ACTA IMEKO
ISSN: 2221-870X
June 2019, Volume 8, Number 2, 62 - 69

Measurement applications in Industry 4.0: the case of an IoT-oriented platform for remote programming of automatic test equipment

Leopoldo Angrisani[1], Umberto Cesaro[1], Mauro D'Arco[1], Oscar Tamburis[2]

[1] University of Naples Federico II, Department of Computer Science and Electrical Engineering, Via Claudio 21, 80125 Naples, Italy
[2] Department of Veterinary Medicine and Animal Productions, Via Delpino 1, 80137, Naples, Italy

ABSTRACT
A laboratory regarded as a site that collects IoT devices, and which allows remote clients to use them as an automatic test equipment (ATE) through a controller acting as service provider, is proposed herein. To assure efficiency and responsiveness, the controller is programmed as a multithreading system that takes advantage of multicore processors. The controller includes a server application that supports communication with clients by means of a TCP/IP protocol. It uses GPIB bus functionalities to control the instruments of the local ATE. It allows several clients to connect and interact with the specific resources of the laboratory. Thanks to the availability of identical sets of resources and to the underlying multithreading philosophy, client requests are processed in tandem rather than according to a classical queuing approach.

Section: RESEARCH PAPER

Keywords: TCP/IP communication; multithreading; multi-core processors; client-server application; execution system

Citation: Leopoldo Angrisani, Umberto Cesaro, Mauro D'Arco, Oscar Tamburis, Measurement applications in Industry 4.0: the case of an IoT-oriented platform for remote programming of automatic test equipment, Acta IMEKO, vol. 8, no. 2, article 9, June 2019, identifier: IMEKO-ACTA-08 (2019)-02-09

Section Editor: Emiliano Sisinni, University of Brescia, Italy

Received July 27, 2018; **In final form** June 18, 2019; **Published** June 2019

Corresponding author: Mauro D'Arco, e-mail: darco@unina.it

1. INTRODUCTION

The term 'Industry 4.0' refers to the fourth industrial revolution we are currently witnessing: the whole sphere of industrial production is in fact undergoing a comprehensive transformation that involves the merger of conventional industrial processes with digital technologies and the Internet. Such phenomenon is intended both (i) to improve existing technological capabilities by complementing them with value-added features and (ii) to enable new ones by exploiting the cooperation between the most recent digital equipment and more traditional assets. A great impact on mass customisation, integration of value chains, and process efficiency has come from the merging of today's sensor technology, interconnectivity, and data analysis capabilities. Accordingly, new open collaboration and innovation paradigms are taking root, always seeking confidence in an efficient quality infrastructure, thus pushing towards relevant changes in societal and economic activities. Companies are in fact not only reorganising their innovation processes by acquiring external information, knowledge, and skills, but they are also providing access to complementary assets outside their boundaries [1]. This situation causes, for instance, increasing deployment of remotely controlled strategic activities in several industrial plants.

The same openness lies behind the IoT technological paradigm, envisioned as a global network of physical devices connected to the Internet [2]. On this basis, authorised users can access equipment from anywhere via Internet in order to inspect data, make decisions, and drive operations [3]-[14].

Despite the level of application, achieving an efficient quality infrastructure lies inevitably in the ability to obtain valid data based on high-precision measurements so that it is appropriate to include such a scenario in the category of 'MetroIndustry 4.0'. In particular, the role of measurement science in the promotion of digital expansion and transformation is being intensively pushed forward by means of research programmes and by setting up new capacity groups. It is also targeted at the centre of the development of new strategies for engineering education 4.0 [15], [16]. Its application in simulations and virtual measuring instruments is therefore counted among the newly identified focuses.

Accordingly, laboratories are upgradable to sites collecting IoT, capable of offering value-added functionalities to the interacting subjects [16]-[24]. From a broader perspective, they can also be thought of as computing systems with access to physical measurement and control instruments as well as connectivity to the Internet—or, in other words, cyberphysical systems (CPSs) [25]. To this end, the present work introduces a proposal for the remote programming of automatic test equipment (ATE) and the development of measurement and automation applications.

Different to several web-based applications, implemented nowadays by means of high-level technologies such as web-services, the proposed solution relies on basic TCP/IP data transfer mechanisms, not involving the hypertext transfer protocol (HTTP) and related concepts [26][25]-[31]. In particular, it consists of a platform, designed as a LabView application, which is initially conceived as a training tool for technicians involved in remote programming tasks. Technicians usually work with hardware directly connected to some development kits or integrated into evaluation boards. Less frequently, technicians are required to program remote instruments without the possibility of acquiring immediate visual feedback or conducting focused inspections on physical features of the instruments during a series of scheduled operations [33]-[37].

The platform user can ask the remote ATE for the execution of measurement applications, as specific services, by programming and specifying a sequence of tasks for a given set of instruments. The server authenticates the connecting clients and interrogates them in order to obtain more details about their request. Thereafter, it processes the request in order to decode the specific tasks in the list, and it then activates a selected set of resources that can include stimulus sources and data acquisition systems. Each task is an action that either defines a list of settings for the addressed instrument or queries the transfer of measurement data. The user can control the timing execution of the tasks by introducing delays between each of them. Several measurement applications typically require a source to stimulate a device under test and, after a controlled delay related to the responsiveness of the device, an observer to inspect its output. The server eventually sends to the client a string response containing the information that is relevant to the outcomes of the measurement process.

By definition, remote laboratories are essential for the process of learning and assimilating scientific concepts. The proposed IoT-oriented approach, specifically envisioned for application in an Industry 4.0 environment, is particularly valuable in addressing responsiveness- and complexity-related issues.

In the following sections, the software architecture of the proposed platform is illustrated, and the code of the main threads is shown, defined by means of the LabView graphical language.

2. PROPOSED PLATFORM

In order to provide a general picture of the macro activities performed by the proposed platform, a high-level unified modelling language (UML)-like activity diagram [38] is shown and annotated in Figure 1.

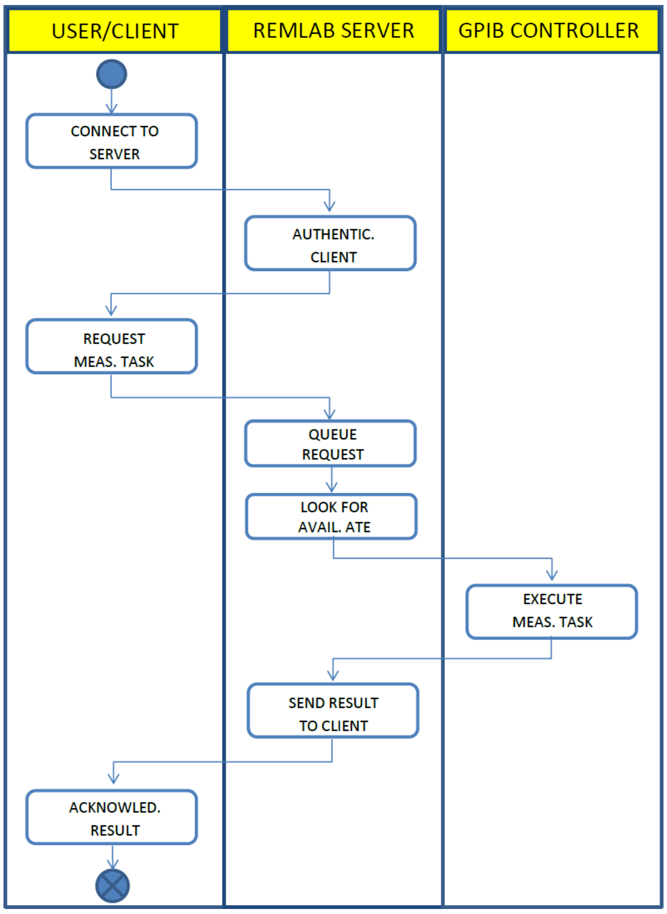


Figure 1. UML-like activity diagram illustrating the macro activities performed by the proposed platform. The user requests the execution of measurement tasks via the remote server. The server authenticates the connecting clients, then it retrieves, queues and processes their request by activating a selected set of resources i.e. a specific ATE. Finally, the server sends a response containing the measurement results.

Then, in order to illustrate each macro activity and provide details related to actual implementations, made in LabView 7.0 SE, in sufficient detail, screenshots of the block diagrams of the most important threads are also given. A general picture of the block diagram that illustrates the operation of the server machine can be found in Appendix I.

The server uses several parallel threads (named TCP connection thread, check-in thread, management thread, and check-out thread) to acquire, process and hand over client requests. It also uses N measurement threads, named GPIB threads, to control the available measurement stations. In the proposed case study, only two ATEs have been used. The laboratory is made up of identical measurement stations, each one including an arbitrary waveform generator (AWG) and a digital storage oscilloscope (DSO), and it allows N concurrent clients to work in parallel.

In brief, the TCP connection thread waits for a client connection and assigns an identifier to any new connection. The check-in thread communicates with newly connected clients and creates a cluster of data specifying the details of the client request. The management thread inspects the queue of active requests as well as the state of available ATEs. If a request is queued and an ATE becomes available, it schedules the client request to the available ATE and wakes up the GPIB thread that is responsible for controlling that ATE. GPIB thread decodes and serves the client request; finally, it queues

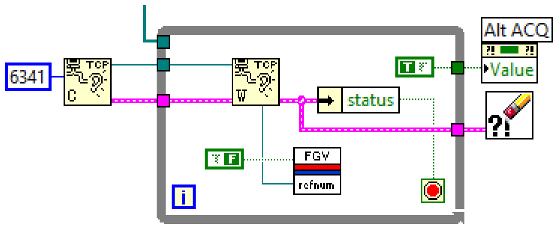


Figure 2. Block diagram of the TCP connection thread.

the results into a response string message. The response string is retrieved by the check-out thread and is sent to the client by means of TCP/IP functions.

The threads exchange data by means of queues implemented with functional global variables (FGVs). Queue structures based on FGV concepts can solely be accessed by means of the methods defined by the programmer and allow threads to exchange data in synchronized way.

In the following section, the names of the main threads are written in bold, while the names of sub-VI and the variables are written between quotation marks.

2.1. TCP connection thread

The thread is typically idle and wakes up on client connection. It is coherent to a typical programming template that uses a while loop inside which a 'TCP wait on client connection' function is cyclically executed, as shown in Figure 2.

The input data for this internal function, namely the listener reference number (green wire) and error info (pink wire), is obtained by a 'TCP create listener' function outside the loop. The function inside the loop waits indefinitely for a new client connection and returns a specific reference number for a newly connected client. The functional global variable 'FGV refnum' implements a queue, wherein both queueing and de-queueing operations can be performed by selecting a Boolean control (true requests for de-queueing all data, false for queueing a single input). Should any error occur in the waiting connections, the **TCP connection thread** halts the server operations. To this end, it quits the while loop and changes the value of the Boolean variable 'Alt ACQ', which acts as stopping signal for the check-in thread. To carry out the change in the variable that is defined in another thread, the variable is addressed through a property node. The error propagates outside the while loop through the pink wire, where it is handled by the LabView standard error handler function.

2.2. Check-in thread

The basic operation cycle of the thread is demonstrated by the block diagram in Figure 3.

First, the queue implemented through the 'FGV refnum' variable, which contains the reference numbers of the clients connected to the server, is flushed. If the queue is not empty, then the state of the 'TCP warden' semaphore is acquired, and if the semaphore provides the necessary permission, sub-VI 'ACQ request' is invoked. The sub-VI is capable of preventing unknown users from hampering the platform by implementing an authentication mechanism. In particular, it acquires a short header message (8 bytes) from the client and scans it, looking first for a 32-bit integer, which should represent an authentication key. If the authentication key is absent, the client connection is immediately closed. Otherwise, the message is scanned further in order to acquire a second 32-bit integer specifying the length of the string that details the client request. The string is subsequently acquired along with an additional read operation expecting as many bytes as are specified in the aforementioned header. The sub-VI 'ACQ request' returns the string to the calling thread i.e. **check-in thread**, which releases the semaphore after completion of the operation. It concatenates the string to the client identifier. The compound string is queued into the variable 'FGV request' and is passed to the management thread through the very same 'FGV request' variable. Note that if the semaphore does not provide the necessary permission, no request is acquired, and the cycle terminates skipping the queueing operation. The repetition of the operation cycle of the check-in thread is delayed for a short time of 250 ms in order to reduce the usage of the central processor when shared with other threads. The cycle repeats indefinitely unless the value of the Boolean variable 'Alt ACQ', which is controlled by the **TCP connection thread**, becomes true.

On the left side of Figure 3, outside the while structure, some initialisation operations are performed. In particular, inspecting the aforementioned portion of the block diagram from the bottom up, one can find that the queue implemented by the variable 'FGV refnum' is initially flushed in order to start the **TCP connection thread** and **check-in thread** with an empty queue. Furthermore, the value of the Boolean control 'Alt ACQ' is set to false, and a semaphore that is later utilised

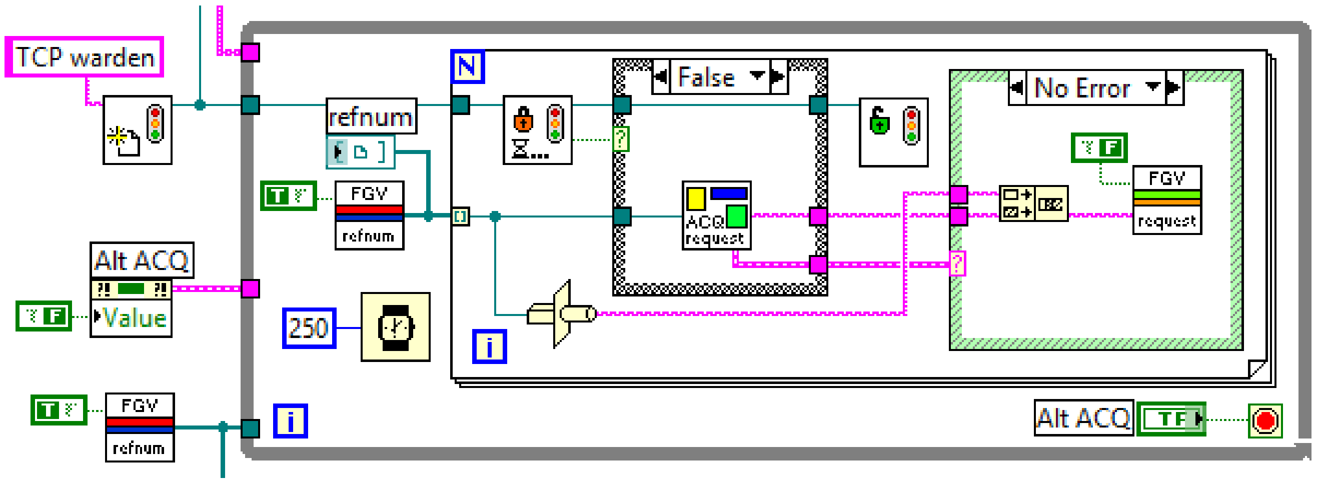


Figure 3. Block diagram of the check-in thread.

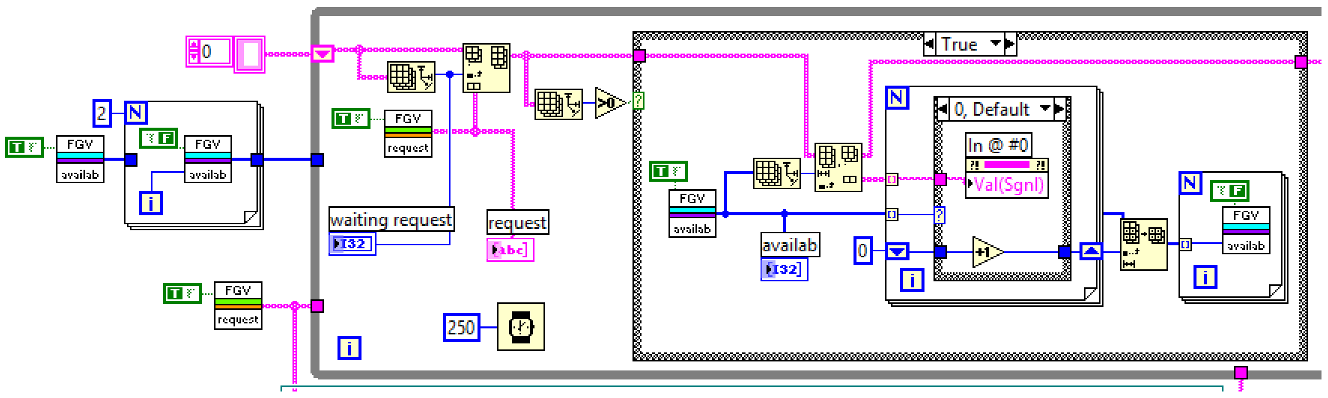


Figure 4. Block diagram of the management thread: core operation.

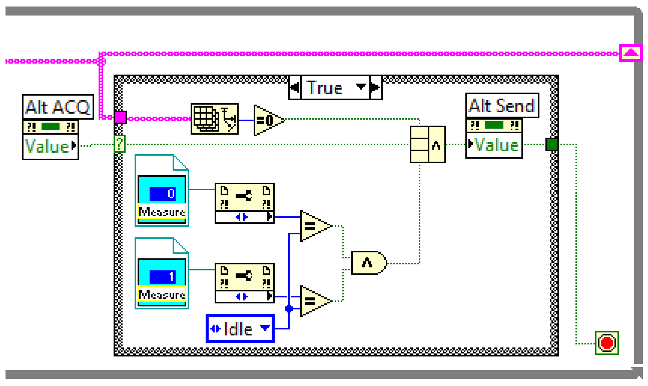


Figure 5. Block diagram of the management thread: stopping condition.

by the **check-in thread** and **check-out thread** is created and named TCP warden.

2.3. Management thread

As shown in Figure , the thread flushes the queue containing the client requests and updates a waiting list appending the requests to those already present in the list.
If the waiting list is not empty, the thread looks for available ATEs by flushing the queue implemented by the 'FGV availab' variable to obtain the list of available ATEs, each one identified by an integer. Then, it retrieves from the waiting list the same number of requests as the number of the available ATEs. The strings that detail the requests are individually signalled to identical instances of the **GPIB thread,** each one controlling an ATE of the available ones. Note that if the number of available ATEs is greater than the number of waiting requests, the **management thread** inserts the identifiers of the ATEs that are not engaged into operations into the queue implemented by 'FGV available', even if they are available.

On the left side of the block diagram, outside the 'while structure', the array of strings used as the waiting list is initialised to an empty array, and the queues implemented by means of 'FGV request' and 'FGV availab' are flushed; the latter is then initialised inserting the number identifying the available ATEs.

The stopping criterion of the thread, as illustrated in Figure , considers a set of conditions. The thread stops if the Boolean control 'Alt ACQ' becomes true, provided that the waiting list of the client requests is empty and all **GPIB threads** are idle.

To acquire the state of the **GPIB threads**, the threads are interrogated through property nodes. The Boolean value that results from the logic multiplication of the aforementioned conditions is also used to set the value of the Boolean control 'Alt Send', which is defined in the **check-out thread**.

Note that like the **check-in thread,** the repetition of the operation cycle of the **management thread** is also delayed for a short time of 250 ms in order to reduce the usage of the central processor when it is shared with other threads.

2.4. GPIB thread

Parallel **GPIB threads** are responsible for the execution of multiple measurement processes. Their software framework is identical, except for the addressed ATE. Each **GPIB thread** is awakened from its idle state on the occurrence of an event, signalled by the **management thread** as soon as the latter finds a client request in the waiting list and an available ATE. Specifically, the **management thread** uses a property node to signal and contextually transfer the string containing the client request to a supplemental thread, shown in Figure , which simply invokes the **GPIB thread** and provides it with the string as an input parameter.

Once invoked, each **GPIB thread** is temporarily locked to a single client request until the completion of the required operations. The block diagram of the prototype **GPIB thread** is shown as an example in Figure .

The thread retrieves the client message, skipping the header 4 bytes that contain the reference to address the client in TCP/IP communication. It then flattens the remaining string in order to recover an array of clusters, each one specifying a measurement task. Each measurement task is detailed in terms of:

- an action upon the GPIB bus (read/write);
- an addressed instrument (DSO/AWG);
- a waiting time before task execution; and
- a string message.

The string message contains either a command message (i.e. in

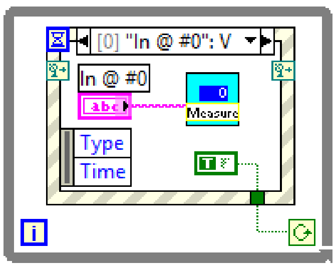


Figure 6. Block diagram of the thread with the event structure that activates the GPIB thread.

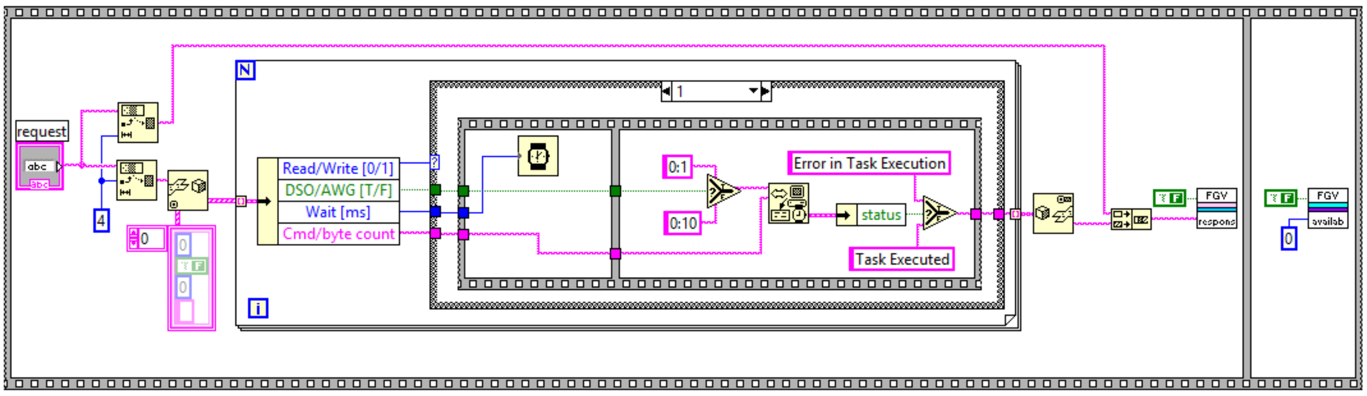


Figure 7. Block diagram of GPIB thread

the case of a write action) or a byte-count specification (i.e. in the case of a read action).

The function decodes the sequence of tasks and performs the required communication via the GPIB interface for each of them. Then, it assembles an array of string messages, each one expressing a response to a specific task. When all tasks contained in the client request have been completed, this array is flattened to a single string and inserted into the 'FGV respons' variable. Finally, the measurement station is made available again by updating the variable 'FGV availab'.

2.5. Check-out thread

The software for the thread is shown in the block diagram in Figure .

The thread flushes the queue implemented with the 'FGV respons' variable. For each string contained in the queue, it recovers the 4-byte header containing the client reference for use in TCP/IP communication, and it finally sends the measurement results contained in the remaining part of the string by invoking the sub-VI 'Send response'. The thread stops if the value of the Boolean control 'Alt Send' is set to true by the **management thread**.

The initialising operations performed by the functions visible on the left side of the block diagram outside the 'while structure' imply flushing the queue implemented by means of 'FGV respons' and setting the value of the Boolean control 'Alt Send' as false.

3. USER SIDE

The user of the platform is required to run a VI client application, the front panel of which is shown in Figure .

In the following demonstration, the front panel is configured by the user in order to program the execution of three sequential tasks according to the following aims [37]-[41]:

1. Set AWG, addressed by means of the Boolean control 'DSO/AWG', to generate a triangular waveform on a high impedance load characterised by 1.850 kHz fundamental frequency, 10 V peak-to-peak amplitude, and zero offset. To this end, the user chooses the action 'Write' through the 'Read/Write' control, since they need to send the configuration data to the remote instrument. The value of the control 'Wait' is set to 0 ms, requiring immediate execution.
2. Set DSO to acquire samples using 100 μs main scale time/div (i.e. 2.5 MSa/s sampling rate) and 1 V volt/div. DSO is also pre-configured to transmit a portion of the acquired record. In particular, no header for the data block is required; channel 1 is the specified data source; single-byte ASCII encoding is demanded; a portion of the acquired record, delimited by means of start and stop values, is selected; the instrument is queried to send the

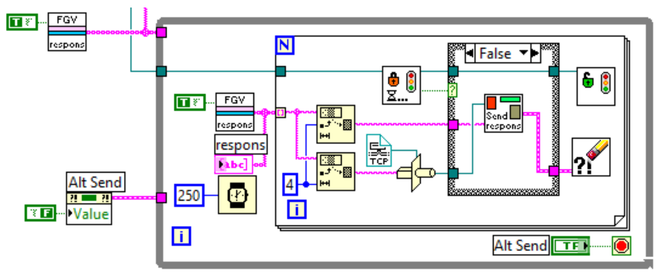
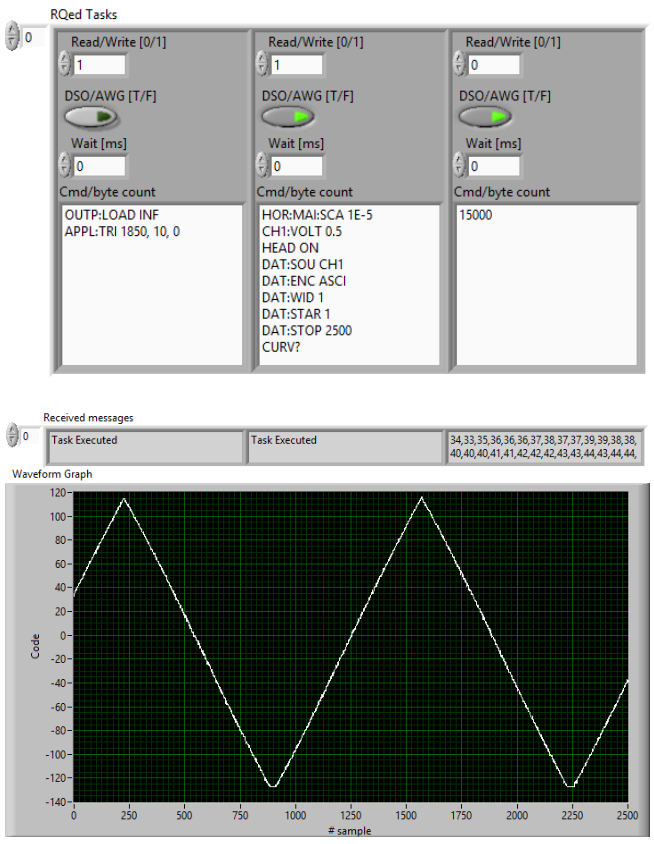


Figure 8. Block diagram of the check-out thread.


RQed Tasks

0

Read/Write [0/1]	Read/Write [0/1]	Read/Write [0/1]
1	1	0
DSO/AWG [T/F]	DSO/AWG [T/F]	DSO/AWG [T/F]
Wait [ms]	Wait [ms]	Wait [ms]
0	0	0
Cmd/byte count	Cmd/byte count	Cmd/byte count
OUTP:LOAD INF APPL:TRI 1850, 10, 0	HOR:MAI:SCA 1E-5 CH1:VOLT 0.5 HEAD ON DAT:SOU CH1 DAT:ENC ASCI DAT:WID 1 DAT:STAR 1 DAT:STOP 2500 CURV?	15000

Received messages

0

Task Executed	Task Executed	34,33,35,36,36,36,37,38,37,37,39,39,38,38, 40,40,40,41,41,42,42,42,43,43,44,43,44,44,

Figure 9. Front panel of the client VI.

selected portion with the last command, which is terminated with a question mark.

3. Collect the samples acquired by the remote DSO, which is analysing the AWG output. To this end, a byte-count value that is sufficiently larger than the expected number of bytes, to be exchanged through the GPIB bus, is specified together with the option 'Read'.

The front panel of the VI contains a waveform graph indicator showing the graphic of the waveform digitised by the DSO present in the ATE. The waveform is first retrieved by the local server through the GPIB interface, and then by the remote client via TCP/IP communication.

4. DISCUSSION

In order to effectively benefit from multithreading technology, high-level programming expertise is required both in the development and debugging phases. In particular, if the application uses the same subVI in different threads, the execution of the subVI must be re-entrant. For instance, at the debugging stage, when it is practical and useful to run multiple client applications on a single PC, the developer has to configure all client subVIs as re-entrant subVIs.

In addition, race conditions due to simultaneous requests by multiple threads of shared resources, such as the GPIB bus in the considered application, or asynchronous modifications of shared variables, must be prevented. Fortunately, the GPIB bus controller can avoid conflicts if instruments are identified by means of different addresses.

On the other hand, the use of FGV has been effective to contrast race conditions issues [39]-[41].

Nonetheless, to implement further optimisations, the programmer can configure the LabView execution system, assigning different priorities (up to six priority levels) to different threads according to their strategic role. The standard execution adopts a separate thread to run user interface activities, while the main threads are divided into daughter threads, containing separable parts of the original processing code and organised into a circular queue. Daughter threads can be retrieved from the queue manage by the execution system and run by different processors in multi-core architectures. When a thread needs to use a control, its execution is suspended, and responsibility is passed to the user interface thread, slowing down the application execution.

More specific remarks are related to the scalability of the platform and to the communication latencies. In theory, each GPIB remote controller can manage up to 15 ATEs, made up of a couple of instruments, using just the primary addresses.

Measurement tasks can be very time consuming, especially if they involve the download of arbitrary waveforms synthesised with up to 32768 samples and the retrieval of long data records acquired by DSOs. Situations like these can cause dangerous bottlenecks at the GPIB bus level for a platform like the proposed one. If all ATEs are charged with heavy tasks, which results in a long occupancy rate for the GPIB bus, the latencies observed on the client side will range from many seconds to even a few minutes. These latencies are unacceptable and are not contemplated in the proposed platform, which limits the TCP/IP communication operations to 25 s by means of a time-out control.

The platform design could be developed either by taking into account statistics related to both the average time to serve a request and the average number of concurrent connections, or by deciding to always avoid disconnections due to time-out occurrences on the client side. In the first case, as described in the present work, a strategy that leaves room for waiting requests to be served can be admitted, accepting the risk for the connected clients to undergo time-outs. In the second case, the server has to deny the acceptance of the client's request when all available ATEs are busy by sending them an invite-to-retry message.

5. CONCLUSIONS

MetroIndustry 4.0 represents the broad scenario in which the achievement of an efficient quality infrastructure lies inevitably on the ability to obtain valid data based on high-precision measurements. The role of measurement science and its application towards simulations and virtual measuring instruments is therefore counted among the newly identified focuses.

Starting from these premises, an IoT-oriented platform realised by means of a LabView application has been proposed. It uses basic TCP/IP data transfer mechanisms to allow clients to access and program remote ATEs so that they can be operated at a distance. The platform is essentially conceived as a training tool for technicians involved in remote instruments programming jobs within an Industry 4.0 environment. The core software modules of the platform have been described in detail. An example of the operations of the platform has been briefly demonstrated. Remarks related to the scalability, complexity, and responsiveness of the system have also been discussed.

ACKNOWLEDGEMENTS

The authors thank A. Tocchi and D. Grillo for their help in setting up and testing the presented platform.

APPENDIX I

A complete picture of all operations performed by the server machine is given in Figure . The server includes the parallel threads: TCP connection thread, check-in thread, management thread, and check-out thread. It also uses further measurement threads, named GPIB threads, to control the available measurement stations.

The threads exchange data between each other by means of queues implemented with FGVs. Queue structures based on FGV concepts can be uniquely accessed via the methods defined by the programmer, and they allow synchronised data exchange between threads.

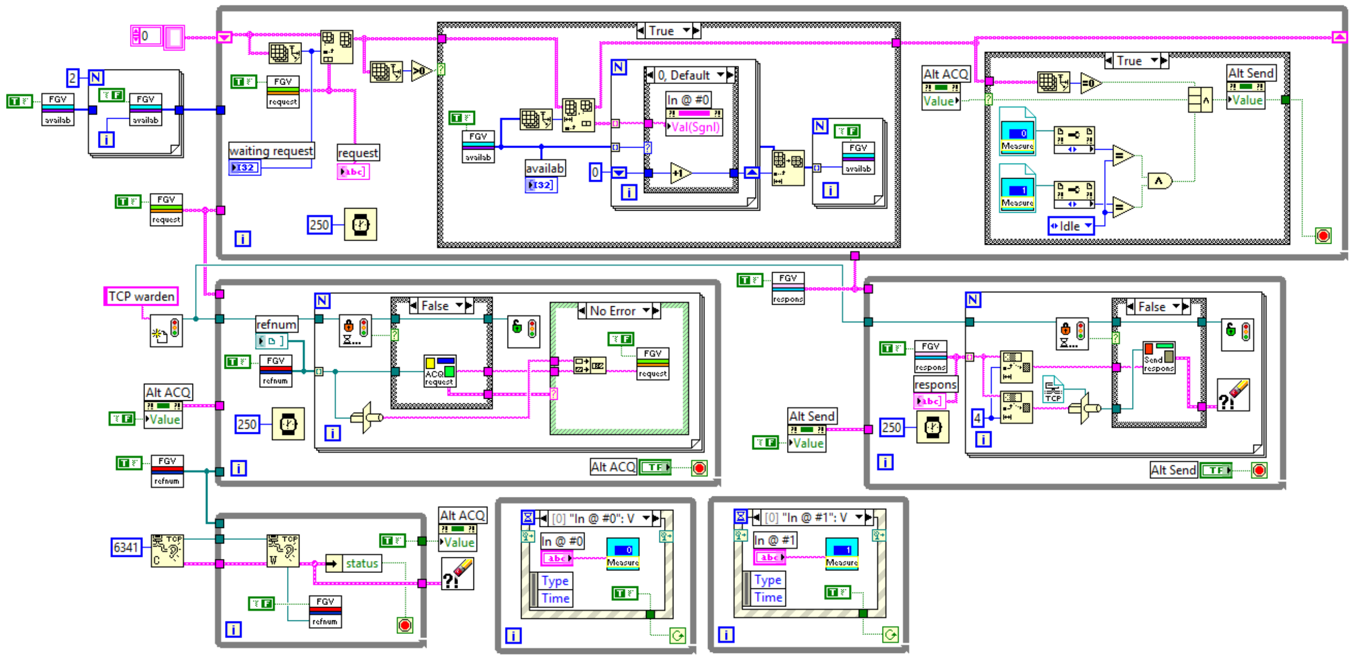


Figure 10. Block diagram illustrating the operation of the server machine of the proposed platform.

REFERENCES

[1] U.Lichtenthaler, Open innovation in practice: an analysis of strategic approaches to technology transactions, IEEE Transactions, 55(1) (2008) pp. 148-157.

[2] L.de la Torre, J.P.Sànchez, S.Dormido, What remote labs can do for you, Physics Today, 69(4) (2016), pp. 48-55.

[3] G.Andria et al., 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., 56(4) (2007) pp. 1124-1134.

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

[5] J.M.G.Palop, J.M.A.Teruel, Virtual work bench for electronic instrumentation teaching, IEEE Trans. Educ., 43(1) (2000) pp. 15-18.

[6] C.C.Ko, B.M.Chen, S.H.Chen, V.Ramakrishnan, R.Chen, Y.Zhuang, A large-scale Web-based virtual oscilloscope laboratory experiment, Eng. Sci. Educ. J., 9(2) (2000) pp. 69-76.

[7] P.Arpaia, A.Baccigalupi, F.Cennamo, P.Daponte, A measurement laboratory on geographic network for remote test experiments, IEEE Trans. Instrum. Meas., 49(5) (2000) pp. 992–997.

[8] G.Canfora, P.Daponte, S.Rapuano, Remotely accessible laboratory for electronic measurement teaching, Comput. Stand. Interfaces, 26(6) (2004) pp. 489-499.

[9] L.Benetazzo, M.Bertocco, F.Ferraris, A.Ferrero, C.Offelli, M.Parvis, V. Piuri, A web based, distributed virtual educational laboratory, IEEE Trans. Instrum. Meas., 49(2) (2000) pp. 349–356.

[10] M.Albu, K.Holbert, G.Heydt, S.Grigorescu, V.Trusca, Embedding remote experimentation in power engineering education, IEEE Trans. Power Syst., 19(1) (2004) pp. 139-143.

[11] P.Arpaia, F.Cennamo, P.Daponte, M.Savastano, A distributed laboratory based on object-oriented measurement systems, Measurement, 19(3-4) (1996) pp. 207-215.

[12] P.Arpaia, A.Baccigalupi, F.Cennamo, P.Daponte, A remote measurement laboratory for educational experiments, Measurement, 21(4) (1997) pp. 157-169.

[13] D.Grimaldi, S.Rapuano, Hardware and software to design virtual laboratory for education in instrumentation and measurement, Measurement, 42(4) (2009) pp. 485-493.

[14] M.Corrado, L.De Vito, H.Ramos, J.Saliga, Hardware and software platform for ADCWAN remote laboratory, Measurement, 45(4) (2012) pp. 795-807.

[15] U.Hernandez-Jayo, J.Garcia-Zubia, Remote measurement and instrumentation laboratory for training in real analog electronic experiments, Measurement, 82 (2016) pp. 123-134.

[16] J.Grodotzki, T.R.Ortelt, A.E.Tekkaya, Remote and virtual labs for engineering education 4.0: achievements of the ELLI project at the TU Dortmund University, Procedia Manufacturing, 26 (2018) pp. 1349-1360.

[17] F.Leccese, 'A cheap Wi-Fi instrumentation remote control platform for electric and electronic didactic laboratory', Proc. of the 17th Symposium IMEKO TC4 - Measurement of Electrical Quantities, 15th International Workshop on ADC Modelling and Testing, and 3rd Symposium IMEKO TC19 - Environmental Measurements, 8-10 Sept., 2010, Kosice, Slovakia, pp. 29-33.

[18] A.Maiti, D.Garbi Zutin, H.D.Wuttke, K.Henke, A.D.Maxwell, A.A.Kist, A framework for analyzing and evaluating architectures and control strategies in distributed remote laboratories, IEEE Transactions on Learning Technologies, 11(4) (2018) pp. 441-455.

[19] L.F.Z.Rivera, M.M.Larrondo-Petrie, L.Ribeiro Da Silva, 'Implementation of cloud-based smart adaptive remote laboratories for education', Proc. of the 2017 IEEE Frontiers in Education Conference (FIE), 2017, Indianapolis, Indiana, pp. 1-5.

[20] N.Fujii, N.Koike, 'IoT remote group experiments in the cyber laboratory: a FPGA-based remote laboratory in the hybrid cloud', Proc. of the 2017 International Conference on Cyberworlds (CW), 20-22 Sept. 2017, Chester, United Kingdom pp. 162-165.

[21] M.Poliakov, K.Henke, H.D.Wuttke, 'Prospects for constructing remote laboratories for the study of cognitive systems', Proc. of the 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), 2017, Bucharest, pp. 965-968.

[22] N.Wang, X.Chen, Q.Lan, G.Song, H.R.Parsaei, S.C.Ho, A novel wiki-based remote laboratory platform for engineering education, IEEE Transactions on Learning Technologies, 10(3) (2017) pp. 331-341.

[23] N.Valov, I.Valova, 'Drying process management laboratory with remote access', Proc. of the 16th International Conference on Information Technology Based Higher Education and Training (ITHET), 10-12 Jul. 2017, Ohrid, Macedonia pp. 1-6.
[24] C.Arguedas-Matarrita et al., 'A teacher training workshop to promote the use of the VISIR remote laboratory for electrical circuits teaching', Proc. of the 4th Experiment@International Conference (exp.at'17), 2017, Faro, Portugal, pp. 1-6.
[25] J.Zalewski, 'Cyberlab for cyberphysical systems: remote lab stations in software engineering curriculum', Proc. of the Fourth International Conference on e-Learning (ICEL2013), 8-10 Jul. 2013, Ostrava, Czech Republic, pp. 1-7.
[26] P.Di Giamberardino, M.Temperini, 'Adaptive access to robotic learning experiences in a remote laboratory setting', Proc. of the 18th International Carpathian Control Conference (ICCC), 2017, Sinaia,Romania, pp. 565-570.
[27] M.Kalúz, J.García-Zubía, M.Fikar, Ľ.Čirka, A flexible and configurable architecture for automatic control remote laboratories, IEEE Transactions on Learning Technologies, 8(3) (2015) pp. 299-310.
[28] P.Orduña, et al., An extensible architecture for the integration of remote and virtual laboratories in public learning tools, IEEE Revista Iberoamericana de Tecnologias del Aprendizaje, 10(4) (2015) pp. 223-233.
[29] A.C.Caminero, S.Ros, R.Hernández, A.Robles-Gómez, L.Tobarra, P.J.T.Granjo, VirTUal remoTe labORatories Management System (TUTORES): using cloud computing to acquire university practical skills, IEEE Transactions on Learning Technologies, 9(2) (2016) pp. 133-145.
[30] A.Chevalier, C.Copot, C.Ionescu, R.De Keyser, A three-year feedback study of a remote laboratory used in control engineering studies, IEEE Transactions on Education, 60(2) (2017) pp. 127-133.
[31] N.Wang, X.Chen, G.Song, Q.Lan, H.R.Parsaei, Design of a new mobile-optimized remote laboratory application architecture for m-learning, IEEE Transactions on Industrial Electronics, 64(3) (2017) pp. 2382-2391.
[32] P.Orduña, L.Rodriguez-Gil, J.Garcia-Zubia, I.Angulo, U.Hernandez, E.Azcuenaga, 'LabsLand: A sharing economy platform to promote educational remote laboratories maintainability, sustainability and adoption', Proc. of the IEEE Frontiers in Education Conference (FIE), 2016, Erie, Pennsylvania, USA, pp. 1-6.
[33] J.M.M.Quintero, P.E.N.Ortiz, D.M.O.Martinez, L.F.C.Alfonso, 'Low cost remote instructional laboratory for control systems courses', Proc. of the International Conference on Interactive Mobile Communication, Technologies and Learning (IMCL), 2016, San Diego, California, pp. 78-82.
[34] L.Angrisani, M.D'Apuzzo, M.D'Arco, Fast transformation for DAC parameter identification, IEEE Transactions on Instrumentation and Measurement, 55 (2006) pp. 2007-2013.
[35] M.D'Apuzzo, M.D'Arco, Sampling and time-interleaving strategies to extend high-speed digitizers bandwidth, Measurement Elsevier, 111 (2017) pp. 389-396.
[36] M.D'Apuzzo, M.D'Arco, A wide-band DSO architecture based on three time interleaved channels, IOP Journal of Instrumentation, 11(8) (2016), pp. 2-9.
[37] A.Baccigalupi, M.D'Arco, A.Liccardo, Parameters and methods for ADCs testing compliant with the guide to the expression of uncertainty in measurements, IEEE Transactions on Instrumentation and Measurement, 66(3) (2017) pp. 424-431.
[38] J.Rumbaugh, I.Jacobson, G.Booch (editors), The Unified Modeling Language User Guide (2nd ed.), Addison-Wesley, Westford MA, 2005, ISBN 0321267974.
[39] National Instruments Application Note 114, Using LabVIEW to Create Multithreaded VIs for Maximum Performance and Reliability, July 2000, Lit. Num. 341419C-01.
[40] National Instruments Application Note 160, Using LabView with TCP/IP and UDP, March 2004, Lit.Num. 342028C-01.
[41] National Instruments Application Note 168, 'LabView Performance and Memory Management', March 2004, Lit. Num. 342078B-01.
[42] BIPM - European Association of Metrology Institutes, Metrology for the digitalization of the economy and society, October 2017 [Online] Available: https://www.bipm.org (Access date: 10/10/2018).