BEST: International Journal of Management, Information Technology and Engineering (BEST: IJMITE)
Vol. 1, Issue 1, Oct 2013, 51-60

Best Journals
Knowledge to Wisdom

IMPLEMENTATION OF OPC-BASED COMMUNICATION BETWEEN TEMPERATURE PROCESS AND DCS ON LABVIEW PLATFORM

N. NITHYARANI

Assistant Professor, Department of Instrumentation & Control, Saranathan College of Engineering, Trichy, Tamil Nadu, India

ABSTRACT

Process automation market is one of the most dynamical industrial sectors. Typically the DCS used in complex process application where large amount of I/O data’s are required, such as oil refineries and chemical plant. DCS are multitasking systems able to handle great common data base and complex algorithms. Using big mainframe computer to control a entire plant was attempted back in the early '60s but it soon became clear that hang-up, one bug or failures could cause the whole plant to shutdown. Distributed control system gives a safety margin for that problems. In the modern industrial automation, LabVIEW is probably the most comprehensive environment for setting up a control/data acquisition system (CS) for a scientific/laboratory experiment. The main intent of this paper is to highlight the salient features of the present DCS used in the process industry and about sharing the temperature process parameters with LabVIEW and Honeywell DCS interfacing through NI OPC server. The NI Communication Server is used to interface the LabVIEW with any third party devices and also can easily analysis the temperature process with LabVIEW.

KEYWORDS: LabVIEW, DCS, NI OPC SERVER, Temperature Process

INTRODUCTION

In any of the Process control and automation application, controller design is the most important part. To control the temperature of outlet fluid of the process tank a conventional PID controller can be used. When PID controllers are used alone, it will can give poor plant performances and it will give some oscillatory response [1]. Due to inherent disadvantages, these conventional techniques were unable to give satisfactory response and results. Model based controller and an internal model based PID controller is developed to control the temperature of the process plant [3]. The main advantage to IMC is that is provides a transparent framework for control system design and tuning. Among the various PID tuning methods, IMC-PID has gained widespread acceptance in all the process industries, chemical industries because of its simplicity, robustness and successful in real time practical applications. IMC-PID controller for better set-point tracking of unstable processes [2]. The ideas behind the real time systems are an accumulation of years of industrial experience, but they could only become a reality with the help of the software platform that is LabVIEW. LabVIEW turned our dreams into a real time systems. In this project, model equations for closed loop temperature control has been developed and mathematically simulated using LabVIEW on the basis of available information and knowledge. The Temperature Control Loop System is studied and the data are collected to Excel by using step test data. The step test data are imported to System Identification Tool in MATLAB. The GUI Tool obtained the process data behavior with a first order plus dead time (FOPDT) dynamic model. The model parameters from the SI Tool is correlated and IMC and IMC-PID Controllers are designed and tuned for the temperature process [3]. By sharing the process parameters with LabVIEW and Honeywell DCS interfacing through NI OPC server [4]. The NI Communication Server is used to interface the LabVIEW with any third party devices and also can easily analysis the process with LabVIEW. The communication framework presented in this project has been described in details to provide a ready to use solution for implementing a DCS with LabVIEW for further analysis.

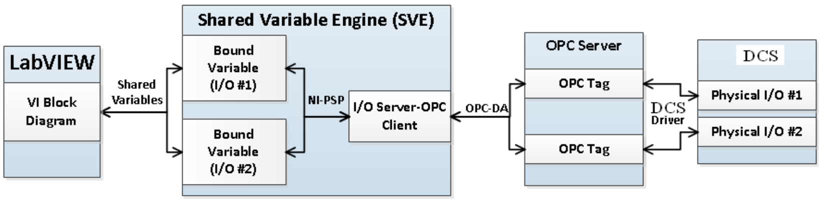


Figure 1: Block Diagram

DISTRIBUTED CONTROL SYSTEM

A Distributed Control System (DCS) refers to a control system which is usually for manufacturing system. DCS is a kind of dynamic system, in which the controller is the main part, the control elements are distributed throughout the system with each component sub-system controlled by one or more controllers. DCS can perform for Complex algorithms to get a accurate control action [5]. By using DCS leads to various advantages like the overall plant optimization, ease in operation and in maintenance tasks. The overall communication through OPC Protocol.

TEMPERATURE PROCESS CONTROL

The aim of the paper is to control or to maintain the temperature of a plant within a desired limit. The temperature controller can be used to control the temperature of any plant. Temperature process is highly a non linear process. Typically it contains three units in the entire setup, a Controller unit, Temperature input unit (heater) and the Control output unit. The Analog-to-Digital (ADC) unit together with temperature sensor forms the temperature input unit for the system. The solid state relay (SSR) driver forms the control output unit for the system. Electric power to the heating element (coil) is supplied through relay contacts in the system. The switching ON/OFF of the relay controls the heat supplied to the plant. Here the process with a liquid tank with a heater, and with two temperature thermocouple sensors [2]. A temperature control system for a liquid tank with heater is simulated. The water is heated by a heater which controlled by the controller. The temperature is measured by a sensor which in practice may be a thermocouple.

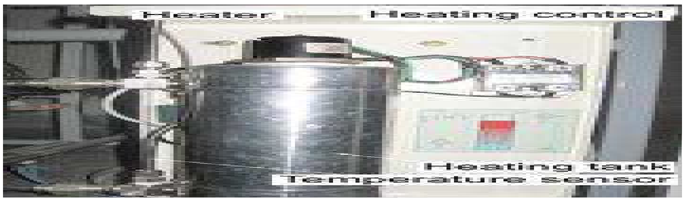


Figure 2: Temperature Control System

The temperature process model on which the simulator is based on the energy balance equation as First-Order system under some assumption for the system as, homogenous conditions of the liquid in the tank. The temperature process model in this paper contains a time delay. The time delay taken in this paper exists between an excitation of the heater and the response in the feedback temperature sensor. In addition the simulator contains a First-Order transfer function representing a time constant in the heater.

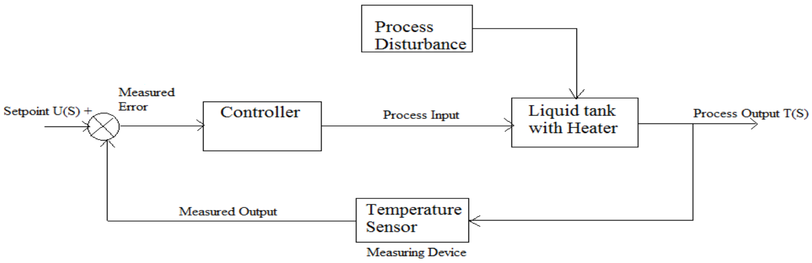


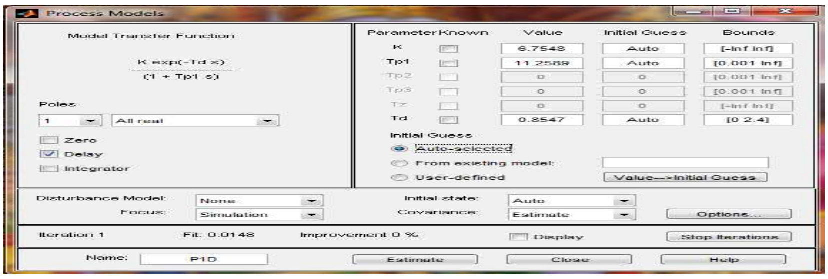
Figure 3: Block Diagram of a Temperature Process Control

Temperatures often have to be set for the large machines or systems or for the process. The process setting should not change when faults occur. Such tasks are undertaken by the closed-loop controllers. The controlled variable is first measured and an electrical signal is created to allow an independent closed-loop controller to control the process variable. The measured output in the controller must then be compared with the desired value or the desired-value set point [2]. The result of this comparison determines any control actions that needs to be taken. Finally a suitable location must be found in the process system where the controlled output can be influenced. In the temperature closed-loop control system the task is to keep the outlet temperature at the desired value or to follow the desired-value set point.

NEED OF CONTROL

The objective of this paper is controller designing and the controller is to maintain the temperature inside the liquid tank at a desired value. Temperature process model is in the form of First Order Plus Delay Time (FOPDT) [1]. Precise temperature control is a challenge faced by various process industries. In practical all the temperature process are highly non-linear and sluggish in nature and the process modeling for the system becomes complex.

The step test data is performed for the process. For fixed input heater supply the temperature is allowed to increase to the desired level from 0°C. At each sample time the data from temperature transmitter i.e., from thermocouple between 4-20 m A is being collected and fed to the process through the interfacing module and the serial port RS 232. The obtained response from the constant input heater supply is called Process Reaction Curve (PRC). From the PRC curve the data are exported to excel. Then by using MATLAB System Identification Tool, the data are imported and process is modeled and approximated as First Order Plus Delay Time (FOPDT) process which includes K, τ and td. The experimental data are approximated to be a FOPDT model and the identified model of the process is: 7/ (12s+1) * e^{-tds}



Model Transfer Function

$$\frac{K \exp(-Td\ s)}{(1 + Tp1\ s)}$$

Poles: 1 All real

□ Zero
☑ Delay
□ Integrator

Parameter	Known	Value	Initial Guess	Bounds
K	□	6.7548	Auto	[-Inf Inf]
Tp1	□	11.2589	Auto	[0.001 Inf]
Tp2	□	0	0	[0.001 Inf]
Tp3	□	0	0	[0.001 Inf]
Tz	□	0	0	[-Inf Inf]
Td	□	0.8547	Auto	[0 2.4]

Initial Guess
● Auto-selected
○ From existing model:
○ User-defined

Disturbance Model: None
Focus: Simulation
Initial state: Auto
Covariance: Estimate

Iteration 1 Fit: 0.0148 Improvement 0 %
□ Display

Name: P1D

Figure 4: Process Model

OPC-OLE FOR PROCESS CONTROL

OLE (Object Linking and Embedding) for Process Control. It is a standard specifies communication of real-time plant data between the control devices from different manufacturers. OPC Foundation is the Interoperability Standard for Industrial Automation. OPC is also about Open Productivity & Connectivity. It is a software interface module that allows windows program to communicate with all type of industrial hardware devices [5]. The NI OPC server is one of the module in a Lab VIEW software that converts the hardware communication protocol used by a DCS (hardware device) into the OPC protocol. The OPC client is an another module in LabVIEW software that needs to connect to the hardware such as HMI. Which sends commands to the hardware device [6]. A major goal of the NI OPC Foundation and the OPC Data Access specification was to eliminate the need of client application vendor's to develop their own proprietary set of communications drivers. OPC improves automation technology. It connects between people, between systems/process and between technologies. Its an enabling technology[7]. It is a three way of communications which includes OPC client, NI OPC server and the process devices. At the lowest level OPC get raw data from the field into a SCADA or into a DCS. The NI OPC that fills the industry-specific needs. It is widely used in the automation and HMI/SCADA applications [8]. The NI OPC Specification was based on the OLE, COM, and DCOM technologies developed by Microsoft for the Microsoft Windows operating system family [6]. The most commonly used OPC specification is OPC Data Access specification, which is used to read and write real-time data.

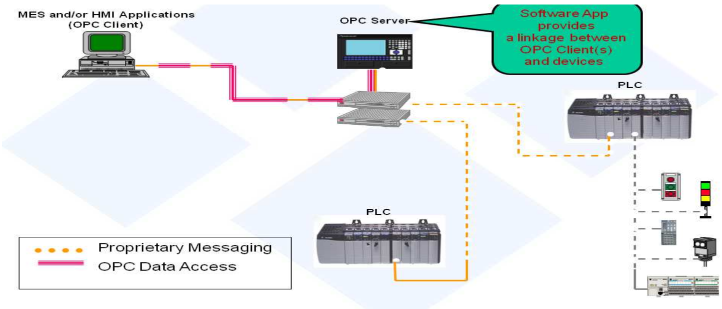
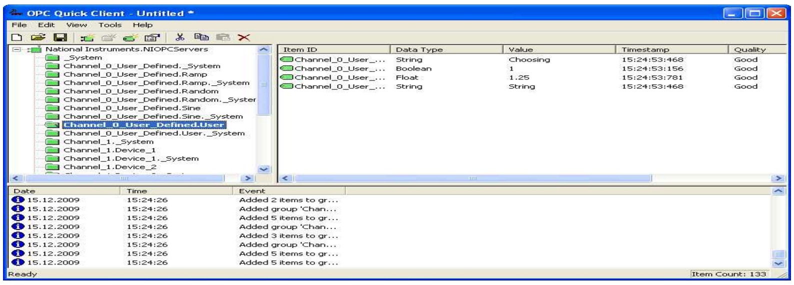

MES and/or HMI Applications
(OPC Client)
OPC Server
Software App provides a linkage between OPC Client(s) and devices
PLC
PLC
Proprietary Messaging
OPC Data Access

Figure 5: OPC Architecture Message

OPC CLIENT

The **OPC Quick Client** has been developed to assist in the test and development of the OPC Data Access 1.0 and 2.0 Servers. The OPC Quick Client does not fully support OPC DA 3.0. The OPC Quick Client supports both local and remote OPC server connections. Remote connections are handled through the operating system's DCOM interface [8].



National Instruments.NIOPCServers
- _System
- Channel_0_User_Defined._System
- Channel_0_User_Defined.Ramp
- Channel_0_User_Defined.Ramp._System
- Channel_0_User_Defined.Random
- Channel_0_User_Defined.Random._Syster
- Channel_0_User_Defined.Sine
- Channel_0_User_Defined.Sine._System
- Channel_0_User_Defined.User
- Channel_0_User_Defined.User._System
- Channel_1._System
- Channel_1.Device_1
- Channel_1.Device_1._System
- Channel_1.Device_2

Item ID	Data Type	Value	Timestamp	Quality
Channel_0_User_...	String	Choosing	15:24:53:468	Good
Channel_0_User_...	Boolean	1	15:24:53:156	Good
Channel_0_User_...	Float	1.25	15:24:53:781	Good
Channel_0_User_...	String	String	15:24:53:468	Good

Date	Time	Event
15.12.2009	15:24:26	Added 2 items to gr...
15.12.2009	15:24:26	Added group 'Chan...
15.12.2009	15:24:26	Added 5 items to gr...
15.12.2009	15:24:26	Added group 'Chan...
15.12.2009	15:24:26	Added 3 items to gr...
15.12.2009	15:24:26	Added group 'Chan...
15.12.2009	15:24:26	Added 5 items to gr...
15.12.2009	15:24:26	Added 5 items to gr...

Ready Item Count: 133

Figure 6: OPC Quick Client

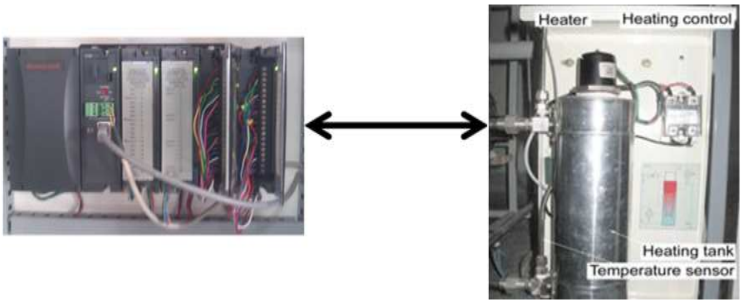
PROJECT EXECUTION

COMPOSITION OF TEST SYSTEM

The composition of the remote real-time communication test system between LabVIEW and DCS. The systematic basic hardware items are shown as following: A PC (Personal Computer) furnished with an ordinary network card; A Honeywell HC900-C30 with Ethernet communication card. This card is supported by some protocols, such as ISO, TCP/IP and UDP, etc. RJ45 interface which is used for connecting with industry Ethernet, and through its own microprocessor. It can deal with the data communication of industry Ethernet independently, and has an only IP address preserved, also it can be put into operation directly through the network; A DI 16*DC 24V digital input module; A DO 16*DC 24V/0.5A digital output module; An AI 8*12BIT analog input module; An AO 4*16BIT analog output module; In addition, the RJ45 industrial Ethernet cable communicate the DCS and the NI OPC server and also the same cable communicate the logical program interfacing with Hybrid control designer.

PROCEDURE

Step 1: The present controlling method, the Honeywell HC900-C30 series DCS control the temperature process.



HC900-C30 DCS TEMPERATURE PROCESS

Figure 7: DCS and Temperature Process Interfacing

Step 2: The Function Block Diagram (FBD) of PID controller for the Temperature process

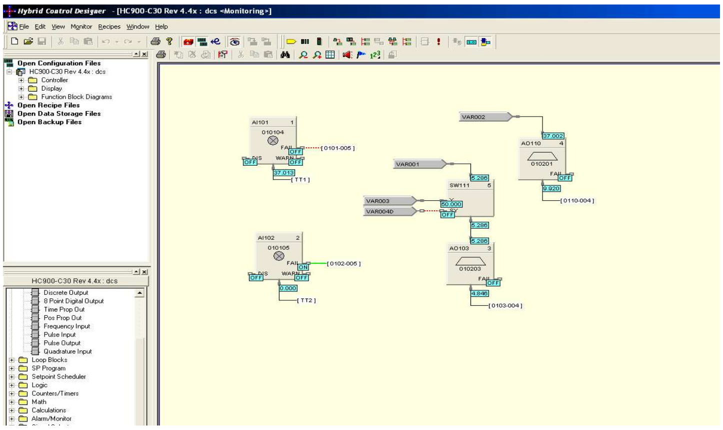
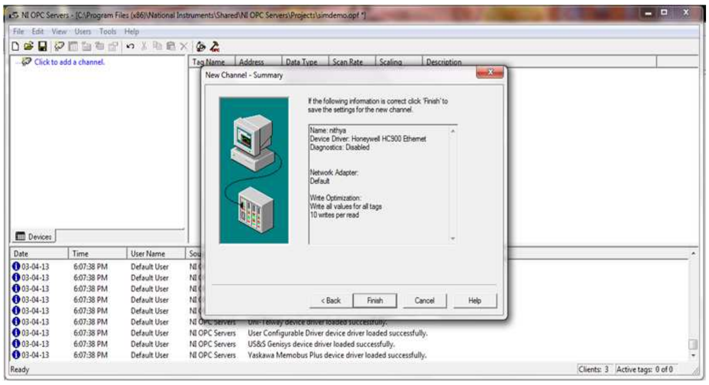


Figure 8: Process Program in DCS

Step 3: Summary of the Lab VIEW and DCS are interfacing using NI OPC Server. The Channel configuration as follows



If the following information is correct click 'Finish' to save the settings for the new channel.

Name: nithya
Device Driver: Honeywell HC900 Ethernet
Diagnostics: Disabled

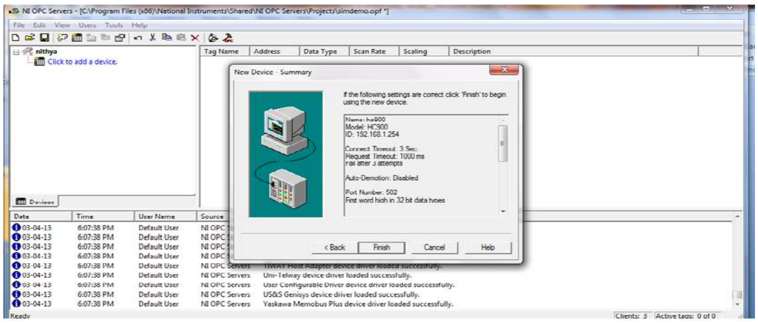
Network Adapter:
Default

Write Optimization:
Write all values for all tags
10 writes per read

Date	Time	User Name	Sou...	
03-04-13	6:07:38 PM	Default User	NI ...	
03-04-13	6:07:38 PM	Default User	NI ...	
03-04-13	6:07:38 PM	Default User	NI ...	
03-04-13	6:07:38 PM	Default User	NI ...	
03-04-13	6:07:38 PM	Default User	NI ...	
03-04-13	6:07:38 PM	Default User	NI OPC Servers	Uni-Telway device driver loaded successfully.
03-04-13	6:07:38 PM	Default User	NI OPC Servers	User Configurable Driver device driver loaded successfully.
03-04-13	6:07:38 PM	Default User	NI OPC Servers	US&S Genisys device driver loaded successfully.
03-04-13	6:07:38 PM	Default User	NI OPC Servers	Yaskawa Memobus Plus device driver loaded successfully.

Figure 9: Summary of Channel Configuration

Step 4: The summary of Device configuration in NI OPC Server.



NI OPC Servers - [C:\Program Files (x86)\National Instruments\Shared\NI OPC Servers\Projects\simdemo.opf *]

Tag Name	Address	Data Type	Scan Rate	Scaling	Description

If the following settings are correct click 'Finish' to begin using the new device.

Name: hc900
Model: HC900
ID: 192.168.1.254

Connect Timeout: 3 Sec.
Request Timeout: 1000 ms
Fail after 3 attempts

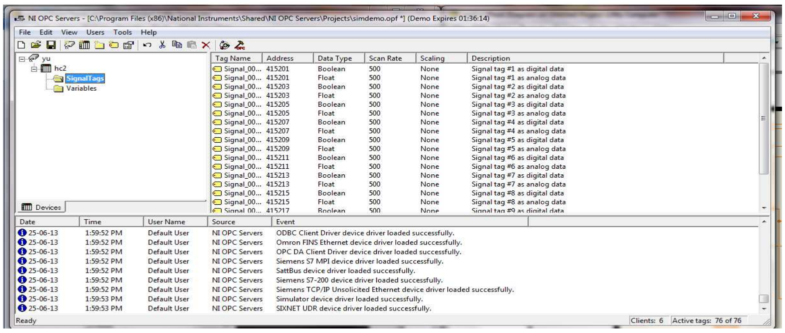
Auto-Demotion: Disabled

Port Number: 502
First word high in 32 bit data types

```
Date        Time        User Name     Source          Event
03-04-13    6:07:38 PM  Default User  NI OPC
03-04-13    6:07:38 PM  Default User  NI OPC
03-04-13    6:07:38 PM  Default User  NI OPC
03-04-13    6:07:38 PM  Default User  NI OPC
03-04-13    6:07:38 PM  Default User  NI OPC Servers  TIWAY Host Adapter device driver loaded successfully.
03-04-13    6:07:38 PM  Default User  NI OPC Servers  Uni-Telway device driver loaded successfully.
03-04-13    6:07:38 PM  Default User  NI OPC Servers  User Configurable Driver device driver loaded successfully.
03-04-13    6:07:38 PM  Default User  NI OPC Servers  US&S Genisys device driver loaded successfully.
03-04-13    6:07:38 PM  Default User  NI OPC Servers  Yaskawa Memobus Plus device driver loaded successfully.
Ready                                                                   Clients: 3  Active tags: 0 of 0
```

Figure 10: Summary of Device Configuration

Step 5: The NI OPC Server and DCS interfacing Signal Tags



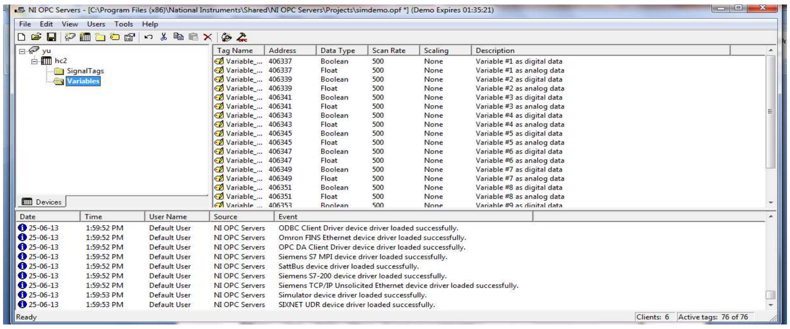
NI OPC Servers - [C:\Program Files (x86)\National Instruments\Shared\NI OPC Servers\Projects\simdemo.opf *] (Demo Expires 01:36:14)

Tag Name	Address	Data Type	Scan Rate	Scaling	Description
Signal_00...	415201	Boolean	500	None	Signal tag #1 as digital data
Signal_00...	415201	Float	500	None	Signal tag #1 as analog data
Signal_00...	415203	Boolean	500	None	Signal tag #2 as digital data
Signal_00...	415203	Float	500	None	Signal tag #2 as analog data
Signal_00...	415205	Boolean	500	None	Signal tag #3 as digital data
Signal_00...	415205	Float	500	None	Signal tag #3 as analog data
Signal_00...	415207	Boolean	500	None	Signal tag #4 as digital data
Signal_00...	415207	Float	500	None	Signal tag #4 as analog data
Signal_00...	415209	Boolean	500	None	Signal tag #5 as digital data
Signal_00...	415209	Float	500	None	Signal tag #5 as analog data
Signal_00...	415211	Boolean	500	None	Signal tag #6 as digital data
Signal_00...	415211	Float	500	None	Signal tag #6 as analog data
Signal_00...	415213	Boolean	500	None	Signal tag #7 as digital data
Signal_00...	415213	Float	500	None	Signal tag #7 as analog data
Signal_00...	415215	Boolean	500	None	Signal tag #8 as digital data
Signal_00...	415215	Float	500	None	Signal tag #8 as analog data
Signal_00...	415217	Boolean	500	None	Signal tag #9 as digital data

```
Date        Time        User Name     Source          Event
25-06-13    1:59:52 PM  Default User  NI OPC Servers  ODBC Client Driver device driver loaded successfully.
25-06-13    1:59:52 PM  Default User  NI OPC Servers  Omron FINS Ethernet device driver loaded successfully.
25-06-13    1:59:52 PM  Default User  NI OPC Servers  OPC DA Client Driver device driver loaded successfully.
25-06-13    1:59:52 PM  Default User  NI OPC Servers  Siemens S7 MPI device driver loaded successfully.
25-06-13    1:59:52 PM  Default User  NI OPC Servers  SattBus device driver loaded successfully.
25-06-13    1:59:52 PM  Default User  NI OPC Servers  Siemens S7-200 device driver loaded successfully.
25-06-13    1:59:52 PM  Default User  NI OPC Servers  Siemens TCP/IP Unsolicited Ethernet device driver loaded successfully.
25-06-13    1:59:53 PM  Default User  NI OPC Servers  Simulator device driver loaded successfully.
25-06-13    1:59:53 PM  Default User  NI OPC Servers  SIXNET UDR device driver loaded successfully.
Ready                                                                   Clients: 6  Active tags: 76 of 76
```

Figure 11: NI OPC Server Signal Tags

Step 6: The NI OPC Server and DCS Interfacing Variables



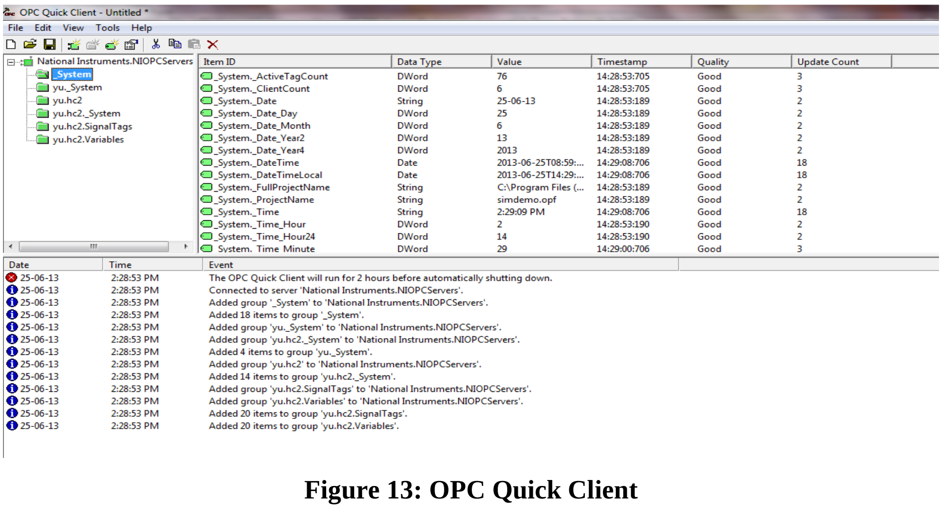
NI OPC Servers - [C:\Program Files (x86)\National Instruments\Shared\NI OPC Servers\Projects\simdemo.opf *] (Demo Expires 01:35:21)

Tag Name	Address	Data Type	Scan Rate	Scaling	Description
Variable_...	406337	Boolean	500	None	Variable #1 as digital data
Variable_...	406337	Float	500	None	Variable #1 as analog data
Variable_...	406339	Boolean	500	None	Variable #2 as digital data
Variable_...	406339	Float	500	None	Variable #2 as analog data
Variable_...	406341	Boolean	500	None	Variable #3 as digital data
Variable_...	406341	Float	500	None	Variable #3 as analog data
Variable_...	406343	Boolean	500	None	Variable #4 as digital data
Variable_...	406343	Float	500	None	Variable #4 as analog data
Variable_...	406345	Boolean	500	None	Variable #5 as digital data
Variable_...	406345	Float	500	None	Variable #5 as analog data
Variable_...	406347	Boolean	500	None	Variable #6 as digital data
Variable_...	406347	Float	500	None	Variable #6 as analog data
Variable_...	406349	Boolean	500	None	Variable #7 as digital data
Variable_...	406349	Float	500	None	Variable #7 as analog data
Variable_...	406351	Boolean	500	None	Variable #8 as digital data
Variable_...	406351	Float	500	None	Variable #8 as analog data
Variable_...	406353	Boolean	500	None	Variable #9 as digital data

```
Date        Time        User Name     Source          Event
25-06-13    1:59:52 PM  Default User  NI OPC Servers  ODBC Client Driver device driver loaded successfully.
25-06-13    1:59:52 PM  Default User  NI OPC Servers  Omron FINS Ethernet device driver loaded successfully.
25-06-13    1:59:52 PM  Default User  NI OPC Servers  OPC DA Client Driver device driver loaded successfully.
25-06-13    1:59:52 PM  Default User  NI OPC Servers  Siemens S7 MPI device driver loaded successfully.
25-06-13    1:59:52 PM  Default User  NI OPC Servers  SattBus device driver loaded successfully.
25-06-13    1:59:52 PM  Default User  NI OPC Servers  Siemens S7-200 device driver loaded successfully.
25-06-13    1:59:52 PM  Default User  NI OPC Servers  Siemens TCP/IP Unsolicited Ethernet device driver loaded successfully.
25-06-13    1:59:53 PM  Default User  NI OPC Servers  Simulator device driver loaded successfully.
25-06-13    1:59:53 PM  Default User  NI OPC Servers  SIXNET UDR device driver loaded successfully.
Ready                                                                   Clients: 6  Active tags: 76 of 76
```

Figure 12: NI OPC Server Variables

Step 7: OPC Quick Client



OPC Quick Client - Untitled *

Item ID	Data Type	Value	Timestamp	Quality	Update Count
_System._ActiveTagCount	DWord	76	14:28:53:705	Good	3
_System._ClientCount	DWord	6	14:28:53:705	Good	3
_System._Date	String	25-06-13	14:28:53:189	Good	2
_System._Date_Day	DWord	25	14:28:53:189	Good	2
_System._Date_Month	DWord	6	14:28:53:189	Good	2
_System._Date_Year2	DWord	13	14:28:53:189	Good	2
_System._Date_Year4	DWord	2013	14:28:53:189	Good	2
_System._DateTime	Date	2013-06-25T08:59:...	14:29:08:706	Good	18
_System._DateTimeLocal	Date	2013-06-25T14:29:...	14:29:08:706	Good	18
_System._FullProjectName	String	C:\Program Files (...	14:28:53:189	Good	2
_System._ProjectName	String	simdemo.opf	14:28:53:189	Good	2
_System._Time	String	2:29:09 PM	14:29:08:706	Good	18
_System._Time_Hour	DWord	2	14:28:53:190	Good	2
_System._Time_Hour24	DWord	14	14:28:53:190	Good	2
_System._Time_Minute	DWord	29	14:29:00:706	Good	3

```
Date        Time        Event
25-06-13    2:28:53 PM  The OPC Quick Client will run for 2 hours before automatically shutting down.
25-06-13    2:28:53 PM  Connected to server 'National Instruments.NIOPCServers'.
25-06-13    2:28:53 PM  Added group '_System' to 'National Instruments.NIOPCServers'.
25-06-13    2:28:53 PM  Added 18 items to group '_System'.
25-06-13    2:28:53 PM  Added group 'yu._System' to 'National Instruments.NIOPCServers'.
25-06-13    2:28:53 PM  Added group 'yu.hc2._System' to 'National Instruments.NIOPCServers'.
25-06-13    2:28:53 PM  Added 4 items to group 'yu._System'.
25-06-13    2:28:53 PM  Added group 'yu.hc2' to 'National Instruments.NIOPCServers'.
25-06-13    2:28:53 PM  Added 14 items to group 'yu.hc2._System'.
25-06-13    2:28:53 PM  Added group 'yu.hc2.SignalTags' to 'National Instruments.NIOPCServers'.
25-06-13    2:28:53 PM  Added group 'yu.hc2.Variables' to 'National Instruments.NIOPCServers'.
25-06-13    2:28:53 PM  Added 20 items to group 'yu.hc2.SignalTags'.
25-06-13    2:28:53 PM  Added 20 items to group 'yu.hc2.Variables'.
```

Figure 13: OPC Quick Client

Step 8: The IMC PID tune set in LabVIEW. The Proportional Gain = 0. 17, Integral Time = 12.5 and Derivative Time = 0.48

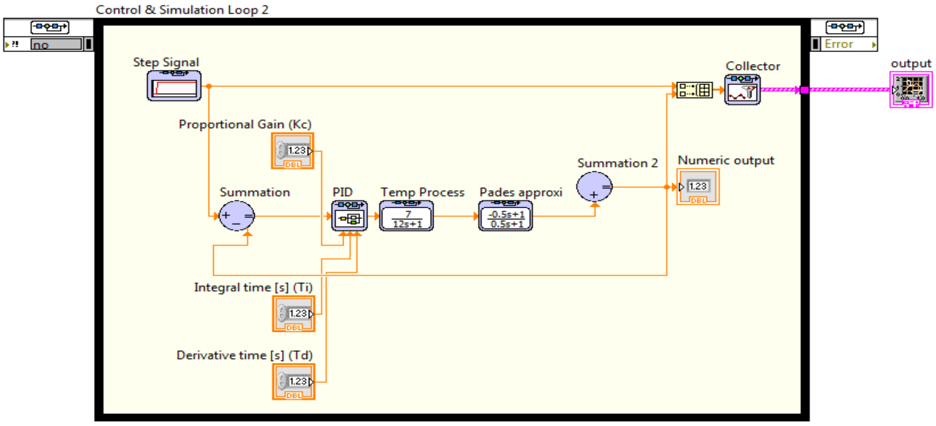


Figure 14: IMC PID Program Block Diagram in LabVIEW

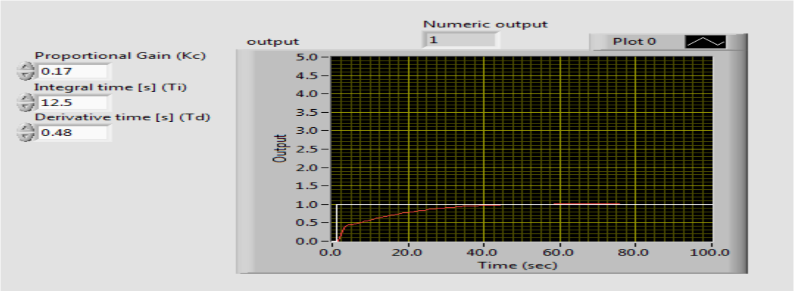

Proportional Gain (Kc) 0.17
Integral time [s] (Ti) 12.5
Derivative time [s] (Td) 0.48
Numeric output 1

Figure 15: Output of IMC PID Controller Parameter

Step 9: The tuning parameters are changed through the LabVIEW to DCS, the result as shown below

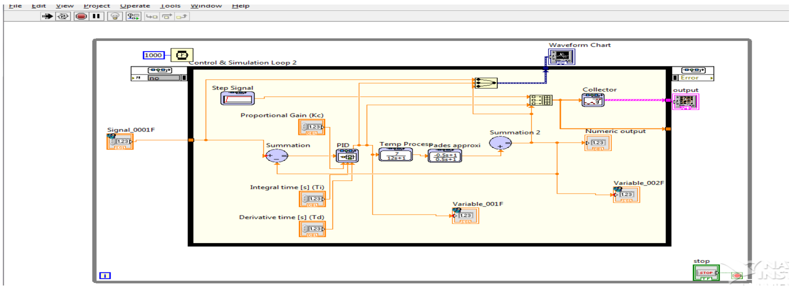


Figure 16: Parameter Controlled by NI OPC Server Block Diagram

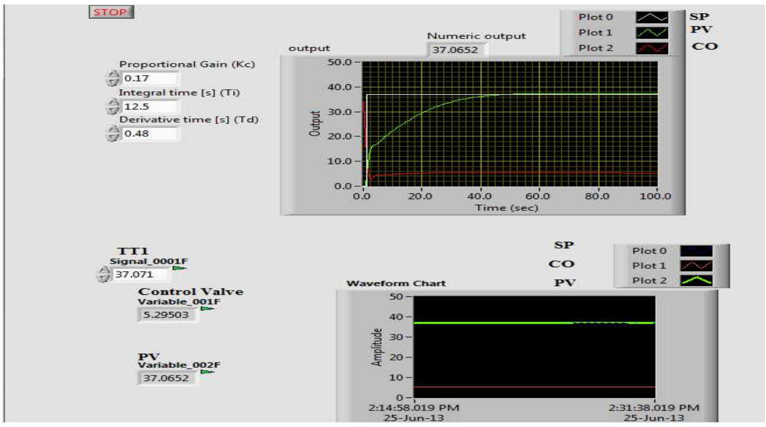

Proportional Gain (Kc) 0.17
Integral time [s] (Ti) 12.5
Derivative time [s] (Td) 0.48
Numeric output 37.0652
TT1 Signal_0001F 37.071
Control Valve Variable_001F 5.29503
PV Variable_002F 37.0652

Figure 17: Parameter Controlled by NI OPC Server Front Panel

RESULTS AND CONCLUSIONS

In the past, client application vendors like LabVIEW, DCS, MATHLAB, Wonderware (InTouch) etc., had to develop a different driver for each control device [12]. The NI OPC standard provides the greatest benefit to client vendors in that they now only have to develop one driver to access data from a process control device [14]. Both of these situations are diagrammed in Figure 18.

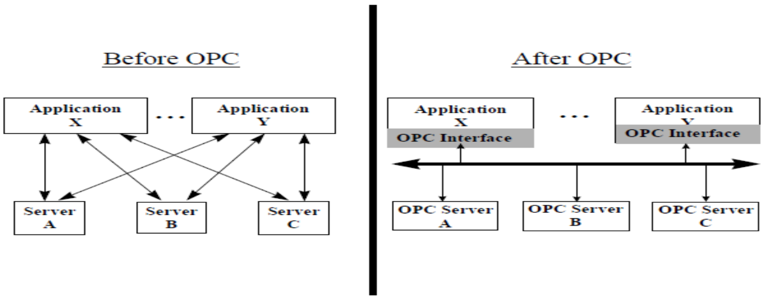


Figure 18: Before and After OPC

If the control device vendor modified the interface to the device, the client vendor would also have to modify the client's driver. OPC isolates the client software from the details of the various systems below the OPC interface [15]. The device vendor is able to modify the functionality under the OPC server interface without affecting the client software [16]. Client vendors can now expend resources on true value added activities for their products in place of the effort required to maintain a library of device drivers.

REFERENCES

1. J.A.Arulandu, Braatz and Horn, I.G., C.J.Gombas, J.G.VanAntwerp and R.D.Braatz, "Improved Filter Design in Internal Model Control " Imd.Eng.Chem.Res.,35(10), 3437-3441
2. Chien, I.L. and P.S.Fruehauf, IMC-Based PID controllers for "Real PID" control. "Consider IMC Tuning To Improve Controller Performance" Chem.Eng.Prog.86(10), 33-41
3. European Journal of Scientific Research ISSN 1450-216X Vol.78 No.1 (2012), pp.65-73EuroJournals Publishing, Inc. 2012 http://www.europeanjournalofscientificresearch.com
4. OPC Foundation. OPC Data Access Custom Interface Standard Version 2.05[EB/OL] (2001-12). http://www.opcfoundation.org.
5. OPC Foundation. OPC Data Access Automation Interface Standard Version 2.02[EB/OL] 1999-02. http://www. opcfoundation .org.
6. Xu Hong, Wang Jianhua, Using standard components in automation industry: A study on OPC Specification [J]. Computer Standards & Interfaces, 2006, 28, 386–395.
7. NI OPC server – www.ni.co.in
8. Winter Are, Winter P. 'Using Microsoft Office for Windows 95'. Que Corporation.
9. Rehbein DA. 'OLE for Process Control (OPC) Primer'. Fisher-Rosemount Systems, Inc.
10. Delta V (whitepaper) of Emerson Process Management
11. Web Access Device Reference Manual, ASI Controls Link OPC Server

12. Kraig Brockschmidt, Inside OLE, Second Edition, Microsoft Press, Redmond, WA, 1995.

13. Microsoft Systems Journal, Q&A, April 1996, pp. 89-101.

14. OLE Automation Programming Reference, Microsoft Press, Redmond, WA, 1996.

15. OLE 2 Programming Reference, Vol. 1, Microsoft Press, Redmond, WA, 1994.

16. OPC Data Access Custom Interface Standard, Version 2.0, OPC Foundation 1998.