Preprints of the 19th World Congress
The International Federation of Automatic Control
Cape Town, South Africa. August 24-29, 2014

Virtualization - an Answer to Secure Development of Online Experiments

Tomáš Frťala and Katarína Žáková

Faculty of Electrical Engineering and Information Technology
Slovak University of Technology in Bratislava
Slovakia
(e-mail: tomas.frtala@stuba.sk, katarina.zakova@stuba.sk)

Abstract: The paper presents a new approach to the management of online laboratories. Virtualization of environment can bring several advantages both for students and teachers. The paper shows the way how such environment can be incorporated to the laboratory ecosystem. It takes into account various virtualization technologies and software. The automatized installation process was proposed on the base of user requirements.

Keywords: virtualization, VirtualBox, emulation, control remote experiments, online laboratory.

1. INTRODUCTION

Development of Internet, Internet technologies, web services and protocols influences not only everyday life including education area but also the way of experimentation. Experiments and all interactive methods of learning are becoming inseparable part of control engineering education. Students and teachers use various multimedia objects – pictures, video, animations, online computations, virtual experiments, online simulations and remote experiments, as well. Today's trend is to approach Internet applications and services not only via desktop computers and notebooks but also using tablets or smart phones. This trend is influencing the area of online experiments, too (Papadopoulos and Leva, 2013). The information that needs to be displayed is more or less identical, it is only concentrated to smaller area depending on the size of display. Nevertheless, considering any type of user end device, the core of experiment remains the same. Examples of various types online experiments and laboratories can be find e.g. in (Ionescu et al., 2013), (Zubia and Alves, 2011), (Bisták, 2006), (Restivo et al., 2009), (Schauer and Ozvoldova, 2008), (Zolotova et. al., 2007).

The paper is oriented mainly to the phase of development of online experiments. It demonstrates how virtualisation of environment can help such development without influencing the real operation of the laboratory.

2. INITIAL ARCHITECTURE

The aim was to continue in the development that was realised in previous years (Janik and Zakova, 2013), (Magyar and Zakova, 2012), (Zakova and Sedlak, 2006). The online laboratory was realized using the classical client/server architecture that is shown in Fig.1. It is to see that all tasks are decentralized among several computers that are arranged into two layers.

Master server has three main roles: it serves as the web server of the laboratory, it is used as a computing centre and it takes care about the whole task management of the developed online laboratory.

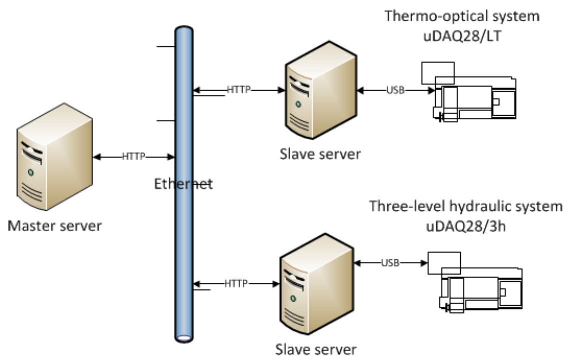

Fig. 1. Initial architecture of online laboratory

Since the master server is used both as a web server and a computation kernel of the online laboratory, all online interactive examples and virtual experiments illustrating various problems (mainly from the area of automation and control engineering) are placed here. Calculations are accomplished either directly in the developed application or in one of computer algebra systems that are installed on the server for this purpose. We oriented mainly to free and open software, so developer and user can find Maxima, Octave, SciLab or OpenModelica environment here.

On the other side the master server also manages approach to the remote experiments. Therefore the master server contains booking system of experiments, user management, etc.

According to the Fig. 1 the laboratory architecture consists of several slave servers, too. They take care about real experiments. Each of these servers communicates with the real plant. It also computes the control signal values that are used for controlling of plant outputs. If control algorithm is calculated inside of some software, this software (e.g. SciLab) is installed on the slave server, too.

It is possible to say, that in the introduced architecture the master server presents a gate to the remote experiments whose control is realised by slave servers.

All servers are communicating between each other via HTTP requests. Each computer is bundled with one operating system (OS) where all applications (used on the computer) are installed (see Fig.2). As far as the operating system we decided to use Linux Ubuntu LTS (Long Term Support). It is based on Debian, it is customisable, stable and has a lot of packages (applications) that can be easily accessible via repositories. It has extended support time and a good reputation also due to the support of specific hardware.

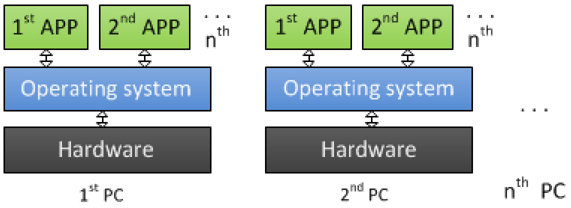

Fig. 2. Typical use of hardware and software

3. IS THERE REASON FOR CHANGE?

The architecture that was presented in previous section allows us to realise all experiments in the desired manner. However, the problem is that the existing online laboratory is all the time under development. All the time new technologies, new services, new experiments are developed and implemented. Very often they are prepared by regular students in frame of their bachelor or diploma thesis. Sometimes the development of the application requires accomplishing changes also to the configuration file of the operating system. It can influence stability of the whole server. In such a case it is not so easy to arrange that applications and experiments (developed sooner) will run 24 hours per day. It is difficult to have a redundant laboratory ecosystem only for development and testing purposes. Therefore we decided to prepare a virtual environment that will have exactly the same functionality and features as the original system. It will be used for developing, debugging and testing all required projects and the stability of the laboratory will be not touched. It means the server with such a virtualised environment can be used both for the development and production phase in the same time without a security risk.

Virtualised environment can run both on the master and slave server, as well. To prepare such ecosystem it is necessary to choose the suitable virtualization method and virtualization software. Our requirements were: to use open source technologies, to enable easy management of the environment, stable solution and long term sustainability.

4. VIRTUALIZATION METHODS

The virtualisation can be solved by several methods. They are introduced in this section.

4.1 Hardware emulation

The basic principle of hardware emulation is very simple. It consists in the effort to substitute a missing physical hardware component with its software equivalent. In this way we can emulate behaviour e.g. of some kind of communication process card or multimedia card. The virtualized hardware does not need to be physically located in the computer (server).

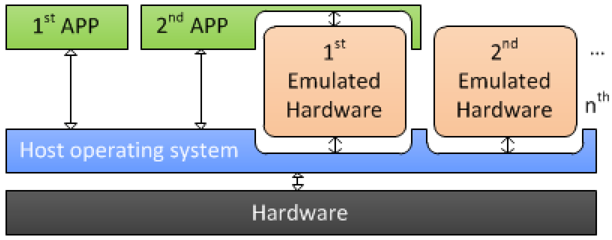

Fig. 3. Emulation of hardware

Since the hardware is emulated by its software equivalent, this type of virtualization is quite slow. The reason is that all tasks for emulated hardware component (e.g. input/output operations) have to be translated to the physical device and the translation is accomplished by the processor.

4.2 Software virtualization

Software virtualization does not emulate hardware but it uses only its abstraction for installation of operating system in a sand box that is called virtual PC.

Virtual PC is an application installed on the native operating system. It enables the exchange of tasks between host and guest operating systems. The guest operating system does not use an emulation of hardware but it uses embedded drivers and selected APIs (Application Programming Interface) of virtual PC to communicate with host hardware. Virtual PC enables to virtualize only operating system not hardware.

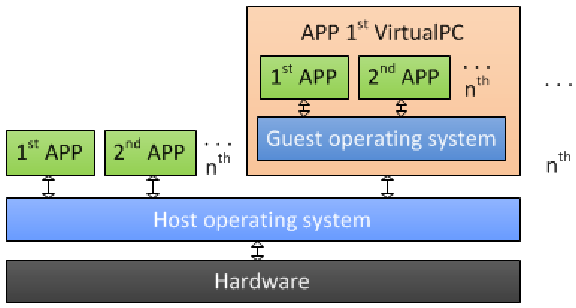

Fig. 4. Application providing software virtualization

4.3 Para virtualization

In difference to the software virtualization the para virtualization requires to interfere with the kernel of operating system.

Para virtualization can run on all computers but to achieve the best results it is good if virtualisation functions are supported by the system processor (e.g. Intel VT-X or AMD-V). The aim is to run several operating systems in parallel. One of these operating systems is acting as the master operating system (host OS). The activity of all other guest operating systems is coordinated by the hypervisor that is controlled by the host operating system. The hypervisor accomplishes input/output operations and tasks directly on hardware of the host system. It enables faster execution of guest operation system requests and better utilization of system resources.

Due to the virtualization layer that acts as an abstraction of the guest operating system, the para virtualization is more powerful than a full virtualization.

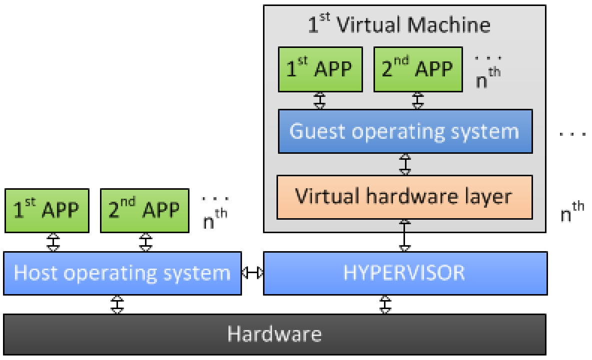

Fig. 5. Para virtualization in host operating system

4.4 Full virtualization

This type of virtualisation enables to emulate all hardware components of computer that are later used for installation of a guest operating system. The emulated hardware represents all parts of the real computer hardware, e.g. CPU, graphic card, mainboard, RAM, etc. In reality these components need not to be in the hosting hardware.

The host computer has a hypervisor layer which provides all needed services to the guest operating systems directly. The hypervisor layer is not the host operating system, it is really just a layer.

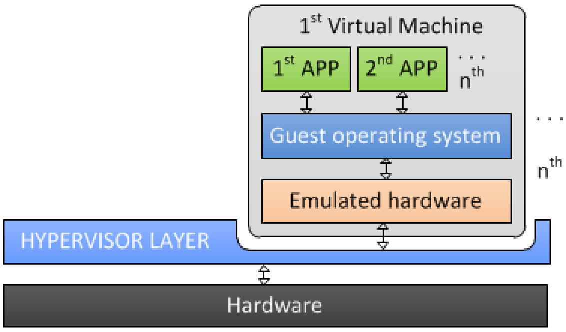

Fig. 6. Full virtualization with emulated hardware

Using associated emulated hardware, operating systems running in virtual machines act more native. Smaller slowdown can be seen when the guest is trying to access hosts peripheries. It is caused by the emulated hardware. Each I/O (input/output) operation needs to be translated – the space on the virtual disc has to be transformed to the place on the physical disc and it takes time.

Full virtualization has to be fostered by the suitable hardware (processor has to support the virtualisation process).

4.5 Summary

Our aim was to build uniform architecture for the whole online laboratory ecosystem. Therefore we decided to use the same virtualisation technology on both master and slave server, as well. To achieve the best possible performance (new plants and applications come into existence and number of experiments is growing), we came to a decision to use the full virtualization as the primary virtualization technique.

5. VIRTUALIZATION SOFTWARE

The selected method of virtualization has to be realized by appropriate software. There exist various proprietary and also free solutions (the overview is in Table 1). In this section we tried to consider those that are the most widely used.

5.1 VirtualBox

VirtualBox is an application that runs on the top of the operating system. It offers software virtualization and it can also support full virtualization using some extensions.

To achieve better performance results, VirtualBox Extension Pack has to be installed in the guest operating system (in virtual machine). The pack adds special support to guest operating system (for e.g. drivers and hardware abstraction). It results to the fact that guest has no information that it is running on the virtualized hardware.

VirtualBox is well known as easy solution for home desktop virtualization. Nevertheless, it can also be used for servers.

VirtualBox is usually controlled by GUI (Graphical User Interface). However, it can also be executed without using GUI extension (in so called headless mode). Then it is managed via CLI (Command-Line Interface).

5.2 VMware

VMware offers several solutions that can be used in frame of built online laboratory (see Table 1). They are oriented either to server virtualization (VMware ESX Server) or to desktop virtualization (VMware Player and VMware Workstation).

As well as VirtualBox, VMware Player and VMware Workstation applications run on the top of the host operating system and offer software virtualization. VMware player is controlled via GUI and VMware Workstation both via GUI and CLI, as well.

Table 1. Overview of virtualization software

Name		Type of virtualization	Host OS	Guest OS	License	Creator	Info
VirtualBox		Software/Full	Windows, Linux, Solaris, FreeBSD	Windows, Linux, Solaris, FreeBSD	GPLv2, proprietary with support	Innotek, now Oracle	www.virtualbox.org
VMware	Player	Software	Windows, Linux	Windows, Linux, Solaris, FreeBSD	Free	VMware	www.vmware.com
	Player Plus	Software			Proprietary		
	WorkStation	Software/Full					
	ESX server	Full/Para	No host OS				
KVM		Para	Linux, FreeBSD	Windows, Linux, Solaris, FreeBSD	GPLv2	Qumranet, now Red Hat	www.linux-kvm.org
XEN		Para	Linux, Solaris, FreeBSD	Windows (with Intel-VT-x and AMD-V), Linux, Solaris, FreeBSD	GPL, payed support	Xensource, now Citrix	www.xenproject.org

In comparison to VirtualBox, VMwares desktop applications are more powerful in graphic experience. They also allow easy installation of guest operating system with more sophisticated wizard which automatically installs the guest operating system with all additional extensions.

In difference to VMware Player and VMware Workstation VMware ESX Server application offers full virtualization. Management of the virtualization runs through standalone GUI desktop application.

5.3 KVM

KVM (Kernel-based Virtual Machine) uses para virtualization and requires also an installation of QEMU[1] (application for software emulation of hardware).

If the host processor does not have virtualization extension (such as Intel VT-X or AMD-V), QEMU is used for the processor emulation, too. This results in performance loss. On the other hand if the virtualisation extension is supported by the hardware, the guest operating system acts near to the native speed. It is caused by the fact that the virtualization runs directly on the level of processor.

KVM itself does not perform any emulation, it just prepares the space for hosts and provides interface for guest. It enables guests to run in virtual machines, feed them with simulated I/O, and provide video visualisation to host.

By default KVM is controlled via CLI. Its management can be extended using web interfaces or by installation of a standalone GUI desktop application.

5.4 XEN

XEN was originally designed to test a new compiled kernel besides the stable kernel. When the kernel of the host operating system boots, it also starts a XEN microkernel (so called main domain) including virtual machines (virtual domains) and other modules.

In comparison to VMware, XEN does not emulate all the hardware, but it provides para hardware. This solution offers faster and better response of the guest operating system, because it acts more natively. If the host processor supports virtualization extension (e.g. Intel VT-X or AMD-V), virtualization runs directly on the level of processor. If not, kernel patches need to be applied to the host operating system.

As well as KVM, XEN is also executed via CLI by default. Its management can be extended by web interfaces or also by installation of a standalone GUI desktop application. XEN is widely used with cloud technology, many web interfaces are very complex and not solving only the virtualization.

5.5 Summary

All introduced products allow to create snapshots of the guest operating systems. This feature is similar to backup of files on external disk storage. Created snapshots save a particular stage of a virtual machine and if necessary, they can be used to revert the virtual machine back to the state that was before. Snapshots can be performed anytime and they can allow students safe development of their applications also in the case when they need to interfere with the configuration of the guest operating system. In each situation they can go back to the previous stage.

After considering all features of the presented virtualization software, we decided to use VirtualBox. There were following reasons:

- easy to use – management via GUI or via web GUI,
- snapshots – easy and clever way of system backup and restore, tree structure of snapshots is also allowed,
- easy installation – no addition kernel patches of host operating system are required,
- good performance also on hardware without support of virtualization technology,
- software virtualization has wide compatibility with arbitrary operating systems and hardware components, in specific conditions full virtualization is also possible,

[1] http://qemu.org/

- platform independent, well documented, free and open source.

6. REALIZATION

Considering the full virtualization technology and VirtualBox environment we modified the architecture of our online laboratory (Fig.7). The virtualisation was realised on both servers (master, slave) and in each case it plays different role.

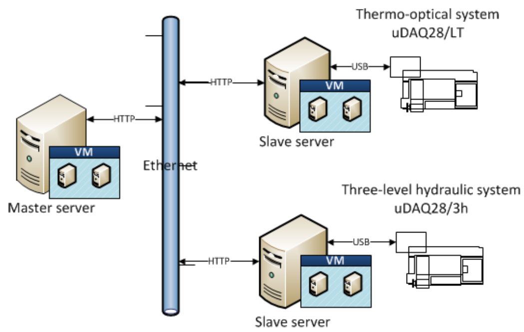

Fig. 7. Final architecture of online lab with virtualization

The virtualisation on the master server enables to run several operating systems (thanks to powerful hardware) that can be used by several students without performance loss. In such a way they can work parallel and not influencing each other. The guest operating system can also offer full desktop environment (Graphic User Interface) for easy operation of tasks. On the master server all tasks that are not connected with the control of a real device can be solved.

The slave server enables students to develop and test their applications directly on the real device. The design of the slave server is calculating with the fact that maximum two guest operating systems will run in parallel. The higher number of operating systems is not necessary, since only one plant is connected to each slave server and in each moment it can be available only to one user. This is ensured by booking system placed on the master server.

It is to note that slave servers (i.e. servers that communicate with real device) have one limitation. The VirtualBox has standard built-in support for communication between guest OS and host Serial Ports, USB or PCI. However, AGP cards and some PCI Express cards are not supported at the moment.

6.1 Virtualization Management

As it was mentioned before, VirtualBox software can be used in two ways: via standard graphical user interface (GUI) and via command line interface (CLI). Our aim was to incorporate the functionality of the programme into the web application. It should help course administrator to manage virtualisation process on the master server. The realisation was done using an extension phpVirtualBox (Moore, 2013) that creates a web interface of the VirtualBox software.

The included web interface enables: to create new virtual machines or to delete old ones, to manage the running of existing machines, to stop, shutdown, restart and suspend virtual machines, to create snapshots and to restore the previous stage, to manage users and rights of users that can work with established virtual machines.

The behaviour of VirtualBox kernel is influenced by custom CLI commands.

6.2 Management of installation packages

Typical installation of any computer (including servers) requires installing the operating system and set of additional software applications. All steps are usually done manually under human supervision and the whole installation and post-installations setups take some time. In addition, not skilled user can have problem to set up all parameters and options.

Linux based software can be installed in three ways: from distribution repository, from binary package or using compilation from a source code. The installation from distribution repository enables to install last stable versions of software that are regularly updated. The installation from binary packages allows installing previous versions of software on the server. It can be useful and practical in the case when the newest version of software can cause incompatibility with other applications on the server. The compilation of software from source code enables to use processor-specific optimizations, to respect hardware support, to accomplish software customizations, to fix bugs, etc.

Thanks to Linux style of software installation it is possible to prepare batch file script that will fully automate all post-install setups according to requirements of the administrator. In spite of the fact that the automated script can combine all types of installation techniques, its manual preparation is not so easy. It requires the knowledge and skills in Linux scripting. The created web application can facilitate the whole preparation process by automatic generation of the installation package (standard or custom). It can be done via user friendly GUI and wizard (Fig.8).

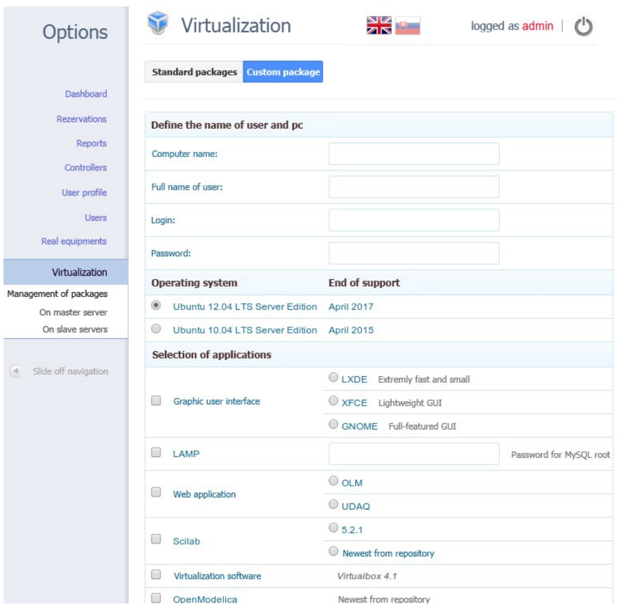

Fig. 8. Dashboard of managing installation packages

The standard package includes pre-configured default set of selected applications and settings for easy deployment of a computer or virtual machine.

The custom package is generated on the base of user requirements. The user needs to specify at least applications that should be installed. Then, the installation package is generated on the server and the user is provided by the URL address that specifies the generated package location.

6.3 Automatic installation

Pre-prepared installation packages can facilitate the whole installation process since the installation can be done automatically. The installation package can be used both for installation of a new physical computer or virtual computer, as well. However, it is necessary to respect that physical (or virtual) computer needs to have access to Internet and be connected to the master server.

In Fig.9 one can see a flowchart for installation of a new physical slave server. The similar procedure can be applied in the case of a new virtual machine installation.

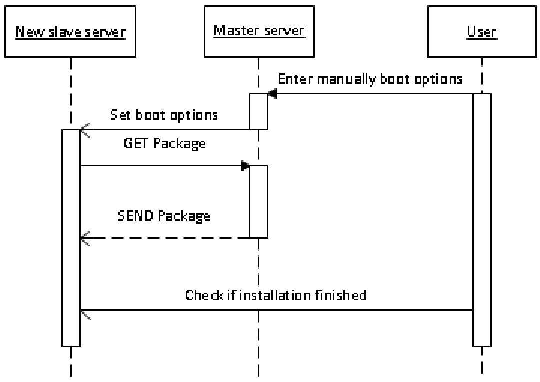

Fig. 9. Installation process of new virtual slave server

The installation process starts after manual entering the URL address corresponding to the installation package that was generated in the previous step. In this way the boot options are specified. Then, the installation request is sent to the master server, who takes control over the whole installation and handles the process instead of the administrator. Later, the slave server asks for sending the installation package. After the installation process unpacks the package, answers all installation questions, installs additional software and does post-install setups.

The administrator can only glance at the process of installation without a necessary intervention. When the installation process is finished, the new computer is ready to use. Thanks to this technique, it is also possible to install several computers at once. It can spare time of administrator.

7. CONCLUSIONS

The virtualisation is used very often by various companies to utilize better available hardware. The paper showed that this technology can be used in online laboratory, too. It offers a possibility to use the same hardware both for development and real service of online experiments. It enables to spare financial resources, equipment, facilities and also available laboratory rooms.

The introduced solution brings benefit both for students and teachers or course administrators, as well. Students can feel more free when they are testing their application in virtual environment without the risk that they influence running of the whole server. On the other hand administrators spare time that would be needed to restore the corrupted server back to its normal operation stage.

ACKNOWLEDGEMENTS

The work has been supported by the grant KEGA 032STU-4/2013 and grants APVV-0772-12 and APVV-0343-12.

REFERENCES

Bisták, P. (2006). Remote Control of Thermal Plant Using Easy Java Simulation. *Int. Conf. on Interactive Computer Aided Learning ICL*'06, Villach, Austria.

Frťala, T. (2013). *Virtualization of enviroment for online computations and experiments*. Diploma thesis, FEI Slovak Univ. of Technology in Bratislava, in Slovak.

Helmke, M., Graner, A., Rankin, K., Benjamin Mako Hill, B. M., Jono Bacon, J. (2012). *The Official Ubuntu Book (7th Edition)*. Prentice Hall, ISBN 978-0133017601

Huba, M., Šimunek, M. (2007). Modular Approach to Teaching PID Control. *IEEE Transactions on Industrial Electronics*, Vol. 54, No. 6, pp. 3112-3120.

Ionescu, C.M., Fabregas, E., Cristescu, S.M., Dormido, S., De Keyser, R. (2013). A Remote Laboratory as an Innovative Educational Tool for Practicing Control Engineering Concepts. *IEEE Trans. on Education*, vol.56, no.4, pp.436,442.

Janík, Z., Žáková, K. (2013) A Contribution to Real-Time Experiments in Remote Laboratories. *International Journal of Online Engineering*, Vol. 9, Issue 1, pp. 7-11.

Magyar, Z., Žáková K. (2012). SciLab Based Remote Control of Experiments. 9th IFAC Symposium on Advances in Control Education ACE'12. Nizhny Novgorod, Russia.

Moore, I. (2013). phpVirtualBox - A web-based front-end to VirtualBox. http://sourceforge.net/projects/phpvirtualbox

Papadopoulos, V., Leva, A. (2013). *Laboratories over the network: from remote to mobile*. In 10th IFAC Symp. on Advances in Control Education,pp.84-89, Sheffield, UK.

Restivo, M. T., Mendes, J., Lopes, A.M., Silva, C.M., Chouzal, F. (2009). A Remote Lab in Engineering Measurement, *IEEE Trans. on Industrial Electronics*, vol. 56, no.12, pp. 4436-4843.

Schauer, F., Ožvoldová, M., Lustig, F. (2008). Real Remote Physics Experiments across Internet – Inherent Part of Integrated E-Learning. *Int. J. of Online Eng.*, 4, No 2.

Zolotová, I., Bakoš, M., Landryová, L. (2007). Possibilities of communication in information and control systems. *Annals of the university Craiova, Series: Automation, Computers, Electronic and Mechatronic*, Vol.4(31), No.2, pp.163-168, ISSN 1841-062.

Žáková, K., Sedlák, M. (2006). Remote Control of Experiments via Matlab. *Int. Journal of Online Engineering*, 2, No. 3.

Zubía, J. G., Alves G. R. (eds.) (2011). Using Remote Labs in Education: Two Little Ducks in Remote Experimentation. University of Deusto, Bilbao, ISBN: 978-84-9830-335-3.