Subject: PEC IVR Inter-Machine Call Transfers using Dialogic HMP firmware

Parwan Electronics Corporation
www.voicesaver.com
Tel. (732)-290-1900 x 3100
February 7, 2019
No. 2019-0204

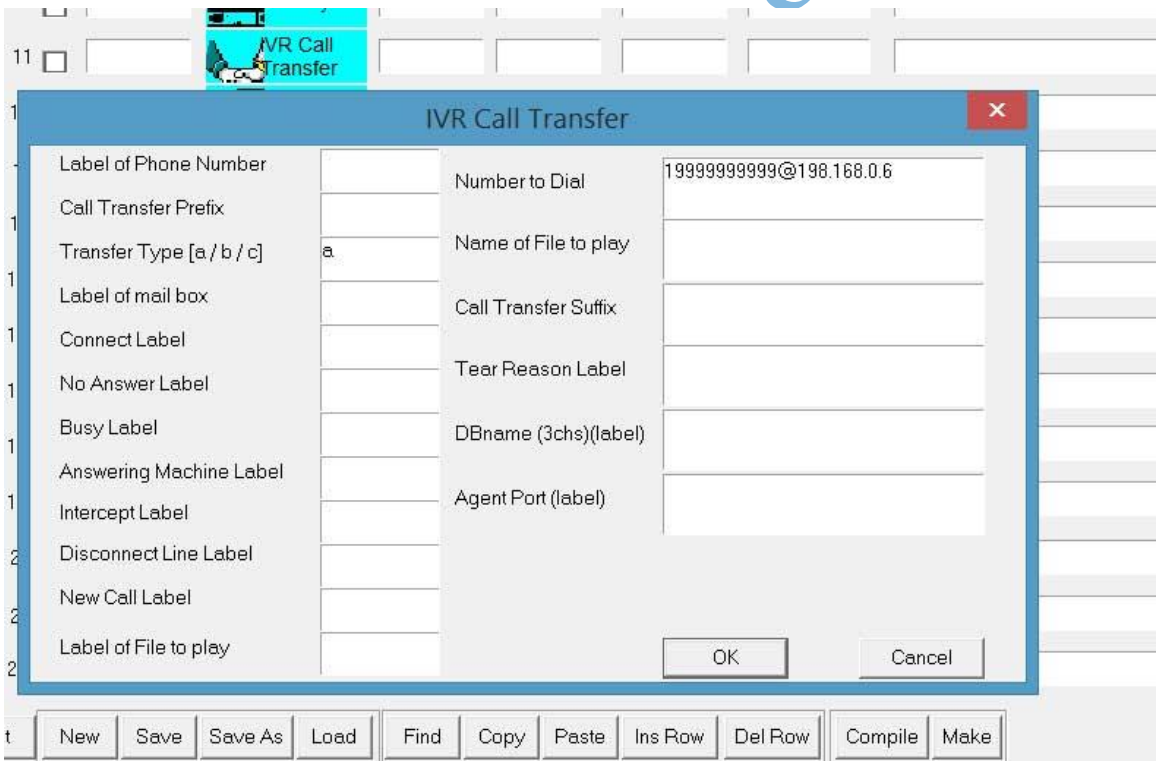
TECHNICAL MEMORANDUM

Description

The PEC Insight™ IVR Call Transfer block supports the blind transfer of a call from one machine running HMP to another machine using HMP. Both machines must be connected to Cisco or other voice gateways that route call to the machines running PEC software. Cisco must be provisioned to route the call to the extension specified in the Number to Dial field of the IVR Call Transfer block.

IVR Call Transfer from the Transferred Machine

The IVR Call Transfer block must be set as follows:

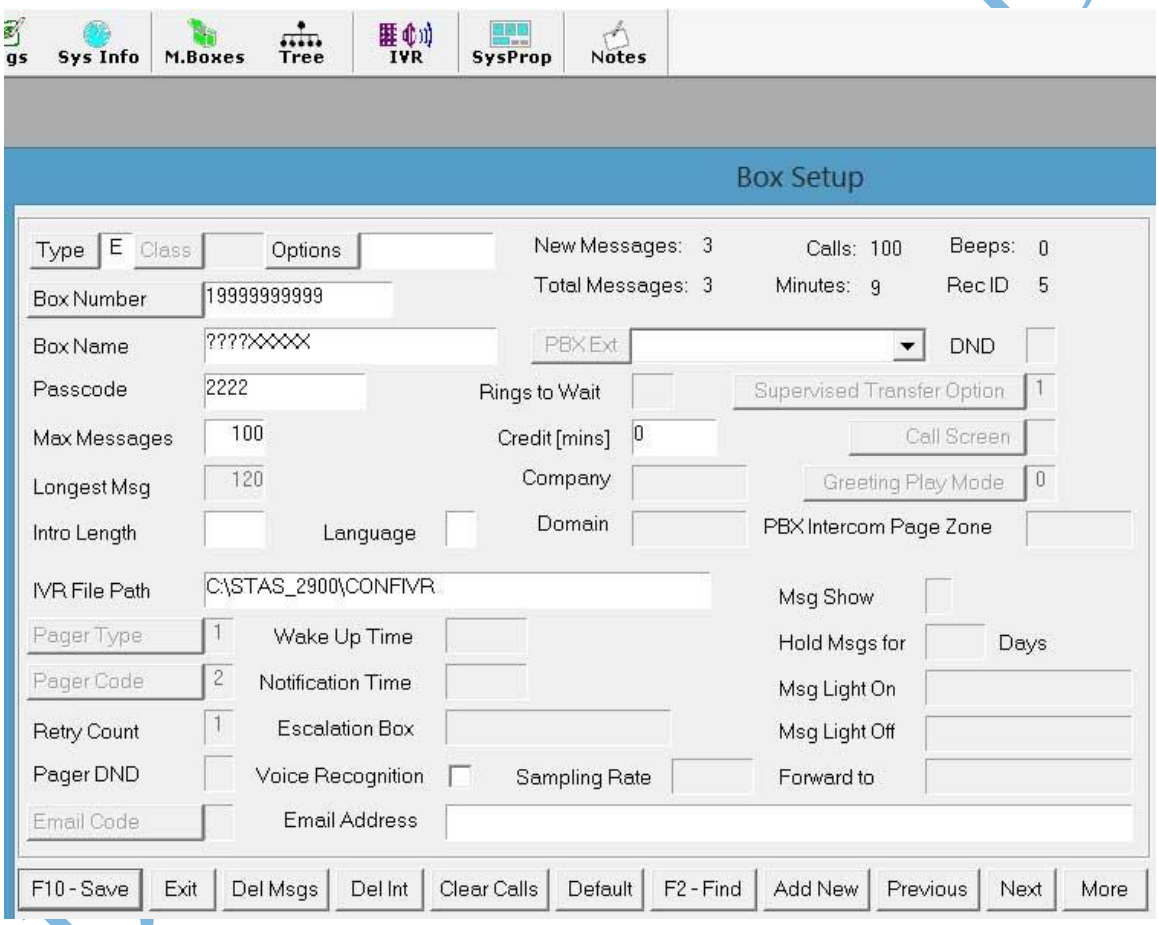


Once the IVR Call Transfer block is processed the software transfers the call to the address specified in the Number to Dial field of the block. Options the address may be specified in the Label. Along with the extension and IP address, the the software sends the five XML tag variables to the transferee machine. In order to the do the software send the "Referred-By: sip:1@1.1.1.1;%s" to Cisco, where %s corresponds the XML tag data. In the example above 19999999999 is the extension number and 198.168.0.6 is the IP

address of the transferred machine. Make sure Cisco routes the call to the transferee machine, Please note your XML tag variable data should not have any blanks in them.

Handling the Call on the Transferee Machine.

The transferee machine receives the call in the mailbox associated with the extension of the Number of Dial of the IVR Call Transfer block. The mailbox on the Transferee machine must an E type mailbox. The software reads the Cisco message received and move the XML tag data into corresponding variables. Cisco sends the "Requested-By: sip:1@1.1.1.1;%s" message. The IVR logic can use the XML variables to do the further processing. The following image shows how the mailbox must be set:



In the example above, the call is received in the 19999999999 mailbox and the IVR in the folder c:\stas_2900\confivr further processes the call.

Cisco Programming

Cisco must be programed to route the call to the proper machine. Before the data is sent to Cisco, the software converts the data for Cisco. The following C code is used:

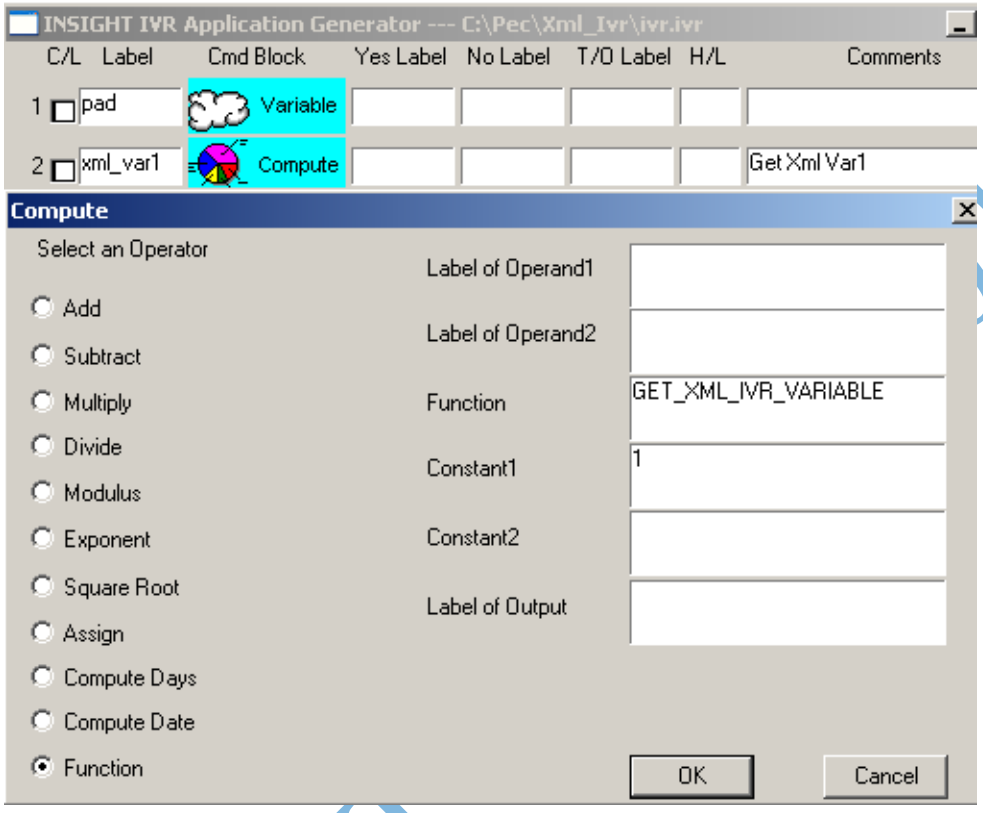
```
#include <stdio.h>
#include <ctype.h>

char rfc3986[256] = {0};
char html5[256] = {0};

void encode( const char *s, char *enc, char *tb)
{
	for (; *s; s++)
	{
		if (tb[*s])
		{
			sprintf(enc, "%c", tb[*s]);
		}
		else
		{
			sprintf(enc, "%%%02X", *s);
		}
		while (*++enc);
	}
}

Void
PEC_Transferred_From_Machine( char *zXml_Var, char *zData_to_Cisco )
{
	char *url ; // ORGIANAL XML DATA
	char *enc ; // ENCODED FOR CISCO

	url = zXml_Var_Data;
	enc = zData_For_Cisco;

	int i;
	for (i = 0; i < 256; i++)
	{
		rfc3986[i] = isalnum(i)||i == '~'||i == '-'||i == '.'||i ==
'_'
			? i : 0;
		html5[i] = isalnum(i)||i == '*'||i == '-'||i == '.'||i ==
'_'
			? i : (i == ' ') ? '+' : 0;
	}

	encode(url, enc, rfc3986);
	puts(enc);

	return 0;
}
```

When the software received the data from Cisco, it decodes the data and move it to the XML tag variables. The following C code is used to do the decoding:

```
#include <stdio.h>
#include <string.h>

Void
PEC_Transferred_To_Machine( char *zXml_Data, char *zData_From_Cisco )
{
      decode( zData_From_Cisco, zXml_Data );
}


inline int ishex(int x)
{
      return      (x >= '0' && x <= '9') ||
            (x >= 'a' && x <= 'f') ||
            (x >= 'A' && x <= 'F');
}

int decode(const char *s, char *dec)
{
      char *o;
      const char *end = s + strlen(s);
      int c;

      for (o = dec; s <= end; o++) {
            c = *s++;
            if (c == '+') c = ' ';
            else if (c == '%' && (!ishex(*s++) ||
                  !ishex(*s++) ||
                  !sscanf(s - 2, "%2x", &c)))
                  return -1;

            if (dec) *o = c;
      }

      return o - dec;
}
```

Reading the Xml Tag Variables by the IVR

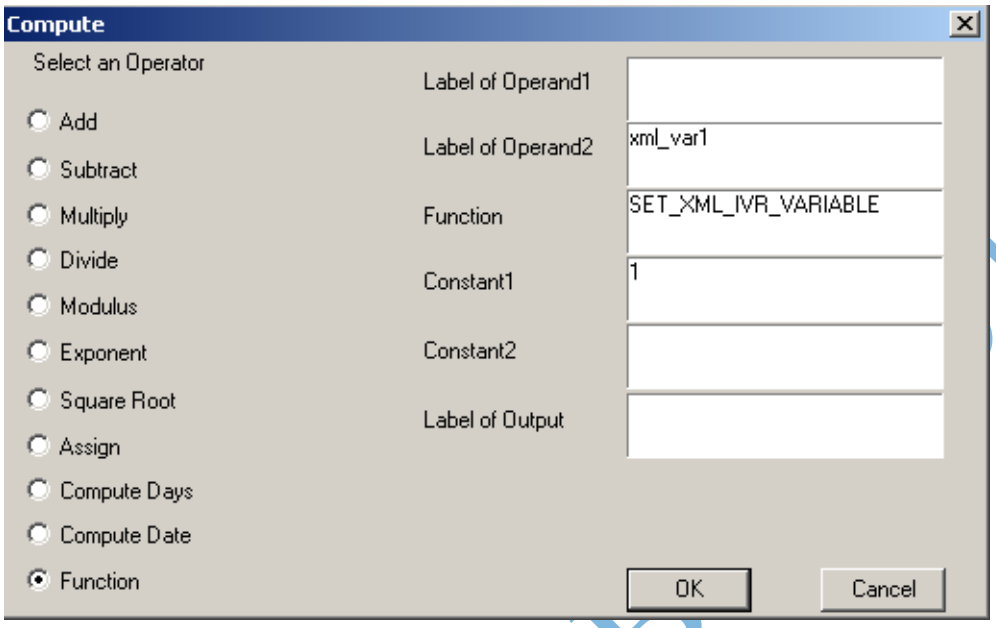
The IVR Applications Developer must use the Compute block of the Applications Generator to get the Xml tag variables into a label variable. Please see the following example:



The above compute block will get the Xml tag Var1 to the xml_var1 label. In order to get the Xml tag Var2, set the Constan1 to 2.

Setting the Xml Tag Variables from the IVR

The IVR Applications Developer must use the Compute block of the Applications Generator to set the Xml tag variables. Please see the following example:



The above compute block sets the Xml tag var1 to the data in the xml_var1 label. To set the Xml tag var2, then set Constant1 to 2.