Roger_RemoteMon.ocx v.1.0

ActiveX Control
requires *RACS* version 3.8 or 3.9

ROGER sp.j.
Gościszewo 59
82-416 Gościszewo
Poland

tel. 055 2720132
fax 055 2720133
www.roger.pl

18 March 2004

Description

The ***Roger_RemoteMon.ocx*** control software is dedicated for those systems integrators which require some new features to be implemented in ***Roger Access Control System***. The application build around this *ActiveX Software Control* can be run on remote computers which communicate with *PR Master* through TCP/IP protocol or can be run on the same computer as *PR Master*. In both cases a *Monitoring Window* of *PR Master* must be activated, when this window is not active a *Roger_RemoteMon.ocx* can not operate properly.

Note: The *Roger_RemoteMon.ocx* v1.0 requires *RACS* version 3.8 or 3.9

In order to use Roger_RemoteMon.ocx ActiveX Control in individual projects the following steps must be done:

1. *Registration of ActiveX control.*
2. *Import of ActiveX Control to dedicated programming environment.*

Registration of ActiveX control

Roger_RemoteMon.ocx ActiveX control should be registered using following command: "RegSvr32 Roger_RemoteMon.ocx".

Import to programming language

The Roger_RemoteMon.ocx ActiveX control can be imported to any programming language which accept ActiveX.
Example:

- *Delphi, C++Builder (Menu Component/Import ActiveX Control…)*
- *MS Visual Studio (Menu Tools/Customize Toolbox/COM Components).*

The API specification

Properties

- property _PRMasterHostIP: WideString – *An IP number of the PC machine on which a PR Master is installed*

- property _PRMasterHostPort: SYSUINT – *The TCP communication port on PC with PR Master*

- property _Login: WideString – *Login procedure, **Name** of program operator*

- property _Password: WideString – *Login procedure, **Password** of program operator*

- property _Connected: WordBool – *true: connect with remote PC, false: disconnect from remote PC*

- property _ShowMessage: WordBool – *when true message box will be displayed*

- property _ShowErrorMessage: WordBool – *when true error messages box will be displayed*

- property _GET_ReaderNamesXML: WideString – *downloads XML string with names of controllers (same as* ***Commands/Controller Commands/…*** *in PR Master.)*

An example of XML structure (including some data):

```
<tables>
   <table>
      <row>
            <col>Network A</col>
            <col>Default</col>
            <col>PR301</col>
      </row>
      <row>
            <col>Network A</col>
            <col>Default</col>
            <col>PR401</col>
      </row>
   </table>
</tables>
```

- property _GET_ZonesXML: WideString – *downloads XML string with names of Access Zones (same as* ***Commands/Zones Commands/...*** *in PR Master)*

An example of XML structure (including some data):

```
<tables>
   <table>
      <row>
            <col>Default</col>
      </row>
   </table>
</tables>
```

- property _GET_UsersXML: WideString – *downloads XML string with names of users (same as* ***View/View Users*** *in PR Master)*

An example of XML structure (including some data):

```
<tables>
   <table>
      <row>
            <col>True</col>
            <col>0</col>
            <col>Master Arnold</col>
            <col>False</col>
            <col>MASTER</col>
            <col></col>
            <col>No group</col>
      </row>
      <row>
            <col>True</col>
```

```
            <col>2</col>
            <col>Smith Mark</col>
            <col>False</col>
            <col>SWITCHER Full (1..49)</col>
            <col></col>
            <col>No group</col>
      </row>
      <row>
            <col>True</col>
            <col>1</col>
            <col>Popkin Jonathan</col>
            <col>False</col>
            <col>SWITCHER Full (1..49)</col>
            <col></col>
            <col>No group</col>
      </row>
   </table>
</tables>
```

- property _GET_GroupsXML: WideString – *downloads XML string with list of user group (same as **View/View Groups** in PR Master)*

An example of XML structure (including some data):

```
<tables>
   <table>
      <row>
            <col>No group (No group)</col>
            <col>0</col>
      </row>
      <row>
            <col>New group (1)</col>
            <col>1</col>
      </row>
      <row>
            <col>New group (2)</col>
            <col>2</col>
      </row>
   </table>
</tables>
```

- property _GET_SubsystemsXML: WideString – *downloads XML string with list of access control networks (same as **View/View Subsystems** in PR Master)*

An example of XML structure (including some data):

```
<tables>
   <table>
      <row>
            <col>True</col>
            <col>0</col>
            <col>Network A</col>
            <col>1</col>
      </row>
   </table>
</tables>
```

Methods

The following methods are avaiable:

- procedure _Entire_AllControllersToON; - *sets all controllers to ON mode, the same as* ***Commands/Entire System/All Controllers to ON*** *in PR Master .*

- procedure _Entire_AllControllersToOFF; - *sets all controllers to OFF mode, the same as* ***Commands/Entire System/All Controllers to OFF*** *in PR Master .*

- procedure _Entire_SetNormalMode; - *sets Door Normal Mode on selected controller, the same as* ***Commands/Entire System/Set Normal Mode*** *in PR Master*

- procedure _Entire_SetUnlockMode; - *sets Door Unlock Mode, the same as* ***Commands/Entire System/Set Unlock Mode*** *in PR Master*

- procedure _Entire_SetConditionalUnlockMode; - *sets Door Conditional Unlock Mode, same as* ***Commands/Entire System/Set Conditional Unlock Mode*** *in PR Master.*

- procedure _Entire_SetBarriedMode; - *sets Door Locked Mode, same as* ***Commands/Entire System/Set Locked Mode*** *in PR Master.*

- procedure _Command_CancelAllAlarms; - *clears all alarms which are active in access control system, same as* ***Commands/Cancel All Alarms*** *in PR Master.*

- procedure _Command_SetSystemClocks; - *sets Clocks of all equipment in access control system , same as* ***Commands/Set System Clock*** *in PR Master.*

- procedure _Controller_CancelControllerAlarm(idx: SYSUINT); - *clears alarm on selected controller, same as* ***Commands/Controller Commands/Cancel Controller Alarm*** *in PR Master. (Parameter idx – controller's index (first = 0) according to list received through property* _GET_ReaderNamesXML*).*

- procedure _Controller_OpenDoor(idx: SYSUINT); - *activate door lock on selected controller, same as* ***Commands/Controller Commands/Open Door*** *in PR Master. (Parameter idx – controller's index (first = 0) according to list received through property* _GET_ReaderNamesXML*).*

- procedure _Controller_SwitchToON(idx: SYSUINT); - *sets controller to ON mode, same as* ***Commands/Controller Commands/Switch to ON*** *in PR Master. (Parameter idx – controller's index (first = 0) according to list received through property* _GET_ReaderNamesXML*).*

- procedure _Controller_SwitchToOFF(idx: SYSUINT); - *sets cotrnoller to OFF mode, same* ***as Commands/Controller Commands/Switch to OFF in PR Master****. (Parameter idx – controller's index (first = 0) according to list received through property* _GET_ReaderNamesXML.*)*

- procedure _Controller_CheckDoorMode(idx: SYSUINT); - *shows actual controller's Door Mode, same as* ***Commands/Controller Commands/Check Door Mode*** *in PR Master. (Parameter idx – controller's index (first = 0) according to list received through property* _GET_ReaderNamesXML*).*

- procedure _Controller_SetNormalMode(idx: SYSUINT); - *sets controller to Normal Mode, same as* ***Commands/Controller Commands/Set Normal Mode*** *in PR Master (Parameter idx – controller's index (first = 0) according to list received through property* _GET_ReaderNamesXML*).*

- procedure _Controller_SetUnlockMode(idx: SYSUINT); - *sets controller to Unlock Mode, same as* ***Commands/Controller Commands/Set Unlock Mode*** *in PR Master. (Parameter idx – controller's index (first = 0) according to list received through property* _GET_ReaderNamesXML*).*

- procedure _Controller_SetConditionalUnlockMode(idx: SYSUINT); - *sets controller to Conditional Unlock Mode, same as* ***Commands/Controller Commands/Set Conditional Unlock Mode****. (Parameter idx – controller's index (first = 0) according to list received through property* _GET_ReaderNamesXML*).*

- procedure _Controller_SetBarriedMode(idx: SYSUINT); - *sets controller to Locked Mode, same as* ***Commands/Controller Commands/Set Locked Mode*** *in PR Master. (Parameter idx – controller's index (first = 0) according to list received through property* _GET_ReaderNamesXML.*)*

- procedure _Zone_SwitchZoneToON(idx: SYSUINT); - *sets all controllers in specified zone to ON Mode, same as* ***Commands/Zones Commands/Switch Zone to ON*** *in PR Master. (Parameter idx – controller's index (first = 0) according to list received through property* _GET_ZonesXML*).*

- procedure _Zone_SwitchZoneToOFF(idx: SYSUINT); - *sets all controllers in selected zone to OFF Mode, same as* ***Commands/Zones Commands/Switch Zone to OFF*** *in PR Master. (Parameter idx – controller's index (first = 0) according to list received through property* _GET_ZonesXML*).*

- procedure _Zone_SetNormalMode(idx: SYSUINT); - *sets all controllers in selected zone to Normal Mode, same as* ***Commands/Zones Commands/Set Normal Mode*** *in PR Master. (Parameter idx – controller's index (first = 0) according to list received through property* _GET_ZonesXML*).*

- procedure _Zone_SetUnlockMode(idx: SYSUINT); - *sets all controllers to Unlock Mode, same as* ***Commands/Zones Commands/Set Unlock Mode*** *in PR Master. (Parameter idx – controller's index (first = 0) according to list received through property* _GET_ZonesXML*).*

- procedure _Zone_SetConditionalUnlockMode(idx: SYSUINT); - *sets all controllers in selected zone to Unlock Mode, same as* ***Commands/Zones Commands/Set Conditional Unlock Mode*** *in PR Master. (Parameter idx –*

controller's index (first = 0) according to list received through property _GET_ZonesXML*).*

- procedure _Zone_SetBarriedMode(idx: SYSUINT); - *sets all controllers in selected zone to Locked Mode, same as Commands/Zones Commands/Set Locked Mode in PR Master. (Parameter idx – controller's index (first = 0) according to list received through property* _GET_ZonesXML*).*

- procedure _ClearEventList; - *clears "EVENTS" window, same as* ***View/Clear Event Window*** *in PR Master.*

- procedure _ClearAlarmList; - *clears "ALARM" window, same as* ***View/Clear Alarm Window*** *in PR Master.*

Events

- procedure _OnConnect(const info: WideString); - *this event occur when connection with PR Master is established. (example: „Connected to PRMaster. Server TCP IP=127.0.0.1 Port=80")*

- procedure _OnDisconnect; - *this event occur when connection with PR Master is discontinued.*

- procedure _OnMessage(const str: WideString); - *this events occur when message box appear on PR Master*

- procedure _OnErrorMessage(const str: WideString); - *this events occur when error message box appear on PR Master*

- procedure _OnMonitorEvent(const xml: WideString); - *this events occur when new event in EVENTS or ALARMS window appear.*

An example of XML structure (including some data):

```
<monEvent>
   <code>10</code>
   <par>3072</par>
   <str>;27-05-2003;13:09:30;Aux. input returned to normal
   cond.;PR301;;;Network A;Default;Internal door</str>
   <type>0</type>
</monEvent>
```

- procedure _OnUpdateSubsPanels(const xml: WideString); - *this event comes when PR Master is refreshing subsystem info (time/date/CPR status).*

An example of XML structure (including some data):

```
<subs>
```

```
      <sub nr="1" cl="-16777201">
         <dt>27-05-2003  13:07:43  Tuesday</dt>
         <ev>0</ev>
         <led>
               <al>8421504</al>
               <t>8421504</t>
               <c>8421504</c>
               <ba>8421504</ba>
               <ac>8421504</ac>
               <bu>8421504</bu>
         </led>
      </sub>
   </subs>
```