Education in informatics at Sofia university - current status and future plans

Roumen Nikolov, Sylvia Ilieva

EDUCATION IN INFORMATICS AT SOFIA UNIVERSITY - CURRENT STATUS AND FUTURE PLANS

Roumen Nikolov
FMI – Sofiia University
5, J. Bouchier str., P.B. 48
Sofia 1164, Bulgaria
roumen@fmi.uni-sofia.bg
http://www-it.fmi.uni-sofia.bg/

Sylvia Ilieva
FMI – Sofiia University
5, J. Bouchier str., P.B. 48
Sofia 1164, Bulgaria
sylvia@fmi.uni-sofia.bg
http://www-it.fmi.uni-sofia.bg/

ABSTRACT

The paper presents some real experiences, emerging models and lessons learnt based on the case of Sofia University - Faculty of Mathematics and Informatics (FMI), and its partners. Sofia University has always played a very important role for the development of the country. The university experiences a lot of challenges related to the overall transformation of the economic and social system in the country, the changing models of education, the new role of the universities in the knowledge-based society and the great demand for Information and Communication Technologies (ICT) specialists. Sofia University is also challenged by the opportunity to be actively involved in the development of the European space of higher education. The outcomes of an internal university project aimed at the establishment of ACM/IEEE based BSc programs in Computer Science, Software Engineering and Information Systems are described and analyzed. Some experience in developing a layer of MSc programs as a live link to the professional ICT society and ICT industry is also described. Some future plans for the development of the Computing Curricula and its coordination with the European ICT education, research and innovation agenda are presented as well.

1. Introduction

Sofia University “St. Kliment Ohridski” is the first school of higher education in Bulgaria. Today it is the largest and most prestigious educational and scientific centre in the country, with over 35,000 students studying in 76 Bachelor’s and over 200 Masters degree programs. Since its establishment the university has always played a very important role for the development of the country. The university experiences a lot of challenges related to the overall transformation of the economic and social system in the country, the changing models of education, the new role of the universities in the knowledge-based society and the great demand for ICT specialists. Sofia University is also challenged by the opportunity to be actively involved in the development of the European space of higher education.

The Faculty of Mathematics and Informatics (FMI) at Sofia University is a leading educational and research organization in the field of Informatics and ICT. An internal university project for re-designing the computing curricula according to the Association for Computing Machinery/Institute of Electrical and Electronics Engineers (ACM/IEEE) guidelines started four years ago. Following the ACM/IEEE CC2005 series recommendations [1] FMI has developed the BSc programs in: Computer Science [1], Software Engineering [3] and Information Systems [4] – see Figure 1. The main aims were to introduce some well established curriculum standards and new style of teaching. The main challenge was how to satisfy recommendations of the ACM/IEEE Computing

Curricula, the Bologna Declaration, and several European ICT curricula recommendations [5], [6].

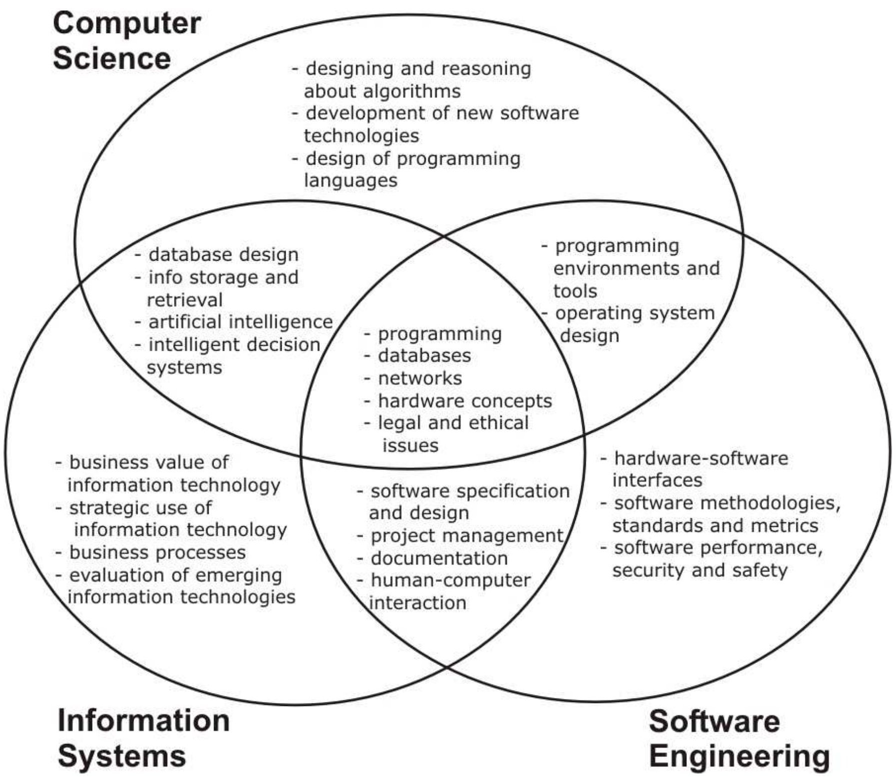

Computer Science
- designing and reasoning about algorithms
- development of new software technologies
- design of programming languages
- database design
- info storage and retrieval
- artificial intelligence
- intelligent decision systems
- programming environments and tools
- operating system design
- programming
- databases
- networks
- hardware concepts
- legal and ethical issues
- business value of information technology
- strategic use of information technology
- business processes
- evaluation of emerging information technologies
- software specification and design
- project management
- documentation
- human-computer interaction
- hardware-software interfaces
- software methodologies, standards and metrics
- software performance, security and safety
Information Systems
Software Engineering

Fig. 1. Intersection of the BSc Computing Curricula in Computer Science, Software Engineering and Information Systems

Quality of education was considered as the major indicator for success while designing and implementing these programs. In addition, FMI offers large number of Master degree programs, such as *Software Engineering (SE), Information Systems*, *eBusiness and eGovernance*, *Artificial intelligence*, *Mobile Technologies and Distributed Systems*, *eLearning*, *Information Security*, *Mechatronics and Robotics*, *Computer Graphics*, *Computational Science and Engineering*, etc.

Most of the MSc programs include compulsory internship student placement. The programs are oriented mostly towards the needs of the global ICT industry and the priorities of the European RTD Programmes in the field of ICT. In order to satisfy the ICT industry needs of technology managers Sofia University established a co-operation with Stevens Institute of Technology, USA, for local delivery of a MSc program in Information Systems [7]. A MSc program in Technology Entrepreneurship and Innovation is under preparation for the next academic year. Such measures would help the country to bridge the existing gap of ICT specialists and managers. The international cooperation opens new opportunities for improving the quality of education and for positioning Sofia University in the European Higher Education space. In the process of negotiation and preparation are some joint Master degree programs with European universities, e.g. a MSc Program in Networked Computing together with University of Reading, UK. A strategic cooperation with many outstanding universities in the frames of the Intel Multi-core Computing Program opens new doors for international cooperation and adaptation of the Computing curricula according the most recent developments in the field and adapting it to some emerging technologies. Such international co-operation programs provide direct mechanisms for increasing the quality of education. They are very attractive for students and motivating for teachers.

One way to have access to contemporary equipment and software is the strong cooperation with the ICT industry. A good example in this direction is the Sofia University Cisco Regional Academy, which was established in 2001 in order to implement the Cisco model of Public-Private-Partnership at a regional and national level. Similar cooperation programs were initiated with Microsoft (Microsoft IT Academy), Oracle (Oracle Academic Initiative), HP (HP training centre), SAP labs, IBM and other international IT vendors. The cooperation with the local ICT industry is considered as a strategic goal of FMI in order to better adapt the computing curricula according to the needs of this industry, and to open doors for a professional carrier for every FMI graduate. Several bi-lateral programs for carrying out student internship programs with some local Information Technology (IT) companies (e.g. Rila Solutions, Fadata, Technologica, Sirma, etc.) were established. The major Bulgarian ICT associations, such as BASSCOM (Bulgarian Association of Software Companies), BAIT (Bulgarian Association of Information technologies) and ASTEL (Telecommunications Association), also provide their services and support for achieving better cooperation between FMI and the Bulgarian ICT companies.

This paper describes some real experiences, emerging models and lessons learnt based on the case of Sofia University

The paper is organized as follows: Section 1 is the Introduction and gives the background information about Sofia University, FMI, its MSc programs and cooperation with industry. Section 2 presents the curricula development process in FMI for the BSc program in Software Engineering. Section 3 describes the real implementation of the BSc of Software Engineering curricula, as part of education in Informatics, and lists some of the main courses. Section 4 focuses on project based learning and internships as essential part of practical units education. Section 5 discusses open issues and perspectives in Sofia University. Section 6 concludes the paper.

2. Curricula Development Process in FMI

A growing demand for university level education is observed in Bulgaria – there is increasing number of candidate-students in conditions of a constant annual enrolment. Changes in the labor market and in the demand for new types of qualifications and a new graduate profile, as well as a demand for lifelong learning/continuing education, are the reasons for increasing the number of part-time students and older students. The only possible solution of this problem is to apply flexible ICT based education at university level based on advanced research outcomes and technology solutions.

An internal university project for re-designing the computing curricula started four years ago with Computer Science curricula. In this section we will describe the process we followed in developing a BSc program in Software Engineering

2.1 Prerequisites and Goals

Bulgaria is among the main destinations of investments of software development companies who are attracted by the well educated and qualified IT workforce. The Bulgarian IT industry is rapidly growing and the shortage of professionally educated software developers is among the main barriers for faster economic development of the country. Since ICT is recognized as one of the priorities of the country development, we are very much concerned with the problem of improving the professional education and training in the field of software development. FMI has got some very good achievements and traditions related to mathematics and computer science (with some mathematical roots). The current tendency shows that software development becomes a more ‘engineering’ type of field where FMI does not have much experience and traditions. It happens, however, that software design differs from all other forms of design, whether these are “hard design” (for material objects) or “soft design” (for processes or policies) [8]. Socha and Walter state that “*Understanding whether designing software is different from designing other things will help our software discipline learn from other disciplines, and contribute to other design disciplines. We believe that this understanding is critical, since most design fields are seeing an increasing role for software as aids to the creation of the design, or of software as part of making a more interactive end product. We believe that in the future hard design processes will adopt and include many of the methods of successful software design practice*.”

Dym et al, define Engineering Design as “*a systematic, intelligent process in which designers generate, evaluate, and specify concepts for devices, systems, or processes whose form and function achieve clients’ objectives or users’ needs while satisfying a specified set of constraints*” *[9].* The output of the engineering design process is generally a set of detailed specifications which can range from a simple text document to complex animated CAD drawings [10]. The steps in a hard design process could be defined as follows:

- Client Statement and Need;
- Problem Definition;
- Conceptual Design;
- Preliminary Design;
- Detailed Design;
- Design Communication;
- Final Design (Fabrication specifications and documentation);

These steps correspond in great extent to the Waterfall software development model originally described by Royce [11]. However, the advance of the Agile Programming models poses some questions like: *“Is Design Dead*?” [12]. The answer of Fawler is “*No*”, but he emphasizes that the nature of the software design has changed: “*For many that come briefly into contact with Extreme Programming (XP), it seems that XP calls for the death of software design. Not just is much design activity ridiculed as "Big Up Front Design", but such design techniques as the UML, flexible frameworks, and even patterns are de-emphasized or downright ignored. In fact XP involves a lot of design, but does it in a different way than established software processes. XP has rejuvenated the notion of evolutionary design with practices that allow evolution to become a viable design strategy. It also provides new challenges and skills as designers need to learn how to do a simple design, how to use re-factoring to keep a design clean, and how to use patterns in an evolutionary style*”.

Fawler identifies a set of specific skills that an XP software designer has to possess, such as:

- A constant desire to keep code as clear and simple as possible;
- Refactoring skills so you can confidently make improvements whenever you see the need;
- A good knowledge of patterns: not just the solutions but also appreciating when to use them and how to evolve into them;

- Designing with an eye to future changes, knowing that decisions taken now will have to be changed in the future;
- Knowing how to communicate the design to the people who need to understand it, using code, diagrams and above all: conversation.

The SE Curricula Task Force at FMI has faced the challenge of lack of any traditions in the engineering field. This was partially compensated by involvement of a number of new professors at FMI who had graduated from some engineering discipline. The team has also faced the lack of enough experience in SE education world-wide. Some of the basic principles integrated into the FMI SE Curricula are described bellow.

In order to enable undergraduates to become professional software engineers, the curriculum designers should note the three essential traits of professionalism that are commonly recognized [13]:

- A body of knowledge unique to the profession;
- High levels of responsibility and accountability via a code of ethics and professional practice;
- Competence and commitment in the profession demonstrated by certification by a professional organization or a legally recognized license to practice.

We identify three different levels of knowledge about a subject *a* person acquires*:*

- **Rote Knowledge**: Rote knowledge entails having a basic understanding of the overall structure of a subject. This knowledge is superficial and de-emphasizes details. A student with rote knowledge of the area has some intuitive understanding of the subject but few tools with which they can solve concrete problems.
- **Application Knowledge**: Application knowledge is deeper than rote knowledge. Students with application knowledge can solve concrete problems given to them that pertain to the subject area. To attain application knowledge some amount of formalization of the subject is required.
- **Correlation Knowledge**: Correlation knowledge is the deepest form of knowledge. Given a problem, the student is able to identify a technique that might be used to solve the problem and then proceed to apply the technique. This knowledge requires deep intuition on the subject and is consequently the most difficult to teach or acquire.

The body of knowledge and skills of the SE program graduates were also adapted to some of the basic ICT job profiles related to software development [8], such as: "*Software and Application Development*", "*Software Architecture and Design*", "*Systems Specialist*", "*ICT Project Manager*", etc.

2.2 Development process

The initial step of the Software Engineering Curricula Task Force at FMI was to create a vision and plan for curricula development. The development process consisted of three main phases – analyses, design and evaluation. We will briefly describe each phase.

2.2.1 Analyses

During this phase some recommendations and views from different stakeholders and committees were analysed. A number of strategic documents related to education in ICT and particularly in Software Engineering were reviewed, such as:

- *Bologna and Lisbon strategies*

- *Career Space Industry Consortium*
- *ACM/IEEE recommendations*
- *Industry companies – Intel, IBM, Microsoft, HP, etc - initiatives for Universities*

The Career Space ICT Consortium believes that there is no one single way to design the best ICT curriculum [5]. On the contrary, if the cultural diversity in Europe is to be used to give competitive advantage to this region, each university must find its own best solution. They believe that a framework based on experience and best practice can lead to a set of useful guidelines, and that the universities can find their own way to success. The Career Space consortium recommends that ICT Curricula should consist of the following core elements (Figure 2):

- a scientific base of 30%,
- a technology base of 30%,
- an application base and systems thinking of 25% and,
- a personal and business skills element of up to 15%.

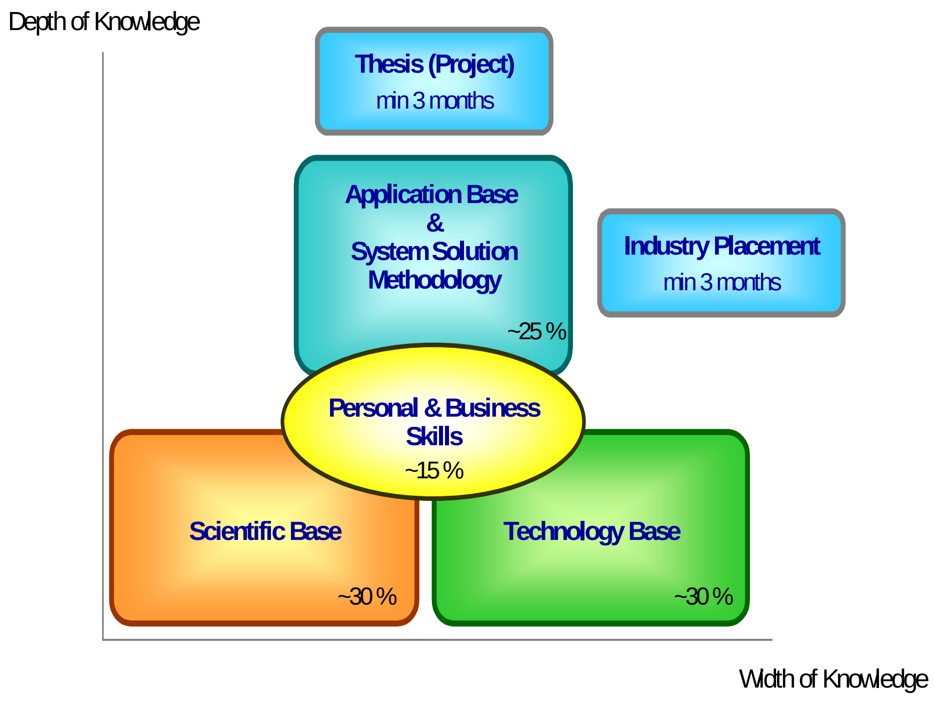


Fig. 2. Scope of Competence, showing Model ICT Curriculum content (adapted from Career Space, 2002)

At the second step a number of well established BSc programs in Software Engineering in Europe, USA and elsewhere were examined and compared. The needs of the Bulgarian ICT industry were also identified through a survey with a specially designed questionnaire followed by a number of individual meetings and discussions. In addition, a representative of the Bulgarian software companies BASSCOM was associated to the FMI SE task force.

The high school background of the prospective students was analyzed and taken into the consideration as well.

2.2.2 Design

This was the most difficult and long lasting phase. First, a template for course description was created, which had to be in line with the requirements of the National Accreditation Agency. An additional requirement was to meet most of the criteria for accreditation of the computing programs of the Accreditation Board for Engineering and Computing [14]. Later on the main courses with their interdependences and relations were identified. Then the documentation of the courses was developed, which included course distribution, teaching hours to topics, the most appropriate text books, etc. In case there was no textbook appropriate, the need of writing such textbook was identified. In both cases the lecturers were obliged to provide a well prepared course reader and lecture presentations in powerpoint. More than a half of the courses were prepared for on-line delivery as well. During the curricula design we were working hard in order to find a well-balanced mix of theoretical and practical units.

2.2.3 Evaluation

Each of the courses and the curricula as a whole were discussed and evaluated by an Advisory Board of specialists respected at national level and a number of external experts in ICT. Based on their feedback the curriculum was refined. A number of surveys among the students as well as among their employers were also organised.

Having in mind that Software Engineering is extremely dynamic field we need to go through those 3 phases iteratively all the time. So we foresee that the courses would be refined on an annual basis and the students are encouraged to give feedback and suggestions.

3. Software Engineering Curricula Implementation at Sofia University

The duration of our BSc program in SE is 4 years. Each course gives the students credits synchronized with the European credit transfer system (ECTS).

The core of software engineering courses cover all the essential elements required by the software engineering profession, such as: *Software Design, Formal Methods, Human Computer Interaction, Measurement and Metrics, Process Models, Project Management, Quality Assurance and Standards, Requirements Specification, Testing, Computer Aided Software Engineering (CASE)Tools*, etc*.*

- **Design and Implementation of Software for the Web.** This course teaches students how to develop software for web applications. The concepts of client-server computing, theories of usable graphical user interfaces, and models for web-based information retrieval and processing are covered. Goals are to understand how to design usable software interfaces and implement them on the web, learn how to build software that accepts information from users across the web and returns data to the user, and understand how to interact with database engines to store and retrieve information. Specific topics that are included are HTML, CGI programming, Java, Java applets, Javascripts, and Java servlets.
- **Software Construction.** The students are taught in in-depth study of software construction using a modern language. Some concepts, such as information hiding, data abstraction, concurrency, and object-oriented software construction, are discussed.
- **Software Requirements and Prototyping.** In-depth study of methods, tools, notations, and validation techniques for the analysis and specification of software requirements. Students participate in a group project on software requirements.

- **Software Design.** The course focuses on concepts and methods for the architectural design of large-scale software systems. Fundamental design concepts and design notations are introduced. Several design methods are presented and compared, with examples of their use. Students participate in a group software design project.
- **Formal Methods and Models in Software Engineering.** This course presents formal mechanisms for specifying, validating, and verifying software systems. Program verification through Hoare's method and Dijkstra's weakest preconditions are also covered. In addition, students will learn about integration of formal methods with existing programming languages, and the application of formal methods to requirements analysis, testing, safety analysis, and object-oriented approaches.
- **Software Project Management.** The main topics discussed in the course are: lifecycle and process models; process metrics; planning for a software project; mechanisms for monitoring and controlling schedule, budget, quality, and productivity; and leadership, motivation, and team building.
- **Software Project Laboratory.** Students are involved in analysis, design, implementation, and management of a software system project. Students work in teams to develop or modify a software product, applying sound principles of software systems engineering. Both industrial and academic standards are used to assess the quality of the work products.
- **Software Engineering Economics.** The course covers: quantitative models of the software lifecycle; cost-effectiveness analysis in software engineering; multiple-goal decision analysis; uncertainty and risk analysis; software cost estimation; software engineering metrics; and quantitative lifecycle management techniques.
- **Object-Oriented Software Development.** The course includes: principles of object-oriented design, development, and programming; relationships between object-oriented design concepts and software engineering principles; techniques of object-oriented design and programming.
- **User Interface Design and Development.** The students are taught in in-depth study of principles of user interface design, development, and programming. It includes user psychology and cognitive science, adaptive user interfaces, icon and window design, command language design, user guidance systems, and collaborative working.
- **Software Testing and Quality Assurance.** The students are taught in concepts and techniques for testing software and assuring its quality. Software testing at the unit, module, subsystem, and system levels is discussed. Automatic and manual techniques for generating and validating test data are taught. The testing process, static vs. dynamic analysis, functional testing, inspections, and reliability assessment are presented.
- **Software Engineering for the World Wide Web.** The course includes detailed study of the engineering methods and technologies for building highly interactive web sites for e-commerce and other web-based applications. Engineering principles for building web sites that exhibit high reliability, usability, security, availability, scalability and maintainability are presented. Methods such as client-server programming, component-based software development, middleware, and reusable components are taught.

- **Special Topics in Software Engineering.** This course covers special topics not occurring in the regular SWE sequence. May be repeated for credit when semester topic is different.
- **Advanced Software Requirements.** This course focuses on state-of-the-art and state-of-the-practice in software requirements engineering. In-depth coverage of selected methods, tools, notations or validation techniques for the analysis and specification of software requirements. The course work includes a project investigating or applying approaches to requirements engineering.
- **Advanced Software Design Methods.** This course covers study of advanced design methods for large-scale software systems, including concurrent, real-time and distributed systems. The course work includes a project investigating or applying software design methods.
- **Directed Readings in Software Engineering.** The students are obliged to do analysis and investigation of a contemporary problem in software engineering. Prior approval by a faculty member who supervises the student's work is required.
- **Thesis.** The Thesis is based on a research project completed under the supervision of a faculty member, which results in a technical report accepted by a three-member faculty committee. The report must be defended in an oral presentation.

A set of courses related to professional practice, computing careers, history of computing, impact of computers on society, legal and ethical responsibilities, computing profession, and computer law were introduced. The typical duration of such courses is one semester:

- SP1. History of computing [core]
- SP2. Social context of computing [core]
- SP3. Methods and tools of analysis [core]
- SP4. Professional and ethical responsibilities [core]
- SP5. Risks and liabilities of computer-based systems [core]
- SP6. Intellectual property [core]
- SP7. Privacy and civil liberties [core]
- SP8. Computer crime [elective]
- SP9. Economic issues in computing [elective]
- SP10. Philosophical frameworks [elective]

4. Education materials

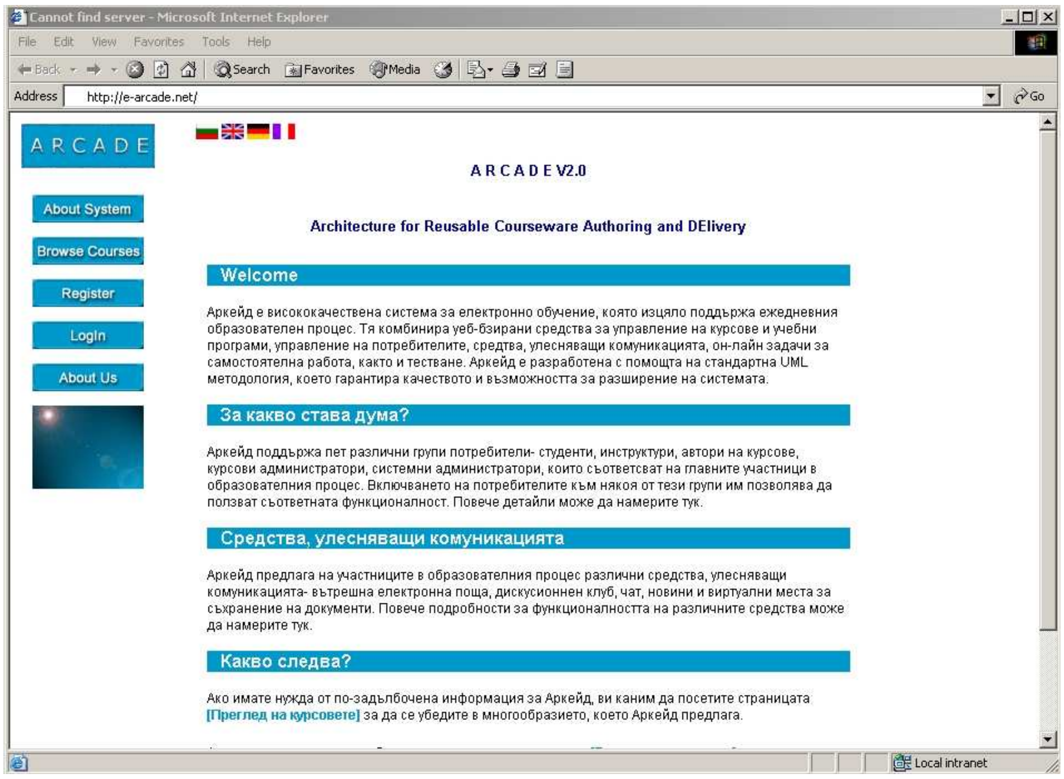
For every course a set of materials is prepared including:

- Presentations
- Tutorials
- Exercises
- Tests
- Projects
- List of recommended papers from conferences, journals or technical reports.

In this section we focus our attention to projects and internships as essential parts of the practical units education.

4.1 Project base learning

The most popular pedagogical model for teaching design is project-based learning (PBL), i.e. students to learn design by experiencing design as active participants [9]. The ability to work as an effective member of a development team is a primary goal of the SE education and one of the the main learning outcomes which the accreditation agencies and commissions would be happy to observe [15]. The SE Master education at FMI is based on team work on class projects which are supposed to produce useful tools expected to enrich either the software environment at the work place of the students or the SE educational environment. The students are involved in development, maintenance and refinement of real software which helps them understand the subject better and adapt to a team-work style. Some of the projects the students work on come from the research areas of the professors, and they might eventually evolve into diploma theses. In some cases such projects mature towards a PhD thesis as well. In other cases projects come from the current internal university needs. An example of such a project is the eLearning Management System ARCADE [16], which was developed by a team of four MSc students guided by a university professor. This system has been used as an eLearning platform for delivering courses at MSc level for more than six years now. The system has been gradually extended and enriched with new functionalities and services and it was used as a basis for additional five MSc and two PhD works.


Cannot find server - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://e-arcade.net/
Go
ARCADE
About System
Browse Courses
Register
Login
About Us
Local intranet

A R C A D E V2.0

Architecture for Reusable Courseware Authoring and DElivery

Welcome

Аркейд е висококачествена система за електронно обучение, която изцяло поддържа ежедневния образователен процес. Тя комбинира уеб-бзирани средства за управление на курсове и учебни програми, управление на потребителите, средтва, улесняващи комуникацията, он-лайн задачи за самостоятелна работа, както и тестване. Аркейд е разработена с помощта на стандартна UML методология, което гарантира качеството и възможността за разширение на системата.

За какво става дума?

Аркейд поддържа пет различни групи потребители- студенти, инструктури, автори на курсове, курсови администратори, системни администратори, които съответсват на главните участници в образователния процес. Включването на потребителите към някоя от тези групи им позволява да ползват съответната функционалност. Повече детайли може да намерите тук.

Средства, улесняващи комуникацията

Аркейд предлага на участниците в образователния процес различни средства, улесняващи комуникацията- вътрешна електронна поща, дискусионнен клуб, чат, новини и виртуални места за съхранение на документи. Повече подробности за функционалността на различните средства може да намерите тук.

Какво следва?

Ако имате нужда от по-задълбочена информация за Аркейд, ви каним да посетите страницата [Преглед на курсовете] за да се убедите в многообразието, което Аркейд предлага.

Figure 3. ARCADE – a Learning Management System developed as a team work of four MSc students

The ARCADE environment (Figure 3) has been designed and developed as a complex environment for collaborative and active e-learning. Its main merits include the following:

- It is based on standards – XML/XSL, Java, IMS, LTSC LOM, QTI, SCORM; UML widely use in development process

- Open - the use of XML import/export will make the system open for information exchange (including reusable courseware content) with other similar systems;
- Modular - different project modules are made as much independent as possible; easy plug-in of new functional modules;
- Portable – Java servlets and JSP technology plus MySQL or Oracle support make ARCADE portable between different platforms and operation systems.

4.2 Internship

Comparing our BSc in SE program with similar programs in UK we can see that we are missing the 1 year placement. Instead, we have provided internships at the end of the last academic year. The internship procedure we have accepted is as follows:

- FMI team contacts Bulgarian private software companies and governmental organizations in order to have every three months the pool of the current requests for interns updated.
- That pool of requests is made available to the students and they are expected to apply for the preferred one.
- In case there are more applications for a given request than needed the selection procedure is started based on students' skills and motivation. In some cases, depending on the company/organization, it is possible that the requests are extended to more people or even that two teams work in parallel on the same task.
- Every student has two tutors – one from the University, and the other from the company
- Each internship ends with evaluation by the company tutor.

Usually after the internship students receive offers for full time position at the same company.

The other new schema for internships is in the process of establishment. It foresees internships at other European universities we are cooperating with.

5. Perspectives and open issues

The future plans of the FMI include the following issues:

- Gradual increase of the number of enrolled BSc and MSc students in the ICT related programs;
- Development of a subset of BSc and MSc programs taught in English and marketing them on the regional level, mostly in the South-Eastern Europe;
- Development of partnership with other European and USA universities, including establishment of joint programs with dual degrees;
- Developing a complete educational virtual environment (Sofia University Virtual Campus) and increasing the courses and programs taught in e-learning mode;
- Establishment of a set of ICT research labs and further involvement of PhDs and other students in concrete research and technology development activities;
- Strengthening the innovation and entrepreneurship capacity of the university and establishment of strong technology transfer links with ICT industry and government;
- Extending the network of partnering organizations from ICT industry both locally and internationally and providing opportunities for cross-sector mobility of students, researchers, professors and ICT professionals.

6. Conclusions

We consider the educational change in the field of ICT at university level as being among the main factors for the successful development of Bulgaria's Knowledge Economy. In addition, the ICT education and professional development of ICT specialists is a very strong factor for building a competitive country economy. Sofia University is prepared to take a leading role in this process.

References

1. ACM/IEEE (2005), Computing Curricula 2005 - The Overview Report, ACM and the IEEE Computer Society
2. ACM/IEEE (2001), Computing Curricula 2001, Computer Science, ACM and the IEEE Computer Society
3. ACM/IEEE (2004), Computing Software Engineering 2004, Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering, Report, ACM and the IEEE Computer Society
4. ACM/AIS/AITP (2002), Model Curriculum and Guidelines for Undergraduate Degree Programs in Information Systems, Association for Computing Machinery (ACM), Association for Information Systems (AIS) & Association of Information Technology Professionals (AITP)
5. Career Space (2001a), Curriculum Development Guidelines: New ICT Curricula for the 21st Century, Designing Tomorrow's Education, http://www.career-space.com/
6. Career Space (2001b), Job Profiles, http://www.career-space.com/
7. MSIS (2007), Master of Science in Information Systems, http://www-it.fmi.uni-sofia.bg/msis/
8. Socha, D., Walter, S. (2006), *Is Designing Software Different From Designing Other Things?*, International Journal of Engineering Education, Special Issue on *Learning and Engineering Design*, Vol. 22 No. 3,
9. Dym, C., Agogino, A., Eris, O., Frey D, Leifer, L (2005), *Engineering Design Thinking, Teaching, and Learning*, Journal of Engineering Education, January
10. Dym C. & X Little, P. (2004), *Engineering Design: A Project Based Introduction*, John Wiley and Sons, Hoboken, NJ,
11. Royce, W. (1970), Managing Development of Large Software Systems, Proceeding of IEEE WESCON, August
12. Fawler, M. (2000), *Is Design Dead?*, http://www.martinfowler.com/articles/xp2000.html
13. Tse, T. H. (2004), Computing Curriculum—Software Engineering: Its Impacts on Professional Software Engineering Education, Proceedings of the 28th Annual International Computer Software and Applications Conference (COMPSAC'04), IEEE
14. ABET (2006), Criteria for Accrediting Computing Programs, http://www.abet.org/
15. Sims-Knight, J. E,. Upchurch, R. L, Powers, T. A., Haden,S Topciu R (2002), *Teams in Software Engineering Education*, 32nd ASEE/IEEE Frontiers in Education Conference, November 6 - 9, Boston, MA, IEEE

16. Bontchev B., T. Iliev (2003), *ARCADE - a Web-based Authoring and Delivery Platform for Distance Education*, 1st Balkan Conference on Informatics (BCI'2003), Thessaloniki, Greece, 21-23 of November.