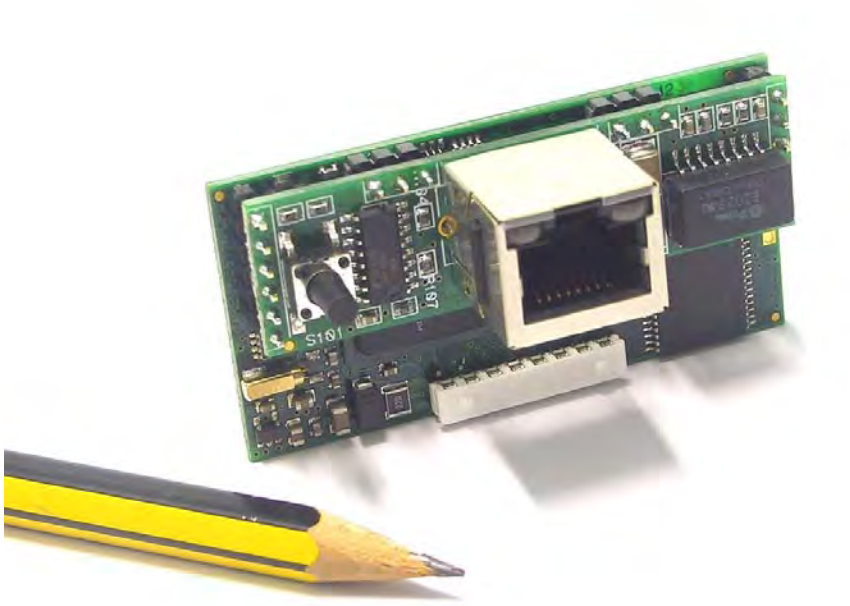
pCOWeb



ENG

User manual

LEGGI E CONSERVA QUESTE ISTRUZIONI
READ AND SAVE THESE INSTRUCTIONS

Technology & Evolution

WARNINGS

CAREL bases the development of its products on decades of experience in HVAC, on the continuous investments in technological innovations to products, procedures and strict quality processes with in-circuit and functional testing on 100% of its products, and on the most innovative production technology available on the market. CAREL and its subsidiaries nonetheless cannot guarantee that all the aspects of the product and the software included with the product respond to the requirements of the final application, despite the product being developed according to start-of-the-art techniques. The customer (manufacturer, developer or installer of the final equipment) accepts all liability and risk relating to the configuration of the product in order to reach the expected results in relation to the specific final installation and/or equipment. CAREL may, based on specific agreements, acts as a consultant for the positive commissioning of the final unit/application, however in no case does it accept liability for the correct operation of the final equipment/system.
The CAREL product is a state-of-the-art product, whose operation is specified in the technical documentation supplied with the product or can be downloaded, even prior to purchase, from the website www.carel.com.
Each CAREL product, in relation to its advanced level of technology, requires setup / configuration / programming / commissioning to be able to operate in the best possible way for the specific application. The failure to complete such operations, which are required/indicated in the user manual, may cause the final product to malfunction; CAREL accepts no liability in such cases.
Only qualified personnel may install or carry out technical service on the product.
The customer must only use the product in the manner described in the documentation relating to the product.
In addition to observing any further warnings described in this manual, the following warnings must be heeded for all CAREL products:
prevent the electronic circuits from getting wet. Rain, humidity and all types of liquids or condensate contain corrosive minerals that may damage the electronic circuits. In any case, the product should be used or stored in environments that comply with the temperature and humidity limits specified in the manual;
do not install the device in particularly hot environments. Too high temperatures may reduce the life of electronic devices, damage them and deform or melt the plastic parts. In any case, the product should be used or stored in environments that comply with the temperature and humidity limits specified in the manual;

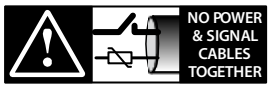
- do not attempt to open the device in any way other than described in the manual;
- do not drop, hit or shake the device, as the internal circuits and mechanisms may be irreparably damaged;
- do not use corrosive chemicals, solvents or aggressive detergents to clean the device;
- do not use the product for applications other than those specified in the technical manual.

All of the above suggestions likewise apply to the controllers, serial boards, programming keys or any other accessory in the CAREL product portfolio.

CAREL adopts a policy of continual development. Consequently, CAREL reserves the right to make changes and improvements to any product described in this document without prior warning.

The technical specifications shown in the manual may be changed without prior warning.

The liability of CAREL in relation to its products is specified in the CAREL general contract conditions, available on the website www.carel.com and/or by specific agreements with customers; specifically, to the extent where allowed by applicable legislation, in no case will CAREL, its employees or subsidiaries be liable for any lost earnings or sales, losses of data and information, costs of replacement goods or services, damage to things or people, downtime or any direct, indirect, incidental, actual, punitive, exemplary, special or consequential damage of any kind whatsoever, whether contractual, extra-contractual or due to negligence, or any other liabilities deriving from the installation, use or impossibility to use the product, even if CAREL or its subsidiaries are warned of the possibility of such damage.

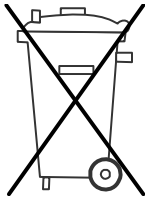


READ CAREFULLY IN THE TEXT!

WARNING:

separate as much as possible the probe and digital input signal cables from the cables carrying inductive loads and power cables to avoid possible electromagnetic disturbance.
Never run power cables (including the electrical panel wiring) and signal cables in the same conduits.

DISPOSAL



The product is made from metal parts and plastic parts.
In reference to European Union directive 2002/96/EC issued on 27 January 2003 and the related national legislation, please note that:

1. WEEE cannot be disposed of as municipal waste and such waste must be collected and disposed of separately;
2. the public or private waste collection systems defined by local legislation must be used. In addition, the equipment can be returned to the distributor at the end of its working life when buying new equipment.
3. the equipment may contain hazardous substances: the improper use or incorrect disposal of such may have negative effects on human health and on the environment;
4. the symbol (crossed-out wheeled bin) shown on the product or on the packaging and on the instruction sheet indicates that the equipment has been introduced onto the market after 13 August 2005 and that it must be disposed of separately;
5. in the event of illegal disposal of electrical and electronic waste, the penalties are specified by local waste disposal legislation.

CONTENTS

PREAMBLE

This manual has been designed with care to allow the detailed use of the product.
Please do not hesitate to contact CAREL Industries S.r.l with any information on imprecision, oversights or suggestions for easier understanding; your suggestions will help improve the quality of the following editions of this document.

Contact: pcoweb@carel.com.

1 PRESENTATION OF THE PRODUCT

pCOWeb – order code PCO1ØØØW*Ø - is an accessory for the ***pCO**** series products (excluding the ***pCOB**** series) and more in general for CAREL products fitted with a serial port and communication via the "CAREL supervisor" data protocol.

pCOWeb acts a "gateway", that is, a translator between the "CAREL supervisor" data protocol and the Ethernet network protocols commonly used to connect the computers in a building.

On the ***pCO**** series products it is installed on the "Serial board" port (also called "serial card 1" or "BMS"), from where it also receives the power required for operation.
The cover supplied protects ***pCOWeb***, especially when removing the network connector.

The ***pCOWeb*** firmware can also be updated by the user.
This manual refers to firmware version A1.4.2. – B1.2.1. To check the version loaded on the ***pCOWeb***, see chapter 8 on page 39 and Figure 9.d on page 42.

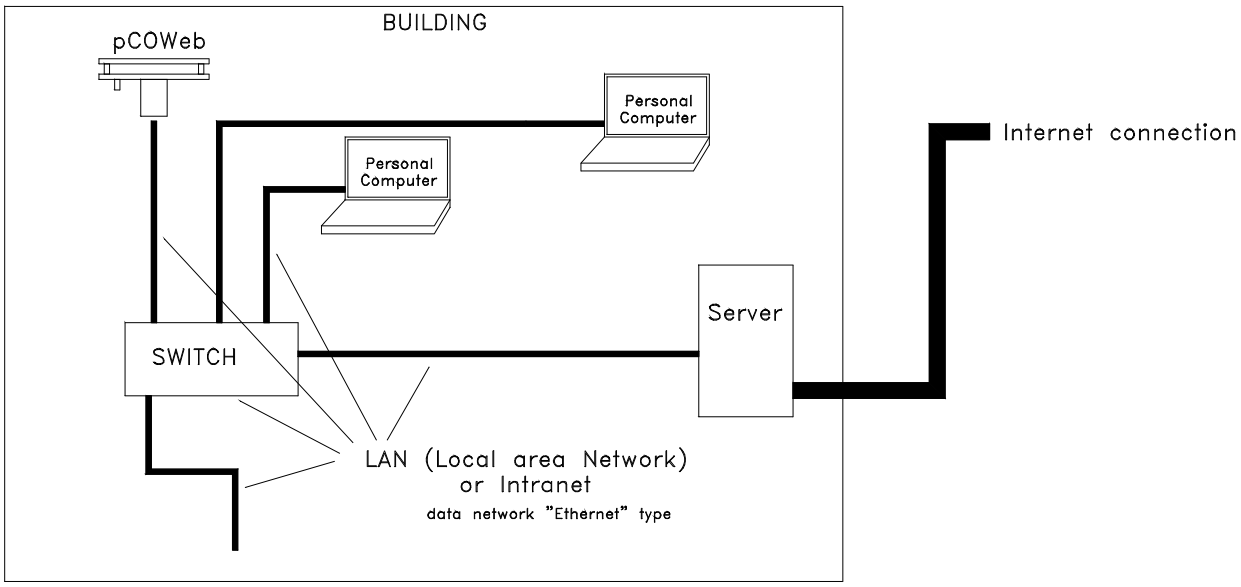


Figure 1.a - ***pCOWeb***: example of network connection

In the example shown in **Figure 1.a** above ***pCOWeb*** is connected to an internal network. If suitably configured, the personal computers in the internal network can communicate with ***pCOWeb*** using one of the various standard network languages included on ***pCOWeb***.
The server connects the external network or "Internet" to the internal network or "Intranet".
Usually in a network, the exchange of data between Internet-Intranet is only enabled for some devices and only for some types of communication. These decisions are defined by the network administrator.

This manual only covers the configurations of the ***pCOWeb*** and the more simple types of networks. For further information, see specific publications on the topic of "data networks".

1.1 FUNCTIONS AVAILABLE

pCOWeb connected to an Ethernet network provides the following functions:

- **WEB server**: used to check or change the operating status of the ***pCO*** controller using a remote computer running, for example, Internet Explorer™ or Mozilla Firefox; dynamic web pages developed by the user can be added; supports CGI and Ajax technologies; supports protection for accessing web pages;
- **Logger**: ***pCOWeb*** can record the values of some of the ***pCO*** controller variables in a file on its non-volatile memory; the file can then be downloaded to a PC using Internet Explorer™;
- **Graphs**: the trends over time of the data saved with the Logger function can be viewed on graphs;
- **E-mail**: ***pCOWeb*** can send e-mails when programmable events occur on the ***pCO*** controller (activation of alarms, exceeding of thresholds) or at set time intervals; a file in XML format can be attached containing the values of the variables;
- **FTP PUSH**: ***pCOWeb*** can send a file in XML format containing values of the variables to a suitably configured computer; the send operations can be programmed in the same way as for send the e-mail messages (upon event or at set times); the file is sent using the FTP protocol;

- **SNMPv1 & v2**: to access ***pCOWeb*** from a computer using supervision software based on the SNMP protocol. ***pCOWeb*** can send programmable enterprise TRAP or INFORM packets for alarm notifications;
- **MODBus over TCP/IP:** ***pCOWeb*** supplies information to a MODBus master requesting it over TCP/IP;
- **BACnet Ethernet ISO8802-2 over 8802-3**: to access the ***pCO*** controller using supervision software based on the BACnet Ethernet protocol;
- **BACnet/IP (Addendum A/Annex J)**: for access using supervision software based on the BACnet/IP protocol;
- **FTP server**: used to copy data files or web pages from/to ***pCOWeb*** in a simple manner, using programs based on dragging icons from one window to another;
- **DHCP**: used to connect ***pCOWeb*** to a local network using the method of automatic addresses assignment by a central server, rather than statically setting the addresses on the individual devices; DHCP is active by default;
- **Plugins**: used to add additional applications developed by CAREL or by the user in script or compiled format;
- **Firmware update**: the ***pCOWeb*** firmware can be updated from a computer;
- ***pCO*** **Application update**: ***pCOWeb*** is able to update the ***pCO*** application (not the BIOS) of every type of ***pCO**** controller (supernode included), except for ***pCO2****.

1.2 USER INTERFACE: BUTTON and LEDS

pCOWeb features (Figure 1.b) a button (PUSHBUTTON) and two indicator lights (STATUS LED and ETHERNET LED).

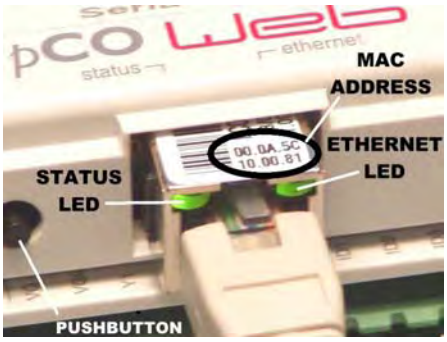


Figure 1.b - MAC address and indicator LEDs

Functions of the button

- When starting up the ***pCOWeb***, this is used to select, for network communication, whether to use the factory parameters or the user parameters (see 3.1.2 on page 12 for the procedure);
- In normal operation, reboots ***pCOWeb*** without needing to disconnect the power supply (see 9.9.1 on page 53 for the procedure).

Meaning of the LEDs

- Status LED: displays information on the communication status between ***pCOWeb*** and the ***pCO***, and must normally be green and flash around 3 times a second; in special circumstances it displays the operation of service activities, such as the restart of the internal program on the ***pCOWeb***, the remote updating of the program, or others. See the table below.

Table 1.a - Status LED signals

Status LED	Meaning	Notes
Green flashing (3 times/sec)	Regular ***pCO-pCOWeb*** communication	When running demanding tasks (sending a large number of notifications), this may be green steady for a few seconds
Red flashing slowly (once every 2 seconds)	***pCO-pCOWeb*** communication not established	Check the settings in paragraph 9.5 on page 44
Single red flash and then flashing green	Single ***pCO-pCOWeb*** communication error, one failed response from the ***pCO*** or attempt to write a variable with an index higher than 207	After 5 failed responses, the Status LED starts flashing red until communication resumes
Red steady	Rescue mode	See 9.10.3 on page 55
[Off, then] green-red repeated in rapid succession, then green steady for 1-2 minutes	***pCOWeb*** reboot phase	See 9.9 on page 53
Green steady for 1-2 minutes	***pCOWeb*** reboot phase	Wait for the conclusion of the reboot
Red – Off slow (1 second – 1 second) repeated 3 times	Recognition of button pressed during the reboot for selecting the factory parameters (rather than the User parameters)	Release the button to confirm, see 9.9.1 on page 53
Red – Off fast (3 times a second) repeated 3 times	During the reboot, confirms the selection of the factory parameters by pressing the button	See 9.9.1 on page 53
Green-red repeated alternating (once a second)	During the firmware update, writing block B to non-volatile memory	Do not interrupt the power supply, see 9.10.1 on page 54
Red steady followed by green-red a number of times	During the update firmware, writing block A to non-volatile memory	Do not interrupt the power supply, see 9.10.1 on page 54

- Ethernet LED: displays the status of the physical network connection (Ethernet connection signals), regardless of whether the network parameters are correct; usually this must be green and flash when the data transits.

Table 1.b - Ethernet LED signals

Ethernet LED	Meaning	Notes
Green steady	Correct Ethernet data connection signals	-
Green flashing	Correct Ethernet data exchange	-
Red	No Ethernet signal detected	See 3.1.3 on page 13

2 INSTALLATION ON THE *pCO* CONTROLLER

2.1 ASSEMBLY

Equipment required:

- a 3 mm flat-head screwdriver;
- a map of the installation (only when installing two or more ***pCOWeb*** devices);
- a pair of scissors.

Figure 2.a – ***pCOWeb*** and the accessories supplied

Figure 2.b – Removing the cover from the ***pCO*** controller

IMPORTANT: to avoid damage, before inserting ***pCOWeb,*** disconnect power to the ***pCO*** controller.

Figure 2.c – Inserting ***pCOWeb*** in the ***pCO*** controller

Figure 2.d – Securing ***pCOWeb*** with the cover

2.2 LABELS SUPPLIED

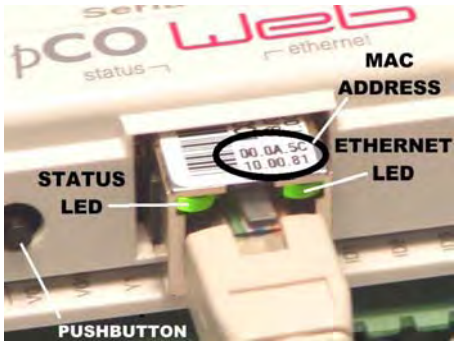


Figure 2.e - MAC address and indicator LEDs

Each ***pCOWeb*** is uniquely distinguished by its own MAC address. The network administrator may at times need to check the MAC address of each ***pCOWeb*** installed.
The MAC address is shown on the label applied to the connector, and on the two labels included in the packaging; in the example shown in Figure 2.e, the MAC address is: 00.0A.5C.10.00.81
Once installed, ***pCOWeb*** may however no longer be accessible.
Therefore, during installation **use the scissors to separate the two labels supplied and apply one in an easily accessible position near the *pCO* controller on the outside of the electrical panel**.

If more than one ***pCOWeb*** device is installed, a map of the installation should be created, applying the second label provided in the packaging onto the map for each ***pCOWeb***, corresponding to its physical position; in this way, the network administrator can be provided with precise documents on where the ***pCOWeb*** devices and corresponding MAC addresses are located.

3 STARTING FOR THE FIRST TIME – ACCESSING *pCOWeb* FROM A COMPUTER

This chapter guides the user, even non-experts, in establishing a connection between the ***pCOWeb*** and a personal computer.
The first part of the chapter describes the ***pCOWeb*** – cable – personal computer (PC) connection, without involving an Ethernet network.
Nonetheless, a ***pCOWeb*** can be accessed from a PC even if these are not connected together directly, but via a network (paragraph 3.2 on page 14); the latter procedure is more complex, as it also requires the settings of the network devices that are normally managed by the administrator. Consequently, it is recommended to first try the direct connection. In any case, once having accessed the device, the basic configurations can be completed (chapter 8 on page 39) and ***pCOWeb*** prepared for connection to the network.

3.1 CONNECTING *pCOWeb* DIRECTLY TO A PC

This connection is used to access ***pCOWeb*** from a computer connected by cable.
Normally this type of connection is used to test operation on the bench or to initially configure ***pCOWeb*** for an installation that does not use "DHCP" automatic address setting (see 9.4 on page 43).
Paragraph 3.2 on page 14, on the other hand, describes the network connection procedure.

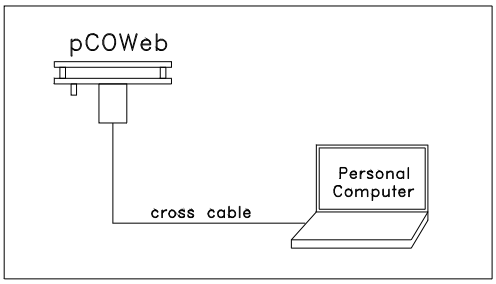


Figure 3.a - Direct PC-***pCOWeb*** connection

Equipment required:

- A computer running, for example, Internet Explorer™ and fitted with an Ethernet network interface; if ***pCOWeb*** is already installed in the final destination and cannot be removed, a portable computer is handier.
- A crossover network cable (recent computers do not require the crossover cable; a normal cable can be tried in any case: if the crossover cable is required, this will be established when connecting).

Starting situation:

- ***pCOWeb*** installed on the ***pCO*** controller (see paragraph 2.1 on page 9);
- ***pCO*** controller NOT powered.
  IMPORTANT: if the ***pCO*** controller is connected in the final installation, before powering up contact the installation manager.

The steps to be completed are as follows:

1. Configuration of the PC for direct connection to ***pCOWeb***.
2. Connection and start-up of ***pCOWeb*** to check correct installation.
3. Activation of the factory network settings (button).
4. Access to the ***pCOWeb*** from the PC.

3.1.1 PC configuration

INFORMATION

The PC can communicate with ***pCOWeb*** if the settings on both devices are correctly aligned.
As the ***pCOWeb*** default settings can only be changed once the connection has been established with the PC, when first accessing the device the personal computer will need to be configured to adapt it to the ***pCOWeb*** default settings.

A – **disconnect the personal computer** from the data network (if connected), and connect it directly to the ***pCOWeb*** using the cable (crossover).

B – **IP ADDRESS AND SUBNET MASK**

INFORMATION

The personal computer must be set not to use the DHCP, but rather the following IP address: 172.16.0.2. The Subnet mask field also needs to be set; the Gateway is not required.
For further information on the meaning of these parameters, see APPENDIX A on page 61 and APPENDIX B on page 62.

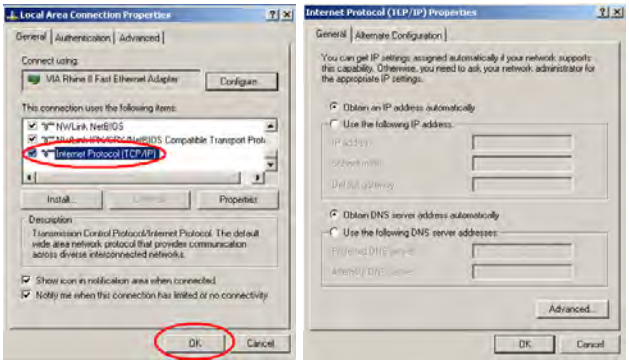


Figure 3.b - Network settings

1. On the Windows PC click the mouse on the "Start" button at the bottom left
2. Choose "Settings"
3. Click "Control panel"
4. Double click "Network and dial-up connections"
5. Double click "Local area connection"
6. Click "Properties": the window shown in Figure 3.b (left) is displayed
7. Double click "Internet Protocol (TCP/IP)": the window shown in Figure 3.b (right) is displayed.

Note down all the settings shown in the new window: this will be useful for returning to the original settings when the procedure is terminated, so that the PC can again communicate with the data network it was previously connected to. In the example shown in the figure, the PC was using an IP address obtained automatically from the DHCP server in the data network.
Then:

8. Click “Use the following IP address” (Figure 3.c);
9. Set the following parameters:

   IP address = 172.16.0.2
   Subnet mask = 255.255.0.0

10. Click the OK button to close all the windows.

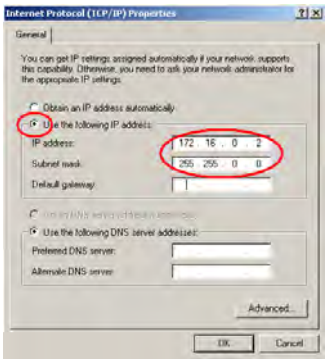


Figure 3.c - Assigning an IP address to the PC

C – PROXY

INFORMATION

The following procedure tells the personal computer to not use the network device called the “proxy” for communication: in fact, the PC is not connected to the network and if the “proxy” is not disabled communication would not be possible.

1. Open the Windows “Control panel”.
2. Double click “Internet options”; the window shown in Figure 3.d above (left) will be displayed.
3. Click “Connections”. Another window (Figure 3.d - right) will be displayed.
4. Click “LAN settings…”

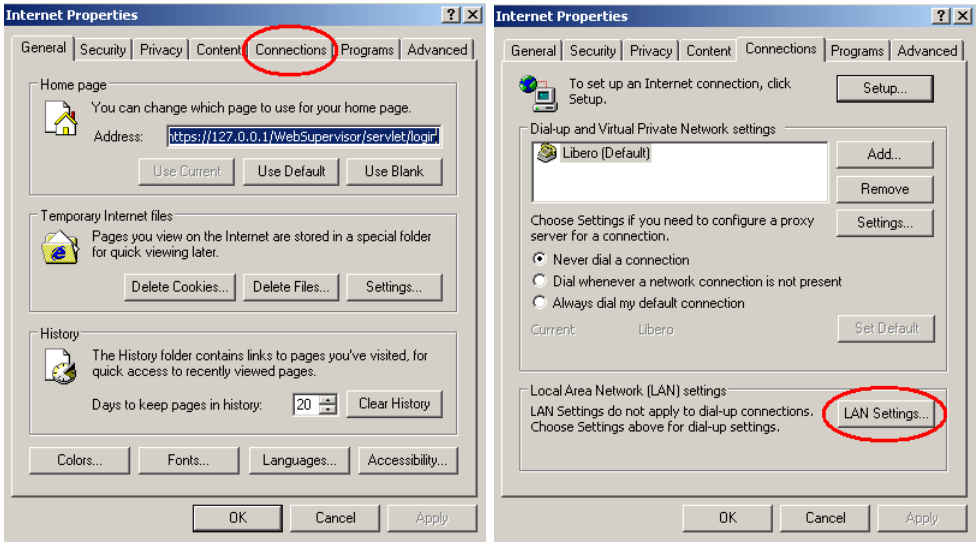


Figure 3.d – Setting the proxy – steps 2, 3, 4

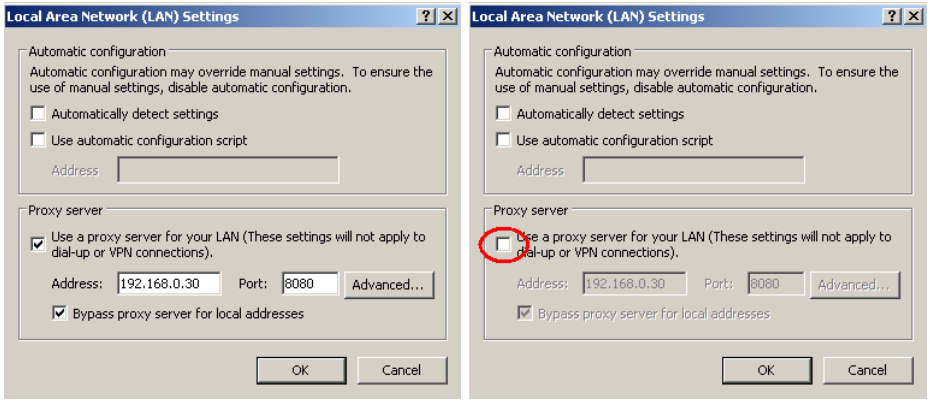


Figure 3.e – Setting the proxy – steps 5, 6, 7

5. Note down the settings.
6. Disable the proxy server.
7. Close the windows using the OK button.

3.1.2 Connection, starting *pCOWeb* and activating the default network settings (Button)

Connection

1. Connect ***pCOWeb*** to the Ethernet connector on the PC using a crossover cable (Figure 3.f below).

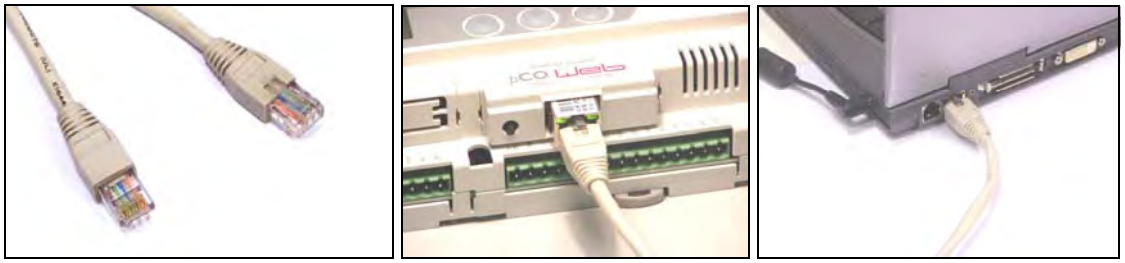


Figure 3.f – ***pCOWeb*** - PC connection

Starting

2. Switch on the ***pCO*** controller.
3. Check that both the indicator LEDs on the ***pCOWeb*** connector come on within a few seconds (Figure 2.e on page 9). If this does not occur, see 3.1.3 on page 13.

Activating the factory network settings (button)

INFORMATION

- the activation of the factory settings or the user settings can only be selected when starting the ***pCOWeb***.
- ***pCOWeb*** will reboot whenever it is restarted.
- ***pCOWeb*** can be restarted without disconnecting the power supply: see 9.9 on page 53.

4. Immediately after reboot, as soon as the Status LED remains on steady GREEN, to activate the factory settings rather than the user settings, hold the button;
5. after around 20 seconds the Status LED, due to the button being pressed, will turn RED and flash slowly 3 times; the button must be released before then end of the 3 flashes;
6. once the red flashes have terminated, the Status LED will turn GREEN and, if the procedure has been performed correctly, immediately after the Status LED will confirm the pressing and release of the button by flashing quickly 3 times RED, and then will come on steady GREEN again for around one minute (completion of the start-up phase);
7. once the start-up phase has been completed, the Status LED will start flashing: ***pCOWeb*** will now start operating; Table 1.a and Table 1.b on page 8 show the meanings of the visual indications represented by the two LEDs.

In this mode ***pCOWeb*** will not use the values of the "User" parameters for communication, but rather the following values (see 9.3 on page 43: View factory bootswitch parameters):

IP address: 172.16.0.1
Subnet mask: 255.255.0.0

NOTE 1 **These values remain active until *pCOWeb* IS RESTARTED (see Figure 3.g below). When next rebooted, if the button is NOT pressed, *pCOWeb* will return to the "User" configuration (see chapter 8 on page 39).**

NOTE 2 **These values are part of the "Bootswitch parameters" and, unlike the user parameters, cannot be modified. The Bootswitch parameters are never copied over the user parameters.**

NOTE 3 **By default, the *pCOWeb* "User" values activate DHCP mode for network communication.**

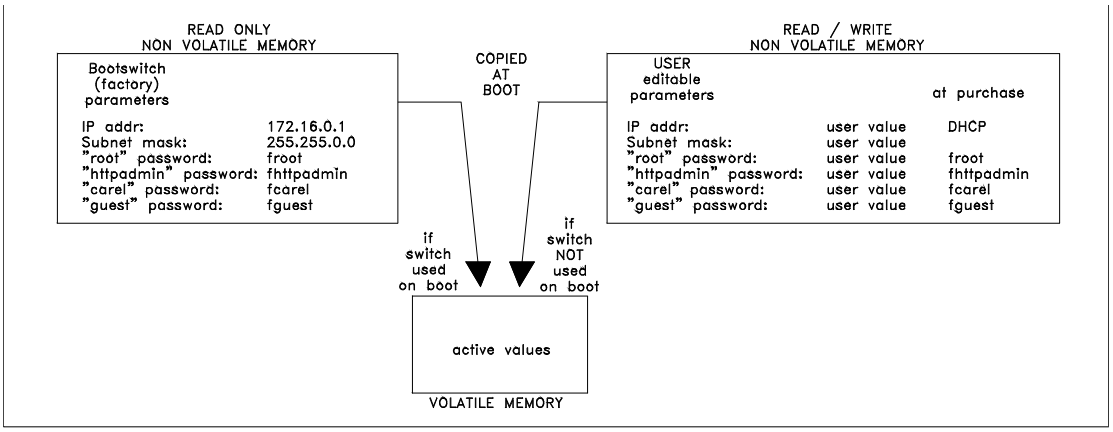


Figure 3.g - Operation of the "Bootswitch parameters" and of the user parameters.

3.1.3 LED signals when starting and during normal operation

Immediately after reboot, the Status LED and Ethernet LED must come on as described below. If remain both off, check:

- the correct installation of ***pCOWeb*** on the ***pCO*** controller;
- the connection of the power supply.

The Status LED (***pCOWeb*** status and ***pCOWeb***-***pCO*** communication, see Table 1.a on page 8) behaves as follows immediately after reboot:

1. Remains off for 2 seconds;
2. then flashes quickly green/red for a further 2 seconds;
3. then comes on GREEN steady for around one and a half minutes (unless the button is pressed, see 3.1.2 on page 12);
4. then starts flashing GREEN or RED:
   - GREEN flashing: ***pCOWeb*** has completed the start-up phase and is communicating correctly with the ***pCO***;
   - RED flashing: ***pCOWeb*** has completed the start-up phase but is not communicating with the ***pCO***: this may be due to incorrect settings of the ***pCOWeb*** or some of the parameters on the ***pCO***; the default parameters ensure correct communication, therefore this situation will only occur if the parameters have been changed. In this case, after having established the PC-***pCOWeb*** connection, check the communication settings (see 9.5 on page 44);
   - RED without flashing: Rescue Mode (see 9.10.3 on page 55).
   - GREEN without flashing for at least 3 minutes: fault; contact CAREL Industries S.r.l. service.

The Ethernet LED (see Table 1.b on page 8) is GREEN in normal conditions. If it remains red, the causes may be:

- the PC is off;
- or the connector has not been inserted on the ***pCOWeb***;
- or the connector has not been inserted on the PC;
- or: the cable is not working, or the PC requires a crossover cable and this has not been used.

3.1.4 Accessing *pCOWeb* from a PC

On the PC open Internet Explorer; in the address field enter the following number, including the dots:

172.16.0.1

then press ENTER.

Figure 3.h - Opening the index.html page

IMPORTANT Before pressing ENTER wait for ***pCOWeb*** to complete the start-up phase (check that the Status LED is flashing normally), otherwise the required web pages will not be sent to the PC.

The ***pCOWeb*** main page "index.html" will be displayed (see Figure 3.i below).

IMPORTANT This page can be customised by the user, and by default appears as shown in Figure 3.i below. If it has been modified or deleted, the custom page or nothing will be displayed; in these cases, remember that a copy of the factory page is available in read-only memory.

To recall the factory page (even if index.html has been deleted or modified), type:

172.16.0.1/defindex.html

(to save this as index.html, access the User memory via FTP, see 4.2 on page 17, and copy the file /usr/local/root/defindex.html to the /usr/local/root/flash/http/ directory, renaming it index.html; any customised index.html page will be overwritten).

Figure 3.i – The default "index.html" page

This page provides access to the page for configuring the functions of the ***pCOWeb***. The network communication parameters should be configured immediately.
See chapter 8 on page 39.

NOTE 1 Remember that if ***pCOWeb*** is now rebooted (for example, switching the ***pCO*** controller off and on again) without pressing the button again as described in 3.1.2 on page 12, ***pCOWeb*** will use the user-set IP address; if this is different than the factory value, it may be impossible to connect to ***pCOWeb*** (this situation is described in paragraph 3.2 on page 14).

NOTE 2 At the end of procedure, restore the previous configuration on the PC so as to be able to access the network again.

3.2 ESTABLISHING A CONNECTION BETWEEN THE PC AND *pCOWeb* VIA A NETWORK

Starting situation:

- ***pCOWeb*** installed on the ***pCO*** controller (see chapter 2 on page 9).
  NOTE: if the controller is connected to the final installation and is not already on, before switching it on contact the installation manager.
- ***pCOWeb*** already connected to the Ethernet network; in this case, a standard cable (NOT a crossover cable) is used to connect ***pCOWeb*** to the network node.
- ***pCOWeb*** network communication parameters already correctly configured.
- Knowledge of the IP address of the ***pCOWeb*** being connected to.
- PC already connected and configured for the data network.

In the following example, the IP address of ***pCOWeb*** is assumed to be 10.0.3.114.

1. On the PC, open Internet Explorer and type the address of the ***pCOWeb*** (example 10.0.3.114), then press ENTER.
   IMPORTANT the ***pCOWeb*** must have completed the start-up phase (check that the Status LED is flashing normally); otherwise the required web pages will not be sent to the PC.
2. If the network administrator has already correctly fitted the network devices, ***pCOWeb*** will be accessible from the PC and the ***pCOWeb*** main page (index.html) will be shown. This page can be customised by the user; by default the page appears as shown in Figure 3.k below.

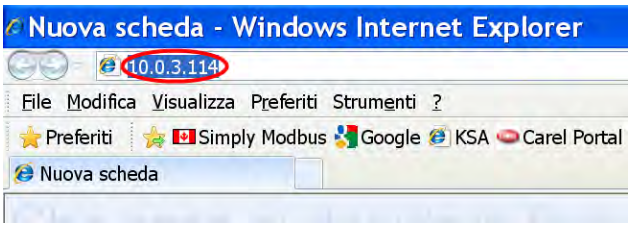


Figure 3.j - Opening the index page

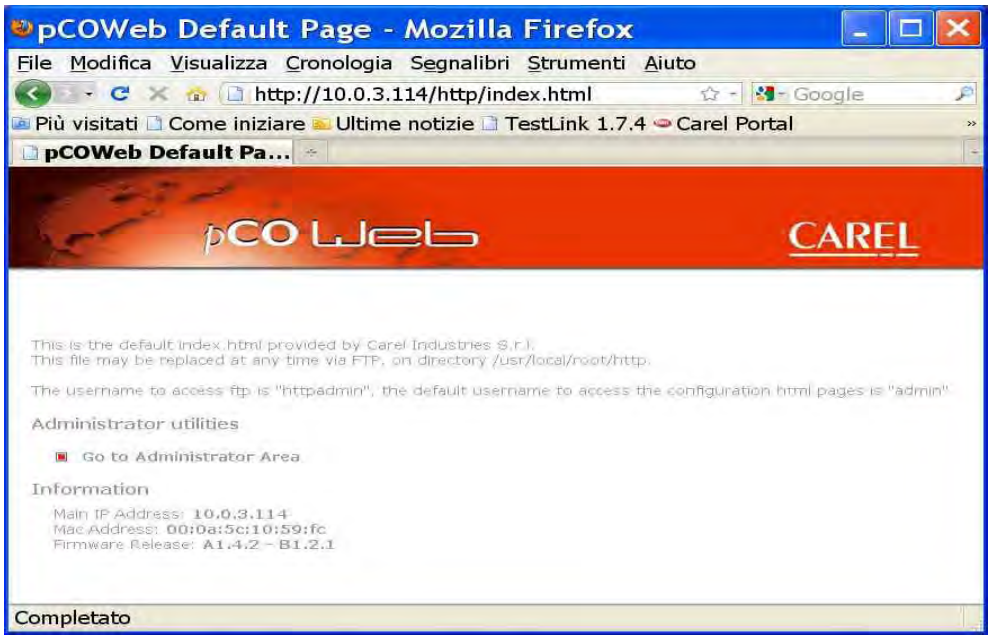


Figure 3.k – The default "index.html" page

If the page is not displayed:

- check that the computer has effective access to the data network (for example, try using Internet Explorer to access other sites with assured accessibility);
- check the indicator LEDs on the ***pCOWeb***, with reference to paragraph 3.1.3 on page 13 (the role of the PC in this case is performed by the switch or the hub);
- consider that the PC can access ***pCOWeb*** only in one of these conditions:
  - the network server features a proxy: in this case, the PCs connected to this type of network have already been set to use the proxy; the network administrator needs to modify the settings of the proxy on the server to make ***pCOWeb*** accessible from the PC;
  - or the network does not feature a proxy or in any case this is not intended to be used: on the PC disable the use of the proxy; if the IP addresses of the PC and ***pCOWeb*** are incompatible for direct communication (see APPENDIX B on page 62) the administrator must set the "gateway" network device for communication between devices with incompatible IP addresses;
  - or the IP addresses of the PC and ***pCOWeb*** are already compatible for direct communication (see APPENDIX B on page 629): in this case it is normally sufficient to disable the proxy on the PC (see point C in paragraph 3.1.1 on page 10); remember that disabling the proxy usually prevents access from the PC to other Internet sites; remember to enable it again when needed.

    To avoid losing access to other sites, as an alternative the proxy can be disabled only for one or a few IP addresses:
    see point C in paragraph 3.1.1 on page 10, however with the variants shown in Figure 3.l on page 15 (example for ***pCOWeb*** with IP address 10.0.3.114).

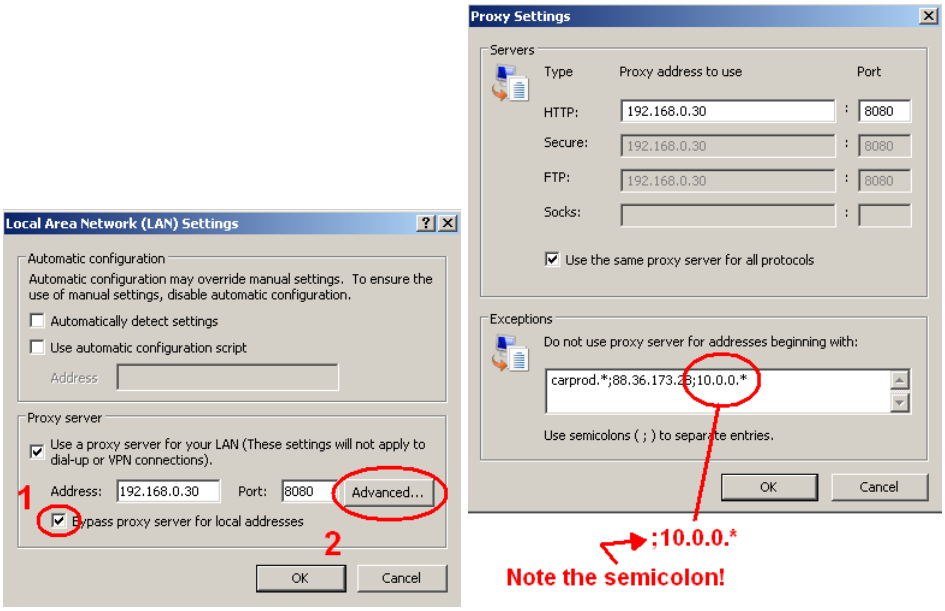


Figure 3.I - Disabling the proxy for some IP addresses

To ensure compatibility for direct communication, as an alternative to the procedures described above, the PC can be configured so that it responds not only to the IP address already set, but also to a second IP address that is compatible for direct communication with ***pCOWeb***.

IMPORTANT In this case, request the support of the network administrator, as any IP address that is assigned to the PC must be previously authorised.

4 FUNCTIONS

This chapter provides a guide to the use of the functions on the ***pCOWeb***.
When parameters need to be set, references are made to chapter 9 on page 39.

4.1 WEB SERVER: CUSTOM PAGES

The "WEB server" function of the ***pCOWeb*** allows a PC running Internet Explorer to display the web pages saved on the ***pCOWeb***.

The web pages can be divided into:
- configuration pages (chapter 9 on page 39), which reside in read-only memory;
- custom pages that can be added by the user, which reside in USER MEMORY space (non-volatile, read / write).

In the custom pages, as well as the typical commands of the HTML standard, other commands ("tags") can be added as per the CAREL standard; this allows read or write access to the supervisor variables of the ***pCO*** controller.

ACCESS RESTRICTIONS
Access restrictions can be defined for all or some custom pages. Whenever a protected page with access restriction is called, ***pCOWeb*** displays a login dialogue box requiring the Username / Password for the specific restriction (see 9.7.1 on page 49).

CGI SCRIPT
CGI scripts can be developed in bash language or compiled languages. These must have the .cgi extension and must reside in the usr-cgi directory (usr/local/root/flash/usr-cgi), otherwise they will not work.
Whenever one or more CGI scripts is modified, click the "Adjust HTML pages attributes" link (see 9.3 on page 43).
For a guide to CGI script for the ***pCOWeb*** see the documents available at http://ksa.carel.com.

4.1.1 Creating HTML pages

To create pages for the ***pCOWeb***, a PC and knowledge of HTML are required. The area dedicated to ***pCOWeb*** on KSA, http://ksa.carel.com, features simple web pages for the standard CAREL applications, which can be used as the starting point for creating custom web pages.
The description of HTML is outside of the scope of this document. Numerous guides are available on the web, for example:
- http://www.w3schools.com/
- http://www.w3.org/MarkUp/
- http://www.htmlgoodies.com/
- http://www.htmlhelp.com/

Also refer to the documentation available at http://ksa.carel.com.

APPENDIX G on page 71 lists the CAREL tags and describes an example of the default demo page resident on the ***pCOWeb***. Other pages are shown in APPENDIX I on page 84 ("Plugins").

To create a simple HTML page, the Notepad application can be used on the PC, typing the following lines and saving the file as "example.html":

```
<html>
<!--tagparser="/pcotagfilt"-->
<h1>pCOWeb Demo Page</h1>
<br>
MAC address: <%macaddress%>
</html>
```

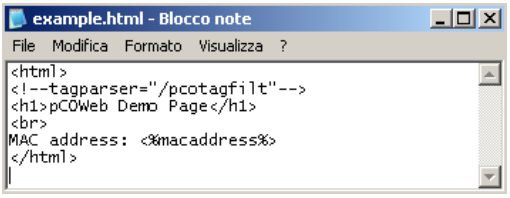


Figure 4.a - Example of a simple HTML page

(the `<%macaddress%>` string is a CAREL tag that returns the MAC address of the ***pCOWeb***).

To load the page created to the user memory, proceed as follows:
1. connect the PC to the ***pCOWeb***, making sure that the PC can access to the HTML pages on the ***pCOWeb*** (see chapter 3 on page 10);
2. access the user memory via FTP (see paragraph 4.2 on page 17);
3. transfer the file of the page created to the area (/usr/local/root/flash/http/) reserved for custom HTML pages.
   NOTE The ***pCOWeb*** web server considers:

   /usr/local/root/

   as the root directory; each page must be located inside this directory (or subdirectory).
4. Display the HTML page loaded by typing the location and name in the Internet Explorer address field:

   http://10.0.0.145/example.html

   then press ENTER.

5. If a directory called "newdir" was created and the page was loaded into this directory, the address would be:

http://10.0.0.145/newdir/example.html

IMPORTANT Browsers such as Internet Explorer, Mozilla Firefox and others create a copy of the pages visited in their cache, and when the page is next called the contents may be loaded from the cache, rather than requesting an update from ***pCOWeb***. This means that at times the desired result is not displayed, above all with dynamic pages such as those used on ***pCOWeb***.
The browsers, nonetheless, feature of a specific command to force them to update the pages from the web server (for example Ctrl+F5 requests the page from the server, F5 could load from the cache).
Ctrl+F5 should be used whenever there is doubt that the result may haven been loaded from the cache on the PC.

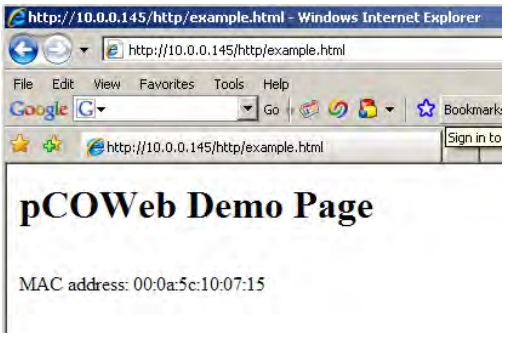


Figure 4.b – How the HTML page created is displayed

4.2 ACCESSING THE USER MEMORY VIA FTP

pCOWeb features an FTP server that is used to access the user memory and load or retrieve files, for example custom HTML pages, configuration files, log files.
To use this function, the PC should be installed with an FTP client; for example SmartFTP™ (http://www.smartftp.com) or FileZilla (http://filezilla-project.org/), used as the example in following procedure (for other software the procedure is similar).

PROCEDURE

1. Download, install and run SmartFTP™ on the PC.
2. Create a new "Remote Browser" and enter the data as shown in the Figure 4.c below.
   NOTE The IP address should be replaced with the address of the ***pCOWeb***; the default Username and Password are: httpadmin / fhttpadmin; paragraph 9.7.2 on page 50 describes how to change this information, and paragraph 9.3 on page 43 shows how to read the current information.
   The following examples assume that the current data being used are httpadmin / fhttpadmin and the IP address is 10.0.0.145.

   Once having selected the OK button, the contents of the user memory will be displayed (path usr/local/root/flash/http/); if this is not the case:
3. check the suggestions shown in chapter 3 on page 10: if the PC cannot access at least one HTML page on the ***pCOWeb***, it will not be able to access via FTP;
4. check that the PC is not running a firewall to block unwanted connections; for example, in Windows XP "Windows Firewall" is normally active, and in the default configuration this blocks FTP communication; to modify the settings, open the "Control panel", select "Windows Firewall" and disable it (Figure 4.d).

   IMPORTANT Some FTP clients do not use the request from ***pCOWeb*** to open the user memory for "httpadmin" (/usr/local/root/flash/http/), but rather show the contents of the root position; the user needs to manually go to /usr/local/root/flash/http/.

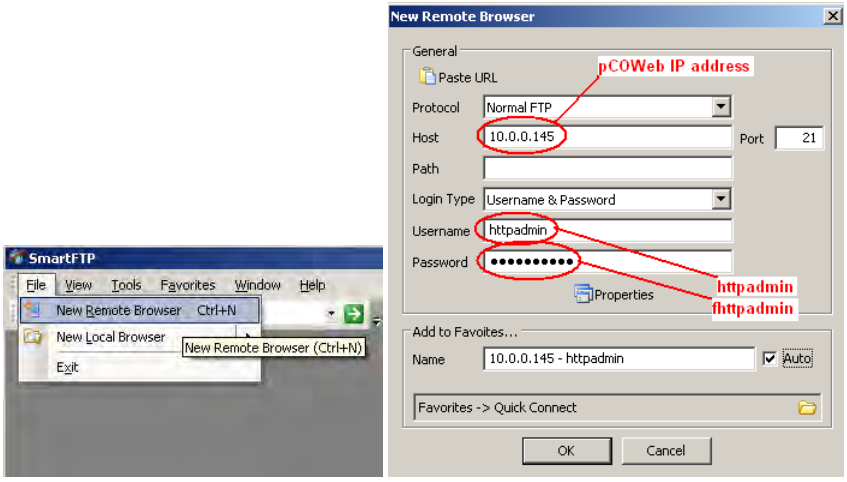


Figure 4.c – SmartFTP™: creating a new "Remote Browser"

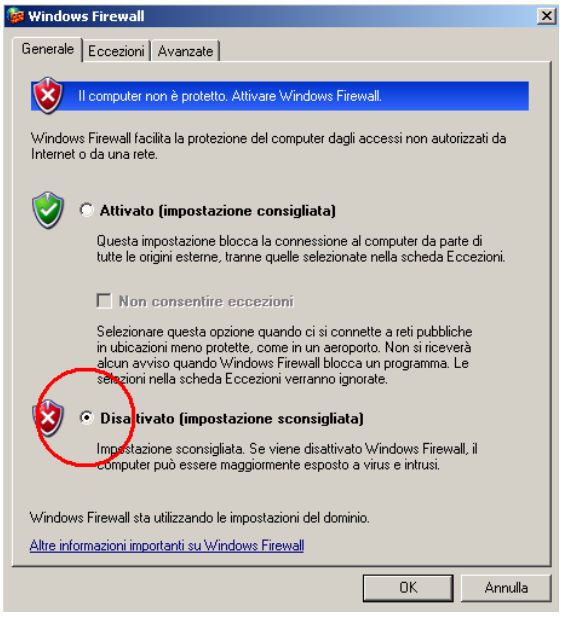


Figure 4.d – Disabling Windows Firewall

5. Create a new "Local Browser" (Figure 4.e);
a new window will be opened showing the contents of the PC (Figure 4.f below).
Then simply drag the directory or files from one window to the other, or alternatively delete the files.

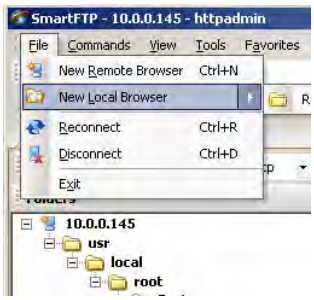


Figure 4.e – SmartFTP: creating a new Local Browser

IMPORTANT The ***pCOWeb*** web server considers:

/usr/local/root/

as the root directory; each page must be located inside this directory (or subdirectory).

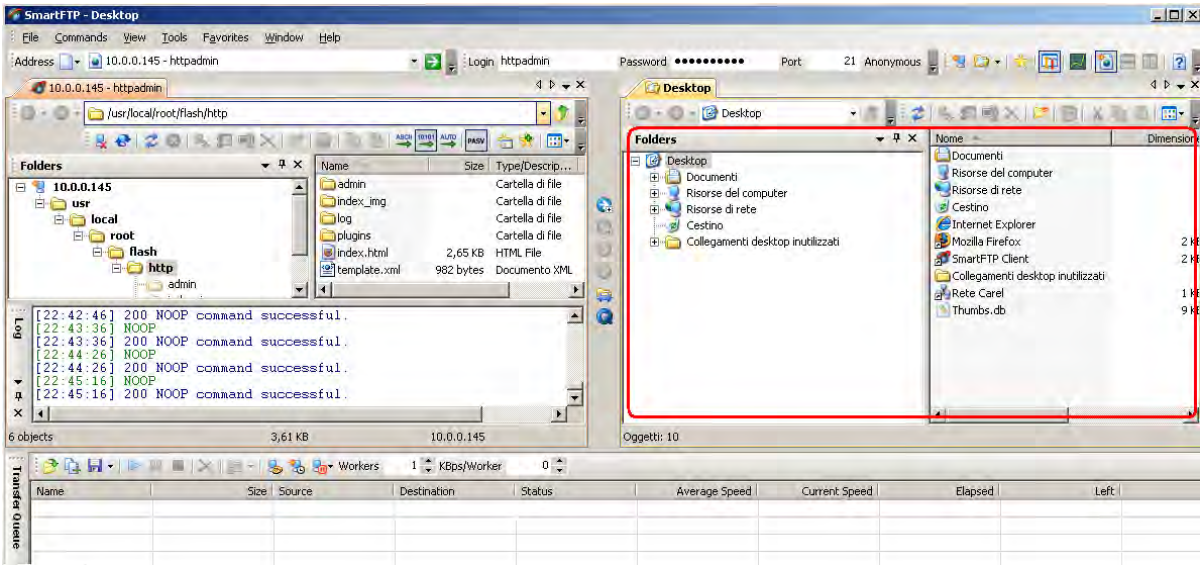


Figure 4.f – SmartFTP with the "Remote Browser" (left) and "Local Browser" (right) windows

The path for entering the web pages and the customised directory is:

/usr/local/root/flash/http/

Figure 4.g describes the memory on the ***pCOWeb*** and highlights the volatile and non-volatile (read/write or read-only) memory areas.

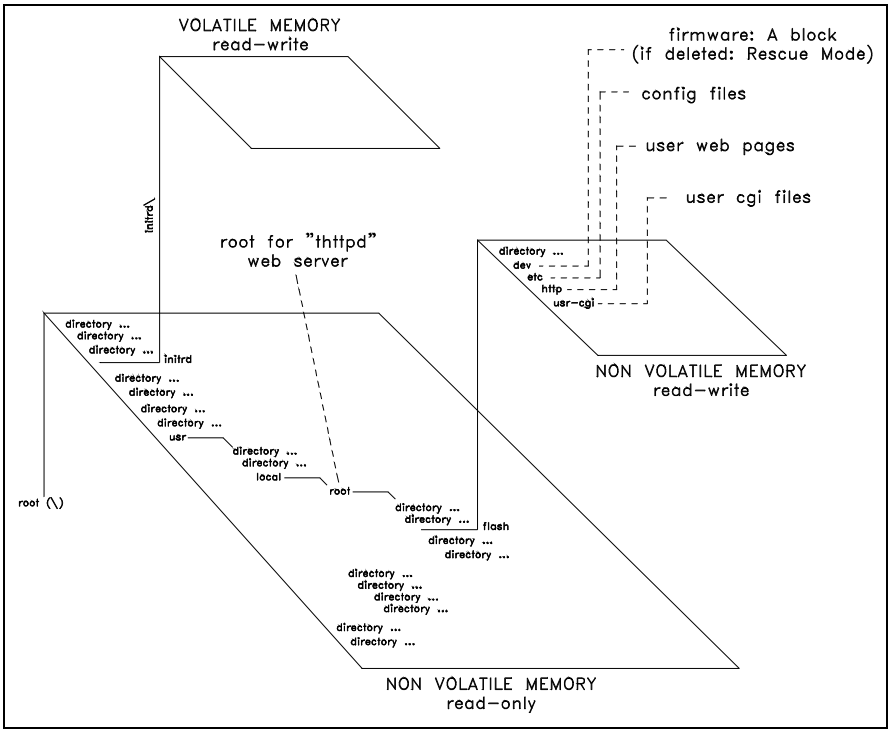


Figure 4.g –***pCOWeb*** memory: volatile and non-volatile memory

6. Alternatively, to use **Internet Explorer** as the FTP client, type:

```
ftp://httpadmin:fhttpadmin@10.0.0.145
```

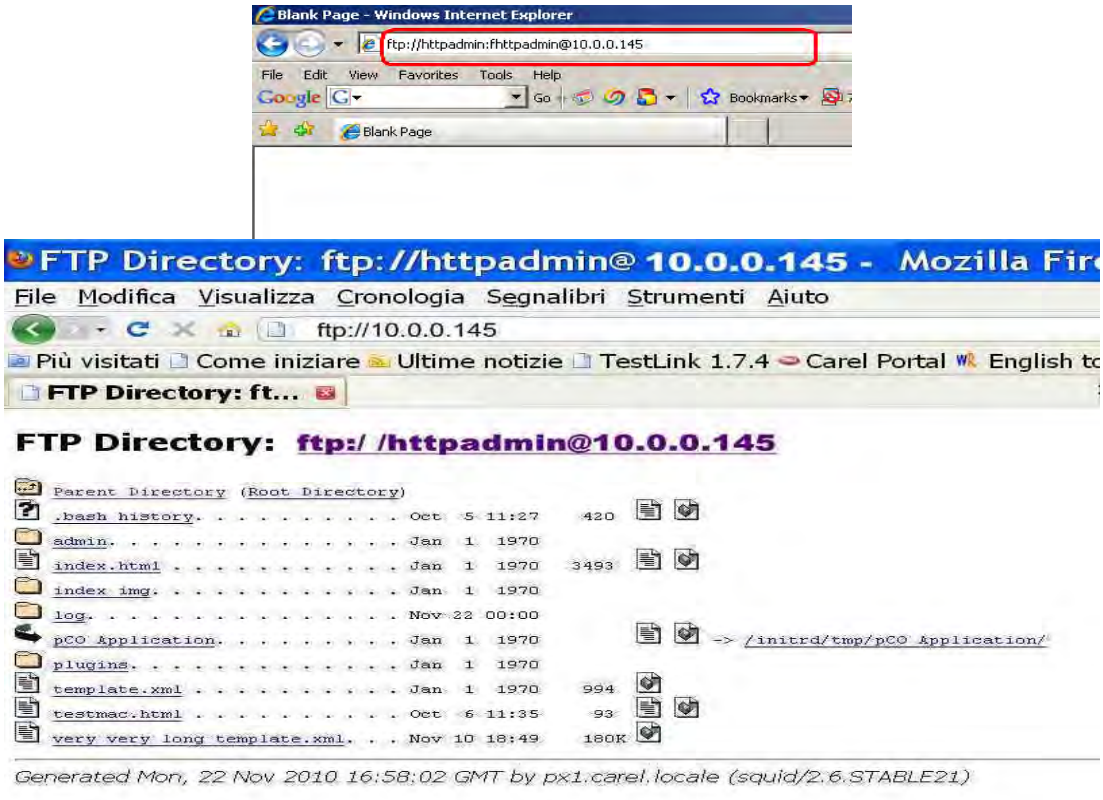


Figure 4.h – Using Internet Explorer as the FTP client

This solution is not recommended, however, as it is not suitable for copying files from the PC to ***pCOWeb***; in addition, previous versions of Internet Explorer had problems in navigating the memory space outside of the areas used for the HTML pages (configuration files …).

7. Alternatively, **Windows Explorer** can be used (see Figure 4.i below), typing in:

```
ftp://httpadmin@10.0.0.145
```

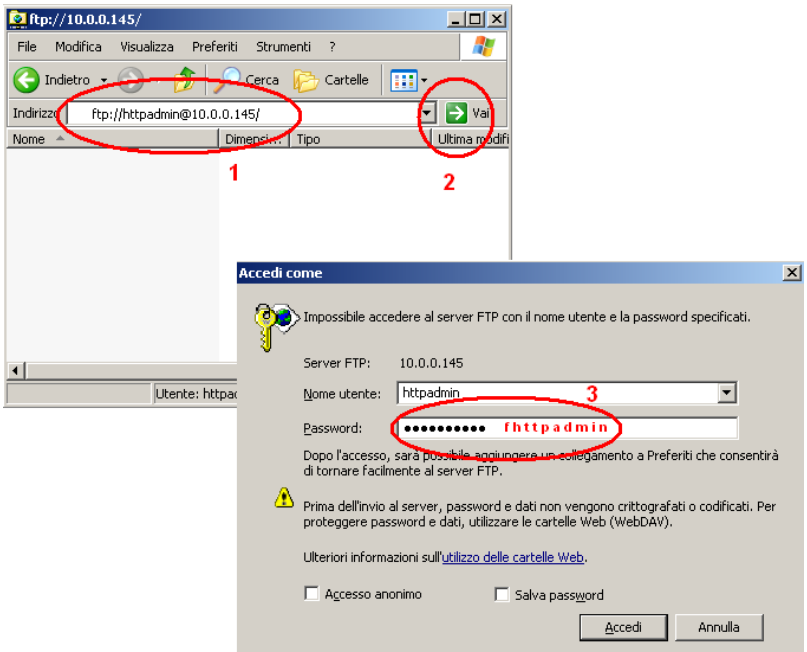


Figure 4.i – Using Windows Explorer as the FTP client

in the address field, confirming with Enter and specifying the password, fhttpadmin:

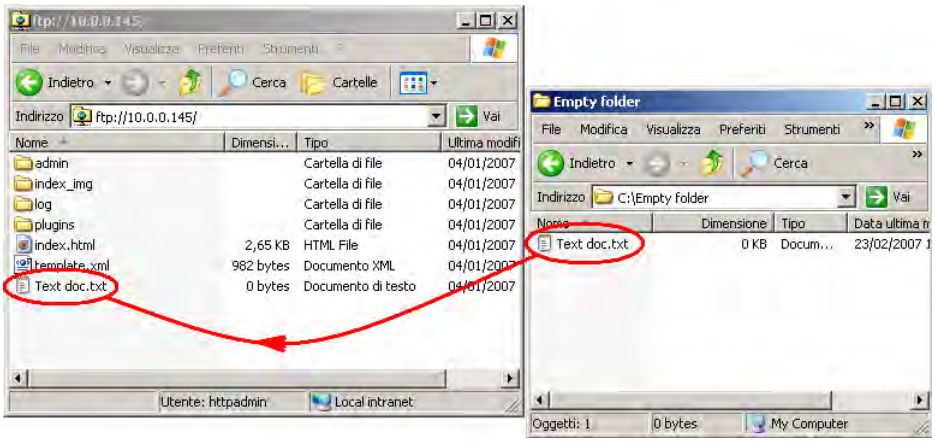


Figure 4.j – Using Windows Explorer as the FTP client

"Explorer" automatically shows the contents of the /usr/local/root/flash/http/ directory, but does not allow navigation outside of this directory. In this case, files can be transferred from *pCOWeb* to the PC and vice-versa.

4.3 EVENT NOTIFICATION: E-MAIL, FTP PUSH, SNMP TRAP/INFORM

Notification messages can be programmed to be sent when asynchronous or scheduled events occurred.
The following types of notification messages are managed:

- E-mail, the body of which can contain customised text or a web page with the values of the *pCO* controller variables read at the moment the message was sent; possibility to attach a custom XML file containing the values of the variables;
- XML file, identical to the one sent by e-mail, but in this case delivered by FTP (FTP PUSH function);
- SNMP TRAP or INFORM messages, which can contain up to 5 values of selected *pCO* variables.

INFORMATION

- To be able to send e-mails, *pCOWeb* must have access to an SMTP server on the local network or the Internet: *pCOWeb* sends the e-mail to the server and a PC can then access the server and download the messages that have been sent to it.
  APPENDIX D on page 66 presents the ArGoSoft Mail Server, a freeware application downloadable from the Internet that, once installed on the PC, can receive e-mails from the *pCOWeb* and then forward them to an e-mail client, for example Microsoft Outlook. This server is recommended for running tests without necessarily having to access an external server.

- To be able to send files via FTP, *pCOWeb* must have access to a PC on the local network or on the Internet that is running an FTP server application that can receive such files.
  APPENDIX E on page 69 presents FileZilla Server, a freeware application downloadable from the Internet that, once installed on the PC, can receive files from *pCOWeb* via FTP.
  This server is recommended for running tests without necessarily having to access an external server.

- To receive the SNMP TRAP or INFORM messages sent by *pCOWeb*, a PC must be available, on the local network or on the Internet, which is running a supervisor based on the SNMP protocol.
  APPENDIX F on page 70 presents Trap Receiver, a freeware application downloadable from the Internet that, once installed on the PC, can receive and show the notifications TRAP / INFORM from *pCOWeb*.
  This server is recommended for running tests without necessarily having to use an SNMP supervisor.

4.3.1 Events generated upon variations in the value of a variable

pCOWeb can be set to send a notification when the value of a *pCO* controller variable changes.
The types of messages sent can be selected separately for each variable. For the settings see 4.3.3 on page 23 and 4.3.4 on page 25.

Digital variables
The notifications can be sent upon the following transitions:

- 0→1 only
- 1→0 only
- both

Example 1
E-mail and FTP PUSH programmed to be sent when digital variable 12 changes from 0→1.
0→1 send both notifications
1→0 nothing sent

Example 2

E-mail and FTP PUSH programmed to be sent when digital variable 12 changes from 0⟶1 and 1⟶0.

0⟶1 send both notifications

1⟶0 send both notifications

Analogue and integer variables

For analogue and integer variables, the value that crosses a programmable threshold generates an "activation" event; if the threshold plus hysteresis is crossed backwards, a "return" event is generated.

For each variable, the following can be set:

- the direction for crossing the threshold considered as "activation";
- send notifications: only upon "activation" or upon "activation" and "return";
- the threshold (numeric value or content of a ***pCO*** variable);
- the hysteresis (numeric value or content of a ***pCO*** variable) for the return event.

In addition, two identification strings can be customised that are common to all the variables (default: "alarm fired" / "alarm reenter"), to be included in the e-mail and SNMP notifications to identify the activation and return events.

Example 1 (Figure 4.k)

FTP PUSH and SNMP TRAP programmed to be sent for analogue variable 8 **only upon activation when increasing**, with threshold 20.5 and hysteresis 1.5.

18⟶20 nothing sent

20⟶20.5 send both notifications; string included: "alarm fired"

20.5⟶22 nothing sent

22⟶19.5 nothing sent

19.5⟶23 nothing sent

23⟶19 nothing sent

19⟶23 send both notifications; string included: "alarm fired"

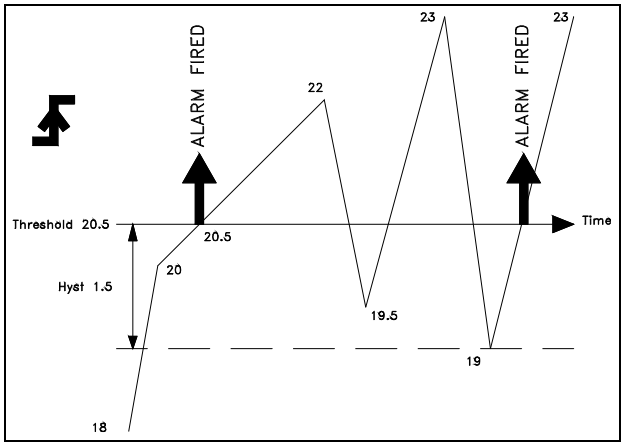


Figure 4.k – Analogue and integer variables - Example 1

Example 2 (Figure 4.l)

FTP PUSH and SNMP TRAP programmed to be sent for analogue variable 8 **upon activation when increasing and on return**, with threshold 20.5 and hysteresis 1.5.

18⟶20 nothing sent

20⟶20.5 send both notifications; string included: "alarm fired"

20.5⟶22 nothing sent

22⟶19.5 nothing sent

19.5⟶23 nothing sent

23⟶19 send both notifications; string included: "alarm reenter"

19⟶20.5 send both notifications; string included: "alarm fired"

20.5⟶23 nothing sent

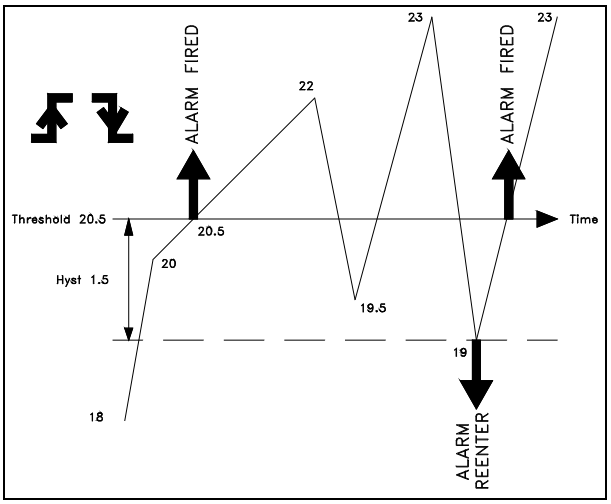


Figure 4.l - Analogue and integer variables - Example 2

Example 3 (Figure 4.m below)

FTP PUSH and SNMP TRAP programmed to be sent for analogue variable 8 **upon activation when decreasing and on return**, with threshold 20.5 and hysteresis 1.5.

23⟶21 nothing sent

21⟶20.5 send both notifications; string included: "alarm fired"

20.5⟶19 nothing sent

19⟶21.5 nothing sent

21.5⟶19.5nothing sent

19.5⟶22 send both notifications; string included: "alarm reenter"

22⟶20.5 send both notifications; string included: "alarm fired"

20.5⟶19 nothing sent

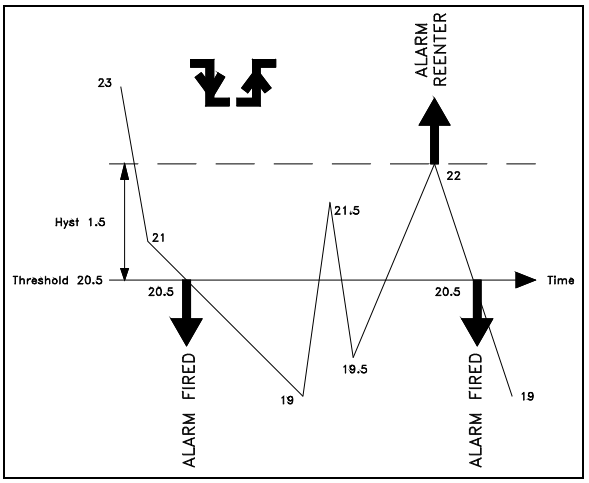


Figure 4.m - Analogue and integer variables - Example 3

4.3.2 Generation of the XML file

When a notification event occurs, ***pCOWeb*** can generate an XML file containing information on the ***pCOWeb*** and the values of the ***pCO*** variables, which can be sent via FTP or as an e-mail attachment.

XML (eXtended Markup Language) is a standard file format for conveying information organised in a diagram; it is recognised by many types of software that manage data that is imported from files. This function is useful when the receiving computer uses one of these software applications.

INFORMATION

- ***pCOWeb*** generates the file using a template based on a special file that must reside on the ***pCOWeb***; as an example see "template.xml" included by default and that can be edited by accessing the user memory via FTP (see 4.2 on page 17).

  NOTE 1 A copy of the default file "template.xml" is available in read-only memory. To call the copy simply access ***pCOWeb*** using the FTP protocol (see 4.2 on page 17) and get the file /usr/local/root/deftemplate.xml.

  NOTE 2 WordPad should be used to edit the template in the Windows operating system: in fact, the ***pCOWeb*** files conform to the Linux standard, which differs from Windows in terms of the different management of the line break control characters; WordPad can import the Linux format and save in Windows format, which can also be read by ***pCOWeb***.

- When sending, ***pCOWeb*** will read the template selected in the notification settings, and will process it based on some simple rules described further on, generating the new XML file to be sent.

HOW *pCOWeb* GENERATES THE XML FILE

pCOWeb reads the template file and copies it character-by-character to the destination file; whenever it recognises a sequence of XML tags, such as in the example shown below, it adds the corresponding information.

Table 4.a - The default "template.xml" file

LINE IN TEMPLATE.XML	MEANING	LINE GENERATED IN XML FILE TO BE SENT
<?xml version = '1.0' encoding = 'UTF-8'?>	Specifies the type of XML standard used	<?xml version = '1.0' encoding = 'UTF-8'?>
<PCOWEB>	The sequences of tags will only be recognised inside sections that start with this tag	<PCOWEB>
<SYSTEM>	Start of the section with the sequences of tags with ***pCOWeb*** properties (that is, not ***pCO***)	<SYSTEM>
<HOSTNAME></HOSTNAME>	Returns the name of the ***pCOWeb*** – always "localhost"	<HOSTNAME>localhost</HOSTNAME>
<DATE></DATE>	Returns date and time the file was generated, in the format YYYYMMDDHHMMSS	<DATE>19700101064832</DATE>
<IP_ADDRESS></IP_ADDRESS>	Returns the IP address of the ***pCOWeb***	<IP_ADDRESS>10.0.3.114</IP_ADDRESS>
<MAC_ADDRESS></MAC_ADDRESS>	Returns the MAC address of the ***pCOWeb***	<MAC_ADDRESS>00:0a:5c:10:07:15</MAC_ADDRESS>
<UPTIME></UPTIME>	Returns the time elapsed since the last reboot of the ***pCOWeb***	<UPTIME>0d2h42m</UPTIME>
<SYS_VERSION></SYS_VERSION>	Returns the firmware version of the ***pCOWeb*** (block B - Bios)	<SYS_VERSION>B1.2.1</SYS_VERSION>
<APP_VERSION></APP_VERSION>	Returns the firmware version of the ***pCOWeb*** (block A - Applications)	<APP_VERSION>A1.4.2</APP_VERSION>
<SEPARATOR_CHAR>.</SEPARATOR_CHAR>	No action. Suggests the character that the XML interpreter can use as the separator for the values of the analogue variables	<SEPARATOR_CHAR>.</SEPARATOR_CHAR>
</SYSTEM>	End of the section with the sequences of tags with ***pCOWeb*** properties (that is, not ***pCO***)	</SYSTEM>
<PCO>	Start of the section with the sequences of tags with ***pCO*** properties	<PCO>
<PCONAME>Template Sample</PCONAME>	No action. Line used to assign a name to the specific ***pCO***; in the example, the name is "Template Sample"	<PCONAME>Template Sample</PCONAME>
<PCOID>1</PCOID>	No action. Line used to assign an identifier number to the specific ***pCO***; in the example, the number is "1"	<PCOID>1</PCOID>
<DIGITAL>	Start of a section for digital variables	<DIGITAL>
<VARIABLE>	Start of the section for one digital variable	<VARIABLE>
<INDEX>1</INDEX>	Specifies the index 1 for the variable to ***pCOWeb***	<INDEX>1</INDEX>
<VALUE></VALUE>	***pCOWeb*** returns the current value (0) of the digital variable with index 1	<VALUE>0</VALUE>
</VARIABLE>	End of the section for the digital variable with index 1	</VARIABLE>
<VARIABLE>	Start of the section for another digital variable	<VARIABLE>
<INDEX>2</INDEX>	Specifies the index 2 for the variable to ***pCOWeb***	<INDEX>2</INDEX>
<VALUE></VALUE>	***pCOWeb*** returns the current value (1) of the digital variable with index 2	<VALUE>1</VALUE>
</VARIABLE>	End of the section for the digital variable with index 2	</VARIABLE>
</DIGITAL>	End of a section for digital variables	</DIGITAL>
<INTEGER>	Start of a section for integer variables	<INTEGER>
<VARIABLE>	Start of the section for an integer variable	<VARIABLE>
<INDEX>1</INDEX>	Specifies the index 1 for the variable to ***pCOWeb***	<INDEX>1</INDEX>
<VALUE></VALUE>	***pCOWeb*** returns the current value (25) of the integer variable with index 1	<VALUE>25</VALUE>
</VARIABLE>	End of the section of the integer variable with index 1	</VARIABLE>
<VARIABLE>	Start of section for another integer variable	<VARIABLE>
<INDEX>2</INDEX>	Specifies the index 2 for the variable to ***pCOWeb***	<INDEX>2</INDEX>
<VALUE></VALUE>	***pCOWeb*** returns the current value (200) of the digital variable with index 2	<VALUE>200</VALUE>
</VARIABLE>	End of the section for the integer variable with index 2	</VARIABLE>
</INTEGER>	End of a section for integer variables	</INTEGER>
<ANALOG>	Start of a section for analogue variables	<ANALOG>
<VARIABLE>	Start of the section for an analogue variable	<VARIABLE>
<INDEX>1</INDEX>	Specifies the index 1 for the variable to ***pCOWeb***	<INDEX>1</INDEX>

<VALUE></VALUE>	***pCOWeb*** returns the current value multiplied by 10 (999.9 x 10 = 9999) of the analogue variable with index 1	<VALUE>9999</VALUE>
</VARIABLE>	End of the section for the analogue variable with index 1	</VARIABLE>
<VARIABLE>	Start of the section for another analogue variable	<VARIABLE>
<INDEX>2</INDEX>	Specifies the index 2 for the variable to ***pCOWeb***	<INDEX>2</INDEX>
<VALUE></VALUE>	***pCOWeb*** returns the current value multiplied by 10 (-12 x 10 = -120) of the analogue variable with index 2	<VALUE>-120</VALUE>
</VARIABLE>	End of the section for the analogue variable with index 2	</VARIABLE>
</ANALOG>	End of a section for analogue variables	</ANALOG>
</PCO>	End of the section with the sequences of tags with ***pCO*** properties	</PCO>
</PCOWEB>	End of the section recognising the sequences of tags	</PCOWEB>

NOTES on using the XML standard

- The syntax of a file with the .xml extension can be validated, for example, by displaying it in Internet Explorer; ***pCOWeb*** nonetheless only recognises the syntax shown above. In particular, it does not recognise the standard XML <text /> tag, which in standard XML is equivalent to the pair <text> + <text/>.
- ***pCOWeb*** does not validate the template file used.
- Information, strings and tags other than those shown above can be entered in any position; if not recognised according to the syntactical limits shown in Table 4.a on page 22, they will be copied textually without any action.

4.3.3 Setting the common properties to all events

Par. 4.3.1 on page 20 describes the operation of the notification events. Below is a description of the setting procedure.

IMPORTANT **For *pCOWeb* to send the notifications, normally the "Gateway" needs to be set in the network configuration (see 9.4.1 on page 44). The gateway configuration is not however required if the address of the *pCOWeb* and the addresses of the receiving network devices are in the same subnetwork. For further information see APPENDIX B on page 62.**

NOTE 1 All the following settings are valid from when they are confirmed; the ***pCOWeb*** does not need to be rebooted.

NOTE 2 The Tests page (see 9.8 on page 51) can be used to manually send all the notifications set, regardless of whether the events occur; once the events have been set, it is recommended to check the sending of the notifications on this page.

NOTE 3 APPENDIX D on page 66, APPENDIX E on page 69 and APPENDIX F on page 70 describe some freeware applications downloadable from the Internet that, once installed on a computer, can receive e-mails, files via FTP and TRAP messages from the ***pCOWeb***.
These should be used for testing without requiring an external server.

PROCEDURE

a. From the ***pCOWeb*** main page (see Figure 9.b on page 41) open the "Events" page (click point 1 in Figure 4.n).
b. Open the window for setting the recipients and the attached files (click point 2 in Figure 4.n).
The settings in this window will be common to all the events generated.

The window is divided vertically into three areas (Figure 4.o on page 24, Figure 4.p on page 24 and Figure 4.q on page 24).
Each has a confirmation button that is only valid for the corresponding area.

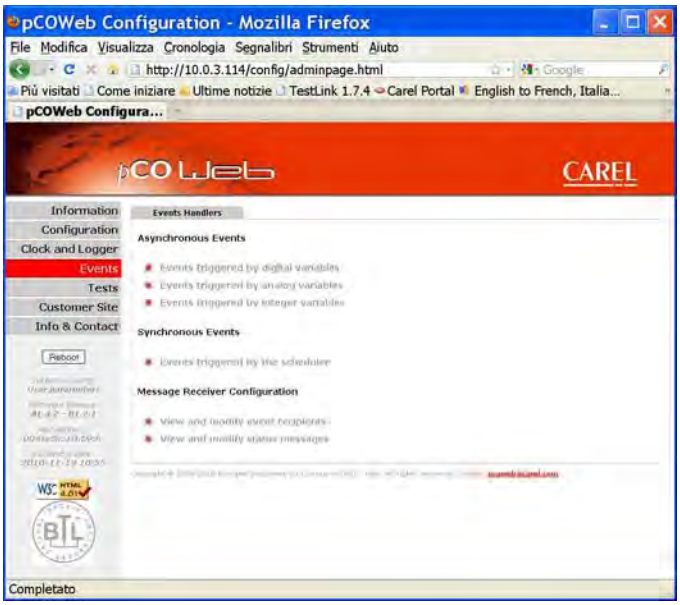


Figure 4.n – Notification events page

"SNMP Hosts": SNMP TRAP recipients (see chapter 6 on page 33).
Up to 5 recipients can be specified; to be able to receive SNMP TRAP notifications, at least the first (Host 1) needs to be specified.

SETTINGS AVAILABLE:

- Hostname: IP address or name of the recipient PC (to use a name, a DNS must be specified, see 9.4.1 on page 44).
- Community: this represents a sort of SNMP keyword that is sent (not encrypted) to the recipient, which can exclude any notifications from communities other than its own; if left empty, the Default trap community will be used (see 6.4 on page 35), but if that too is empty, the TRAP will not be sent to the Host.

Figure 4.o – SNMP Hosts configuration for sending the TRAP messages

- **"E-mail Configuration"**: e-mail recipients.
  - Username / Password: Username and Password to access the SMTP server used by ***pCOWeb*** to send the messages; for some servers this information is required, while for others it isn't.
  - E-Mail Account (required for sending e-mail notifications): in the message received this will be shown as the sender's e-mail address.
    If this address is required to send e-mails to the ***pCOWeb***, remember that ***pCOWeb*** does not feature a client for reading or downloading the messages; these must then be read / downloaded using a client (for example "Microsoft Outlook") that accesses the e-mail server corresponding to the domain address in this field (in the example "provider1.com"); the corresponding MAILBOX must have been created on the server: also see the APPENDIX D on page 66.
  - Identification (optional): in the message received this will be displayed as the sender's name.
  - Reply to (optional): if the recipient selects the "Reply to" option in the e-mail client used, this represents the return e-mail address that the reply will be sent to.
  - **Destination #N (at least #1 required): e-mail addresses that *pCOWeb* will send the messages to.**
  - SMTP Server Address (required): IP address or name of the server that ***pCOWeb*** connects to in order to send the messages; if a name is specified instead of an IP address, a DNS must have been specified (see 9.4.1 on page 44).
  - XML template for attachment (required for attachments): this is used to choose the XML template resident on the ***pCOWeb*** that will be used as the basis for generating the attached XML file (see 4.3.2 on page 22); "Choose" opens a list of files with the .xml extension in /usr/local/root/flash/http; if the file name is entered, the path must be specified starting from the root (/).
  - Attached file name (optional): name that will be attributed to the attached XML file; if preceded by the "(date)" string, the name will start with the date / time on the ***pCOWeb*** at the instant the file was generated, in the format YYYY-MM-DD_hhmmss; leaving this field empty, the name will be assigned automatically, made up of the date / time, in the format: YYYY-MM-DD_hhmmss.xml.
  - Offset from UTC (optional): this represents the offset between local time (where ***pCOWeb*** is installed) and UTC. It has to be expressed, as examples, +0100 (Paris, Rome, Berlin time), -0500 (New York time) or +0530 (Mumbai time)

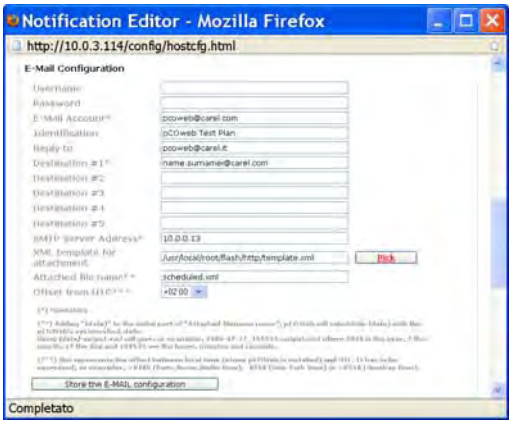


Figure 4.p – E-mail recipient configuration

- **"Ftp Push Configuration"**: FTP PUSH recipients for sending XML files.
  - Input Filename (required): used to choose the XML template resident on ***pCOWeb*** that will be used as the basis for generating the XML file sent (see 4.3.2 on page 22); "Choose" opens a list of files with the .xml extension in /usr/local/root/flash/http; if the file name is entered, the path must be specified starting from the root (/).
  - Output file name (required): name that will be attributed to the attached XML file; if preceded by the "(date)" string, the name will start with the date / time on the ***pCOWeb*** at the instant the file was generated, in the format YYYY-MM-DD_hhmmss.
  - Enabled (at least one required): enables the recipients, where the file will be push for the events relating to the variable.
    IMPORTANT: any Recipients enabled after one that is not enabled will be ignored.
    Example: 1: Yes 2: No 3: Yes 4: Yes 5: No gives as result that Recipients 3 and 4 will not be used.
  - Server Address (required to send notifications via FTP): IP address or name of the FTP server that ***pCOWeb*** will connect to in order to deliver the file; if a name is specified instead of an IP address, a DNS must have been specified (see 9.4.1 on page 44); for an example of an FTP server, see APPENDIX E on page 69.
  - Server Port (optional): specify only if a TCP port other than the default 21 is to be used.

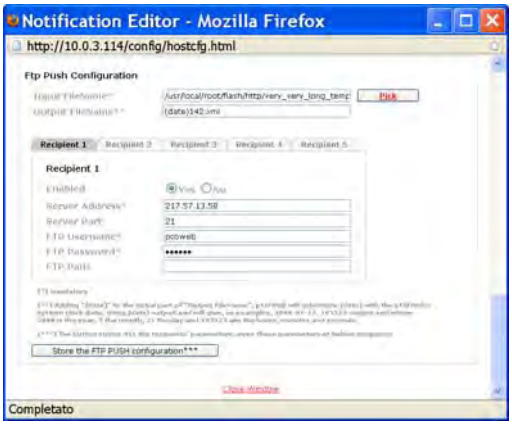


Figure 4.q – FTP PUSH configuration

- FTP Username / FTP Password (required): Username and Password to access the FTP server;
  IMPORTANT: Username and Password must have a write access, otherwise pCOweb can not push the file inside the directory.
- FTP path (optional): path of the server, in the default directory for the specified Username, where the sent file will be written; the path must exist, otherwise the file cannot be received.

c. Open the window for setting the strings identifying the events for integer and analogue variables (click point 3 in Figure 4.n on page 23).
The settings in this window (Figure 4.r) will only be used for e-mail and SNMP TRAP/INFORM messages and will be common to the events generated by all the integer and analogue variables.
In the e-mail notifications, the suitable string will be automatically added to the "Subject" field, separated by a dash; in the SNMP TRAP notifications, the OID will be included automatically, with the corresponding string and contents, based on the event, as follows (see 6.2 on page 34):

`1.3.6.1.4.1.9839.1.3.1.1.0` (OID for the Alarm fired event string)
`1.3.6.1.4.1.9839.1.3.1.2.0` (OID for the Alarm reentered event string)

SETTINGS AVAILABLE:
- Alarm fired: string that will be included in the notifications when the threshold is exceeded.
- Alarm reentered: string that will be included in the notifications when returning back below the threshold, with hysteresis.

Figure 4.r – Setting the strings sent for Activation - Return

4.3.4 Setting the notifications set upon variations in the variables

IMPORTANT Some settings described below ("Event Handlers", "Trap Configuration" and "E-Mail Configuration") are also used for the scheduled events (see 4.3.5 on page 27).

NOTE: all the following settings are valid from when they are confirmed; the ***pCOWeb*** does not need to be rebooted.

PROCEDURE

1. Open the window for setting the events corresponding to the **DIGITAL** variables (click point 4 in Figure 4.n on page 23).
   A summary window is displayed for enabling the events relating to the variation of all the digital variables (Figure 4.s below). Moving the mouse, for example, over the section relating to digital variable 1 shows the tooltip "Digital Variable 1"; clicking opens the window, divided into three sections, for setting the events linked to digital variable 1.

   IMPORTANT: The range of variables related to the notifications goes from 0 to 207, even if the ModBus Extended protocol is running. Variables' addresses for notifications can go just from 0 to 207.
   IMPORTANT: The Submit buttons only save the values for the corresponding section.

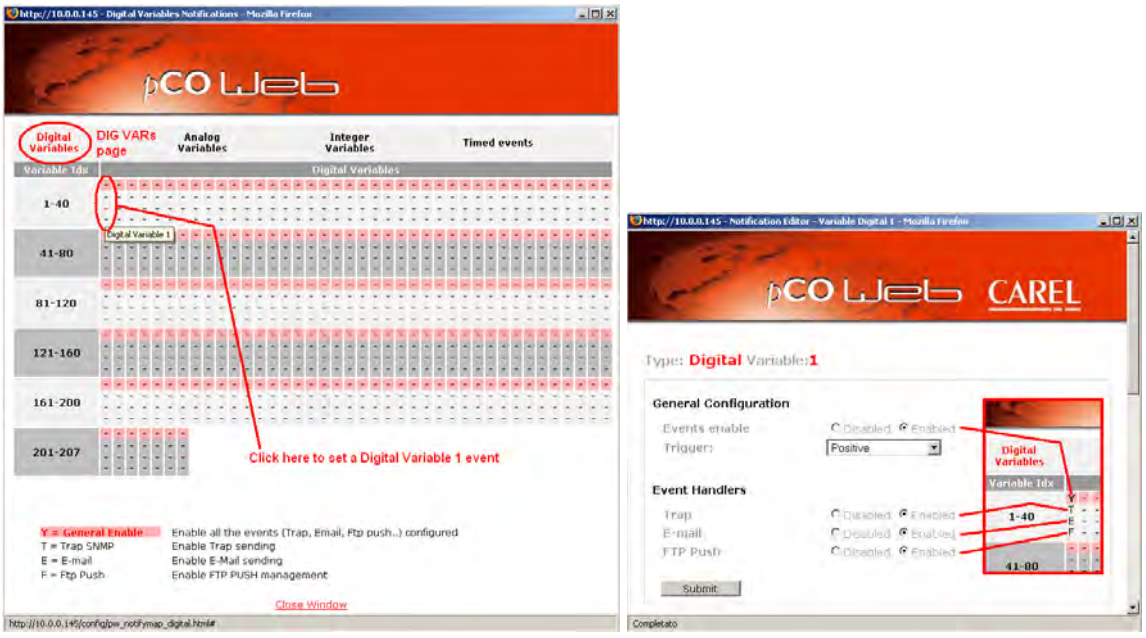


Figure 4.s - Events generated by digital variables

Type: **Digital** Variable: 1 : the settings in window only refer to the digital variable with index 1.

- TOP SECTION – COMMON SETTINGS: **General Configuration** and **Event Handlers** (Figure 4.s - right - on page 25)
  - Events enable: enable all the events generated by variations to the variable.
  - Trigger: a notification event is generated when there are following variations in the value of the variable:
    Positive: 0→1;
    Negative: 1→0;
    Positive & negative: 0→1 or 1→0;
  - Trap / E-mail / FTP Push: select one or more types of notification to be sent when the event occurs.
    IMPORTANT: these decisions are also used to send notifications for scheduled events (see 4.3.5 on page 27).

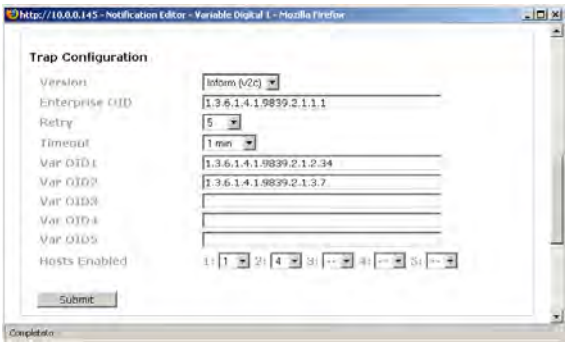


Figure 4.t - TRAP configuration for an event

- MIDDLE SECTION: **Trap Configuration**
  - Version: type of version; V1: conforms to SNMPv1; Inform (v2c): conforms to SNMPv2c. Select in accordance with the version of the SNMP supervisor that will receive the notifications.
  - Enterprise OID (**required sending the TRAP message**): description of the TRAP object; sent together with the TRAP, allows the receiver to associate the correct meaning with the message based on the information included in the MIB description file.
    IMPORTANT: ***pCOWeb*** sends the TRAP only if the string entered respects the syntax of an OID.
  - Retry / Timeout (optional, only used if Version=Inform (v2c)): specifies the maximum number of send attempts after the first, and the time interval for retrying the TRAP (INFORM); when reception is acknowledged by the receiver, the send attempts are stopped.
  - Var OID #N (optional): specifies an SNMP object whose OID and corresponding value will be attached with the message; for example, specifying the OID of a ***pCO*** variable, when the TRAP is sent ***pCOWeb*** will acquire the numeric value of the ***pCO*** variable specified by the OID set and will attach this with the message, together with the OID.
  - Hosts Enabled (at least one required): enables the recipients (Hosts), from those set in Figure 4.o on page 24, that the TRAP will be sent to for the events relating to the variable.
    IMPORTANT: any Hosts enabled after one that is not enabled will be ignored.
    Example: 1: 1 2: -- 3: 3 4: 4 5: --
    Hosts 3 and 4 will not be used.

- BOTTOM SECTION: **E-Mail Configuration**
  - Subject (optional): in the message received, the text entered will appear as the Subject.
  - Body: select the contents for the body of the e-mail message:
    From file: the name of a file with the .html extension can be selected; "Choose" opens the list of the .html files in /usr/local/root/flash/http/; if entering the name of the file, remember that the starting directory for the search is /usr/local/root/.

Figure 4.u - E-mail configuration for an event

NOTE If the file includes the line:

```
<!--tagparser="/pcotagfilt"-->
```

the contents of the file, before being added to the body of the message, will be processed by the ***pCOWeb*** WEB server (see APPENDIX G on page 71); in this way, by creating simple HTML pages, the body of the message can contain the value of some ***pCO*** variables read at the moment the message was sent.
pCOWeb does not support the encapsulation of images inside the e-mail messages sent: alternatively, the HTML page can contain links to image files; the e-mail client will highlight these with bookmarks and the links can be opened by clicking the right mouse button. To simply copy the text of the document without this being processed, do not include the line shown above.

From text: alternatively, the text entered in the text box on the side can be added. ***pCOWeb*** will automatically add the date / time string `Date: dd/mm/yyyy hh:mm`, and then the text entered to the body of the message.

  - Enable Attachment: enable the sending of the attachment, see "XML template for attachment" and "attached file name" on page 24

2. Open the window for setting the events relating to the **ANALOGUE** variables (click point 5 in Figure 4.n on page 23 or click "Analogue Variables" in the settings summary window – Figure 4.v on page 27).
   A summary window is displayed for enabling the events relating to the variation of all the analogue variables. Moving the mouse, for example, over the section relating to analogue variable 7 shows the tooltip "Analogue Variable 7"; clicking opens the window, divided into three sections, for setting the events linked to analogue variable 7.

   IMPORTANT: The range of variables related to the notifications goes from 0 to 207, even if the ModBus Extended protocol is running. Variables' addresses for notifications can go just from 0 to 207.
   IMPORTANT: The Submit buttons only save the values for the corresponding section.

Type: **Analogue** Variable: **7** : the settings in window only refer to the analogue variable with index 7.

- TOP SECTION – COMMON SETTINGS: **General Configuration** and **Event Handlers** (Figure 4.v - right)
  Paragraph 4.3.1 on page 20 describes the generation of events based on a threshold. Below is the meaning of the settings on the page.

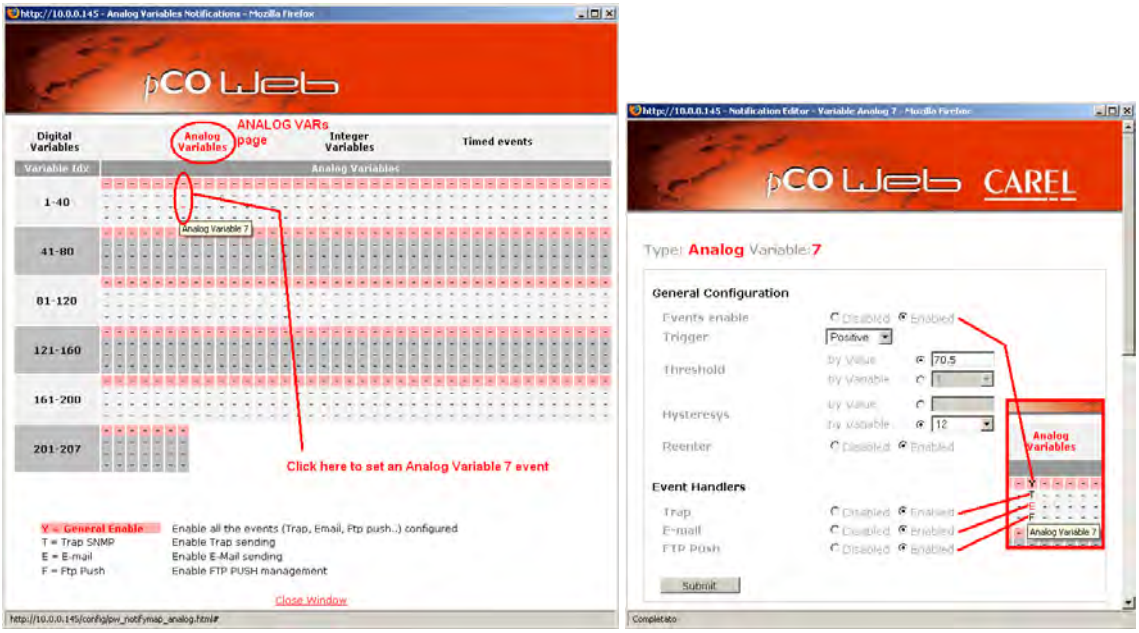


Figure 4.v - Events generated by analogue variables

  - Events enable: enable all the events generated by variations to the variable.
  - Trigger: an "Activation" notification event is generated when there are following variations in the value of the variable:
    Positive: the value of the variable crosses the threshold while increasing;
    Negative: the value of the variable crosses the threshold while decreasing.
  - Threshold / Hysteresis: crossing threshold / return hysteresis, either values that are entered or analogue variables that can be selected.
  - Reenter: enable the generation of a notification event when returning.
  - Event Handlers: select the type of notification that will be sent when the events set above occur.
    IMPORTANT: these decisions are also used to send notifications for scheduled events (see 4.3.5 below).

- MIDDLE AND BOTTOM SECTION: **Trap Configuration** and **E-Mail Configuration**
  These sections and the corresponding functions are identical to those described for digital variables.

3. Open the window for setting the events corresponding to the **INTEGER** variables (click point 6 in Figure 4.n on page 23, or click "Integer Variables" on the settings summary window – Figure 4.w).
   The windows and the settings are similar to those for the analogue variables.
   It should be noted that in this case Threshold / Hysteresis can be set as values entered or as integer variables that can be selected.

Figure 4.w - Opening the integer variable summary settings

4.3.5 Scheduled events (generated at time intervals)

pCOWeb can be set to generate events for sending notification when a set and repeated time interval expires.

PROCEDURE

1. Open the window for setting the scheduled events (click point 7 in Figure 4.n on page 23).

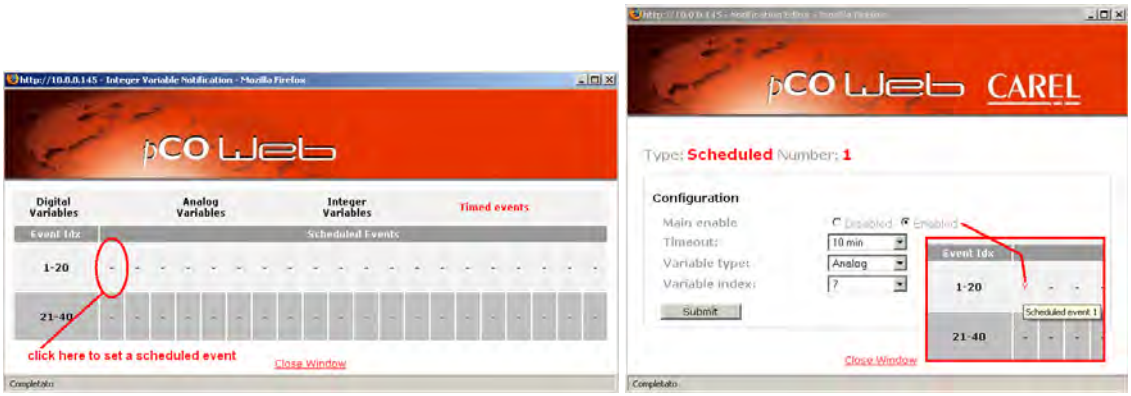


Figure 4.x - Scheduled events

Type: Scheduled Number: 1: settings for scheduled event 1; overall 40 settings windows are available.

SETTINGS AVAILABLE:

- Main enable: enable the generation of scheduled events for this window.
- Timeout: repeat time; the first event will be generated after the Timeout has elapsed from when the "Submit" button was selected; the following events will be generated when the "Timeout" has elapsed again.
- Variable type / Variable index: the scheduled events in the current setting window use some decisions set in the windows for the events based on variations (see 4.3.4 on page 25) related to the variable selected by these two settings.
  The shared settings are in the following sections:
  Events Handlers
  Trap Configuration
  E-mail Configuration

NOTE The SNMP TRAP and e-mail notifications relating to scheduled events for analogue and integer variables always automatically include the string entered in the "Alarm fired" field in Figure 4.r on page 25.

5 CLOCK AND LOGGER

5.1 CONFIGURING CLOCK WITH *pCO* SYNCHRONIZATION

pCOWeb features an internal clock that keeps its settings only while power is supplied.
When rebooting, the date / time is reset to: 1970-01-01 00:00. ***pCOWeb*** can be configured so that every minute it sets its own clock based on the time set on the ***pCO*** controller.
For this to occur, the application program running on the ***pCO*** must continuously transmit the information on five selected integer supervisory variables.

The page shown in Figure 5.a is used to enable the synchronisation function and specify the indexes of the five integer supervisory variables.

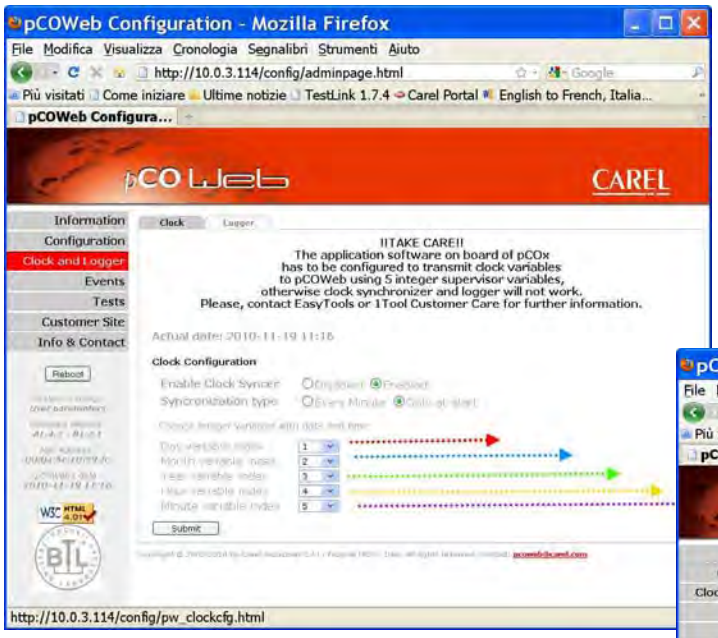


Figure 5.a - Setting clock synchronisation

The page shown in Figure 5.b is used to show how integers' values are related to the day, month, year, hour and minute data sent in supervision from the ***pCO*** application software.

Figure 5.b shows as first integer a variable with value '19'; if in the variable list of the ***pCO*** application you find that the first integer represents the variable current_day you can set in Figure 5.a "Day variable index" = '1' and the ***pCOWeb*** will show in the date '19' i.e. the 19th day of the month.

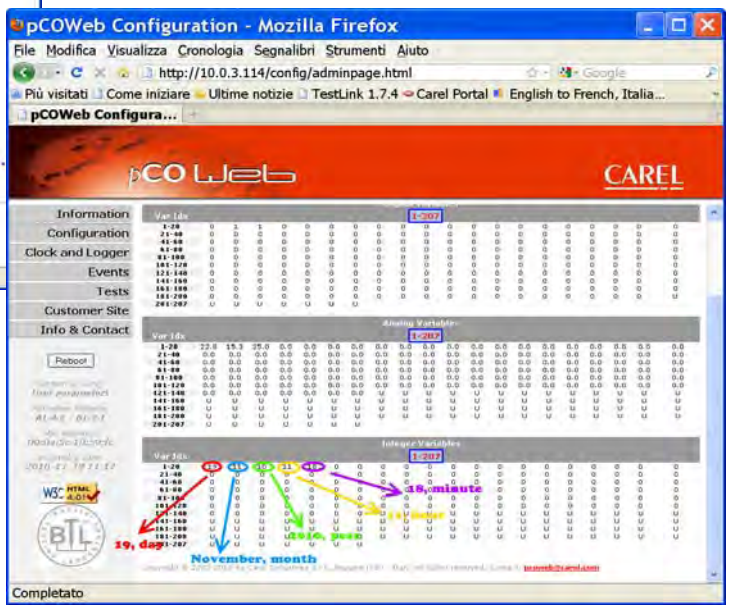


Figure 5.b – Relation between indexes and values

NOTE 1 The settings will be valid as soon as they are confirmed; ***pCOWeb*** does not need to be rebooted.

NOTE 2 (For version < 1.4.2) The time indicated on the left of the page only changes when the page is completely refreshed (for example, by pressing F5 on the computer keyboard).

5.2 CONFIGURING CLOCK WITHOUT *pCO* SYNCHRONIZATION

1. Using BACnet protocol commands (e.g. Time Sync/UTC Time Sync in BACset: refer to BACset manual for more information).

2. Installing the NTP plugin (downloadable from http://ksa.carel.com) it is possible to let the ***pCOweb*** run the time supplied by a NTP server, in fact the NTP plugin contains the file called ntp_client.conf, which includes the following lines:
   ```
   NTP_URL=10.0.0.7
   UTC_OFF=+2
   ```
   The first line defines the NTP server used for communication, an IP address (the NTP server address) is required to let ***pCOWeb*** reaches the NTP server.
   The second line gives the possibility to set an offset to the UTC time (2 hours in the example above).

NOTE 1 When using these methods, synchronisation with the time on the controller must be disabled

5.3 LOGGER AND GRAPHS

pCOWeb can save the values of max 20 variables sampled at regular time intervals to non-volatile memory ("Logger" function).
pCOWeb can create also a graph using the vector format SVG and representing in the same graph up to 5 of the logged variables. SVG files can be natively viewed using Mozilla Firefox, Chrome and Safari browser, while instead Internet Explorer 8 needs an external plug-in (e.g.: http://www.adobe.com/svg/viewer/install/)

IMPORTANT: the logging function will not start if the date of the ***pCOWeb*** is earlier than 01-01-2006. For the date / time settings see 5.1 and 5.2 on page 29.
IMPORTANT: The range of addresses of loggable goes from 0 to 207, even if the ModBus Extended protocol is running.

START OR DISPLAY THE RECORDS

1. Open the Clock and Logger page (Figure 5.c).
2. Click the "Logger" tab.
   The settings are located in the bottom section of the page (Figure 5.d below); they becomes active just after the "Submit" confirmation.

Figure 5.c - Opening the Logger and Graph settings

SETTINGS AVAILABLE:

- Check logger status and free space available: a window is opened that shows the status of the logger ("Running" / "Not Running" / "Waiting for a valid clock configuration") and the space occupied and free in the non-volatile memory.
- Time sample: sample time, common to all the variables.
- Compression (.gz): the data files created are .cvs; if Compression is enabled, compressed files in .gz format are created, and can be opened using various decompression tools, for example the "Filzip" freeware, http://www.filzip.com (but not with the decompression tool included by default in the Windows operating system); this is used to save space in the memory, as the data files are made up of highly repetitive data, a situation that allows significant compression.
- Variable 1-20: enable the logging of the variables;
- Name: Identifier which will be used in the graphical representation.;
- Type: Type of variable (Digital/Analogue/Integer);
- Index: Numeric identifier of the variable in the database of variables which could be monitored by the monitoring system (index);
- Short Description: A short description which will be used in the CSV file to describe the variable;
- Group: see below;
- Colour: Colour of the line used to represent the variable in the graph.

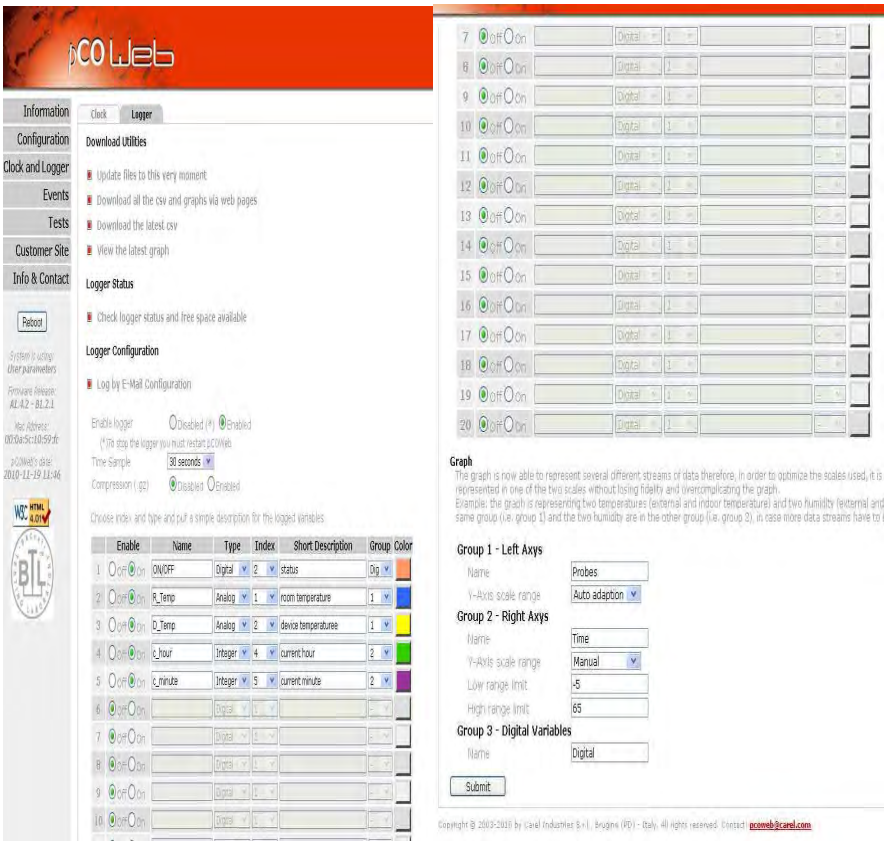


Figure 5.d – Logger and graphs: configuration

GROUPS AND SCALES
The graph is able to represent up to five different streams of data and therefore, in order optimize the scales used, it is necessary to define two or more sets of data (temperature, pressure, power consumption...) so that they can be gathered together and therefore represented in one of the two scales (one on left-hand side of the graph and one of the right-hand side) without losing resolution and quality or overcomplicating the graph.
Each and single variable represented in the graph has to be assigned to one of the following groups:

- Group 1 - Left Axis
- Group 2 – Right Axis
- Group 3 – Digital Variables (Dig)

To correctly represent the variables in the graph it is very important to define meaningful ranges for the scales, these ranges can be obtained by:

- Auto Adaption: the graph will automatically calculate the highest and lowest value reached by the all the variables of the same group and automatically adjust the scale
- Manual: Limits are defined by the user, if one or more values overtake the boundaries they will be showed like if they are going of the graph.

IMPORTANT:
pCOWeb offers also the possibility to make graph in BMP format (the only available format before 1.4.2);

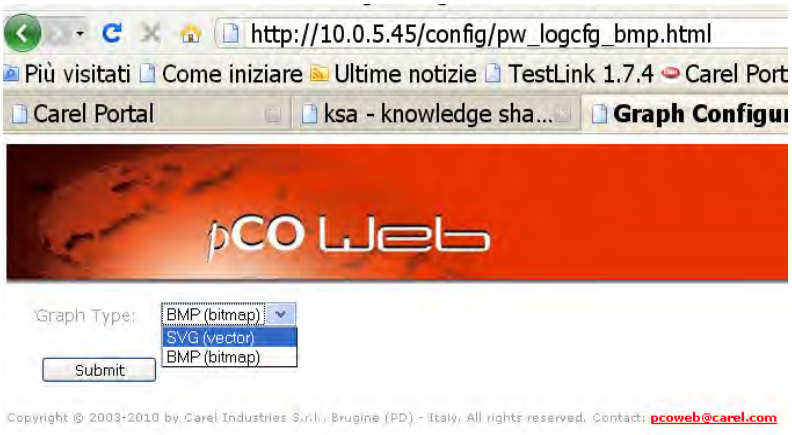


Figure 5.e – Relation between indexes and values

pCOWeb creates a file in .bmp format containing a simple graph of the trend in the values of just one of the logged variable. There is a specific html page to set the type of the graph called "pw_logcfg_bmp.html" typing in the browser http://10.0.5.45/config/pw_logcfg_bmp.html

pCOWeb will display the page on the Figure beside:

After switching pcoweb to the favourite graph format, it is necessary to go back to the logger page and configure again the logged variables. After this, to let the new settings work it is mandatory to reboot.

HOW THE RECORDS ARE SAVED
Whenever the specified time interval expires, the sampled values are added to a temporary file resident in VOLATILE memory; at 00 minutes each hour, the data is downloaded to a file (history_diskbuffer) in proprietary format in NON-VOLATILE memory (/usr/local/root/lash/http/cache).
At midnight each day this file is recopied in .cvs text format (compressed when enabled) in NON-VOLATILE memory accessible to the user, in the "/usr/local/root/lash/http/log" directory, organised into monthly subdirectories (2007-02, 2007-03, etc...); the graph file is created (if enabled) in .svg format in the same memory space.

The logging mode is optimised for a minimum number of writes to non-volatile memory, so as to reduce the degradation of the memory (minimum guaranteed write cycles: 100,000).
NOTE: in the event of power failures, this logging mode limits the maximum duration of data loss to one hour.

Nonetheless, the .cvs / .svg files for the current day can be created manually, so as to obtain all the data saved until that moment.

In the "/usr/local/root/flash/http/log" directory, ***pCOWeb*** automatically saves the powerup-log.cvs file containing the records, with the date / time of the events: "Start firmware" and "Stop firmware", respectively "Power-up" and "Power-down" (the latter is only written when the Stop follows a reboot while pressing the physical button or selecting the button on the configuration page; in this way, two consecutive "Power-up" events indicate that there was a power failure between the two events.

The .cvs files and the graphs appear as shown in Figure 5.f.
The first line of the .cvs files lists the types of variables enabled for logging, the second line the indices and the third the descriptions attributed; these are followed by the lines with the records.

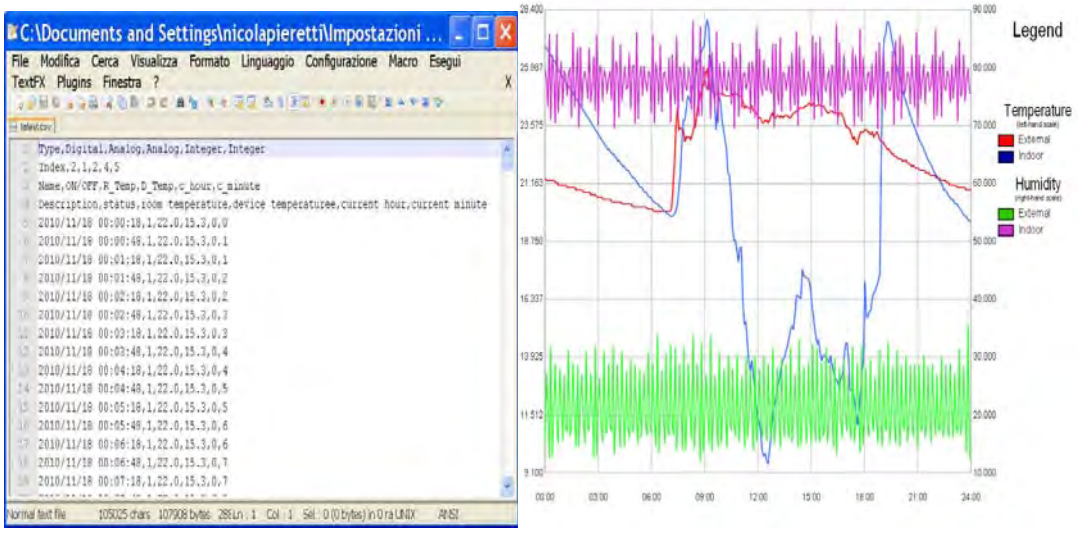
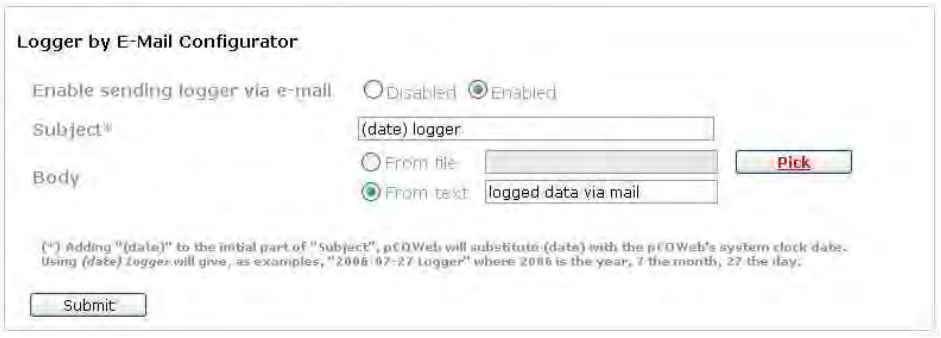


Figure 5.f – Logger and graphs: end results

LOGGER VIA E_MAIL

There is the possibility to send an e-mail at midnight every day. Just enable the sending logger in Figure 5.g



- Subject: It gives the object of the mail
- Body: it allows to create a specific body for the mail sent, from a html file (downloaded via FTP in the http folder) or the body could be a text typed directly in the empty field

Figure 5.g – Logger and graphs: configuration

DOWNLOADING THE RECORDS:

Thee top section of the page (Figure 5. sotto) features the buttons for displaying and downloading the data.

- Update cvs file and graph: manually create the .cvs file (and, if enabled, the .bmp) relating to the current day, so as to acquire all the data saved until that moment; during the update, a window is displayed that shows the activity in progress; the procedure may take around one minute; the window must be closed manually at the end.
  If this button has not been pressed at least once, the data for the current day will only be available starting from the next day (in any case, at midnight the files are overwritten, including the data for the entire day).
- Download all the cvs and graphs (the password for the Username httpadmin has to be entered, which by default is "fhttpadmin"): a window is opened (this may take some time) showing the list of directories for the months in which records have been saved; to download, select the directory and then click a .cvs file; clicking a .bmp file opens a window that displays the graph.
- Download only the latest cvs file: this function only downloads the .cvs file for the current day; the file must first have been created using the "Update cvs file and graph" function.
- View last graph: a window is opened that displays the graph relating to the current day; the file must first have been created using the "Update cvs file and graph" function.

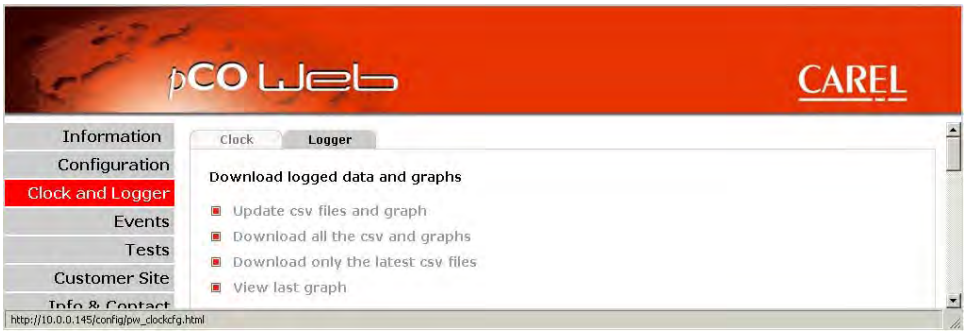


Figure 5.h – Discharge logger and graphs

IMPORTANT Whenever the configuration of the Logger is changed during the day, ***pCOWeb*** retains the values saved until that moment but updates the first three lines of the header (see Figure 5.f - left); if the selection of the logged variables is changed and the records saved until that moment need to be retained, proceed as follows:

1. before changing the configuration, save the data to the PC by first selecting Update cvs file and graph, then Download all the cvs and graphs;
2. disable all the variables currently selected for logging;
3. manually delete the file "history_diskbuffer" in the /usr/local/root/flash/http/cache directory by accessing the ***pCOWeb*** via FTP, with the "root" Username / Password (default "froot"); make sure not to modify other files / directories in this phase, as the "root" Username, in opposition to the case of "httpadmin", has no restrictions;
4. reboot ***pCOWeb***;
5. then restart the Logger, selecting the new variables for logging.

6 SNMP

SNMP (Simple Network Management Protocol) is a protocol used in Ethernet networks for controlling and setting the parameters for the network devices, for example switches and network printers.

A complete description of the SNMP protocol is not within the scope of this document. The system integrator, who is usually responsible for setting the various parameters, checking network communication and setting up the supervision system, should know how SNMP works.

A simple system based on the SNMP protocol normally features a series of devices, each containing an SNMP Agent, as well as a central supervisor called the NMS – Network Management Station – which periodically queries the devices, acquiring the status and where necessary setting the operating parameters.

The simplicity of its messages means that SNMP is becoming increasingly widespread, above all for the control of industrial devices.

pCOWeb includes an SNMP v2c Agent, that is, an application that responds to network queries in SNMP protocol version 1 and 2c.

The ***pCOWeb*** SNMP Agent is developed using the Net-SNMP open source package, version 5.0.9; refer to this application for further information.

TRAP/INFORM
pCOWeb features a TRAP generator that sends notification messages to an NMS in SNMP TRAP or INFORM v2C format when events occur relating to the ***pCO*** variables, as set by the user. The TRAP/INFORM messages are useful because they are sent by the Agent, thus providing information in real time regardless of whether they have been queried by the supervisor. To set the SNMP TRAP notifications, see paragraph 4.3 on page 20.

6.1 OVERVIEW OF SNMP

OID
The main concept found in the SNMP protocol involves the definition of the variables, which in this language are called objects.
Each object, within an Agent, is identified by a unique sequence of numbers, separated by decimal points, as follows:

```
1.3.6.1.4.1.9839.2.1.2.45
```

Reading from left to right, each sequence can be graphically represented as a unique path of branches in a tree, in which the numbers on the left are closer to the root.

The SNMP standard has issued a series of rules for using the numbers in the tree, and has reserved some specific paths for common uses or specific organisations. The standard also allows as each number to be replaced by a name.
In the example, the path is rewritten with the standardised name, in brackets:

```
1(iso).3(organization).6(dod).1(internet).4(private).1(enterprises)
```

The path shown above is reserved for companies ("enterprises"), for specific developments.

Within that path, any company can apply for and obtain a specific identifier, and, within that branch, allocate the identifiers of the objects defined in its own products.
The "`enterprises`" identifier assigned to CAREL is `9839`. Therefore the path:

```
1.3.6.1.4.1.9839
```

identifies the CAREL space within each SNMP device.

Inside the CAREL space, the objects that correspond to the ***pCO*** variables and some typical objects of the ***pCOWeb*** have been added; for the description of this space, see 6.2 on page 34.
As the SNMP Agent is based on the Net-SNMP package, it also supports all the typical objects of that application. For information on this part of the SNMP tree in ***pCOWeb***, see the documents on Net-SNMP (www.net-snmp.sourceforge.net).

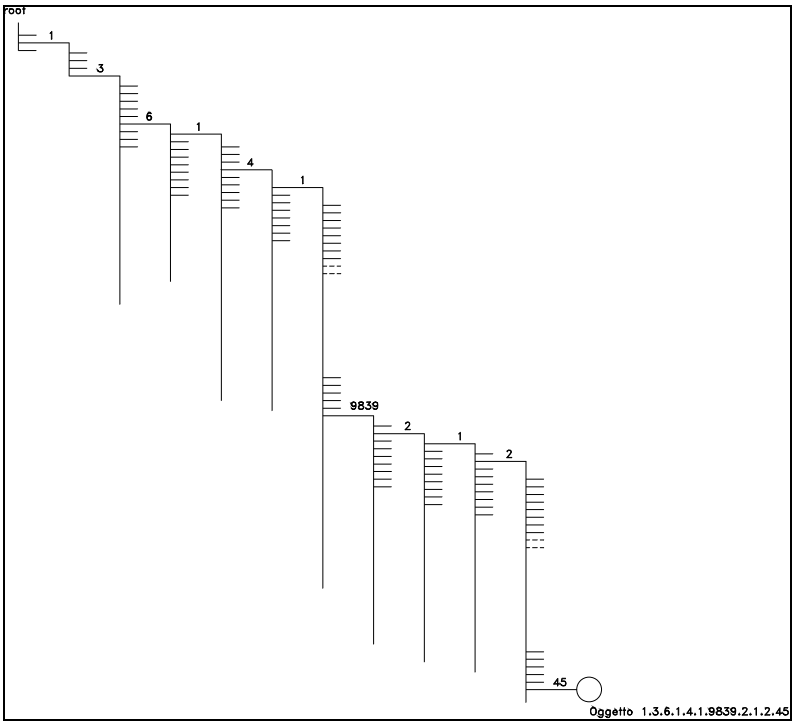


Figure 6.a - SNMP tree – example of a ***pCOWeb*** OID (Object IDentifier)

BASIC SNMP COMMANDS

The basic SNMP commands used by the NMS manage the variable read/write operations.

- snmpget / snmpset: read and write a specified SNMP object respectively;
- snmpgetnext: based on an initial SNMP object, reads the object implemented in the Agent that is next to the one specified, considering the lexicographical order of the path of the MIB tree;
- snmpwalk: based on an initial SNMP object, reads the entire part of the SNMP tree implemented in the device, next to the specified object (in reality, this command uses a sequence of snmpgetnext commands to request the "next" object each time).

COMMUNITY

SNMP v2 does not feature authentication (Username / Password). In version 2 the data is also exchanged without encryption.
Nonetheless, version 2c uses an identifier word (sent without encryption) called the "Community" that acts as a filter for the reception of the data or executing commands.
When an NMS needs to read the value of an object in an Agent, it must send a "ReadOnly Community" or a "ReadWrite Community" string that matches the identifier set in the Agent.
The same is true for the "ReadWrite Community" string when writing data.
Similarly, when an Agent sends an SNMP notification message (TRAP or INFORM), it specifies a "Trap Community"; if this does not coincide with the recipient "Trap Community", the message will be ignored.

6.2 THE *pCOWeb* SNMP TREE

The format of the CAREL ***pCOWeb*** OID (object identifier) is:

`1.3.6.1.4.1.9839(CAREL).A.B[.C.D]`

where:

A=1: *pCOWeb* properties

A.B

`1.1.0` Agent release (read-only), 2 for firmware A1.3.5 – B1.2.1, 3 for firmware A1.4.2 – B1.2.1

`1.2.0` Agent code (read-only), 2 ("***pCOWeb***")

A.B.C.D

`1.3.1.1.0` `ALARM FIRED` string (read / write)

`1.3.1.2.0` `ALARM REENTERED` string (read / write)

A=2: *pCO* properties

A.B.C.D

`2.0.10.1.0` communication status with ***pCO*** (read-only)
`0`=***pCO*** offline `1`=communication attempt `2`=***pCO*** online

`2.0.11.1.0` No. of communication attempts with ***pCO*** (read-only)
`0` when ok or, during `offline`, each 5 attempts

`2.1.t.i.0` ***pCO*** variables (read / write)
`t`: type of variable (`1`: Digital `2`: Analogue `3`: Integer)
`i`: variable index

NOTE 1 SNMP requires the use of the final zero when referring to scalar objects, that is, not tables or lists. All ***pCOWeb*** objects are scalar.
NOTE 2 Reading the ***pCO*** variables with indices that are not managed by the BIOS version used return insignificant values
NOTE 3 When a value is undefined, i.e. the summary web page shows "U", SNMP browser returns "-858993460".
NOTE 4 A variable is always read/write, however the value will be retained only if featured in the ***pCO*** application program.

Example:

`1.3.6.1.4.1.9839.2.1.`**`2.45.`**`0` type **2** ***pCO*** variable (analogue) with index **45**.

6.3 MIB FILE

The objects contained in an SNMP Agent are normally described using a document called the MIB FILE, the format of which is codified by SNMP.

The document must be written using the syntax specified by SNMP and must be uploaded to the NMS supervisor when this queries the Agent or receives the TRAP message: if the NMS contains the MIB file, it can use this an "interpreter" that, as well as containing the list of all objects available on the Agent, describes the properties, for example the description, the writability, the type, etc.…

As the MIB file contains the physical meaning of each object, it can only be created once the ***pCO*** application has been defined.
For this reason, upon request CAREL only provides the MIB FILES for the standard CAREL ***pCO*** applications, which can however be used to as the basis for creating a MIB file for custom applications; once created, numerous sites are available on the web for validating the MIB FILES.
One recommended site is: http://www.simpleweb.org/ietf/mibs/validate/.

6.4 BASIC SNMP CONFIGURATIONS FOR *pCOWeb*

The page shown in Figure 6.b below summarises the basic SNMP configurations for ***pCOWeb***.

SNMP System Configurations

- Read/Write Community – Read only Community: specify the strings in accordance with the system NMS (see 6.1 on page 33).
- Read/Write Network – Read only network: subnetworks that ***pCOWeb*** accepts Read/Write or Read only queries from; useful for restricting access only to some NMS; for unlimited access, leave "default".
- System contact: e-mail address to contact for the management of the installation; corresponds to the SNMP standard OID:
  `1.3.6.1.2.1.1.4.0`
- System name: installation identifier name; corresponds to the SNMP standard OID:
  `1.3.6.1.2.1.1.5.0`
- System location: physical location of the installation; corresponds to the SNMP standard OID:
  `1.3.6.1.2.1.1.6.0`

Figure 6.b - SNMP: basic configurations

SNMP Default traps configuration

- Default trap community: community enclosed with the TRAP notifications (see 4.3 on page 20) if the Host Community fields on the page shown in Figure 4.o on page 24 are empty; if this Community is also left empty, the Host TRAP notifications with the empty Community will NOT be sent.
- Hosts enabled: recipients enabled to send the "System traps" (see below) defined on the page shown in Figure 4.o on page 24.

System traps

pCO protocol failure: a TRAP will be generated whenever ***pCO-pCOWeb*** communication is interrupted or re-established; this TRAP can be set, as follows:

- Enabled: enabled to be sent.
- Trap OID: extended OID that describes the TRAP sent; the information will be sent without being processed by ***pCOWeb***.
- Acknowledge / Ack interval: enable, select the maximum number of repeats and the time interval for sending the TRAP until reception is acknowledged (similar to the Retry / Timeout fields corresponding to the TRAP messages generated by the events relating to the ***pCO*** variables, see Figure 4.t on page 26).

7 BACNET

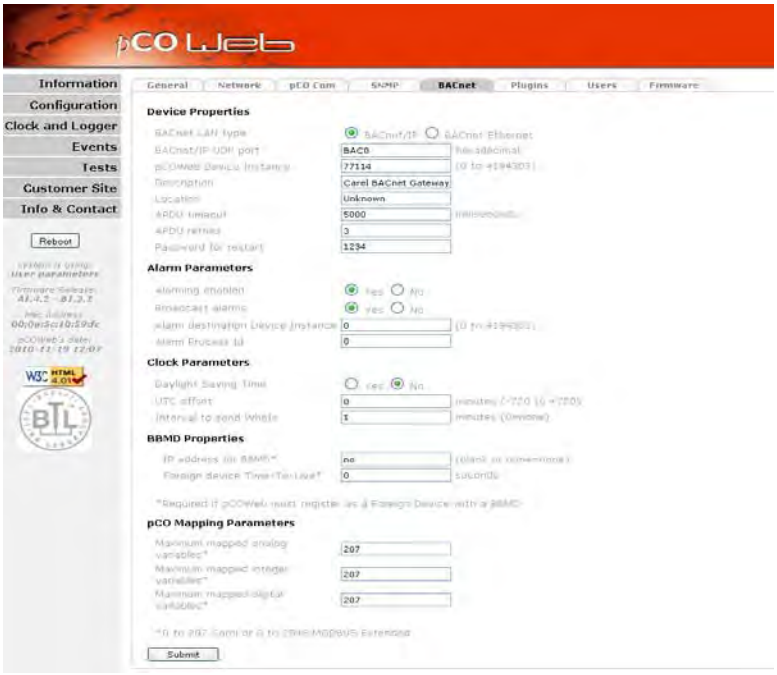


Figure 7.a - BACnet: basic configuration

pCOWeb can recognise queries sent by a supervisor that uses the BACnet protocol (Building Automation Control Networks), in the following two versions:

BACnet/IP (Addendum A/Annex J)
BACnet Ethernet ISO8802-2 over 8802-3

The two standards use the same physical means for carrying the data (Ethernet RJ-45 network) but differ as regards the different ways the data packets are encoded. If BACnet is used, during installation the proper version needs to be set to coincide with the type used by the supervisor.

A complete description of the BACnet protocol is not within the scope of this document. The system integrator, who is usually responsible for setting the various parameters, checking network communication and setting up the supervision system, should know how BACnet works.
The basic BACnet parameter configuration page is shown in Figure 7.a above; for advanced configuration, go to http://ksa.carel.com and download the ***BACset*** software and the ***BACset*** user guide; the main screen of this program is shown in Figure 7.b. All settings become active at next ***pCOWeb*** reboot.

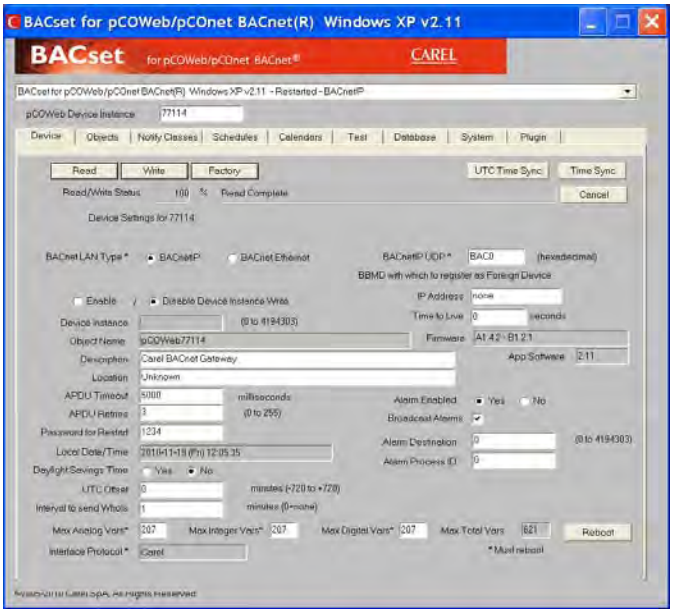


Figure 7.b - ***BACset*** software for the advanced ***pCOWeb*** BACnet configuration

BACset can be used to set all the properties of the BACnet objects supported by ***pCOWeb***, save them on ***pCOWeb*** or on the PC for later use when required.
The BACnet configuration is saved on ***pCOWeb*** in a number of files in the /usr/local/root/flash/etc/sysconfig directory in the same way as the other configurations.

BACNET PICS
BACnet is a scalable protocol, that is, it features a vast array of functions, some of which are obligatory, others whose implementation is left to the discretion of the manufacturer. For a product that uses BACnet, the document that analytically describes the type of functions implemented is extremely important.
This document is called PICS (Protocol Implementation Conformance Statement) and for the ***pCOWeb*** is available at http://ksa.carel.com.

NOTE1: The clock on the ***pCOWeb*** can be set using specific commands in the BACnet protocol. When using this method, it is recommended to disable synchronisation with the time on the ***pCO*** controller (see 5.1 on page 29).

NOTE2: The number of mapped variables in Figure 7.a has to be coherent with the number of variables sent in supervision. Moreover when ModBus Extended protocol is running is necessary that BACnet mapped variables are less or equals to the number of variables sent in supervision, see Figure 9.l on page 45 and in every case they could not be more than 2048 for each type (digital, analogue or integer)

7.1 BACnet/Carel Mapping

For each ***pCOWeb*** that interfaces to a Carel controller using a standard Carel ***pCO*** connection there can be a maximum of 670 BACnet objects, which includes the Device Object. Of the 670 objects, there are always 16 Notification Class objects, 16 Calendar objects, 16 Schedule objects and there can be up to 207 Analogue Values, 207 Analogue Values or Multi-state Values and 207 Binary Values. The number of each type can be configured prior to use. From the factory all 621 Analogue Value and Binary Value objects are mapped. Object_Identifiers are assigned according to the following table:

Table 7.a – BACnet/Carel mapping

Object Type	Instance Range	*pCO* Mapping
Analogue Value	1-207	A001-A207
Analogue Value or Multi-state Value	1001-1207	I001-I207
Binary Value	1-207	D001-D207

From the factory, I001-I2007 are mapped as Analogue Value 1001-Analog Value 1207. Each of these may be reprogrammed as Multi-state Values on an individual basis, by writing to the Object_Identifier property. BACnet objects which are mapped but for which there are no corresponding physical ***pCO*** points return an unreliable-other (7) value for the Reliability property of the corresponding object and an undetermined value for the Present_Value property (usually=0 or 0.0).
The Object_Name properties for all the objects, except the Notification Class objects are writable and can be up to 32 characters in length. By default the Analogue Value/Multi-state Value and Binary Value objects are named A001-A207, I001-I207 and D001-D207 corresponding to their ***pCO*** mapping.

7.2 BACnet/MODBUS Mapping

For each ***pCOWeb*** that interfaces to a Carel controller using a Carel MODBUS Extended connection there can be a maximum of 6193 BACnet objects, which includes the Device Object. Of the 6193 objects, there are always 16 Notification Class objects, 16 Calendar objects, 16 Schedule objects and there can be up to 2048 Analogue Values, 2048 Analogue Values or Multi-state Values and 2048 Binary Values. The number of each type can be configured prior to use. Object_Identifiers are assigned according to the following table:

Table 7.b – BACnet/MODBUS mapping

Object Type	Instance Range	MODBUS Mapping
Analoghe Value	100001-102048	A0001-A2048
Analogue Value or Multi-state Value	200001-202048	I0001-I2048
Binary Value	100001-102048	D0001-D2048

By default I0001-I2048 are mapped as Analogue Value 100001-Analog Value 102048. Each of these may be reprogrammed as Multi-state Values on an individual basis, by writing to the Object_Identifier property. BACnet objects which are mapped but for which there are no corresponding physical MODBUS points return an unreliable-other (7) value for the Reliability property of the corresponding object and an undetermined value for the Present_Value property (usually=0 or 0.0).
The Object_Name properties for all the objects, except the Notification Class objects are writable and can be up to 32 characters in length. By default the Analogue Value/Multi-state Value and Binary Value objects are named A0001-A2048, I0001-I2048 and D0001-D2048 corresponding to their MODBUS mapping.

7.3 BACnet features

7.3.1 Alarming

BACnet Intrinsic Alarming is supported for the Analogue Values (Out_of_Range event types), Binary Values (Change_of_State event types) and Multi-state Values (Change_of_State event types). All optional properties related to Intrinsic Alarming are included and writable where appropriate. Use of Notification Class Objects may be optionally bypassed. In this scheme, if an object's Notification_Class property is set to 0, three standard BACnet properties that are normally included in Notification Class objects, specifically Ack_Required, Issue_Confirmed_Notifications and Priority, are included for each Analogue Value, Binary Value and Multi-state Value object and take the place of the Notification Class. Although these properties are not standard or optional for Analogue Value, Binary Value and Multi-state Value objects, they are treated in a standard BACnet way. Also in this scheme, two properties, Process_Identifier and Recipient, are included for the Device Object and take the place of the Notification Class Object. Although these properties are not standard or optional for Device objects, they are treated in a standard BACnet way. The Recipient property is limited to the Object_Identifier of a Device object. If the Notification_Class for an object is 0, it will be reported as 1 in alarm events.
For all alarms, the default Message Text that is always included by default is of the form:

"nnnnnn (Binary Value xxx) ChangeOfState v"
"nnnnnn (Analogue Value xxx) OutOfRange v"

Where: nnnnn is the Object_Name, xxx is the object instance, ChangeOfState or OutOfRange is the Event_Type, v is the to-state (i.e. Normal, Offnormal, Fault, High Limit or Low Limit)
The Message Text can be customized by replacing any of the Event_Type text and/or to-state text with customized text up to 32 characters long each. In addition, the Object_Name property can be replaced with the Description property. Alarm Message Text can be customized on an object by object basis by writing to the following proprietary properties for the object:

Table 7.c – Alarm Messages Customization

Property name	EnumeratedProperty value	Datatype	Value
PROP_ALARMCUSTOM	3000	Boolean	True=Alarms are customized for this object False=Use the default Message Text format
PROP_ALARMPROPERTY	3001	Character string	Object_Name or Description
PROP_ALARMCOS	3002	Character string	String (up to 32 characters) to replace Event_Type ChangeofState text
PROP_ALARMOOR	3003	Character string	String (up to 32 characters) to replace Event_Type OutofRange text
PROP_ALARMNORMAL	3004	Character string	String (up to 32 characters) to replace to-state Normal text
PROP_ALARMFAULT	3005	Character string	String (up to 32 characters) to replace to-state Fault text
PROP_ALARMOFFNORMAL	3006	Character string	String (up to 32 characters) to replace to-state Offnormal text
PROP_ALARMHILIM	3007	Character string	String (up to 32 characters) to replace to-state High Limit text
PROP_ALARMLOWLIM	3008	Character string	String (up to 32 characters) to replace to-state Low Limit text

7.3.2 COV Subscriptions

The DS-COV-B and/or DS-COVP-B BIBB support a maximum 250 simultaneous subscriptions. The following properties are eligible for COV reporting using the SubscribeCOVProperty Service:

- Present_Value
- Status_Flags
- Reliability
- Event_State
- Out_of_Service

7.3.3 Commandability

BACnet commandability is supported for the Analogue Values, Binary Values and Multi-state Values on an individual object basis. In addition to being optionally selected as commandable, each of these individual objects may be programmed as read-only or writable but not commandable. This condition is controlled by writing to the proprietary property 1014 for the appropriate object. For the proprietary property 1014, a value of 0 indicates read-only, 1 indicates writable but not commandable and 2 indicates commandable. The Priority_Array and Relinquish_Default properties are present only if the proprietary property 1014 is 2 for an individual object. By default, the proprietary property 1014 has a value of 1 (writable but not commandable) for backward compatibility with older versions of the ***pCOWeb*** when delivered from the factory.

7.3.4 Schedules

BACnet schedules are supported and made visible through the 16 Schedule and 16 Calendar objects. For each Calendar object, there can be from 0 to 16 dates in the Date_List. For each Schedule, both the Weekly_Schedule and Exception_Schedule properties are supported. Each object's Exception_Schedule list can have from 0 to 4 entries. Each Weekly_Schedule and each Exception_Schedule can have from 0 to 6 Time_Values. Each object's List_Of_Object_Property_References property can have from 0 to 64 entries. External objects are not supported. Following a restart, schedules are not executed until the ***pCOWeb*** clock has been set using a TimeSynchronization or UTCTimeSynchronization. Schedules are checked approximately every 10 seconds for a minute rollover, so scheduled values may not change until 10 to 15 seconds after the minute turns over.
Carel

7.3.5 Default Factory Settings

Default values for the ***pCOWeb*** are as follows:

Table 7.d – Factory Settings

Property	Default Value
Device Instance	77000
LAN type	1=BACnet/IP
BACnet/IP UDP	BAC0 (hexadecimal) or 47808 (decimal)
Alarming Enabled	False
Number of Analogue Variables Mapped	207
Number of Integers Variables Mapped	207
Number of Digitals Variables Mapped	207
IP Address for BBMD	None
Foreign Device Time-To-Live	0 (seconds)

8 MODBUS OVER TCP/IP

8.1 OVERVIEW

MODBus is a communication protocol developed by Modicon systems. In simple terms, it is a way of sending information between electronic devices. The device requesting the information is called the MODBus Master and the devices supplying information are MODBus Slaves. In a standard Modbus network, there is one Master and up to 247 Slaves, each with a unique Slave Address from 1 to 247. The Master can also write information to the Slaves.

8.2 BASIC DESCRIPTION

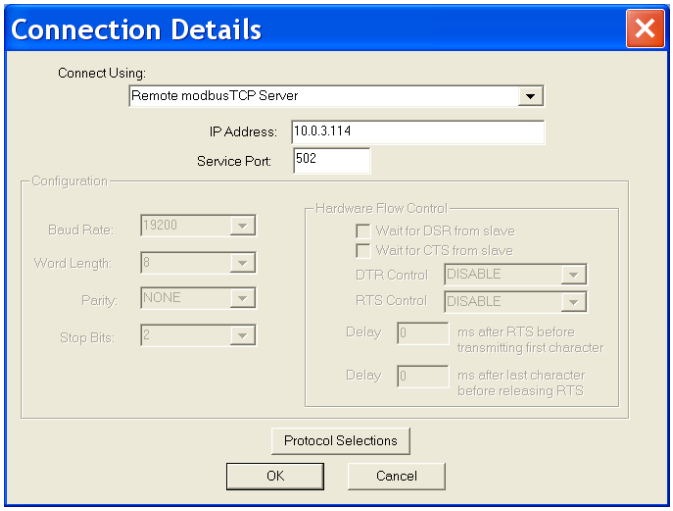


Figure 8.a – Remote connection example

To get your modbus device working, these are the basic things you need to know:

- Port 502 TCP (default by protocol, cannot be changed)
- Address format: Decimal (digital number 1 = coil 1)
- Variable bindings (version <=1.3.5)
  - Digitals: Coils 1-207
  - Analogues: Registers 1-207
  - Integers: Registers 5001-5207
- The ranges of variables for version 1.4.2 (when ***pCO*** communicates using Modbus extended protocol) have been extended as explained here below:
  - Digital variables: coils from 1 up to 2048
  - Analogue variables: registers from 1 up to 5000
  - Integer variables: registers from 5001 to 10000
- Variable types: Signed Integers (mandatory in some software to correctly read/write the variables)

8.3 COMMANDS SUPPORTED

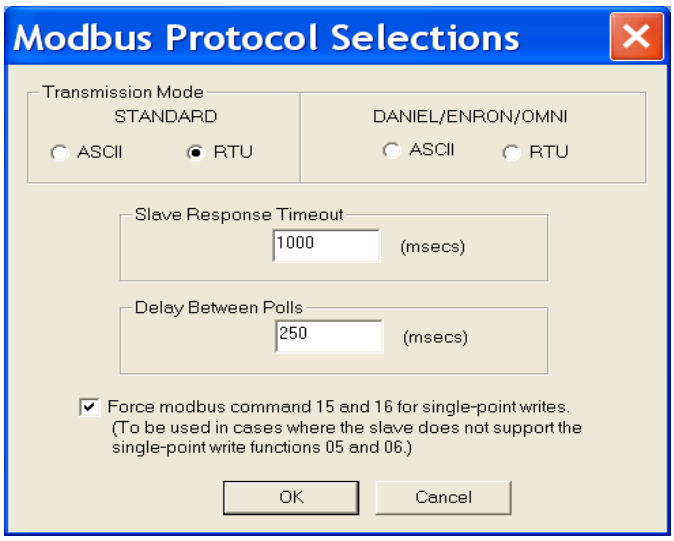


Figure 9.a – Protocol settings

These are the only commands supported by the ***pCOWeb*** in modbus TCP/IP

- 01: Read coils
- 03: Read holding registers
- 05: Write single coil
- 06: Write single registry
- 15: Write multiple coils
- 16: Write multiple registers

9 BASIC CONFIGURATION AND AUXILIARY FUNCTIONS

pCOWeb features a number of HTML pages for setting the functions. These pages are accessible from a PC.

Some settings will be active as soon as they are confirmed, while others, where indicated, require the ***pCOWeb*** to be rebooted.

9.1 BACKING UP THE CONFIGURATION OF THE *pCOWeb*

All the settings are saved in special text or binary file in the /usr/local/root/flash/etc/sysconfig directory; to run a backup, simply connect to ***pCOWeb*** via FTP (see 4.2 on page 17) and copy all the files to a PC.

IMPORTANT This directory only contains the parameter settings. To completely backup all the ***pCOWeb*** custom settings, the custom HTML pages, the contents of the Plugin directories, the log file directory and in general the entire contents of the /usr/local/root/flash/http/ directory and any other directories created manually must be copied.

NOTE1: To completely back up the ***pCOWeb*** configuration after copying the files to the new ***pCOWeb*** a reboot is required
NOTE2: To copy a back up from a ***pCOWeb*** to an other one is necessary that both ***pCOWeb*** have the same firmware version

In addition, a function is available that deletes all the changes made by the user (settings or files added) and returns ***pCOWeb*** to its default status (see 9.3 on page 43).

9.2 ACCESSING THE CONFIGURATION PAGE

To access the configuration pages:

1. connect to ***pCOWeb*** using a PC (chapter 3 on page 10);
2. start Internet Explorer, opening the main page (Figure 3.i on page 13);
   the main page may have been customised and consequently will be different from Figure 3.i.
3. If the page appears as shown in Figure 3.i, click the "Go to Administrator area" link. An authentication dialogue box similar to the one shown in Figure 9.a below will be opened. Continue as described in paragraph 9.2.1 on page 41.
   NOTE The authentication dialogue box is enabled by default, however it may be been disabled by the user (for the disabling procedure see paragraph 9.2.1 on page 41); in this case the dialogue box will not open; continue as described in paragraph 9.2.2 on page 41.
4. If, on the other hand, the main page is different because it has been customised, the "Go to Administrator area" link may not be available.
   In this case, open the copy of the main page available in the read-only memory, typing the following address in Internet Explorer:

   `http://<pCOWeb IP address>/defindex.html`

   The IP address entered refers to the ***pCOWeb*** being configured: when using the button at ***pCOWeb*** starting (procedure described in paragraph 3.1 on page 10), the IP address will be 172.16.0.1.

   In the more general situation of connection via the network, there may be more than one ***pCOWeb*** connected; consequently, the IP address of the ***pCOWeb*** in question needs to be known; if necessary, contact the network administrator. If the connection fails, see paragraph 3.2 on page 14.
   If there are no problems, an authentication dialogue box similar to the one shown in Figure 9.a below will be opened. Continue as described in paragraph 9.2.1 on page 41.

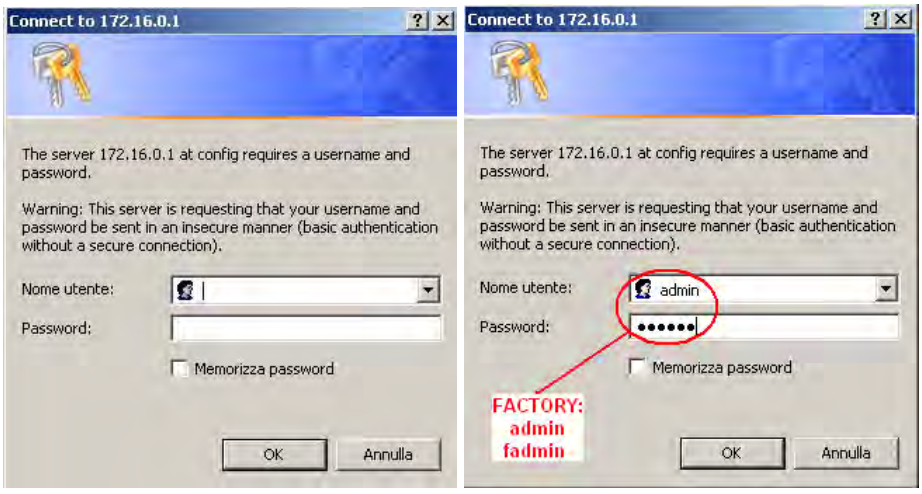


Figure 9.a – Authentication dialogue box for accessing the Administrator area

NOTE The authentication dialogue box is enabled by default, however it may be been disabled by the user (for the disabling procedure see 9.2.1 on page 41); in this case the dialogue box will not open; continue as described in paragraph 9.2.2 on page 41.

9.2.1 Authentication dialogue box for accessing the Administrator area

Following the previous points, an authentication dialogue box is displayed on the PC screen (Figure 9.a on page 40); complete the fields with the access information, then select OK.
The default settings are:

Username: **admin** Password: **fadmin**

IMPORTANT

- These settings are case sensitive!
- The login data can be customised by the user (see 9.7.1 on page 49); the data to be entered then are always the custom settings, regardless of whether ***pCOWeb*** is started while pressing the button.
- The authentication dialogue box can be disabled or the default restored; proceed as follows:
  1. access the user memory via FTP (see 4.2 on page 17);
  2. (to disable the dialogue box) delete the file:

     /usr/local/root/flash/http/admin/.htpasswd

  3. (or alternatively, to restore the default data, "admin/fadmin") copy the file:

     /usr/local/root/defadmin/.htpasswd

     to the following directory:

     /usr/local/root/flash/http/admin/

     overwriting the current .htpasswd file;
     for further information see 9.7.1 on page 49.
- Internet Explorer (as well as other applications) temporarily saves the information used to access a site (this data is normally deleted when closing Internet Explorer). In this way, when next attempting to access the configuration page without have closed Internet Explorer, the data may not need to be entered again: Internet Explorer will automatically send the previously acquired data.

NOTE 1 Unlike index.html and the other ***pCOWeb*** pages that the user can customise or load to adapt ***pCOWeb*** to the specific application, all the configuration pages described below CANNOT BE DELETED OR CUSTOMISED by the user; in fact, they are contained in the ***pCOWeb*** firmware and can only be modified following firmware updates published by CAREL and downloaded to the product. Nonetheless, a copy of the pages can be made and edited as desired. The directory that contains these files is /usr/local/root/config/.

NOTE 2 As already mentioned at the beginning of the manual, the following description refers to the configuration page relating to ***pCOWeb*** firmware version A1.4.2 - B1.2.1. To check the ***pCOWeb*** firmware version, see paragraph 9.2.2 below.

9.2.2 Configuration - Starting page: Information (*pCOWeb* Summary page)

If the login data entered are correct the following page will be displayed, as shown in Figure 9.b. The section in the middle (Body) is the "information page" and can be refreshed by clicking Information (the "Menu" section is not refreshed).

pCOWeb information page

The "information page" shows a table that represents a "snapshot" of the values of all the ***pCO*** variables. The table is divided into three parts: digital, analogue, integer variables. The indices of the variables are shown on the left.
The values displayed on the screen are NOT updated automatically, but only whenever the page is refreshed or opened again (see Figure 9.c on page 42).

The letter "U" (Undefined) shown for some variables means that the value could not be acquired for that variable from the ***pCO***.

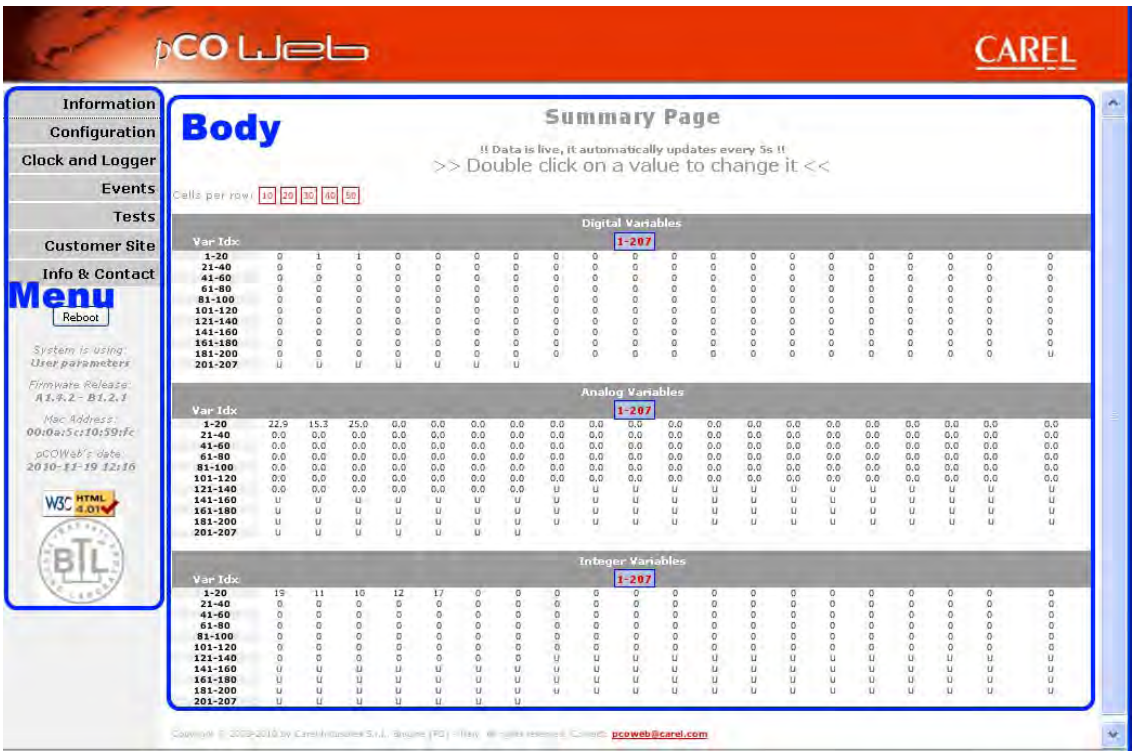


Figure 9.b – Configuration – starting page: Summary

NOTE 1 The older versions of the *pCO* operating systems (BIOS) do not manage the entire range of variables. In this case "U" is shown in the corresponding locations.

NOTE 2 If all the variables are shown with a "U" there is a problem in communication with the *pCO*. In this case, check the *pCO*-*pCOWeb* communication settings (see 9.5 on page 44)

NOTE 3 If some variables show a "U" among other valid values, check that the application on the *pCO* uses the supervision atoms correctly; the most frequent causes are: variables written too frequently, or two different variables that use the same supervisor index; see the technical documents on the *pCO* application development environment.

NOTE 4 It is possible to set a variable directly from the Summary page, just double clicking on the interested variable a pop-up window (see Figure 9.aa) will be displayed as described in 9.8 on page 51 where the reading/writing function is described

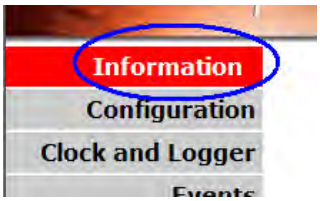


Figure 9.c – Click to refresh the Body section shown in Figure 9.b

For further information on exchanging the values of the variables between the *pCO* and *pCOWeb* see APPENDIX C on page 65.

MENU AREA

The "Menu" area is always the same for all the main configuration pages.
It contains:

- A - buttons for opening the main configuration page.
- B - "Reboot" button; when selecting this, *pCOWeb* will be rebooted without requesting confirmation. This is required in some phases of the configuration.
- C – System is using (see 3.1.2 on page 12):
  - *Factory Parameters*: *pCOWeb* is using the Bootswitch parameters (Figure 3.g on page 12).
  - *User Parameters*: *pCOWeb* is using the parameters set by the user.
- D – displays the *pCOWeb* firmware version:
  A: Applications B: Bios.
  IMPORTANT: do not confuse this with the firmware version ("BIOS") of the *pCO*!
- E – displays the *pCOWeb* MAC address (see 2.2 on page 9).
- F – displays the *pCOWeb* date and time at the moment of the last update requested for the page displayed from the PC (for example with F5).
- G – symbol that the HTML page conforms to the HTML 4.01 standard.

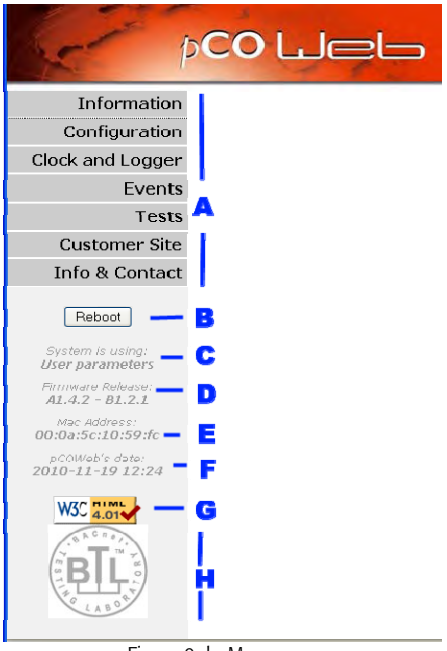


Figure 9.d - Menu area

9.2.3 Displaying the main page

When a configuration page is displayed, the main page can be recalled by clicking "Customer Site"; a new window will be opened with the main page, index.html where featured, customised via FTP by the user when creating the site in relation to the *pCO* application program used; if not, the page displayed will be the default (see Figure 3.i on page 13).

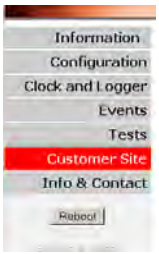


Figure 9.e - Displaying the main page

9.2.4 Useful contacts

Clicking "Info & Contact" displays the following page with useful contacts at CAREL.

Figure 9.f – Useful contacts at CAREL

9.3 GENERAL INFO AND RESTORING THE DEFAULT SITUATION

The page shown in Figure 9.g can be opened to display basic information on the ***pCOWeb*** and restore the ***pCOWeb*** default situation.

- View used/free disk space: shows a summary of the space occupied in the user memory (Figure 9.h below);
  - Size / Used / Avail / Use%: overall size / used / available / % of the user memory space.

  NOTE The space in non-volatile memory reserved for the user is managed in compressed format; the information saved in this space (configurations, custom HTML pages, log files, …) will take up less space than the native size of the file; the values shown in Figure 9.h refer physical space occupied by the already compressed information.

- View factory bootswitch parameters: shows a summary of the factory settings that ***pCOWeb*** will use if rebooted with the button pressed (see 3.1.2 on page 12);
  - DEFIP / DEFNETM: IP address / subnet mask;
  - PROOT / PHTTP / PCAREL / PGUEST: password respectively for the "root" / "httpadmin" / "carel" / "guest" Usernames in the operating system running on ***pCOWeb*** (see 9.7.2 on page 50).
- View network configuration: shows a summary of the network settings that ***pCOWeb*** is using, divided between ETH0 / ETH0:1 / ETH0:2 / ETH0:3 (these can only be displayed if the corresponding logical interface is being used, see 9.4 below) / DNS SERVERS / GATEWAY; especially useful when operating with DHCP: shows the effective information acquired from the server and currently used.
- Fix HTML pages and CGIs rights: in the previous firmware versions, once an HTML page or CGI executable was loaded via FTP, this process needed to be run to automatically attribute the transferred file the rights needed to be recognised by the ***pCOWeb*** operating system; currently this is only necessary for executable CGI files.
- Delete all user files and settings: deletes all the settings and files added by the user, and restores the default situation from a copy located in the read-only area of the non-volatile memory.

Figure 9.g - General info and restore default situation

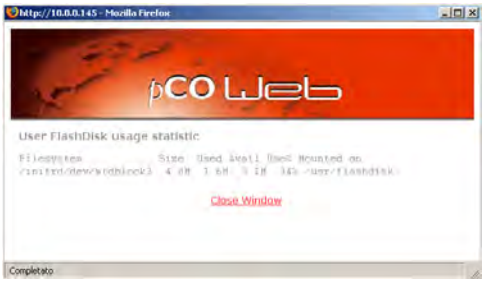


Figure 9.h - Summary of the space available in the user memory

The following options are available (Figure 9.i):

- Yes, but keep network configuration: do not reset the default network configuration; useful for continuing to connect to ***pCOWeb*** using the same IP address, recommended if ***pCOWeb*** cannot be rebooted while holding the button;
- Yes, delete all: also reset the default network configuration ("DHCP"); in this case, if this setting was not previously used, to contact ***pCOWeb*** the button will need to be held when rebooting to activate the default IP address: 172.16.0.1 (see 3.1.1 on page 10 and 3.1.2 on page 12).
- Regardless of the above choice, after confirming with the Submit button, a report is shown with a request to reboot the ***pCOWeb***; the reboot is required: some settings will only be reset after reboot; until that time, the HTML pages will show the incorrect setting of some parameters (Example: baud rate = 300, see 9.5 on page 44), and these will only be correct after the reboot.

Figure 9.i – Restoring ***pCOWeb*** to the default situation

IMPORTANT If the authentication data to access the configuration page set by the user is different from the default values (admin / fadmin), ***pCOWeb*** will request authentication (admin / fadmin) for the configuration page immediately, starting from the page in response to the deletion of the user files.

9.4 SETTING THE NETWORK COMMUNICATION PARAMETERS

For ***pCOWeb*** to communicate with the data network it is installed in, a number of network communication parameters need to be correctly set.
The assistance of the network administrator is required to establish if ***pCOWeb*** can be connected, and to understand the essential data relating to the installation.
To better understand the meaning of the procedure for setting these parameters, see APPENDIX A on page 61 and APPENDIX B on page 62. The following operations are in any case possible even without having read the appendices.

First of all, it must be established whether or not the network uses automatic address setting (DHCP); ask the network administrator.

- **network with DHCP:** in this case ***pCOWeb*** is already ready in the factory configuration and no operation is required. The network administrator will require documents that show the physical positions of the various ***pCOWeb*** devices and the corresponding MAC addresses (see 2.2 on page 9).
  To check the setting of DHCP mode on the ***pCOWeb***, follow the same procedure as described below for the "network without DHCP".
- **network without DHCP:** in this case, before connecting ***pCOWeb*** to the network, the DHCP system, active by default, must be disabled, and a different specific address entered for each ***pCOWeb***. These procedures are described below.

9.4.1 Network configuration

This is the most important configuration. Without this, in fact, ***pCOWeb*** will not be able to communicate correctly with the Ethernet network.

Starting by the main configuration page – Information (Figure 9.b on page 41), click "Configuration", then click the "Network" tab (Figure 9.j).

The page shown in Figure 9.k below is displayed.
This page is used to set the following fundamental user network parameters:

IP address
NetMask

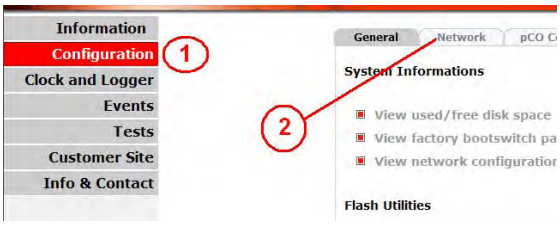


Figure 9.j – Opening the network configuration page

NOTE 1 The values set will be saved to non-volatile memory only when selecting the "Submit" button.

IMPORTANT The values set will only be used when ***pCOWeb*** is next rebooted.

NOTE 2 If ***pCOWeb*** is started when the button is pressed (paragraph 3.1.2 on page 12), the "Bootswitch" settings will be used instead of these values (see Figure 3.g on page 12).

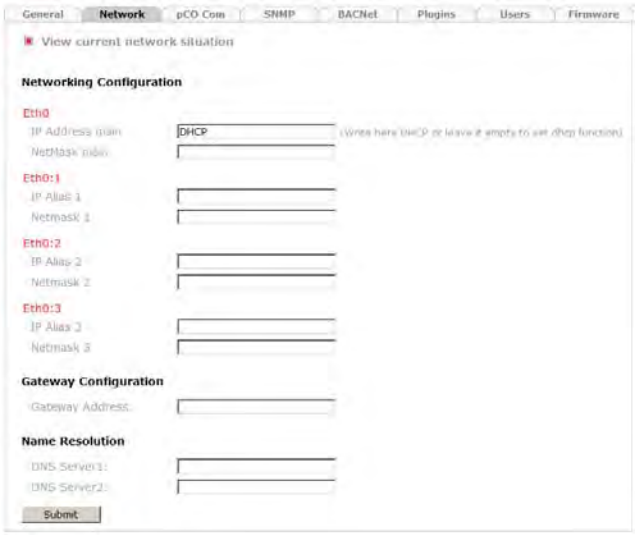


Figure 9.k - Configuration - Network

SETTINGS AVAILABLE:

- Eth0 (required)
  - For networks with automatic address setting (DHCP server), write DHCP or dhcp or alternatively leave empty. The NetMask, if set, will not be used.
  - For networks that do not use automatic address setting (DHCP), enter the IP address and the NetMask established for this ***pCOWeb*** by the network administrator.
    Make sure to use the correct format; examples:
    10.0.3.114
    255.255.0.0
    without spaces.

  APPENDIX A on page 61 and APPENDIX B on page 62 provide further information on the meaning of the parameters.
- Eth0:1 – Eth0:2 – Eth0:3 (optional): irrespective of whether the DHCP method is used, up to a maximum of 3 other IP addresses can be set that ***pCOWeb*** will respond to; entering "DHCP" in these fields will produce no result: Eth0 is the only parameter enabled for the activation of DHCP.
  IMPORTANT If Eth0 = DHCP, the addresses Eth0:1-Eth0:2-Eth0:3 will only be operational when the server has assigned a dynamic IP and only during its period of validity (called "lease").
- Gateway Address (not used with DHCP operation): used in cases where notifications are to be sent in response to events (see 4.3 on page 20).
  Set the IP address for the Gateway in the subnetwork that the ***pCOWeb*** Eth0 IP address is part of.
  NOTE The gateway configuration is not required if the address of ***pCOWeb*** and the receiving network devices belong to the same subnetwork. For further information see APPENDIX B on page 62.
- Name resolution (DNS): required only if the recipients of the notifications (e-mail / TRAP / FTP PUSH) will be specified not using IP addresses but rather names (for example: "working_pc.net").
  Set the IP address of at least one server with Domain Name Server functions.
  For further information see APPENDIX B on page 62.

9.5 SETTINGS RELATING TO *pCOWeb - pCO* COMMUNICATION

INFORMATION
pCOWeb-pCO communication occurs by default using the CAREL supervisor protocol; starting from version A142, ***pCOWeb*** is able to communicate with ***pCO**** controller using protocol Modbus Extended (BMS_PROTOCOL = 30 on 1tool and SERIAL1_PROTOCOL = 30 on Easytool, BIOS > 4.00). Simple Modbus (BMS_PROTOCOL = 3) is not supported.
This enables ***pCOWeb*** to exchange up to 12048 variables (5000 analogue, 5000 integer and 2048 digital variables) with the ***pCO*** application, which has to be specifically designed to use the new extended range of variables to the monitoring system.
To enable the modbus extended protocol ***pCOWeb*** has to be told:

- The address used by the ***pCO*** in the serial port BMS, this is identified by the system variable BMS_ADDRESS in the ***pCO*** controller and in the ***pCO*** application, default is 1;
- The highest index for each type of variable, default is 2048 for digital variables, 5000 for analogue and integer variables; it is strongly advised to limit the number of variables read by ***pCOWeb*** to those which are actually managed by the software to improve drastically the performances of the card. All the configuration web pages of ***pCOWeb*** will automatically adapt to the new limits

IMPORTANT: The number of variables read by ***pCOWeb*** must be equal or superior to the number of variables mapped in the BACnet configuration (see Figure 7.a) on page 36
IMPORTANT: **if the "CAREL supervisor" protocol or "ModBus Extended" protocol has not been selected on the *pCO*, communication will not be possible**.

NOTE ***pCOWeb***, unlike other devices that communicate using the CAREL supervisor protocol, does not require the setting of the supervisor serial address (IDENT) while using ModBus Extended it is required.

The fixed settings for the ***pCOWeb*** serial port are:

- Frame: 8 bits;
- Parity bit: none;
- Stop bits: 2.

The speed can be set between 300-600-1200-2400-4800-9600-19200(default)-38400-57600-115200 bit/s and must coincide with the baud rate of the ***pCO*** supervisor port, selected on a special screen on the application interface, the protocol can be set between Carel and Modbus Extended as we can see in Figure below. The default values for ***pCOWeb*** are 19200 baud and Carel protocol, and can be changed as required.

IMPORTANT

- The set value will only be used when ***pCOWeb*** is next rebooted.
- If the speed set for ***pCOWeb*** is different from the value set for the ***pCO***, communication will not be possible.

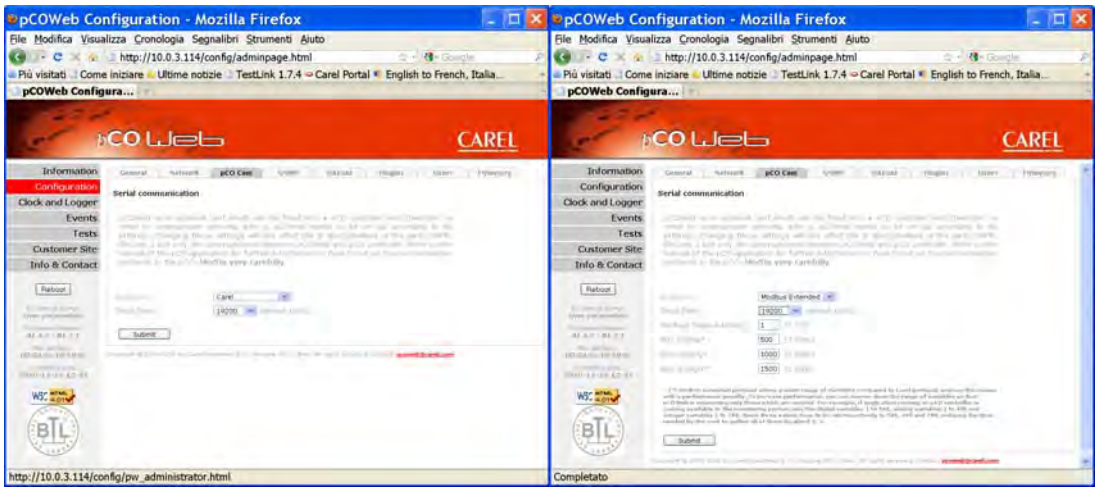


Figure 9.I – Setting the ***pCOWeb***-***pCO*** communication speed

9.5.1 Extended range ad functionalities

All the protocols and some of the functionalities of ***pCOWeb*** have been extended to cover the entire range of variables made available by modbus extended. ***pCOWeb*** returns "Undefined" if requesting a variable which is not available on the database of modbus extended (***pCO*** application doesn't support the specific index) and "Error" if requesting a variables which is out of the boundaries defined in the configuration page "***pCO*** Com".

- Web Pages

All the web pages designed for ***pCOWeb*** <= A135 are fully compatible with version A142 and modbus extended. The ranges of variables have been extended as explained here below:

- •Read/write digital variables
  - -<%var(0,1,1,0,1)%> up to <%var(0,1,2048,0,1)%>
- •Read/write analogue variables
  - -<%var(0,2,1,-3276.8,3276.7)%> up to <%var(0,2,5000,-3276.8,3276.7)%>
- •Read/write integer variables
  - -<%var(0,3,1,-32768,32767)%> up to <%var(0,3,5000,-32768,32767)%>

- SNMP

Fully compatible with version <=A135; Trap event notifications are available only for the first 207 variables (range has not been extended from A135). The ranges of variables have been extended as explained here below:

- •Digital variables: OID from 1.3.6.1.4.1.9839.2.1.1.1.0 up to 1.3.6.1.4.1.9839.2.1.1.2048.0
- •Analogue variables: OID from 1.3.6.1.4.1.9839.2.1.2.1.0 up to 1.3.6.1.4.1.9839.2.1.2.5000.0
- •Integer variables: OID from 1.3.6.1.4.1.9839.2.1.3.1.0 up to 1.3.6.1.4.1.9839.2.1.3.5000.0

- Modbus TCP/IP

Fully compatible with version <=A135. The ranges of variables have been extended as explained here below:

- •Digital variables: coils from 1 up to 2048
- •Analogue variables: registers from 1 up to 5000
- •Integer variables: registers from 5001 to 10000

- BACnet

Fully compatible with version <=A135 if using Carel protocol, with modbus extended the mapping of the variables into BACnet objects changes completely to allow a larger number of variables.

- •Digital variables: Binary value objects from 1 up to 207 (Carel Protocol), 100001 up to 102048 (Modbus extended protocol)
- •Analogue variables: Analogue value objects from 1 up to 207 (Carel Protocol), 100001 up to 102048 (Modbus extended protocol)
- •Integer variables: Analogue/Multistate value objects from 1001 up to 1207 (Carel Protocol), 200001 up to 202048 (Modbus extended protocol)

- Clock Synchronization

Variables used to synchronize the clock have been extended to cover the entire range of integer variables.

- Logger

With the new version of firmware, logger has been deeply improved; anyway only variables with index from 1 to 207 can be logged and graphed by ***pCOWeb***.

9.6 PLUGINS

Plugins are auxiliary applications that can be installed on ***pCOWeb*** and allow custom functions to be added easily and automatically.
There are two types of Plugins:

- distributed by CAREL, available at http://ksa.carel.com.
- custom, created for a specific application.

To be able to create a Plugin, good knowledge is required of the GNU/Linux operating system and bash scripting and, to create compiled binary files, these need to adhere to certain requirements. The rules for creating Plugins are summarised in APPENDIX I on page 84.

9.6.1 Installing a Plugin

Access the ***pCOWeb*** main configuration page (see 9.2 on page 40), then open the Plugin page (Figure 9.m).

➢ Plugins found: lists the Plugins currently installed that have at least one HTML configuration page. In the example shown in Figure 9.m there are no Plugins installed.

Figure 9.m - Installing / uninstalling the Plugins

PROCEDURE FOR INSTALLING THE CAREL "NTP" PLUGIN

The "NTP" Plugin (Network Time Protocol) is used to synchronise the time on ***pCOWeb*** with an Internet time server.

1. Download the "NTP" Plugin from http://ksa.carel.com; if the Plugin to be installed is contained in a zip file, first unzip it using any zip file decompression program (for example, "Filzip" freeware, http://www.filzip.com), so as to create a directory called "install-plug-xxx" (xxx is ignored by ***pCOWeb***, and is only used to identify the function of the Plugin).

2. Access the ***pCOWeb*** user memory (see 4.2 on page 17) with the "httpadmin" Username, and display the contents of the "/usr/local/root/flash/http/" directory.

3. Copy the "install-plug-xxx" directory and all its contents to the "/usr/local/root/flash/http/" directory on ***pCOWeb***; then press F5 to refresh the page shown in Figure 9.m above, which will then appear as shown in Figure 9.n.
   The page includes the new section called "Install/uninstall plugins" that is used to choose which Plugins to install from those that have already been copied, or which to uninstall from those already installed. This section is only displayed if at least one Plugin is installed or if at least one Plugin to be installed has been copied.

Figure 9.n – Plugin page: ***pCOWeb*** has found one or more Plugins to be installed

4. Click "Install/uninstall plugins"; a page as shown in Figure 9.o will be displayed.
   The "Install plugins" section will show 'Install plugin "Network_Time_Protocol"', the text displayed is taken from the pluginname file in the Plugin installation directory copied from the computer to ***pCOWeb***.
   If the "/usr/local/root/flash/http/" directory contains more than one installation directory, link will be displayed for each Plugin.

Figure 9.o – Install/Uninstall Plugin page: list of Plugins ready to be installed

5. To start the installation procedure, click 'Install plugin "Network_Time_Protocol" '.
   IMPORTANT Confirmation is not requested before starting the procedure, and the installation, if succeeded, will delete the "install-plug-xxx" directory on ***pCOWeb*** copied from the computer.

Figure 9.p – "NTP" Plugin installation report

At the end of the installation procedure (which may last some tens of seconds), a report is displayed describing the outcome of the installation of each file; if the installation was not successful, the Plugin installed will need to be manually removed.
The report is as shown in Figure 9.p.

In this case all the files making up the Plugin were installed correctly. If the installation procedure attempts to overwrite a file that already exists (for example, re-installing the same Plugin, or two Plugins that use two files with the same name), the result will be as shown in Figure 9.q.

Figure 9.q – "NTP" Plugin installation report if the Plugin is already installed

The following messages are displayed in the installation report:

[ok]	file installed correctly
[Yet present]	file already present
[!!!]	file not installed correctly

NOTE If two Plugins include two files with the same name and the same path, during installation the old file will never be overwritten with the new one, and as a consequence uninstalling the second Plugin will not delete the file in the first.

Once having completed the installation of the Plugin, the Plugin page will be as shown in Figure 9.r.

Figure 9.r – Plugin page after having installed "NTP"

The "Plugins found" section shows the list of Plugins installed: The previous text "No plugins found" in Figure 9.r above has been replaced by the link to the "Network time Protocol", which opens the HTML page for setting the Plugins present in "/usr/local/root/flash/http/plugins/"; the link ("Network time Protocol") is defined on the same HTML page.

6. Set the parameters for the Plugin installed, clicking the "Network Time Protocol" link (for the "NTP" Plugin, the configuration page is shown in Figure 9.s).

7. Select the "Install/uninstall plugins" link shown in Figure 9.r above again to return to the "Install plugins" section; the page will be as shown in Figure 9.t.
It can be noted that the link for the installation of the Plugin is no longer present, and in its place, if there are no other Plugins available for installation, the text "No install scripts found" is shown, and below, in the "Uninstall plugins" section, is the link to uninstall the Plugin already installed.

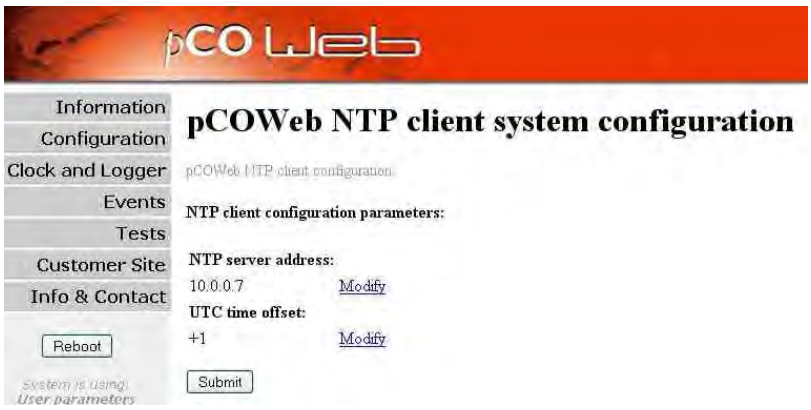


Figure 9.s – Page for setting the "NTP" Plugin parameters

IMPORTANT If you make some changes via FTP to the Plugin files installed, at the end you must fix their access rights by clicking the "Fix HTML pages and CGIs rights" link (see 9.3 on page 43).

To install other Plugins, repeat all the above operations.

For firmware version before A.1.4.2, to make the Plugin installed operational, the *pCOWeb* must be rebooted.

9.6.2 Uninstalling a Plugin

To uninstall, for example, the NTP Plugin installed in the previous pages, with reference to Figure 9.r on page 47:

1. click "Install/uninstall plugins", obtaining a page similar the one shown in Figure 9.t; the "Uninstall plugins" section shows the list of all the Plugins currently installed;

2. to start the uninstallation procedure, click the link for the Plugin to be removed.
   IMPORTANT: Confirmation is not requested before starting the procedure.

Figure 9.t – Install/Uninstall Plugin page after having installed "NTP"

At the end of the uninstallation procedure, a report is displayed, similar to the one created during the installation procedure (see Figure 9.u).

The following messages are displayed by the uninstallation report:

[ok]file removed correctly
[!!!] file not removed correctly

If a file has not been removed correctly (it may have been removed previously or the permission has been changed manually), manual access via FTP is required for complete removal.

IMPORTANT: for firmware version before A.1.4.2, *pCOWeb* must be rebooted to make the changes effective.

Figure 9.u – "NTP" Plugin uninstallation report

9.7 PROTECTION AND ACCESS CONTROL

9.7.1 Access rights to the custom HTML pages

Access restrictions can be defined to the custom HTML pages: to access these from a web browser (for example Internet Explorer) Username/Password authentication will be required.
pCOWeb requires authentication if and only if the directory that contains the html page also contains the special file called:

.htpasswd
(note the dot before the letter h)

Once the authentication data has been entered in the browser, these are compared against the data saved in .htpasswd; if the data does not match, access from the browser will be denied.

The access restrictions to the HTML pages MUST NOT BE CONFUSED WITH THE AUTHORISATION FOR USERS OF THE OPERATING SYSTEM described further on; for example, the password specified to access a page is still valid even ***pCOWeb*** is started with the button; if forgetting the password, to access the HTML pages the access restrictions must be deleted (see this paragraph below).

SETTING OR EDITING THE HTML ACCESS RESTRICTIONS
Click the points shown in Figure 9.v below:

Figure 9.v – Protection and access control

thus obtaining:

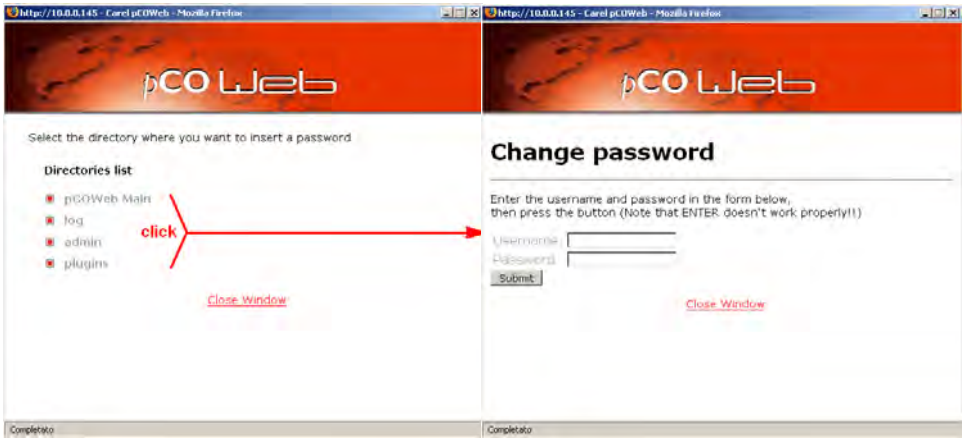


Figure 9.w – Directory list for setting access restrictions to the HTML pages

IMPORTANT

- For this information ***pCOWeb*** is case sensitive!
- The changes will be active as soon as they are confirmed; ***pCOWeb*** does not need to be rebooted.
- Using the button when rebooting ***pCOWeb*** (see 3.1.2 on page 12) causes no variations in the restrictions: those already set are still valid.
- The Username and Password fields of an already created restriction will be ever displayed as empty.
- Selecting Submit creates the restriction even if the fields are empty; during authentication, the corresponding fields should be left empty.
- To skip the creation of the access restriction, close the dialogue box without pressing "Submit".

Directories list (Figure 9.w above)

- pCOWeb Main: used to set the Username/Password for the HTML pages contained in the /usr/local/root/flash/http directory (no affect on the subdirectories).
- log / admin / plugins / …: list of the directories and related subdirectories contained in /usr/local/root/flash/http (the index_img directory will not be displayed); the list shows both the directories with active restrictions that those without restrictions.
  The Username/Password for the selected directory can be set and/or changed.

IMPORTANT NOTE The "admin" directory contains the .htpasswd file that is used for authentication to the ***pCOWeb*** configuration page contained in /usr/local/root/config/ (see 9.2 on page 40). That directory (read-only) in fact includes a permanent link (name: .htpasswd) to the /usr/local/root/flash/http/admin/.htpasswd file.

- To cancel the restriction, access the memory via FTP and delete the following file:

  /usr/local/root/flash/http/admin/.htpasswd

- To restore the default restriction, access the user memory via FTP and copy the file:

  /usr/local/root/defadmin/.htpasswd

  to the directory:

  /usr/local/root/flash/http/admin/

TIP
The links, as described in the IMPORTANT NOTE above, can be used to attribute the same authentication to multiple directories: after having generated, for a directory, the .htpasswd file as shown in Figure 9.w on page 49, simply add a link in each directory (symbolic link, typical of Unix-like systems) to the .htpasswd file generated.

For example, to share the .htpasswd file located in /usr/local/root/flash/http/admin/ for the /usr/local/root/flash/http/ directory, proceed as follows:

1. from a Windows terminal access ***pCOWeb*** by selecting Accessories / Prompt and then entering:

   `telnet 10.0.3.114` (IP address or name of the ***pCOWeb***);

2. at the login request, type:

   `httpadmin`

   and the password (see 9.7.2 below);
3. type:

   `cd /usr/local/root/flash/http/`

   (new directory for sharing the same Username and Password as /usr/local/root/flash/http/admin);
4. type:

   `ln -s /usr/local/root/flash/http/admin/.htpasswd .htpasswd`

   (this creates a link called ".htpasswd" that points to the /usr/local/root/flash/http/admin/.htpasswd file generated previously).

REMOVING AN HTML ACCESS RESTRICTION
The only way to remove access restrictions that have already been set for the HTML pages contained in a directory is to access the ***pCOWeb*** memory via FTP (see 4.2 on page 17) and manually delete the .htpasswd file in the directory in question.

9.7.2 Users of the operating system

The functions of the GNU/Linux operating system that ***pCOWeb*** is based on can be accessed by logging in with one of the following four default, non-modifiable usernames:

Table 9.a - Operating system Usernames and Passwords

Username	Password when starting without pressing the button: default value (modifiable)	Password when starting with the button (see 3.1.2 on page 12) (non-modifiable)
root	froot	froot
httpadmin	fhttpadmin	fhttpadmin
carel	fcarel	fcarel
guest	fguest	fguest

A modifiable Password is provided as default for each of the four users. The modification does not affect the password that is used to access ***pCOWeb*** when started with the button pressed (see 3.1.2 on page 12), which is non-modifiable.
The Password can be modified to allow only authorised personnel to access ***pCOWeb***.

IMPORTANT

- For this information ***pCOWeb*** is case sensitive!
- The changes will be active as soon as they are confirmed; starting from version A.1.4.2 ***pCOWeb*** does not need to be rebooted.

Figure 9.x – Changing the password for the operating system access Usernames

"root" USERNAME
Access with the "root" identifier is used internally to run critical applications for which the operating system requires special permission; logging in with this Username means there are no restrictions; therefore, it is important to set a Password other than the default "froot" to prevent potentially dangerous outside access.

"httpadmin" USERNAME
This is the only Username of interest to the user of the ***pCOWeb*** and must be used to access the user memory via FTP.
This type of identification only allows functions relating to the memory, as established by CAREL. Setting a new Password for this User prevents unwanted access to the memory.

"carel" AND "guest" USERNAMES
These are used internally to run applications. They have no interest for the user of the ***pCOWeb***, except to change the default Password so as to increase security.

IMPORTANT If the password is forgotten, the only way to access the operating system functions protected by the Username/Password is to start ***pCOWeb*** with the factory Passwords pressing the button, as described in 3.1.2 on page 12.

9.8 VARIOUS TESTS: PING - *pCO* VARIABLES – NOTIFICATIONS - VERBOSITY

The "Tests" page features the following tests:

- Ping a remote Host: ***pCOWeb*** can be used to send a "ping" command to a computer. This is useful for checking whether or not communication is possible between ***pCOWeb*** and a computer (cable problems, IP address or subnet mask settings).
  IMPORTANT: the Gateway normally needs to be set, see 9.4.1 on page 44.

  Specify the name (a DNS must be set – see 9.4.1 on page 44) or the IP address, then select the "Ping" button; the command asks the recipient to respond to acknowledge reception; after sending the command, ***pCOWeb*** will display the outcome on the confirmation page; see Figure 9.z below.

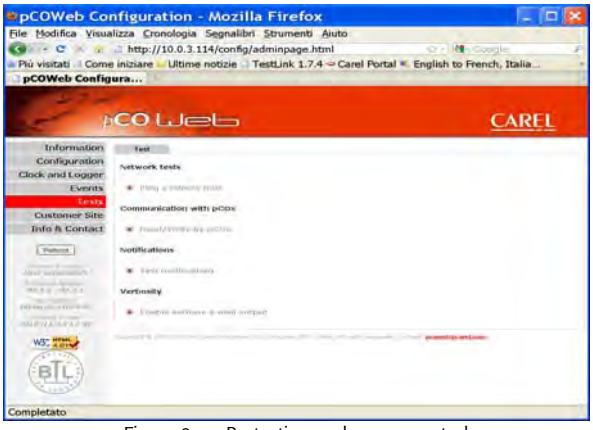


Figure 9.y – Protection and access control

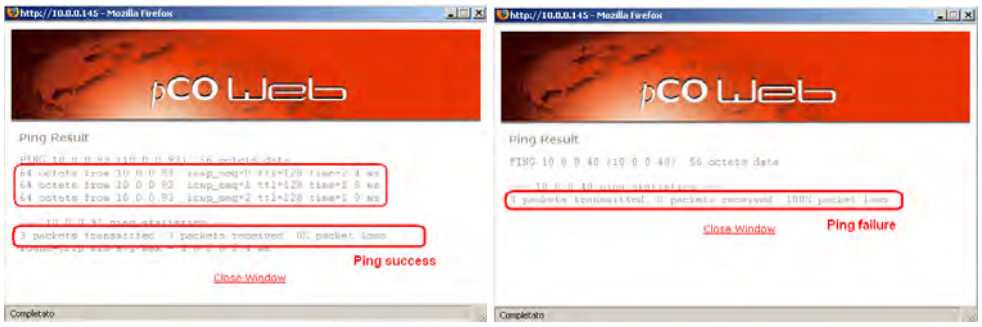


Figure 9.z - Results of the "Ping" command

- Read/Write by pCOx: is used to read or write the contents of a ***pCO*** variable using the page shown in Figure 9.aa below;
  - Choose the variable: choose the variable to be read or written;
    IMPORTANT The selection is also common to the write function and only becomes effective after selecting the "Read" button; the "Write" button does not confirm the selection made in the "Choose the variable" section, but rather writes the values shown on line with the "Write" button and highlighted in Figure 9.aa;
  - Current value: value read; this may be numeric or "U" (for the meaning of "U" see 9.2.2 on page 41);
  - Variable <Type> <index> new value: enter the new value for the variable and select Write, then wait for the outcome page and check whether the value has been accepted (= the ***pCO*** application can write the variable) or has remained unvaried (= the ***pCO*** application manages the variable as read-only);
  - Operation result legend (Undefined / Ok / Timeout): describes the outcome of the last read or write operation:
    - Ok: after a write operation, ***pCOWeb*** has received acknowledgement from ***pCO*** with the current value of the variable;
    - Timeout: if 10 seconds elapse from the write operation without the ***pCO*** sending the acknowledgement message containing the current value of the variable; possible situations:
      – the value sent to the ***pCO*** is the same as the existing value;
      – the index of the variable is outside of the limits managed by the BIOS version of the ***pCO*** used;
      – communication between ***pCO*** and ***pCOWeb*** is interrupted; for further information see APPENDIX C on page 65.
    - Undefined: returned for all read operations, and for the write operations that do not send any request to the ***pCO***, that is, when a numeric value has been entered that is not allowed:
      [Digital: 0/1], [Analogue: -3276.8 to 3276.7], [Integer: -32768 to 32767].

Figure 9.aa - ***pCO*** variables test

- Test notifications: send all the notification messages set, regardless of whether the corresponding events occur or time interval has expired. For details of the notifications see 4.3 on page 20. The message displayed once you executed the test is:

Figure 9.bb - ***pCO*** variables test

NOTE 1 The notifications may take a lot of time to be sent, depending on the number of messages set; during this activity the Status LED could stop flashing; the completion of the notifications will be signalled by the confirmation page.

NOTE 2 Running this test does not ensure that all the messages have effectively been delivered: check that all the conditions described in 4.3 on page 20 are Ok so that ***pCOWeb*** can deliver the messages.

- Verbosity: this function allows verifying the sending email logs. Switching in the debug mode (Figure 9.cc) it will open a specific window (see Figure 9.dd) where it will be possible to analyse the mail traffic.

Figure 9.cc – enable Verbose

On the right is possible to see an example of communication between *pCOweb* and the e-mail server

NOTE: *pCOweb* must be restarted when finished monitoring traffic to server.

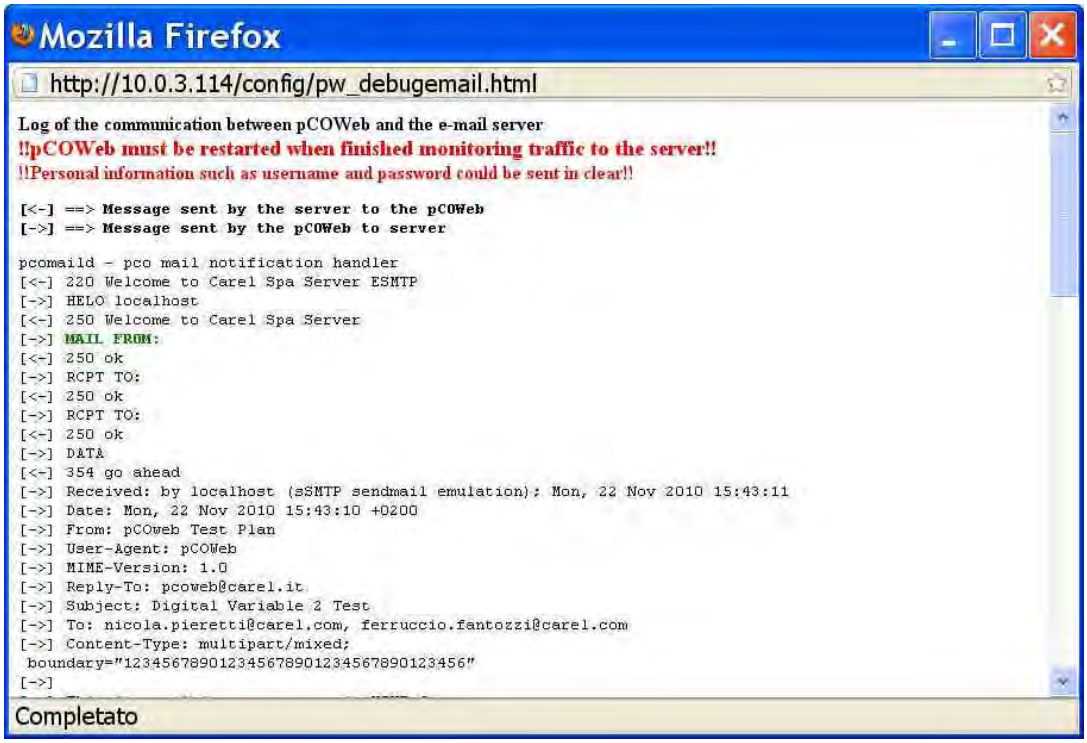


Figure 9.dd – Mail logs

9.9 RESTARTING *pCOWeb*

Some settings require the *pCOWeb* to be rebooted.

The reboot can be performed in the following ways:

- Disconnect the power supply for a few seconds (the LEDs must go off), then reconnect (if the operation is to be performed after setting the value of a parameter, wait until the setting has been confirmed, to allow time for the confirmation to be sent via the browser to *pCOWeb*);
- or alternatively: select the "Reboot" button available on the configuration web pages (Figure 9.d on page 42);
- or alternatively: use the button shown in Figure 1.b on page 8 with the procedure described in 9.9.1 below.

9.9.1 Restarting *pCOWeb* using the button

PROCEDURE

1. Check that the Status LED is flashing continuously green or red, or is on steady red ("Rescue mode", see 9.10.3 on page 55); in any other situations, the procedure cannot be started.
2. Select and hold the button for at least 5 seconds, then release it.
   IMPORTANT: *pCOWeb* ignores the button when pressed for more than 10 seconds, interpreting this as accidental.

After a few seconds, the Status LED stops flashing, it may change colour and switch off, then it will flash very quickly red-green-red-green for a few seconds, confirming the reboot; after 1-2 minutes, the completion of the reboot phase will be signalled by the regular flashing of the Status LED, or the steady red colour ("Rescue mode", see 9.10.3 on page 55).

9.10 FIRMWARE UPDATE

The *pCOWeb* firmware can be updated by uploading it from a computer via the Ethernet connection, using the configuration page.
The firmware is divided into two separate blocks, each with its own version. The current blocks and versions are:

A1.4.2 (block containing most of the application programs)
B1.2.1 (block containing the BIOS, that is, the GNU/Linux operating system and the basic applications).

To check the firmware version on the *pCOWeb* see Figure 9.d on page 42.

INFORMATION

- Every update operation loads only a block per time.

- *pCOWeb* does not check for coherence between the two blocks.
- There is no need to update both blocks if one of these is already up-to-date.
- The order of updates (A then B, or B then A) is indifferent.
- An earlier block than the current one can also be loaded.
- Currently there is no incompatibility between the versions of the two blocks; for outdated versions check the documents supplied.
- Updating the firmware does not affect the user settings, as the non-volatile memory is divided into areas. Nonetheless, as the information on the sub-division of the areas is contained in block B, in the future CAREL may need to move the sub-division information following the addition of new functions. In that case, the change will be described in the documents, and the data will need to be backed up (see 9.1 on page 40).

IMPORTANT If when updating block A the procedure is interrupted due to a power failure, the block A transferred is not valid and upon next reboot *pCOWeb* will enter Rescue Mode (see 9.10.3 on page 55); the normal functions will no longer work, but *pCOWeb*, due to presence of a valid block B, will continue to operate to allow block A to be loaded again, although with a slightly different procedure.

IMPORTANT **If when updating block B the procedure is interrupted, the product will be need to be sent to CAREL for reprogramming in the factory.**

NOTE Rescue Mode is also activated by updating outdated firmware on *pCOWeb* (for example if starting from version A1.2.7 - B1.1.5 and updating to A1.3.1 - B1.2.1); in fact, the last versions have different limits in the allocation of the non-volatile memory that cause the corruption of the existing block A.

9.10.1 Procedure for updating the firmware from web pages

The firmware updates are distributed by CAREL divided into two separate files included in the same compressed file. For example, the update relating to the version that this manual refers to is:

UPDATE FILE NAME on http://ksa.carel.com: **pCOWeb_A142-B121.zip**

Block name	Block type	Update file name	Date
A.1.4.2	Applications	flash_apps_A1.4.2.bin	2010.10.30
B.1.2.1	Bios	flash_sys_B1.2.1.bin	2006.08.02

INFORMATION
pCOWeb recognises the type of block not by name, but rather by some information it contains.

TIP: if updating block A only on a series of *pCOWeb* devices, use the procedure shown in paragraph 9.10.2 on page 55.

PROCEDURE

1. Download the update to a directory on the computer and decompress it using any ZIP decompression utility (for example, "Filzip" freeware, http://www.filzip.com).
2. Open the *pCOWeb* firmware update page (Figure 9.ee).
3. Use the "Browse" button to choose the file corresponding to the block being updated.
4. Select the "Submit" button: the block will start being transferred to the VOLATILE memory on *pCOWeb* (the non-volatile memory is not modified in this phase, however if at this point the procedure needs to be stopped, *pCOWeb* will have to be rebooted).
   Wait for the confirmation page to be displayed (see Figure 9.ff), then close the window and wait until the page shown in Figure 9.ee above turns into that shown in Figure 9.gg on page 55, describing the outcome of the test on the transferred block.
5. Check that the type of block recognised is as expected ("FLASH APPS" string highlighted in Figure 9.gg on page 55).

Figure 9.ee – Firmware update page

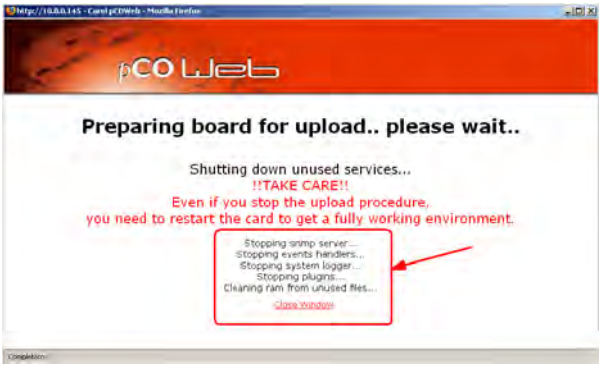


Figure 9.ff – Confirmation of block transferred

6. To start to write the block to NON-VOLATILE memory select "Proceed and Reboot" (no confirmation is requested).
   For block A, the effective writing to memory corresponds to the phases in which the Status LED is on steady red; for block B, on the other hand, this corresponds to the phase in which the Status LED alternates red-green, starting some tens of seconds after pressing the button.

At the end ***pCOWeb*** will be rebooted automatically. After around 2 minutes, the updated main configuration page can be opened to check that the firmware version has been changed (see Figure 9.d on page 42).

Figure 9.gg – Block transferred analysis report – Start update or cancel

9.10.2 Procedure for updating the firmware via FTP (block A only)

INFORMATION

- The firmware (only block A) can also be updated via FTP.
- An already updated ***pCOWeb*** that the firmware can be downloaded from is required.
- The "flash_sys_Bx.x.x.bin" file used in paragraph 9.10.1 on page 54 cannot be used for this procedure.

PROCEDURE

1. Access an already updated ***pCOWeb*** via FTP (see 4.2 on page 17), going to the directory:

   ```
   /usr/local/root/flash/dev/
   ```

2. Take the following two files:

   ```
   romapps.img
   romapps.img.chk
   ```

   and copy them to a temporary directory on the computer.
3. Access the same directory on the ***pCOWeb*** being updated via FTP.
4. Delete the romapps.img and romapps.img.chk files contained in the directory.
5. Copy the files from the computer to ***pCOWeb***.
6. ***pCOWeb*** must be rebooted to complete the update.

IMPORTANT: if one or both files listed above are missing or are not valid, when rebooted ***pCOWeb*** will go into Rescue Mode (see 9.10.3 below).

9.10.3 Rescue mode

INFORMATION

- When ***pCOWeb*** is rebooted it first runs the block B firmware, then checks the presence and validity of block A, starting up as normal if everything is Ok.
- If block A is not valid, ***pCOWeb*** enters Rescue Mode to allow block A to be reloaded.
  The Status LED is on steady red.
  - In this mode, the normal functions are not available and access to the standard configuration page is not possible, as those pages are contained in block A.

HOW TO RELOAD THE BLOCK "A" FIRMWARE

1. Access ***pCOWeb*** by entering the IP address (during the Rescue Mode, the pages need to be refreshed by pressing Ctrl-F5, otherwise unexpected results may occur due to the use of the browser's cache memory); the page shown in Figure 9.hh on page 56 will be displayed.
2. Click "Provide firmware image"; (see Figure 9.ii on page 56), the page used to choose the block A file will be displayed; select "Upload" to transfer the block to the VOLATILE memory (during this phase the block will only be copied to the VOLATILE memory and checked).

3. When the page showing "Upload result: Ok" is displayed, select "Press here to start immediately": the file will be written to NON-VOLATILE memory.
4. The end of the procedure will be signalled by the page shown in Figure 9.jj below; then reboot ***pCOWeb***.

5. Wait at least 2 minutes, then load the main page by entering the name of the ***pCOWeb*** or the IP address in the address field and pressing Enter.

NOTE **After rebooting the Rescue Mode page may be displayed again. Usually this is simply a problem with the browser.**
To make sure that ***pCOWeb*** is no longer in Rescue Mode, check the Status LED; if flashing green, the Rescue Mode page displayed is one saved in by the browser; refresh the page by pressing Ctrl-F5; if this doesn't work, try deleting the browser's cache memory, and if this still doesn't work, close and open the browser again (all the instances running).

Figure 9.hh – The "Rescue Mode" page

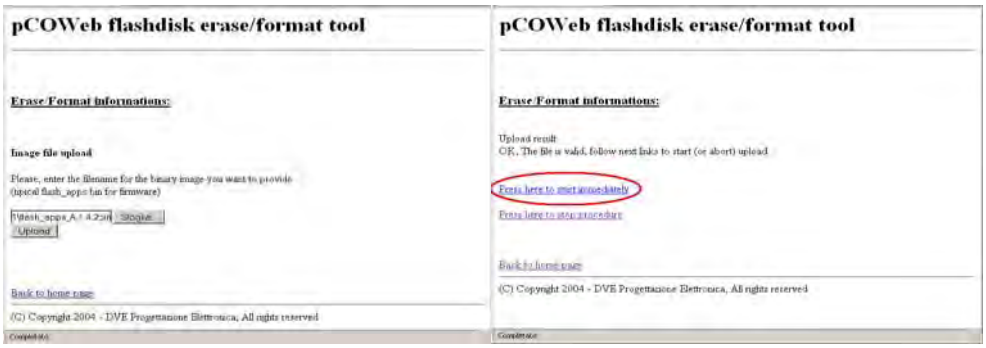


Figure 9.ii – Reloading block A

If, on the other hand, Rescue Mode is still active even after rebooting ***pCOWeb***, there may be errors in the formatting of the non-volatile memory. Proceed as follows:

1. Click the "Completely format flashdisk" link shown in Figure 9.kk below to regenerate the data structure of the non-volatile memory.
2. Choose: "Yes, please, erase all the flashdisk, all the user configurations and pages will be lost": all the HTML pages entered by the user and all the configurations will be lost, including the network settings. When next rebooting the button will need to be pressed (see 3.1 on page 10), using the default address 172.16.0.1 to open the Rescue Mode pages; make sure that the computer is configured correctly as described in 3.1 on page 10 to access the address 172.16.0.1.
3. Then repeat the procedure for loading block A described above.

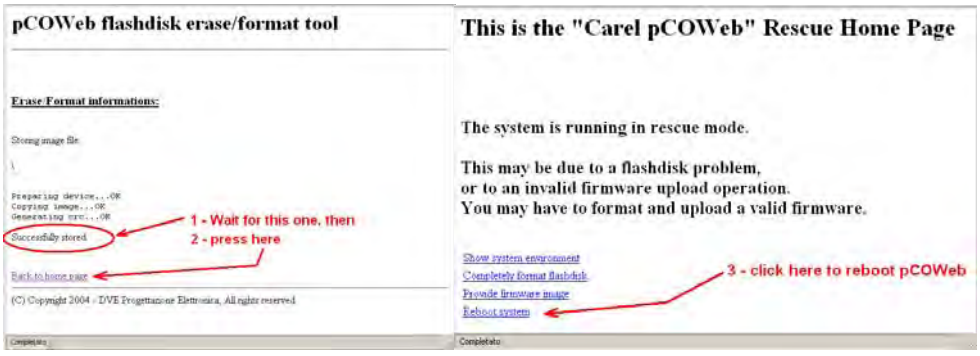


Figure 9.jj – When loaded, reboot ***pCOWeb***

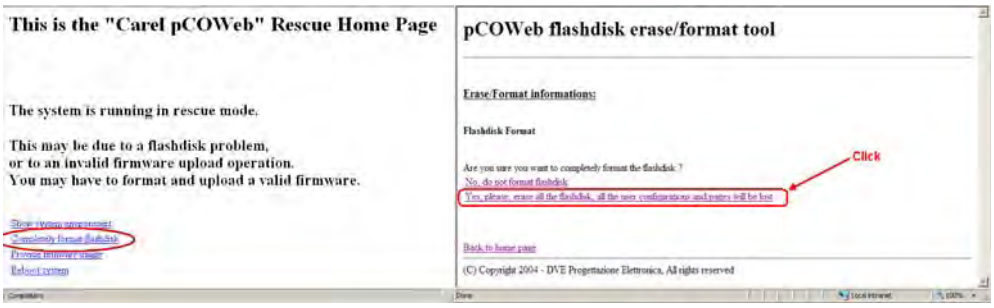


Figure 9.kk – Formatting the non-volatile memory

9.11 pCO APPLICATION UPDATE

pCOWeb is able to update the *pCO* application of every type of *pCO* controller, except for *pCO2* (only because *pCO2* doesn't automatically restart when the upload has been completed).

To update the *pCO* application, when Carel protocol is running ,you need either:

•A bios version >=4.10 installed, as starting from this specific versions of bios the *pCO* switches automatically from Carel protocol to Winload protocol (SERIAL1_PROTOCOL=4 or BMS_PROTOCOL = 4) as soon as the *pCOWeb* sends a specific command.
• An application which is able to switch to Winload protocol (SERIAL1_PROTOCOL=4 or BMS_PROTOCOL = 4) on the serial port 1 (BMS) when a dedicated digital variable changes from "0" to "1"; this doesn't depend on the protocol used between *pCO* and *pCOWeb*.

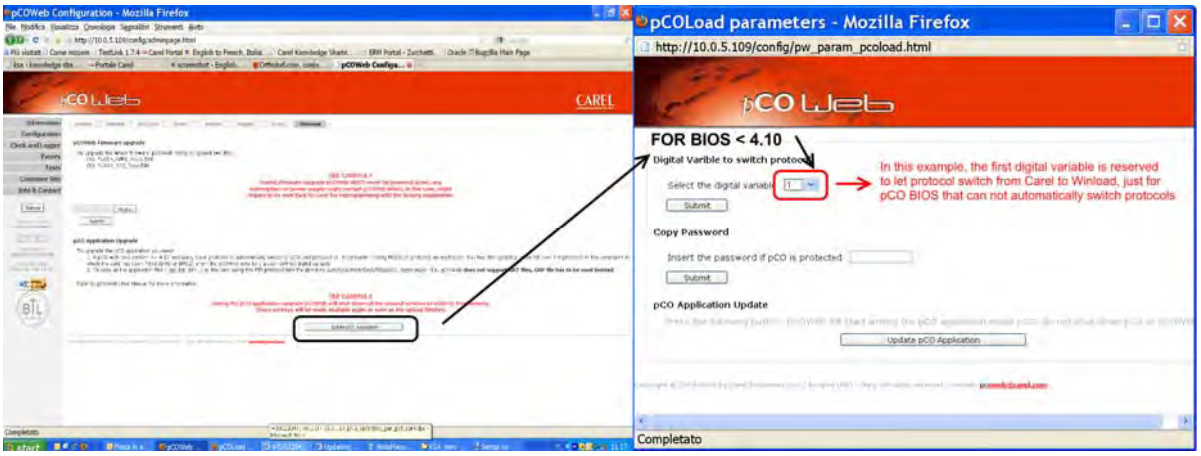


Figure 9.ll Digital variable necessary for BIOS < 4.10 in Carel protocol

This required auto switch protocol is not supported with modbus extended, and it is not possible to reserve a digital variable during the update procedure, as described below.

To update the *pCO* application, when Modbus Extended is running:

•An application which is able to switch to Carel protocol (SERIAL1_PROTOCOL=1 or BMS_PROTOCOL = 1) on the serial port 1 (BMS) when a dedicated digital variable changes from "0" to "1".
In this case, the procedure is quite different from the one described for Carel protocol with BIOS < 4.10, in fact the switching of the variable to change protocol has to be performed before pressing: "Update *pCO* Application":

Figure 9.mm Manual switch to change protocol

After switching the above-described variable from "0" to "1", a double reboot for ***pCOWeb*** is necessary: first to go from Modbus Extended to Carel (in the ***pCO*** Com, see picture 9.l), after this first reboot the ***pCO*** Application Upload can be run:

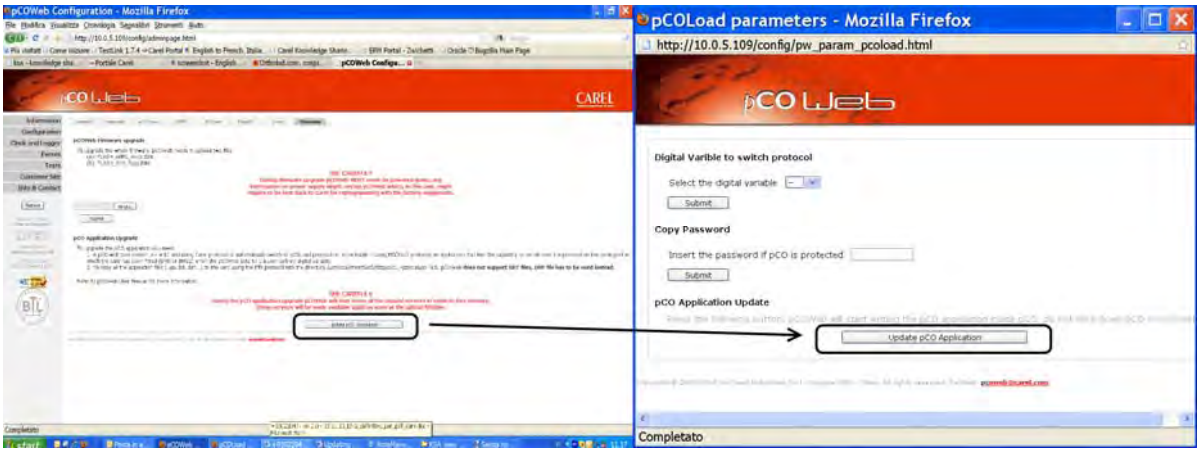


Figure 9.nn ***pCO*** Application Upload

Then, once that the Upload procedure has been completed it is necessary to perform the second reboot of ***pCOWeb*** to go back from Carel to Modbus extended (always in the ***pCO*** Com). <u>This is mandatory if using modbus extended protocol.</u>

pCOWeb supports:

- •BLB, BIN, BLX (Strategy)
- •IUP (User interface)
- •GRP (graphic resources, available only with 1Tool)
- •DEV (default parameters)
- •PVT, LCT (bios logger configuration)

pCOWeb doesn't support:

- •GRT
- •BIOS
- •BOOT

VERY IMPORTANT

It is standard in custom ***pCO*** application and in each Carel standard application that, in order to avoid memory mismatching, to re-initialize the unit to the default configuration when the application updates, so that is strongly advised to upload, together with the applications' files, a consistent DEV file for the specific application (i.e. downloading it from another controller with similar configuration and the new software).

Updating the ***pCO*** application is a very critical procedure for the unit, it implies to switch it off before taking any action, ***pCO*** will de-energize every digital and analogue output during the firmware upgrade without taking in consideration any safety timing or specific procedure defined in the software.

Please, go to http://ksa.carel.com/ and download the step by step guide "Updating ***pCO*** Application via ***pCOWeb***.pdf"

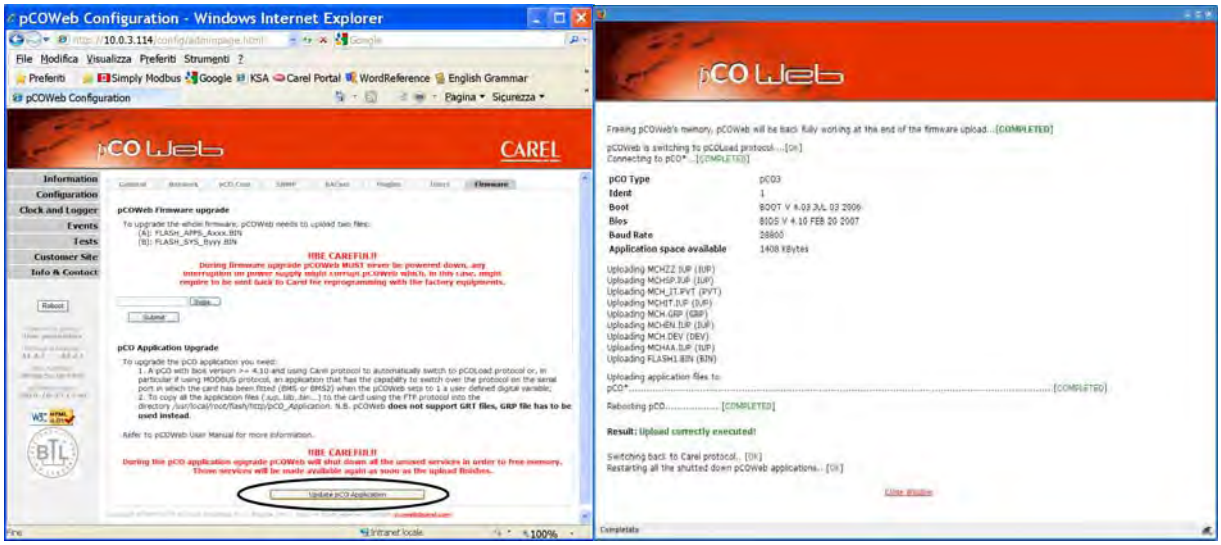


Figure 9.oo – ***pCO*** Application Upload correctly executed

10 TECHNICAL SPECIFICATIONS

CPU:	ARM7 TDMI@74MHz clock
Memory:	16MB RAM, 8 MB FLASH (around 3MB available for web pages and user data).
Operating system:	GNU/LINUX 2.4.21 – file system for the non-volatile user memory: JFFS2
Firmware	A1.4.2 – B 1.2.1
Ethernet interface:	RJ45 connector for 10BaseT Ethernet; use a class 5 shielded cable, max 100 m.
Protocols managed:	HTTP (web server: thttpd 1.1) SMTP DHCP FTP DNS SNMP v1 (Net-SNMP 5.0.9) SNMP v2c (Net-SNMP 5.0.9) BACnet Ethernet ISO8802-2/8802-3 BACnet/IP (Addendum A/Annex J) MODBus TCP/IP
Operating conditions:	0T55 °C, 20/80% RH non-condensing
Storage conditions:	-20T70 °C, 20/80 % RH non-condensing
Environnemental pollution:	Normal
CAREL order code:	PCO1ØØØWBØ

APPENDIX A MAC ADDRESS - STATIC OR AUTOMATIC IP ADDRESS (DHCP).

Each ***pCOWeb*** has a MAC address ("Medium Access Control") that is unique at a worldwide level from all other network devices. In addition, each ***pCOWeb*** connected in the data network must be set with a unique IP address. If the network that the ***pCOWeb*** is connected to uses the automatic "DHCP" system for assigning the address, a manual setting of the addresses will not be needed. Vice-versa, each individual ***pCOWeb*** will need to be set one-by-one the IP address.

MAC address

Each network device is distinguished from any other device connected on the network by the MAC address.
The MAC address is a unique number at a worldwide level. It is attributed to the device in the factory and subsequently cannot be modified.
It is made up of 16 alphanumeric characters, in the following range: 0, …, 9, A, …, F. For easier interpretation, this number is often represented in groups of two characters; an example of MAC address is:

```
00.0A.5C.10.00.81
```

The data network is a connection that carries messages from one network device simultaneously to all the other devices connected. Therefore, normally a network device also receives messages sent to other devices.
Each network device only considers a message if the destination MAC address matches its own. All the other messages are ignored, except for the messages sent to a special MAC address called "broadcast": FF.FF.FF.FF.FF.FF, which are considered by all the devices, even if the destination address does not correspond to their own. The broadcast address is used to send "messages for all". For this reason, no network device can have the broadcast address as its own address.

IP address

In the data networks, the devices are also uniquely identified using another address, called the "IP address".
IP stands for Internet Protocol.
Each device is therefore set with its own IP address.
Within a network, each IP address must be unique.
The correct setting of this address is fundamental for the exchange of data between the various devices.

The IP address is made up of four numbers separate by dots. Each of the four numbers is between 0 and 255. An example of an IP address is:

```
10.0.0.204
```

Static IP

The attribution of an IP address to a network device, for example ***pCOWeb***, is subject to a number of rules; therefore, when ***pCOWeb*** is installed in a data network, the network administrator must be contacted for a valid IP address.

Only the network administrator who has the overall view of the entire configuration of the installation can in fact choose a valid address for the ***pCOWeb***. In systems that do not use the automatic DHCP system, each individual ***pCOWeb*** will need to be set with the IP address provided by the administrator (see 9.4.1 on page 44).

Dynamic IP (DHCP) (default method for *pCOWeb*)

In some networks, the IP address is automatically assigned to a network device by the central server via the data network itself (this is called the DHCP server).
The operation that uses an IP address provided by a central server is described as the "dynamic" IP address.
This setting avoids, for systems that use DHCP, having to set the correct address on the ***pCOWeb*** and (computers) one-by-one, allowing significant time savings.
In this case, the network administrator will have configured the DHCP server with a table of MAC addresses ↔ IP addresses, and will be able to distinguish the various devices by the MAC address and a table that describes, for each MAC address, the physical position of the device in the installation (see 2.2 on page 9). In this case, the devices that are connected must be first have been configured to use this method for exchanging the IP address.
pCOWeb has DHCP active by default.
In these networks, therefore, the IP address is decided by the network administrator who configures the DHCP server, and who must be contacted to ask which address has been attributed.
If DHCP is active, up to three additional static IP address can be assigned – see 9.4.1 on page 44).

APPENDIX B IP ADDRESSES, PROXY SERVER, SUBNET MASK, DNS, GATEWAY

IP ADDRESSES

RECEPTION
A message that reaches a device is accepted if the destination IP address contained in the message coincides with the receiver's address.

TRANSMISSION
A PC can send data only to a limited range of IP addresses, called the subnetwork. The definition of the subnet for a given PC depends on the combination [IP address – Subnet mask].
A practical test to establish whether two IP addresses belong to the same subnetwork, is as follows:

NETMASK – PRACTICAL TEST
If the Subnet mask of the two devices are identical and the IP addresses corresponding to the "255" positions of the Subnet mask are identical, the two IPs belong to the same subnetwork.

Example 1: same subnetwork

SUBNET MASK	**255.**	**255.**	0.	0
IP1	**10.**	**0.**	1.	34
IP2	**10.**	**0.**	2.	245

The device with IP address 1 communicates directly with the device with IP address 2.

Example 2: different subnetworks

SUBNET MASK	**255.**	**255.**	0.	0
IP1	**10.**	**1.**	1.	34
IP2	**10.**	**0.**	2.	245

The device with IP address 1 CANNOT DIRECTLY communicate with the device with IP address 2 (a Gateway device will be needed).

PROXY SERVER

For some types of message, the PC can also be configured so that the path travelled by the message includes (if featured in the local network) a special device called a proxy server, which performs some additional tasks.

An example of such tasks is the one that is normally performed when requesting access to Internet sites from a PC on a local network; in this case, each PC accesses the Internet via a central server on the network, which has the task of managing the traffic to the outside.
Normally, the server features a proxy so as to store the last pages accessed in its memory, and, if on the websites these have not been changed (date / file size), rather than download them from the sites it provides its own copy, thus avoiding useless data traffic; in general, a proxy returns a result that depends on the rules set on the proxy.

If ***pCOWeb*** is installed in a network with a proxy server, the browser on the PCs in the network are normally already set to send their HTML page requests to the proxy. In this case, if the network server has not yet been set to locate the ***pCOWeb*** in the network, a personal computer will not be able to access the ***pCOWeb*** HTML pages via the proxy. Consequently, the easiest solution is to disable the proxy on the PC used to contact the ***pCOWeb***. This means however that the PC will not be able to access the Internet via the intranet.

FURTHER INFORMATION

The mechanism for the transmission of a data packet is illustrated below.

NOTE A network device can contact another device only if its *MAC address* is known (or the MAC address of the corresponding gateway that it connects through) and this is included in the transmission packet; in fact, the first filter for reception on each network device is a network board that rejects all the messages that are not sent to its own *MAC address*.

The transmission mechanism can be analysed as follows.

A – request from a program running on a PC to contact a site; role of the communications manager.

It is assumed that a program running on a PC requests the PC operating system to send a certain message to a server, for example www.myserver.com; the program sends the request to the communications manager on the PC; the communications manager is a program that is part of the operating system and that can be considered a "switchboard operator" for the PC: it receives transmission requests from the programs running on the PC, and receives messages from the outside, forwarding them to the programs running.

B – search for the IP address relating to the site's server.

Once the request has been received, the communications manager on the PC looks up its "table of names-IP addresses" to find the association between the requested name (URL: www.myserver.com) and the IP address of the server; if the association is not found, it connects to a special network service called "*DNS* - Domain Name Server", whose role is to supply the IP addresses corresponding to the names, in the same way as a telephone directory; the IP address of the DNS is known to the communications manager, being specified by the user in the configuration of the PC.

NOTE In the following examples, to simplify the explanations, it is assumed that the DNS server is located within the network; it must also be remembered that there are no limits to fact whether the DNS server is located within or outside of the network.

B1 – search for the MAC address of the DNS server

In reality, as described, to call the DNS server the communications manager must first acquire the MAC address; to do this, first of all it looks up its "table of IP addresses-MAC addresses" (ARP table); if it can't find the address, it sends a message using ARP protocol (Address Resolution Protocol); the message is sent to the "broadcast" address (see APPENDIX A on page 61) and consequently is received by all the devices. The message asks for a response from the network device whose IP address matches the address of the DNS server; as soon as the DNS responds, the PC saves the MAC address in the ARP table so that this can be used again in the future, without having to send the request each time; the table is not static but is constantly updated.

B2 – the DNS provides the IP address of the site.

Once the communications manager has the MAC address of the DNS server, it will send the name www.myserver.com, and the DNS will respond with the corresponding IP; if the table does not contain the address, the PC will not be able to access www.myserver.com and a warning message will be displayed.
On a PC, the same behaviour can be simulated using the "nslookup" function in Microsoft Windows™:

```
C:\>nslookup www.google.com
Server:  ns4.myserver.dns
Address:  10.0.0.14

Response from an untrusted server:
Name:    www.l.google.com
Address:  216.239.59.147
Aliases:  www.google.com
```

C – the IP address acquired may not be suitable to contact the server directly: function of the subnet mask.

At this stage, the communications manager checks whether or not it can contact the server's IP address directly; if not enabled, it will send the request to contact a network "helper" called the "gateway", which will forward the message to the recipient (and will do the same for the response); the gateway can be contacted by the PC as the user will have specified the IP address provided by the network administrator; this IP must be directly contactable.

Direct contact depends on the combination of the following numbers:

- IP address of the destination
- IP address of the personal computer
- subnet mask of the personal computer

The communications manager, using these three numbers, establishes whether the recipient's IP address belongs to the range (called the "subnet") of IP addresses it is enabled to communicate with, without having to go through the gateway.
The destination IP addresses it can communicate directly with satisfy the following criterion:

the bits of the destination IP address corresponding to those that in the Subnet mask are equal to 1, must have the same value as the bits of the source IP address

NOTE The subnet mask cannot have just any value; it must follow a number of rules, including:

- **from the left, at least the first bit must be 1**
- **continuing to the right, the first 0 bit will only be followed by 0 bits**

Examples

```
Subnet mask              255.     255.     224.       0
                  11111111 11111111 11100000 00000000

Source IP                 10.       0.      66.      64
                  00001010 00000000 01000010 01000000


Dest IP 1 (direct)        10.       0.      69.     240
                  00001010 00000000 01000101 11110000

Dest IP 2 (by gateway)    10.      12.      66.      64
                  00001010 00001100 01000010 01000000

Dest IP 3 (by gateway)    10.       0.     130.      64
                  00001010 00000000 11000010 01000000
```

In other words, the valid range (subnet) for the addresses that can be contacted directly from this PC will be:

```
ALL DIRECT IP's RANGE:
Dest IP #1              255.     255.      64.       0
                  00001010 00000000 01000000 00000000

Dest IP #2              255.     255.      64.       1
                  00001010 00000000 01000000 00000001

Dest IP #3              255.     255.      64.       2
                  00001010 00000000 01000000 00000010
. . . . . .       . . . . . .

Dest IP (#8192)         255.     255.      95.     255
                  00001010 00000000 01011111 11111111
```

D - (destination IP address contained in the subnet)

D1 – search for the MAC address of the site's server.

Similarly to step B1, to send the message the communications manager must first acquire the MAC address corresponding to the IP; first of all, it looks up its "table of IP addresses - MAC addresses"; if this is not found, it searches the network by sending an ARP message and waiting for the response from the server whose IP address is equal to the destination; as soon as the server responds, the PC saves the MAC address in the table so that this can be used again in the future, without having to send the request each time.

D2 – effectively send the message

Finally, the personal computer sends the message to the server with the MAC address acquired.

E - (destination IP address outside of the subnet: send to the gateway)

In this case, the personal computer will send the message to a special network device called a gateway, whose IP address will have been specified in the configuration of the PC. The IP address of this device must belong to the same subnet as the personal computer.

The message will be sent to the gateway after the personal computer has acquired the corresponding MAC address, as described in step B1 or D1.

The gateway, if configured by the network administrator to allow communication between the two subnets in question, will forward the message to the IP destination address; once the response is received, this too will be forwarded to the original sender (the personal computer).

Overall, then, the gateway allows the personal computer to access all other IP addresses outside of its own subnetwork.

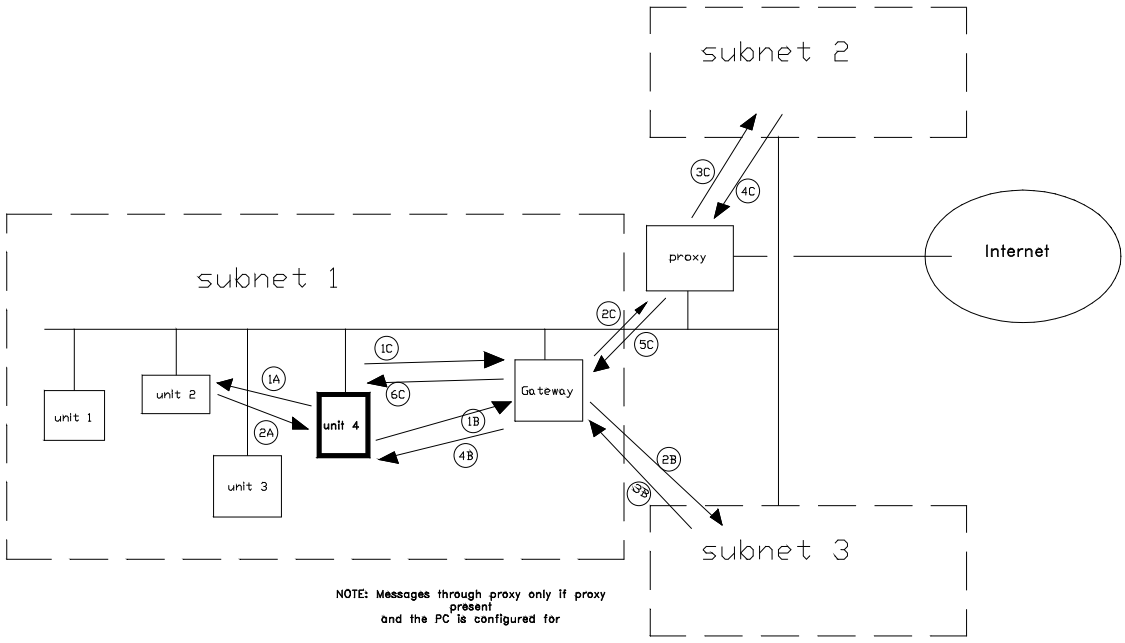


Figure B.a - Communication in a network

APPENDIX C APPLICATION - *pCO – pCOWeb* COMMUNICATION

Figure C.a below describes a write request generated by a high level ***pCOWeb*** application.

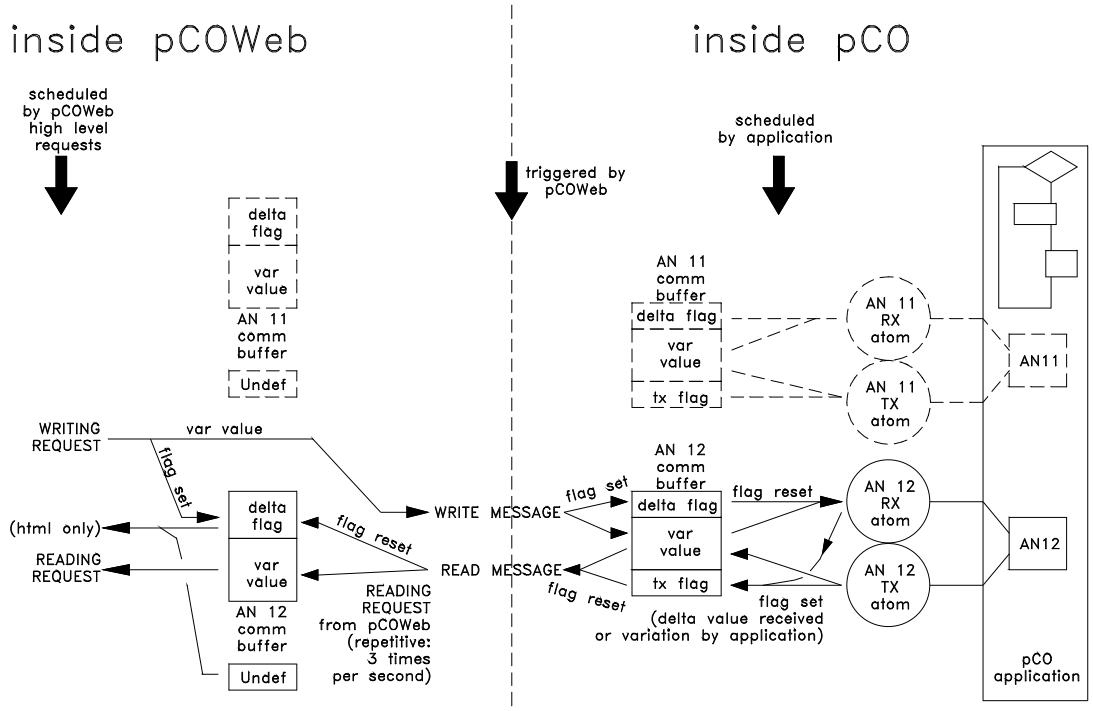


Figure C.a - Communication between ***pCOWeb*** - ***pCO*** atoms

REQUEST TO WRITE A VARIABLE FROM ***pCOWeb*** TO ***pCO***

1. The variation request (WRITING REQUEST) is not transcribed to the ***pCOWeb*** table (comm buffer: var value) but rather is sent directly to the ***pCO***.
2. At the same time, on the ***pCOWeb*** the variation sent flag is raised (comm buffer: delta flag "set"); this flag is ONLY used by the HTML pages when reading to establish whether the variations sent by the page with the "Submit" button have been confirmed by the reception of the new values on the ***pCO***: while there are still some variations that have not yet been confirmed by the ***pCO***, ***pCOWeb*** does not generate the confirmation html page on the browser; a 10 seconds Timeout is reset whenever each new confirmation is received; the html confirmation page will be generated and sent back when all confirmation have been received OR when the Timeout expires.
3. The value is received by the ***pCO*** and saved in the corresponding communication buffer (comm buffer: var value), together with a variation received flag (comm buffer: delta flag "set"); the flag is only raised if the value received is different from the existing value.
4. If the application running on the ***pCO*** contains the atom for receiving the variable (RX ATOM), when the atom is run it detects the raised flag, resets the flag (comm buffer: delta flag "reset") and transfers the new value to the variable (example: AN12) for use within the application; at the same time it "books" the re-transmission of the variation by raising the flag in the communication buffer on the ***pCO*** (comm buffer: tx flag "set").
5. If the application running on the ***pCO*** contains the atom for sending the variable, whenever this is run it checks whether the value of the communication buffer (comm buffer: var value) is different from the value of the variable in the application (AN12); if it is different, it transfers the value of the variable in the application to the communication buffer and "books" the re-transmission of the variation by raising the flag in the communication buffer (comm buffer: tx flag "set").
6. With the following update request from ***pCOWeb*** to the ***pCO***, the latter will send the value and lower the delta flag (comm buffer: delta flag "reset").
7. ***pCOWeb*** will receive and save the value and the reception in its buffer (comm buffer: var value, delta flag "reset").

CONCLUSIONS

A. In a ***pCO*** application, a read-only variable does not perform the reception atom and is only varied from the application. Each value sent by ***pCOWeb*** will not produce any variation in the value of the application, but if the two values are different, the send atom will also book an echo containing the value of the variable contained in the ***pCO*** application.
B. On power-up, the ***pCOWeb*** communication buffer is loaded with a series of "U"-s (Undefined); each value received from the ***pCO*** replaces the corresponding "U" loaded at power-up.
C. If communication with the ***pCO*** is interrupted, after a Timeout of a few seconds, the entire ***pCOWeb*** communication buffer is initialised again with the "U"-s (the "Status" LED starts flashing red).
D. On power-up, or when communication is restored, ***pCOWeb*** sends ***pCO*** a command to read all the variables, regardless of the variations; ***pCO*** then sends all the values of all the variables managed by the communication buffer, irrespective of whether the atoms are present or not.
E. The ***pCO*** BIOS versions prior to 4.02 do not manage all 207 variables for all three indices, that is, have communication buffers that do not extend across the range 1-207. This means that there may be some "U"-s for the last variables.
F. The ***pCO*** BIOS versions superior to 2.31 supports Carel protocol version 3.0, ***pCOWeb*** requires this protocol version or superior.

APPENDIX D ArGoSoft: A FREEWARE MAIL SERVER

Various SMTP servers are available on the Internet.
Below is a description of the ArGoSoft Mail Server freeware (version 1.8.8.9), downloadable at the link:
`www.argosoft.com`.

After installation, when first started the screen shown in Figure D.a below is displayed.

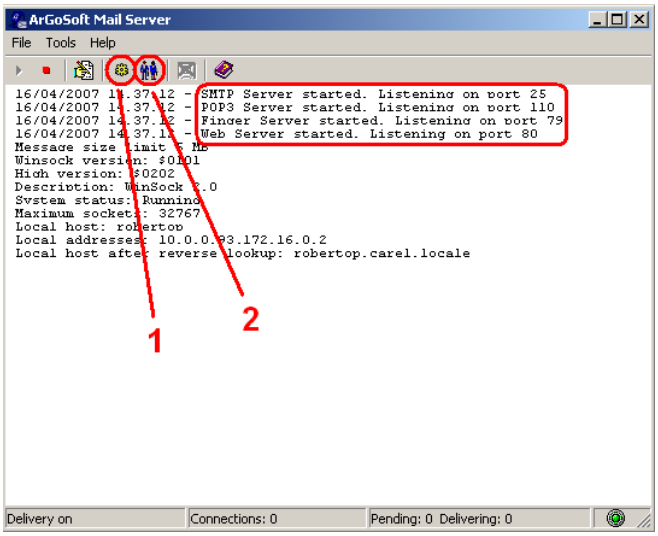


Figure D.a - ArGoSoft Mail Server: opening screen

The "Listening on port …" messages confirm that:

- SMTP Server started: the server that ***pCOWeb*** will send the messages to be running.
  IMPORTANT: if another SMTP manager is already active, port 25 will be busy and the service will not be started. Close the other manager.
- POP3 Server started: the server that the e-mail client will download the messages from is running.
  IMPORTANT: if another POP3 manager is already active, port 110 will be busy and the service will not be started. Close the other manager.
- Finger Server started: optional service not used by ***pCOWeb***; ignore.
- Web Server started: the server to access the MAILBOX from an Internet Browser is running.
  IMPORTANT: if port 80 is busy, the web server will not be started; the Skype application, for example, uses port 80; another port can be selected in the configuration (for access via browser the port will need to be specified), or alternatively close Skype or any other applications that occupy port 80.

CONFIGURING THE SERVER

1. SET THE DOMAINS

   INFORMATION
   - The part of an e-mail address to the right of the "@"is called the domain (name@domain).
   - More than one domain can be added to ArGoSoft.
   - The server will only receive the messages whose domain corresponds to one domain specified on the settings.

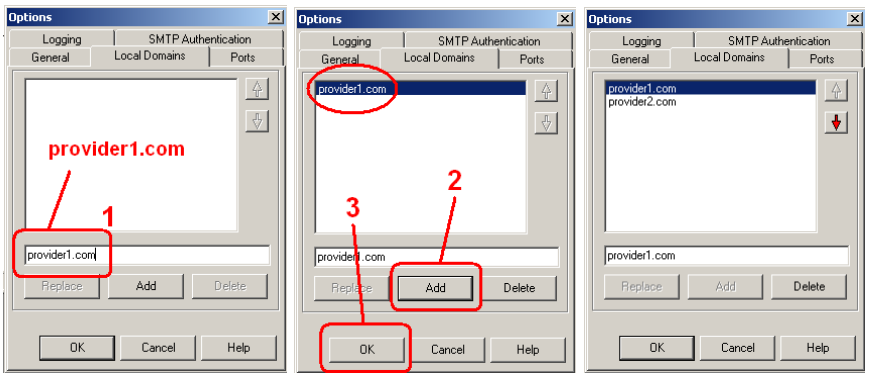


Figure D.b - ArGoSoft Mail Server: setting the domains

   PROCEDURE
   Click point 1 in Figure D.a above; with reference to the example settings illustrated for ***pCOWeb*** in Figure 4.p on page 24, enter the two domains (see Figure D.b above):

```
provider1.com
provider2.com
```

2. CREATE THE MAILBOX
   Click point 2 in Figure D.a above. Create two MAILBOXES, entering the User Name and Password (in Figure D.c on page 67 "pcoweb_heater_pCO" / "password" and "supervisor" / "password" have been used). This information will only be used by the client (for example Microsoft Outlook) to receive the messages.

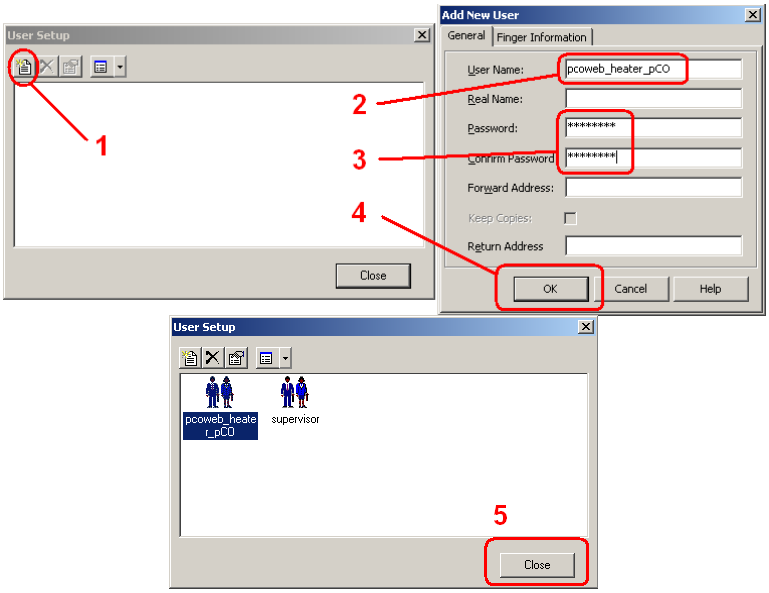


Figure D.c - ArGoSoft Mail Server: creating a MAILBOX

IMPORTANT When, in Figure 4.p on page 24, the properties are set for sending the e-mail notifications from ***pCOWeb***, if ArGoSoft is used, it is mandatory not to specify the login Username / Password (leave the corresponding fields empty).

NOTE Even if the pcoweb_heater_pCO@provider1.com MAILBOX is not created, ***pCOWeb*** still sends the messages to supervisor@provider2.com; the first MAILBOX has simply been proposed as an example for any messages sent to ***pCOWeb***. These messages are not managed by ***pCOWeb*** but must be displayed by browser or downloaded by a client.

ACCESSING THE SERVER VIA WEB

The ArGoSoft server features access to MAILBOXES via web pages.
To do this, simply open an Internet browser that can access the computer where ArGoSoft is running, and type the name or the IP address of the computer; if the same computer is running the browser and ArGoSoft, simply enter the address 127.0.0.1.
Once the access page has been displayed, select the "Login" button and enter the Username ("pcoweb_heater_pCO") and Password ("password") set for the MAILBOX.

NOTE If when installing ArGoSoft, port 80 was occupied and a different port is used (for example 81), the port must be specified as follows:

```
<name or IP address>:81
```

Example: `127.0.0.1:81`

STOPPING THE ArGoSoft SERVER

After ArGoSoft has been started and the server has been configured, the program can be closed. The server will remain operating.
To also stop the server, open ArGoSoft again and proceed as shown in Figure D.d.

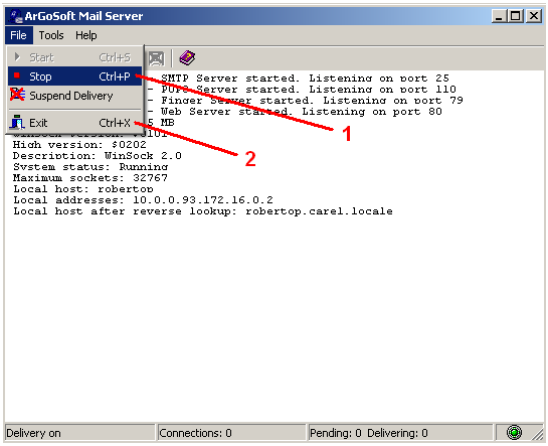


Figure D.d - Stopping the ArGoSoft Mail Server

CONFIGURING THE Microsoft Outlook CLIENT

To be able to use an e-mail client (for example Microsoft Outlook) to get the messages delivered, for instance, to the supervisor@provider2.com MAILBOX on the server, the client needs to be configured.

PROCEDURE

- From the Microsoft Outlook menu bar, choose:
  - "Tools"
  - "Accounts…"
  - "Add an e-mail account"
- Choose the type of server: POP3

- Enter the information :

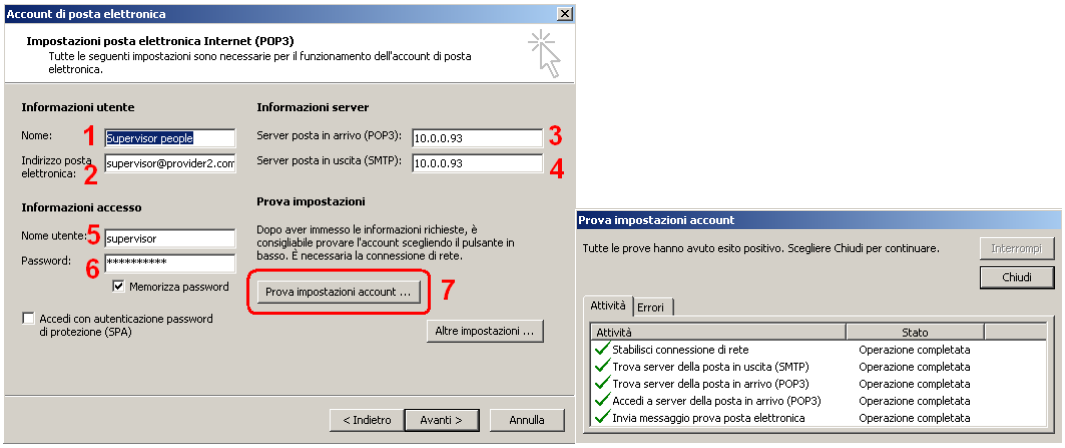


Figure D.e - ArGoSoft Mail Server: configuring Microsoft Outlook

1 - Name: `Supervisor people`
2 – E-mail address: `supervisor@provider2.com`
3 and 4 – Incoming mail server (POP3) and Outgoing mail server (SMTP): name or IP address of the computer where the ArGoSoft server has been installed.

NOTE If the computer is different from the one running Outlook, the two computers must be able to exchange data. It is recommended to use the same computer. In this case, enter 127.0.0.1 (internal address of the computer visible to all the internal applications).

5 – User name: as set for the ArGoSoft MAILBOX: "`supervisor`".
6 – Password: password set when creating the MAILBOX: "`password`".
7 – Test account settings...: select this button to test the Microsoft Outlook settings: a test message will be sent and then received by the MAILBOX.

At this stage, the ArGoSoft server is ready to receive e-mails from ***pCOWeb*** and deliver them to Outlook.

APPENDIX E FileZilla Server: A FREEWARE FTP SERVER

Various FTP servers are available on the Internet.
Below is a description of the FileZilla Server freeware (version 0.9.23beta) downloadable at the link:
`http://filezilla-project.org/`.

When installation has been completed, run FileZilla Server and the FileZilla Server Interface application for making the settings: the authentication dialogue box for connection to the server is displayed: select Ok.

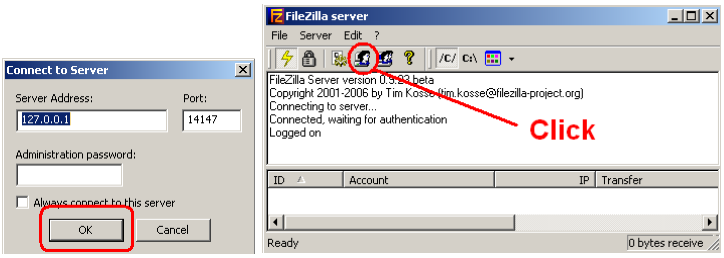


Figure E.a - FileZilla Server Interface: authentication dialogue box and opening screen

CONFIGURING THE USERS

INFORMATION

When ***pCOWeb***, using the FTP PUSH function on page 24, connects to a FTP server to send a file, authentication is performed by sending the Username and Password; the user must already be configured in the server, otherwise the connection will fail. The example below shows how to set a user on FileZilla Server; the procedure is the same for setting other users.

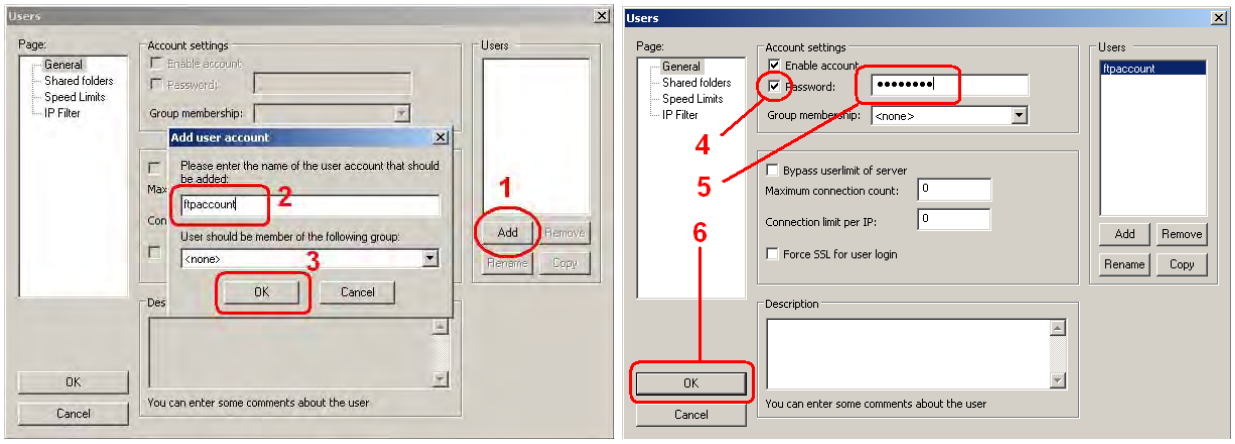


Figure E.b - FileZilla Server Interface: creating the "ftpaccount" user

1. Open the user configuration page (see Figure E.a above, click the Users icon);
2. With reference to Figure E.b above (points 1, 2, 3), type the Username of the new user (for the FTP PUSH example illustrated on page 24, enter "`ftpaccount`");
3. Again with reference to Figure E.b above (points 4, 5, 6), set the access Password for the "ftpaccount" user created above (for example "`password`").
4. With reference to Figure E.c (points 7, 8, 9, 10), set the shared directory for "ftpaccount" that ***pCOWeb*** will be able to copy the files into.

The settings are now complete. The FileZilla Server Interface can now be closed; FileZilla Server will keep running.

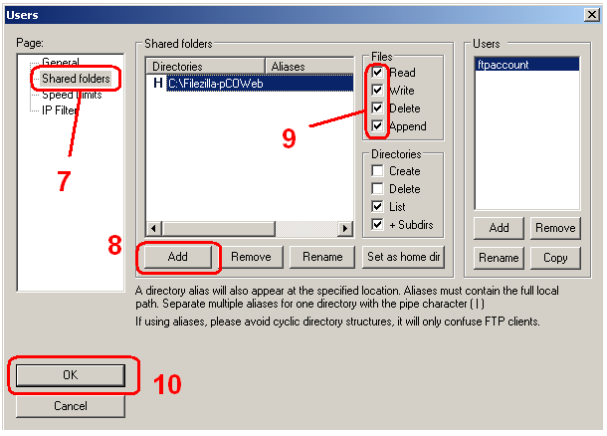


Figure E.c - FileZilla Server Interface: choosing the shared directory for "ftpaccount"

STOPPING / RESTARTING FileZilla Server

If needing to stop or restart the server, from the Windows Programs menu, procedures as shown in Figure E.d.

Figure E.d - FileZilla Server: stopping / rebooting

APPENDIX F Trap Receiver: A SIMPLE TRAP / INFORM RECEIVER

Various TRAP receivers are available on the Internet.
Below is a description of the Trap Receiver freeware (version 5.00) downloadable at the link:
http://www.ncomtech.com/trapreceiver.html.

After installation, when starting Trap Receiver requires confirmation to start the TrapRcvr service (Figure F.a).
When started no configuration is required. Figure F.b shows the Trap Receiver window highlighting a Trap message that has been received; double click to see the details of the TRAP (INFORM).

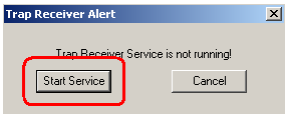


Figure F.a - Trap Receiver: confirming the start of the service

STOPPING / RESTARTING Trap Receiver

The TrapRcvr service can be stopped from Windows: Start / Settings / Control Panel / Administrative Tools / Services: search for the "TrapRcvr" service and click the right mouse button on the row corresponding to the service and choose "Stop".
The service will be automatically restarted when Trap Receiver is next run.

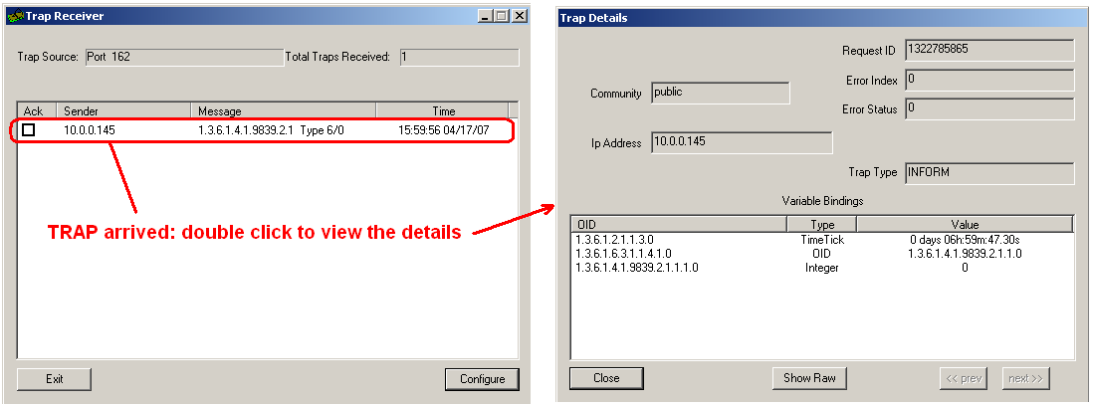


Figure F.b - Trap Receiver: confirming the start of the service – details of the TRAP messages

APPENDIX G CAREL TAGS FOR *pCOWeb* HTML PAGES -THE PW_DEMO.HTML PAGE

Each CAREL tag entered on an html page is used to read or modify the ***pCO*** controller data or the ***pCOWeb*** settings. This makes the html page dynamic, that is, it contains the values of variables.

The tags are managed by an internal program on ***pCOWeb*** with tag parser functions, called "pcotagfilt". In fact, the web server on ***pCOWeb*** has been programmed to recognise the following row in an html page (the row position is irrelevant):

```
<!--tagparser="/pcotagfilt"-->
```

following which the web server searches for the "pcotagfilt" program in the default root position for the web server ("/usr/local/root"), and then starts it, sending all the contents of the html page.

The "pcotagfilt" program recognises the CAREL tags, replaces them with the results, and returns the resulting html page.

IMPORTANT If the row

```
<!--tagparser="/pcotagfilt"-->
```

is not included in the html page, the CAREL tags will not work, and will be shown on the PC as they are written.

NOTE Browsers such as Internet Explorer, Mozilla Firefox and others create a copy of the pages visited in their cache, and when the page is next called the contents may be loaded from the cache, rather than requesting an update from ***pCOWeb***. This means that at times the desired result is not displayed, above all with dynamic pages such as those used on ***pCOWeb***.
The browsers, nonetheless, feature of a specific command to force them to update the pages from the web server; for example, by using F5 both Internet Explorer and Mozilla Firefox update the page, but by using Ctrl+F5 they do it without using the cache contents.
Ctrl+F5 should be used whenever there is doubt that the result may haven been loaded from the cache on the PC.

CAREL TAGS FOR HANDLING THE *pCO* VARIABLES

"var": read / write a variable

Used to read or write the value of a ***pCO*** controller supervisor variable.

SYNTAX
<%var(0, *VariableType*, *VariableIndex*, *[MinValue]*, *[MaxValue]*)%>

where:

- 0 (zero): required; reserved for future extensions;
- *VariableType*: 1: Digital, 2: Analogue, 3: Integer;
- *VariableIndex* (1 to 207 for Carel protocol/5000 for ModBus Extended protocol): choose the variable;
- *[MinValue] (Int.: -32768 to 32767 step 1, An.: -3276.8 to 3276.7 step 0.1),* optional when writing, no affect when reading: ***pCOWeb*** will not send the ***pCO*** a value less than *MinValue*;
- *[MaxValue] (Int.: -32768 to 32767 step 1, An.: -3276.8 to 3276.7 step 0.1),* no affect when reading, optional when writing: ***pCOWeb*** will not send the ***pCO*** a value greater than *MaxValue*.

READING
Returns the value of the variable saved in a table on ***pCOWeb***. The table is updated based on the variations that ***pCO*** sends to ***pCOWeb*** around 3 times a second, as displayed by the green flashes of the Status LED.
The result of the read operation may be:

- value of the variable;
- or: "U" (Undefined): indicates that the value of the variable contained in the table on the ***pCOWeb*** is not valid.
  Possible causes of the value "U":
  - ***pCOWeb*** has not yet received a value for the variable from the ***pCO***; wait around one minute and retry;
  - alternatively: communication between the ***pCOWeb*** and the ***pCO*** is interrupted (the Status LED on the ***pCOWeb*** flashes red, all the variables are equal to "U" on the page shown in Figure 9.b on page 41): check the correspondence of the communication parameters between ***pCOWeb*** and the ***pCO*** application (see 9.5 on page 44);
  - alternatively: the address of the variable is outside of the range managed by the ***pCO*** BIOS firmware (not a recent version); In the example shown in Figure 9.b on page 41, the ***pCO*** only manages the first 199 digital variables, the first 127 analogue variables and the first 127 integer variables.

Example (Figure G.a on page 72)
Reading the value of digital variable 5:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<!--tagparser="/pcotagfilt"-->
<head>
  <meta content="text/html; charset=ISO-8859-15" http-equiv="content-type">
```

```
  <title>pCOWeb Configuration</title>
</head>
<body>
<div style="color: red">Variable Digital 5 value is: <%var(0,1,5)%></div>
</body>
</html>
```

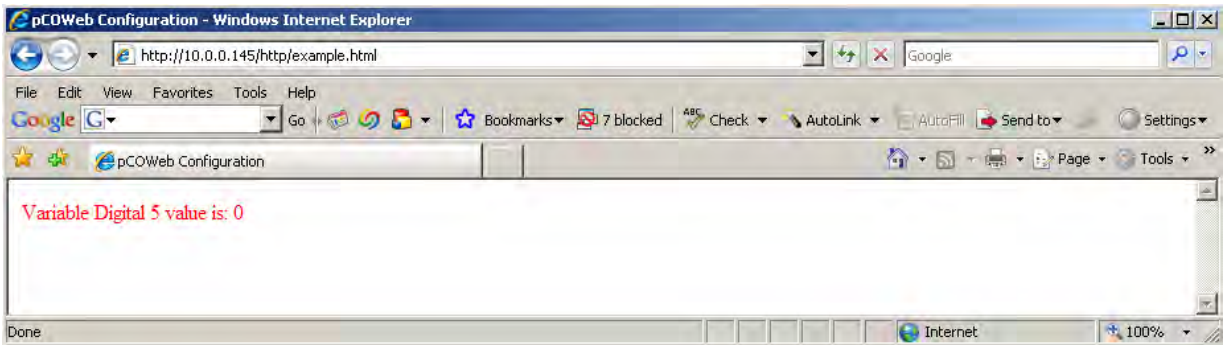


Figure G.a – The "var" TAG used for reading

WRITING

To enter a variable write operation on an html page, use the *<form>* statement (for the syntax of <form>, see the documents on the HTML language at http://www.htmlhelp.com/reference/wilbur/block/form.html). Inside the form section, the **var** tag is used together with the "**script**" statement.

Example (Figure G.b below and Figure G.c on page 73)

Writing a new value for the integer variable with index 9 (note: to try this example on ***pCOWeb***, replace index 9 with the index of a read/write variable for the application loaded on the ***pCO***).

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<!--tagparser="/pcotagfilt"-->
<head>
  <meta content="text/html; charset=ISO-8859-15" http-equiv="content-type">
  <title>pCOWeb Demo Page</title>
</head>
<body bgcolor='#ffffff'>
<h1 style="text-align: center">pCOWeb Demo Page</h1>
<br>
<form method="GET" action="example.html">
<input type="text" name="?script:var(0,3,9,-32768,32767)" value="<%var(0,3,9)%>">
<input type="submit" value="Submit">
</form>
</body>
</html>
```

The **?script:var(0,3,9,-32768,32767)** section places a text box on the page for entering the value using the keyboard.
The **<%var(0,3,9)%>** section fills the text box with the value of the variable; if omitted, the value will not be indicated.

Figure G.b – The "var" TAG used for writing

After having entered a new value, selecting the Submit button sends ***pCOWeb*** the value entered for the variable and a new request for the page specified in the "form" statement ("example.html").
pCOWeb will perform the following operations:

1. send the value entered to the ***pCO*** (for multiple variables, it will send all the values entered);
2. wait to receive the echo of the value sent by the ***pCO*** (for multiple variables, it will wait for the echo of all the values);
3. when the echo is received, the value will be written to the table on the ***pCOWeb*** and the page will be returned to the PC (for multiple values, it will wait for the last echo to be received); the fields will contain the values received as echoes from the ***pCO***.

Figure G.c – The "`var`" TAG used for writing: write completed

NOTE The ***pCO*** does NOT send an echo in the following cases:

- the ***pCO*** is sent the same value as already saved for the variable;
- one of the situations in which "var" returns "U" (see above, description of "var" for reading).

If waiting for some echo, ***pCOWeb*** will return the page to the PC after a Timeout of 10 seconds from the reception of the last echo; the fields relating to the variables for which no echo has been received will contain the values saved prior to the variations being sent. For further details see APPENDIX C on page 65.

"`setres`": outcome of a write page

Returns a value that depends on the result of the last access request to the ***pCO*** supervisor variables in which **`var`** commands are used (see above).

SYNTAX
<%setres('*UndefinedText*', '*OkText*', '*TimeOutText*')%>

The three customisable fields are returned in the following cases:

- *UndefinedText:* the ***pCO*** has not been sent a request for the variation of the variables;
- *OkText*: ***pCOWeb*** has received all the echoes for the variations sent to the ***pCO***;
- *TimeOutText*: ***pCOWeb*** not has received all the expected echoes.

IMPORTANT The result of setres does NOT represent the correct writing or reading of a ***pCO*** supervisor variable; in fact, the following special cases exist:

- for the write request, the ***pCO*** returns an echo even when the variable is in read-only and the value sent to the ***pCO*** is different from the one in the application; the echo returned sends the unchanged value contained in the ***pCO*** application;
- if the ***pCO*** receives the write request for a variable with a value that is the same as the current one, no echo will be generated for that variable.

For these reasons, this tag should only be used to check the operation of the HTML pages during their construction.
For further information on communication with the ***pCO***, see APPENDIX C on page 65.

Example (Figure G.d on page 74)
Displaying the result of the write operation:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<!--tagparser="/pcotagfilt"-->
<head>
  <meta content="text/html; charset=ISO-8859-15" http-equiv="content-type">
  <title>pCOWeb Demo Page</title>
</head>
<body bgcolor='#ffffff'>
<h1 style="text-align: center">pCOWeb Demo Page</h1>
<br>
<form method="POST" action="example.html">
<input type="text" name="?script:var(0,3,9,-32768,32767)" value="<%var(0,3,9)%>">
<input type="submit" value="Submit">
</form>
Operation result: <%setres('Undefined','Ok','Timeout')%>
</body>
</html>
```

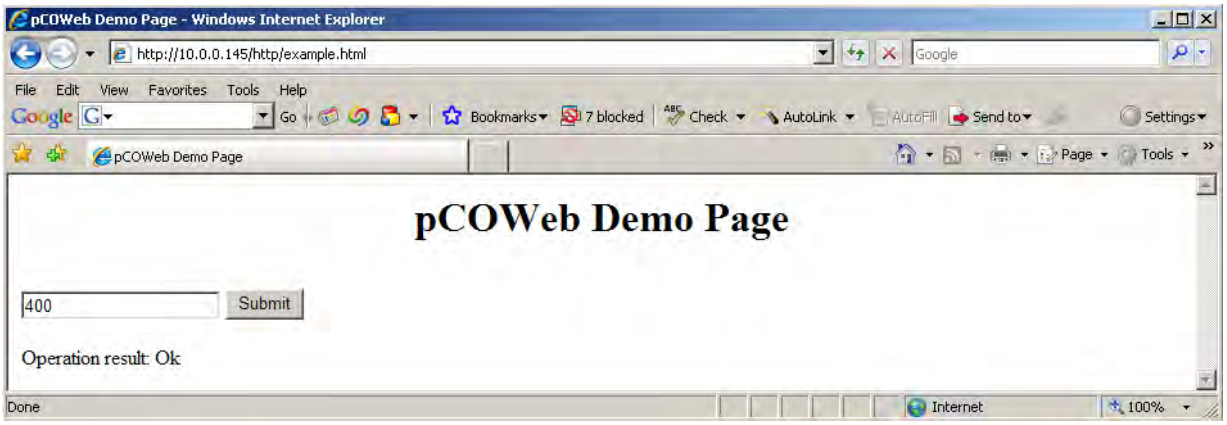


Figure G.d – The "setres" TAG: example

CAREL TAGS FOR HANDLING THE *pCOWeb* CONFIGURATION FILES

pCOWeb saves the user settings in a number of files (paragraph 9.1 on page 40), in the following format:

```
<parameter>=<value>
```

An example of these files (SNMP TRAP configuration, 'snmptrap' file) is as follows:

```
r1_enabled=1
r1_trigger=0
r1_dest1=1
r1_dest2=0
r1_dest3=0
r1_dest4=0
r1_trapoid=1.3.6.1.4.1.9839.2.1.1.0
r1_ack=0
r1_time=1
host1=10.0.0.131
community1=carel
enabled1=1
enabled2=0
enabled3=0
enabled4=0
pcoprotfail_enab=0
pcoprotfail_ack=0
pcoprotfail_time=1
r1_valoid1=1.3.6.1.2.1.1.5.0
```

The following tags are available for handling the configuration files:

- "getdb": read a parameter from file;
- "setdb": write a parameter to file;
- "checkdbsel": search for a parameter in a file and check the value (useful for html "select" tools);
- "checkdbradio": search for a parameter in a file and check the value (useful for html "radiobutton" tools).

"getdb": read a *pCOWeb* parameter from a file

Returns the value of a *pCOWeb* parameter read from a specified file.

SYNTAX
<%getdb('*Filename*', '*ParameterName*')%>

IMPORTANT: the names of the *pCOWeb* files are case sensitive.
The file is searched by default in /usr/local/root/flash/etc/sysconfig/, which contains the *pCOWeb* configuration files. To use a new file of parameters for new applications, a path other than the default can be specified, always starting from root ("/"); nonetheless, it is recommended to save the files in the default path so as to group all the *pCOWeb* parameter settings in the same location.

Example (Figure G.e en page 75)
The 'commcfg' file contains the *pCOWeb* and *pCO* communication configurations. The row that defines the baud rate is:

```
speed=19200
```

The following html page:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<!--tagparser="/pcotagfilt"-->
<head>
  <meta content="text/html; charset=ISO-8859-15" http-equiv="content-type">
  <title>pCOWeb Demo Page</title>
</head>
<body bgcolor='#ffffff'>
<h1 style="text-align: center">pCOWeb Demo Page</h1>
<br>
BAUD RATE = <%getdb('commcfg','speed')%><br>
</body>
</html>
```

will generate the result shown in Figure G.e below.

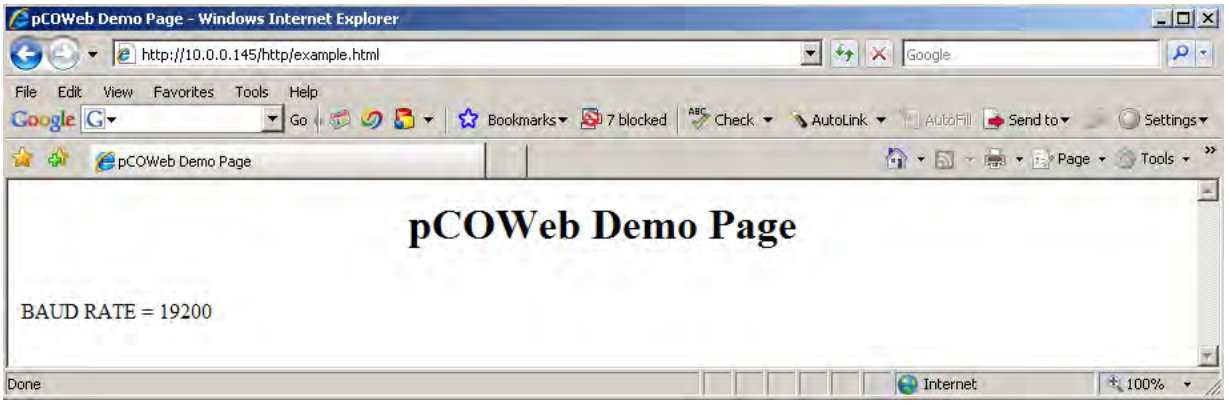


Figure G.e – The "getdb" TAG: example

"setdb": write a *pCOWeb* parameter to a file

Writes or modifies the value of a ***pCOWeb*** parameter to a specified file.

SYNTAX
?script:setdb('*Filename*', '*ParameterName*')

The same observations as made for the getdb tag also apply here.

Example (Figure G.f on page 76)
Refer to the same 'commcfg' file as used for the getdb tag.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<!--tagparser="/pcotagfilt"-->
<head>
  <meta content="text/html; charset=ISO-8859-15" http-equiv="content-type">
  <title>pCOWeb Demo Page</title>
</head>
<body bgcolor='#ffffff'>
<h1 style="text-align: center">pCOWeb Demo Page</h1>
<br>
<form method="POST" action="example.html">
      <input type="text" name="?script:setdb('commcfg','speed')"
      value="<%getdb('commcfg','speed')%>">
      <input type="submit" value="Submit">
</form>
</body>
</html>
```

Figure G.f – The "setdb" TAG: example

"checkdbsel": check whether the value of a parameter in a file is "selected"

Returns the "selected" string if the value of the parameter contained in the file indicated by the tag is the same as indicated in the tag itself, otherwise it returns an empty string.

SYNTAX
<%checkdbsel('*Filename*', '*ParameterName*', '*Value*')%>

Used in the construction of drop-down menu with the html "select" command, selects the current value of the parameter.

Example (Figure G.g below)
With reference to the 'commcfg' configuration file already seen for the previous tags; a drop-down menu is created similar to the one used on the configuration page shown in

Figure 9.l on page 453, so as to allow the selection between the values 9600, 19200 and 38400; if the parameter in the file is equal to 19200, the menu will show that value as already being selected.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<!--tagparser="/pcotagfilt"-->
<head>
  <meta content="text/html; charset=ISO-8859-15" http-equiv="content-type">
  <title>pCOWeb Demo Page</title>
</head>
<body bgcolor='#ffffff'>
<h1 style="text-align: center">pCOWeb Demo Page</h1>
<br>
<form method="POST" action="example.html">
   <select name="?script:setdb('commcfg','speed')">
      <option value="9600" <%checkdbsel('commcfg','speed','9600')%>>9600</option>
      <option value="19200" <%checkdbsel('commcfg','speed','19200')%>>19200</option>
      <option value="38400" <%checkdbsel('commcfg','speed','38400')%>>38400</option>
   </select>
   <input type="submit" value="Submit">
</form>
</body>
</html>
```

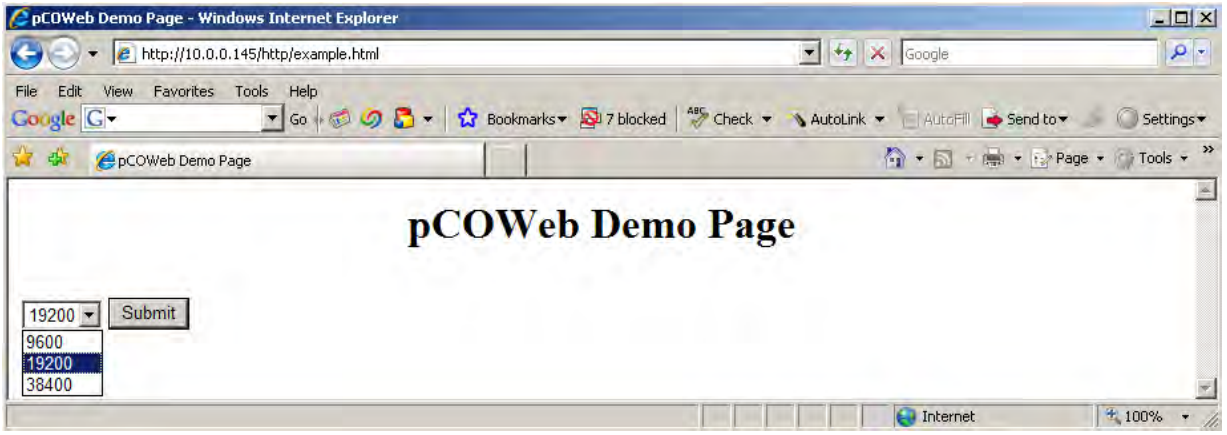


Figure G.g – The "checkdbsel" TAG: example

"checkdbradio": check whether the value of a parameter in a file is "checked"

This works in exactly the same way as <%checkdbsel%>, but returns **"checked="checked""** instead of **"selected"** and is used for "radio" buttons, that is, those buttons in which one selection excludes the others.

Example (Figure G.h below)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<!--tagparser="/pcotagfilt"-->
<head>
  <meta content="text/html; charset=ISO-8859-15" http-equiv="content-type">
  <title>pCOWeb Demo Page</title>
</head>
<body bgcolor='#ffffff'>
<h1 style="text-align: center">pCOWeb Demo Page</h1>
<br>
<form method="POST" action="example.html">
   <input type="radio" name="?script:setdb('clockcfg','clock_sync_enabled')" value="0"
id='en clock a' onclick="Check EnVar('1')"
<%checkdbradio('clockcfg','clock_sync_enabled','0')%>>Disabled
   <input type="radio" name="?script:setdb('clockcfg','clock_sync_enabled')" value="1"
id='en_clock_b'
onclick="Check_EnVar('1')"<%checkdbradio('clockcfg','clock_sync_enabled','1')%>>Enabled
   <input type="submit" value="Submit">
</form>
</body>
</html>
```

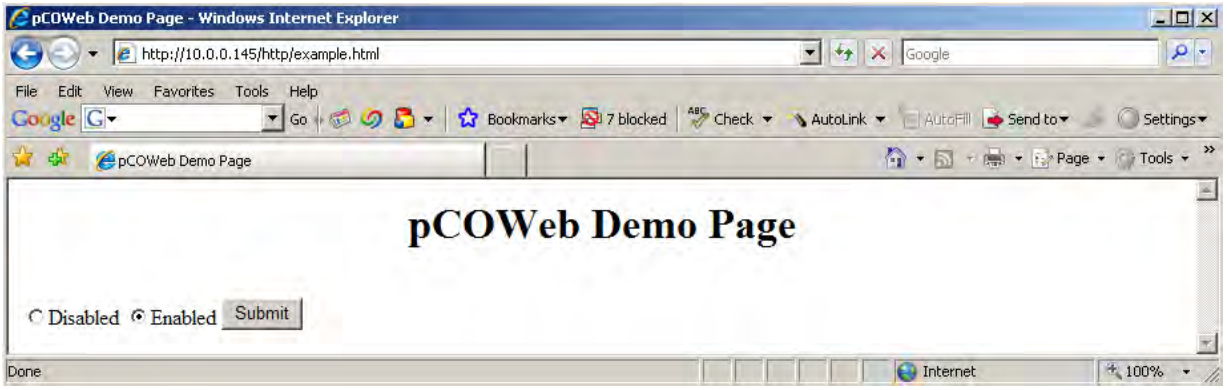


Figure G.h – The "checkdbradio" TAG: example

INFORMATION TAGS: macaddress, fw_release, bootvalues, ipaddr_eth0, date

These return information relating to ***pCOWeb***.

SYNTAX AND MEANINGS

<%macaddress%>	MAC address
<%fw_release%>	***pCOWeb*** firmware version
<%bootvalues%>	Type of parameters used when starting: User/Bootswitch, see 3.1.2 on page 12
<%ipaddr,eth0%>	IP address of the eth0 interface; use eth0:1, eth0:2, eth0:3 for the others
<%date%>	Current time on the ***pCOWeb*** in the format 1970-01-03 20:27

Example (Figure G.i en page 78)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<!--tagparser="/pcotagfilt"-->
<head>
  <meta content="text/html; charset=ISO-8859-15" http-equiv="content-type">
  <title>pCOWeb Configuration</title>
</head>
<body>
<div>The ethernet Mac Address is: <%macaddress%></div>
<div>The pCOWeb firmware release is: <%fw_release%></div>
```

```
<div>The pCOWeb started using: <%bootvalues%></div>
<div>pCOWeb IP address (Eth0): <%ipaddr,eth0%></div>
<div>pCOWeb date/time: <%date%></div>
</body>
</html>
```

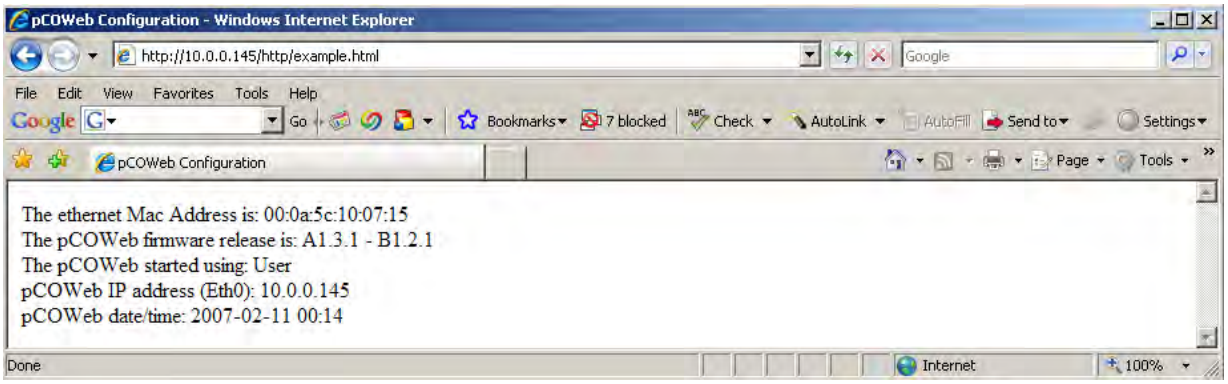


Figure G.i – The information TAGS: example

SPECIAL TAGS

These are used in the configuration web pages; their use is closely related to the operating system, and incorrect usage may cause instability on the ***pCOWeb***.

- "`factdefcfg`": extracts from non-volatile read-only memory the factory values of the parameters used by pressing the button when starting the ***pCOWeb*** (paragraph 3.1.2 on page 12).
  Used in the link to "View factory bootswitch parameters" (see Figure 9.g on page 43).
- "`rcresult`": directly returns the result of certain operations from the operating system. This is particularly delicate, as if not used correctly may require the ***pCOWeb*** to be rebooted.
- "`env`": returns all the information available from the ***pCOWeb***.
  This takes a long time to run; if when running (that is, until the web page is returned) other web pages are requested, ***pCOWeb*** will need to be rebooted. To use this, type the following command directly into the address field in Internet Explorer:
  `http://172.16.0.1/config/result.html?%3fscript%3arccmd%28%27do_log%27%29`
  replacing the address 172.16.0.1 with the IP address of ***pCOWeb*** in question.
- "`do_flashroom`": returns the size of the remaining free space in the area of non-volatile memory reserved for the web pages.
  One example of this is "View used / free disk space" on the configuration page (see Figure 9.h on page 43);
- "`do_ifconfig`": returns the current configuration of the Ethernet network, including the IP address obtained from the server in DHCP mode; one example is "View network configuration" on the configuration page (see Figure 9.g on page 43).

TYPICAL ERRORS INVOLVING THE TAGS

These are some typical errors that may be committed when using the tags inside the web pages.

- The tag is not replaced by the corresponding value.
  Solution:
  - check that the following row is included in the html page:
    `<!--tagparser="/pcotagfilt"-->`
  - check that the editor used to create the page does not replace the special characters used in the tags with the corresponding html code; for example, the "<" and ">" characters may have been replaced with "`&lt`" and "`&gt`" respectively.
- The web pages containing tags are not displayed.
  Causes: the syntax of the tag has not been observed.
  Solution: make sure the "`%`" characters have been entered correctly inside the tags; the last tag on the page is often forgotten.
- The value displayed on the web pages does not change when the variable changes.
  Causes: the web page has been made by copying the source of another page (for example, displaying "Source code" in the browser); it does not work because the page obtained does not contain the tags, but rather the values that pcotagfilt has already replaced them with.

EXAMPLE: THE PW_DEMO.HTML PAGE

Below is a description of the 'pw_demo.html' page, available by accessing ***pCOWeb*** via FTP at the path /usr/local/root/config.
For further information on the creation of the HTML pages, go to http://ksa.carel.com/, under the section ***pCOWeb***.

The following row is useful for having the page validated by automatic software, so as to check the correct usage of the language: it declares the type of document, and the corresponding standard and version.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
```

The following row opens the section that will contain the HTML statements. This is closed by </html>.

```
<html>
```

The following rows in standard html language simply represent comments.
The last of the three rows is nonetheless essential for the ***pCOWeb*** pages and MUST be included in the page (see the start of APPENDIX G on page 71).
This tells ***pCOWeb*** which translator must be used for the proprietary tags. ***pCOWeb*** currently features just one tag translator for CAREL tags, so the row is always:
`<!--tagparser="/pcotagfilt"-->`
If this row is not entered, no CAREL proprietary tags will work.

```
<!-- The tag filter must be specified in the first 10 lines,
     after the '<html>' statement, with the syntax: -->
<!--tagparser="/pcotagfilt"-->
```

NOTE Unlike the comment, the position of the row in the page is irrelevant.

The following row opens the HEAD section, which will contain some special html functions that have no effect on the display of the page.

```
<head>
```

The `<meta...` row tells the browser to use a certain set of characters and other properties.
The `<link...` row loads the style sheets that will be used for formatting the page.
The `<title>title... </title>` row attributes the title to the html page window opened by the browser.

```
  <meta content="text/html; charset=ISO-8859-15" http-equiv="content-type">
  <link rel="stylesheet" href="style.css" type="text/css">
  <title>pCOWeb Demo Page</title>
```

The following row closes the HEAD section.

```
</head>
```

The following row selects the colour used for the background of the page.

```
<body bgcolor='#ffffff'>
```

The following row writes the text **pCOWeb Demo Page**, in h1 font size, with centred horizontal alignment.
This is followed by the line break,
.

```
<h1 style="text-align: center">pCOWeb Demo Page</h1>
<br>
```

The following row tells the browser that it must draw an area for acquiring data entered by the user, and, when confirmed, send the data to the "pw_demo.html" page.
The list of controls for this area ends with the </form> command.
The command defines the start of an area with centred horizontal alignment, the <table> command places the subsequent objects in a table, the <tr> command defines a new row in a table, the <td> command defines a new cell in a table.
Note that the page has the structure as a general table, in turn containing three tables, each of which contains 10 ***pCO*** variables.
For each variable, the html "input" command is used together with the CAREL "var" tag.

```
<form method="GET" action="pw_demo.html">
<center>
<table> general table containing the following three tables
  <tr>
```

`<td>` first cell in the general table: table of digital variables

```
    <table style="font-size: 12px; padding: 0px; margin: 0px; padding-right: 30px; border-right: 1px solid #cccccc" cellpadding="0">
      <tr>
        <td style="font-size: 14px; font-weight: bolder; height: 40px; text-align: center" colspan="2">Digital Variables</td>
      </tr>
      <tr>
        <td style="width: 90px">Var 1:</td>
        <td><input type="text" name="?script:var(0,1,1,0,1)" value="<%var(0,1,1)%>"></td>
      </tr>
```

```
        <tr>
          <td>Var 2:</td>
          <td><input type="text" name="?script:var(0,1,2,0,1)" value="<%var(0,1,2)%>"></td>
        </tr>

        ……

        <tr>
          <td>Var 10:</td>
          <td><input type="text" name="?script:var(0,1,10,0,1)" value="<%var(0,1,10)%>"></td>
        </tr>
      </table>

    </td>
```

`<td>` second cell in the general table: table of analogue variables

```
      <table style="font-size: 12px; padding: 0px; margin: 0px; padding-left: 30px; padding-right: 30px; border-right: 1px solid #cccccc"
cellpadding="0">
        <tr>
          <td style="font-size: 14px; font-weight: bolder; height: 40px; text-align: center" colspan="2">Analog Variables</td>
        </tr>
        <tr>
          <td style="width: 90px">Var 1:</td>
          <td><input type="text" name="?script:var(0,2,1,-3276.8,3276.7)" value="<%var(0,2,1)%>"></td>
        </tr>

        ……

        <tr>
          <td>Var 10:</td>
          <td><input type="text" name="?script:var(0,2,10,-3276.8,3276.7)" value="<%var(0,2,10)%>"></td>
        </tr>
      </table>

    </td>
```

`<td>` third cell in the general table: table of integer variables

```
      <table style="font-size: 12px; padding: 0px; margin: 0px; padding-left: 30px" cellpadding="0">
        <tr>
          <td style="font-size: 14px; font-weight: bolder; height: 40px; text-align: center" colspan="2">Integer Variables</td>
        </tr>
        <tr>
          <td style="width: 90px">Var 1:</td>
          <td><input type="text" name="?script:var(0,3,1,-32768,32767)" value="<%var(0,3,1)%>"></td>
        </tr>

        ……

        <tr>
          <td>Var 10:</td>
          <td><input type="text" name="?script:var(0,3,10,-32768,32767)" value="<%var(0,3,10)%>"></td>
        </tr>
      </table>
    </td>


  </tr>
</table>

</center>
```

`<br>` line break

The following rows are comments and consequently are not active

```
<!--pCO set result:  <img src="/config/img/<%setres('undefined','green','red')%>.gif" alt="<%setres('undefined','correct','error
timeout')%>" align="bottom"><br><br>-->
```

The following row is an example of the “setres” tag used to colour the texts

```
pCO set result: <div style="display: inline; font-weight: bold; color: <%setres('gray">Undefined','green">Communication ok','red">Timeout
Error')%></div><br>
```

Addition of the “Submit” button

```
  <input style="width: 100px; margin-left: 20px; margin-top: 10px" type="submit" value="Submit">
</form>
```

End of the page

```
<div id="back-bottom">
(C) Copyright 2003-2010 by Carel Industries S.r.l.., Brugine (PD) - Italy. All rights reserved. Contact: <a
href="mailto:pcoweb@carel.com">pcoweb@carel.com</a><br>
Linux porting and software implemented by DVE Progettazione Elettronica, Varese (Italy). <a
href="mailto:info@dveprojects.com">info@dveprojects.com</a><br>
</div>
</body>
</html>
```

APPENDIX H Library “pw_ajax.js” and CGI “xml.cgi”

Overview

To cope with the increasing requests to integrate ***pCOWeb*** in enterprises system, it had became mandatory for the ***pCOWeb*** to integrate a standard way of communicate using the HTTP protocol, that is why the ***pCOWeb*** now is able to send information using XML over HTTP (which is the standard way to use the XML).

Minimum Requirements

This application note is addressed to people with the following knowledge:

- Discrete knowledge of HTML
- Discrete knowledge of JAVASCRIPT

How to read variables

Starting from version A135 (and with major improvements in version A142) ***pCOWeb*** implements a library called “pw_ajax.js” that allows users to read the values of the variables without using the standard ***pCOWeb*** HTML tags.

Example (need to copy the file “pw_ajax.js” from the config folder to the same folder of the html file):

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
  <meta content="text/html; charset=ISO-8859-15" http-equiv="content-type">
  <title>Happy Island</title>
  <script src="pw_ajax.js" type="text/javascript"></script>

  <script type="text/javascript">

  // arrays used by the javascript library, DO NOT USE OTHER NAMES!
  digitals = new Array;
  integers = new Array;
  analogs  = new Array;

  var timestamp;

function getVariables() {

  // N.B. The function getParams is defined in the file pw_ajax.js
  // This reads variables Digitals 1 and 2, Integers 11 and 12 and
  // analogues 101 and 102
  // and then copy them into the arrays declared here above to the
  // cells with the index equal to their supervisry index
  // e.g. analogue 101 is copied into the array called "analogs" to the index 101
  // The string is composed by: TYPE|START_INDEX|END_INDEX|TYPE2|START_INDEX2...
  // It can contain up to 5 different ranges of variables
  getParams('/config/xml.cgi','D|1|2|I|11|12|A|101|102');

function parseResults() {

  // The following instructions assign the values of the variables
  // in the arrays to an element of the HTML page to be visualized

  document.getElementById("digital1").innerHTML=digitals[1];
  document.getElementById("digital2").innerHTML=digitals[2];
  document.getElementById("integer11").innerHTML=integers[11];
  document.getElementById("integer12").innerHTML=integers[12];
  document.getElementById("analog101").innerHTML=analogs[101];
  document.getElementById("analog102").innerHTML=analogs[102];

  document.getElementById("currentTime").innerHTML=timestamp;

  // Refresh the variables every 5 seconds (the time is ms)
  timer=setTimeout('getVariables()',5000);

}
</script>
</head>

<body onLoad="getVariables();">
<!-- The onLoad is necessary to load the variables as soon as the page loads -->

Current time is: <span id="currentTime"></span><br>

<table>
  <tr>
    <td>Digital 1 is:</td><td><span id="digital1"></span></td>
  </tr>
    <td>Digital 2 is:</td><td><span id="digital2"></span></td>
  </tr>
    <td>Integer 11 is:</td><td><span id="integer11"></span></td>
  </tr>
    <td>Integer 12 is:</td><td><span id="integer12"></span></td>
  </tr>
    <td>Analogue 101 is:</td><td><span id="analog101"></span></td>
  </tr>
    <td>Analogue 102 is:</td><td><span id="analog102"></span></td>
  </tr>
</table>

</body>
</html>
```

As you can see from the example here above, the line <!--tagparser="/pcotagfilt"--> doesn’t need to be included in the code because the ***pCOWeb*** doesn’t have to parse the page as values are coming from the xml, reducing the workload of the card, the values will be loaded on the arrays by the function getVariables().

API

The library “pw_ajax.js” needs the following functions, which have to be used in the javascript section of the html page:

- getParams (defined in "pw_ajax.js", therefore it has to be only executed in the html page)
- parseResults (from A142 – it is only executed by the pw_ajax.js library, therefore it has to be declared in the html page).

It needs the following arrays to be declared in the html page:

- digitals
- integers
- analogs

And one variable (optional):

- timestamp (A142)

getParams

The function getParams, using the (very common) browser function "XMLHttpRequest" (http://en.wikipedia.org/wiki/XMLHttpRequest), retrieves values from ***pCOWeb*** executing "xml.cgi".
"xml.cgi" is an application custom made for ***pCOWeb*** to generate an xml file containing the values requested by the arguments of the function getParams, the URL that the browser will request to the ***pCOWeb***, following the example, is:

http://<ip_address_pCOWeb>/config/xml.cgi?D|1|2|I|11|12|A|101|102

The syntax of the query is "xml.cgi? + VAR_TYPE|START_INDEX|END_INDEX|", the string allows to request many ranges of variables, up to the maximum length allowed by the browser.

Starting from version A142, "xml.cgi" is able to recognize the argument "N" positioned at the beginning of the request, so that the xml file returned by ***pCOWeb*** will use the new and more performing format; the request now will look like:

http://<ip_address_pCOWeb>/config/xml.cgi?**N**|D|1|2|I|11|12|A|101|102

The library "pw_ajax.js" in "/config" of the firmware >=A142 is automatically using the new format for the xml file; therefore, for the applications designed for firmware A135 in order to use the new format, they have to:

- if the web pages are importing the library using the path "/config/pw_ajax.js", nothing has to be done to update them
- if the web pages are using a local copy of the library, the new library will have to be copied in the folder to update it, the new library is fully compatible with the old one.

For custom made pages which are not using the standard "pw_ajax.js" library to use the new format for the xml file, it is necessary to adjust the string sent by the browser and to adapt the parser of the xml to the new format; keeping the string as for version A135 will use the "old" xml format.

Copy and pasting the same URL to the browser, it will return the following XML (values contained depend on the application running on the ***pCO*** controller):

New format

```
<PCOWEB t="2010-03-22 11:48 ">
  <PCO>
    <DIGITAL>
      <O I="1" V="1"/>
      <O I="2" V="0"/>
    </DIGITAL>
    <INTEGER>
      <O I="11" V="0"/>
      <O I="12" V="0"/>
    </INTEGER>
    <ANALOG>
      <O I="101" V="0.0"/>
      <O I="102" V="0.0"/>
    </ANALOG>
  </PCO>
</PCOWEB>
```

Old format

```
<PCOWEB>
  <PCO>
    <DIGITAL>
      <VARIABLE>
        <INDEX>1</INDEX>
        <VALUE>1</VALUE>
      </VARIABLE>
      <VARIABLE>
        <INDEX>2</INDEX>
        <VALUE>0</VALUE>
      </VARIABLE>
    </DIGITAL>
    <INTEGER>
      <VARIABLE>
        <INDEX>11</INDEX>
        <VALUE>0</VALUE>
      </VARIABLE>
      <VARIABLE>
        <INDEX>12</INDEX>
        <VALUE>0</VALUE>
      </VARIABLE>
    </INTEGER>
    <ANALOG>
      <VARIABLE>
        <INDEX>101</INDEX>
```

```
        <VALUE>0.0</VALUE>
      </VARIABLE>
      <VARIABLE>
        <INDEX>102</INDEX>
        <VALUE>0.0</VALUE>
      </VARIABLE>
    </ANALOG>
  </PCO>
</PCOWEB>
```

After receiving the XML, the library parses it and loads the information to the three arrays declared on the top of the javascript code of the HTML page, it is very important that the name of the arrays have to be kept to "analogs", "digitals" and "integers".
The current time of the ***pCOWeb*** received will be loaded into the variable "timestamp".

parseResults (only from version A142)

Starting from Version A142, the "pw_ajax.js" library is using a new function called "parseResults" which is executed as soon as the browser receives the xml from ***pCOWeb***, so that the values shown in the html can be updated without waiting for the next run of the javascript code.
Up to version A135 the function responsible to read from the javascript arrays (digitals, integers and analogs) was executed just after having sent the request for the xml file to the ***pCOWeb*** before having the new values loaded into the arrays, therefore it was necessary to wait for the next time the function is executed to have the values shown on the web page, the web pages were always showing the values received on the previous request.
This reduces the time-to-values when the web page is loaded for the first time and the responsively of the web page when values change in the ***pCO***.

This function is not defined in "pw_ajax.js" which is, instead, only using it, the function has to be declared in the html page, as shown in the example.
To be compatible with web pages developed with firmware < A142, this function is optional and is executed only if defined in the html file.

Tips

- Because of the not synchronicity of the function XMLHttpRequest, the very first time the web page loads, the values will be "undefined", this can be resolved in two ways:
    1. Preloading the arrays with the values obtained by the ***pCOWeb*** using the HTML tags:
       digitals[1]="<%var(0,1,1)%>";
    2. Handling the exceptions on the javascript code
       if (!digitals[1]) digitals[1]="-"

  This will show "-" when the element 1 of the array digitals is undefined
- Web pages developed for version A135 of ***pCOWeb*** will be working for version A142 with no problems at all.

Advantages

1. Web pages can reside in a Apache or IIS web server, they will have to connect to the ***pCOWeb*** only to download data, which is extremely useful for integration or to overtake the limitations of the embedded web server of the ***pCOWeb*** (for example php or asp or .net could be used to generate dynamic web pages)
2. Any web developer is able to develop the page without knowing the ***pCOWeb*** tags
3. The values can be automatically updated without the horrible refresh of the entire page
4. Adding or removing variables to the page takes seconds
5. Integration with Google gadgets, MacOSx widgets or flash animations using a standard interface

Disadvantages

1. Values are retrieved seconds after the page is loaded, tricks can be used to preload the values on the web page using the standard tags of ***pCOWeb***, if the web pages lies in the card.
2. Up to firmware A135 the requests keep the ***pCOWeb*** busier than the standard ***pCOWeb*** tags, and the more people are connected to the ***pCOWeb*** the worse are the performance. This is not true anymore with firmware >= A142 as the cgi has been drastically improved.

APPENDIX I STRUCTURE OF A *pCOWeb* PLUGIN

For the description and the installation of the Plugins, see 9.6 on page 46.

A Plugin is made up of six elements (Figure I.a):

- Configuration files ("conf_users" directory)
- HTML configuration page ("plugins" directory)
- Executable files ("bin_users" directory)
- Start-up scripts (directory "rc_users" directory)
- Plugin name (contained in the "pluginname" file)
- Installation directory ("install-plug-xxx")

IMPORTANT The position of the files in the Plugin is essential for the correct installation and uninstallation of the Plugin.

In the following descriptions the examples refer to the NTP Plugin (Network Time Protocol) for the automatic synchronisation of the ***pCOWeb*** clock with a time server, using the NTP protocol; the Plugin is available at http://ksa.carel.com.

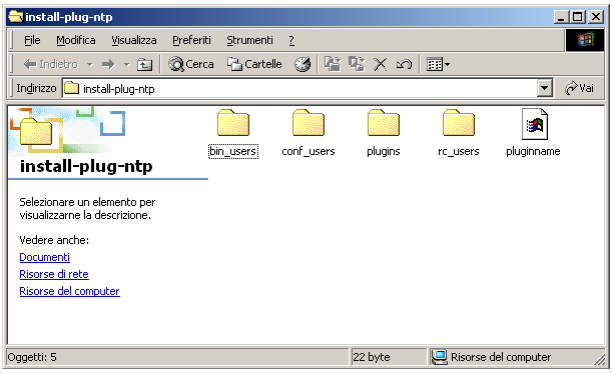


Figure I.a – Elements making up a Plugin

CONFIGURATION FILES

These are ALL the files in the "conf_users" directory of the Plugin being installed.
They contain all the settings defined by the user to customise the functions of the Plugin.
During installation these files are copied to the "/usr/local/root/flash/conf_users/" directory, which corresponds to the $CONF_USERS system variable.
To create or edit a configuration file, use the Plugin web pages or edit it manually using a text editor, accessing the user memory via FTP.

The NTP Plugin only contains the file called "ntp_client.conf", which includes the following line:

```
1:    NTP_URL=10.0.0.131
2:    UTC_OFF=+2
```

that defines the IP address of the NTP server used for communication and the offset from the UTC (i.e. Greenwich Meridian Time).

HTML CONFIGURATION PAGES

These are ALL the files in the "plugins" directory of the Plugin being installed.
They are the HTML pages used to create the configuration files required for the correct use of the Plugin, and are installed in the "/usr/local/root/flash/http/plugins/" directory.

NOTE The presence of .html or .htm files inside the Plugins directory will ensure that the "Configuration / Plugins" section (see Figure 9.r on page 47) contains the link to the configuration page ("Network time Protocol"), and the text "No plugins installed" is no longer displayed; for this reason, when developing a Plugin, at least one HTML page for configuration should be provided, containing at least the name of the Plugin.

Structure of the ntp_client.html page for configuring the NTP Plugin

Below are the contents of the 'ntp_client.html' file; the sections of interest are highlighted in bold and commented on individually; the rest is the HTML code required to correctly generate the page.

```
1     <!--pluginname=Network time Protocol-->
2     <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
3     <html>
4     <!--tagparser="/pcotagfilt"-->
5     <head>
6       <meta content="text/html; charset=ISO-8859-15" http-equiv="content-type">
7       <title>pCOWeb Ntp client configuration</title>
8     </head>
9
10    <script type="text/javascript">
11    function modify(obj)
12    {
13    if (document.getElementById(obj).style.visibility=="visible")
14          document.getElementById(obj).style.visibility="hidden";
15    else
16          document.getElementById(obj).style.visibility="visible";
17    }
18    </script>
```

```
19
20    <h1>pCOWeb ftp client System Configuration</h1>
21    <div style="color: #999999; font-size: 14px">pCOWeb Ntp client
configuration</div>
22
23    <form method="POST" action="">
24    <h4>Ntp client configuration parameters:</h4>
25    <table>
26    <tr>
27    <td><B>Ntp server Address:    </B></td>
28    </tr>
29    <tr>
30          <td><%getdb('conf_users/ntp_client.conf','NTP_URL')%></td>
31          <td><a href=javascript:modify("ntp_url")> Modify </a></td>
32          <td><input id=ntp_url style="visibility: hidden" type="text"
name="?script:setdb('conf_users/ntp_client.conf','NTP_URL')"
value="<%getdb('conf_users/ftp_client.conf','NTP_URL')%>"></td>
33    </tr>
34    </table>
35    <p>
36    <input type="submit" value="Submit">
37    </form>
38    <br>
39    <a href="/config/scanplugins.cgi">Back to plugins page</a>
40    </html>
```

DESCRIPTION OF THE LINES
(also see APPENDIX G on page 71)

1: Syntax that defines the text shown on the link on the "Configuration/Plugins" HTML page (Figure 9.r on page 47); if not present, the text shown on the link is the name of the HTML file.
IMPORTANT This should not be confused with the pluginname file, located in the Plugin installation directory, which is only used during the installation and uninstallation phase.

4: Syntax telling the ***pCOWeb*** to process the page and replace the CAREL tags with the corresponding values; this is required to be able to write and read the configuration files.

23: The HTML ***form*** statement is used to send information from the html page to the ***pCOWeb***; note that the ***method*** must always be ***POST***.

30: The ***getdb*** tag is used to display the value of a setting in the configuration file specified. The example refers to the value associated with NTP_URL in the ntp_client.conf file located in "/usr/local/root/flash/etc/sysconfig/conf_users/"; the full path does not need to be specified if the file is located in a subdirectory of "/usr/local/root/flash/etc/sysconfig /"; another directory can be specified by correctly indicating the full path, starting from the root ("/"), nonetheless it is recommended to use the suggested directory so as to maintain coherence in the structure of all the Plugins developed.

32: This HTML statement defines an input field used to display and modify the value associated with a parameter in a configuration file. The ***setdb*** statement can be used to modify the value associated with the variable when the "Submit" button is selected on the form.

36: Closing the page, the "Submit" button is displayed.

39: Link that, selecting the "Submit" button, refreshes the display of the ***pCOWeb*** Plugin configuration page; Administrator authentication (see 9.2.1 on page 41) is required to display the page.

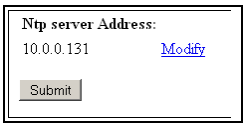


Figure I.b – Field used to set the address of the NTP server

The field displayed to change the value of the NTP_URL parameter is as shown in Figure I.b.

EXECUTABLE FILES

These are ALL the files in the Plugin "bin_users" directory of the Plugin being installed.
They represent the executable files (binary files or bash scripts) that manage the functions of the Plugin.
During installation the executable files are installed in the "/usr/local/root/flash/bin_users/" directory, which corresponds to the $BIN_USERS system variable; the path is included in the $PATH system variable.
During installation, each of these files are attributed the permission to be executed as envisaged by the Linux operating system, shown in Figure I.c.

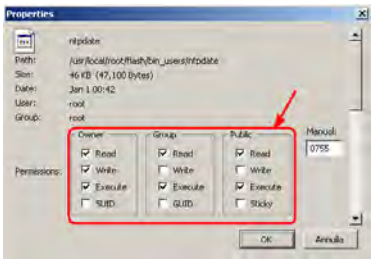


Figure I.c – Linux permission for executing the "ntpdate" files

The following executable files are included in the NTP Plugin:

- "ntpdate" (this handles communication with the NTP server and writes the time to the system clock);
- "ntp.sh" (this handles the management); the latter is described in detail below.

The ntp.sh script file

This runs the "ntpdate" binary file, deletes any temporary files created by "ntpdate" and, if this is involved in the periodical updating of the time, manages the cycle of updates.

```
1     #!/bin/sh
2     . /etc/profile > /dev/null
3     . $CONF_USERS/ntp_client.conf
4
5     while [ 1 ]; do
6           ntpdate ${NTP_URL} > /dev/null
7           date
8           rm /initrd/tmp/ntp
9           sleep 3600
10    done
```

DESCRIPTION OF THE LINES

1: Specifies the command executor; currently the only one featured on the ***pCOWeb*** is the bash shell ("Bourne Again SHell"), therefore "/bin/bash".
2: Read the "/etc/profile" file that contains the definition of the $CONF_USERS, $RC_USERS, $BIN_USERS variables, to simplify the management of the paths inside the Plugin.
3: Read the Plugin configuration file "ntp_client.conf" in the $CONF_USERS directory, so as to be able to use the variables set in the configuration (for example, from the html page) in the script.
5: Infinite cycle to allow the script to continuously run the statements.
6: Update the time using "ntpdate", requesting the data from the server defined in $NTP_URL, and redirect the standard output to "/dev/null" to eliminate any unnecessary messages.
7: Display the time obtained above.
8: Delete a temporary file created by "ntpdate", otherwise "ntpdate" cannot be run again.
9: Wait 3600 seconds before running the next statement (synchronisation); alternatively this could have been defined as a parameter in "ntp_client.conf" and then easily recalled, for example, by the $SLEEP_TIME variable.

START-UP SCRIPTS

These are ALL the files in the Plugin "rc_users" directory on the Plugin being installed.
They are used to be able to run the Plugin functions automatically when the system is started.

During the installation the scripts are installed in the "/usr/local/root/flash/rc_users/" directory, the path string is also returned by the $RC_USERS system variable.
During installation, each of these files are attributed all the permissions required by the Linux operating system, in the same way as for the executable files seen above (Figure I.c on page 85).

NOTE The execution permissions are also used by the system to decide which scripts to run automatically; to prevent a Plugin from being run at start-up, reset the execution bits ("Execute") relating to the script.

Structure of the start-up scripts

For the start-up script to run correctly, it must have the following structure (the example refers to the NTP Plugin):

```
1     #!/bin/bash
2
3     . /etc/init.d/functions/highlight
4
5     case "$1" in
6           start)
7           showMsg "Starting Ntp script..."
8           (ntp.sh &) > /dev/null
9           showStatus "$?"
10          ;;
11
12          stop)
13          showMsg "Stopping Ntp Script..."
14          id=$(pidof -x ntp.sh)
15          kill $id
16          showStatus "kill$id"
17          ;;
18
19          restart)
20          showMsg "Restart ntp Script..."
21          $0 stop
22          $0 start
23          ;;
```

```
24
25          *)
26          echo $"Usage: $0 {start|stop|restart}"
27          exit 1
28          ;;
29      esac
```

DESCRIPTION OF THE LINES

1: Specifies the command executor, currently the only one featured on the ***pCOWeb*** is the bash shell ("Bourne Again SHell"), therefore "/bin/bash".
3: Add the colouring function for the log messages (see the "highlight" script).
5-6: When the script is run, parameter "`$1`" is equal to "`start`".
7: `showMsg` (see lines 13-18 of the "highlight" script) displays the text "`Starting Ntp script...`" in the system log.
8: This line runs the "ntp.sh" script (see the previous section); note that the standard output is redirected to "/dev/null" to avoid unnecessary messages.
9: `showStatus` displays the result with the related colouring (see lines 20-37 of the "highlight" script); note the variable "`$?`", with the value returned by the previously run statement.
12-17: As for the start section, the stop section is made up of the same parts, and is called when the ***pCOWeb*** is shutdown or when accessed by the user.
19-23: Runs the script again, first the stop section and then the start section, used to have the Plugin reread the new configuration files; it cannot be run from the web page but only from a remote console.
24-28: This is the section of the script that is run in the event where "`$1`" is not equal to `start`, `stop` or `restart`; a message is displayed to indicate that the script has been run with an incorrect value of "`$1`".

"Highlight" script

This script is used to define the standard display used in the various scripts on the ***pCOWeb***.

```
1       export escGreen=`echo -e "\033[1m\033[32m"`
2       export escRed=`echo -e "\033[1m\033[31m"`
3       export escYell=`echo -e "\033[1m\033[33m"`
4       export escNorm=`echo -e "\033[m"`
5       export msgOk=`echo -e "[\033[1m\033[32mOK\033[m]"`
6       export msgDone=`echo -e "[\033[1m\033[32mDone\033[m]"`
7       export msgYes=`echo -e "[\033[1m\033[32mYes\033[m]"`
8       export msgNo=`echo -e "[\033[1m\033[32mNo\033[m]"`
9       export msgError=`echo -e "[\033[31mError\033[m]"`
10      export msgFail=`echo -e "[\033[31mFailed\033[m]"`
11      export msgBad=`echo -e "[\033[31mBad\033[m]"`
12
13      #
14      #  Echo script message
15      #
16      function showMsg(){
17            printf "%-70s" "$1"
18      }
19
20      #
21      #  Show status message
22      #
23      function showStatus(){
24            case "$1" in
25            0)    echo "$msgOk";;
26            1)    echo "$msgFail";;
27            ok)   echo "$msgOk";;
28            error)      echo "$msgError";;
29            bad)  echo "$msgBad";;
30            yes)  echo "$msgYes";;
31            no)   echo "$msgNo";;
32            done) echo "$msgDone";;
33            failed)     echo "msgFail";
34            kill*)      test "$1" = "kill" && echo '[Not runn]' || echo $msgOk;;
35            *)    echo "$1";;
36            esac
37      }
```

DESCRIPTION OF THE LINES

1: The colour of the characters written afterwards becomes green.
2: The colour of the characters written afterwards becomes red.
3: The colour of the characters written afterwards becomes yellow.
4: The colour of the characters written afterwards returns to the default.
5: Displays the text [Ok]
6: Displays the text [Done]
7: Displays the text [Yes]
8: Displays the text [No]
9: Displays the text [Error]
10: Displays the text [Failed]
11: Displays the text [Bad]
13-18: Display the value of the variable "$1" and write the next character after 70 characters from the start of the line displayed; if the output of "$1" is "Test variable", the following is displayed:

```
$ Test variable·····································     $
```

where $ represents the shell prompt.
20-37: Displays different signals in the log based on the value of "`$1`" sent to the function.

PLUGIN NAME

This the file located in the Plugin directory with the name "pluginname".
It contains the Plugin name that will be displayed for the link during the installation and uninstallation of the Plugin.
IMPORTANT: the contents of "pluginname" must not contain spaces.
In the case of the NTP Plugin, the file contains the string "`Network_Time_Protocol`", and the following link is displayed: '**Install plugin "Network_Time_Protocol**" ' (Figure 9.0 on page 46).

PLUGIN DIRECTORY

For a Plugin to be automatically recognised as able to be installed by the ***pCOWeb***, the directory that will be copied via FTP (see 9.6.1 on page 46) and that contains all the files and all the directories must be called "Install-plug-xxx", where xxx is any name used to identify the contents of the Plugin, yet has no special meaning for the ***pCOWeb***.

CHARACTERISTICS OF THE *pCOWeb* GNU/Linux OPERATING SYSTEM

Currently the ***pCOWeb*** can run two types of executable files:

- Bash script
- Compiled executables

"bash" is an interpreted language used to perform simple tasks or cyclical binary executions; it is a standard language and is widely described on the Internet.

For a compiled executable to be able to run on the ***pCOWeb***, it must have the following characteristics:

- Architecture: ARM
- Recommended compiler: gcc 2.95.3
- Linux Kernel: 2.4.21
- Glibc: 2.2.3
- Textutils: 2.1
- Fileutils: 4.0
- Sh-utils: 2.0
- Inetutils: 1.3.2

INDEX

NOTES