EXOR
Tech-note

# Using the SCM10 TWS Module

This Technical Note is an introduction to the use of the SCM10 Web Server module. This document does not cover all the technical details you will need to develop a complete application; it represents a guideline summarizing all the main steps required to use the SCM10 TWS (**T**iny **W**eb **S**erver). This manual contains all the references to the specific documents describing in more detail the topics covered here.

## Contents

## 1 Introduction

The SCM10 module transforms any UniOP operator panel into a standard web server. This novel approach will allow you to locate the web server as close as needed to the machine.

The web server module allows remote access to the data available in the operator panel using standard Internet protocols. The SCM10 module has a built-in Ethernet interface that makes the connectivity to the factory network extremely easy and effective. A simple and standard Web browser is the only tool needed to remotely access the data in the module.

The SCM10 contains a fully capable web server with 32 MB of flash memory to store web pages. The web server is accessible using the on-board 10BASE-T Ethernet interface.

The SCM10 module is compatible with the format of the UniOP communication modules and it has to be mounted in the standard plug available in every UniOP panel.
Programming the web server is performed using the ftp protocol through the Ethernet interface.
The web application can be created using standard HTML editing tools.

The web server in the SCM10 can obtain live data from the UniOP where it is plugged-in. From this point of view the combination UniOP+web server is actually acting as a gateway between industrial devices and the web. The web server can access PLC data using the communication protocol that is loaded in the UniOP panel. This is true for all the communication protocols currently supported by UniOP (with the only restriction that today's operator panels can only accept one optional module).

Web applications can use either the CGI (***C**ommon **G**ateway **I**nterface*) protocol or the ASP (***A**ctive **S**erver **P**ages*) technique to gather live data from the panel. See later in this manual for a brief introduction to these techniques.

***Note:** operation with the SCM10 requires UniOP firmware version V5.30 or higher.*
*All firmware types support the SCM10 module.*

## 2 Architecture

The Tiny Web Server (TWS) is based on a free source embedded web server[1]. The SCM10 implementation of the embedded web server adds some specific features to the original off-the-shelf product specifications. This includes, for instance, the possibility to sample external data with a predefined rate and store them into a trend data structure, for later retrieval by the client module.

[1] Goahead webserver 2.1. See http://www.goahead.com

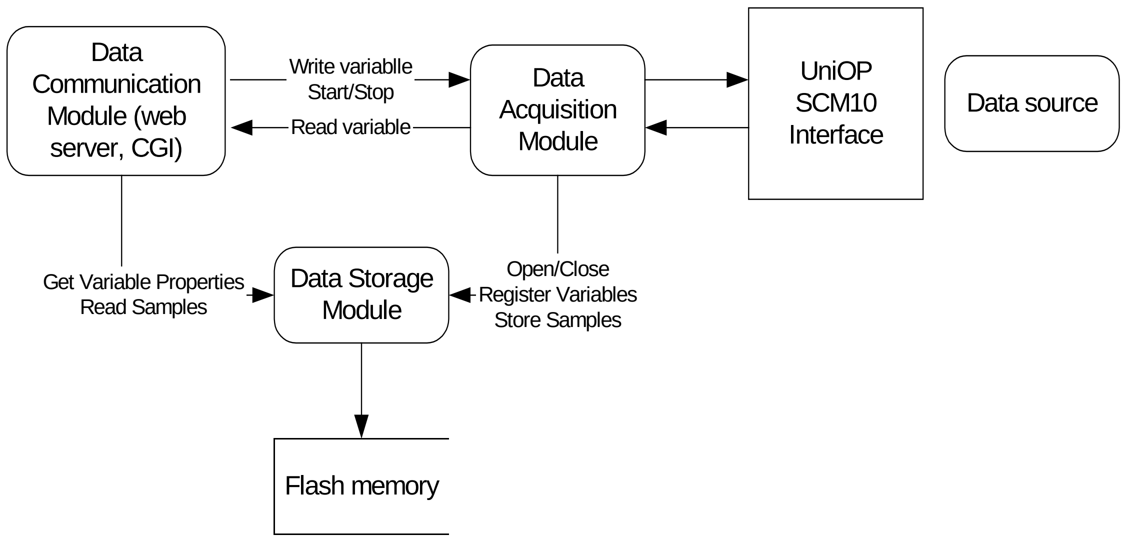

Figure 1

Figure 1 shows the server architecture with the various modules and their interfaces.
The TWS is composed of different modules, each one dedicated to a specific function.
The operation of the SCM10 module depends on the UniOP panel for the communication (that is the data exchange taking place between the controller and the panel via the serial port).

The Web Server in the SCM10 module prepares communication requests for the panel using a special interface. The operator panel will retrieve the data requested by the web server from the external controller connected via the serial line. It will then return the requested data to the web server.

The Data Acquisition Module is responsible for this activity. Data gathered from UniOP is stored locally on the web server (Data Storage Module); it is available to the client application (Data Communication Module).
This approach makes the memory in the external controller (PLC or other) fully accessible to the web server.

## 2.1 Data Communication Module

This module consists of:
1. Embedded web server responsible for handling HTTP requests.
2. CGI functions (activated on client request), that can read data either from the Data Acquisition Module or from Data Storage Module and sends them back to the client.

## 2.2 Data Storage Module

This module, which interfaces with Data Communication Module and Data Acquisition Module, is dedicated to store data obtained from the UniOP communication channel in a specified format.

## 2.3 Data Acquisition Module

The Data Acquisition Module acquires fresh data from the data source and makes them available to both the Data Storage Module and the Data Communication Module.

# 3 Interface with Client

There are several ways for a client application to interact with the web server.

## 3.1 The CGI (Common Gateway Interface) Standard

The Common Gateway Interface (CGI) is a standard for interfacing external applications with Web servers. **CGI** stands for **Common Gateway Interface;** it is a standard ("common") way of getting away from the server (gateway) to communicate ("interface") with other processes.
A plain HTML document that the Web server **retrieves** is **static**, which means it exists in a constant state: a text file that does not change.
A CGI program instead is **executed** in real-time, so that it can produce **dynamic** information.
CGI is not a programming language. It is a simple protocol that can be used to communicate between Web servers and CGI compliant applications. A CGI script can be written in any language that can read STDIN, write to STDOUT, and read environment variables, i.e. virtually any programming language, including C, Perl, or even shell scripts.

It is important that, whatever the CGI program does, it should not take too long to process. Otherwise, the users will just be staring at their browser waiting for something to happen.

Since a CGI program is executable, it is basically the equivalent of letting the world run a program on your system, which normally is not the safest thing to do. Therefore, there are some security precautions that need to be implemented when it comes to using CGI programs.
There are eight possible types of communication between the client and the server.
These types are:

| | |
|---|---|
| Getting trend samples | *getdata* CGI |
| Read operation on PLC variables | *readVariable* CGI |
| Write operation on PLC variables | *writeVariable* CGI |
| Getting trend samples for Excel client | *getexceldata* CGI |
| Format operation on local storage disk | *FormatDisk* CGI |
| Printing a system configuration report | *CGIGetSysConfig* CGI |

All the different types of Client access are described in details in the “RU05DV01.doc” document; please refer to this manual for additional and more specific information.

## 3.2 The ASP (Active Server Pages) standard

ASP stands for Active Server Pages. ASP is a server-side technology that is used to display dynamic content in the web pages. ASP in itself is not a language but it actually uses VBScript or JScript to display dynamic content. ASP is more of a technology used by VBScript / JScript on the server side.

Normally, when someone visits a web site, his browser sends a request for a certain file, such as index.htm. The server will then return this file to the client computer, and the file itself will be displayed in the browser as shown in Figure 2.

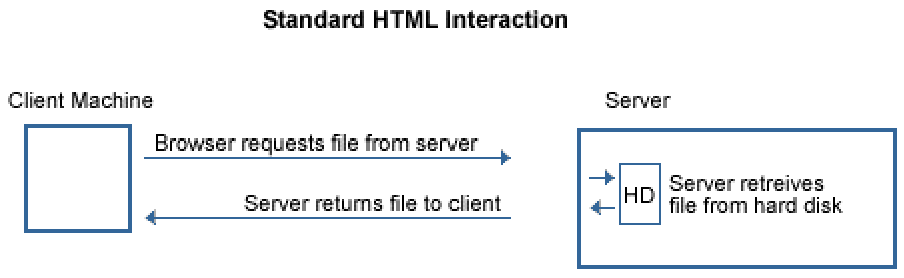
Standard HTML Interaction
Client Machine
Browser requests file from server
Server returns file to client
Server
HD
Server retreives file from hard disk

Figure 2

Active Server Pages, instead, allow, for instance, Visual Basic scripts to be processed on the server before sending the file back to the client.

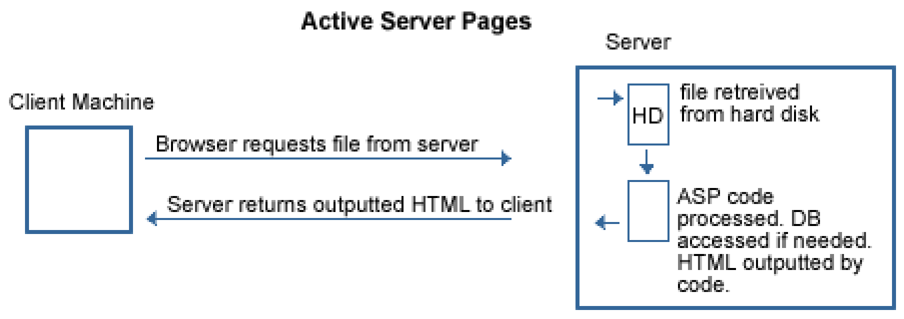
Active Server Pages
Server
Client Machine
Browser requests file from server
Server returns outputted HTML to client
HD
file retreived from hard disk
ASP code processed. DB accessed if needed. HTML outputted by code.

Figure 3

This results in a number of benefits:

- unlike client-side code, such as the JavaScript that makes ad windows pop-up, the browser does not need to understand VB - it does not even get to see it.
- the VB code can act on information passed to the page, such as from an Internet form or a query string (this is data passed in the URL after the ‘?’). It can then use this information to display data, or retrieve specific information from a database
- only the smallest amount of data is sent to the client - only the HTML that the page outputs is sent, not the VB code. This also means that the author’s VB programming work is protected from visitors.

## 3.3 The Java Gateway

The use of the CGI approach can be simplified a lot by using a simple additional piece of software that makes the interface to CGI commands easier.

We can provide a prototype for an invisible Java applet acting as a bridge between the JavaScript inside a web page and the TWS CGI interface.

Thanks to this applet, JavaScript can be used as “glue language” between TWS and whatever ActiveX we may have inside the web page (e.g. a gauge, a meter). The applet further extends the options to connect the TWS to third party software systems.

This applet, in a way, plays the same role of the OPC Data Automation Interface, that is a software bridge that can be accessed by a script language from within an application. In the case of the OPC, the application may be MS Excel and the scripting language is VBA; in the case of SCM10 the application is an Internet browser and the scripting language is JavaScript. To better understand the comparison, please check the related documentation about OPC and Data Access contained in the manual TN154.

The Java Class developed for this application corresponds to the file called *ICUGateway.class* and it is available also in source code in the file named *ICUGateway.java*.

The use of the Java applet interface avoids the need to prepare the explicit CGI commands, using instead a more user-friendly approach based on the use of class methods.

In order to better understand how the Java class operates and what exactly it does, please check the following example.

The example is written in HTML and JavaScript. This sample program is ready-to-run.
Use the following sequence to run the sample application using Microsoft application programs:

- copy the text contained in the box and paste it in a text document created with Windows Notepad
- start a session of Microsoft FrontPage
- create a new project
- click on the HTML tab on the bottom of the application form
- delete the default entries made after the creation of the new project
- paste the text copied from Notepad

```
<html>
<head>
<title>ICU Gateway Example</title>
<script language="JavaScript">
   <!--

var Interv = 10

function startClock() {
   Interv = Interv - 1

   var now = new Date()
   var dummystr = parseInt(now.getTime() / 1000)
   delete now

   if (Interv < 0) {
        document.myform.voltage.value = document.ICUGTW.readVariable("Voltage");
        document.myform.current.value = document.ICUGTW.readVariable("Current");
      Interv = 10
   } else {
        document.myform.time.value = Interv;
    }
   timrID = setTimeout("startClock()", 1000)
}

   //-->
</script>

</head>
<body onLoad="startClock()">
<H1>Sends / Gets Data to/from the ICU</H1>
<form name="myform">
<table>
<tr>
<td>
<table>
<tr>
<td> Voltage </td>
<td> <input type="text" NAME="voltage"> </td>
</tr>
<tr>
<td> Current </td>
<td> <input type="text" NAME="current"> </td>
</tr>
```

```
<tr>
<td> Data will be updated in </td>
<td> <input type="text" NAME="time"> </td>
<td> sec...</td>
</tr>
</table>
</td>
</tr>
<tr>
<td>
<table>
<tr>
<td>
<input type="button" value="GE_L1 ON"
       onClick="document.ICUGTW.writeVariable('GE_L1', '1');">
</td>
<td>
<input type="button" value="GE_L2 ON"
       onClick="document.ICUGTW.writeVariable('GE_L2', '1');">
</td>
<td>
<input type="button" value="GE_L3 ON"
       onClick="document.ICUGTW.writeVariable('GE_L3', '1');">
</td>
</tr>
<tr>
<td>
<input type="button" value="GE_L1 OFF"
       onClick="document.ICUGTW.writeVariable('GE_L1', '0');">
</td>
<td>
<input type="button" value="GE_L2 OFF"
       onClick="document.ICUGTW.writeVariable('GE_L2', '0');">
</td>
<td>
<input type="button" value="GE_L3 OFF"
       onClick="document.ICUGTW.writeVariable('GE_L3', '0');">
</td>
</tr>
</table>
</td>
</tr>
</table>
</form>
<applet name="ICUGTW" code="ICUGateway.class"
   codebase="http://194.185.135.202:8432/demo/default.html "
   width="0" height="0">
  <param name="TRACE_CGI" value="on">
</applet>
</body>
</html>
```

The example uses the class methods “readVariable” and “writeVariable” to read and write some data items present in the controller connected to the UniOP panel hosting the SCM10 module.
In the example the variables are referenced as tag names:

- GE_L1
- GE_L2
- GE_L3
- Voltage
- Current

The correspondence between the tag names and the actual PLC data is defined explicitly in the VARS.INI file as described in the RU05DV01.doc and RU05DV02.doc documents. The tag file must be loaded in the web server

# 4 TWS Configuration and Operating Instructions

This chapter contains a brief introduction to the steps required to operate the web server module.

## 4.1 Web Server Operation Modes

The web server can only be in 4 different operation modes; the presentation below includes a description of the operation modes and the events that can generate transitions from one mode to the next one.

| Mode | Description | Transitions |
|---|---|---|
| OM 1 | Disk not formatted | Only one CGI can be executed: *FormatDisk* CGI.<br>Every request will be solved in a response to inform users that the (flash) disk is not formatted<br>After the user has formatted the disk using *FormatDisk* CGI, TWS goes to *OM 2*. |
| OM 2 | Disk formatted but Data Acquisition Module cannot start | The FTP server is active and ready for connection.<br>Every other request will be solved in a response to inform the users that there is no enough information to start the Data Acquisition Module (a detailed report about missing information is notified). |
| OM 3 | Normal TWS operation (Data Acquisition Module is active) | Once the user has uploaded valid files, the Data Acquisition Module is automatically started. |

Prerequisites for the successful startup of Data Acquisition Module are:
- Disk space check passed
  - All the necessary configuration files are present and contain valid data

Data Acquisition Module supports connection to the communication driver loaded in the UniOP panel hosting the TWS module; the Data Acquisition Module can be considered an option since an application made for the TWS could also run without the use of the Data Acquisition Module. In this case there is not interaction between the TWS and the UniOP communication protocol.

The CGI system commands described above must be sent to the TWS module using a standard web browser typing the required string in the address entry of the browser.

## 4.2 Web Server Configuration

The web server may be configured using a specific set of initialization files that must be copied in the root directory of the web server (flash) disk.

The TWS configuration files for the web server setup are listed below; all configuration files described in this section are ASCII files where each line contains a pair in the format "Parameter=Value".

- TWS.INI
- VARS.INI
- PASS.INI
- PLC.INI

The information contained in the initialization files is described in greater detail in the document RU05DV01.doc.

The initialization files must be uploaded to the TWS flash disk using an FTP client; the FTP server is always active and available.

Since accessing the FTP server provides users with the capability of changing the TWS functionality, the login between FTP server and client requires an authentication procedure; the user name and password must be entered when the appropriate dialog is displayed in the browser. Password and user name are defined in the configuration file PASS.INI. If authenticated, the user will receive a message from TWS indicating whether the FTP client has been successfully connected

**Note:** *If there is no PASS.INI file in server root directory, the authentication procedure can be cleared typing any user name and no password.*
*The TWS FTP server dos not support PASV mode.*

## 4.3 Network parameters

The network parameters of the Tiny Web Server can be configured via the UniOP Configuration Mode IP setup menu.

**Note:** *the network parameters setup via UniIP Configuration Mode requires panel Firmware version V5.30 or above and TWS firmware version V2.03B or above.*

The following parameters can be set:

- IP address
- Subnet Mask
- Gateway/Router IP address

## Appendix A. - Diagnostic Information

SCM10 is equipped with a bi-color LED indicator (red and green). This indicator is used to display the operating status of the TWS.

| **LED** | **Status** | **Description** |
|---|---|---|
| **Green** | On | the device is operating (TWS running) |
| | Blink | the device is running but no application is present. |
| **Red** | On | if the red LED remains ON a fault condition is present; in this situation the green status LED flashes an error code and service is needed. |

The following table describes the additional possible green LED status when red LED is fixed ON.

| **Green LED Status (Red LED is on)** | **Fault or Error Description** |
|---|---|
| flashes 1 time | EEPROM user section checksum test failed |
| flashes 2 times | EEPROM factory section test failed |
| flashes 3 times | SRAM test failed |
| flashes 4 times | Application checksum failed |
| flashes 5 times | Loopback test failed |
| flashes 6 times | SSFDC detection failed |