International Journal of Computational Intelligence Systems, Vol. 7, No. 1 (February 2014), 37-49

Advanced control software framework for process control applications

Shazzat Hossain
UMPEDAC[*]*, Department of Computer System and Technology*
University Malaya, Kuala Lumpur, 50603, Malaysia
E-mail: shazzat@siswa.um.edu.my
www.um.edu.my

M.A. Hussain
UMPEDAC, Department of Chemical Engineering,
University Malaya, Kuala Lumpur, 50603, Malaysia
E-mail: mohd_azlan@um.edu.my
www.um.edu.my

Rosli Bin Omar
Department of Electrical Engineering,
University Malaya
Kuala Lumpur, 50603, Malaysia
E-mail: rosli_omar@um.edu.my
www.um.edu.my

Received 2 October 2011

Accepted 26 May 2013

Abstract

Industries are now moving towards PC-based control from PLC (Programmable Logic Control) based control as the PC (personal computer) is easily available and capable of implementing various control strategies to improve productivity. Advances in both hardware and software technology are expediting this move. Absence of a user friendly, cost effective and easy to apply advanced control strategy based software is the motivation for this work. In this paper we reviewed issues of software application in process control systems. We present the development of an advance process control software tool and its application on a pH neutralization plant. Neural network model based DIC (Direct Inverse Control) strategy is implemented to control the process. The application and results shows the viability and robustness of the advanced control software and its superiority over the conventional PID (proportional–integral–derivative) controller.

Keywords: Process control, software tool, Neural Network, Direct Inverse Control, pH Neutralization Process.

[*] UM Power Energy Dedicated Advanced Center

1. Introduction

Advances in both hardware and software technology enable us to build smart systems. Process control system is also not an exception. Industrial automation or control software is being used in various branches of industry such as fault diagnosis, condition monitoring, plant process monitoring and process control [1]. Due to the increase of software use and complexity of process system itself, various challenges such as delivering cost-effective, efficient and flexible solutions for control are coming forward for today's engineers [2]. At the same time it is expected that technologies incurring big money and longer time to fix or replace, should be avoided. So to meet all these challenges, most companies are relying on software for automatic control of the industrial process as software is more adaptable, easily replaceable and cost effective [3]. Furthermore hardware is not ever lasting, it is subjected to limited life time, not robust, and expensive to replace or modify. Moreover, nowadays computer technologies have become cheaply available. Development of engineering software is done with lots of essential features either in the graphical presentation or applying more complex control strategies to enhance the control performance in control system engineering [4]. Nowadays software can handle complex task with priority towards precision, speed, flexibility and reliability.

Leading process control companies mostly use advanced process control technology to drive their costs down, increase their throughput, and improve control performance. However medium and small scale industries are still using the PLC based control or embedded conventional control due to the fact that the online control software environment along with advanced control facility requires expertise to use them effectively. Most companies only have good user interface to visualize the control system but lack in control strategies selection. Some provide good control strategies but is difficult to configure the controller settings. Today advanced control strategy solutions are also readily available but not easy to use for the average user, need expertise knowledge and above all they are expensive, platform dependent and are proprietary module dependent.

On the other hand, in order to keep the software scalable, reusability is a very important issue [4-6]. Reusability [5, 7] is the likelihood that a segment of source code that can be used again to add new functionalities with slight or no modification. At the same time intelligent (AI) strategy based control [8-13] is now a trend as process system is becoming complex which is difficult to cope with conventional control strategy that requires accurate mathematical model. Many of them use neural network based control strategy [14-18]. Distributed implementation is also necessary to make the software available across several machines[19].

Based on these, our goal in this research is to develop an innovative and advanced software tool which will provide advanced control strategies (AI based) in a user friendly, GUI (Graphical User Interface) based and cost effective manner. In the later section of the literature review the evolution of software technology in connection to process control industry and current state is described. The design and implementation section explains various design issues related to this software and their implementation. A case study involving online control using this developed software is presented at the end.

In the later section of this paper, the background study of various software and related topics (involving neural networks) followed by the design and implementation of the proposed architecture is discussed. Conclusion is given in the last section which is preceded by a case study of real-time control on a neutralization plant.

2. Literature Review

During the period when personal computers (PCs) were not adopted in industrial automatic control system, all the irregularities that occurred in the industrial process system were checked manually and monitored by experienced or trained operator [1]. At that time all operations were handled manually or in a semiautomatic way. Due to the manual operation, abnormal conditions could not be monitored or handled in real time manner. Measurements and calculations for correcting any abnormality done by human operator were prone to mistake. Therefore it was highly necessary to automate the process operations such as monitoring and control to improve the operating efficiency. The advances in microelectronics, and the increasing speed of

microprocessors during the last four decades, have led to the development of miniature hardware components such as the programmable logic controllers (PLC) [20]. There are many vendors producing PLC based instruments such as Allen-Bradley, Siemens, and General Electric. This PLC approach offers an easy way to integrate hardware with high quality components. However there are some disadvantages as well with the PLC approach, that can be summarized as follows [21]:

- PLCs are typically a propriety system. Thus selecting a particular vendor and PLC family result in customers being locked only to the vendor's compatible corresponding line of products.
- Particular vendor specific product line is specific in nature, which is not easily customizable.
- Programming the PLC is traditionally limited to ladder logic, though later on the cross-development from PC has been supported. However PLC based system is generally based on PLC architecture, which is inherently closed in nature.

For fixed control strategies industrial process control application using PLC served well. However the industrial automation system is becoming more complex and control strategies changes to cope with system dynamics. Due to the increasing complexity of the process system, new sensors are being used [20]. Sometime the new sensors are difficult to interface with the traditional PLC hardware. New communication interface, and data acquisition functionality make these new sensors not always compatible to interact with the PLCs [22]. Due to the above mentioned reasons, the PLCs are being phased away in some industrial automation applications [21]. Instead of PLC, system designers are moving towards a second solution path, the PC based automation. Cheap availability of computing resources like memory and CPU (Central Processing Unit) has lead the industrial engineers to select the PC based control [23]. PC based control technology now offers reliability and functionality with capabilities that allow substantial productivity gains [24]. This alternative offers the standard development platform, various software frameworks, highly object oriented programming languages and development environments which are capable of building user friendly GUI (Graphical User Interface) and good quality control system software. Advances in both computing hardware resource and software technology enable the control engineers to encompass complex control strategies in software [25]. Due to the increasing complexity in the process system, limitations of classical methods for real-time control become apparent. According to [26], control system software need to be distributed (components can be in different computers), intelligent (capability to cope with ill posed behavior autonomously), integrated, open (non proprietary), and heterogeneous (ability to execute on a variety of computational platforms).

Now it is expected to have all the above aspects in control system software [26] in order to cope with the increasing complexity of control systems and software systems themselves. Complexity in control's implementation algorithms necessitates their design at a higher level of abstraction rather than the procedural code [25].

Numerous commercial software tools have been developed recently in this regard to provide advanced control strategies through user friendly ways. The most common analysis and design tool is from Mathworks with the product software name Matlab. Mathworks facilitates control engineers to design a control system by developing Simulink package, which is used in conjunction with Matlab to provide a block diagram based GUI [27]. For the implementation (on-line operation) purpose, the Real-Time-Workshop toolbox is required along with Matlab[28]. Moreover, despite the strong toolboxes and highly capable computing, the Mathworks generated code is not component-based rather it is monolithic (not modularized). Due to this fact, it is difficult to test or update the code. Along with this, Matlab code is not always open in nature, i.e. third party software components are not easy to incorporate with Matlab code. LabView is another software development environment [29]. It has smart GUI and hardware interfacing facility along with advanced control strategies. It has its own hardware interfacing devices. But it requires that the LabVIEW run-time engine be installed to use it, which makes it restricted, in use. MATRIXx is another software which incorporates control system design and analysis [22]. It provides interactive simulation facility and it requires code generation and compilation which requires expertise and not for the average user. Paragon software is good in data acquisition and storage but it only has

conventional control strategy [30]. LABTECH software supports open standard I/O hardware and data acquisition in real-time manner [31] but it also lacks advanced control strategies. JOVIAL [8], RTL/2 [32], Pearl [9] and Ada [5] are in the line of high-level language platform in this mission of easing the control software development process and make it cost effective. Though the inception of these programming systems certainly benefited control engineers, the effects have not been up to expectation [25]. One of the main reasons behind this is the dependencies to communicate between the various self modules. OCCAM is an exception at this point which effectively deals with communication and concurrency issues [33]. Though for control system of limited size, OCCAM has proven to be successful but for large scale systems its capability is not proven. Moreover none of them provide heterogeneous, distributed and open solution. It was not possible before due to the technological limits then, but today technologies are advanced enough to bring all the expected properties together as in this work. In Table 1 some of the leading software used in process industries are presented along with their functionality:

Table 1: Data acquisition and control software review

Software	Functionality	Limitations
Matlab (Simulink)	Matlab provides advanced control. Online control can be achieved by real-time workshop (one of its modules).	It is relatively expensive, proprietary product which requires exper knowledge to use. Matlab code is not component based so it is difficult to write scalable product.
LabView	LabView has advanced control solution with nice GUI (Graphical User Interface).	It also is relatively expensive and requires their proprietary hardware to connect to system. It only supports limited hardware communication interface.
GENESIS	It's a good modularized software and user friendly. Provides conventional control facility.	Absence of advanced control strategy. It also is relatively costly.
Paragon	Good in data acquisition and storage. Built in facility to exchange with third-party software.	Provides only conventional control.
LABTECH	Good in real-time data acquisition and supports open standard I/O hardware. Provides better data visualization.	Only have conventional control strategy only.
PITOPS Plus	PID Tuning Simulator/Optimizer, Dynamics Identification, Advanced Control, Model-Based Control, Feedforward, Cascade Control, DCS functions.	Only used for simulation not applicable for online control.
APROMON	Online Control Quality Monitoring Software.	Provides online control using conventional strategy not advanced control.
Process Control CBT	Practical Process Control and APC Training designed for Control Room	It's for training and simulation purposes not for online implementation.
PICONECT	Real-Time data collection	Only provides data acquisition facility not control
PILOGGER	High Speed Data Monitoring	Control is not its function but rather visualization for monitoring.
iTools	Process data logging, monitoring, setpoint program, Mini-SCADA	Set point changing and monitoring but no control strategy in it.

Table 1 (*Continued*)

Opto 22 FactoryFloor	It's Opto22 hardware based software, which has user friendly user interface for process monitoring and data logging.	Even though it can provide PID based online control but it cannot provide advanced control strategies.
MATRIXx	It is basically a design and analysis tool for control systems. It provides complex system simulation. Used by various industrial giant such as Ford, General Motors and Kaiser Aluminum.	It can be implemented for real-time application but requires other software modules to convert it to C code, which is tedious and not user friendly.
OCCAM	It is a concurrent programming language [33] which helps to build interactive application.	In larger scale application its capability is not proven.

The software mentioned in the table show that most of the software are for data acquisition, data logging and monitoring. Some of them provide control strategies but only for simulation not for online control. Online control strategies are also limited to basic control strategy implementations such as the PID control strategy. Moreover many of the software mentioned above are somehow dependent on other proprietary hardware or software components. Due to these reasons we have developed a software architecture that will provide an easy to apply advanced control strategies for online implementation, as described in the next few sections. The new architecture provides a scalable, reusable component based distributed software development framework, which is innovative in nature and a major contribution of this work.

3. Software Design and Implementation

Designing is an important part of software development. It involves the decomposition of a system into its constituent parts [34]. A good design is necessary to the successful implementation and evolution of a system. Before detail designing of the software system, the requirements need to be specified. Ideally, most of the requirements for control system software can be obtained from the role of software in modern control systems, as described by [25]. As per normal requirements, software for control system needs to be able to handle data acquisition from the system being controlled through sensing devices and also for tracking discrepancy between actual and the desired state and applying the corrective action after the necessary calculation. Notifying the user with system information is also the job of modern control software. Today's control system is complex and getting more so and control strategies are subject to changes anytime [8]. So the software developed should be capable of handling future modifications in control strategies, while keeping its performance acceptable. In order to do this, the software system should be designed in a highly modular fashion. Efficient use of exciting technologies and their up-to-date features enable us to build good control software design. Furthermore most software systems for control are needed to be distributed, integrated, open, and heterogeneous [26]. All these criteria for modern control system are incorporated in our software framework accordingly. To make our software distributed and modular, we separate the whole system into three parts:

- Main GUI (Graphical User Interface) module
- CIGM (Control Input Generator Module)
- Hardware module

This modularization is shown in Fig.1

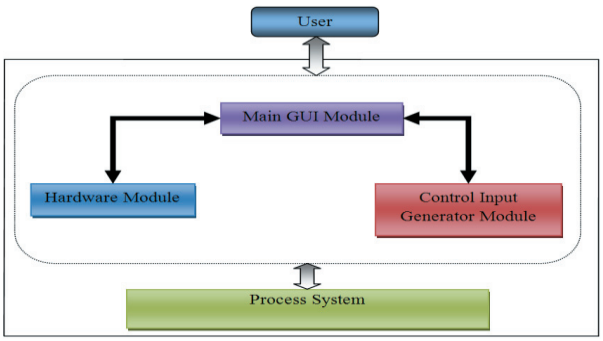

Fig. 1: Generalized Software Structure

The Main GUI module is the interface between the user and the software. It's the top level module of the software system. It displays the process parameters in various forms such as plots, charts, and percentages. The GUI module receives user commands and passes it to the appropriate modules of the software. The hardware module is the part of the software which interacts with the process system and maintains communication. It sends the control input generated from the software to the process system and read from the process system for necessary process parameters of the CIGM with updated values and pass it to the appropriate software modules. The control input generator module is the heart of this software framework. This module is responsible for providing the control strategy and proper control input to the process determined by the control strategy set by the user. All the modules are very loosely coupled to each other, which is the main criterion of object oriented programming [35], that helps to build the software system to achieve easy achievement, and with adaptable features.

We use the MVC (Model View Controller) architectural pattern for the Main GUI implementation, while multithreading is used for separate purposes. Hardware communication, control input generation and displaying the results are all handled by separate threads of execution to make the software interactive. Thread execution and thread interaction are shown in Fig.2:

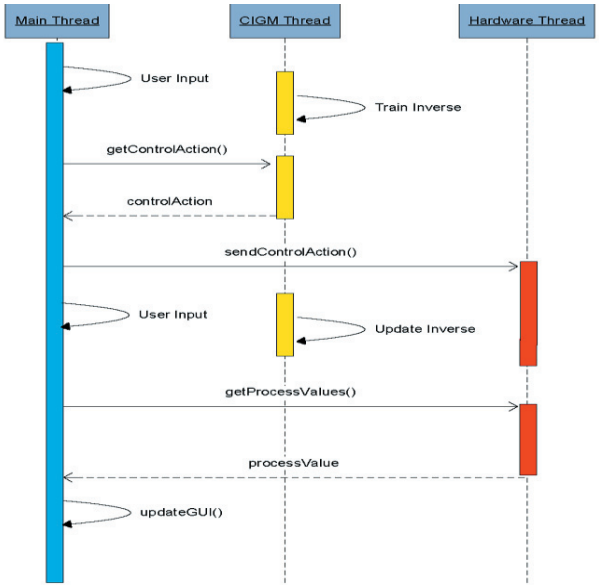

Fig.2: Threads execution and interaction

Distributed architecture is achieved through webservice implementation. Control input generator is deployed in a separate machine and used through the service. As a case study implementation we implemented the software interface for pH (Neutralization) control system as shown in Fig.15 for online control. In MVC implementation there is a model class which holds all process related data which is filled and changed by the controller class. All views are updated by fetching data from the model class. Interactions among the model, view and controller are shown in Fig.3:

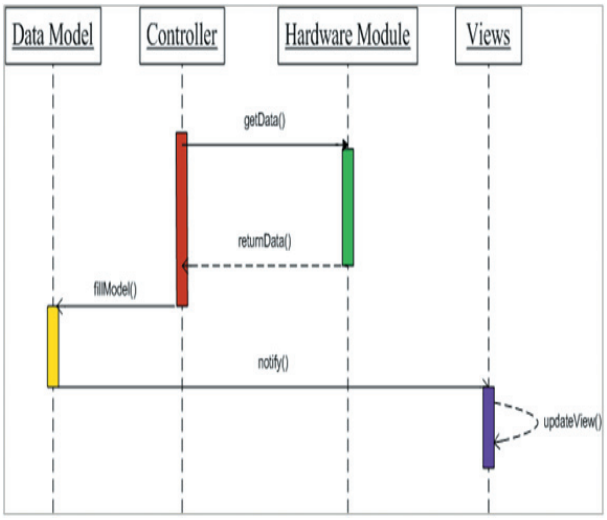

Fig.3: Model, Views and Controller interactions

Due to Webservice implementation we don't need to make several installations for the control input generator module (CIGM) which is under webservice and can be accessed from other machines running the GUI module. That result into controlling several plants without having proportional hardware resources which will reduce cost.

Hardware module is the part of the software for maintaining signal communications. It is used to maintain communication between the process model class and the process system. In our software framework we develop it as a separate module. Our purpose is to support virtually all I/O (Input/output) card available or to come in the future. In this regard we designed it in a very simplified manner. Most of the I/O card comes with the driver software along with the legacy API (Application Programming Interface). In order to support legacy API and make it expandable we built managed-DLL (Dynamic Link Library) component for

every I/O card of our interest. We implement a wrapper class to contain all the particular I/O card specific settings, while communicating with the external interface only through some general functions. For this purpose we declare an interface class, while all the wrapper class of I/O card's legacy code implements this interface as shown in the following code:

```
Interface HardwareModule
        {
                Signal getSignal();
                void sendSignal(Signal signal);
        }
```

During the execution of online process control, data flow from one module to another. All the data flow can be shown as in the following DFD as of Fig.4. The DFD is a graphical representation of the "flow" of data through the software and the process system.

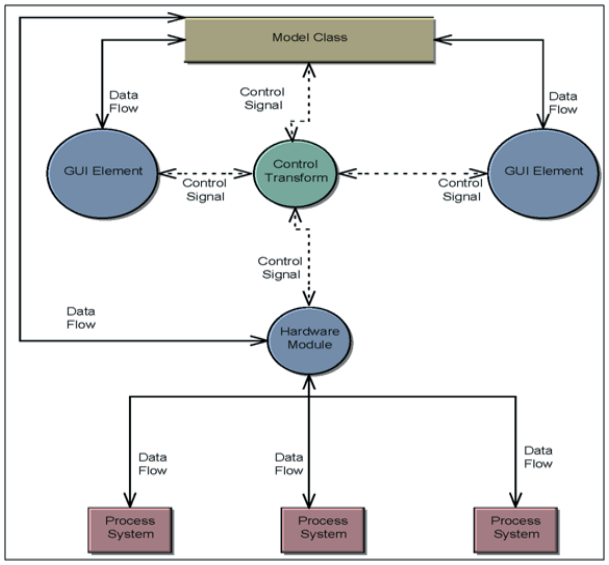

Fig.4: DFD (Data Flow Diagram) of the software architecture

During the visualization of process data shown in Fig. 12, 14, 15, data stored in the model class flows to those GUI elements, while the control action data is sent to the control valve hardware by the control unit module of the software. Control transform module shown in the DFD diagram above contains both the controller class of MVC and control action generation functionality. All the implemented control strategy is incorporated in this "control transform" module. Either PID or neural network based controller action is generated from this module and passed to process system hardware (in this case, control valve).

4. Neural network tool implementation

The neural network tool is an important part of the development of advanced control strategies in this software. It consists of user interfaces which help the user in training the process data to get the trained neural network. Preprocessing for training the data, such as loading, scaling of the data can be performed by using this tool together with testing and validation of the model. The trained neural network is saved and can be used later on from the software. As per our goal we used the open loop online data for training the inverse of the process, and then used it as the controller to implement the neural network model based DIC (Direct Inverse Control) strategy. When neural networks originally were proposed for controlling unknown non-linear systems, one of the first methods being reported was on training a network to act as the inverse of the system and use this as a controller [36]. Training in this case includes mapping of the neural network to the required model by optimization of the various neural network parameters. The system to be controlled can be described by:

$$y(t+1) = g[y(t), \dots y(t-n+1), u(t), \dots u(t-m)]$$

Where y(t) is the output of the process at time t, u(t) represents the control input feed to the system at time t. The desired network is then the one that isolates the most recent control input, u(t), i.e.

$$u(t) = g^{-1}[y(t+1), y(t), \dots y(t-n+1), u(t-1), \dots u(t-m)]$$

Where g^{-1} represents the inverse of the process. The DIC structure can be depicted as in Fig.5.

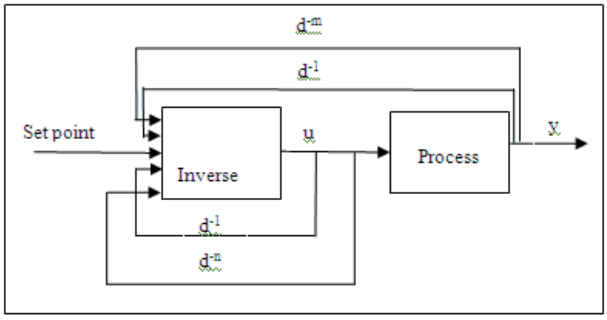

Fig.5: DIC Structure

Here in the following sections the NNTool (Neural Network Tool) is described with its GUI. NNTool is used for modeling the processing using neural network algorithm.

A case study is presented in the later section with online control result. In order to train the neural network model to learn the system dynamics, i.e., system inverse in this case, we need to load the open loop system dynamics data shown in Fig.11(a, b, c, d). Data can be loaded into the NNTool system by going to the 'Training Data' menu. Validation or testing data also can be loaded in the same way, as shown in Fig.6. Once the user clicks on the Training Data/ Validation Data/ Testing Data menu under the Load menu, the window as shown in Fig.6 comes up to user to select by browsing the local file system and choose the appropriate data for the corresponding parameter.

Fig.6: Data Loading GUI

After loading data it can be scaled by using the following the GUI as shown in Fig 7.

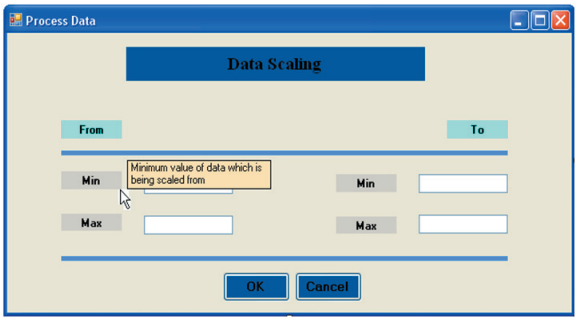

Fig.7: Data Scaling GUI

Before training we need to select the input and output pattern of the neural network being trained. Selection of input output can be done using the GUI as given in Fig.8.

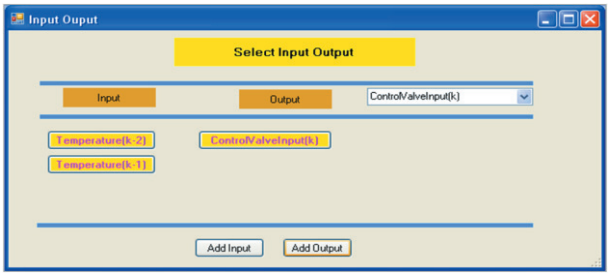

Fig.8: Input/Output specifier GUI

Once the data is loaded and scaled and the input/output pattern chosen, it is ready for training and from the software the training routine can be invoked by going to the 'Training' menu. Training window GUI is shown in Fig.9.

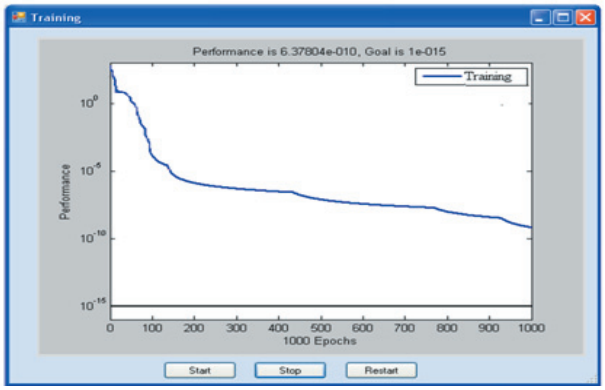

Fig.9: Training GUI

After training it is ready to be used as controller as part of the DIC based control strategy. In the next section, a case study is presented by controlling the pH (Neutralization) process using this strategy, which has been implemented in the software.

5. Case Study (pH-Neutralization control system)

In this case study to justify the applicability of our software, we control the neutralization process system for control of its pH. The pH system is described followed by control result such as set point tracking of process variable (pH) and manipulated variable (Base flow in this case).

5.1. *pH System Description*

The control of pH in neutralization plant is one of the challenging processes in the process industry due to its severe nonlinearities. It plays an important role in neutralizing excess reagents acidic by-products to obtain

high gain of expected products [10]. In this work we have controlled pH in a CSTR (Continuous stirred-tank reactor) acting as the neutralization plant. The acidity of the leaving solution is measured by a digital pH meter. We control the base flow rate to control the pH while keeping the acid flow at constant rate with a valve opening of 50%. The output of the pH meter is used by the control software's data acquisition part to generate the appropriate control signal (valve opening for base flow) to track the set point. The CSTR mainly has two inputs (acid flow and base flow line) and one output. In Fig.10 the schematic diagram of the pH plant is shown.

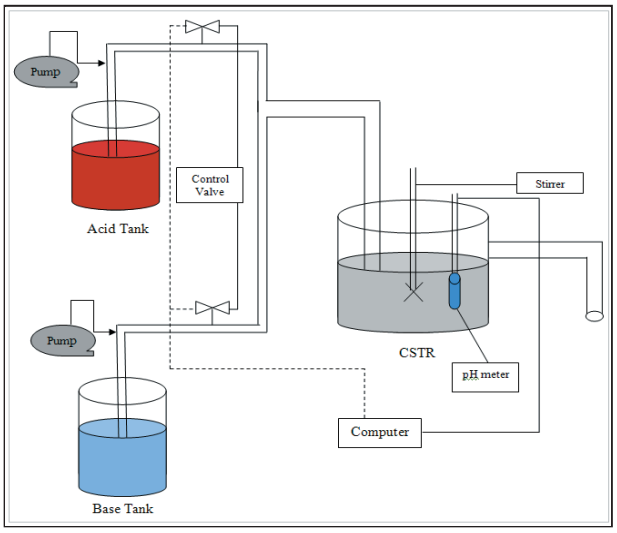

Fig.10: pH Plant with CSTR

The acid and base flow from two separate tanks come through these input lines. The output line is used to pass out the dilution to keep the process as a continuous process. The acid tank and base tank have a volume of 200 liters each and we used 0.1M HCl and 0.1M NaOH as acid and base respectively. Two separate pumps and control valves are attached through the flow line from each of acid and base tank to the CSTR. Computer software takes the signal from pH meter and send appropriate control signal to the base flow control valve. Pump for the acid and base flow-stream can supply from 0 to 10 liter/min depending on the control valve opening attached to the flow-stream line.

5.2. *Data acquisition and training*

Here in this example we used the pH process data for showing the NNTool training procedure. Multiple datasets are obtained by giving arbitrary excitation to the process system which are used for training, validation and testing. In Fig.11 (a, b, c, d) all the data sets are shown. Each data set contains different number of data points, as shown in figure 11 (a, b, c, d) they have 400, 240, 200, and 350 data points respectively. Both the base flow and pH data values are scaled from 0 to 1 for simplification. The real range of base flow is 0 to 12 liters/min and pH is 4 to 9. We trained our inverse neural network model with pH values within 4 to 9 only, as our interest was to operate around the neutralization point. Three datasets were used for training and testing while another was used for validation

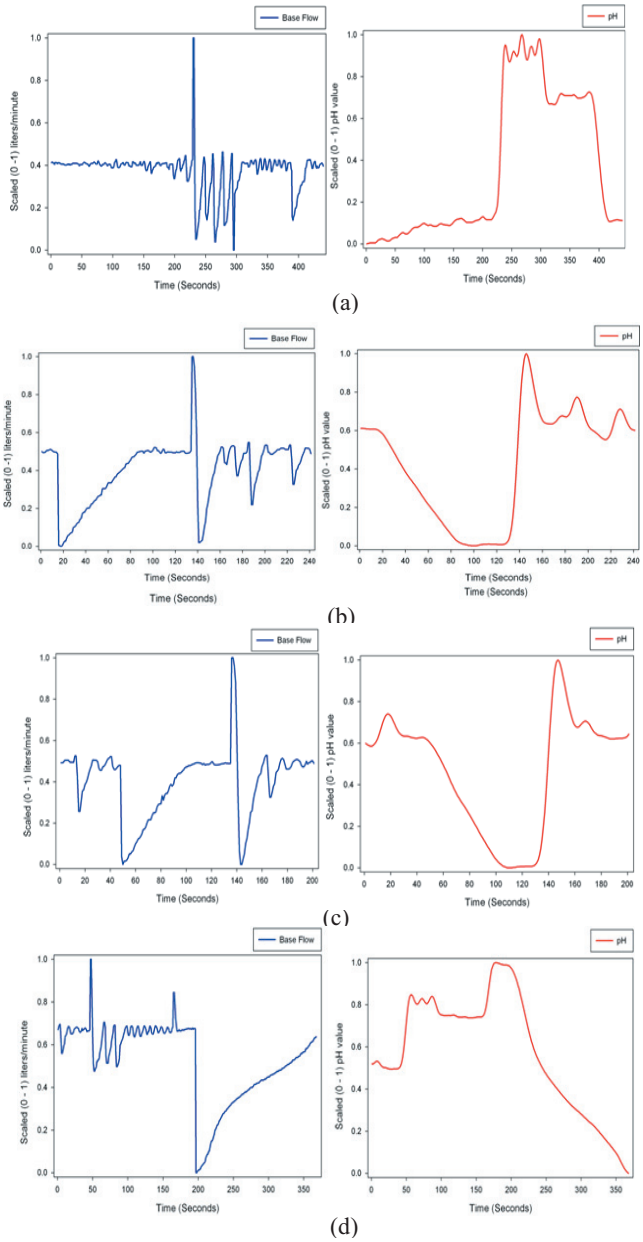

Fig.11: Training Data Set

Numerous variations of input-output configuration were tried to get the best neural network to represent the inverse model of process. In this case, five delayed inputs of pH measurement and two delayed input of the base flow reading gave the best training result. Table 2 presents the training results with various configurations of the neural network. Where MSE (Mean Square Error) measured by the formula

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(O_i - net_i)^2 ----(1)$$

Where, O_i and net_i is the actual output and trained net output respectively for the i^{th} training set within the n total sets.

Table 2: Training result with various input-output delays (pH Process)

pH Delay	Base Flow Delay	Neurons	MSE
3	1	6	0.2286
3	2	8	0.4998
4	1	10	0.1995
5	1	12	0.0523
5	2	12	0.0082
5	2	14	0.0341
6	1	10	0.1907
6	2	14	0.3045
7	1	10	0.2151

After getting acceptable training result the software can be used for testing and validation for the trained network as shown in Fig.3. It requires the necessary data to be loaded beforehand. In this case we get the following testing and validation result shown in Fig.12.

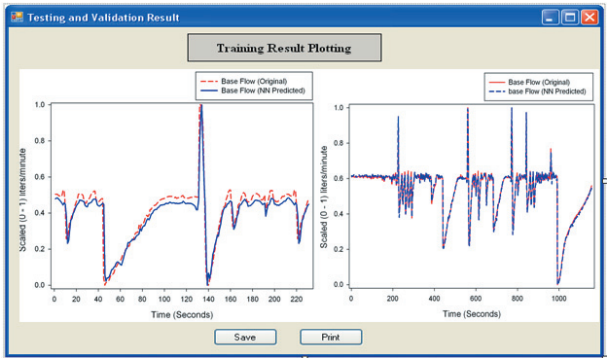

Figure 12: Testing and validation result

5.3. *Control and Set Point Tracking Results (pH Neutralization System*

Once the training, testing and validation are done for the process inverse model, the trained network is ready to be applied as direct inverse controller (DIC) for online control of the pH control system. Software graphical user interface for pH control system is shown in Fig.13

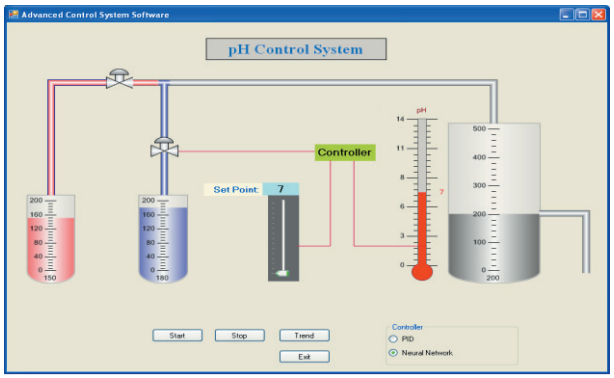

Fig.13: Main software GUI for pH control system

User can load a pre-trained network if a trained network already exists. We used the previously trained network as the inverse controller. In the following Fig.14, online control and set point tracking results for pH control system are shown.

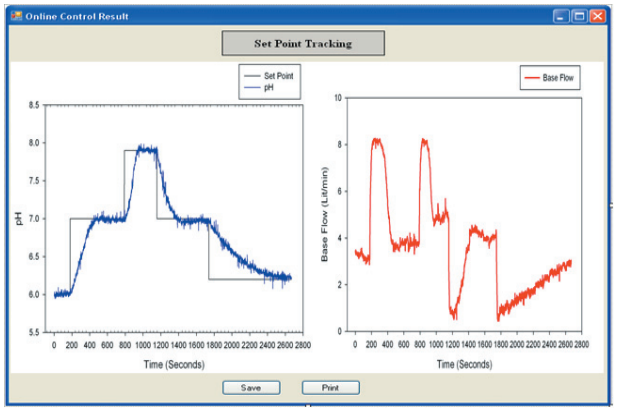

Figure 14: Set point tracking of pH neutralization process (Neural Network Controller)

Fig.14 shows both the set point tracking and the corresponding base flow rate which is being controlled by the software. Base flow change helps the pH control system to track the set point. All the set points between 6 and 8 are covered in this run. Neutralization point, i.e., pH 7, is important in process industries and is difficult

to keep the process at this value. Here the neural network based DIC controller shows convincing control result. The developed software also facilitates the conventional PID control strategy, where the PID action is governed by the following equation [32, 37]:

$$U(t) = K_p e(t) + K_i \int_0^t e(t)dt + K_d \frac{d}{dt} e(t) \text{-- (2)}$$

K_p, K_i, K_d are controller tuning parameters for getting optimal control. The online control result by using the PID control is shown in Fig.15 as comparison. The result for PID shows oscillatory behavior which is not desirable for control of such a system in practice.

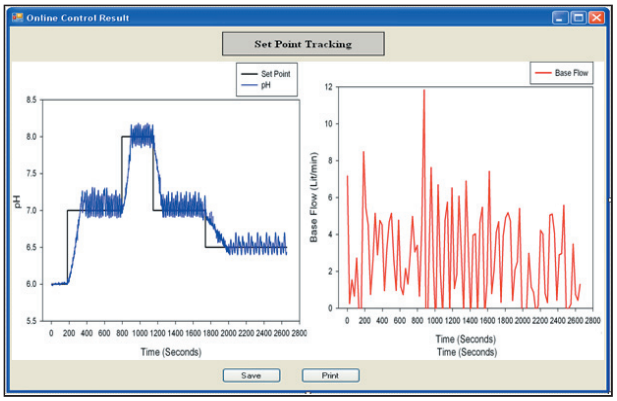

Figure 15: Set point tracking of pH neutralization process (PID Controller)

Apart from visually comparable performance of the conventional PID and neural network based controller, numerical comparison is given as below:

Table 2: Numerical comparison of PID (Conventional) and neural network based control performance

Strategy / Measure	PID (Conventional)	Neural Network Based
MSE	0.005287	0.00352
ISE	284.10	189.20
IAE	541.52	432.49

MSE: Mean square error
ISE: Integral of the square of the error
IAE: Integral of the absolute value of error
Mathematical formula of different error measures[38] are given as below

$$ISE = \int_0^\infty e^2 dt \text{ - - - - - - - - - - -(3)}$$

$$IAE = \int_0^\infty |e| dt \text{ - - - - - - - - - - -(4)}$$

All the error measures given in Table 2, shows better performance by the neural network based controller. Even though it seems the errors are close in value, but the fact is in real applications once the setpoint is achieved it doesn't change over a long time. Here in our real-time experiment we implemented with various setpoint values and changed when it settled for a while. The oscillations when using the PID controller contributes larger error value while neural network controller gives less oscillation.

6. Conclusion

In this research, a user friendly software for control system utilizing advanced control strategy (artificial neural network) is developed. Evolution of the different software system is presented and requirement and expectation for the modern technology based software for control system is explained. We developed this software framework to provide an easy-to-apply, user friendly but advanced control strategy for process industries. It can be used as a general purpose software where neural network based control is expected to be applied.

As mentioned in the design section that MVC pattern is followed and CIGM is installed as a webservice. The MVC architectural pattern allows the future development to be independent in different sections of the Model, View or controller without hampering or depending on another component, while the webservice implementation gives the software a distributed character.

In the current architecture the hardware communication is handled through the developed dll library which is not modifiable. In case of supporting a new hardware we need to write new dll for that hardware module and have

to be put in certain locations which is not user friendly. This is a limitation of this architecture. The hardware module can be specified through certain properties in an XML file so that normal user can just put the specifications of the new hardware module and the software system should be able to generate necessary dll for that module. That will make the software more user friendly. Also adding new control strategies will definitely enhance the software.

Real-time Control results and implementation of latest software development technology which ensures the architectural scalability and reusability, supports the architecture and development strongly.

Acknowledgement

Authors are grateful to UMPEDAC (University Malaya Power Energy Dedicated Advanced Center) and MOSTI (Ministry of Science, Technology & Innovation, Malaysia) for providing fund for this work.

References

1. Lingfeng, W. and C.T. Kay, *Modern Industrial Automation Software Design*. 2006.
2. NASA Tech Briefs. . *Industrial Automation Software: Simplified Solutions for Complex Processes*. 2004; Available from: http://findarticles.com/p/articles/mi_qa3957/is_200409/ai_n9437482/.
3. Auslander, D.M., Ridgely, J.R., Ringgenberg, J.D.,, *Control Software for Mechanical Systems: Object-Oriented Design in a Real-Time World* 2002: Prentice Hall PTR.
4. Xiaojun, L., et al., *Heterogeneous Modeling and Design of Control Systems*, in *IEEE Control Systems Magazine*. 2003.
5. Speck, A., *Reusable industrial control systems.* Industrial Electronics, IEEE Transactions on, 2003. **50**(3): p. 412-418.
6. Yunfei, L. and H. Wei. *A software design method in the field of industrial control*. in *Sensing Technology, 2008. ICST 2008. 3rd International Conference on*. 2008.
7. Matsumoto, Y. *Experiences from software reuse in industrial process control applications*. in *Software Reusability, 1993. Proceedings Advances in Software Reuse., Selected Papers from the Second International Workshop on*. 1993.
8. Zengxian, G., et al. *Research and application of intelligent control software for flotation reagent process*. in *Intelligent Control and Automation (WCICA), 2010 8th World Congress on*. 2010.
9. Yingjun, G., C. Dongfang, and Z. Guang. *Application of Intelligent Control Techniques for Temperature-Humidity Control in Industrial Workshops*. in *Intelligent Information Technology and Security Informatics, 2009. IITSI '09. Second International Symposium on*. 2009.
10. Bharathi, N., J. Shanmugam, and T.R. Rangaswamy, *Intelligent Control of pH in a Neutralization Process* Asian Journal of Information Technology, 2007. **6**(6): p. 7.
11. Antnio, F. and C. Manuel, *Intelligent Control and Integration Software for Flexible Manufacturing Cells.* Industrial Informatics, IEEE Transactions on, 2007. **3**(1): p. 3-11.
12. Chuanying, Y. and H. Dexian. *Research and Application of Advanced Control Software Platform based on iFIX*. in *Intelligent Control and Automation, 2006. WCICA 2006. The Sixth World Congress on*. 2006.
13. Baysal, C.V. and A.M. Erkmen, *An intelligent inference system for robot hand optimal grasp preshaping.* International Journal of Computational Intelligence Systems, 2010. **3**(5): p. 656-673.
14. Mevawalla, Z.N., G.S. May, and M.W. Kiehlbauch. *Neural networks for advanced process control*. in *Advanced Semiconductor Manufacturing Conference (ASMC), 2010 IEEE/SEMI*. 2010.
15. Taek-Beom, K., et al. *Development of integrated process control system utilizing neural network for plasma etching*. in *Electronics Manufacturing Technology Symposium, 1995. 'Manufacturing Technologies - Present and Future', Seventeenth IEEE/CPMT International*. 1995.
16. Pappa, N. and J. Shanmugam. *Neural network based predictor for control of cascaded thermal process*. in *Intelligent Sensing and Information Processing, 2004. Proceedings of International Conference on*. 2004.
17. Mevawalla, Z.N., G.S. May, and M.W. Kiehlbauch, *Neural Network Modeling for Advanced Process Control Using Production Data.* Semiconductor Manufacturing, IEEE Transactions on, 2011. **24**(2): p. 182-189.
18. Kasparian, V. and C. Batur. *Neural Network Structure for Process Control using Direct and Inverse Process Model*. in *American Control Conference, 1992*. 1992.
19. Yih Ping, L., C. Shean-Shyong, and C. Jau-Woie. *Design of distributed control system software using client-server architecture*. in *Industrial Technology, 1996. (ICIT '96), Proceedings of The IEEE International Conference on*. 1996.
20. Cecil, L.S., *Digital control of industrial processes.* ACM Comput. Surv., 1970. **2**(3): p. 211-241.
21. Borges, J. *PC Vs. PLC for Machine and Process Control*. 1997; Available from: www.realtime-info.be/magazine/97Q4/1997q4_p071.pdf.
22. National Instrument. *MATRIXx Document: User Guide*. 2007; Available from: http://www.ni.com/pdf/manuals/370769c.pdf.

23. Hanselmann, H., *Implementation of digital controllers--A survey.* Automatica, 1987. **23**(1): p. 7-32.
24. Gee, D. *The how's and why's of PC based control*. in *Pulp and Paper Industry Technical Conference, 2001 Conference Record of.* 2001.
25. Boasson, M., *Control systems software.* Automatic Control, IEEE Transactions on, 1993. **38**(7): p. 1094-1106.
26. Sanz, R. and K.E. Arzen, *Trends in software and control.* Control Systems Magazine, IEEE, 2003. **23**(3): p. 12-15.
27. Matworks. *Simulink® 7 Getting Started Guide*. 2011; Available from: http://www.mathworks.com/help/pdf_doc/simulink/sl_gs.pdf.
28. Matworks. *Real-Time Workshop User's Guide (pdf)*. 2011; Available from: http://www.mathworks.com/help/pdf_doc/rtw/rtw_ug.pdf.
29. National Instrument. *LabVIEW™ User Manual*. 2011; Available from: http://www.ni.com/pdf/manuals/320999e.pdf.
30. Nematron. *Paragon - Powerful and flexible HMI/ SCADA*. 2011; Available from: http://www.nematron.com/products/legacy/paragon.html.
31. LABTECH. *LABTECH Overview*. 2011; Available from: http://www.labtechsoftware.com/LabTech_feature_sheet.pdf.
32. Yun, L., A. Kiam Heong, and G.C.Y. Chong, *PID control system analysis and design.* Control Systems, IEEE, 2006. **26**(1): p. 32-41.
33. SGS-THOMSON Microelectronics, *OCCAM 2.1 reference manual*. 1995.
34. Hans van, V., *Software Engineering: Principles and Practice*. 2008: Wiley Publishing. 740.
35. Bernd, B. and A.D. Allen, *Object-Oriented Software Engineering; Conquering Complex and Changing Systems*. 1999: Prentice Hall PTR. 553.
36. Hussain, M.A. and L.S. Kershenbaum, *Implementation of an Inverse-Model-Based Control Strategy Using Neural Networks on a Partially Simulated Exothermic Reactor.* Chemical Engineering Research and Design, 2000. **78**(2): p. 299-311.
37. Hu, Y. and D. Li. *Tuning of PID controller for fractional order systems*. in *Control Conference (CCC), 2011 30th Chinese*. 2011.
38. Saji, K.S. and K.M. Sasi. *A novel controller tuning of pH neutralization process*. in *Signal Processing, Communication, Computing and Networking Technologies (ICSCCN), 2011 International Conference on*. 2011.