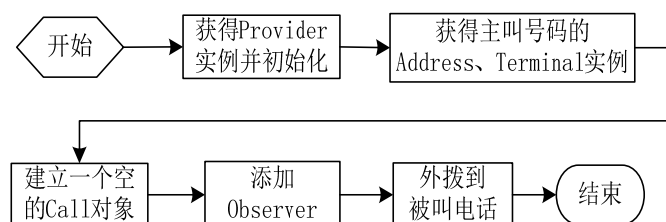


图 3 呼叫功能流程图

基本实现步骤如下：

(1) 获得 Provider 实例：首先通过 JTAPIPeerFactory 获得一个 JTAPIPeer 对象实例；然后使用 JTAPIPeer 对象的 getProvider() 方法获得一个 Provider 对象并初始化，语句如下：JtapiPeer.getProvider(providerName, login, passwd); 其中 ProviderName: 为 CUCM 的 IP 地址；login: 为赋予控制设备权限的用户；passwd: 为用户的密码。当获得一个 Privoder 对象后，即与 CUCM 建立了 TCP 连接，实现了与 CUCM 的握手。

(2) 获得主叫电话的 Address 实例：通过 Provider.getAddress()方法获得当前用户控制线路上的电话号码。

(3) 获得主叫号码的 Terminal 实例：根据 Address 实例，通过 Provider.getTerminal()方法得到其对应的 Terminal 实例。

(4) 建立一个 Call 对象实例：通过 Provider.createCall()方法产生一个空的 Call 对象。

(5) 添加 Observer: 为各种对象添加 Observer，通过 Observer 可以对整个外拨过程进行消息分析。在 Provider 对象初始化以后，调用 provider.addObserver()将 Observer 添加到 Provider 对象上；在创建 Address 和 Terminal 实例之后，调用 addCallObserver()方法将 CallObserver 添加到 Address 和 Terminal 对象上，只有在 CallOberver 关联到 Provider 对象上的 addresses 和 terminals 之后 JTAPI 才能进行拨打电话；在创建 Call 对象实例之后，调用 addObserver()方法将 Observer 添加到 Call 对象上。

(6) 外拨被叫电话：调用 Call.connect()方法实现两方呼叫连接。调用该方法需提供主叫 Terminal 对象和主叫 Address 对象，并提供被叫 Address 对象。Call.connect()会返回呼叫过程中发起端和目标端的两个 Connection 对象。

4.2 会议功能实现

会议是指将多个成员加入到同一个电话呼叫中，同时进行交流。会议由发起方和发起方定义的会议成员共同参与。会议功能的实现首先需要建立会议发起方与其中一个会议成员之间的呼叫，然后依次将其它的会议成员添加到此呼叫中，实现多方会议。如图 4 为会议功能的流程图。

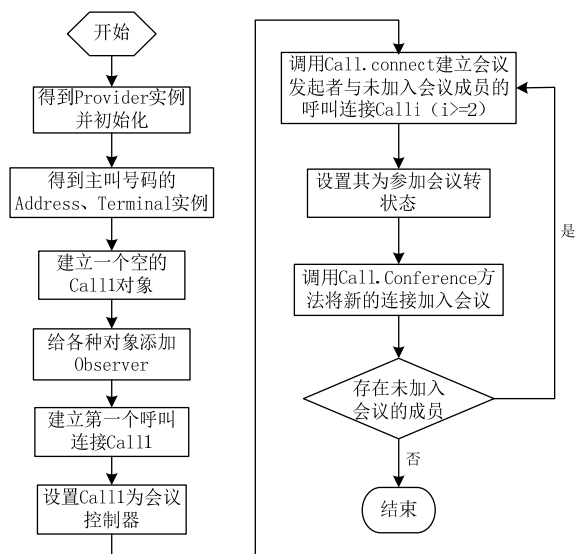


图4 会议功能流程图

会议功能的实现步骤:

(1) 建立会议发起者与会议成员之间的一个普通呼叫 Call1, 其基本实现与建立一个普通外拨操作类似, 通过 setConferenceController()方法将 Call1 设置为会议控制器。

(2) 依次向呼叫中加入其他的会议成员。新建一个会议发起者与未加入会议成员的呼叫 Call2, 调用 setConferenceEnable()方法设置其为可参加会议状态。

(3) 调用 Call1.conference(Call2)方法, 将 Call1 和 Call2 中的两条呼叫记录合并为一条呼叫记录, 形成三方通话。

(4) 若还有未参加会议的成员, 新建一个会议发起者与未加入会议成员的呼叫 Call3, 调用 setConferenceEnable()方法设置其为可参加会议状态, 调用 Call1.conference(Call3)方法合并为一个呼叫记录, 重复本步骤, 直至所有成员都加入会议。

4.3 强插功能实现

强插是指调度员可以在监听通话的过程中, 可以强行插入到正在进行的通话中, 组成一个三方会议, 以对此通会话进行指导。强插流程与三方会议类似。

强插功现主要包含以下几个步骤:

(1) 首先建立一个两方通话的呼叫连接 Call1。得到此呼叫通话双方的其中一方地址(Address), 通过 setConferenceController()方法将 Call1 设置为会议控制器。

(2) 通过调用 Provider.createCall()方法建立一个新的呼叫 Call2, 并调用 setConferenceEable()方法将其设置为允许会议状态。

(3) 调用 Call2.connect()方法建立强插者与通话中的一方的连接。

(4) 通话建立成功后, 通过调 Call1.conference(Call2)方法建立会议。其中 Call1 是正在进行的通话, Call2 为准备强插者与正在通话的一方建立的通话。

4.4 强拆功能实现

强拆是指调度员在监听的过程中, 可以对任何一部正常通话过程中的话机强行拆断并与之通话。它的实现过程类似于呼叫转接, 区别在于呼叫转接操作的发起者是正在进行通话的一方, 而强拆操作的发起者不属于通话的任何一方。强拆功能的实现是将相关信息进行适合转接操作的处理后, 转入转接操作进行处理。

强拆过程的具体实现如下:

(1) 首先建立一个两方通话的呼叫连接 Call1, 得到其对应的呼叫其中一方的 TerminalConnection。

(2) 利用 `Provider.createCall()` 方法建立一个空的 `Call2` 对象, 并通过 `setTransferEnable()` 设置此通呼叫可以进行转接操作。

(3) 利用方法 `Call2.consult(Terminalconnection, number)` 建立呼叫协商。其中 `TerminalConnection` 为 `Call1` 需要拆除一方对应的终端连接, `number` 为强拆者的号码。

(4) 在通话协商成功后, 通过调用方法 `Call1.transfer(Call2)` 完成通话的转接操作, 实现原 `Call1` 中另一方与强拆者的号码通话, 实现强拆功能。

5 结论

本文针对电力调度业务的需求, 结合思科统一通信平台, 提出了一种新的多媒体调度系统架构, 架构中的调度服务器是系统的核心, 它在调度台与思科 CUCM 之间起到桥梁的作用, 调度系统中的几乎所有通讯功能都通过调度服务器实现。本文结合调度台与调度服务器之间的消息传送, 使用 Java 编程调用 JTAPI 标准接口实现了两方呼叫、多方会议、强插、强拆等基本调度功能, 通过调试运行, 可以很好的完成多媒体调度系统的基本调度功能。本系统基于统一通信软交换平台, 可以方便的集成多媒体业务, 这种在 Java 平台上基于 JTAPI 接口开发的调度服务器, 不仅具有强大的电话功能和跨平台移植特性, 而且具有开放的体系结构、标准结构和开放的应用编程接口, 大大加速了调度业务的开发、生成和部署。

参考文献

- [1] 刘相锋. 智能调度系统中的故障分析与研究[D]. 济南: 山东大学, 2011.
- [2] 吴卫凯. 基于呼叫中心的调度软件设计[D]. 长沙: 国防科学技术大学, 2006.
- [3] 桂海源, 张碧玲. 软交换与 NGN [M]. 北京: 人民邮电出版社, 2009.
- [4] 刘俊年, 李红, 李勇军, 等. 利用 JTAPI 技术实现基于 Internet 的 CTI 服务[J]. 四川大学学报, 2003, 40(5): 849-852.
- [5] 袁飞艳. 基于 VoIP 的机场调度系统的研究与实现[D]. 贵阳: 贵州大学, 2009.
- [6] Dennis Hartmann. Implementing Cisco Unified Communications Manager [M]. 北京: 人民邮电出版社, 2011.
- [7] 封晓燕, 崔燕明, 温永怡. 利用软交换实现多媒体调度通信的特色功能[J]. 电力系统通信, 2010, 31(8): 9-13.
- [8] 双锴, 杨放春. 软交换中呼叫关系模型与业务视图[J]. 北京邮电大学学报, 2009, 32(5): 53-57.
- [9] 林峰, 胡牧, 蒋元晨, 等. 电力调度综合数据平台体系结构及相关技术[J]. 电力系统自动化, 2007, 31(1): 61-64.

作者简介:

田文锋 (1984-), 男, 通信作者, 硕士, 江苏徐州人, 工程师, 主要从事电力通信软交换技术方面的研究, Email: tianwenfeng920@yahoo.com.cn;

卜宪德 (1978-), 男, 硕士, 工程师, 主要研究方向: 电力系统通信;

郭云飞 (1976-), 男, 硕士, 工程师, 主要研究为: 电力系统通信;

朱 亮 (1988-), 男, 学士, 工程师, 主要研究为: 电力系统通信。