第 2 章

PSTN、PBX 及呼叫中心业务

我们在第 1 章学习了 PSTN 和 VoIP 的基本概念和术语，在本章我们接着了解一下传统的电话网及交换设备。这些服务有的是读者已经熟悉的，有的可能没听说过。有一些传统业务在 VoIP 时代实现起来异常简单，而有一些业务可能已经不需要了。

为了更深入地理解这些业务，在本章我们也对一些基本的概念，如中继线、IP-PBX 等加以深入介绍。

另外，考虑到相当一部分读者可能把 FreeSWITCH 应用到呼叫中心或相关业务中，在本章我们也对呼叫中心业务进行了简单的介绍，并简单讨论一下在 FreeSWITCH 中的实现思路。

2.1 PSTN 业务

我们还是从传统的 PSTN 业务开始讲起。除了为用户提供基本的语音通话外，PSTN 还能提供一些附加的业务，如叫醒服务、呼叫转移等，为人们的生产和生活提供方便。这些业务有的可以由运营商为用户设置，有的也可以由用户拨打某些特殊的号码自行激活和取消。下面我们分类来介绍一下。

2.1.1 POTS

POTS（Plain Old Telephone Service）即普通老式电话业务。POTS 是沿用国外的叫法，在国内我们称之为新业务。当然，新业务这种叫法这还是沿用数年前的叫法。这些业务有的是收费的，有的是不收费的。它们的接入号码通常以“ * ”开头。古老的话机是转盘式的，使用脉冲方式拨号，只能拨 0 ~ 9 的号码，现在在现实生活中已很少用了，但大家从电影上经常能看到。现代的话机多为按键式，使双音频方式拨号，上面有 0 ~ 9、* 和 # 字等键。其中，* 字键通常读作“星”[⊖]。有的话机上还有 A、B、C、D 键，但这些键很少用到。

⊖ 有些运营商的话务员也将“*”读作“米”，笔者觉得不是很通用。

另外有的话机上还有各种其他的功能键，如号码翻查、重拨等。在某些新业务中（如三方通话），会用到话机上的叉簧，快速拍一下可以给交换机传递相关信号，某些话机上有专门的 Flash 键、R 键或闪断键。下面仅列举几种典型的业务：

- ❑ 缩位拨号（Abbreviated dialing）。通过事先登记的代码代替长号码，可以减少记忆难度，加速拨号。比如，拨“ **1”可以拨叫之前指定的号码（比如 12345678），该功能比较实用。
- ❑ 呼叫前转（Call Forwarding，有时也称呼叫转移）。分三种基本情况：无条件转移，即任何来电转移至事先登记的号码；遇忙转移，若被叫忙，则转移；无应答转移（或称久叫不应转移），若指定时间内无应答，则转移。其中大部分交换机上无条件转移的登记方式为“ *57* 电话号码 #”，取消方式为“ #57#”。登记成功后，所有到该话机的来电会转到所登记的电话号码。如在话机 A 上操作“ *57*B#”，则所有对 A 的呼叫都会转移到 B 上。适用于将家里或办公室电话转移到手机上的情况。运营商也经常使用该功能作一些特殊的业务，如改号通知，即通过后台操作将某一号码转移至特定的语音平台，实现类似“您拨的电话已改号，新的号码是 XXX”的功能。
- ❑ 新转移方式。在移动电话及小灵通等出现后，又有了新的转移方式，如不在服务区转移等。
- ❑ 立即热线（Hotline）。拿起电话不用拨号即自动拨打某号码，方便拨号。笔者在北京某银行网点用过该项业务，拿起电话直接连接到他们的自助语音服务。
- ❑ 延迟热线（Delayed Hotline）。与立即热线差不多，区别是摘机后会延迟一段时间（如 5 秒）再自动拨号，在特殊场合有用。
- ❑ 呼叫等待（Call Waiting）。被叫忙时，主叫仍会听到正常的回铃音（或个性化的语音提示：请不要挂机，您拨打的电话正在通话中……），而交换机会通过特殊的提示音提示有新电话呼入，被叫可选择是否接听，或在两者间切换。
- ❑ 三方通话（Three Way, Conference Call）。通过比较复杂的操作实现三方通话，某些交换机支持最多五方的通话（会议电话），更多方的电话会议系统需要专门的平台。
- ❑ 来电显示（Caller ID Presentation）。就是在被叫话机上显示主叫方的电话号码。
- ❑ 呼出限制（Call Barring，又称密码限呼）。呼叫某类号码（如长途电话）前提示先输入密码，在电话费还是很贵的年代比较有用。该功能可以使用密码限制话机能否打长途等，也可以限制小孩乱打电话。
- ❑ 免打扰服务（Do Not Distrurb，DND）。登记该业务后，如果有来电，交换机会提示主叫用户被叫用户不想被打扰。不过，在实际应用中，直接拔掉电话线比登记这个容易多了。
- ❑ 叫醒服务（Alarm Call，又称闹钟服务）。登记后在相应的时间电话会振铃。在这个手机异常普及的年代相信一般人不会用这个功能了，但在酒店应用中还是非常普遍的。
- ❑ 遇忙回叫（Completion of Call to Busy Subscriber 或 Auto CallBack，CCBS）。如果被叫忙，则主叫可以按一个特殊的号码以登记该业务，待被叫空闲后双方话机会自动振

铃，接听后双方进行通话，省了好多重复拨叫的操作。不过，该业务一般限制主、被叫用户都在同一交换机上，因而实际应用意义不大（最早发明该业务时可能人们的交际范围不广，大家都在同一交换机上）。

2.1.2 商务业务

商务业务是由运营商提供的，主要是为企业用户服务，一般有以下几种。

（1）模拟中继线

模拟中继线又称为用户小交换机[⊖]，它主要提供号码连选功能。典型应用是提供一个总机号（又称引示号）及若干条中继线（实际上就是普通的电话线）。当有人拨总机号时，交换机会根据指定的策略选择一条空闲的中继线呼入。而用户端通常会接 PBX 设备，下设分机。当用户呼出时，通过用户端的 PBX 设备选择一条空闲的线路。用户可选择是否显示总机号。

（2）数字中继线

如果用户需要的中线数量较多，数字中继线能提供更稳定的服务，设备通常支持 2Mbit/s 的一号信令或 30B+D 的 ISDN 信令，有的设备也支持 7 号信令。

（3）虚拟网

虚拟网又称商务组（Business Call Group，BCG）或汇线通（Centrex）业务。虚拟网主要提供在无需用户端 PBX 设备的情况下，实现网内（组）电话互拨小号，通常小号间的通话是免费的，但要比普通电话多收部分月租费。

虚拟网与模拟中继线的区别是，它的每路电话都是直线，可以直接呼入呼出，但需要占用更多的 PSTN 号码资源。它与普通电话的区别是，网内可以互拨小号。

（4）立即计费

传统的 PSTN 需要通过额外的系统来计算通话费用，通常需要有一段时间的滞后。而立即计费主要用于酒店等需要立即计费的场合，通常使用 ISDN 信令配合用户端的话务台软件实现。

（5）VPN

VPN（Virtual Private Network) 的全称是虚拟专用网，有别于 Internet 上的 VPN。它主要是用在连接大型企业在不同城市的分支机构中，可实现公司内部互拨小号，有时也称为广义虚拟网。

2.1.3 其他增值业务

传统的语音业务所带来的收入比例越来越低，因此，各大运营商都纷纷推出基于数据库和计算机系统的各种增值业务以增加收入。这些业务包括预付费业务（电话卡类业务等）、800 业务、400 业务以及彩铃、电话秘书台（语音信箱）等。当前，受苹果商店和安卓市场的影响和启发，随着云和大数据时代的到来，大家也都意识到靠几个大厂家研发的产品和服

⊖ 注意，这里说的模块中继线并不是“中继线”本身，而是指一种功能，可以理解为模拟中继线功能或用户小交换机功能。

务是永远跟不上时代发展的，因而各大运营商也在积极研发和试点新型的能力开放平台，力求将沉睡在运营商网络中的通信能力、数据等通过 API 或其他接口方式提供给开发人员和用户，吸引更多的开发者，从而创造更丰富的应用，为社会创造更多的价值。

2.2　PBX 业务

PBX（Private Branch eXchange）的全称是专用小交换机。该设备一般安装在企业内部。PBX 的上端通过运营商提供的模拟或数字中继线连接到 PSTN，而下端则直接接企业内部的话机。

企业使用 PBX 的好处是可以自己控制内部呼叫，而且内部通话免费。它通常可以提供呼叫保持、自动选线、呼叫前转、呼叫转移等基本功能，比较高级的小交换机还可以提供自动总机、三方通话、语音信箱等功能。在此，我们仅就经常使用的呼叫转移业务和同组代答业务简述如下。

2.2.1　呼叫转移

企业用总机一般有人工话务员，话务员接听电话后可能会根据需要将来话转移到其他分机[⊖]。转移有两种，一种是称为盲转（Blind Transfer），即被叫一方不管三七二十一将来话转至第三方号码，至于第三方号码是否可用，是否有人接听，则全然不管；另一种称为协商转（Attended Transfer），被叫一方先通过一些操作将来话置于 Hold 状态，主叫一方听音乐，被叫一方呼叫第三方号码，第三方接听后，被叫可询问第三方是否愿意接听，然后再执行转移操作或挂机。关于这两种转移操作，我们在 15.1 节还会详细讨论。

2.2.2　同组代答

通过逻辑上将一些分机分配到一个组，组中其他电话振铃时，组内的任意人可以拿起话筒拨叫一个特殊号码将正在振铃的某一分机上的呼叫接到本机上来。在办公室人数较少又不想漏接电话的情况下可使用本业务。

2.3　PBX 与中继线

用户或企业 PBX 要想打通外面的电话，或者外面的电话需要打进来，需要走运营商提供的中继线，以接入到 PSTN 网上去。理解中继线的概念对于理解 PBX 以及 PSTN 是非常重要的，中继的接入方式决定了我们如何拨号，读者在学习中也可以思考一下为什么使用这

⊖ 与上一节所述的呼叫前转不同（虽然人们有时也将前转称为呼叫转移，但在本书中，我们认为前转和转移是不同的概念），“前转”来自英文 Call Forwarding，是由交换机实现的，在电话到达被叫用户之前就被转移到其他号码了；而“转移”来自英文的 Transfer，是被叫用户接听后，将来话转移到其他号码。所以，我们认为，“前转”是一项业务，而“转移”指一个动作。

种拨号方式，我们能如何配置以提供更好的用户体验等。因此，在这里我们单独拿出一节来说明。

下面我们以模拟中继线为例，通过一则故事来说明中继线与 PBX 的关系。

假设我们刚开了一家公司，需要 7 部电话，于是向运营商（在 PSTN 交换机上）申请了 7 条模拟中继线。前面已经指出，实际上就是 7 条普通的电话线，只是运营商在 PSTN 交换机端对我们这 7 条线（也可以解释为 7 个号码）做了特殊的数据设置，将其逻辑上分为一个组，并为该组设了一个总机号。我们有幸选到了一个很酷的号码——88888888（它可以是一个虚拟的号码，或者是其中某一条中继线的真实号码）。而其他的中继线号则可能是 44440001 ~ 44440007。现在，我们把这 7 条线都接上话机。如果有人呼叫 88888888，则 PSTN 交换机会从 7 条线中自动选择一条空闲的线路呼入，因此某个电话会就会振铃。如果有多个电话呼入，只要同时呼入的电话不超过 7 个，我们的电话就都有机会振铃，因而我们可以同时对外为 7 个人同时提供服务。一般来说，当有电话呼入时，交换机有两种选线策略——顺序选线和循环选线。所谓顺序选线，就是每次都从 44440001 开始，寻找一条空闲的线路进行呼入；而循环选线则是每次都从上一次呼叫的下一个开始选起，使用这种选线方式，每个话机接到的电话数会比较平均。公司安装的 7 部电话的结构示意如图 2-1 所示。

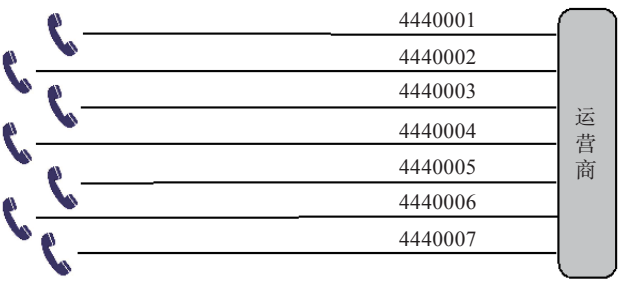

图 2-1 运营商为公司安装了 7 部电话

为维护企业形象，当有人呼出时，不管是从哪个分机呼出，都显示总机号 88888888。当然，也可以设置显示单线的号码（如 44440004），这个要在 PSTN 交换机端设置，一旦设置后，用户端不能动态更改。

一个月后，公司发展到 21 个人，因此需要 21 部电话。但由于一般不会出现所有人同时都在打电话的情况，故安装 21 条线有些浪费。因此我们买了一个小交换机，把原来的 7 条中继线接到小交换机的外线接口上，而把每个人的话机接到小交换机的内线口上，这样，每个人就都有了一个分机号，从 601 到 621，而 PSTN 端的配置不变，如图 2-2 所示。当客户打总机号时，PSTN 交换机仍然会选择一条线进入我们的小交换机，这时候，选线方式已经不像以前那样重要，因为现在是小交换机在接电话，对它来说，7 条线哪条都一样。就这样，小交换机接了电话，并播放“您好，欢迎致电某某公司，请直拨分机号，查号请拨 0……”，如果客户按某一分机号，则对应分机振铃，电话接通。

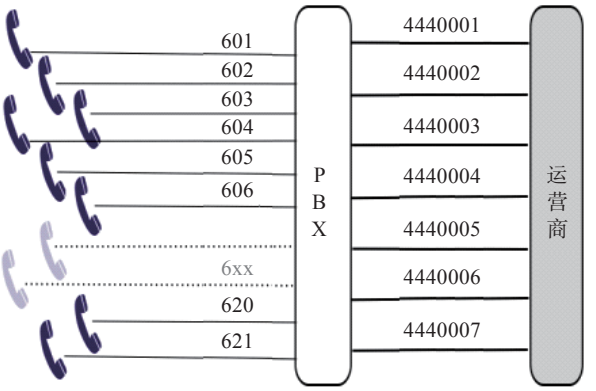

图 2-2　小交换机（7 条外线，21 条内线）

有了小交换机，内部通话就免费了。但出现了另外一个问题，就是如果拨打外线，则需要先拨一个特殊的数字，一般是 0 或 9。有的小交换机会送二次拨号音，即你拿起电话，听到小交换机的拨号音，拨了 0 之后，则听到外部 PSTN 交换机的拨号音，表明你可以拨打外线了。总之，小交换机会选择一条空闲的中继线对外呼叫。

上述例子中，21 : 7 称为集线比，即 3 : 1。集线比是由话务量决定的，如果同时通话的人数比较多，那我们可能会把中继线增加到 12 条，集线比就降为 21 : 12，约为 2 : 1 了。

即使增加了线路，也经常会遇到这样的情况：由于打进来的电话太多，占用了太多的线路，经常一个电话都打不出，因此，我们联系运营商，将中继线分为三组，其中 4 条只进不出，4 条只出不进，4 条能出能进。在电信术语中，分别叫做单出，单入和双向[⊖]，而北京联通则分别称为发专、受专和双向。当然，这种分配方式降低了总体线路的使用率，为此，我们把每个组都增加 1 条线，现在中继线总共达到 15 条。

又过了几天，有客户反映这样的情况，正常上班期间打电话经常无人接听，需要打好几遍；而同时，内部也有人反映往外打电话时有时拨 0 没反应，再试一次就好了。我们没有处理这种问题的经验，只好请教 PBX 专家，专家说可能是某条外线断了。因为，如果有一条线断了，当有电话呼入时，交换机仍会向主叫方送回铃音，跟被叫端没接话机是一样的。但到底是哪条线断了，却不好查。由于双方都是自动选线。我们只好将每条线都从小交换机上拔下来，接上话机试一试[⊜]，以确定是哪条线断了。

还算比较幸运，我们找到了断线的号码，联系运营商，很快修好了，把所有线路都插回小交换机，一切恢复正常。

⊖ 注意，对电信部门来说，出和入跟我们是相反的，因为我们（小交换机）的出，对应它们（PSTN 交换机）的入。

⊜ 某些小交换机也支持指定端口拨打功能，即在拨打真正号码前先拨几个特殊的数字，可以选择从指定的线路出去。实际上还有一种方法：我们知道，一般来说，每条单线的号码也都是可以呼入的，只是外人不知道而已，故只需要依次拨打所有的单线号码就可以知道是哪条线断了。当然，这依赖于 PSTN 交换机端的设置，有些城市（甚至同一城市不同的交换机都有不同的设置）默认设置单线是不允许呼入的。

几天后，老板又很幸运地搞到了一个新号码 66666666，该号码并未加入中继线组，而是直接扯了根线拉到老板办公桌上。为了能拨打内线，他不得不在办公桌上放两部电话，另一部专门打内线。后来技术人员小张仔细阅读了 PBX 的说明书，发现该小交换机功能还比较强，就进行了以下设置：将 66666666 这个号码接到小交换机上，仍给老板一个内线电话，同时在小交换机上进行设置，只有老板打出时才走 66666666 这个端口；而对于打入的电话，也不播放“欢迎致电 XX 公司…”，而是直接向老板电话振铃。这种拨入方式叫做 DID，即对内直接呼叫（Direct Inbound Dial）。

接下来，随着公司的发展，加入的中继线条数越来越多，维护起来更加复杂。比如，像我们刚才遇到的情况，其中有一条线断了，在很长的一段时间内根本不知道，即使知道了，要找到是哪条线也非常麻烦。后来，当公司发展到 100 人的时候，购买了新设备，并将模拟中继线换成了两条 E1 数字中继线，可同时支持 60 路通话。

公司发展一帆风顺，电话量也越来越多，公司有了很多分支机构，也有了更多客户，需要更复杂的语音菜单及更智能的电话分配策略，而更换专门的电话系统不仅价格昂贵，而且跟现有业务系统进行集成难度也很大。在综合考虑了多种解决方案以后，技术人员开始学习 FreeSWITCH……

2.4 IP-PBX[⊖]业务

在上一节中，我们最初买的模拟和数字小交换机是基于电路实现的，在这里我们将它们称为传统的 PBX。同时我们也欣喜地看到，我们的技术人员已经开始学习和研究 FreeSWITCH 了。FreeSWITCH 的默认配置就是一个家用或小型企业级的 PBX，它是由纯软件实现的，基于 IP 网进行通信，因而又称为 IP-PBX。

IP-PBX 首先是一个 PBX（Private Branch eXchange），它具有传统 PBX 的绝大部分功能。另外，由于使用了 IP 通信，它能通过 IP 网提供语音、视频以及即时消息通信。这些通信不仅可以在企业内部网上进行，也可以通过 Internet 在外网甚至 PSTN（Public Switched Telephone Network）电话间进行。

由于大部分 PBX 功能都是用软件实现的，因而实现起来成本相对来讲都非常廉价，并且非常易于增加新功能，如多方会议、使用 XML-RPC 等控制正在进行的通话、IVR、TTS/ASR（Text To Speech/Automatic Speech Recognition）、支持通过模拟或数字线路与 PSTN 网络对接，支持通过 SIP、IAX、H.323 或 Jingle（Google 对 XMPP 协议的扩展）以及其他协议与其他通信系统进行互联互通等。

与传统的 PBX 相比，IP-PBX 支持更多的新特性（或更易于支持某些特性）：

- ❑ 无限分机数量、无限自动话务台、无限语音信箱；
- ❑ 更易于通过 API 与其他应用系统（如 CRM 等）集成；

⊖ 本节大部分内容来自 Wikipedia: http://en.wikipedia.org/wiki/IP_PBX，http://zh.wikipedia.org/wiki/ 网络电话交换机。

❑ 支持远端电话分机、支持软电话；
❑ 支持高级用户接口，如电话跟随、统一消息、电话录音、语音邮箱、传真集成等；
❑ 根据时间路由、自定义路由规则；
❑ 支持从语音邮箱回拨、语音邮箱转 Email；
❑ 支持电话监听、耳语；
❑ 支持根据名字呼叫；
❑ 支持话务员控制台、拖拽转移、点击拨号等；
❑ 支持多人会议室；
❑ 支持高级 IVR（Interractive Voice Response）；
❑ 支持自动呼叫分配（Automatic Call Distribution，ACD）；

除了上面所述的这些特性外，IP-PBX 更易于部署，尤其是基于 IP 的通信更加廉价。但是，IP-PBX 并不是解决所有问题的良药，老技术向新技术过渡总要有些取舍。比如，在 IP 环境中，IP 电话终端更加智能，如摘机检测、挂机检测、收号等，这些原来由 PBX 或交换机实现的功能现在都在终端上实现了，因而在 PBX 上很难获得这些信息的细节。另外，当它与传统的 PBX 或 PSTN 网络对接时，还需要相应的 VoIP 网关来实现。当然，现在国内某些运营商也开始试验性地提供基于 IMS 或 SBC 的 SIP 中继，这样对接起来就方便多了。

2.5　呼叫中心

基于企业级的 PBX 和 IP-PBX 的通信还只是局限于基础的通信层。而随着企业规模的扩大及用户对服务要求的提高，企业更需要在业务逻辑和管理层方面为用户提供更好的服务。当这些服务可以通过远程电话支持的方式解决的情况下，一种称为呼叫中心的业务（Call Center）便产生了。

在呼叫中心中，有专门的话务员为客户提供服务。呼叫中心通常能同时处理大量的通话，并且为了给客户更好的服务体验，呼叫中心的通信系统通常通过技术手段与 CRM（Customer Ralationship Management，客户关系管理）系统集成。

本节我们简单介绍呼叫中心的基本概念及其在语音业务中的应用模式、技术发展情况以及 FreeSWITCH 在呼叫中心应用中的优势。

2.5.1　什么是呼叫中心

呼叫中心又称客户服务中心，它是一种基于 CTI 技术、充分利用通信网和计算机网的多项功能集成，并与企业连为一体的一个完整的综合信息服务系统，利用现有的各种先进的通信手段，高效地为客户提供高质量、高效率、全方位的服务。初看起来呼叫中心好像是企业在最外层加上一个服务层，实际上它不仅为外部用户，也为整个企业内部在管理、服务、调度、增值方面起到非常重要的统一协调作用[⊖]。

⊖　摘自百度百科：http://baike.baidu.com/view/4061981.htm。

通俗地讲，呼叫中心是企业或机构建立的以电话为主要手段，为客户提供服务与沟通的部门组织及信息系统。百姓生活中常见的 110、119、120 这些应急服务电话，及电信运营商的客服电话、电话银行等都是呼叫中心的具体应用。在呼叫中心中通常是由座席代表通过电话为客户提供相应的服务与沟通。根据呼叫中心业务量不同，可以同时处理的电话量和座席代表的人数也有所不同。较小规模的呼叫中心只有几个人，大规模的呼叫中心会达到几千人。

2.5.2 呼叫中心的历史

1956 年美国泛美航空公司建成了世界上第一家呼叫中心，在 20 世纪 80 年代，呼叫中心在欧美等发达国家的电信企业、航空公司、商业银行等领域得到了广泛的应用。20 世纪 90 年代中后期，随着中国经济的发展，呼叫中心概念被引入国内。今天，呼叫中心在家电企业、邮电、银行、航空、铁路、保险、股票、房地产、旅游、公共安全等众多的行业间搭建起了企业与客户、政府与百姓之间的一座桥梁，与百姓的日常生活息息相关。

呼叫中心技术的发展可以分为以下几个阶段。

（1）第一代呼叫中心

第一代呼叫中心最早出现在民航服务领域，用于接受旅客的机票预订业务。第一代呼叫中心的系统主要在早期 PBX 的基础上增加了电话排队功能，那时甚至不能称为呼叫中心，而称为热线电话，其全部服务由人工完成。

（2）第二代呼叫中心

IVR（Interactive Voice Responce，交互式语音应答）系统的出现，标志着第二代呼叫中心的开始。在呼叫中心中利用 IVR 系统可以将大部分常见问题交由系统设备通过语音播放、DTMF（双音多频，电话机上面的数字按键所发出的频率）按键交互解决。例如我们在日常生活中常用的 121121 天气预报、117 报时电话，通过电话银行进行余额查询、转账等业务都是通过 IVR 系统自动实现的。在第二代呼叫中心中，IVR 系统的大量使用，可以大大减少人工业务的受理数量和人工座席的工作强度，同时可以为客户提供 7 × 24 小时全天候、不间断的服务。

（3）第三代呼叫中心

随着计算机技术的发展，CTI（Computer Telephony Integration，计算机电话集成）技术的诞生与应用，标志着第三代呼叫中心时代的开始。CTI 技术实现了电话交换机系统与计算机系统的集成，即实现了语音与数据的同步。客户信息与资料采用数据库方式存储，座席代表可以在处理电话服务的同时可以从计算机系统中调取和修改客户信息数据，为客户提供个性化的服务。CTI 技术的使用，推动了呼叫中心更大范围地使用。与此同时，呼叫中心中出现了专门用于电话录音的录音设备，对座席代表与客户的通话进行录音、存储和查询。

相比之前的呼叫中心系统，CTI 技术的使用使得呼叫中心大部分功能实现了自动化。从客户电话接入到最终问题的解决，整个过程被完整地记录了下来。

（4）第四代呼叫中心

前三代呼叫中心均是以电话为主要的服务渠道。在 2000 年，伴随着互联网以及移动通

信的发展与普及，将电子邮件、互联网、手机短信等渠道接入呼叫中心，成为第四代呼叫中心的标志。第四代呼叫中心也称为多媒体呼叫中心或联络中心（Contact Center）[⊖]。它相对传统呼叫中心来说接入渠道丰富，同时引入了多渠道接入与多渠道统一排队等概念。

（5）下一代呼叫中心

目前，已经有厂商提出了第五代呼叫中心的概念。下一代呼叫中心的发展方向是在第四代多媒体呼叫中心的基础上，更多地融入了依托于互联网技术的媒体渠道与沟通渠道。例如：社交网络、社交媒体（如微博、微信等媒体渠道），依托于互联网的文本交谈、网上音频、网上视频等沟通渠道。

2.5.3　呼叫中心的分类

对于呼叫中心的分类可以有多种维度，如技术架构、呼叫类型、建设规模、功能、使用性质、呼叫方式、部署地点等。从呼叫方式上讲，它主要分为外呼型呼叫中心（如电话营销）、客服型呼叫中心（如客户服务）以及混合型呼叫中心（营销和客服融合）。除此之外，在这里，我们仅针对技术架构进行简单讨论。

按照呼叫中心系统所采用的技术架构的不同，呼叫中心可以分为交换机、板卡、软交换（IPCC）三种类型。

1. 交换机类型的呼叫中心

交换机类型的呼叫中心的呼叫中心系统是在交换机基础上构建而成的。呼叫中心系统平台主要由交换机系统、CTI 中间件、IVR 系统、录音系统组成。用于呼叫中心系统平台常见的交换机系统有：Avaya、西门子、华为、阿尔卡特 – 朗讯等。图 2-3 是一个典型交换机类型的呼叫中心平台系统架构图。

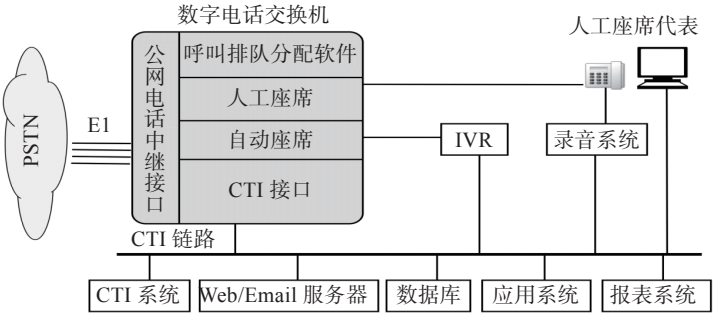

图 2-3　交换机类型的呼叫中心架构示意图

交换机类型的呼叫中心系统平台语音接入平台采用的是成熟稳定的交换机系统，采用专用硬件设备，具有产品成熟稳定、接入能力强、处理能力大等特点，适用于大中规模呼叫

⊖ Call Center 与 Contact Center 的缩写都是“CC”，由于 Contact Center 涵盖的业务面更广，因而现代大多数人倾向于使用联络中心这一概念。但呼叫中心这一名词已深入人心，因此，本书中仍然使用呼叫中心。

中心的建设。同时，交换机类型的呼叫中心系统平台具有系统架构复杂（需要由多个系统组成，通常会由多个厂家提供），安装部署及运维难度大，建设成本高等缺点。

2. 板卡类型的呼叫中心

板卡类型的呼叫中心系统平台语音接入平台采用语音板卡实现。语音板卡按照功能分为中继卡和用户卡两种类型，分别用于中继线的接入和座席端电话终端设备的连接。板卡类型的呼叫中心系统配合会议卡和传真卡还可以实现电话会议和电子传真的功能。随着 VoIP 技术的普及，一些语音板卡厂商也推出了自己的 IP 板卡，可以支持 IP 座席或 IP 中继的接入。图 2-4 是一个典型板卡类型的呼叫中心平台系统架构图。

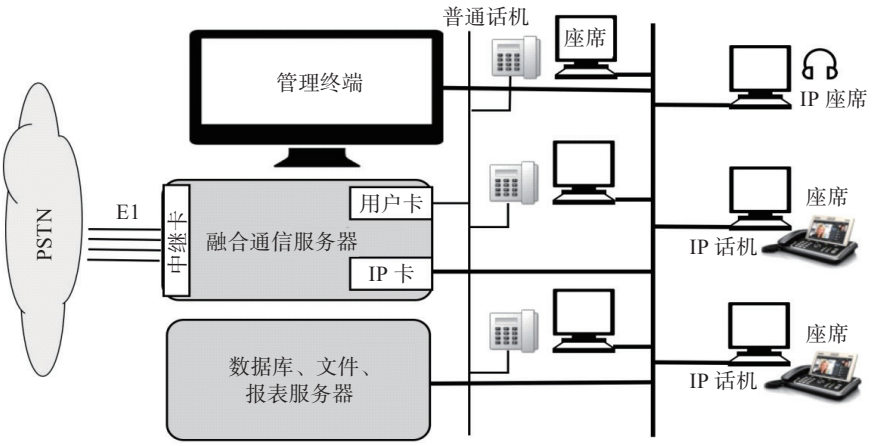

图 2-4 板卡类型的呼叫中心架构示意图

板卡类型的呼叫中心系统平台语音接入平台采用工控机加语音板卡的方式实现，其最大特点是系统结构简单。板卡类型的呼叫中心系统特别适用于中小规模呼叫中心的建设。板卡类型的呼叫中心系统安装部署及运维技术难度低，相对于交换机类型的呼叫中心系统其建设成本低。对于小规模的呼叫中心通常一台服务器就可以提供一套功能齐全的呼叫中心系统。

板卡类型呼叫中心的缺点是其受限于语音板卡的容量及服务器主板总线槽位数量，单台服务器对语音呼叫的处理能力有限（通常小于 120 路）。虽然语音板卡厂商也提供多台设备堆叠扩展的解决方案（通常采用过机卡的方式），但是实现难度较大、稳定性差，因而很少有厂商支持这种方式。板卡类型呼叫中心通常采用通用的硬件平台和软件系统，所有呼叫过程需要上层应用程序开发实现，其系统的功能和稳定性取决于开发商的技术能力和产品成熟度。相对于交换机厂商几十年的产品成熟度，国内厂商所提供的板卡类型呼叫中心解决方案在系统的稳定性和功能上还有很大的差距。

3. 软交换类型的呼叫中心

随着基于 VoIP 的软交换技术的发展，特别是一些优秀的开源软交换项目的出现，在呼叫中心系统建设方面出现了以软交换技术为核心的呼叫中心系统。软交换类型的呼叫中心不

仅解决了板卡类型呼叫中心接入能力受限于硬件板卡的问题，同时还具备板卡类型呼叫中心系统结构简单、部署灵活、低成本的优势。除 E1 接入外，现在有的运营商也提供 SIP 线路，较交换服务器也可以绕过 E1 网关而直接通过运营商的 SBC 接入 PSTN。图 2-5 是一个典型软交换类型呼叫中心的系统架构图。

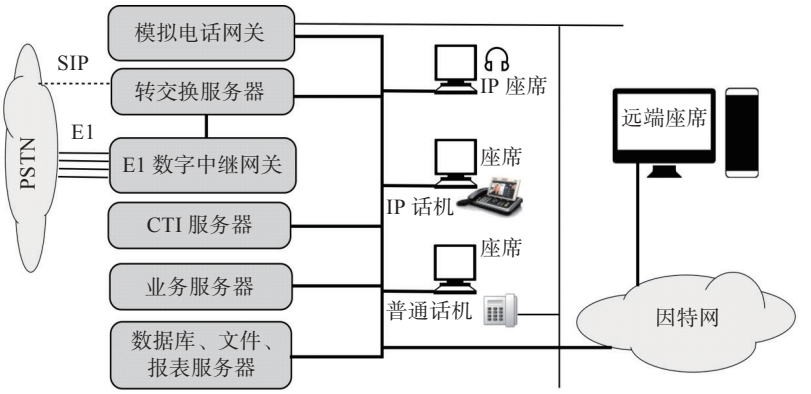

图 2-5　软交换类型的呼叫中心架构示意图

软交换类型的呼叫中心系统基于 VoIP 技术，具备先天的分布式部署的优势，特别适用于座席职场分散、接入分散的呼叫中心。软交换类型呼叫中心的系统容量可以通过多台服务器集群的方式平滑扩展，可以满足大型呼叫中心对于容量和性能的要求，是呼叫中心技术的发展方向。但是，软交换类型呼叫中心系统的性能和稳定性与呼叫中心厂商自身的研发能力有很大的关系。早期大多数软交换类型呼叫中心都是基于 Asterisk 或其他开源项目演变而来，受限于这些项目早期版本自身的问题，在市场上留下了性能差、不稳定的印象。这不是软交换技术本身的问题，而更多的是呼叫中心厂商自身技术能力的问题，它们大多采用“拿来主义”、“站在巨人肩膀上”的策略，在通信底层不愿进行太多的研发和投入，另外即使自己解决了某些问题一般也不会向开源社区公开，因而对开源软件的生态环境没有太大的贡献。

2.5.4　呼叫中心的主要技术指标

呼叫中心是一种劳动密集型产业。一个呼叫中心少则几个人，多则成百上千人。为了对呼叫中心工作人员进行有效管理，通常会通过一些量化技术指标来制定各种 KPI（Key Performance Indications，即关键绩效指标），作为衡量座席代表工作指标。

每个呼叫中心会根据其具体的业务类型、企业要求来制定自己的 KPI 指标，其数量和标准也不尽相同。常见的 KPI 指标有接通率、呼入项目占有率、呼出项目工作效率、服务水平、客户满意度、平均振铃次数、监听合格率、一次性解决问题率等。一般来说，这些指标的得分越高，绩效就越好。

❑ **接通率**：对于呼入业务类型的呼叫中心，接通率是指 IVR 终极服务单元的接通量与人工座席接通量之和与进入呼叫中的呼叫量之比；对于呼出业务类型的呼叫中心，接

通率是指座席呼出电话后接通量与呼出电话总量之比。

- ❑ **呼入项目占有率**：呼入项目占有率是指在某一统计时间段内，人工座席处理电话的总时长与实际登录系统时长的比率。
- ❑ **呼出项目工作效率**：呼出项目工作效率是指在某一统计时间段内，人工座席处理电话的总时长与实际登录系统时长的比率。
- ❑ **服务水平**：服务水平是对于呼入项目呼叫中心的一个技术指标，是指在某一统计时间段内应答呼叫数量占呼叫中心接入呼叫数量的百分比。
- ❑ **客户满意度**：客户满意度一般是指接受电话服务的客户对于呼叫中心所提供服务的满意程度。一般通过定期对客户进行满意度调查，或在每一次电话服务结束后系统自动对客户满意度进行调查获得数据。
- ❑ **平均振铃次数**：平均振铃次数是指呼入业务类型的呼叫中心在某一统计时间段内，客户听到 IVR 或人工座席接起电话之前所听到的振铃次数之和与总呼叫次数的比值。
- ❑ **监听合格率**：监听合格率是指在某一统计时间段内，质检人员通过监控、电话录音等手段抽检座席的服务质量的合格率。
- ❑ **一次性解决问题率**：一次性解决问题率是在某一统计时间段内，不需要客户再次拨打呼叫中心电话也不需要座席员回拨或转接电话就可以解决客户所提出问题的电话量所占座席员接起电话量总数的比率。

2.5.5 CTI 中间件

在 2.5.2 节我们讲到，CTI 中间件是第三代呼叫中心的重要标志。那么，什么是 CTI 中间件呢？ CTI（Computer Telephony Integration）的字面意思为：计算机电话集成。早先的电话交换机（即 PBX）是一个独立、封闭的系统设备。随着计算机技术的发展和呼叫中心需求的提出，交换机设备厂商开始考虑为交换机增加一个可以受计算机系统控制的接口，由计算机系统通过某种协议获取交换机用户话机的状态信息以及对呼叫的控制命令。这种连接和控制接口称为 CTI-Link。

交换机的厂商繁多，不同交换机厂商所提供的 CTI-Link 的接口协议不尽相同，常见的有北电网络的 Merisian Link、Avaya 的 ASAI/TSAPI 以及欧洲交换机厂商所普遍遵循的 ECMA（欧洲计算机制造协会）提出的 CSTA 标准。因此，要针对每一个交换机厂商的 CTI-Link 协议进行开发是一件很繁杂的工作，因此人们提出了 CTI 中间件的概念。CTI 中间件在下层通过对各种 CTI-Link 协议的包装和抽象，屏蔽了各种交换机的不同，在上层为呼叫中心业务软件开发人员提供统一的 API 开发接口，这样开发人员开发的程序不仅能支持丰富的业务逻辑，还能适用于各种不同的交换机，增加了系统的灵活性，也提高了开发效率。同时 CTI 作为一种中间件产品，也一直长盛不衰。图 2-6 是 CTI 中间件在呼叫中心系统中的位置。

早期比较著名的 CTI 中间件有 CT-CONNECT、Genesys、Quintus 等。随着呼叫中心技术的发展，CTI 中间件的功能也在不断丰富。CTI 中间件由最初单纯的 CTI-Link 协议转换

和标准化的单一功能发展成为涵盖多渠道统一接入、智能 ACD 呼叫分配、智能预测外呼、现场监控工具、数据统计分析等多个功能模块的系统平台。

在互联网尤其是移动互联网技术飞速发展的今天，人们对电话与互联网的集成又提出了新的要求，因而各厂商也增加了一些面向互联网的接口。但是，传统的 CTI 技术由于其产生的年代及其本身的基础架构的局限性，一般来讲是不适合互联网的。而且，随着使用 VoIP 软交换设备替代传统的硬件交换机，以及运营商网络由传统的 PSTN 向 IMS 及 LTE 的发展，理论上讲，原来的以电路交换为基础的电话系统现在也都变成了计算机了，因而计算机 – 电话系统集成也就失去了它原本的意义。现代的 VoIP 电话系统直接可以提供更现代、更开放的集成接口，SIP、3GPP、Webservice、REST、大规模并发和集群等新型的协议和开发部署方式由于其更易于与互联网集成而受开发者青睐，因而，CTI 领域也必将迎来一场新的革命。

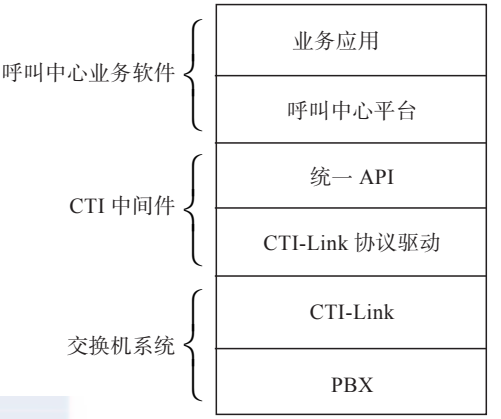

图 2-6　CTI 在呼叫中心系统中的位置

2.5.6　FreeSWITCH 在呼叫中心的应用

我们可以使用 FreeSWITCH 项目构建一个软交换类型的呼叫中心系统。本小节我们来看一下 FreeSWITCH 中和呼叫中心相关的几个主要功能。

（1）语音交换功能

FreeSWITCH 首先是一个 IP-PBX，具备类似交换机的语音交换功能。通过路由设置或程序控制，通过 FreeSWITCH 可以将客户拨打进的呼叫分配到指定的座席代表所在的话机终端。同样，座席代表也可使用话机终端通过 FreeSWITCH 发起呼叫，拨打客户的固定电话或移动电话。语音交换功能是构成呼叫中心接入平台最基本的功能。

（2）媒体处理功能

FreeSWITCH 具备媒体处理功能，可以进行录音、放音、DTMF 按键分析、产生 DTMF 按键的操作。利用 FreeSWITCH 的媒体处理功能可以实现呼叫中心系统中的 IVR 系统和录音系统的功能。

（3）媒体监播功能

FreeSWITCH 具备媒体监播功能。利用 FreeSWITCH 的媒体监播功能，可以实现呼叫中心所需的监听、强插、耳语等功能。

（4）电话会议功能

在某类呼叫中心业务中（如：电话外语翻译），需要多方参与通话时可以使用 FreeSWITCH 所提供的电话会议功能实现。

（5）电子传真功能

FreeSWITCH 支持收、发电子传真的功能，可以完全替代传统基于传真卡的语音板卡传真系统。

（6）排队功能

FreeSWITCH 提供 mod_fifo（先进先出模块）以及 mod_callcenter 模块用于实现呼叫中心的排队和 ACD 功能。当然，开发者也可以利用 FreeSWITCH 所提供的接口自己完成排队策略的控制模块。

通过以上关于对 FreeSWITCH 与呼叫中心相关功能的介绍我们可以发现，使用 FreeSWITCH 开发出一套功能完善的呼叫中心系统是完全没问题的。同时，FreeSWITCH 良好的性能和稳定性也为呼叫中心系统提供了可靠的保障。对于大规模呼叫中心的应用可以提供 HA（双机热备方案）和 Cluster（集群）方案来解决对于高可用性和大容量呼叫中心的需求。

2.6 小结

本章着重介绍了传统的 PSTN 网络和 PBX 系统所能实现的基本业务和增值业务。其中有一些业务是比较小众的，可能大部分人一生也不会用到；而很大一部分业务是在我们的工作和生活中常常用到的，只是可能没怎么在意。当然，对于这个领域的从业者来讲，这些都应该是非常熟悉的。但无论如何，在学习 FreeSWITCH 的过程中会涉及这些业务的方方面面。在本章，既有简单的罗列，也有生动的故事，目的就是带领对业务不熟悉的读者循序渐进地了解这些业务，对熟悉业务的读者也统一一下思想。读者在学习中不妨也深入思考一下：这些业务为什么会出现？它们给人们带来了怎样的便利？各种业务的出现都是为了解决谁的问题，是主叫用户、被叫用户，还是运营商？

本章也简单介绍了从 PBX 到 IP-PBX 的演变。值得一提的是，IP-PBX 能提供更丰富、灵活的功能，但有些人可能更喜欢传统的 PBX 简单。而且 IP-PBX 为兼容旧的设备和用户使用习惯往往采取一些妥协的办法，因而不一定能充分发挥其优势。旧技术到新技术的演变总要有一个过程，用户从认识、接受、慢慢习惯到熟练应用需要一个过程，各厂商的推动、研发以及利益方面的考虑也需要时间。新技术不是解决所有问题的良药，总有人会怀旧，但历史的车轮永远是向前的，新技术取代旧技术肯定是一个不可逆转的趋势。

最后，本章也用了大量的篇幅讲了呼叫中心。并不是因为呼叫中心这一概念有多重要，而是从它的历史和发展中可以看出交换机技术的发展以及进步，业务层五花八门的需求也将交换机的性能和功能都发挥到了极致。在呼叫中心部分我们也着重讲了 CTI 的概念，以及其将面临的挑战和革命。

总之，通过这两章的学习，我们已经了解了一些基础知识，端正了思想，统一了认识。接下来，便可以大步进入我们的 FreeSWITCH 之旅了。

第 3 章

初识 FreeSWITCH

在前面几章，我们用了很大的篇幅介绍了电话通信的背景和基础知识，以及电信业务的知识。对于刚刚跨入通信（或电信）领域的读者来说，熟悉这些背景知识以及里面提到的各种名词术语，有助于理解后面要学到的知识。通信领域涉及的面非常广泛，可以说，里面的很多术语或知识点单独拿出来都可以写成一章或一本书。我们本书的重点是 FreeSWITCH，因此从本章开始，我们正式进入 FreeSWITCH 的学习。学习本书的好处在于，即使你对前两章的内容不是很了解，也可以通过对 FreeSWITCH 的学习去反过来理解前面的知识。

在本章，我们将先讲解 FreeSWITCH 的基本概念，然后通过安装和简单的配置做成一个实际可用的 PBX 系统，并进行电话注册和拨打测试。目的是先给读者一个宏观、快速的体验，然后再在后面的章节中逐步强化和深入。读者可以循序渐进，一步一步成长为 FreeSWITCH 领域的高手。

3.1 什么是 FreeSWITCH？

什么是 FreeSWITCH？这一问题恐怕是初次见到本书的人首先要问的。很遗憾，我们一直到本章才回答这个问题。其实好多人问这一问题，并不是期望我们能给 FreeSWITCH 来下一个准确的定义，而是想知道，它到底能做什么。下面我们就来看一下 FreeSWITCH 的概念和功能。

3.1.1 FreeSWITCH 的概念

FreeSWITCH 是一个开源的电话交换平台。官方给它的定义是——世界上第一个跨平台的、伸缩性极好的、免费的、多协议的电话软交换平台[⊖]。由这个定义我们可以得出以下几点：

❑ FreeSWITCH 是**跨平台**的。它能原生地运行于 Windows、Max OS X、Linux、BSD 及

⊖ The World's First Cross-Platform Scalable FREE Multi-Protocol Soft Switch - http://www.freeswitch.org。

Solaris 等诸多 32/64 位平台（甚至，也有人成功地将它应用于 Linksys NLS2 平台及 Raspberry Pi 上[⊖]）。

- ❑ FreeSWITCH 具有很强的**可伸缩性**。FreeSWITCH 从一个简单的软电话客户端到运营商用级的软交换设备几乎无所不能。
- ❑ FreeSWITCH 是免费的。它采用 MPL 1.1[⊜]协议授权，意味着任何人都可以免费使用并获取源代码，任何人都可以修改、发布甚至出售自己的应用。
- ❑ FreeSWITCH 支持 SIP、H323、Skype、Google Talk 等多种通信协议，并能很容易地与各种开源的 PBX 系统（如 sipXecs、Call Weaver、Bayonne、YATE 及 Asterisk 等）通信，它也可以与商用的交换系统（如华为、中兴的交换机或思科、Avaya 的交换机等）互通，如图 3-1 所示。

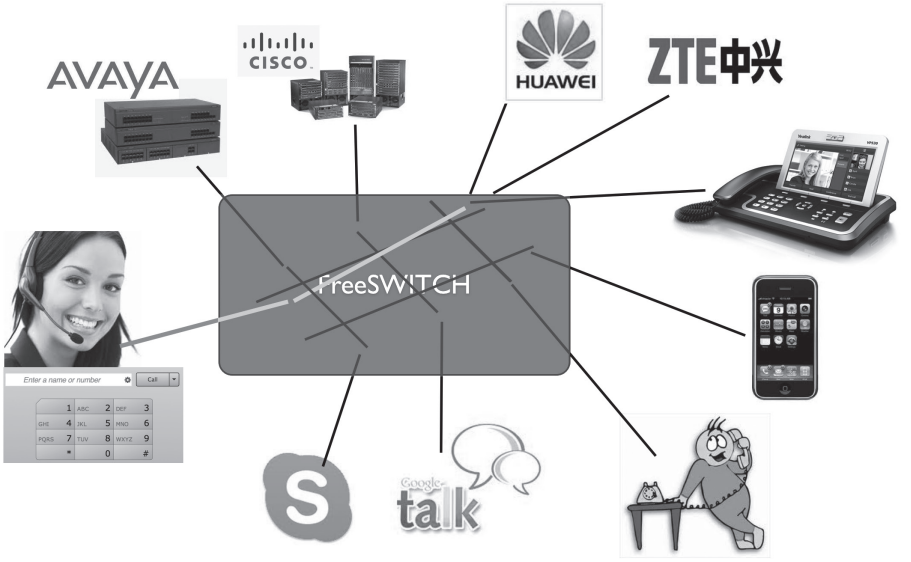

图 3-1 FreeSWITCH 支持多种协议

- ❑ FreeSWITCH 可以用作一个简单的交换引擎、一个 PBX、一个媒体网关或媒体支持 IVR 的服务器，或在运营商的 IMS 网络中担当 CSCF 或 Application Server 等。
- ❑ FreeSWITCH 遵循相关 RFC 并支持很多高级的 SIP 特性，如 Presence、BLF、SLA 以及 TCP、TLS 和 sRTP 等。它也可以用作一个 SBC 进行透明的 SIP 代理（proxy）以支持其他媒体，如 T.38 等。
- ❑ FreeSWITCH 支持宽带及窄带语音编码，电话会议桥可同时支持 8、12、16、24、32

⊖ Raspberry Pi(http://www.raspberrypi.org/) 是一个装有 ARM CPU 的信用卡大小的计算机。事实上，笔者办公室的 IP-PBX 就是一台运行着 FreeSWITCH 的 Raspberry Pi。

⊜ Mozilla Public License。详见：http://www.mozilla.org/MPL/1.1/。

及 48kHz 的语音。

❑ 从技术上讲，FreeSWITCH 是一个 B2BUA ⊖，它作为一个背靠背的用户代理用来帮助通信的双方进行实时的语音视频通信，如图 3-2 所示。

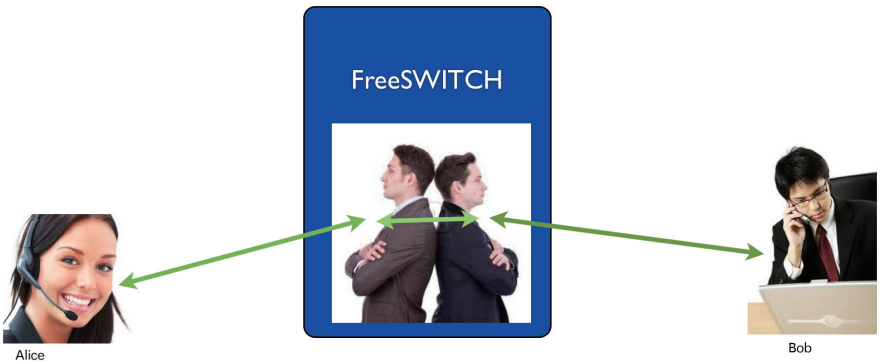

图 3-2　FreeSWITCH 是一个 B2BUA

3.1.2　FreeSWITCH 的功能

FreeSWITCH 是一个 B2BUA，所以它能做的工作非常多。在国外，很多 ISP 和运营商把它作为关键的软交换设备，处理成千上万路的并发通话；也有的把它用于呼叫中心，与各种企业级的应用系统（如 CRM、ERP 等）集成；在国内，也已经有很多应用案例，其被广泛用于金融、保险、电力、石油、煤炭等领域的呼叫中心、企业通信以及应急指挥调度平台等。从这一方面讲，它是传统的电话交换系统及商业的电话交换系统良好的替代品。除了简单的替代以外，它往往还提供更多的新功能、更灵活的数据集成能力和更快速的应用开发能力，在业务需求千变万化的今天显得格外有生命力。

另外，在当今的移动互联、物联网与大数据、云计算盛行的时代，好多厂商和互联网的创业者也把 FreeSWITCH 用于通信领域的“云”平台。FreeSWITCH 诞生的年代和背景、良好的设计架构以及活跃的技术支持社区都是它能在“云”平台上成功的坚实基础。

上面讲了 FreeSWITCH 的一些典型应用场景，下面看看它的典型功能：

❑ 在线计费、预付费功能
❑ 电话路由服务器
❑ 语音转码服务器
❑ 支持资源优先权和 QoS 的服务器
❑ 多点会议服务器
❑ IVR、语音通知服务器

⊖ Back-to-back User Agent，背靠背的用户代理。事实上，B2BUA 的概念会贯穿本书的始终，读者最好能理解它。

- ❑ VoiceMail 服务器
- ❑ PBX 应用和软交换
- ❑ 应用层网关
- ❑ 防火墙 /NAT 穿越应用
- ❑ 私有服务器
- ❑ 第三方呼叫控制应用
- ❑ 业务生成环境运行时引擎
- ❑ 会话边界控制器
- ❑ IMS 中的 S-CSCF/P-CSCF/I-CSCF
- ❑ SIP 网间互联网关
- ❑ SBC 及安全网关
- ❑ 传真服务器、T.30 到 T.38 网关

更多关于 FreeSWITCH 的特点和指标，可以参考 http://wiki.freeswitch.org/wiki/Specsheet。

3.2 快速体验

FreeSWITCH 的功能确实非常丰富和强大，在进一步学习之前我们先来一次完整的体验。

FreeSWITCH 默认的配置是一个 SOHO PBX（家用电话小交换机），那么我们本节的目标就是从零开始安装，实现分机互拨电话，测试各种功能，并通过添加一个 SIP-PSTN 网关拨打 PSTN 电话。这样，即使你没有任何使用经验，也应该能顺利学完本章，从而建立一个直观的认识。在体验过程中，你会遇到一点稍复杂的配置，如果不能完全理解，也不用担心，我们在后面会详细介绍。当然，如果你是一个很有经验的 FreeSWITCH 用户，那么大可跳过本章。

3.2.1 安装基本 FreeSWITCH 系统

在学习和使用 FreeSWITCH 之前，我们首先要安装一个基本的 FreeSWITCH 系统。FreeSWITCH 是跨平台的，大多数人使用各种 Linux 系统；很大一部分的开发者使用 Mac 平台进行开发；另外，也有很多用户在 Windows 平台上学习和使用它。因此，我们将分别介绍一下这几大主流平台的安装方法和应该注意的问题。FreeSWITCH 的开发非常活跃，因面版本更新很快，所以，我们首先从选择一个安装版本开始。

1. 版本简介

到本书截稿时止，FreeSWITCH 最新的版本是 1.4.beta。

FreeSWITCH 的版本号很有规律：版本号有 3 部分构成，以点隔开。其中，第 1 位为主版本号，第 2 位为次版本号，第 3 位用作补丁及更新的标志。其中，从第 2 位看，偶数的版本为稳定版，奇数的版本为开发版。开发版更新的内容在经过测试后会合并到稳定版中。如果有大的功能变化或改进，则稳定版和开发版版本两者的编号都会加 2。例如，上一个稳

定版本为 1.2，其对应的开发版为 1.3。最初的 1.2 由 1.2-rc1（Release Candidate，候选版）、1.2-rc2、到 1.2.0、1.2.1 等组成，到本书截稿时为止，最新的一个稳定版本是 1.2.22。

FreeSWITCH 使用 Git 进行版本控制。1.2 版本单独由一个 1.2.stable 的分支进行管理。其中，每一个发行版都会对应 Git 里的一个 Tag，如 v1.2.10、v1.2.12 等。而 1.2.stable 分支则永远是 1.2 版中最新的版本（可以看成是稳定分支中的不稳定版）。

FreeSWITCH 支持 32 位及 64 位的 Linux、Mac OS X、BSD、Solaris、Windows 等众多平台。某些平台上有编译好的安装包，但作者建议有一定基础的用户从源代码安装，因为这样便于版本的切换与升级。

在实际安装过程中，我们尽量选用比较新的版本。然而，某些版本在某些平台上有一些已知的问题，因此，具体的版本选择我们将在安装时再介绍。

2. 在 Windows 上安装

如果仅仅是为了学习和使用，在 Windows 平台上可以使用已经编译了的安装包。另外，为了完整性，本章也包含从源代码编译安装的步骤。本节假设读者已经熟悉 Windows 平台上的软件安装方法，在实际安装过程中仅对应该注意的事项加以说明。

（1）使用安装包安装

Windows 用户可以直接下载安装文件，下载地址为 http://files.freeswitch.org/windows/installer/。然后根据自己的系统选择不同目录，32 位系统的用户选择 x86 目录，64 位系统的用户选择 x64 目录。freeswitch.msi 是最新的安装程序，一般隔几天就会更新一次版本。笔者的测试环境是 32 位的 Windows XP，下载界面如图 3-3 所示。

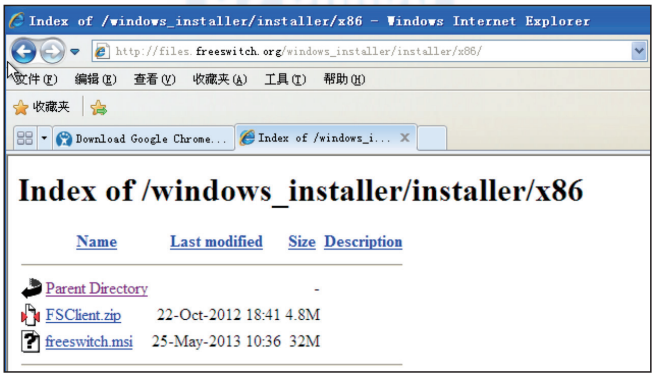

图 3-3　下载 FreeSWITCH Windows 版

如同安装其他程序一样，我们全部选择默认设置即可，也就是说只要连续单击“Next”按钮就能安装完毕。安装完成后选择”开始菜单”→“所有程序”→“FreeSWITCH”→“FreeSWITCH”便可以启动 FreeSWITCH 了，启动后的界面如图 3-4 所示。

如果安装过程中你没有修改默认安装路径的话，那么 FreeSWITCH 的实际安装路径是：c:\Program Files\FreeSWITCH，配置文件在该目录的 conf 目录下。

图 3-4 Windows 上的 FreeSWITCH 控制台界面

（2）从源代码安装

如果从源代码安装，则首先要下载源代码。在此我们以 1.2.10 版为例，其下载地址是：http://files.freeswitch.org/freswitch-1.2.10.tar.gz。

除此之外，也可以 Git 仓库获取源代码。Git 是 FreeSWITCH 使用的版本控制工具，从 Git 仓库获取源代码的好处是可以随时更新，并可以很方便地切换到不同的代码分支，甚至“倒回”到任意提交点。

如果从 Git 仓库获取源代码，需要先在 Windows 上安装 Git。使用哪个 Git 版本不是很重要，笔者用的是从 https://code.google.com/p/msysgit/downloads/list?q=full+installer+official+git 下载的 1.8.3-preview 版。

安装 Git 很简单，一般来说双击安装文件并连续单击“Next”按钮即可安装完毕。不过，在 Windows 平台编译 FreeSWITCH 有几个要注意的事情，因此在安装 Git 的过程中我们也需要注意以下问题，并做适当的选择：

- 将 FreeSWITCH 的源代码放到一个“干净”的目录下。为避免有时候遇到奇怪的问题，最好把代码放到一个比较不容易出问题的目录下，如可以放到 C:\src\freeswitch 或 D:\src\freeswitch 下，这两个都是比较好的目录。而像 C:\My Documents（有空格）或 C:\ 源代码中文目录 \freeswitch（有中文）之类的则在编译或使用时可能会有问题。
- Git 相关的环境变量。Git 是从 UNIX 系统上移植过来的一个命令行工具，因此需要一些相关的环境变量。在安装时有三个选项（见图 3-5），笔者建议使用第三项，这样最省心。当然，第三项与 Windows 系统的命令会有少量冲突，如 find 等。但实际上，你可能永远不会用到 Windows 上的命令行工具，因此，在安装过程中果断选择第三项可以省去不少麻烦。

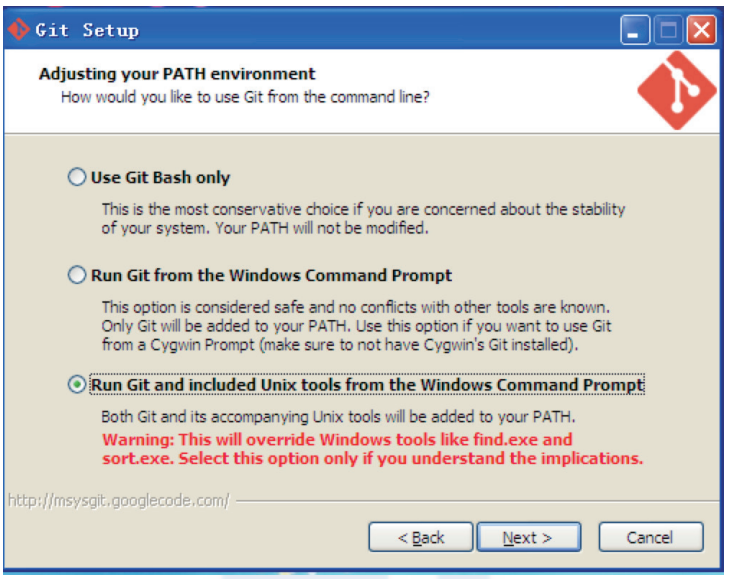

图 3-5　安装 Git 时选择自动包含所有相关的环境变量

❑ 关闭 Git 的自动换行符转换。众所周知，Widows 使用“回车 + 换行”（“\r\n”，又称作“CRLF”）做换行符，而 UNIX 仅使用“\n”。Git 可以自动在不同的换行符间转换。但问题是，有时候自动转换不靠谱，尤其是对于 FreeSWITCH 这样大型的项目，所以笔者一般在安装 Git 时就关掉这一选项（否则在编译阶段可能会出奇怪的错误），如图 3-6 所示。

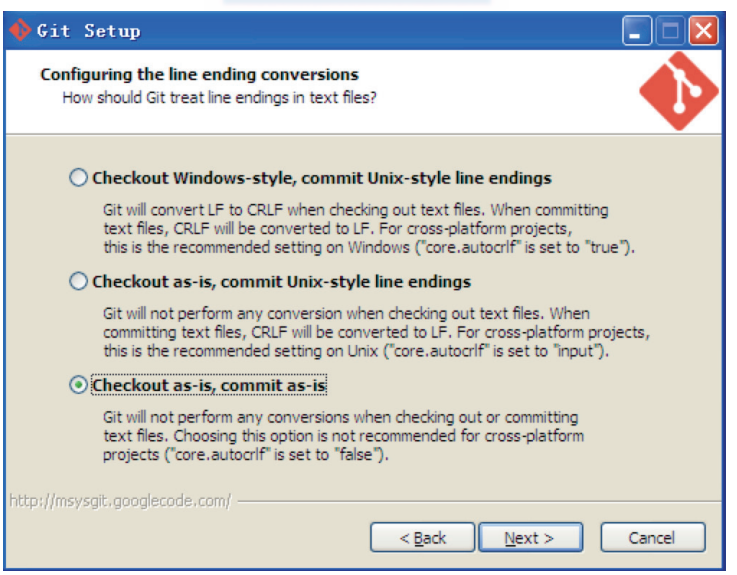

图 3-6　不使用自动换行符转换功能

接下来可以连续按“Next”按钮直到安装完毕。Git 安装完毕后就可以切换到命令行方式，使用 git clone 命令把远程的版本仓库复制到本地了：

```
git clone git@git.freeswitch.org/freeswitch.git
```

复制完毕后，默认的分支是 master 分支，即最新的分支。FreeSWITCH 对不同版本的安装包在 Git 仓库中有不同标签与之相对应。使用如下命令可以列出所有的标签（tag，为节省篇幅，省略了一部分输出）：

```
C:\src\freeswitch> git tag
v1.2.0
v1.2.1
v1.2.10
v1.2.21
v1.2.22
v1.2.9
v1.5.7
```

可以用以下命令检出对应的标签并建立一个新的本地分支，（我们在这里仍然使用 1.2.10 版）：

```
C:\src\freeswitch> git checkout -b v1.2.10
Switched to a new branch 'v1.2.10'
```

当然，如果你不习惯使用这种命令和工具，则可以下载 Tortoise Git 图形界面工具，下载地址为 https://code.google.com/p/tortoisegit/wiki/Download。

Tortoise Git 也允许通过 AutoCrlf 复选框选择是否开启自动换行符转换，为避免它自动转换，我们应该保证该复选框是非选中状态的，如图 3-7 所示。

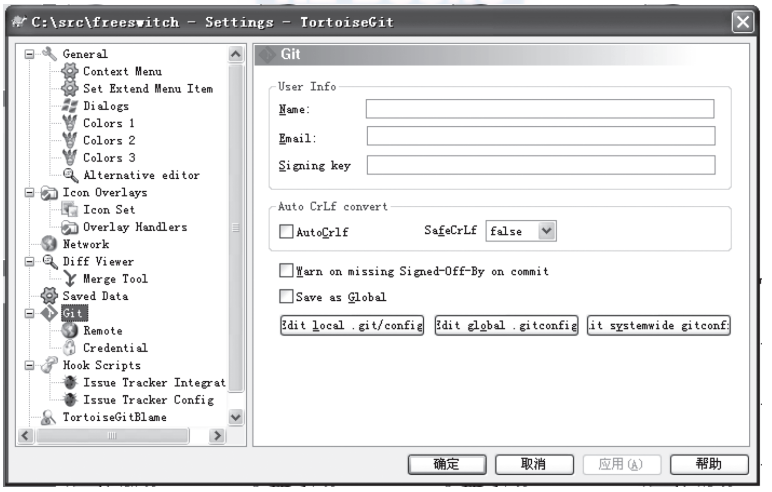

图 3-7　Tortoise Git 的 CRLF 设置

使用图形界面的方式对 FreeSWITCH 的源代码进行复制会比命令行方式直观一些，如图 3-8 所示。

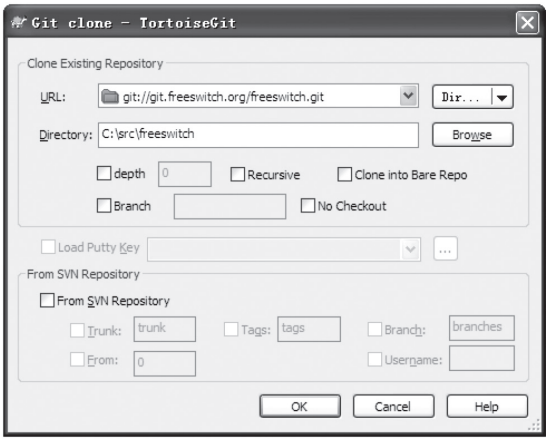

图 3-8　使用 Tortoise Git 复制 FreeSWITCH 代码库

复制完毕后，可以使用右键菜单，通过选择相应的菜单项检出（checkout）相应的标签或分支，在这里就不多介绍了。

有了 FreeSWITCH 源代码，接下来还需要下载编译工具。Microsoft 提供 Visual Studio 工具进行开发。FreeSWITCH 中有 VS2005、VS2008、VS2010 以及 VS2012 的工程文件。VS2008 及以前的支持已经不再更新了，因此不推荐使用。VS2010 和 VS2012 目前是官方支持的版本。在此，笔者使用 VS2010 Express 版为例加以说明。

FreeSWITCH 的源代码目录下有一个名为 Freeswitch.express.2010.sln 的 Solution 文件，双击鼠标打开它，然后选择菜单项“调试”→“生成解决方案”，或按快捷键 F7，就可以进行编译了。不出问题的话，编译成功后将会在源代码目录下的 Win32 目录下出现 Debug 或 Release 目录（取决于编译前的选择，默认为 Debug），编译完成的目标文件都会在这些目录下。

图 3-9 所示是使用 VS2010 正在编译 FreeSWITCH 源代码时的界面。

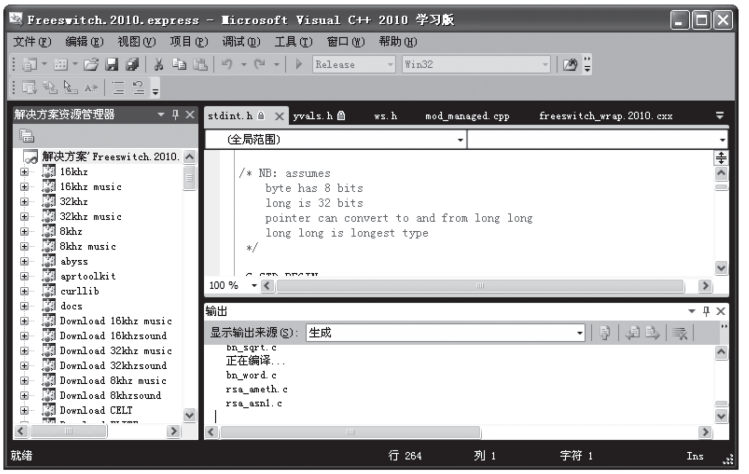

图 3-9　在 Windows 上使用 VS2010 编译

3. 在 Linux 系统上安装

在开始本小节的讲解之前有一点需要和读者声明一下，就是以下内容是在假定读者已经有了一定的 Linux 的基本知识并且已经安装了 Linux 的情况下进行的。若读者没接触过 Linux，则建议不采用这种方法，或者去网上搜集相关资料，自行学习 Linux 相关知识。限于篇幅，本书不再介绍与 Linux 相关的基础知识。

在安装之前，我们需要先准备安装环境。Linux 有多种发行版（发行套件）。一般来说，大部分主流的 Linux 发行版都是可以运行 FreeSWITCH 的，但不排除某些发行版的内核、文件系统、编译环境、LibC 版本会有一些问题。所以，如果你在安装或使用 FreeSWITCH 的过程中遇到问题时想获得社区支持，最好选择一种大家都熟悉的发行套件[⊖]。另外，编译安装 FreeSWITCH 要依赖一些基础的 Linux 软件包，在不同的发行版平台上可以用以下不同的命令安装：

CentOS：

```
    yum install -y autoconf automake libtool gcc-c++ ncurses-devel make zlib-devel
libjpeg-devel
    yum install -y openssl-devel e2fsprogs-devel curl-devel pcre-devel speex-devel
sqlite-devel
```

Ubuntu/Debian:

```
    apt-get -y install build-essential automake autoconf git-core wget libtool
    apt-get -y install libncurses5-dev libtiff-dev libjpeg-dev zlib1g-dev libssl-
dev libsqlite3-dev
    apt-get -y install libpcre3-dev libspeexdsp-dev libspeex-dev libcurl4-openssl-
dev libopus-dev
```

除此之外，如果你想从 Git 仓库中下载源代码安装 FreeSWITCH，则需要事先安装 Git。CentOS 5 默认的软件仓库中可能没有 Git，如果你需要在 CentOS 5 上使用 Git 安装，则可以先安装 rpmforge（http://pkgs.repoforge.org/rpmforge-release/），然后再安装 Git。CentOS 6 的 yum 源中已经包含了 Git，因此不需要 rpmforge 了。关于如何在你的发行版上安装 Git 请参考有关资料。一般来说，在 Ubuntu 或 Debian 上可以使用如下命令来安装：

```
apt-get install git-core
```

在 CentOS 6 上则可以使用如下命令：

```
yum install git
```

在准备好相关 Linux 环境以后，就可以安装 FreeSWITCH 了。以下的安装步骤跟选用哪种 Linux 发生套件关系不大。从以下三种安装方式可任选其一，默认安装位置都是 /usr/

⊖ FreeSWITCH 开发者使用的平台是 Debian 7，社区中也有许多人在使用 CentOS（5、6）和 Ubuntu。在将要发布的 FreeSWITCH1.4 版中，官方已决定不再支持 Debian6 及 CentOS5 等低版本的 Linux 系统，但 FreeSWITCH1.2 版仍支持，总之选择操作系统的原则是，如果你对 FreeSWITCH 和 Linux 都不熟悉的话，建议选择最新版本的 Debian 或 CentOS，这样遇到问题可以比较方便地得到帮助（开发者并不使用 Ubuntu）。如果你选择一个生辟的或比较小众的发行版，若遇到问题则很难找到能帮助你的人。当然，如果你是往新的系统移植，建议至少熟悉了 FreeSWITCH 以后再做。

local/freeswitch。安装过程中会下载源代码目录，请保留，以便以后升级及安装配置其他组件。

（1）从 Git 仓库安装

从代码库安装能让你永远使用最新的版本，如果安装过程中遇到问题也能够方便地回退到先前的版本。首先我们使用下列命令来从 Git 仓库中获取 FreeSWITCH 的源代码：

```
git clone git://git.freeswitch.org/freeswitch.git
```

如果需要安装特定的版本，则可以切换到对应的 Tag。如安装 1.2.22，你可以执行：

```
cd freeswitch                    # 进入源代码目录
git checkout -b v1.2.12          # 根据一个 Tag 检出到一个本地分支
```

或

```
git checkout -b v1.4.beta        # 从远程分支检出一个本地分支
```

当然，如果对 Git 比较熟悉，你也可以直接在复制时指定一个分支：

```
git clone -b v1.4.beta git://git.freeswitch.org/freeswitch.git
```

总之，在 Linux 上得到源代码并检出适当的 Tag 或分支（新手推荐选择安装时最新的稳定版）后，便可以执行下列命令进行安装（注意下列命令要在 FreeSWITCH 源代码目录中执行）：

```
./bootstrap.sh
./configure
make install
```

上面的命令是在 Linux 上从源代码安装软件的标准过程。首先第 1 行执行 bootstrap.sh 以初始化一些编译环境，第 2 行配置编译环境，第 3 执行编译安装。

（2）解压缩源码包安装

注意，这里我们使用本书截稿时最新的 1.4.beta6 版，如果你安装的时候，应该检查一下是否有更新的版本出现。

使用 wget 可以获取源代码安装包。下列命令会首先使用 wget 下载安装包，然后使用 tar 解压缩，最后使用 cd 命令进入源代码目录：

```
wget http://files.freeswitch.org/freeswitch-1.4.0.beta6.tar.bz2
tar xvjf freeswitch-1.4.0.beta6.tar.bz2
cd freeswitch-1.4.0
```

接下来的配置安装就很简单了，具体如下：

```
./configure
make install
```

可以看到，与上一种方法不同的是，它不需要执行 bootstrap.sh（源代码在打成 tar 包前已经执行过了，因而不需要 automake 和 autoconf 工具），便可以直接配置安装。

（3）最快安装

这是史上最快的安装方式，如果你对 UNIX 类的编译系统比较熟，或者跟作者一样需要

经常安装系统，你不妨试一试这种方式㊀：

```
wget http://www.freeswitch.org.cn/Makefile && make install
```

以上命令会使用 wget 下载一个 Makefile，然后使用 make 执行安装过程。安装过程中它会从 Git 仓库中获取代码㊁，实际上执行的操作跟前面的安装方式相同。

4. 在 Mac 系统上安装

苹果公司的 Mac 系统是理想的开发者平台，尤其是苹果 iPhone 和 iPad 在全世界范围内的成功，使得该平台吸引了大量的开发者。而且大多数的 FreeSWITCH 开发者也都在使用 Mac。事实上，本书就是在 Mac 系统上使用 Sublime Text 2 编辑器写成的，本书的大部分环境和截图也是在 Mac 系统上做的。

如果你想在 Mac ㊂系统上安装 FreeSWITCH，则需要先下载安装 Apple 的 Xcode 工具（https://developer.apple.com/xcode/），并选择菜单 Preferences -> Downloads 安装命令行工具（Command Line Tools）㊃，如图 3-10 所示。

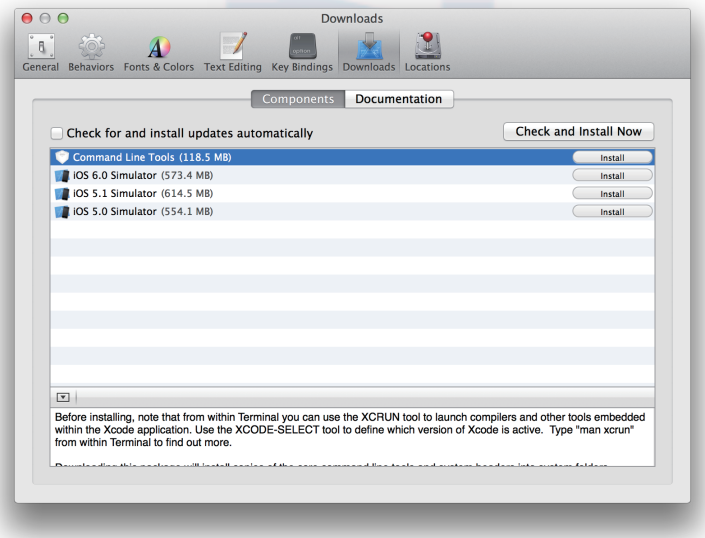

图 3-10　安装 Apple Xcode 及 Command Line Tools

除此之外，FreeSWITCH 也依赖于一些第三方的库。要安装第三方的库，在 Mac 平台

㊀ 在本书截稿时，FreeSWITCH 1.4 还没有发布正式版，因而你拿到的最新的 FreeSWITCH 版本在实际安装时可能跟上面描述的安装过程有出入。如果按照上面几种安装方式遇到问题的话，尝试读一下这里提到的 Makefile 的文件的内容也许会有帮助。

㊁ 根据发行版的不同它还可以自动选择使用 apt-get（Debian/Ubuntu）、yum（CentOs）或 Homebrew（MacOSX）来安装依赖的包和库。对此感兴趣的读者可以自行看一下下载得到的 Makefile 文件。

㊂ 本例是在 Mac OS X 10.8.4 上做的，Apple 公司在 2013 年 10 月份发布了 Mac OS X 10.9，但笔者尚没有升级。

㊃ 本例是在 Mac10.84 上写的，如果在 Mac10.9 上，请使用 xcode-select-install 命令安装。

上一般使用 Macports、Flink 和 Homebrew 等包管理工具。Homebrew 是比较新的工具，安装和使用起来都很方便。如果你还没有安装，可以用以下命令安装[㊀]：

```
ruby -e "$(curl -fsSL https://raw.github.com/mxcl/homebrew/go)"
```

安装完 Homebrew 后，可以先试一下安装 Git 和 libtiff 库，安装 FreeSWITCH 时也需要用到它们：

```
brew install git
brew install libtiff
```

在你的系统上使用以上命令时，若系统提示你没有权限，则可以在命令前面加上 sudo，如安装 libtiff 库的命令就可写为：

```
sudo brew install libtiff
```

其他的安装步骤就全部跟 Linux 上一样了，如：

```
git clone git://git.freeswitch.git/freeswitch.git
cd freeswitch
./bootstrap.sh
./configure
make install                    # 或 sudo make install（根据是否有权限）
```

5. 安装声音文件

在后面的例子中我们需要一些声音文件。声音文件有两种，一种是提示音，用于通话期间的语音提示，如 VoiceMail 的提示音，支持 TTS 功能的提示音等；另一种是音乐，用于在 Hold 状态时播放，即所谓的 Music on Hold（MOH）[㊁]。

在 Windows 系统上，这些声音文件是默认安装的。而在 Linux 或 Mac 上安装这些声音文件也异常简单。你只需在源代码目录中执行：

```
make sounds-install
make moh-install
```

安装过程中将自动从 files.freeswitch.org 下载相关的语音包，并解压缩到相关的安装路径中（默认安装在 /usr/local/freeswitch/sounds 下）。

另外，FreeSWITCH 支持 8kHz、16kHz、32kHz 及 48kHz 的语音[㊂]。与上面的声音文件相对应的高清声音文件可以选择安装。如以下命令安装 16kHz 的声音文件：

```
make cd-sounds-install
make cd-moh-install
```

㊀ 由于 Homebrew 的安装地址可能变化，请到官方网站（http://mxcl.github.io/homebrew/index_zh-cn.html）查看最新的安装方法。

㊁ 在实际应用中，如果通话的双方中有一方不想让对方听到自己的讲话，或者在不挂机的情况下拨打另一路电话时，就需要将电话置于 Hold（保持）状态，这时候 FreeSWITCH 需要向对方播放保持音乐，即 MOH。另外，在拨打某些呼叫中心客户号码的时候经常遇到“座席繁忙，请等待…”，然后是一连串的音乐，这些音乐也可以称为 MOH。

㊂ 很少有其他电话系统支持如此多的抽样频率。普通电话是 8kHz，某些新的 IP 话机支持更高的抽样频率，更高频率意味着更好的语音质量（细节听起来更逼真）。

6. 安装完成后的操作

FreeSWITCH 使用 make install 安装完成后，会显示一个有用的帮助，它会提示你接下来可以用哪些 make 命令执行一些其他的操作（如我们刚才安装声音文件的命令，在这里就可以看到）。下面笔者在默认的帮助信息后增加了一些中文的注释，读者可以在学习中自行练习一下。

```
+---------- FreeSWITCH install Complete ----------+
+ FreeSWITCH has been successfully installed.     +
+                                                 +
+       Install sounds:                           +  安装声音文件
+       (uhd-sounds includes hd-sounds, sounds)   +
+       (hd-sounds includes sounds)               +
+       ----------------------------------        +
+                make cd-sounds-install           +  CD 音质的声音文件
+                make cd-moh-install              +
+                                                 +
+                make uhd-sounds-install          +  超高清声音文件
+                make uhd-moh-install             +
+                                                 +
+                make hd-sounds-install           +  高清声音文件
+                make hd-moh-install              +
+                                                 +
+                make sounds-install              +  标准声音文件
+                make moh-install                 +
+                                                 +
+       Install non english sounds:               +  安装其他语言的声音文件
+       replace XX with language                  +
+       (ru : Russian)                            +  如 ru 代表俄语
+       ----------------------------------        +
+                make cd-sounds-XX-install        +
+                make uhd-sounds-XX-install       +
+                make hd-sounds-XX-install        +
+                make sounds-XX-install           +
+                                                 +
+       Upgrade to latest:                        +  升级到最新版本
+       ----------------------------------        +
+                make current                     +
+                                                 +
+       Rebuild all:                              +  重新编译
+       ----------------------------------        +
+                make sure                        +
+                                                 +
+       Install/Re-install default config:        +  安装（或重新安装）配置文件
+       ----------------------------------        +
+                make samples                     +
+                                                 +
+       Additional resources:                     +
+       ----------------------------------        +
+       http://www.freeswitch.org                 +  官方网站
+       http://wiki.freeswitch.org                +  官方 Wiki
+       http://jira.freeswitch.org                +  官方的缺陷跟踪工具
+       http://lists.freeswitch.org               +  邮件列表
+                                                 +
```

```
+       irc.freenode.net / #freeswitch              +   IRC 聊天室
+-------------------------------------------------+
```

至此，FreeSWITCH 就已经安装完了。在 UNIX 类操作系统上，其默认的安装位置是 /usr/local/freeswitch（下文所述的路径全部相对于该路径）。两个常用的命令是 bin/freeswitch 和 bin/fs_cli（我们下面会讲到它们的用法），为了便于使用，建议将这两个命令做符号链接放到你的搜索路径中，如：

```
ln -sf /usr/local/freeswitch/bin/freeswitch /usr/bin/
ln -sf /usr/local/freeswitch/bin/fs_cli /usr/bin/
```

接下来 FreeSWITCH 就应该可以启动了。通过在终端中执行 freeswitch 命令 (如果你已做符号链接的话，否则要执行 /usr/local/freeswitch/bin/freeswitch) 可以将 FreeSWITCH 启动到前台。启动过程中会有许多 log 输出，第一次启动时会有一些错误和警告，可以不必理会[⊖]。启动完成后会进入系统控制台，并显示类似的提示符“ freeswitch@localhost>”(以下简称 freeswitch>)。通过在控制台中输入 shutdown 命令可以关闭 FreeSWITCH。

如果您想将 FreeSWITCH 启动到后台 (Daemon，服务模式)，可以使用 freeswitch -nc (即 No console)。后台模式没有控制台，如果想关闭 FreeSWITCH，可以直接在 Linux 提示符下通过 freeswitch -stop 命令实现。

不管 FreeSWITCH 运行在前面还是后台，都可以使用客户端软件 fs_cli 连接到它并对它进行控制。使用方法为：

```
/usr/local/freeswwitch/bin/fs_cli
```

当然，如果上面已经做了符号连接也可以直接运行 fs_cli。任何时间想退出 fs_cli 客户端，都可以输入 /exit 或按 Ctrl + D 组合键，也可以直接关掉终端窗口。

3.2.2　连接 SIP 电话

FreeSWITCH 最典型的应用是作为一个服务器（它实际上是一个背靠背的用户代理，即 B2BUA），并用电话客户端软件（一般叫软电话）连接到它。虽然 FreeSWITCH 支持 IAX、H323、Skype、Gtalk 等众多通信协议，但其最主要的协议还是 SIP。支持 SIP 的软电话有很多，笔者比较常用的是 X-Lite 和 Zoiper。这两款软电话都支持 Linux、Mac OS X 和 Windows 平台，免费使用但是不开源。在 Linux 上你还可以使用 Ekiga 软电话，它是开源的。

强烈建议在同一局域网上的其他机器上安装软电话，并确保麦克风和耳机可以正常工作。当然，如果你没有多余的机器做这个实验，也可以在同一台机器上安装。只是需要注意，软电话不要占用 UDP 5060 端口，因为 FreeSWITCH 默认要使用该端口，这是新手常会遇到的一个问题。你可以通过先启动 FreeSWITCH 再启动软电话来避免该问题（后者如果在启动时发现 5060 端口已被占用，一般会尝试选择其他端口），另外有些软电话允许你修改本

⊖ FreeSWITCH 在第一次启动时由于没有必要的数据库表，它会打印一些出错信息，并自创建这些表。只要下次启动时不出现错误，就可以暂时不必理会。

地监听端口[⊖]。

在 UNIX 类平台上，通过输入以下命令可以知道 FreeSWITCH 监听在哪个 IP 地址上，记住这个 IP 地址（:5060 以前的部分），下面要用到：

```
netstat -an | grep 5060

udp        0      0 192.168.0.9:5060          0.0.0.0:*
```

FreeSWITCH 默认配置了 1000 ~ 1019 共 20 个用户，你可以随便选择一个用户进行配置，配置过程如下：

1）在 X-Lite 上右击，选“ Sip Account Settings...”，单击“ Add”添加一个账号，填入以下参数（Zoiper 可参照配置）：

```
Display Name: 1000
User name: 1000
Password: 1234
Authorization user name: 1000
Domain: 你的 IP 地址，就是刚才你记住的那个
```

2）其他都使用默认设置，单击“ OK”按钮就可以了。然后单击“ Close”按钮关闭 Sip Account 设置窗口。这时 X-Lite 将自动向 FreeSWITCH 注册。注册成功后会显示“ Ready. Your username is 1000”，另外，左侧的“拨打电话”（Dial）按钮会变成绿色的，如图 3-11 所示。

值得一提的是，笔者使用的是一个旧版本的 X-Lite，之所以这么做，是因为考虑到大家可能对这个版本的 X-Lite 更熟悉一些。新版本的 X-Lite 界面如图 3-12 所示。

图 3-11 XLite 软电话注册后

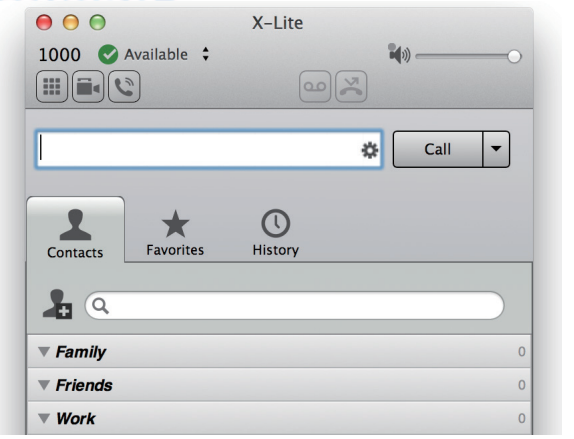

图 3-12 XLite 软电话 (新版)

激动人心的时刻就要来了。输入“9664”按回车（或按绿色拨打电话按钮），就应该能

⊖ 特别注意，如果你是在 Linux 上，并且在同一台机器上使用 Ekiga 的话，肯定会遇到这个问题。你需要手工使用 gconf_editor 来更改 Ekiga 使用的端口，当然，也可以改 FreeSWITCH 的端口，如果你会的话。

听到保持音乐（MOH）。如果听不到也不要气馁，看一下控制台上有没有提示什么错误。如果有“ File Not Found”之类的提示，多半是声音文件没有安装，重新查看 make moh-install 是否有错误。接下来，可以依次试试拨打表 3-1 所示的号码。

表 3-1　默认号码及说明

号码	说明
9664	保持音乐
9191	注册 ClueCon
9192	在 log 中显示 Channel 信息
9195	echo，回音测试，延迟 5 秒
9196	echo，回音测试
9197	milliwatte extension，铃音生成
9198	TGML 铃音生成示例
9180	铃音测试，使用远端生成的回铃音
9181	铃音测试，产生英式铃音
9182	铃音测试，使用音乐当铃音，彩铃
9183	先应答，然后发送英式铃音
9184	先应答，然后发送音乐铃音
9178	收传真
9179	发传真
5000	示例 IVR
4000	听取语音信箱
33xx	电话会议，48kHz（其中 xx 可为 00 ~ 99，下同）
32xx	电话会议，32kHz
31xx	电话会议，16kHz
30xx	电话会议，8kHz
2000-2002	呼叫组
1000-1019	默认分机号

详情见 http://wiki.freeswitch.org/wiki/Default_Dialplan_QRF。

另外，也许你想尝试注册另外一个 SIP 用户并在两者间通话。此时最好是在同一个局域网中的另外一台机器上启动另一个 X-Lite ，并使用 1001 注册，注册完毕后就可以在 1000 上呼叫 1001，或在 1001 上呼叫 1000 。当然，你仍然可以在同一台机器上做这件事（比方说用 Zoiper 注册为 1001 ），需要注意的是，由于你机器上只有一个声卡，两者可能会争用声音设备。特别是在 Linux 上，有些软件会独占声音设备。如果同时也有一个 USB 接口的耳机，那就可以设置不同的软件使用不同的声音设备。

如果你手边有硬件的 IP 话机，你也可以试一试。与传统的话机相比，IP 话机更加“智能”，功能也更丰富。因为硬件话机的设置方法和软件电话大同小异，所以只要明白上述软电话的设置，即可知道如何设置硬件话机了。我国产的话机质优价廉，在国际上都有

很好的口碑。下面我们分别以国产的亿联和潮流的话机为例，熟悉一下硬件话机注册到FreeSWITCH 的配置。

亿联（Yealink ㊀）话机是在国内能找到的质量比较好的话机，而且它有好多独有的特性。我们在后面的章节会讲到它的其他特性，这里我们先看看基本的配置。话机本身有一个液晶显示屏，并可以通过按键设置账号信息，但那样配置起来比较烦琐。在液晶屏上找到话机的 IP 地址以后㊁，用浏览器打开，界面如图 3-13 所示。

图 3-13 亿联话机的账号配置界面

账号配置跟软电话差不多，“显示名称”可以随便填，“注册名称”和“用户名称”这里我们都用 1002，“密码”是默认的 1234，“ SIP 服务器”处输入你的 IP 地址，其他的都保留默认设置，然后单击“提交”按钮。如果一切顺利，就能看到“账号状态”显示为“注册上”，这时就可以拨打 1000 或 1001 了。

潮流（Grandstream ㊂）话机也是质量不错的话机，配置和使用也比较方便。它的配置界面如图 3-14 所示。其中“账号名”可以随便填，“ SIP 服务器”中输入你的 IP 地址，“ SIP 用户 ID”、“认证 ID”及“名称”都填入 1003，“密码”也是默认的 1234。保存并提交后即可注册。潮流话机的注册状态是在单独的“状态”页面中显示的。

笔者使用这几款话机注册后相互拨打，彼此都能通，声音质量也很不错。

㊀ www.yealink.cn。

㊁ 一般使用 DHCP 启动后会自动获得一个 IP 地址，否则，也可以设一个静态的 IP。

㊂ www.grandstream.com.cn。

图 3-14　潮流话机的账号配置界面

3.3　配置 FreeSWITCH

FreeSWITCH 配置文件默认放在 conf/ 下，它由一系列 XML 配置文件组成。最顶层的文件是 freeswitch.xml，系统启动时它依次装入其他一些 XML 文件并最终组成一个大的 XML 文件。基本的目录结构和主要配置文件如表 3-2 所示。

表 3-2　配置文件的目录结构

conf/ 目录和文件	说明
├── vars.xml	一些常用变量
├── switch.xml	主配置文件，它会使用 include 语句装入其他文件
├── autoload_configs	目录，存放自动加载的配置文件
│　├── modules.conf.xml	配置当 FreeSWITCH 启动时自动装载哪些模块
│　├── *.xml	一般来说每个模块都有一个配置文件
├── chatplan	聊天计划
├── dialplan	拨号计划
│　├── default.xml	默认的拨号计划配置，一般用于内部用户路由
│　├── public.xml	默认的拨号计划配置，一般用于外部来话路由
├── directory	用户目录
│　└── default	默认的用户目录配置
│　　├── *.xml	SIP 用户，每用户一个文件

（续）

conf/ 目录和文件	说明
├── ivr_menus	IVR 菜单
├── jingle_profiles	连接 Google Talk 的相关配置
├── lang	多语言支持
│ ├── en	英语
│ └── fr	法语
├── mrcp_profiles	MRCP 的相关配置，用于跟第三方语音合成和语音识别系统对接
├── sip_profiles	SIP 配置文件
│ ├── internal.xml	一个 SIP profile，或称作一个 SIP-UA，监听在本地 IP 及端口 5060
│ │	一般供内网用户使用
│ ├── externa.xml	另一个 SIP-UA，用作外部连接，端口 5080
└── skinny_profiles	思科 SCCP 协议话机的配置文件

下面我们先通过学习添加一个新的 FreeSWITCH 用户来简单熟悉一下 FreeSWITCH 的配置文件。

FreeSWITCH 默认设置了 20 个用户（1000 ~ 1019），如果你需要更多的用户，或者想通过添加一个用户来学习 FreeSWITCH 配置，只需要简单执行以下三步：

1）在 conf/directory/default/ 中增加一个用户配置文件。

2）修改拨号计划（Dialplan）使其他用户可以呼叫到它。

3）重新加载配置使其生效。

例如我们想添加用户 Jack，分机号是 1234。只需要到 conf/directory/default 目录下，将 1000.xml 复制到 1234.xml 中。打开 1234.xml，将所有 1000 都改为 1234。并把 effective_caller_id_name 的值改为 Jack，然后存盘退出，命令如下：

```
<variable name="effective_caller_id_name" value="Jack"/>
```

接下来，打开 conf/dialplan/default.xml，找到下面一行

```
<condition field="destination_number" expression="^(10[01][0-9])$">
```

将其改为

```
<condition field="destination_number" expression="^(10[01][0-9]|1234)$">
```

熟悉正则表达式的读者应该知道，“^(10[01][0-9])$”匹配被叫号码 1000 ~ 1019。因此我们修改之后的表达式就多匹配了一个 1234。FreeSWITCH 使用 Perl 兼容的正则表达式（PCRE）。

现在，回到控制台或启动 fs_cli，执行 reloadxml 命令或按快捷 F6，使新的配置生效。

找到刚才注册为 1001 的软电话（或启动一个新的，如果你有足够的机器的话），把 1001 都改为 1234 然后重新注册，这时就可以与 1000 相互进行拨打测试了。如果没有多台机器，在同一台机器上运行多个软电话可能有冲突，这时可以直接进入 FreeSWITCH 控制台使用如下命令进行测试：

```
freeswitch> sofia status profile internal reg      (显示多少用户已注册)
freeswitch> originate user/1000 &echo              (同上)
freeswitch> originate user/1000 9999               (相当于在软电话 1000 上拨打 9999)
freeswitch> originate user/1000 9999 XML default (同上)
```

其中，echo 程序是一个很简单的程序（App），它只是将你说话的内容原样再放给你听，在测试时很有用，在本书中我们会经常用它来测试。

3.4　FreeSWITCH 用作软电话

也可以把 FreeSWITCH 简单地用作一个软电话（可以看作用 FreeSWITCH 做了一个 X-Lite）。虽然相比而言，FreeSWITCH 比配置 X-Lite 略微复杂一些，但你会从中得到更多好处：FreeSWITCH 是开源的，更强大、灵活。关键是它是目前笔者所知道的唯一支持 CELT 高清通话的软电话。

FreeSWITCH 使用 mod_portaudio 模块支持你本地的音频设备，该模块默认是不编译的。在你的源代码目录下执行如下命令，以安装该模块：

```
make mod_portaudio
make mod_portaudio-install
```

其他的模块也可以依照上面的方式进行重新编译和安装。安装完成后到控制台中执行：

```
freeswitch> load mod_portaudio
```

如果得到“Cannot find an input device”之类的错误，则可能是你的声卡驱动有问题。如果是提示“+OK”就是成功了。接着执行 pa devlist 命令，可以看到如下输出：

```
freeswitch> pa devlist

API CALL [pa(devlist)] output:
0;Built-in Microphone;2;0;
1;Built-in Speaker;0;2;r
2;Built-in Headphone;0;2;
3;Logitech USB Headset;0;2;o
4;Logitech USB Headset;1;0;i
```

以上是笔者的笔记本电脑上的输出，它列出了所有的声音设备。其中，3 和 4 最后的“o”和“i”分别代表声音输出（out）和输入（in）设备。在你的电脑上可能不一样，如果你想选择其他设备，可以使用命令进行修改。例如下列命令可以选择使用笔者电脑上内置的麦克风和耳机：

```
freeswitch> pa indev #0
freeswitch> pa outdev #2
```

至此你就有了一个可以用命令行控制的软电话了。尝试输入以下命令：

```
freeswitch> pa looptest       (回路测试，echo)
freeswitch> pa call 9196      (呼叫 9196)
freeswitch> pa call 1000      (呼叫 1000)
freeswitch> pa hangup         (挂机)
```

如上所示，你可以呼叫刚才试过的所有号码。现在假设想从 SIP 分机 1000 呼叫到你，那么需要修改拨号计划 (Dialplan)。用你喜欢的编辑器编辑以下文件并放到 conf/dialplan/default/portaudio.xml 中：

```
<include>
  <extension name="call me">
    <condition field="destination_number" expression="^(me|12345678)$">
      <action application="bridge" data="portaudio"/>
    </condition>
  </extension>
</include>
```

然后，在控制台中按 F6 或输入以下命令使之生效：

```
freeswitch> reloadxml
```

在分机 1000 上呼叫 me 或 12345678（你肯定想为自己选择一个更酷的号码），然后在控制台上应该能看到类似 [DEBUG] mod_portaudio.c:268 BRRRRING! BRRRRING! call 1 的输出（如果看不到，按 F8 能得到详细的 log），这说明你的软电话在振铃。多按几个回车，然后输入 pa answer 就可以接听电话了。输入 pa hangup 可以挂断电话。

当然，你肯定希望在振铃时能听到真正的振铃音而不是看什么 BRRRRRING。好办，选择一个好听的声音文件（WAV 格式），编辑 conf/autoload_configs/portaudio.conf.xml，将 ring-file 一行修改为下面的样子，其中，value 指定你的声音文件的路径：

```
<param name="ring-file" value="/home/your_name/your_ring_file.wav"/>
```

然后重新加载模块：

```
freeswitch> reloadxml
freeswitch> reload mod_portaudio
```

再打打试试，看是否能听到振铃音了？

如果你用不习惯字符界面，可以看一下 FreeSWITCH-Air（http://www.freeswitch.org.cn/download），它是使用 Adobe Air 开发的，为 FreeSWITCH 提供一个简洁的软电话的图形界面。另外，如果你需要高清通话，除需要设置相关的语音编解码器（codec）外，你还需要有一个好的耳机才能达到最好的效果。笔者使用的是一款 Logitech 的 USB 耳机。除此之外，还有两款基于 FreeSWITCH 的软电话，分别是 FSComm[⊖]（QT 实现）和 FSClient[⊜]（C# 实现）。

3.5 配置 SIP 网关拨打外部电话

如果你拥有某个运营商提供的 SIP 账号，那么你就可以通过配置 SIP 来拨打外部电话了。该 SIP 账号（或提供该账号的设备）在 FreeSWITCH 中称为 SIP 网关（Gateway）。添加一个网关只需要在 conf/sip_profiles/external/ 中创建一个 XML 文件，名字可以随便起，如

⊖ http://wiki.freeswitch.org/wiki/FSComm

⊜ http://wiki.freeswitch.org/wiki/FSClient

gw1.xml，然后在该文件中输入如下代码：

```
<gateway name="gw1">
    <param name="realm" value="SIP 服务器地址，可以是 IP 或 IP:端口号 "/>
    <param name="username" value="SIP 用户名 "/>
    <param name="password" value=" 密码 "/>
</gateway>
```

如果你的 SIP 网关还需要其他参数，可以参阅同目录下的 example.xml，但一般来说上述参数就够了⊖。你可以重启 FreeSWITCH，或者执行以下命令使用之生效：

```
freeswitch> sofia profile external rescan
```

显示一下网关的注册状态：

```
freeswitch> sofia status
```

如果显示 gateway gw1 的状态是 REGED，则表明已正确地注册到了网关上。你可以先用命令试一下网关是否工作正常：

```
freeswitch> originate sofia/gateway/gw1/xxxxxx &echo
```

以上命令会通过网关 gw1 呼叫号码 xxxxxx（可能是你的手机号），被叫号码接听电话后，FreeSWITCH 会执行 echo 程序，你应该就能听到自己的回音了。

当然，世界 SIP 网关五花八门，你实际操作起来可能不会这么顺利。如果真的在这里遇到问题，那么你大可继续往下阅读，相信你读完本书之后，跟任何网关对接的复杂问题都能迎刃而解了。

3.5.1　从某一分机上呼出

如果网关测试正常，你就可以配置从你的 SIP 软电话或 portaudio 呼出了。由于我们是把 FreeSWITCH 当作 PBX 用，所以需要选一个出局字冠。常见的 PBX 一般是内部拨小号，打外部电话就需要加拨 0 或先拨 9 。当然，这是你自己的交换机，你可以用任何你喜欢的数字（甚至是字母）。继续修改拨号计划，创建一个新的 XML 文件—— conf/dialplan/default/call_out.xml，内容如下：

```
<include>
  <extension name="call out">
    <condition field="destination_number" expression="^0(\d+)$">
      <action application="bridge" data="sofia/gateway/gw1/$1"/>
    </condition>
  </extension>
</include>
```

其中，“ ^0(\d+)$”为正则表达式，“ (\d+)”匹配 0 后面的所有数字并存到变量 $1 中。然后通过 bridge 程序通过网关 gw1 打出该号码。当然，建立该 XML 后需要在控制台中执行 reloadxml 使之生效。

⊖ 当添加网关对外注册时，FreeSWITCH 就相当于一个 SIP 客户端，读者可以对比一下，看与配置 X-Lite 有何不同。

3.5.2 呼入电话处理

如果你的 SIP 网关支持 DID[⊖]，那么你需要知道呼入的 DID 号码。一般来说，呼入的 DID 就是你的 SIP 号码，如果你不知道，也没关系，学习完第 6 章你就能学到如何取得这个值了。创建以下 XML 文件并放到 conf/dialplan/public/my_did.xml 中：

```
<include>
  <extension name="public_did">
    <condition field="destination_number" expression="^(你的 DID)$">
      <action application="transfer" data="1000 XML default"/>
    </condition>
  </extension>
</include>
```

在 FreeSWITCH 中执行 reloadxml 使之生效。上述配置会将来话直接转接到分机 1000 上。在后面的章节中你会学到如何更灵活地处理呼入电话，如转接到语音菜单或语音信箱等。

3.6 小结

本章涵盖了 FreeSWITCH 在 Windows、Linux、MacOSX 三大平台上从安装、配置到调试、使用等相关内容。

如果你能顺利走到这儿，则说明你对 FreeSWITCH 已经爱不释手了。如果你卡在了某处，或某些功能未能实现，这也不是你的错，主要是因为 FreeSWITCH 博大精深，笔者不能在短短的一章内把所有的方面解释清楚。在后面的章节中，你会学到更多的基本概念，从而更加深入地了解 FreeSWITCH 的哲学，也会学到更多的调试技术和技巧，那时解决任何问题都会是小菜一碟了。

最后需要注意的一个问题是，由于版权的限制或某些其他原因，FreeSWITCH 在从源代码编译安装时会从网上自动下载相关的工具、第三方库以及声音文件等（尤其是在 Windows 系统上）。因此，如果编辑安装时没有互联网环境，可能会出现很多奇怪的错误。这时候，笔者建议初学者在有互联网环境的条件下编译，并保证能顺利连接 http://files.freeswitch.org[⊜]。如果你已经对 FreeSWITCH 比较熟悉了，又必须要在内网环境下编译，可以事先将需要的文件下载好，并放到对应的位置。通过对比有互联的环境下的编译日志能够找到这些位置，关于在不能访问互联网的情况下的编译和安装留给有兴趣的读者自己去练习，在此我们就不多介绍了。

⊖ Direct Inbound Dial，即对内直接呼入。

⊜ files.freeswitch.org 由 CDN 提供，因此在世界各地可能连接到不同的 IP 地址。如果不能正常连接时可以尝试找出你所处区域解析到的 IP 地址，并向官方说明该问题。或者，可以尝试使用其他的 DNS 服务器，如 Google 的 8.8.8.8 及 4.4.4.4 等。