Combined Teaching of Intelligent Building Design and Component Programming

Michał Śmiałek, Marcin Seligowski

Abstract — **The paper presents an approach towards teaching of intelligent building design combined with component programming of software modules for intelligent building control. This approach is supported by a laboratory simulator and a component-based environment. The simulator is composed of several EIB modules and a house control model. The component environment consists of low-level components for intelligent bus communication. The teaching approach uses the simulator to teach programming of intelligent building modules and construction of external software control components.**

I. Introduction

INTELLIGENT buildings are most probably the future of building construction. The "intelligence" of such building lies in their electrical installations. Such installations allow for controlling of all the electrical appliances in a very flexible way. An example of such an intelligent system is the EIB (European Installation Bus) [2,3,4]. The EIB systems consist of distributed modules that are connected through a bus cable (Fig. 1). The sensor modules (bottom ones on Fig. 1) need no supply of high voltage. Only the actuator modules that actually switch the appliances need power supply.

The EIB modules are very flexible, but this flexibility means also that the configuration of such a module is complex. Every module is a microprocessor-based device that can have an appropriate program loaded and requires a quite complicated process of parameterization. The modules are connected to the EIB bus line through an intelligent transmission module (Fig. 2). This module contains a choke that separates the direct current bus voltage from the data transmission accelerating current voltage. It also contains data telegram handling devices.

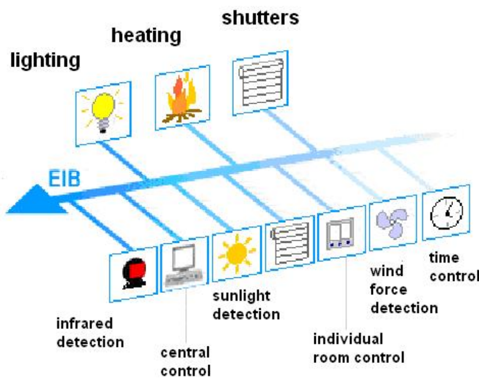

Fig. 1. Connections in the EIB system.

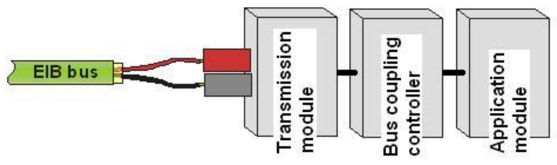

Fig. 2. Structure of the EIB modules.

The bus coupling controller contains a microprocessor with memory (ROM, RAM and EEPROM) that manages the transmission and control functions of the module. Here, the actual program with its parameters is stored and executed. Finally, the application module contains appropriate buttons, displays, switches, or physical detectors. This is the actual "user interface" of the module.

Though the devices are completely autonomous, they need an external PC computer to upload their programs and parameters. The PC has to be connected to the bus when uploading, but after programming, the computer is no longer necessary. The programming can be done with a special PC application – ETS. This application can handle a database of devices with their bus addresses. It manages all the parameters of the devices and enables their transmission.

Additionally to the ETS application, other PC applications can be installed in order to serve as system controllers. In this case the PC has to be connected to the bus permanently ("central control" on Fig. 1). Such applications should conform to the EIB standards of communication. These standards can be easily conformed when an appropriate library of software components is used [8]. The EIB standard supplies us with such a library (Falcon) that enables creating of applications based on the COM [6] or .NET [7] component programming standard.

II. Lab simulator of an intelligent building

In order to teach the design and programming of intelligent buildings in the EIB standard, an appropriate building simulator is proposed. The simulator contains several sensor and actuator EIB modules with appropriate devices connected. The connected devices simulate a real house. This approach enables the students to program the devices and see the results immediately on a model of a house.

The lab simulator serves as an aid in teaching and thus is designed to show the main aspects of the EIB technology. Its front panel (see Fig. 3) contains all the devices arranged according to their function in the system. It also contains the description of all the connections and a miniature model of a house.

Authors are with the Institute of Theory of Electrical Engineering, Measurement and Information Systems, Warsaw University of Technology, ul. Koszykowa 75, 00-662 Warszawa, Poland, e-mail: *smialek@iem.pw.edu.pl*

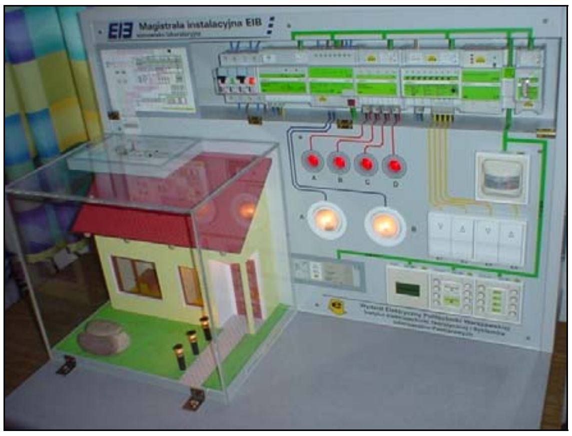
Fig. 3. Overall view of the EIB system lab simulator.

The top row of elements (Fig. 3) is formed of actuator modules and a power supply with circuit breakers. Only these devices need high voltage power supply. The actuators are the dimming module, binary output module and voice module. With these modules the student can control two lighting circuits with dimming capabilities, four lighting circuits with on-off capability and eight voice channels (voice messages). Such capabilities have proven to be satisfactory for teaching and for testing of PC applications.

Below the actuators (on the right) there are three rows of sensor modules. These contain a movement detector, regular switches connected to the binary inputs of the voice module, two push-button modules and an LCD display module. Again, this set of sensors allows the students or testers to design a variety of intelligent house applications. The sensors are supplemented with a serial port (RS-232) that enables communication with an external PC.

To enhance the educational impact of the simulator, it has been supplied with an extensive system of descriptions and an additional model of a house. The descriptions comprise of diagrams showing the electrical circuit layout (Fig. 4) and connector lines showing the actual connections. The model of a house contains lights and window shutters that are connected parallel to the lights on the simulator's front panel. Especially, the window shutters enable interesting experiments with lighting control. The house is supplied with an additional light sensor also connected to the bus. This allows the students to design even more sophisticated applications. One of such applications is constant room lighting. Depending on the outside lighting conditions, and a user-determined lighting level, the application can control the shutters and dimmers to keep constant lighting level in a room.

To realize the above constant lighting application, another EIB module is necessary, which is the logic module placed in the upper row of devices. The logic module gives yet more teaching possibilities. Together with an appropriate PC application it allows for teaching of intelligent logic controller programming. Programming is done graphically as a set of connected logic devices (AND, OR, timer, multiplexer, etc.).

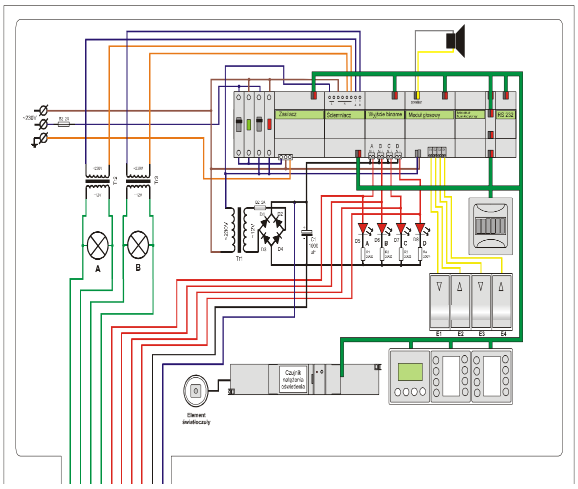

Fig. 4. General diagram showing the electrical connections of the simulator.

III. PC BUS CONTROL ENVIRONMENT

The intelligent house lab simulator can be combined with an environment for teaching of component programming [5]. Such an environment supplies the students with a set of low level software components that enable easy bus communication. It relieves the students from the burden of inventing all the lower layers of the component-based application. The main assignment to the students is to write a set of high-level components that would form a software application that would control certain features of the EIB simulator.

The component environment consists basically of three main components (Fig. 5). The main logical component is "EIB Switch", the component responsible for EIB communication is "Falcon Connect" and the supportive user interface component is "Switch UI".

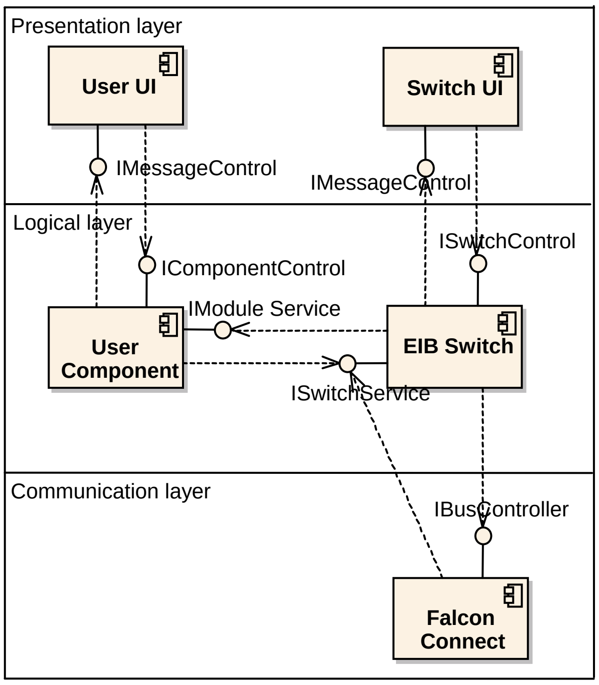

Fig. 5. General view of the component framework architecture.

All the components are designed and written according to the .NET component technology standard. The three framework components supply other components with an appropriate set of external operations grouped into "interfaces". The Falcon Connect component uses appropriate components from the Falcon EIB communication package of components. These components were written in the COM technology, and appropriate techniques were used to make them usable from the .NET environment.

The framework was designed to support education of proper design of software architectures based on component technologies. The components are distributed among three layers. The top layer is the Presentation layer that contains user interface components. The next layer is the Logical layer which contains components supporting the application logic. Finally, the Communication layer contains components responsible for bus communication and also responsible for database communication (not shown).

A student wishing to write an application based on the above framework has to implement an interface that inherits operations from the "`IModuleService`" interface. One of the main operations of this interface is "`putMessage()`", which should handle messages incoming from the bus. This message along with other offered by the interface should be implemented by the student in a component placed in the Logical Layer (User Component on Fig. 5). For the reverse direction communication the student's component should use the "`sendMessage()`" operation from the EIB Switch component. This message is handled by the component that sends it appropriately to the Falcon Connect component and finally – to the EIB bus.

The example usage of the above operations together with a complete flow of messages is shown on Fig. 6. This figures contains a UML [1] sequence diagram that orders messages temporally from top to bottom. The sequence starts when the user presses a button on the user interface. The User UI component makes a call ("`sendMessage`") to the IComponentControl interface. Then, appropriate component makes another call ("`sendTelegram`") to the IBusControl interface. When the message is sent successfully to the bus, the return message acknowledges it. Finally, an acknowledgement window is displayed on the user interface through the IMessageControl interface.

IV. TEACHING APPROACH

The EIB Simulator and the EIB Component Environment (see. Fig. 7) are two novel elements of the overall environment for teaching of intelligent building design and component programming. Together with the ETS Module Configuration Application and Logical Module Application they form a complete educational framework. It has to be noted that this framework can be used both for teaching students majoring in electrical engineering or majoring in computer engineering. The first group of students benefits from training in design of intelligent electrical installation. The second group benefits from training in component and logical circuit programming.

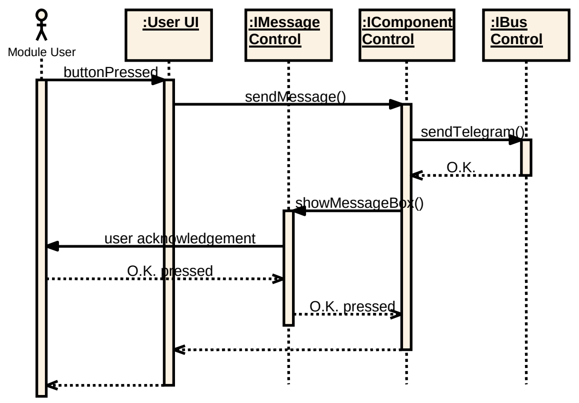

Fig. 6. Sequence diagram for sending a telegram to the EIB bus.

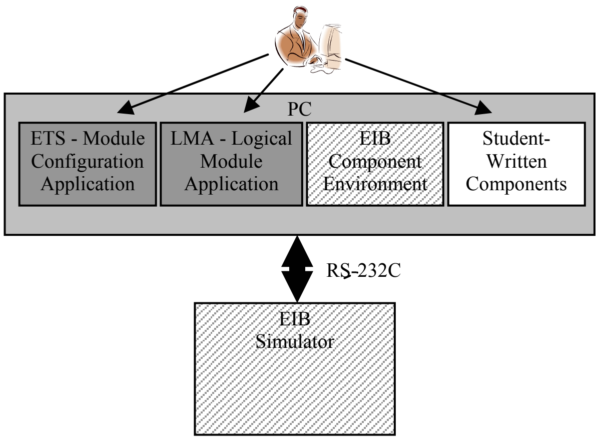
PC
ETS - Module Configuration Application
LMA - Logical Module Application
EIB Component Environment
Student-Written Components
RS-232C
EIB Simulator

Fig. 7. Procedure for using the proposed teaching framework.

The procedure for using the proposed educational framework is shown on Fig. 7. The student uses ETS and LMA to program and configure EIB modules of the EIB Simulator. ETS can be used for the entry-level assignments. With this tool only basic functionality of the simulator can be utilized, as only simple programs of the actuator and sensor modules can be configured here. Nevertheless, the tool gives the students a necessary introduction to the problems of bus transmission, addressing and proper structuring of the intelligent building designs.

With LMA, the students can also configure the Logical Module of the simulator. With the logical module, more sophisticated applications can be designed. This module is programmed with a visual language that depicts logical circuits. Assignments that require the use of the logical module can be given to higher-level electrical engineering students or to computer engineering students in a logic circuit teaching module.

For the most complex tasks of an intelligent building, the resources of the EIB Simulator are not sufficient. This is where PC control modules are necessary. The students install the EIB Component Environment on their local PCs or use readily installed components in the labs. Then they use .NET programming environment to develop their own components. The components can be tested without the EIB Simulator first, and the final test can be performed with the simulator connected.

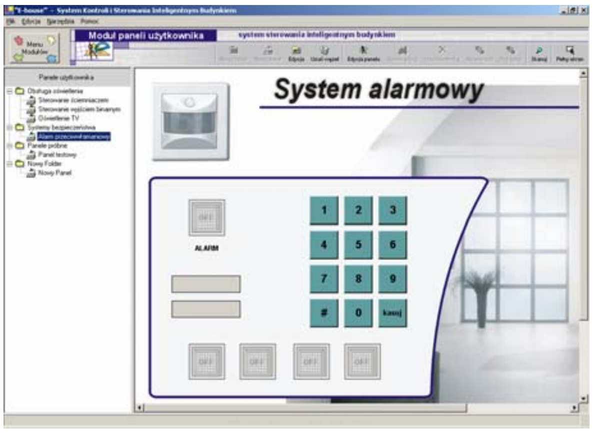

Fig. 8. User interface of a panel module written as a student assignment.

Assignments that involve writing of PC applications are very suitable for students that take component programming courses. This is where they can gain knowledge on general component programming and .NET programming in particular. The fact that the students can control a real house through the appropriate .NET interfaces is an additional motivation for learning. Further motivation is the layout and clarity of the EIB Component Environment. It serves students as a template for future design of component-based systems.

An example of such a student-written application is the alarm module shown on Fig. 8. The alarm module is a part of a larger panel module application [4]. The panel module allows for designing and commissioning of user-specified panels. The figure shows part of the user interface of the module, i.e. one of the designed panels. This example shows a sophisticated application with logic necessary to implement a system that reacts on the telegrams sent by the movement detector of the EIB Simulator.

V. CONCLUSIONS

The constructed EIB system simulator together with the EIB component environment forms a complete framework for efficient teaching of intelligent building design and component programming. The current experience with the framework shows that the scope of the simulator covers all the features necessary in performing student assignments and in experiments associated with the EIB system. The component environment seems to have clear structure enabling the students to create good architectural models and learn good practices in component programming.

The current state of the teaching framework allows for very interesting laboratory assignments where teams of students construct sophisticated near-commercial systems. It is possible to assign projects to groups of students that collectively design and construct component-based applications communicating with the EIB simulator. Further development of the presented teaching framework would allow on-line remote testing of the application through the internet or the teaching intranet. This should include the construction of EIB-IP gateways to the framework and the use of Web cameras.

REFERENCES

[1] G. Booch, J. Rumbaugh, I. Jacobson, “The Unified Modeling Language User Guide”, Addison-Wesley, 1998.
[2] EIBA, “Project Engineering for EIB Installations, Basic Principles”, EIBA sc, 1998.
[3] T. Sauter, D. Dietrich, W. Kastner, “EIB, Installation Bus System“, Publicis, 2001.
[4] M. Seligowski, “Stanowisko laboratoryjne magistrali EIB z programowym modułem paneli użytkownika dla inteligentnego budynku – Master degree thesis”, Warsaw University of Technology, 2004.
[5] C. Szyperski, “Component Programming – Beyond Object-Oriented Programming, 2nd ed.”, Addison-Wesley, 2002.
[6] J. Templeman, R. J. Mueller, J. Mueller, “COM Programming with Microsoft .NET”, Microsoft Press, 2003.
[7] J. Templeman, D. Vitter, “Visual Studio .NET: The .NET Framework Black Book”, Coriolis Group Books, 2002.
[8] G. WesterMeir, T. Weinzierl, F. Schneider, “Interfacing the EIB Bus with Windows Computers”, EIB-Proceedings, part 3, 2000.