# Internet-oriented measurement and automation project for Industry 4.0 educational program

L. Angrisani, F. Bonavolontà, M. D'Arco, A. Liccardo
Dep. Computer Science and Electric Engineering
University of Naples Federico II
Via Claudio 21, 80125 Napoli, ITALY

O. Tamburis
Dep. Veterinary Medicine and Animal Productions
University of Naples Federico II
Via Delpino 1, 80137 Napoli, ITALY

***Abstract*— In the paper a remote laboratory as educational tool in measurement and automation courses is presented. Industry 4.0 and IoT paradigms require future information engineers to be able to use enabling technologies not only for the automation of measurements, but also for their remote management. For this purpose, a solution is proposed that makes the students an active part in the process of realization of the remote laboratory. An architecture involving typical laboratory instruments, introduced during measurement courses, and a server to allow remote access to laboratory equipment is implemented. The realization of the software architecture, however, is left to the students who, under teacher's guide, learn how to use the tools and programming environments to build up remote measurement stations.**

***Keywords—Remote laboratories; automatic measurement stations; remote stations didactics.***

## I. INTRODUCTION

In engineering teaching courses, laboratory experiences are a fundamental stage of learning [1]. In this context, remote laboratories are a very useful tool as they allow students to perform experiments 24 hours a day from any Internet-enabled workstation [2]-[7].

Cost advantage is the main characteristic of a remote laboratory: in a traditional laboratory it is in fact necessary to create an appropriate number of laboratory stations with respect to the number of students, while in a remote laboratory the stations are accessible at any time. In this way students are able to perform (and repeat, if necessary) the assigned experiments with a reduced number of stations, therefore saving the hardware equipment [8]-[19]. In addition, the remote laboratory can be powered by including other experiments that involve advanced devices located in physical laboratories distributed throughout the territory. Further advantages are related to safety, since the operator is physically distant from the laboratory and can conduct experiments without risks [20].

Until few years ago, the design and implementation of a remote laboratory was a multidisciplinary task that required the intervention of various experts from different sectors [21]-[25]: in order to be able to remotely control an automatic measurement station, it is in fact necessary for the student to gain knowledge regarding the didactic experiments of the course, and then to focus the attention on those aspects that need therefore to be transferred via a remote experience. Moreover, the knowledge of the laboratory instrumentation is necessary to understand how the remote user can interact with the instruments and design the client interface. Eventually, the experience in computer science is necessary to realize the hardware architecture of the laboratory, to figure out the most suitable communication protocol, as well as to implement the software, both server- and client-side, based on the characteristics of the remote experiment and the type of clients.

Currently, the creation of a remote laboratory requires less effort and can be quickly achieved even by those who do not possess relevant experience in communication networks and IT skills, thanks to the tools made available to users from different software environments. As a consequence, the implementation of a remote measurement station can be assigned as a project within the measurement and automation courses. Students have in this way the opportunity to learn and use the enabling technologies of remote monitoring and automation [26]. Although the capability to create automatic measurement and control stations is no longer sufficient in the typical Industry 4.0 processes, the ability to know how to communicate with a remote device is anyway required, as well as the capability to implement internet-oriented measurement processes [29]-[32].

The paper presents the structure of the remote laboratory, with the intent to lead students during the creation and testing of their own software for remote control of the instrumentation. The article is organized as follows: after a brief description of the bases of the remote laboratories, given in Section II, the server architecture is described in Section III; Section IV then provides details on the server software for controlling the connected instruments, and Section V describes the structure of the client software. Finally, some conclusions are drawn in Section VI.

## II. REMOTE LABORATORY ASSET

The scheme of a typical remote laboratory is shown in Fig. 1. The server machine is physically connected to the laboratory equipment, and communicates with the devices via a proper interface. The server is also connected to the Internet network, in order to receive remote requests.

The clients' machines are geographically distributed personal computers that can communicate with the server through Internet connections.

For what concerns the software architecture, there are no rules on the software environment for the implementation of server and client programs; it is only required that both server

and client applications are able to exchange data according to the TCP/IP protocol. The authors, in particular, have used LabVIEW environment, exploiting their experience in the realization of programs for automatic measurement stations, but the approach described in the paper can be extended to all software environments that allow configuring TCP/IP communication parameters.

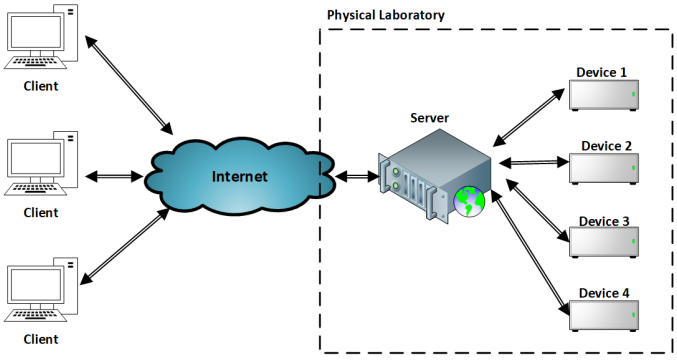


Fig.1 Block diagram of a remote laboratory based on server/client paradigm.

The server application can be seen as a translator between the TCP/IP protocol and the interface adopted for the communication with the laboratory equipment. It receives via Internet the commands that the client wants to send to a specific instrument and forwards them to the device, adapting them (if needed) to the device interface. Similarly, it forwards the response from the local device back to the client via TCP/IP connection.

Usually the client needs to send a sequence of messages to the laboratory instrumentation in order to perform an experiment. The server must process these messages during a single client connection: it is important in fact to avoid that, when performing the experiment of a client, the server receives and processes messages due to the connection of another client. The messages exchanged between client and server applications must be therefore structured so that in a single connection the client completes the tasks structuring the experiment.

The authors suggest to use the array type to this purpose and describe, in the following sections, how the client and software interact with each other.

## III. Server Software

The server application has to receive client messages. The first server operation is then the creation of a TCP listener. LabVIEW offers its library built-in function to create a listener that allows selecting some parameters as the Ethernet board (if more than one board is present) and the port number of the connection.

The port number should be in the range 49152-65535 that is the range of the so-called dynamic and/or private ports, according to Internet Assigned Numbers Authority (IANA) specifications.

It has to be noticed that some operating systems do not meet IANA specifications – Windows, for example, allows the use of ports ranging within 1024 and 5000, which is an interval included in the range of ports classified by IANA as user or registered ports. It is still possible to create a listener on these ports, but the possibility of receiving an error message due to a conflict on the port has to be taken into account. The library function that creates the listener returns a handle that uniquely identifies the listener.

Once the listener is designed, a function that waits for the client TCP connection is necessary. This function can be configured for waiting indefinitely a client connection or a timeout can be set, so that when the timeout expires an error message is returned. If a client connection occurs, this function returns the handle for identifying the connection, the IP address and the port of the remote client.

When the connection is established, the server receives the client message through a TCP read function and sends messages to the client through a TCP write function.

Specifically, the TCP read is a flexible function that can be configured in order to obtain different behaviors, depending on the selected mode. For example, the user can configure a termination character that, when received, indicates the end of the client message, or specify that the function has to immediately return the received bytes regardless the expected length of the client message. Alternatively, user can specify the number of bytes that have to be received from the client and a timeout interval: if a number of bytes lower than that expected is received when the timeout expires, the user can select if this function has to return an empty string or the partial message received (in both cases an error condition is raised).

In particular, the authors suggest the last configuration since the error signal is monitored; if all the expected bytes have not been received, the connection is closed.

The expected client message is structured as shown in Fig. 2. The header is composed by eight bytes. The first four bytes represent the client ID adopted for the client authentication; if the client ID is not in the list of clients authorized to use the laboratory, the server closes the connection. The other four bytes of the header express the length in bytes of the remaining data block, referred in the following as payload.

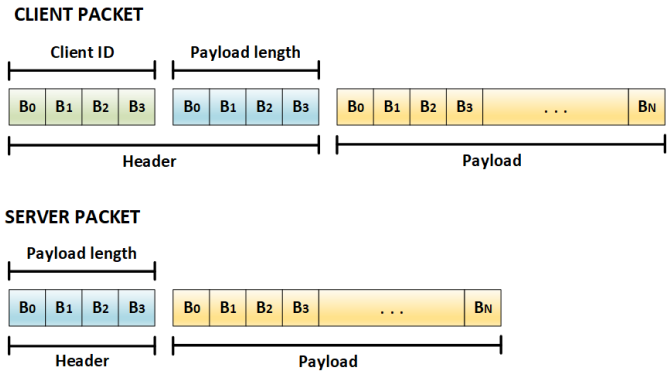


Fig.2 Structure of client and server messages.

The payload contains the directive for the laboratory equipment, organized as a sequence of tasks that have to be properly processed by a program called Instrument and Actuator Controller, described in detail in the following section.

Header and payload are received through two different TCP read functions. The first one reads the header so that the server program can verify the client ID and then obtain the length of the payload. After sending an acknowledge message, a second TCP read function is performed: it is configured for reading a number of bytes equal to the length received in the header, but also a timeout interval is set, to prevent that the server program is blocked waiting to receive the expected number of bytes. If the timeout expires, the connection is closed, regardless the payload received until then.

The payload contains the data to be transmitted to the laboratory devices, and is organized as an array that is a sequence of structured data, corresponding to particular tasks. The array is passed to the Instrument and Actuator Controller that, interfacing with the local devices, executes the tasks. Some tasks cause a response from the device, such as the result of a measurement. To allow the client to quickly retrieve the device response, the Instrument and Actuator Controller maintains the order relationship between the task array and the response array. Specifically, for each task that is executed the Instrument and Actuator Controller adds a string to the response array that will be returned to the client: if the device has returned data, the string that is appended to the array is exactly the data returned by the device; alternatively, an empty string is appended. In this way, the client receives an array whose size is the same as the array that has been sent; since the client knows the position of a query within the task array, he knows where to find the device response in the response array.

The server program sends the response array to the client using a TCP write function. Actually, the server sends a packet similar to those sent by the client, as shown in Fig. 3, but the bytes related to the identification are omitted for the server. The header, then, is only composed by the four byte expressing the length of the payload.

Finally, the server closes the connection using the handle for identifying the client connection. It is worth noticing that the described operations are indefinitely iterated in a while loop, in order to accept and process other client connections.

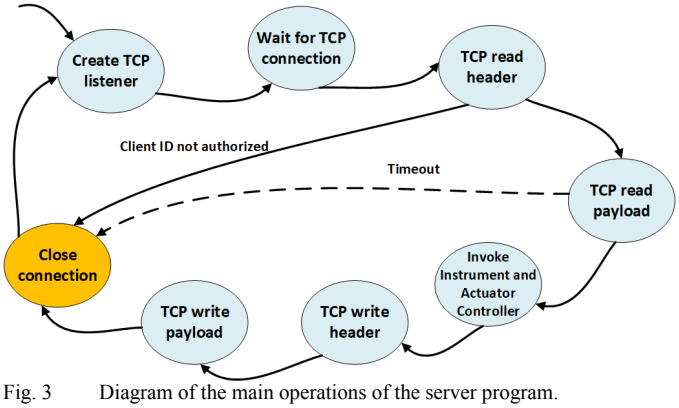


Fig. 3 Diagram of the main operations of the server program.

## IV. Instrument and Actuator Controller

The Instrument and Actuator Controller communicates with the laboratory device, according to one or more interfaces. Many software environments (including LabVIEW) offer, also for this type of communication, library functions for communicating with devices through the interfaces most commonly adopted in industrial processes. In the developed laboratory the IEEE 488 interface is adopted.

The Controller receives the array of tasks contained in the payload of the client message. A task is an operation performed with a specific instrument that can be the transmission of configuration commands or the data reading from the instrument buffer. The task is composed by the following fields: (i) the device ID identifies through a symbolic name the device involved by the task; (ii) the operation, that can be *write* or *read*; (iii) the field is dependent on the operation: if the operation is write, it contains the data to be sent to the device, otherwise it is a number representing the number of bytes to be read from the device; (iv) a delay that the Controller has to wait before executing the required operation.

In order to better describe the task structure, in Fig. 4 a basic example of received array of tasks is shown. In the example the client wants to configure the source, namely an arbitrary waveform generator (AWG), for obtaining a square wave signal with frequency of 2.5 kHz and peak-to-peak amplitude of 1 Vpp. The client wants as well to measure the true rms voltage of the signal by means of a digital multi-meter (DMM). The device ID of the *first* task, that is AWG, indicates that such task involves the generator. In particular, the task is a write operation and the message to be written contains the command for properly configuring the generator, listed in the programmer's manual of the instrument. The *field delay* equal to zero indicates that the write operation can be executed immediately.

The device ID of the *second* task is equal to DMM, which means that the task involves the digital multi-meter. The task performs a write operation and sends to the multi-meter the configuration commands (available in the programmer's manual of the instrument) for measuring the rms voltage of the input waveform.

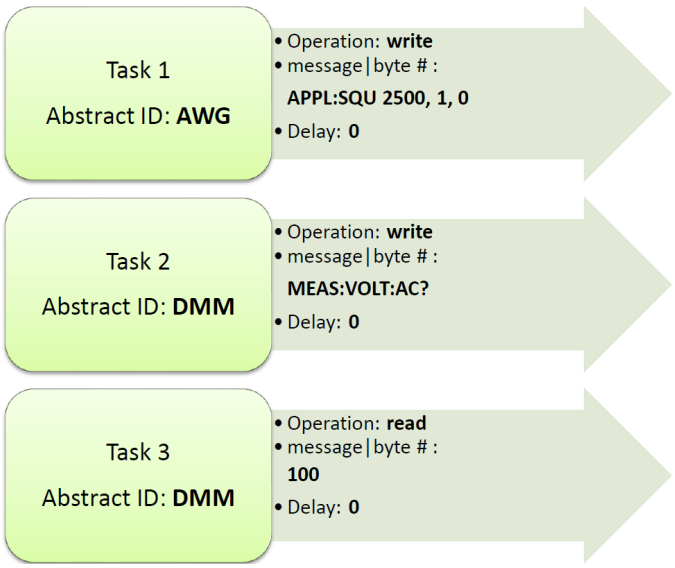


Fig. 4 Example of array of tasks received by the Instruments and Actuator Controller.

The *third* task aims at acquiring the measurement result. The device ID is still DMM, but the operation is equal to read.

This time the message field contains the number of bytes to read, set equal to 100.

The response array built by the Instrument and Actuator Controller, according to the example of Fig. 4, has size equal to three. The first and second elements are empty strings, corresponding to the write operations; the third array element is the string returned by the digital multi-meter in the read operation performed in the third task.

## V. Client Software

Authors chose LabVIEW environment for the implementation of the client software as well, but the described operations can be performed in other environments.

The client program has to establish an Internet connection with the server in order to send the tasks to be executed. A proper library function is used to such scope. In order to correctly configure the open TCP function, the remote user needs to know the physical IP address of the server and the remote port number. This function receives also a timeout parameter, since if the connection is not established and the timeout expires, the function returns an error condition. Another output of the open TCP function is the handle that uniquely identifies the connection session.

Following, the client has to send the array of task to the server. To this aim, the client has first to build the array, in dependence on the operation sequence that the remote user wants to perform. It has to be noticed that the client can be implemented by following two different approaches, depending on the level of knowledge that the remote user is supposed to have. The basic user is who simply has to perform operations with the laboratory devices, without having the skills to create an automatic measurement station or to communicate with a remote laboratory. In this case, the software interface is very user friendly and the client software builds the array of tasks based on the selections. The specialized user, to whom the laboratory is mainly addressed, after having designed the process to be executed, is able to describe the tasks, to organize them into a single array and to use the TCP/IP functions to transmit the latter to the server.

In both cases the client program, after the construction of the array of tasks, has to send it as a string, which represents the payload of the message to be transmitted. According to the protocol described in the previous section, the client has to evaluate the length of the payload, in order to build the header.

The header is obtained by concatenating the four bytes representing the authentication ID, and the four bytes expressing the length of the payload.

The transmission of header and payload is performed through a library function TCP write. The client program must include a wait function, proportional to the number and complexity of the tasks sent, in order to allow the server to perform all the required tasks.

After the waiting time has elapsed, the client can receive the server response. Also, this operation is performed with the TCP read library function, which returns the array of responses relative to each task, from which the client can extract the responses of interest of the queried devices.

The last operation is to close the connection; the TCP close function is used, which requires in input the connection handle returned by the open function.

For the sake of clarity, a diagram of the main operations of the client software is shown in Fig. 5.

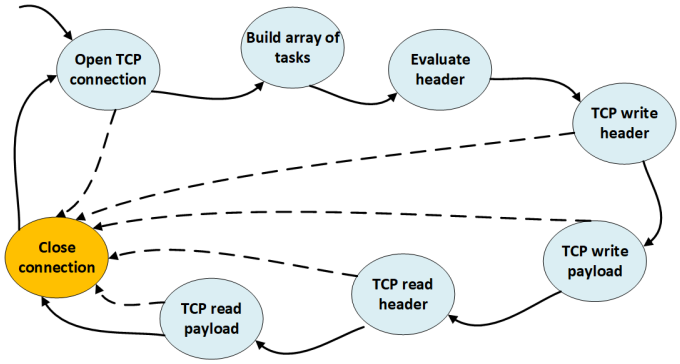


Fig. 5 Diagram of the main operations of the client program; the dashed lines indicates a timeout event.

## VI. Conclusions

In the paper an educational approach aimed at teaching the implementation of measurement processes geographically distributed throughout the territory is proposed. The authors in fact believe that remote laboratories should no longer be tools just to make students design automated measurement as if they were physically in the laboratory. Rather, students can be trained on fundamental themes of IoT and Industry 4.0 as pillars of openness [33], making them an active part in the realization of the remote laboratory itself.

As part of the project proposed to students regarding the realization of a remote laboratory, students are invited to also consider issues related to the management of errors and the management of concurrent client requests.

## References


[1] L. Angrisani, P. Arpaia, F. Bonavolontà, R. Schiano Lo Moriello, "Academic FabLabs for industry 4.0: Experience at University of Naples Federico II", (2018) IEEE Instrumentation and Measurement Magazine, 21 (1), art. no. 8278802, pp. 6-13.

[2] U. Lichtenthaler, "Open Innovation in Practice. An analysis of strategic approaches to technology transactions", IEEE Transactions, Vol. 55, no. 1, pp. 148-157, January-March 2008.

[3] L. de la Torre, J.P.Sànchez and S. Dormido, "What remote labs can do for you", Physics Today, Vol. 69, no. 4, 2016.

[4] G. Andria et alii "Remote didactic laboratory 'G. Savastano', the Italian experience for e-learning at the technical universities in the field of electrical and electronic measurement: architecture and optimization of the communication performance based on thin client technology" IEEE Trans. on Instrum. and Meas., vol. 56, no. 4, 2007, pp. 1124-1134.

[5] A. Baccigalupi, U. Cesaro, M. D'Arco, A. Liccardo, "Web-based networking protocol for expanding IEEE-488 ATE capabilities", IEEE International Workshop Measurements and Networking (M&N), Capri, Italy, 2011, pp.100-104.

[6] J. M. G. Palop and J. M. A. Teruel, "Virtual work bench for electronic instrumentation teaching", IEEE Trans. Educ., vol. 43, no. 1, 2000, pp. 15–18.

[7] C. C. Ko, B. M. Chen, S. H. Chen, V. Ramakrishnan, R. Chen, and Y. Zhuang, "A large-scale Web-based virtual oscilloscope laboratory experiment", Eng. Sci. Educ. J., vol. 9, no. 2, 2000, pp. 69–76.

[8] P. Arpaia, A. Baccigalupi, F. Cennamo, and P. Daponte, "A measurement laboratory on geographic network for remote test experiments", IEEE Trans. Instrum. Meas., vol. 49, no. 5, 2000, pp. 992–997.

[9] G. Canfora, P. Daponte, and S. Rapuano, “Remotely accessible laboratory for electronic measurement teaching”, Comput. Stand. Interfaces, vol. 26, no. 6, 2004, pp. 489–499.

[10] L. Benetazzo, M. Bertocco, F. Ferraris, A. Ferrero, C. Offelli, M. Parvis, and V. Piuri, “A Web based, distributed virtual educational laboratory”, IEEE Trans. Instrum. Meas., vol. 49, no. 2, 2000, pp. 349–356.

[11] M. Albu, K. Holbert, G. Heydt, S. Grigorescu, and V. Trusca, “Embedding remote experimentation in power engineering education”, IEEE Trans. Power Syst., vol. 19, no. 1, 2004, pp. 139–143.

[12] P. Arpaia, A. Baccigalupi, F. Cennamo, P. Daponte, “A remote measurement laboratory for educational experiments, Measurement, Volume 21, Issue 4, August 1997, Pages 157-169.

[13] Domenico Grimaldi, Sergio Rapuano, “Hardware and software to design virtual laboratory for education in instrumentation and measurement”, Measurement, Volume 42, Issue 4, May 2009, Pages 485-493.

[14] M. Corrado, L. De Vito, H. Ramos, J. Saliga, “Hardware and software platform for ADCWAN remote laboratory”, Measurement, Volume 45, Issue 4, May 2012, Pages 795-807.

[15] Unai Hernandez-Jayo, Javier Garcia-Zubia, “Remote measurement and instrumentation laboratory for training in real analog electronic experiments”, Measurement, Volume 82, March 2016, Pages 123-134.

[16] N. Wang, X. Chen, Q. Lan, G. Song, H. R. Parsaei and S. C. Ho, "A Novel Wiki-Based Remote Laboratory Platform for Engineering Education", in IEEE Transactions on Learning Technologies, vol. 10, no. 3, pp. 331-341, July-Sept. 1 2017.

[17] N. Valov and I. Valova, "Drying process management laboratory with remote access", 2017 16th International Conference on Information Technology Based Higher Education and Training (ITHET), Ohrid, 2017, pp. 1-6.

[18] C. Arguedas-Matarrita et al., "A teacher training workshop to promote the use of the VISIR remote laboratory for electrical circuits teaching", 2017 4th Experiment@International Conference (exp.at'17), Faro, 2017, pp. 1-6.

[19] P. Di Giamberardino and M. Temperini, "Adaptive access to robotic learning experiences in a remote laboratory setting", 2017 18th International Carpathian Control Conference (ICCC), Sinaia, 2017, pp. 565-570.

[20] A. Tocchi, V. Roca, L. Angrisani, F. Bonavolontà, R. Schiano Lo Moriello, “First step towards an IoT implementation of a wireless sensors network for environmental radiation monitoring” , I2MTC 2017 - 2017 IEEE International Instrumentation and Measurement Technology Conference, Proceedings, DOI: 10.1109/I2MTC.2017.7969754

[21] L. Angrisani, F. Bonavolontà, R. Schiano Lo Moriello, A. Andreone, R. Casini, G. Papari, D. Accardo (2014). “First steps towards an innovative compressive sampling based-THz imaging system for early crack detection on aereospace plates”, In: 2014 IEEE Metrology for Aerospace (MetroAeroSpace). p. 488-493, ISBN: 9781479920693, Benevento, 29-30/05/2014, doi: 10.1109/MetroAeroSpace.2014.6865974

[22] P. Orduña et al., "An Extensible Architecture for the Integration of Remote and Virtual Laboratories in Public Learning Tools", in IEEE Revista Iberoamericana de Tecnologias del Aprendizaje, vol. 10, no. 4, pp. 223-233, Nov. 2015.

[23] A. C. Caminero, S. Ros, R. Hernández, A. Robles-Gómez, L. Tobarra and P. J. T. Granjo, "VirTUal remoTe labORatories Management System (TUTORES): Using Cloud Computing to Acquire University Practical Skills", in IEEE Transactions on Learning Technologies, vol. 9, no. 2, pp. 133-145, April-June 1 2016.

[24] A. Chevalier, C. Copot, C. Ionescu and R. De Keyser, "A Three-Year Feedback Study of a Remote Laboratory Used in Control Engineering Studies", in IEEE Transactions on Education, vol. 60, no. 2, pp. 127-133, May 2017.

[25] N. Wang, X. Chen, G. Song, Q. Lan and H. R. Parsaei, "Design of a New Mobile-Optimized Remote Laboratory Application Architecture for M-Learning", in IEEE Transactions on Industrial Electronics, vol. 64, no. 3, pp. 2382-2391, March 2017.

[26] F. Bonavolontà, A. Tedesco, R. Schiano Lo Moriello, A. Tufano, “Enabling wireless technologies for industry 4.0: State of the art”, 2017 IEEE International Workshop on Measurement and Networking, M&N 2017 - Proceedings, DOI: 10.1109/IWMN.2017.8078381.

[27] L. Angrisani, P. Arpaia, F. Bonavolontà, M. Conti, A. Liccardo, (2017,). “LoRa protocol performance assessment in critical noise conditions,” In 2017 IEEE 3rd International Forum on Research and Technologies for Society and Industry (RTSI) (pp. 1-5). September 2017.

[28] P. Orduña, L. Rodriguez-Gil, J. Garcia-Zubia, I. Angulo, U. Hernandez and E. Azcuenaga, "LabsLand: A sharing economy platform to promote educational remote laboratories maintainability, sustainability and adoption", 2016 IEEE Frontiers in Education Conference (FIE), Erie, PA, USA, 2016, pp. 1-6.

[29] F. Bonavolontà, et al. "New approach based on compressive sampling for sample rate enhancement in DASs for low-cost sensing nodes." Sensors 14.10 (2014): 18915-18940.

[30] J. M. M. Quintero, P. E. N. Ortiz, D. M. O. Martinez and L. F. C. Alfonso, "Low cost remote instructional laboratory for control systems courses", 2016 International Conference on Interactive Mobile Communication, Technologies and Learning (IMCL), San Diego, CA, 2016, pp. 78-82.

[31] J. Grodotzki, T. R. Ortelt and A. E. Tekkaya, “Remote and Virtual Labs for Engineering Education 4.0: Achievements of the ELLI project at the TU Dortmund University”, in Procedia Manufacturing, Vol. 26, 2018, pp. 1349-1360.

[32] J. Zalewski, J, “Cyberlab for cyberphysical systems: remote lab stations in software engineering curriculum”, The Fourth International Conference on e-Learning (ICEL2013), 2013, pp. 1-7.

[33] I. Bonacci, O. Tamburis, “Empowering openness: The case of CROs-related innovation networks in the Italian bio-pharmaceutical sector”, International Journal of Learning and Intellectual Capital, Vol. 13, Issue 2-3, 2016, pp. 184-201.

.