Utilization of the PLC as a Web Server for Remote Monitoring of the Technological Process

Ján KAČUR, Milan DURDÁN and Marek LACIAK
Institute of Control and Informatization of Production Processes,
Faculty BERG, Technical University of Košice,
Košice, Slovak Republic,
E-mail: jan.kacur@tuke.sk

***Abstract*—This paper presents how it is possible to utilize PLC in the task of the Web server. Described is configuration of the PLC and its connection to the computer network. Next is described principle of creation dynamic web page and described are function for process variable data read and write. User can create dynamic web page that uses combination of some programming language e.g. HTML, ASP (Active Server Pages) and Java Script. The process variables can be displayed on remote PC independently on operating system. There is only web browser needed. In the paper is also described principle of the desktop application that was programmed in the Java language. This application directly communicates with remote PLC and has own visualization of the technological process. For communication with the PLC the opened and wide HTTP web protocol is used. Described principles present some alternative for well-known technology OPC. Inscribed principles were verified in application for temperatures measurement in the underground gasification process and for system of indirect temperature measurement.**

Keywords— PLC; remote monitoring; HTML; ASP; Java; Java Script; process visualization

I. INTRODUCTION

Ethernet/internet-based data monitoring brings new capabilities and unprecedented access to process measurement and control. Using standard process sensors, such as thermocouples pressure transducers, flow meters or other sensors that produce a standard analog or pulse output, we can monitor, control or log data in almost any location across the hall, on the other side of the street, across town, on opposite ends of the country, or anywhere around the world. Using the power of the internet the remote monitoring takes on new meaning. With Ethernet-based, internet-enabled instrumentation and monitoring system, remote access can be anywhere a smartphone has a signal. From the simplest application, viewing data through the web-browser on your iPhone, Blackberry, Android device or laptop, to more sophisticated uses, such as sending a text or e-mail message when an alarm occurs, or transmitting a data log file over the internet from a remote location to a central office. A user can access this data anytime, anywhere, 24 hours a day, 365 days a year, wherever you have internet access. Many application and instruments can now log data to an SD card; the data can be read directly on a PC or Mac, or can be downloaded remotely, over your Ethernet network or the internet. Modern application technology has blurred the definition of remote access in recent years. As both people and services can require remote access, the suitability of technological solutions becomes hard to discern. Many communications protocols are being encapsulated in web traffic and web-based applications are managing increasingly complex communications inside HTTP and HTTPS channels. Further, internet-facing web-based applications are becoming universally accessible; yet, they manage the security of the data that they are presenting to various user communities, in some respects the data are being brought to the consumer [1, 2, 3].

In this paper we regard two technological processes: steel coils annealing and underground coal gasification (UCG). In the first case we regard only temperature measurement especially indirect measurement of the inner temperatures based on mathematical model. In the second case we regard temperatures measurement for experimental gasification on laboratory gasifier. In both of these industry fields was performed a wide research on our workplace [4, 5].

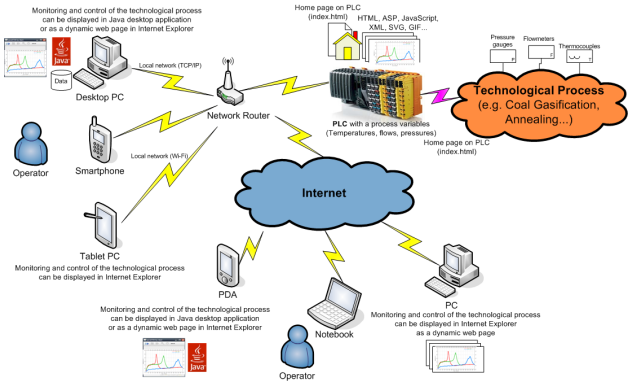

Figure 1. Ideological scheme of the remote monitoring.

Control center for monitoring of the process and performing manual actuating to the process must be placed on safety distance of heat aggregate. Interconnection of operator's PC for visualization and monitoring with control PLC may be provided through industry bus or LAN and TCP/IP protocol. The advantage of TCP/IP is possibility of usage wireless connection e.g. Wi-Fi. There is possible to provide monitoring from any PC with wireless networking technology support.

978-1-4673-4490-6/13/$31.00 ©2013 IEEE

We propose two principles of the remote monitoring. In the first presented principle a dynamic web page is placed on PLC. Page is dynamically updated on a remote PC in the Internet Explorer on some PC or mobile equipment (chapter V). In the second the monitoring is performed by self-programmed application in the Java programming language with utilization of its HTTP capabilities (chapter VI). There the web page is not needed. Values of the process variables are directly transferred between PLC and remote PC. In both of those presented principle the B&R PLC (class X20) as a web server is used (Fig. 1)

II. PLC AS A WEB SERVER

Starting with Version AR3.08 (Automation Runtime), the B&R Automation Runtime SG4 platforms are equipped with a web server. The firmware (marked as SG in a various versions) is a component of the PLC operating system of B&R Automation Runtime. An important characteristic of web pages is that they allow information to be accessed and represented independently of which operating system or tool (browser) is used. With a web page on the machine, the most important data can be called up from any terminal, PC, laptop or web-capable device on the intranet if necessary. The Web interface provides neutral, secure and universal access to the automation. Interactive Web sites also allow operation of a machine or system. The web pages can be created using regular HTML editors. Automation Runtime process variables can be output or modified using web pages. A standard web browser can be used to call up web pages and files from the B&R web server. This can also be done using a Java or .NET application. Web pages can be more than simply called up in a browser. They can also be requested and further-processed within other applications. The B&R web server can be used for a variety of actions: from simple requests sent by files stored or created on the PLC to the processing of collected data via dynamic pages and Java script for diagnostics purposes or in an application running on a client (e.g. written in .NET or Java). The web server is available for SG4 starting with AR 2.82. MIME types are used to describe the type of data transferred (e.g. text/html, text/css, image/gif, image/ico, text/javascript, image/jpeg, video/mpeg, image/png, image/svg+xml, text/plain, text/xml).

Hypertext Transfer Protocol (HTTP) is a standard Client/Server Internet protocol that transfers web content over a TCP/IP network. The Programmable Logic Controller (PLC) with an Ethernet module acts as an HTTP Server (web content provider), which responds to requests from HTTP Clients (web browsers). The PLC supports HTTP Client authentication, requiring the web browser to enter a Username and Password, before web content can be accessed. Properly authenticated, an HTTP Client can access web content previously stored in the PLC file system. This web content can include standard static HTML files and other web resource files (images, sound, etc.). In addition, the PLC HTTP server allows the HTTP Client to display and edit PLC register data, by embedding specially named dynamic HTML files into a static HTML file. To start browsing web resources stored in the PLC file system, enter the appropriate Uniform Resource Locator (URL) into the HTTP Client (web browser). As a minimum, the URL should include the target PLC's IP address and a filename. For example, if the PLC's IP address is 147.232.37.14, the following URL will access the PLC index.html file in the PLC file system root directory:

http://147.232.37.14/index.html

A PLC HTML file is either static or dynamic. These files can be created with a simple text editor or by using a 3rd party HTML Editor. Files saved from the application can be viewed or copied from any PC in the network using a standard browser. No additional software needs to be installed. The web server makes it possible to display and enter process values in a standard Internet browser, as well as to use them in web-capable client applications. Clients can utilize web-capable languages such as ASP, or JavaScript to access the dynamic web pages on the web server. The open and widely used HTTP web protocol is used. This allows for manufacturer-independent solutions. Data is exchanged via dynamic pages or the web service integrated in the B&R web server. This makes it possible to access any process variable on the PLC. The web service functionality can also be configured to improve security. As a result, it is simple to display or modify process variables in an application. HTML pages to be used on the B&R web server can be created quickly and easily in standard HTML editors. They can also be created in standard text editors. With a real PLC or simulated (virtual PLC – Arsim), the result can be viewed immediately in a browser (Fig. 2).

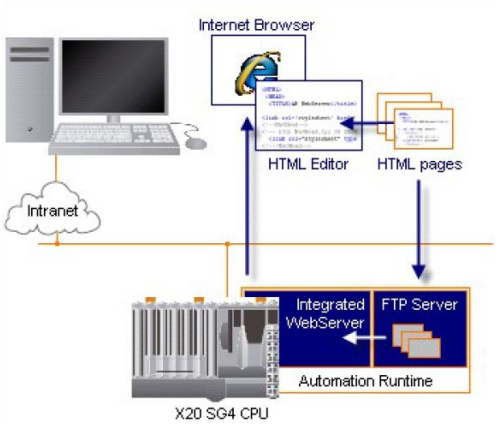

Figure 2. Scheme of the PLC utilization as a Web server

Possible applications:

- Remote maintenance
- Data collection
- Processing collected data
- Open and closed loop control
- Configuring parameters

Another way for continually transmission of monitored values into remote PC is utilization of OPC (OLE for Process Control) technology. This technology enable to transmission values of the process variables from PLC or panel into remote PC where OPC server runs. Windows OPC server runs like resident application and data of the process variables provides for applications of type OPC client. As OPC client the suitable

are applications like SCADA/HMI systems e.g. InTouch, Promotic, Control Web and other. For PLC and panels from Bernecker and Rainer may be used own B&R PVI OPC server or it is possible to use Windows OPC server. Built-in B&R OPC server runs directly on target (B&R PLC or B&R Power Panel). Disadvantage of this solution is a little complicated configuration of the project. Everyone variable must be configured in the OPC declaration file as e.g. OPC tag and in the OPC Mapping file where is specified which variables are accessary on data transfer between PLC and PC (Fig. 3). This configuration is not needed when the web server is used for data transfer.

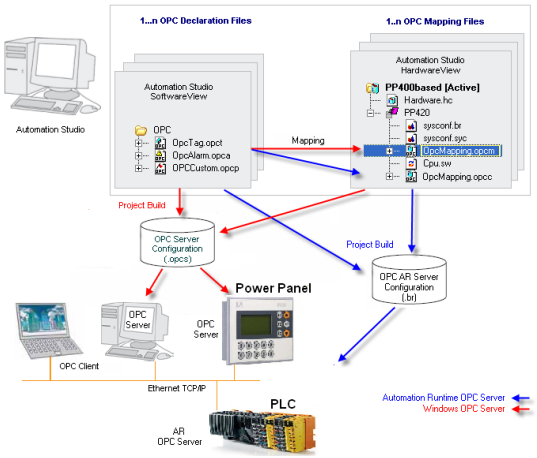

Figure 3. Dataflow of the OPC configuration

When built-in Web server is active user can save complete structure of web application (web page) into PLC or Power Panel. Web application can contain own visualization of measured temperatures. Source files of web page may be copied into PLC via FTP protocol. With the help of FTP client files must be copied into web directory e.g. ftp://147.232.37.14/Web placed on CompactFlash card (directory F:\web\) which is inserted to the PLC or Power Panel. For displaying of the process variable on web page is needed to use scripting language ASP (Active Server Pages) or JavaScript. Files of type HTML or HTM we can use for displaying of static parts of the web page. This page we can display on remote PC through internet explorer by typing of IP address of the PLC or Panel [6].

III. CONFIGURATION OF THE PLC FOR REMOTE MONITORING

In Automation Studio (AS), the web server is configured in the physical view, under the node for the CPU, by selecting Open / AR Configuration from the menu. Starting with AR 3.08, the web server can be configured directly in AS. The default web server configuration can be used right away. The most common MIME types are pre-configured so that web pages will be correctly displayed in the browser. On Fig. 4 we can see example of the AR configuration window where user can setups web page directory, port number, default index page DNS parameters, ASP Goform service and MIME types. On Fig. 5 is depicted Ethernet configuration window of the automation project. In this window user can performs setup of the PLC IP address, mask, and port number. Also there is possible to use DHCP. In practice we use B&R PLC X20 CP 1483-1 with 100 MHz CPU, 64 KB DRAM, 128 KB SRAM with TCP/IP Ethernet interface and several I/O X20 modules.

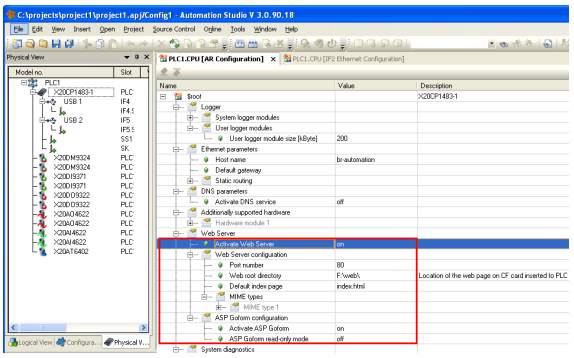

Figure 4. Configuration of the PLC web server in the AS project

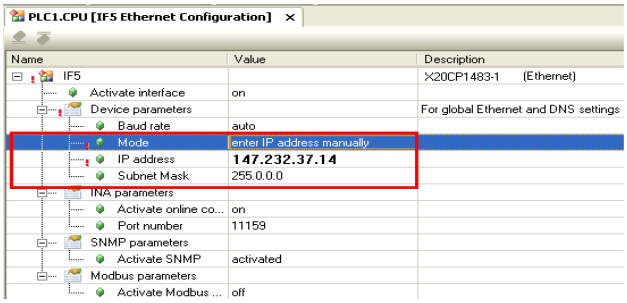

Figure 5. Configuration of the PLC IP address in AS

IV. PROGRAMMING OF THE WEB PAGES FOR PLC

Active page elements are different than static page elements because parts of the content are added to the document when the request is made. The page stored on the CompactFlash card contains instructions, which are replaced by the respective contents before being sent out. This means that the page sent out is no longer a 1:1 copy of the page on the server. When using the B&R web server, process variables can be read and written using ASP commands. The use of ASP users makes it possible to define groups with certain permissions for accessing individual document contents. A page requested by the client is first parsed in the server and the embedded ASP commands are executed. The server replaces the tags embedded in the page with the corresponding process value and passed on to the browser. This makes it possible to send pages with dynamically generated contents in addition to static pages. The following ASP commands can be used:

- ReadPLC("VariableName"); - Outputs the value of a process variable in the page that is called up.
- WritePLC("VariableName", "Value"); - Writes values specified on the page to process variables when a page is called up.
- EnvPrint("Parameter"); - Outputs environment variables of the web server on the page.
- WebPrint("Parameter"); - Outputs a parameter on the page that is transferred in the URL when the .asp page is called.

Interactive commands can be addressed using forms. The necessary information is recorded in the form. When the form is sent, the browser generates a request to the ASP Goform web service. Simple ASP commands make it possible to add current PLC values to web pages and other documents when called via the server or to change specific values by calling up pages. List of possibilities:

- Access process variables
- Read status information
- Set and read process-relevant PVs
- (Colored) display of PV information
- Status-dependent display
- Display dynamic HTML pages
- Create HTML pages with auto refresh
- Set page-dependent PLC states
- Read environment variables from the web server
- Dynamic extension of web pages
- Control visibility and controller based on easy group rights.

ASP Goform is a web service. Interactive page elements (e.g. forms) provide options for entering values that can be transferred to the PLC. When using the B&R web server, process variables can be written and read on the server using ASP commands or on the client using the ASP Goform web service. The ASP Goform web service can be addressed in the browser using forms or JavaScript. In addition to this, any web-capable client is able to write or read PVs using the ASP Goform interface. Application possibilities

- ASP Goform independent of ASP commands
- ASP Goform with JavaScript
- ASP Goform solely parameter transfer / request - without redirect evaluation
- ASP Goform with ASP as mix

ASP commands and ASP Goform access can be optimally combined in the same document. Web server doesn't support integration of programs (e.g. loops, mathematical functions, etc.) or PHP scripts. A PLC's variables can be accessed using the Goform interface. This files PV_Access.js and response.asp to be located in the web directory. Access to the variables of a PLC is programmed in the PV_Access.js file. The user can include the JavaScript in his ASP file and use the functions pvAccess.WritePV and pvAccess.ReadPV to access the variables and to read or write the values. Transferring via FTP is best when developing web pages, since it allows the files to be sent to and from the target system quickly and easily. A functioning Ethernet connection to the target system is a requirement for transferring web pages to the web server. In order to establish a connection you need to know the IP address or host name of the PLC. When accessing via a remote network or a proxy, an FTP connection is only possible when the respective ports are open. Next follow practice applications.

V. Remote Monitoring for Annealing Process and Indirect Measurement

In term of the steel coils annealing quality is needed to know temperature inside the batch and places with the lowest temperature during annealing process. On present in the world is this temperature non-measurable in a real conditions and annealing time is estimated by empirical. At first in the annealing process the steel coils must be warmed and then must be enabled their cooling in line with technological rule defined for given type of the steel. Operator must regard requirement of minimization specific consumption and maximization quality of annealed coils for given type of the bell furnace. This problem is possible to solve with system of indirect measurement of inner temperatures in the steel coil. Proposed system is based on direct measurement of atmosphere temperature in furnace space between protection bell and coil. Thermocouple measure the atmosphere temperature, which inputs as a value for software realization i.e. mathematical model created for calculation of temperature array inside of massive batch. Model performs continually calculation based on measured atmosphere temperature. The task of proposed system of indirect measurement is from direct measured atmosphere temperature by indirect to measure temperatures on the batch's surface and of them to calculate (to indirect measure) temperatures inside the batch [7, 8].

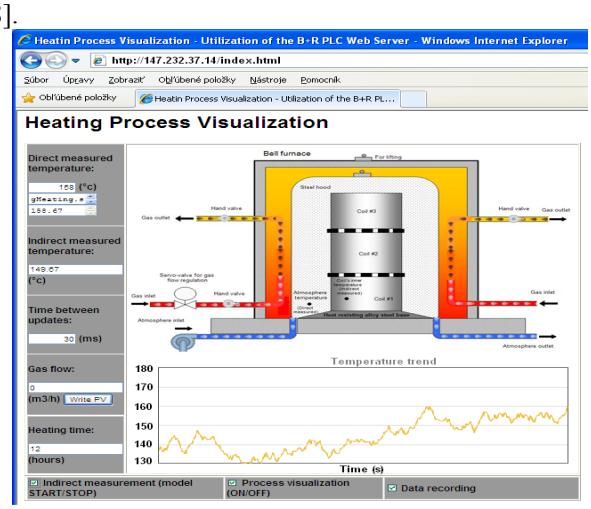

Figure 6. Dynamic web page of the temperature measurement in annealing

For remote monitoring of the temperatures in the annealing process in the bell furnace we proposed to use monitoring based on the dynamic web page (Fig. 6) and utilization of the PLC or Power Panel. PLC should be connected into local network of industry hall through network router. At the same time router provides connectivity of another computers through cabling of company intranet or Wi-Fi. PLC has built-in 100 Mbits Ethernet interface. If router is switch-on then setups reserved IP address for connected PLC. PLC is possible to configure through B&R Automation Studio. As a remote device for page view can be used PC, Tablet PC or Smartphone with OS Android. A PLC's variables can be accessed using the Goform interface. The files PV_Access.js and response.asp must be located in the web directory. Access to the variables of a PLC is programmed in the PV_Access.js file. We

included these JavaScript in our HTML file and used the functions pvAccess.WritePV and pvAccess.ReadPV to access the variables and reading or writing the values.

```
pvAccess = {}
pvAccess.Func = function()
{
  function AccessPV(name, rValue, wValue)
  {
    var url = '/goform/ReadWrite';  var data = 'redirect=/response.asp&variable=' + escape(name);
    if(rValue != null && rValue != "")
    { data += '&value=' + escape(rValue); data += "&write=1"; }
    else
    { data += '&value=none'; data += "&read=1"; }
    var xmlHttp = null;
    try { xmlHttp = new XMLHttpRequest(); // Mozilla, Opera, Safari sowie Internet Explorer (ab v7)
    } catch(e) {  try { xmlHttp  = new ActiveXObject("Microsoft.XMLHTTP"); // MS Internet Explorer (ab v6)
        } catch(e) { try { /* MS Internet Explorer (ab v5)*/ xmlHttp  = new ActiveXObject("Msxml2.XMLHTTP");
            } catch(e) { xmlHttp  = null; }
        }
    }
    if (xmlHttp) { xmlHttp.open('POST', url , 1);  xmlHttp.onreadystatechange = function () {
          if (xmlHttp.readyState == 4) {
            if(wValue != null) {
              wValue[3] = xmlHttp.responseText;  wValue[3] = wValue[3].replace("<!-- B&R ASP Webserver -->","");
              wValue.value = wValue[3]; return wValue;
            }
          }
      );
      xmlHttp.send(data);
    }
  }
  {  // public function
    this.WritePV = function(name, value)
    {  ccessPV(name,value); }
    this.ReadPV = function(name,wValue)
    {  return AccessPV(name, null, wValue); }
  }
}  pvAccess = new pvAccess.Func();
```

Figure 7. Demo code of the PV_access.js library

Example of the PV_Access.js as JavaScript library that was used is shown on Fig. 7. The file response.asp only call <% WebPrint("val"); %> command and is used by PV_Access.js. On Fig. 8 is depicted HTML part of the web page that paints input/output visual components on the page. This code also links libraries PV_Access.js, jquery.js and jquery.flot.js. The first script defines functions that enable to read and write process variable (ReadPV, WritePV). The second and the third script enable to paint graphs and trends. jQuery library is free for download from the internet.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
 <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <title>Heatin Process Visualization - Utilization of the B+R PLC Web Server</title>
    <link href="layout.css" rel="stylesheet" type="text/css">
     <!--  Utilization of the jquery libraries for data trend plotting -->
    <script language="javascript" type="text/javascript" src="/jquery.js"></script>
    <script language="javascript" type="text/javascript" src="/jquery.flot.js"></script>
    <!--  Utilization of the PV_Access library where function for PLC variables reading are placed -->
    <script src="/PV_Access.js"></script>
 </head>
 <body> <h1>Trend of temperatures</h1>
  <div id="placeholder" style="width:600px;height:300px;"></div>
<!--  ... Visualization of the process graphic and I/O visual elements (input output) ...-->
<h1>Heating Process Visualization</h1>
<table width="91%" border="1">
  <tr>
    <td width="26%" bordercolor="#999999" bgcolor="#999999">
      <p><strong>Direct measured temperature:</strong></p>
      <p>
        <input id="measuredVal" type="text" value="" style="text-align: right; width:5em">
        <strong> (°C) </strong>
        <textarea id="Variable10"  name="Variable10" rows="1" cols="10">&lt;Name of PV&gt;</textarea>
        <textarea id="Value10"  name="Value10" rows="1" cols="10">&lt;value&gt;</textarea>
        </p></td>
  <!--  ... Visualization based on SVG graphic ...-->
  </tr>
   <!--  ... Another visual element ...-->
</table>
```

Figure 8. Demo code of the HTML part process visualization page

These scripts are included in the followed JavaScript part where ReadPV or WritePV functions are used (Fig. 9). The final web page depicted on Fig. 6 shows bell furnace with some coils and some text box for display of the measured temperatures or temperature from indirect measurement if model would be implemented on PLC [9, 10]. The web page is prepared also for actuating of the annealing by entering of required gas flow to the bell furnace. This page can be used for annealing monitoring on a real industry plant and can be displayed in the web browser on a various device (PC, Tablet PC, or Smartphone).

```
<script type="text/javascript">
$(function () {
    var data = [], totalPoints = 300;
    function get_PLC_Data() { // Reading one temperature from PLC
      var Variable = "gHeating.status.actTemp";
      var Value = "%f.1f";  // PLC value data format
      $("#Variable10").val(String(Variable));  // name of the PLC variable is written to the EditBox
      pvAccess.ReadPV(document.getElementById('Variable10').value, document.getElementById('Value10') );
      measuredVal.value = parseInt( parseFloat(Value10.value) ) ;
      return measuredVal.value;
    }
    // ...
    function getMyData() { // Realtime data reading from PLC
     if (data.length > 0) data = data.slice(1);
        while (data.length < totalPoints) {
            var prev = data.length > 0 ? data[data.length - 1] : 50;
            var y = get_PLC_Data();  if (y < 0) y = 0; if (y > 100) y = 100;
            data.push(y);
        }
        var res = []; for (var i = 0; i < data.length; ++i) res.push([i, data[i]])
        return res;
    }
    // ...
    var updateInterval = 30; // Initial refresh interval
    $("#updateInterval").val(updateInterval).change(function () {
      // ...here is code for some conditional setup of the refresh interval
    });
    // Setup plot and plotting
    var options = { series: { shadowSize: 0 }, yaxis: { min: 0, max: 1500 }, xaxis: { show: false } };
    var plot = $.plot($("#placeholder"), [ getMyData() ], options);
    function update() {
        plot.setData([ getMyData() ]); plot.draw(); setTimeout(update, updateInterval);
    }
    update();
});
</script>
```

Figure 9. Demo code of the JavaScript part process visualization page

VI. REMOTE MONITORING OF THE GASIFICATION PROCESS

Second presented principle for remote monitoring is utilization of the Java capabilities. Underground Coal Gasification is a method for energy obtaining from underground coal bed without utilization complicated mining machines ways. This method uses principles of thermal coal decomposition where syngas is produced with help of injected oxidizer [11]. Automated monitoring and control of the UCG sometimes need intervention of the specialists. Operator must know adequate to react on various states of underground gasifier e.g. temperature or calorific value decrease in reaction zone. The quality transformation of the coal to the syngas is necessary periodically to inspect and to analyze measured process variables (e.g. temperatures). Control PLC must be placed around of the equipment for measurement and control considering length of the signal wires). Control center for process monitoring and manual control intervention can be placed also outside gasifier operation. Even if explosion or poisoning risk on underground gasifier operation is lower than in the case of conventional mining, remote monitoring is comfortable and flexible form of the process control. Of course there exists a lot of various another ways how to perform remote monitoring. We propose the principle where monitoring is performed by self-programmed application in the Java programming language ant its HTTP support capabilities. In the laboratory testing of the UCG were used two laboratory gasifiers (generators), control PLC B&R X20 and powerful desktop computer with OS Win XP. A remote monitoring was tested later apart from the real experiments. There was created desktop Java application that runs on a JVM (Java Virtual Machine) on a various operating system (e.g. Linux, Windows). The main window of this application is depicted on Fig. 10. This process picture monitors temperatures from experimental coal gasifier. For each thermocouple is created process variable on PLC. Measured temperatures are through these variables and HTTP protocol transferred to the remote computer. Application runs as web client and measured temperatures are displayed in an edit boxes and text fields. Graphic user interface (GUI) of this application was created in Easy Eclipse IDE that has built in Visual Editor. The several graphical components from the Java Swing and Awt package can be used by drag and drop technique on the main form. For reading data from the web

server (PLC) was created library the BR_ASP_Request.java where basic readPV and writePV methods (class functions) are programmed (Fig. 11).

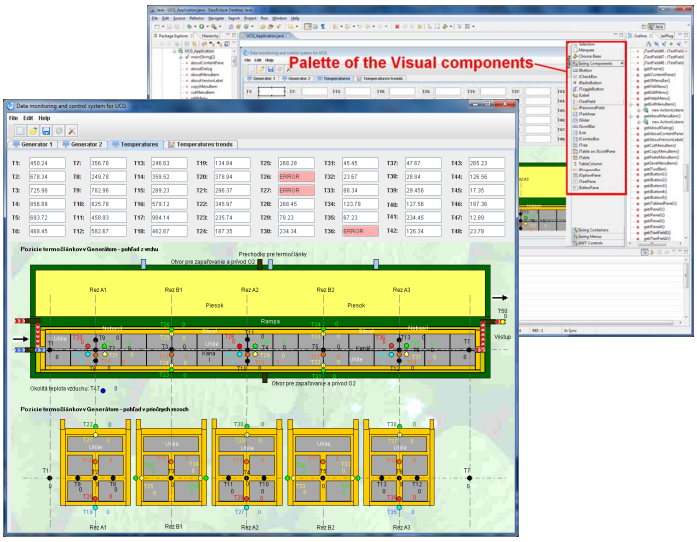

Figure 10. Java Desktop application for temperatures monitoring in gasifier

```
package BR_ASP;
import java.io.BufferedReader; import java.io.InputStreamReader; import java.io.OutputStreamWriter;
import java.net.URL; import java.net.URLConnection; import java.net.URLEncoder; import java.util.logging.Level;
import java.util.logging.Logger;

public class BR_ASP_Request
{  public static String readPv(String variable, String target)
    { try { String data = URLEncoder.encode("redirect", "UTF-8") + "=" + "/response.asp";
            data += "&" + URLEncoder.encode("variable", "UTF-8") + "=" + URLEncoder.encode(variable, "UTF-8");
            data += "&" + URLEncoder.encode("value", "UTF-8") + "=";
            data += "&" + URLEncoder.encode("read", "UTF-8") + "=" + URLEncoder.encode("1", "UTF-8");
            URL url = new URL("http://" + target + ":80/goform/ReadWrite");
            return pvAccess(data, url); }
        catch (Exception ex) { Logger.getLogger(BR_ASP_Request.class.getName()).log(Level.SEVERE, null, ex); }
        return "Error"; }

    public static String writePv(String variable, String value, String target)
    {  try { String data = URLEncoder.encode("redirect", "UTF-8") + "=" + "/response.asp";
             data += "&" + URLEncoder.encode("variable", "UTF-8") + "=" + URLEncoder.encode(variable, "UTF-8");
             data += "&" + URLEncoder.encode("value", "UTF-8") + "=" + URLEncoder.encode(value, "UTF-8");
             data += "&" + URLEncoder.encode("write", "UTF-8") + "=" + URLEncoder.encode("1", "UTF-8");
             URL url = new URL("http://" + target + ":80/goform/ReadWrite");
             return pvAccess(data, url); }
        catch (Exception ex) { Logger.getLogger(BR_ASP_Request.class.getName()).log(Level.SEVERE, null, ex); }
        return "Error"; }

    public static String pvAccess(String data, URL url)
    {   String line = null; String response = "";  try { URLConnection conn = url.openConnection();
              conn.setConnectTimeout(3);  conn.setDoOutput(true);
              OutputStreamWriter wr = new OutputStreamWriter(conn.getOutputStream());
              wr.write(data);  wr.flush();  wr.close();
              BufferedReader rd = new BufferedReader(new InputStreamReader(conn.getInputStream()));
              while ((line = rd.readLine()) != null)
              {  line = line.replace("<!-- B&R ASP Webserver -->","" ); response += line; }
              wr.close(); rd.close(); return response; }
        catch (Exception e) {   System.out.println(e.toString()); return e.toString(); } } }
```

Figure 11. Demo source code of the BR_ASP_Request.java library

```
package BR_ASP;
import java.awt.Font;  import java.awt.event.ActionEvent;  import java.awt.event.ActionListener;
import java.awt.event.ItemEvent; import java.awt.event.ItemListener; import javax.swing.ImageIcon;
import javax.swing.JButton;  import javax.swing.JCheckBox; import javax.swing.JFrame; import javax.swing.JLabel;
import javax.swing.JPanel; import javax.swing.JTextField; import javax.swing.Timer;

public class WebServerTest extends JFrame{
  ActionListener update; // event management
  Timer timer;  // for automatic data refresh
  JLabel label1;  // text label for thermocouple number visualization
  JTextField field1;  // text field for data output
  JLabel picLabel;  // for graphic scheme visualization

  public WebServerTest() { // constructor of the class
        JPanel panel = new JPanel(); panel.setLayout(null);
        label1 = new JLabel("T1: ");  label1.setFont(new Font("Serif", Font.PLAIN, 12));
        label1.setBounds(10, 20, 150, 80);
        field1 = new JTextField();  field1.setBounds(100, 45, 150, 25);
        // checkbox for timer on/off
        JCheckBox checkbox = new JCheckBox("Read data", false);
        checkbox.setFocusable(true); checkbox.setBounds(270, 35, 300, 50);
        // visualization of the gasification generator scheme
        picLabel = new JLabel(new ImageIcon("scheme1.PNG")); picLabel.setBounds(50, 250, 500, 500);
        // Timer "tick" - timer event management - data refresh
        update = new ActionListener() { public void actionPerformed(ActionEvent actionEvent) {
        String picValue = BR_ASP_Request.readPv("gHeating.status.actTemp", "147.232.37.14"); // reading PLC data
        field1.setText(picValue); } };
        timer = new Timer(100, update);  // setup initial timer period (100 ms)
        checkbox.addItemListener(new ItemListener() { public void itemStateChanged(ItemEvent event) {
        // We determine, whether the check box is selected. Then we add or remove the listener.
        if (!timer.isRunning()) {  timer.start();  } else { if (timer.isRunning()) { timer.stop(); } }}  });
         panel.add(picLabel); panel.add(field1); panel.add(label1);  panel.add(checkbox); add(panel);
         pack(); setDefaultCloseOperation(EXIT_ON_CLOSE); setTitle("Web server test");
         setSize(640, 320); setResizable(true); setLocationRelativeTo(null); setVisible(true);  }

  public static void main(String[] args) { WebServerTest myWebServerTest = new WebServerTest();  } }
```

Figure 12. Demo source code of the data visualization in the Java window

These methods are used for process data reading and writing and were implemented in the final Java application (Fig. 10). Sample source code where readPV function is implemented for cyclic display one temperature to the text field is shown on Fig. 12. Data update is performed through the Timer as a non-visual component and its event management.

VII. Conclusion

In this paper were presented a various ways for remote monitoring of the technological process. There were presented principles where PLC was used as Web server and data was transferred through HTTP protocol. Data monitoring was demonstrated on two practical application – annealing and coal gasification. Both of this industry fields were experimentally studied in the recent research. For temperature monitoring during annealing was proposed dynamic web page based on HTML, JavaScript and ASP Goform web service. The page is accessible from PC or from various remote devices connected to the network. There was created Java application for gasification that can runs on the remote PC. In this case the web page is unnecessary. Application directly communicates with PLC through the web service. Measured temperatures from the experimental gasifier are displayed through Java visual components. Both of these applications can be useful in the future research of the mentioned fields as an alternative to the OPC technology.

Acknowledgment

This work was supported by the Slovak Research and Development Agency under the grant VEGA no. 1/0746/11, and VEGA no. 1/0729/12.

References

[1] P Chandra, D. M. Dobkin, “Wireless Networking: Know It All”, Elsevier , 2008.

[2] D. Caro, “Wireless Networks for Industrials Automation”, 3rd eidition, ISA 2008.

[3] I. Leššo, P. Flegner, K. Feriančiková, “System as a basic tool in solution of tasks on the basis of system science knowledge base and informatics” In: SGEM 2012: 12th International Multidisciplinary Scientific GeoConference : conference proceedings: Vol. 4, June 17-23, 2012, Albena, Bulgaria. - Sofia : STEF92 Technology Ltd., 2012, pp. 237-244.

[4] M. Malindžáková, J. Futó, “Tunnel boring machine design and reliability parameters of a dynamic rock-indentor system”, In: Communications. Vol. 14, no. 4, 2012, pp. 79-82.

[5] M. Pástor, L. Lukáč, “Preparation and feeding the charge to the gasifying generator”, In: Acta Metallurgica Slovaca, Vol. 2, no. 1, 2011, pp. 163-167.

[6] Bernecker and Rainer Automation Studio, “B&R Help Explorer v 2.1.1” for B&R Automation Studio v 3.0.90. Bernecker + Rainer Industrie - Elektronik Ges.m.b.H, 2010.

[7] K. Kostúr, "Principles of Indirect Measurement Temperature", In: ICCC 2012: Proceedings of the 13th International Carpathian Control Conference: Podbanské, Slovak Republic, May 28-31, 2012, Piscataway: IEEE, 2012, pp. 336-339.

[8] M. Laciak, M. Durdán, J. Kačur, “Utilization of indirect measurement in the annealing process of the steel coils”, In: Acta Metallurgica Slovaca. Vol. 18, no. 1, 2012, pp. 40-49.

[9] K. Kostúr, M. Durdán, “Model for indirect temperature measurement of the massive batch = Model pre nepriame meranie teplôt masívnej vsádzky”, In: Automatizácia a informatizácia strojov a procesov. - Bratislava : STU, 2004, pp. 11.

[10] K. Kostúr, “Regulation of the heating furnace in tube rolling mill”, In: Metalurgija 45 (4) 2006 , pp. 303-306.

[11] J. Kačur, M. Laciak, M. Durdán, “Remote monitoring and control of the UCC process”, In: ICCC 2011 : Proceedings of the 12th International Carpatian Control Conference, Velké Karlovice, Czech Republic, May 2011, pp. 180-184.