Development and Application of Remote Laboratory for Embedded Systems Design

http://dx.doi.org/10.3991/ijoe.v11i3.4519

Anzhelika Parkhomenko, Olga Gladkova, Eugene Ivanov, Aleksandr Sokolyanskii, Sergey Kurson
Zaporizhzhya National Technical Universit, Zaporizhzhya, Ukraine

***Abstract*—This paper presents Embedded Systems' Hardware-Software CoDesign and is an overview of the approach based on using ready platforms. Comparable analysis of well-known platforms from different vendors is given. Application of remote laboratories of embedded system rapid prototyping is offered. Remote lab architecture, a set of experiments and Use-Case scenarios are described. The proposed approach improves the current state-of-the-art work in the area of embedded systems design to accelerate stages of hardware-software integration and testing.**

***Index Terms*—Embedded systems' design; Hardware and software platform; Remote laboratory; Web-interface; Experiment set; Program code; Control system of the mobile objects; Support design decision.**

I. INTRODUCTION

Today Remote Laboratories are being successfully developed and implemented worldwide. According to observations, the vast majority of already implemented experiments and installations with remote access are used as pure educational resources and the possibilities for professional researchers or designers are not being fully explored. On the other hand, the Embedded Systems (ESs) market is permanently evolving and requires the creation of more complex systems in shorter periods of time. Under these conditions, the development of systems "from scratch" is not effective. Therefore, one of the key areas to improve the efficiency of Embedded Systems design is the accumulation of technical solutions for reuse. This reuse of knowledge can be illustrated and implemented in remote laboratories. The disunity of descriptions of already developed components (hardware units, programs, algorithms, implementations, etc.) hinders their reuse, but the application of remote experiments allows the designer to obtain information about ready hardware/software platforms and components for making decisions on Embedded Systems realization.

II. INVESTIGATION OF THE EMBEDDED SYSTEM DESIGN FEATURES

At present experts represent and analyze various high-level design methodologies for the embedded systems. Design flow of embedded systems has been presented in various publications, e.g. in [1-4]. The main directions are shown in [1]:

- Object-oriented design.
- Hardware/software CoDesign.
- Platform-based design.
- Actor-oriented design.
- Mixed-language design.
- Aspect design.

Hardware-Software CoDesign is considered one of the possible approaches for design of ESs. CoDesign techniques have both advantages and disadvantages. On the one hand, the ES design is complicated, but on the other hand, there is significant improvement of the final product characteristics in comparison with alternative versions of design solutions. Generally, the embedded systems design flow looks like Fig. 1.

The established practices of embedded systems design show that hardware decisions are initially selected. Further, based on those decisions, the software superstructure will be made. But this has a great risk of delaying the whole product design chain [2]. After the hardware and software components have been implemented, they must be separately verified and integrated, and whole system must be tested again.

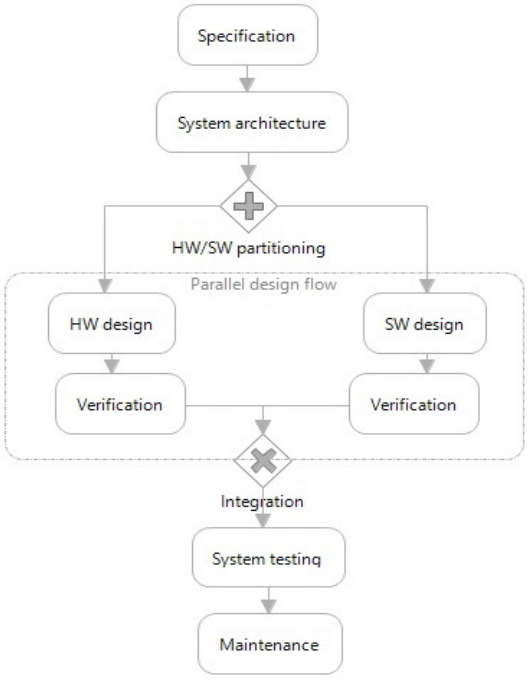

Figure 1. Classical design flow of embedded systems

Since the problem of reducing the design time remains relevant, this approach of embedded system design has not been successful and requires improvement. Today, in the field of ES design, the use of ready hardware and software platforms is very popular to accelerate development of products and to reduce the time to market.

Research has shown there are many different manufacturers of hardware and software platforms: Arduino, Texas Instruments, Parallax Inc., Netmedia, Microchip, Digilent, MBED, Raspberry PI, Cyclone. Each platform has a form factor and functionality, so the designer's choice depends on the task. Thus, the designer of an ES has to know the potential of the ready platforms existing in the market and be able to make responsible decisions concerning application of a certain hardware-software platform. Unfortunately, the information offered on the website does not always allow the right and reasonable choice to optimally implement the project. Acquisition of a huge number of various platforms for analysis of their opportunities is unacceptable for small and medium-sized enterprises despite the relatively low prices [5].

Due to growth in the popularity of ready hardware-software platforms, a number of the companies (e.g., Autodesk) offer applications for virtual simulation of their work. For example, it is possible to allocate simulators for Arduino: 123D Circuits, Virtualbreadbord, Simulino, Virtronics, Simuino, Arduino Simulator and others. However, the simulation does not replace real work with the hardware and software. In fact, instead of a real physical process, the simulator allows study of only its mathematical model. Furthermore, many software simulators are not free, require time for studying, and have limited functionality and an incomplete element base.

There are also powerful packages of circuitry design, such as ISIS PROTEUS, Altium Designer and others. However, the use of their full functionality is not always required, and the cost is quite high.

Therefore, creation of a specialized laboratory for research of ready hardware-software platforms based on remote experiment is an actual task.

Application of remote laboratories will make changes to the embedded systems design flow (Fig. 2).

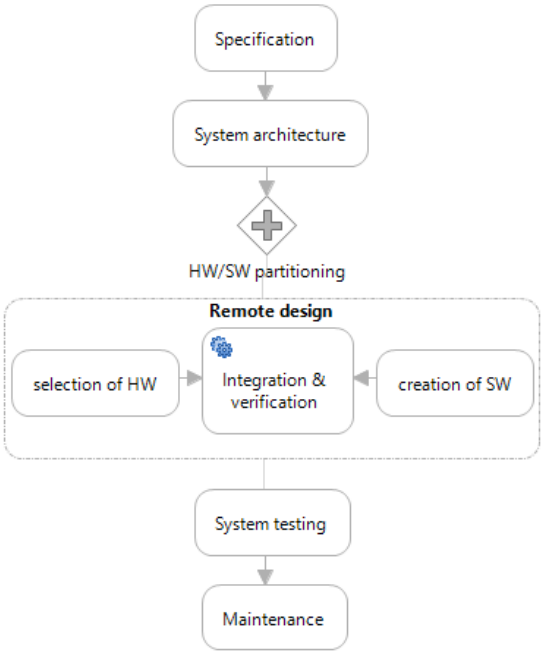

Figure 2. Modified design flow

Introduction in the design process of a remote lab will allow users to

- Extract and analyze information about specifications of ready hardware and software components for their reuse.
- Perform selection of hardware platforms.
- Develop program code.
- Test compatibility and working capacity of hardware and software parts of the projectable system.
- Check if the devices functions to the specification requirements.
- Observe the experiment performance.

Remote testing of components allows quick assessment of the feasibility of their application and decisions on their integration into the project, thereby reducing the risks and the time/expenses for the design.

III. DEVELOPMENT AND APPLICATION OF REMOTE LABORATORY FOR THE SUPPORT OF ADOPTION OF DESIGN DECISIONS

The specialized remote laboratory RELDES (REmote Laboratory for Embedded Systems Design) is focused on providing developers with a fast and effective choice of standard design decisions on the basis of ready hardware-software platforms.

Fig. 3 gives an overview of the RELDES infrastructure. The laboratory provides remote controlled access to Arduino-based experimental equipment. Arduino is a powerful microcontroller with a large volume of memory that allows creation of complex electronic devices. Additionally, the hardware-computing Arduino platform is one of the most popular and simple.

There are already many articles comparing various development boards [6-8], so for comparison the following well-known boards were chosen: Arduino Uno, LaunchPad, PCDuino, Raspberry PI, BeagleBone Black. The comparative Table 1 is given below.

TABLE I.
WELL-KNOWN PLATFORMS' COMPARATIVE TABLE

	Microcontroller boards		**Computer boards**		
	Arduino Uno	***Launchpad MSP430***	***PCDuino 3***	***Raspberry PI***	***Beagle Bone Black***
Price	20$	10$	59$	35$	45$
MC/CPU	16MHz, ATmega 328	16MHz, MSP430 G2553	1GHz ARM Cortex A8	700 MHz ARM 1176JZFS	1GHz TI Sitara AM3359 ARM Cortex A8
RAM	2KB	512B	1GB	512MB, SDRAM	512MB, DDR3
Pins	14, 6	20, 3	32	2x13, 8	2x46
Overall size (mm)	68.6x53.4	66x51	121x65	86x54	86.4x53.3

It is clear that Arduino and LaunchPad are in a different category than PCDuino, Raspberry PI, Beagle Bone Black. Arduino and LaunchPad are microcontrollers. A

microcontroller is just one tiny part of a computer. The Arduino can be programmed in C, but it cannot run an operating system.

On the other hand, the Rasperry Pi and PCDuino are computers. These devices need an operational system to work. Raspberry Pi and PCDuino can run the operating system alone [6].

Arduino and LaunchPad are perfect for electronics projects and prototyping. They allow rapid, cheap, prototyping of embedded systems. The major differences between Arduino and LaunchPad are the cost and the memory available. Arduino's advantages lie in community code libraries and vast amount of extension boards (so called shields) which provides users with both hardware and software tools to achieve their goals. On the other hand, LaunchPad has very few available extension boards.

So, in comparison to other similar platforms, Arduino has a number of advantages:

- The project was developed with open code of software, that works as the network project/community, allowing participants to exchange experience and ready applied practices, further accelerating the process of development and debugging.
- The microcontroller and expansions have a low cost.
- The programming environment (OS Windows, Macintosh OSX and Linux: 32/64bit) has simplicity and a cross-platform.

The stand with the experiments is connected to the laboratory server via the serial interface. The computer with Linux Debian OS acts as the server for the remote laboratory. The server provides access to experiment programming and the video stream of the laboratory. The video stream is implemented through the use of "ffmpeg." The server processes web client requests and carries out the following actions depending on the obtained data:

- Compilation of received initial code (Arduino, with use of the console utility Ino).
- Returns the compilation results (using HTTP protocol) and sends the request for compilation..
- Uploading binaries to the microcontroller (provided the controller is free).
- Client queuing (if the experiment is occupied).

The RELDES process diagram is shown in Fig. 4. To get access to the experiments, users need to be registered. Access to the experiments page is closed to unregistered users. Users can upload their program code to a microcontroller on the Arduino board via the Web-client. With the Web-interface, the user is able to create code or upload files with the initial code. The Lab server compiles the received initial code. Being compiled, the generated code is uploaded to the microcontroller on the Arduino board. If the program has errors, the server generates and returns the compilation results. Therefore, users have a possibility to create and edit their program code directly in the client's Web-interface without any need for the development tools on the user's PC.

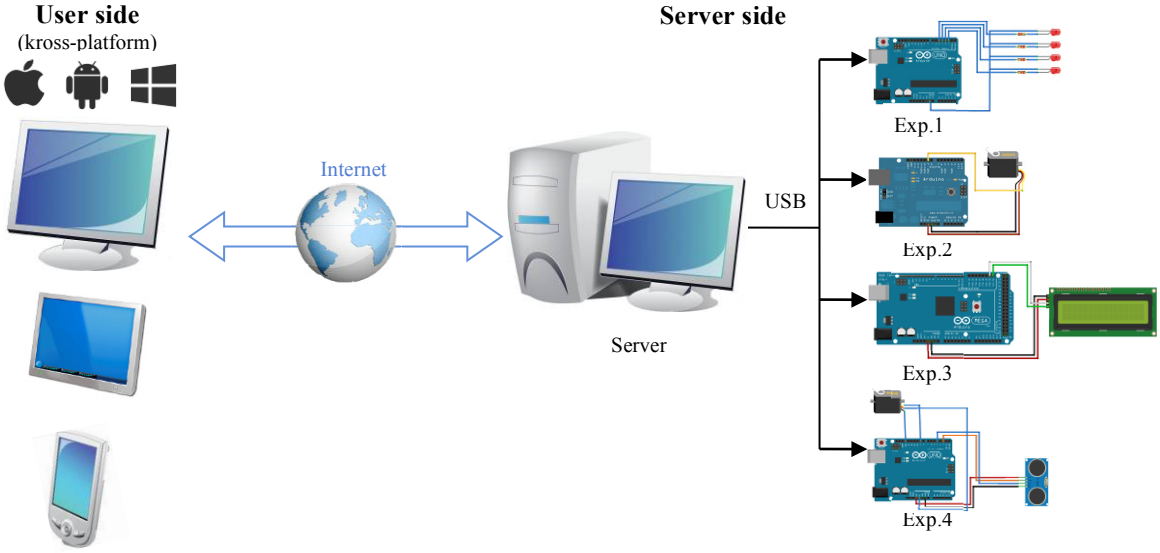

Figure 3. Overview of the RELDES infrastructure

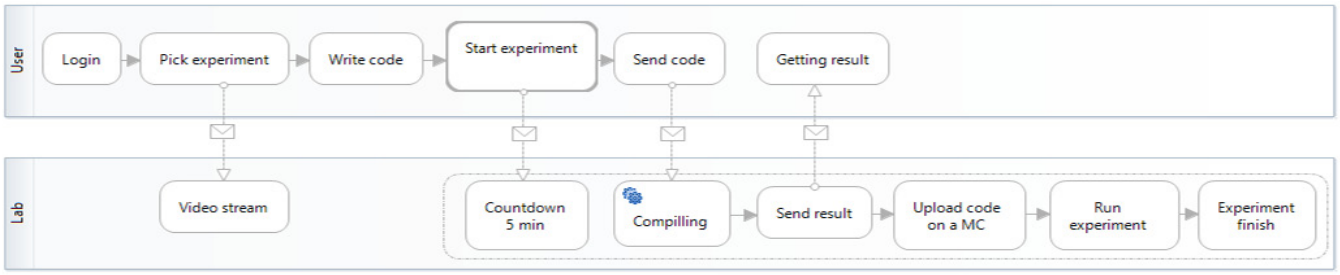

Figure 4. Laboratory process diagram

Client-server communication is carried out via a web-interface implemented with HTML+CSS+JS. The service runs on the Apache server with use of the relational MySQL database. Fig. 5 illustrates an impression of the experiment page Web-interface.

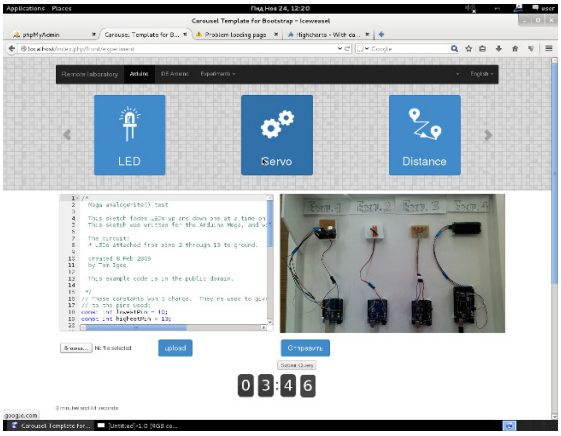

Figure 5. RELDES web-interface

The user friendly slider for fast switching between experiments is located at the top of the page. The remaining area is divided for displaying video and code editing. At the start of the work with a remote experiment, a user can observe the time remaining to the end of the experiment. Besides the web browser, no additional software is required to access the remote laboratory

IV. THE LIST OF REMOTE EXPERIMENTS

The set of experiments depends on the specifics of the designed ES. For example, as standard decisions for design of the mobile objects control system [9,10], experiments based on the Servos, the liquid crystal display, the LEDs, the distance sensor, etc. are offered. The proposed set of experiments can be expanded for other hardware-software platforms and tasks connected to the realization of various classes of ES.

Experiment 1. Traffic Lights. This experiment allows users to control a set of LEDs with an Arduino board. LEDs are connected to the digital pins of the board. The resistors are used to restrict the current flow through the diodes. Diodes have common ground. (Fig.6). Scenarios of experiment: Sequential lights, Running lights, Brightness change.

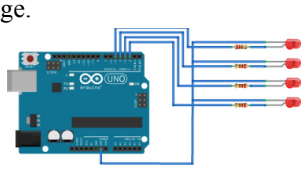

Figure 6. Model of the experiment Traffic Lights

The scenario of the experiment allows setting and controlling parameters of the traffic light such as the pin number and pin mode. Function pinMode is used for that. Also, to control diodes' brightness analogWrite call is used. Its arguments are pin number and voltage value. Voltage value is not used directly as the parameter of analogWrite, values ranging from 0 to 255, is used instead. To make it possible for the user to see a diodes' brightness change, function delay is called. It stops the board's work for amount of time (in ms), stated as argument of the function.

Experiment 2. Servomotor. In this experiment, the user is allowed to control Servo through a digital pin. Servo has 3 pins: vcc, ground and control pin (Fig.7). Scenarios of experiment: Two position turns, Smooth turns.

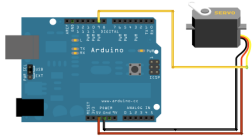

Figure 7. Model of the experiment servomotor

The program uses the Servo.h library, which contains Servo class implementation. It is used to communicate with the motor. To set the position of the Servo method, write is used. An angle is passed to the method as a parameter. The function delay halts the board's work for a desirable time in milliseconds. It is necessary to give the Servo time to perform before the next command is given. Not all Servos have a turning range of 0-180 degrees. It varies depending on the model.

Experiment 3. LCD display. In this experiment, the user is allowed to control the liquid crystal display. The display has an integrated controller and is not accessed by Arduino directly. Communication is done via the I2C serial protocol (Fig.8). Scenarios of experiment: Static text, Scrolling text.

First, the LiquidCrystal_I2C.h and Wire.h libraries are included. Wire.h implements the I2C protocol, and LiquidCrystal_I2C.h implements communication with the LCD via the I2C. A LiquidCrystal_I2C class object is created at the beginning of the code with parameters specific to a given LCD controller. The LCD parameters are set to begin the call. Its parameters are the number of rows and columns. The backlight()/noBacklight() function is called to enable/disable the LCD's back light.

The cursor position is set by calling the setCursor (column, row) function. The clear() function is called to remove all the symbols from the LCD. The delay(milliseconds) function is called to delay the activity of the board for a stated time in milliseconds. Function print(string) is called to print the symbols starting from the cursor position.

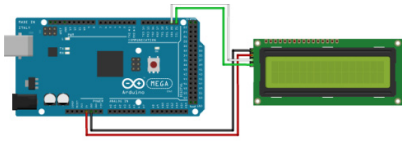

Figure 8. Model of the experiment LCD display

In this case it is done by setting the parameters of the LCD screen with a call to begin the c parameters, the number of columns and rows of characters. Next, f-tion print is called, which displays a specified set of characters. The second function of the loop is executed in an infinite loop. In this case, it is performed by setting the cursor to the beginning of the second line (numbered from zero produced) by calling the setCursor and printing time of the board in seconds.

Experiment 4. Sonic distance sensor. In this experiment the user works with a sonic distance sensor. The sensor has 4 pins: vcc, ground, trigger and signal. (Fig.9). Scenarios of experiment: Distance measurement, Distance sensor with Servo.

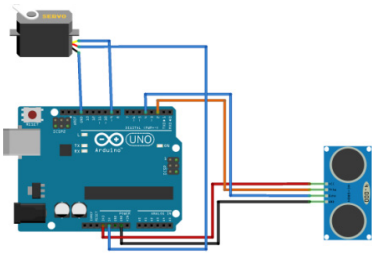

Figure 9. Model of the experiment Ultrasonic sensor distance

The user receives data from the sensor through the serial interface, using the serial class. In this function the serial interface is initialized with a baud rate of 9600, and sensor pins modes are set with pinMode call. Variables with the sensor pins are declared before the setup function. The distance sensor has two modes: measuring and sending data. To set it in measuring mode pulse with 2 microseconds, length must be sent to the sensor (trig pin) by calling the digitalWrite function. Its parameters are pin number and pin value. To form the pulse the function is called three times: firstly with LOW value to make sure the pin is not set to HIGH, then with HIGH value to create the pulse itself and lastly with LOW value again to end the pulse. Those calls are separated by time delays. Delays between calls are made with the delayMicroseconds function. PulsIn() is called to read the date from the sensor. This function's arguments are pin number and expected value (HIGH or LOW). Returned value will represent travel time of a sonic signal measured by the sensor. MicrosecondsToCentimeters and microsecondsToInches are used to convert that value to distance functions. Results are returned to the user by print method of serial interface.

In the second scenario, the sensor is attached to the Servo. It allows user to turn the sensor manually and measure the distance to different objects. The Servo position is set by the write method of Servo object. Angle values are stored in the array. After the Servo position is set, command to measure distance is called. To do so, logical 1 pulse with duration of 10 microseconds is sent to the sensor with digitalWrite call. To read the measurements, pulseIn is called. Results are sent to the user via the serial interface by calling the print method.

V. CONCLUSIONS

Development of remote laboratories for complex hardware-software design of the embedded systems is an actual task because the problems in this area demand new techniques, technologies and tools of ES design.

Benefits of our laboratory:

- Cross-platform (use any device and operation system).
- Accessibility (use our hardware instead of spending money).
- Configurability (write your own code or use code templates).

Practical application of the developed laboratory will allow the designer to make the right decisions on the expediency of the application of a ready platform and possible options of the project development.

Future work will be focused on expanding the range of provided hardware-software platforms and set of experiments for solving tasks from different areas of embedded systems design.

REFERENCES

[1] Platunov, A.E., Postnikov, N.P., "High-level design of embedded systems", p.121, SPb.: NIU ITMO, 2011. http://dx.doi.org/10.1109/jproc.2011.2182009

[2] Teich, J., "Hardware/Software Codesign: The Past, the Present, and Predicting the Future", Proceedings of the IEEE, Vol.100, pp. 1411-1429, Germany, May 13, 2012.

[3] Abdurohman, M., Kuspriyanto, Sutikno, S., Sasongko, A., "The New Embedded System Design Methodology For Improving Design Process Performance", International Journal of Computer Science and Information Security, Vol. 8, No. 1, pp. 35-43, 2010.

[4] Wolf H., Wayne, "Hardware-Software Co-Design of Embedded Systems", Proceedings of the IEEE, Vol.82, NO. 7, pp. 967-989, Germany, July 1994.

[5] Parkhomenko, A., Gladkova, O., "Interactive remote laboratory for research of hardware-software platforms", Proceedings of the International Scientific-Practical Conference «Internet-Education-Science-2014», pp. 111-113, Vinnytsia: VNTU, October 14 - 17, 2014.

[6] Santos, R., Arduino vs Raspberry PI vs Beagle Bone Black – 2013. [Electronic source] http://randomnerdtutorials.com/arduino-vs-raspberry-pi-vs-beaglebone-vs-pcduino/

[7] Allan, A., Which Board is Right for Me? – 2014. [Electronic source] http://makezine.com/magazine/make-36-boards/which-board-is-right-for-me/

[8] Leonard, M. How to Choose the Right Platform: Raspberry Pi or BeagleBone Black? – 2014. . [Electronic source] - http://michaelhleonard.com/raspberry-pi-or-beaglebone-black/

[9] Parkhomenko, A.V., Gladkova, O.N., "Virtual Tools and Collaborative Working Environment in Embedded System Design", Proceedings of XI International Conference on Remote Engineering and Virtual Instrumentation (REV2014), pp. 91-93, Porto: Polytechnic, Portugal, February 26-28, 2014. http://dx.doi.org/10.1109/rev.2014.6784230

[10] Tsmots, I., Teslyuk, V., Vavruk, I., "Hardware and software tools for motion control of mobile robotic system", Proceedings of XII International Conference «The Experience of Designing and application of CAD Systems in Microelectronics (CADSM 2013)», p. 368, Lviv Polytechnic, Lviv, 2013.

AUTHORS

Anzhelika Parkhomenko is with the Zaporizhzhya National Technical University, Software Tools Department, 69063 Zaporizhzhya, Ukraine, Zhukovskogo 64, (e-mail: parhom@zntu.edu.ua).

Olga Gladkova is with the Zaporizhzhya National Technical University, Software Tools Department, 69063 Zaporizhzhya, Ukraine, Zhukovskogo 64, (e-mail: gladkovaolga9@mail.ru).

Eugene Ivanov is with the Zaporizhzhya National Technical University, Software Tools Department, 69063 Zaporizhzhya, Ukraine, Zhukovskogo 64, (e-mail: skycat4best@gmail.com).

Aleksandr Sokolyanskii is with the Zaporizhzhya National Technical University, Software Tools Department, 69063 Zaporizhzhya, Ukraine, Zhukovskogo 64, (e-mail: sanya_sokol1992@mail.ru).

Sergey Kurson is with the Zaporizhzhya National Technical University, Software Tools Department, 69063 Zaporizhzhya, Ukraine, Zhukovskogo 64, (e-mail: kurson-serg@yandex.ru)

This article is an extended and modified version of a paper presented at the International Conference on Remote Engineering & Virtual Instrumentation (REV2015), held in Bangkok, Thailand, 25 - 28 February 2015. Submitted 13 March 2015. Published as resubmitted by the authors 04 May 2015.