Efficient Web-based Monitoring and Control System

Ahmed M. Mohamed
Electrical Engineering Dept.,
Aswan Faculty of Engineering,
Aswan, Egypt
ahmed@engr.uconn.edu

Hosny A. Abbas
Qena Paper Company
Qena, Egypt
hosnyabbas@yahoo.com

***Abstract*— Computer-based supervisory control and data acquisition (SCADA) systems have evolved over the past four decades, from standalone, compartmentalized operations into networked architectures that communicate across large distances. There is an emerging trend comprising SCADA and conventional IT units toward consolidating some overlapping activities. This trend is motivated by cost savings achieved by consolidating disparate platforms, networks, software, and maintenance tools. For reasons of efficiency, maintenance, economics, data acquisition, control platforms have migrated from isolated in-plant networks using proprietary hardware and software to PC-based systems using standard software, network protocols, and the Internet. In this paper, we present a new approach for web based SCADA systems that adapt to the behavior of the target application. In addition, we take into account the real time constraints that imposed by the nature of the problem. We show that our approach is more efficient than other approaches in terms of consuming as little as possible of the available resources (computational power and network bandwidth).**

Keywords – Automation; SCADA; OPC DA; Web Services.

I. INTRODUCTION

As the technical capabilities of computers, operating systems, and networks improved, organizational management pushed for increased knowledge of the real-time status of remote plant operations. These capabilities are known collectively as Supervisory Control and Data Acquisition or SCADA. As the name indicates, it is not a full control system, but rather focuses on the supervisory level. SCADA systems are vital components of most nations' critical infrastructures. They control pipelines, water and transportation systems, utilities, refineries, chemical plants, and a wide variety of manufacturing operations. Automation technology produces large amounts of data. This typically represents physical variables such as temperature, current, pressure or other production data such as number of units, error information and so on. The data comes from equally diverse sources: controllers, level measuring devices, weighing machines, scanners and similar. These data end up being crunched or otherwise retained by all sorts of software applications. It may need to be placed straight into visualization software for HMI (Human Machine Interface).

One of the most familiar automation protocols now is OPC protocol. OPC stands for OLE (Object linking and Embedding) for Process Control or Open Process Control [2,9]. It provides a mechanism to provide data from a data source and communicate the data to any client application in a standard way. A vendor can now develop a reusable, highly optimized server to communicate to the data source, and maintain the mechanism to access data from the data source/device efficiently**.** Providing the server with an OPC interface allows any client to access their devices. The utility of OPC has now reached the point where automation without OPC is unthinkable. This interface is supported by almost all SCADA, visualization, and process control systems [3]. OPC delivers connectivity and interoperability benefits to measurement and automation systems in the same way that standard printer drivers deliver connectivity and interoperability benefits to word processing. Both OPC server and OPC clients are COM objects. It makes no difference who developed these objects, when they were developed, or in what programming language they were written. Not only the services of the OPC servers on the same computer are available but also those of all servers which can be accessed over the network. Our objective is to integrate the existing OPC server data access (OPC DA) with the Internet to achieve an efficient web-based SCADA system. In our approach we consider many design parameters such as how frequent the target data change, the consumption of the available resources and the real time constraints that imposed by the nature of the application. This paper is organized as follows; in Section II we give a brief background for some of the solutions that have been developed for this problem. In Section III, we present the motivation behind our research. We introduce our approach in Section IV. In Section V, we present evaluation for our approach.

II. BACKGROUND

There are many solutions that have been proposed to solve this problem. These solutions can be classified into four categories.

A. DCOM

Most of the previous work used DCOM for LANs communication with OPC DA server for example, Xiaofeng Lee et al. [18] used DCOM communication between an OPC DA client and OPC DA server then they transfer the OPC DA client data to XML format to be able to access these data through Internet with an XML-DA client which

 ISBN: 978-1-61208-134-2

communicate to an XML server (Web server) to get data. Truong Chau et al. [16] used DCOM to enable the C# server script to access OPC DA through LAN and because the OPC DA client is a .NET client they used an OPC .NET wrapper to make the transformation from .NET to COM and COM to .NET. Zhang Lieping et al. [19] uses the OPC DA Toolbox which is integrated in MATLAB 7 and above editions, this Toolbox enables MATLAB applications to communicate with OPC DA servers through DCOM then the user can simply and conveniently realize the operation to the OPC objects. Unfortunately, using DCOM through Internet is avoided for many reasons such as, DCOM is windows dependent platform, difficult to configure, has very long and non-configurable timeouts, and cannot be used for Internet communication. That is why DCOM is best suitable for LANs where there are less number of nodes and small delay times.

B. XML

XML is a platform-independent which is an important feature to achieve interoperability between different applications that is running on different platforms. The other advantage of OPC XML-DA is the simple administration as it is based on SOAP and XML. Xiaofeng Lee et al. [18] suggested designing an information integration system which will adopt OPC DA to OPC XML-DA; the design includes three layers structure, data source layer, data transfer layer and client layer. His conclusion was that because of adopting industry standard OPC interface and web service transfer interface, the remote monitoring system based on OPC XML-DA technology makes it convenient to update and expand system. But if we analyze this approach we find that there is an overhead in layer2 because of the COM-XML transformer, also the authors didn't expose to the problem of client data update is there a data polling mechanism that enable the client to get the new data. Similar work used XML and/or OPC XML-DA techniques such as [3, 15, 17]. Due to possible performance limitations, OPC XML-DA is unlikely to be used for real time applications, although it is commonly used as a bridge between the enterprise and control network.

C. Webservice

Webservices as defined by the W3C as "a software system designed to support interoperable machine-to-machine interaction over a network. Nunzio M. Torrisi et al. [8] proposed what they called CyberOPC which is a communication system that anticipates the use of a gateway station called CyberOPC, which will process messages sent to the OPC towards the public network and vice versa. Their proposed communication system is targeted to best effort network with minimum bandwidth reserved for periodic traffic. As they claimed that the necessity to satisfy time-critical and security requirements for remote control has stimulated the study of a new protocol for process control. And to obtain maximum interoperability with existing factory floor technologies, they built their communication project over the OPC technology. This approach doesn't solve the problem of periodic data requests from the remote client to update its old data with new data, and when the remote client makes a request, the CyberOPC will not check if there is a change in these data from the last sent data or not so if the control process has a high frequency data change, the remote client has to increase its periodic data requests which will affect the server efficiency and network bandwidth.

Duo Li et al. [3] suggested an approach to implement the function of real time monitoring which provide periodically updated data to operators, the target data which saved in a database is marked and mapped to an HTML file, say file2, in a web server, the database server automatically refreshes file2 with the latest data whenever the target data in the server is updated. To show target data in an HTML file a Java applet is developed that connects to file2 to get the latest data periodically and display them in the required format, another HTML file ,say file1,in which the applet is included, is developed to establish a basic human-machine interface (HMI) and complete some initiations.

D. AJAX

Actually AJAX (Asynchronous JavaScript and XML) [5] started a new line in web based SCADA systems because it enables us to create more interactive web pages which are our way to replace traditional desktop SCADA application. AJAX can be considered as the future of the Internet because of its ability to simulate desktop applications by the new features it offers like asynchronous client-server calls and partial-page updates. Truong Chau et al. [16] used AJAX in their approach to get the OPC DA data to the client web page with high user interactivity. But because of the problems of the HTTP protocol as a stateless protocol they had to use a simple data polling mechanism by using an AJAX Timer to poll for new OPC DA data set from the OPC DA server through the web server. When using an AJAX style of programming the old, classic programming approach must be given up. There is no form submit any more that posts all the client state to the server and requests for a complete new page description using HTML. Instead of loading several pages until the functionality is done only one page is loaded and will stay in the browser until the end of functionality.

III. MOTIVATION

None of the mentioned approaches considered the feasibility of the realization of their work. If we look closely to [3, 13, 15, 17] we will find out that they use XML and/or OPC XML-DA technologies which have disadvantages such as:

- Not suitable for transferring large data volumes
- XML technology is generally slower than COM
- The interaction parameters are coded using XML, which leads to an overhead.
- An OPC XML-DA Service is stateless.

 ISBN: 978-1-61208-134-2

Also in [12], to allow each node (client or server) to start sending data to the other node independently at any time (bidirectional communication) the authors have to maintain the connection in a permanent open state by making the server continuously sending data (whether there is a data change or not) to the client which affects the performance of the system specially with large number of clients and low network bandwidth, and this may lead to server crash.

In [14], the authors will need to find a way to periodically update file2 which may be a loop or any other way, also the java applet which embedded in the HTML file1 will need to periodically use a Timer to get the new data from file2 and then they will face the problem of synchronization between the applet Timer interval and the process of updating file2 with new data and this will be difficult, also they ignored the heavy network load and the need for large server memory to handle the large number of requests and the heavy load on the server CPU. It is very difficult (nearly impossible) to guarantee a real time behavior with such limitations. The only solution is to spend money to increase the network bandwidth and the server CPU speed and memory capacity (The hardware resources). In our work we try to find alternative, much cheaper solutions to these problems. Also in [8], the designed CyberOPC doesn't solve the problem of periodic data requests from the remote client to update its old data with new data, and when the remote client makes a request, the CyberOPC will not check if there is a change in these data from the last sent data or not so if the control process has a high frequency data change, the remote client has to increase its periodic data requests which will affect the server efficiency and network bandwidth.

Finally, in [16], the authors did not take any attention to the efficiency and performance of their approach, the Timer they used will get the data without any care if there is a significant data changed or not, also how they could specify the Timer interval, there are two cases for that, first, when the interval is small i.e. 1 second (high frequency data polling) by this way the server and network will suffer, second if the interval is large they will loss some data change events that maybe very important and the system will not be considered as a Real-Time monitoring system. So this approach still needs some modifications which enable us to get optimal efficiency and performance, for this reason we choose this approach to be our reference approach. We believe that AJAX can be the way to make web applications compete with desktop applications by the interesting features it offers [5].

IV. THE PROPOSED APPROACH

The main challenge will be to design and implement a new approach for providing a direct web access to controllers using OPC DA server that is aware of the consumption of the available resources and try to fulfill the real time constraints as well. We have two options to achieve this target, first using DCOM, second using a web server and modern IT technologies. If we used the first option, we will face the DCOM problems (see section II). Therefore, we will use the second option, which is web server and modern IT technologies. However, with this option we are going to face many design and implementation challenges such as, HTTP limitations, responsiveness, real time constraints and the efficiency of the approach. In a classic client/server application, there is a static communication link between client and server but with web applications, this static communication does not exist thus the need for a persistent communication mechanism by which a client can communicate with the server independent of any specific action taken by the user. We want that communication take place even if the user is not clicking or using any of the controls in the user interface.

A. System Overview

Our target system consists of a web server, an OPC server and a Programmable Logic Controller (PLC) units which will be connected in the same plant network, which may be industrial Ethernet network or Fieldbus network (Profibus, MPI... etc), in our case it will be an Ethernet LAN because it is now a standard and familiar protocol in industry. This LAN connected to internet through an ADSL router which has a local (virtual) IP number and an international (Real) IP number. A Client (browser) can connect to the Webservice residing in the web server though internet using HTTP protocol, then the Webservice will connect to the OPC server in the same LAN using DCOM to get or set the new Data as shown in Figure 1. To achieve our goals we will use a modular architecture in the design of our web based SCADA system. As shown in Figure 2. Our system consists of three main modules, which are:

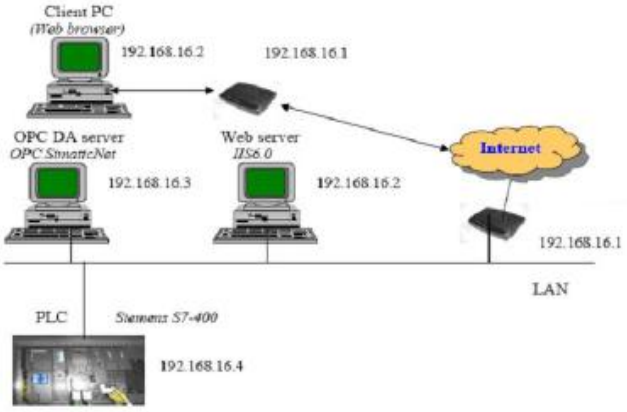

Figure 1: Proposed System Overview

1. The Client, which is a web browser that will be used to run and display the designed SCADA web page and will send the required requests to update the web page with the new data. The requests will be sent asynchronously (in background) by the AJAX engine and the web page will still be responsive. When the request's response comes, the AJAX engine will forward the returned data to the web page then the web

page will send another request and so on. A new request will be sent only if the response to its predecessor request comes.

2. The web server which will wait for the HTTP client's requests and forward these requests to the Webservice which in turn will run the proper web method (WM1 if the request asks for new data and WM2 if the request contains set-points to the OPC DA server). The WM1 will wait for a data change in the OPC DA server cache and then return the new data to the client. WM2 will write the coming client's set-points to the OPC DA server and return to the client without waiting. Also, it includes an OPC DA Wrapper**,** as we mentioned that OPC DA protocol is based on Microsoft COM technology. However, the Webservice, which we use to access this protocol, is implemented in modern .NET technology. So, we will have to find a way to access a legacy technology from a modern technology, the OPC FOUNDATION solved this problem by creating wrappers to enable .NET clients to access COM servers.
3. The OPC DA server, which is the interface to the physical control process (PLC), the OPC DA server has two layers, one to provide the standard COM interface to its clients and another one, which is a vendor specific driver to communicate with the PLC.

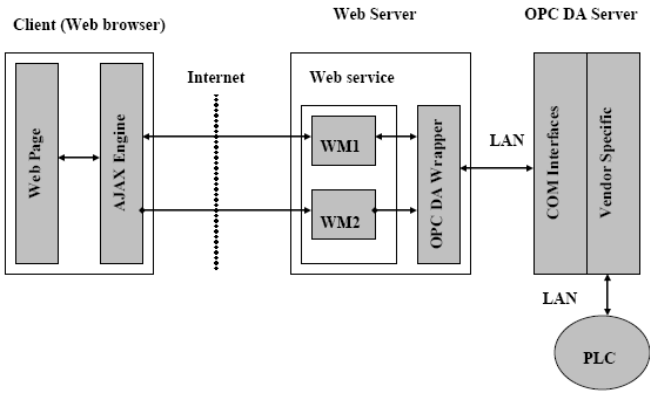

Figure 2: Architecture of the Proposed System

B. *Proposed Approach Design*

From the discussion of previous work and because of the Internet infrastructure, we conclude that data polling is the only solution, which is robust and viable; our main challenge will be implementing a data polling that is effective and wastes as little as possible of the available resources**.** A traditional data poll such as the one used by Truong Chau et al. [16] will query the server, get an answer, and then wait until the next query. The client side has to choose the rate of polling the server for a new data (i.e., a request every 5 seconds) without taking into account the frequency of data change (fast or slow). The waiting period until the next poll is a dead time during which neither the client nor the server can communicate with each other.

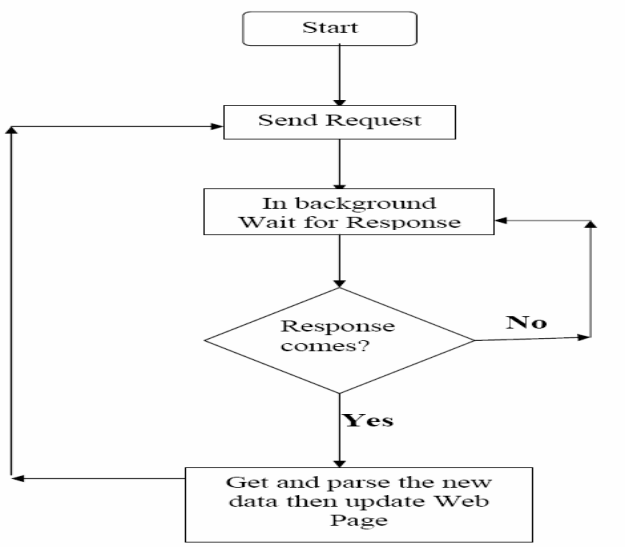

Figure 3: The client side

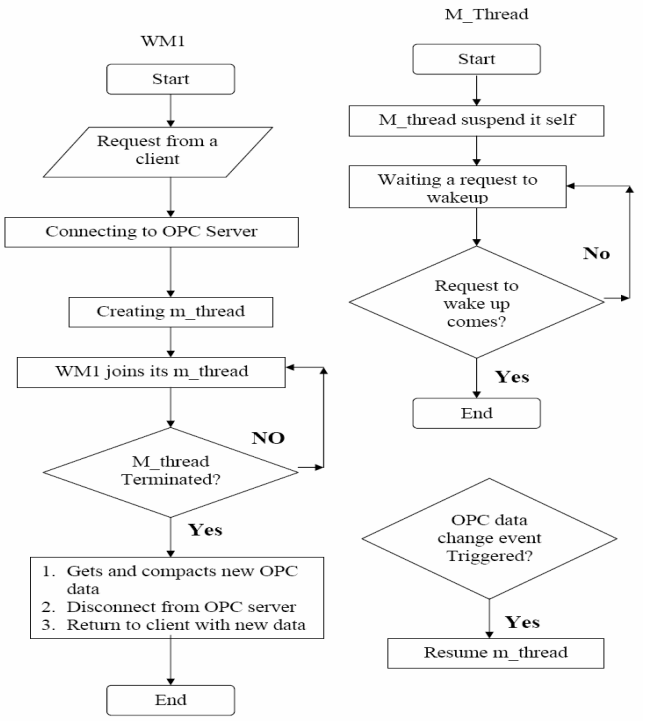

Figure 4: The server side Webservices

In our approach we do not choose a constant data query period instead we adapt to the frequency of data change of the application. This can be achieved by converting the dead time into a wait time created by the web server. The client sends a request and waits for the potential of data change. The server responds to the client only when there is a data change. Otherwise the request is put in a waiting queue at the server. The design flow charts of our approach is shown in Figure 3 (for client side) and Figure 4 (for server side). For complete description of the design and the code of each module, please check [1].

 ISBN: 978-1-61208-134-2

V. EVALUATIONS

Any control process has a number of measurable physical quantities, which are measured by different sensors, and then the raw signals transferred to the PLC to be converted by the A/D converters to digital form to be manipulated by the PLC processing unit. The OPC DA server updates its cache by the new values available in the PLC according to a specified update rate. So a change in the control process quantities leads to a change in the OPC DA server cache. Hence, to simulate a typical control process we will create a software simulation application, which connects to the real OPC DA server, and continuously change its cached data. The simulation application created using VB.NET 2008 and designed to offer many options such as the data change rule (Fixed or Random), the number of data changes (Limited or Unlimited), and the range for Timer interval in case of random data change can be specified, and the changed data range can be specified too. In addition, there is a facility to start and stop data changing at any time to check the behaviors of our proposed approach and other approaches with/without data change. The exact time of each data change can be stored in a database for analysis purposes. In all figures presented in this section we changed the OPC data according to the uniform distribution with parameters (1, 3). Now to show the extent to which we achieved our goals we will compare our approach results to Truong Chau et al. [16] approach results, we will use windows Task Manager to monitor the web server's performance measures and network bandwidth utilization. Applying Truong approach and using simulated OPC data. We choose the timer interval (request rate) for Truong Chau et al. [16] approach to be 1 second as they suggested in their paper. When we run the experiment, we got the CPU usage as shown in Figure 5.

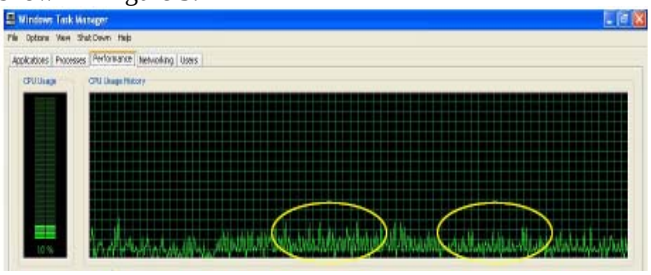

Fig. 5 Truong Approach (CPU load) with Timer interval = 1 sec

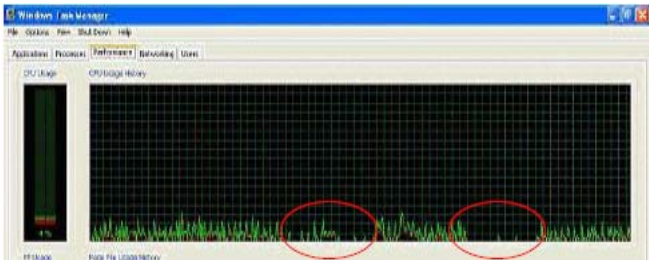

Figure 6: Our approach test results (CPU Load).

As we see in Figure 5, even if there is no data change (check the circled areas) the server still working approximately with the same CPU usage rate since there is a request coming from the client every one second. However, in our approach the CPU usage is approximately zero when there is no data change as shown in Figure 6 and that because the client does not send a new request until it receives a response to the current request and the server does not send the response until there is a data change. This means that our approach uses the available resources (CPU power and network bandwidth) only when there is a need to do so (significant data change). Thus we use less computational power than previous approaches which proves that our approach is more efficient. Also the data transferred through network with Truong approach is shown in Figure 7, as shown his approach transfers data continuously through network (suppose his timer interval is 1 second) regardless the data changed or not. However, with our approach the data transferred only if there is a data change as indicated by the circled areas in Figure 8.

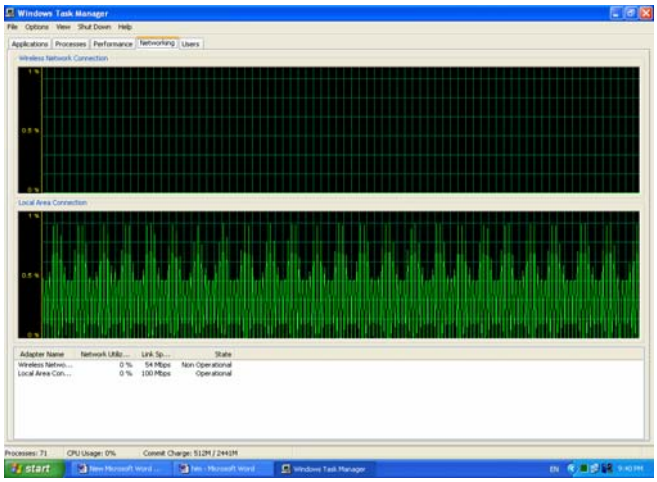

Figure 7 Truong Approach Network utilization with T =1 sec

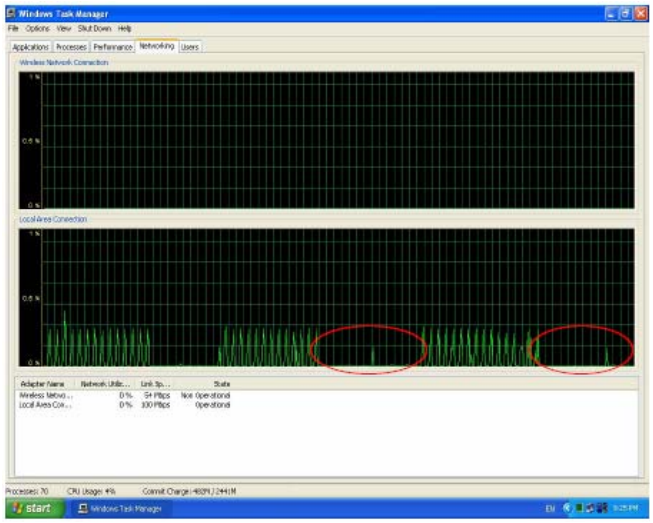

Figure 8: Our approach Network utilization

 ISBN: 978-1-61208-134-2

Another experiment we performed using our simulation was to calculate the number of request to the web server for each change in the target application data. We change the rate of data change to be no changes at all, 1 change / 4 seconds, 1 change / 2 seconds, 1 change/second. Applying Truong Chau et al. and our approach we got the following numerical data in Table 1.

TABLE I. NUMBER OF REQUESTS TO THE WEB SERVER

Number of Actual data changes / Minute	Number of client requests/Minute	
	Truong Chau. [16] with T=1 Second	***Our approach***
0	**60**	**0**
15	**60**	**15**
30	**60**	**30**
60	**60**	**60**

As we can see, our approach can adapt to the change behavior of the application. The server will send data (use the network) to the client only when there is a data change. While in previous approaches the server keeps sending data to the client even if there is not a data change. For a complete set of experiments, please check [1].

CONCLUSION

As IT technologies progress, web applications will replace desktop applications in most of computer applications fields especially industrial applications like SCADA systems which are now very important in process control. AJAX finally enables us to make an efficient web based SCADA systems because of the features it offers like asynchronous client-server communication and partial page update. In addition, Webservices can be used as server side scripting. Webservices provide a language-neutral, environment-neutral programming model that accelerates application integration inside and outside the enterprise. Application integration through Webservices yields flexible loosely coupled business systems. Using AJAX in a proper way can give us a very good performance in the server and network. In this paper we used AJAX to implement a persistent communication between client and server. In addition, the threading mechanism used in the Webservice was the key concept to achieve this persistent web communication. Because of the Internet infrastructure, we concluded that data polling is the only viable solution to get new data. We showed how our approach for web based monitoring and control system adapts to the behavior of the target application by responding to the client only when there is a data change (the frequency of requests matches the frequency of the application data change). Also, that our approach is more efficient than previous approaches. In addition, we take into account the real time constraints that imposed by the nature of the problem. By using as little as possible of the available resources (computational power + network bandwidth), we improve the responsiveness of the application.

References

[1] Abbas, Hosny and Mohamed, Ahmed "Efficient Web Based SCADA System" Master Thesis, http:// www. engr. uconn. edu /~ahmed/Thesis_Hosny.pdf, Mar. 2011.

[2] Byres research, OPC security WP#1, July 17, 2010.

[3] Duo Li and Yoshoizumi Serizawa, "*Concept Design for a Web-based SCADA system*", Proceedings of Transmission and Distribution Conference and Exhibition 2002: Asia Pacific, IEEE/PES, Vol. 1, pp. 32 – 36, 2002, JAPAN.

[4] http://www.controlglobal.com/articles/2004/229.html, Aug. 18, 2010.

[5] http://www.helium.com/items/580436-advantages-and-disadvantages-of-ajax, Sep 6, 2010.

[6] http://www.w3.org/TR/WD-script-970314, Mar 14, 2010.

[7] Mike Clayton, Philippe Gras, and Juan Oses "A SCADA-WEB INTERCONNECTION WITH TCP IN JAVA", URL: *http://ess.web.cern.ch/ESS/GIFProject/PVSSJava/pvssweb.0.8.pdf*, Nov, 20, 2010

[8] Nunzio Torrisi and João Oliveira," Remote control of CNC machines using the CyberOPC communication system over public networks", The International Journal of Advanced Manufacturing Technology, Vol. 39, pp. 570-577, 2007

[9] OPC Foundation, "OPC DA 3.0 Specification [DB/OL]", Mar.4, 2010

[10] OPC Foundation, "*OPC XML-DA Specification Version 1.0*", Released July 12, 2003.

[11] Shamdutt Kamble, Venkateswara Rao, and Shailendra Jain, "*OPC Connectivity to Remote Monitoring & Control*", White paper, Wipro Technologies company, www.wipro.com.

[12] Shekhar Kelapure, Sastry Akella, and Gopala Rao, "*Application of Web Services In SCADA Systems*", The International Journal of Emerging Electric Power Systems Vol. 6, No 1, pp. 1-15, 2006.

[13] Shaung-Hua. Yang and Lili Yang, "*Guidance on Design of Internet-based Process Control Systems*", ACTA AUTOMATICA SINICA, Vol. 31 NO.1, pp. 56-63, 2005.

[14] Thomas Bauer, Roland Heymann, and Heinz Christoph, "System and method for transmitting OPC data via Internet using an asynchronous data connection", US patent number: US 7,302,485 B2, 2007.

[15] Thomas Dreyer, David Leal, Andrea Schröder, and Michael Schwan, " *ScadaOnWeb – Web Based Supervisory Control and Data Acquisition*", Proceedings of 2nd International Semantic Web Conference, pp. 788-801, 2003, FL,USA.

[16] Truong Chau and Nguyen Khai, "*WEB-BASED DATA MONITORING AND SUPERVISORY CONTROL*", Proceedings of the International Symposium on Electrical & Electronics Engineering. pp. 7-13, 2007, Vietnam.

[17] Vu Van Tan, Dae-Seung Yoo, and Myeong-Jae Yi, "*Design and Implementation of Web Service by Using OPC XML-DA and OPC Complex Data for Automation and Control Systems*", Proceedings of the Sixth IEEE International Conference on Computer and Information Technology pp.263-265, 2006, Korea.

[18] Xiaofeng Lee and Jianfeng Hu,"*Design and Research of Remote Monitoring System based on OPC XML-DA*", Proceedings of 2008 International Pre-Olympic Congress on computer science, V(1), pp. 147-151, 2008, China

[19] Zhang Lieping, Zeng Aiqun, and Zhang Yunsheng, "*On Remote Real-time Communication between MATLAB and PLC Based on OPC Technology*" Proceedings of the 26th Chinese Control Conference, pp. 545-548, 2007, China

 ISBN: 978-1-61208-134-2