# Remote control software for thermo-optical plant

**M. Kalúz** [*], **Ľ. Čirka** [*], **M. Fikar** [*]

[*] *Slovak University of Technology in Bratislava*
*Faculty of Chemical and Food technology*
*e-mail: {martin.kaluz, lubos.cirka, miroslav.fikar}@stuba.sk*
*url: http://www.kirp.chtf.stuba.sk*

**Abstract:** This paper describes development of a remote control laboratory. The main aim of this work is to create computer software for remote access and control of thermo-optical device uDAQ28/LT. We describe solutions that have been chosen for meeting the requirements for a fully user-friendly and an easy to use application. We choose the software programming platform Adobe Flash for client side application development and we develop a solution based on technologies PHP, MySQL, and MATLAB for server side.

*Keywords:* Remote Laboratory, Process Control

## 1. INTRODUCTION

Today, in a field of automation and process control education, it is necessary to create quality conditions, also by enabling students to use real experimental devices, on which they can carry out their experiments and improving their skills. This aim can be fulfilled by several different ways.

One of the most used solution is real technological laboratory. However, from the students view, it can be used only during school opening hours and it depends on accessibility of teachers. But there is another way how to provide students access to real experiments. It is remote laboratory that can be used for control of real technological plants from any location with access to Internet.

There are some solutions of remote laboratories over the Internet, created by software developers from another universities.

Interesting solution comes from FEI STU in Bratislava, where remote control software for the same thermo-optical device uDAQ28/LT has been developed. This solution is based on Java Applet and Java Servlet communication, and MATLAB used as main control software. Used technology is described in detail in Bisták and Beránek (2006). This solution was later extended by new reservation system (Bisták and Čirka, 2009) and has been applied to another type of device (hydraulic plant) (Žilka et al., 2008).

A DSP(Digital Signal Processor)-based remote control laboratory was proposed by Hercog et al. (2007). Their solution is based on in-house developed control hardware with MATLAB as control software and client application created in LabVIEW graphical programming environment. It is used for control of some real systems like electric motor, robot mechanism, power converter, etc.

Another remote control laboratory was created at Department of Industrial Systems Engineering of University Miguel Hernández in Elche, Alicante, Spain. Puerto et al. (2010) suggested remote control of two mechanical devices, a DC motor and sliding cylinder. The client side of software is built on HTML/PHP technologies. The communication part of software is based on Common Gateway Interface (CGI) and the main control software is MATLAB.

We develop a remote laboratory that can be accessed by Internet, using a web browser, but propose a different combination of technologies than above mentioned. Our remote control application is created as Adobe Flash program, framed in regular web page, accessible through e-learning system Moodle (Dougiamas, 2005). It also provides a video stream from remote web camera that is aimed at controlled device, so the remote user can observe the behavior of the experiment in real time. The Flash application is used on the client side. The server side with the real controlled device uDAQ28/LT consists of PHP/MySQL Apache-type server containing programs serving communication between every parts of our software solution, MATLAB that directly controls remote device, and Java database software drivers for providing access to MySQL from MATLAB.

In comparsion with client applications from Bisták and Beránek (2006), and Puerto et al. (2010), based on HTML/PHP and Java Applet, the Adobe Flash programing platform is more suitable for graphical design development and also it provides more options for graphical user interface (GUI) customization. From the simplicity point of view and software/hardware requirements, Java and HTML/PHP solutions seems to be better, but not so far. Hercog et al. (2007) used for building GUI the LabVIEW environment, which is directly designed for such use, but it is questionable if LabVIEW is an appropriate environment for building features like interface for communication between users, dynamically generated forms and fully animable GUI. The main advantages of our solution is the possibility of direct storage and data managment from experiments. This advantage stems from the using MySQL

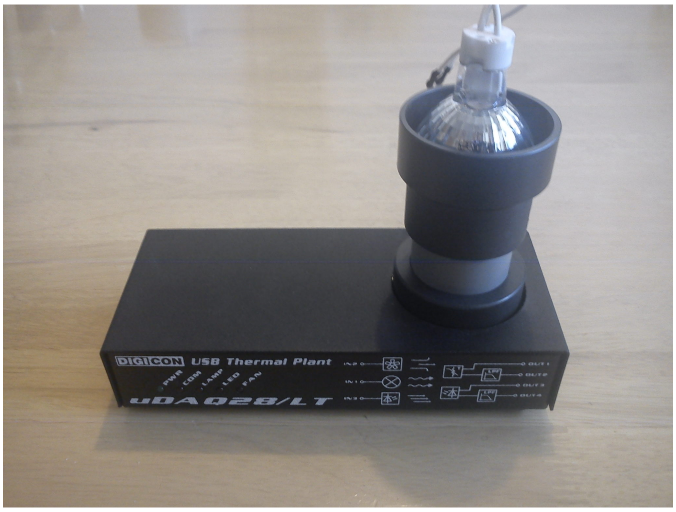


Fig. 1. Thermo-optical device uDAQ28/LT

database system, also as communication channel. Another difference is that we use technology based on asynchronous communication, so we can prevent the crash of experiment due to short-time losses of connection between client and server. MySQL, PHP layer, and MATLAB are located on the same physical server, therefore in the case of link outage between client PC and server, the local communication channels on server side stay unaffected by this issue, so the measuremnet remains running and data are collected in MySQL. In this case, user does not lose the measured data. The experiment is turned off only in case of longer-time link outage. Using PHP is fast and efficient way to secure communication between Flash application on client side and MySQL.

This is not our first attempt of using Adobe Flash platform in software development. Recently, we created an virtual laboratory based on Flash (Čirka et al., 2010), and MATLAB computation online laboratory based on Java (Kalúz et al., 2010).

## 2. UDAQ28/LT DEVICE

For purposes of remote laboratory and application testing, we chose a thermo-optical device uDAQ28/LT (Fig. 1) created by DigiCon Corporation (Huba et al., 2006). It is originally designed for education in fields of automation and process control. Device can be connected directly to computer by universal serial bus (USB) and after simple driver installation it can communicate with MATLAB. Manufacturer provides with device the installation CD including software drivers, Simulink schemes, and user manual. Plant can be controlled by 3 analog inputs (bulb voltage, fan voltage, and LED voltage), and has 4 primary outputs (light intensity, temperature inside light tube, fan rpm, and voltage taken by fan).

## 3. SOFTWARE SOLUTIONS

Our software solution combines several technologies, which were chosen depending on selected criteria. The main criteria were data storing ability, connection robustness, program processing rate, easy-to-use graphical user interface, and low system and hardware requirements.

In this section, we will explain the realization of links between every part of chosen technology and its principles.

Every user instruction from Flash application is processed on server side by PHP scripts.

PHP is used in three different ways.

(1) The first is a group of scripts, which serves application timing and simple computations with results directly sent back to application.
(2) The second group of PHP scripts is used to serving connection between Flash application and MySQL database system.
(3) The third group of scripts is used for executing of server system commands.

We use MySQL system for storing data like device states, running experiment states, MATLAB states, connection states, user accounts, and other. The uDAQ28/LT device on server-side is directly controlled by MATLAB/Simulink. We have created a Simulink block function, which sends the measured data to database in every sample time period of running experiment. These data are collected by Flash application and sent to user. The connection between MATLAB and MySQL is performed by Java database connection driver (JDBC).

The communication between every part is shown in Fig. 2. Technology of experiment execution and data collecting is as follows. MATLAB is executed by PHP script that sends execution command directly to operating system command line. This command is built in client-side application and consist of two parts. The first is system command for MATLAB execution with program setup parameters, and the second is command for MATLAB command line. This second part contains all instructions for experiment execution, as names of Simulink schemes and m-files to run, input parameters, and other instructions for MATLAB. When MATLAB/Simulink program is running, the specific m-file script writes to MySQL information about its state (1 – MATLAB is running). Another state is collected from Simulink. These states are periodically checked by client application, and if both are set to 1, it means that experiment is properly running, and client-side application switches to screen for data observation. Data collecting is based on asynchronous technology. Every new measured state from device is sent by Simulink to MySQL and periodically collected by client application. Sampling period for Simulink experiment is set up by user and for data collection by Flash application is set to 0.2 seconds. MATLAB uses JDBC for database connection and Flash application connects to database through PHP layer. After experiment is finished, MATLAB is automatically turned off by its script. It can be turned off also manually by the user from client application.

## 4. APPLICATION OVERVIEW

The remote laboratory can be used by anyone who is registered as user of Moodle. After sign in to remote laboratory course, user fills a simple registration form to create a new account. After registration, user receives a confirmation e-mail with account activation link and can sign in to remote control application.

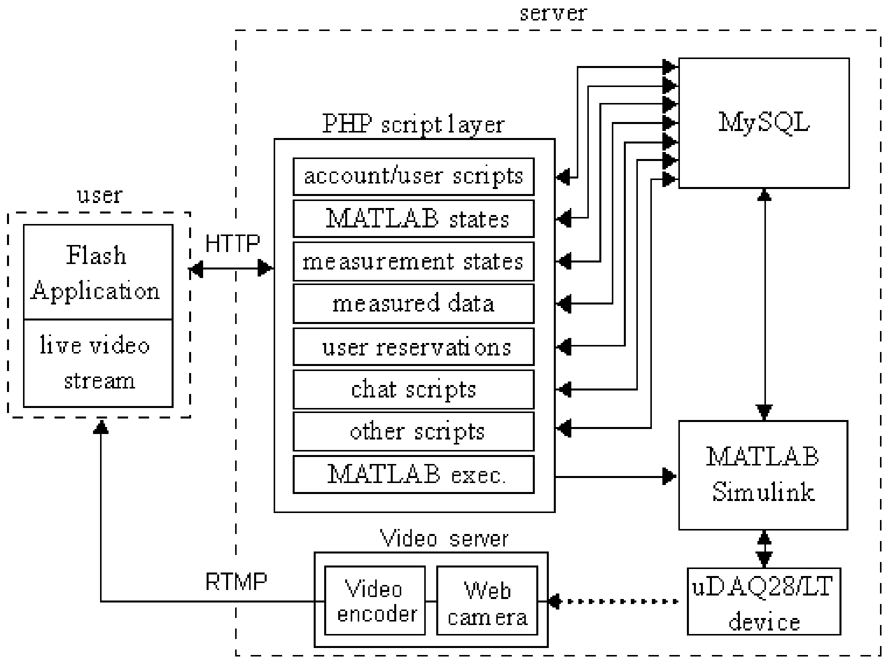


Fig. 2. Scheme showing connections between client and server side of software solution, and controlled device

The Flash application is accessible through any web browser, because it is embedded into regular web page. The only requirement to run the application is Adobe Flash Player (downloadable software plug-in). The application web site is located on e-learning system LMS Moodle, and it is accessible through the main web site of our department.

Application GUI consists of several screens. The first is login interface (Fig. 3), where user has to put his login name and password to sign up for experiment.

In the actual version of our solution, it is necessary to use accout login option directly in client application, because the actual version of application is aimed to be independent on web location. Due to fact that our solution is a part of remote laboratory course, located on Moodle, in future we plan to create extension that will provide access to application user accout directly through Moodle, by using its session cookies.

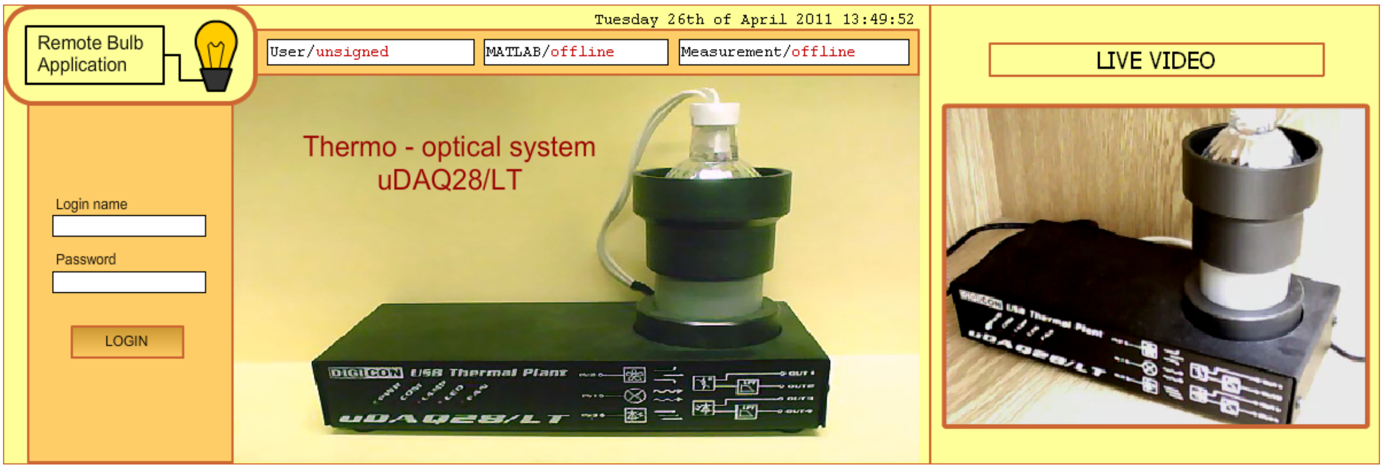
On the second screen (Fig. 4) user can reserve the time session for experiment and communicate with other signed up users by chat. Application provides option for reservations management, like creation, removal, and sort.

Before proceeding to the experiment itself a new time reservation has to be created. Flash program scripts run in background of application to compare new reservation with those which are saved in database to avoid time collision between experiments. One user can register maximum three reservations with maximal duration of 30 minutes (sufficient time for performing the experiment on uDAQ28/LT).

If session reservation is finished, user can switch to another screen, where he can set up the experiment. On setup screen (Fig. 5) user chooses Simulink scheme which will be executed in remote MATLAB. All Simulink schemes are located at server and registered in MySQL database system. Content of each scheme is automatically detected by parsing software. This software part is necessary for gathering informations about Simulink model files (MDL-files) and their content. In application, when user selects scheme for measurement, the parsing PHP script reads the MDL file and detect all important objects in it. The script looks for Simulink blocks and their parameters, and sends information about them back to Flash application, where the input form is dynamically generated. This way, user can view and edit all important parameters that model requires. The script detects most commonly used blocks and their parameters, like constant, step, transfer function, and PID controller blocks. User can change their values before measurement starts and also select parameters that may be changed during experiment.

Application contains all default simulation schemes provided by device manufacturer, but there is also an option to upload custom scheme created by user. For this purpose we have created upload form that provides all necessary features important for keeping software security and operationality. Before a new schema is uploaded to server and saved to database, the security PHP script checks if file has proper extension, size, and if contains all necessary parts, like database connector block and block for data export. In last step PHP script reads file to look for dangerous content (for example system commands writen by user).

When all input parameters for experiment are correctly filled, user can execute the experiment. After connection between user-side application and MATLAB is established and MATLAB/Simulik is properly running, the application switches to experiment screen (Figs. 6 and 8). User can observe experiment results and remote device states and also can send commands for setting up new values of device inputs (Fig. 7).



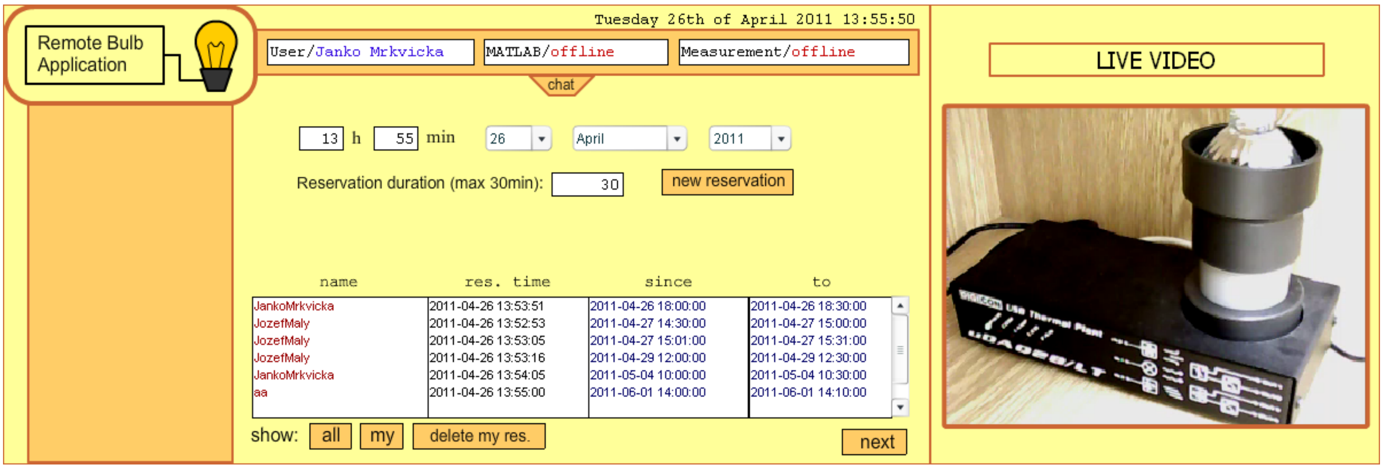
Tuesday 26th of April 2011 13:49:52

User/unsigned

MATLAB/offline

Measurement/offline

Fig. 3. First screen of Flash application, showing the picture of remote device and login form



Tuesday 26th of April 2011 13:55:50

User/Janko Mrkvicka

MATLAB/offline

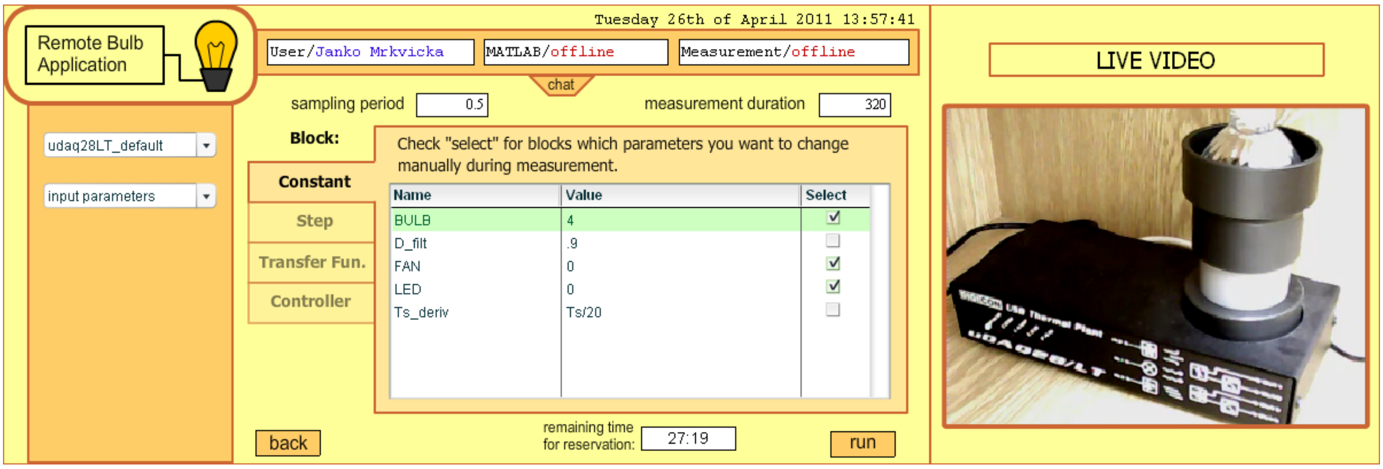
Measurement/offline

13 h 55 min 26 April 2011

Reservation duration (max 30min): 30

| name | res. time | since | to |
|---|---|---|---|
| JankoMrkvicka | 2011-04-26 13:53:51 | 2011-04-26 18:00:00 | 2011-04-26 18:30:00 |
| JozefMaly | 2011-04-26 13:52:53 | 2011-04-27 14:30:00 | 2011-04-27 15:00:00 |
| JozefMaly | 2011-04-26 13:53:05 | 2011-04-27 15:01:00 | 2011-04-27 15:31:00 |
| JozefMaly | 2011-04-26 13:53:16 | 2011-04-29 12:00:00 | 2011-04-29 12:30:00 |
| JankoMrkvicka | 2011-04-26 13:54:05 | 2011-05-04 10:00:00 | 2011-05-04 10:30:00 |
| aa | 2011-04-26 13:55:00 | 2011-06-01 14:00:00 | 2011-06-01 14:10:00 |

Fig. 4. Second screen of Flash application, showing reservation system for experiment



Tuesday 26th of April 2011 13:57:41

User/Janko Mrkvicka

MATLAB/offline

Measurement/offline

udaq28LT_default

input parameters

sampling period 0.5

measurement duration 320

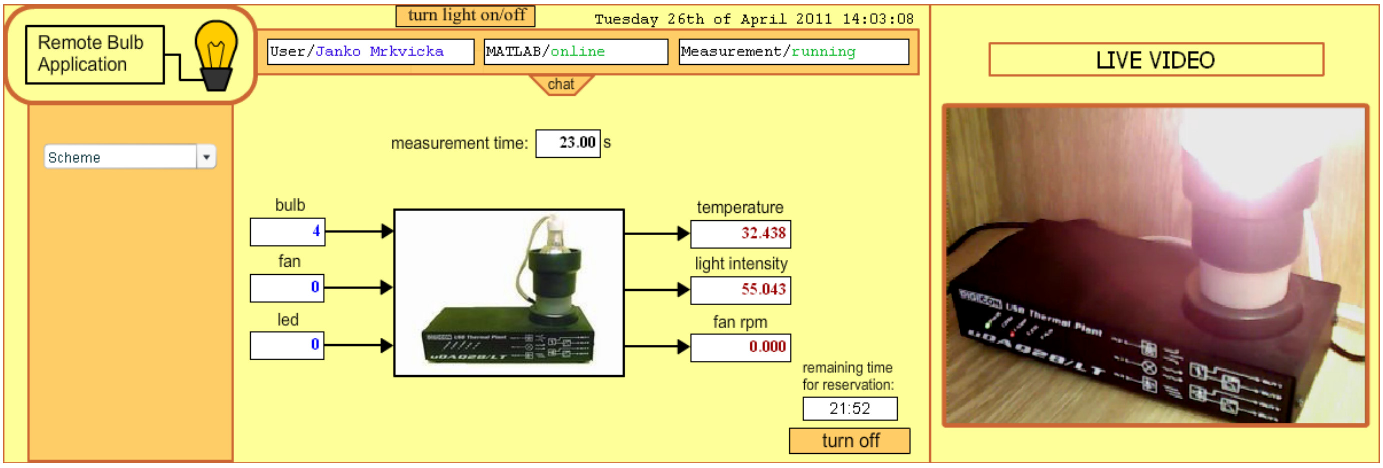
Block:

Check "select" for blocks which parameters you want to change manually during measurement.

| Name | Value | Select |
|---|---|---|
| BULB | 4 | ☑ |
| D_filt | .9 | □ |
| FAN | 0 | ☑ |
| LED | 0 | ☑ |
| Ts_deriv | Ts/20 | □ |

remaining time for reservation: 27:19

Fig. 5. Input form for experiment parameters associated to chosen scheme



Tuesday 26th of April 2011 14:03:08

User/Janko Mrkvicka

MATLAB/online

Measurement/running

measurement time: 23.00 s

bulb: 4

fan: 0

led: 0

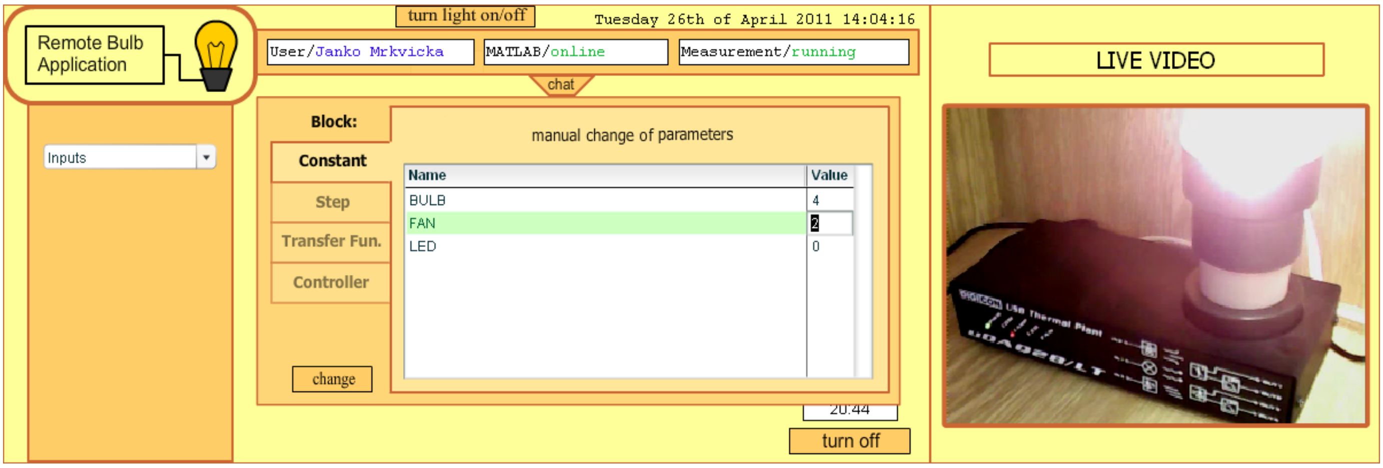
temperature: 32.438

light intensity: 55.043

fan rpm: 0.000

remaining time for reservation: 21:52

Fig. 6. Running experiment (input/output view)



Tuesday 26th of April 2011 14:04:16

User/Janko Mrkvicka

MATLAB/online

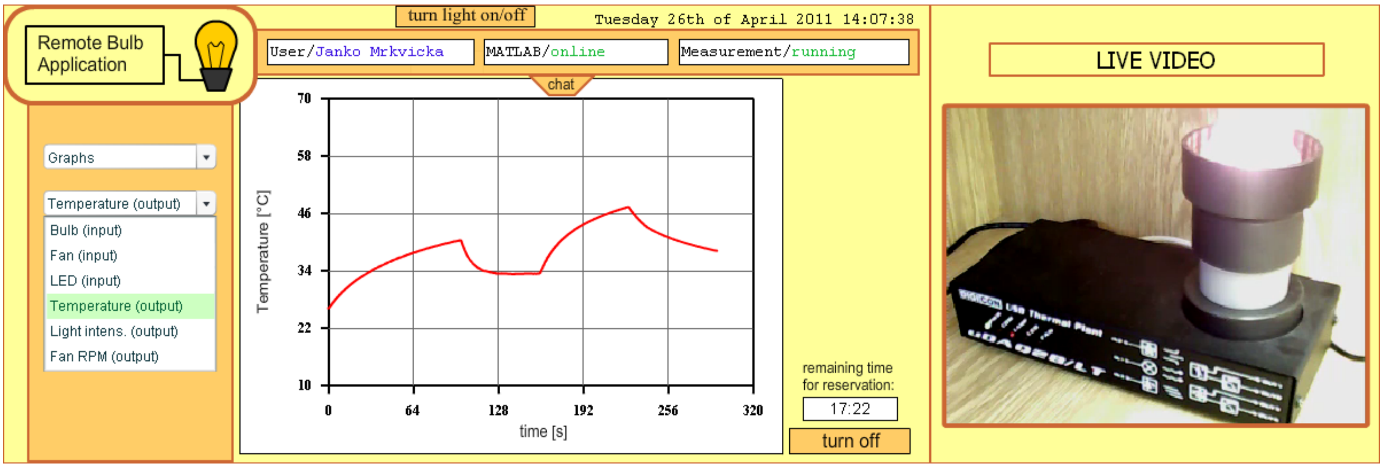
Measurement/running

Block:

manual change of parameters

| Name | Value |
|---|---|
| BULB | 4 |
| FAN | 2 |
| LED | 0 |

Fig. 7. Running experiment (input form where user can change parameters in real time)



Tuesday 26th of April 2011 14:07:38

User/Janko Mrkvicka

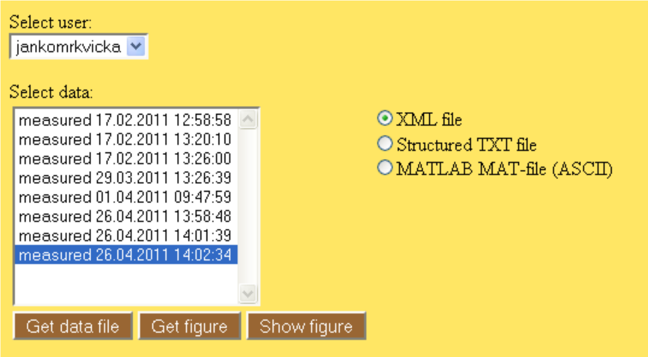
MATLAB/online

Measurement/running

remaining time for reservation: 17:22

Fig. 8. Running experiment (selected variable graph view)

When experiment is finished, user can download all measured data from database in chosen format. Application can provide data in the form of structured XML file, plain text file or MATLAB m-file. Every experiment result is stored in database with unique identificator, and can be accessed later. Web module for data export is shown in Fig. 9.



Select user:
jankomrkvicka

Select data:
measured 17.02.2011 12:58:58
measured 17.02.2011 13:20:10
measured 17.02.2011 13:26:00
measured 29.03.2011 13:26:39
measured 01.04.2011 09:47:59
measured 26.04.2011 13:58:48
measured 26.04.2011 14:01:39
measured 26.04.2011 14:02:34

● XML file
○ Structured TXT file
○ MATLAB MAT-file (ASCII)

Fig. 9. Data export form. Results are sorted by user name and time of measurement

## 5. CONCLUSION

Remote laboratories can be a suitable way to improve education in field of automation engineering and process control. Our proposed solution presents one of many different ways how can remote laboratory be realized. We have chosen technologies which can easily handle features that are required for this kind of solution. MySQL database system is suitable for fast storage and data management and it is also robust enough for programs with high database access traffic. Adobe Flash Application on user side provides easy-to-use GUI and can be run directly through web browser without any installation procedure.

## ACKNOWLEDGMENTS

The authors are pleased to acknowledge the financial support of the Cultural and Educational Grant Agency KEGA of the Slovak Republic under grant No. 3/7245/09 and of the Scientific Grant Agency of the Slovak Republic under the grant 1/0071/09.

The paper is supported by a grant (No. NIL-I-007-d) from Iceland, Liechtenstein and Norway through the EEA Financial Mechanism and the Norwegian Financial Mechanism. This project is also co-financed from the state budget of the Slovak Republic.

## REFERENCES

Bisták, P. and Beránek, M. (2006). Java client server applications for remote laboratory based on matlab and com. In *7th Int. Conference Virtual University VU06*, 135–139. Bratislava, Slovakia.

Bisták, P. and Čirka, Ľ. (2009). Reservation of remote laboratory using Moodle LMS. In M. Huba (ed.), *Proceedings of 10th International Conference Virtual University*. FEI STU in Bratislava, E-academia Slovaca, Bratislava.

Čirka, Ľ., Kalúz, M., Kvasnica, M., and Fikar, M. (2010). Virtual laboratory. In *Proceedings of the 9th International Scientific–Technical Conference Process Control 2010*, C029a–1–C029a–8. University of Pardubice, Kouty nad Desnou, Czech Republic.

Dougiamas, M. (2005). Moodle – a free, open source course management system for online learning. `http://moodle.org`.

Hercog, D., Gergič, B., Uran, S., and Jezernik, K. (2007). A DSP-based remote control laboratory. *IEEE Transactions on Industrial Electronics*, 54, 3057–3068.

Huba, M., Kurčík, P., and Kamenský, M. (2006). *Thermo-optical device uDAQ28/LT*. STU Bratislava, Illkovičova 3, Bratislava.

Kalúz, M., Čirka, Ľ., and Fikar, M. (2010). MATLAB builder JA in control engineering education at FCFT STU. In *Technical Computing Bratislava 2010*, volume 18, 1–5.

Puerto, R., Jiménez, L.M., and Reinoso., O. (2010). Remote control laboratory via internet using matlab and simulink. In *Computer Applications in Engineering Education*, volume 18, 694–702.

Žilka, V., Bisták, P., and Kurčík, P. (2008). Hydraulic plant remote laboratory. *International Journal of Online Engineering*, 4, 69–73.