http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 12 (19): 4823-4830, 2013
ISSN 1812-5638 / DOI: 10.3923/itj.2013.4823.4830

Using OPC Standard for Control System Integration

[1]Yang Maying, [2]Jin Xiaoming, [3]Zhang Shuji, [3]Xu Yi and [3]Liu Bingjie
[1]College of Information Engineering, Zhejiang University of Technology, 310014, Hangzhou, China
[2]Institute of Cyber-systems and Control, Zhejiang University, 310027, Hangzhou, China
[3]Zhejiang Supcon Software Co., Ltd., 310053, Hangzhou, China

Abstract: This study describes an OPC (OLE for Process Control) based framework for integrating distributed control systems. The proposed framework combines OPC Client/Server technologies to integrate data and services. The OPC client interface for JX 300X DCS and S7-300 PLC is developed, the design and implementation of the integration technology are presented in this study. Experiments on performance of the client interface and OPC-based remote control are also conducted and presented.

Key words: Control system integration, OLE for process control, distributed control systems

INTRODUCTION

There are a wide range of process control applications and control devices, including Distributed Control Systems (DCS), SCADAs, PLCs, etc. They have different originality but can meet the plant control and supervising demand currently However, these devices often have different data access interfaces which prohibits direct intercommunication. For the time being, effective performance monitoring and remote control of these devices have been a primary interest for many commercial and industrial businesses. Due to the advances in the Internet, the ability to acquire information and even to control devices over the Internet is becoming desirable to the general public as well as professionals (Chang *et al*., 2006). Manufacturing enterprises are looking at advanced distributed object technologies as a means of integrating the islands of automation that currently exist within their worldwide operations. Distributed object technologies allow the seamless exchange of information across plant and enterprise networks.

OPC (OLE for Process Control),based on Microsoft DCOM technology, consists of a standard set of interfaces, properties and methods that can be used for device communication in process control and manufacturing applications. OPC has emerged as the worldwide industry standard since 1990s (Chistrolm, 1998), enabling connectivity and interoperability of plant floor information between disparate fieldbus networks, programmable controllers, distributed control systems, condition monitoring, plant asset management and production management systems. Nowadays, OPC has play an important role in process data interchange. Commercial DCS and control equipments usually provide OPC client or server capabilities. Nevertheless, in order to be independent of commercial products, it's necessary to develop an OPC client interface which can acquire data in real time from a series of processes, display the information in different formats, receive commands from the operators and send them to the process, in addition to other tasks such as data storage, control, alarm signaling, etc. Unfortunately, among the existing OPC client software products, it is difficult to meet the user-specified needs to connect OPC servers of different fieldbus standards (Hong and Jianhua, 2006; Schwarz and Boercsoek, 2007).

In this study, we developed a pilot OPC DA client interface based on OPC DA 2.0 standard which can connect a PLC controller and a DCS controller. The design and implementation of the integration are presented, with an application on a pilot experiment.

CONTROL SYSTEM INTEGRATION ARCHITECTURE AND CONFIGURATION

There are different levels of control system integration, suah as, integration of fieldbus at DCS's I/O layer, as DeltaV system of Fisher-Rosemount Co., or integration of fieldbus at DCS's network layer, etc. In this study, we developed an opc-based interface integration of DCS and PLC system, as shown in Fig. 1.
where, the OPC DA client has the following capacities:

Corresponding Author: Yang Maying, College of Information Engineering, Zhejiang University of Technology, 310014, Hangzhou, China

- It has basic interface of OPC DA which can connect with DCS and PLC servers simultaneously
- Process data are sampled and saved to the database,which can be displayed online or offline through remote server by OS/ES
- Software was compiled by C++

A detailed hardware realization of system integration of JX 300X DCS, made by SUPCON Co. and SIMATIC S7-300 PLC, made by SIEMENS Ltd., is illustrated in Fig. 2.

where, for JX300X DCS, SCnet II was used as process control network and the Ethernet card was used as the communication interface card. The AdvanTrol-X3.16 software was used for DCS configuration. JX300X DCS OPC V.1.0.0.2 was used as the OPC server.

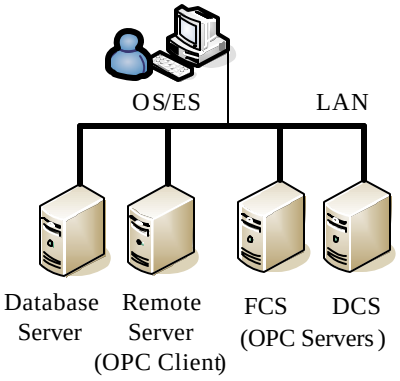

Fig. 1: OPC-based control system integration architecture

It should be noted that the resulting *.SCO file was added to the JX300X DCS OPC server, thus the automation instruments' data address can be obtained, to facilitate the data interchange with OPC DA client.SIMATIC STEP7 V5.3 and SIMATIC NET 6.0 were used for S7-300 configuration, where the hardware configuration can be shown in Fig. 3.

For the OPC server configuration, with SIMATIC NET 6.0, "Station Configuration Editor" was opened in the "Start" menu, items added, as illustrated in Fig. 4.

After PLC hardware configuration, SIMATIC Manager was opened, SIMATIC 300 (1) was downloaded to PLC and PC Station (FCS) to PC internal (local) (as shown in Fig. 5).

Thus the data transfer between PLC OPC server and OPC client can be realized.

OPC CLIENT INTERFACE DESIGN

Our client software was developed with Microsoft Visual C++ version 6.0, the required running environment was Microsoft Windows Pro 2000, OPC specification (OPC DA 2.0) and SDK package were needed. File"opccomn.h", "opccomn_i.c","opcda.h","opcda_i.c",

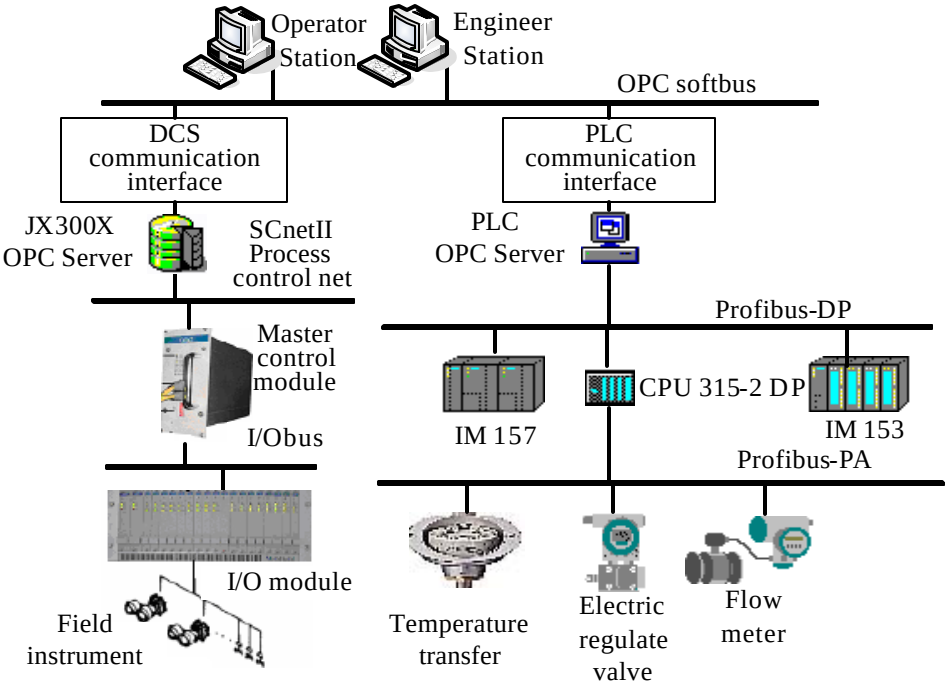

Fig. 2: hardware integration of JX 300X DCS and S7-300 PLC

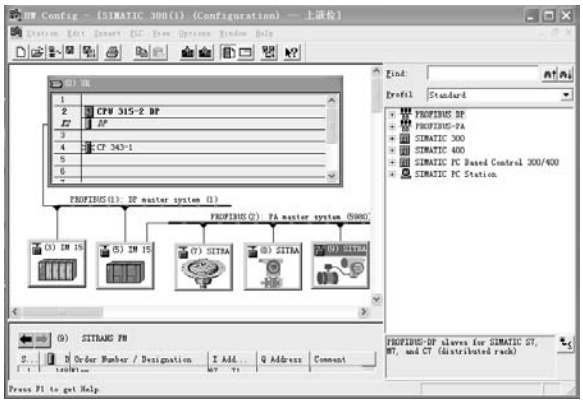

Fig. 3: S7-300 hardware configuration

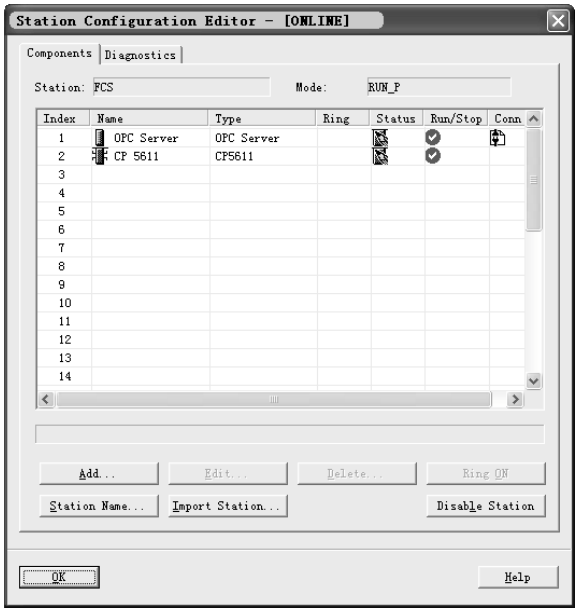

Fig. 4: PC Station configuration

"opcenum_i.c"in SDK package should be compiled to generate file "stdafx.h", added with "OpcEnum.exe", to establish a new project. "_WIN32_DCOM" was added in menu "Project"→"Settings…"→"C/C++"→"Preprocessor Definition", to visit OPC server remotely (OPC Foundation, 2011; Atzori *et al*., 2010; Guo *et al*., 2011; Chang and Liu, 2013; Huang and Chang, 2013; Mou *et al*., 2010; Mercurio *et al*., 2009).

The intercommunication task facing the OPC client include: establishing a server object; adding groups and items; data reading and writing; disconnecting OPC servers. The diagram can be shown in Fig. 6.

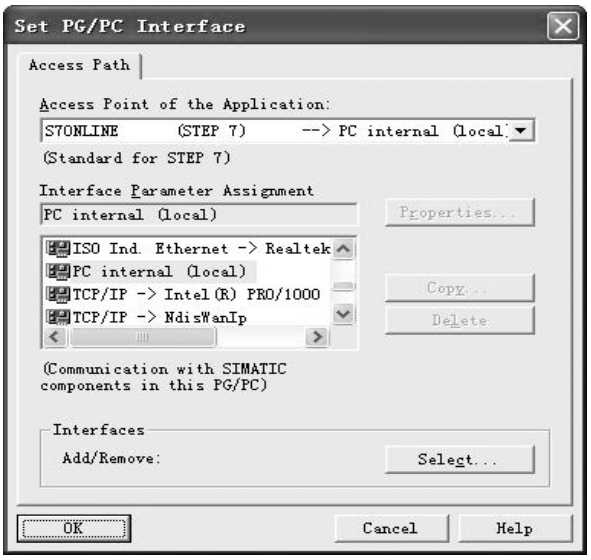

Fig. 5: PC/PG interface setting

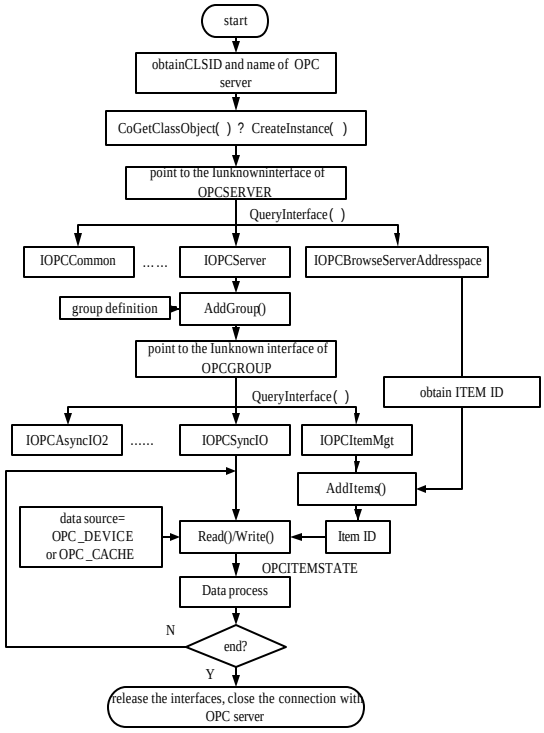

Fig. 6: Interchange diagram between client and server

Investigating OPC servers: It's important for the client software to investigate and find all the OPC servers. There are 3 steps for the OPC DA client to investigate all the OPC servers:

```
--Step 1: including head files:
#include <atlbase.h>;
#include <objbase.h>;
#include <comdef.h>;
#include "opcenum_i.c";
#import "OpcEnum.exe" no_namespace
--Step 2: defining CLSID by "OPC Data Access Servers Version 2.0";
static const CLSID CATID_OPCDAServer20 =
{ 0x63d5f432, 0xcfe4, 0x11d1, { 0xb2, 0xc8, 0x0, 0x60, 0x8, 0x3b, 0xa1,
0xfb } };
--Step 3: using IOPCServerList (provided by OpcEnum.exe) to get function
EnumClassesOfCategories(), to look for the OPC servers.
```

Creating OPC server object: The task of OPC client is to call the OPC server object interface and perform the including OPC items. Following are some codes describing the relation establishment with OPC servers and obtain of the server's interface pointer:

```
CLSID CLSID_OPCServer;
IOPCServer* pOPC=NULL;
 //defining the server's interface variable
char *szProgID =servName.GetBuffer(0);
 //changing server name to pointer
hr=CoInitialize(NULL);
 //initializing COM library
hr = CLSIDFromProgID(A2W(szProgID) andCLSID_OPCServer);
 //transforming to 128-bit unique indicator
hr = CoCreateInstance(CLSID_OPCServer, NULL, CLSCTX_ALL,
IID_IUnknown,
(LPVOID *)andpUnkn);
 //creating server object instance,
// connecting the server,
//obtaining server interface indicator
serverUnkHr[serverIndex]=pUnkn;
```

Creating OPC group object: OPC DA standard has defined the functions of AddGroup() and RemoveGroup() in IOPCServer interface. Once we get the IOPC Server pointer, we can call the 2 functions. Specification of the fastest rate at which data changes may be treated for items in this group should also be given. Here we used a group dialog block to set the parameters (Fig. 7).

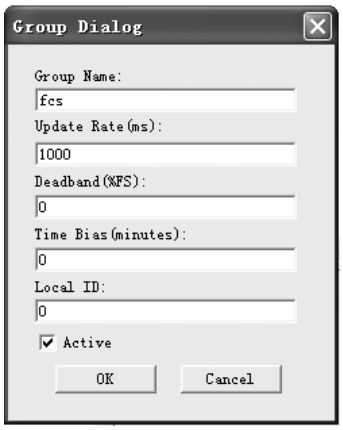

Fig. 7: Group dialog group

Creating OPC items: By investigating IOPCBrowse ServerAddressSpace interface, the client can browse the available data items in the server. Either flat or hierarchical address spaces can be designed to work well over a network. Here the CObList is used to save information and define items:

```
OPCITEMDEF* m_tempItems;
m_tempItems=new OPCITEMDEF[dwItems]; //for Item information
for (int i=0;i<dwItems;i++)
{
m_tempItems[i].szAccessPath =pwStr; //access path
m_tempItems[i].szItemID = itemString ; //item name
m_tempItems[i].bActive= TRUE; //item W/R status
m_tempItems[i].hClient= listServerCount+i;
//client handle
m_tempItems[i].dwBlobSize = 0; //dead zone size
m_tempItems[i].pBlob = NULL;
m_tempItems[i].vtRequestedDataType = 0;
 //data type
m_tempItems[0].wReserved=0; //returned data type
}
pOPCItemMg->AddItems(count, m_tempItems,
andm_pItemResult,andm_pErrors);
```

Data saving and reading, curve drawing: ADO (ActiveX Data Object) (Chistrolm, 1998) is used for data access. Main steps may include:

- Introduce ADO DLL: adding codes in application file "stdafx.h":

```
#import "c:\program files\common files\system\ado\msado15.dll"
no_namespace rename("EOF", "adoEOF")
```

- Initialize OLE/COMlibrary environment,linking database with Connection object
- After connection, execute SQL command.
- record proposal, disconnect and free object.

SYSTEM IMPLEMENTATION

Process description: The developed OPC client was used to construct pilot control equipment, shown in Fig. 8. where, component 1 and 2 denote the water tanks, whose liquid level can be sampled, component 3 is an electrical heater, component 4 and 5 are a lagged tube and a heat exchanger, respectively. All the inner temperature of the 3 tanks can be sampled. Component 6 and 7 are regulating valves.

OPC client test: The OPC based distributed control system was applied in the process. Firstly, the OPC client can show the implementation status, as illustrated in Fig 9 and 10.

It can be seen from Fig. 9 and 10 that the proposed OPC client has effective communication with distributed control equipment, the control responses can be shown clearly in the OPC client.

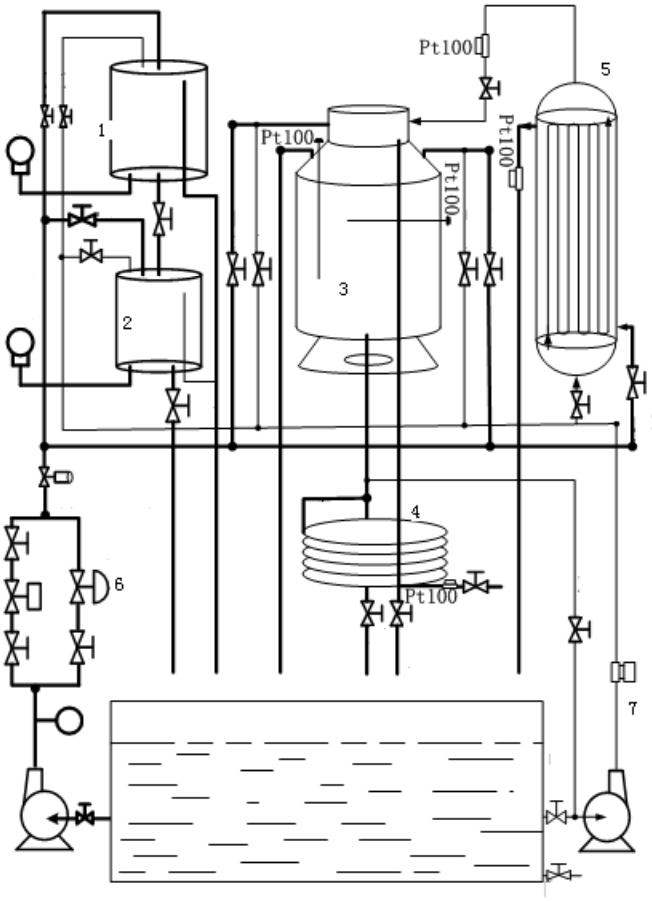

Fig. 8: Pilot process control equipment

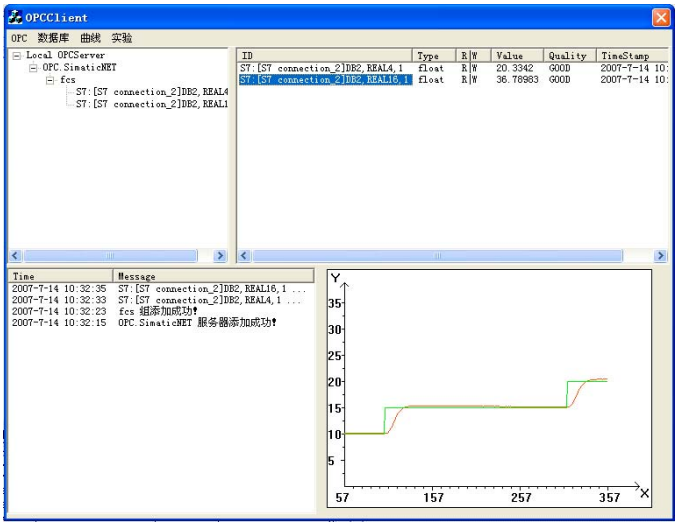

Fig. 9: OPC-based distributed control (status of PLC-controlled tank level)

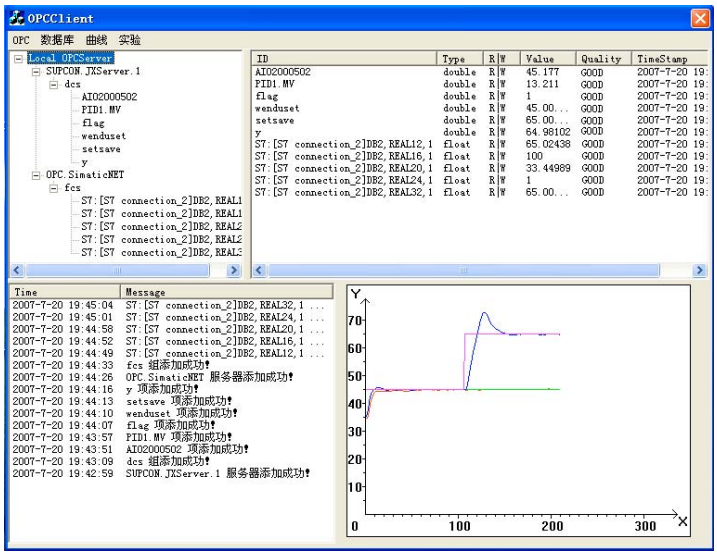

Fig. 10: OPC-based distributed control (status of DCS-controlled heater temperature)

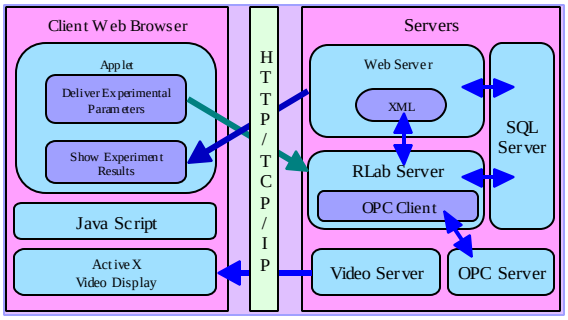

Fig. 11 OPC-based remote control system architecture

OPC-based Remote control experiment: Based on the OPC Client/Server data exchange mode and the OPC client, we constructed an OPC-based remote control laboratory system, as shown in Fig. 11.

The Browser/Server mode was applied in the OPC-based remote control system, where Java Applet and ActiveX technologies were used to develop the web client, data saving and delivering were realized with the help of XML and OPC technologies. Here the formerly developed OPC client acted as part of the remote control laboratory server (Rlab Server), to interfere with OPC servers. The Rlab Server was an application program independent of the Web Server, it administrated the users and data, etc. Fig. 12 shows briefly the functions of the Rlab Server.

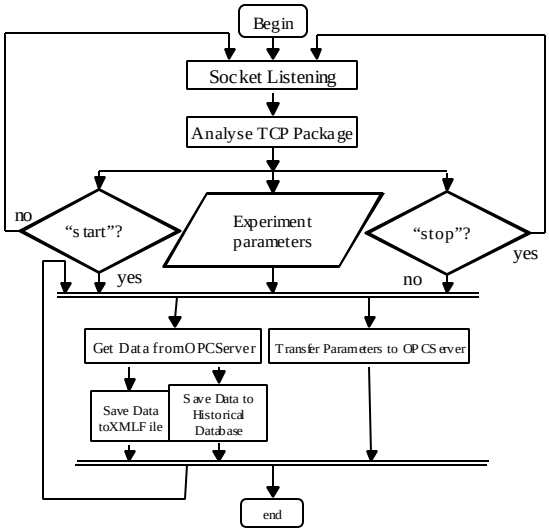

Fig 12 Simplified diagram of Rlab Server

With the remote control laboratory system, the experimental status can also be seen on the web client. Fig. 13 presents one of the control results.

These experimental results show that the OPC-based industrial control systems can achieve interconnection and information sharing among different control system networks, the OPC client interface is effective. Meanwhile, the OPC client provides a feasible way to realize the complex control algorithms (Skogestad, 2003).

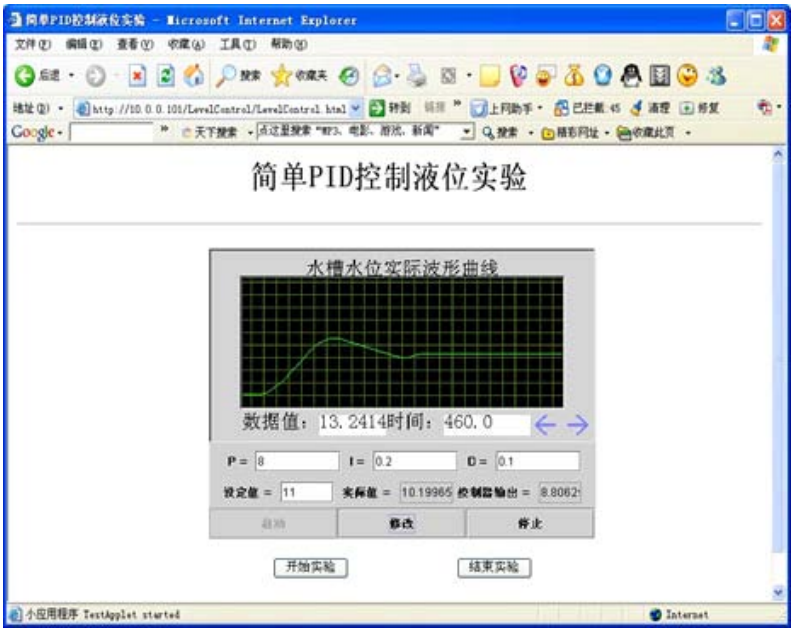

Fig. 13: Remote control results shown on the web client

CONCLUSION

In this study, we proposed an OPC-based control system integration platform which includes networked PLC-control and DCS-control and client-server platform. Based on the vendor provided OPC servers, the client interface was designed. A case study of a pilot control implementation was carried out to demonstrate the feasibility of the integration platform.

The OPC based integration platform presented in this study employs standard communication protocol and technology, namely, OPC technology, realizes data and services integration and inter-operation among distributed control systems. It utilizes the flexibility and universality of the OPC interface to implement the integration and inter-operation of distributed controls in LAN. In the meantime, it avoids the security risk of deploying OPC on the Internet directly. It can be invoked by FDD (Fault Diagnosis Detect) applications and plant-wide control and optimization applications.

ACKNOWLEDGMENT

The authors would like to thank for the support by the National High Technology Research and Development Program of China (863 Program 2009AA043204) and the national science and technology support program (No. 2012BAF05B01).

REFERENCES

Atzori, L., A. Iera and G. Morabito, 2010. The internet of things: A survey. Comput. Networks, 54: 2787-2805.

Chang, M. and Y.M. Liu, 2013. Design and implementation of the OPC client for the wet film thickness measurement system of galvanized line. Metall. Ind. Autom., 37: 54-58.

Chang, W.F., Y.C. Wu and C.W. Chiu, 2006. Development of a web-based remote load supervision and control system. Int. J. Electr. Power Energy Syst., 28: 401-407.

Chistrolm, A.L., 1998. OPC data access 2.0 technical overview. OLE for Process Control and Factory Automation, Intellution Inc., USA., October 1998.

Guo, X.J., J.C. Miao and Z.F. Wu, 2011. Data exchange client development based on OPC in instantaneous profile gauge. Atomic Energy Sci. Technol., 45: 1015-1019.

Hong, X. and W. Jianhua, 2006. Using standard components in automation industry: A study on OPC specification. Comput. Stand. Interfaces, 28: 386-395.

Huang, J.H. and X.M. Chang, 2013. Development and realization of OPC clients based on rapid development toolkit. Control Instrum. Chem. Ind., 40: 894-897.

Mercurio, A., A. Di-Giorgio and P. Cioci, 2009. Open-source implementation of monitoring and controlling services for EMS/SCADA systems by means of web services-IEC 61850 and IEC 61970 standards. IEEE Trans. Power Delivery, 24: 1148-1153.

Mou, H., C. Zheng and L. Hu, 2010. Implementation of component-based OPC in data acquisition. Metall. Ind. Autom., 34: 62-64.

OPC Foundation, 2011. OPC data access custom interface specification 3.0. OPC Foundation, Scottsdale, AZ., USA.

Schwarz, M.H. and J. Boercsoek, 2007. Survey on Ole for Process Control (OPC). Proceedings of the 7th WSEAS International Conference on Applied Computer Science, November 21-23, 2007, Venice, Italy, pp: 186-191.

Skogestad, S., 2003. Simple analytic rules for model reduction and PID controller tuning. J. Process Control, 13: 291-309.