TECHNICAL REFERENCE GUIDE

blueButler™

Intelligent Digital Recorder

blueButler™ iDR Technical Reference Guide

Table of Contents

blueButler Documentation

This Technical Reference Guide provides details on how to integrate other applications with blueButler using the blueButler Web Services. It also describes how to add or modify XML Scripts in the blueButler .NET client application and other technical aspects of the product.

The Supervisor Guide describes how supervisors, managers and others access the blueButler Intelligent Digital Recording (iDR) application.

The blueButler User Guide provides instructions on how individual agents and users can access their recording files and install and use the blueButler pop-up client.

For information on the Administrator tasks associated with blueButler iDR, other blueButler applications or installing blueButler software, please contact your blueButler technical representative.

SECTION 1 – Overview

About blueButler

blueButler iDR is a state-of-the-art digital call recording system that is designed to improve efficiency within your organization. The blueButler iDR platform has a number of technical interfaces that are described in this guide. Choose the interface that is applicable to the task that you want to perform.

WEB SERVICES Interface

The **WEB SERVICES** interface allows you to integrate your own application software with blueButler iDR. This interface can be used to pass data variables to be stored in the SQL record for the recording, it can also be used to start/stop/pause/resume recording, and to play a recording from within your application.

XML SCRIPTS

The **XML SCRIPTS** interface is used to provide online scripts that prompt users on what to say while they are talking to callers. It is completely flexible in allowing you to determine what type of information to display on the screen. The use of the Script is tracked in the blueButler SQL Events table allowing you to correlate the scripts with your call recording files.

FILE IMPORT UTILITY

The File Import utility program can be used to import User Profiles into blueButler. It is useful for when the system is initially being configured or when a large number of users are being added to the system.

SQL DATABASE

Every call recording in blueButler has an associated record in the **SQL DATABASE**. These records have many fields that can be used to create reports using third-party report writers and/or queries from other application software. The SQL tables in blueButler are published for read and write access providing you with the opportunity to use the data records to meet your specific needs.

SECTION 2 – WEB SERVICES Interface

This section describes how to use the blueButler Web Services interface to integrate and/or control the call recording service from another software application.

Security Methods

All of the Web Service commands require a login to ensure that the recordings are kept secure. There are two security methods provided – one method is used when all requests are submitted from a server and the other method is used when individual user systems access blueButler. Note that the user must be set up in blueButler with their Windows login entered into the 'Alternate Login ID' field of their blueButler user profile.

Server Access Method

When the Web Services are accessed from a server, you need to set up a login account on the domain that the blueButler server is connected to. That login account will have access to the entire blueButler system allowing you to control it from one server.

Note: when you use the Server Access method there are some Web Services that require you to append the user's blueButler User ID (known as the {buid} – it is either the extension or the agent id of the user) to the end of the Web Service request in order to identify the user's recording that you are referencing in the request.

User Desktop Access Method

When your application program runs on each user's desktop computer, the login account from the desktop computer is used to authenticate the Web Service request. For this method, there is no requirement to append the {buid} to the end of the request as the user id is already known.

Web Services – MP3 File Access

This group of Web Service commands is used to retrieve MP3 call recording files from blueButler.

GetMP3

http://{blueServer}/bluebutler/web/service.svc/MP3/{refNum}

Purpose: stream the MP3 audio file to the local system's media player

Variable: {refnum} is the Reference Number for the MP3 recording file

Notes:

This method uses Windows authentication to determine if the user is allowed access to the file. If this Web Service is accessed from a server, the Windows account on the server should be set to allow access to all

recording files. If you do not set the login account, it will use Anonymous and the Web Service will deny access to the file. If there is no domain, create the account on the server instead of in Active Directory in which case your domain would be <blueButler name server name>.

An alternative method of accessing the MP3 file is to use the ArchiveFileName field in the DrData table in blueButler's SQL database. This provides the pathname of the MP3 file allowing you direct access to the file rather than using the Web Service.

PlayByPhone

http://{blueServer}/bluebutler/web/service.svc/PlayByPhone/{refNum}/{phone_number}

Purpose: blueButler calls out to the phone number provided to play the MP3 audio file over the phone

Variable: {refnum} is the Reference Number for the MP3 recording file

{phone_number} is the phone number to be called

Notes:

This service requires the blueButler Dialogic Option to be installed and connected to your PBX in order to establish the phone connection.

ViewLog

http://{blueServer}/bluebutler/web/service.svc/Log

Purpose: return the list of recording records for the user

Variable: none

ViewLogToday

http://{blueServer}/bluebutler/web/service.svc/Log/Today

Purpose: return the list of the current day's recording records for the user

Variable: none

Web Services – SQL Database Updates

This group of Web Service commands is used to update the SQL database records in blueButler.

GetReferenceNumber

http://{blueServer}/bluebutler/web/service.svc/GetReferenceNumber

http://{blueServer}/bluebutler/web/service.svc/GetReferenceNumber/{buid}

Purpose: used to get the unique blueButler Reference Number for the active call

Variable: none

Notes:

When you use the Server Access method you need to append the user's blueButler User ID (known as the {buid} – it is either the extension or the agent id of the user) to the end of the Web Service request in order to identify the user's recording that you are referencing in the request.

UpdateDrData

http://{blueServer}/bluebutler/web/service.svc/Data/Update/{refnum}?subject={subject}&comments={comments}&isImportant={isImportant}&isBookmark={isBookmark}&direction={calltype}&digits={digits}&custom1={drdata1}&custom2={drdata2}&custom3={drdata3}&custom4={drdata4}&custom5={drdata5}&custom6={drdata6}&custom7={drdata7}&custom8={drdata8}&custom9={drdata9}&custom10={drdata10}

Purpose: to update one or more data fields in the SQL record for the call recording

Variable: {refnum} is the Reference Number for the MP3 recording file

Optional: {subject} is the Subject field in the record – maximum length is 255 characters

{comments} is the Comments field in the record – maximum length is 255 characters

{isImportant} is the Important field in the record – set to True or False

{isBookmark} is the Bookmark field in the record – set to True or False

{calltype} is the Call Type field in the record – set to In or Out

{digits} is the Dialed Digits field in the record – set to a numeric string

{drdata1} through {drdata10} are the Custom Data fields in the record

Notes:

Only include the fields that you want to update. The other fields can be omitted from the request. This service can be used to update an active call as well as MP3 call recordings that are in the SQL database.

Web Services – On-Demand Recording

This group of Web Service commands is used to control when and how the blueButler recording service records phone calls for specific users. All the OnDemand Web Services return a {refnum} that uniquely identifies the recording. Use {refnum} to identify this record when you call the **UpdateDrData** Web Service to add data to the SQL record for the call recording.

OnDemand_Start

http://{blueServer}/bluebutler/web/service.svc/OnDemand/Start

http://{blueServer}/bluebutler/web/service.svc/OnDemand/Start/{buid}

Purpose: used to start recording an MP3 file

Variable: none

Notes:

When you use the Server Access method you need to append the user's blueButler User ID (known as the {buid} – it is either the extension or the agent id of the user) to the end of the Web Service request in order to identify the user's recording that you are referencing in the request.

OnDemand_Pause

http://{blueServer}/bluebutler/web/service.svc/OnDemand/Pause

http://{blueServer}/bluebutler/web/service.svc/OnDemand/Pause/{buid}

Purpose: used to pause recording

Variable: none

Notes:

If the call ends in the pause state, the recording file is closed automatically and saved. When you use the Server Access method you need to append the user's blueButler User ID (known as the {buid} – it is either the extension or the agent id of the user) to the end of the Web Service request in order to identify the user's recording that you are referencing in the request.

OnDemand_Resume

http://{blueServer}/bluebutler/web/service.svc/OnDemand/Resume

http://{blueServer}/bluebutler/web/service.svc/OnDemand/Resume/{buid}

Purpose: used to resume recording

Variable: none

Notes:

When you use the Server Access method you need to append the user's blueButler User ID (known as the {buid} – it is either the extension or the agent id of the user) to the end of the Web Service request in order to identify the user's recording that you are referencing in the request.

OnDemand_Stop

http://{blueServer}/bluebutler/web/service.svc/OnDemand/Stop

http://{blueServer}/bluebutler/web/service.svc/OnDemand/Stop/{buid}

Purpose: used to stop recording

Variable: none

Notes:

If the call ends in the stop state, the recording file is closed automatically and saved. When you use the Server Access method you need to append the user's blueButler User ID (known as the {buid} – it is either the extension or the agent id of the user) to the end of the Web Service request in order to identify the user's recording that you are referencing in the request.

OnDemand_Delete

http://{blueServer}/bluebutler/web/service.svc/OnDemand/Delete

http://{blueServer}/bluebutler/web/service.svc/OnDemand/Delete/{buid}

Purpose: used to delete the recording file for the active call

Variable: none

Notes:

This service is used when you start recording a call but then decide to delete it before it is stored in blueButler. When you use the Server Access method you need to append the user's blueButler User ID (known as the {buid} – it is either the extension or the agent id of the user) to the end of the Web Service request in order to identify the user's recording that you are referencing in the request.

OnDemand_Save

http://{blueServer}/bluebutler/web/service.svc/OnDemand/Save

http://{blueServer}/bluebutler/web/service.svc/OnDemand/Save/{buid}

Purpose: used to save a recording file

Variable: none

Notes:

This service is used to set the "Keep" indicator for users who are configured for On-Demand recording. If the "Keep" indicator is set when the call ends, the recording is saved otherwise it is discarded. When you use the Server Access method you need to append the user's blueButler User ID (known as the {buid} – it is either the extension or the agent id of the user) to the end of the Web Service request in order to identify the user's recording that you are referencing in the request.

Web Services – Agent ID

This group of Web Service commands is used to assign an Agent ID to the active call when it is recorded.

AgentLoginByExtension

http://{blueServer}/bluebutler/web/service.svc/LoginAgentByExtension/{extension}/{buid}

Purpose: used to associate recordings to a specific agent id rather than the extension number

Variable: {extension} is the Agent ID that you want to assign

{buid} is the blueButler User ID

Notes:

This service will override the agent id processing in the recording service for the phone if it is also activated.

AgentLogout

http://{blueServer}/bluebutler/web/service.svc/LogoutAgent/{buid}

Purpose: used to disassociate an agent id that was previously assigned to a user profile

Variable: {buid} is the blueButler User ID

SECTION 3 – XML SCRIPTS Interface

The blueButler .NET Client application includes an XML Script feature that can be configured to meet your specific requirements. This section describes how to add or modify the scripts.

The scripts are located in the blueButler directory **C:\blueC802\blueButler\Data\<default>** where **<default>** is the name of the folder that contains the configuration and the scripts associated with it. You can have any number of folders and sub folders within the <default> folder to help organize larger numbers of scripts or more complex scripts.

Example:

C:\blueC802\blueButler\Data\<default>
\Privacy Policy
\Product1
\Product2
start.xml
end.xml

You might also choose to segment scripts by department. In that case, the scripts would each be given a different directory name. For instance, instead of using <default>, you might have separate folders blueButler\data\CustomerSupport, blueButler\data\Sales, etc.

Tab(s)

Every script will have at least one tab at the top, but it's possible to add several tabs. Each tab can have access to it's own drop down list to select different choices. NOTE: buttons are currently not supported.

Example:

```
<Popup_Tab Caption="Compliance">

    <!-- Caption: What gets displayed -->
    <!-- File: Where to go when selected -->
    <!-- Complete: Setting to "True" quits the sequence for this call -->

   <Popup_Options Caption="Please select a script">

    <Popup_Option Caption="Privacy Consent" blueButler="privacy" File="privacy\start.xml"/>

   </Popup_Options>

  </Popup_Tab>
```

This sample section of a script shows a new tab called "Compliance". The tab has one item in the drop down list at this time. The label in the drop down list is "Privacy Consent" and it will call the script located

in the privacy folder called start.xml. It will also send a message to blueButler with the word privacy that will be stored in the Events table in the SQL database for this recording.

Drop-down List

In a script you can have a drop down list. This is very useful when there will be several options to choose from when starting a call, or within in a specific section of a script.

Example:

```
<Popup_Options Caption="Please select a script">

  <Popup_Option Caption="Privacy Consent" blueButler="privacy" File="privacy\start.xml"/>
  <Popup_Option Caption="Auto" blueButler="Auto" File="auto\start.xml"/>
  <Popup_Option Caption="Life" blueButler="Life" File="life\start.xml"/>
  <Popup_Option Caption="Commercial" blueButler="Commercial" File="commercial\start.xml"/>

</Popup_Options>
```

This sample section has the label "Please select a script" and then the drop down list has 4 different options. Each of these options points to a different folder, with a start.xml script in each.

Button Selection

You can also include a 'button' in the script that the user can click on. There are no icons or images required – use an ASCII text string as the icon.

Example 1:

```
<Popup_Buttons>
  <Popup_Button Caption="Pass" File="start.xml"/>
  <Popup_Button Caption="Fail" File="privacy/end.xml" Separator="Hidden"/>
</Popup_Buttons>
```

In this example you will display "Pass | Fail". If you click on Pass, it goes to start.xml. If you select Fail, it selects end.xml. The end Seperator="Hidden" should be at the end of each list of buttons so that you do not have the ending "|" icon at the end of the display.

Example 2:

```
<Popup_Buttons>
  <Popup_Button Caption="&lt;&lt;" File="start.xml"/>
  <Popup_Button Caption="&gt;&gt;" File="finish.xml" Separator="Hidden"/>
</Popup_Buttons>
```

In this example we wanted to have forward and back arrows to indicate next and back. To create this type of 'icon' we have used the greater-than and less-than symbols. This example will look like "<< | >>". Clicking >> will select the finish.xml script.

Commands

Caption - This is the name, label or title that will be assigned. Used in popup_button, popup_caption, and popup_tab.

File - This will allow the script to call the xml script that is specified. Syntax can also have a folder name in the command: *File="privacy/end.xml"*

bluebutler - This command will insert text into the Events table of the SQL database for that recording. You can have as many events in a script as you want. *bluebutler="privacy"* would insert that word into the database, and you would know when during that recording (time stamp) that the user completed the privacy script.

On-Demand Commands

OnDemand="save" -- KEEP
OnDemand="delete" -- DISCARD (default for OnDemand user)
OnDemand="pause" -- PAUSE
OnDemand="start" -- KEEP (if OnDemand user) then START or RESUME if it was paused
OnDemand="resume" -- RESUME

NOTE: If the call is already in START state and a second START is issued, the initial recording will be discarded and a new recording begun.

SAMPLE Scripts

Below are a few sample scripts to refer to when you create or modify your own scripts.

Sample Script – Privacy Consent

```
<?xml version="1.0" encoding="utf-8" ?>
- <!--
 Displays text "Privacy Consent" with options Pass and Fail
  -->
- <Popup_Configuration>
  - <Popup_Tabs>
    - <Popup_Tab Caption="Privacy Consent">
      - <Popup_Options Caption="Remember to complete verbal privacy
        consent! In accordance with privacy legislation, we need to
        obtain your permission in order to collect, use, or retain your
        personal information. The personal information we collect
        includes, but is not limited to, your name, address, date of
        birth, driver's license and insurance history. This information
        will be used for the purposes of obtaining claims and licensing
        reports, preparing your proposal, and then upon acceptance of
        such, the ongoing maintenance and servicing of your policy(s).
        We will share your personal information with the insurers with
        whom we have arranged insurance programs that may be of
        interest to you. We do not sell your personal information to
        unrelated third parties. Do you agree?">
        - <!--
         Caption: What gets displayed
          -->
        - <!--
         File: Where to go when selected
          -->
        - <!--
         Complete: Setting to "True" quits the sequence for this
          call
          -->
        </Popup_Options>
```

```
      </Popup_Tab>
    </Popup_Tabs>
  - <Popup_Buttons>
      <Popup_Button Caption="Pass" File="start.xml" />
      <Popup_Button Caption="Fail" File="privacy/end.xml"
        Separator="Hidden" />
    </Popup_Buttons>
  </Popup_Configuration>
```

Sample Script – Main Screen

```
<?xml version="1.0" encoding="utf-8" ?>
- <!--
 Displays text "Home" and “Please select a script”
  -->
- <Popup_Configuration>
  - <Popup_Tabs>
    - <Popup_Tab Caption="Home">
      - <!--
       Caption: What gets displayed
        -->
      - <!--
       File: Where to go when selected
        -->
      - <!--
       Complete: Setting to "True" quits the sequence for this call
        -->
      - <Popup_Options Caption="Please select a script">
          <Popup_Option Caption="Privacy Consent"
            blueButler="privacy" File="privacy\start.xml" />
          <Popup_Option Caption="Auto" blueButler="Auto"
            File="auto\start.xml" />
          <Popup_Option Caption="Life" blueButler="Life"
            File="life\start.xml" />
          <Popup_Option Caption="Commercial"
            blueButler="Commercial" File="commercial\start.xml" />
        </Popup_Options>
      </Popup_Tab>
    </Popup_Tabs>
    <Popup_Buttons Close="Visible" />
  </Popup_Configuration>
```

SECTION 4 – FILE IMPORT Utility

User profiles in blueButler can be created using the File Import utility described in this section.

File Import

Importing user profile records into blueButler is a two-step process. First, import the extensions and then import the devices.

Step 1 – Create import files

Create the two import files – one for the extensions (e.g. C:\Extensions.csv) and the other for the phones (e.g. C:\Phones.csv). The files contain multiple fields that are mapped to corresponding fields in the blueButler SQL tables. You can change the order of the fields in the file and remove the optional fields that you are not using. Some of the fields that are commonly imported include:

Field	Description
buid	unique blueButler User ID (created and managed by the system)
blueEmployeeBuid	blueButler User ID
blueExtension	Extension number
blueAgentID	Agent ID (when Agent IDs are used to identify the user at a phone)
sn	Last name (Surname)
givenName	First name
userPassword	Password
blueDrPrivilege	User security (privilege) level
mail	Email address
blueCostCenter	Department or Cost Center for the user
druid	Device ID
blueDeviceType	Type of recording device
blueDrSerialNumber	IP address or MAC address for IP phone
blueDrTypeName	Model of recording device
blueRecordingMode	Automatic or On-Demand recording
blueRecordKeys	Keys to be recorded on the phone
blueTransferKey	Transfer key identifier
servuid	Server name

Example:

Phones.csv

blueEmployeeBuid	Servuid	blueAgentID	blueRecordKeys	blueDeviceType	blueDrSerialNumber
11001	Main	Yes	0,1,2	IP Device	00:1B:1F:1D:A1:10

Extensions.csv

blueExtension	buid	blueDrPrivilege	blueRecordingMode
11001	11001	Full Features	On-Demand

Step 2 – Open a cmd prompt

Open a **cmd** prompt in Windows and go to the **C:\bluec802\bin** directory.

Example:

```
C:\Documents and Settings\Administrator>cd..
C:\Documents and Settings>cd..
C:\>cd bluec802
C:\blueC802>cd bin
```

Step 3 – Import the Extensions file

Run the **ImportUsers cmd** line utility and enter the file to import when prompted.

C:\blueC802\bin>java -classpath .\;..\sqljdbc.jar;butler.jar;blueutil.jar ImportUsers

Example:

```
ImportUsersStarting
Enter the name of the import csv file:

C:\Extensions.csv

Importing data, please wait.....................................
Number of records read from import file = 35
Number of records imported = 35
ImportUsersStopping
```

Step 4 – Import the Phones file

Run the **ImportUserDrs cmd** line utility and enter the file to import when prompted.

C:\blueC802\bin>java -classpath .\;..\sqljdbc.jar;butler.jar;blueutil.jar ImportUserDrs

Example:

```
ImportUserDRsStarting
Enter the name of the import csv file:

C:\Phones.csv

Importing data, please wait.....................ERROR: Record Number 17: invalid
ip address or mac Address
ERROR: line 17 was rejected.....................
Number of records read from import file = 35
Number of records imported = 34
Number of records rejected = 1
send to blueIntercept: drrestart
blueIntercept returned: '-done'
java.net.ConnectException: Connection refused: connect
ImportUserDRsStopping

C:\blueC802\bin>
```

Note, the File Import utility will inform you if there are errors in the file. In this example one of the MAC addresses was incorrect.

Step 5 – Verify the import

Check in the blueButler .NET client application to see that all the users and phone devices were created properly. Note, there is no undo function so be careful to check the import data carefully; if you make mistakes you will need to manually edit the data using the blueButler user interface.

Step 6 – Restart the recording services

The blueButler recording service must be restarted on the server where the phones were added before the import will take effect. Restarting the blueButler server is the recommended method to follow to ensure all services are restarted correctly after an import.

SECTION 5 – SQL DATABASE

At the core of the blueButler system is a SQL database that contains tables of data records associated with the call recording files and the user profiles. These records have many fields that can be used to create reports using third-party report writers and/or queries from other application software. The SQL tables in blueButler are published for read and write access providing you with the opportunity to use the data records to meet your specific needs.

SQL Tables

The tables below list the fields in the primary data tables used in blueButler.

DrData table

This table contains the records for the call recording files. In particular, the **ReferenceNumber** field has the value of the unique reference number for the recording and fields **DrData 1** through **DrData10** are available to be used to associate your specific data with the call recording file.

TABLE DrData		
Field Name	**Type**	**Value**
drdatauid	VARCHAR(255)	NOT NULL PRIMARY KEY
DrComments	VARCHAR(255)	NULL
druid	VARCHAR(255)	NULL
DrSubject	VARCHAR(255)	NULL
DrCallerId	VARCHAR(255)	NULL
DrDialedDigits	VARCHAR(255)	NULL
Important	VARCHAR(255)	NOT NULL DEFAULT 'No'
SuperBuid	VARCHAR(255)	NULL
EmployeeBuid	VARCHAR(255)	NULL
DrStartTime	DATETIME	NULL
DrDuration	INT	NOT NULL DEFAULT 0
DrData1	VARCHAR(255)	NULL
DrData2	VARCHAR(255)	NULL
DrData3	VARCHAR(255)	NULL
DrData4	VARCHAR(255)	NULL
DrData5	VARCHAR(255)	NULL
DrData6	VARCHAR(255)	NULL
DrData7	VARCHAR(255)	NULL
DrData8	VARCHAR(255)	NULL
DrData9	VARCHAR(255)	NULL

TABLE DrData		
Field Name	**Type**	**Value**
DrData10	VARCHAR(255)	NULL
FileName	VARCHAR(255)	NULL
ArchiveFileName	VARCHAR(255)	NULL
EmployeeName	VARCHAR(255)	NULL
BuidCostCenter	VARCHAR(255)	NULL
SuperEval	VARCHAR(255)	NULL
RecordingServer	VARCHAR(255)	NULL
DNIS	VARCHAR(255)	NULL
EmailAddress	VARCHAR(255)	NULL
SendSMS	VARCHAR(255)	NOT NULL DEFAULT 'No'
GotFile	VARCHAR(255)	NOT NULL DEFAULT 'No'
IsDistList	VARCHAR(255)	NOT NULL DEFAULT 'No'
DistName	VARCHAR(255)	NULL
drconfiguid	VARCHAR(255)	NULL
SuperRating	VARCHAR(255)	NOT NULL DEFAULT '0'
PopUpStatus	VARCHAR(255)	NULL
AttachRecording	VARCHAR(255)	NOT NULL DEFAULT 'No'
UserTag	VARCHAR(255)	NOT NULL DEFAULT 'No'
ReferenceNumber	VARCHAR(255)	NULL
GroupFolder	VARCHAR(255)	NULL
FileType	VARCHAR(255)	NOT NULL DEFAULT 'MP3'
RestoreTimeSet	VARCHAR(255)	NOT NULL DEFAULT 'No'
RestoreDateLimit	DATETIME	NULL
ProcessingDone	INT	NOT NULL DEFAULT 0
FileStatus	VARCHAR(255)	NULL
GetTries	INT	NOT NULL DEFAULT 0
ConvertTries	INT	NOT NULL DEFAULT 0
ArchiveTries	INT	NOT NULL DEFAULT 0
ArchiveStatus	VARCHAR(255)	NULL
SplitMessage	VARCHAR(255)	NOT NULL DEFAULT 'No'
SplitReference	VARCHAR(255)	NULL
SplitPart	VARCHAR(255)	NULL
SplitPartNum	INT	NOT NULL DEFAULT 1
CallType	VARCHAR(255)	NULL
ListenedTo	VARCHAR(255)	NOT NULL DEFAULT 'No'
ReleasePressed	VARCHAR(255)	NOT NULL DEFAULT 'No'
TransferPressed	VARCHAR(255)	NOT NULL DEFAULT 'No'
EvalUser	VARCHAR(255)	NULL
FinalDestination	VARCHAR(255)	NOT NULL DEFAULT 'Yes'
HoldTime	INT	NOT NULL DEFAULT 0
TotalTime	INT	NOT NULL DEFAULT 0
CallHeld	VARCHAR(255)	NOT NULL DEFAULT 'No'
AbandonedOnHold	VARCHAR(255)	NOT NULL DEFAULT 'No'
TransferedTo	VARCHAR(255)	NULL
DrStartTimeOfDay	INT	NULL

TABLE DrData		
Field Name	**Type**	**Value**
OriginalSuffix	VARCHAR(255)	NOT NULL DEFAULT ''
EmailAddress2	VARCHAR(255)	NULL
EmailAddress3	VARCHAR(255)	NULL
NotifyDVDCopy	VARCHAR(255)	NOT NULL DEFAULT 'No'
CopyDVDComplete	VARCHAR(255)	NOT NULL DEFAULT 'No'
InTransit	VARCHAR(255)	NOT NULL DEFAULT 'No'

DrUserComments table

This table contains records that are linked to the DrData table records. The purpose is to allow long comment strings to be added by Users to a record without affecting the size of all DrData records.

TABLE DrUserComments		
Field Name	**Type**	**Value**
CommentId	INT	NOT NULL IDENTITY(1,1) PRIMARY KEY
drdatauid	VARCHAR(255)	NULL
Comment	VARCHAR(255)	NULL

DrSysComments table

This table contains records that are linked to the DrData table records. The purpose is to allow long comment strings to be added by Supervisors or Managers to a record without affecting the size of all DrData records.

TABLE DrSysComments		
Field Name	**Type**	**Value**
CommentId	INT	NOT NULL IDENTITY(1,1) PRIMARY KEY
drdatauid	VARCHAR(255)	NULL
Comment	VARCHAR(255)	NULL

Event table

This table contains records that are created whenever an event occurs that is related to a DrData recording record. For instance, the XML Scripts can add event records when a user selects a command in the script.

TABLE DrEvent		
Field Name	**Type**	**Value**
DrEventUID	INT	NOT NULL AUTO_INCREMENT PRIMARY KEY
EventId	VARCHAR(255)	NOT NULL
ReferenceNumber	VARCHAR(255)	NULL
TimeStamp	DATETIME	NULL
buid	VARCHAR(255)	NULL
EventTypeId	VARCHAR(255)	NOT NULL

DrRealTime table

This table contains records that are created when a call recording is in progress for an active call. These records contain information about the active call.

TABLE DrRealTime		
Field Name	**Type**	**Value**
Id	INT	NOT NULL IDENTITY(1,1) PRIMARY KEY
druid	VARCHAR(255)	NULL
DrStartTime	DATETIME	NULL
ReferenceNumber	VARCHAR(255)	NULL
DrCallerId	VARCHAR(255)	NULL
DrDialedDigits	VARCHAR(255)	NULL
DNIS	VARCHAR(255)	NULL
CallType	VARCHAR(255)	NULL

Alarm table

This table contains records of any event in blueButler that is considered unusual. They are used to notify the Administrator of potential problems in the system.

TABLE Alarm		
Field Name	**Type**	**Value**
alarmuid	VARCHAR(255)	NOT NULL PRIMARY KEY
Type	VARCHAR(255)	NULL
Sender	VARCHAR(255)	NULL
Server	VARCHAR(255)	NULL
Time	DATETIME	NULL
Event	VARCHAR(255)	NULL
Ack	VARCHAR(255)	NOT NULL DEFAULT 'No'

Users table

This table contains the records for each user profile that has been added to blueButler. The **buid** field is the unique key value for each record.

TABLE Users		
Field Name	**Type**	**Value**
buid	VARCHAR(255)	NOT NULL PRIMARY KEY
cn	VARCHAR(255)	NULL
sn	VARCHAR(255)	NULL
userpassword	VARCHAR(255)	NULL
mail	VARCHAR(255)	NULL
givenName	VARCHAR(255)	NULL
Title	VARCHAR(255)	NULL
HomeNumber	VARCHAR(255)	NULL
SecurityLevel	VARCHAR(255)	NULL

TABLE Users		
Field Name	**Type**	**Value**
OfficeLocation	VARCHAR(255)	NULL
WorkHours	VARCHAR(255)	NULL
LastLogon	DATETIME	NULL
CurrentForwardURL	VARCHAR(255)	NULL
ActiveScheduleRecord	DATETIME	NULL
LastVoiceMail	DATETIME	NULL
EmailPassword	VARCHAR(255)	NULL
SendVoiceMail	VARCHAR(255)	NULL
OverrideCyclic	VARCHAR(255)	NULL
AskCallerID	VARCHAR(255)	NULL
MobileTelephoneNumber	VARCHAR(255)	NULL
PagerTelephoneNumber	VARCHAR(255)	NULL
VoiceMailSent	VARCHAR(255)	NULL
CurrentForwardType	VARCHAR(255)	NULL
Extension	VARCHAR(255)	NULL
MailboxNumber	VARCHAR(255)	NULL
MailBoxPassword	VARCHAR(255)	NULL
AltContactExt	VARCHAR(255)	NULL
LastPasswordChange	DATETIME	NULL
AlternateContacts	VARCHAR(255)	NULL
OptionMessage	VARCHAR(255)	NULL
EmailPin	VARCHAR(255)	NULL
LastPinChange	DATETIME	NULL
AlternateUserID	VARCHAR(255)	NULL
WirelessInfoReceived	VARCHAR(255)	NULL
WirelessEmailLevel	VARCHAR(255)	NULL
MessageType	VARCHAR(255)	NULL
ForwardSetting	VARCHAR(255)	NULL
SetupWorkdaySchedule	VARCHAR(255)	NULL
RemoteFWDPassword	VARCHAR(255)	NULL
CallingPINOld1	VARCHAR(255)	NULL
CallingPINOld2	VARCHAR(255)	NULL
PasswordOld1	VARCHAR(255)	NULL
PasswordOld2	VARCHAR(255)	NULL
MakeCallState	VARCHAR(255)	NULL
MakeCall	VARCHAR(255)	NULL
DepartmentHead	VARCHAR(255)	NULL
AutoPassword	VARCHAR(255)	NULL
VoiceEmailType	VARCHAR(255)	NULL
Sendnotise	VARCHAR(255)	NULL
FailedLogins	VARCHAR(255)	NULL
FailedPin	VARCHAR(255)	NULL
SendPolicy	VARCHAR(255)	NULL
LockoutLogin	VARCHAR(255)	NULL
LockoutPin	VARCHAR(255)	NULL

TABLE Users		
Field Name	**Type**	**Value**
SMSAddress	VARCHAR(255)	NULL
MessOptionDefault	VARCHAR(255)	NULL
SuppressPopups	VARCHAR(255)	NULL
MP3Playback	VARCHAR(255)	NULL
MP3ConvQuality	VARCHAR(255)	NULL
MP3MessQuality	VARCHAR(255)	NULL
QuickResponse	VARCHAR(255)	NULL
MobilePassword	VARCHAR(255)	NULL
VirtualExtension	VARCHAR(255)	NULL
CallingPIN	VARCHAR(255)	NULL
SurveyID	VARCHAR(255)	NULL
PlayMessage	VARCHAR(255)	NULL
CostCenter	VARCHAR(255)	NULL
VoiceEmailAttach	VARCHAR(255)	NULL
IncludeFileLink	VARCHAR(255)	NULL
MP3ID3Setting	VARCHAR(255)	NULL
AuthCode	VARCHAR(255)	NULL
BuidFileName	VARCHAR(255)	NULL
NameRecorded	DATETIME	NULL
GreetuidActive	VARCHAR(255)	NULL
VoxNameFile	VARCHAR(255)	NULL
GreetingDefault	VARCHAR(255)	NULL
GreetingActive	VARCHAR(255)	NULL
UserFlag	VARCHAR(255)	NULL
ChairCode	VARCHAR(255)	NULL
MemberCode	VARCHAR(255)	NULL
SaveMessageDays	INT	NULL
Language	VARCHAR(255)	NULL
NameFile	VARCHAR(255)	NULL
NewMessages	INT	NULL
LastMessage	DATETIME	NULL
AdminListen	VARCHAR(255)	NULL
AutoMessagePlay	VARCHAR(255)	NULL
MWI	VARCHAR(255)	NULL
VoiceEmailReply	VARCHAR(255)	NULL
OneNumberOnlyUser	VARCHAR(255)	NULL
AllowDrReplay	VARCHAR(255)	NULL
DrPrivilege	VARCHAR(255)	NULL
drconfiguid	VARCHAR(255)	NULL
DrConfigHist	VARCHAR(255)	NULL
DriveName	VARCHAR(255)	NULL
IsDefault	VARCHAR(255)	NULL
PreferredDrive	VARCHAR(255)	NULL
DrAllowDelete	VARCHAR(255)	NULL
DrEmailMsgs	VARCHAR(255)	NULL

TABLE Users		
Field Name	**Type**	**Value**
DrAdminListen	VARCHAR(255)	NULL
DrListener	VARCHAR(255)	NULL
Popup	VARCHAR(255)	NULL
RecordingMode	VARCHAR(255)	NULL
Executive	VARCHAR(255)	NULL
DNISGroup	VARCHAR(255)	NULL
LastLoginListener	VARCHAR(255)	NULL
DrViewer	VARCHAR(255)	NULL
Integrate3rdParty	VARCHAR(255)	NULL
IsEvaluator	VARCHAR(255)	NULL
AcdPassword	VARCHAR(255)	NULL

CostCenter table

This table contains the records for the cost centers / departments that users are grouped into in blueButler.

TABLE CostCenter		
Field Name	**Type**	**Value**
costcenteruid	VARCHAR(255)	NOT NULL PRIMARY KEY
DepartmentHead	VARCHAR(255)	NULL
Comments	VARCHAR(255)	NULL
CellPhoneBrand1	VARCHAR(255)	NULL
CellPhoneModel1	VARCHAR(255)	NULL
Country	VARCHAR(255)	NULL
Carrier1	VARCHAR(255)	NULL
Carrier2	VARCHAR(255)	NULL
CellPhoneBrand2	VARCHAR(255)	NULL
CellPhoneModel2	VARCHAR(255)	NULL
ContractTerm	VARCHAR(255)	NULL
WirelessServices	VARCHAR(255)	NULL
CellPhoneFeatures	VARCHAR(255)	NULL
OtherMobileDevices	VARCHAR(255)	NULL
PolicyLevel	VARCHAR(255)	NULL
SendNotise	VARCHAR(255)	NULL
PolicyID	INT	NULL
PolicyEmailLevel	VARCHAR(255)	NULL
PolicyInfoReceived	VARCHAR(255)	NULL
drconfiguid	VARCHAR(255)	NULL
DrConfigHist	VARCHAR(255)	NULL
Integrate3rdParty	VARCHAR(255)	NULL