机 密

文档号：**W-3-3-XXX-B**
版本号：**V4.0**
日　期：**2007/09/03**
作　者：

SoftPhone 使用手册

修订历史记录

版本号	日 期	描 述	作 者

目　　录

1. 概　述 1

1.1 目　标 1
1.2 读　者 1
1.3 参考文档 1
1.4 符号说明 1

2. 接口描述 2

2.1 OCX 应用环境 2
2.2 OCX 的原理 2
2.3 OCX 对业务系统状态进行控制的原理 3
2.3.1 使用 OCX 自己提供的按钮界面 3
2.3.2 业务系统自己绘制按钮界面 3
2.4 OCX 的基本功能说明 4
2.4.1 坐席配置 4
2.4.2 坐席签入 5
2.4.3 坐席签出 6
2.4.4 坐席自动签入 6
2.4.5 调整 OCX 的按钮的大小 6
2.4.6 设置自绘制按钮的状态 6
2.4.7 取得软电话状态描述 10
2.4.8 电话进入 10
2.4.9 应答处理 10
2.4.10 挂机处理 11
2.4.11 外拨处理 11
2.4.12 保持处理 11
2.4.13 取消保持处理 12
2.4.14 暂停处理 12
2.4.15 单步转接处理 12
2.4.16 磋商处理 14
2.4.17 磋商转接处理 16
2.4.18 三方会议处理 16
2.4.19 发送传真处理 16
2.4.20 接收传真处理 17
2.4.21 远程桌面监控处理 19
2.4.22 随路数据处理 20
2.4.23 转自动流程处理 21
2.4.24 自动话后结束处理 22
2.4.25 IVR 语音信箱留言查听处理 22
2.4.26 Email 呼叫处理 24
2.4.27 TextChat 以及 WebCall 呼叫处理 27

3. 基于 VB 的 DEMO 程序的搭建 28

3.1 程序搭建过程 28

3.1.1 创建工程 28

3.1.2 在工程中添加 OCX 控件 29

3.1.3 在界面上添加 OCX 控件 30

3.1.4 完成主界面 32

3.1.5 完成相关界面 32

3.1.6 Login 的代码实现 33

3.1.7 暂停处理的实现 34

3.1.8 电话应答的实现 36

3.1.9 保持的实现： 37

3.1.10 单步转接的实现： 37

3.1.11 外拨的实现： 40

3.1.12 磋商的实现： 40

3.1.13 暂停的实现： 41

3.1.14 会议的实现： 42

3.1.15 监听的实现： 42

3.1.16 强插的实现： 43

3.1.17 拦截的实现： 43

3.1.18 状态栏显示： 43

3.1.19 注册表配置： 43

3.1.20 自定义座席按钮： 43

4. 嵌入 HTML 文件的实现 50

4.1 OCX 控件插入 50

4.2 设计基本界面 50

4.3 设置按钮的初始状态： 51

4.4 按钮的标题及状态变化处理 52

4.5 登录的实现 55

4.6 登出的实现 55

4.7 暂停的实现： 56

4.8 应答和挂机的实现： 56

4.9 保持和取消保持的实现： 56

4.10 单步转接的实现： 57

4.11 外拨的实现： 57

4.12 磋商的实现: 58

4.13 磋商转接的实现: 58

4.14 磋商进入三方会议的实现: 59

4.15 传真的实现: 59

4.16 监听的实现: 59

4.17 强拆的实现: 59

4.18 拦截的实现: 60

4.19 新呼叫进入事件处理： 60

4.20 播放工号的实现：.....60
4.21 状态显示:.....60
4.22 配置:.....60

5. 基于 C#.NET 的 DEMO 程序的搭建.....62

5.1 程序搭建过程.....62
5.1.1 创建工程.....62
5.1.2 在工程中添加 OCX 控件.....62
5.1.3 在界面上添加 OCX 控件.....63
5.1.4 完成主界面.....65
5.1.5 完成相关界面.....66
5.1.6 Login 的代码实现.....67
5.1.7 暂停处理的实现.....68
5.1.8 电话应答的实现.....70
5.1.9 保持的实现：.....71
5.1.10 单步转接的实现：.....72
5.1.11 外拨的实现：.....74
5.1.12 磋商的实现：.....74
5.1.13 暂停的实现：.....75
5.1.14 会议的实现：.....76
5.1.15 监听的实现：.....76
5.1.16 强插的实现：.....77
5.1.17 拦截的实现：.....77
5.1.18 状态栏显示：.....77
5.1.19 注册表配置：.....77
5.1.20 自定义座席按钮：.....77

6. 基于 DELPHI 的 DEMO 程序的搭建.....88

6.1 程序搭建过程.....88
6.1.1 创建工程.....88
6.1.2 OCX 控件注册.....88
6.1.3 在界面上添加 OCX 控件.....89
6.1.4 完成主界面.....91
6.1.5 完成相关界面.....91
6.1.6 Login 的代码实现.....92
6.1.7 暂停处理的实现.....93
6.1.8 电话应答的实现.....95
6.1.9 保持的实现：.....97
6.1.10 单步转接的实现：.....97
6.1.11 外拨的实现：.....98
6.1.12 暂停的实现：.....100
6.1.13 会议的实现：.....101
6.1.14 监听的实现：.....101

6.1.15 强插的实现：.....102
6.1.16 拦截的实现：.....102
6.1.17 状态栏显示：.....102
6.1.18 注册表配置：.....102
6.1.19 自定义座席按钮：.....102

7. 基于 VISUALC++6.0 的 DEMO 程序的搭建.....111

7.1 程序搭建过程.....111
7.1.1 创建工程.....111
7.1.2 在工程中添加 OCX 控件.....111
7.1.3 在界面上添加 OCX 控件.....112
7.1.4 完成主界面.....114
7.1.5 完成相关界面.....114
7.1.6 Login 的代码实现.....115
7.1.7 电话应答的实现.....119
7.1.8 保持的实现：.....121
7.1.9 单步转接的实现：.....121
7.1.10 外拨的实现：.....123
7.1.11 磋商的实现：.....125
7.1.12 暂停的实现：.....125
7.1.13 会议的实现：.....126
7.1.14 监听的实现：.....126
7.1.15 强插的实现：.....127
7.1.16 拦截的实现：.....127
7.1.17 状态标签 CStatic 显示：.....127
7.1.18 注册表配置：.....127
7.1.19 自定义座席按钮：.....127

1.概　述

1.1 目　标

详细介绍 CTXWare SoftPhone Control(OCX 控件)编程接口，为基于该控件开发座席业务系统的开发人员提供相关的技术支持。

1.2 读　者

基于 SoftPhone Control 开发座席业务系统的技术人员。

1.3 参考文档

序号	文档名称	版本号	作 者

1.4 符号说明

- **MIS**：Manager Info System（管理信息系统），即呼叫中心人工服务业务系统。
- **SoftPhone**：在本文中概念等同于“软话机”或“OCX 控件”

2. 接口描述

2.1 OCX 应用环境

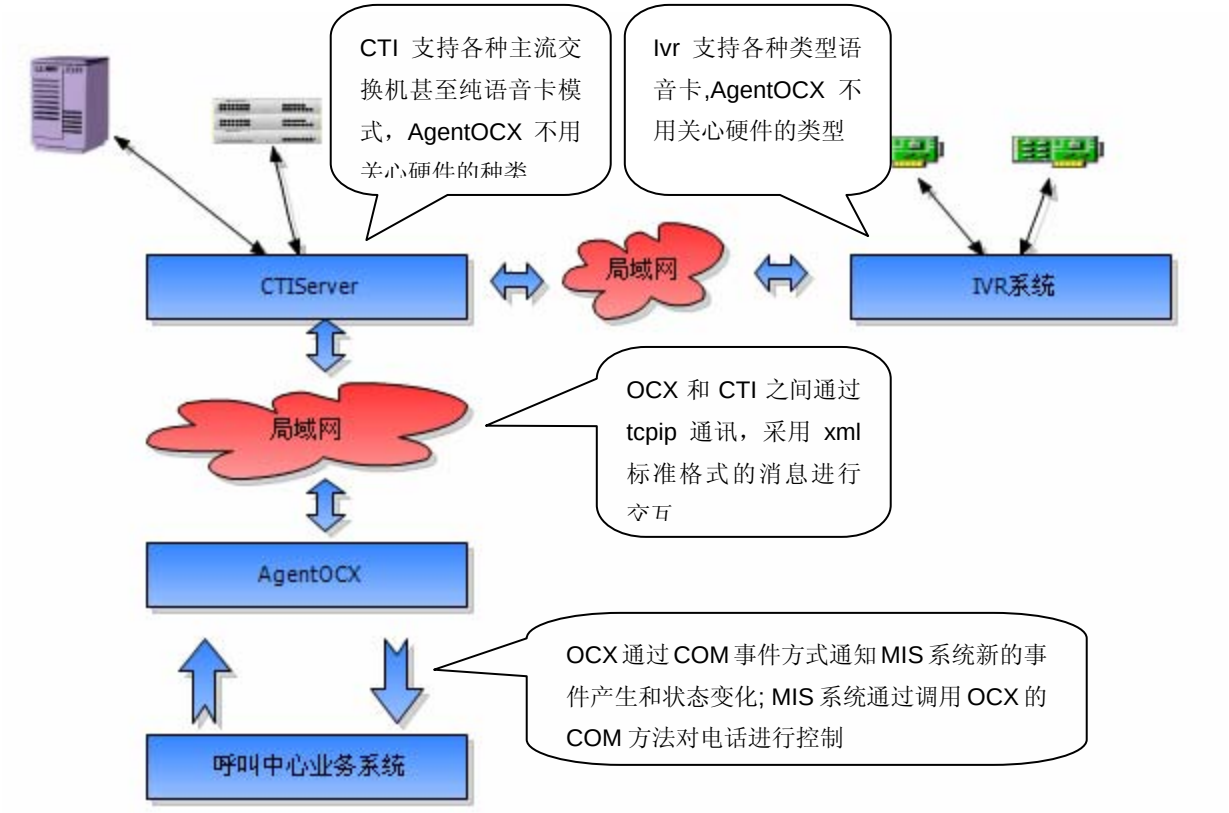
CTXWare Softphone ocx 可以适用于支持 COM 组件的第 3 方 C/S 开发环境，也可以嵌入 B/S 应用的网页中。

常见的支持 COM 组件的 C/S 开发环境有：Visual C++,Visual Basic,Delphi,C++ Builder,PowerBuilder， C#，VB.NET,VC.Net 等。

在 B/s（浏览器-服务器）的应用环境中，可以有两种方式和业务系统相整合：

- CTXWare Softphone ocx 可以作为 ActiveX 控件嵌入网页中；
- 可以使用 C/S 开发环境制作一个外壳，CTXWare Softphone ocx 嵌入在外壳中。外壳携带一个 IE 浏览器控件,利用浏览器控件来显示 B/s 的客户端网页；

2.2 OCX 的原理



图表 1 OCX 的原理

OCX 的原理如上图所示，其设计特点如下：

1） AgentOCX 和呼叫中心业务系统(MIS 系统)之间通过标准的 COM 协议进行交互。OCX 通过 COM 事件方式通知 MIS 系统新的事件产生和状态变化; MIS 系统通过调用 OCX 的 COM 方法对电话进行控制。

2） AgentOCX 和 CTI 之间通过 tcpip 通讯，发送 XML 消息包进行交互。

3） CTI 可以支持各种主流的交换机甚至纯语音卡环境；而不论什么样的硬件环境，CTI 和 AgentOCX 之间的接口是统一的，和硬件环境无关的。这种 ocx 和硬件环境的松耦合模式，使得呼叫中心坐席应用可以支持广泛的硬件配置，具有很好的扩展性。比如，当呼叫中心集成商第一次使用 CTXWare 中间件的应用中，初期采用的是小规模的语音卡环境；而随着业务的扩大，话务量很快增长，系统硬件升级为大规模的交换机模式，那么现有的坐席端业务系统不需要作改变即可平滑升级。

4） Ocx 和 IVR 系统之间是通过 CTI 进行交互的，比如呼叫从 IVR 系统分配到坐席，或者坐席把呼叫转回 IVR 系统，都通过 CTI 进行中间控制处理。因此 ocx 和 IVR 系统的语音卡种类无关。Ocx 和 IVR 系统硬件的松耦合关系也为系统应用提供了很好的扩展性。

2.3 OCX 对业务系统状态进行控制的原理

Ocx 的使用分为两种模式：

1） 使用 OCX 自己提供的按钮界面；

2） 业务系统自己绘制按钮界面；

2.3.1 使用 OCX 自己提供的按钮界面

当使用 OCX 提供的按钮界面时，OCX 提供如下的界面：

如果业务系统使用 OCX 自己提供的按钮界面，则可以不用考虑软电话按钮的各种状态；则本节后面描述的内容可以不用继续阅读；

2.3.2 业务系统自己绘制按钮界面

大多数情况下，OCX 的界面和业务系统的风格不统一。因此，业务系统更希望自己绘制按钮和界面。

开发过呼叫中心坐席端软件的工程师都知道，坐席端对电话控制部分的状态逻辑是最麻烦的问题。坐席端软件界面一般提供了 10 个甚至更多的软电话控制按钮供坐席使用，不同的情况下按钮的状态都不一样，组合出很多的状态。

很多中间件厂商的坐席端 ocx 或者 dll 提供的只是单纯的 API 函数；因此调用某个单个功能确实很简单，比如坐席端软件要发起外拨，点击外拨按钮的时候，将调用类似如下的函数：

makecall (“114”)或者 ocx.makecall(“114”)。

如上的单个功能的调用很简单，但是处理按钮的状态很繁琐；比如点击外拨按钮后，软电话应该进入一种“外拨中的中间状态”，此时很多按钮的状态都应该属于禁用状态（灰色），并且外拨按钮不允许被重复点击，并且应该提供一个“取消外拨”的按钮可供坐席取消尚未接通的外拨操作。

上述只是一个非常简单的例子，而实际的情况下，坐席软电话会存在上百种状态，状态之间的跳转组合非常多；而尤其在大量坐席（几十个甚至上百个）同时使用的情况下，坐席之间的转接、磋商、会议、监听频繁发生，使得处理坐席的状态更加具有难度，并且更加容易出错误和不稳定；

举例说明：

在正常通话情况下，收到系统的呼叫挂断事件后，坐席软电话应该进入话后处理（或者示闲）状态；但是，在磋商过程中，收到系统的挂断事件后，必须区分是哪一个呼叫的挂断（此时坐席电话上具有两个呼叫，一个是被保持的呼叫、一个是外拨的呼叫），如果是被保持一方的呼叫挂断，则本坐席应该进入通话状态（而非话后或者示闲）→继续和被磋商者通话；如果是被磋商者通话被挂断，则本坐席仍然应该进入通话状态（和被保持的通话者通话，）而被保持的呼叫应该调用取消保持。

上述就是一个状态变化的例子。由于在不同状态下会产生各种事件，所以状态的变化图是非常繁琐的。

为了不让坐席端业务系统开发人员陷入这种繁琐的状态处理过程中，CTXWare AgentOCX 提供了方便的处理状态的机制。

CTXWare AgentOCX 提供了专门帮助业务系统处理状态的几个事件：

- EVTButtonStatus(btnName as String,btnTitle As String, bEnable As Long) 本事件在状态改变时被触发，并携带 3 个参数：btnName 表示按钮的名称、btnTitle 表示按钮的标题、bEnable 表示按钮是可用还是禁用。业务系统只需要处理本事件，并修改相应的按钮标题和可用/禁用状态，即可完美实现复杂的状态机。可以说，业务系统只需要处理此事件，不需要关心按钮界面复杂的状态机→复杂状态机处理简化成了简单的按钮基本属性操作。
- EVTReturnStatusCH(sStatus as string) 本事件在状态改变时被触发，参数 sStatus 传递了当前状态的中文名称；业务系统可以直接把中文状态名称在自己的界面中显示出来。

上述方式的特点如下：

- 通过提供上述两个状态事件，AgentOCX 把状态机的复杂处理完全和业务系统屏蔽开来，使得可能需要一周甚至一个月的编程工作量简化为半天或者一天的编程工作量。
- 如果自己处理软电话按钮的状态，必须对 CTI 的底层工作机制有所了解（比如必须了解保持、磋商、会议、转接、监听等等的状态原理）。而不同种类的交换机、甚至纯粹语音卡环境对上述 CTI 状态的处理都不相同，会产生很多工作量。
- 业务系统开发人员在处理状态机制时，在实际的应用中可能存在 Bug，影响了项目的进度；而 CTXWare AgentOCX 把状态的处理封装起来，对外提供简洁易用的接口，并且久经考验更加稳定。

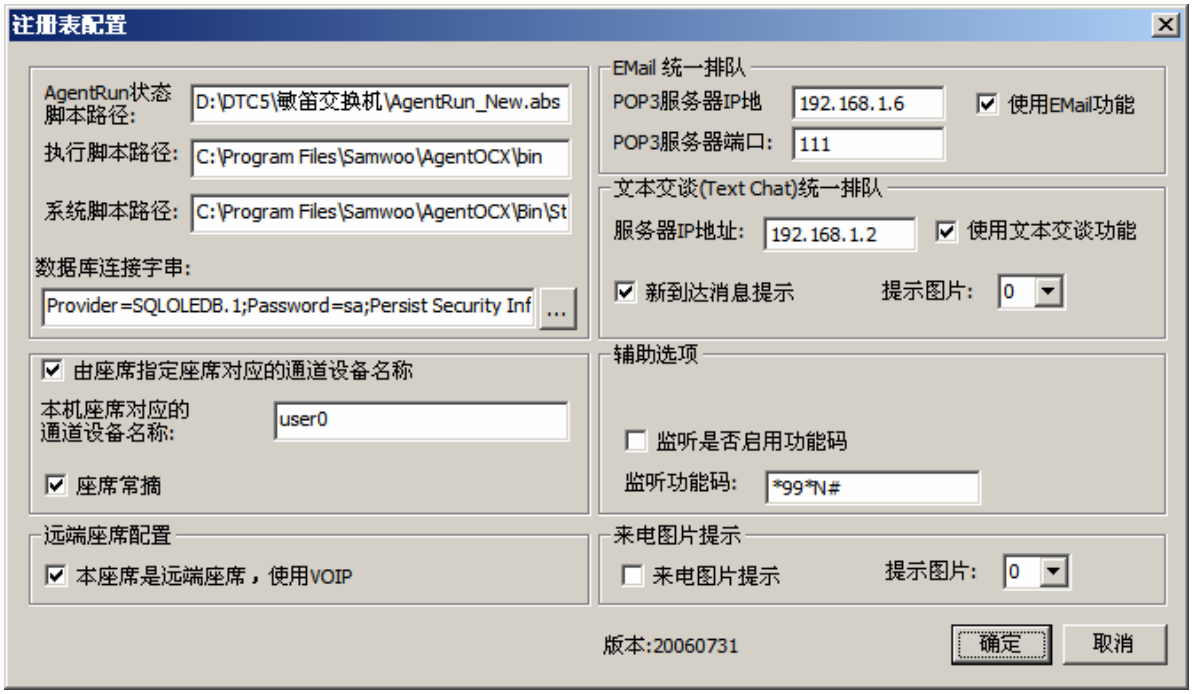
2.4 OCX 的基本功能说明

本章描述 OCX 的基本功能。

2.4.1 坐席配置

使用坐席之前必须配置坐席。调用 ShowConfig 方法。

ShowConfig 方法会弹出如下界面：



图表 2 坐席配置界面

关键的配置项描述如下：

- 数据库连接字串：AgentOCX 需要使用的数据库连接。注意，为了支持 CTIServer 的多机热备，所以 AgentOCX 不会直接配置 CTIServer 的 ip 地址，而是从数据库中取得当前的主 CTIServer 的 Ip 地址。
- 本机坐席对应的通道设备名称：本坐席所需要操作的通道号。对于交换机环境，这个参数就是坐席要使用的电话的号码；对于纯语音卡环境，这个参数是坐席卡的通道名称；对于东进、三汇卡，名称为 user0、user1、。。。userX，对于 Dialogic 语音卡，名称为 msiBxCx 等。
- 来电图片提示：是否在来电时，显示图片通知。（类似 msn 消息通知）

2.4.2 坐席签入

坐席登录需要设置 AgentOCX 的 AgentID 和 Password 属性，并调用 LogIn 方法。

例子如下：

```
#001 ocx.AgentID = “Agent2”
#002 ocx.Password = “2”
#003 ocx.Login
```

上述例子设置 ocx 的登录工号为 Agent2，密码为 2，然后调用 Login 方法。

注意：Login 方法为异步方法。登录的结果由事件通知。登录结果的事件有如下两个：

- EVTLoginFailed(reason As String) →登录失败；携带一个参数:reason，内容为登录失败的原因；
- EVTLoginSuc()→登录成功；

2.4.3 坐席签出

坐席签出调用 LogOUT 方法。LogOut 底层会关闭和 CTIServer 的 Tcpip socket 连接。LogOut 没有相关的事件。

注意：LogOUT 不会影响现有物理电话的通话。即：如果坐席已经和客户通话，则 LogOUT 不会挂断通话。

2.4.4 坐席自动签入

坐席登录后，在以下情况下，AgentOCX 可以自动签入：

- 网络意外中断；如网络抖动、网线被拔下等；一旦网络恢复正常，AgentOCX 会自动签入坐席；
- CTIServer 异常重新启动；这种现象会发生在和交换机的 Link 链路 Restart 过程或者 CTIServer 的自恢复过程，此时 AgentOCX 会自动侦测 CTIServer 的运行并自动连接；

注意：如果调用了 LogOUT 方法，AgentOCX 会停止自动签入。如果没有主动调用 LogOUT 方法，任何情况的和 CTIServer 之间的失去通讯，AgentOCX 都会尝试自动签入；自动签入的机制保证了在网络出现抖动等情况下的坐席端的持续工作。并且自动签入不会影响现有的物理通话。

自动签入不需要调用任何方法。

2.4.5 调整 OCX 的按钮的大小

在坐席业务系统界面直接使用 OCX 自带的按钮时，可以设置 ocx 的 PicSize 属性来调整 ocx 的按钮的大小。如下图的 PicSize 为 320：

2.4.6 设置自绘制按钮的状态

在坐席业务系统界面使用自绘制的按钮时，通过处理 EVTButtonStatus(btnName as String,btnTitle As String, bEnable As Long)事件来设置按钮的状态。有关设置按钮状态的详细描述见“OCX 对业务系统状态进行控制的原理”一节。

2.4.6.1 定义按钮

坐席业务界面应该添加如下的按钮：

按钮名称	说明
btnHook	摘机、挂机
btnHold	保持、取消保持（注：是单步转接/盲转）
btnTransfer	转接、取消转接
btnDialOut	外拨、停止外拨
btnConsultation	磋商、取消磋商
btnAuto	磋商转接（注：是两步转接）

btnFax	发送传真
btnPause	暂停、恢复（注：即小休功能）
btnConference	三方会议（注：是两步会议，必须磋商后才能进入会议）
btnListen	监听
btnDisconnect	强插
btnRopCall	拦截

注意：上述按钮的命名是建议；因为处理 EvtButtonStatus 事件时会方便一些。

2.4.6.2 处理 EvtButtonStatus 事件

当软电话底层的状态机发生变化时，会触发 EvtButtonStatus；EvtButtonStatus 的参数描述如下：

- Name→按钮的名称；字符串类型
- Title→按钮的标题；Title 返回的都是英文，可以转换成合适的中文；字符串类型
- Enable→按钮的允许/禁止；字符串类型

处理 EvtButtonStatus 事件的代码如下（以 visual basic 为例子）：

先把 Title 转换成中文（中文的选择可以按需要修改，比如“OffHook”可以显示为“摘机”，也可显示为“应答”）

```
#001 Select Case Title
#002        Case "OnHook"
#003            Title = "挂机"
#004        Case "OffHook"
#005            Title = "摘机"
#006        Case "HOOKOFFCONSULT"
#007            Title = "接受"
#008        Case "Hold"
#009            Title = "保持"
#010        Case "HoldCancel"
#011            Title = "取消"
#012        Case "Transfer"
#013            Title = "转接"
#014        Case "CancelTransfer"
#015            Title = "取消"
#016        Case "DialOut"
#017            Title = "外拨"
#018        Case "CancelDialOut"
#019            Title = "取消"
#020        Case "Consultation"
```

```
#021            Title = "磋商"
#022        Case "CancelConsultation"
#023            Title = "取消"
#024        Case "StopConsultation"
#025            Title = "磋商结束"
#026        Case "ConsultTransfer"
#027            Title = "磋商转接"
#028        Case "Auto"
#029            Title = "磋商转接"
#030        Case "OutPhone"
#031            Title = "自动"
#032        Case "CancelOutPhone"
#033            Title = "取消"
#034        Case "Play"
#035            Title = "放音"
#036        Case "PlayCancel"
#037            Title = "结束"
#038        Case "Fax"
#039            Title = "传真"
#040        Case "FaxStop"
#041            Title = "结束"
#042        Case "Pause"
#043            Title = "暂停"
#044        Case "Continue"
#045            Title = "恢复"
#046        Case "ContinueDialTask"
#047            Title = "放弃回访"
#048        Case "Listen"
#049            Title = "监听"
#050        Case "CancelListen"
#051            Title = "结束"
#052        Case "Disconnect"
#053            Title = "强插"
#054        Case "Conference"
#055            Title = "会议"
```

```
#056            Case "CancelConference"
#057                Title = "取消"
#058            Case "RopCall"
#059                Title = "拦截"
#060            Case "LOGINSUCC"
#061                Title = "登录成功"
#062            Case "LOGINFAIL"
#063                Title = "登录失败"
#064            Case "TRANSFERFAIL"
#065                Title = "转接失败"
#066 End Select
```

下面设置各个按钮的标题和状态。注意，下面的代码段使用了 SetButtonOption 函数，是设置某个按钮的标题和状态的自定义的函数：

```
#001 Select Case Name
#002            Case "Hook"
#003                SetButtonOption Title, Enable, Me.btnHook
#004            Case "Hold"
#005                SetButtonOption Title, Enable, Me.btnHold
#006            Case "Transfer"
#007                SetButtonOption Title, Enable, Me.btnTransfer
#008            Case "DialOut"
#009                SetButtonOption Title, Enable, Me.btnDialOut
#010            Case "Consultation"
#011                SetButtonOption Title, Enable, Me.btnConsultation
#012            Case "Auto"
#013                SetButtonOption Title, Enable, Me.btnAuto
#014            Case "Fax"
#015                SetButtonOption Title, Enable, Me.btnFax
#016            Case "Pause"
#017                SetButtonOption Title, Enable, Me.btnPause
#018
#019            Case "Conference"
#020                SetButtonOption Title, Enable, Me.btnConference
#021            Case "Listen"
#022                SetButtonOption Title, Enable, Me.btnListen
#023            Case "Disconnect"
#024                SetButtonOption Title, Enable, Me.btnDisconnect
#025            Case "RopCall"
#026                SetButtonOption Title, Enable, Me.btnRopCall
#027            Case "Init"
```

```
#028                SetButtonInit
#029
#030 End Select
```

注意：上述代码中有一个特殊之处，就是：Name 参数传回“Init”，表示进入初始状态，此时应该把所有的按钮的状态都设置为禁用。→登出，或者和 CTI 通讯中断时，会触发此状态。

2.4.7 取得软电话状态描述

软电话状态变化时，会触发 EVTReturnStatusCH 事件，携带如下参数：

Status→状态的中文名称；字符串类型

可以处理 EVTReturnStatusCH 事件，并把状态中文显示在业务界面上。

2.4.8 电话进入

电话进入时，会触发 CallArrive 事件。CallArrive 事件携带以下参数：

ANI→主叫号码

DNIS→被叫号码

Data→保留不用。有关随路数据的传递见“随路数据一节”

业务系统可以处理 CallArrive 事件来进行弹屏处理。

2.4.9 应答处理

应答处理有几种情况：

- 普通呼叫进入；
- 被磋商呼叫进入；

针对普通呼叫进入和针对被磋商呼叫进入的处理不一样；普通呼叫进入，调用：CmdAnswer；被磋商呼叫进入，调用：CmdConsultAnswer。

辨别是哪种呼叫进入，可以根据按钮的标题来判断，如果是普通呼叫进入，则标题为“摘机”，如果是被磋商呼叫进入，则标题为“接受”。

以下是例子代码：

```
#001 If Left(Me.Hook.Caption, 2) = "摘机" Then
#002        Me.aOCX1.CmdAnswer
#003 ElseIf Me.Hook.Caption = "接受" Then
#004        Me.aOCX1.CmdConsultAnswer
#005 Else
#006            Me.aOCX1.CmdOnHook
#007 End If
```

注：上述代码判断 btnHook 的标题，如果不是“摘机”也不是“接受”，则必然是“挂机”，则调用挂机函数。

2.4.10 挂机处理

软电话挂机调用 CmdOnHook 即可。

例子代码见应答处理一节。

2.4.11 外拨处理

外拨处理提供两种模式：

1）“点击外拨按钮”→“弹出界面输入外拨号码”→“发起外拨”

2）“从界面输入或者选择外拨号码”→“发起外拨”

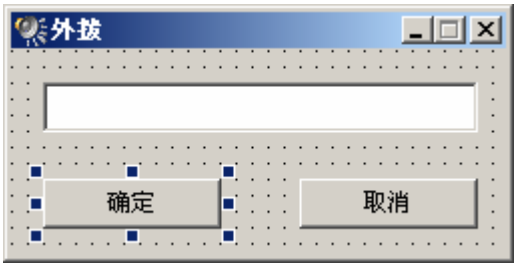
第一种模式下的步骤：

1）调用 AgentOCX 的 CmdDialOut 方法；

2）等待 AgentOCX 的 EVTDialOut 事件；该事件触发后，自己显示外拨界面，供填入被叫号码；

3）调用 AgentOCX 的 DialOut 方法；DialOut 方法携带一个参数：Dnis，为要拨打的电话号码；

外拨界面的一个例子如下：



图表 3 外拨界面

第二种模式下的步骤：

直接调用 AgentOCX 的 DialOut 方法；DialOut 方法携带一个参数：Dnis，为要拨打的电话号码；该模式一般用于在业务系统的客户信息列表中直接点选外拨，而不是先点击外拨按钮并输入外拨号码的时候。

2.4.12 保持处理

软电话保持调用 CmdHold 方法。

例子代码如下：

```
#001 If Me.Hold.Caption = "保持" Then
#002       Me.aOCX1.CmdHold
#003 Else
#004       Me.aOCX1.CmdHoldStop
#005 End If
```

注意，上述代码判断如果 btnHold 的标题为“保持”，则调用 CmdHold。否则，则调用 CmdHoldStop 来取消保持。

2.4.13 取消保持处理

取消保持调用 CmdHoldStop。例子代码见“保持处理”一节。

2.4.14 暂停处理

暂停处理调用 AgentOCX 的 CmdPause 方法。调用暂停之前，也可以让坐席选择暂停的原因。设置暂停的原因调用 SetAuxCode 方法。SetAuxCode 携带一个参数：AuxCode；AuxCode 为字符串类型，内容为“01”、“02”…、“09”、“10”。AuxCode 的定义在 AUX 表格中。在

2.4.15 单步转接处理

AgentOCX 的单步转接提供了两种模式：转接外线和转接坐席。

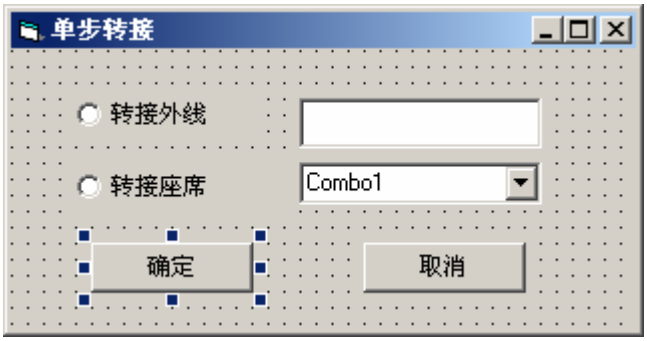
- 转接外线：这种方式下，软电话会把电话转接到一个电话号码上。（可以是携带出局号码的外部号码，也可以是分机号码）
- 转接坐席：这种方式下，软电话会从 CTIServer 取得目前空闲的坐席列表，并可以选择一个坐席进行单步转接。这种情况下，CTIServer 实际上根据被选择坐席工号定位到坐席的分机号码然后进行单步转接。

单步转接的发起过程：

- 发起单步转接：调用 CmdTransfer；
- 接收 CTIServer 返回的空闲坐席列表：处理 EVTTransfer 事件，EVTTransfer 事件携带一个参数 AgentList，AgentList 的结构为：AgentID=IsMonitor;AgentID1=IsMonitor1;….；其中 AgentID 为空闲坐席的工号，IsMonitor 为本坐席是否为班长席；例子如下：Agent1=1;Agent2=0;表示空闲坐席为 Agent1（班长席）,Agent2（非班长席）
- 调用 CmdTransferToAgent 进行转接。CmdTransferToAgent 携带 4 个参数：sAgentID（被转坐席工号），sAni（保留不用）, sDnis（保留不用）,Data（保留不用）。第一个参数 sAgentID 为被转接的空闲的工号（号码不以$开头表示是坐席工号）或者被叫号码（号码以$开头表示后面跟的是被转接的电话号码）。

2.4.15.1 业务系统自己绘制单步转接对话框

业务系统典型的处理方式为：点击 btnTransfer（转接按钮）后，调用 CmdTransfer；响应 EVTTransfer 事件，并弹出如下模式的转接选择界面：



图表 4 单步转接界面

弹出上述界面后，根据客户选择转接的模式进行转接。

注意：上述界面中，如果点击“取消”按钮，则应该调用 CmdTransferStop 方法取消转接的执行。

例子代码：

对于软电话 btnTransfer 的处理：

```
#001 If   Me.btnTransfer.Caption = "转接" Then
#002        Me.aOCX1.CmdTransfer
#003
#004 Else
#005        Me.aOCX1.CmdTransferStop
#006 End If
```

注意：上述代码中，如果 btnTransfer 的标题为“转接”，则调用 CmdTransfer；否则，其标题必然为“取消”，此时应该调用 CmdTransferStop。

对于单步转接的弹出界面的处理的例子：

点击确定按钮：

```
#001 If Option2.Value = True Then
#002        If (Combo1.Text = "") Then
#003                MsgBox "请选择正确的座席"
#004                Exit Sub
#005        End If
#006        frmMain.aOCX1.CmdTransferToAgent Combo1.Text, "", "", ""
#007        Unload Me
#008 Else
#009        If Len(Text1) = 0 Then
#010                MsgBox "请输入正确的转接电话号码"
#011                Exit Sub
#012        End If
#013        frmMain.aOCX1.CmdTransferToAgent "$" & Text1.Text, "", "", ""
#014        Unload Me
#015 End If
```

注意：上述代码中，如果选择空闲坐席，则调用 CmdTransferToAgent 方法，第一个参数为空闲坐席的工号；如果是转接外线，则调用 CmdTransferToAgent 方法，第一个参数为$+电话号码。

2.4.15.2 使用 OCX 自带的单步转接对话框

使用 OCX 自带的单步转接对话框，业务系统典型的处理方式为：点击 btnTransfer（转接按钮）后，调用 CmdTransfer；响应 EVTTransfer 事件，并调用 aOCX 的 ShowTransferDlg 方法。ShowTransferDlg 会弹出 AgentOCX 自带的对话框，该对话框已经对 CTIServer 返回的空闲坐席列表做了解析。

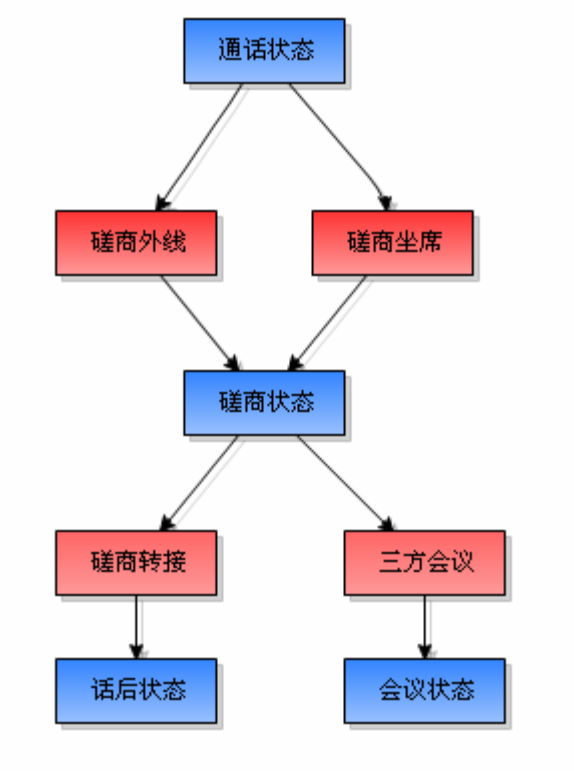
2.4.16 磋商处理

磋商的过程为：客户 1 与坐席通话过程中，坐席把客户 1 保持住，并与另外一方坐席或者外线号码（可以是携带出局号码的外部电话号码，也可以是内部分机号码）进行通话的过程。

AgentOCX 提供了两种磋商方式：

- 磋商外线：即把客户 1 保持住后，与第 3 方电话号码进行通话，第 3 方电话号码可以是携带出局号码的外部电话号码，也可以是内部的分机号码。
- 磋商内线：把客户 1 保持住后，与第 3 方坐席进行通话。

进入磋商状态后，可以选择磋商转接和进入三方会议。磋商状态变化图如下图所示：



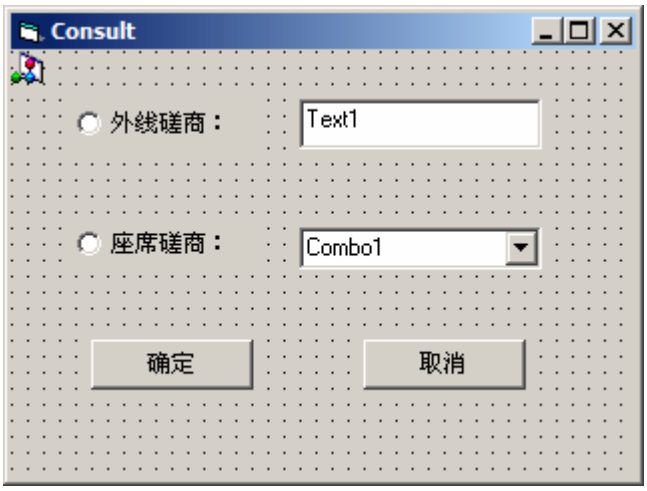
图表 5 磋商状态变化图

磋商发起过程：

- 发起磋商：调用 CmdConsult
- 接收 CTIServer 返回的空闲坐席列表：处理 EVTConsult 事件，EVTConsult 事件携带一个参数 AgentList，AgentList 的结构为：AgentID=IsMonitor;AgentID1=IsMonitor1;….；其中 AgentID 为空闲坐席的工号，IsMonitor 为本坐席是否为班长席；例子如下：Agent1=1;Agent2=0;表示空闲坐席为 Agent1（班长席）,Agent2（非班长席）
- 选择磋商坐席模式，则调用 CmdConsultToAgent 方法：CmdConsultToAgent 携带 3 个参数，第一个为被磋商的坐席工号，第 2 和第 3 个参数保留不用。
- 选择磋商外线模式，则调用 CmdConsultToOutLine 方法：CmdConsultToOutLine 方法携带 1 个参数，为被磋商的电话号码。

2.4.16.1 业务系统自己绘制磋商对话框

使用业务系统自己绘制磋商对话框，业务系统典型的处理方式为：点击 btnConsult（转接按钮）后，调用 CmdConsult；响应 EVTConsult 事件，并弹出如下模式的转接选择界面（由业务系统自己绘制）：



图表 6 磋商界面

上述界面中，如果点击取消，则应该调用 CmdConsultCancel 函数来取消磋商；

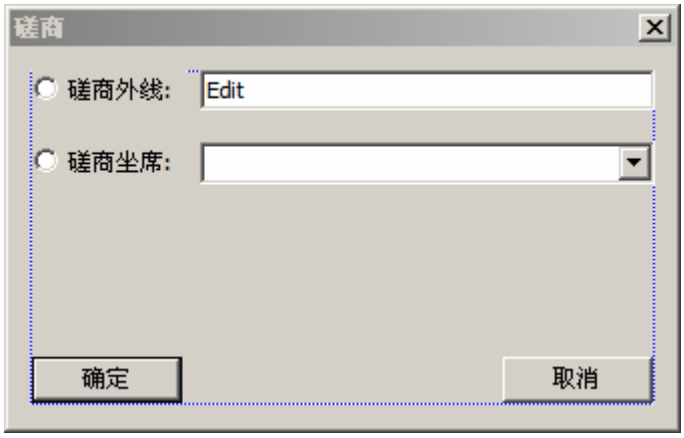
如果点击确定，则应该根据磋商的方式进行调用，如果是外线磋商，则调用 CmdConsultToOutLine，如果磋商坐席，则调用 CmdConsultToAgent。

点击确定后的典型代码如下：

```
#001 If Option1.Value = True Then
#002      If Trim(Text1.Text) = "" Then
#003           MsgBox "请输入正确的电话号码！"
#004           Exit Sub
#005      End If
#006      frmMain.aOCX1.CmdConsultToOutLine Me.Text1.Text
#007 Else
#008      If Trim(Combo1.Text) = "" Then
#009           MsgBox "请选择正确的座席！"
#010           Exit Sub
#011      End If
#012       frmMain.aOCX1.CmdConsultToAgent Me.Combo1.Text, "", ""
#013 End If
```

2.4.16.2 使用 OCX 自带的磋商对话框

使用 OCX 自带的磋商对话框，业务系统典型的处理方式为：点击 btnConsult（转接按钮）后，调用 CmdConsult；响应 EVTConsult 事件，并调用 ocx 的 ShowConsultDlg 方法。ShowConsultDlg 会调用 AgentOCX 自己绘制好的对话框，该对话框已经对 CTIServer 返回的空闲坐席列表作了解析，对话框如下图所示：



图表 7 调用 OCX 自己的磋商对话框

2.4.17 磋商转接处理

磋商转接处理就是两步转接，和单步转接不一样，磋商转接必须先进入磋商状态，才能发起。

进入磋商状态后，磋商转接的按钮是可用的，点击后，调用 OCX 的 CmdConsultTransfer 方法。CmdConsultTransfer 方法携带 4 个参数，都没有用处→给空值即可。

2.4.18 三方会议处理

三方会议必须在磋商状态才能发起。

进入磋商状态后，会议按钮是可用的，点击后，调用 OCX 的 CmdConference 即可。

2.4.19 发送传真处理

AgentOCX 发送传真的原理为：

4）AgentOCX 选择要发送的传真文件；支持 WORD、HTML、TEXT、EXCEL 等格式的文档；

5）AgentOCX 通过 ftp 把要发送的传真文件发送到 FaxServer 所在的服务器的 C:\FaxReceived 目录下；FaxServer 一般就是 IVRServer 所在的服务器，因为在呼叫中心大部分应用中，传真服务器和 IVR 服务器共用传真资源。

6）AgentOCX 把一条记录写入到 FaxServer 使用的数据库对应的 FAX_INFO_CONVERT 表格中。FAX_INFO_CONVERT 表格存储的是需要被转换并发送的传真文档信息。

7）FaxServer 会定期检测 FAX_INFO_CONVERT 表格，并把 FAX_INFO_CONVERT 表格中的传真转换记录进行处理，把 WORD、HTML、TEXT、EXCEL 等格式的文档转换成系统所支持的传真文档（Dialogic、三汇等支持 tif 传真文档，东进支持 bfx 传真文档）。

8）FaxServer 把转换好的 tif 或者 bfx 的信息写入到 FAX_INFO 表格中；

9）AAHost 的 ACS Server(自动外拨服务器)会定期检测 FAX_INFO 表格，并根据 FAX_INFO 表格的被叫号码进行自动外拨；（多条传真任务会进行批量外拨）

10） 外拨成功的传真任务会执行 ACS Server(自动外拨服务器)设置的自动传真流程（在 AAManager 中设置，如 auto_fax），该流程为流程编辑器编辑的自动发送传真的流程。ACS Server 启动自动发送传真的流程时，会把传真任务的 id 作为流程全局变量 sFaxID 传入到 auto_fax 流程中。

11） 自动发送传真的流程会根据 sFaxID 到 Fax_INFO 表中定位该传真任务对应的传真文件，并在流程中进行发送。

12） ACS Server 会根据传真发送的结果填写 FAX_INFO_HIS 传真发送历史记录。

13） 如果传真外拨失败，ACS Server 会自动定期尝试外拨。如果外拨失败达到指定次数，如 3 次，则 ACS Server 会把失败结果写入 FAX_INFO_HIS 传真发送历史记录。

如上述过程中，可见对于 AgentOCX 来说，发送传真就是一个上传的过程。其他的转换、外拨、发送、错误处理都由系统来完成。因此，对于 AgentOCX 来说，发送传真调用的函数很简单，如下所示：

aOCX.CmdFax

即调用 CmdFax 即可。CmdFax 会弹出上传对话框，并要求输入被发送的电话号码和传真文件。

注：有关传真的处理还包括自动传真接收，自动传真发送等。有关传真的具体细节，请见流程编辑器的帮助文档。

2.4.20 接收传真处理

AgentOCX 的接收传真处理原理为：

- 客户通过传真机打入 IVR 系统，IVR 流程执行到“接收传真节点”时，客户把传真文件发送到“接收传真节点”指定的服务器的路径上。
- IVR 流程在接收完毕传真后，在接收传真节点的操作中把接收的传真信息写入到 FaxLog 表中。其中 FaxLog 的 FLAG 字段为 2 表示是未处理的传真呼叫。
- CTIServer 循环检测 FaxLog 表格，看是否有未处理的传真呼叫。如果发现有新的传真呼叫，则找出空闲的符合条件的坐席，并发送新传真呼叫信息到所选择的坐席；选择了坐席后的未处理的传真呼叫的 FLAG 字段会设置为 8，CustomerID 字段会更新为处理坐席的坐席工号。
- AgentOCX 收到新传真呼叫信息后，触发 EVTNewFaxRecordArrive 事件。
- 业务系统收到 EVTNewFaxRecordArrive 后，从 FaxLog 表格中查找 CustomerID 为本坐席工号，并且 FLAG=8 的记录，列出来； 然后业务系统选择一条要处理的传真任务，调用 AgentOCX 的 DownloadAndShowFax 即可下载处理传真。调用 AgentOCX 的 DownloadAndShowFax 方法后，业务系统应该把相关的传真任务的 FLAG 参数改为 1，表示该传真已经被处理。

例子代码：

下面是查询属于本座席的新传真任务的例子代码(vb)：

```
On Error Resume Next
    '从数据库中取得未听 IVR 留言纪录
    grdEdit.Clear False
```

```
Dim cn As Object
Set cn = CreateObject("ADODB.Connection")
cn.ConnectionString = ConnectionString
cn.Open

Dim sql As String



'RecordLog 表格，如果 FLAG=8，表示未被处理的 Fax 留言


sql = "select seq,CallerNum,F_PATH from FaxLog where FLAG=8 and CustomerID = '" & gAgentID & "'"

Dim rs
Set rs = cn.Execute(sql)

If rs.EOF = True Then
    Exit Sub
End If

rs.MoveFirst

While rs.EOF = False
    With grdEdit
        .AddRow
        .CellText(.Rows, 1) = rs(0)
        .CellIcon(.Rows, 1) = 0
        .CellText(.Rows, 2) = rs(1)
        .CellText(.Rows, 3) = rs(2)



    End With
    rs.MoveNext
Wend
```

以下例子代码是选择了一条新传真纪录后的处理过程：

```
CurIndex = grdEdit.CellText(grdEdit.SelectedRow, 1)
faxFile = grdEdit.CellText(grdEdit.SelectedRow, 3)
aOCX1. DownloadAndShowFax grdEdit.CellText(grdEdit.SelectedRow, 3)

Dim sql As String
```

```
sql = "update faxlog set Flag = 1,ReadTime = GETDATE() where seq = " & CurIndex


Dim cn As Object
Set cn = CreateObject("ADODB.Connection")
cn.ConnectionString = ConnectionString
cn.Open

cn.Execute sql
```

2.4.21 远程桌面监控处理

AgentOCX 提供了远程桌面监控的功能。具有班长权限的坐席可以选择当前登录在系统的坐席的列表，并选择一个坐席进行远程桌面的监控。远程桌面的监控只能看到对方的桌面，无法对其进行操作。远程桌面监控无需第 3 方软件的支持。

注意：要使用远程桌面监控，必须开启 AAManager 的 CTIServer 配置的远程桌面监控的选择项。

要发起远程桌面监控，则调用 AgentOCX 的 QueryAgentAddresses 方法既可。

QueryAgentAddresses 的处理过程如下：

- AgentOCX 发送 Query 命令到 CTIServer，CTIServer 返回当前登录的坐席的列表给 AgentOCX；
- AgentOCX 显示当前登录坐席列表的界面；
- 业务系统选择要监控的坐席，发起监控；
- AgentOCX 会显示监控的界面，并进行监控连接。

远程监控的界面如下所示：

图表 8 选择要远程监控的座席

图表 9 座席远程桌面监控

2.4.22 随路数据处理

当呼叫从 IVR 转接到坐席，或者是从坐席转接到 IVR 时，或者坐席之间互相转接时，都需要传递随路数据。这里的随路数据不是主叫号码和被叫号码等基本数据，而是应用中更加灵活的数据。比如，客户在 IVR 流程中输入的帐号和密码信息，转接到坐席后，坐席不需要再次询问客户这些信息，就能够在坐席软件中显示出来，这些就是随路数据。

CTXWare 提供了一对函数来对随路数据进行存取：

DoSetAssociatedData 和 DoGetAssociatedData

- DoSetAssociatedData 是设置随路数据(可以在 IVR 流程编辑器中调用，也可以在坐席端调用)。该函数接收两个参数，第一个参数为随路数据的名称（自己定义），第二个参数为随路数据的内容。两个参数都是字符串类型。
- DoGetAssociatedData 是取得随路数据(可以在 IVR 流程编辑器中调用，也可以在坐席端调用)。该函数具有一个参数，为随路数据的名称。该函数返回随路数据对应的内容。参数和返回数值都是字符串类型。

注意：DoSetAssociatedData 可以多次调用，系统中可以存储最多 20 个随路数据定义。如果两次调用的随路数据名称一样，则最后调用的将覆盖前面调用的。一旦电话挂断，随路数据将全部清除。

例子 1：

在 IVR 中，希望将用户名称和用户性别传送给坐席。则在节点操作中，写如下指令：

```
DoSetAssociatedData “Name”,”张三”

DoSetAssociatedData “Sex”,”男”
```

呼叫转接到坐席后，坐席通过调用 AgentOCX 的函数 DoGetAssociatedData 即可取出数据：

```
sName = ocx1.DoGetAssociatedData(“Name”)  →返回“ 张三”

sSex = ocx1.DoGetAssociatedData(“Sex”) →返回“ 男”
```

例子 2：

坐席需要将呼叫转回 IVR 中，并利用 IVR 播放一个实现录好的语音文件。则坐席端调用如下：

```
ocx1.DoSetAssociatedData “VoiceName”,”c:\hello.vox”
```

呼叫转接到 IVR 后，IVR 流程通过以下操作取出随路数据：

```
sVoiceName = DoGetAssociatedData(“VoiceName”)
```

例子 3：

坐席将呼叫转接到另外的坐席，则在发起转接的坐席端，调用：

```
ocx1.DoSetAssociatedData “Name”,”张三”
```

在被转接的坐席端，调用：

```
sName = ocx1.DoGetAssociatedData(“Name”)  →返回“ 张三”
```

2.4.23 转自动流程处理

呼叫可以从坐席处理转回到自动流程。转自动流程处理一般有如下典型应用模式：

- 让客户收听已经录制好的语音。比如系统中录制了大量的政策法规语音，坐席可以把呼叫转到 IVR，并传递随路数据到 IVR，让 IVR 播放预先录制好的语音给客户；
- 让客户输入用户名称和密码，由 IVR 系统对用户名称和密码进行检验。这样，可以不由坐席人工参与对用户名和密码进行检验；

座席转自动流程处理的方法为：cmdAuto2；

cmdAuto2 的携带 2 个参数。

- 第一个参数 FlowName：为要转的目标流程的名称；注意，第一个参数为目标流程在系统 IVR 工程中的名称，而不是目标流程的路径。比如，目标流程的路径为：c:\toauto.sbs，而目标流程在工程中的名称为 auto，则 cmdAuto2 方法的第一个参数应该是“auto”。

- 第二个参数指定转自动流程的方式：”BLINDTRANSFER”表示转接模式，其他表示会议。转接模式是指，调用 cmdAuto2 后，客户直接和 IVR 进行通话，坐席退出通话；会议模式是指，调用 cmdAuto2 后，客户、IVR 和座席进行三方通话；此时在 ivr 流程中，坐席可以对客户进行指导，在自动流程退出后，同一坐席可以继续服务该客户。

转自动之前，坐席端可以通过 DoSetAssociatedData 来指定随路数据；而目标流程可以根据随路数据选择播放哪些语音，或者是跳转哪些分支，这样可以实现由坐席端指定转到自动流程的哪个节点分支。

2.4.24 自动话后结束处理

缺省情况下，在坐席通话结束后，坐席会自动进入话后处理状态。但是某些情况下，希望坐席通话结束后，能够自动回到空闲状态。

自动话后结束处理过程如下：

1）响应 AgentOCX 的 EVTWrapUp 事件；

2）在 EVTWrapUp 事件的处理函数中调用 AgentOCX 的 WrapEnd 方法。

2.4.25 IVR 语音信箱留言查听处理

IVR 提供了语音信箱（自动语音留言）的功能。语音留言的处理过程如下：

1） 客户在 IVR 流程中进入到语音留言节点（就是录音节点），开始留言录音。

2） 录音完毕后，IVR 流程在操作中把留言文件的信息、客户信息等写入到 RecordLog 表格。并把 FLAG 字段设置为 2，表示新的留言纪录。

3） CTIServer 循环查找 RecordLog 表格的相关纪录的 FLAG=2 的纪录，如果发现，则查找一个合适的坐席，并发送 NEWIVRRECORD 事件到该座席。同时，把 RecordLog 表格的相关纪录的 FLAG 设置为 8，CustomerID 设置为所选座席的工号。

4） AgentOCX 收到 CTIServer 的 NEWIVRRECORD 通知后，触发 EVTNewIVRRecordArrive 事件。

5） 业务系统收到 EVTNewIVRRecordArrive 事件后，从 RecordLog 表格查询 FLAG=8 并且 CustomerID=本坐席工号的纪录，列出来。

6） 新 IVR 留言的查听是通过转自动的方式通过坐席电话实现的。通过随路数据设置了要听的留言的语音文件名，然后调用 CmdToAuto2 转到事先做好的自动流程（如 ListenIVRRecord）。

例子代码：

查询新留言的例子代码：

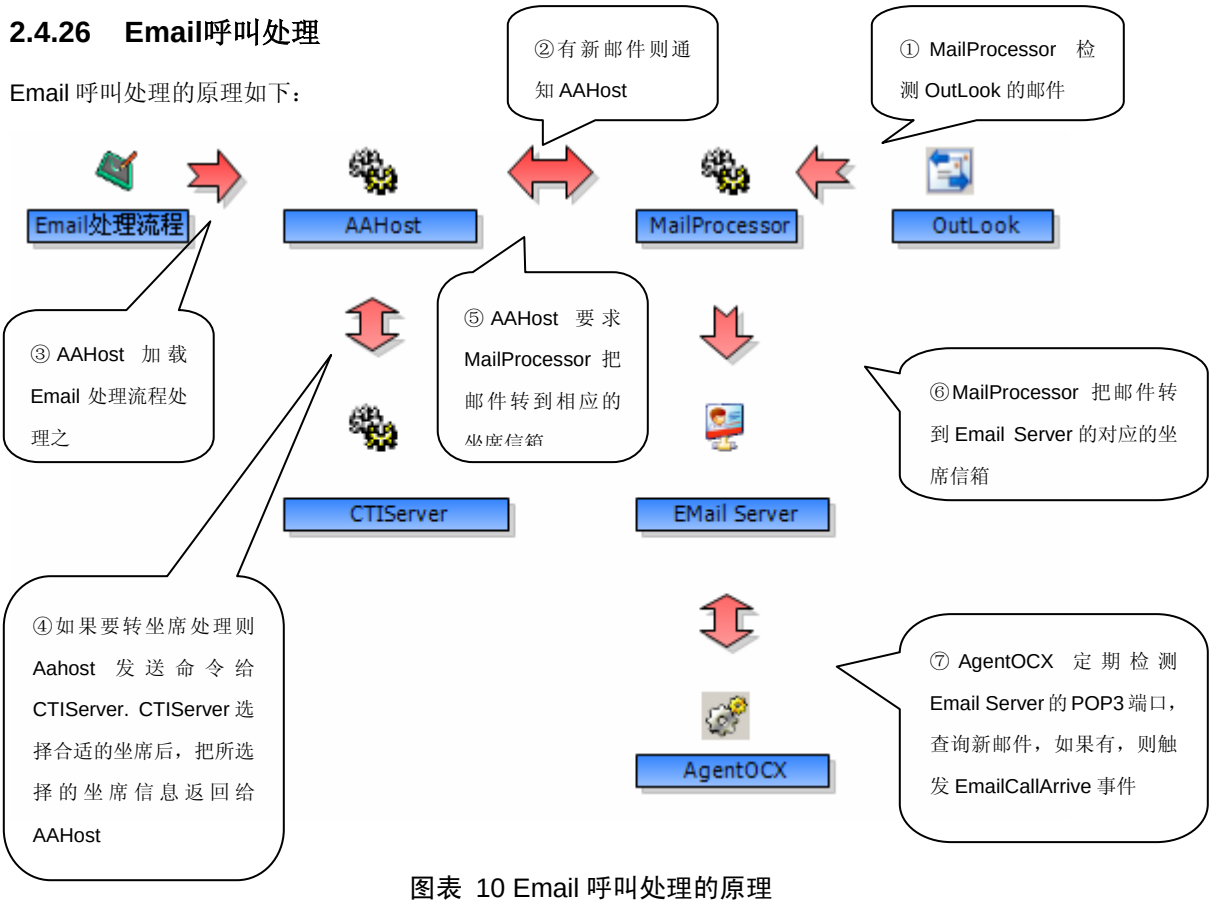
```
grdEdit.Clear False

    Dim cn As Object
    Set cn = CreateObject("ADODB.Connection")
    cn.ConnectionString = ConnectionString
    cn.Open

    Dim sql As String

    'RecordLog 表格，如果 FLAG=2，表示未被处理的 IVR 留言

    sql = "select seq,CallerNum,CalleeNum,StartTime,EndTime,DATEDIFF(s,StartTime,EndTime),F_PATH from RecordLog where FLAG=8 and CustomerID = '" & gAgentID & "'"

    Dim rs
    Set rs = cn.Execute(sql)

    If rs.EOF = True Then
        Exit Sub
    End If

    rs.MoveFirst

    While rs.EOF = False
        With grdEdit
            .AddRow
            .CellText(.Rows, 1) = rs(0)
            .CellIcon(.Rows, 1) = 0
            .CellText(.Rows, 2) = rs(1)
            .CellText(.Rows, 3) = rs(2)
            .CellText(.Rows, 4) = rs(3)
            .CellText(.Rows, 5) = rs(4)
            .CellText(.Rows, 6) = rs(5)
            .CellText(.Rows, 7) = rs(6)

        End With
        rs.MoveNext
    Wend
```

留言查听的代码：

```
CurANI = grdEdit.CellText(grdEdit.SelectedRow, 2)
CurDNIS = grdEdit.CellText(grdEdit.SelectedRow, 3)
sFileName = grdEdit.CellText(grdEdit.SelectedRow, 7)
CurIndex = grdEdit.CellText(grdEdit.SelectedRow, 1)
aOCX1.DoSetAssociatedData "AgentID", gAgentID
aOCX1.DoSetAssociatedData "DeviceName", m_DeviceName
aOCX1.DoSetAssociatedData "RECORDFILE", grdEdit.CellText(grdEdit.SelectedRow, 7)
aOCX1.cmdAuto2 "ListenIVRRecord", ""
```

2.4.26 Email呼叫处理

Email 呼叫处理的原理如下：



图表 10 Email 呼叫处理的原理

Email 呼叫处理的原理如上图所示。描述如下：

1） CTXWare MailProcessor 是一个 Windows 服务，用来定期地检测 OutLook 的邮件。因此需要在 Email 服务器上配置 OutLook 邮件账号，账号就是呼叫中心对外公布的 Email Address。Email Address 可以是一个或者多个地址。

2） CTXWare MailProcessor 如果检测到新的邮件，则通知 AAHost 有新的 Email 呼叫。

3） AAHost 会加载 Email 处理流程，Email 处理流程会对 Email 的内容进行分析处理；对于要自动回复的邮件，Email 处理流程可以自动回复。对于需要坐席人工处理的邮件，会执行**电子邮件转坐席节点**。

4） **电子邮件转坐席节点**会发送转坐席命令给 CTIServer，可以指定要转的坐席的技能组。CTIServer 收到转接命令后，进行 ACD 统一处理，选择合适的坐席后，会返回 AAHost 转接结果。

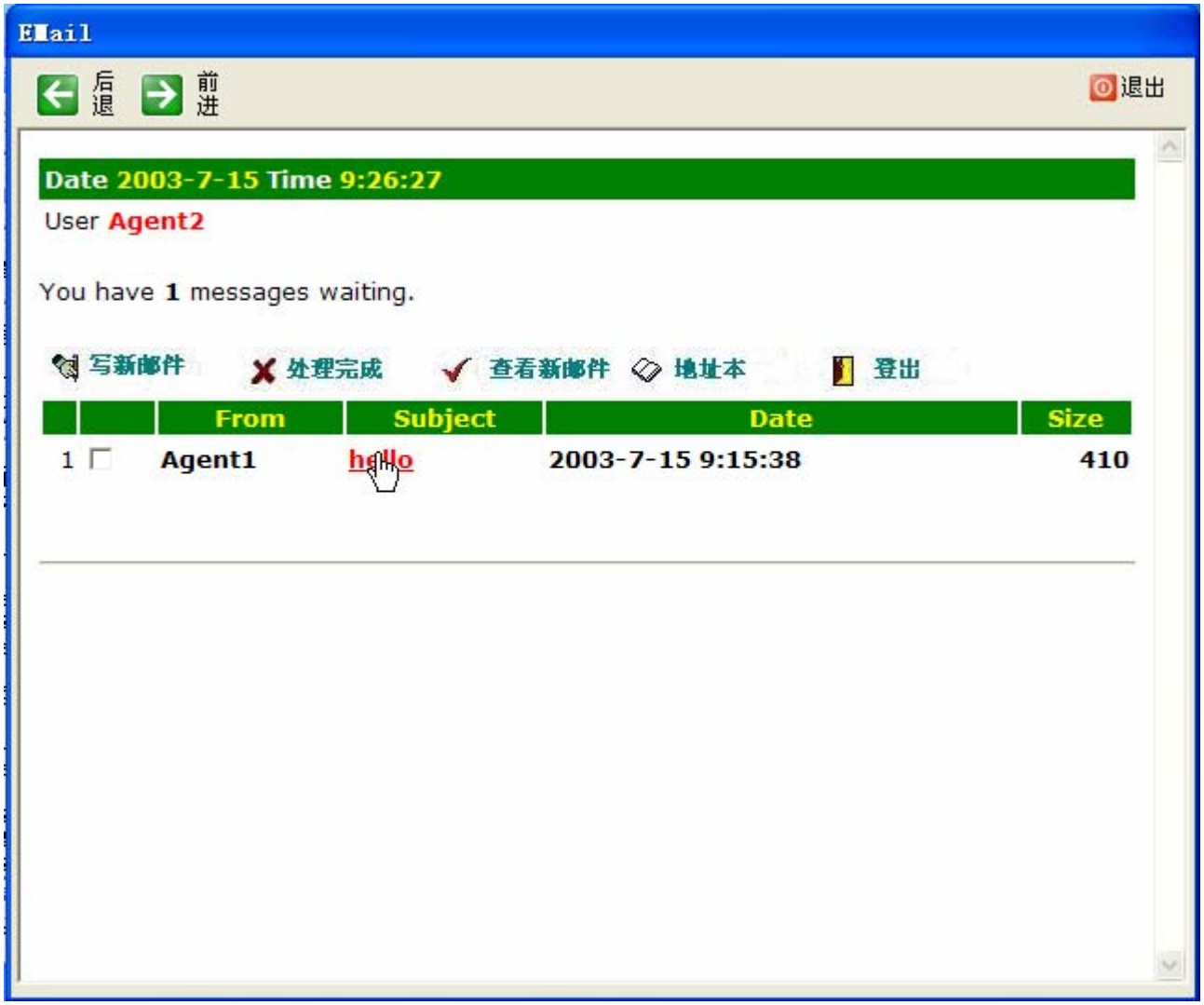
5） AAHost 收到 CTIServer 返回的转接结果信息后，会给 MailProcessor 发送指令，要求 MailProcessor 把该邮件转到该座席对应的邮箱。

6） MailProcessor 收到 AAHost 的转坐席邮箱的命令后，操作 CTXWare 内置的 Email Server，把新邮件转接到 Email Server 对应的坐席的邮箱。注意：MailProcessor 会自动为每一个坐席在 Email Server 中创建新的邮箱。

7） AgentOCX 定期检测 Email Server 的 Pop3 端口，检测到属于自己的新邮件后，会触发 EmailCallArrive 事件。

8） 业务系统检测到 EmailCallArrive 事件后，可以调用 AgentOCX 的 GoMail 方法进入自己的邮箱进行处理。

Email 的处理界面如下图所示：



图表 11 EMail 处理界面

座席在处理 Email 时，可以直接把无法处理的邮件转发到其他坐席。在新建邮件过程中，收件人地址写上对方坐席的工号，即可把邮件转发到该座席。

注意，如果要使用 Email 功能，必须在坐席的配置中设置 Email Server 的 Pop3 地址和端口。CTXWare 内置的 Email Server 的 Pop3 端口为 111。具体配置细节请参考“坐席配置”一节。

坐席也可以列出当前未处理的 Email 的标题和发件人信息。可以调用 AgentOCX 的 CmdGetMailCount 取得当前未处理邮件的数目。并可以循环调用 CmdGetMailDetails(n)取得第 n 个邮件的具体信息。注意：n 是 0 开头的索引。

下面是列出当前未处理的 Email 的标题和发件人信息的例子代码：

```
Dim i As Integer
TotalCount = aOCX1.CmdGetMailCount

For i = 0 To TotalCount - 1
        Dim retval As String
        retval = aOCX1.CmdGetMailDetails(i)

        If retval <> "INVALID_XML" Then

            Dim From As String
            Dim Attach As String
            Dim Subject As String
            Dim sDate As String
            Dim sTo As String
            Dim sBody As String

            '解析出字段
            Dim xNodeTmp As Object
            Dim xDocTmp As MSXML.DOMDocument
            Set xDocTmp = CreateObject("MSXML.DOMDocument")
            xDocTmp.createProcessingInstruction "xml", " version='1.0'"

            xDocTmp.loadXML retval
            Set xNodeTmp = xDocTmp.childNodes(0)

            From = xNodeTmp.Attributes.getNamedItem("FROM").nodeValue
            Attach = xNodeTmp.Attributes.getNamedItem("ATTACHS").nodeValue
            Subject = xNodeTmp.Attributes.getNamedItem("SUBJECT").nodeValue
            sDate = xNodeTmp.Attributes.getNamedItem("DATE").nodeValue
            sTo = xNodeTmp.Attributes.getNamedItem("TO").nodeValue
            sBody = xNodeTmp.Attributes.getNamedItem("BODY").nodeValue

            With grdEdit
                .AddRow
                .CellText(.Rows, 1) = i + 1
                .CellIcon(.Rows, 1) = 14

                .CellText(.Rows, 2) = From
                .CellText(.Rows, 3) = sTo
                .CellText(.Rows, 4) = Subject
                .CellText(.Rows, 5) = Attach
                .CellText(.Rows, 6) = sBody
```

```
                .AutoHeightRow (.Rows)

            End With

        End If
    Next i
```

由于 CTXWare 内置的 Email Server 支持 Pop3 和 www 接口，所以也可以通过一下的 url 进入到某坐席的邮箱：

http://gAgentID:gAgentID@192.168.100.1:12345/msg1

其中 gAgentID 为所需要查看的 AgentID，192.168.100.1 为 Email Server 的 ip,12345 为 Email Server 的 www 接口的缺省端口。Msgn 为取得第 n 条新邮件。

下面的 url 用来删除第 1 条邮件。

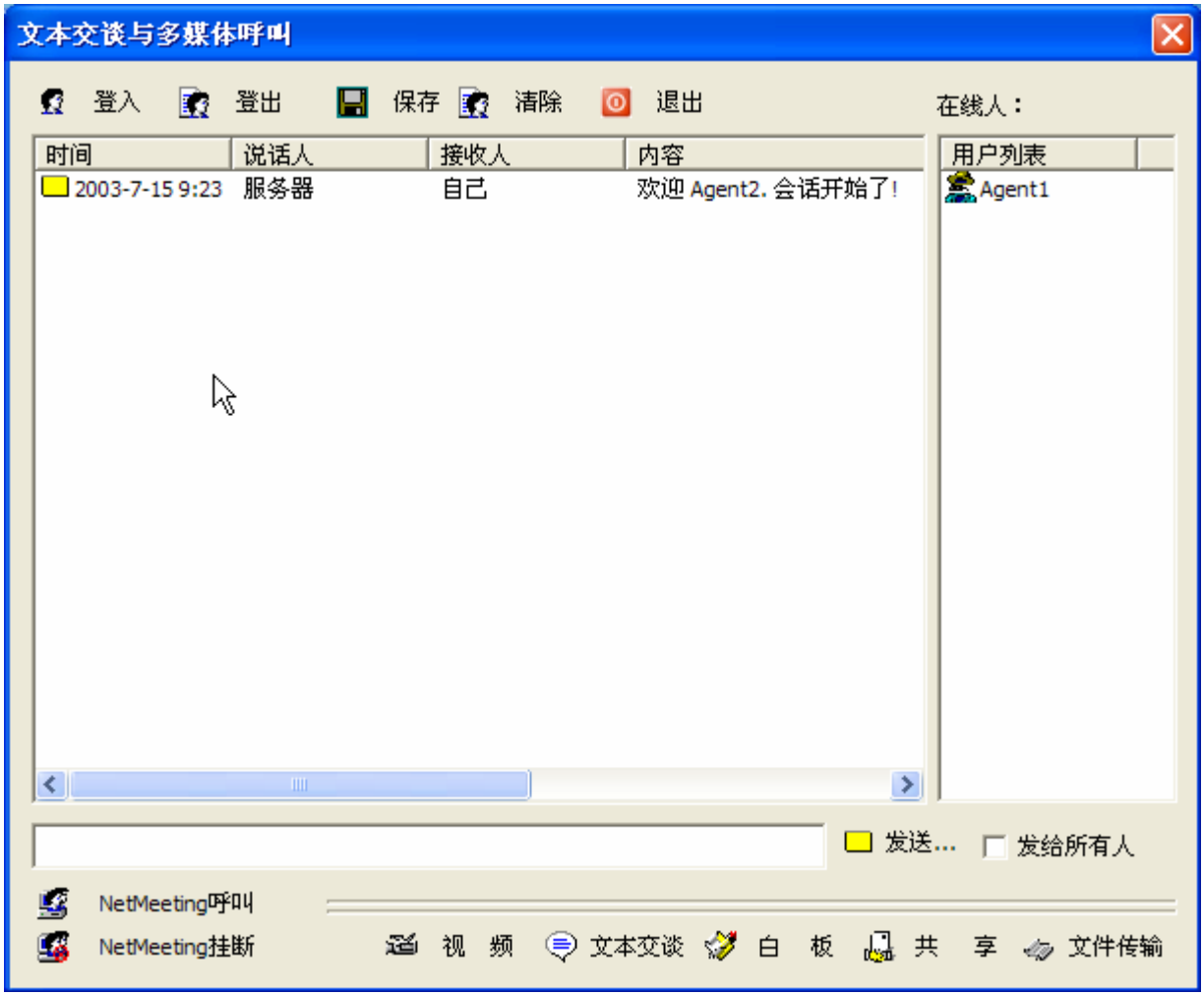
http://gAgentID:gAgentID@192.168.100.1:12345/delete?msg1=on

2.4.27 TextChat 以及 WebCall 呼叫处理

当坐席登录到 CTXWare 的 TextChatServer 时，会触发 TextChatLoginSucc 事件。此时可以通过调用 AgentOCX 的 ShowTextChatDlg 来进行文本交谈。

当坐席失去和 CTXWare 的 TextChatServer 的连接时，会触发 TextChatLogOuted 事件。

TextChat 和 WebCall 的界面如下：



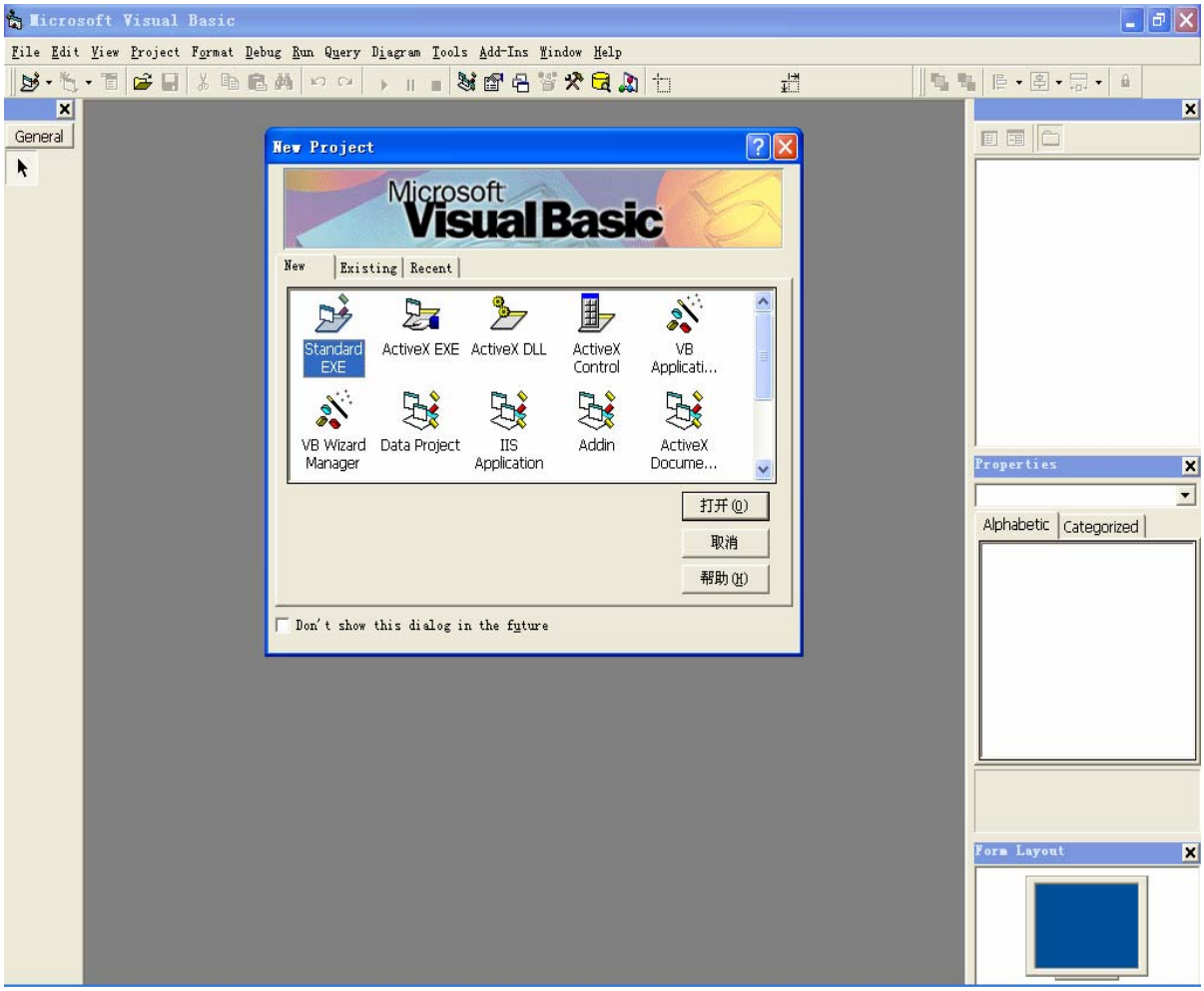
图表 12 TextChat 和 WebCall 的界面

3. 基于 VB 的 DEMO 程序的搭建

3.1 程序搭建过程

3.1.1 创建工程

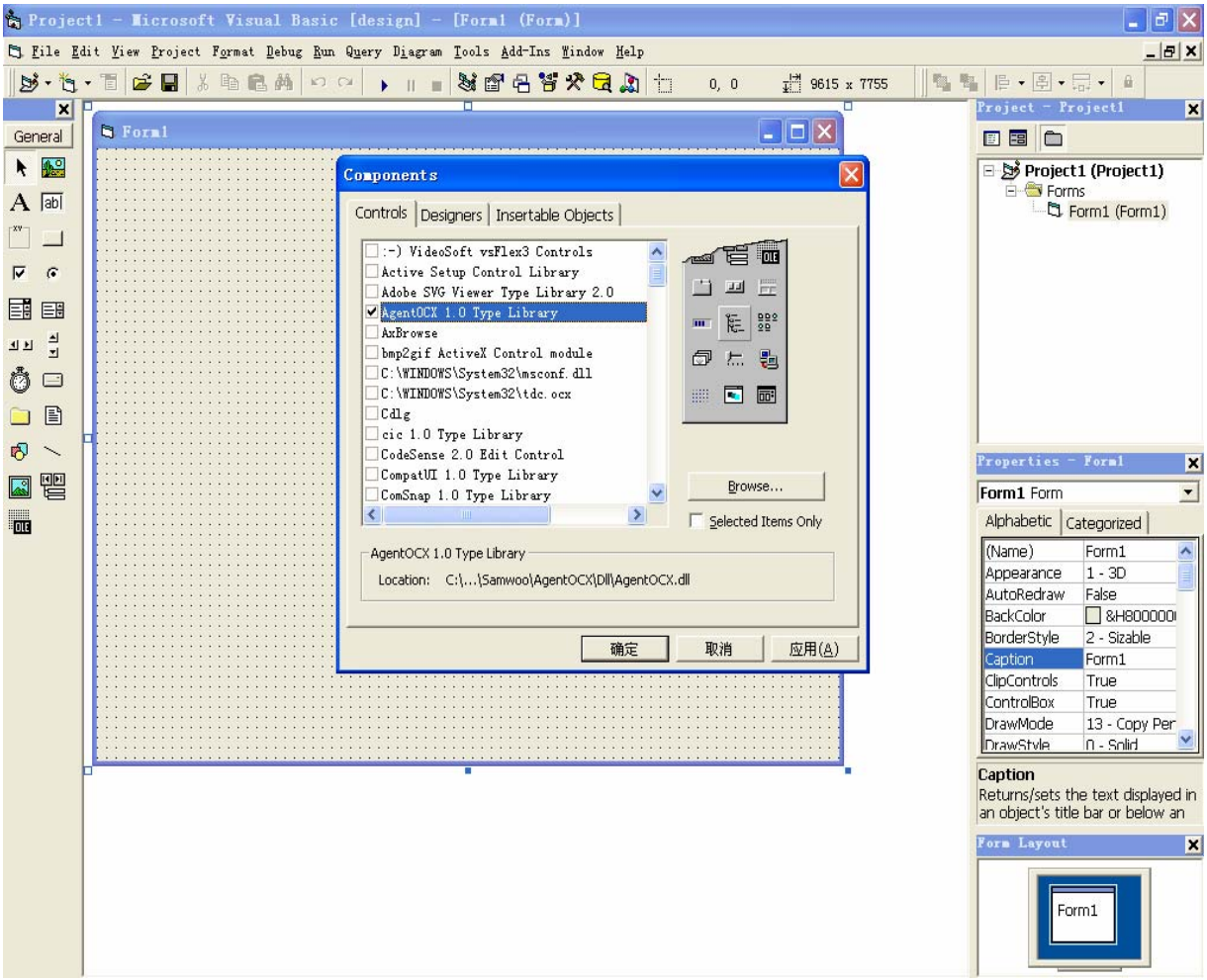
首先要将所提供的 AgentOCX.dll 注册，然后创建 VB 工程，选择 Standard.exe 模式



图表 13 vb 程序搭建选择 Standard Exe

3.1.2 在工程中添加 OCX 控件

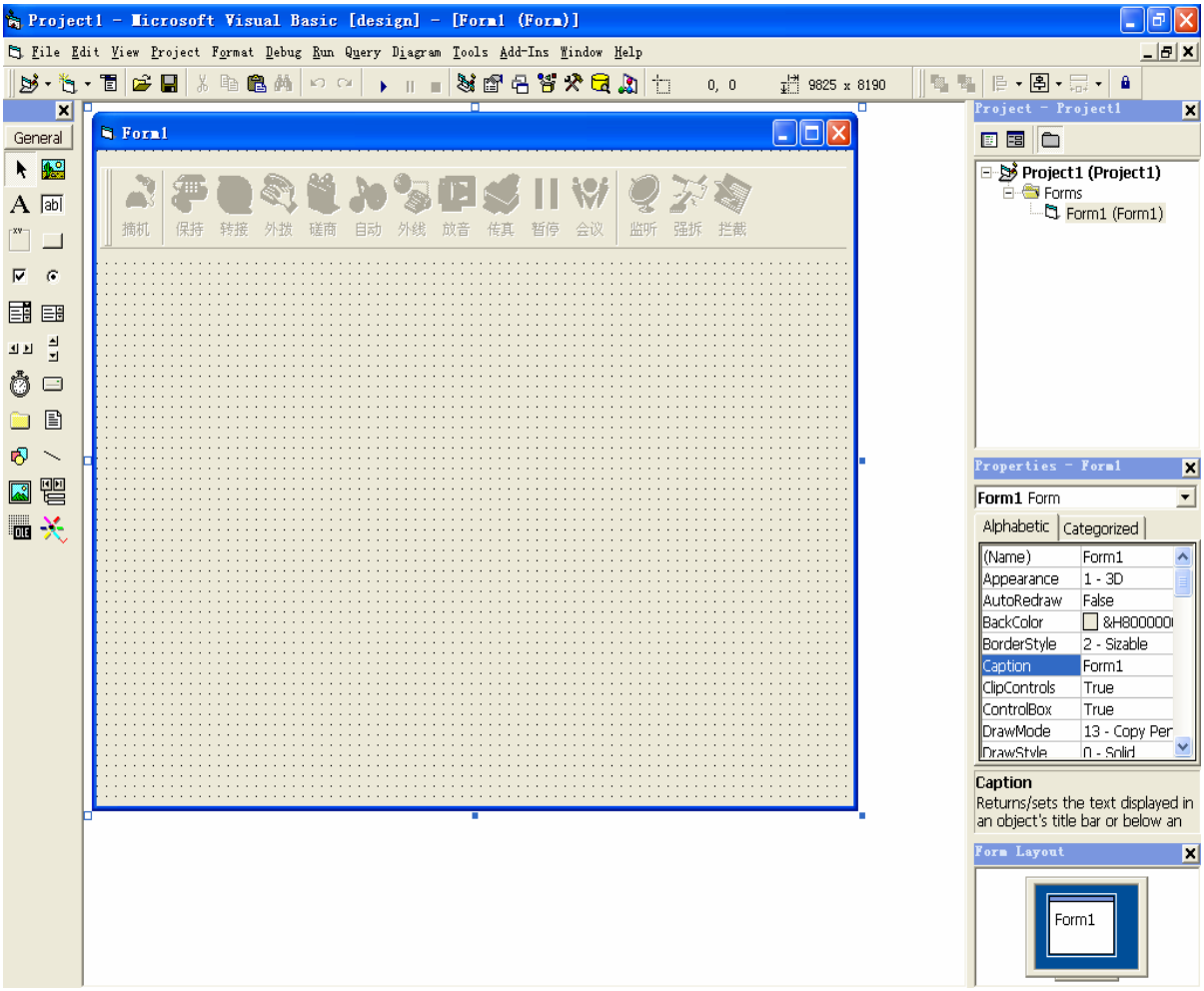
在 VB 工程的 Components 选项卡中选中 AgentOCX 1.0 Type Library 项，在 VB 工程左侧 ToolBox 栏会显示 OCX 控件的图标:



图表 14 选择 AgentOCX Component

3.1.3 在界面上添加 OCX 控件

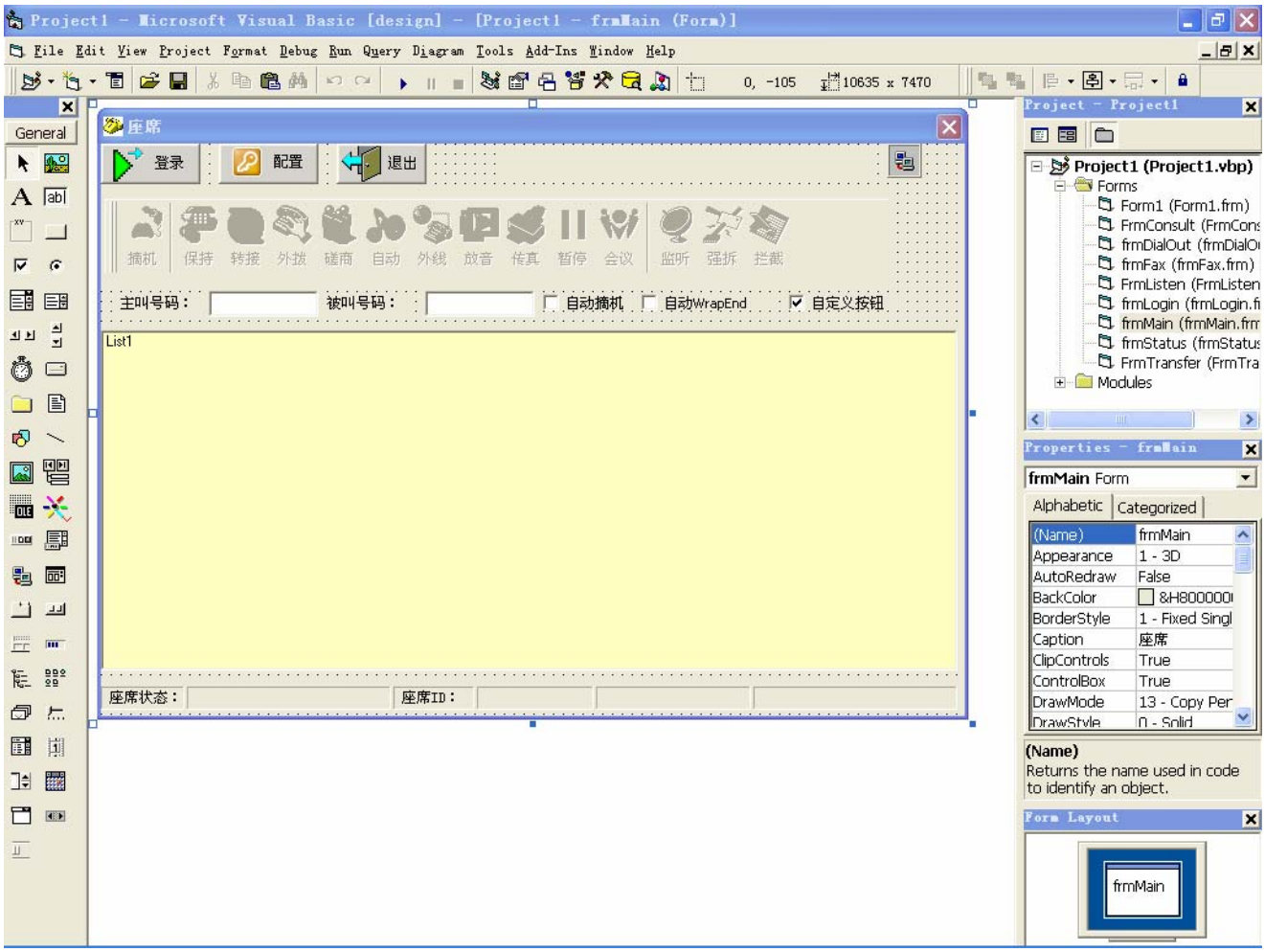
选中 AgentOCX 的图标，在 VB 工程的 Form1 上画出控件的外形，为了方便整个控件的显示控制，也可将它放置在 Picturebo 控件上。如图：



图表 15 把 AgentOCX 贴在界面上

3.1.4 完成主界面

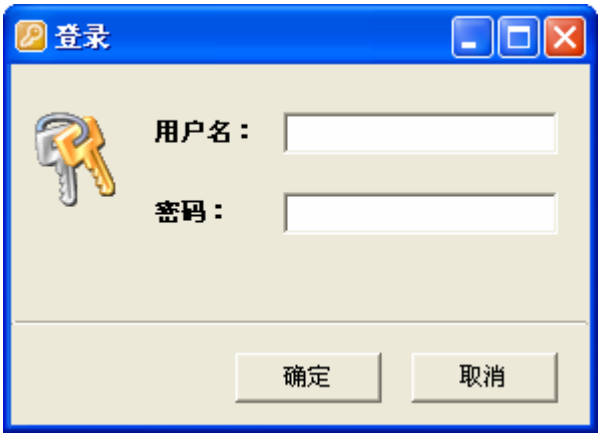
在界面上添加显示主叫和被叫的文本框及登录、注销按钮以及显示座席实时状态的状态条



图表 16 完成主界面

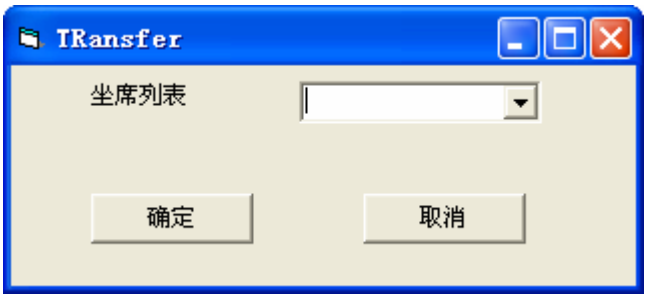
3.1.5 完成相关界面

分别作出登陆、外拨（外线、会议）、传真、转接、监听用的 Form

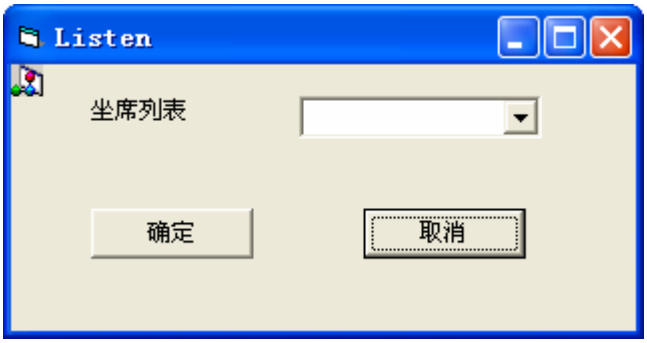


图表 17 登录对话框

图表 18 外拨对话框



图表 19 转接对话框



图表 20 监听对话框

3.1.6 Login 的代码实现

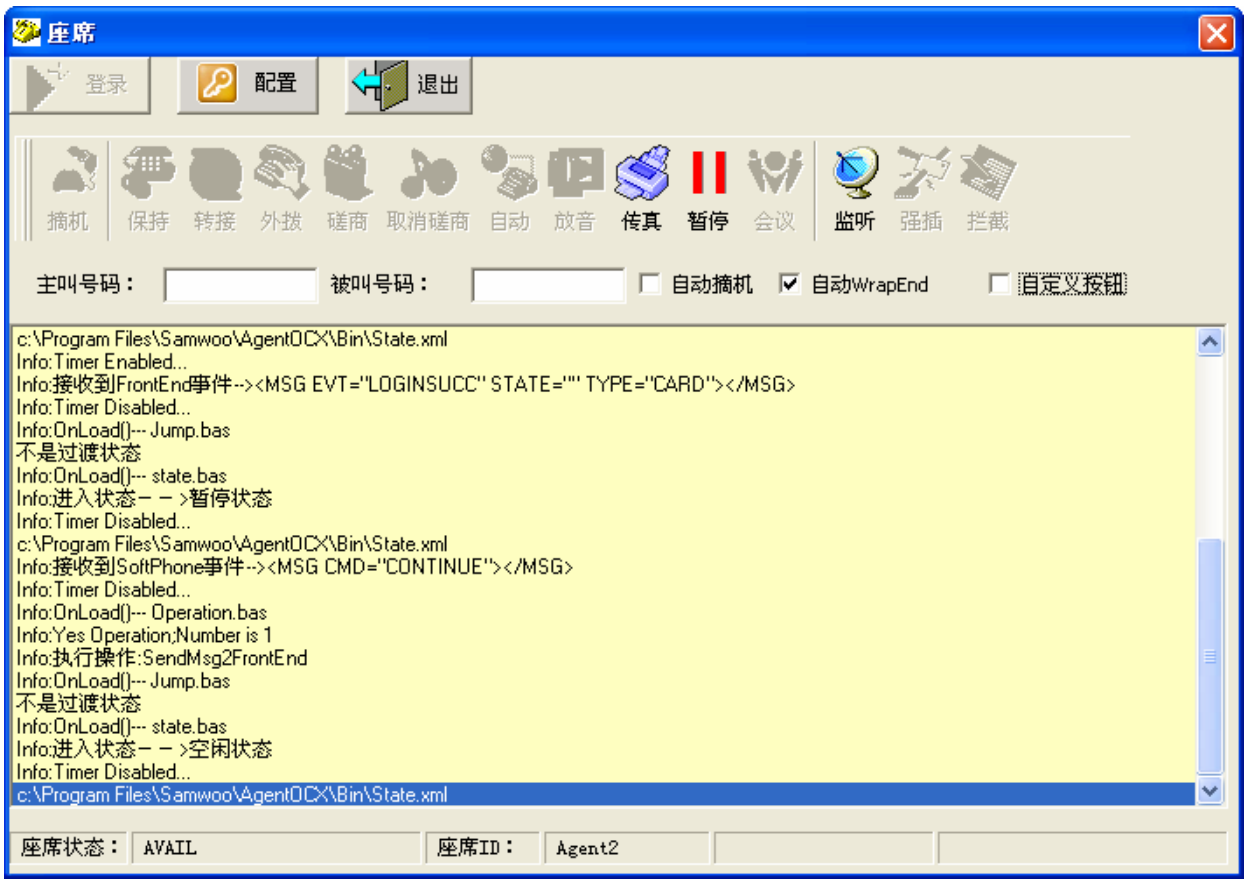
界面的工作完成之后就可以进入编码的阶段了，在图表 16 所示的登录界面中，需要提供登录的 AgentName 和 Password，调用 OCX 控件的 Login 函数即可。代码如下：

```
Private Sub cmdCancel_Click()
Unload Me
End Sub

Private Sub cmdConfirm_Click()
```

```
        frmMain.aOCX1.AgentID = Trim(Me.txtUserName.Text)
    frmMain.aOCX1.Password = Trim(Me.txtPassword.Text)
    frmMain.aOCX1.LogIn
    frmMain.StatusBar1.Panels(4).Text = Trim(Me.txtUserName.Text)
    Unload Me
    End Sub
```

3.1.7 暂停处理的实现

登录成功后的座席界面如图所示：

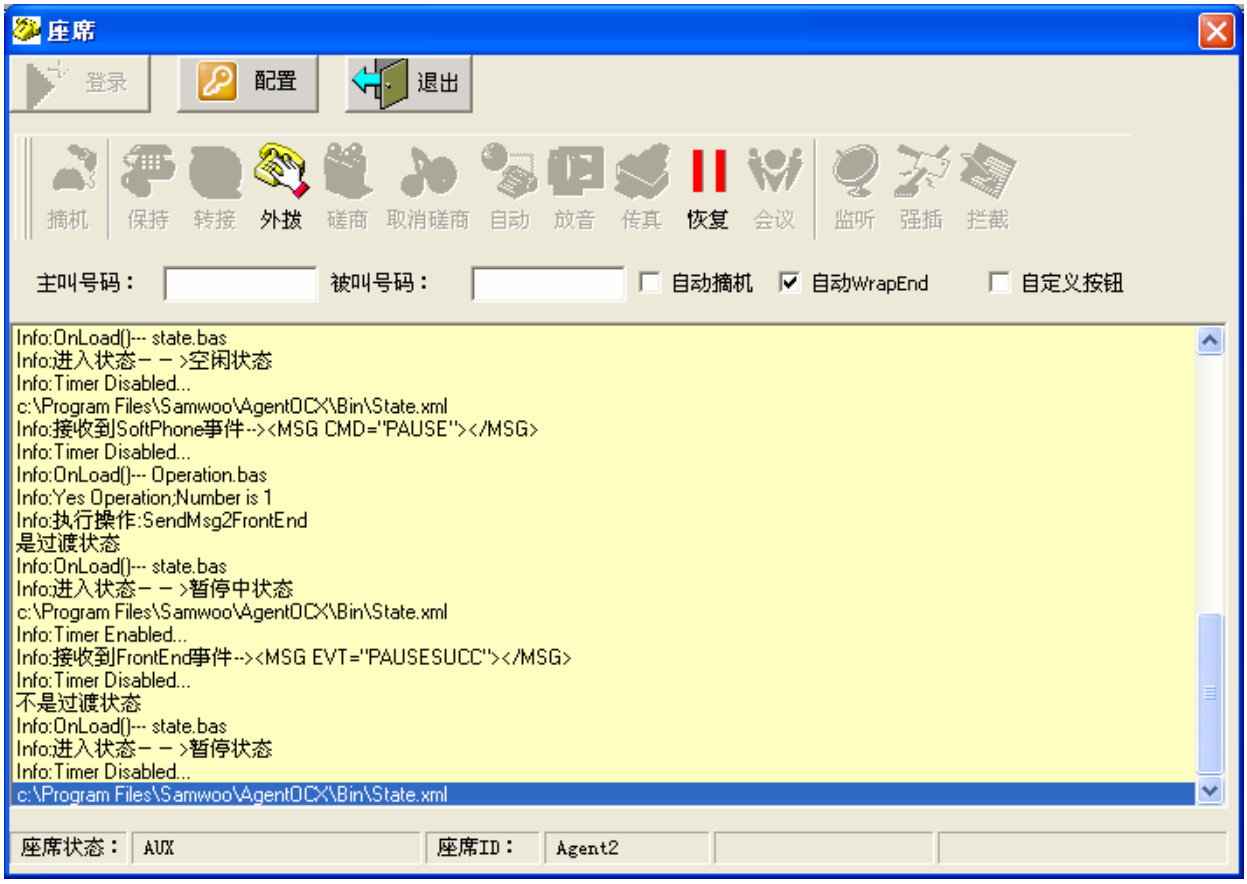


图表 21 登录成功主界面

座席登录成功后处于空闲状态，可以进行软电话的相关操作，如：传真、监听。

底下的状态条显示了当前的座席状态及登录座席的 ID 信息。

当座席点击暂停时，座席进入暂停状态：

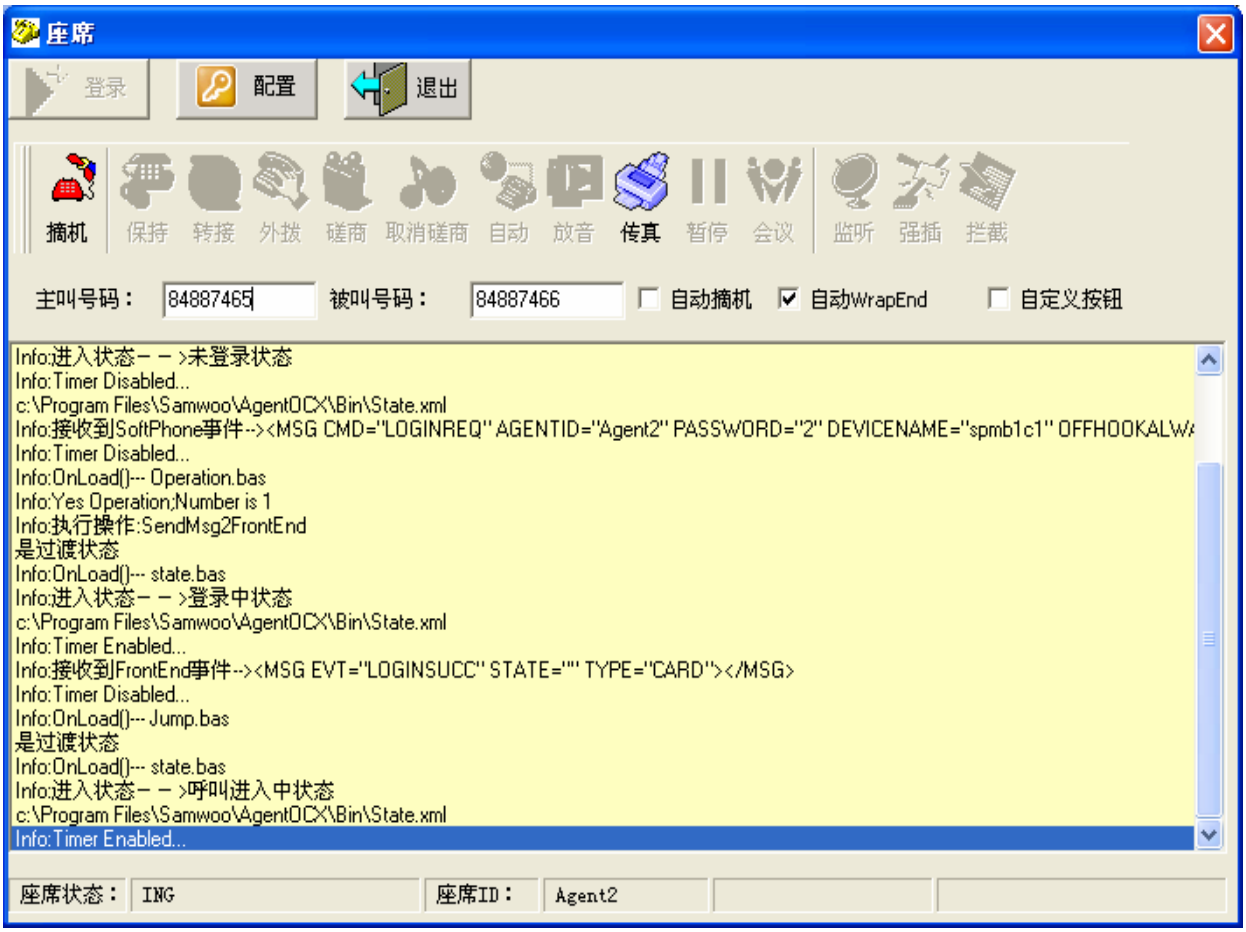


图表 22 暂停状态主界面

这时座席除了外拨之外不能进行其他任何操作，只有当恢复后进入服务状态才可以进行相关操作，这是为座席休息和话后处理提供的功能，不允许座席在空闲状态下发起外拨主要是为了避免座席在外拨的同时又电话进入而发生冲突。

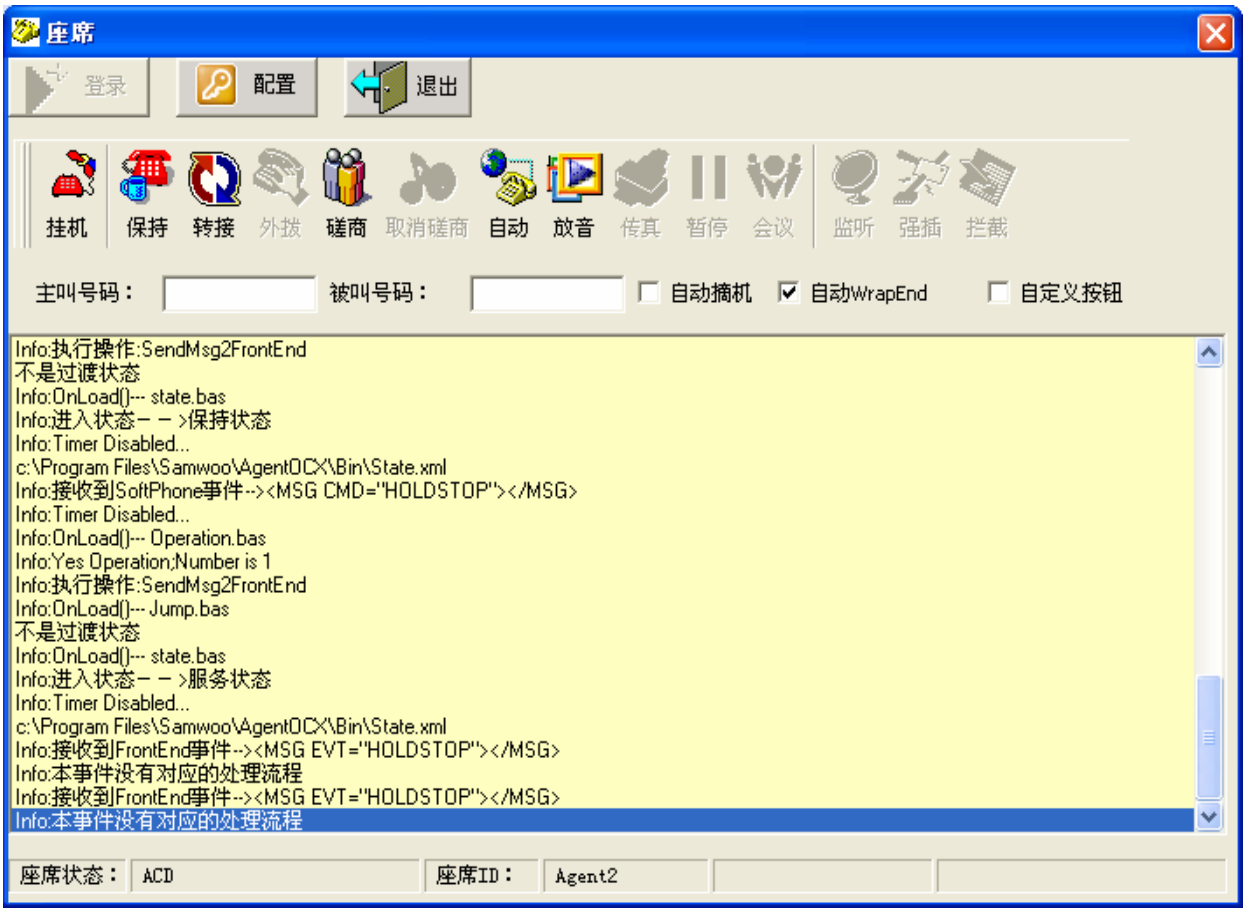
3.1.8 电话应答的实现

当有电话打到座席，座席的状态如图所示：



图表 23 呼叫进入中状态主界面

摘机后话机状态变为：



图表 24 摘机应答后的主界面

此时座席处于服务状态，MIS 端可以通过 CallArrive()函数获得通话的主叫和被叫。

代码如下：

```
Private Sub aOCX1_CallArrive(ByVal ANI As String, ByVal DNIS As String)
    txtANI.Text = ANI
    txtDNIS.Text = DNIS
        End Sub
```

此时座席可以进行转接、磋商、自动、会议操作，或挂断当前的通话，挂断当前通话话机进入话后状态，话后状态与暂停状态类似。

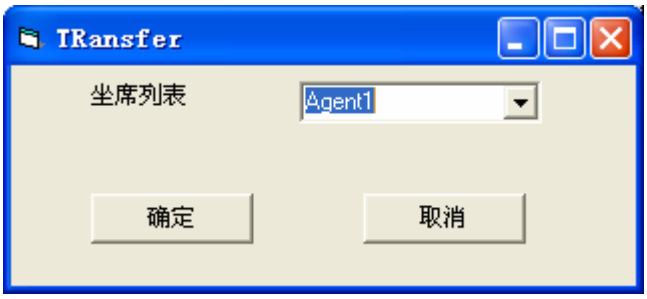
3.1.9 保持的实现：

保持当前的通话，给用户放音。用于座席的特殊情况，如查询其他资料等。

3.1.10 单步转接的实现：

即座席把当前通话转给同技能组的其他空闲座席

座席在服务状态时可以点击转接按钮，这时系统会自动查询目前登录的处于空闲状态的座席，并以列表的形式显示出来，座席要选择特定的空闲座席进行转接：



图表 25 转接界面

转接后座席转入话后状态，而被转接的座席相当于接收来话。

转接、磋商、监听所对应的事件返回的参数为 Agent 的列表，格式为 AgentName1=0;AgentName2=1;，需要 MIS 端解析该字串。在例程里有详尽的代码。

代码实现为:

1．模拟发送转接请求

```
Private Sub Transfer_Click()
    If Left(Me.Transfer.Caption, 2) = "转接" Then
        Me.aOCX1.CmdTransfer
    Else
        Me.aOCX1.CmdTransferStop
    End If
End Sub
```

2．获取当前处于空闲状态的座席

```
Private Sub aOCX1_EVTTransfer(ByVal AgentList As String)
    FrmTransfer.AgentList = AgentList
    FrmTransfer.Show 1
End Sub
```

3．对当前空闲坐席 Agentlist 字串进行解析，并在下拉列表里进行显示

```
Private Sub Form_Load()
Set Agents = New Collection

    Dim pos As Integer
    Dim posStart As Integer

    pos = 0
    posStart = 1
```

```
    pos = InStr(posStart, AgentList, ";")

    Dim tmpAgent As String

    Dim pos111 As Integer
    Dim tmpAgent1 As String

    While (pos >= 1)
        tmpAgent = Mid(AgentList, posStart, pos - posStart)
        pos111 = InStr(1, tmpAgent, "=")
        If (pos111 >= 1) Then
            tmpAgent1 = Left(tmpAgent, Len(tmpAgent) - 2)
        End If

        Agents.Add (tmpAgent1)
        posStart = pos + 1
        pos = InStr(pos + 1, AgentList, ";")

    Wend

    Dim i As Integer

    For i = 1 To Agents.Count
        Me.Combo1.AddItem Agents.Item(i)
    Next

    If Agents.Count > 0 Then
        Me.Combo1.ListIndex = 0
    Else
        Me.Combo1.Text = ""
    End If
End Sub
```

4．调用 CmdTransferToAgent 函数实现转接

```
Private Sub CmdOK_Click()
    If (Combo1.Text = "") Then
        MsgBox "Please select correct Agent"
        Exit Sub
    End If
frmMain.aOCX1.CmdTransferToAgent Combo1.Text, "", ""
Unload Me
```

```
End Sub
```

点击确定，调用 aOCX1.CmdTransferToAgent Combo1.Text, "", ""，确保所选的被转接座席不为空。

3.1.11 外拨的实现：

座席在暂停状态下可以点击外拨按钮，点击外拨按钮时，CTI Server 会上返一个 EVTDialOut 事件，MIS 系统在捕捉到这个事件后应该弹出所作的外拨界面，并调用 DialOut 函数发起外拨。如图所示：

图表 26 外拨界面

代码如下：

```
Private Sub aOCX1_EVTDialOut()
    frmDialOut.Caption = "DialOut"
    frmDialOut.Show vbModal
End Sub
```

在一个 command 按钮的 click 事件中。

```
Private Sub Command1_Click()
    If Me.Caption = "DialOut" Then '外拨的操作
        frmMain.aOCX1.DialOut Trim(Me.txtPhoneNum.Text)
    End If
    Unload Me
End Sub
```

3.1.12 磋商的实现：

座席屏蔽当前的通话，同班长席、专家座席或其他座席磋商相关业务或其它事宜，磋商结束可以将通话转接也可以将通话恢复（其中获取当前空闲座席列表代码和获取转接座席列表代码一样）。

当座席处于服务状态（接听电话或外拨成功等）时，座席可以进行磋商操作，磋商的操作与转接的部分逻辑有些类似，点击磋商按钮系统会自动查询当前登录且处于空闲状态的座席，并以列表的形式显示出来，座席可以选择空闲的座席发起磋商请求。如图：

图表 27 磋商主界面

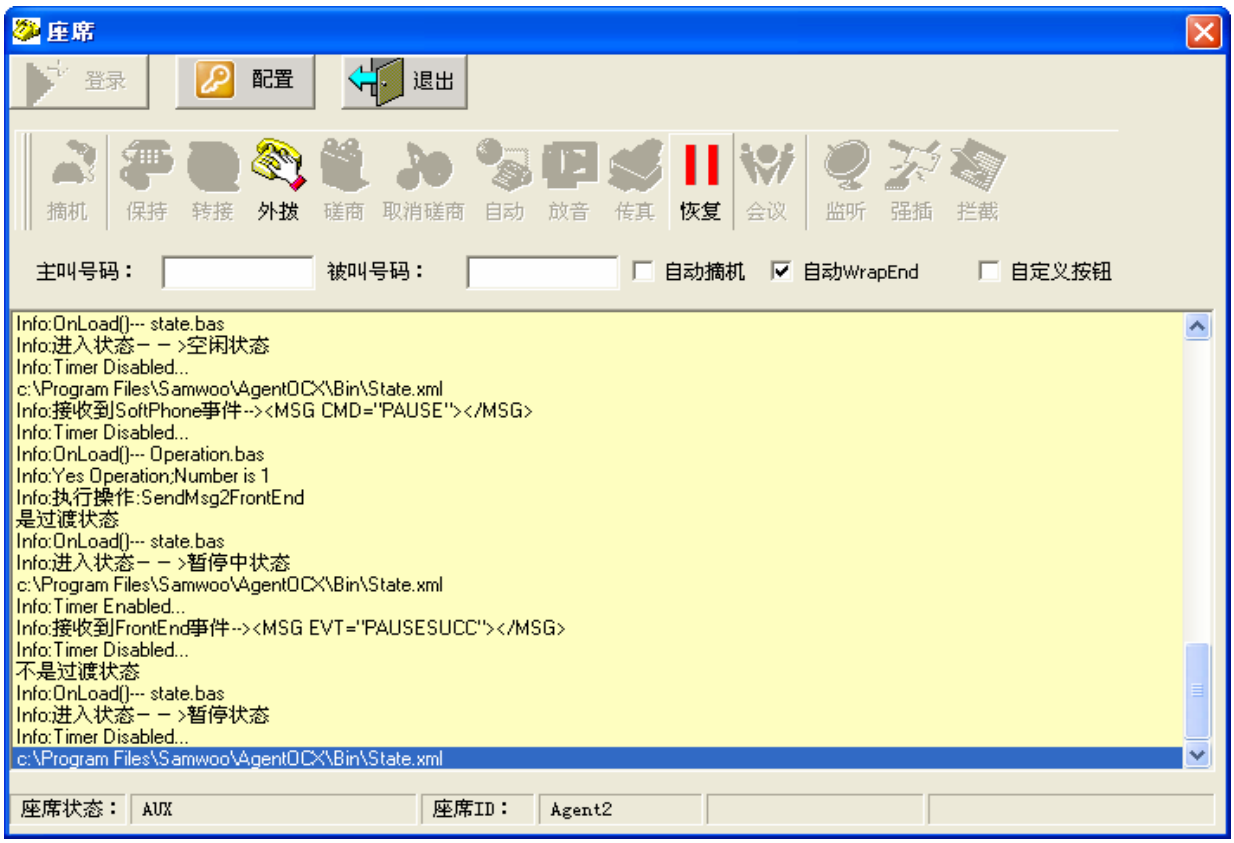
磋商分为内线磋商，外线磋商两类，内线磋商指的是在登录的座席之间发起的磋商，外线磋商是对于呼叫中心外部的外线之间进行磋商通话。

对于不同的磋商对象(外线、内线)要调用不同的函数，CmdConsultToAgent 用于内线磋商，CmdConsultToOutLine 用于外线磋商；取消磋商调用 CmdConsultCancel；在发起磋商以后，对方未摘机，可以点击用户界面上的停止来停止磋商。

磋商成功后，座席与被磋商座席通话，用户端播放音乐。当遇到难以解决的问题或一些特殊情况需要被磋商座席解决时，座席可以点击磋商转接按钮，将该通话转接到被磋商座席，或者直接点击会议按钮形成三方会议，或者座席点击结束磋商按钮重新接通被屏蔽的通话。

3.1.13 暂停的实现：

暂停座席，座席处于暂停状态，不能做任何业务，只有点击恢复才可重新进入空闲状态。如图：



图表 28 暂停状态界面

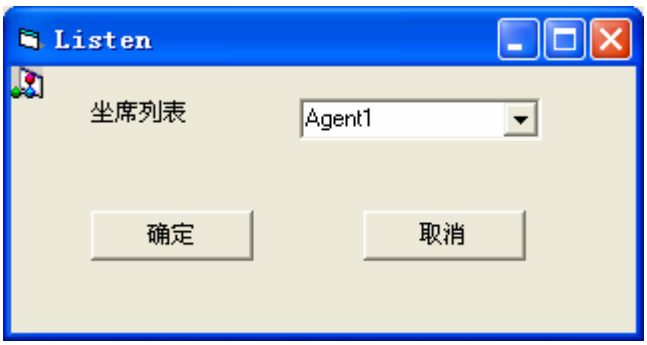
3.1.14 会议的实现：

会议的功能目前主要为三方通话
当座席处于通话状态时可以将第三方加入到会议中，它的入口也必须为磋商，即将在磋商中的三方加入到会议中。

3.1.15 监听的实现：

班长或有较高权限的座席可以监听其他座席与用户的通话

当座席处于空闲状态时可以做监听操作，点击监听按钮后系统会自动查找目前已登陆的处于服务状态的座席，并以列表的形式显示出来（其中获取当前空闲座席列表代码和获取转接座席列表代码一样），座席可以选择座席发起监听请求。如图：



图表 29 监听界面

调用函数 CmdListenToAgent 发起监听，取消时调用 CmdListenStop。

监听成功后，座席进入监听状态，可以监听到被监听座席和客户的谈话，处于监听状态的座席还可以进行拦截和强拆操作。

3.1.16 强插的实现：

强行进入三方会议，由发起监听的座席发起强插功能，实现与被监听方的三方会议。

3.1.17 拦截的实现：

强行将被监听座席同客户的通话中断，并将通话拦截到本座席，本座席转入服务状态。

3.1.18 状态栏显示：

状态栏可以显示当前座席的实时状态，当前状态可以通过上返事件 EVTReturnStatus(ByValue rstatus As String)来获得，参数 rstatus 即为座席的状态值。

代码如下：

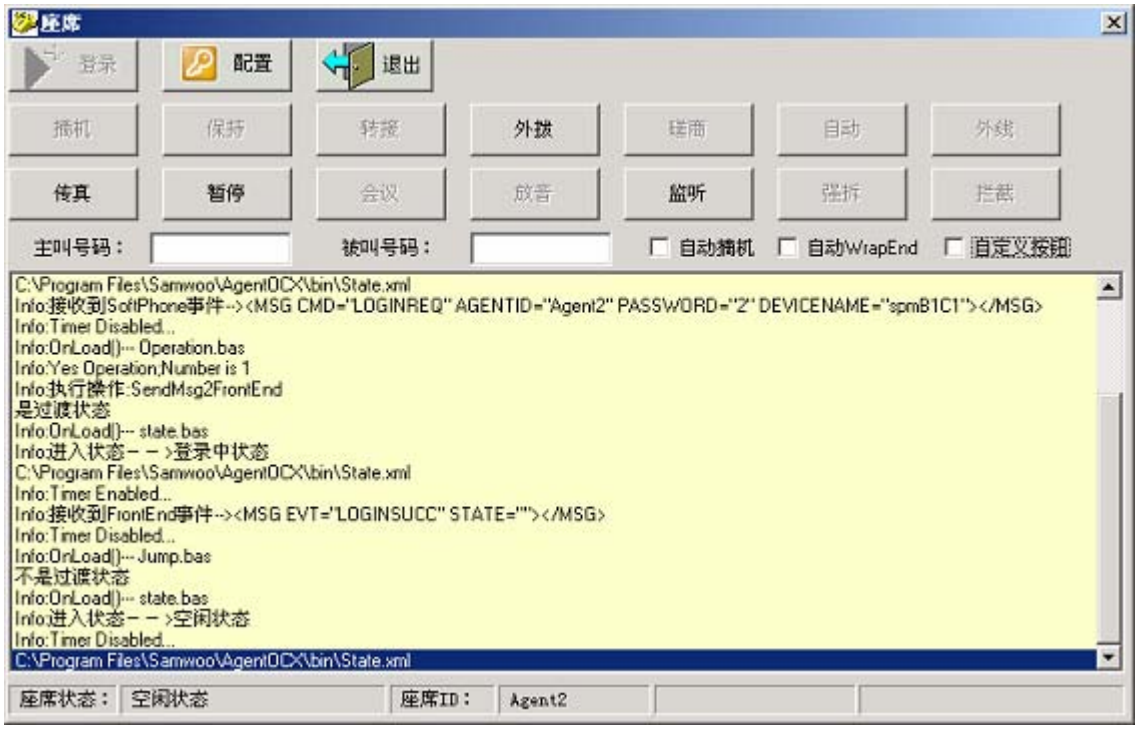
```
Private Sub aOCX1_EVTReturnStatus(ByVal rstatus As String)
    Me.StatusBar1.Panels(2).Text = rstatus
    Me.StatusBar1.Refresh
End Sub
```

3.1.19 注册表配置：

对于一些需要设置的注册表信息，系统提供可视化的借口给用户，用户可以通过调用 ShowConfig 函数来完成注册表信息的配置，点击 OK 按钮即可完成注册表信息的更新。

3.1.20 自定义座席按钮：

MIS 系统的代码编写人员可以自己定义座席控件的各个按钮，以达到界面风格的统一。如图：



图表 30 自定义按钮主界面

每个按钮对应一个特定的模拟向 CTI Server 发送消息的事件，系统可以根据 aOCX1_EVTButtonStatus() 的上返事件的值来确定个按钮的状态。代码如下:

```
Private Sub Conference_Click()
    If Me.Conference.Caption = "会议" Then
        Me.aOCX1.CmdConference
    Else
        Me.aOCX1.CmdMakeCallStop
    End If
End Sub

Private Sub Consultation_Click()
    If Me.Consultation.Caption = "磋商" Then
        Me.aOCX1.CmdConsult
    Else
        Me.aOCX1.CmdConsultStop
    End If
End Sub

Private Sub DialOut_Click()
    If Me.DialOut.Caption = "外拨" Then
        Me.aOCX1.CmdDialOut
```

```
    Else
        Me.aOCX1.CmdMakeCallStop
    End If
End Sub

Private Sub Disconnect_Click()
    Me.aOCX1.CmdDisconnect
End Sub

Private Sub Fax_Click()
    Me.aOCX1.CmdFax
End Sub

Private Sub Hold_Click()
    If Me.Hold.Caption = "保持" Then
        Me.aOCX1.CmdHold
    Else
        Me.aOCX1.CmdHoldStop
    End If
End Sub

Private Sub Hook_Click()
    If Me.Hook.Caption = "摘机" Then
        Me.aOCX1.CmdAnswer
    Else
        Me.aOCX1.CmdOnHook
    End If
End Sub

Private Sub Listen_Click()
    If Me.Listen.Caption = "监听" Then
        Me.aOCX1.CmdListen
    Else
        Me.aOCX1.CmdListenStop
    End If

End Sub

Private Sub OutPhone_Click()
    If Me.OutPhone.Caption = "自动" Then
```

```
        Me.aOCX1. CmdAuto
    Else
        Me.aOCX1. CmdAutoCancel
    End If
End Sub

Private Sub Auto_Click()
    If Me.Auto.Caption = "磋商转接" Then
        Me.aOCX1.CmdConsultTransfer "", "", "", ""
    End if
End Sub

Private Sub Pause_Click()
    If Me.Pause.Caption = "暂停" Then
        Me.aOCX1.CmdPause
    Else
        Me.aOCX1.CmdContinue
    End If
End Sub

Private Sub Play_Click()
    If Me.Play.Caption = "放音" Then
        Me.aOCX1.CmdPlay
    Else
        Me.aOCX1.CmdPlayStop
    End If
End Sub

Private Sub RopCall_Click()
    Me.aOCX1.CmdRopCall
End Sub

Private Sub Transfer_Click()
    If Me.Transfer.Caption = "转接" Then
        Me.aOCX1.CmdTransfer
    Else
        Me.aOCX1.CmdTransferStop
    End If
End Sub
```

```
Private Sub aOCX1_EVTButtonStatus(ByVal Name As String, ByVal sTitle As String, ByVal Enable As Long)
    Debug.Print Name, Title, Enable
```

```
    Select Case sTitle
        Case "OnHook"
            sTitle = "挂机"
        Case "OffHook"
            sTitle = "摘机"
        Case "HOOKOFFCONSULT"
            sTitle = "接受"
        Case "Hold"
            sTitle = "保持"
        Case "HoldCancel"
            sTitle = "取消"
        Case "Transfer"
            sTitle = "转接"
        Case "CancelTransfer"
            sTitle = "取消"
        Case "DialOut"
            sTitle = "外拨"
        Case "CancelDialOut"
            sTitle = "取消"
        Case "Consultation"
            sTitle = "磋商"
        Case "CancelConsultation"
            sTitle = "取消"
        Case "StopConsultation"
            sTitle = "磋商结束"
        Case "ConsultTransfer"
            sTitle = "磋商转接"
        Case "Auto"
            sTitle = "磋商转接"
        Case "OutPhone"
            sTitle = "自动"
        Case "CancelOutPhone"
```

```
            sTitle = "取消"
        Case "Play"
            sTitle = "放音"
        Case "PlayCancel"
            sTitle = "结束"
        Case "Fax"
            sTitle = "传真"
        Case "FaxStop"
            sTitle = "结束"
        Case "Pause"
            sTitle = "暂停"
        Case "Continue"
            sTitle = "恢复"
        Case "ContinueDialTask"
            sTitle = "放弃回访"
        Case "Listen"
            sTitle = "监听"
        Case "CancelListen"
            sTitle = "结束"
        Case "Disconnect"
            sTitle = "强插"
        Case "Conference"
            sTitle = "会议"
        Case "CancelConference"
            sTitle = "取消"
        Case "RopCall"
            sTitle = "拦截"
        Case "LOGINSUCC"
            sTitle = "登录成功"
        Case "LOGINFAIL"
            sTitle = "登录失败"
        Case "TRANSFERFAIL"
            sTitle = "转接失败"
    End Select
```

```
Select Case Name
    Case "Hook"
        SetButtonOption sTitle, Enable, Me.Hook
    Case "Hold"
        SetButtonOption sTitle, Enable, Me.Hold
    Case "Transfer"
        SetButtonOption sTitle, Enable, Me.Transfer
    Case "DialOut"
        SetButtonOption sTitle, Enable, Me.DialOut
    Case "Consultation"
        SetButtonOption sTitle, Enable, Me.Consultation
    Case "Auto"
        SetButtonOption sTitle, Enable, Me.Auto
    Case "OutPhone"
        SetButtonOption sTitle, Enable, Me.OutPhone
    Case "Fax"
        SetButtonOption sTitle, Enable, Me.Fax
    Case "Pause"
        SetButtonOption sTitle, Enable, Me.Pause
    Case "Conference"
        SetButtonOption sTitle, Enable, Me.Conference
    Case "Play"
        SetButtonOption sTitle, Enable, Me.Play
    Case "Listen"
        SetButtonOption sTitle, Enable, Me.Listen
    Case "Disconnect"
        SetButtonOption sTitle, Enable, Me.Disconnect
    Case "RopCall"
        SetButtonOption sTitle, Enable, Me.RopCall

End Select

Public Sub SetButtonOption(ByVal strTitle As String, ByVal strEnable As Integer, strButton As CommandButton)

    strButton.Caption = strTitle
    strButton.Enabled = strEnable

End Sub
```

4. 嵌入 html 文件的实现

本章讲述在 html 文件上嵌入 OCX 的方法。例子使用 UltraEdit 文件编辑工具。Html 文件上使用 vbscript 作为脚本语言。

4.1 OCX 控件插入

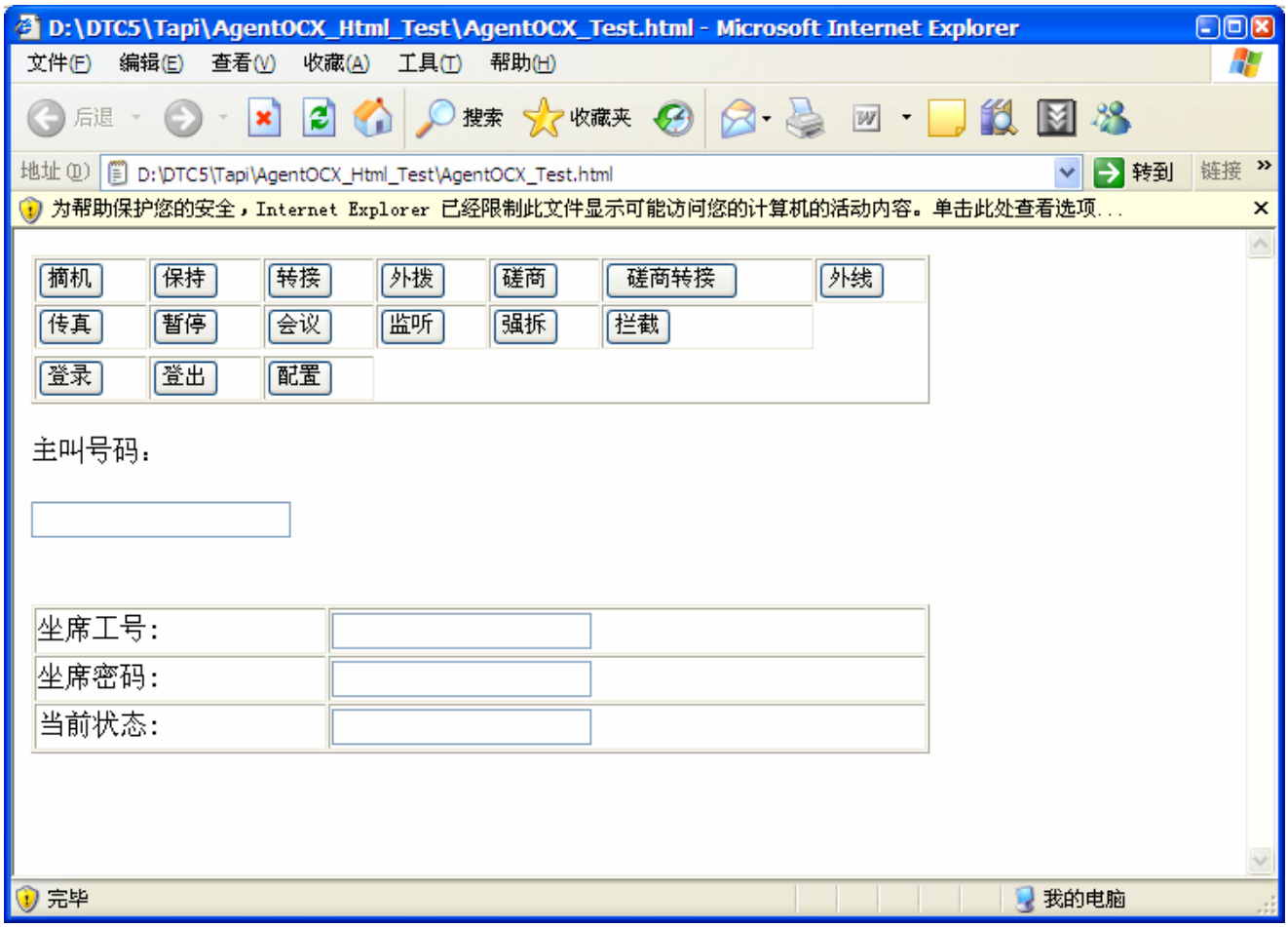
在 html 文件上加入以下语句：

```
<OBJECT id=aOCX style="WIDTH: 151px; HEIGHT: 23px"
data=data:application/x-oleobject;base64,np6Npeezi02KZ8WyuR3P+QADAACbDwAAYQIAAA==
classid=clsid:A58D9E9E-B3E7-4D8B-8A67-C5B2B91DCFF9></OBJECT>
```

以上语句把本页面的 OCX 实例命名为 aOCX。

4.2 设计基本界面

基本界面设计如下图所示：



图表 31 HTML 坐席主界面

其中页面的最上两行按钮为软电话功能按钮。包括：

- 摘机(btnHook)
- 保持(btnHold)
- 转接(btnTransfer)
- 外拨(btnDial)
- 磋商(btnConsult)
- 磋商转接(btnAuto)
- 外线(btnOutPhone)
- 传真(btnFax)
- 暂停(btnPause)
- 会议(btnConference)
- 监听(btnListen)
- 强拆(btnDisconnect)
- 拦截(btnRopCall)

第三行按钮为：

- 登录(btnLogin)
- 登出(btnLogOut)
- 配置(btnConfigure)

主叫号码显示新呼叫的主叫号码；坐席工号和坐席密码用来进行软电话登录。当前状态显示当前软电话所处的状态。

4.3 设置按钮的初始状态：

增加一个函数 SetButtoninit 用来设置按钮的初始状态：

```
Sub SetButtoninit

    btnHook.disabled = true
    btnConference.disabled = true

    btnConsult.disabled = true
    btnAuto.disabled = true
    btnDial.disabled = true
    btnDisconnect.disabled = true
    btnFax.disabled = true
    btnHold.disabled = true
```

```
        btnListen.disabled = true
        btnOutPhone.disabled = true
        btnPause.disabled = true
        btnRopCall.disabled = true
        btnTransfer.disabled = true

End Sub
```

添加一个页面初始函数 window_onload 如下:

```
Sub window_onload

        SetButtonInit
        'btnLogOut.disabled = true

End Sub
```

4.4 按钮的标题及状态变化处理

系统可以根据 aOCX_EVTButtonStatus() 的上返事件的值来确定个按钮的状态。代码如下:

```
Sub aOCX_EVTButtonStatus(sName,sTitle,bEnable)
        Select Case sTitle
            Case "OnHook"
                sTitle = "挂机"
            Case "OffHook"
                sTitle = "摘机"
            Case "HOOKOFFCONSULT"
                sTitle = "接受"
            Case "Hold"
                sTitle = "保持"
            Case "HoldCancel"
                sTitle = "取消"
            Case "Transfer"
                sTitle = "转接"
            Case "CancelTransfer"
                sTitle = "取消"
            Case "DialOut"
                sTitle = "外拨"
            Case "CancelDialOut"
                sTitle = "取消"
            Case "Consultation"
                sTitle = "磋商"
```

```
Case "CancelConsultation"
    sTitle = "取消"
Case "StopConsultation"
    sTitle = "磋商结束"
Case "ConsultTransfer"
    sTitle = "磋商转接"
Case "Auto"
    sTitle = "磋商转接"
Case "OutPhone"
    sTitle = "自动"
Case "CancelOutPhone"
    sTitle = "取消"
Case "Play"
    sTitle = "放音"
Case "PlayCancel"
    sTitle = "结束"
Case "Fax"
    sTitle = "传真"
Case "FaxStop"
    sTitle = "结束"
Case "Pause"
    sTitle = "暂停"
Case "Continue"
    sTitle = "恢复"
Case "ContinueDialTask"
    sTitle = "放弃回访"
Case "Listen"
    sTitle = "监听"
Case "CancelListen"
    sTitle = "结束"
Case "Disconnect"
    sTitle = "强插"
Case "Conference"
    sTitle = "会议"
Case "CancelConference"
    sTitle = "取消"
Case "RopCall"
    sTitle = "拦截"
Case "LOGINSUCC"
    sTitle = "登录成功"
```

```
        Case "LOGINFAIL"
            sTitle = "登录失败"
        Case "TRANSFERFAIL"
            sTitle = "转接失败"
    End Select

    'MsgBox sName & ".." & sTitle & "..." & bEnable & ":" & bDisable
    Dim bDisable
    if bEnable = 0 then
        bDisable = true
    else
        bDisable = false
    end if

    Select Case sName
        Case "Hook"
            btnHook.value = sTitle
            btnHook.disabled = bDisable
        Case "Hold"
                btnHold.value = sTitle
                btnHold.disabled = bDisable
        Case "Transfer"
                btnTransfer.value = sTitle
                btnTransfer.disabled = bDisable
        Case "DialOut"
                btnDial.value = sTitle
                btnDial.disabled = bDisable
        Case "Consultation"
                btnConsult.value = sTitle
                btnConsult.disabled = bDisable
        Case "Auto"
                btnAuto.value = sTitle
                btnAuto.disabled = bDisable
        Case "OutPhone"
                btnOutPhone.value = sTitle
                btnOutPhone.disabled = bDisable
        Case "Fax"
                btnFax.value = sTitle
                btnFax.disabled = bDisable
        Case "Pause"
                btnPause.value = sTitle
                btnPause.disabled = bDisable
```

```
        Case "Conference"
                btnConference.value = sTitle
                btnConference.disabled = bDisable
        Case "Play"

        Case "Listen"
                btnListen.value = sTitle
                btnListen.disabled = bDisable

        Case "Disconnect"
            'SetButtonOption Title, Enable, Me.Disconnect
            btnDisconnect.value = sTitle
            btnDisconnect.disabled = bDisable
        Case "RopCall"
            btnRopCall.value = sTitle
            btnRopCall.disabled = bDisable
        Case "Init"
            SetButtonInit
        'Case Else
          '    SetButtonInit
    End Select

End Sub
```

4.5 登录的实现

添加登录按钮(btnLogin)的 click 事件，如下：

```
Sub btnLogin_onclick
    aOCX.AgentID = AgentID.value
    aOCX.Password = AgentPwd.value
    aOCX.LogIn

End Sub
```

4.6 登出的实现

添加登出按钮(btnLogOut)的 click 事件，如下：

```
Sub btnLogOut_onclick
    aOCX.LogOUT
    SetButtonInit
```

```
    btnLogin.disabled = false

End Sub
```

4.7 暂停的实现：

添加暂停按钮(btnPause)的 Click 事件，如下：

```
Sub btnPause_onclick

    If Left(btnPause.value, 2) = "暂停" Then
        aOCX.CmdPause
    Else
        aOCX.CmdContinue
    End If

End Sub
```

4.8 应答和挂机的实现：

添加应答按钮(btnHook)的 Click 事件处理，如下：

```
Sub btnHook_onclick

    If Left(btnHook.value, 2) = "摘机" Then
        aOCX.CmdAnswer


    ElseIf btnHook.value = "接受" Then
        aOCX.CmdConsultAnswer
    Else
        aOCX.CmdOnHook
    End If

End Sub
```

4.9 保持和取消保持的实现：

添加保持按钮(btnHold)的 Click 事件处理，如下：

```
Sub btnHold_onclick

    If Left(btnHold.Value, 2) = "保持" Then
        aOCX.CmdHold
```

```
    Else
        aOCX.CmdHoldStop
    End If

End Sub
```

4.10 单步转接的实现：

添加单步转接按钮(btnTransfer)的 Click 事件处理，如下：

```
Sub btnTransfer_onclick

    If Left(btnTransfer.value, 2) = "转接" Then
        aOCX.CmdTransfer

    Else
        aOCX.CmdTransferStop
    End If

End Sub
```

发起单步转接后，系统会把空闲的坐席列表通过 EVTTransfer 事件通知客户端，添加 OCX 的 EVTTransfer 事件，如下：

```
Sub aOCX_EVTTransfer(AgentList)

    aOCX.ShowTransferDlg

End Sub
```

4.11 外拨的实现：

添加外拨按钮(btnDial)的事件处理，如下：

```
If Left(btnDial.value , 2) = "外拨" Then
    aOCX.CmdDialOut
else
    aOCX.CmdMakeCallStop
end if
```

并处理 EVTDialOut 事件：

```
Sub aOCX_EVTDialOut

    aOCX.ShowDialOut

```

```
End Sub
```

4.12磋商的实现:

添加磋商按钮(btnConsult)的事件处理，如下：

```
Sub btnConsult_onclick

    If Left(btnConsult.value, 4) = "磋商结束" Then
        aOCX.CmdConsultStop
    ElseIf Left(btnConsult.value, 2) = "磋商" Then
        aOCX.CmdConsult

    Else
        aOCX.CmdConsultStop
    End If

End Sub
```

发起磋商请求后，系统会把目前空闲的坐席列表通过 EVT_Consult 发送给客户端，添加 OCX 的 EVT_Consult 事件处理，如下：

```
Sub aOCX_EVTConsult(AgentList)
    aOCX.ShowConsultDlg

End Sub
```

当磋商成功后，OCX 会上返 EVT_ConsultSucc 事件，添加如下处理：

```
Sub aOCX_EVTConsultSucc(sAgentID)
    DAgentID = sAgentID
End Sub
```

其中 DAgentID 用来存储磋商的对方 AgentID 或者对方的电话号码

4.13磋商转接的实现:

添加磋商转接按钮(btnAuto)的事件处理，如下：

```
Sub btnAuto_onclick
    aOCX.CmdConsultTransfer DAgentID,"","","",""
End Sub
```

4.14磋商进入三方会议的实现:

添加会议按钮(btnConference)的事件处理，如下：

```
Sub btnConference_onclick
    If Left(btnConference.value, 2) = "会议" Then
        aOCX.CmdConference
    Else
        aOCX.CmdMakeCallStop
    End If
End Sub
```

4.15传真的实现:

添加传真按钮(btnFax)的 Click 事件处理，如下：

```
Sub btnFax_onclick
    aOCX.CmdFax
End Sub
```

4.16监听的实现:

添加监听按钮(btnListen)的 Click 事件处理，如下：

```
Sub btnListen_onclick
    If Left(btnListen.value, 2) = "监听" Then
        aOCX.CmdListen
    Else
        aOCX.CmdListenStop
    End If
End Sub
```

并处理 EVTListen 事件：

```
Sub aOCX_EVTListen
    aOCX.ShowListenDlg
End Sub
```

4.17强拆的实现:

添加强拆按钮(btnDisconnect)的 Click 事件处理，如下：

```
Sub btnDisconnect_onclick
    aOCX.CmdDisconnect

End Sub
```

4.18拦截的实现:

添加拦截按钮(btnRopCall)的 Click 事件处理，如下：

```
Sub btnRopCall_onclick
    aOCX.CmdRopCall
End Sub
```

4.19新呼叫进入事件处理：

新呼叫进入，会产生 EVTCallArrive 事件：

```
Sub aOCX_CallArrive(sAni,sDnis,sData)
    bAgentIDSpeaked = false
    Ani.value = sAni

End Sub
```

4.20播放工号的实现：

处理摘机成功事件 EVTAnswerSucc,如下：

```
Sub aOCX_EVTAnswerSucc
 if bAgentIDSpeaked = false then
    aOCX.CmdPlayAgentIDWelcome
    bAgentIDSpeaked = true
end if
End Sub
```

其中 bAgentIDSpeaked 是全局变量。

4.21状态显示:

处理 EVT_ReturnStatusCh 事件，如下：

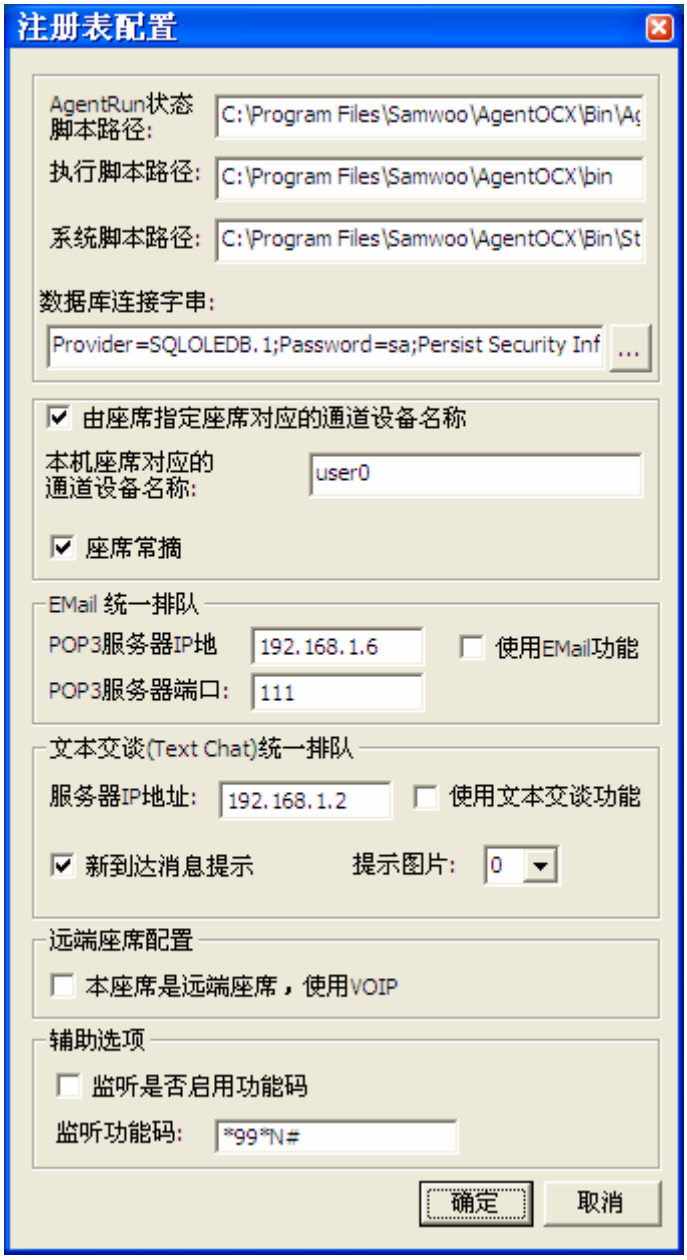
```
Sub aOCX_EVTReturnStatusCH(status)
    txtStatus.value = status
End Sub
```

4.22配置:

处理配置按钮(btnConfigure)的 Click 事件，如下：

```
Sub btnConfigure_onclick
    aOCX.ShowConfig

End Sub
```

会显示如下对话框：



图表 32 AgentOCX 配置界面

其中必须配置两个选项：

- 数据库连接字串：必须配置为 CTIServer 对应的 SQLServer 数据库，并选择 TF_CMS 为缺省库。
- 本坐席对应的通道设备名称：选择本坐席内线通道的名称。对于东进卡，命名规则为 user0,user1..userN。

5. 基于 C#.NET 的 DEMO 程序的搭建

5.1 程序搭建过程

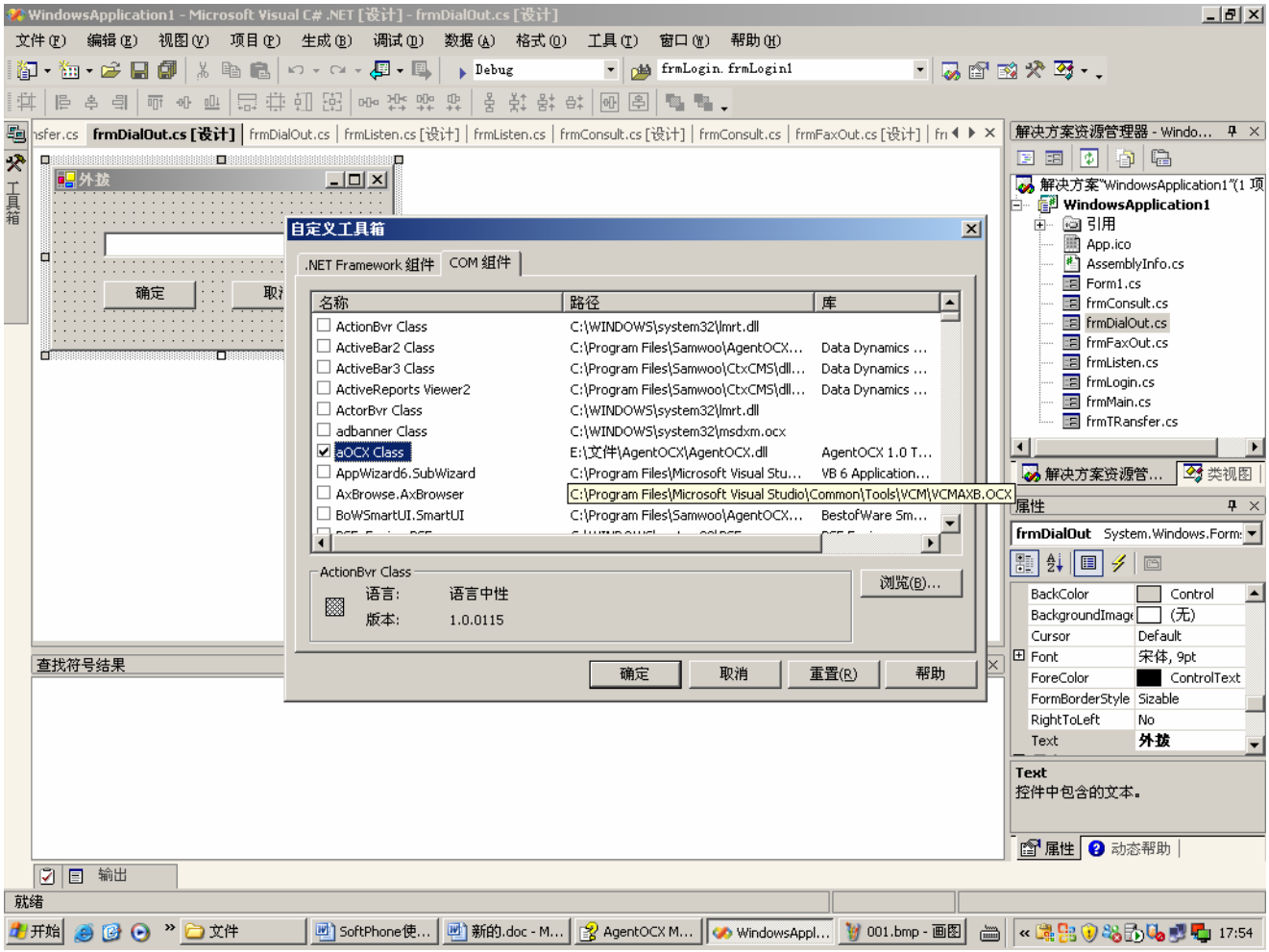
5.1.1 创建工程

首先要将所提供的 AgentOCX.dll 注册，然后创建 C#项目，选择 Windows 应用程序。

图表 33 C#程序搭建选择 Windows 应用程序

5.1.2 在工程中添加 OCX 控件

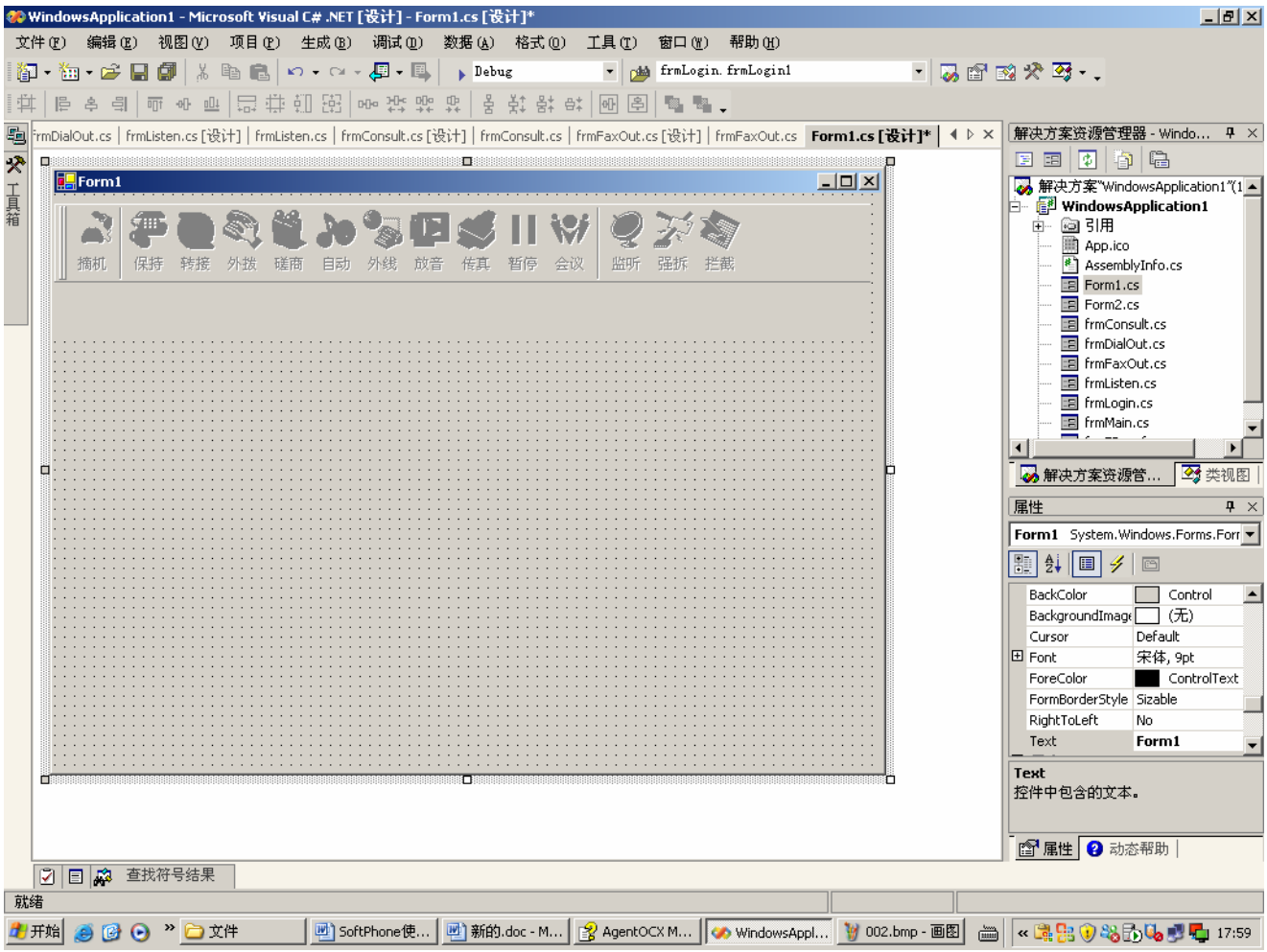
在 C#工程的工具菜单下的添加/移除工具箱选项卡中选中 COM 组件下的 aOCX Class 项，在 C#工程左侧工具箱中会显示 OCX 控件的图标:



图表 34 选择 AgentOCX Component

5.1.3 在界面上添加 OCX 控件

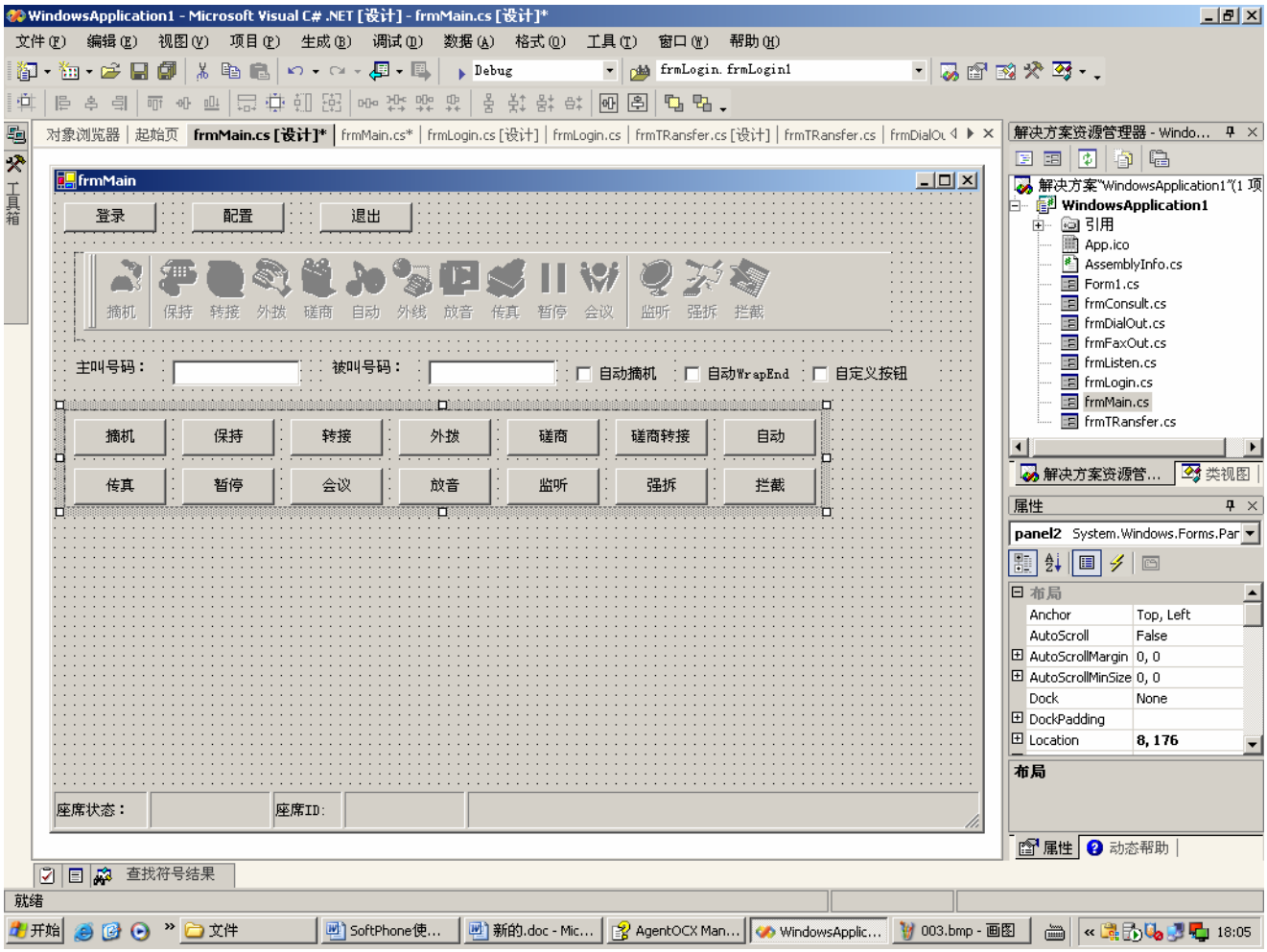
选中 AgentOCX 的图标，在 C#工程的 Form1 上画出控件的外形，为了方便整个控件的显示控制，也可将它放置在 Picturebo 控件上。如图：



图表 35 把 AgentOCX 贴在界面上

5.1.4 完成主界面

在界面上添加显示主叫和被叫的文本框及登录、注销按钮以及显示座席实时状态的状态条

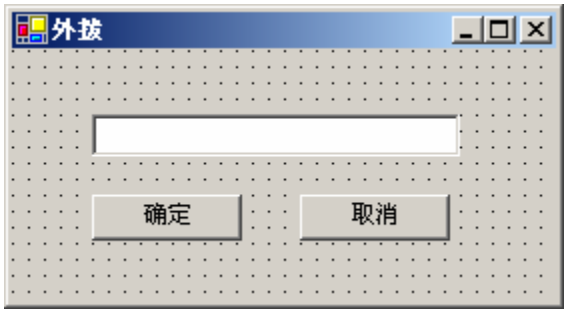


图表 36 完成主界面

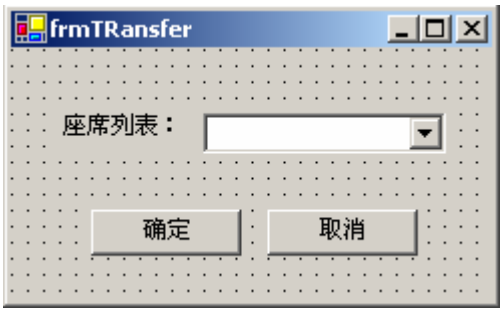
5.1.5 完成相关界面

分别作出登陆、外拨（外线、会议）、传真、转接、监听用的 Form

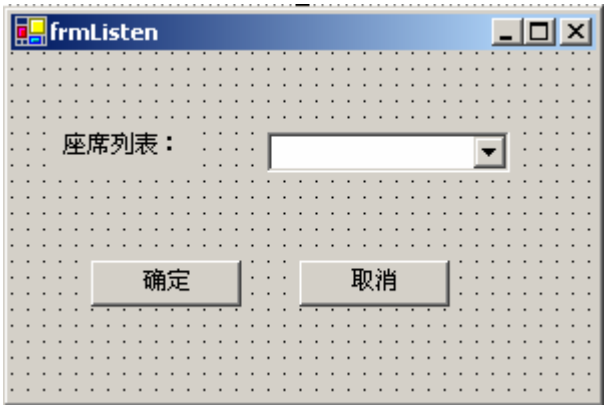
图表 37 登录对话框



图表 38 外拨对话框



图表 39 转接对话框



图表 40 监听对话框

5.1.6 Login 的代码实现

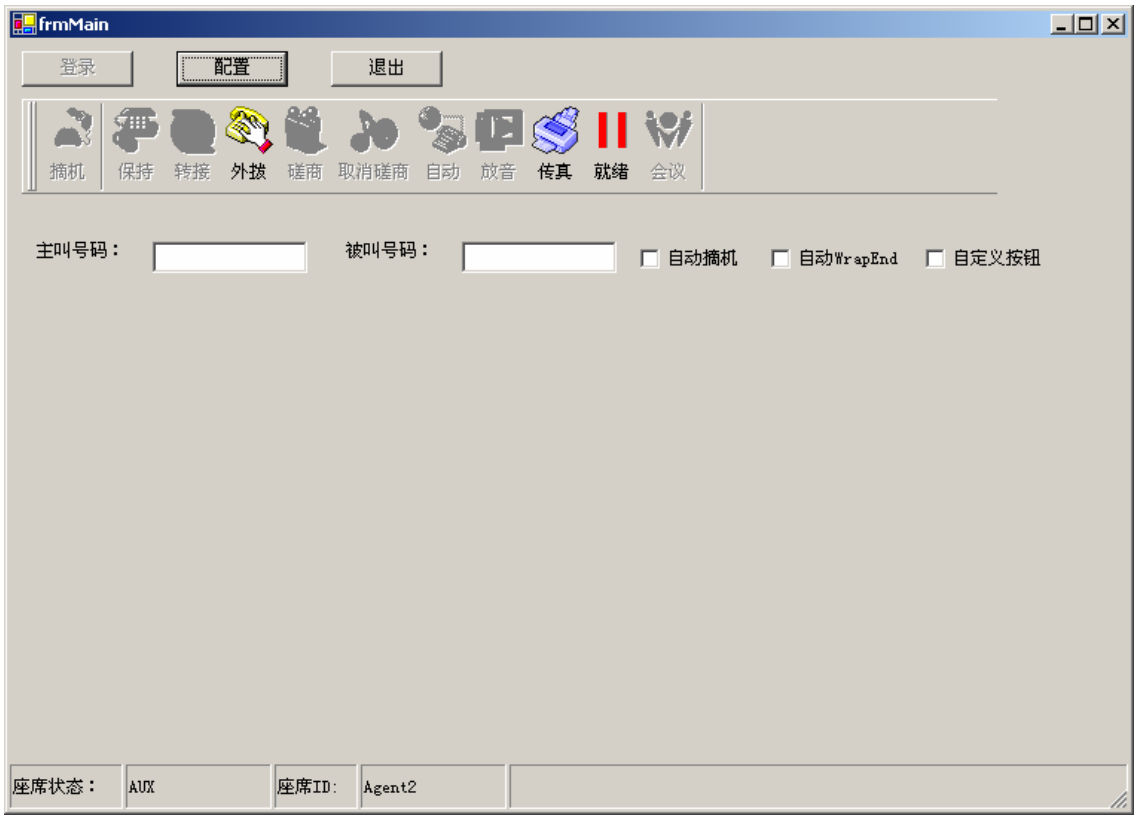
界面的工作完成之后就可以进入编码的阶段了，在图表 16 所示的登录界面中，需要提供登录的 AgentName 和 Password，调用 OCX 控件的 Login 函数即可。代码如下：

```
private void cmdCancel_Click(object sender, System.EventArgs e)
{
    this.Close();
}
private void cmdConfirm_Click(object sender, System.EventArgs e)
{

    if(this.txtUserName.Text=="")
    {
        MessageBox.Show("请您输入用户名！");
    }
    else if(this.txtPassword.Text=="")
    {
        MessageBox.Show("请您输入密码！");
    }
    else
    {
        this.frmMain1.axaOCX2.AgentID=this.txtUserName.Text.Trim();
        this.frmMain1.axaOCX2.Password=this.txtPassword.Text.Trim();
        this.frmMain1.axaOCX2.LogIn();
        this.frmMain1.statusBarPanel4.Text=this.txtUserName.Text.Trim();
        if(this.frmMain1.cheZButton.Checked)
        {
            this.frmMain1.axaOCX2.SetToolsVisible (0);
```

```
                this.frmMain1.panel2.Visible=true;
                this.frmMain1.panel1.Visible=false;
            }
            else
            {
                this.frmMain1.axaOCX2.SetToolsVisible (1);
                this.frmMain1.panel2.Visible=false;
                this.frmMain1.panel1.Visible=true;
            }
            this.Close();
        }
    }
```

5.1.7 暂停处理的实现

登录成功后的座席界面如图所示：

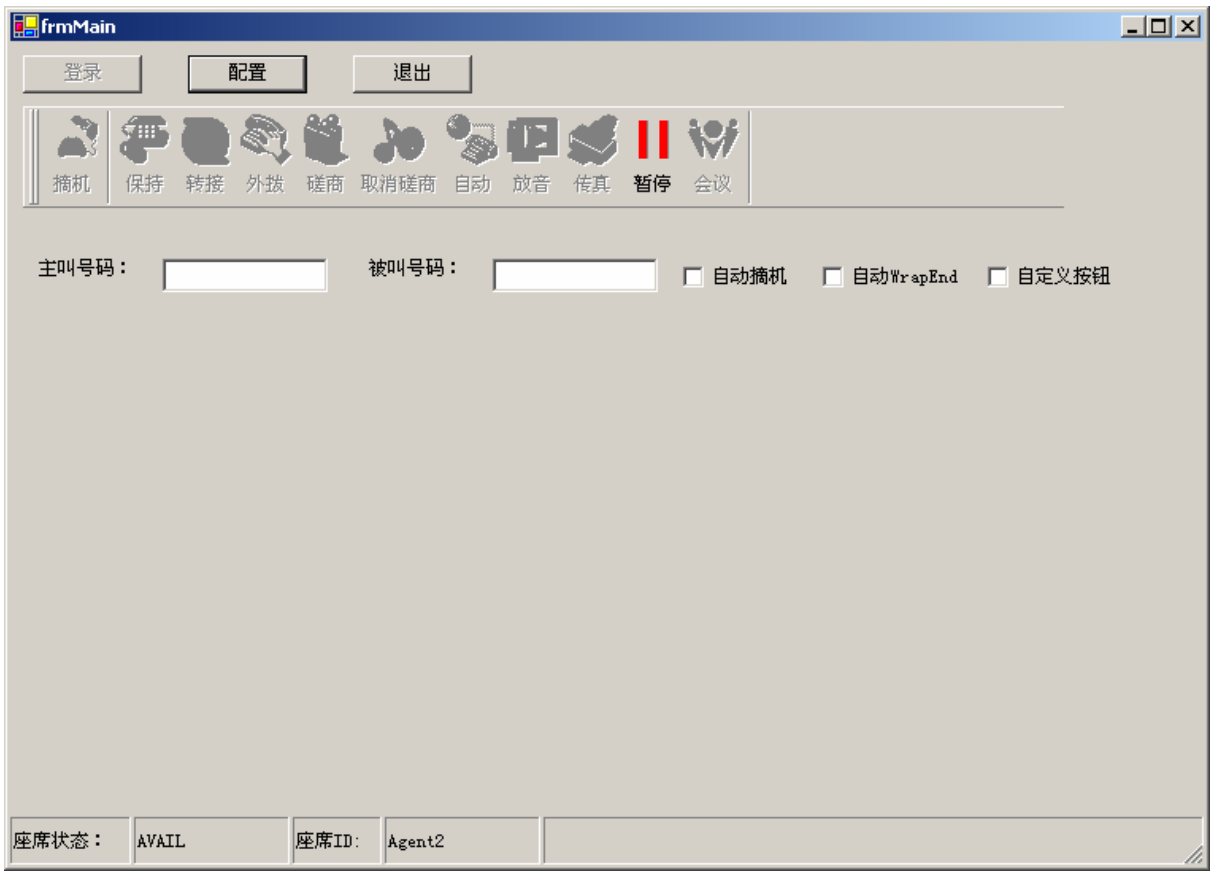


图表 41 登录成功主界面

座席登录成功后处于空闲状态，可以进行软电话的相关操作，如：传真、监听。

底下的状态条显示了当前的座席状态及登录座席的 ID 信息。

当座席点击暂停时，座席进入暂停状态：

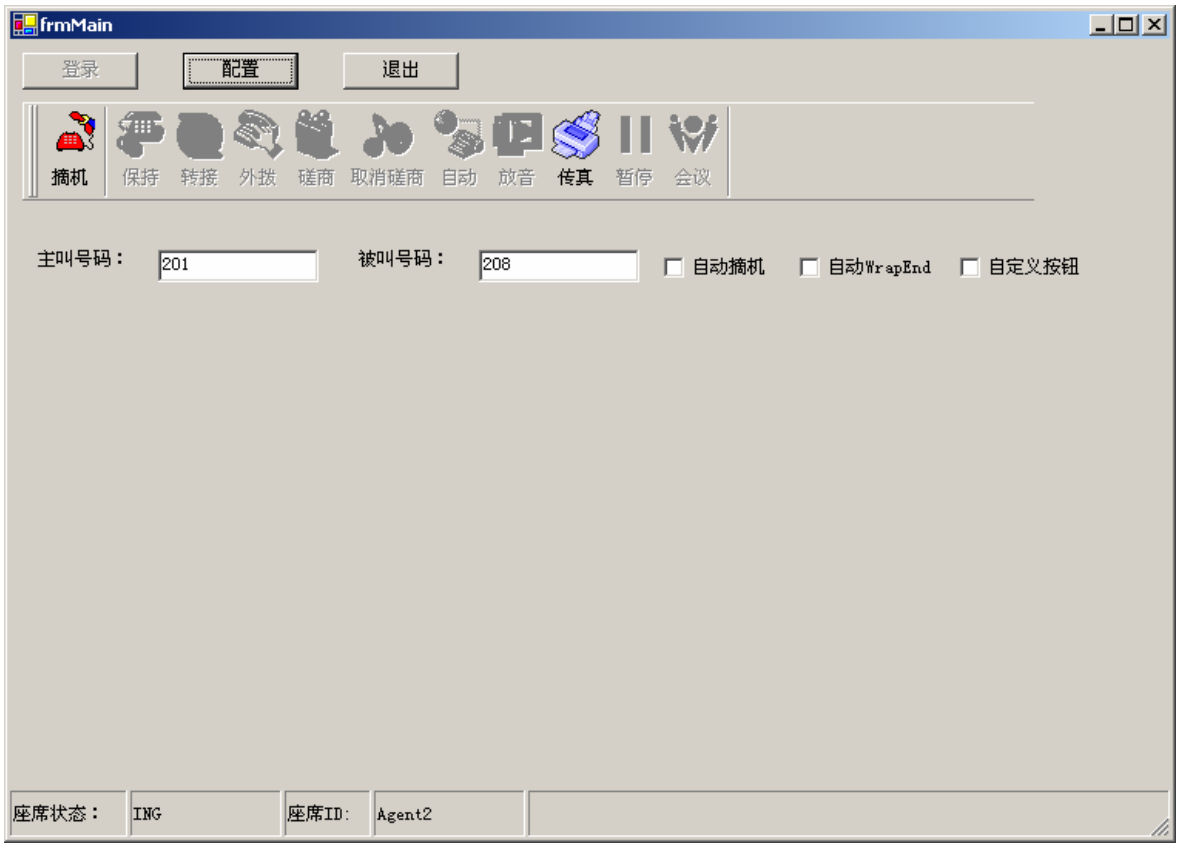


图表 42 暂停状态主界面

这时座席除了外拨之外不能进行其他任何操作，只有当恢复后进入服务状态才可以进行相关操作，这是为座席休息和话后处理提供的功能，不允许座席在空闲状态下发起外拨主要是为了避免座席在外拨的同时又电话进入而发生冲突。

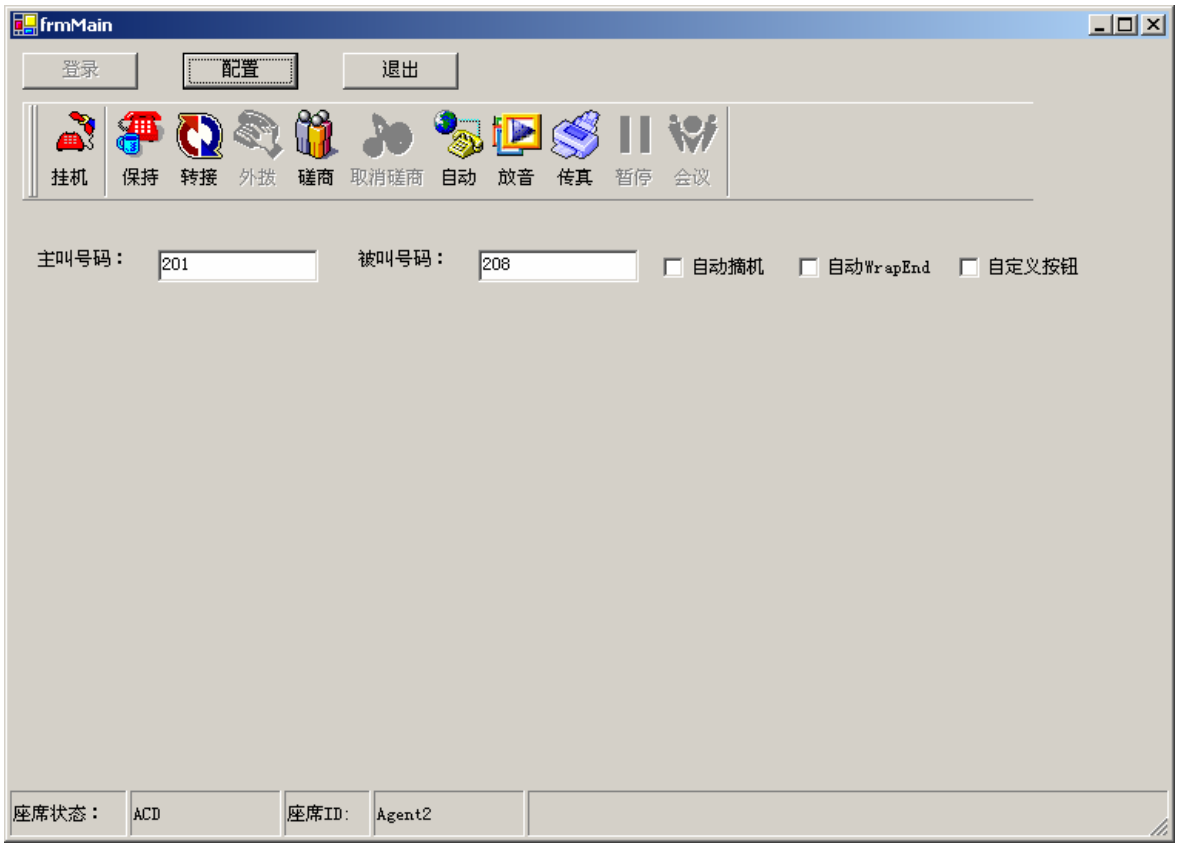
5.1.8 电话应答的实现

当有电话打到座席，座席的状态如图所示：



图表 43 呼叫进入中状态主界面

摘机后话机状态变为：



图表 44 摘机应答后的主界面

此时座席处于服务状态，MIS 端可以通过 CallArrive()函数获得通话的主叫和被叫。

代码如下：

```
private void axaOCX2_CallArrive(object sender, AxAGENTOCXLib._IaOCXEvents_CallArriveEvent e)
{
    this.txtANI.Text=e.ani;
    this.txtDNIS.Text=e.dnis;
    if (this.cheAutoHook.Checked)
    {
        this.axaOCX2.CmdAnswer();
    }
}
```

此时座席可以进行转接、磋商、自动、会议操作，或挂断当前的通话，挂断当前通话话机进入话后状态，话后状态与暂停状态类似。

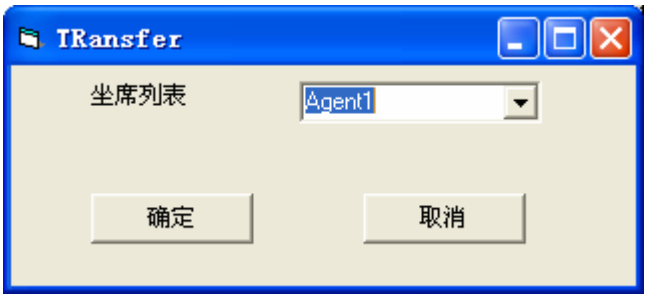
5.1.9 保持的实现：

保持当前的通话，给用户放音。用于座席的特殊情况，如查询其他资料等。

5.1.10 单步转接的实现：

即座席把当前通话转给同技能组的其他空闲座席

座席在服务状态时可以点击转接按钮，这时系统会自动查询目前登录的处于空闲状态的座席，并以列表的形式显示出来，座席要选择特定的空闲座席进行转接：



图表 45 转接界面

转接后座席转入话后状态，而被转接的座席相当于接收来话。

转接、磋商、监听所对应的事件返回的参数为 Agent 的列表，格式为 AgentName1=0;AgentName2=1;，需要 MIS 端解析该字串。在例程里有详尽的代码。

代码实现为:

1．模拟发送转接请求

```
private void cmdTransfer_Click(object sender, System.EventArgs e)
{
    if(this.cmdTransfer.Text=="转接")
    {
        this.axaOCX2.CmdTransfer();
    }
    else
    {
        this.axaOCX2.CmdTransferStop();
    }
}
```

2．获取当前处于空闲状态的座席以及对当前空闲坐席 Agentlist 字串进行解析，并在下拉列表里进行显示

```
private void axaOCX2_EVTTransferEvent(object sender, _IaOCXEvents_EVTTransferEvent e)
{
    string strAgent,strAgent1;
    int pos2,i;
    bool temp;
    frmTRansfer frmTRansfer1=new frmTRansfer();
    frmTRansfer1.frmMain1=this;
    frmTRansfer1.Show();
```

```
            strAgent=e.agentList;
            i=strAgent.Length;

            if(i>0)
            {
                temp=true;
            }
            else
            {
                temp=false;
            }
            while(temp)
          {
              pos2=strAgent.IndexOf(";",0,i);
              strAgent1=strAgent.Substring(0,pos2-2);
              frmTRansfer1.cmbAgent.Items.Add(strAgent1);
              strAgent=strAgent.Remove(0,pos2+1);
              i=strAgent.Length;
              if(i<=0)
              {
                 temp=false;
              }
          }
        }
```

3．调用 CmdTransferToAgent 函数实现转接

```
        private void cmdConfirm_Click(object sender, System.EventArgs e)
        {
            if(this.cmbAgent.Text=="")
            {
                MessageBox.Show("请您选择正确的座席！") ;

            }
            else
            {
                this.frmMain1.axaOCX2.CmdTransferToAgent(this.cmbAgent.Text,"","","");
                this.Close();
            }
        }
```

点击确定，调用 aOCX1.CmdTransferToAgent t(this.cmbAgent.Text,"","","")，确保所选的被转接座席不为空。

5.1.11 外拨的实现：

座席在暂停状态下可以点击外拨按钮，点击外拨按钮时，CTI Server 会上返一个 EVTDialOut 事件，MIS 系统在捕捉到这个事件后应该弹出所作的外拨界面，并调用 DialOut 函数发起外拨。如图所示：

图表 46 外拨界面

代码如下：

```
private void axaOCX2_EVTDialOutEvent(object sender, EventArgs e)
{
    frmDialOut frmDialOut1=new frmDialOut();
    frmDialOut1.frmMain1=this;
    frmDialOut1.Show();
}
```

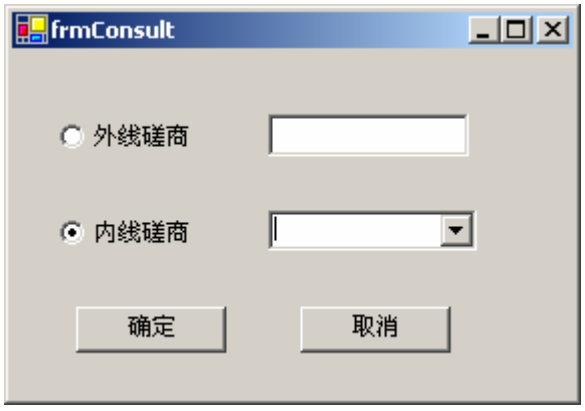
在一个 command 按钮的 click 事件中。

```
private void cmdDialout_Click(object sender, System.EventArgs e)
{
    if(this.cmdDialout.Text=="外拨")
    {
        this.axaOCX2.CmdDialOut();
    }
    else
    {
        this.axaOCX2.CmdMakeCallStop();
    }
}
```

5.1.12 磋商的实现：

座席屏蔽当前的通话，同班长席、专家座席或其他座席磋商相关业务或其它事宜，磋商结束可以将通话转接也可以将通话恢复（其中获取当前空闲座席列表代码和获取转接座席列表代码一样）。

当座席处于服务状态（接听电话或外拨成功等）时，座席可以进行磋商操作，磋商的操作与转接的部分逻辑有些类似，点击磋商按钮系统会自动查询当前登录且处于空闲状态的座席，并以列表的形式显示出来，座席可以选择空闲的座席发起磋商请求。如图：



图表 47 磋商主界面

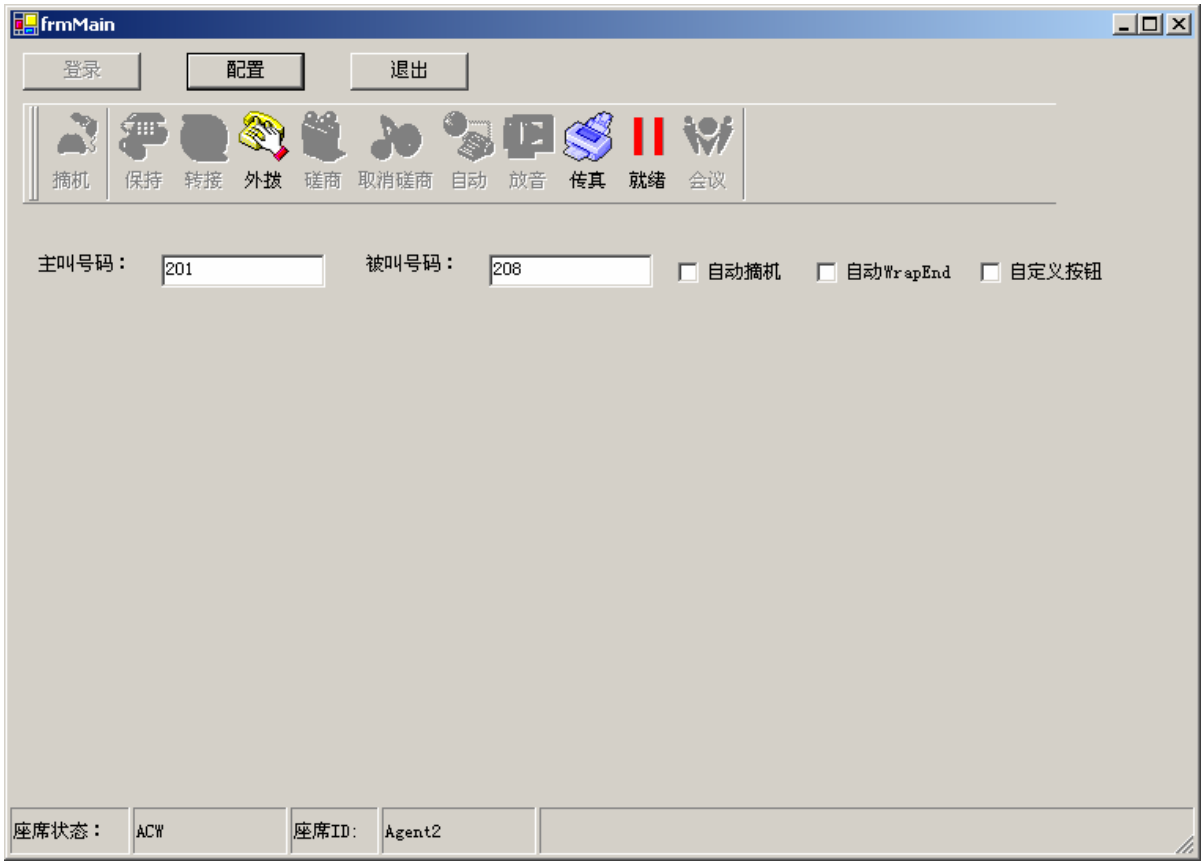
磋商分为内线磋商，外线磋商两类，内线磋商指的是在登录的座席之间发起的磋商，外线磋商是对于呼叫中心外部的外线之间进行磋商通话。

对于不同的磋商对象(外线、内线)要调用不同的函数，CmdConsultToAgent 用于内线磋商，CmdConsultToOutLine 用于外线磋商；取消磋商调用 CmdConsultCancel；在发起磋商以后，对方未摘机，可以点击用户界面上的停止来停止磋商。

磋商成功后，座席与被磋商座席通话，用户端播放音乐。当遇到难以解决的问题或一些特殊情况需要被磋商座席解决时，座席可以点击磋商转接按钮，将该通话转接到被磋商座席，或者直接点击会议按钮形成三方会议，或者座席点击结束磋商按钮重新接通被屏蔽的通话。

5.1.13 暂停的实现：

暂停座席，座席处于暂停状态，不能做任何业务，只有点击恢复才可重新进入空闲状态。如图：



图表 48 暂停状态界面

5.1.14 会议的实现：

会议的功能目前主要为三方通话
当座席处于通话状态时可以将第三方加入到会议中，它的入口也必须为磋商，即将在磋商中的三方加入到会议中。

5.1.15 监听的实现：

班长或有较高权限的座席可以监听其他座席与用户的通话

当座席处于空闲状态时可以做监听操作，点击监听按钮后系统会自动查找目前已登陆的处于服务状态的座席，并以列表的形式显示出来（其中获取当前空闲座席列表代码和获取转接座席列表代码一样），座席可以选择座席发起监听请求。如图：

图表 49 监听界面

调用函数 CmdListenToAgent 发起监听，取消时调用 CmdListenStop。

监听成功后，座席进入监听状态，可以监听到被监听座席和客户的谈话，处于监听状态的座席还可以进行拦截和强拆操作。

5.1.16 强插的实现：

强行进入三方会议，由发起监听的座席发起强插功能，实现与被监听方的三方会议。

5.1.17 拦截的实现：

强行将被监听座席同客户的通话中断，并将通话拦截到本座席，本座席转入服务状态。

5.1.18 状态栏显示：

状态栏可以显示当前座席的实时状态，当前状态可以通过上返事件 EVTReturnStatus(ByValue rstatus As String)来获得，参数 rstatus 即为座席的状态值。

代码如下：

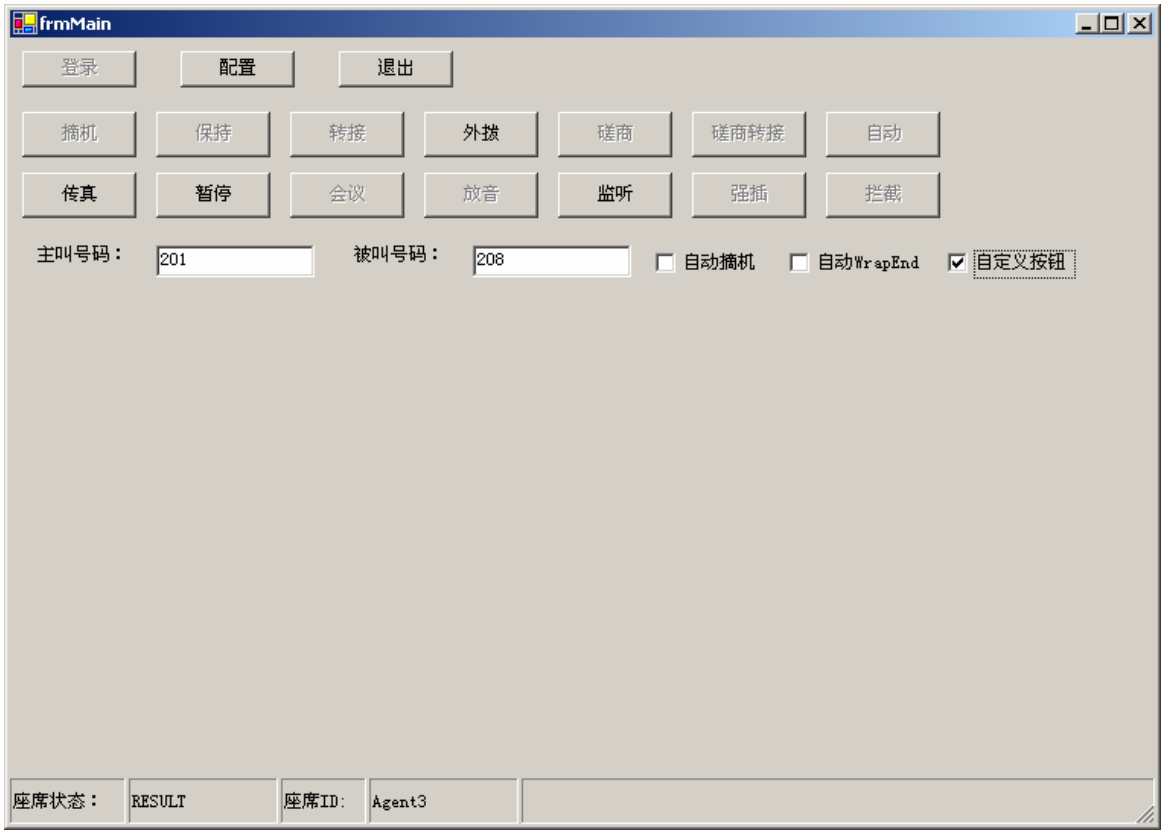
```
private void axaOCX2_EVTReturnStatusEvent(object sender, _IaOCXEvents_EVTReturnStatusEvent e)
{
    this.statusBarPanel2.Text=e.rstatus;
    this.statusBar1.Refresh();
    if(e.rstatus=="连接断开")
    {
        this.cmdLogin.Enabled=true;
    }
    else if(e.rstatus=="空闲状态")
    {
        this.LabelInfo.Visible=false;
    }

}
```

5.1.19 注册表配置：

对于一些需要设置的注册表信息，系统提供可视化的借口给用户，用户可以通过调用 ShowConfig 函数来完成注册表信息的配置，点击 OK 按钮即可完成注册表信息的更新。

5.1.20 自定义座席按钮：

MIS 系统的代码编写人员可以自己定义座席控件的各个按钮，以达到界面风格的统一。如图：



图表 50 自定义按钮主界面

每个按钮对应一个特定的模拟向 CTI Server 发送消息的事件，系统可以根据 aOCX1_EVTButtonStatus() 的上返事件的值来确定个按钮的状态。代码如下:

```
private void cmdHook_Click(object sender, System.EventArgs e)
{
    if(this.cmdHook.Text=="摘机")
    {
        this.axaOCX2.CmdAnswer();
    }
    else if(this.cmdHook.Text=="接受")
    {
        this.axaOCX2.CmdConsultAnswer();
    }
    else
    {
        this.axaOCX2.CmdOnHook();
    }
}
private void cmdHold_Click(object sender, System.EventArgs e)
{
```

```
        if(this.cmdHold.Text=="保持")
        {
            this.axaOCX2.CmdHold();
        }
        else
        {
            this.axaOCX2.CmdHoldStop();
        }
    }

    private void cmdTransfer_Click(object sender, System.EventArgs e)
    {
        if(this.cmdTransfer.Text=="转接")
        {
            this.axaOCX2.CmdTransfer();
        }
        else
        {
            this.axaOCX2.CmdTransferStop();
        }
    }

    private void cmdDialout_Click(object sender, System.EventArgs e)
    {
        if(this.cmdDialout.Text=="外拨")
        {
            this.axaOCX2.CmdDialOut();
        }
        else
        {
            this.axaOCX2.CmdMakeCallStop();
        }
    }

    private void cmdConsultation_Click(object sender, System.EventArgs e)
    {
        if(this.cmdConsultation.Text=="磋商")
        {
            this.axaOCX2.CmdConsult();
        }
```

```
        else if(this.Text=="磋商结束")
        {
            this.axaOCX2.CmdConsultStop();
        }
        else
        {
            this.axaOCX2.CmdConsultStop();
        }
    }
    private void cmdAuto_Click(object sender, System.EventArgs e)
    {
        if(this.cmdAuto.Text=="磋商转接")
        {
            this.axaOCX2.CmdTransferToAgent("","","","");
        }
        else
        {
            this.axaOCX2.CmdTransferStop();
        }
    }

    private void cmdOutphone_Click(object sender, System.EventArgs e)
    {
        if(this.cmdOutphone.Text=="自动")
        {
            this.axaOCX2.CmdAuto();
        }
        else
        {
            this.axaOCX2.CmdAutoCancel();
        }
    }

    private void cmdFax_Click(object sender, System.EventArgs e)
    {
        this.axaOCX2.CmdFax();
    }

    private void cmdPause_Click(object sender, System.EventArgs e)
    {
        if(this.cmdPause.Text=="暂停")
```

```
        {
            this.axaOCX2.CmdPause();
        }
        else
        {
            this.axaOCX2.CmdContinue();
        }
    }

    private void cmdConference_Click(object sender, System.EventArgs e)
    {
        if(this.cmdConference.Text=="会议")
        {
            this.axaOCX2.CmdConference();
        }
        else
        {
            this.axaOCX2.CmdMakeCallStop();
        }
    }

    private void cmdPlay_Click(object sender, System.EventArgs e)
    {
        if(this.cmdPlay.Text=="放音")
        {
            this.axaOCX2.CmdPlay();
        }
        else
        {
            this.axaOCX2.CmdPlayStop();
        }
    }

    private void cmdListen_Click(object sender, System.EventArgs e)
    {
        if(this.cmdListen.Text=="监听")
        {
            this.axaOCX2.CmdListen();
        }
        else
        {
```

```
                this.axaOCX2.CmdListenStop();
            }
        }

        private void cmdDisconnect_Click(object sender, System.EventArgs e)
        {
          this.axaOCX2.CmdIntrude();
        }

        private void cmdRopcall_Click(object sender, System.EventArgs e)
        {
            this.axaOCX2.CmdRopCall();
        }
    private void axaOCX2_EVTButtonStatusEvent(object sender, _IaOCXEvents_EVTButtonStatusEvent e)
        {
            switch(e.sTitle)
            {

                case "OnHook":
                        e.sTitle = "挂机";
                       break;
                case "OffHook":
                        e.sTitle = "摘机";
                       break;
                case "HOOKOFFCONSULT":
                        e.sTitle = "接受";
                       break;
                case "Hold":
                        e.sTitle = "保持";
                       break;
                case "HoldCancel":
                        e.sTitle = "取消";
                       break;
                case "Transfer":
                        e.sTitle = "转接";
                       break;
                case "CancelTransfer":
                        e.sTitle = "取消";
                       break;
```

```
case "DialOut":
        e.sTitle = "外拨";
        break;
case "CancelDialOut":
        e.sTitle = "取消";
        break;
case "Consultation":
        e.sTitle = "磋商";
        break;
case "CancelConsultation":
        e.sTitle = "取消";
        break;
case "StopConsultation":
        e.sTitle = "磋商结束";
        break;
  case "ConsultTransfer":
        e.sTitle = "磋商转接";
        break;
  case "Auto":
        e.sTitle = "磋商转接";
        break;
  case "OutPhone":
        e.sTitle = "自动";
        break;
  case "CancelOutPhone":
        e.sTitle = "取消";
        break;
  case "Play":
        e.sTitle = "放音";
        break;
  case "PlayCancel":
        e.sTitle = "结束";
        break;
  case "Fax":
        e.sTitle = "传真";
        break;
  case "FaxStop":
```

```
            e.sTitle = "结束";
            break;
        case "Pause":
            e.sTitle = "暂停";
            break;
        case "Continue":
            e.sTitle = "恢复";
            break;
        case "ContinueDialTask":
            e.sTitle = "放弃回访";
            break;
        case "Listen":
            e.sTitle = "监听";
            break;
        case "CancelListen":
            e.sTitle = "结束";
            break;
        case "Disconnect":
            e.sTitle = "强插";
            break;
        case "Conference":
            e.sTitle = "会议";
            break;
        case "CancelConference":
            e.sTitle = "取消";
            break;
        case "RopCall":
            e.sTitle = "拦截";
            break;
        case "LOGINSUCC":
            e.sTitle = "登录成功";
            break;
        case "LOGINFAIL":
            e.sTitle = "登录失败";
            break;
        case "TRANSFERFAIL":
            e.sTitle = "转接失败";
```

```
                break;
            }

            switch(e.name)
            {

                case "Hook":

                    SetButtonOption (e.sTitle, e.enable,this.cmdHook);
                        break;

                case "Hold":

                    SetButtonOption (e.sTitle, e.enable, this.cmdHold);
                        break;

                case "Transfer":

                    SetButtonOption (e.sTitle, e.enable, this.cmdTransfer);
                        break;

                case "DialOut":

                    SetButtonOption (e.sTitle, e.enable, this.cmdDialout);
                        break;

                case "Consultation":

                    SetButtonOption (e.sTitle, e.enable, this.cmdConsultation);
                        break;

              case "Auto":

                    SetButtonOption (e.sTitle, e.enable, this.cmdAuto);
                        break;

                case "OutPhone":

                    SetButtonOption (e.sTitle, e.enable, this.cmdOutphone);
                        break;

                case "Fax":
```

```
                    SetButtonOption (e.sTitle, e.enable, this.cmdFax);
                        break;

                case "Pause":

                    SetButtonOption (e.sTitle, e.enable, this.cmdPause);
                        break;

                case "Conference":

                    SetButtonOption (e.sTitle, e.enable, this.cmdConference);
                        break;

                case "Play":

                    SetButtonOption (e.sTitle, e.enable, this.cmdPlay);
                        break;

                case "Listen":

                    SetButtonOption (e.sTitle, e.enable, this.cmdListen);
                        break;

                case "Disconnect":

                    SetButtonOption (e.sTitle, e.enable, this.cmdDisconnect);
                        break;

                case "RopCall":

                    SetButtonOption (e.sTitle, e.enable, this.cmdRopcall);
                        break;

            }
        }
public void SetButtonOption(string strTitle,int strEnable,System.Windows.Forms.Button strButton)
        {
          bool temp;
            if(strEnable>0)
            {
                temp=true;
```

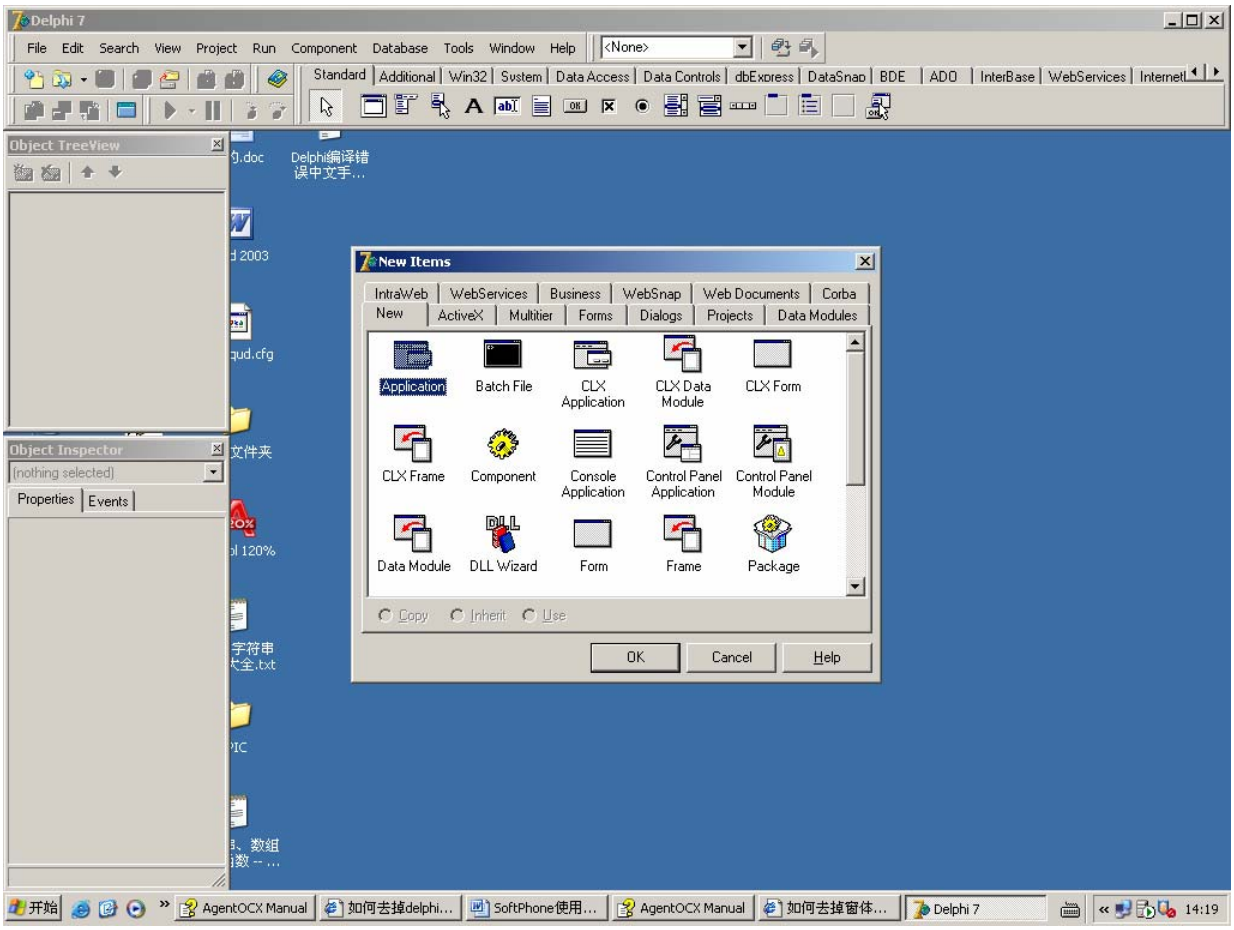
```
            }
            else
            {
                temp=false;
            }
        strButton.Text=strTitle;
        strButton.Enabled=temp;
    }
```

6. 基于 DELPHI 的 DEMO 程序的搭建

6.1 程序搭建过程

6.1.1 创建工程

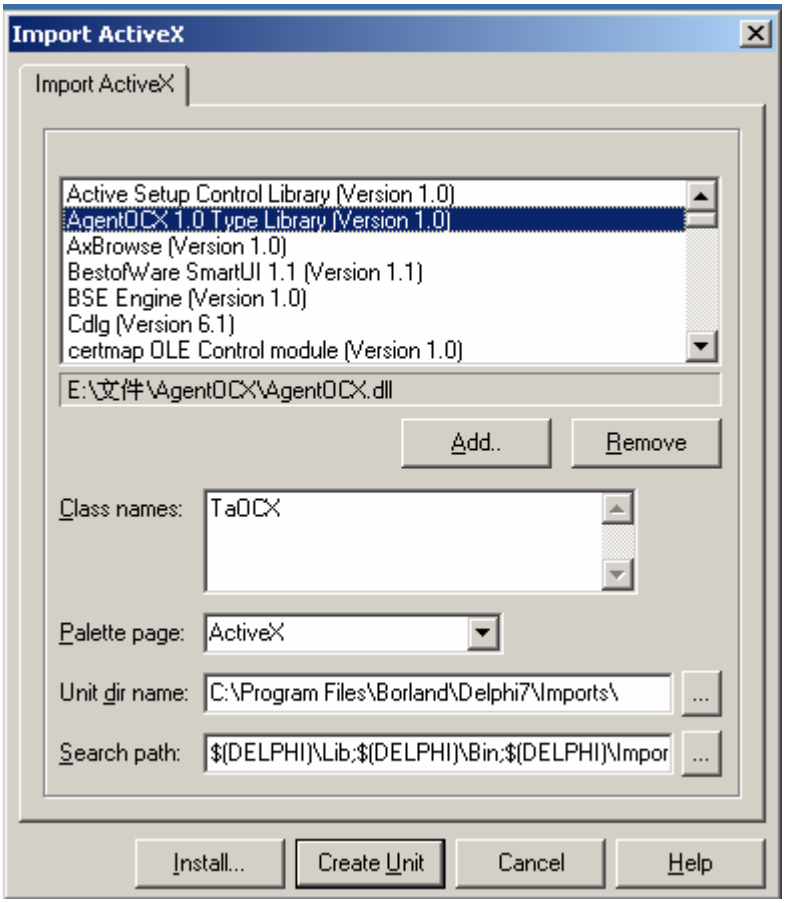
首先要将所提供的 AgentOCX.dll 注册，然后创建 DEPHI 工程，选择 Application 模式



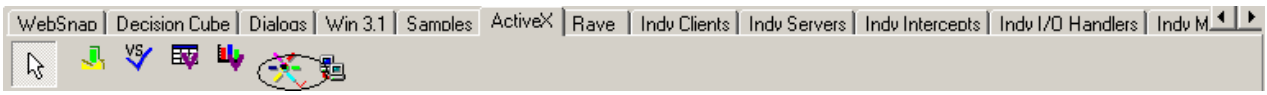
图表 51 delphi 程序搭建选择 Application

6.1.2 OCX 控件注册

在菜单的 Component 选项中点击 Import ActiveX，显示如下窗口（图 2），选择 AgentOCX1.0 Type Library(Version 1.0)，点击 Install，则会在 Delphi 的 Componet 栏中的 ActiveX 部分增加如图 3 所示的 SoftPhone Control。为了和 CTI Server 进行 TCP/IP 通信，需要用相同的方法添加 Microsoft WinSock Control6.0(Version1.0)。



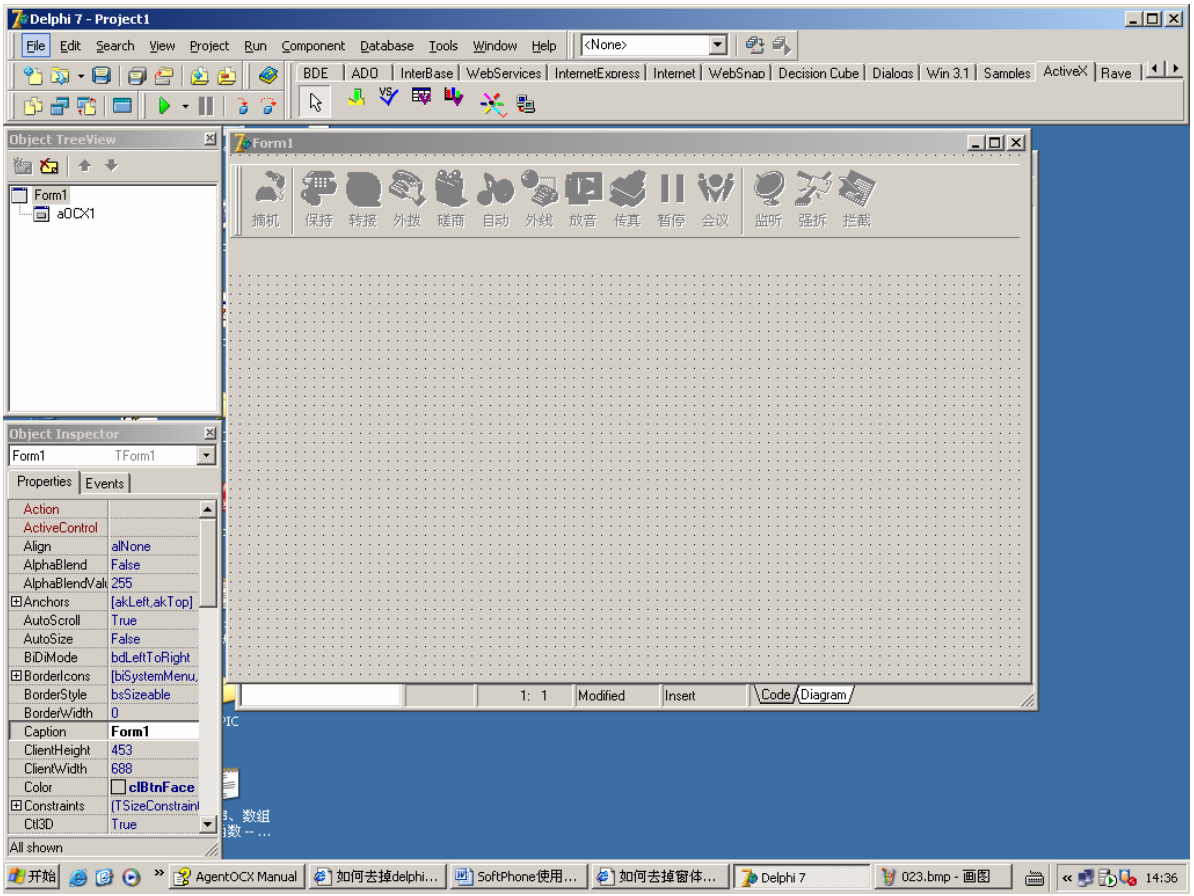
图表 52 选择 AgentOCX Component



图表 53

6.1.3 在界面上添加 OCX 控件

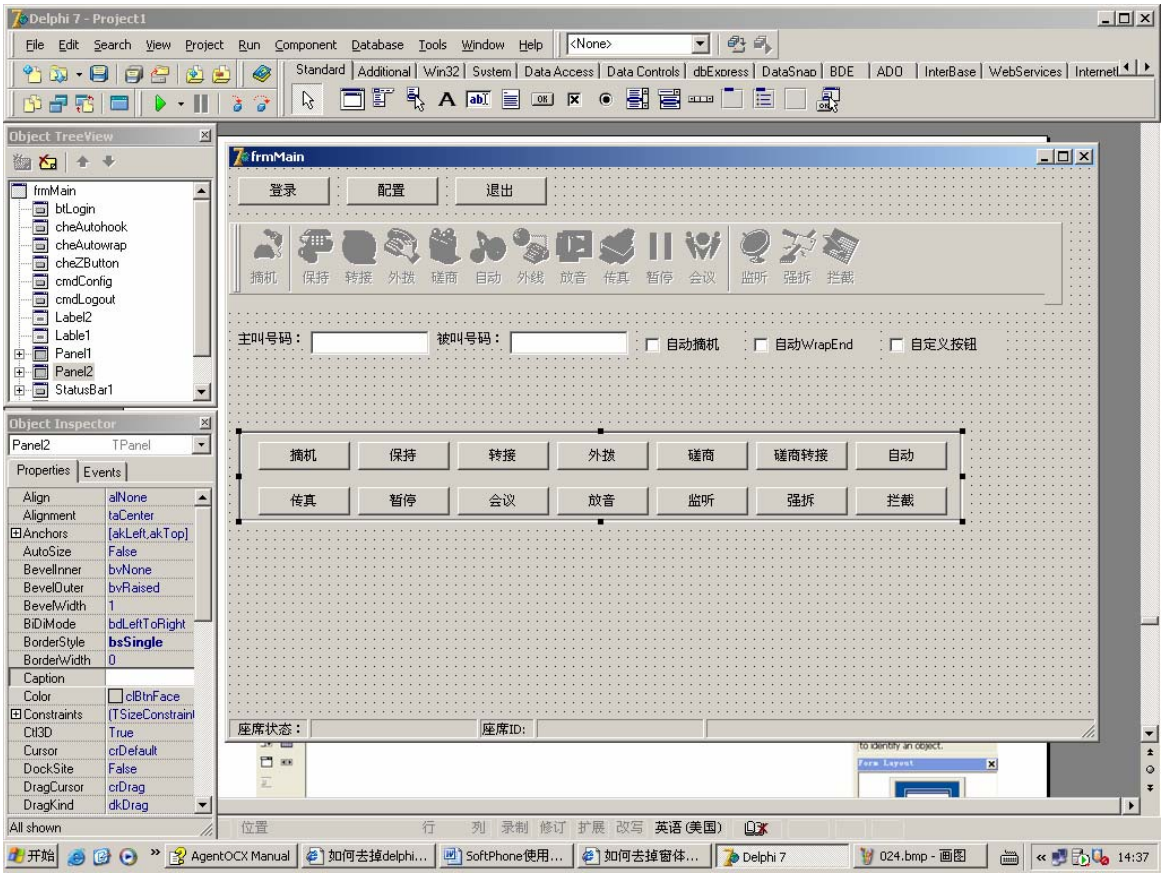
选中 AgentOCX 的图标，在 DELPHI 工程的 Form1 上画出控件的外形，为了方便整个控件的显示控制，也可将它放置在 Picturebo 控件上。如图：



图表 54 把 AgentOCX 贴在界面上

6.1.4 完成主界面

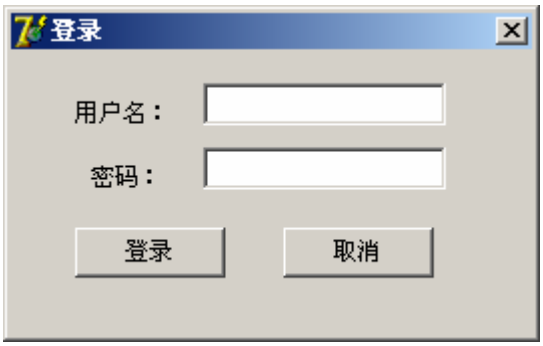
在界面上添加显示主叫和被叫的文本框及登录、注销按钮以及显示座席实时状态的状态条



图表 55 完成主界面

6.1.5 完成相关界面

分别作出登陆、外拨（外线、会议）、传真、转接、监听用的 Form



图表 56 登录对话框

图表 57 外拨对话框

图表 58 转接对话框

图表 59 监听对话框

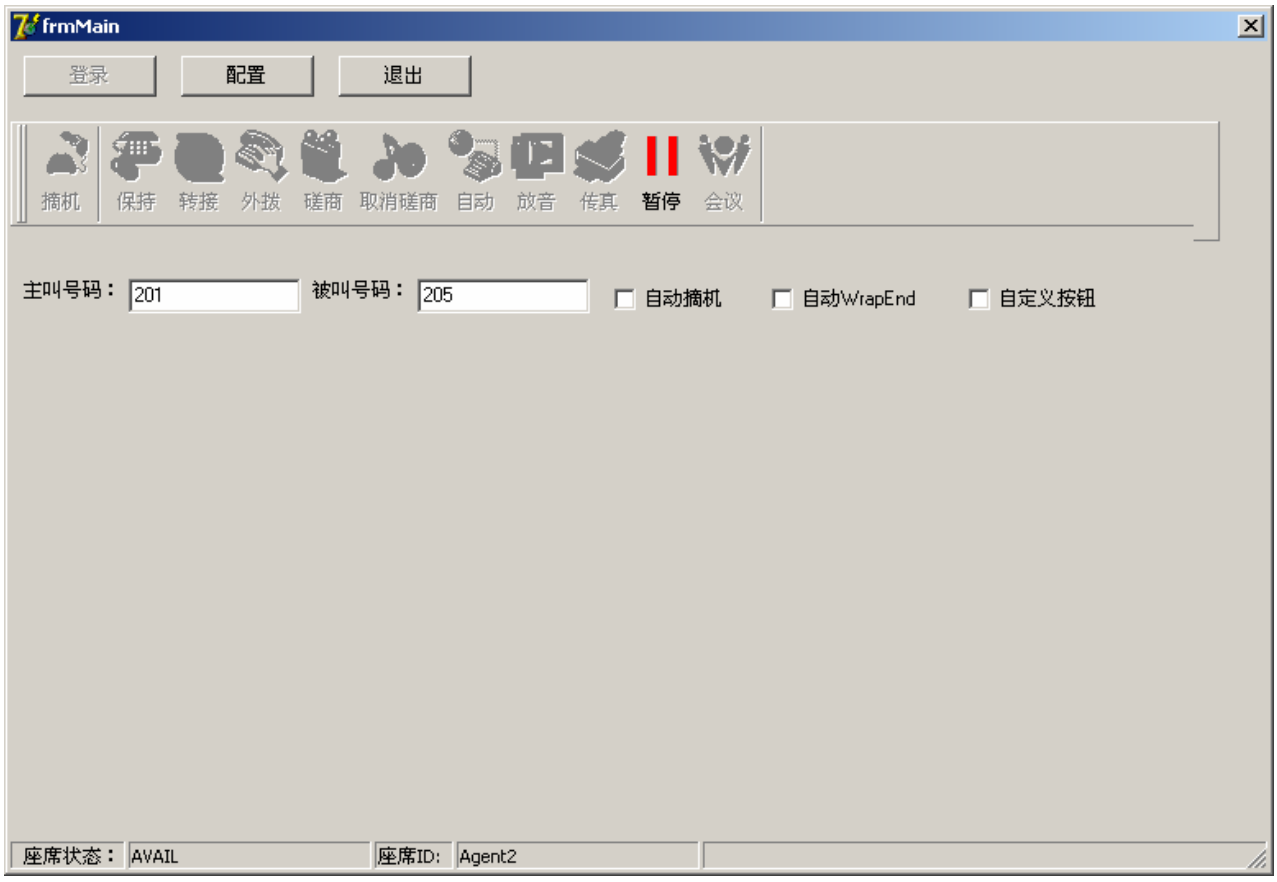
6.1.6 Login 的代码实现

界面的工作完成之后就可以进入编码的阶段了，在图表 56 所示的登录界面中，需要提供登录的 AgentName 和 Password，调用 OCX 控件的 Login 函数即可。代码如下：

```
procedure TfrmLogin.cmdLoginClick(Sender: TObject);
begin
      frmMain.StatusBar1.Panels[3].Text:=frmLogin.txtUser.Text;
      frmMain.aOCX1.AgentID:=frmLogin.txtUser.Text;
      frmMain.aOCX1.Password:=frmLogin.txtPassword.Text;
      frmMain.aOCX1.LogIn;
      frmLogin.Close;
end;
procedure TfrmLogin.cmdNLoginClick(Sender: TObject);
begin
        frmLogin.Close;
end;
```

6.1.7 暂停处理的实现

登录成功后的座席界面如图所示：

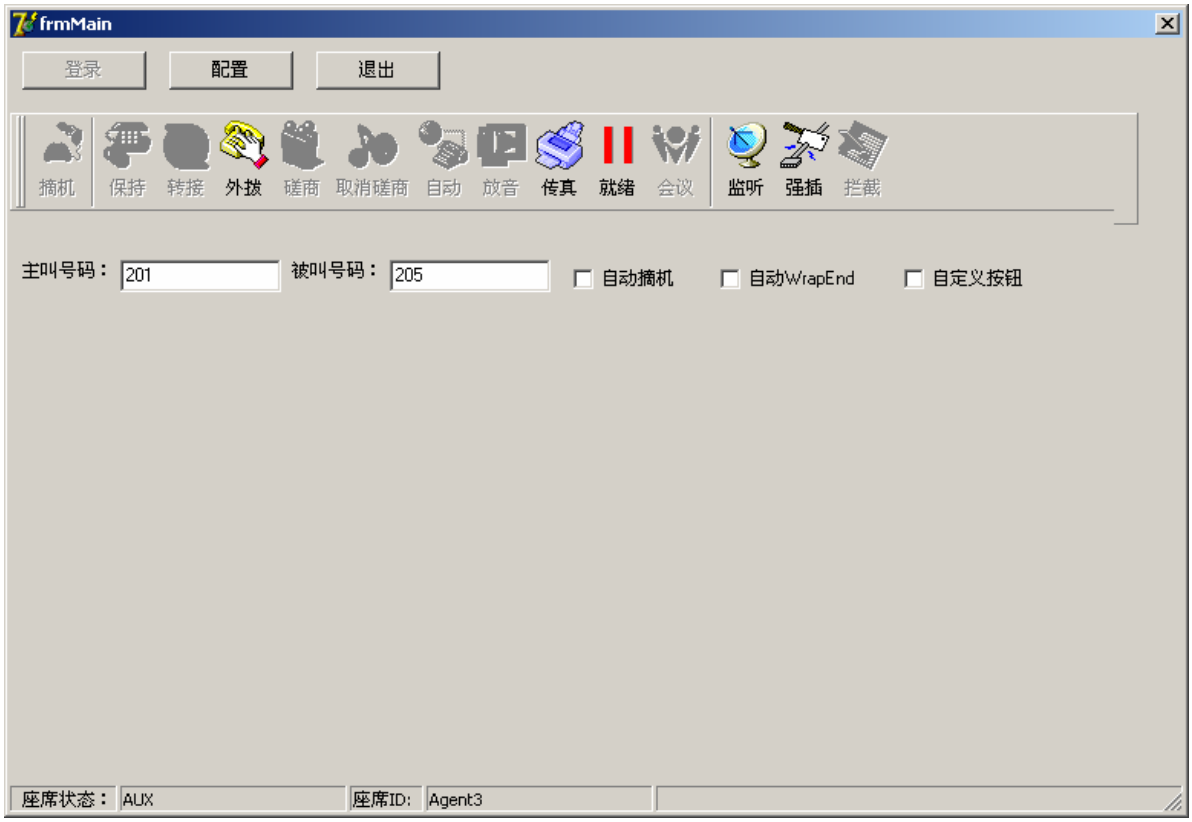


图表 60 登录成功主界面

座席登录成功后处于空闲状态，可以进行软电话的相关操作，如：传真、监听。

底下的状态条显示了当前的座席状态及登录座席的 ID 信息。

当座席点击暂停时，座席进入暂停状态：

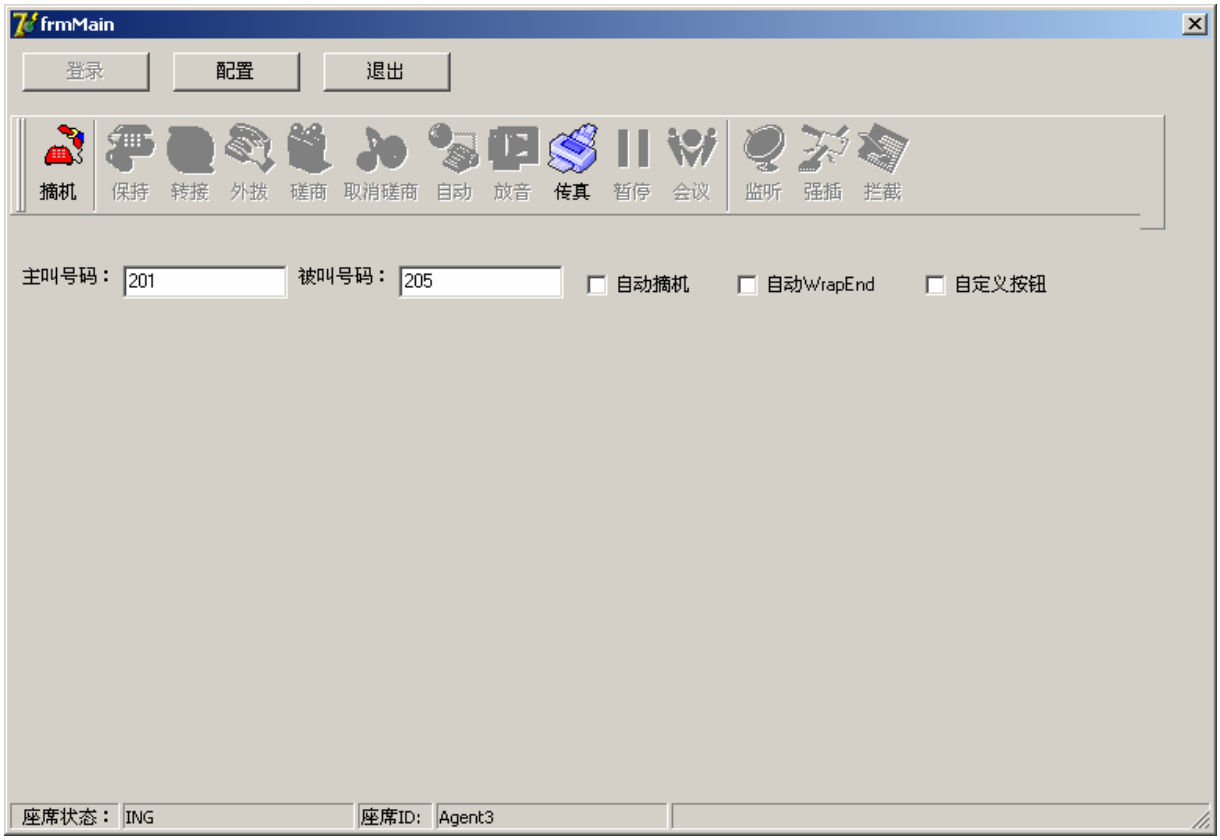


图表 61 暂停状态主界面

这时座席除了外拨之外不能进行其他任何操作，只有当恢复后进入服务状态才可以进行相关操作，这是为座席休息和话后处理提供的功能，不允许座席在空闲状态下发起外拨主要是为了避免座席在外拨的同时又电话进入而发生冲突。

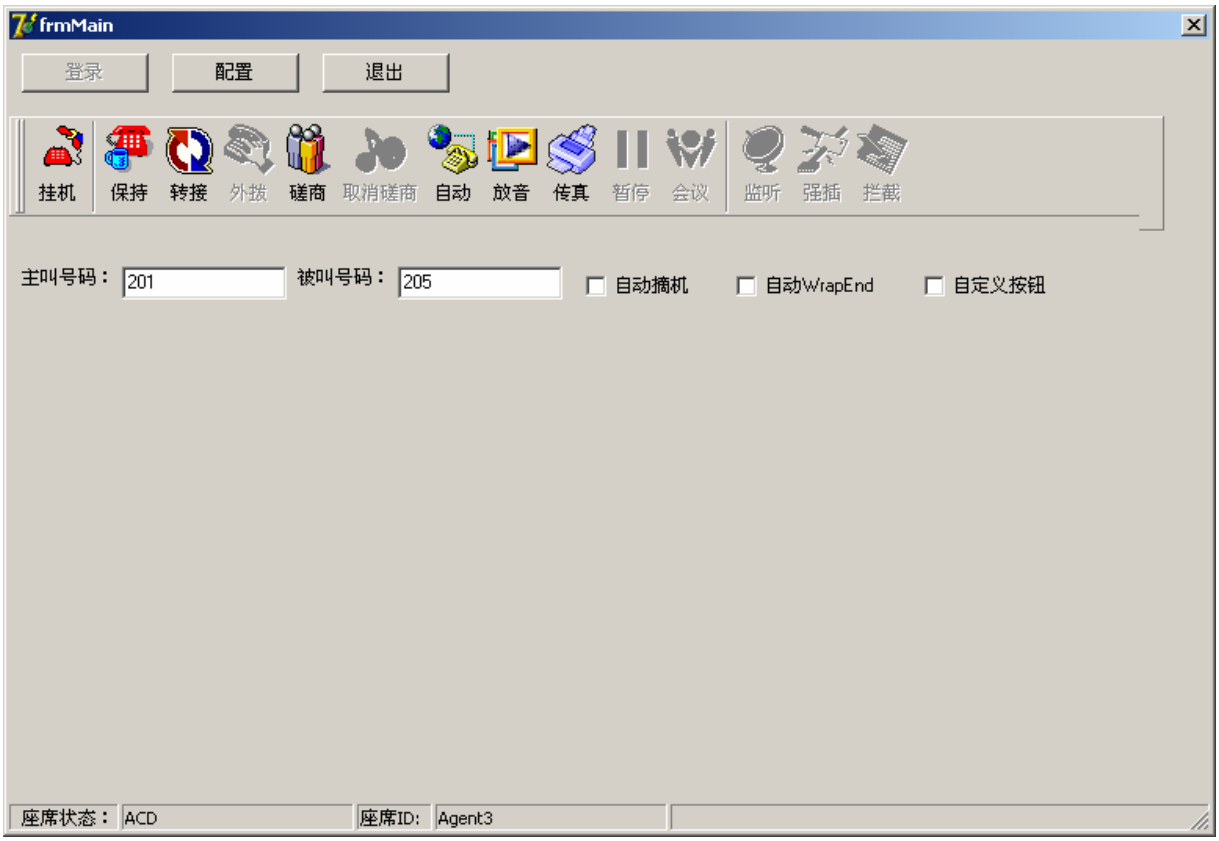
6.1.8 电话应答的实现

当有电话打到座席，座席的状态如图所示：



图表 62 呼叫进入中状态主界面

摘机后话机状态变为：



图表 63 摘机应答后的主界面

此时座席处于服务状态，MIS 端可以通过 CallArrive()函数获得通话的主叫和被叫。

代码如下：

```
procedure TfrmMain.aOCX1CallArrive(ASender: TObject; const ani, dnis,
   Data: WideString);
begin
   frmMain.txtANI.Text:='';
   frmMain.txtDNIS.Text:='';
   frmMain.txtANI.Text:=ani;
   frmMain.txtDNIS.Text:=dnis;
   if frmMain.cheAutohook.Checked=True then
      begin
         frmMain.aOCX1.CmdAnswer;
      end;
end;
```

此时座席可以进行转接、磋商、自动、会议操作，或挂断当前的通话，挂断当前通话话机进入话后状态，话后状态与暂停状态类似。

6.1.9 保持的实现：

保持当前的通话，给用户放音。用于座席的特殊情况，如查询其他资料等。

6.1.10 单步转接的实现：

即座席把当前通话转给同技能组的其他空闲座席

座席在服务状态时可以点击转接按钮，这时系统会自动查询目前登录的处于空闲状态的座席，并以列表的形式显示出来，座席要选择特定的空闲座席进行转接：

图表 64 转接界面

转接后座席转入话后状态，而被转接的座席相当于接收来话。

转接、磋商、监听所对应的事件返回的参数为 Agent 的列表，格式为 AgentName1=0;AgentName2=1;，需要 MIS 端解析该字串。在例程里有详尽的代码。

代码实现为:

1．模拟发送转接请求

```
procedure TfrmMain.cmdTrasnferClick(Sender: TObject);
begin
  if frmMain.cmdTrasnfer.Caption='转接' then
  begin
      frmMain.aOCX1.CmdTransfer;
  end
  else
  begin
      frmMain.aOCX1.CmdTransferStop;
  end;
end
```

2．获取当前处于空闲状态的座席以及对当前空闲坐席 Agentlist 字串进行解析，并在下拉列表里进行显示

```
procedure TfrmMain.aOCX1EVTTransfer(ASender: TObject;
  const AgentList: WideString);
var
```

```
  strAgent,strAgentID:string;
  i,pos1:Integer;
begin
  frmTransfer.cmbAgentID.Items.Clear;
  frmTransfer.Show;
  strAgent:=AgentList;
  i:=Length(strAgent);
  while i>0 do
  begin
    pos1:=Pos(';',strAgent);
    strAgentID:=MidStr(strAgent,1,pos1-3);
    frmTransfer.cmbAgentID.Items.Add(strAgentID) ;
    Delete(strAgent,1,pos1);
    i:=Length(strAgent);
  end;
end;
```

4．调用 CmdTransferToAgent 函数实现转接

```
procedure TfrmTransfer.btConfirmClick(Sender: TObject);
begin
   if frmTransfer.cmbAgentID.Text='' then
   begin
    ShowMessage('请您选择正确的座席');
   end
   else
   begin
    frmMain.aOCX1.CmdTransferToAgent(frmTransfer.cmbAgentID.Text,'','');
    frmTransfer.Close;
   end
end;
```

点击确定，调用 aOCX1.CmdTransferToAgent Combo1.Text, "", ""，确保所选的被转接座席不为空。

6.1.11 外拨的实现：

座席在暂停状态下可以点击外拨按钮，点击外拨按钮时，CTI Server 会上返一个 EVTDialOut 事件，MIS 系统在捕捉到这个事件后应该弹出所作的外拨界面，并调用 DialOut 函数发起外拨。如图所示：

图表 65 外拨界面

代码如下：

```
procedure TfrmMain.aOCX1EVTDialOut(Sender: TObject);
begin
  frmDialout.Caption:='外拨';
  frmDialout.Show;
end;
```

在一个 command 按钮的 click 事件中。

```
procedure TfrmDialout.btConfirmClick(Sender: TObject);
begin
  if frmDialout.EditPhone.Text='' then
    begin
      ShowMessage('请您输入电话号码！');
    end
  else
    begin
      if frmDialout.Caption='外拨' then     //外拨的实现
      begin
          frmMain.aOCX1.DialOut(Trim(frmDialout.EditPhone.Text));
      end
      else if frmDialout.Caption='会议' then
            begin
                frmMain.aOCX1.Conference(Trim(frmDialout.EditPhone.Text));
            end
            else
            begin
                frmMain.aOCX1.OutPhone(Trim(frmDialout.EditPhone.Text));
            end;
      frmDialout.Close;
    end;
```

```
end;
```

磋商的实现：

座席屏蔽当前的通话，同班长席、专家座席或其他座席磋商相关业务或其它事宜，磋商结束可以将通话转接也可以将通话恢复（其中获取当前空闲座席列表代码和获取转接座席列表代码一样）。

当座席处于服务状态（接听电话或外拨成功等）时，座席可以进行磋商操作，磋商的操作与转接的部分逻辑有些类似，点击磋商按钮系统会自动查询当前登录且处于空闲状态的座席，并以列表的形式显示出来，座席可以选择空闲的座席发起磋商请求。如图：

图表 66 磋商主界面

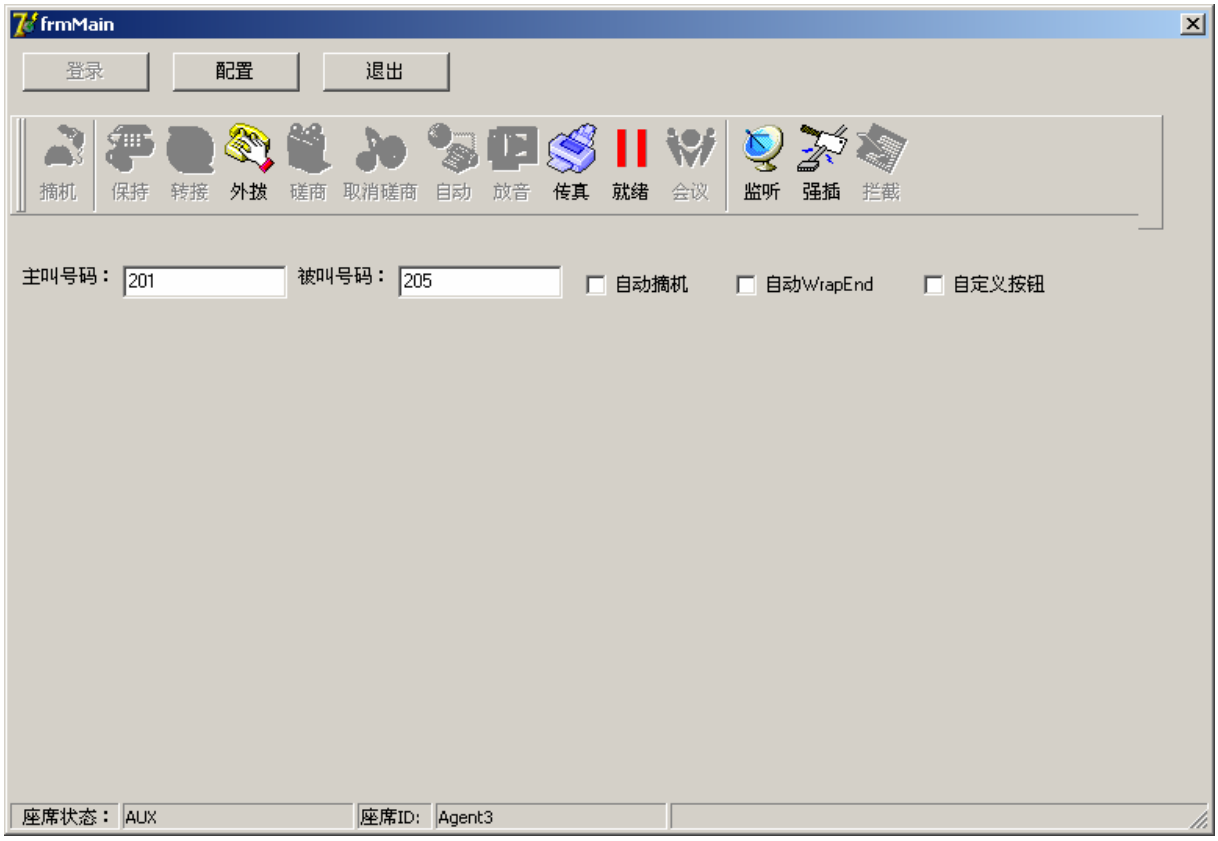
磋商分为内线磋商，外线磋商两类，内线磋商指的是在登录的座席之间发起的磋商，外线磋商是对于呼叫中心外部的外线之间进行磋商通话。

对于不同的磋商对象(外线、内线)要调用不同的函数，CmdConsultToAgent 用于内线磋商，CmdConsultToOutLine 用于外线磋商；取消磋商调用 CmdConsultCancel；在发起磋商以后，对方未摘机，可以点击用户界面上的停止来停止磋商。

磋商成功后，座席与被磋商座席通话，用户端播放音乐。当遇到难以解决的问题或一些特殊情况需要被磋商座席解决时，座席可以点击磋商转接按钮，将该通话转接到被磋商座席，或者直接点击会议按钮形成三方会议，或者座席点击结束磋商按钮重新接通被屏蔽的通话。

6.1.12 暂停的实现：

暂停座席，座席处于暂停状态，不能做任何业务，只有点击恢复才可重新进入空闲状态。如图：



图表 67 暂停状态界面

6.1.13 会议的实现：

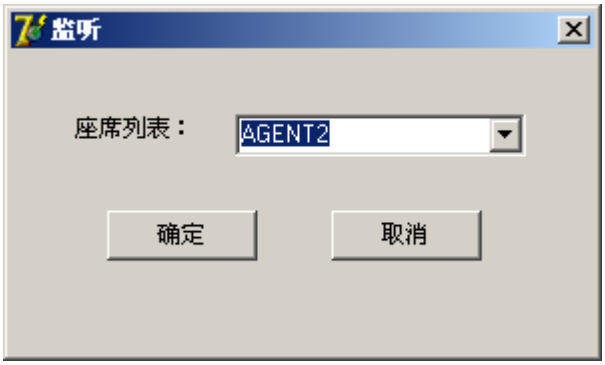
会议的功能目前主要为三方通话

当座席处于通话状态时可以将第三方加入到会议中，它的入口也必须为磋商，即将在磋商中的三方加入到会议中。

6.1.14 监听的实现：

班长或有较高权限的座席可以监听其他座席与用户的通话

当座席处于空闲状态时可以做监听操作，点击监听按钮后系统会自动查找目前已登陆的处于服务状态的座席，并以列表的形式显示出来（其中获取当前空闲座席列表代码和获取转接座席列表代码一样），座席可以选择座席发起监听请求。如图：



图表 68 监听界面

调用函数 CmdListenToAgent 发起监听，取消时调用 CmdListenStop。

监听成功后，座席进入监听状态，可以监听到被监听座席和客户的谈话，处于监听状态的座席还可以进行拦截和强拆操作。

6.1.15 强插的实现：

强行进入三方会议，由发起监听的座席发起强插功能，实现与被监听方的三方会议。

6.1.16 拦截的实现：

强行将被监听座席同客户的通话中断，并将通话拦截到本座席，本座席转入服务状态。

6.1.17 状态栏显示：

状态栏可以显示当前座席的实时状态，当前状态可以通过上返事件 EVTReturnStatus(ByValue rstatus As String)来获得，参数 rstatus 即为座席的状态值。

代码如下：

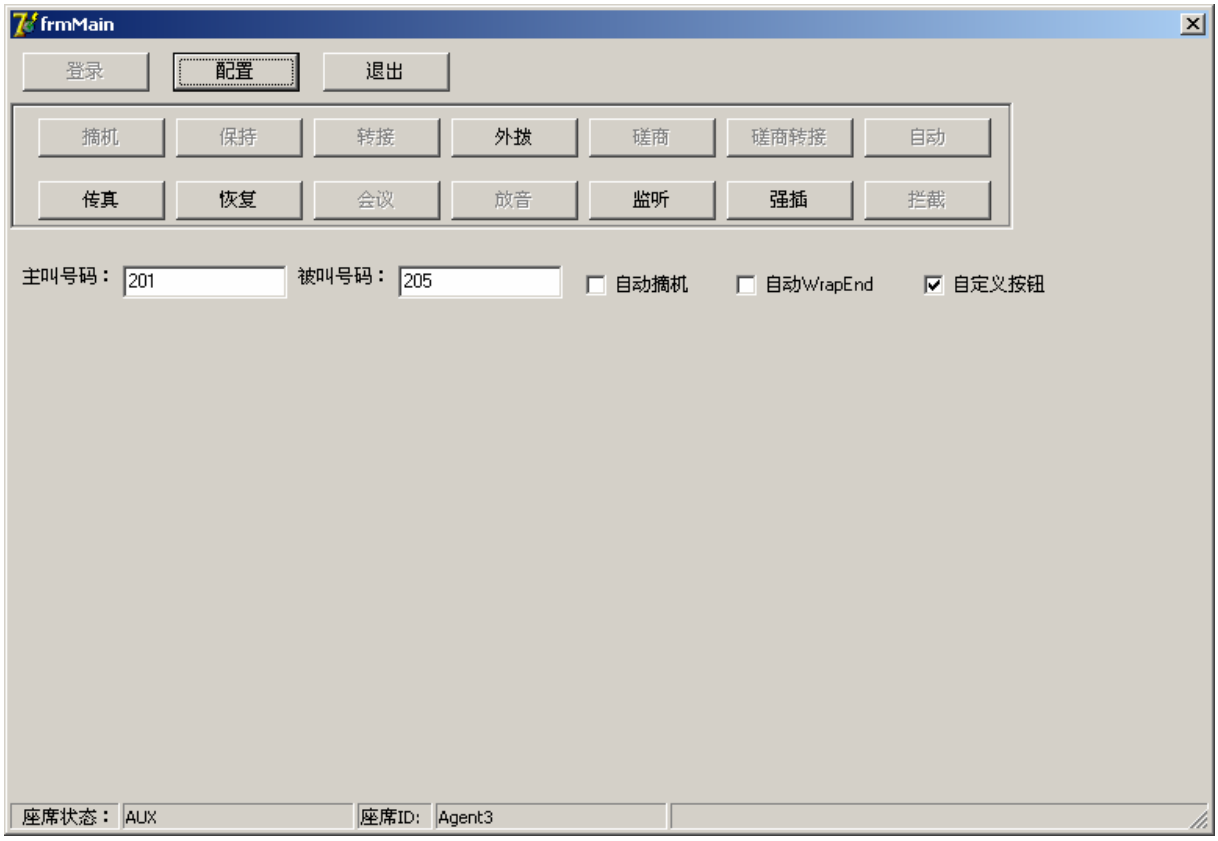
```
procedure TfrmMain.aOCX1EVTReturnStatus(ASender: TObject;
   const rstatus: WideString);
begin
   frmMain.StatusBar1.Panels[1].Text:=rstatus;
   if rstatus='连接断开' then
   begin
       frmMain.btLogin.Enabled:=true;
   end;
end;
```

6.1.18 注册表配置：

对于一些需要设置的注册表信息，系统提供可视化的借口给用户，用户可以通过调用 ShowConfig 函数来完成注册表信息的配置，点击 OK 按钮即可完成注册表信息的更新。

6.1.19 自定义座席按钮：

MIS 系统的代码编写人员可以自己定义座席控件的各个按钮，以达到界面风格的统一。如图：



图表 69 自定义按钮主界面

每个按钮对应一个特定的模拟向 CTI Server 发送消息的事件，系统可以根据 aOCX1_EVTButtonStatus() 的上返事件的值来确定个按钮的状态。代码如下:

```
procedure TfrmMain.cmdAutoClick(Sender: TObject);
begin
        frmMain.aOCX1.CmdConsultTransfer(",",",",");
end;

procedure TfrmMain.cmdConferenceClick(Sender: TObject);
begin
  If frmMain.cmdConference.Caption = '会议' Then
  begin
     frmMain.aOCX1.CmdConference;
  end
  else
  begin
     frmMain.aOCX1.CmdMakeCallStop;
  end;
end;
```

```
procedure TfrmMain.cmdConsultationClick(Sender: TObject);
begin
   If frmMain.cmdConsultation.Caption = '磋商' Then
   begin
       frmMain.aOCX1.CmdConsult;
   end
   Else if frmMain.cmdConsultation.Caption='磋商结束' then
          begin
              frmMain.aOCX1.CmdConsultStop;
          end
   else
   begin
       frmMain.aOCX1.CmdConsultStop;
   end;
end;

procedure TfrmMain.cmdDialoutClick(Sender: TObject);
begin
   If frmMain.cmdDialout.Caption = '外拨' Then
   begin
       frmMain.aOCX1.CmdDialOut;
   end
   Else
   begin
       frmMain.aOCX1.CmdMakeCallStop;
   end;
end;

procedure TfrmMain.cmdDisconnectClick(Sender: TObject);
begin
   frmMain.aOCX1.CmdIntrude;
end;

procedure TfrmMain.cmdFaxClick(Sender: TObject);
begin
   frmMain.aOCX1.CmdFax;
end;

procedure TfrmMain.cmdHoldClick(Sender: TObject);
begin
```

```
  If frmMain.cmdHold.Caption = '保持' Then
  begin
      frmMain.aOCX1.CmdHold;
  end
  Else
  begin
      frmMain.aOCX1.CmdHoldStop;
  end;
end;

procedure TfrmMain.cmdHookClick(Sender: TObject);
begin
       If frmMain.cmdHook.Caption = '摘机' Then
       begin
           frmMain.aOCX1.CmdAnswer;
       end
       Else if frmMain.cmdHook.Caption='接受' then
            begin
                frmMain.aOCX1.CmdConsultAnswer;
            end
       else
       begin
           frmMain.aOCX1.CmdOnHook;
       end;
end;

procedure TfrmMain.cmdListenClick(Sender: TObject);
begin
  if frmMain.cmdListen.Caption='监听' then
  begin
      frmMain.aOCX1.CmdListen;
  end
  else
  begin
      frmMain.aOCX1.CmdListenStop;
  end;
end;

procedure TfrmMain.cmdOutPhoneClick(Sender: TObject);
begin
```

```
    If frmMain.cmdOutPhone.Caption = '自动' Then
    begin
        frmMain.aOCX1.CmdAuto;
    end
    else
    begin
        frmMain.aOCX1.CmdAutoCancel;
    end;
end;

procedure TfrmMain.cmdPauseClick(Sender: TObject);
begin
  if frmMain.cmdPause.Caption='暂停' then
  begin
      frmMain.aOCX1.CmdPause;
  end
  else
  begin
      frmMain.aOCX1.CmdContinue;
  end;
end;

procedure TfrmMain.cmdPlayClick(Sender: TObject);
begin
  if frmMain.cmdPlay.Caption='放音' then
  begin
      frmMain.aOCX1.CmdPlay;
  end
  else
  begin
      frmMain.aOCX1.CmdPlayStop;
  end;
end;

procedure TfrmMain.cmdRopcallClick(Sender: TObject);
begin
  frmMain.aOCX1.CmdRopCall;
end;

procedure TfrmMain.cmdTrasnferClick(Sender: TObject);
begin
```

```
  if frmMain.cmdTrasnfer.Caption='转接' then
  begin
      frmMain.aOCX1.CmdTransfer;
  end
  else
  begin
      frmMain.aOCX1.CmdTransferStop;
  end;
end;
procedure SetButtonOption(sTitle:Widestring;strEnable:integer;strButton:TButton);
  var temp:Boolean;
begin
  strButton.Caption:=sTitle;
  if strEnable>0 then
     begin
       temp:=True;
     end
  else
     begin
       temp:=False;
     end;
  strButton.Enabled:=temp;
end;

procedure TfrmMain.aOCX1EVTButtonStatus(ASender: TObject;const    Name,
  sTitle: WideString; Enable: Integer);
  var sTitle1:WideString;
begin
      begin
      if sTitle='OnHook' Then
      sTitle1:='挂机'
      else if sTitle='OffHook' then
      sTitle1:='摘机'
      else if sTitle='HOOKOFFCONSULT' then
      sTitle1:='接受'
      else if sTitle='Hold' then
      sTitle1:='保持'
      else if sTitle='HoldCancel' then
      sTitle1:='取消'
      else if sTitle='Transfer' then
```

```
sTitle1:='转接'
else if sTitle='CancelTransfer' then
sTitle1:='取消'
else if sTitle='DialOut' then
sTitle1:='外拨'
else if sTitle='CancelDialOut' then
sTitle1:='取消'
else if sTitle='Consultation' then
sTitle1:='磋商'
else if sTitle='CancelConsultation' then
sTitle1:='取消'
else if sTitle='StopConsultation' then
sTitle1:='磋商结束'
else if sTitle='ConsultTransfer' then
sTitle1:='磋商转接'
else if sTitle='Auto' then
sTitle1:='磋商转接'
else if sTitle='OutPhone' then
sTitle1:='自动'
else if sTitle='CancelOutPhone' then
sTitle1:='取消'
else if sTitle='Play' then
sTitle1:='放音'
else if sTitle='PlayCancel' then
sTitle1:='结束'
else if sTitle='Fax' then
sTitle1:='传真'
else if sTitle='FaxStop' then
sTitle1:='结束'
else if sTitle='Pause' then
sTitle1:='暂停'
else if sTitle='Continue' then
sTitle1:='恢复'
else if sTitle='ContinueDialTask' then
sTitle1:='放弃回访'
```

```
else if sTitle='Listen' then
sTitle1:='监听'
else if sTitle='CancelListen' then
sTitle1:='结束'
else if sTitle='Disconnect' then
sTitle1:='强插'
else if sTitle='Conference' then
sTitle1:='会议'
else if sTitle='CancelConference' then
sTitle1:='取消'
else if sTitle='RopCall' then
sTitle1:='拦截'
else if sTitle='LOGINSUCC' then
sTitle1:='登录成功'
else if sTitle='LOGINFAIL' then
sTitle1:='登录失败'
else if sTitle='TRANSFERFAIL' then
sTitle1:='转接失败'
end;

begin
if Name='Hook' then
   SetButtonOption(sTitle1, Enable, frmMain.cmdHook)
else if Name='Hold' then
   SetButtonOption(sTitle1, Enable, frmMain.cmdHold)
else if Name='Transfer' then
   SetButtonOption(sTitle1, Enable, frmMain.cmdTrasnfer)
else if Name='DialOut' then
   SetButtonOption(sTitle1, Enable, frmMain.cmdDialout)
else if Name='Consultation' then
   SetButtonOption(sTitle1, Enable, frmMain.cmdConsultation)
else if Name='Auto' then
   SetButtonOption(sTitle1, Enable, frmMain.cmdAuto)
else if Name='OutPhone' then
   SetButtonOption(sTitle1, Enable, frmMain.cmdOutPhone)
else if Name='Fax' then
   SetButtonOption(sTitle1, Enable, frmMain.cmdFax)
else if Name='Pause' then
   SetButtonOption(sTitle1, Enable, frmMain.cmdPause)
```

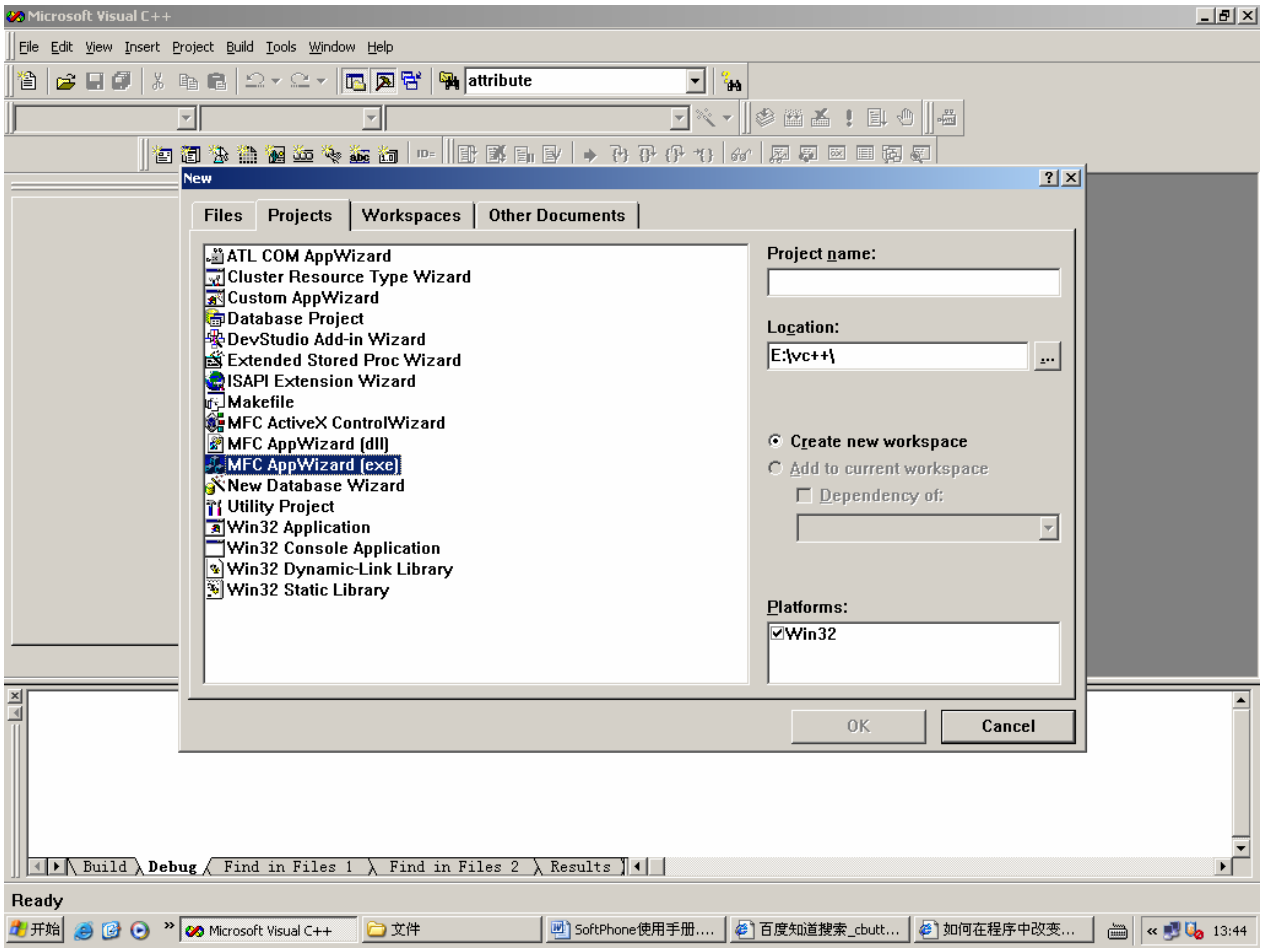
```
        else if Name='Conference' then
          SetButtonOption(sTitle1, Enable, frmMain.cmdConference)
        else if Name='Play' then
          SetButtonOption(sTitle1, Enable, frmMain.cmdPlay)
        else if Name='Listen' then
          SetButtonOption(sTitle1, Enable, frmMain.cmdListen)
        else if Name='Disconnect' then
          SetButtonOption(sTitle1, Enable, frmMain.cmdDisconnect)
        else if Name='RopCall' then
          SetButtonOption(sTitle1, Enable, frmMain.cmdRopcall)
        end;
end;
```

7. 基于 VisualC++6.0 的 DEMO 程序的搭建

7.1 程序搭建过程

7.1.1 创建工程

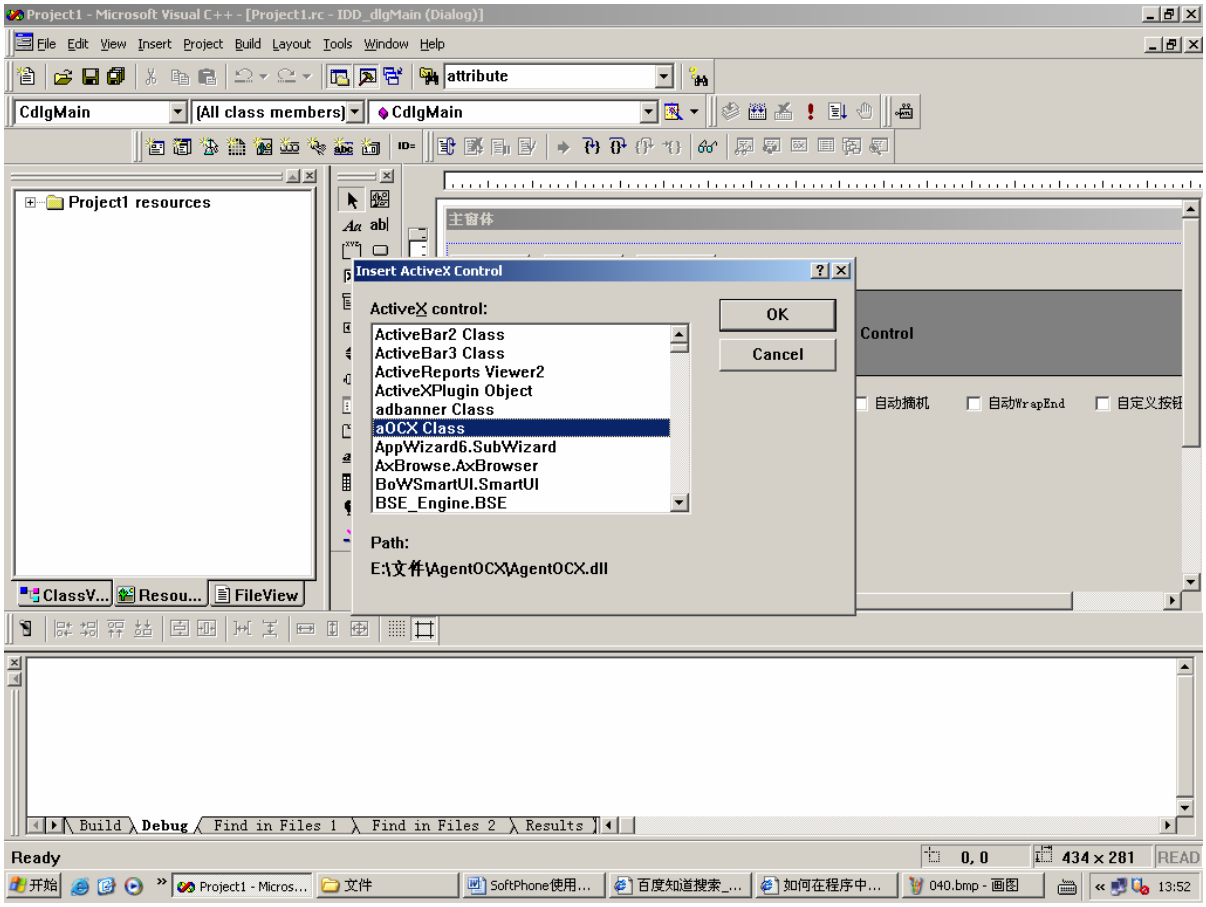
首先要将所提供的 AgentOCX.dll 注册，然后创建 VC++项目，选择 MFC AppWizard 应用程序。



图表 70 C#程序搭建选择 Windows 应用程序

7.1.2 在工程中添加 OCX 控件

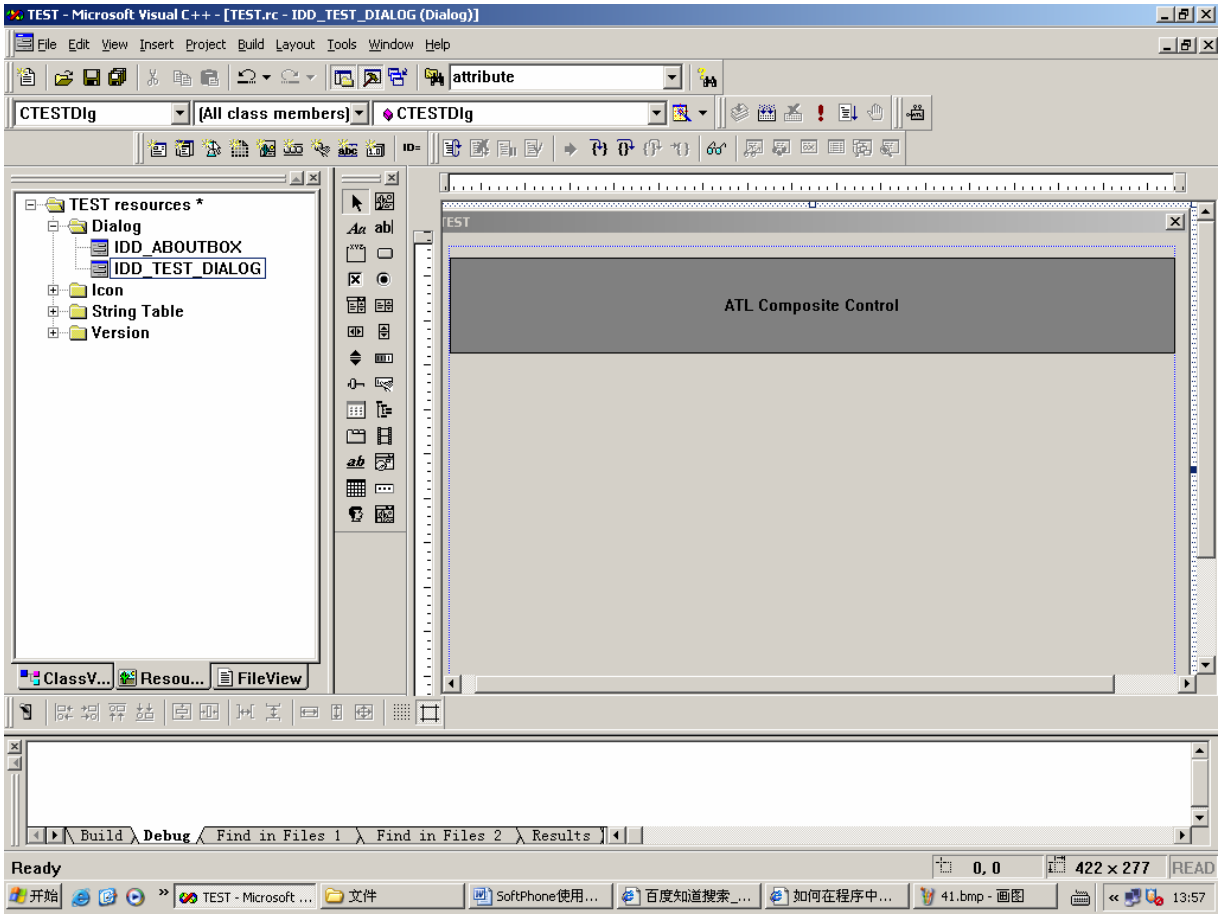
在 VisualC++工程的工某个对话框（Dialog）上右键选 Insert ActiveX Control，弹出 ActiveX 控件插入窗体，选择 aOCX Class 项，在 VisualC++工程工具箱中会显示 OCX 控件的图标:



图表 71 选择 aOCX Class 项

7.1.3 在界面上添加 OCX 控件

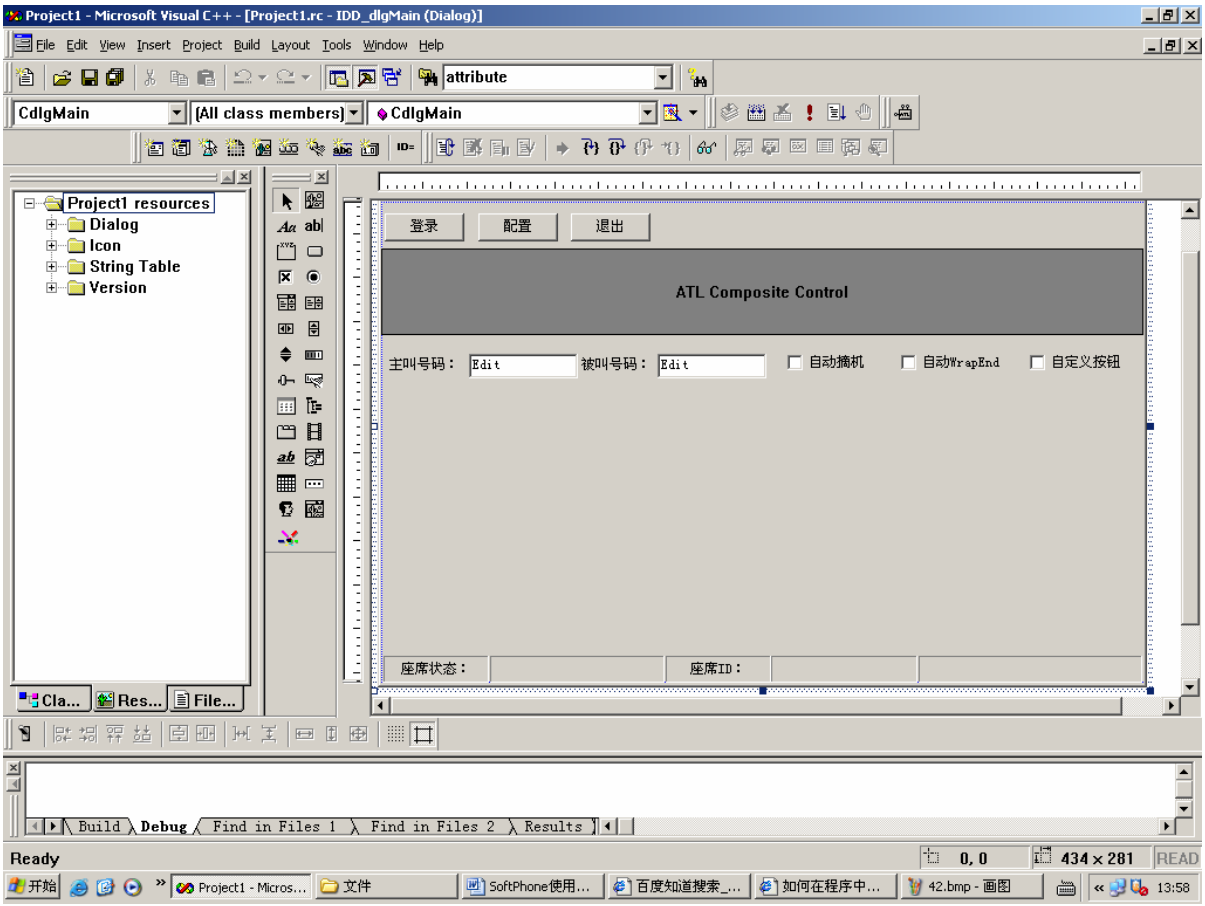
选中 AgentOCX 的图标，在 VisualC++工程的 Diallog1 上画出控件的外形，为了方便整个控件的显示控制，也可将它放置在 Picturebox 控件上。如图：



图表 72 把 AgentOCX 贴在界面上

7.1.4 完成主界面

在界面上添加显示主叫和被叫的文本框及登录、注销按钮以及显示座席实时状态的状态条



图表 73 完成主界面

7.1.5 完成相关界面

分别作出登陆、外拨（外线、会议）、传真、转接、监听用的 Form

图表 74 登录对话框

图表 75 外拨对话框

图表 76 转接对话框

图表 77 监听对话框

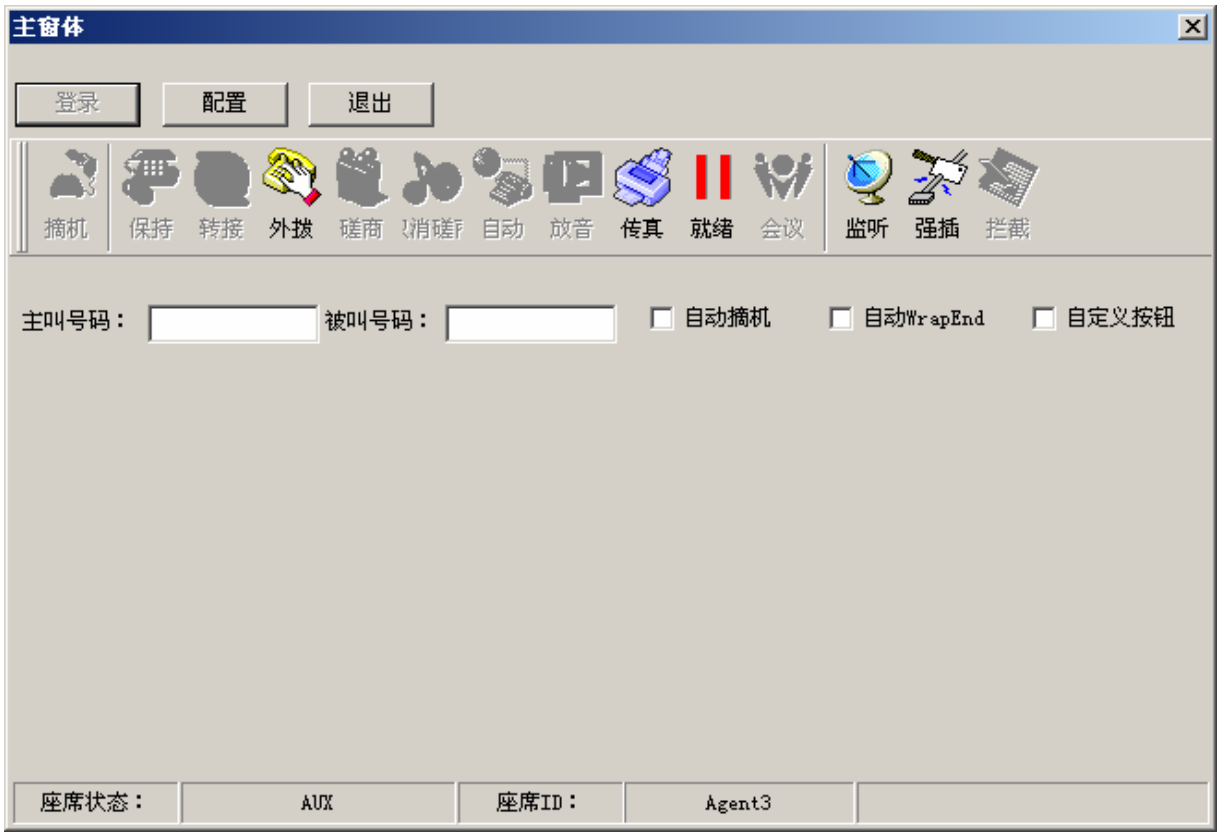
7.1.6 Login 的代码实现

界面的工作完成之后就可以进入编码的阶段了，在图表 16 所示的登录界面中，需要提供登录的 AgentName 和 Password，调用 OCX 控件的 Login 函数即可。代码如下：

```
void CdlgLogin::OnbtCancel()
{
   this->OnCancel();
}
void CdlgLogin::OnbtConfirm()
{
   this->UpdateData(true);
   if(this->usr.IsEmpty())
   {
      this->MessageBox("请您输入用户名！","警告",MB_OK|MB_ICONQUESTION);
       return;
   }
   if(this->pwd.IsEmpty())
   {
      this->MessageBox("请您输入密码！","警告",MB_OK|MB_ICONQUESTION);
       return;
   }


   this->dlgMain->AgentOCX.SetAgentID(usr);
   this->dlgMain->AgentOCX.SetPassword(pwd);
   this->dlgMain->AgentOCX.GetAgentID();
   this->dlgMain->AgentOCX.GetPassword();
   this->dlgMain->AgentOCX.LogIn();
   this->dlgMain->static2=usr;
   this->dlgMain->UpdateData(false);
   this->OnCancel();
}
```

暂停处理的实现

登录成功后的座席界面如图所示：

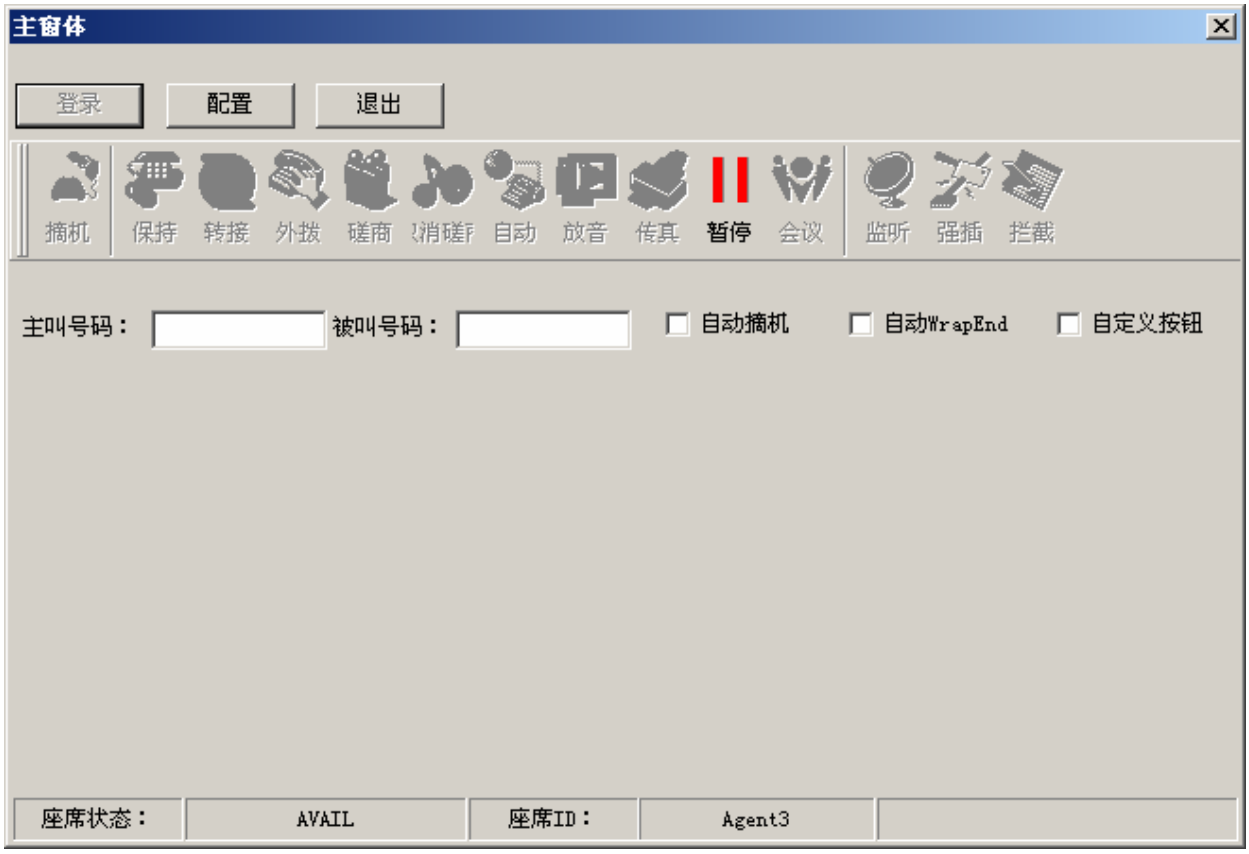


图表 78 登录成功主界面

座席登录成功后处于空闲状态，可以进行软电话的相关操作，如：传真、监听。

底下的状态条显示了当前的座席状态及登录座席的 ID 信息。

当座席点击暂停时，座席进入暂停状态：

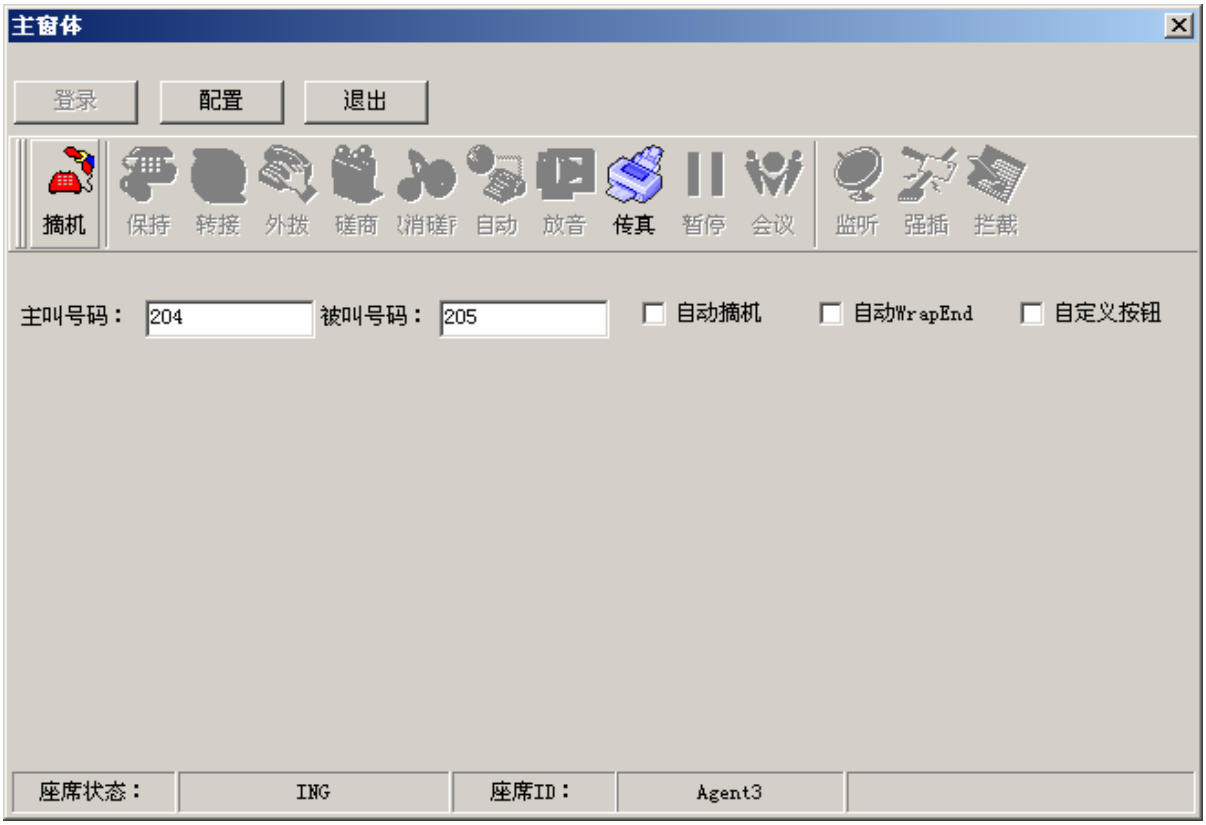


图表 79 暂停状态主界面

这时座席除了外拨之外不能进行其他任何操作，只有当恢复后进入服务状态才可以进行相关操作，这是为座席休息和话后处理提供的功能，不允许座席在空闲状态下发起外拨主要是为了避免座席在外拨的同时又电话进入而发生冲突。

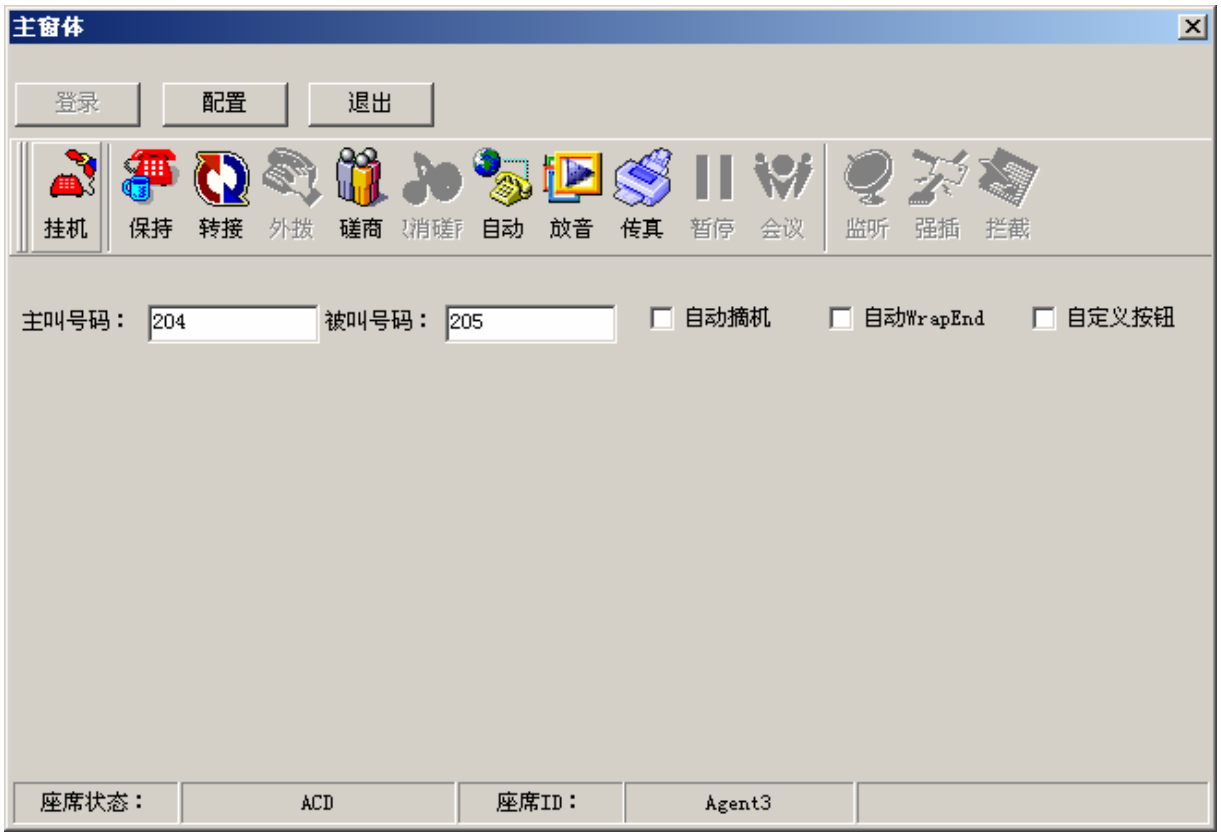
7.1.7 电话应答的实现

当有电话打到座席，座席的状态如图所示：



图表 80 呼叫进入中状态主界面

摘机后话机状态变为：



图表 81 摘机应答后的主界面

此时座席处于服务状态，MIS 端可以通过 CallArrive()函数获得通话的主叫和被叫。

代码如下：

```
void CdlgMain::OnCallArriveAocx1(LPCTSTR ani, LPCTSTR dnis, LPCTSTR Data)
{
	this->UpdateData(true);
	this->edANI="";
	this->edDNIS="";
	this->edANI=ani;
	this->edDNIS=dnis;

	if(normal1)
	{
	   this->AgentOCX.CmdAnswer();
	}
	this->UpdateData(false);
}
```

此时座席可以进行转接、磋商、自动、会议操作，或挂断当前的通话，挂断当前通话话机进入话后状态，话后状态与暂停状态类似。

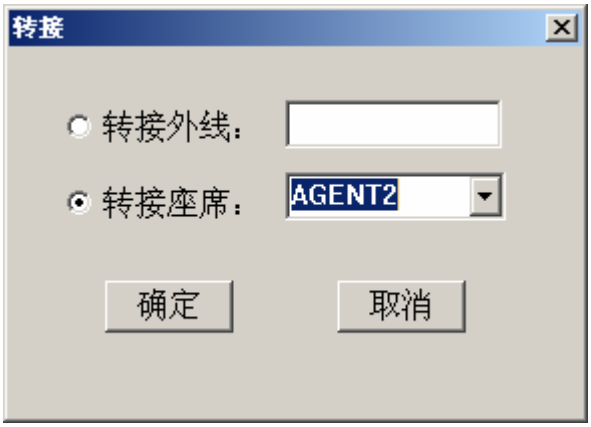
7.1.8 保持的实现：

保持当前的通话，给用户放音。用于座席的特殊情况，如查询其他资料等。

7.1.9 单步转接的实现：

即座席把当前通话转给同技能组的其他空闲座席

座席在服务状态时可以点击转接按钮，这时系统会自动查询目前登录的处于空闲状态的座席，并以列表的形式显示出来，座席要选择特定的空闲座席进行转接：



图表 82 转接界面

转接后座席转入话后状态，而被转接的座席相当于接收来话。

转接、磋商、监听所对应的事件返回的参数为 Agent 的列表，格式为 AgentName1=0;AgentName2=1;，需要 MIS 端解析该字串。在例程里有详尽的代码。

代码实现为:

1．模拟发送转接请求

```
void CdlgMain::OnTransfer()
{
	CString str;
	this->m_Transfer.GetWindowText(str);
	if(str=="转接")
	{
		this->AgentOCX.CmdTransfer();
	}
	else
	{
```

```
            this->AgentOCX.CmdTransferStop();
        }



}
```

2．获取当前处于空闲状态的座席以及对当前空闲坐席 Agentlist 字串进行解析，并在下拉列表里进行显示

```
void CdlgMain::OnEVTTransferAocx1(LPCTSTR AgentList)
{
        CdlgTransfer dlgTransfer;
        dlgTransfer.dlgMain=this;
        dlgTransfer.AgentList=AgentList;
        dlgTransfer.DoModal();


}
BOOL CdlgTransfer::OnInitDialog()
{
        CDialog::OnInitDialog();

       CString strAgentID;
        int i,pos;
        i=AgentList.GetLength()-1;
        while(i)
        {
            pos=AgentList.Find(";",0);
            strAgentID=AgentList.Mid(0,pos-2);
           this->cmbTransfer.AddString(strAgentID);
            AgentList=AgentList.Delete(0,pos+1);
            i=AgentList.GetLength();
           i--;
        }
        return TRUE;    // return TRUE unless you set the focus to a control
                        // EXCEPTION: OCX Property Pages should return FALSE
}
```

3．调用 CmdTransferToAgent 函数实现转接

```
            void CdlgTransfer::OnbtTConfirm()
{
       CString str;
        this->cmbTransfer.GetWindowText(str);
        if(rdOutphone)
        {
            if(str=="")
```

```
        {
            MessageBox("请您选择正确的座席！","提示",MB_OK);
             return;
        }
        this->dlgMain->AgentOCX.CmdTransferToAgent(str,"","","");
    }
    else
    {
       this->UpdateData(true);
       if(this->edTransfer="")
       {
           MessageBox("请您输入电话号码！","提示",MB_OK);
            return;
       }
       this->dlgMain->AgentOCX.CmdTransferToAgent("$"+edTransfer,"","","");
    }
    this->OnCancel();
}
```

点击确定，调用 AgentOCX.CmdTransferToAgent(str,"","","")，确保所选的被转接座席不为空。

7.1.10 外拨的实现：

座席在暂停状态下可以点击外拨按钮，点击外拨按钮时，CTI Server 会上返一个 EVTDialOut 事件，MIS 系统在捕捉到这个事件后应该弹出所作的外拨界面，并调用 DialOut 函数发起外拨。如图所示：

图表 83 外拨界面

代码如下：

```
void CdlgMain::OnEVTDialOutAocx1()
{
    CdlgOutdial dlgOutdial;
    dlgOutdial.dlgMain=this;
```

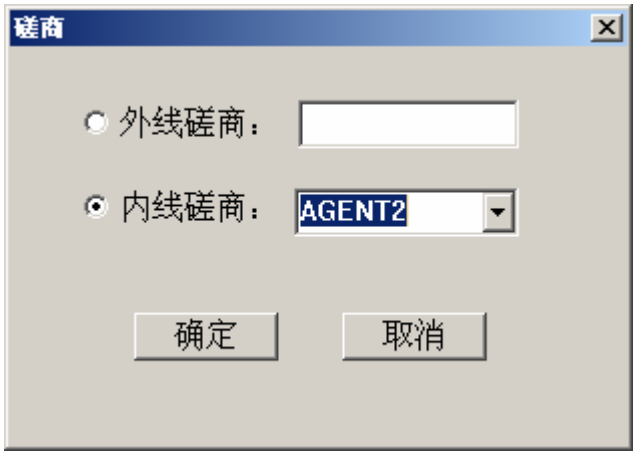
```
        dlgOutdial.OSign=1;
        dlgOutdial.DoModal();
}


 void CdlgOutdial::OnbtoConfrim()
{
        CString str;
        this->UpdateData(true);
        if(this->edNum="")
        {
            MessageBox("请输入电话号码！","提示",MB_OK);
            return;
        }
        this->GetWindowText(str);
        if(str=="外拨")
        {
            this->dlgMain->AgentOCX.DialOut(str);
        }
        else if(str=="外线")
        {
            this->dlgMain->AgentOCX.OutPhone(str);
        }
        else if(str=="会议")
        {
            this->dlgMain->AgentOCX.Conference(str);
        }
        this->OnCancel();

}
void CdlgOutdial::OnbtoCancel()
{
        CString str;
        this->GetWindowText(str);
        if(str=="会议")
        {
           this->dlgMain->AgentOCX.CancelConference();
        }
        this->OnCancel();
```

```
}
```

7.1.11 磋商的实现：

座席屏蔽当前的通话，同班长席、专家座席或其他座席磋商相关业务或其它事宜，磋商结束可以将通话转接也可以将通话恢复（其中获取当前空闲座席列表代码和获取转接座席列表代码一样）。

当座席处于服务状态（接听电话或外拨成功等）时，座席可以进行磋商操作，磋商的操作与转接的部分逻辑有些类似，点击磋商按钮系统会自动查询当前登录且处于空闲状态的座席，并以列表的形式显示出来，座席可以选择空闲的座席发起磋商请求。如图：



图表 84 磋商主界面

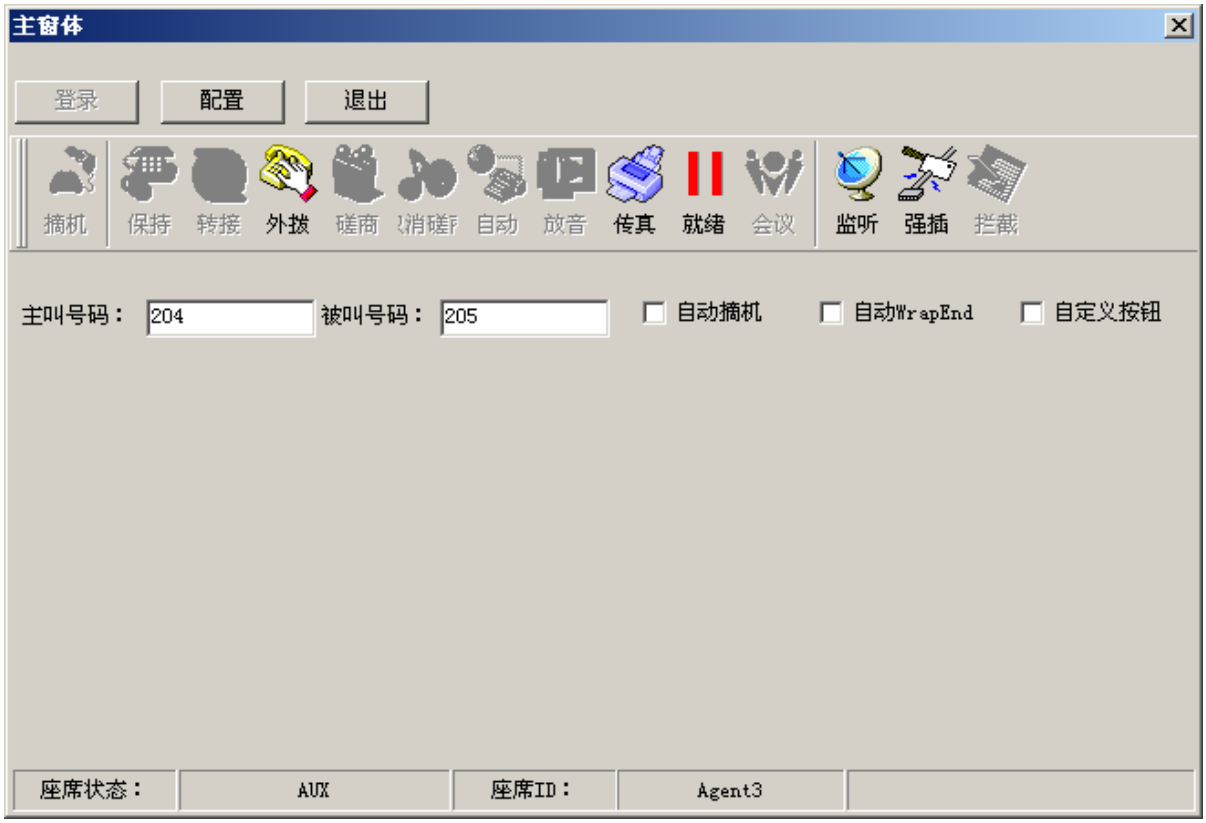
磋商分为内线磋商，外线磋商两类，内线磋商指的是在登录的座席之间发起的磋商，外线磋商是对于呼叫中心外部的外线之间进行磋商通话。

对于不同的磋商对象(外线、内线)要调用不同的函数，CmdConsultToAgent 用于内线磋商，CmdConsultToOutLine 用于外线磋商；取消磋商调用 CmdConsultCancel；在发起磋商以后，对方未摘机，可以点击用户界面上的停止来停止磋商。

磋商成功后，座席与被磋商座席通话，用户端播放音乐。当遇到难以解决的问题或一些特殊情况需要被磋商座席解决时，座席可以点击磋商转接按钮，将该通话转接到被磋商座席，或者直接点击会议按钮形成三方会议，或者座席点击结束磋商按钮重新接通被屏蔽的通话。

7.1.12 暂停的实现：

暂停座席，座席处于暂停状态，不能做任何业务，只有点击恢复才可重新进入空闲状态。如图：



图表 85 暂停状态界面

7.1.13 会议的实现：

会议的功能目前主要为三方通话

当座席处于通话状态时可以将第三方加入到会议中，它的入口也必须为磋商，即将在磋商中的三方加入到会议中。

7.1.14 监听的实现：

班长或有较高权限的座席可以监听其他座席与用户的通话

当座席处于空闲状态时可以做监听操作，点击监听按钮后系统会自动查找目前已登陆的处于服务状态的座席，并以列表的形式显示出来（其中获取当前空闲座席列表代码和获取转接座席列表代码一样），座席可以选择座席发起监听请求。如图：

图表 86 监听界面

调用函数 CmdListenToAgent 发起监听，取消时调用 CmdListenStop。

监听成功后，座席进入监听状态，可以监听到被监听座席和客户的谈话，处于监听状态的座席还可以进行拦截和强拆操作。

7.1.15 强插的实现：

强行进入三方会议，由发起监听的座席发起强插功能，实现与被监听方的三方会议。

7.1.16 拦截的实现：

强行将被监听座席同客户的通话中断，并将通话拦截到本座席，本座席转入服务状态。

7.1.17 状态标签 CStatic 显示：

状态栏可以显示当前座席的实时状态，当前状态可以通过上返事件 EVTReturnStatus(ByValue rstatus As String)来获得，参数 rstatus 即为座席的状态值。

代码如下：

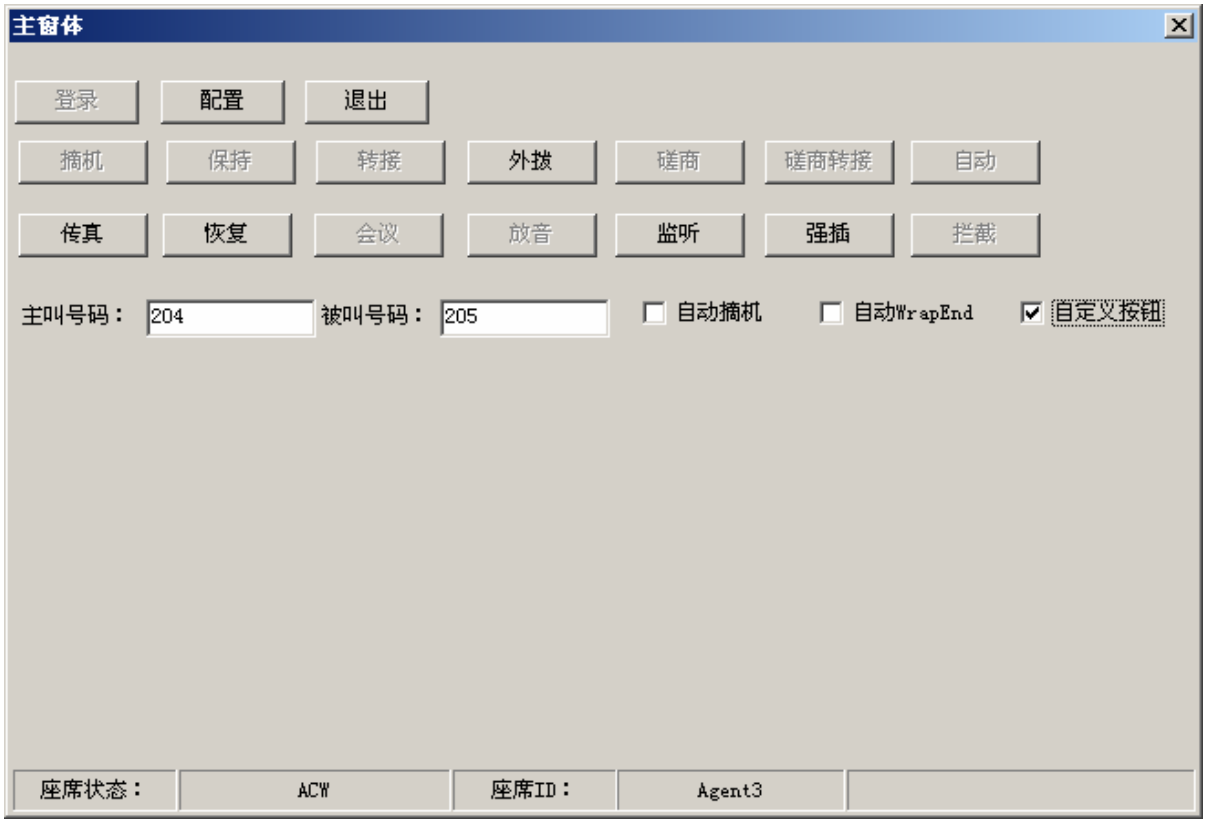
```
void CdlgMain::OnEVTReturnStatusAocx1(LPCTSTR rstatus)
{

    this->static1=rstatus;
    this->UpdateData(false);
    if(rstatus=="连接断开")
    {
        this->btLogin.EnableWindow(true);
    }

}
```

7.1.18 注册表配置：

对于一些需要设置的注册表信息，系统提供可视化的借口给用户，用户可以通过调用 ShowConfig 函数来完成注册表信息的配置，点击 OK 按钮即可完成注册表信息的更新。

7.1.19 自定义座席按钮：

MIS 系统的代码编写人员可以自己定义座席控件的各个按钮，以达到界面风格的统一。如图：



图表 87 自定义按钮主界面

每个按钮对应一个特定的模拟向 CTI Server 发送消息的事件，系统可以根据 aOCX1_EVTButtonStatus() 的上返事件的值来确定个按钮的状态。代码如下:

```
  void CdlgMain::OnHook()
{
	CString str;
	this->m_Hook.GetWindowText(str);
	if(str=="摘机")
	{
		this->AgentOCX.CmdAnswer();
	}
	else if(str=="接受")
	{
		this->AgentOCX.CmdConsultAnswer();
	}
	else
	{
		this->AgentOCX.CmdOnHook();
	}
```

```
}

void CdlgMain::OnHold()
{
	CString str;
	this->m_Hold.GetWindowText(str);
	if(str=="保持")
	{
		this->AgentOCX.CmdHold();
	}
	else
	{
		this->AgentOCX.CmdHoldStop();
	}
}

void CdlgMain::OnTransfer()
{
	CString str;
	this->m_Transfer.GetWindowText(str);
	if(str=="转接")
	{
		this->AgentOCX.CmdTransfer();
	}
	else
	{
		this->AgentOCX.CmdTransferStop();
	}
}

void CdlgMain::OnDialOut()
{
	CString str;
	this->m_DialOut.GetWindowText(str);
	if(str=="外拨")
	{
		this->AgentOCX.CmdDialOut();
	}
	else
	{
		this->AgentOCX.CmdMakeCallStop();
```

```
	}
}

void CdlgMain::OnConsultation()
{
	CString str;
	this->m_Consultation.GetWindowText(str);
	if(str=="磋商")
	{
		this->AgentOCX.CmdConsult();
	}
	else if(str=="磋商结束")
	{
		this->AgentOCX.CmdConsultStop();
	}
	else
	{
		this->AgentOCX.CmdConsultStop();
	}

}

void CdlgMain::OnAuto()
{
	CString str;
	this->m_Auto.GetWindowText(str);
	if(str=="磋商转接")
	{
		this->AgentOCX.CmdConsultTransfer("","","","");
	}

}

void CdlgMain::OnOutPhone()
{
	CString str;
	this->m_OutPhone.GetWindowText(str);
	if(str=="自动")
	{
		this->AgentOCX.CmdAuto();
	}
```

```
	else
	{
		this->AgentOCX.CmdAutoCancel();
	}


}

void CdlgMain::OnFax()
{
	this->AgentOCX.CmdFax();

}

void CdlgMain::OnPause()
{
	CString str;
	this->m_Pause.GetWindowText(str);
	if(str=="暂停")
	{
		this->AgentOCX.CmdPause();
	}
	else
	{
		this->AgentOCX.CmdContinue();
	}


}

void CdlgMain::OnConference()
{
	CString str;
	this->m_Conference.GetWindowText(str);
	if(str=="会议")
	{
		this->AgentOCX.CmdConference();
	}
	else
	{
		this->AgentOCX.CmdMakeCallStop();
```

```
	}


}

void CdlgMain::OnPlay()
{
	CString str;
	this->m_Play.GetWindowText(str);
	if(str=="放音")
	{
		this->AgentOCX.CmdPlay();
	}
	else
	{
		this->AgentOCX.CmdPlayStop();
	}


}

void CdlgMain::OnListen()
{
	CString str;
	this->m_Listen.GetWindowText(str);
	if(str=="监听")
	{
		this->AgentOCX.CmdListen();
	}
	else
	{
		this->AgentOCX.CmdListenStop();
	}


}

void CdlgMain::OnDisconnect()
{
	this->AgentOCX.CmdIntrude();
}
```

```
void CdlgMain::OnRopCall()
{
       this->AgentOCX.CmdRopCall();

}

void CdlgMain::SetButtonOption(LPCTSTR sTitle, long Enable, CButton* strButton)
{
    strButton->SetWindowText(sTitle);
    strButton->EnableWindow(Enable);
}

void CdlgMain::OnEVTButtonStatusAocx1(LPCTSTR Name, LPCTSTR sTitle, long Enable)
{
      if(strcmp(sTitle,"OnHook") == 0)
  {
          sTitle = "挂机";
  }
  else if(strcmp(sTitle,"OffHook")==0)
  {
          sTitle="摘机";
  }
  else if(strcmp(sTitle,"HOOKOFFCONSULT")==0)
  {
          sTitle="接受";
  }
  else if(strcmp(sTitle,"Hold")==0)
  {
          sTitle="保持";
  }
  else if(strcmp(sTitle,"HoldCancel")==0)
  {
          sTitle="取消";
  }
  else if(strcmp(sTitle,"Transfer")==0)
  {
          sTitle="转接";
  }
  else if(strcmp(sTitle,"CancelTransfer")==0)
  {
          sTitle="取消";
```

```
}
else if(strcmp(sTitle,"DialOut")==0)
{
	sTitle="外拨";
}
else if(strcmp(sTitle,"CancelDialOut")==0)
{
	sTitle="取消";
}
else if(strcmp(sTitle,"Consultation")==0)
{
	sTitle="磋商";
}
else if(strcmp(sTitle,"CancelConsultation")==0)
{
	sTitle="取消";
}
else if(strcmp(sTitle,"StopConsultation")==0)
{
	sTitle="磋商结束";
}
else if(strcmp(sTitle,"ConsultTransfer")==0)
{
	sTitle="磋商转接";
}
else if(strcmp(sTitle,"Auto")==0)
{
	sTitle="磋商转接";
}
else if(strcmp(sTitle,"OutPhone")==0)
{
	sTitle="自动";
}
else if(strcmp(sTitle,"CancelOutPhone")==0)
{
	sTitle="取消";
}
else if(strcmp(sTitle,"Play")==0)
{
	sTitle="放音";
}
```

```
else if(strcmp(sTitle,"PlayCancel")==0)
{
	sTitle="结束";
}
else if(strcmp(sTitle,"Fax")==0)
{
	sTitle="传真";
}
else if(strcmp(sTitle,"FaxStop")==0)
{
	sTitle="结束";
}
else if(strcmp(sTitle,"Pause")==0)
{
	sTitle="暂停";
}
else if(strcmp(sTitle,"Continue")==0)
{
	sTitle="恢复";
}
else if(strcmp(sTitle,"ContinueDialTask")==0)
{
	sTitle="放弃回访";
}
else if(strcmp(sTitle,"Listen")==0)
{
	sTitle="监听";
}
else if(strcmp(sTitle,"CancelListen")==0)
{
	sTitle="结束";
}
else if(strcmp(sTitle,"Disconnect")==0)
{
	sTitle="强插";
}
else if(strcmp(sTitle,"Conference")==0)
{
	sTitle="会议";
}
else if(strcmp(sTitle,"CancelConference")==0)
```

```
{
        sTitle="取消";
}
else if(strcmp(sTitle,"RopCall")==0)
{
        sTitle="拦截";
}
else if(strcmp(sTitle,"LOGINSUCC")==0)
{
        sTitle="登录成功";
}
else if(strcmp(sTitle,"LOGINFAIL")==0)
{
        sTitle="登录失败";
}
else if(strcmp(sTitle,"TRANSFERFAIL")==0)
{
        sTitle="转接失败";
}

if(strcmp(Name,"Hook")==0)
{
        SetButtonOption (sTitle, Enable,&(m_Hook));//(CButton*)GetDlgItem(IDC_Hook));
}
else if(strcmp(Name,"Hold")==0)
{
        SetButtonOption (sTitle, Enable,&(this->m_Hold));
}
else if(strcmp(Name,"Transfer")==0)
{
        SetButtonOption (sTitle, Enable,&(this->m_Transfer));
}
else if(strcmp(Name,"DialOut")==0)
{
        SetButtonOption (sTitle, Enable,&(this->m_DialOut));
}
else if(strcmp(Name,"Consultation")==0)
{
        SetButtonOption (sTitle, Enable,&(this->m_Consultation));
}
else if(strcmp(Name,"Auto")==0)
{
```

```
            SetButtonOption (sTitle, Enable,&(this->m_Auto));
    }
    else if(strcmp(Name,"OutPhone")==0)
    {
        SetButtonOption (sTitle, Enable,&(this->m_OutPhone));
    }
    else if(strcmp(Name,"Fax")==0)
    {
            SetButtonOption (sTitle, Enable,&(this->m_Fax));
    }
    else if(strcmp(Name,"Pause")==0)
    {
            SetButtonOption (sTitle, Enable,&(this->m_Pause));
    }
    else if(strcmp(Name,"Conference")==0)
    {
        SetButtonOption (sTitle, Enable,&(this->m_Conference));
    }
    else if(strcmp(Name,"Play")==0)
    {
        SetButtonOption (sTitle, Enable,&(this->m_Play));
    }
    else if(strcmp(Name,"Listen")==0)
    {
        SetButtonOption (sTitle, Enable,&(this->m_Listen));
    }
    else if(strcmp(Name,"Disconnect")==0)
    {
        SetButtonOption (sTitle, Enable,&(this->m_Disconnect));
    }
    else if(strcmp(Name,"RopCall")==0)
    {
        this->SetButtonOption(sTitle, Enable,&(m_RopCall));
    }
}
```