ISBN: 978-628-95207-4-3. ISSN: 2414-6390. Digital Object Identifier: https://dx.doi.org/10.18687/LACCEI2023.1.1.1581

Improvements Implemented in guiEditor and its Use in Teaching of Process Control Systems

Eduardo J. Adam, Dr.[1], and E. Sergio Burgos, Mg.[2]
[1]Universidad Nacional del Litoral Facultad de Ingeniería Química, Argentina, eadam@fiq.unl.edu.ar
[2]Universidad Tecnológica Nacional Facultad Regional Paraná, Argentina, sergioburgos@frp.utn.edu.ar

Abstract– ***This paper presents an overview of the recent enhancements implemented in guiEditor, a graphical user interface (GUI) editor for GNU Octave. The focus is on the improved functionality and usability of the tool, designed to facilitate development of applications for teaching industrial process control systems. The potential of this tool for solving classroom problems and supporting laboratory experiences for students is demonstrated.***

The success achieved in the implementation of guiEditor is evidence to its efficacy and encourages further development of the tool. These promising results motivate future research to explore potential of guiEditor in facilitating the development of educational tools for industrial process control systems.

Keywords — didactic object, process control system, engineering education, GNU Octave, GUI Editor

I. Introduction

The use of software as a didactic tool in engineering education is widely accepted, particularly in simulating and designing complex systems using rigorous mathematical models ([1] and [2] among others). Proprietary software, such as Matlab from MathWorks or Wolfram Mathematica, among many others, are commonly utilized in undergraduate, postgraduate, and research activities across various engineering fields. However, the high cost of these software packages can be prohibitive for many academic institutions and students, necessitating the exploration of cost-effective alternatives.

While some companies offer economic solutions to this problem, such options may be limited in terms of functionality and scope. As a result, it is crucial to investigate open-source software tools that can provide a comparable level of functionality without the associated costs.

The academic community, motivated by the licensing costs associated with these proprietary applications, began to explore open source tools a few years ago. Particularly, in this work, emphasis will be made on GNU Octave [3] as a calculation software applied to topics related to Control Systems.

Upon commencing work with GNU Octave, a limitation that was encountered was the challenge of executing parameterized scripts from the console. In both academic and research contexts, interactive experimentation that allows for manipulation of operating conditions or parameters can significantly enhance the efficacy of investigations. This shortcoming served as the impetus for exploring alternatives that could facilitate the development of graphical user interfaces (GUIs) for process control systems, thereby enhancing teaching and research efforts in the field.

Digital Object Identifier: (only for full papers, inserted by LACCEI).
ISSN, ISBN: (to be inserted by LACCEI).
DO NOT REMOVE

Through this inquiry about open source software alternatives as a possibility to developed GUIs suitable for undergraduate and postgraduate educational tasks in the classroom and laboratory settings, as well as for research simulation endeavors. By enabling more interactive and user-friendly manipulation of parameters and variables, the implementation of GUIs in GNU Octave has the potential to significantly enhance the performance of process control systems, facilitating more efficient and effective teaching and research in this critical field.

With version 4 of GNU Octave, it became possible to represent visual controls incorporated in the Qt framework, an environment that was also used to develop its user interface. With this, the possibility of developing a graphical environment that allowed the construction of interactive graphical interfaces for GNU Octave as script files was observed. This is how guiEditor emerged, a visual development environment that has been published ([4] and [5]) as an open source application and that is currently available in its version 2 [6].

It is also important to note that guiEditor has been made available to the community through its source code on the GitLab repository, where binaries for MS Windows and Ubuntu Linux are also available. Successful tests have been carried out by building this application from its source code on a Raspberry PI 4 and using it in this architecture.

This paper aims to present some internal features of guiEditor together with a brief description of some new features implemented, their functionalities, and some cases in which this GUIs editor was applied for the development of applications for teaching control systems in engineering careers.

For this purpose, the present work is organized as indicated below. Section II comments on the general structure of the developed application. Then, Section III presents the new features implemented in the guiEditor in his version 2.0. Subsequently, in Section IV, applications developed for teaching process control systems are briefly introduced. Finally, Section V presents the conclusions.

II. Structure of an application

Projects developed using guiEditor are designed to be GNU Octave packages that are necessarily associated with a dialog window. Therefore, the project name will defines the name of the application in GNU Octave. When invoked, it display the main window of the application, allowing the user to interact with it.

To structure the development process, the representation of an application is stored using a directory structure and an XML file for each dialog window. These files contain both the source code to be executed for the controls and their attributes. The GUI editor generates the representation of each window and displays the associated source code according to the selected control. While each control has a single default event, it is possible to explicitly link more callbacks from the source code.

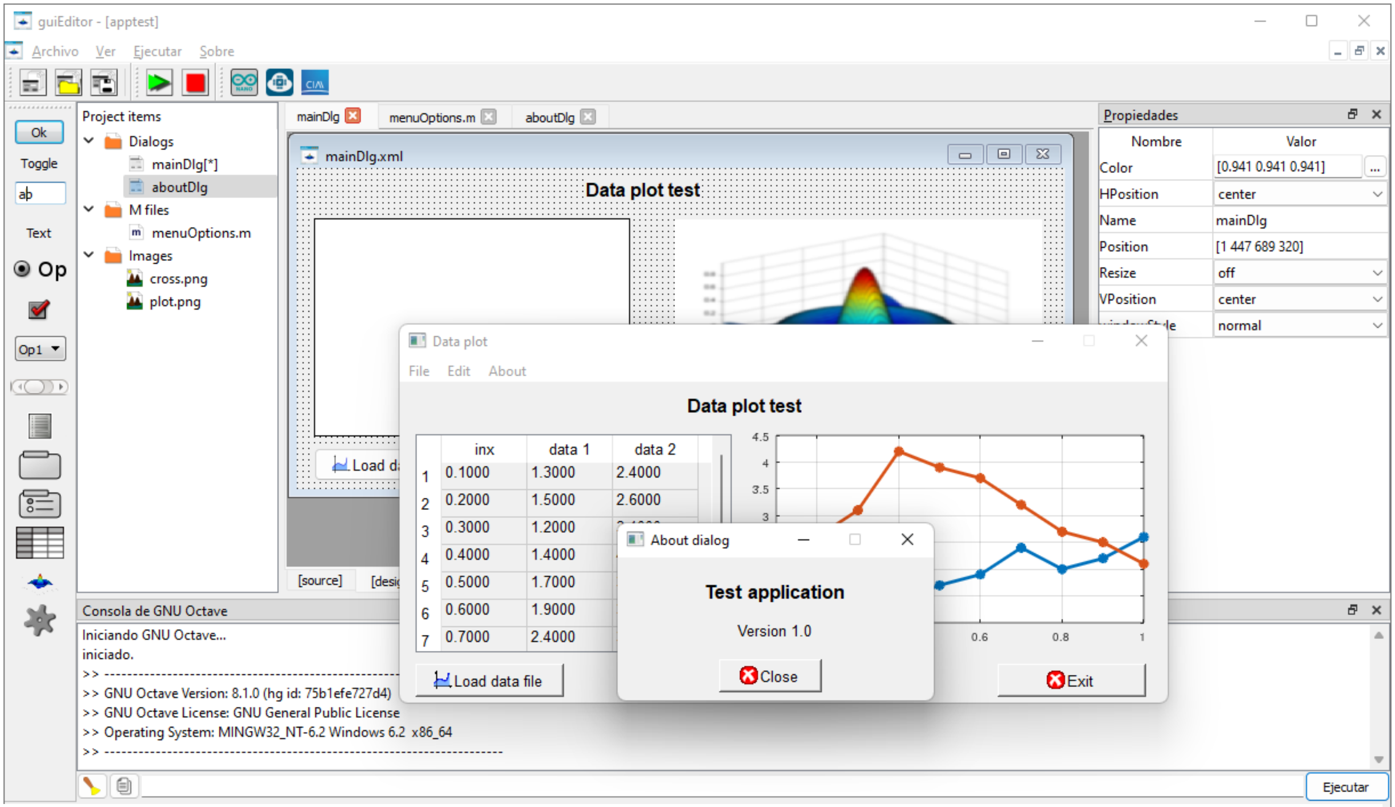
Controls are built in the script file using invocations to the uicontrol, uipanel, and uitable functions. To achieve a representation similar to that used in object-oriented programming, the identifiers of each control are stored in a structure with the same name as the dialog window to which they belong. This approach creates an effect akin to a composition/aggregation relationship. However, since this logic hides the figure on which the dialog window is built, it is useful to be able to reference it. Therefore, an additional field called 'figure' is added to store its identifier. With this, it is possible to modify its attributes or use it as an argument for other GNU Octave functions.

When working with packages, a common problem is the need to store information and/or have libraries incorporated into a project. To solve this, two alternatives are proposed. Firstly, files included in the 'lib' folder of a project will be incorporated into the search path when an application starts to execute. This makes it possible to use libraries transparently. Secondly, each project, when executed, creates a global variable containing the physical path from which the application is executed. With this information, it is possible to access other folders, libraries, or data files.

To facilitate interaction with external hardware, a control has been incorporated to communicate with Arduino development boards. This control depends on the “instrument-control” package of GNU Octave and allows programming of a default firmware on the development board, configuration of basic aspects such as the functionality of input/output pins, and communication with the application. Reprogramming of the board is accessible by double-clicking on the control, while other aspects can be accessed through the property inspector. The communication implementation has been made without a particular purpose, allowing for reading and writing of analog and digital ports, as well as controlling servo motors. In order to allow for the development of specific applications, the source code executed on the microcontroller is added to the project, so that the developer can modify it using the tools provided by the Arduino framework. The firmware structure implements a state machine to process a packet-based protocol, making it easy to extend its functionality by adding new messages.

III. New features implemented in the guiEditor

As the development of applications using guiEditor has progressed, possible improvements have been identified and implemented. The following are the most notable changes that



Nombre	Valor
Color	[0.941 0.941 0.941]
HPosition	center
Name	mainDlg
Position	[1 447 689 320]
Resize	off
VPosition	center
windowStyle	normal

	inx	data 1	data 2
1	0.1000	1.3000	2.4000
2	0.2000	1.5000	2.6000
3	0.3000	1.2000	
4	0.4000	1.4000	
5	0.5000	1.7000	
6	0.6000	1.9000	
7	0.7000	2.4000	

```
Iniciando GNU Octave...
iniciado.
>> ------------------------------------------------------------------
>> GNU Octave Version: 8.1.0 (hg id: 75b1efe727d4)
>> GNU Octave License: GNU General Public License
>> Operating System: MINGW32_NT-6.2 Windows 6.2  x86_64
>> ------------------------------------------------------------------
```

Fig. 1 Appearance of guiEditor version 2.0 during the execution of an application under development using GNU Octave version 8.1.0 and Microsoft Windows 11 as the operating system.

are visible to the user (Fig. 1):

1- The most noticeable change is the redesign of the application's graphical interface for simpler user interaction. Instead of working with a global project manager, each project is now managed from a window that contains it.
2- In addition, instead of working with one window per document, it is now proposed to use tabs to simplify access to the different files.
3- In the case of files related to dialog windows, they are now treated as a single unit, switching between the graphical representation and the associated source code through dedicated tabs.
4- Error localization has been improved. During the application development process, when executing it from the editor, if an error occurs, the location reported by GNU Octave may not correspond to the location of the code written by the developer due to the code added during the application generation process. To resolve this issue, a mapping between both files was incorporated, allowing errors to be located simply by double-clicking on the output console.
5- Accompanying the evolution of the functionalities incorporated by GNU Octave with respect to graphical controls, new features were added to allow for their graphical editing.

IV. Applications

A. *A toolbox for Teaching Control Systems (`LTITOOL`)*

To improve dictation and comprehension in the Instrumentation and Process Control course offered as part of the Chemical Engineering and Food Engineering undergraduate programs, a tool called `LTITOOL` was developed. This tool was created using interactive graphical interfaces developed using guiEditor [4].

The primary objective of developing `LTITOOL` was to enhance student understanding of the fundamentals of control theory through four fundamental topics related to the course, including:

1. Open -loop systems.
2. Closed-loop systems under unitary feedback.
3. Feedback systems, not necessarily unitary, against changes in load and set point.
4. Advanced control strategies. Specializing in feedforward and cascade control.

Besides, the toolbox includes additional tools, where although some of them are under development, one of them that allows the optimal adjustment of controllers stand out.

Figure 2 shows, by way of example and without going into further detail, the toolbox main window that shows the functionalities implemented is presented.

Figure 3 shows the main window of the application used for analysis and simulation studies of closed-loop systems with unitary feedback. The dynamic simulation inside and outside the main window, as well as a root locus plot, can be observed. Students can also access additional information related to stability, settling time, gain and phase margin of the closed-loop system, among other data. Moreover, all the developed applications have an additional window that provides further information through graphs that show closed-loop zero and pole maps, root locus, Bode, and Nyquist diagrams, as shown in Figure 4.

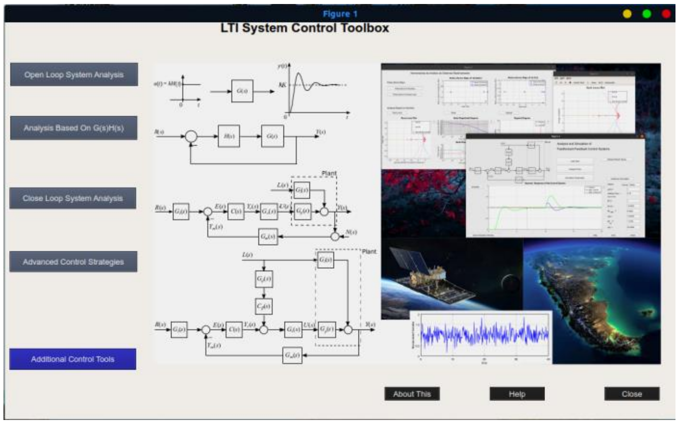


Figure 1
LTI System Control Toolbox
Open Loop System Analysis
Analysis Based On G(s)H(s)
Close Loop System Analysis
Advanced Control Strategies
Additional Control Tools
About This
Help
Close

Fig. 2 Ltitool toolbox main window

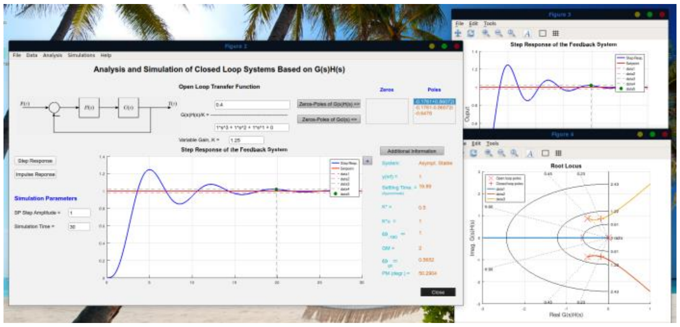

Analysis and Simulation of Closed Loop Systems Based on G(s)H(s)
Open Loop Transfer Function
Zeros
Poles
Step Response of the Feedback System
Step Response
Impulse Reponse
Simulation Parameters
Additional Information
Close
Root Locus

Fig. 3 Main window for analysis and simulation for closed-loop systems with unitary feedback.

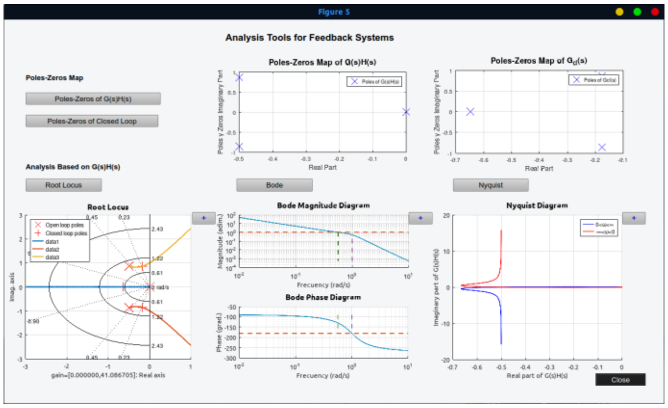


Fig. 4 Main window for analysis tools for closed-loop systems.

To evaluate the effectiveness of the developed toolbox, at the end of the course, we asked students to complete a survey about their experience with LTITOOL. Students responded on a rating scale and were given sufficient time (10 minutes) to complete their responses. The main questions asked are presented below, and the results obtained are summarized in Table I.

1. It was easy to understand the software management (Yes/No/Not much)
2. Rate the graphical interface developed (Poor 1 - Very Good 5)
3. The software was useful for understanding the topics taught in class (Not much 1 - Much 5)
4. Which of the following topics the software helps you understand them for? (Not much 1 - Much 5)
   (a) Open loop dynamics
   (b) Root locus
   (c) Frequency domain
   (d) Stability
   (e) Tuning PID controllers
   (f) Closed loop dynamic behaviour
   (g) Feedforward control
   (h) Cascade control

At this point it is important to note that the data presented in this work are broader than those presented in the work by [4], since here it is include 2 more years of data collection.

According to the results in Table I, it is observed that for the first question, most students find the developed software very easy to manipulate, which is a merit of the developed graphical environment. For the remaining questions, the results obtained in all questions were very good, exceeding the average value of 4 over a maximum value of 5.

In light of these results, it is clear that LTITOOL's development and implementation proved to be a successful approach to enhancing students' knowledge and understanding of the aforementioned topics. Future research may seek to extend the tool's capabilities and assess its effectiveness in different educational contexts.

TABLE I
SURVEY ABOUT LTITOOL APP

Qst.	Answer		
	Yes	No	Not Much
1	63	1	2

	Min				Max			
	1	2	3	4	5	Min	Max	Avg.
2	0	1	7	26	32	2	5	4.34
3	0	0	1	19	46	3	5	4.68
4(a)	0	1	12	20	33	2	5	4.29
4(b)	0	0	7	16	43	3	5	4.54
4(c)	1	4	12	17	32	1	5	4.14
4(d)	0	0	11	18	37	3	5	4.39
4(e)	1	3	15	24	23	1	4	3.98
4(f)	0	1	9	19	37	2	5	4.39
4(g)	3	9	13	19	22	1	5	3.73
4(h)	7	10	12	18	19	1	5	3.48

B. Remote experimental with a thermal system prototype for process control

In this work, we present a remote laboratory (RLab) development that allows for laboratory experiments to be carried out using GUIs with GNU Octave, based on the work of Adam et al. [7]. The authors utilized guiEditor to build a small SCADA system for remote experimentation with low investment costs for software and hardware.

Specifically, the laboratory experience developed consists of acquiring temperature data inside an electric furnace (see Fig. 5) and sending the remotely monitored data through low-cost instrumentation.

Particularly, the objectives sought with this experimental work are those listed below.

1 Monitor the furnace temperature. These data are recorded with a PT100 and sent to the computer for data acquisition in the laboratory through an ARDUINO UNO board.
2 Send the data to the server at the Telematics Center (TC) of the UNL.
3 Present the data on the computer monitor of the laboratory.
4 Record the data of the experience in a file with formats MAT or CVS.
5 Check the connection status between the computer inside the laboratory and the server at TC.

The RLab operates through an internal wired connection that links the Faculty of Chemical Engineering (FIQ) with the TC-UNL building. This connection facilitates the exchange of information between the laboratory and the TC, enabling both parties to send and receive data, as illustrated in Figure 6.

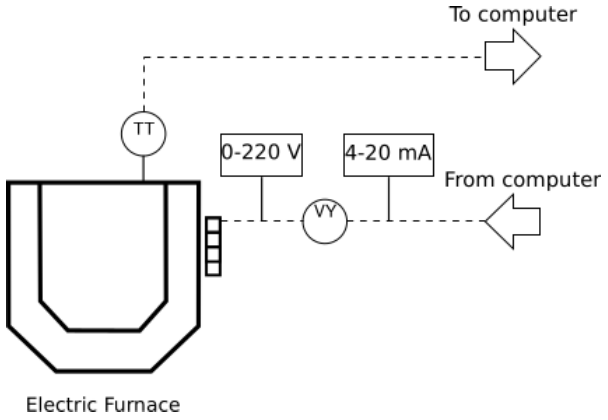


Fig. 5 Laboratory equipment for remote experiment

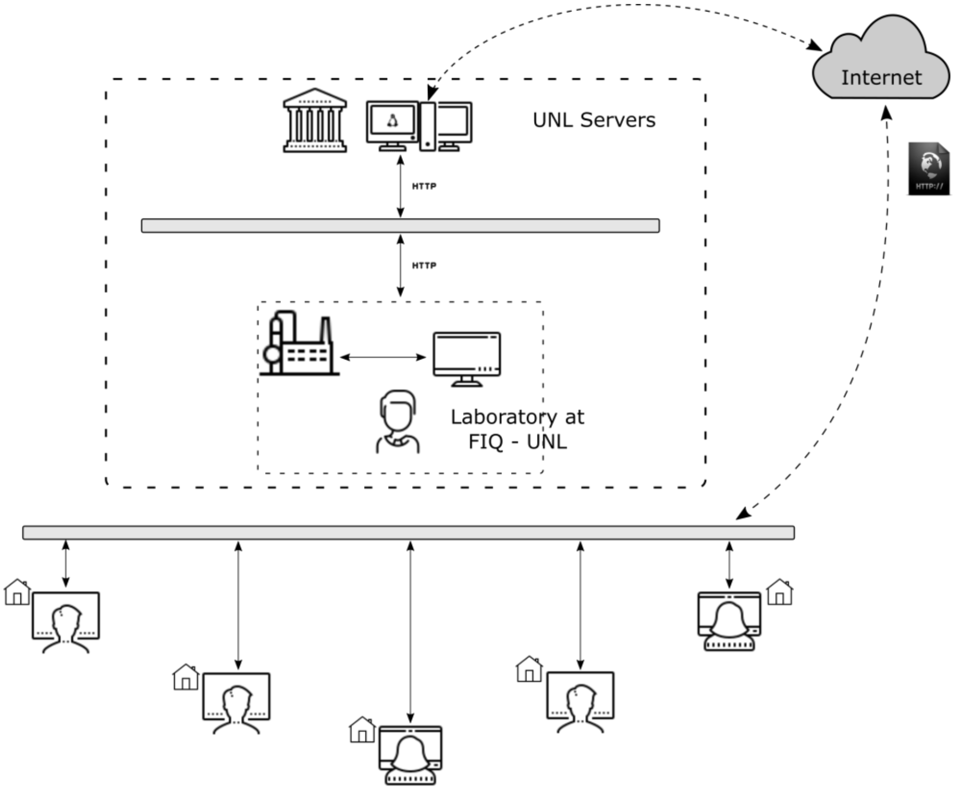
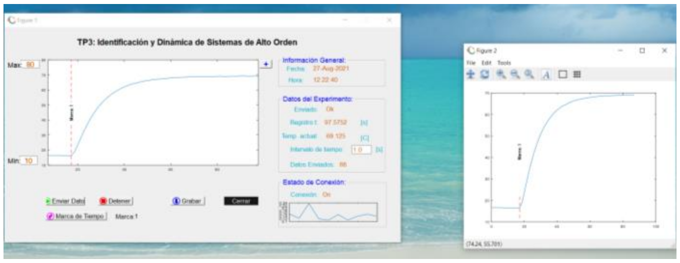


Fig. 6 Information management scheme from and to the RLab

Figure 7 illustrated the *master* application utilized by the teacher in the laboratory to transmit data to the server in the CT.

Figure 8 shows the client application used by students to receive remotely real-time data from the server in the CT.



TP3: Identificación y Dinámica de Sistemas de Alto Orden

Max: 80
Min: 10

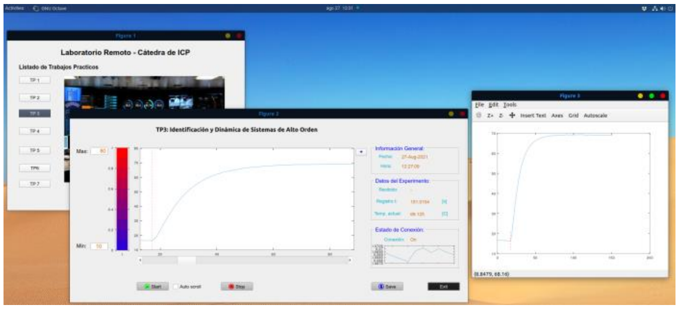
Información General:
Fecha: 27-Aug-2021
Hora: 12:22:40

Datos del Experimento:
Enviado: Ok
Registro t: 97.5752 [s]
Temp. actual: 69.125 [C]
Intervalo de tiempo: 1.0 [s]
Datos Enviados: 86

Estado de Conexión:
Conexión: On

(74.24, 55.701)

Fig. 7 Master application to transmit data



Laboratorio Remoto - Cátedra de ICP

Listado de Trabajos Practicos

TP3: Identificación y Dinámica de Sistemas de Alto Orden

Max: 80
Min: 10

Información General:
Fecha: 27-Aug-2021
Hora: 12:27:09

Datos del Experimento:
Recibido: -
Registro t: 181.0194 [s]
Temp. actual: 69.125 [C]

Estado de Conexión:
Conexión: On

(8.8479, 68.16)

Fig. 8 Client application to receive data.

In order to measure the performance of the experiment remotes an the developed software, students were asked to complete a survey about the experience they had with remote activities carried out.

In contrast to the previous survey, the current study focused on remote experiments and the SCADA software developed for remote furnace operation. The survey questionnaire included the following key questions, with the corresponding results presented in Table II.

A. Questions about remote activities

1. Qualify the remote experimental activities.
2. Were you able to carry out the activities that were realized during the remote experimental activities?
    a) Without problems.
    b) I have problems to start up the connection.
    c) I found some difficulties to follow the activities.
    d) I could not do the activities.

B. About the developed SCADA software.

1. Was the software developed easy to understand and use?
2. Rate the developed graphical interface.
3. Was the software was useful to understand the topics developed in classes?
4. Do you consider that the help window is useful to assist them during the remote experimental activities?

TABLE II
SURVEY ABOUT REMOTE EXPERIMENT AND DEVELOPED SOFTWARE

A. About Remote Experimental Activities

Qst.	Min				Max		Min	Max	Avg.
	1	2	3	4	5				
1	0	1	15	25	21		1	4	4.06

Qst.	(a)	(b)	(c)	(d)			
2	44	12	4	2		(d)	(a)

B. About the Developed SCADA Software

Qst.	No	Not Much	Yes
1	3	17	42

Qst.	Min				Max		Min	Max	Avg.
	1	2	3	4	5				
2	0	1	6	36	19		2	4	4.18
3	0	4	6	30	22		2	4	4.18

Qst.	No		Yes
4	17		75

Based on the data presented in Table II, it can be inferred that the remote experiments conducted in this study were highly successful and well-received by the participating students. The developed SCADA software was found to be user-friendly and intuitive, as evidenced by the positive feedback received on the graphical interface and the ease with which students were able to follow the class topics. Moreover, the availability of a help window was highly appreciated by the students, indicating the importance of providing adequate support and guidance in remote learning environments.

V. RESULTS AND CONCLUSIONS

Subsequently, two applications developed using this tool were presented. The first application demonstrates its use for teaching control systems, allowing students to study and delve into the main topics of this degree subject. Results collected through surveys demonstrate the high level of acceptance among users of this toolbox.

The second application was specifically designed to facilitate experimental work by allowing students to remotely connect to the process control laboratory. Through the survey presented earlier, it can be concluded that the SCADA developed with this application is user-friendly and intuitive, while also providing easy remote access to laboratory equipment.

Moreover, it is important to highlight that guiEditor is available in the GitLab repository for the academic community, providing free and open access. This development contributes to promoting transparency and collaboration in the educational software development community. In addition, a snap version was incorporated in July 2022 to simplify its use in Linux, which has already registered almost 300 users from 54 countries, according to metrics provided by the snapcraft.io site. This suggests that, beyond what is stated in this document, the tool has been considered useful by members of the free software community.

It is important to acknowledge that third-party works have been published in the academic literature, demonstrating the educational value and applications of the tool presented in this study. Notably, references [8] and [9] provide examples of such works, highlighting the potential of the tool to enhance the quality of teaching and learning in various educational contexts.

Like other similar open source projects, it is a project in continuous development, so it is expected to continue improving and expanding its functionalities as the community continues to find it of interest.

REFERENCES

[1] J. T. Trejo, A. F. Pérez, M. C. Huamán, J. A. Quiroz, R.T. Huamán, and V. B. Olivera. "Experiences in the Usage of Octave on Improving Learning Mathematics in an Engineering Faculty". In 2021 IEEE World Conference on Engineering Education (EDUNINE) (pp. 1-4). IEEE. (2021, March)

[2] V. P. Pavlović, and J. T. Ilić. "Implementation of GNU Octave in a University Course of General Physics". In 8th International Scientific Conference Technics and Informatics in Education (2020)

[3] J. W. Eaton, D. Bateman, and S. Hauberg, "Gnu octave", 2013.

[4] E.S. Burgos and E. J. Adam, "Graphical User Interface Editor for Octave Applications", *Engineering Reports*, pp. 1–15, July 2020.

[5] E.S. Burgos and E. J. Adam, "The guiEditor proyect site", https://gitlab.com/labinformatica/guieditor/, 2020.

[6] E. S. Burgos and E. J. Adam, "The guiEditor II project site", https://gitlab.com/labinformatica/guieditor_ii/, 2023

[7] E. J. Adam, E.S. Burgos, M. F. Picó, and M. E. Morelli, "Prácticas de Sistemas de Control Mediante Laboratorio Remoto", 5to Congreso Argentino (CADI), 3er Congreso Latinoamericano de Ingeniería (CLADI) y 11vo Congreso Argentino de Enseñanza de la Ingeniería (CAEDI), October 2021

[8] J. P. Correia Barbosa, J. G. Lima Neto, P. A. Linhares Freitas, R. de Araújo Bezerra, and R. do Nascimento Rodrigues, "Desenvolvimento de Software Educacional para Cálculo do Momento de Inercia de Área para Perfis Estruturais". Revista de Engenharia e Tecnologia, 15(1), 2023.

[9] J. G. Lima Neto, J. P. Correia Barbosa, J. V. C. B. Holanda Gomes, R. de Araújo Bezerra, R. do Nascimento Rodrigues, "Análise Estructural não Linear Estática de um Protótipo Baja Sae Pelo M.E.F.", Revista de Engenharia e Tecnologia, 14(3), 2022.