Web-based Remote Control by LiveConnect

Reinhard Langmann

University of Applied Sciences Duesseldorf/Process Informatics Lab, Duesseldorf, Germany

***Abstract* — This paper deals with the application of LiveConnect for the remote control of real devices/stations over the Web. In this context, both the concept of Lean Web Automation and a flexible Java-based application tool have been developed ensuring a fast and secure process data transfer between device-server and Web browser by the subscriber/publisher principle.**

***Index Term:* Web-based remote control, Lean Web Automation, teletechnology, Web Access Kit.**

For approximately 2-3 years, standard IT methods have been made available for the automation of technical processes and have increasingly been used in this area - Internet Technology for Automation being the keyword. This covers in particular:

- the application of Ethernet and TCP/IP as communication structure between components of automation systems and
- the use of the Web browser as universal control interface for automated devices, machines and plants.

Several well-known research & development (R&D) projects deal with the extension and adaptation of the TCP/IP transmission protocol in order to be able to use Ethernet systems for real-time communication (and replace the traditional fieldbus systems). First results are available and various market leaders in automatic control engineering and industrial alliances push forward in this area [1]. The basis of this work are Ethernet and Ethernet-based local area networks. Automation functions can be distributed freely in these networks, though proprietary protocols are used occasionally. Shifting of these functions into the Internet is not intended. By using Internet technology (Web server, HTML, XML ...), however, selected functions for a limited range of tasks (e.g. remote service) can be carried out over the Internet.

In other projects Web servers (embedded Web servers) integrate automation components, thus enabling a uniform access to automation plants by means of a Web browser. These solutions are used either as Web-based extensions to existing process visualization systems (SCADA systems) or for operating/monitoring as well as for remote service purposes for embedded Web controllers over the Internet.

Fig. 1 illustrates various state-of-the-art structures of automation equipment in the Internet environment.

The well-known works assume that the basic automation functions (signal processing, control functions) remain in the local automation equipment and be performed by a controller (PLC or PC controller,

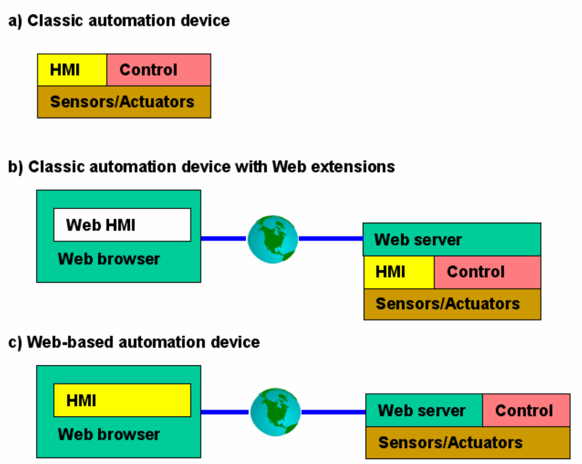

Figure 1: State-of-the-art structures of automation equipment in the Internet environment

embedded Controller). Thus no changes in the structure of the automation systems are necessary.

I. LIVECONNECT AND JAVA-APPLETS

Control and automation functions by means of Web technology need access to process data. However, the standard Web technologies are not yet ready for accessing automation interfaces. Once a fast and reliable display of data images of plants on Web pages is possible, all current Web technologies could be used and guarantee a simple and flexible set-up of distributed control configurations.

The only possibility to ensure an event-based and fast display of external data (process data) on a Web page is the LiveConnect principle [2]. *LiveConnect* stands for a mechanism within the Web browser enabling communication between Java Applets and Plug-ins on the one side and for JavaScript on the other. The use of Java Applets in particular (contrary to ActiveX objects) results in reliable, platform-independent and therefore Web-usable solutions.

For the developer LiveConnect essentially presents itself as a definition of interfaces which are used when developing a Java applet.

Basically, communication between JavaScript and Java applet can run bi-directionally. On the one hand an applet can be controlled by method calls by means of JavaScript, and on the other an applet has access to the DOM (Document Object Model) and thus on the JavaScript objects of the Web browser:

- *JavaScript controls Applet*:

  Variables and methods which are defined as public in the applet can be used in JavaScript. Methods can supply return values.

- *Applet controls JavaScript*:
  For the contact to JavaScript objects a Java Virtual Machine must contain a Java Package which is not included in the standard package of Java. This package is netscape.javascript. It has been included in Netscape Navigator since version 3.0, and in Internet Explorer since version 4.0.

If an access to distant process data is implemented in the Java applet via e.g. TCP Sockets or RMI, then the process data can be made available via appropriate methods in JavaScript by LiveConnect.

interface, e.g. to an OPC server, a driver for a fieldbus system or an embedded controller. The W2<process> proxy is equal to a distributed application following the client/server architecture. The W2<process> proxy server is an independent application which is implemented on the process data server.

The functions of the W2<process> proxy client are performed by a Java applet. The applet is stored in the memory of the process data server. When requested by a process data client (PD client), the applet program code is downloaded by the process data server via its Web server and processed in the Web browser of the process data client.

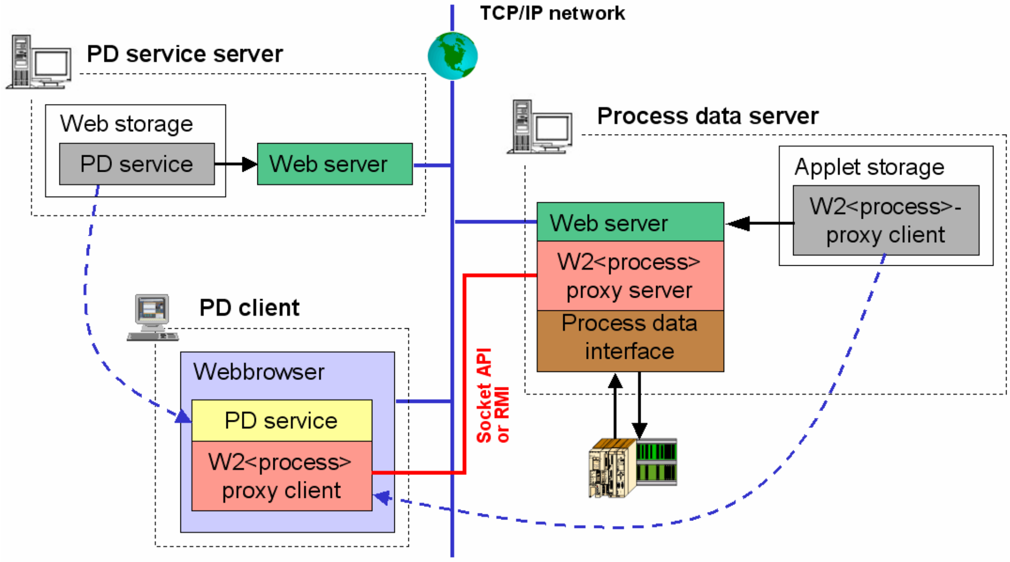

Figure 2: Distributed structure for the Web-based operation of automation devices by the LWA concept

II. Lean Web Automation as basic concept

Developed in the Process Informatics Lab of the University of Applied Sciences Duesseldorf – the concept of Lean Web Automation (LWA) proposes, on the basis of LiveConnect, a solution for distributed automation using standardized Internet technologies [3].

The automation devices attached to a network (Intranet/Internet) make available only their process data (sensors/actuators). The effective control and automation functions are loaded and used from resources which are available in the net.

A. Process data in the net

Principal item of the LWA is the fast, reliable and safe supply of the sensor and actuator data of automation equipment in the Intranet/Internet. The process data communication is realized by a Java-based application model *W2<process> proxy* using Live-Connect. This model provides an interface to the process data of an external automation equipment similar to a proxy of the respective automation function in the process area of a Web client (browser) (see fig. 2).

The substitute symbol <process> stands for the practical realization of the appropriate process data

The PD client (any computer in the net) takes over the function of the master for the automation device.

The W2<process> proxy client creates a permanent dynamic data link to the W2<process> proxy server in the process data server and – over the connection to the process data interface – to the actual process data of the automation device. The data connection between the W2<process> proxy client and server is achieved with the help of the Java technology available for the realization of dynamic communication in distributed applications, Socket API or RMI (RMI = Remote Method Invocation). The process data exchange is realized by an own OPC-like telegram protocol.

B. Distribution of automation functions

The processing of process data is made by process data services (PD service) which are downloaded from a PD service server via the Web to the PD client and are executed there.

In LWA process data service describes the program code necessary for the execution of the automation functions. A PD service can contain operation and user interface functions as well as functions for the processing of process data. So it is possible to visualize e.g. current trends of analog process data by means of a suitable PD service (e.g. a graphical XY presentation service) in the PD client.

The program code of a process data service is e.g. a makro created in an ECMA-Script-compliant programming language. In addition, a Java applet or other Web objects (ActiveX object, Flash object, …) can be used as user specific PD services.

The communication between the W2<process> proxy client and the PD service takes place on request as well as event-controlled over the LiveConnect mechanism.

III. Remote Control via Web Access Kit

Web Access Kit (WAK) for OPC is the software tool of the LWA concept which uses OPC as an interface to the process [4].

The WAK tool is already available for several embedded Web controllers: e.g. for the Ethernet based I/O module "Web-IO" of Wiesemann&Theis (Wuppertal, Germany) and for the "BC660" of Beck IPC (Wetzlar, Germany), a very compact controller with embedded Web server and Ethernet interface.

The current version WAK V1.3 consists of the following components:

- WAK data server (= W2OPC proxy server),
- WAK proxy applet (= W2OPC proxy client),
- server-assistant,
- client-assistant.

Fig. 3 shows the component structure of the Web Access Kit and illustrates its function mode. See [5] to get access to an HTML training system for WAK (selfwak) and [6] to download the trial version of WAK V1.3 Professional.

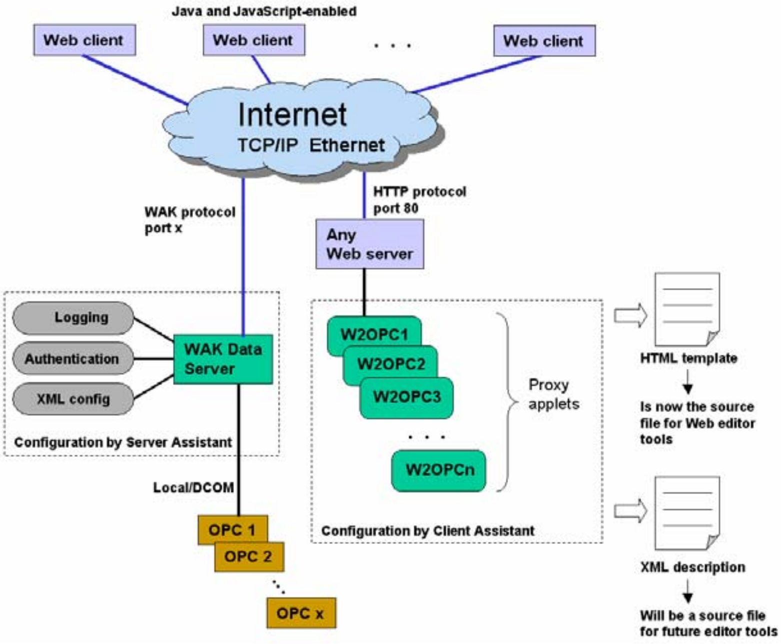

Figure 3: Component structure of the Web Access Kit for OPC

A. *WAK data server*

The *WAK data server* is a Java application and enables the access to process data of arbitrary OPC servers. Via a configurable TCP port, the WAK server creates a permanent data link to the WAK proxy applets which are loaded in an HTML page. A server assistant guarantees an adaptable and flexible configuration available as XML file.

The data server can be described by the following characteristics:

- multithreading functionality,
- configurable refresh rate for process data (up to max. OPC refresh rate),
- configurable TCP port,
- precisely adjustable user rights,
- extensive logging mechanism (user, access, error),
- safety procedures at connection abort.

B. *WAK proxy applet*

A WAK proxy applet delivers the process data by means of a Java/JavaScript API (API - Application Programmers Interface) with approximately 20 methods to other Web objects in the HTML page of the Web client. Standard Web tools enable the user to create Web-based automation functions (control, operation, visualization ...) while applying all abilities of Web objects. Additionally, the proxy applet possesses a graphical interface for the transferred process data which can be switched on and off via an applet parameter for diagnostic purposes.

The number of proxy applets simultaneously connected to the WAK data server is not limited. As many as desired WAK data servers can be operated on the process data server which access to other OPC servers in each case.

C. *Example for remote control with W2OPC proxy*

A W2OPC proxy applet is embedded into a Web page with an HTML editor. The necessary OPC variables (OPC items) of a certain OPC server as well as other configuration settings are indicated in the applet parameter set. The parameter set can either be created manually by the user or be generated automatically with the help of the WAK client-assistant.

The API of the W2OPC proxy applet provides synchronous and asynchronous methods for read/write accesses to one or more process variables.

The process data (Items) are addressed via their *ClientHandles*. With n items in an applet, ClientHandles from 0 to n-1 are assigned to the individual items. During value changes of a process date**,** an event-controlled notification of a PD service by a W2OPC proxy applet is realized by triggering of the special method `<appletID>_onDataChange`.

The code example in fig. 4 explains the application of the WAK proxy applet for the realization of a control algorithm as PD service:

After registration of an event receiver by means of the function `proxy1_onInit` the two process data `input1` and `output1` are subscribed for changes of their values (subscriber/publisher principle). With each change of the values, the W2OPC proxy applet calls the function `proxy1_on DataChanged` via LiveConnect. Inside of this function the control function `control` is called which changes the `output1` according to the control algorithm.

```
...
<script language="JavaScript" type="text/javascript">
function proxy1_onInit(){
document.applets["W2OPCProxy1"].addW2OPCPrEventListener(
"dataChanged","proxy1_onDataChanged");
  }
function proxy1_onDataChanged(dchEvent){
  var numItems = dchEvent.getNumItems();
  for (var i = 0; i < numItems; i++) {
   var  index = dchEvent.getClientHandle(i);
    var strValue = dchEvent.getValue(index);
   if (index ==0){
     control(strValue)
   }
 }
}
function control(input_value){
//Control Algorithm
...
}
</script>
...
<applet code="wak.w2opcpr.W2OPCProxy.class"
codebase="http://195.37.240.99/wak13/applets/"
archive="w2opcproxy.jar" name="W2OPCProxy1" height="0"
width="0" mayscript>
      <param name="Copyright" value="2003 m+iTEC">
      <param name="AppletID" value="proxy1">
      <param name = Port value = "6565">
      <param name="OPCVar" value="input1,output1">
    </applet>
...
```

Figure 4: Web site (part of code) of simple remote control

This method enables the simple generation and application of event-based sequence programs as PD service. The control is performed by the Web client PC. In the appropriate station/device a controller is not needed. The component structure is shown in fig. 5 (compare to fig. 1).

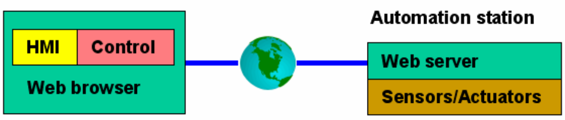

Figure 5: Web-based Remote Control Structure

D. *Security and reliability*

For a secure and reliable application, additional methods on the level of the process data communication are embedded in the LWA concept beside the well-known Web-based safety methods:

- *User administration:* For the access to process data a flexible and fine adjustable user administration with group rights is integrated.
- *Access and error logging:* All accesses to the process data are logged – related to user, process date and Web client.
- *Error management in case of client crash or connection abort:* A connection interruption is recognized in the process data server by the W2OPC proxy and can be evaluated as a function of the application (e.g. set all process data outputs to zero).

Furthermore, redundant communication structures can easily be developed with WAK. In these structures, several TCP communication channels can be used for the transmission of the same process data, see [6] for an example.

E. *Time performance*

Naturally, time performance is an important characteristic of a controller. The time performance of the described Web-based controller is mainly determined by the following aspects:

- Transmission time inside the network (Internet/Intranet),
- Processing time of the PD service

First studies concerning the time performance of the process data transmission between OPC server and Web client including the data transfer to an event receiver by means of WAK resulted in values specified in table 1.

TABLE I. TRANSMISSION TIME FOR PROCESS DATA VIA THE INTERNET

	Transmission time of 10 process data, these values are changed at the same time [ms]
Intranet	20 … 50
Internet	800 … 1500

The processing time of the control program needed by the PD service in the Web client depends on how it is realized. In each case the communication between the LiveConnect interface of the W2OPC proxy applet and the actual control program (e.g. in JavaScript or also by means of a Java applet) requires a defined time. Depending on the parameter transfer and browser type, a simultaneous change of 10 process data needs a time range of 1 to 10 ms.

In summary, simple sequence programs with a JavaScript source code extent of a few hundred lines source code and a restricted amount of process data can be operated via the Internet in a time slot pattern of approximately 200 ms [7].

IV. APPLICATIONS

A. *Web Programmable Logic Controller (WebPLC)*

In the context of the development of a tele-practical course for PLC training according the IEC 61131 standard in the Duesseldorfer Telelabor a didactical WebPLC as PD service was implemented [8][9].

The WebPLC contains a Java applet with PLC editor (IL language) and an integrated run time environment. Communication with the distant station is done event-based by WAK server and W2OPC proxy. Trainees can use this WebPLC to program and control assembly procedures for a model car via the Internet.

B. *JavaScript Controller*

For demonstrating a pure JavaScript controller, a digitally positionable lifting station with four positions was set up. The lifting station is controlled by the Web client and a light position detector. See [9] to test the controller (fig. 6). The users can also develop their own JavaScript programs and control the lifting station from his PC via the Internet.

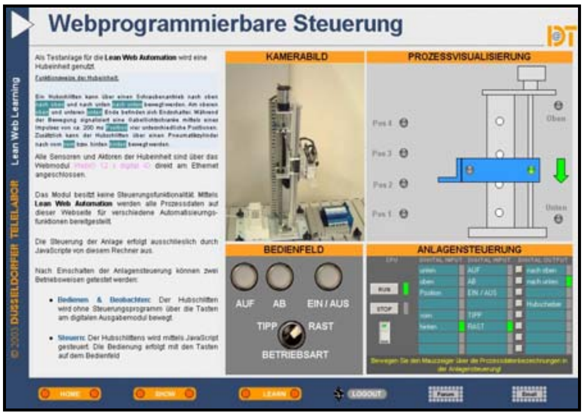

Figure 6: Remote Control of a lifting station via JavaScript

C. *Web-based User Interfaces*

Current emphasis in applying the Web Access Kit is put on graphical user interfaces for automated stations, machines and devices e.g. in the context of Web-based teleservice solutions as well as real-time visualization in the Web [7]. Several examples can be tested under [10].

V. SUMMARY AND PREVIEW

With its flexiblity and transparency, Lean Web Automation will meet future requirements for individuality and openness of automation systems and simultaneous cost reduction in operation and maintenance. A strong focus is on Web-based remote visualization and operating.

Remote control of didactical plants have already been implemented successfully. Further developments of the LWA concept are planned in the following fields:

- Increased reliability of the process data access even with short connecting interruption and time delays.
- Investigation of usage and security problems with consideration of the passing of Intranet/Internet borders, coding of process data and resumption of aborted automation functions with automatic restart.
- Development of different PD services for the process data processing and preprocessing in the Web client. Testing of different visualization technologies for the LWA concept (e.g. with SVG graphics, flash animations and dynamic 3D VRML visualizations).
- Investigation of different service structures and management models for the use of automation functions as PD services in the Web (e.g. "Rent a PC service").

In a new project "Ego secundus" (project term 2005 - 2007) the described concept will be used for the Web-based controlling of a mobile robot platform.

REFERENCES

[1] ZVEI, "Ethernet in der Automation", Studie, 2003

[2] C. Kühnel, S. Mintert; "JavaScript", http://www.javascript-workshop.de/buch/12.html, JavaScript-Workshop online.

[3] R. Langmann, "A New Approach for Automation of Distributed Systems", *Automation Panorama*, 2004.08, No. 4, pp. 23 - 39.

[4] R. Langmann, "Von JavaScript zum technischen Prozess," *Elektronik*, 17/2004, pp. 42 – 46.

[5] Homepage of m+iTEC, http://www.lean-web-automation.com

[6] Web-based Learning System: Web Access Kit for OPC, http://www.lean-web-automation.com/selfwak/wak.htm, m+iTEC, 2005.

[7] R. Langmann, " Schnelle Prozessvisualisierung im Web", *A&D Newsletter*, März 2005, pp. 71 – 73.

[8] R. Langmann et al, "The German Remote Lab Network CONTROL-NET", *Proc. of the 21th World Conference on Open Learning and Distance Education* (CD-ROM), Hongkong, 18 - 21.02.2004.

[9] Homepage of the Telelab Duesseldorf, http://www.telelabor.de

[10] Examples for Web-based Graphical User Interfaces, http://www.lean-web-automation.com/

AUTHOR

Reinhard Langmann is Professor at the University of Applied Sciences Duesseldorf and the head of the Process Informatics Lab, 40474 Duesseldorf, Germany (e-mail: R.Langmann@ fh-duesseldorf.de).

Manuscript received May 31, 2005. This work was supported in part by Ministry for Science and Research of the country North Rhine-Westphalia under the grant "Telepraktikums-Kit".