

Work in Progress - Module-Based Active Learning Approach for Introductory Level of Computer Engineering Curriculum

Tatyana Yakhno, Emine Ekin, and Tevfik Aktuglu
 tatyana.yakhno@izmir.edu.tr, emine.ekin@deu.edu.tr, tevfik.aktuglu@deu.edu.tr

Abstract - The paper describes a new approach to Computer Engineering curriculum design which has been made in Engineering Faculty of Dokuz Eylul University of Izmir, Turkey. This approach is based on a Module-Based Active Learning model. Knowledge units of introductory level are packed into modules each with duration of 5 weeks. All theoretical knowledge in the modules are connected into a concept map which shows the integration of knowledge inside the module. During the module study students have to implement a project which is a core of a module. This approach allows us to include the engineering design from the very beginning of education. Preliminary assessment shows the increasing level of self-confidence and satisfaction among students and instructors.

Index Terms – Curriculum design, Curriculum integration, Module-based learning, Project-based learning.

INTRODUCTION AND MOTIVATION

Rapidly growing technologies (such as Internet, networking, mobile computing, intelligent information systems, semantic Web, etc.) require serious revision of Computer Engineering educational programs. The paper describes a new approach to Computer Engineering curriculum design which has been made in Engineering Faculty of Dokuz Eylul University of Izmir, Turkey. The aim of our computer engineering curriculum is to produce programmers and software engineers who will work in industry and produce different types of software. Therefore all students need to learn to adapt to the rapidly evolving nature of the field.

The general revision of the curriculum was motivated by several factors:

- general dissatisfaction with students' skills and abilities in project design and development,
- desire to integrate engineering practice into the curriculum,
- students' low self-confidence, lack of communication skills, and absence of teamwork experience.

For computer engineers design relates to software and hardware components of modern computing systems and computer-controlled equipment. The purpose to link different courses and develop better design abilities leads us to the concept of curriculum integration. Curriculum integration is the process of experiencing and understanding connections and, because of this, seeing things whole. Generally integration can be defined as a curriculum approach that links together knowledge, skills and attitudes across the subject areas to develop a more powerful understanding of key ideas and prepares students for professional practice [1].

CURRICULUM STRUCTURE

As a basic teaching methodology we have used a module-based learning approach, which blends theory and practice, with theory guiding practice [2,3]. Following the IEEE/ACM recommendations [4] we have prepared the layout of the knowledge units taking into account the following principles:

- Keep the balance between software and hardware units.
- Select the sequence of units' components in a way which allows us to obtain meaningful integration of units.
- For the first and second year keep 3 knowledge units in one semester.

The main constructing block in our curriculum becomes a **module**, which contains integrated concepts, notions and problems from above mentioned knowledge units. Every module includes:

- Theoretical presentations;
- Problem solving sessions;
- Project design and implementation sessions;
- Lab settings.

The main consolidating component in the module is a **project**. The duration of one module is 5 weeks (we have three modules in one semester). Such a time period allows students to design a project and at the same time theoretical knowledge is integrated and related to projects. The whole

curriculum is packed into modules and includes: introductory modules, intermediate modules, and advanced modules. The main advantage of such an approach consists in continuous project design experience starting from the very beginning of education.

Introductory projects are specified by the university instructors and integrate the introductory courses of the computer engineering discipline and basic algorithmic courses. One of the difficult problems in curriculum design is the choice of the first programming language. Some instructors prefer to choose currently fashionable industry language, e.g. C# or Java, or a language that will be used later in upstream courses such as operating systems. We do agree that after graduating from the university computer engineering students must be able to program well, but we do believe that university education is not supposed to be a vocational training school for the latest industrial programming languages and programming tools. University should be a place and time when students are exposed to principled ideas on a regular and rigorous basis and see how these principles apply in the real world.

When we teach a programming language we have to focus on program design principles instead of the use of language constructs. Therefore the first language should introduce the principles of program design, state them explicitly as habits and have students practice them with numerous examples. In our department, where we have a 3-term long algorithms and programming track, we have made our choice in favor of starting with a dynamically typed functional programming language: Scheme [5].

For the first semester, where we cover the Discrete Mathematics, Computer Literacy, Algorithms and Programming knowledge units, we have 3 modules which are:

Module 1: Introduction to Programming.

Module 2: Functional Decomposition and Compound Data Types .

Module 3: Data Abstraction .

CASE STUDY: MODULE SCENARIO

As an illustration of our approach let us consider the second of our modules : Functional Decomposition and Compound Data Types.

Topics covered in this module are represented as a concept map and include:

1. *Graphs and Trees*
2. *Infix, prefix, postfix notation*
3. *Spanning trees, shortest path.*
4. *Graph models. Euler and Hamiltonian cycle*
5. *Tree recursion; Process types; Divide-and-conquer; Iteration;*
6. *Record types; Higher-order-functions; Major higher-order-functions as iteration construct.;*
7. *Direct access structure: Array;*
8. *Basic sorting and searching algorithms.*
9. *Introduction to operating systems utilities*

10. Networks

The project specified in this module aims to acquaint the students with the object-based design approach, as well as graphs and shortest path algorithms. Before delivering this project, the students have been equipped with basic knowledge on algorithms, lists, recursion, and the Scheme programming language in the first module.

As a project students have to design and implement a light-weight reservation system for an airline company, which includes at least the following operations:

- reservation of a seat,
- purchase of a ticket,
- cancellation of a reserved seat,
- returning a ticket,
- changing the departure time of a purchased ticket or a reserved seat,
- passenger list of a particular flight, and
- waiting list of a particular flight.

The Scheme implementation is expected to support simultaneous access of multiple users through web browser and windows-based interfaces.

SUMMARY

The impact of the proposed project-based model on student learning can be preliminarily summarized as follows:

- Students' satisfaction is significantly increased.
- Students' design ability is enhanced.
- Students' understanding on course material is improved.
- Students' presentation skills and team-working skills are significantly improved.

Class presentation performance showed that the students become more confident and more organized in presenting their design.

This project-based learning approach was applied for the first time in Turkey for Computer Engineering education. Except learning and teaching impacts we have got very rich social impacts. In particular, team work during a project design has helped students to overcome social and cultural restrictions, and engaged students with appreciation of effectiveness of cooperative work and group learning.

REFERENCES

1. Beane, James. A. "Curriculum integration: Designing the core of democratic education." Teachers College Press, Columbia University, 1997.
2. Albanese, M. A., Mitchell, S. "Problem-based learning: A review of literature on its outcomes and implementation issues." *Academic Medicine*, Vol. 68, No 1, 1993, p.52-81.
3. Perrenet, J. C., Bouhuijs, P.A., Smits J. "The suitability to of problem-based learning for engineering education: Theory and practice." *Teaching in Higher Education*, Vol.5, No 3, 2000, p.345-358.
4. Computer Engineering. Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering. *IEEE Computer Society, ACM.*, 2004.
5. Felleisen M., Findler R. B., Flatt M., Krishnamurthi S., "The Structure and Interpretation of the Computer Science Curriculum." *Journal of Functional Programming*, Vol. 14 , No 4, 2004, p. 365 – 378.