

Brekeke PAL

Version 2.x

Developer's Guide

Brekeke Software, Inc.

Version

Brekeke PAL v2.x Developer's Guide

Revised December 2009

Copyright

This document is copyrighted by Brekeke Software, Inc.

Copyright © 2009 Brekeke Software, Inc.

This document may not be copied, reproduced, reprinted, translated, rewritten or readdressed in whole or part without expressed, written consent from Brekeke Software, Inc.

Disclaimer

Brekeke Software, Inc. reserves the right to change any information found in this document without any written notice to the user.

Trademark Acknowledgement

- ◆ *LINUX is a registered trademark of Linus Torvalds in the United States and other countries.*
- ◆ *Red Hat is a registered trademark of Red Hat Software, Inc.*
- ◆ *Windows is a trademark or registered trademark of Microsoft Corporation in the United States and other countries.*
- ◆ *Mac is a trademark of Apple Computer, Inc., registered in the U.S. and other countries.*
- ◆ *Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.*
- ◆ *Other logos and product and service names contained in this document are the property of their respective owners.*

- 1. BREKEKE PAL 5**
 - 1.1 What’s New in Version 2.4 5**
 - 1.2 What’s New in Version 2.3 5**
 - 1.3 System Requirements..... 5**
 - 1.4 Installation..... 6**
 - 1.5 Uninstall 6**
 - 1.6 Upgrading Previous PAL Application projects..... 6**

- 2. DEVELOPING WITH BREKEKE PAL 7**
 - 2.1 Verify your Brekeke PBX and IP Phone Setup 7**
 - 2.2 Configure Web Service at Brekeke PBX 7**
 - 2.3 Add PAL to your Visual Studio Component ToolBox 8**
 - 2.4 Add the PAL Control to the Project Form 8**
 - 2.5 Set PAL Control Properties 8**
 - 2.5.1 TimerInterval Property..... 8
 - 2.5.2 LocalPort..... 8
 - 2.5.3 PbxAddress..... 9
 - 2.5.4 PbxPort 9
 - 2.5.5 WebService..... 9
 - 2.5.6 LocalRegPort 9
 - 2.5.7 PbxRegPort..... 9
 - 2.5.8 Tenant 9
 - 2.6 Initialize the Control 9**
 - 2.7 Subscribe for Notifications 9**
 - 2.8 Monitor events 10**

2.9	Complete Your Application.....	10
2.10	Terminology	11
2.10.1	TalkerID	11
2.10.2	RoomID	11
3.	THE PAL CONTROL.....	12
3.1	Properties.....	12
3.2	Events.....	12
3.2.1	notify_status(ByVal user As String, ByVal talkerID As Integer)	12
3.2.2	notify_statusEx(ByVal user As String, ByVal talkerID As Integer, ByVal statusType As Integer, ByVal time As Long)	12
3.2.3	notify_talker(ByVal tevent As PAL.TalkerEvent).....	12
3.2.4	notify_vmail(ByVal user As String)	14
3.2.5	notify_registered(ByVal user As String, ByVal reg_status As Integer)	14
3.3	Functions	15
3.3.1	announce(ByVal user As String, ByVal talkerid As Integer) As String	15
3.3.2	attendedTransfer(ByVal user As String, ByVal talkerid As Integer) As String	15
3.3.3	barge(ByVal user As String, ByVal talkerid As Integer) As String.....	16
3.3.4	blindTransfer(ByVal user As String, ByVal talkerid As Integer) As String	16
3.3.5	callEnd(ByVal talkerid As Integer) As String.....	16
3.3.6	callForward(ByVal roomid As String, ByVal userlist As String) As String	17
3.3.7	Public Function getDisplayName(ByVal user As String) As String.....	17
3.3.8	Public Function getDisplayOther(ByVal user As String) As String	17
3.3.9	getLogID (ByVal talkerID As Integer) As String	18
3.3.10	getParkerIDbyCallee(ByVal user As String, ByVal callee As String) As Integer.....	18
3.3.11	getParkerIDs(ByVal user As String) As String	18
3.3.12	getParkNumber(ByVal user As String, ByVal talkerid As Integer) As String	19
3.3.13	getParkNumbers(ByVal user As String) As String	19
3.3.14	getRoomID (ByVal talkerID As Integer) As Integer	19
3.3.15	getRoomIDbyCallee(ByVal user As String, ByVal callee As String) As Integer.....	20
3.3.16	getSipUri(ByVal user As String) As String	20
3.3.17	getSipUriOther(ByVal user As String) As String	20
3.3.18	getStatus(ByVal statusType As Integer) As String.....	21

3.3.19 getTalkerIDbyCallee(ByVal user As String, ByVal callee As String) As Integer.....	21
3.3.20 getTalkerIDs(ByVal user As String) As String.....	21
3.3.21 initialize() As String.....	22
3.3.22 monitor(ByVal user As String, ByVal talkerid As Integer) As String.....	22
3.3.23 park(ByVal talkerid) As String.....	22
3.3.24 parkCancel(ByVal talkerid) As String.....	23
3.3.25 parkPickup(ByVal user As String, ByVal parkNumber As String) As String.....	23
3.3.26 parkEx(ByVal talkerid As Integer, ByVal retriever As String) As String.....	23
3.3.27 pbxVersion() As String.....	24
3.3.28 recordingStart(ByVal talkerid) As String	24
3.3.29 recordingStop(ByVal talkerid As Integer) As String	24
3.3.30 remoteControl(ByVal talkerid As Integer, ByVal action As String) As String	24
3.3.31 showStatus(ByVal user As String) As String.....	25
3.3.32 subscribeStatus(ByVal user As String, ByVal user_array() As String, ByVal expireSeconds As Integer) As String.....	25
3.3.33 subscribeRegistered(ByVal user As String, ByVal user_array() As String, ByVal expireSeconds As Integer) As String.....	26
3.3.34 subscribeVmail(ByVal user As String, ByVal user_array() As String, ByVal expireSeconds As Integer) As String	26
3.3.35 threePCC(ByVal caller As String, ByVal callee As String) As String	26
3.3.36 threePCCEx(ByVal user As String, ByVal src As String, ByVal dest() As String, ByVal mode As String) As String.....	27
3.3.37 totalParkerIDs(ByVal user As String) As Integer	27
3.3.38 totalTalkerIDs(ByVal user As String) As Integer	27
3.3.39 transferCancel(ByVal talkerid As Integer) As String	28
3.3.40 tutor(ByVal user As String, ByVal talkerid As Integer) As String.....	28
3.3.41 vmailCount(ByVal user As String) As Integer.....	28

1. Brekeke PAL

Brekeke PAL (Brekeke PBX Active Library) is a Windows Control Library that allows the creation of companion applications for the Brekeke PBX. Some possible applications that may be developed with the Brekeke PAL include Operator Consoles, Speed Dial Panels, Phone Call Recorders, and Line Status Panels.

1.1 What's New in Version 2.4

The following changes were introduced since the version 2.4:

(REQUIRES Brekeke PBX Version 2.4.x)

- ◆ New Status event that include more exact status and timestamp
- ◆ Minor bug fixes.

1.2 What's New in Version 2.3

The following changes were introduced since the version 2.3.x:

(REQUIRES Brekeke PBX Version 2.3.x)

- ◆ Subscribe/Notify for the Registrar now accepts "all" as parameter
- ◆ Subscribe/Notify for Status now accepts "all" as parameter
- ◆ PAL is compatible with the new Brekeke PBX Multi-Tenant Edition
- ◆ New Status event that include more exact status and timestamp
- ◆ Log Id to access Call Log information stored in Third Party database
- ◆ Improved third party call control function allows external agent as call source
- ◆ Added new remote control function to take phones off-hook and on-hold
- ◆ Minor bug fixes.

1.3 System Requirements

Development with Brekeke PAL requires the following:

- ◆ Windows Platform
- ◆ Brekeke PBX with PAL Option enabled or Evaluation Edition.
The first two numbers in the version should be the same between Brekeke PAL and Brekeke PBX. (ex. Brekeke PAL version 2.3.3.0 and Brekeke PBX 2.3.8.2)
- ◆ Java Runtime Environment (JRE) Version 6 or later. (Brekeke PAL Version 2.3 can work with JRE 1.5 or 6.)
- ◆ Visual Studio 2005 (or later) or Visual Express for Development (.NET development)

1.4 Installation

Prepare the development environment as follows:

1. Download and install the Java Runtime Environment Version 6.
2. Uninstall any previous version of Brekeke PAL (see Section 1.5 below).
3. Install Brekeke PBX (see Section 1.3 for Version requirement). The Brekeke PBX does not have to be installed on the same machine as the development environment.
4. Run the PAL Installer.

1.5 UnInstall

If, for some reason, the Brekeke PAL needs to be uninstalled:

1. Go to the Windows Control Panel.
2. Select Add or Remove Programs.
3. Remove the Brekeke PAL.

1.6 Upgrading Previous PAL Application projects

If you have existing Visual Studio Projects that use the Brekeke PAL Component, you will need to rebuild your solutions using the new Brekeke PAL version as follows:

1. Backup your existing code.
2. Be sure that the Microsoft Visual Studio IDE is closed because it locks some dlls from changes.
3. Install the new Brekeke PAL version as described in Section 1.4 above.
4. Go to the PAL installation directory and copy the files "Interop.OperatorBean.dll" and "Pal.dll" to your projects "bin\Debug" and "bin\Release" folders.
5. Open the Microsoft Visual Studio IDE and rebuild your solutions.

2. Developing with Brekeke PAL

Creating a companion application for the Brekeke PBX is now very simple. Read completely through Sections 2.1 to 2.10 for the basic development steps. Read Section 3 to learn about all the PAL Control's capabilities.

2.1 Verify your Brekeke PBX and IP Phone Setup

Applications developed using Brekeke PAL are usually software companions to desktop phones or soft phones. Make sure your desktop IP phones or softphones are registered with the Brekeke PBX. Please consult the proper Brekeke Manuals if you need help with this. Place some test calls to make sure the phone system and the Brekeke PBX are running properly.

2.2 Configure Web Service at Brekeke PBX

The PAL Control makes use of the built-in Brekeke Web Service which is controlled using the Brekeke PBX Administration Tool. Only Clients with IP Addresses that match the regular expression pattern defined in the Brekeke PBX Administration Tool will be allowed to consume from the web service. Define the valid client IP addresses is necessary in order to send commands to the Brekeke PBX.

1. Login Brekeke PBX Administration tool with Admin privileges.
2. At [Options] > [PBX system settings] > [Valid client IP Pattern], enter a regular expression define the Clients' IP addresses pattern.

For example, "192.168\..*" will allow all PAL clients whose IP addresses starting with "192.168"

Set the following parameters at Brekeke PBX Administration Tool > [Options] > [Advanced] field to control PAL clients usage of Brekeke Web Service (since Brekeke PBX version 2.4.*).

Restart Brekeke PBX from Administration Tool with the settings below

1. **statusboard.passthrough.events**

Description: events which can be got by PAL clients.

Values: event codes, separated by comma (,)

Current Event codes: 35 (HOLD), 36 (UNHOLD)

Example:

```
statusboard.passthrough.events = 35,36
```

Send HOLD and UNHOLD status to PAL clients.

2. **statusboard.savedata.switch**

Description: Save the current status for PAL or not

Values: true or false

Example:

```
statusboard.savedata.switch = false
```

PAL clients cannot get extensions current status from Brekeke PBX at the time PAL application starts. PAL clients can only get new event status after PAL starts running..

2.3 Add PAL to your Visual Studio Component ToolBox

To add PAL to your project toolbox:

1. Select Choose Toolbox Items from the Tools menu or right-click on the Toolbox and select Choose Items from its shortcut menu.
2. Select and add the Brekeke PAL Control by browsing for the PAL.dll file in the PAL installation directory.

2.4 Add the PAL Control to the Project Form

To create an instance of the Brekeke PAL Control:

1. Open the project's form in the Forms Designer. In the Toolbox, you will see the new PAL control.
2. Drag the PAL control onto your form. An instance of the control is created and added to the Component Tray. The Brekeke Logo will appear on your form. This logo can be hidden by changing the hide property of the control to TRUE. The default control name will be Pal1.

2.5 Set PAL Control Properties

Customize the control by setting some properties. Right click on the Brekeke PAL control and select "Properties" to see the properties sheet for the control.

2.5.1 TimerInterval Property

Set the "timerInterval" under the Misc tab of the properties for PAL1. This step is optional. This property controls the polling rate to detect notification events. The default setting is 250 milliseconds.

2.5.2 LocalPort

The port on the client machine through which Brekeke PAL will send Status and Voicemail notifications.

2.5.3 PbxAddress

Set the IP Address of the Brekeke PBX Server.

2.5.4 PbxPort

The port number used by the Brekeke PBX Server.

2.5.5 WebService

Set the URL of the Brekeke PBX Web Service. The Brekeke PBX Web Service is usually in `http://<host>/pbx/services/pal` where `<host>` is the host server name or host IP address of your Brekeke PBX Server.

2.5.6 LocalRegPort

The port on the client machine through which Brekeke PAL will send Registrar notifications.

2.5.7 PbxRegPort

The port on the PBX Server for Registrar related interactions.

2.5.8 Tenant

Applicable only when using Brekeke Multi-Tenant PBX. Specify which Tenant is interacting with PAL. For Single Tenant Brekeke PBX, this should remain as "-" (the dash character).

2.6 Initialize the Control

After configuring the Control's properties, the control must be initialized prior to calling any other methods or functions. The Brekeke PAL Control is initialized by calling the "initialize" function.

2.7 Subscribe for Notifications

Brekeke PAL uses a subscribe-notify mechanism to receive information from the Brekeke PBX. In your form's Load event, call the subscribe functions (see Section 3).

There are three types of notifications that you can subscribe for:

- ◆ Registrar notifications will allow you to see which users have registered
- ◆ Line status notifications will allow you to see the state of a phone
- ◆ Voicemail notifications will allow you to see if voicemail is available

For example,

```
Dim UserArray() As String = {"4001","4007","4008"}
```

```
Dim UserAll() As String = {"all"}
```

```
Private Sub Form1_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles  
Me.Load
```

```
    PAL1.subscribeRegistered("4008", UserAll, 6000 ) ` example of using "all"
```

```
    PAL1.subscribeStatus("4008", UserArray, 6000 )
```

```
    PAL1.subscribeVmail("4008", UserArray, 6000)
```

```
End Sub
```

The first parameter is your desktop extension. The second parameter is a string array of extension numbers for which you want to receive notifications about line status and voice mail status. The last parameter indicates how many seconds before the subscription expires (in seconds).

2.8 Monitor events

If your subscription was successful, Brekeke PBX will start sending notification events to the PAL control. PAL will generate notify_status events for line status notification events. For voicemail notification events, PAL will generate notify_mail events. Monitor for these events to display line status and voicemail changes.

For example, we received a voicemail notification

```
Private Sub Pall_notify_vmail(ByVal user As String) Handles Pall.notify_vmail
```

```
    If (user = AppUser) Then
```

```
        vmailCountLabel.Text = Pall.vmailCount(user)
```

```
    End If
```

```
End Sub
```

2.9 Complete Your Application

Use the functions of the Brekeke PAL control to create your application. Please see the next section for a full description of all available functions.

2.10 Terminology

This section explains some terms used in PAL function calls and development.

2.10.1 TalkerID

When one agent calls another agent, each agent is considered a talker and is assigned a Talker ID (TID). Each agent can be involved in multiple conversations, hence each can have multiple Talker IDs.

2.10.2 RoomID

When one agent calls another agent; conceptually, a room is created and the talkers are placed in the room. Each room is assigned a Room ID.

3. The PAL Control

This section is a full description of the PAL Windows Control Library, including properties, events, and functions.

3.1 Properties

`timerInterval` - This is the number of milliseconds between polls for line status and voicemail status events. The default setting is 250 milliseconds.

3.2 Events

3.2.1 `notify_status(ByVal user As String, ByVal talkerID As Integer)`

Available and deprecated in Brekeke PAL version 2.4, replaced by “notify_talker” event.
Raised when a line status event has been detected. `User` is the extension for which the event was raised. The `talkerID` corresponds to the user.

3.2.2 `notify_statusEx(ByVal user As String, ByVal talkerID As Integer, ByVal statusType As Integer, ByVal time As Long)`

Available and deprecated in Brekeke PAL version 2.4, replaced by “notify_talker” event.
Raised when a line status event has been detected. `User` is the extension for which the event was raised. The `talkerID` corresponds to the user. `statusType` is the type of status that occurred. `time` is a timestamp of when the status occurred given as a long. To determine the `statusType` use the new function `getStatus()` in Section 3.3 below.

3.2.3 `notify_talker(ByVal tevent As PAL.TalkerEvent)`

New event in Brekeke PAL version 2.4. Raised when a line status event has been detected. `PAL.TalkerEvent` is a class of `PAL`, whose fields contain the information related to the extension of the current event.

The following shows the fields of `PAL.TalkerEvent` and usage examples:

`tevent.Status`

Description:

You can compare `tevent.Status` type with the constant fields in the table below directly without changing status type to status string by using `getStatus()`

Return: Call status type as integer.

PAL.TalkerEvent class has the following constant fields for the status type:

CALLING	Making a call to the user's UA. (Brekeke PBX works as UAC.)
INCOMING	Receiving a call from the user's UA. (Brekeke PBX works as UAS.)
CALL_SUCCESS	Start talking as UAC.
ANSWER_SUCCESS	Start talking as UAS.
END_TALKING	Disconnect a call.
RINGING	Receiving a RINGING signal from the user's UA.
HOLD	The call was placed on hold.
UNHOLD	The held call was retrieved.
ERR	An error happened.

Example:

```
If tevent.Status = PAL.TalkerEvent.CALL_SUCCESS Then
```

tevent.User

Description: Get the user of which the event is raised

Return: The extension for current event

tevent.TalkerID

Description: Get the Talker ID of which the event is raised

Return: The Talker ID for current event

tevent.DisconnectedByTheOther

Description: check if the current session is disconnected by the other side

Return: true or false

tevent.DisplayName

Description: Get display name of the user of which the event is raised

Return: the display name as string

tevent.DisplayNameOfTheOther

Description: Get the other user's display name of the conversation

Return: the other party display name as string

tevent.LogID

Description: Get the log ID assigned to the conversation

Return: the log ID as string

tevent.Q850Code

Description: Get the Q.850 code generated when a call has been rejected.

Return: q.850 code as short

tevent.ResponseCode

Description: Get the response code of the event

Return: response code as integer

tevent.RoomID

Description: Get RoomID assigned to the conversation

Return: room ID as string

tevent.TheOtherNumber

Description: Get the other user number of the conversation

Return: the other side user number as string

tevent.Time

Description: Get timestamp of when the status occurred

Return: timestamp as long

3.2.4 notify_vmail(ByVal user As String)

Raised when a line status event has been detected. User is the extension for which the event was raised.

3.2.5 notify_registered(ByVal user As String, ByVal reg_status As Integer)

Raised when a registrar event has been detected. User is the extension for which the event was raised. reg_status is 0 if the user became unregistered. reg_status is 1 if the user became registered.

3.3 Functions

This section contains descriptions of all the supported functions.

3.3.1 announce(ByVal user As String, ByVal talkerid As Integer) As String

Description: Allows an announcement to be made to talkers in a conversation. The Announcer is the user given as the first parameter. The announcement will be audible to all talkers in the same room as the talker identified by the talkerid parameter. The announcer is able to announce, but will not be able to listen to the conversation between talkers.

Parameters:

user – extension number of the user making the announcement

talkerid – an integer representing the TalkerID of the user to whom we want to make the announcement.

Returns: A success message or an error message.

Related Functions: barge, monitor, tutor

3.3.2 attendedTransfer(ByVal user As String, ByVal talkerid As Integer) As String

Description: Perform an attended transfer from the talker identified by parameter talkerid to the extension identified by the parameter user.

Parameters:

user – the extension to transfer the call to

talkerid – the talkerid of the talker whose conversation will be transferred. Usually this one of the Talker IDs assigned to the CoAct control's User.

Returns: A success message or an error message.

Related Functions: blindTransfer, cancelTransfer

3.3.3 **barge(ByVal user As String, ByVal talkerid As Integer) As String**

Description: Allows a user to barge into a conversation. The barging user is the extension specified in the user parameter. The barging user will be able to listen and speak to all talkers in the conversation.

Parameters:

user – the extension of the user who will be barging into the conversation

talkerid – an integer representing the TalkerID of one of the users whose conversation will be barged into

Returns: A success message or an error message.

Related Functions: announce, monitor, tutor

3.3.4 **blindTransfer(ByVal user As String, ByVal talkerid As Integer) As String**

Description: Perform a blind transfer from the talker identified by parameter talkerid to the extension identified by the parameter user.

Parameters:

user – the extension to transfer the call to

talkerid – the talkerid of the talker whose conversation will be transferred. Usually this one of the Talker IDs assigned to the CoAct control's User.

Returns: A success message or an error message.

Related Functions: attendedTransfer, cancelTransfer

3.3.5 **callEnd(ByVal talkerid As Integer) As String**

Description: End a call

Parameters:

talkerid – the talkerid

Returns: A success message or an error message.

3.3.6 callForward(ByVal roomid As String, ByVal userlist As String) As String

Description: Forward a conversation to a list of users

Parameters:

roomid – RoomID assigned to the conversation

userlist – space separated list of user extensions to whom the conversation will be forwarded

Returns: A success message or an error message.

Related Functions: getRoomID

3.3.7 Public Function getDisplayName(ByVal user As String) As String

Description: Return the display name for this user.

Parameters:

User – the user.

Returns: The display name as a string or the empty string.

Related Functions: getDisplayOther, getSipUri, getSipUriOther

3.3.8 Public Function getDisplayOther(ByVal user As String) As String

Description: Return the display name for the other extension with whom the user is conversing.

Parameters:

User – the user.

Returns: The other user's display name as a string or the empty string.

Related Functions: getDisplayName, getSipUri, getSipUriOther

3.3.9 getLogID (ByVal talkerID As Integer) As String

Description: Returns the third party call log ID assigned to the conversation. Requires use of third party database for the Call Log. Contact Brekeke for details.

Parameters:

talkerID – the talker ID of one of the talkers in the room

Returns: A String representing the ID. Returns empty string if no ID is found.

Related Functions: none.

3.3.10 getParkerIDbyCallee(ByVal user As String, ByVal callee As String) As Integer

Description: Returns the parked talkerID for the user. This function is useful if the extension of the other talker is known.

Parameters:

user – the user who parked the call

callee – the user extension of the other talker

Returns: An integer representing a talker ID. On error, this returns a -1.

Related Functions: totalTalkerIDs

3.3.11 getParkerIDs(ByVal user As String) As String

Description: Returns a list of space separated parked talker ids for the parameter user

Parameters:

user – the user for whom we want to retrieve the talker IDs

Returns: A string of space separated talker IDs. When no talker IDs are assigned, this returns the empty string.

Related Functions: totalTalkerIDs

3.3.12 getParkNumber(ByVal user As String, ByVal talkerid As Integer) As String

Description: Returns the number to retrieve a parked call corresponding to the talkerid

Parameters:

User – extension number of user who parked the call

talkerid – the talkerid of the parked call

Returns: A string representing the retrieve number for picking up a parked call. When no retrieve numbers are available, this returns an empty string.

Related Functions: park, parkPickup, parkCancel, getParkNumbers

3.3.13 getParkNumbers(ByVal user As String) As String

Description: Returns all retrieve numbers that are assigned to the user

Parameters:

User – extension number of user who parked the call

Returns: A string representing the retrieve numbers for picking up a parked call. When no retrieve numbers are available, this returns an empty string.

Related Functions: park, parkPickup, parkCancel, getParkNumber

3.3.14 getRoomID (ByVal talkerID As Integer) As Integer

Description: Returns the ID number assigned to the conversation.

Parameters:

talkerID – the talker ID of one of the talkers in the room

Returns: An integer representing a room ID. On error, this returns a -1.

Related Functions: callforward

3.3.15 getRoomIDbyCallee(ByVal user As String, ByVal callee As String) As Integer

Description: Returns the ID number assigned to the conversation. This function is useful if the extension number of the other talker is known.

Parameters:

user – the extension number of one of the talkers in the room

callee – the extension number of the other talker

Returns: An integer representing a room ID. On error, this returns a -1.

Related Functions: callforward

3.3.16 getSipUri(ByVal user As String) As String

Description: Return the SIP URI for this user.

Parameters:

User – the user.

Returns: The SIP URI as a string or the empty string.

Related Functions: getDisplayName, getDisplayOther, getSipUriOther

3.3.17 getSipUriOther(ByVal user As String) As String

Description: Return the SIP URI for the other extension with whom the user is conversing.

Parameters:

User – the user.

Returns: The other user's SIP URI as a string or the empty string.

Related Functions: getDisplayName, getDisplay Other, getSipUri

3.3.18 getStatus(ByVal statusType As Integer) As String

Description: Return the description corresponding to the value indicated by statusType.

Parameters:

statusType – integer which is a statusType code.

Returns: Description of what the statusType code represents.

Related Functions: None.

3.3.19 getTalkerIDbyCallee(ByVal user As String, ByVal callee As String) As Integer

Description: Returns the talkerid. This function is useful if the extension number of the other talker is known.

Parameters:

user – the user

callee – the extension number of the other talker

Returns: An integer representing a talker ID. On error, this returns a -1.

Related Functions: getTalkerIDs, totalTalkerIDs

3.3.20 getTalkerIDs(ByVal user As String) As String

Description: Returns a list of space separated talker ids for the parameter user

Parameters:

user – the user for whom we want to retrieve the talker IDs

Returns: A space separated string of talker IDs. If no talker IDs are assigned, this returns an empty string.

Related Functions: getTalkerID, totalTalkerIDs

3.3.21 initialize() As String

Description: Connects the control to the Brekeke PBX prior to other methods and function calls

Parameters: None.

Returns: Success or Error String.

Related Functions: None.

3.3.22 monitor(ByVal user As String, ByVal talkerid As Integer) As String

Description: Allows a user to listen in on a conversation. The listener is the user given as the first parameter. The listener is able to hear, but will not be able to speak to the talkers.

Parameters:

user – extension number of the user who will monitor the call

talkerid – the talkerid of one of the talkers in the conversation

Returns: A success message or an error message.

Related Functions: barge, announce, tutor

3.3.23 park(ByVal talkerid) As String

Description: park a call.

Parameters:

talkerid – talkerId whose call is to be parked

Returns: A string representing the retrieve number or an error message.

Related Functions: parkPickup, parkCancel, getParkNumber, parkEx

3.3.24 parkCancel(ByVal talkerid) As String

Description: Cancel parking a call. This only works before the parking phone is placed back onhook. Once the phone is back onhook, the park is complete and cannot be cancelled.

Parameters:

talkerid – talkerid whose parked call is to be cancelled

Returns: A success message or an error message.

Related Functions: park, parkPickup, getParkNumber, parkEx

3.3.25 parkPickup(ByVal user As String, ByVal parkNumber As String) As String

Description: pickup a parked call

Parameters:

user – the extension number where you want to pickup the parked call

parkNumber – the retrieve number

Returns: A success message or an error message.

Related Functions: park, parkPickup, parkCancel, getParkNumber, parkEx

3.3.26 parkEx(ByVal talkerid As Integer, ByVal retriever As String) As String

Description: park a call.

Parameters:

talkerid – talkerid whose call is to be parked

retriever – the number that can be used to unpark or retrieve a parked call

Returns: A string representing the retrieve number or an error message.

Related Functions: parkPickup, parkCancel, getParkNumber, park

3.3.27 pbxVersion() As String

Description: Retrieve the Brekeke PBX Version Number

Parameters:None.

Returns: A string representing Brekeke PBX Version Number.

Related Functions: None.

3.3.28 recordingStart(ByVal talkerid) As String

Description: record a conversation

Parameters:

talkerid – talkerid whose conversation will be recorded. The recording will go into this talker's voice mailbox.

Returns: A success message or an error message.

Related Functions: recordingStop

3.3.29 recordingStop(ByVal talkerid As Integer) As String

Description: stop recording

Parameters:

talkerid – talkerid of user whose recording is to be stopped

Returns: A success message or an error message.

Related Functions: recordingStart

3.3.30 remoteControl(ByVal talkerid As Integer, ByVal action As String) As String

Description: remotely take a phone off-hook or place a call on hold

Parameters:

talkerid – talkerid of user whose phone will be controlled

action – either “talk” which takes a phone off-hook, or “hold” which will place the call on hold

Returns: A success message or an error message.

Related Functions: None

Notes: This function is supported on certain Polycom and Snom phones.

3.3.31 showStatus(ByVal user As String) As String

Description: Show the line status for the user.

Parameters:

user – the extension number whose status is being checked

Returns: A string showing the line status for a user. On error, an error message is returned.

Possible status states are:

- ◆ Available – the line is onhook and the user is available
- ◆ Connecting – the dialed call is progressing
- ◆ Ringing – the callee's line is ringing
- ◆ Talking – the call was a success and the callee answered
- ◆ Undefined – the call was put on hold

The Callee is also returned. For example: Ringing 4001

Related Functions: subscribeStatus

3.3.32 subscribeStatus(ByVal user As String, ByVal user_array() As String, ByVal expireSeconds As Integer) As String

Description: Subscribe to receive notification about line status

Parameters:

user – extension where the notifications are to be sent

user_array – an array of extension numbers for which we want to receive status notifications

expireSeconds – how long before the subscription expires in seconds

Returns: A success message or an error message.

Related Functions: showStatus, subscribeVmail, subscribeRegistered

3.3.33 subscribeRegistered(ByVal user As String, ByVal user_array() As String, ByVal expireSeconds As Integer) As String

Description: Subscribe to receive notification about register/unregister events

Parameters:

user – extension where the notifications are to be sent

user_array – an array of extension numbers for which we want to receive status notifications

expireSeconds – how long before the subscription expires in seconds

Returns: A success message or an error message.

Related Functions: showStatus, subscribeVmail, subscribeStatus

3.3.34 subscribeVmail(ByVal user As String, ByVal user_array() As String, ByVal expireSeconds As Integer) As String

Description: subscribe to receive notification about voicemail status

Parameters:

user – extension where the notifications are to be sent

user_array – an array of extension numbers for which we want to receive voicemail notifications

expireSeconds – how long before the subscription expires in seconds

Returns: A success message or an error message.

Related Functions: showStatus, subscribeStatus, vmailCount, subscribeRegistered

3.3.35 threePCC(ByVal caller As String, ByVal callee As String) As String

Description: Place a call between caller and callee. The caller's line will ring first. Once the caller pick's up, then the callee's phone will ring.

Parameters:

caller – the caller's extension or phone number

callee – the callee's extension or phone number

Returns: A success message or an error message.

Related Functions: None.

3.3.36 threePCCEx(ByVal user As String, ByVal src As String, ByVal dest() As String, ByVal mode As String) As String

Description: Place a call between caller and callee. The caller's line will ring first. Once the caller pick's up, then the callee's phone will ring.

Parameters:

user – String representing the call owner.

src – String representing the from-url.

dest – String array representing the to-urls. Calls can be simultaneously made to different to-urls.

mode – Strings: "1" or "2". Type "1" will simultaneously call the from-url and the tourl then connect them. Type "2" will call the from-url first. When the from-url picks up, The to-url is called, then the two calls are connected.

Returns: A success message or an error message.

Related Functions: None.

3.3.37 totalParkerIDs(ByVal user As String) As Integer

Description: Returns the total number of Parked Talker IDs that are assigned to the user

Parameters:

user – the user's extension number

Returns: Number of talker IDs that are parked for the user. Returns 0 if there are none.

Related Functions: getParkerID, getParkerIDs

3.3.38 totalTalkerIDs(ByVal user As String) As Integer

Description: Returns the total number of Talker IDs that are assigned to the user

Parameters:

user – the user's extension number

Returns: Number of talker IDs assigned to a user. Returns 0 if there are none..

Related Functions: getTalkerID, getTalkerIDs

3.3.39 transferCancel(ByVal talkerid As Integer) As String

Description: Cancel an attended transfer

Parameters:

talkerid – talkerid which is in process of being transferred

Returns: A success message or an error message.

Related Functions: attendedTransfer

3.3.40 tutor(ByVal user As String, ByVal talkerid As Integer) As String

Description: Allows a user to listen in on a conversation and speak to one of the talkers. The talker being tutored is specified using the second parameter. The other talker who is not being tutored will not be able to hear the tutor.

Parameters:

user – extension number of the user who is the tutor

talkerid – the talkerid of one of the talkers in the conversation who will be tutored

Returns: A success message or an error message.

Related Functions: monitor, barge, announce

3.3.41 vmailCount(ByVal user As String) As Integer

Description: Get the number of voice messages

Parameters:

user – the user whose voice mail is to be checked

Returns: Number of new voice mail messages. Returns 0 when there are no messages.

Related Functions: subscribeVmail