Recent Innovations in Mechatronics (RIiM) Vol. 5. (2018). No. 1.
DOI: 10.17667/riim.2018.1/1

Overview of modern teaching equipment that supports distant learning

Csongor Márk Horváth, Trygve Thomessen
PPM AS,
Trondheim, Norway
csongor@ppm.no, trygve.thomessen@ppm.no

Gábor Sziebig
Department of Industrial Engineering,
UiT - The Arctic University of Norway, Narvik, Norway
gabor.sziebig@uit.no

Abstract— Laboratory is a key element of engineering and applied sciences educational systems. With the development of Internet and connecting IT technologies, the appearance of remote laboratories was inevitable. Virtual laboratories are also available; they place the experiment in a simulated environment. However, this writing focuses on remote experiments not virtual ones. From the students' point of view, it is a great help not only for those enrolling in distant or online courses but also for those studying in a more traditional way. With the spread of smart, portable devices capable of connection to the internet, students can expand or restructure time spent on studying. This is a huge help to them and also allows them to individually divide their time up, to learn how to self-study. This independent approach can prepare them for working environments. It offers flexibility and convenience to the students. From the universities' point of view, it helps reduce maintenance costs and universities can share experiments which also helps the not so well-resourced educational facilities.

Keywords—e-learning, remote laboratories, distant learning

I. Introduction

Laboratory is a key element of engineering and applied sciences educational systems. With the development of Internet and connecting IT technologies, the appearance of remote laboratories was inevitable. Virtual laboratories are also available; they place the experiment in a simulated environment. However, this writing focuses on remote experiments not virtual ones. From the students' point of view, it is a great help not only for those enrolling in distant or online courses but also for those studying in a more traditional way. With the spread of smart, portable devices capable of connection to the internet, students can expand or restructure time spent on studying. This is a huge help to them and also allows them to individually divide their time up, to learn how to self-study. This independent approach can prepare them for working environments. It offers flexibility and convenience to the students. From the universities' point of view, it helps reduce maintenance costs and universities can share experiments which also helps the not so well-resourced educational facilities.

The initial focus of remote experiment development was to design reliable experiments and to solve all the emerging issues such as proper software for controlling, solution for video and audio streaming, management of login of students etc. There still are new architectures designed (the most important currently is the development of remote experiments in HTML5 based environment so portable devices can play a role) but many new researches focuses on the pedagogical aspects [1]. One major component of a well-structured remote educational environment is the communication panel. These criteria can be met by adopting chat programs in remote experiments and adding forums or discussion boards to them. These functions help to establish the desired collaborative environment.

II. Common Architectures

This section is a review of the common architectures of remote experiments frequently applied so far, focusing on their basic elements [2]. These elements are the user interface, the web server, the lab server and the controllers. This partitioning of the architecture helps the better understanding of the working principle.

A. Software components

User interface is a virtual environment that runs in the user's web browser and handles all administration process. Usually, it requires server-side programming languages (e.g. PHP, ASP, JSP) and a graphical user interface (GUI) to retrieve user data from the database. The GUI is built from plugins and animation technology embedded into HTML code to resemble the real lab workbench (e.g. Flash, ActiveX controls, Java Applets). Other scripting languages, such as JavaScript can be also used to add server independent interactive elements to the webpage. Another language is AJAX, it retrieves information from the server without the need of web page refreshing. Both of these languages are recently being utilized expansively with the spread and recognition of HTML5 technology. Another solution is to use a software installed on the user's computer and connected to the database server. However, from the user's point of view, this design lacks the simplicity of a web page being utilized.

The web servers are hosting the web sites and the database, running on dedicated server computers. The most commonly used servers are Apache and Microsoft IIS and the most commonly used databases are MySQL, Microsoft SQL and Oracle. The basis of its operation is the following:

the user sends requests through the web page and these requests are forwarded to the lab server from the web server in the form of XML messages through TCP/IP over HTTP layer [3].

B. physical instrumentation

The lab server is the server-PC that is directly connected to the physical instrumentation of the experiment and hosts the control software. The control is on the user's requests. The lab server software can be written in multi-purpose programming languages such as C, C# and C++ or proprietary graphical programming languages can be applied such as LabVIEW and MATLAB. Moreover, it can be a language attached to the proprietary controller (e.g. a PLC). The control software can communicate with the controller and instruments via multiple ports such as USB, RS- 232, Ethernet, serial port, parallel port and General Purpose Interface Bus (GPIB-IEEE- 488.2).

The controller is a programmable device and therefore can be applied in different experiments. The most commonly used devices are Programmable Logic Controller (PLC), Programmable Logic Device (PLD), Field Programmable Gate Array (FPGA), microcontrollers and Complex Programmable Logic Device (CPLD). The connection between the controller and the controlled objects can be designed via connectors, A/D, D/A converters, I2C-based electronic boards etc. Furthermore, in some applications a „relay switching matrix is used to switch and route the connection between instruments and experiment elements" [4],[5]. In other examples, the utilization of a measurement card can be satisfactory like in real-time image processing experiments [6]. In many cases, LabVIEW with commercial measurement platforms such as NI ELVIS and CompactRIO has facilitated the design of a remote laboratory.

C. Webserver on LabVIEW and MATLAB

As it was stated before, two outstanding software for lab servers to be written in are LabVIEW and MATLAB. LabVIEW is based on a graphical programming environment, containing graphical icons called virtual instruments (VIs) which imitate the physical instruments. It is suitable for developing, testing, controlling and executing measurement. LabVIEW provides a built-in web server allowing through the Remote Panel module the VIs embedment and direct control from a web browser. The necessity of low-level TCP programming is eliminated with a technology called DataSocket which allows multiple data exchange among users and applications. The drawback of this technology is that web publishing tool requires the LabVIEW run-time engine on the user's computer. MATLAB is ideal for intensive numerical calculations, algorithm development, data visualization and analysis. MATLAB enables web access capability of the applications by toolboxes, offering for interoperability and data exchange with other GUI applications [7],[8]. Thus, there is no need to have a local version of MATLAB or be familiar with its programming.

It is obvious that a hybrid architecture utilizing LabVIEW and MATLAB can give high efficiency and performance. Signal acquisition and GUI is designed with LabVIEW and numerical calculations and signal processing can be facilitated by MATLAB. There are many ways LabVIEW helps this integration with MATLAB (Active X automation technology, DLL and COM technology and MathSricpt RT module). In addition, multi-purpose programming languages can also be utilized to design lab server programs like C, C# and C++. Moreover, hardware drivers can be built using LabVIEW and MATLAB in form of DLLs and be deployed in multi-purpose programming language codes. Although, these solutions require dedicated experienced software engineers.

D. Client – Server communication

Client-server communication can be established by Common Gateway Interface (CGI), ActiveX and Java Applets, Rich Internet Applications (RIAs), Asynchronous JavaScript and XML (AJAX).

CGI is a standard programming interface, defining the HTTP communication between the web servers and the external programs. The provider server runs the main body of the CGI software, only input and output forms sent across the web components, eliminating the need for external software components to work in all browser and operation system [9]. However, there are multiple security risks. First, it can provide web access by unauthorized person, second, a bad CGI script written or uploaded is also a potential risk. Although there are solutions to it but CGI offers a basically slow design because a new process has to be started every time a HTTP request comes in and the database connection has to be reopened for the next instance.

ActiveX controls work only in Windows based environments and can be written in languages only supporting COM component development (C/C++, Java, .NET, LabView etc.). ActiveX controls are common in LabVIEW built-in web server applications so front panel can be embedded and interactive content can be displayed in a browser. ActiveX controls have full access to Windows operating system and even though there is a registry of digital signatures created to verify ActiveX controls, this is a potential risk. Therefore, users have to know what is the purpose of the control and check if it works accordingly.

Java Applets are more secure; each Java application can be customized concerning what actions it is permitted to control. Moreover, unsigned applets run in a protected area in the memory called sandbox. This model does not allow these applets to interfere with other processes or each other. Signed applets can be permitted by the user to run out of the sandbox. In addition, there are security libraries providing authentication, code signing, permissions, encryption and certificate implementation. Java Applets are written in Java programming language which is transformed into byte code so it is platform and web browser independent. The byte code is loaded and compiled by the client and executed using a Java Virtual Machine (JVM). It restarts every time the browser starts afresh. There are different versions of Java Runtime Environment (JRE), therefore users might need to download a new version if they want to be sure that the most recently created applets can run in the environment.

In addition to security issues, there are other differences between ActiveX and Java Applets. Every time Java applets are launched, they are downloaded while ActiveX is downloaded only once and are updated automatically. Moreover, ActiveX is faster because it is based on 32-bit code against the Java Applets' 8-bit code. But the function of the two technologies are similar: they are downloadable plugins from the web server so that on the computer a rich dynamic content can be realized.

Adobe Flash is the most commonly used RIA, it integrates graphics, animations, multimedia and interactivity into web pages. The required plugin has to be downloaded only once which reduces the load time of the application, the load on the server and bandwidth requirements. For security reasons, this technology is also based on the sandbox working principle. With an increase in the number of students using portable devices (tablet, smartphones), its usefulness decreases. This is the reason why HTML5 based technology starts to supplant designs utilizing Flash. Architectures based on AJAX and JavaScript technologies allows rich and dynamic web applications to be created without the need for additional plugins. AJAX enables JavaScript to directly communicate with the web server using XMLHttpRequest object. Instead of reloading the whole web page, it allows the web browser to update only the changed data. Apart from XMLHttpRequest object, all the technologies utilized by AJAX such as HTTP, XML, JavaScript etc. is supported by all web browsers by default. These technologies are for requesting, retrieving, converting and presenting the data on the web page. The major drawback of AJAX is, that does not provide video and audio capabilities.

E. *Remote Desktop sharing*

Remote desktop sharing is fundamentally different from the previous architectures because it does not require a middleware to be constructed, no virtual panels has to be created that can be accessed from a web page. Instead, the lab server connected to the experiment and the control software running on it can be accessed (viewed and controlled) directly. Restrictions can be deployed such as the remote user accessing the lab server computer has no authorization to modify the configuration, to shut down the computer or to view and to reconstruct the file architecture at the hard disk. The two common mentioned drawbacks are the lack of security and that it is bandwidth intensive. The latter with the development of internet technologies is longer an issue, particularly not in developed countries.Security issues can be solved with an extra security layer with stronger encryption, Secure Shell (SSH) and Virtual Private Network (VPN) tunneling are both proper solutions [10].

The typically adopted remote desktop sharing technologies are the Virtual Network Computing (VNC) the and Remote Desktop Protocol (RDP). VNC is cross- platform, based on image transferring and allows multiple connections. VNC only supports video transmission. RDP technology was originally developed by Microsoft but there are RDP servers and clients for other operating systems. Instead of image transferring, it utilizes data transferring which makes it a faster solution than VNC. However, file transfer between the client and the remote server and multiple access is not supported. RDP is by default more secure than VNC and it has built-in bandwidth reduction features. Moreover, it supports video and audio signal transmitting as well.

Overall conclusion is that one of the best design is to use plugin based web pages. However, with the emergence of HTML5 based technologies AJAX and JavaScript can be utilized in remote desktop solutions (either VNC or RDP based) so that clients do not need plugins in their web browsers such as *Apache Guacamole* (supporting both VNC and RDP), *noVNC*, *ThinVNC* and *FreeRDP-WebConnect*. Also web browsers nowadays have a much more frequent release cycle which means that a new technology can be faster adopted. The previous examples are all open-source projects but naturally, there are proprietary software such as *Thinfinity* (RDP based), *Remote Spark* (based on RDP but supports VNC as well) and *Ericom AccessNow* (RDP based) which may not be affordable in an educational facility. Beside the provided support, they implement solutions to low-bandwidth and high-latency networks and further encryption solutions are also offered.

III. Remote Laboratory Experiments

A. *Managed and unmanaged experiments in WebLab-Deusto*

The aforementioned alignment of structures is noticeable in the WebLab-Deusto environment [11]. In this context, remote laboratories are investigated how they can be integrated into the WebLab-Deusto system. This is basically a remote experiment management system with many features adopted from learning management systems (LMS). The most important aim of this project is to reduce duplicity among remote experiments and their features and therefore cut the required workhours invested in a newly designed remote laboratory. The result is that many universities can share laboratories throughout the world and can be accessed in a uniform system no matter where the user requests access from.

In WebLab-Deusto there are managed and unmanaged experiments. Managed experiments are composed of two basic components. The client runs in a web browser and can be based on Java Applets, Adobe Flash etc. or it can be a separate software that has to be installed on the client side. It is responsible for interaction with students and for proper translation of user requests to the server.

Server is connected to the experimentation devices (such as electronic boards, robotic arms etc.) located in the university. The code (e. g. Java, .NET, C++, Python, LabVIEW) running on the server receives requests from students and controls the interaction with the hardware and ideally helps avoiding harmful requests.

This is a clear client-server architecture and WebLab-Deusto supports the integration of this type of remote experiments by offering both client and server side libraries and software components that will be in the middle of the structure. The aim of these modifications are communication handling, avoiding access to be given to unauthorized students and logging all the commands sent and received from the experiment (user tracking). Moreover, Secure Sockets Layer is provided with the help of the software components in the middle of the architecture. Overall, security and tracking issues are handled by the WebLab-Deusto platform that is why it is called managed.

An example based on FPGA experiment is presented above. After a student has logged in, waited in the queue in case other students were using the experiment and sent a file that will be programmed in the FPGA the virtual control panel appears. To create it in the WebLab-Deusto environment, several libraries can be utilized for Java Applets, Adobe Flash and JavaScript, so the client only needs to perform calls to a common API to send and receive commands and files from the server [11]. As a result, switches, buttons and the screen of the webcam appear in the web page (Fig. 1.). Also in the background it is defined that after a button has been pressed what command has to go through the API to the server.

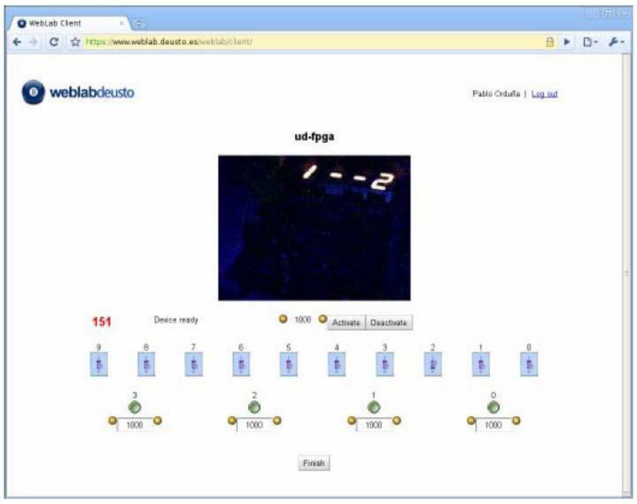


Fig. 1.: FPGA experiment in the WebLab-Deusto platform [11]

On the server side another set of libraries can be utilized such as C, C++, Java, .NET, Python or LabVIEW. Each time a command or file is sent to the server, it enables the proper response to be displayed in the web page. Although an additional layer is utilized in the architecture, depending on the configuration it does not even add latency to the system or just a little bit of delay is caused. This approach facilitates the development of secure remote experiments that deploy web applications. Furthermore, all the communication is managed through HTTP/HTTPS ports, therefore no issues will emerge concerning institutional firewalls or HTTP proxies.

Another example is the integration of the VISIR project [12]. The client side is developed in Adobe Flash and WebLab-Deusto provides a library for that technology. Therefore, communication layer was replaced by the WebLab-Deusto library. Moreover, although VISIR project has its own user tracking method, it was replaced because WebLab-Deusto's solution is more fine-grained, provides more precise and detailed information about users' activity.

Unmanaged experiments in the WebLab-Deusto environment require no skills in web development. Authentication, authorization, scheduling, basic load balancing, user tracking are still managed by the platform but the client contacts the server on its own without additional server side libraries provided by WebLab-Deusto. The architecture (Fig. 2.) is based on VirtualBox which is a virtual machine (VM) provider. Whenever a user requests an experiment, a virtual machine is loaded. Access is given to the user by a password generated at the initialization of the VM which is shared with the user through WebLab-Deusto. Access is performed through VNC and RDP with SSH and the client needs an applet within the web browser or a specific client. More precisely, VNC and SSH are supported in Linux VMs and VNC and RDP are supported in Microsoft VMs

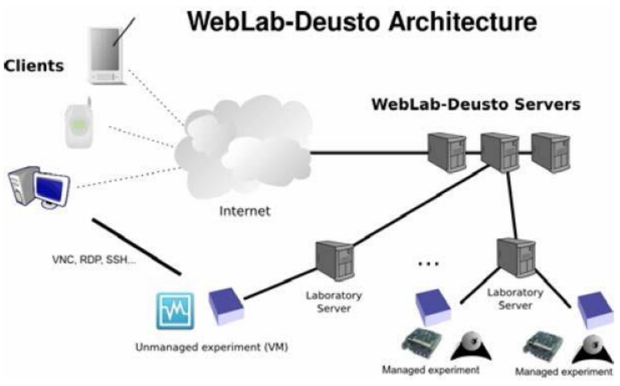


Fig. 2.: WebLab-Deusto architecture [11]

The importance of the unmanaged experiments is that it allows remote laboratories utilizing any desktop technology to be integrated in the WebLab-Deusto platform. Moreover, in case of for example a LabVIEW based remote experiment, developers need to be able to use LabVIEW but it is not required of them to develop in web environment. (In case of a remote experiment utilizing not just LabVIEW but its additional module Remote Panels, integration into WebLab-Deusto becomes much more complex because of authentication issues [12]. There is a solution but it inherits the original restrictions of Remote Panel technology such as it cannot run on mobile devices unlike any other remote experiment embedded into the WebLab-Deusto- platform.) However, there are security issues in comparison with managed experiments because VNC is not considered secure by default, althogh RDP is secure by design. Communication is not handled through WebLab-Deusto (except for the generated password being securely sent through it), therefore they are managed directly which results in network problems such as firewalls blocking the ports. Also universities utilizing HTTP proxy will not be able to be integrated in the platform because these connections are forbidden. Moreover, there is no solution for user tracking because WebLab-Deusto is not part of the main communication. But these drawbacks can be bearable in light of the fact that the time and effort spent on developing the remote laboratories can be significantly reduced.

B. *A remote laboratory utilizing LabVIEW Remote Panels*

There are examples of using LabVIEW in remote laboratories without the Remote Panels module such as in the VISIR project [13]. In this example instrumentation server is developed in LabVIEW and the user interacts with an Adobe Flash environment in a browser which is connected to the measurement server developed in C++.

With Remote Panels, the locally created virtual instrumentation can be published to the web. LabVIEW comes with a web server that automatically deploys the application and on the client side the user can see the virtual instrumentation. This makes LabVIEW an efficient software to develop remote laboratories such as MIT iLabs [14]. Furthermore, working in LabVIEW environment is easier than writing codes with multi-purpose programming languages but designing remote experiments is relatively even easier than using any other solution (except for remote desktop sharing). Although this tool has its disadvantages. First, users have to install a plug-in in their web browsers, second, to some extent it lacks security. For example, authentication is limited, there is no support for firewall restrictions and HTTP proxies. And third, it is not available for mobile devices such as smartphones or tablets because it does not support HTML5. And last, not all the web browsers support the plug-in that uses the LabVIEW runtime.

Optional authentication is available, the server defines an algorithm to establish which passwords are paired with which usernames and based on this setting only the right user is able to access the system. Moreover, it can be defined if a user can have an „only read” access or full control of the system.

The plug-in during its loading establishing a new TCP connection to the LabVIEW server [12]. Transport is applied through plain TCP pocket instead of HTTP and it uses the same port as the LabVIEW web server so no additional firewall management is needed. However, it is not possible to deploy LabVIEW in a server that runs any other web server such as Apache or IIS with other web applications. This problem can be circumvented by using a different, non-standard port but then firewall management problems emerge.

There is a newly developed tool by National Instruments addressing the listed problems. It is called LabVIEW Web UI Builder; its aim is to publish web interface but it enables a higher level of decoupling of the final user interface from the LabVIEW runtime. This design is a little bit closer to create entirely web based remote laboratories. This solution requires Microsoft Silverlight which is a competitor of Adobe Flash and Java Applets. Although Silverlight is an existing plug-in and therefore is no need to license it from LabVIEW, it still has its drawbacks. It is not supported by Linux and smart devices and also this development is not in line with current trends like „clientless”, more precisely plug-in free HTML5 based solutions which is suitable for mobile devices.

IV. Remote Desktop Sharing Technologies

Based on how users can access the remote experiment, it has two types [15]. First type is when users control the experiment through a virtual instrumentation or virtual reality environment. The other basic architecture allows the user to directly control the remote physical devices of the experiments by granting remote access to the control software installed on the computer which is connected to the experiment. These solutions are based on the so-called remote desktop technology. These designs utilize either Virtual Network Computing (VNC) or Remote Desktop Protocol (RDP).

Remote desktop solutions provide direct access to lab servers, users directly see GUIs of the experiment controlling software. The need for intermediate web-based solutions is eliminated. They are two basic approach to remote desktop designs. First, the user/client in order to access the remote/virtual desktop has to install a software. The other design only needs a web-browser and the latest solutions are based on HTML5 which requires no additional plug-ins to the browser. Currently these systems offer the most convenient user experience, requiring only a widespread web browser supporting HTML5 (e. g. Mozilla Firefox, Google Chrome).

A. *Virtual Network Computing (VNC)*

VNC is a platform independent thin-client technology because it works at the frame buffer level (RFB is a simple remote frame buffer protocol). RFB protocol allows the server to update the frame buffer displayed on the viewer. Thin-client in a short summary is a technology that enables the user (client) not to work with a traditional computer integrating hard disk and powerful CPU. All the computing capacity is available at the server, it basically does all the work and the thin-client that the user works with is only for displaying the content of the powerful server and to handle keyboard and mouse events. Hence, allowing the client to remotely control the desktop.

The VNC technology allows various encoding schemes for the pixel data and it results in huge flexibility concerning network bandwidth, client drawing speed and server processing speed [16]. However, all clients and servers must support the so-called raw encoding because this is the lowest prevailing denominator. The rectangle is a pixel data transported in left-to-right scanline manner. According to the abilities of the server, the thin-client and connection between the two, other encoding types can be deployed.

Another type is the copy rectangle encoding. The client already has the pixel data, the server simply sends the reference X, Y coordinate to start copy the pixel data for the rectangle. This saves bandwidth and require few byte data transfer to perform such operations. VNC also features adaptive update control, when the server can define which part of the screen to update and what encoding.

Since VNC is open-source, many software are based on it with special features, developments and optimizations. The main field of these developments are encryption and compression. For example, developers can set an optimal relation between compression, network availability and client side computing power. If the goal is to create an architecture that consumes low level of network resources, it is possible to deploy a high level of compression on the server side. On the other hand, decompression on the client side requires higher level of computing resources. Some of the developed software are TightVNC, UltraVNC, RealVNC, noVNC, ThinVNC and Apache Guacamole.These last three are HTML5 based, all the former software are either need a plug-in (e.g. Flash and JAVA) or a whole independent program to be installed on the client side.

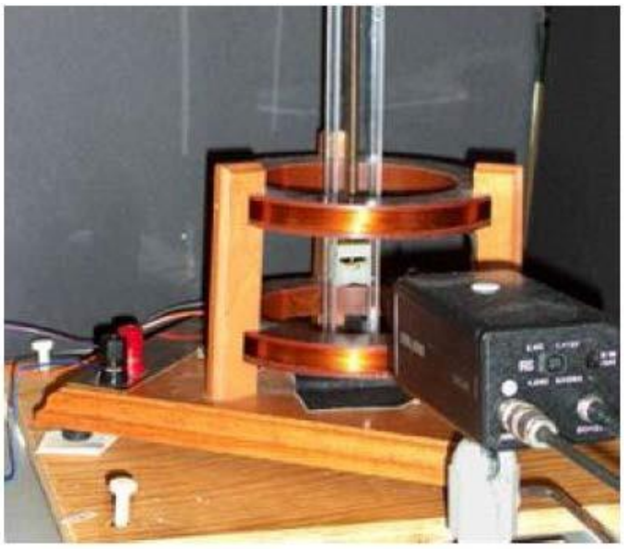


Fig. 3.: Force on a Dipole Experiment at MIT [18]

Furthermore, researchers can develop solutions for a given project. An example of this is the idea which splits the frame into high and low motion parts [17]. First, a Dynamic Image Detection Scheme is proposed which automatically detects the high motion part of the frame with only 1% of extra CPU load. After deciding of a frame part whether it is high or low motion, the system sends the frame to the client but the high motion parts are first encoded with a H.264 encoder. It is obvious that many changes can be applied to the VNC source code, therefore many utilizations can be found. One of them is distance learning and experimenting.

An example of VNC utilization is „The Force On A Dipole Experiment" [18] which is part of MIT's iLab Project (Fig. 3.). The experiment „consists of a small magnet suspended vertically by a spring in the center of two horizontally mounted coils. An electrical current may be directed through the top or, through both coils. A video camera is stationed to observe the position of the magnet."

With a LabView application, users can change the amplitude and frequency of the current and specify which coil the current is directed to. The program displays the input current, the top coil current and the position of the magnet (Fig. 4.). This experiment set-up is augmented with a software package called Project Wonderland. With the help of this software 3D virtual environment can be rendered. Wonderland supports VNC and after it is installed on the client side, it can establish connection with the server which is in the remote lab and where LabView control software and Wonderland server side is installed. This way, student can remotely control the LabView environment which directly controls the physical experiment. This system allows multiple clients to connect simultaneously to the experiment.

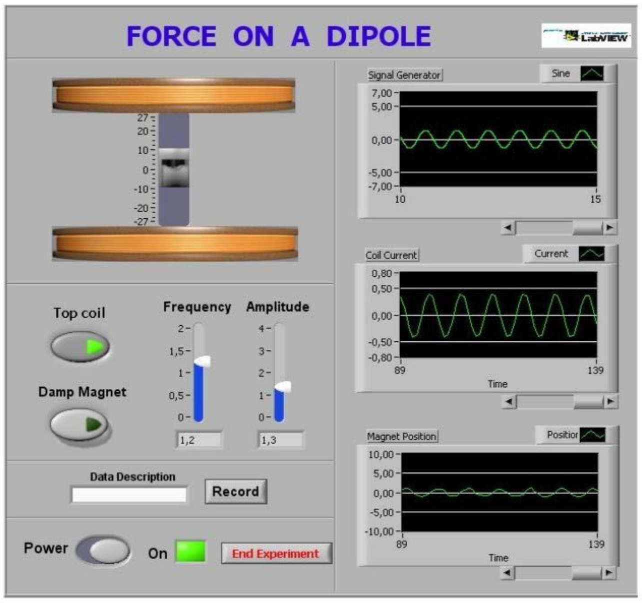


Fig. 4.: LabVIEW front panel for Force on a Dipole experiment [18]

Another example is presented from the University of Technology, Sydney (UTS) [19]. Currently, there are five different experiment equipment at UTS (Embedded Operating System Experiment and FPGA Experiment for Computer Systems Engineering students, PLC Experiment for Mechanical and Mechatronics Engineering students, Loaded Beam Experiment for Civil and Construction Engineering students (Fig. 5.) and Coupled Water-Tanks for Mechanical Engineering Students). For each experiment, basic architecture is similar but the interfaces and access mechanisms are different. For example, Coupled Water-Tanks experiment utilizes a LabVIEW application, for industrial PLCs a Windows based development environment is applied.

The main design goal was to manage concurrent and flexible access to the laboratories. The former was achieved by the so-called Arbitrator software that was developed for this goal. It authenticates requests for connection to labs and then allocates the equipment if it is unused. If it is occupied than a queuing mechanism manages the requests. When a student finishes a session, the Arbitrator reclaims the apparatus and it changes its state to unoccupied and allocates the lab among the pool of free devices so it can be accessed for the next session.

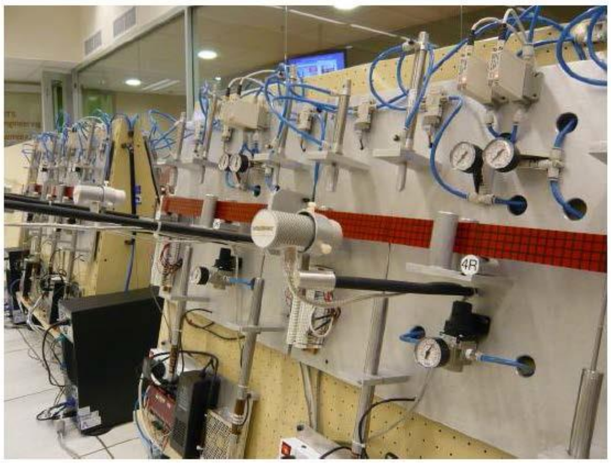


Fig. 5.: Physical structure of the Loaded Beam Experiment [19]

The goal of flexible access was reached based on virtualization. Most of the laboratories requires special proprietary software like LabVIEW or a tool to construct PLC programs. The aim is to somehow relieve the students of the task of installing licensed products. Even if the student is given the proper permission to install the software, it gives an additional level of complexity to the process. It is preferable if the students' only challenge is to successfully perform the laboratory exercise. Virtualization technologies (in this case VMWare was utilized) allow the university to equip every experiment with a lab server on which virtual machines can be created with the required software installed on them. When a student logs in and the required experiment is allocated to him/her, the system gives the IP address of the virtual machine connected to that experiment. Then students can use the installed software and control the physical experiment from their computers via VNC.

One major drawback of this architecture (Fig. 6.) was the lack of supporting multiple access to the experiment. One virtual machine is dedicated to a single user after login. Although Arbitrator grants multiple access to a lab, it is only enough for the student to watch the video stream (because it operates through the web server). It means that while one student controls the experiment, all the other logged in students can just observe. It damages the desired student-student and student-professor interactions and teamwork.

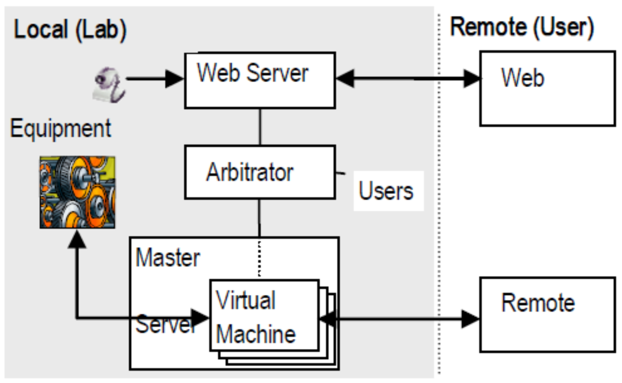


Fig. 6.: Original architecture [19]

To solve these issues, a new architecture was developed (Fig. 7.). It is based on VNC but an altered version of it called AVNC. Although VNC protocol supports multiple access by default, AVNC's great feature is that it is possible to remotely display not the whole desktop but just one chosen software. Although during developing this architecture, its multiple accessibility was its main appeal. AVNC can be embedded directly into a web interface so users are in no need of an installation, they can simply have access using their web browsers (with additional plug-ins). The only installation needed is on AVNC's server side, it has to be installed on the virtual machine. When it is turned on, the VNC server should be automatically started as well.

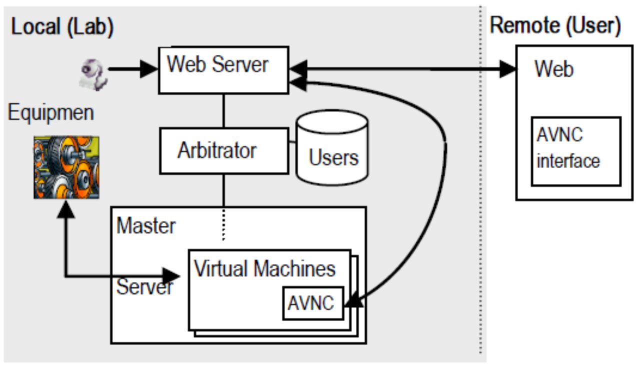


Fig. 7.: New architecture [19]

A chat program was also embedded in the architecture. Not only does it help students to discuss the experiment, it enables tutors to help answer questions. First, when students are not yet logged in the experiment, tutors describe the experiment and the curriculum connected to it. Second, during experimentation, tutors can „walk around", observe the students' work and help them. This assistance can be initiated by the student: he/she can ask for help which results in displaying a message to the tutor. Also teachers can initiate assistance if students are not able to continue the experiment. Furthermore, tutors can even take control of the experiment.

Mulfari et al. **Hiba! A hivatkozási forrás nem található.** described a VNC based system for supporting assistive technology in cloud computing. It is based on virtualization and utilizes noVNC which is a HTML5 remote desktop client proxy/web application. It offers an open source Python Websocket proxy server and JavaScript libraries to create the HTML5 VNC client web application. The architecture is discussed in details in the Apache Guacamole section, because noVNC can be replaced with Guacamole and the same researchers designed both systems with the only difference being Apache Guacamole deployed instead of noVNC.

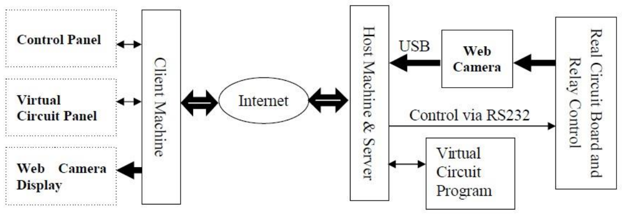


Fig. 8.: Architecture of the system utilizing LabVNC [21]

Another utilization example of the VNC technology integrates LabVNC [21]. On Fig. 8. is shown the system architecture. LabVNC is in a web server of the host computer and users can connect to it with either a Java-enabled (plug-in required) web browser or with a dedicated software previously installed on the client's side. Login requires a personal IP address and password.

Once a student is logged in, he/she can set the desired parameters using the virtual panel. This commands are forwarded to the server connected to the plant. Then commands are processed: first, the parameters sent are transmitted to the virtual circuit. Second, signals are sent to a single chip via RS-232 serial communication port. This chip controls a number of relays which are connected to the actual physical circuit board. Based on the received command, components of the real circuit board are manipulated by the relays.

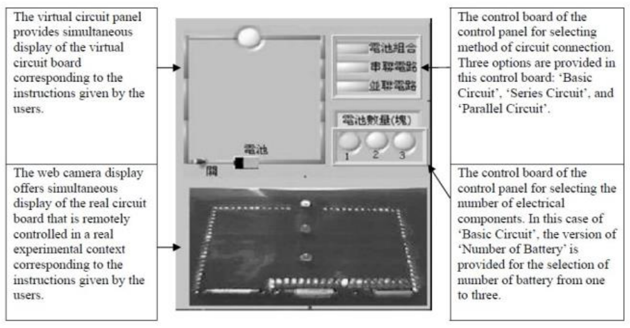

The virtual circuit panel provides simultaneous display of the virtual circuit board corresponding to the instructions given by the users.
The web camera display offers simultaneous display of the real circuit board that is remotely controlled in a real experimental context corresponding to the instructions given by the users.
The control board of the control panel for selecting method of circuit connection. Three options are provided in this control board: 'Basic Circuit', 'Series Circuit', and 'Parallel Circuit'.
The control board of the control panel for selecting the number of electrical components. In this case of 'Basic Circuit', the version of 'Number of Battery' is provided for the selection of number of battery from one to three.
電池組合
串聯電路
並聯電路
電池數量(塊)
電池
1
2
3

Fig. 9. Interface of the LabVNC based system [21]

Results generated by the virtual circuit board are sent to the server. Then the panel of the virtual circuit generates the proper virtual computer graphics and this panel is available to the student via LabVNC. The results generated by the real circuit board are transmitted to the student on video by the web camera which is connected to the server by USB ports. This way students can compare the results of the virtual and the physical circuit board (Fig. 9).

Security issues solved in [22], what is a cloud based system and based on SSH tunnelling (Fig. 10.). SSH process run on both server and client side. A client side SSH sends data through SSH port to the SSH server and vice versa. The configuration file of SSH is modified so login can be managed without password. Cloud controller generates a key pair, it is downloaded to the SSH client and only users who are given this key can access the virtual instance.

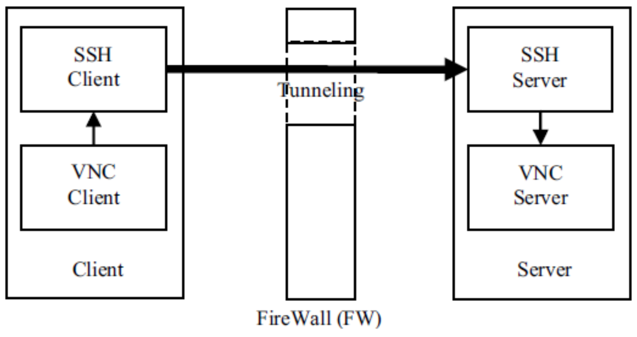


Fig. 10.: SSH tunneling [22]

An interesting project that is still worth mentioning a VNC based remote desktop control for smartphones running on Android operating system by Shisode and Dotre [23]. In this architecture, smartphone is the client and the remote computer that is accessed is the server. A VNC server must be installed on the computer and a viewer is required to be installed on the mobile. The client is programmed using the Android system, the remote desktop server is implemented in Java. For successful access, the user must know the IP address of the computer he/she wants to remotely control. Originally, the system works in Wi-Fi range but can be expanded with a cloud-based environment so that the mobile can truly be accessed from any location. Several functions have been developed such as keyboard shortcuts, file transfer protocol etc.

B. Remote Desktop Protocol (RDP) based examples

RDP technology is detailed in Section 2. An RDP based solution is presented in [24](Fig. 11.). On the client side, web based visualization of the remote experiment takes place. On the web server, the learning management system is stored with the database of users' profile. The next component is the laboratory server, each remote experiment has one and is in connection with the measurement server with the help of the so-called Bridge Service. The measurement server contains the control software which in this case is LabVIEW and the terminal server which allows remote desktop access. The last component is the plant itself connected to LabVIEW via GPIB-IEEE-488.2 which is responsible for the controlling of the physical devices. With the help of remote desktop protocol, the existing instrumentation was successfully integrated in a remote laboratory structure without the need to develop a whole new software for web-based access. Only an RDP terminal server was installed on the measurement server. On the client side, a Java-enabled web browser is enough for remote access of the experiment. The remote desktop technology that was utilized is based on a software developed by the authors called Laboratory applet. The original RDP software that was proposed turned out to be too bandwidth consuming (ProperJavaRDP applet, as its name suggests, a Java environment must be installed in the browser in the form of a plugin).

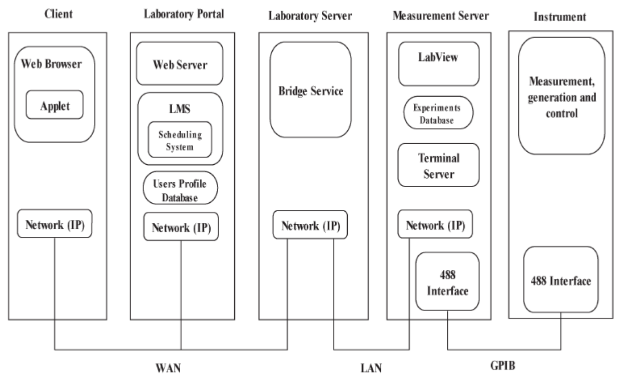


Fig. 11.: RDP based architecture [24]

Another example based on RDP [25] integrates an ActiveX module into the client's web browser called Terminal Server Advanced Client (TSAC). It is released by Microsoft and can be run in Internet Explorer. It is a relatively old architecture from 2006 but apart from the ActiveX module that has to be installed in the browser, it offers quite a smooth and convenient experience for the student. Also, the development of virtual instrumentation in the web environment is not necessary because the LabVIEW control can directly be accessed.

C. TeamViewer and Examples if its Utilization

TeamViewer is a VNC based proprietary solution for remote support, access, control, online meeting, conferences and file transferring. Also, for non-commercial use it can be utilized according to the company's policy (some of the features are restricted). It has built-in features like auto configuration and simplified authentication, built-in security, chat and can connect to computers behind firewalls. It can automatically re-establish a connection in case of network malfunctions. To connect to another computer, TeamViewer has to be running on both computers (server and client as well). Once it is started on a computer, it generates an ID and password and for the remote user to access and control this computer, he/she must have the ID and password, otherwise remote access cannot be established. The password is generated every time TeamViewer is launched, the ID remains the same, it acts like a mobile phone or an e-mail address for the computer. It supports remote access from smartphones and tablets as well as web browsers and a dedicated TeamViewer application. It can be operated as Windows system service what enable reachability even before logging in to the Operation system, furthermore cross-platform connections are also possible [26]. Moreover, based on the available bandwidth, quality and speed can be adjusted.

Crainic and Preitl [27]utilize TeamViewer in a remote laboratory where the control of a robotic arm is practiced. The laboratory equipment integrates an RV-2AJ robotic arm (Mitsubishi Co.), a CR1-571 controller, a compressor with a minimum pressure of 8 bars, a computer as a server where TeamViewer is installed on and a camera for observing the experiment. Another example is presented in [28]. It integrates an NI Smart Camera, PLC and a KUKA robot. The system helps students to viewing the exact status of the robot during practice without the need to be present in the laboratory. Fig. 12. presents the architecture of the remote experiment. There are three main components: the remote computer of the user, the KUKA controller and the central computer/local server. At their computers, students can set the parameters of the experiment, watch and control the experiment process utilizing TeamViewer while the experimental data can be gathered by the user. KUKA controller receives control signals from the central computer through the PLC and controls the robot accordingly. The central computer is responsible for managing the experiment and TeamViewer is installed and run there. Moreover, it hosts the video monitoring camera system and a developed human- machine interface.

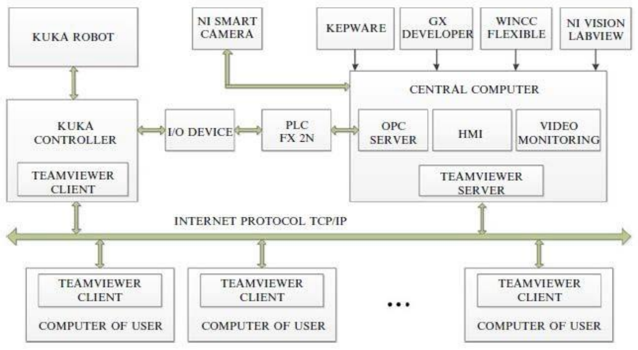
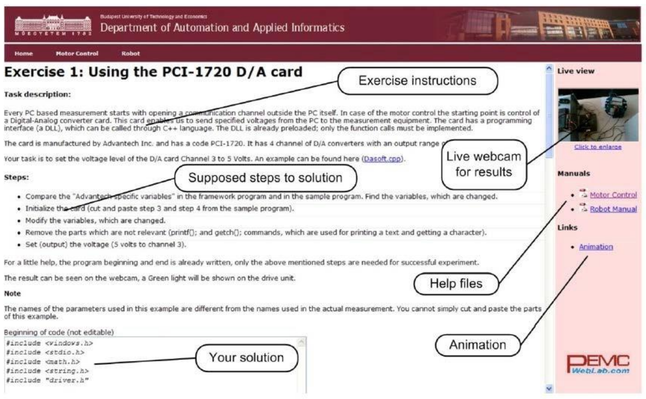


Fig. 12.: Architecture of the KUKA robot experiment [28]

V. A SERVO MOTOR CONTROLLING REMOTE EXPERIMENT

An example of remote experiments at the Department of Mechatronics, Optics and Mechanical Engineering Informatics at the Budapest University of Technology and Economics (BUTE) is aimed at allowing students to remotely control ethologically inspired robot [29],[30] and a servo motor [31]. The basic elements of the architecture are the following: a D/A card which provides the analog signal from the PC to start and stop the motor and it also provides the reference signal. Another card is needed as well (A/D card or counter card) which receives the signal of the encoder and forwards it to the PC (this is how the position of the motor can be defined). A real-time clock is also necessary for timing the sampling and the scheduling of executing the tasks.

The system is web based where students can type in their programming codes in C++. The framework is given; students have to write the variables in the code. For this, there is a motor control manual available at the web page (Fig. 13.) and also students learn about the system during lectures and they can download animations explaining the theoretical background [32],[32](as shown in Fig. 13). The students design a controller and analyzed its operation by integrated control and circuit simulation [34]. Finally, they write a simple controller code. Once the code is typed in the proper field, users have to click Upload. If the compiling is successful, the results are presented at the web page and can be downloaded as well. On the web page a web camera image is available so when the motor is under experiment, it can be tracked visually.



Exercise 1: Using the PCI-1720 D/A card

Task description:

Every PC based measurement starts with opening a communication channel outside the PC itself. In case of the motor control the starting point is control of a Digital-Analog converter card. This card enables us to send specified voltages from the PC to the measurement equipment. The card has a programming interface (a DLL), which can be called through C++ language. The DLL is already preloaded; only the function calls must be implemented.

The card is manufactured by Advantech Inc. and has a code PCI-1720. It has 4 channel of D/A converters with an output range o

Your task is to set the voltage level of the D/A card Channel 3 to 5 Volts. An example can be found here (Dasoft.cpp).

Steps:

- Compare the "Advantech specific variables" in the framework program and in the sample program. Find the variables, which are changed.
- Initialize the card (cut and paste step 3 and step 4 from the sample program).
- Modify the variables, which are changed.
- Remove the parts which are not relevant (printf(); and getch(); commands, which are used for printing a text and getting a character).
- Set (output) the voltage (5 volts to channel 3).

For a little help, the program beginning and end is already written, only the above mentioned steps are needed for successful experiment.

The result can be seen on the webcam, a Green light will be shown on the drive unit.

Note

The names of the parameters used in this example are different from the names used in the actual measurement. You cannot simply cut and paste the parts of this example.

Beginning of code (not editable)

```
#include <windows.h>
#include <stdio.h>
#include <math.h>
#include <string.h>
#include "driver.h"
```

Fig. 13.: Web page of the first exercise with explanatory labels [34]

The main task is to implement a PID controller (or PI, P) to the servo motor and analyze how the motor reacts to each controller. Moreover, exercises are built on each other, students have to learn the basics of every component of the remote experiment. First, the programming of the D/A card is studied, this is the component which helps the user send specified voltages to the equipment from the PC. Second, the working principle of the real-time clock is investigated and then it is studied how students can write programs that make use of the counter card. This is the component that retrieves data from the encoder and converts it to numerical data. This value is read by a function which is executed in every tick of the real-time clock. After these measurements, further, more complex tasks can be executed (PID control of the motor, open-loop and closed-loop measurements, investigation of the stick-slip phenomena etc.)

In details, the architecture is depicted in Fig. 14. Students can make request through the web page to the PC located next to the experiment (Siemens industrial PC Pentium 4, 2.8 GHz).Requests are then transmitted to the D/A card (Advantec PCI- 1720) so an analog reference torque signal can be set to the servo drive which transforms the signal into PWM form. When the servo motor (Maxon A-max 26 (110961)) is in work, its built-in encoder (Maxon Digital Encoder HP HEDL 5540) provides signal to the counter card (Advantec PCI-1784). The information from the encoder is transformed into numerical data which is processed and visualized on the PC using MATLAB. The created graph is then sent to the web page and can be downloaded. In addition, there is a camera which offers surveillance of the experience to the students.

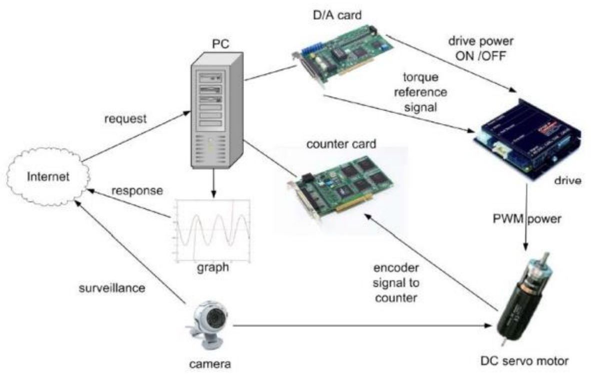


Fig. 14.: Architecture of the remote experiment [34]

VI. APPLICATION EXAMPLES OF APACHE GUACAMOLE

A. *User Interface for Disabled People*

The main goal of this section is to present an application of the Guacamole software and therefore help the better understanding of its structure and working principle.The example presented in this section was designed to help disabled people to use their own personalized desktop via an internet browser from anywhere [35]. The main goal of the architecture is to eliminate the need for a software to be installed on behalf of the disabled person. The only element the user has to work with is a HTML5 based web browser so the remote desktop can be used in a web browser without additional plug-ins. The reason why this proposed architecture is promising is that basically the very same or similar structure would be satisfactory for distant learning or remote experiments. The main difference is that in the cited publication, on the virtual desktop assistive technology (AT) tools have to be installed like screen reader or magnifier. In case of distance learning other software can be installed which students are in need of. The proposed structure integrates Guacamole (a HTML5 remote client web application and gateway), Virtual Network Connection (VNC, technology to control remote desktops), Oracle VirtualBox virtualization software package (virtual machine monitor or hypervisor) and CLEVER (middleware to manage virtual infrastructure).

The whole structure is based on cloud computing; the infrastructure is based on a data center where physical servers run virtual machines (VM). These VMs support computing environment, operating system and desktop applications are installed and can be accessed from a networked computer by using a remote desktop software. This means that users can interact with these programs installed on the VM as if they were installed on the user's computer. Although the cited publication focuses on the characteristics of the design from the aspect of assistive technology, all the characteristics of the utilized architecture are in favor of distant learning as well. Since the goal of the paper is to collect the features and demonstrate how such a design can be applied in remote learning and experimenting, I will only focus on the subject accordingly. The writing is not focusing on assistive technologies.

The basic idea is to create a virtual computer including general purpose software and ones that are crucial for engineering studies and experimentations. Students can access and interact with a laboratory set up on the campus from their home or anywhere they want. Moreover, students do not need an up-to-date hardware for accessing the content, all the computational tasks are fulfilled on the virtual machine.

This matter was also mentioned in another example [36] at the Coastal Carolina University, Computing Sciences Department plans to employ virtual machines in software engineering courses. They also use Oracle's Virtual Box and they experienced that for some students who use lower-end hardware, the performance of the virtual machines was quite slow. In order to eliminate this issue, they plan to give remote access to software on virtual machines via HTML5 enabled web browser to the students. The idea is to utilize Guacamole remote display proxy and this way, the computational tasks will burden the servers, not the students' computers. Also software can be upgraded simultaneously for everyone, students do not need to update their own software on their own computers. Therefore, students and teachers are in no need of constant consulting on the different versions of the software and can focus entirely on useful tasks.

The main element of the architecture (Fig. 15.) is the Guacamole software which is a HTML5 remote desktop client and gateway web application. A remote desktop server daemon is installed in the virtual machine (in this case a VNC server) for the remote control of the virtual machine. A hypervisor (VirtualBox) and a Virtual Infrastructure Manager (VIM in our case CLEVER) is applied for the virtual machines. The latter allows the cloud provider to control a cluster of physical servers. On each server, a hypervisor allows the running of one or more virtual machines and every one of them is connected to the internet. On each VM a remote desktop server daemon is installed which allows the user to remotely access and control the virtual machine. On the user end of the structure, the HTML5 remote desktop client/gateway application allows the user to access their VM using a web browser. The user has only one thing to do, he/she has to login in the web application in order to access his/her VM. As a result, the virtual desktop appears in the browser and the user can interact with it by using the keyboard and mouse.

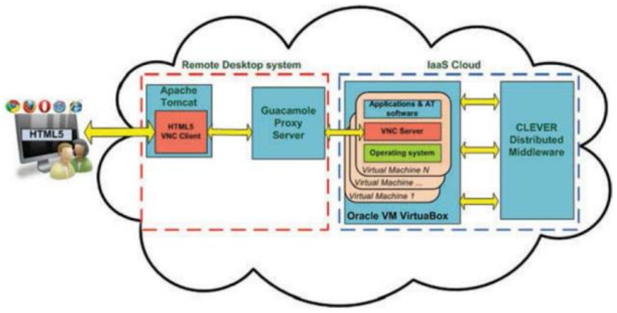


Fig. 15. Architecture of the system [36]

The VIM provides an infrastructure of virtual environment including a pool of networked virtual machines. The VIM has to be secure, scalable, modular, interactive and fault-tolerant and CLEVER is fit for these requirements. Also CLEVER provides the control of Linux-based cluster environment in which the virtual machines can be dynamically managed. Moreover, CLEVER can perform the following operations: start, shutdown, destroy and migrate a VM; monitor the performance and behavior of VMs (memory, CPU, storage); provide defense against hardware and software failures; manage disk images; create new virtual machine based on templates (including operating system and software).

The connection between the virtual machines and the web browser is established based on VNC and Guacamole. VNC technology is based on a graphical desktop sharing system that utilizes the Remote Frame Buffer (RFB) [37]. It works over a network and transmits the keyboard and mouse events between computers and graphical screen updates in the other direction. A standard VNC system integrates three basic elements: the server is on the machine that shares its desktop and it allows the client to take control of the its software. The client or viewer is the application that watches, controls and interacts with the server. The VNC protocol (RFB) is based on graphic primitives from server to client and event messages from client to server.

First, in the virtual environment a VNC remote desktop server daemon is installed, this is the basis of the user being able to access the VM remotely. To perform this interaction by a web interface, Guacamole is implemented. The architecture includes guacd which acts as a Guacamole proxy server and runs on physical Linux server. (It is also possible to install the Guacamole server on the physical server where the virtual machines run but it is a less transparent and secure architecture.) Guacd is a daemon process and is applied in a way that it works like an interface between the VNC server daemon process on the virtual machine and the VNC client developed as a HTML5 web application. Guacd integrates Java pieces that dynamically loads support for remote desktop protocols (client plug-ins) and connects them to remote desktops based on messages received from the web application (it runs in the background and listens for TCP connections from the web application). Guacd does not understand any remote desktop protocols (like VNC), it is able to decide which protocol support (client plug-in) needs to be loaded. Once a client plug-in is loaded, it is the element that is in control of the communication between itself and web application. This is the case until the client plug-in terminates. The web application uses JavaScript functions (the Guacamole API) so it can work with the proxy server. It is also based on a Canvas tag which is responsible for viewing remote screen content and to handle keyboard and mouse events. Whenever the user presses a key on the keyboard or performs a mouse action on the Canvas, the web application turns the detected input event into commands to control the virtual machine. The web application has to be deployed in a servlet container like Apache Tomcat. In addition, it manages authentication in order to prevent unauthorized access to virtual machines.

Since the publication focuses on assistive technology tools, it is vital to have audio connection. The problem is that VNC does not support this, audio signals produced by the software on the virtual machine cannot be heard remotely. Although this and all other assistive technology aspects are out of the boundaries of this writing, there are two reasons why it is useful to look into the topic briefly. Both of the aspects help to have a better picture of the Guacamole software. First, it is open-source and therefore can be modified freely according to the user needs. Second, Guacamole support RDP (another remote desktop protocol) not just VNC. The matter of modifying the source code emerged when the authors aimed at changing keyboard input events. Typically, disabled people use different devices so that emulating mouse actions and typing are easier and Guacamole was modified in a way so that is supports these solutions. RDP was important from the author’s point of view because it was necessary for them that the system supports native audio redirection e.g. for a screen reader application and RDP supports both video and audio. The solution they first proposed was not sufficient (it needs a new subsystem on the Linux physical server, two separate data flows are created and according to their measurements delay is too high) so they turned to RDP solutions. Sound is important for distant learning or experiments; it helps the student to perceive the process of a given experiment more thoroughly.

Basically, the architecture of the system utilizing RDP is not changed (Fig. 16.) [38]. The only difference is that in this architecture the virtual machine has to integrate an RDP server instead of a VNC server. The requirement of a HTML5 enabled web browser being utilized so that the virtual machine can be accessed from remote by PC, laptop, tablet, smartphone, remains the only condition laid down to the users. If the reference system is the one that is expanded with an audio streaming server, the RDP- based solution is a simplification of the architecture. Both video and audio signals are sent back to the web browser eliminating the need for an audio redirection subsystem. The authors' measurement confirmed that this new revised system works better, experiments have shown significantly shorter time delay.

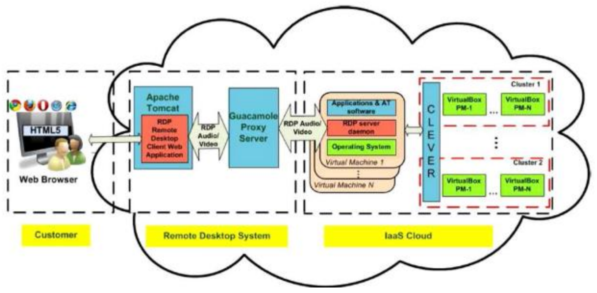


Fig. 16.: Architecture of the RDP-based revised system [38]

The aim of this section was to show an example of connecting and controlling remote desktops, environments with the help of Guacamole. The presented structure could be adapted in a remote learning or laboratory scenario. For instance, students could have an own virtual machine in a laboratory during a semester or a remote experiment could be connected to a server and through virtual machines, it could be controlled.

B. RP-SMARF platform

An example of utilizing Guacamole is a cloud-based platform for supporting research collaboration. The initiative is called RP-SMARF [39] and the aim of this system is to link together scattered privately-owned elements such as data banks, software or hardware elements. It „unifies geographically dispersed heterogeneous resources" [39] for researchers, this way a software developed by one research team or a batch of data gathered by another team can be shared with those other researchers and research team who the given team wants to give access to. There is a built-in authorization system which helps the researchers easily grant or deny access to his/her work.

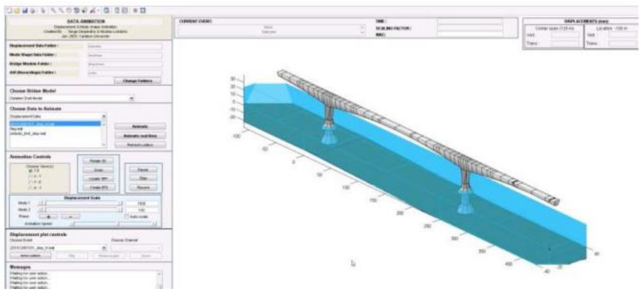


Fig. 17.: SPLASH in the RP-SMARF platform [39]

An example case depicts the system even more properly; it is about an instrumented bridge. The data collected from the instruments such as acceleration vibration data can be accessed from Carleton University by a research group on demand. The bridge operator shares sensor data so the remote researchers can assess structural health of the bridge. The application which the research team works with is also invocated in the RP-SMARF system so the result of data analysis is immediately accessible for the provider of the data as well. The software which in this example is used is SPLASH and it can simulate the effects of certain weather conditions on the bridge. A screenshot of SPLASH in the RP-SMARF platform is seen in Fig. 17. In order to share the distributed resources (high performance clusters and servers, virtual machines on servers, storages devices that serve as sensor data repositories and databases that host archival data) a cloud-based system must be installed.

The architecture (Fig. 18.) is based on a control server that manages a database of accessible resources and permissions. There is a remote agent in each site that gives access to the site's storage and computing resources behind the firewall. This access is to data based on SFTP or the native file system of the remote agent and the access to computing resources is performed via SSH. To the applications, access and control is provided by Guacamole software via VNC/RDP. To sum it up, the remote agent provides access to stored data and computing resources via the control server, the Guacamole server provides control of the computer by-passing the control server because it only needs screen information.

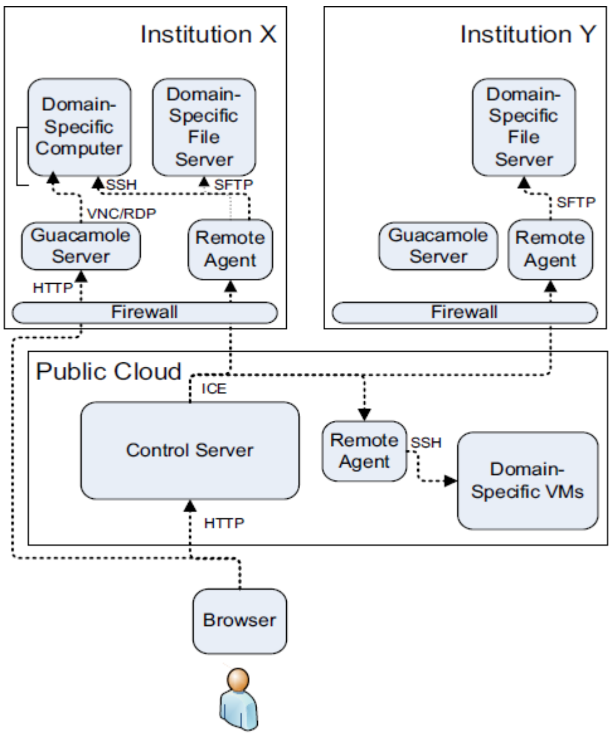


Fig. 18.: Architecture of the system [39]

The client is browser-based; this is where the Guacamole web application runs. It gives a graphical user interface of the RP-SMARF platform and interacts with the control server. The control server maintains information about the resources available, access permissions and jobs which are currently running or were recently run. All this information is in an SQL database. The remote agent interacts with the control server to start and stop jobs. Its other task is a file system service; it is „used by the control server to allow end-user interaction with the files accessible via the remote agent" [32]. The Guacamole server is basically a web server and it allows GUI-based control to devices via Remote Desktop Protocol (RDP) for Windows-based devices and via Virtual Network Connection (VNC) for other operating systems. architecture that was detailed and the publication emphasizes the data sharing and accessing among researchers and although there is less attention on the desktop sharing and on the control of shared software, it is more important in a distant learning or experimentation framework. It is obvious that the Guacamole software is sufficient for that goal. Otherwise SPLASH or any other software that researchers use to evaluate data could not be shared on a screen.

C. *InstantLab*

Another example of utilizing Guacamole software comes from Hasso Plattner Institute, Potsdam, Germany [40]. The authors teach operating system classes and during the years they faced the same problem over and over again which is: the number and version of platforms are constantly increasing. In order to conduct these operating system experiments, infrastructure must continuously evolve. And it is not only the already mentioned variety of platform versions but also the hardware has to maintained and updated. The proposed solution of the authors is to move the whole course into the cloud. This move frees the students and teachers from constant maintenance, versioning and configuration.

The new cloud-based platform is called *InstantLab* and it uses virtualization technology to solve the aforementioned problems. Experiments run in the university datacenter so students' computers are no longer function as the computational platform for the experiments. Students' devices are independent from the datacenter where software run, the students' devices are merely thin clients. Since there is one center of the system where software are installed, students do not have to configure and update their devices one by one. Instead, updates are made in the center and experiments come in containers. These integrate „virtual machine images and setup exact execution environment required by the experiment" [40]. Again, students do not need to setup the systems on their own so every student have the exact same and desired environment and software package.

The proposed architecture is based on *OpenNebula* software package (Fig. 19.). The main component is a web application (utilizing a Java based web framework called play!) through which users can start and conduct experiments. The experiments can be accessed and controlled through a terminal connection which originates from the web browser. Remote desktop access is realized with the help of the Guacamole software: a proxy server connects to the running virtual machines using VNC protocol. The information gathered is then retrieved to the client side web browser by JavaScript and using HTML5 element Canvas the remote desktop is drawn.

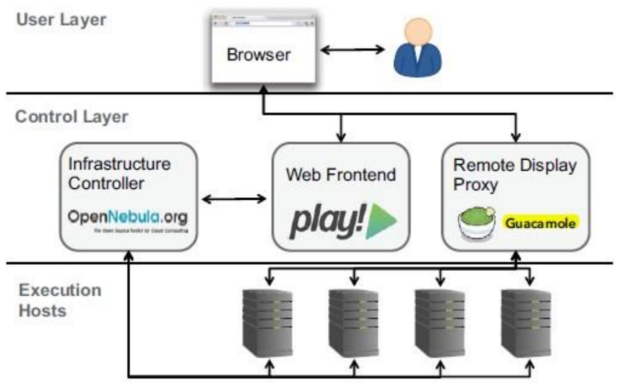


Fig. 19.: Proposed architecture [40]

This is a pure distant learning and experimenting scenario and based on this example, it is without question that Guacamole is a perfect HTML5 based remote desktop displaying technique and can be smoothly applied in distant learning systems.

D. Adaptive User Interfaces in Cloud Platform

Since cloud and virtualization technologies are quite fresh approaches in the field of education, examples are current, the one presented in this section is from 2014 [41].The main focus around this topic is to find an easy and fast way for the user to connect to the cloud and the content stored in it. The expectation is to eliminate the need for installing special software or plug-ins on the client side. The expansion of HTML5-enabled web browsers and web based applications utilizing its advantages has led to new solutions. One of them is Guacamole and although it is still very new, there are quite a few examples of its utilization. This section presents one of them.

Remote desktop virtualization enables to move the client's desktop to the cloud from its traditional PC environment so the desktop is stored in a remote virtual machine. Energy expenses can be reduced and it allows users not to update their hardware so frequently, thus saving costs. Moreover, not so powerful devices are enough to connect to the remote desktop since all the data, applications, memory are stored in a center. Only this remote center has to be a powerful machine, what remains of the users' PC is a so-called thin client without the need to have a hard-drive. These thin-clients, servers and software as services make up the thin client or remote desktop model.

Current utilizations of remote desktop technology are based on one instance of the operation system running on the server. It supports synchronous login from multiple users but this architecture has its difficulties e.g. imbalance in loads of different servers. In an educational scenario, this kind of difficulties could be more easily managed: in many cases, there is a finite and exact number of students logged into a remote experiment at the same time. However, a virtualized server environment where multiple isolated virtual machines can be created is a sufficient architecture for educational purposes as well. „A virtual machine is a software implementation of a machine that executes related programs like a physical machine. Each virtual machine includes its own system kernel, operating system, supporting libraries and applications." [41] Managing virtual machines needs a so-called hypervisor, under its supervision virtual machines can run simultaneously. It also decouples the virtual machines from the physical ones and it allows the virtual environment to be started and run on a different physical machine.

The aim of the separation of the execution and the presentation of applications (this is the basis of thin client technology) is to relieve the users of complex tasks in terms of software installation. This is significant in an educational context. It is desired that all the students have the same version of a software and it works perfectly and a centralized installation, configuration, upgrading and troubleshooting process helps to achieve this target. This centralization can be facilitated by web based software technologies but compatibility problems might emerge with browsers and also it needs huge efforts to set up such an environment. The other solution is the already detailed desktop virtualization together with HTML5 enabled web browser based accessibility. The main advantage of this architecture is that the user only needs a HTML5 compliant web browser (Firefox, Chrome, Internet Explorer, Safari, Opera etc.) and any of the existing operating systems. Not only does this solution eliminate the need to install special clients, it works on any device because of the HTML5 technology (PC, laptop, tablet, smartphone).

One of the most important criteria of a remote desktop virtualization solution in an educational context is its price. There are two proprietary products investigated in the cited publication: Citrix XenDesktop and VMware View. The main elements of XenDesktop are XenServer, Desktop Delivery Controller, Desktop Receiver and Virtual Desktop Provisioning. XenServer uses „Xen hypervisor to create and run multiple virtual machines on the physical machine" [41]. Desktop Delivery Controller „authenticates users, manages user's virtual desktop environments and establishes connections between users and the virtual desktops" [41]. Desktop Receiver is the application installed on user's device, Virtual Desktop Provisioning „provisions server, creates and manages virtual desktops from a single desktop image on demand" [41].

VMware View has quite similar components. VMware Server is a computer virtualization tool; it runs on a host operating system. View Client Software is the element that is installed on the user's computer or thin client. It is responsible for accessing the desktop running on the virtual machine. View Connection Server is for user authentication and after authorization, it redirects requests to appropriate virtual machine. View Administrator is a web based application and it is for „administrators to configure View Connection Server" and „to deploy and manage View desktops" [41]. The important thing is that although the architecture is somewhat similar to the one to-be-proposed but none of them is web based. So the creation of a new system is required for multiple reasons: it has to be open-source and the client side has to web browser based.

A brief overview of the proposed system (Fig. 20.) is the following: when a client requests a remote desktop from the (terminal) server, it establishes connection to the web server and opens a widget so a *Websocket* connection can be made to the server. HTTP connection is also available because although HTML5 enabled web browsers support Web sockets, it may not be supported by the utilized firewall or VPN system.) There is a terminal proxy at the gateway of the network. It interrupts the communication and checks if the requested session is already present. If it is, it provides information about the session to the web server and therefore acts as a bridge between client and central server.

If the session is not present, it creates a new one and sends the request from the client to the appropriate server where the proper virtual machine runs (depicted as orange submachines of a terminal server in Fig. 20.). The server receives this request and in return sends back a response,

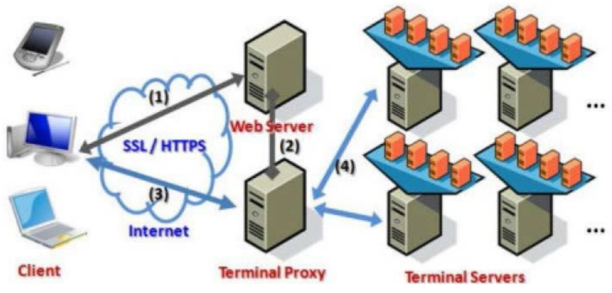


Fig. 20.: System architecture [41]

The software architecture is shown in Fig. 21., it is entirely web based. The hardware integrates CPU, memory, hard disk and several other physical devices. The hypervisor in this system is called KVM, it allows to run multiple virtual machines running Linux or Windows. Libvirt API is utilized so it can control and manage the hypervisor. It is open source as well as the other components and it is a „daemon and management tool for managing platform virtualization" [41]. PHP is a server-side scripting language for web development and MySQL is a relational database management system that runs as a server. It gives access to many users simultaneously to a number of databases. Guacamole was already introduced in previous sections, it is a HTML5 web application that supports graphical access through remote desktop protocols in a browser without any additional plug-ins. Google Web Toolkit is „a development toolkit for building and optimizing complex browser-based applications and provides a set of core Java APIs and widgets. These allow us to write AJAX applications in Java and then compile the source to highly optimized JavaScript that runs across all browsers" [41]. Basically, with the help of Google Web Toolkit, web application development is realized without the need to learn how to write JavaScript and HTML codes. Although Guacamole in itself would be sufficient to establish connection between the client's web browser and the virtual machine with one simple web page in the browser, it is useful to integrate it in a well-developed page.

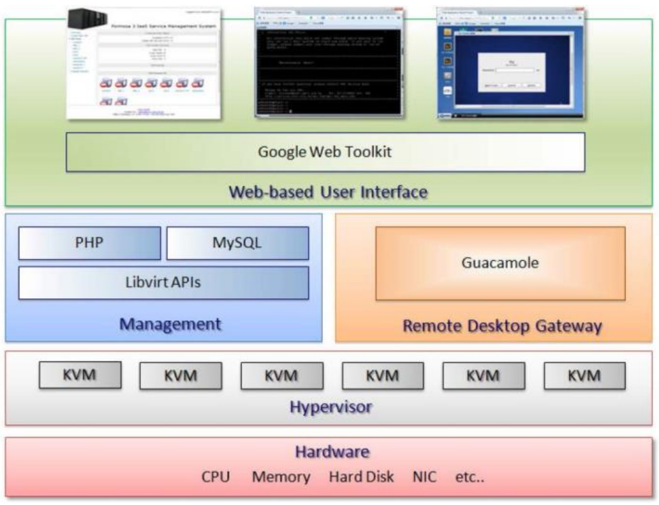


Fig. 21.: Software architecture [41]

Guacamole is utilized as the terminal proxy, it works based on web standards like JAX, JSON and HTML5 and the remote desktop access is realized via Virtual Network Computing (VNC), Remote Desktop Protocol (RDP) and SSH (Secure Shell). To access remote desktop, user initiates connection and so a terminal widget on the developed web page opens and the Guacamole web application/client connects to the Guacamole server. The Guacamole client is written in JavaScript and is within an Apache Tomcat server which technically is integrated in the Guacamole server (see Fig. 22.) The Guacamole web application reads the Guacamole protocol (request from user) and forwards it to the *guacd*. This is the element that actually acts as a proxy and it translates (graphical) output from VNC/RDP into XML and the same happens in reverse.

In other words, it interprets the messages of the Guacamole protocol (request from user) which results in connecting to virtual machine or machines. This is the element that is responsible for the connection between the web browser and the virtual/remote desktop(s). This translation between VNC/RDP and XML and the web browser's HTML5 rendering engine significantly defines the speed of the whole system. Therefore, there might be minor differences in the platform due to the utilized web browser [42].

The proposed system is a perfect example of how to set up a cloud based system which utilizes virtualization and remote desktop technologies, both perfect tools for remote educational and experimental context. Although Guacamole is a fresh software package, a few publications suggest that it is a very promising tool to set up remote desktop scenarios where only a HTML5 compliant web browser is needed of the student.

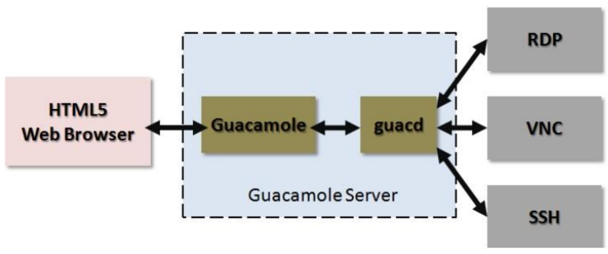


Fig. 22.: Apache Guacamole architecture [42]

VII. Pedagogical Considerations, Learning Management

Previous sections were basically focused on how to design reliable, effective and convenient remote laboratories, experiments. This section expands the environment of these systems and gives solutions to place them in learning management systems (LMS) based on pedagogical considerations.

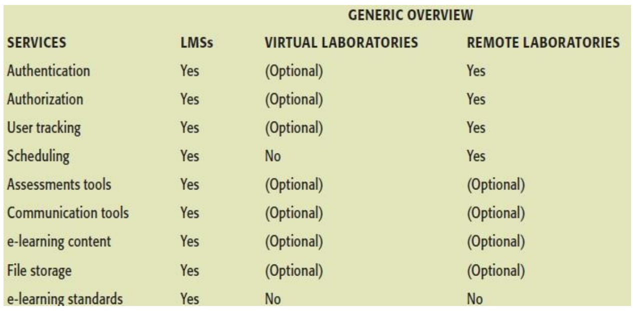
Sancristobal et al. [43] in their paper investigated very thoroughly the issue of integration of remote laboratories into learning management systems (LMS). Most of the remote laboratories are developed independently from other systems. Therefore, all of the common features (e. g. booking, authentication) are developed from scratch every time a new remote experiment is designed and it consumes a lot of effort and time. The article suggests another approach: an LMS should be developed which supports the easy integration and sharing of remote laboratories from all over the world.

LMS, virtual and remote laboratories are all tools that emerged together with the spread and development of internet technologies but they have not been viewed in an expanded system, they mostly exist independently from each other. However, the aim is for students to be able to work in a well-defined learning system where theory and practice, traditional and e-learning tools are utilized in one system. Learning scenario is a social scenario where students and teachers interact with each other and work together and this system is to be preserved in e-learning contexts as well.

As it mentioned above, remote laboratory developers recognized this and mostly these labs are combined with web-learning tools and traditional hands-on laboratories, or even some kind of learning framework but these are ad hoc solutions and are suitable for only one or two remote experiments to be integrated to the course of a university. It consumes a significant amount of effort to integrate a new remote experiment to an existing system. The solution to this is to design a system to which new remote laboratories can easily be integrated. If this design criterion is fulfilled, sharing these laboratories among universities becomes easier as well. Without a system like this, different software and codes created for each experiment can hardly or cannot be reused by other facilities.

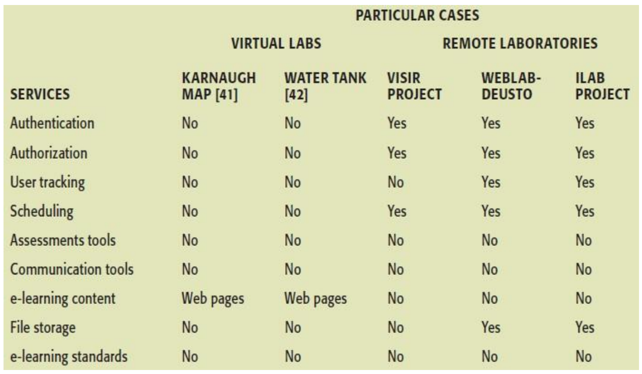
Even if the only goal is to design a collaborative learning context around the remote laboratories, it is a better approach to integrate it in an existing LMS instead of developing a structure from scratch with the very same features. Table 1 shows all the items of a satisfactory e-learning scenario, it is obvious that LMSs perform better than features developed around a single remote experiment. This remains unchanged if we compare several particular remote laboratories, not all the features are available (Table 2.).

Table 1: Comparison of LMSs and remote laboratories (and virtual laboratories)



	GENERIC OVERVIEW		
SERVICES	LMSs	VIRTUAL LABORATORIES	REMOTE LABORATORIES
Authentication	Yes	(Optional)	Yes
Authorization	Yes	(Optional)	Yes
User tracking	Yes	(Optional)	Yes
Scheduling	Yes	No	Yes
Assessments tools	Yes	(Optional)	(Optional)
Communication tools	Yes	(Optional)	(Optional)
e-learning content	Yes	(Optional)	(Optional)
File storage	Yes	(Optional)	(Optional)
e-learning standards	Yes	No	No

Table 2: Comparison of a few remote laboratory frameworks



	PARTICULAR CASES				
	VIRTUAL LABS		REMOTE LABORATORIES		
SERVICES	KARNAUGH MAP [41]	WATER TANK [42]	VISIR PROJECT	WEBLAB-DEUSTO	ILAB PROJECT
Authentication	No	No	Yes	Yes	Yes
Authorization	No	No	Yes	Yes	Yes
User tracking	No	No	No	Yes	Yes
Scheduling	No	No	Yes	Yes	Yes
Assessments tools	No	No	No	No	No
Communication tools	No	No	No	No	No
e-learning content	Web pages	Web pages	No	No	No
File storage	No	No	No	Yes	Yes
e-learning standards	No	No	No	No	No

In order to clarify once and for all what a learning management system is here lies a definition from [43]: „An LMS is a framework that delivers and manages instructional content, identifies and assesses individual and organizational learning or training goals, tracks the progress toward meeting those goals, and collects and presents data for supervising the learning process of an organization as a whole." There are proprietary LMSs like Blackboard but in an educational context it is more beneficial to deploy open-source free software. First, because it helps to reduce costs and second, there is a high probability as to modifications of the source-code might be needed. Examples of open-source LMSs are Moodle, Sakai, dotLRN or Claroline.

The three most important part of an LMS are the database, the Web server and its services. Database is where all the data is stored for proper working of the learning management system like a table of students' name associated with the coursed they attend. The web server provides a web browser based environment through which content can be delivered and services can be executed using templates. Services are packages of source codes each responsible for a functionality of the LMS like forums or surveys.

Sancristobal et al. [43] modified the source code of Moodle and as a result, the following three step procedure is available for teachers to integrate remote experiments in a learning management system. First step is the adding of a remote lab to the LMS. It is done by the administrator and the created interface makes it quick and easy to add a title, description etc. to the lab. Second step is adding features to the experiment like forums, chatrooms or exams. This step is made by the teacher but he/she does not have to write a single line of code, he/she can use a menu-driven interface. Third step is the login of students, they log in to the Moodle course, choose a laboratory, click on its link and carry out the experiment. It is scheduled by queues so students might have to wait. If the teacher offers a calendar function to reserve time, students have to reserve before carrying out the experiment but they do not have to wait when they log in.

A. Moodle

Moodle is an open-source learning management system and over the years it has been widely utilized and accepted. Not only it is a free software, it is very flexible, anyone can develop and add new features and modules to it. For successful installation, Moodle requires a web server and a database because it stores most of its data in a database which is written in PHP language. Also Moodle recommends MySQL database [44].

Moodle can integrate many features such as chats, forums, online booklets, variety of questions, collections of problems and exercises, lecture notes, text-based and html-formatted documents, multimedia resources: graphics, video, audio, PowerPoint, Flash-based applications and Java applets. Moodle has a strong content creating skill due to its built-in HTML-editor.

In order to smoothly manage remote experiments, a booking module is utilized (it was written in PHP - [44]), it allows the student to choose an unoccupied date from the calendar and reserve a one-hour long timeslot. It is possible to allow multiple students to access the experiment, in this case a chat program (synchronous communication tool) is available. However, it is not active if only one student is

„present" at the experiment. The teacher when creating a remote experiment can insert a booking link which enables the students to use this module. Also teachers in editing mode can maximize the allowed number of students to an experiment and can give a title to the remote lab and insert a glossary or other contents helping the student better understand the experiment. The booking module can easily be added to a course or experiment by the teacher or administrator, it can be chosen from a drop-down menu.

A huge development stage of Moodle was when the so-called EJSApp module was released. The past few years, Easy Java Simulation (EJS) has grown to be a widespread tool to create virtual and remote labs. It is a tool specifically designed to create discrete computer simulations. Recent releases of EJS support connection with external applications such as LabView and MATLAB/Simulink. The result is that virtual laboratories and GUIs to remote labs and experiments can be created. EJSApp allows these experiments to be easily integrated into Moodle. With its utilization, user access to each experiment can be controlled, resizing options are available for the embedded applets and creates backups of the experiments [45].

The idea behind Moodle and e-learning is that multimedia tools that enables the creation of activities makes the learning process friendlier and easier. It results in more motivated and interested students, material can be presented to student which otherwise could not be due to lack of time and any material from the internet can be linked into Moodle. It was built in accordance with a socio-constructivist, collaborative teaching approach which gives a strong knowledge based on active, interactive and collaborative (e.g. teamwork in a remote laboratory) learning. Moodle provides an environment that allows a discovery-based in e-learning and it invokes the student inquiry towards a topic.

„Studies have shown that distance education cannot be a complete substitution for traditional face-to-face teaching model. The lack of face-to-face interaction between the teachers and students will most likely reduce the motivation of the students to learn" [46]. Therefore, in case of distant learning, it is an effective way to study to discuss problems between students and teachers.

Teachers can organize, manage and deliver course related contents; they can assess students' work by Moodle based feedback and evaluation tools. Students can also give feedback in qualitative or quantitative form. For example, the assignment module gives teachers the opportunity to attach evaluation to the corrected assignment. Not only can students upload their finished work in any format, teachers can include an audio-based evaluation to the assignment. Forms are automatically corrected by the system and in case of open questionnaires teachers can review it and comment. „The feedback is given to each student in his/her pace. The ability to monitor each individual student from amongst a huge number, providing them with the right feedback at the correct time is one of the most advantageous features of Moodle" [47]. Students have the opportunity to interact with each other, based on synchronous and asynchronous modules. The former is the chat module, the latter is the forum. Participants can share their questions, ideas and help each other and even teachers can be part of the conversation or just view the dialogue and interact in need. Viewing student conversations helps teachers to better understand where the difficulties lies and which parts are easier to understand.

VIII. Google for Education

Google for Education is a whole network of applications, tools and devices, a cloud-based system. It helps reduce time and effort spent on organizational tasks. Not only can all the data of a project be saved and stored automatically so it can be retrievable, students and teachers can be in contact outside of the classroom with the help of smart devices able to connect to the internet. Therefore, it is an ideal system to be applied in a remote laboratory and implicitly, in a whole e-learning system.

All the students and teachers need is a Google account which many of them already have since they use Gmail and that account is suitable for all Google Applications. This means that the students do not need a whole new pack of accounts, passwords, they don’t have to get accustomed to a new system. This is a huge advantage of Google for Education, moreover, it is free just like the Gmail service.

The heart of the system is the so-called Classroom; this is basically the control room [48]. For the sake of simplicity, I will use the phrases „teacher” and „class” but naturally, these applications can be used in higher education just as well. The professor can create a Classroom in his/her subject for the students taking part in the current semester. The basis of the system is that teachers and students can stay in touch outside of the classroom in a paperless way. First, teachers have to set up the Classroom (he/she needs a Google account). For this, there is a step by step guide provided by Google and if it is not sufficient, they can ask for an IT professional from Google to help. According to Google, the installation and setting of the Classroom should not take more than a few minutes. Since this is an application of Google, we do not need to install a whole new program, it works from a web browser just like Gmail.

After the setup of the Classroom, teachers can create a new course and invite the students (an email is sent to them). Within the course, projects or assignment can be created and the system sends every single student his/her own copy. Then teacher can follow which student has finished the task and who are the ones still working on it. From the students’ aspect, to join the Classroom, they have to accept the invitation sent by email. After that, all the content that the teacher uploads (including new assignments, projects and those helping the understanding of the curriculum and also material concerning the subject) will be at one place. There is an additional feature that helps to simplify the sharing of materials both for students and teachers: The Classroom share button for websites. With the help of this function, a whole website can be shared with one click, there is no need for copying links or utilization of another application, it directly shares the material with all the students in the Classroom. Every time the teacher uploads something, a warning email will be sent to the students.

Another feature of the Classroom is that students and the teacher can communicate with each other in real time. It is possible to message only the teacher but both the teacher and students can send a message to everyone as well. Teachers have the authority to moderate announcements and messages. Students when finished with the task can send the document to the teacher in many forms: attached document, link, shared document from cloud (Google calls its service Drive) or even photos. Once a student’s work is finished and turned in (after the expiration of the deadline it can be blocked), the assignment cannot be modified unless the teacher allows it. The teacher can grade the work and return comments which will be sent to the student in a spreadsheet which is updated every time a new finished work is graded by the teacher.

A huge advantage of the Google for Education system is that it incorporates Google other well-known (and similarly free) and widely used applications. It significantly reduces the time of getting accustomed to all the features because many already knows them, the only difference is basically that they used them separately before. Most obvious example is Gmail which is a web browser based e-mail client. For educational purposes Documents and Drive are better examples. Documents allows people to create and edit docs, spreadsheets and presentations and with the help of Drive (Google’s cloud storage), these documents can be shared and then edited together in real time and are saved automatically every time the content is altered. This structure is the basis of the whole Education system.

For a project that a whole class is working on or for just one student to store his/her deadlines, it is essential to have a calendar and Google has a feature for that (Calendar). It is synchronized with Gmail and Drive so deadlines can be automatically migrated to the Calendar. Sites and Forms are also very useful for educational purposes. What is common in these two applications is that they help to build structures from blocks and eliminates the need to write code which results in less time spent on creating the sites and forms.

With Sites, both teachers and students can create websites easily, for example if the teacher wants to create one for a topic. For students it is helpful, if they assigned to create a website. Forms are even more helpful, teachers can for example ask the students to evaluate the subject, the materials available to the subject, the teachers work itself or to find out if a remote laboratory was helpful for the students. Many times, for their tasks (for example final project) students need to assess people’s view, attitude in a topic and for that too, Forms is a very useful tool. Once the form is created, it can be shared in the Classroom or anywhere else and its link can be sent via email. For the owner of the form, an evaluation spreadsheet is generated and every single response is saved into it.

The aforementioned importance and necessity of communication between teachers and students calls for a video call application and Google has the answer to it in the form of Hangouts. This tool can be applied in remote laboratories when professors want to establish a visual connection between the students and the laboratory. Of course for this a program has to be written in order to automatically turn on the camera in the lab and Hangouts application when it is needed. Otherwise someone has to be in the laboratory to turn the camera on and then the remote laboratory would lose its purpose which is for the laboratory to be always accessible. Although if the laboratory is remote because there is not enough space for the students to be there physically and the goal is to address students distant from the classroom, Hangouts might be an option without any alteration to establish a virtual classroom.

In the introduction I mentioned that Google offers devices as well. The application of these devices is voluntary, meaning that Google Apps run on any computer which is able to connect to the internet with a web browser. Moreover, these applications are available for iOS and Android devices and even if the smartphone or tablet is offline, information on assignments can be viewed because of the caching system (every time the device is online, it downloads the latest information so it can be available offline.) Therefore, there is no need to buy their laptops or tablets but they offer a wide range of products which are optimized for the Education system.

The main advantages of Google for Education system are the following: it is versatile because it integrates many different kind of applications, it has a tremendous focus on collaboration, these applications are well-known among people, therefore there is no need to learn to navigate in a whole new environment, it is easily installed and it is free. For schools (grammar schools and universities as well) it is of huge importance to find solutions to educational problems, to improve their standards and to reduce the time spent on managing administrative issues in an inexpensive way. A significant percentage of a school’s teaching budget is for covering the costly licenses of professional programs and learning management systems are no expectation. An easy and effective way to reduce these costs is the use of free programs. Although Google Classroom was released in August 12, 2014 and there are not many published examples of its utilization, most of its elements have been used for years and are reliable software. There is no major obstacle in the way of its spread and I see it as an adequate solution to build up an e-learning system.

A. Utilization of Google Applications

There is one publication about the role of Google Apps in an e-learning system [49]. The author approaches the topic from a pedagogical point of view. He cites models from the past and his observation is that although there is not one perfect model because of various contexts, environments and one model can easily vary while the „given course unfolds”, his statement is that best models’ intersection is their active and collaborative nature. With the spread of online / remote courses and laboratories (MOOCs – Massive Open Online Courses), a new era of teaching and learning model is coming. The basis of it is that time spent in a classroom are focused on the collaborative work and discussions with the professor and lectures and other presentations are watched outside of the school with the help of the internet and e- learning systems. He calls this set-up „flipped classrooms” because the nature of time spent in the classroom reverts.

Many of the already well-known Google Apps are built on this collaborative environment, students exchange ideas and solutions to problems and exams in cloud- stored documents (Google Drive, Documents – Fig. 23.). Many educational groups are established in social networks (Facebook, Google+) so this whole collaborative system is not strange to students, moreover, the apps they use are fit for creating e- learning frameworks.

The article proposed a Google Application-based e-learning system a year before Google Classroom was available. He listed the teaching and learning needs and all the possible Google Apps solution to the given need. Based on Google Sites, he created a template that integrated all of them. There were needs which could have more than one Google Apps match (e.g. for communication Hangout, Gmail, Google+). The fact that Google launched its Education framework means that there is potential in it and the author’s work was not in vain and above all was not unnecessary. Therefore, it is worthy to mention the results of the evaluation of this system even if its representativeness is limited. He presented this framework to a BSc course (11 student) and a MSc course (10 student) and to the question „How would you judge the course’s overall quality based on the grading scale A-F?” the response was positive. 8 out of 10 forms returned by the BSc students rated the course as A (one form returned as B, another returned as C). In response to the same question, 8 out of 10 forms returned by MSc students rated the course as B (one form returned as A, another returned as C).

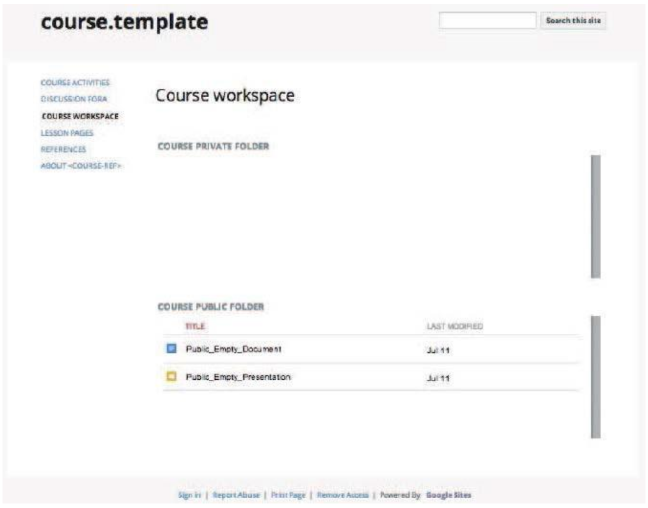


Fig. 23.: Google Drive folders to build up the course workspace [49]

Although Moodle and other commonly used e-learning platforms can be misused for only storing slides and lecture notes, they are still a satisfying framework of education thanks to their concept: they support group activity. One of the author's most important statement is that this Google Apps based framework is not free of limitations and is not capable of replacing Moodle or other learning management systems because of the lack of an automatic grading system and the „mechanisms for submitting student deliverables are limited" [49]. In light of the existence of Google for Education and particularly the Classroom application, it is safe to say that both problems listed by the author are eliminated. Therefore, Google for Education cloud- based e-learning system is definitely a good basis to build a collaborative e-learning model on.

IX. Conclusion

Since the appearance of remote laboratories many architectures were designed. New tools and trends have always been built into these designs. Currently, the fastest developing technology is HTML5 and cloud based remote laboratories and for that, remote desktop technologies provide perfect solutions. For example, with Guacamole, in a HTML5 enabled web browser a remote desktop can be displayed without any additional plugin. Also HTML5 supports portable devices such as smartphones and tablets and this accelerates the spread of remote experiments.

Another trend is to integrate remote laboratories in learning management systems (LMS). This topic has been around for years now but it is becoming more and more significant. With open-source LMSs like Moodle, developments in remote experiments can be followed because new modules are continuously developed. Also Google with its framework called Google for Education is aiming to gain a foothold among LMSs and it has every chance to do so.

References

[1] D. Lowe, C. Berry, S. Murray, E. Lindsay, "Adapting a Remote Laboratory Architecture to Support Collaboration and Supervision," in Proceedings of the Sixth International Conference on Remote Engineering and Virtual Instrumentation, pp. 103-108, 2009

[2] M. Tawfik, E. Sancristobal, S. Martin, G. Diaz, M. Castro, "State-of-the-Art Remote Laboratories for Industrial Electronics Applications," in Technologies Applied to Electronics Teaching (TAEE), pp. 359-364, doi: 10.1109/TAEE.2012.6235465, 2012

[3] J. Garcia-Zubia, P. Orduna, D. Lopez-de-Ipina, and G. R. Alves, "Addressing Software Impact in the Design of Remote Laboratories," Industrial Electronics, IEEE Transactions on, vol. 56, pp. 4757-4767, 2009.

[4] M. J. Callaghan, J. Harkin, T. M. McGinnity, and L. P. Maguire, "Intelligent User Support in Autonomous Remote Experimentation Environments," Industrial Electronics, IEEE Transactions on, vol. 55, pp. 2355-2367, 2008.

[5] N. Sousa, G. R. Alves, and M. G. Gericota, "An Integrated Reusable Remote Laboratory to Complement Electronics Teaching," Learning Technologies, IEEE Transactions on, vol. 3, pp. 265-271, 2010.

[6] A. Kalantzopoulos, D. Markonis, and E. Zigouris, "A Remote Laboratory for Real-Time Digital Image Processing on Embedded Systems," International Journal of Online Engineering (iJOE), vol. 5, pp. 25-29, 2009.

[7] E. Irmak, R. Bayindir, I. Colak, and M. Soysal, "A remote laboratory experiment for 4-quadrant control of a DC motor," Computer Applications in Engineering Education, vol. 19, pp. 747-758, 2011.

[8] C. A. Ramos-Paja, J. M. R. Scarpetta, and L. Martinez-Salamero, "Integrated Learning Platform for Internet-Based Control-Engineering Education," Industrial Electronics, IEEE Transactions on, vol. 57, pp. 3284-3296, 2010.

[9] B. Hanson, P. Culmer, J. Gallagher, K. Page, E. Read, A. Weightman, and M. Levesley, "ReLOAD: Real Laboratories Operated at a Distance," Learning Technologies, IEEE Transactions on, vol. 2, pp. 331-341, 2009.

[10] C. Longzheng,Y. Shengsheng, Z. Jing-li, "Research and Implementation of Remote Desktop Protocol Service Over SSL VPN," in proceedings of the 2004 IEEE International Conference on Service Computing (SCC'04), pp. 502-505, doi: 10.1109/SCC.2004.1358052, 2004

[11] P. Orduña, J. Irurzun,L. Rodriguez-Gil,J. Garcia-Zubia,F. Gazzola, D. López-de-Ipiña, "Adding New Features to New and Existing Remote Experiments Through Their Integration in WeblLab-Deusto", in International Journal of Online Engineering (iJOE), vol 7, Special Issue 2, "REV2011", pp. 33-39, 2011

[12] P. Orduña, J. García-Zubia, L. Rodriguez-Gil, J.Irurzun, D. López-de-Ipiña, D Gazzola, "Using LabVIEW Remote Panel in Remote Laboratories: advantages and disadvantages,"in proceedings of the Global Engineering Education Conference (EDUCON 2012 IEEE), doi: 10.1109/EDUCON.2012.6201134, pp. 1-7, 2012

[13] I. Gustavsson, J. Zackrisson, L. Håkansson, I. Claesson, T.L. Lagö, "The VISIR project – an Open Source Software Initiative for Distributed Online Laboratories", in proceedings of Remote Engineering & Virtual Instrumentation Conference (REV), Porto, Portugal, June 25 – 27, 2007

[14] Harward, V. Judson, et al. "The iLab Shared Architecture: A Web Services Infrastructure to Build Communities of Internet Accessible Laboratories," in proceedings of the IEEE , vol.96, no.6, pp.931-950, 2008.

[15] A. Melkonyan, A. Gampe, M. Pontual, G. Huang, D. Akopian, "Facilitating Remote Laboratory Deployments Using a Relay Gateway Server Architecture," in proceedings of IEEE Transactions on Industrial Electronics, Vol. 61, No.1, 2014

[16] T. Richardson, Q. Stafford-Fraser, K. R. Wood, A. Hopper: Virtual Network Computing, IEEE Internet Computing, January-February 1998

[17] Tan, Kheng-Joo, et al. "A remote thin client system for real time multimedia streaming over VNC." in proceedings of the IEEE International Conference on Multimedia and Expo (ICME), 2010.

[18] T. Scheucher, P. H. Bailey, C. Gütl, V. J. Harward, “Collaborative Virtual 3D Environment for Internet-Accessible Physics Experiments,” International Journal of Online Engineering (iJOE), vol. 5., Special Issue 1: REV2009, doi: 10.3991/ijoe.v5s1.1014, 2009

[19] D. Lowe, C. Berry, S. Murray, E. Lindsay, "Adapting a remote laboratory architecture to support collaboration and supervision." in Proceedings of the Sixth International Conference on Remote Engineering and Virtual Instrumentation, pp. 103-108, 2009.

[20] D. Mulfari, A. Celesti, M. Villari, A. Puliafito, “Using Virtualization and noVNC to Support Assistive Technology in Cloud Computing,” in Proceedings of the IEEE 3rd Symposium on Network Cloud Computing and Applications, doi: 10.1109/NCCA.2014.28, 2014

[21] S. C. Kong, Y. Y. Yeung, X. Q. Wu, “An experience of teaching for learning by observation: Remote-controlled experiments on electrical circuits,” in Computers and Education 52, pp. 702-717, doi: 10.1016/j.compedu.2008.11.011, 2009

[22] S. Kibe, T. Koyama, M. Uehara, “The Evaluations of Desktop as a Service in an Educational Cloud,” in Proceedings of the 15th International Conference on Network-Based Information Systems, doi: 10.1109/NBiS.2012.43, 2012

[23] A. P. Shisode, M. R. Dhotre, “PC Applications on Android Mobile for Remote Desktop Control,” in International Journal of Innovative Research in Computer and Communication Engineering, vol. 3., Issue 6, pp. 5741-5748, doi:10.15680/ijircce.2015.0306133, 2015

[24] 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,” in IEEE Transactions on Instrumentation and Measurement, vol. 56, No. 4., pp. 1124-1134, 2007

[25] S. Rapuano, F. Zoino, “A Learning Management System Including Laboratory Experiments on Measurement Instrumentation,” in IEEE Transactions on Instrumentation and Measurement. vol. 55, No. 5, pp. 1757-1766, October 2006

[26] S. Hubalovsky, “Remote desktop access us a Method of Learning of Programming in Distance Study,” in Proceedings of the 14th International Conference on Interactive Collaborative Learning (ICL), pp. 450-455, 2011

[27] M-F. Crainic, S. Preitl, “Virtual Laboratory for a Remotely Operating Robot Arm,” in Proceedings of the 9th IEEE International Symposium on Applied Computational Intelligence and Informatics (SACI), pp. 101-104, 2014

[28] F-H. Jen, T. C. Do,” A Proposed System for Practicing Indutrial Robot Remotely,” in Intelligent Technologies and Engineering Systems vol 234, pp. 365-372, doi: 10.1007/978-1-4614-6747-2_44, 2013

[29] B. Nagy, M. Papp,L. Raj, R.T. Fekete, “Development of a behavior engine in ethorobotics system and implementation on a mobile robot in LabVIEW environment”, in Proceedings of the International Symposium on Small-scale Intelligent Manufacturing Systems (SIMS). Narvik, Norway. pp. 121-125, 2016

[30] R.T. Fekete, B. Korcsok, L. Raj, “Affective psychological basis of behavioural modelling and simulation in ethorobotics”, in Proceedings of the CERiS'15: Workshop on Cognitive and Etho - Robotics in iSpace, Budapest, Hungary, pp. 30-39,2015

[31] G Sziebig, B Takarics, P Korondi, “Control of an embedded system via internet”, in IEEE Transactions on Industrial Electronics 57 (10),pp. 3324-3333, 2010

[32] V. Fedak, P. Bauer,V. Hájek,H. Weiss, B. Davat,S. Manias, I. Nagy, P. Korondi, R. Miksiewicz, P. Duijsen, P. Smékal, “Interactive E-Learning in Electrical Engineering”, in Proceedings of the 15th International Conference on Electrical Drives and Power Electronics (EDPE), Podbanské, pp. 368-373, 2003

[33] H. Weiss, A. Schmidhofer, A. Schmid, V. Hajek, B. Davat, S. Manias, I. Nagy, P. Korondi, K. R. Jardan, R. Miksiewicz, V. Fedak, P. Smekal, P. Bauer, P. van Duijsen, “Animated and Interactive E-Learning Concept and Realization,” in Proceedings of the IASTED International Conference on Web-Based Education (WBE), 2004

[34] P. Bauer , P. Korondi, P.J. van Duijsen, “Integrated Control and Circuit Simulation for a Motion Control System”, in Proceedings of the 10th European Conference on Power Electronics and Applications (EPE), 2003

[35] D. Mulfari, A. Celesti, M. Villari & A. Puliafito, “Using Virtualization and Guacamole/VNC to Provide Adaptive User Interface to Disabled People in Cloud Computing,” in Proceedings of the 10th International Conference on Ubiquitous Intelligence & Computing, pp. 72-79, DOI: 10.1109/UIC-ATC.2013.42, 2013

[36] C. Cox, M. A. Murphy, “Hands-on, web service based, software architecture lab component for software engineering course,” Computing Sciences Department, Coastal Carolina University, 2015

[37] D. Mulfari, A. Celesti, M. Villari, A. Puliafito, “Providing Assistive Technology Applications as a Service Through Cloud Computing,” in Assistive Technology, The Official Journal of RESNA, 27:1, pp. 44-51, doi: 10.1080/10400435.2014.963258, 2015

[38] D. Mulfari, A. Celesti, M. Villari, “A computer system architecture providing a user-friendly man machine interface for accessing assistive technology in cloud computing,” in The Journal of Systems and Software, JSS-9402, 2014

[39] A. McGregor, D. Bennett, S. Majumdar, B. Nandy, J. O. Melendez, M. St-Hilaire, D. Lau, J. Liu, “A Cloud-based Platform for Supporting Research Collaboration,” in Proceedings of the 8th International Conference on Cloud Computing, DOI: 10.1109/CLOUD.2015.162, pp 1107-1110, 2015

[40] C. Neuhaus, F. Feinbube, A. Polze, “A Platform for Interactive Software Experiments in Massive Open Online Courses,” in Transactions of the SPDS: Journal of Integrated Design and Process Science 18 (1), pp. 69-87, DOI: 10.3233/jid-2014-0010, 2014

[41] S-T. Wang, H-Y. Chang, “Development of Web-based Remote Desktop to Provide Adaptive User Interfaces in Cloud Platform,” in International Journal of Computer, Electrical, Automation, Control and Information Engineering Vol.8., No.8., 2014

[42] J. Bickford, P. Giura, “Safe Internet Browsing using a Transparent Virtual Browser,” in Proceedings of the 2nd International Conference on Cyber Security and Cloud Computing, 2015

[43] E. Sancristobal Ruiz et al., “Virtual and Remote Industrial Laboratory: Integration in Learning Management Systems,” in Proceedings of the IEEE Industrial Electronics Magazine, vol. 8., Issue: 4., doi: 10.1109/MIE.2012.2235530, pp. 45-58, 2014

[44] J. M. M. Ferreira, A. M. Cardoso, “A Moodle extension to book online labs,” in International Journal of Online Engineering, vol 1, No 2, 2005

[45] “Moodle plugind directory”, Online, accessed: 10/12/17 12:43:28 PM, https://moodle.org/plugins/mod_ejsapp

[46] Y-H. Wang, Y-H. Tseng, C-C. Chang, “Comparison of Students’ Perception of Moodle in a Taiwan University against Students in a Portugese Univerity,” in Proceedings of the International Conference on Web-Based Learning (ICWL), pp. 71-78, 2013

[47] S. Kotzer, Y. Elran, “Learning and teaching with Moodle-based E-learning environments, combining learning skills and content in the fields of Math and Science & Technology,” in Proceedings of the 1st Moodle Research Conference, pp. 122-131, 2012

[48] “Google for Education”, Online, accessed: 14/12/17 09:37:12 AM https://www.google.com/edu/

[49] Jose Manuel Martins Ferreira, “Flipped classrooms: From concept to reality using Google Apps,” in Proceedings of the 11th International Conference on Remote Engineering and Virtual Instrumentation (REV), pp. 204-208, DOI: 10.1109/REV.2014.6784256, 2014