A Contribution to Real-Time Experiments in Remote Laboratories

http://dx.doi.org/10.3991/ijoe.v9i1.2339

Zoltán Janík and Katarína Žáková
Slovak University of Technology, Bratislava, Slovakia

***Abstract*—The paper is focused on realization of hard real-time control of experiments in on-line laboratories. The presented solution utilizes already developed on-line laboratory portal that is based on open-source Scilab environment. The customized solution is based on Linux RTAI platform with RTAI-XML server, Comedi and jRTAILab with support of ScicosLab environment. It generates real-time executable code that is used to operate student experiments performed on Humusoft CE152 Magnetic Levitation plant.**

***Index Terms*—computer aided engineering, control design, online services, student experiments, RTAI.**

I. INTRODUCTION

Nowadays, one can follow an emerging trend of increasing interactivity in educational process using various multimedia teaching materials, animations and on-line experiments (virtual, remote, hybrid). Especially, the on-line experimentation is very important mainly at technical universities. The interactivity helps students to achieve necessary skills and understand better topics that are taught.

In this paper, we focus on the demonstration of one experiment in the area of automation and control. The experiment is realized by the remote server that communicates with web-based client application. Such kind of experiments enables students to be in contact with real devices without their presence in the laboratory rooms. The disadvantage of remote experimentation is that students can lose the feeling of staying in the laboratory since all tasks (including model and parameter identification, control, visualization of results, etc.) can be done only via command window on their personal computer. In spite of that we tried to achieve hard real-time control on the server side to offer students the closest possible contact with the plant.

II. REAL-TIME SUPPORT

Desktop simulation environments such as Simulink in Matlab or Scicos, respectively Xcos in Scilab already contain tools for real-time control (Simulink Coder – formerly known as Real-Time Workshop, respectively Scicos-HIL). However, these tools and their equivalents are dependent on the operating system they are running on. Thus, in addition to real-time support in applications, it is also necessary to consider real-time control support directly in the operating system. Windows, Mac OS or standard Linux distributions do not match this request. For such reason, it is necessary to use operating system with patched kernel. Very useful is for example an open-source solution – the patched Linux with RTAI platform that we currently use in our department.

Real-Time Application Interface for Linux (RTAI) provides guaranteed hard real-time scheduling with all of the features and services of standard Linux with necessary hard real-time extensions. The most important feature of RTAI is the ability to use a hard real-time code both in the kernel space and in the user space while using a single LXRT (LinuX Real-Time) scheduler.

In RTAI-patched Linux, the operating system is run as the lowest priority task of a small real-time operating system. There are no changes to its operation from the viewpoint of the user or the kernel, except that it is permitted to execute only when there are no real-time tasks executing. Thus, special real-time tasks may execute whenever needed, regardless of what other tasks Linux may be performing.

RTAI extension for Linux [2] is being developed since 1999 and it has been created as an environment for implementing inexpensive technique for data acquisition and systems for digital control. RTAI's core is distributed under General Public License (GPL) and the user part under Lesser General Public License (LGPL). RTAI is a part of Real-Time Suite [6] together with RTAI-Lab, Comedi and other supporting software.

III. SCIWL ON-LINE LABORATORY

Our effort is focused on deployment of Real-Time Suite to one of our current on-line laboratory web portals that allow students to perform experiments on real plants. The laboratory portal is called SciWL and it is described in [13]. The portal is based on open-source technologies, such as Apache web server, PHP scripting language, MySQL database server and Scilab environment. It provides communication link between Scilab and the client web application for uploading custom control algorithms and for continuous observation of measured results. User is allowed to modify either the pre-defined PID controller by changing its parameters, or to define custom algorithm. Creating a custom control algorithm can be achieved by inserting Scilab code that is executed in Scifunc block during the experiment. In addition to these two possibilities, we have added another way to modify the control algorithm. The new option enables to change the block structure of the controller in the graphical user interface by using drag & drop method. It uses a graphical editor embedded directly into the client web application [10], [11]. The editor offers comfortable way to modify the block scheme with the used controller in a similar way that desktop solutions offer (e.g. Xcos). Consequently, the end users do not have to have a local

installation of Scilab/Xcos if they need to define custom controller that is more complex than the pre-defined PID. Another advantage of this solution is that the user modifies only the part of the scheme that contains the controller. Thus, the user can focus on the important part of the scheme and he does not have to think about blocks that have no effect on the control (e.g. blocks for communication with a real plant). The rest of the scheme remains untouched, which is also a security benefit (user cannot modify the scheme in a way that could harm the real equipment).

In the present time, SciWL portal covers two experiments (thermo-optical plant and hydraulic plant) that use USB bus for communication with the server. Our aim is to extend the number of experiments and to enable remote control of the magnetic levitation plant CE152 provided by Humusoft company ([9], see Fig. 1) that enables to demonstrate control problems associated with nonlinear unstable systems. In difference to previous devices, this plant communicates via 14bit Humusoft MF624 communication board. The fact that the magnetic levitation is a fast system makes communication and control via USB almost impossible due to large latency of USB protocol.

Considering two plants that were primarily implemented in the SciWL portal (the thermo-optical and hydraulic plant), the RTAI control was not applicable because the USB protocol is not suitable for communication in hard real-time mode. However, this standard and reliable solution is a good choice for control of the magnetic levitation plant. We have chosen the Real-Time Suite as a supporting solution for implementation of the magnetic levitation into SciWL.

IV. IMPLEMENTATION OF HARD REAL-TIME CONTROL OF MAGNETIC LEVITATION IN SCIWL

The magnetic levitation plant is one-dimensional system consisting of metal ball, copper coil and inductive ball position sensor. This system offers to control the ball height using the voltage/current convertor. Fig. 2 illustrates the on-line environment for work with the magnetic levitation experiment.

Implementing the magnetic levitation experiment into the on-line laboratory environment required installation of Real-Time Suite and Comedi driver for Humusoft MF624 DAQ board on the laboratory server. We have combined the Comedi driver described in [15] with standard RT Suite installation together with some minor modifications required by the MF624 driver (e.g. running Comedi in legacy mode).

Each block scheme that we want to run in the hard real-time mode has to be compiled into a binary executable code using Scicoslab CACSD environment (which is very similar to Scilab). Thus, each user-modified controller has to be automatically compiled before its execution on the laboratory server. As it was mentioned before, the SciWL portal offers several possibilities. User can choose either the standard PID controller, own controller written as Scicoslab code in Scifunc block, or a custom controller in a form of a block scheme.

The advantage of a PID controller is that it never changes its structure – only controller parameters change. Thanks to this fact, the PID controller block scheme does not require to be recompiled after any change of its properties. For this reason, one can always use the same

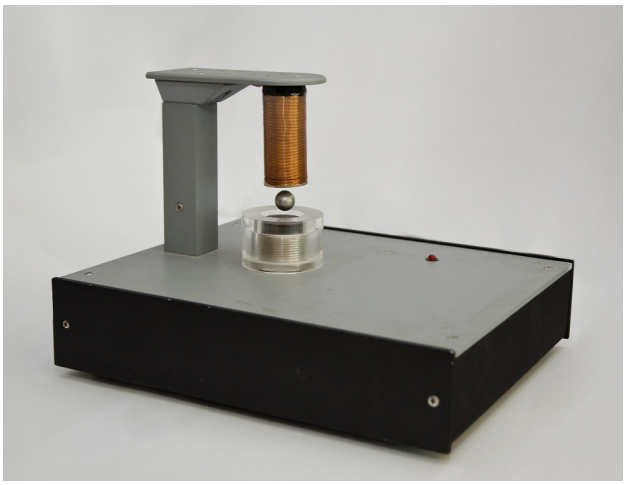

Figure 1. CE 152 Magnetic Levitation

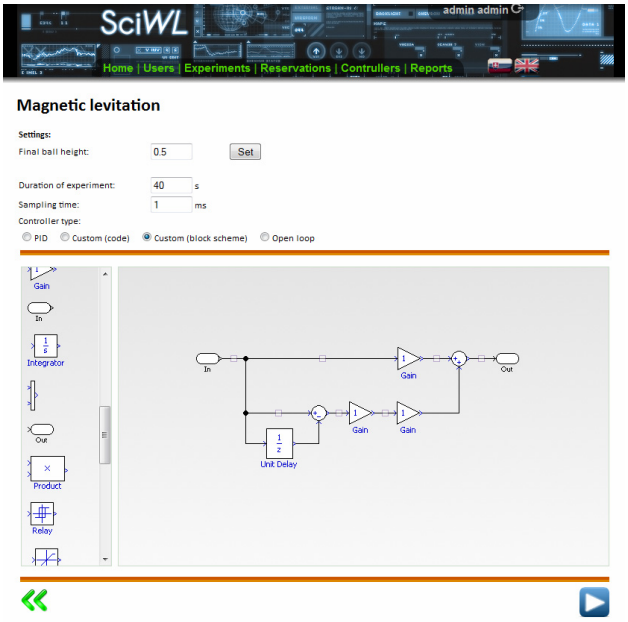

Figure 2. Screenshot of Magnetic levitation experiment in SciWL portal

binary file for PID controller. However, the other types of custom controllers may cause significant changes in the scheme structure and therefore they require recompilation almost every time the user changes the control algorithm.

The current real-time server configuration is based on Ubuntu 10.04 LTS with patched Linux kernel 2.6.32-rtai. The server uses RTAI 3.8.1, Comedi 0.7, ScicosLab 4.4.1 and RTAI-XML 1.0.

V. LOCAL REAL-TIME EXPERIMENTS

Standard scenario of building an RTAI-based experiment offline using local desktop installation of Scicoslab/Scicos is similar to this procedure: User creates a block scheme in Scicos (e.g. in Fig. 3) with all necessary blocks for controller, communication with the real plant (analog data input and output), experiment monitoring (scopes), etc. When the scheme is finished, user selects all blocks except the clock and converts them into a superblock. The content of the superblock can be seen in Fig. 3 where the clock is replaced by the red clock input block.

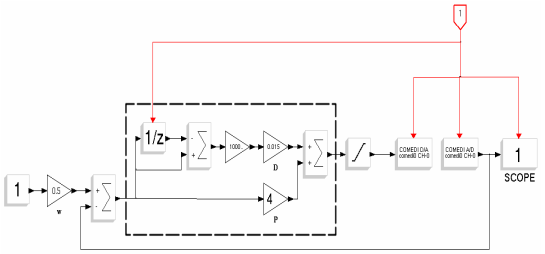

Figure 3. Scheme of superblock for magnetic levitation control

After this step, he or she highlights the super block and chooses "Set Target" from the RTAI menu and "RTAI CodeGen" afterwards. Scicoslab starts the compilation and informs the user about the progress. The compilation results in creating of a binary executable file that can be run outside the Scicoslab's environment. This procedure is illustrated in flowchart in Fig. 4 where each step has to be done by user himself.

VI. REAL-TIME EXPERIMENTS IN ON-LINE LABORATORY

For obvious reasons, the procedure described in previous section is not applicable for experiments implemented in on-line laboratory where each step except the scheme design has to be done without user's interaction and it requires certain level of automation.

Firstly, the user does not even have to create the whole block scheme but only the single part with the controller (e.g. the part of the scheme marked with dashed rectangle in Fig. 3 which has the same structure as the scheme in Fig. 2). The rest of the scheme can be pre-prepared and the user-defined controller is automatically placed into this scheme by an automated script.

Secondly, the user does not have to concern about creating superblocks and other standard procedures that are necessary to build the scheme for RTAI properly. The final scheme with user's control algorithm has to be compiled also without user's interaction.

The standard scheme compilation in local Scicoslab's desktop environment is performed by a macro saved in RTAICodeGen_.sci file. However, a customized compilation macro is required for the purposes of on-line laboratory due to the fact that the standard method uses a number of GUI interactions that are not applicable in text mode of Scicoslab that we are using for automatic scheme compilation.

When the user wants to run the experiment with a modified controller, the on-line laboratory web page sends a request to the laboratory server. The request contains the block scheme of user-defined controller (in a format that is suitable for transfer via HTTP) and metadata containing additional scheme information and experiment settings. The remote server combines the pre-defined framework scheme (the one with communication blocks, measurement, etc.) with the user-defined scheme received via HTTP. Afterwards, it remotely executes Scicoslab with special startup script as an argument. The Scicoslab is executed in non-interactive mode without loading its GUI. It can be realized by the command:

```
scicoslab -nwni -f "startup.sce"
```

The startup script is responsible for loading all necessary macros and functions that are normally available in desktop version of Scicos:

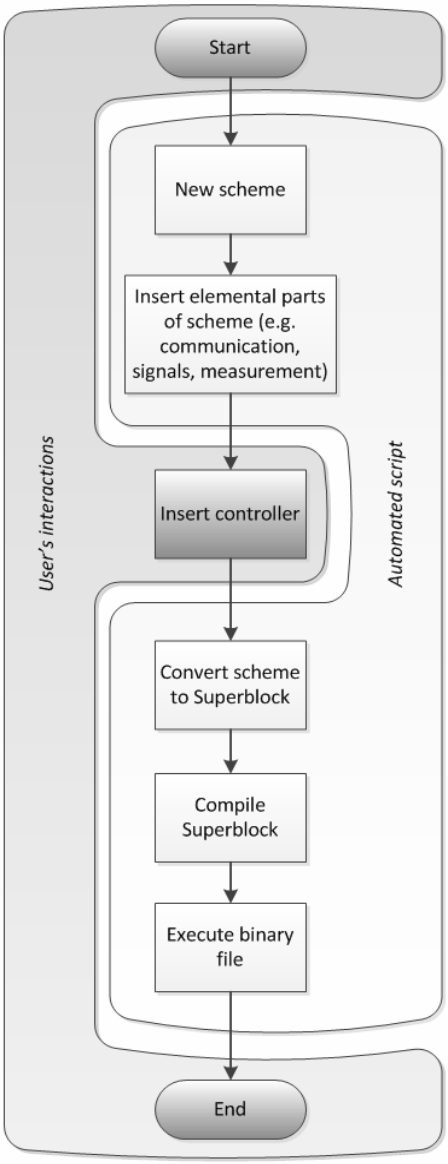

Figure 4. Flowchart – live cycle of a scheme with custom controller

- The files containing required definitions of functions are available in macros/scicos directory (inside Scicoslab's root directory) and they are loaded with the standard load function:
  ```
  load SCI/macros/scicos/lib;
  ```
- Regarding the fact that the compilation will require the functions for each block that is present in the compiled scheme, the startup script has to load also the definitions of block functions in macros/scicos_blocks directory and in each of its subdirectory. To prevent loading each subdirectory separately, all functions can be loaded with the following command:
  ```
  exec(loadpallibs,-1);
  ```
- An important prerequisite for running the compilation is RTAI-Lib toolbox [6] loaded into Scicoslab environment. In the desktop solution, the RTAI-Lib can be loaded from Toolboxes menu after clicking the RTAI toolbox option. This step may be substituted with command-line equivalent that is also used in our startup script:
  ```
  exec('SCI/contrib/RTAI/loader.sce');
  ```

After this step, the non-interactive version of Scicoslab is ready to perform tasks related to work with Scicos schemes and RTAI toolbox. However, we cannot rely on the standard way of scheme compilation as it was mentioned earlier in this section. As a result, we need to load and use our customized compilation macro called RTAI-CodeGenCustom. The core aspects of the customization are removing all GUI or mouse interactions with user and other modifications related to absence of GUI version of Scicos (for example, the selection of compiled superblock is automatic instead of the selection performed with mouse). The customized file must be loaded consequently after the RTAI-Lib toolbox is loaded in Scicoslab:

```
exec('SCI/contrib/RTAI/macros/RTAICodeGen
Custom_.sci');
```

Finally, the startup script loads the combined Scicos scheme, executes the custom real-time code generation and terminates the Scicoslab environment:

```
load("scheme.cos");
RTAICodeGenCustom_(function parameters);
exit;
```

The whole process can be observed in the console window in SciWL portal environment where the user can see the same output as in the standard Scicoslab desktop version. An example of such text output can be seen in Fig. 5.

After successful compilation, the binary file is ready to be executed. We use RTAI-XML server [6] to access the experiment remotely. When a SciWL client connects to RTAI-XML server, the compiled scheme is executed and it starts to exchange data continuously between the on-line laboratory portal and the experiment itself. The experiment can be monitored using the open-source jRTAILab applet [6] as seen in Fig. 6. We are also developing custom interface for monitoring RTAI experiments.

The described workflow in online environment is also shown in Fig. 4. The user's interactions are marked with dark background. The rest operations are performed automatically by the server scripts.

VII. RESULTS

Previously, the magnetic levitation experiment has been running on the Windows server supported by Matlab's Real-Time Workshop. We have used the same PC configuration to be able to compare these two different solutions – RT Workshop on Windows vs. Linux RTAI. The laboratory server is powered by a dual-core Intel Pentium 4 2,8 GHz (Prescott) processor with 12kB L1, 16kB L2 and 1024 kB L3 cache, Intel 865PE chipset and 2 slots of 256 MB PC3200 DDR SDRAM Apacer (200MHz; 2,5-4-4-8 timing; Non-ECC).

The standard sampling time of 1 millisecond for magnetic levitation experiment in RT Workshop on Windows XP can be shortened down to 50 microseconds but with lower quality of control at shorter sampling time values (which is caused by occasional misses of calculation deadlines specified by timing constraints given to real-time task). In addition, sampling times near this value and lower have significant difficulties with maintaining desired ball height. Further decreasing of sampling time causes the host PC to freeze (hard reset required).

When we use Linux RTAI with the same solver settings (ode4 – Runge-Kutta) as on Windows, the experiment

```
        -------------------------------------------
                      ScicosLab-4.4.1
                   A Simple Matter of Conviction

             Copyright (c) 1989-2011 (INRIA, ENPC)
                (Tue Apr 19 12:54:45 CEST 2011)
        -------------------------------------------


Startup execution:
  loading initial environment
shared archive loaded
Link done

 Scicos-RTAI Ready

 Executing "[CCode,FCode]=gen_blocks()"...

 Executing "[Code,Code_common]=make_standalonert();"...

 Executing "files=write_code(Code,CCode,FCode,Code_common);
      "...

 Executing "Makename=rt_gen_make(rdnom,files,archname);"...

 Executing "ok=compile_standalone();"...

 Compiling standalone
```

Figure 5. Console output from Scicoslab

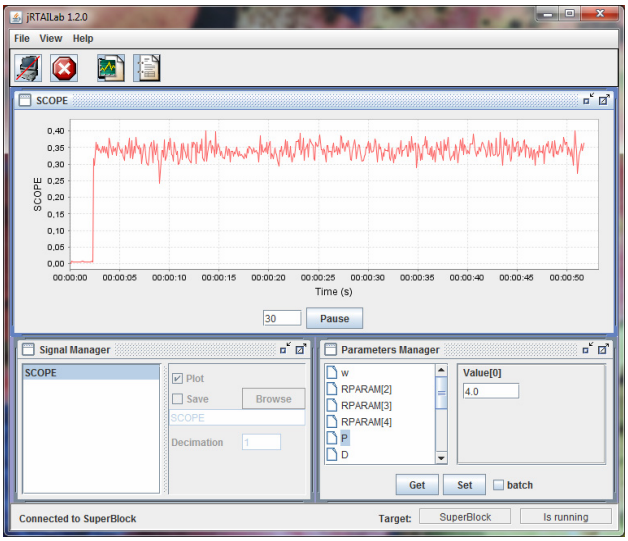

Figure 6. Remote observation of the experiment in jRTAILab

works without problems with sampling periods down to 40 microseconds, which is near the physical limits of the MF624 DAQ board. Sampling periods around 30 and less milliseconds cause that the host operating system stops responding to interrupts from keyboard and mouse and consequently freezes (hard reset required). Due to the limits of the DAQ board, we could not test shorter sampling times. On the other hand, the tested values are far below the maximal sampling times that are required to properly control the magnetic levitation plant.

VIII. CONCLUSION

Until now, the hard real-time control experiments were almost exclusively a domain of desktop systems. However, more and more real-time experiments appear in public on-line laboratories. Unfortunately, they lack certain level of interaction with user since the user can execute usually only a pre-defined experiment with ability to change only a limited number of properties. Thanks to the newly added graphical editor of block schemes for

building custom controllers and thanks to the support of additional plant with the support of hard real-time control, we have managed to enhance possibilities of on-line laboratory system and we have demonstrated a new possible trend in evolution of on-line laboratory systems. Together with the rest of features of the SciWL portal, this solution could be used as a very flexible supporting application for on-line courses in the area of automation and control.

REFERENCES

[1] P. Bisták, "Remote Control of Thermal Plant Using Easy Java Simulation," *Int. Conf. on Interactive Computer Aided Learning - ICL*'06. Villach, Austria, 2006.

[2] R. Bucher and S. Balemi, "Scilab/Scicos and Linux RTAI – A Unified Approach," *IEEE Conference on Control Applications*. Toronto, Canada, 2005.

[3] S. L. Campbell, J.-P. Chancelier, and R. Nikoukhah, "Modeling and Simulation in Scilab/Scicos with ScicosLab 4.4," ISBN 978-1-4419-5526-5, 2010.

[4] Digiteo, SciLab website, http://www.scilab.org, 1989-2011

[5] J. Erdfelt and D. Drake, LibUSB Homepage, Online, http://www.libusb.org/

[6] A. Guiggiani, "RealTime Suite," Online, http://www.rtaixml.net/realtime-suite/, 2011.

[7] I. Gustavsson, K. Nilsson, J. Zackrisson, L. Hakansson, J. G. Zubía, G. R. Alves, U. Hernandez, R. J. Costa, T. Lago, and I. Claesson, "The VISIR Open Lab Platform 5.0 - an architecture of a federation of remote laboratories," 8*th Intern. Conference on Remote Engineering and Virtual Instrumentation* (*REV*'11), Brasov, Romania, 2011.

[8] M. Huba and M. Šimunek, "Modular Approach to Teaching PID Control," *IEEE Transactions on Industrial Electronics*, ISSN 0278-0046, Vol. 54, No. 6, pp. 3112-3120, 2007.

[9] Humusoft, "CE 152 Magnetic Levitation Model website," Online, http://www.humusoft.cz/produkty/models/ce152/, 1991-2012.

[10] Z. Janík and K. Žáková, "Online Design of Matlab/Simulink Block Schemes," In: *International Journal of Emerging Technologies in Learning (iJET)*. ISSN 1863-0383. - Vol. 6, Special Issue 1, pp. 11-13, 2011.

[11] Z. Janík and K. Žáková, "Online Design of Matlab/Simulink and SciLab/Xcos Block Schemes," *14th International Conference on Interactive Collaborative Learning and 11th International Conference Virtual University*. Piešťany, Slovakia. ISBN 978-1-4577-1746-8. p. 241-247, September 2011.

[12] M. Jáno and K. Žáková, "SciLab Based Remote Control of Thermo-Optical Plant," *Int. Journal of Online Engineering* (*iJOE*), Vol. 7, No. 4, pp. 10-15, 2011.

[13] Z. Maygar and K. Žáková, "SciLab Based Remote Control of Experiments," *9th IFAC Symposium on Advances in Control Education ACE'12*. Nizhny Novgorod, Russia, 2012.

[14] H. V. Neher, "The Role of Experimental Work," *American Journal of Physics*, 30 (3). pp. 186-190. http://resolver.caltech.edu/CaltechAUTHORS:NEHajp62a, 1962. http://dx.doi.org/10.1119/1.1941966

[15] P. Píša and R. Lisový, "COMEDI and UIO drivers for PCI Multifunction Data Acquisition and Generic I/O Cards and Their QEMU Virtual Hardware Equivalents," In: *Thirteenth Real-Time Linux Workshop*. Schramberg. pp. 101-106. ISBN 978-3-00-036193-7, 2012.

[16] M. T. Restivo, J. Mendes, A. M. Lopes, C. M. Silva, and F. Chouzal, "A Remote Lab in Engineering Measurement," *IEEE Trans. on Industrial Electronics*, vol. 56, no.12, pp. 4436-4843, 2009. http://dx.doi.org/10.1109/TIE.2008.2011479

[17] F. Schauer, M. Ožvoldová, and F. Lustig, "Real Remote Physics Experiments across Internet – Inherent Part of Integrated E-Learning," *Int. Journal of Online Engineering* (*iJOE*), 4, No 2, 2008.

[18] C. Schmid, „Internet - basiertes Lernen," *Automatisierungstechnik*, 51, No. 11, pp. 485-493, 2003. http://dx.doi.org/10.1524/auto.51.11.485.19585

[19] P. Schmidt, "Creating a C Function Block in Scicos," Online. http://www scicos.inria.fr/ScicosCBlockTutorial.pdf, 2009.

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

[21] J. G. Zubía and G. R. Alves, *Using Remote Labs in Education: Two Little Ducks in Remote Experimentation*. University of Deusto, Bilbao, ISBN: 978-84-9830-335-3, 2011.

AUTHORS

Z. Janík and **K. Žáková** are with the Faculty of Electrical Engineering and Information Technology, Slovak University of Technology, Ilkovičova 3, 812 19 Bratislava, Slovakia (e-mail: zoltan.janik@stuba.sk, katarina.zakova@stuba.sk).

This work has been supported by the Slovak Grant Agency, Grant VEGA No. 1/0656/09. It is an extended and modified version of a paper presented at the International Conference on Interactive Collaborative Learning (ICL2012), held 26 - 28 September 2012, in Villach, Austria. Received 29 November 2012. Published as resubmitted by the authors 24 January 2013..