MANAGER TOOLKIT
ICX

KEY WORDS: ICX, LEGRAND, USRMGR

EVOLUTIONS

Revision	Writer	Action	Date
1.0	ED	Initial version	21/03/2016

content

Scope of this document

The scope of this document is to describe and explain how to set-up the use of the svmgrICX.dll.

Introduction

The svmgrICX.dll enables to communicate with the COMMEND Intercom Exchange GE800 using TCP/IP**.**

The DLL is client and open the communication channel with the GE800 server. Thereafter the DLL waits for ICX messages for indication of events. The DLL can also sends out ICX Commands for control of event in the intercom Server.

A keep alive is sent cyclically by the task in orderto validate the connection to the GE 800 server.

To get information regarding the svmgrICX task while running the application, please check the event viewer for lines starting with "svmgrICX..."

Project structure

Activating the functionality

In order to activate the task it is necessary to copy the svmgrICX.dll into the BIN directory of PcVue and to rename the file svmgrICX_UsrMgr.dat into usrmgr.dat.

If the file svmgrICX_UsrMgr.dat is missing you can simply create a text file named usrmgr.dat with the following content

```
[USRMGR\usrmgrICX]
DLL = svmgrICX.dll
```

Configuring the task

The task is configured via a single file located in the project PER directory.

YourProjectName\PER

usrmgrICX.ini: Used to configure the usrmgrICX.dll task.

File description

usrmgrICX.ini

Please note that if an entry is omitted from the file or if the file is not present in the project then its default value will be applied by the task.

The file format is:

```
[TaskParameters]
AttributeNumber = 14
AssociationName =
Key = ICX#
SuffixParam = _PARAM
ParamResetString =

[_GE800Server]
IpAddress  = 192.168.1.1
Port          = 3671
ReconnectionPeriodInSec = 10
WatchdogInSec = 60
```

Where:

[TaskParameters]

AttributeNumber: Specify the extended attribute used for configuring a ICX variable.
Format: Number from 3 to 16. If omitted, **default is 16**

AssociationName: Specify the association name in a redundant architecture. The task is only active on the active server.
Format: String. If omitted, **default is an empty string**

Key: Specify the string the task has to find in the extended attribute specified above to consider the variable as and ICX variable.
Format: String. If omitted, **default is ICX#**

SuffixParam: Specify the variable parameter suffix one has to create to be able to pass the interphone called or that called you.
Format: String. If omitted, **default is _PARAM**

ParamResetString: Specify the string that is set to the variable varname_PARAM when it is reset.
Format: String. If omitted, **default is an empty string**

[STATION_NAME_GE800Server]: Specify the definition of the GE800 interface for the station "station_name". If you are not in a multu stion architecture the key is [_GE800Server]

IpAddress: Specify the Ip address of the GE800 Server the task has to connect to.
Format: String. If omitted from the file **default is 127.0.0.1**

Port: Specify the port used to communicate with the interface.
Format: Number
If entry omitted from the file the **default is 18000**

ReconnectionPeriodInSec: Specify the period in second at which the driver try to reconnect to a failed or in error interface.
Format: Number. If omitted, **default is 10 s**

WatchdogInSec: Specify watchdog period after which if we do not receive any packet form the server we consider that there is a connection error. The keep alive frame sent by the task is sent every WatchdogInSec /2 secondes.
Format: Number
If entry omitted from the file the **default is 60 s**

Message and command supported

The ICX protocol offers a very large range of messages and command. Please note that the driver supports only the short format frame structure.

Task	Type	Param1	Param2	Description	Attribute value	Variable type	Behaviour
42 5B 5B	13 21 22	aaaa - -	bbbb bbbb bbbb	Call to 'bbbb' from 'aaaa' Public call request from interphone 'bbbb' Public call request from interphone 'bbbb'	aaaa:CallRequest	BIT*	Variable with attribute aaaa:CallRequest is set to 1
42	12	aaaa	bbbb	Call in progress between 'aaaa' and 'bbbb'	aaaa:CallInProgress	BIT*	Variable with attribute aaaa:CallInProgress and bbbb:CallInProgress are set to 1
42 42	10 14	aaaa aaaa	bbbb bbbb	Call ended between 'aaaa' and 'bbbb' Call ended 'bbbb' busy	aaaa:CallEnded	BIT*	Variable with attribute aaaa:CallEnded and bbbb:CallIEnded are set to 1
42	05	aaaa	XXYY	Default XX with state YY State YY = 00 : No fault YY = 01 : Fault active	aaaa:DefaultXX	BIT	Variable with attribute aaaa:DefaultYY is set to 0 or 1 according to the default state
40	80	aaaa	FFFF	Cancel call for 'aaaa'	aaaa:CancelCall	BIT command	Forcing the variable to 1 with extended attribute aaaa:CancelCall will force the call cancel for aaaa
40	80	aaaa	bbbb	Buildup call from 'aaaa' to 'bbbb'	aaaa:BuildupCall	REGISTER command	Forcing the variable to value bbbb with extended attribute aaaa:BuildupCall will force the call to be made

5B	40	Cxxx	-	Output xxx open	Cxxx	Cxxx	
5B	41	Cxxx	-	Output xxx closed	Cxxx	Cxxx	
42	AF	Dxxx	C200	Input Dxxx opened	Dxxx	Dxxx	
42	AF	Dxxx	C204	Input Dxxx closed	Dxxx	Dxxx	

* aaaa:CallRequest, aaaa:CallInProgess and aaaa:CallEnded are mutually exclusive that means that when one variable is set to 1 all the other respective variable for the interphone are set to 0.

The user can record the interphone number called or that has called by creating a TEXT variable with suffix _PARAM. For instance is a BIT variable with the attribute aaaa:CallInProgress is called PARK_03.PHONE_99.IN_COMM. The TEXT variable PARK_03.PHONE_99.IN_COMM_PARAM will be set with the with the Interphone number that you are colling or that as call you accordingly.

Variable definition

To define an interphone in the supervisor you simply need to specify in the extended attribute (number specify in the usrmgrICX.ini file) of the variable:

Interphone activity variable definition

ICX#InterphoneNumber:Id

ICX#: Key to consider the variable to be an ICX variable

InterphoneNumber: Interphone number as receive in the frame for instance if the interphone number is 109 the user must enter **F109**.

Id: For each interphone a minimum of 3 variables must be defined.

CallRequest : set to 1 when the interphone is requesting a call.Reset to 0 otherwise.

CallInProgess : Set to 1 when the communication is established with the destination or when another interphone called the interphone.

CallEnded : Set to 1 when the communication is ended.

If you wish to record the interphone that is being called or that is the caller you simply need to create a text variable with the correct suffix (default value being _PARAM).

Interphone control variable definition

ICX#InterphoneNumber:Id[#InterfaceIP]

ICX#: Key to consider the variable to be an ICX variable

InterphoneNumber: Interphone number as receive in the frame for instance if the interphone number is 109 the user must enter **F109**.

Id: For each interphone a minimum of 3 variables must be defined.

CancelCall#InterfaceIP : If the variable is set to one it forces the GE800 server to cancel the call for the interphone specified.

BuildupCall#InterfaceIP : If the variable is set to a value different to 0 forces the GE800 server to build up a call for the interphone specified to the interphone specify by the register value.

For booth these parameter, you must specify an additional parameter that give the task the address of the GE800 server to send the command to for intake 127.0.0.1. If no interface IP is specified the command is simply ignored by the task.

Interphone default variable definition

ICX#InterphoneNumber:DefaultYY

ICX#: Key to consider the variable to be an ICX variable

InterphoneNumber: Interphone number as receive in the frame for instance if the interphone number is 109 the user must enter **F109**.

DefaultYY: represent the default number YY that is received. The variable is set to 1 when the default is on and set to 0 when the default is off

Interphone Input/Output variable definition

ICX#Cxxx:Val or ICX#Dxxx:Val

ICX#: Key to consider the variable to be an ICX variable

Cxxx: Output number xxx.

Dxxx: Input number xxx.

Trace

Additional trace can be triggered using the SCADA BASIC verb

TRACE(1/0,11,FlagBit);

Where

1/0	1: Turn ON the additional trace
	0: Turn OFF the additional trace
11	Default value not to be modified
Flagbit	hexadecimal combination of flag
	001 To view request not yet managed by the driver
	002 To view all request received

Example

TRACE(1,11,"003"); 'Turn on additional trace for read and write

TRACE(0,11,"003"); 'Turn off all traces.