A virtual laboratory exercise for teaching development of industrial computer applications

George Hassapis, Niovi Pavlidou
Department. of Electrical and Computer Engineering, Aristotle University of Thessaloniki
541 24 Thessaloniki, Greece
chasapis@eng.auth.gr, niovi@eng.auth.gr

Abstract: This paper addresses the issue of teaching the development of industrial control applications by the use of industrial computers. These computing devices have a modular hardware architecture able to receive a varying number of interfaces for sampling signals from industrial-type sensors and updating signals which drive a varying number of actuators. Also they run under the control of real-time operating systems. The development involves the elaboration of the hardware and the software architectures, the code generation and the software testing. Teaching is done through the realization of a process control system by using a professional suite of software tools.

1. INTRODUCTION

The wide spread introduction of the industrial computers [1,2] into process plants has posed new requirements on the control education. These requirements call for combining skills in hardware configuration and software development of these specialized devices with those in traditional control system analysis and design.

As various educators believe [3] most of the curricula in control systems theory and engineering cover quite adequately the analysis and design aspects but they do not expose the student to the broader and practical issues of a complete control system. In order to bring control systems education a step closer to real life, curricula must be enhanced with work that deals with the implementation of a control system application which involves the use of an industrial computer.

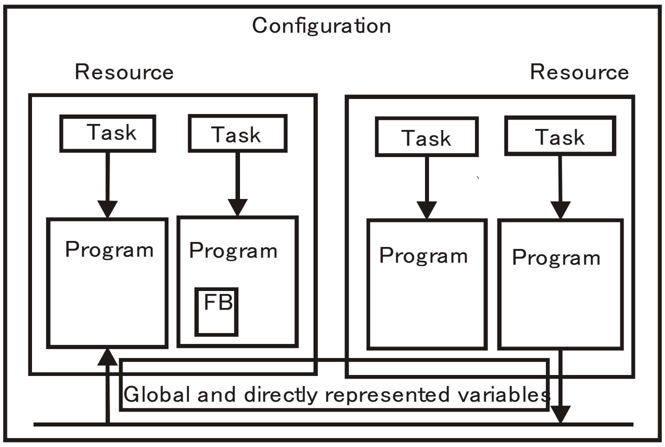
The work presented in this paper proposes an exercise which addresses all the practical issues encountered in a real-life application of an industrial computer and a way of teaching all the phases of such a development. It is based on the use of the IsaGraf [4] software engineering workbench, that is a suite of tools that the professionals are using for developing applications for this class of computing devices. This work may be considered complementary to that presented in Ref. [5] to [8] which involves the use of professional process simulators and CACSD(Computer-Aided-Control-Systems-Design) software for training students on control systems engineering.

2. ARCHITECTURE AND PROGRAMMING OF A SOFT LOGIC COMPUTER

Industrial computers are characterized by a modular hardware architecture consisting of a small number of processing boards and a large number of input and output modules. Processing boards and input/output modules are placed on a number of frames communicating between each other and distributed in the industrial complex. Each frame is equipped with a back plane bus and slots, which the various modules are plugged in. Input modules are of various types, each type being able to sample either analogue or discrete signals. The analogue signals emanate from devices measuring physical quantities such as temperature, level, flow and the like. The discrete signals are coming from devices identifying logical states of physical variables, such as the lowest or highest level of liquid in a vessel. Output modules provide the analogue or discrete signals for adjusting controlled variables to desired values.

Depending on its type, an input or output module may receive or supply a specific number of similar signals from the sensing or to the actuating devices. The circuitry of each module that is used to receive or transmit each one of these signals is usually called a channel of the module. Typical modules are available for 8, 16 or 32 channels of discrete signals or 2 to 8 channels of analogue signals.

A real-time operating system manages the use of the computer resources, schedules the execution of the entities of the application software and provides the routines for sampling and updating the input and output modules. The sampling is taking place by assigning symbolic addresses to each channel of each module. It also provides the environment for executing the application software according to the model recommended in the IEC1131-3 standard [9]. This model defines the way software should be structured and run and defines its main entities. It also specifies the syntax and semantics of a unified suite of programming languages that can realize in software these entities. A schematic presentation of the model is illustrated in Fig. 1.



FB: Function Block

Figure 1: Software model

According to this model, the software for a particular control problem is characterized by its architecture, which consists of resources, tasks and programs. A resource is the operating system abstract view of the computer platform, a task is a scheduling facility that executes either periodically or in response to a state change of a particular Boolean variable and a program is the largest software entity that can be declared at the resource level. A resource is characterized by its configuration, which includes the definition of the type of the processing boards, the placement in the frame of input/output modules and the assignment of symbolic addresses to the channels of the input/output modules. A program contains input, output, internal variable declarations, a body of data processing instructions and a class of software entities that are called function blocks. A function block allows a specified algorithm or set of actions to be applied to a given set of data in order to produce a new set of output data. Control over the execution of different programs and function blocks is achieved by assigning them to different tasks.

3. APPLICATION DEVELOPMENT METHODOLOGY

On the basis of the above model, one can write down a list of actions that the development of software for an industrial computer requires. These actions may be summarized in the following list.

1. Configuration of the hardware platform
2. Architectural description of the application software
3. Coding the software entities of the application software
4. Testing the application software.

What the configuration of the hardware involves has been explained in the previous section. Once this action is completed the architecture of the software has to be described. This can be done by the constructs of a language defined in the IEC 1131-3 standard and involves practically the description of the software decomposition to tasks, programs and function blocks. Also, it involves the assignment of the program and function block execution to the defined tasks and the declaration of the common data objects, that is the data structures and the variables that can be accessed by any program or function block of the application software.

The coding of the software entities involves the writing of the source code of each program, function block and task of the software. Source code writing for each individual entity can be done in any of the languages supported by the IEC 1131-3 standard. The last step in this work involves the compilation and correction of syntactic errors of all the coded entities and the library routines that may have been included in these entities.

Testing the software is the work of verifying that the execution of the developed software will perform the logical functions that the designer has conceived. Simulation of the software execution on a host computer is a well-established technique for a first level testing.

The efficient implementation of all the above-mentioned methodological phases of the application development would require the availability of supporting software tools. IsaGraf [4] is a suite of such tools, developed and build to support the IEC 1131-3 recommendations. One of these tools is the so-called project manager, which supports the implementation of the configuration of the platform and the description of the software architecture. A number of editors is the other set of tools, which allow the writing of the source code of the individual entities of the software in different languages. A multi-language compiler, a debugger and a target system simulator are included in this suite of tools.

In the following the use of the IsaGraf suite for teaching the development of an application of an industrial computer through the study of a specific process control problem is presented.

4. THE PROCESS CONTROL PROBLEM

In Fig. 2 the schematic diagram of a chemical process with its control instrumentation is shown. The process involves the formation of a liquid solution with specific pH value by mixing water with hydrochloric acid solution.

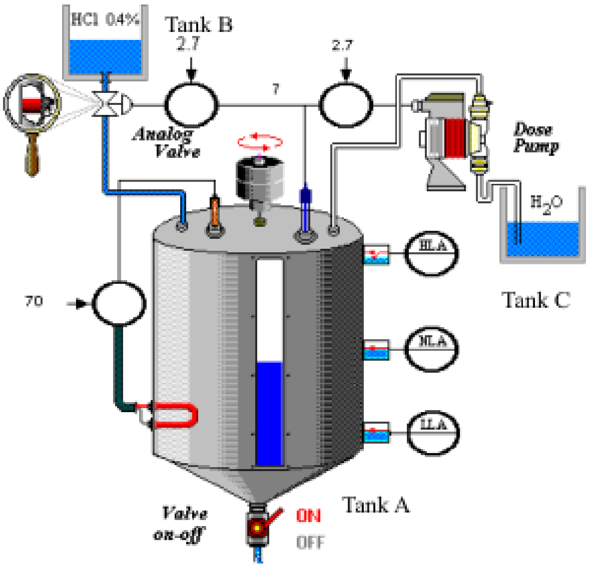


Figure 2: The chemical process of the pH adjustment of a liquid solution.

Initially, water is fed into tank A at a quantity equal to half of the tank volume. Tank B contains a hydrochloric acid solution with a concentration equal to 0.4% of the overall liquid volume. Tank C is filled with water. It is required to form a homogeneous solution, which will have a pH value of 2.7 at the temperature of 70 ^{0}C. The process of forming the homogeneous solution involves the adjustment of the volume of the hydrochloric solution and the water that is poured into tank A from tanks B and C and the heating and stirring the mixture. A mixer located at the top of tank A stirs the mixture. An electric heater, located at the bottom of tank A is used for heating the mixture.

The process operation must be controlled by a system that performs the following functions.

- control of the flow rate of the hydrochloric acid solution with the purpose of increasing the pH value,
- control of the flow rate of the water from tank C when the pH value must be decreased,
- control of the solution temperature in tank A in parallel with the control of the pH value,
- control of the stirring process of the solution in tank A,
- identification and alarm annunciation of abnormal operating conditions, such as tank A flooding or liquid solution level being below a minimum,
- monitoring the flow rates of the hydrochloric acid solution and the water, the pH value and the solution temperature by the use of bar graphs and analogue value indicators,

- raising alarms for the conditions of the low liquid level (LLA) and the high liquid level (HLA) in tank A,
- bypassing the automatic mode of operation and controlling manually the flows at the outlets of tanks A, B and C and the temperature in tank A.

The control loops, the alarms, the monitoring of variables and the manual controls are realized in the following way:

- For implementing the control of the liquid flow from tank B a loop is formed by inserting a pH measurement electrode in tank A, by placing an analogue valve with a linear characteristic curve at the outlet of tank B and by controlling its position through the use of a PI (Proportional plus Integral) algorithm.
- For the control of the water flow rate from tank C a dosimetric pump with 2-lit/pulse-flow rate is placed at the outlet of tank C.
- For draining the contents of tank A an on/off solenoid valve is connected to the bottom outlet of tank A.
- A J-type thermocouple is assumed to monitor the temperature changes of the solution in tank A. A heater is switched on by a silicon-controlled rectifier every time the temperature drops below the desired value and is switched off when the temperature exceeds the desired value.
- Two level switches for identifying the lowest and highest level of the liquid solution in tank A have been placed at the positions of tank A marked with the LLA and HLA symbols respectively.
- Four On/Off switches are used to set the mode of operation of the following actuators from manual to automatic and vice versa:
  - the analogue valve that controls the flow of the hydrochloric solution,
  - the dosimetric pump that controls the flow of water from tank C,
  - the solenoid valve at the bottom of tank A and
  - the heater.

5. THE TEACHING APPROACH TO THE APPLICATION DEVELOPMENT

In the considered teaching approach the student is guided to implement the process control functions by applying the methodology for the software development that was presented in section 3. For the implementation of the control functions the student is provided with semi-tailored solutions, which is required to complete and test.

5.1 Hardware configuration

For the phase of the computer configuration, a virtual computer frame is given and the student is asked to select from a library of input and output modules those needed by the application and insert them to the appropriate slots of the frame. The selection and insertion can be done by simple drug-and-draw actions of the mouse in the environment of the project management tool of the IsaGraf suite. The library contains simulated versions of input and output modules with 8, 16 and 32 digital channels, modules with 2 analogue channels and modules for displaying and inserting alphanumeric data with up to 32 characters. For the simulation of the modules a C–code program was prepared which defines a data structure with the number of the channels of the virtual module and the type of data that each channel may accept, i.e. Boolean, integer, short or long real number, message or time type of data.

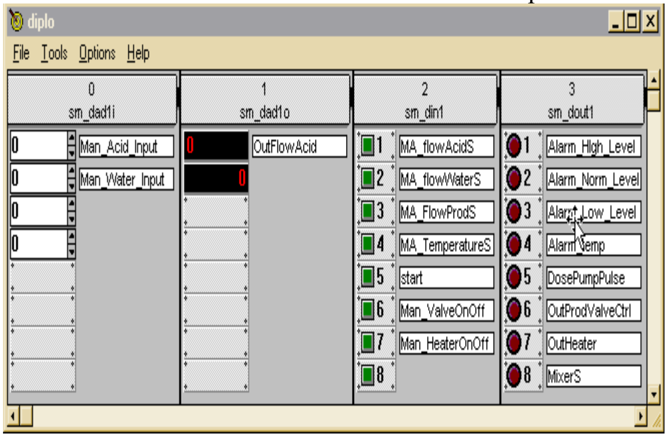
Referring to the specific application of the pH control, one easily observes in Fig. 2 that the soft logic computer must be able to receive signals from two analogue sensors, the pH electrode and the temperature thermocouple. Also, it must be able to receive discrete signals from:

- the three level switches in tank A,
- the switches that will select the manual or automatic mode of operation of the control loops of the hydrochloric acid and water flows, the temperature in tank A, and the flow of the product from the bottom of the tank A,
- the start/stop switch of the process.

It should provide one analogue output for the control of the analogue valve and eight discrete outputs for:

- the control of the dosimetric pump,
- the heater,
- the solenoid valve at the bottom of the tank,
- the temperature alarm,
- the alarm indicator of a low solution level (LLA),
- the alarm indicator of a high solution level (HLA),
- the indicator of the normal level condition (NLA).

Also, for the display of the flows of the hydrochloric acid solution and the water, for the display of the temperature and the pH values the appropriate alphanumeric display modules must be included. For inserting preset values for the manual control of the analogue and dosimetric pump and the set point value of the PI algorithm a module of alphanumeric data entry must be selected. The student at this phase is expected to select the proper modules from the library of the available modules and insert them to the appropriate slots of the frame. This action must be combined later with the action of assigning names of variables to the channels of each module, which takes place during the phase of defining the software architecture. The assigned variables are related with the received and sent signals from and to the process. This combined action is expected to result to a frame filled in the way shown in Fig. 3. In this figure, the tags shown next to each channel of the module 'sm_din1' correspond to the names of the variables used in the programs to denote values of signals emanating from the switches that define the manual or automatic mode of operation of the various actuators. The variable names 'MA flowAsidS', 'MA_FlowWaterS', MA_FlowProductS', MA_TemperatureS' 'Man_ValveOnOff' and 'Man_HeaterOnOff' refer respectively to the switches that set in manual or automatic mode of operation the



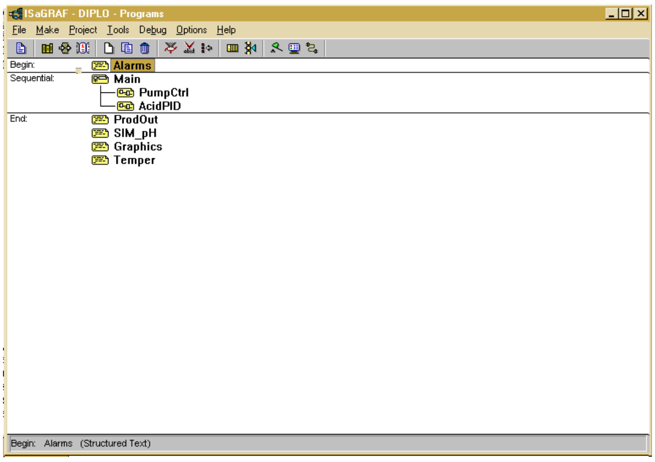
0 sm_dad1i		1 sm_dad1o		2 sm_din1		3 sm_dout1	
0	Man_Acid_Input	0	OutFlowAcid	1	MA_flowAcidS	1	Alarm_High_Level
0	Man_Water_Input	0		2	MA_flowWaterS	2	Alarm_Norm_Level
0				3	MA_FlowProdS	3	Alarm_Low_Level
0				4	MA_TemperatureS	4	Alarm_Temp
				5	start	5	DosePumpPulse
				6	Man_ValveOnOff	6	OutProdValveCtrl
				7	Man_HeaterOnOff	7	OutHeater
				8		8	MixerS

Figure 3: Window for configuring the hardware of the industrial computer.

analogue valve, the dosimetric pump, the temperature controller, the solenoid valve and the heater respectively. The 'start' variable denotes the state of the start/stop switch, which initiates and interrupts the whole cycle of the pH adjustment. The module 'sm_dout1' provides the eight discrete outputs of the computer. The names of the variables 'DosePumpPulse', 'OutProdValve', 'OutHeater', 'MixerS' shown in this module, refer respectively to the signals produced for the control of the dosimetric pump, the solenoid valve, the heater, the mixer and for trigerring alarm indicators. When the temperature exceeds a limit the output with the variable name 'Alarm temp' is triggered. When a high or low or normal level of solution is monitored in tank A the outputs with the variable names 'Alarm High Level', 'Alarm Low Level', and 'Alarm Norm level' are activated. The module 'sm_dad1o' provide a continuous display of the dynamic flow changes of the hydrochloric acid (OutFlowAcid), the flow of the water (OutFlowWater), the pH value (pH) and the temperature in the tank (Temperature). The module 'sm_dad1i' allows the entry of the set point value in the PI control loop of the analogue valve (Set point) and the preset values for the manual control of the position of the analogue valve (Man_Acid_Input) and the dosimetric pump (Man_Water_Input).

5.2 Software Architecture

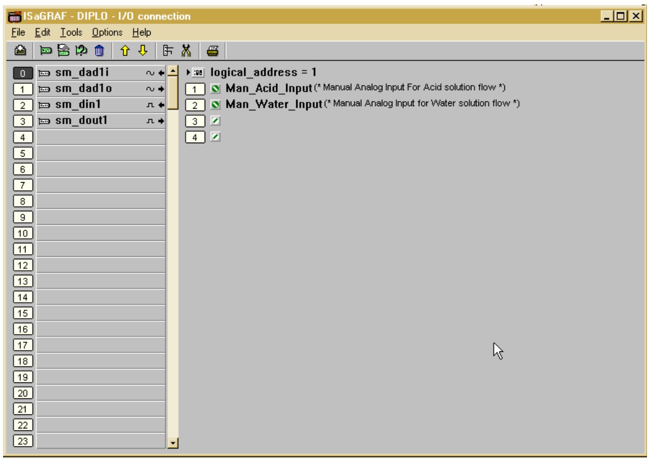
In this phase the student is provided with a recommended software architecture, which is expected to read and understand by using the project management tool of the IsaGraf suite. The tool translates the provided data to the source code of the architecture description language that is recommended in the IEC 1131-3 standard. The window in Figure 4 shows how the architecture is defined by the use of the project management tool. Based on what was said in section 3, the architecture considers that the computer platform has a single resource, which runs three tasks, the Begin, Sequential and End tasks. The control and monitoring functions have been decomposed to a number of programs, some of them being independent and some having the parent-child relation. In Table 1 the control or monitoring functions performed by each program are described. The next step of this phase concerns the definition of the common objects, that is, the various types of variables that are visible by all the tasks and programs of the software. Again, the student is required to invoke the lists of these variables and identify them. The last step involves the assignment of variable names to the channels of the input and output modules of the computer. This is done through the graphical presentation of the addresses of the channels and the writing in the field that is next to the address of each channel the symbolic name of the appropriate variable. As an example, in Fig. 5 the assignment of the variable names to the channels of the discrete input card 'sm_din1' is shown.



Begin: Alarms
Sequential: Main
PumpCtrl
AcidPID
End: ProdOut
SIM_pH
Graphics
Temper

Begin: Alarms (Structured Text)

Figure 4: Graphical description of the software architecture.



0 sm_dad1i
1 sm_dad1o
2 sm_din1
3 sm_dout1

logical_address = 1
1 Man_Acid_Input (* Manual Analog Input For Acid solution flow *)
2 Man_Water_Input (* Manual Analog Input for Water solution flow *)
3
4

Figure 5: Assignment of variable names to the addresses of the channels of the input and output modules.

For the selected module, denoted by the slot number in black in the left side of the diagram, a list of numbered channels is provided on the right side of the diagram. The student may inscribe next to each channel number the name of the variable that he wishes to assign to this channel. At this point the description of the software architecture is concluded and the student is advised to proceed with the writing of the source code of the programs that are listed in Table 1. This can be done by calling the language editor, which each program is going to be written to, from the project management tool. As an option, instead of writing the code the student may obtain the full source code of the programs with comments, which can study and understand the functions they perform. The completion of this phase is followed by the compilation phase in which all the written codes for the architecture description and the programs are converted to executable code.

5.3 Control System Testing

Once the syntactically correct executable code is produced, the phase of testing the logic of the software can be initiated. Simulation of the code execution is an option offered by the suite of tools. In this option the developed software runs on a host computer under user defined scenarios of input values. The simulator on the host computer imitates the way application software is executed on a target industrial computer and provides either a pre-programmed graphical display of the monitored variables and controls of the process operation or allows the use of a custom-made display. Through this display a software execution scenario can be set up and the output results can be monitored either by running the software on an instruction-by-instruction basis or by inserting breakpoints in the code and run sections of the software. These two modes of software execution allow the student to identify programs and sections of code, which do not behave the expected way and modify them accordingly. The student in this phase of software development can obtain the full source code of the programs with comments, which can study and understand the functions they perform. He is also provided with the display shown in Figure 6 and is asked to test the execution of the software.

Table 1: Description of the functions performed by the programs of the software

Program Name	Program Type	Description
Alarms	Parent	LLA and HLA check and annunciation
Main	Parent	Co-ordinates the call of the AcidPID and PumpCtrl subprograms
AcidPID	Child	Implements the PI control of the flow of the hydrochloric acid solution to tank A
PumpCtrl	Child	Implements the flow control of the water from tank C to tank A.
ProdOut	Parent	Controls the solenoid valve at the bottom of tank A
SIM_pH	Parent	pH computation - simulation of the pH electrode operation
Graphics	Parent	Imports the graphical presentation of the bar graph and pictures of the virtual instruments, prepared by the Easy Icons and IconMaster packages. Updates the dynamic changes of the values of the measured quantities.
Temper	Parent	Temperature computation simulation of the temperature sensor operation

Although the student can work out any testing scenario, a recommended one is also given according to which the student can act as a typical plant operator. First, he has to initiate the start-up procedure and verify that the developed system acts the expected way. Next, he has to run the process at its normal mode of operation and at the end he has to test the alarming functions For the start-up of the process the automatic/manual switches of the analogue valve, the heater, the dosimetric pump and the solenoid valve, corresponding to the variables 'MA_flowAcidS', 'MA_flowWaterS', 'MA_flowProdS', 'MA_TemperatureS' respectively (see Fig.6), are set to the manual mode of operation. Then, the analogue input 'Man_Water_Input' is loaded with the value of 300 strokes, which is a preset value for the dosimetric pump.

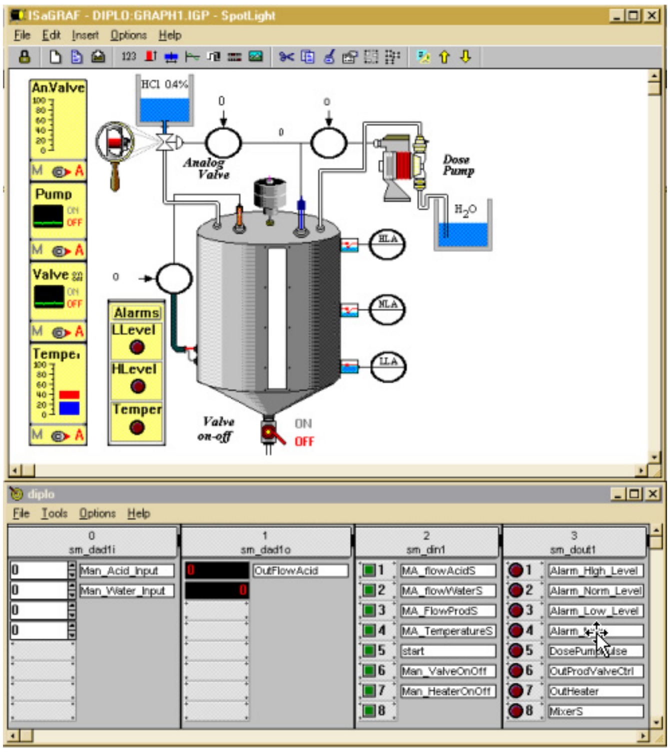


Figure 6: The display of the variables and controls of the process operation.

At this mode of operation, one expects to see the analogue and the solenoid valves closed, the heater off and the tank A to be filled with a volume of 600 lit of pure water from tank B. If the software runs correctly, by pressing the start push button the student can see the volume indicator of tank A to rise and when the NLA mark is reached the dosimetric pump to shut off and the pH indicator to display the value of 7. This is the pH value of the water. The next operator action would be to adjust manually the flow of the hydrochloric solution in order to form a solution with the desired pH value. This can be realized by loading the analogue input 'Man_Acid_Input' with the value of 50% of the analogue valve opening and observe how the pH value changes as liquid from tank C flows into tank A. When a pH value in the range of 2.5-2.7 is monitored all the switches are set from the manual to automatic mode of operation. The expected software behavior is to see after a short period of time the pH value of the solution to reach a value in the range of 2.68-2.72, the temperature of the solution to reach a value in the range of 69-71^{0}C and the mixer operation to start. The mixer operation continues until a command to drain the contents of tank A is given by the operator. This command can be given by setting the manual /automatic switch of the solenoid valve to the on position. During the stirring period the pH and temperature values continue to have values in the just mentioned ranges, as long as the level of the liquid solution in tank A is below the HLA mark. To test the operation of the alarming functions the start-up procedure can be repeated but at this time a large value greater than 600 lit, i.e. 650 lit, must be loaded in the 'Man_Water_Input'. One should expect to see the level of the liquid in tank A to reach a point above the HLA mark while the alarm indicator Hlevel flashes. After draining tank A below the mark LLA, one should expect to see the alarm indicator Llevel to flash.

By performing the tests that have been described or other tests that the student can think of, one may verify the correctness of the developed software. The reader may carry out the experiment described in this paper by accessing the site mentioned in Ref [10].

6. CONCLUSIONS

This work has presented an exercise for teaching the development methodology of an industrial computer application. The exercise concerned the start-up, normal mode of operation and alarm annunciation of abnormal operating conditions of a process that is used to adjust the pH of a chemical solution. The exercise through the use of a suite of software tools has demonstrated how one can implement the four phases of the methodology. The respective work involves the computer hardware configuration, the description of the software architecture, the writing of the code of the individual entities of the software and its testing by simulating its execution on a host computer.

REFERENCES

1. MotioNet.com Inc., Industrial Computer Directory., http://www.motionnet.com/cgibin/search.exe?a=cat&no=213, 2001
2. Crispin A. J., Programmable logic controllers and their engineering application. London: McGraw-Hill, (1997).
3. Antsaklis P., Başar T., DeCarlo R., McClamroch N. H., Spong M., and Yurkovich S., Report on the NSF/CSS Workshop on New Directions in Control Engineering Education. IEEE Contr. Syst. Mag., 19, 53 (1999),.
4. Isagraf, ISAGRAF IEC 1131-3 Soft Logic, User's Guide. Seyssins,France : CJ International, (1998).
5. Cooper D. and Dougherty D., Control Station:An Interactive Simulator for Process Control Education. International Journal of Engineering Education, 17, 276 (2001).
6. Shin D., Yoon E.S., Park S.J., Lee E.S., Web-based interactive virtual laboratory system for unit operations and process systems engineering education. Computers & Chemical Eng., 24, 1381 (2000),.
7. Blackley J.J., Irvine D.A., Teaching programmable logic controllers using multimedia-based courseware. International Journal of Electrical Engineering Education, 37, 305 (2000).
8. Copinga G.J.C., Verhaegen M.H.G., and Van de Ven M.J., Toward a Web-Based Study Support Environment for Teaching Automatic Control. IEEE Control Sys. Mag., 20, 8 (2000),.
9. IEC-1131-3, Programmable controllers-Part 3: Programming languages. Geneva: International Electrotechnical Commission, (1993).
10. Hassapis G., Industrial Informatics. http://indinf.ee.auth.gr/myweb, 2002)