Brekeke PBX

Version 2

ARS Plug-in Developer's Guide

Brekeke Software, Inc.

Version

Brekeke PBX Version 2 ARS Plug-in Developer's Guide
Revised February 2010

Copyright

This document is copyrighted by Brekeke Software, Inc.

Copyright © 2010 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. Introduction

This plug-in interface provides enhancements on ARS Route Search functionality of Brekeke PBX. This plug-in is available for Brekeke PBX version 2.1 or later. You will need to use the Java programming language to create your own plug-in.

ARS Route Search function uses conditions (matching patterns) for SIP headers (From, To, etc.). Those conditions, written as regular expressions, make ARS Search very flexible. Powerful uses of the ARS Plug-in include:

- Searching a telephone directory for a caller's number. For example, if the caller is in Do-Not-Call list, you can then decline the call.
- Searching for the least cost route by country number and area code number (Least Cost Routing).
- Searching a telephone directory for a caller's name using the caller's number. You can then change the display name with the caller's name.

Note: For simple searches, you can use [Options] menu > [Notes] for making data list in the PBX admin tool. There are default plug-ins for [Notes] menu, so you don't have to create your own plug-in for it.

2. Requirements

Creating an ARS Plug-in requires:

- JDK 1.4 or later
- Brekeke PBX v2.1 or later

3. Steps to Create an ARS Plug-in

1. Create a java class. You can use any name for your java package and class.

2. Create a method in your class with the following format:

    public static String plugin(String param)

    or

    public static boolean plugin(String param)

Although this example uses the function name "plugin", you may use any function name you desire.

4. Installation and Setup

4.1. Compile your class and place it in the classpath

Place your compiled class file in exact directory structure of package name under the directory:

<Brekeke PBX install_directory> /webapps/pbx/WEB-INF/classes

For example:

class name = YourClass, package name = com.yourdomain

Place "YourClass.class: under the directory:

<Brekeke PBX install_directory> /webapps/pbx/WEB-INF/classes/com/yourdomain

or compress your class file into a jar file and place the jar file into

<Brekeke PBX install directory>/webapps/pbx/WEB-INF/lib

4.2. Configure Brekeke PBX to use the new plug-in using the Admintool

1. Log into the Brekeke PBX Admin tool.

2. Navigate to ARS menu > Settings.

3. Edit an existing Route or Create a New Route.

4. Under Patterns > Matching patterns, set the following fields:

[Plugin] field

Specify your plug-in class and method name including your package name.

For example: yourpackage.YourPluginClass.yourMethod

[Param] field

Set the parameter which will be passed to your plug-in. You can use the text strings specified using bracket ( ) in the fields (such as To, From, etc.) for setting the parameter.

The variables for referring to those text strings in each field are:

[From]   &f1, &f2 …. &f9

[To]   &t1, &t2 …. &t9

[User]   &u1,&u2 …. &u9

[Return] field

Boolean or String will be returned by the plug-in.

- If your plug-in returns a boolean, this field will not be used. If your plug-in returns true, this ARS pattern will be applied.

- If your plug-in returns a String, you can set the matching condition using regular expressions in this field. If the condition is fulfilled, this ARS pattern will be applied. You can use the variables (&p1, &p2, ….., &p9) which refer to the text strings in [Return] field to set each field in Deploy Patterns.

5. A Sample Plug-in

5.1.   Code Sample – ARSMatchingSample.java

```java
package  yourpackage;

import java.util.*;
import java.util.regex.*;
import com.brekeke.pbx.common.*;

public class ARSMatchingSample {

        private static long lastmodified = 0;
        private static ArrayList patternlist = null;

        public static synchronized String regex( String param ) throws Exception {
                long l = NoteUtils.lastModified( "Regex" );
                if( l == 0 ){
                        return null;
                }
                if( lastmodified != l ){
                        lastmodified = l;
                        String s = NoteUtils.read( "Regex" );
                        if( s == null ){
                                return null;
                        }
                        ArrayList al = new ArrayList();
                        StringTokenizer st = new StringTokenizer( s );
                        while( st.hasMoreTokens() ){
                                String token = st.nextToken();
                                Pattern pt = Pattern.compile( token );
                                al.add( pt );
                        }
                        patternlist = al;
                }
                for( int i = 0; i < patternlist.size(); i++ ){
                        Pattern pt = (Pattern) patternlist.get(i);
                        Matcher mt = pt.matcher( param );
                        if( mt.matches() ){
                                return mt.group(1);
                        }
                }
                return null;
        }
}
```

This sample program refers to a note called "Regex" and matches using regular expressions. The program processes matching data strings in all rows in order starting from the top row. When the program finds a matched row, it returns the matched character strings inside parenthesis (). When the program can not locate a matched row, it returns null. (When null is returned, the patterns are considered mismatched.) The content of the note is cached in the variables. When a note is renewed, the program will re-read the content.

5.2. Example of ARS Setup

Set the following fields in ARS menu > Patterns >Matching patterns. Choose appropriate patterns (IN or OUT) depending on the environment.

Matching Pattern

From	
To	sip:(.+)@
User	
Plugin	yourpackage. ARSMatchingSample regex
Param	&t1
Return	(.+)

Deploy Pattern

From	
To	sip:&p1@domain.com
DTMF	
Target	

In the above example, "user-info" section of To header is set as a parameter. The remainder value "&p1" is drawn out and set at caller's To header.

5.3. Example of Notes Setup

Select the menu [Options] > [Notes]. PBX plug-ins can use the text in the notes.

Name	Description
Name	Name of the note.
Description	A brief description of the note
User access level	Access level Select from "No Access", "Read only", "Read/Write"
Note	Text field where you can write your own notes.

Name this note "Regex".

```
^1650(.+)$
^1888(.+)$
^1800(.+)$
```

When the number is starting with "1650", "1888", or "1800", the rest of the number will be drawn out of parameter.

6. Default Plug-ins for Notes

Brekeke PBX offers the following default plug-ins that use Notes.

contains

Parameters:	Note Name, search string
Returns:	Boolean. Search the specified Note for the specified search string.
	If any row in the Note matches with the search string, return true.

lookup

Parameters:	Note Name, search string, column index for search (default=1), column index for return (default=2)
Returns:	String. This plug-in will search the specified Note.
	Comma separated values should be set in each row in the Note. This method will search "the column for search" for the specified text string. If there is a row which matches with the text string (the first occurrence), the value in "the column for return" of the corresponding row will be returned. If there is no match, text string with the length 0 will be returned.

7. Class NoteUtils API

Package: com.brekeke.pbx.common

This class is used to access Notes. When referring to or editing a note, the methods in this class should be used.

7.1. read

public static String **read**(String name)

Read contents of the note

Parameters:

name   Name of the note

Returns:

Contents of the note

7.2.   write

public static boolean **write**(String name, String text)

Write into the note

Parameters:

name   Name of the note

text   Character string which will be written in the note

Returns:

Returns "true" on success, "false" on fail

7.3.   lastModified

public static long **lastModified**(String name)

Returns the time when the note was last edited.

Parameters:

name   Name of the note

Returns:

Return time of the last edit using "long" value. "OL" will be returned when the note does not exist or error occured

7.4. exists

public static boolean **exists**(String name)

Look for the specified note

Parameters:

name   Name of the note

Returns:

Return "true" when specified note exists, if not, returns "false".