RESEARCH PAPERS
FACULTY OF MATERIALS SCIENCE AND TECHNOLOGY IN TRNAVA
SLOVAK UNIVERSITY OF TECHNOLOGY IN BRATISLAVA

2016 Volume 24, Number 39

EMBEDDED PLC WEBSERVER AND POSSIBILITIES OF ITS UTILIZATION

Michal KOPČEK[1,2]

[1]SLOVAK UNIVERSITY OF TECHNOLOGY IN BRATISLAVA,
FACULTY OF MATERIALS SCIENCE AND TECHNOLOGY IN TRNAVA,
INSTITUTE OF APPLIED INFORMATICS, AUTOMATION AND MECHATRONICS,
ULICA JÁNA BOTTU 2781/25, 917 24 TRNAVA, SLOVAK REPUBLIC,
[2]SKARTEK S.R.O. TRSTÍNSKA 9, 917 01 TRNAVA, SLOVAK REPUBLIC
e-mail: michal.kopcek@stuba.sk / m.kopcek@skartek.com

Abstract

The concept Industry 4.0 is a top theme of recent times. Even if it represents the so-called 4th industrial revolution its main ideas could be used also to support the education of subjects from IT and automation. Furthermore, the basic IT and web technologies inherited by this concept are very well known to students. This paper introduces the embedded webserver of PLC and the options for its optimal utilization. The web server is essentially a simple and widely well-known technology, therefore it was chosen as an example.

Key words

webserver, PLC S7-1200 and S7-1500, Industry 4.0, education

INTRODUCTION

The concept Industry 4.0 is a top theme of recent times. Besides its application in the industrial environment, it could also be used in education. The advantages of such implementation are obvious. It keeps teachers in touch with modern trends in industrial automation and makes training more attractive for students.

A lot of technologies within Industry 4.0 have their origin in web technologies and informatics. However, their implementation on the process level is limited by hardware which usage is primarily dedicated to control the processes. Therefore, this paper is focused on the options of implementation of the so called user web pages using the embedded webserver, which is not necessary for primary control functions. This technology on one side offers added value for production management and on the other hand the students are pretty familiar with the web page development.

The S7-1200 and S7-1500 PLCs (Programmable Logic Controller) were chosen as the example to keep the explanation more clear and due to the fact that the Simatic PLCs have greatest share in the institute laboratories. The webservers of these two types of PLC are fundamentally the same and so the procedure of the web page implementation is practically identical (SIEMENS 2015).

OBJECTIVE AND METHODS

The main objective of this paper is to provide basic knowledge about embedded webserver technology, which is one of the fundamental parts of Industry 4.0. Furthermore, it also aims to show some good practice to use the system resources optimally.

Fundamental scientific methods like analysis, experiment, deduction and creation of hypothesis were used. The analysis offers a closer view on the functioning of the webserver and creation of user web pages as well as the implementation of good practice to optimize the system behavior. The employment of deduction and hypothesis creation based on the results of analysis and experiments leads to conclusions and discussion about possible applications of the embedded PLC webserver.

TESTING OF THE PLC WEBSERVER

Starting up the web server

As mentioned before, the S7-1200 and S7-1500 PLC webservers were chosen for the experiment of this paper. The test application was developed using the TIA Portal V13, which is development software for programming Siemens PLCs. The web pages were coded using Notepad++ and tested in web browsers (Chrome, Internet Explorer).

To begin the work it is necessary to start up the embedded webserver in the device configuration of the PLC by checking appropriate parameters in four steps as shown in Fig 1. Because of security reasons it is also necessary to create user and assign privileges to be able to browse the web pages. User "Admin" with all privileges was used during the experiment.

After enabling the web server it is already possible to connect to it by entering the IP address of the PLC in the web browser. Figure 2 shows the default PLC web page before and after logging in. As could be seen in Fig. 2 the web page provides many useful functions for e.g. PLC diagnostics, tag monitoring, file management etc.

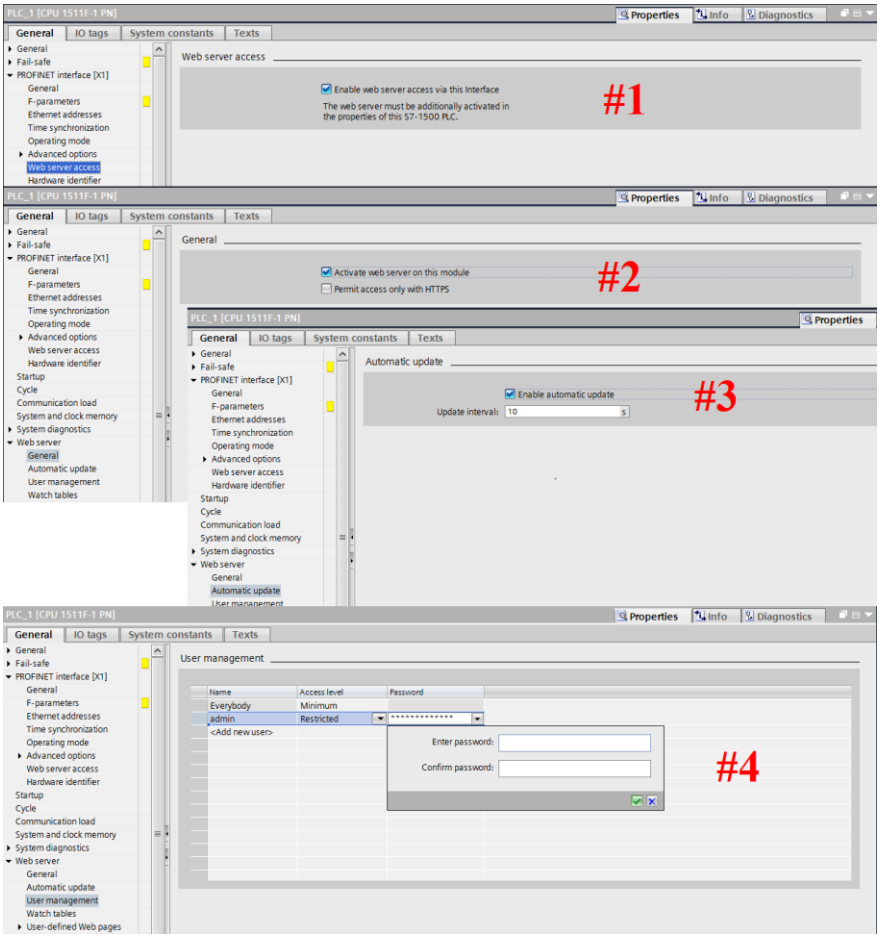


Fig. 1 *Enabling the webserver and user management in the PLC hardware configuration. Source: Author*

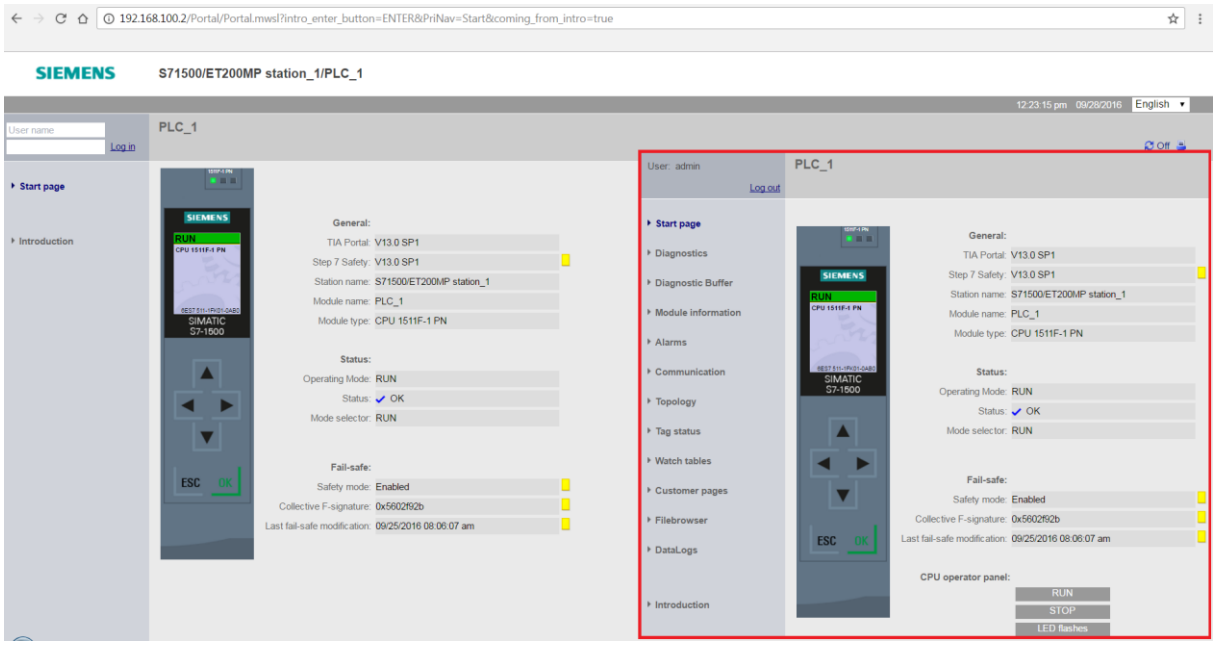


Fig. 2 *The PLC start web page. Source: Author*

Processing of the user web pages in PLC

The custom user web pages offer unlimited opportunities to visualize data according to the customer needs from simple overviews to fully-featured web based HMI (Human Machine Interface). The user web pages are developed in HTML 5 using CSS and JavaScript, while the page files need to be loaded to the PLC (SIEMENS 2016). This is done in the device configuration of the PLC as shown in Fig. 3.

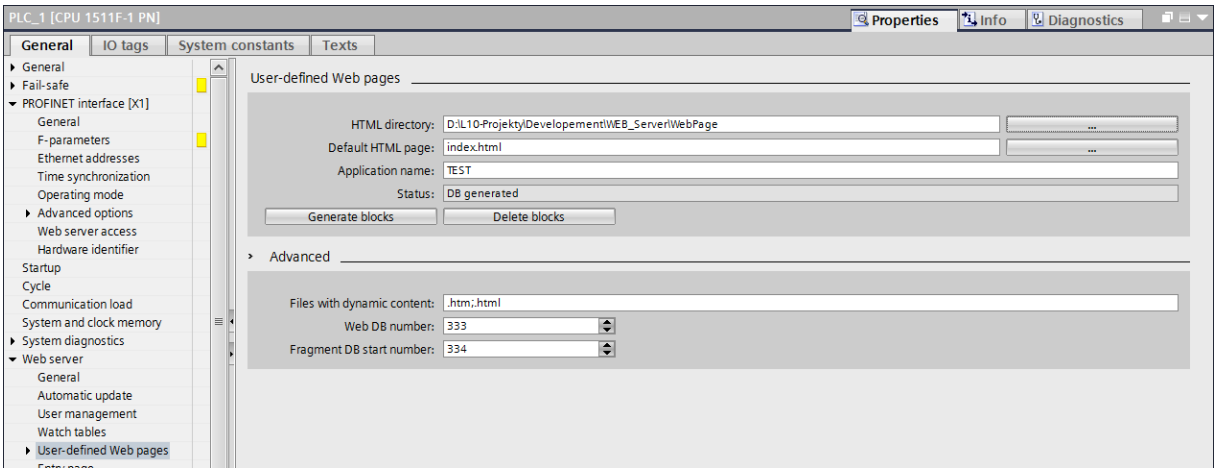


Fig. 3 *The web page linking and generating in the PLC configuration. Source: Author*

After linking the page files, it is necessary to generate data blocks, where the PLC stores the data. The web data block (DB333) and fragment data block (DB334 and further). The fragment data block stores the content of the web page files as array of bytes, while one array includes one file. If the maximum number of bytes that can be contained in a data block is exceeded, the generating procedure creates the next data block. To run the user web pages it is also necessary to add the WWW function into the PLC program, which in conjunction with the DB333 controls the processing of fragments.

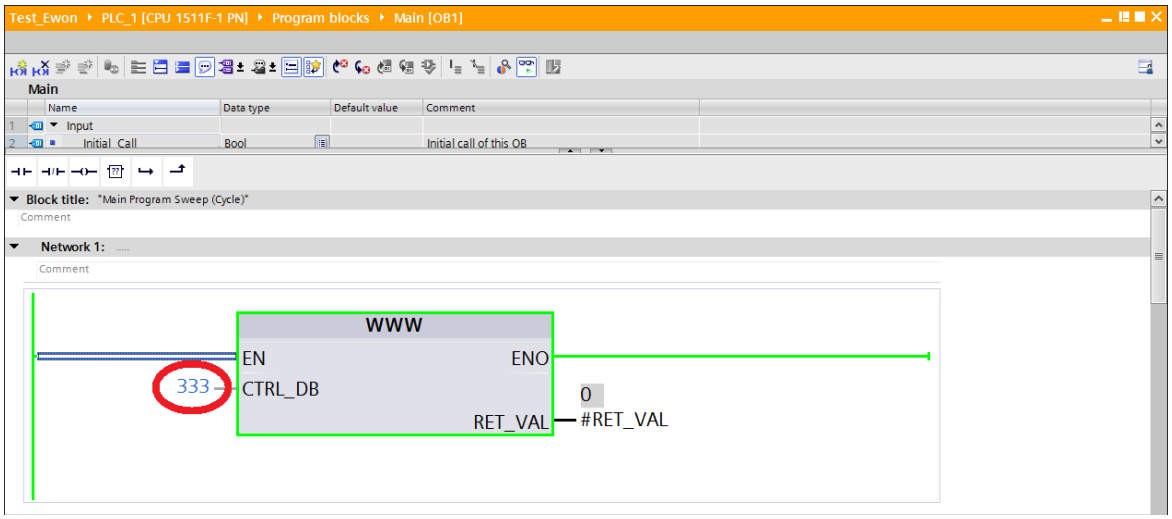


Fig. 4 *The WWW function with linked DB333. Source: Author*

Creating the user web pages

A simple web page shown in Fig. 5 was created within the experiment. Three counters and a data log file representing variables to be read from the PLC and the reset button representing the variable to be written to the PLC were implemented in the PLC program. It also needs to be mentioned, that this kind of applications naturally works with PLC variables, so-called tags. These tags could represent physical inputs and outputs of the PLC as well as any other type of variable in the PLC memory. The tags are linked to the web page through special script command in the HTML code, which is recognized by the web server. To get a fully functioning application it is necessary to continuously read the tags values, this is done by the automatic update function of the webserver. However, the automatic refresh of the web page leads to flashing of the page in the web browser. To avoid this, the variable exchange should be done in the background. Therefore the page was split into 3 files (Fig. 6):

- index.html – contains the HTML code of the web page,
- input.html – contains an XML structure with input variables,
- output.html – contains an XML structure with output variables.

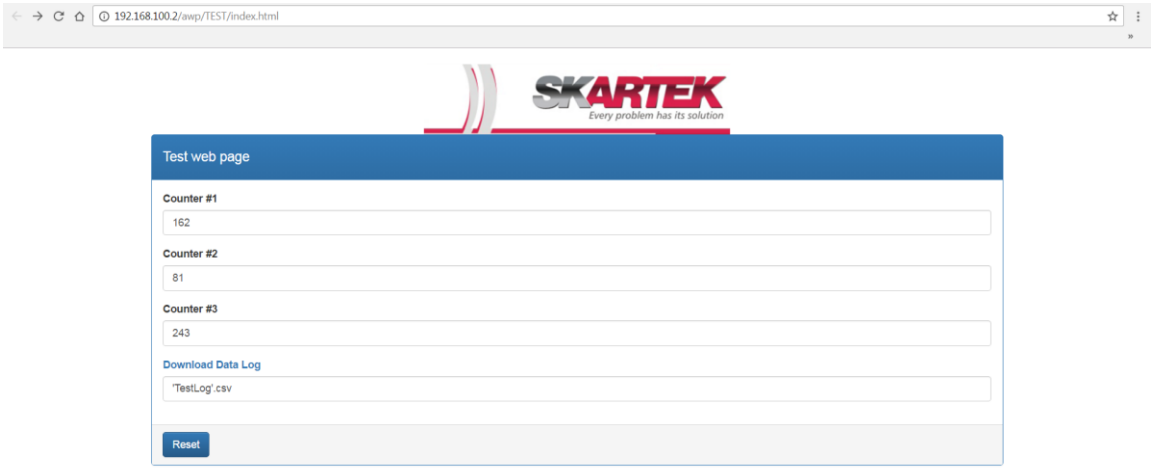


Fig. 5 *The test web page. Source: Author*

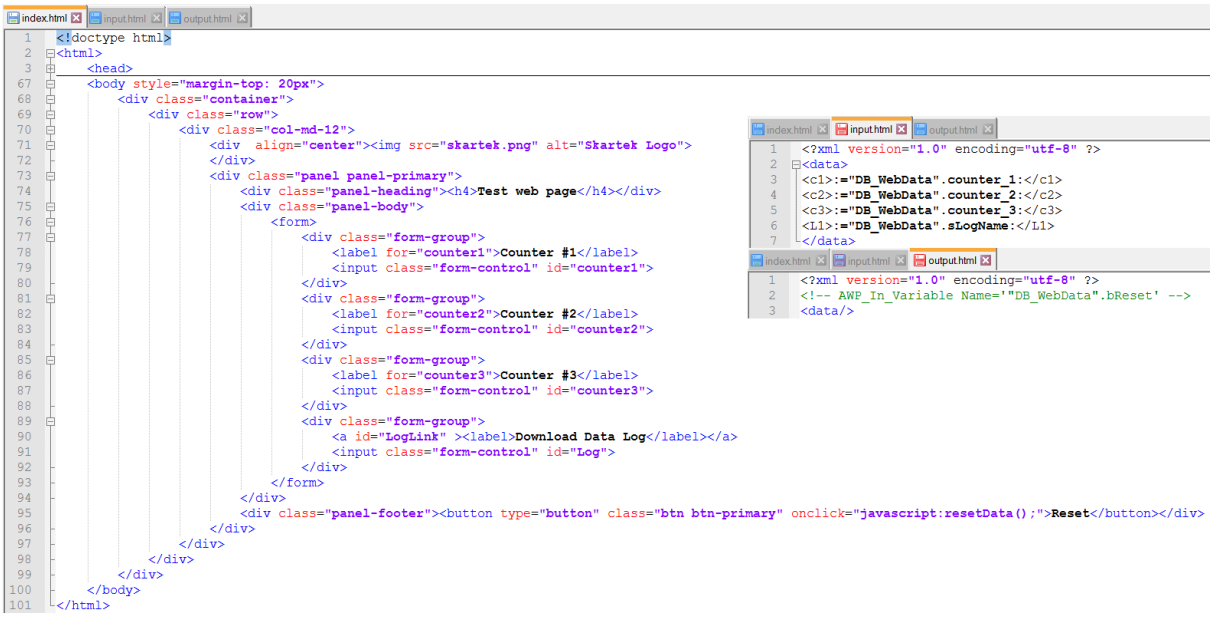


Fig. 6 *The source code of the test web page. Source: Author*

It is evident, that also some functions to link and provide the data from XML files to the main HTML file have to be used. Simple java script functions were implemented into the *index.html* file. Figure 7 shows the source code. The function *refreshData()* reads the *input.html* file and refreshes the values of the PLC tags. The function *resetData()* writes the tag to the PLC through the *output.html* file.

```
function refreshData() {

    $.ajax({
        type: "GET",
        url: "input.html",
        dataType: "xml",
        success: function (msg) {

            $("#counter1").val($(msg).find("c1").text());
            $("#counter2").val($(msg).find("c2").text());
            $("#counter3").val($(msg).find("c3").text());
            $("#Log").val($(msg).find("L1").text()+".csv");

            setTimeout(function() { refreshData() }, 1000);
        },
        error: function (jqXHR, textStatus, errorThrown) {

            $("#counter1").val("Failure! Status: " + textStatus + " Description: " + errorThrown);
            $("#counter2").val("Failure! Status: " + textStatus + " Description: " + errorThrown);
            $("#counter3").val("Failure! Status: " + textStatus + " Description: " + errorThrown);
            $("#Log").val("Failure! Status: " + textStatus + " Description: " + errorThrown);

            setTimeout(function() { refreshData() }, 1000);
        }
    });
    var adrs = "../../DataLogs/"+$("#Log").val()
    var x = document.getElementById("LogLink");
        x.setAttribute("href",adrs);

}
```

```
function resetData() {

    $.ajax({
        type: "POST",
        url: "output.html",
        data: "%22DB_WebData%22.bReset=1",
        dataType: "xml",
        success: function (msg) {
            alert("Reset successful!");
        },
        error: function (jqXHR, textStatus, errorThrown) {
            alert("Failure! Status: " + textStatus + " Description: " + errorThrown);
        }
    });

}
```

Fig. 7 *The source code of the functions. Source: Author*

To be able to use the advantages of JavaScript libraries just to avoid scripting of standard functions, it is necessary to include the library to the *index.html* file. For test purposes the JQuery library was used (jQuery 2016). Similarly also the style and formatting of the web page could be done by including CSS stylesheets (Bootstrap 2016).

It is evident that every file added to operate the user web page increases the consumption of the PLC memory. Two approaches of the web page creation were used to show the demands on the system resources.

The first way (case A in further text) is to add all incorporated files (*.js, *.css) into the user web page folder and load them to the PLC memory. Of course, the more complex web page is implemented, the higher the demands on the PLC memory will be. However, if the fact is taken into account, that the JavaScript is executed in the web browser and that the web browser could have connection to the internet separated from the PLC network, then it is not necessary to upload the auxiliary libraries to the embedded webserver. So the second approach (case B in further text) is to include the auxiliary libraries and files using URL address pointing to any PC based webserver located within the PLCs private network or even to a webserver on the internet. This way of linking stylesheets and libraries to the user web page is explained by the following figure.

```
<head>
    <meta charset="utf-8">

    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.0/jquery.min.js"></script>
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap-theme.min.css">
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>

    <script type="text/javascript">
</head>
```

Fig. 8 *Linking of the aux. files from location outside of the PLC memory. Source: Author*

The further text is focused on comparison of both approaches and their influence on the PLC memory requirements.

RESULTS

First of all, the comparison of the summary of file sizes is presented. As could be seen from Fig. 9 the difference is remarkable already at that simple application. The optimized version B is approximately nine times smaller than version A.

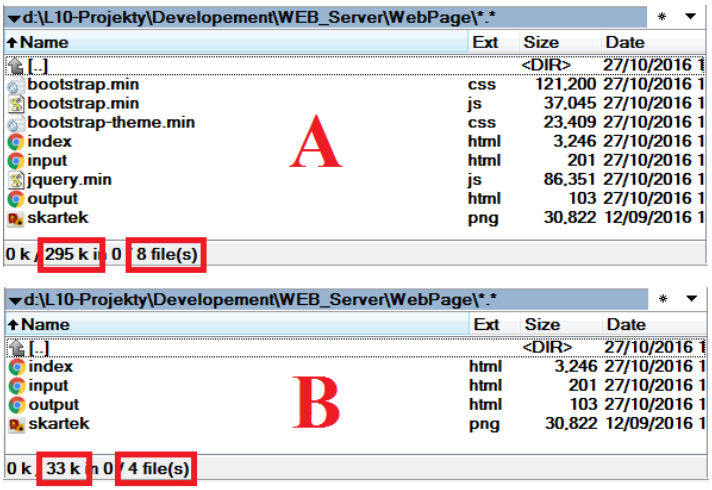


Fig. 9 *Comparison of the summary size of auxiliary files. Source: Author*

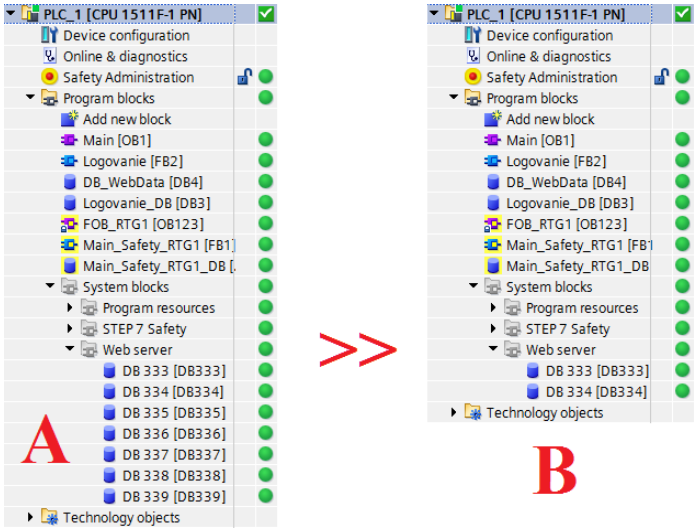


Fig. 10 *Comparison of the amount of consumed PLC data blocks. Source: Author*

The higher count of auxiliary files leads to a higher amount of PLC data blocks as shown in Fig. 10. While the non-optimized version A consumes six fragment data blocks, the optimized version B needs only one.

Finally, the comparison of consumed PLC memory space is shown in Fig. 11. It is evident, that the non-optimized version A puts remarkably higher demands on the PLC memory space than the optimized version B. The difference is approximately 3%, which is a big portion of memory space from the PLCs point of view.

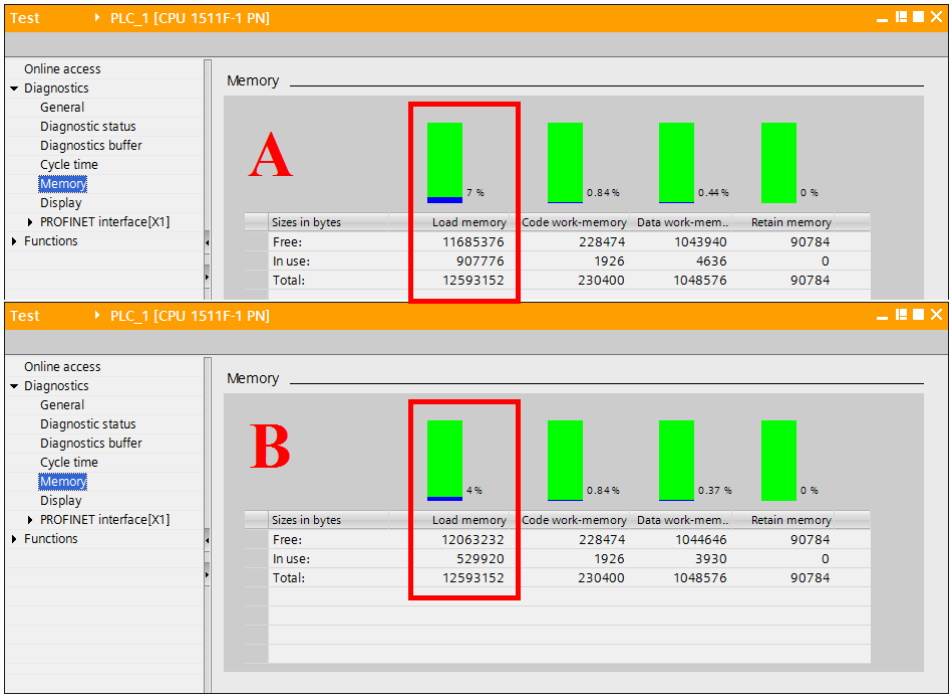
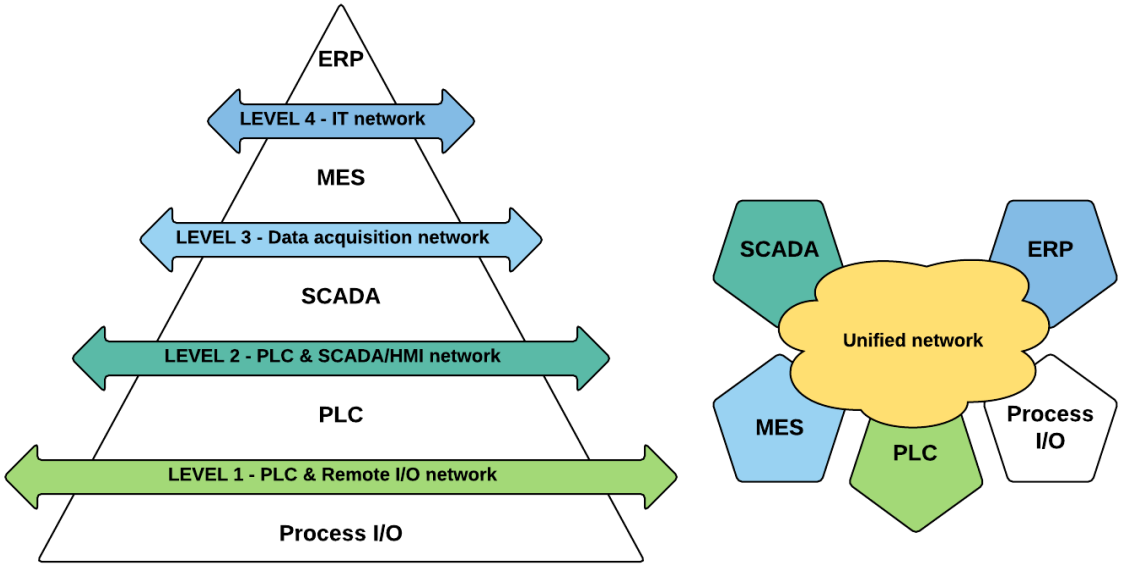


Fig. 11 *Comparison of the consumed PLC memory space. Source: Author*

DISCUSSION AND CONCLUSION

The experiment described in this paper offers a possible approach to the application of embedded PLC web server with the optimal use of system resources as proven by the above presented results. The embedded PLC web server in conjunction with an auxiliary PC based web server offers a wide range of user web pages utilization besides the production data acquisition e.g. linking the device documentation. The application of these technologies appears to be much easier and less hardware demanding, assuming the transformation of the traditional pyramidal model of decentralized control systems in accordance with the Industry 4.0 concept as shown in Fig. 12 and keeping the principles of computer networks security.

As stated in the introduction of this paper the principles and technologies of the Industry 4.0 concept could be used with advantages also for education. The specialized institute laboratories consist of complex models of industrial processes e.g. water treatment, food and beverages production or assembly line. The ongoing research is besides the transformation of the control systems due to Industry 4.0 also focused on the application of technologies such as RFID, QR code, IoT, virtual models, etc. to operate the mentioned laboratories and making the process of teaching more efficient. One of the primary aims is to motivate students by incorporating their favorite gadgets (tablet, cell phone) into the education. Another important aim is to support the teacher during the exercises and during debugging of the students work or maintenance.



Traditional pyramidal model *Industry 4.0 model*

***Fig. 12** Comparison of decentralized control systems models. Source: Author*

An application example could be marking separate parts of the complex model factory, for example with a QR code. This code could include a link to a user web page on a related embedded PLC web server. The user page could contain actual diagnostic information, data of the simulated production and other information related with the PLC job, on the other hand there could be links redirected to an auxiliary PC based web server for downloading system documentation (PI&D diagram, electrical and / or pneumatic diagrams, operator manual etc.) and last but not least assignments for exercises and projects.

The testing of the embedded PLC web server with the respect to the system resources and the vision of applying the Industry 4.0 concept into education processes could be considered as the main benefits of this article.

Acknowledgement

At this point the author wants to thank to J. Imrich for the valuable discussion and advice about scripting.This publication is the result of implementation of the projects: "Modernization of the Automatic Control Hardware course by applying the concept Industry 4.0" (KEGA 040STU-4/2016) and "UNIVERSITY SCIENTIFIC PARK: CAMPUS MTF STU - CAMBO" supported by the Research & Development Operational Program funded by the EFRR.

References:

1. Creating User-defined Web Pages on S7-1200 / S7-1500 – Application description. SIEMENS ©2015, [cit. 2016-10-30]. Entry ID: 68011496 Available online: https://support.industry.siemens.com/cs/ww/en/view/68011496
2. jQuery library. The jQuery Foundation ©2016, [cit. 2016-10-15]. Available online: http://jquery.com/
3. Web page theme. Bootstrap ©2016, [cit. 2016-10-15]. Available online: https://www.bootstrapcdn.com/

4. SIMATIC S7-1500, ET 200SP, ET 200pro Web server – Function manual. SIEMENS ©2016, [cit. 2016-10-30]. Document ID: A5E03484625-AE Available online: https://support.industry.siemens.com/cs/ww/en/view/59193560

ORCID:

Michal Kopček 0000-0001-8466-598X