IRRODL
THE INTERNATIONAL REVIEW OF RESEARCH IN OPEN AND DISTANCE LEARNING

Multi-Country Experience in Delivering a Joint Course on Software Engineering – Numerical Results

Zoran Budimac[1], Zoran Putnik[1], **Mirjana Ivanović**[1], Klaus Bothe[2], Katerina Zdravkova[3], and Boro Jakimovski[3]
[1]University of Novi Sad, Serbia, [2]Humboldt University Berlin, Germany, [3]Ss Cyril and Methodious' University in Skopje, Macedonia

Abstract

A joint course, created as a result of a project under the auspices of the 'Stability Pact of South-Eastern Europe' and DAAD, has been conducted in several Balkan countries: in Novi Sad, Serbia, for the last six years in several different forms, in Skopje, FYR of Macedonia, for two years, for several types of students, and in Tirana, Albania, in the form of a crash, seven-day course, for the last two years. In this paper, we will put an emphasis on the assessment methods used within these courses, and compare them with the 'original' course that has been conducted at the Humboldt University in Berlin for almost a decade. Having a good environment for comparisons we draw some conclusions about teaching software engineering in different environments.

Keywords: Common course; joint project; multiple universities; software engineering

Introduction

Coinciding with the trends in European high education, under the auspices of the 'Stability Pact of South-Eastern Europe' and 'DAAD, Deutscher Akademischer Austausch Dienst' ('German Academic Exchange Service'), a project was established in 2001. The main idea of the project was to create and develop common courses in several fields of computer science. Also, the intention of the project was to enable the usage of shared materials for courses at a wide range of universities in countries participating in the project.

The project joined participants from 15 universities and nine countries: Germany, Serbia, FYR of Macedonia, and Bulgaria being the core members, and Croatia, Bosnia and Herzegovina, Romania, Albania, and Montenegro, as associate members. More about the project, its goals, and members can be found in the Joint SE course homepage (2013), as well as publications by Zdravkova, Bothe, and Budimac (2003) and Bothe et al. (2003, 2005, 2009), while the experiences gained were described by Budimac et al. (2008, 2009, 2011).

The general goal of the project is improvement and adjustment of educational processes in South-Eastern Europe, with respect to the current and modern trends of countries within the European Union, and from the start it managed to fulfil several more specific goals:

- inclusion of 'Software Engineering' as a stand-alone course and a core course in universities' curricula of participating countries;
- consensus in creation of a joint course, including determination and selection of appropriate topics to be included as the basis for the common pool of topics;
- creation and development of joint teaching, examination, and assessment material for the selected topics: slides, case studies, assignments, exam questions, literature...;
- establishment of e-Learning facilities, used both as a simple repository of teaching materials and more subtle materials in a form of e-Lessons;
- establishment of a research and education framework as the basis for future educational and scientific cooperation.

These goals are implemented through cooperation in creation, improvement, and enhancement of teaching materials, and production of a distributed, Internet-based, multilingual university course. The joint course in software engineering originated from the course that has been conducted at the Humboldt University in Berlin for several years. Its main objective was to present introductory notions and principles of the discipline, including a wide spectrum of sub-areas suggested by the ACM and IEEE societies (Computing Curricula, 2001) and others (Bran et al., 2001). Thus, the course

covers more than 85% of the basic lessons suggested in 'Curricular guidelines for undergraduate programs in computing.'

Since this paper will deal mostly with exercises and assessment, it is interesting to notice that the course is accompanied by a pool of case studies discussed during lectures and processed through assignments. From this pool lecturers/instructors are free to select the most suitable one(s). In addition, there is a pool of assignments referring both to the course contents and case studies, thus forming the base for specific exercises. Together with the assignments, sample solutions, correction hints, and typical errors are collected and presented to students when appropriate. Through these additions, flexibility is added to the course.

After practical experiences in running (almost) the same course in four countries with five educational directions, we collected numerical data that could enable understanding of possible differences between conducted courses with respect to students and their environment. Thus, the goal of this paper is not only to describe our project but to present our conclusions based on collected numerical data. In our efforts to understand possible differences we used a field research technique (collecting existing records) and partly also the survey research method. Both approaches belong to qualitative research methodologies whose main aim is to understand and possibly predict the behaviour of the phenomenon under investigation.

Related Work

Our international educational joint project is not alone in its endeavour to improve teaching, and there are other similar projects that can be found in books and research papers. There are for example three other projects dealing with the same field of software engineering, such as those explained by Modesitt (2002), Doberkat et al. (2004), and Hilburn et al. (2003). Others we can mention are Ariadne (2013) and Merlot (2013) which are involved in some other fields of computer science, but still gather participants from different countries.

What we see as a major difference between our project and those mentioned is the methodology of course creation. While the mentioned courses created a relatively independent set of courses, from which participants can choose and adjust those they prefer and can include into their curricula, our aim was to create the course as a whole, including complete teaching materials, lectures, assignments, case studies, exam questions, and even lists of typical errors made by students. By this approach, our aim was to ease material reusability and enable use of these resources even to those lecturers who do not have software engineering as a primary field of interest.

Since as a consequence and the follow-up of our project, within a Tempus project (TEMPUS course homepage, 2013), a whole curriculum for master studies in the field of software engineering was created, two other projects that had the same approach as this

are worth mentioning. As presented by Caplinskas and Vasilecas (2003), the idea of the *MOCURIS* project was to develop the whole project, regarded as "... a system composed of courses, modules, labs, projects, and other components." Pretty much the same problem we encountered is mentioned in this paper, and that is the problem of "existing staff abilities." As the authors say "... no one separate university is able to implement such curriculum separately, because of the shortage of human and other resources." The same solution was implemented by both projects, and that is accumulation of resources of the partners, exchange of not only ideas and opinions, but also of teaching staff as necessary.

The second project that started with the need to exchange experiences and views, but concluded with the development of the whole master curriculum, is mentioned by Tibaut et al. (2013). They created an e-Learning environment for an international master level program in information technologies for the field of architecture. This environment "... integrates resources (units of study, learning management system, virtual classroom, teachers and students) from five European universities." What differentiates this project from ours is the fact that while we aimed to combine expertise and experiences of lecturers, project members, and create *joint* courses, the ITC-Euromaster project decided to join individually and separately created courses, accepted by the project consortium, and offer their use to the other project members via an e-Learning system that is the integral part of a project. This meant also inclusion of lecturers through a video conferencing system, opposite to our approach where lecturers are present in person, until individual project participants develop their own teaching staff.

Structure of the Joint Course

During the last several years, the same version of the course has been conducted by some participants of the project.

The Humboldt University in Berlin has the longest tradition in conducting the course, where it has been conducted for more than a decade.

At the University of Novi Sad, Faculty of Sciences, the course has been conducted in several different ways:

- for undergraduate students of computer science, eight years,
- for undergraduate students of the direction 'Professor of Geography and Informatics', also eight years,
- two years for postgraduate students of computer science. After that, as a part of Tempus project CD-JEP-18035-2003 (TEMPUS course

homepage, 2013), master study curriculum in software engineering was created and conducted (Bothe at al., 2009a).

At the 'Ss Cyril and Methodius' University in Skopje, Faculty of Natural Sciences and Mathematics, the course has been conducted for undergraduate students of computer science, the first time during the school year 2007-08, and since then.

At the Polytechnics University of Tirana, a seven-day crash-course was conducted by a professor from Berlin and assistant from Novi Sad in the spring semester of 2006-07 and afterwards each year.

Even though all of the lecturers had the freedom to choose the methods of course conduction, the basic structure of a course is rather similar at all universities, as agreed within the project.

The course consists of 28 topics that cover most of the introductory notions from the software engineering field. The complete list of 28 topics is given below.

Part I: Introduction to Software Engineering

Topic 1: *What is software engineering?* gives insight into motivation, areas of interest, definitions, and history of the field.

Topic 2: *Quality criteria for software products* covers classification and definitions of software quality criteria, mentioning standards dealing with questions of quality.

Topic 3: *Software process models – introduction* explains the activities of software development, gives overview of models, and introduces details on selected process models.

Topic 4: *Basic concepts and software development documents* gives overview of concepts and documents created during the software development process.

Part II – Requirements Engineering (Analysis and Definition)

Topic 5: *Results of the 'Analysis and definition' phase* – Feasibility study, product model, and requirement document are discussed here.

Topic 6: *Cost estimation* – Costs, factors, and presentation of the 'function point' method are covered in the topic.

Topic 7: *Basic concepts of the function-oriented view* is devoted to function-tree and data-flow diagrams.

Topic 8: *Basic concepts of data-oriented view* – Data dictionary and entity relationship concepts are presented in the topic.

Topic 9: *Basic concepts of rule-oriented view* is concerned with concepts of rules, decision tables, and trees.

Topic 10: *Structured analysis* introduces notions of context diagram, DFD hierarchy, mini-specification, and shows the process of structured analysis.

Topic 11: *Basic concepts of state-oriented view* – Petri-nets, state automata, and activity diagrams are covered within this topic.

Topic 12: *Basic concepts of scenario-based view* is dedicated to concepts of collaboration diagrams and sequence diagrams.

Topic 13: *Object-oriented analysis* covers the transition process from object-oriented analysis to design, explaining activities, class diagrams, use cases, UML, and shows the process of object-oriented analysis, using case studies.

Topic 14: *Formal software specifications and program verification* – Z, algebraic, and Hoare logic are presented.

Part III - Design

Topic 15: *Overview of design activities* presents notions of software architecture, specification of components, quality assurance, giving an overview of some software architectures.

Topic 16: *Structured design* is mostly dedicated to structure charts, illustrating the process of structured design.

Topic 17: *Object-oriented design* covers the transition process from object-oriented analysis to design, explaining characteristic activities in more detail.

Part IV – Implementation and Testing

Topic 18: *Implementation* – Principles, methods, guidelines for the implementation phase of software development are presented.

Topic 19: *Systematic testing* gives a classification of testing methods, and discusses review/audit, control-flow and data-flow oriented methods.

Topic 20: *Functional testing* is dedicated to the most important testing methods, including a presentation of testing tools.

Part V – Advanced Topics

Topic 21: *Software metrics* introduces several metric methods: cyclomatic complexity, Halstead, LOC, object-oriented metrics, and presents several CAME tools.

Topic 22: *Maintenance* – Types, requests, costs and planning of maintenance are covered.

Topic 23: *Reverse engineering* discusses notions of software repair, reengineering, and restructuring, including presentation of CARE tools.

Topic 24: *Quality of software development process and its standardization* covers standards such as ISO 9000, and capability assessment models.

Topic 25: *Introduction to software ergonomics* – The most important notions of graphical user interfaces, standards, and guidelines are presented.

Topic 26: *User manuals* – Dedicated to principles and guidelines for writing user manuals.

Topic 27: *Project management* – Planning, organization, people management, and control are the most important notions covered.

Topic 28: *Configuration management* – The topic is dedicated to motivation, explains activities, and discusses CVS.

The second essential component of the course is usage of complex case studies. Currently, there are two case studies in use, 'Seminar organization' and 'XCTL software', while the development of several additional ones is in progress. Case studies are used after the presentation of theoretical topics, and the main idea is to show students the ways of application of theory and skills gained earlier, on a real software product.

- 'Seminar organization' represents a software system (Balzert, 1998) created for management of a company dealing with creation, organization, and presentation of external courses: contacts with clients and companies, communication with lecturers and participants, hotels and travel agencies, and so on. At the moment, this case study is used in 10 topics as an illustrative example.

- The second case study represents the software system 'XCTL', a real, existing system, used at the Department of Physics, Humboldt University, Berlin. This legacy system has been reengineered, refactored, and partially rewritten, in order for some additional features to be added. The case study is used in four topics as an illustration.

The third essential component of the course is team assignments, and in this paper we will mostly concentrate on these. There is a pool of assignments created, which can be used both for illustration of theory and for testing the acquired knowledge. An agreement within the project has been reached that the assignments will be given to *teams* of students.

Depending on the number of students, from 5 to 20 teams were created, numbering from 3 to 5 students. Members of the team are self-chosen which enables easier organization of teamwork, which is also a useful practice elsewhere (e.g., Bielikova and Navrat, 2004). After presentation of an appropriate topic, an assignment is given with a deadline of 2–3 weeks for solving. Teams have to organize meetings, discuss the assignment, distribute obligations, perform work, and submit a written report. Points are then assigned to these solutions and all of the team members receive the same number of points.

A minimum number of points required for a student to qualify for the final exam is settled at 50% of the possible maximum number of points. How those points influence the final grade is not the same: In Germany and Albania students were required to achieve 50% of the points, which qualifies them for the final exam, but does not influence the final grade. In Serbia and in FYR of Macedonia, besides the limit of 50%, the number of points influences the grades.

The Assignments

For the first time, during the school year 2004-05 a complete, absolutely identical course, with the same case studies and the same assignments, was held in Germany (Berlin) and in Serbia (Novi Sad).

In the following years, the Faculty of Natural Sciences and Mathematics of the 'Ss Cyril and Methodious' University of FYR of Macedonia in Skopje adopted the same lecturing style for the software engineering course. Later on, the course was conducted in a slightly different style at the Polytechnic University of Tirana, but with the same general structure. It was conducted as a seven-day crash course, where most of the topics were presented to students, including case studies, while only four assignments were given to students.

A pool of nine assignments has been created. They are:

- Assignment 1: *Review of* '*preliminary requirements specification*' *and* '*requirements specification.*' Case study 'Seminar organization' is used, and both requirements specifications are part of it. Students have to find misunderstandings, discrepancies, and errors and write a report with suggestions on how to solve the problems.

- Assignment 2: *Application of a function-point method*. Again, requirements specification for 'Seminar organization' is used and students have to estimate costs, expressed in human-power, for development of this software.

- Assignment 3: *Review of a product model resulting after structured analysis*. Data-flow diagrams for 'Seminar organization' software are presented, including

some errors in them. Students are required to recognize those errors and suggest correct diagrams.

- Assignment 4: *Development of a part of a static model through creation of a use-case diagram and class diagram*. As students are already familiar with use-case and class diagrams from other courses, we test their creativity on a part of a static model.

- Assignment 5: *Definition of a formal specification for several given operations*. After being introduced to formal specification of several classic operations, students have to develop their solutions for some additional operations.

- Assignment 6: *Review of a solution for the fourth assignment of a different team*. Teams were faced with another teams' solution of assignment 4. Analysis of other teams' solutions exposes the students to a different view on the same problem.

- Assignment 7: *Measuring the quality of given software*. The implementation of 'Seminar organization' system is used as a case study to be measured.

- Assignment 8: *Specification of a regression test*. Students are required to develop a regression test package for a given example program, using the given testing tool.

- Assignment 9: *Creation of a classification tree*. Practicing usage of classification tree method for function-oriented tests. Students are expected to specify a systematic test for a given business process, from the requirements specification for the 'Seminar organization' case study.

In practice, the following procedure is applied: Teams are given specific tasks and are expected to produce results before a given deadline. Each member of a team has to read, think about, and reflect on a task *before* the team meeting. During several team meetings, the team discusses and solves the task. Occasionally, one class is organized where the most interesting and provocative solution is presented by the members of the team submitting it.

For assignments, students are divided into teams, according to their choice. This approach has several advantages. The first is simplicity from the managerial point of view. Second is that the opportunity for a student to sign up for a team of their choice creates a tendency to base the choice on personal relationships. Thus, time needed for adjustments and adaptation of team members is shortened. Thirdly, efficiency of teams created this way tends to be rather high.

The mentioned advantages are inclined to minimize most of the real-life problems that arise. Teams *do* get started without additional effort, extra meetings are organized more easily, and usually there is a natural pressure on team members 'not to disappoint their friends.' Still, there are two disadvantages of this approach. The first one is obvious – there is a risk that team quality can (and usually does) vary significantly. The second

drawback is that every now and then, some of the groups have complaints concerning individual team members. To 'fire' them and cancel their participation in a team is more difficult when team members are friends. Thus, there is a common mistake of some of the members covering for other, non-working members. Hopefully, bad experience within their studies might steer students away from this habit.

We will also mention here that *not all* of the assignments are performed each year, at each university. Besides personal choice of the lecturer, technical elements influence such a decision, for example, assignments 8 and 9 require specific tools, available only in Berlin, while some of the other assignments require additional time so they are not appropriate for usage in the crash course in Tirana.

Another important point is the fact that the 'correct solution' for each of the assignments, presented to students by lecturers, is created *in cooperation*, and based on the *combined practice and knowledge* of lecturers from Berlin and Novi Sad. Those experiences are based on solutions previously submitted by students.

The Assignment Results

In this section, we will present results of the assignments gained at different universities. We will start with the two groups of students from Novi Sad, continue with the results achieved in Tirana, then in Skopje, and finish with the results of Berlin students.

Novi Sad Students

Average results and number of points students gained per assignment give us some interesting insights. Let us first present results for two groups of Novi Sad students, students of the 'Computer Science' study programme, and students of the 'Professor of Geography and Informatics' programme.

Table 1

Assignment Points for Novi Sad Students of Computer Science

Novi Sad	Nr of Students	Average Points Assgn 1	Average Points Assgn 2	Average Points Assgn 3	Average Points Assgn 4	Average Points Assgn 5	Average Points Assgn 6	Average Points Assgn 7	Total Points Assgn
2007	45	81.11%	66.67%	63.78%	73.11%	75.78%	88.61%	68.52%	74.05%
2008	54	73.89%	74.53%	80.38%	79.90%	80.68%	94.32%	95.45%	81.75%
2009	60	81.67%	75.42%	88.00%	75.56%	80.67%		95.00%	81.85%
2010	66	77.73%	75.99%	85.76%	77.42%	78.30%	94.38%	91.67%	82.18%
2011	47	80.20%	69.63%	80.20%	69.33%	50.40%	87.25%	85.50%	73.56%
Average		78.92%	72.45%	79.62%	75.06%	73.17%	91.14%	87.23%	78.68%

Even though the number of students constantly grows, the percentage of gained points for assignments shows a more or less regular behaviour.

Results of assignments for the first year are significantly different than those for the following years. We think that the reasons are twofold: inexperienced lecturers during the first year and inexperienced assistants who checked the solutions for the first time. Results of assignments for the last year (gray-coloured area in the table) are also sometimes radically different. Since the same thing also happened in Skopje, we believe that the reason for this is the fact that during this year, the first generation of students studying according to Bologna principles approached this course. While this alone can influence results, the more important thing in our opinion is the fact that a lot of not-so-good students from previous generations finally made the effort of passing to the final year of their studies, in order to avoid switching to a new curriculum. Namely, the curriculum created in accordance with Bologna principles introduced several new courses, discarding of some of the old courses, changing the way courses are assessed, which was highly undesirable for older students.

The worst results and the lowest number of points are usually gained for assignment number 2 (application of the function-point method). Even though it is quite straightforward – a conclusion that has also been made by students – it seems that the assignment has enough hidden difficulties such that it is a regular practice that not many teams reach the maximum number of points. Assignment number 6 (review of a solution of another team's assignment) has the highest average, but it represents simply the ability of team members to defend their own opinion, so it is not of a high expertise level.

The really best results and the highest number of points are usually gained for assignment number 7 (measuring of the quality of software), denoted in Table 1 by a rectangle. Again, there are two reasons, in our opinion. It is again a straightforward and relatively simple task, where most of the required answers are given by the installed software. The second important reason is that this is the last assignment, given at the end of the semester, when students are aware and experienced in how and what they need to do to solve their task.

The assignment that requires the highest level of 'creative' work, assignment number 4 (creation of use-case and class diagrams), denoted in the table by an eclipse, also has a rather low average. The main reason for this we consider to be the lack of experience with real-life work, no practical abilities and skills, and possession of scholarly knowledge only.

The average total points achieved by students are sufficient for them to approach the second part of the exam, and even more it is close to 80% of possible points, which we consider very good.

Worth noting is also assignment number 5 (definition of formal specification), but only in comparison with the solutions of other groups, so we will discuss this topic in more detail later. Here, we would just note that again, results from last-year students (gray area) are far lower than the results of any of the previous generations.

At the end, we note that the average of the total points for the last generation is again far below most of the others, and far below average. As a reminder, that is the generation consisting of the first enrolment of students studying by the Bologna principles, plus remains of previous generations, finally making it to their last year of study. What lies behind this, in our opinion, are several things. For 'Bologna' students

- there is a much larger group of 'elective courses' for this generation of students, which naturally invited students to skip some of the more difficult ones that are needed for assignment solving;
- there exists a much higher percentage of exam passing within this generation, because of changed methods of knowledge assessment. This in turn allowed a greater number of students to pass to the next study years, but with somewhat less knowledge, and considerably lower grades, on average.

For 'old' students

- they have been struggling with their studies for years, having lower knowledge and grades, passing exams only after several attempts and/or years of trying;
- by the time they reached the final year, a high percentage of them had to find a job to support themselves, and their studies suffered greatly because of that.

Novi Sad Students – Professors Programme

Let us now look into the results of students of the Novi Sad programme 'Professor of Geography and Informatics'.

Table 2

Assignment Points for Students of the Programme Professor of Geography and Informatics

Novi Sad Professors	Nr of Students	Average Points Assgn 1	Average Points Assgn 2	Average Points Assgn 3	Average Points Assgn 4	Average Points Assgn 5	Average Points Assgn 6	Average Points Assgn 7	Total Points Assgn
2007	4	57.50%	50.00%	62.50%	75.00%	65.00%	87.50%	83.33%	67.74%
2008	11	62.00%	54.55%	61.82%	69.05%	72.73%	89.77%	75.00%	68.73%
2009	7	71.43%	44.64%	81.43%	57.14%	52.86%		83.33%	64.29%
2010	15	53.33%	50.83%	72.00%	41.67%	72.70%	59.00%	72.73%	59.30%
2011	12	70.00%	70.88%	73.30%	44.42%	40.00%	75.00%	83.33%	63.02%
Average		62.85%	54.18%	70.21%	57.46%	60.66%	77.82%	79.54%	64.62%

The number of students in this programme is relatively small for making definitive conclusions, so we will just compare these results with those from the previous table.

Results of assignments for the first year are not much different from the results from other years, as they are for the students of the computer science programme. A possible reason for this is that they are both low enough, so there cannot be too much difference.

Again, the worst results are gained for assignment number 2 (function-point method) and assignment number 4 (creation of use-case and class diagrams), the 'difficult one' and the 'creative one.' Similarly, the best results are again gained for assignment number 7 (measuring of the quality of software), 'the final one.'

Contrary to the computer science students, within this group it happened on several occasions that on average, students failed to achieve the necessary 50% of points – for assignment number 2 in school year 2009, for assignment number 4 during school years 2010 and 2011, and for assignment number 5 during school year 2011 (see circled areas in Table 2).

Again, average total points achieved by students were sufficient for them to approach the second part of the exam.

With this group of students, the difference for the last generation of students is *not* that obvious. The reason for this, in our opinion, is the fact that they *do not* all belong to the same generation, since (each year) more than half of these students remain from some of the previous years. This, in turn, may also partly explain why results for all of the assignments for this group are weaker than for the computer science programme.

Tirana Students

The second country where the course has been conducted is Albania. At the Polytechnic University of Tirana, during the school year 2006-2007, a seven-day crash course for the selected students of the 2nd year of master studies was conducted by a professor from Germany and assistant from Serbia, for the first time. In the following years, the same crash course was conducted.

Because of the shortage of time and physical constraints, it has been decided that this group of students will have to solve only four assignments, namely assignments 1 (review of requirements specifications), 2 (function-point method), 5 (definition of algebraic specification), and 7 (measuring of the quality of software). Particulars of tasks were also slightly different – the first assignment they had to solve was before the course started in order to get introduced to the requirements specification. Three other assignments had to be solved after the course was finished, and they were given two weeks per assignment. The results are presented in Table 3.

Table 3

Assignment Points for Tirana Master Students

Tirana	Nr of Students	Average Points Assgn 1	Average Points Assgn 2	Average Points Assgn 5	Average Points Assgn 7	Total Points Assgn
2009	17	78.24%	80.59%	80.00%	98.24%	84.26%
2010	15	69.30%	74.00%	76.70%	95.30%	78.83%
2011	15	66.00%	78.70%	80.70%	92.70%	79.53%
Average		71.18%	77.86%	79.13%	95.41%	80.87%
Novi Sad		77.76%	75.31%	79.88%	94.04%	79.74%

It can be noticed that these results are quite comparable to the results of Novi Sad students of computer science. In the last, gray row of the table, percentages disregarding the first, introductory year and the last, mixed year are given. As expected – *master* students from Tirana gained slightly higher numbers of points per assignment, while, again as expected, students studying in non-mother tongue achieved less than maximal results. As a consequence, master students were not much better than graduate students in total numbers.

Skopje Students

The third country that has been engaged in conducting a common course was FYR of Macedonia. The Faculty of Natural Sciences and Mathematics in Skopje started with the conduction of this course in 2007-08 and has been conducting it since. The data we present here concern two groups of students: students of 3-year studies and students of 4-year studies (according to Bologna principles). The following year besides these two groups another group appeared: students of 4-year studies, studying *not* according to Bologna principles. Here are the results.

Table 4

Assignment Points for Students from Skopje

Skopje		Nr of Students	Average Points Assgn 1	Average Points Assgn 2	Average Points Assgn 3	Average Points Assgn 4	Average Points Assgn 5	Average Points Assgn 6	Total Points Assgn
2010	3-year	46	46.34%	88.46%	97.14%	51.27%			66.90%
	4-year	75	61.44%	96.72%	89.74%	67.94%			76.76%
2011	3-year	51	55.49%	53.33%	73.92%	35.53%	56.74%	34.88%	51.65%
	4-year	118	58.36%	72.89%	74.57%	30.35%	43.01%	12.96%	48.86%
	"old"	80	61.54%	71.63%	83.26%	57.40%	92.21%	82.22%	74.71%

During the first year in Skopje only the first four assignments were used. The decision of the lecturers was that 'formal specifications' have been tested enough within other courses, while they did not manage to lecture on the topic of 'software metrics,' so this assignment could not be used. Instead, they added one more assignment from the field of 'product models,' which we will not consider here. During the second year, having more experience with the course, lecturers decided that the course would have six assignments. The first, obvious thing that could be noticed is the fact that students of the 4-year studies achieved much better results than students of the 3-year studies (except for the 3rd assignment – Review of a product model). This is due to the general observation in Skopje that the students of 4-year studies are more skillful and possess a better background in informatics than students of 3-year studies.

Here again we can notice a big difference between results for the first and the second year of conducting the course. For example, for the first year (Table 4, gray area), an observation can be made about the distribution of points between assignments. Namely, assignments 1 (Review of requirements specifications) and 4 (Development of a part of a static model) in both groups were solved much worse than the other two assignments (Function-point method and Review of a product model). At the end, on average, students of the 4-year studies had around 15% better results than the students of the 3-year studies. For the second school year (Table 4, white area), differences are not that high, which again might have to do with the experience of the assistant. For this school year, of much greater importance is the difference between groups of students. Students of the 3-year and 4-year studies (studying according to Bologna principles) have lower results for each assignment than the 'old' students. In some cases, that difference is extreme – for assignment number 7, they received only 35% and 13% of points, while the 'old' students received 82% of points on average! The difference is significant also for assignments number 5 and 6, and considerable for assignment number 4. This, being rather similar to the situation in Novi Sad, keeps the question about peculiarities and lower results of Bologna students still open.

Berlin Students

How do all of the mentioned results compare to Berlin students? For the course conducted in Berlin, statistics for three years are available (Table 5).

Table 5

Assignment Points for Berlin Students

Berlin	Nr of Students	Average Points Assgn 1	Average Points Assgn 2	Average Points Assgn 3	Average Points Assgn 4	Average Points Assgn 5	Average Points Assgn 7	Total Points Assgn
2007	52	88.57%	78.41%	75.00%	72.27%	65.00%	86.73%	77.14%
2009	85	86.88%	80.63%	86.25%	74.67%	75.63%	78.00%	80.34%
2011	64	87.14%	87.62%	87.62%	87.62%	81.00%	91.90%	87.15%
Average		87.53%	82.22%	82.96%	78.19%	73.88%	85.54%	81.54%
Berlin differences			9.21%	12.62%	15.35%	16.00%	13.90%	13.42%
Novi Sad differences			1.46%	7.62%	4.35%	2.38%	3.79%	3.92%

Except for the first assignment, percentages for Berlin students are quite different between years. As can be seen in the gray area of Table 5, differences for each year are much higher in Berlin than in Novi Sad. The fact that between 2007 and 2009 an assistant has been changed in Berlin probably caused such differences. As it has been already concluded for Novi Sad results, 'inexperience' of an assistant can have influence on grades and number of points awarded to students. This can also be confirmed if we look at the average percentage of points awarded for *all* of the assignments per year. While in Novi Sad, Tirana, and Berlin (during the first two years) the average percentage of points gained in total is around 80–82%, *inexperienced assistants* had quite different results: 74% in Novi Sad during the first year and 87% during the first year for the new assistant in Berlin.

The worst results by far the German students achieved with the 5th assignment (Formal specifications). While both Novi Sad and Tirana students gained around 80% of points for it on average, Berlin students had only 74% for no apparent reason.

Assignment number 7 (Measuring the quality of software) had a similar trend as assignment number 5 when compared with other groups. This assignment, which was the easiest one for Novi Sad and Tirana students, for Berlin students was not that successful, and they gained 'only' 85% on average. Still, this difference might have one quite simple explanation – the assistant for Novi Sad and Tirana students was the same, with the same criteria, while Berlin students had a different assistant. With the change of assistant in Berlin results became closer – Berlin students on the last year gained 92% of points on average.

Results of Tests

In Serbia, Albania, and FYR of Macedonia, the second part of the exam was organized through tests consisting of theoretical questions. The actual structure of the tests varies for each particular country, but the general form is the same. There is a repository of around 400 questions covering the whole curriculum. There were two tests covering the curriculum (in Tirana), or three (in Skopje), or four (in Novi Sad). Still, total numbers of questions and points that can be gained is the same in each case: two tests with 30 questions, three tests with 20 questions, or four tests with 15 questions. In total, 60 points can be gained within tests added to 40 points that could be gained through assignments give the total of 100 points. In Berlin, the second part of the exam is performed orally, so here we will compare only statistics for the rest of the countries.

In Novi Sad, there were four tests organized during the semester, with the exception of the first year. Calculated averages account only for the last three years, since there was a different distribution of questions and topics in tests during the first year. The following are student results (Table 6).

Table 6

Test Points for Novi Sad Students of Computer Science

Novi Sad	Average Points Test 1	Average Points Test 2	Average Points Test 3	Average Points Test 4	Total Points Tests
2007	67.77%	67.70%	77.50%		70.99%
2008	68.07%	66.09%	66.92%	63.95%	66.25%
2009	70.41%	71.35%	67.54%	70.89%	70.05%
2010	68.63%	70.00%	54.01%	53.33%	61.49%
2011	66.07%	58.67%	53.40%	44.47%	55.65%
Average	68.29%	66.53%	60.47%	58.16%	63.36%

Since the 3rd test during school year 2010-11 (Table 6, gray area), we can notice significantly lower results than before.

The explanation is twofold. Throughout the year, current standings for all of the students are known and published, together with the final grade gained up to that moment. More importantly, students are aware that *after* the semester, there is one additional possibility to do the test(s) again. Those who passed some tests, but are not satisfied with their success, must either accept these points, or re-take *all* of the tests. So as the last test approaches, students start calculating their possibilities: If they estimate they will not get as many points as they want, they submit an 'empty' test, fail it, and re-take it later.

The second point is again related to the school year 2010-11, and students representing a mixture of those studying by Bologna principles and those repeating their studies for several years. For 'Bologna' students, there is a high percentage of students arriving to the final year of studies, compared to the number of enrolled students at the beginning, and there are more students with less knowledge, lower grades, and missing (elective) courses needed as pre-knowledge for this course. For the other part of the group, it is obvious that we are dealing with students struggling with their studies, students with much lower knowledge. So, as a result, averages for *each* of the tests are the lowest compared to the previous years. The low-point is the last test of the last year, when – on average – students achieved only 44-47% of points. Even the average number of points, for *all* of the tests, is the lowest for all years – just 55-65%, compared to 62-71% for the previous years.

The results of tests obtained by another group of students from Novi Sad, professors of Geography and Informatics, are as follows (Table 7).

Table 7

Test Points for Novi Sad Students of Geography and Informatics

Novi Sad Professors	Average Points Test 1	Average Points Test 2	Average Points Test 3	Average Points Test 4	Total Points Tests
2007	51.25%	62.50%	66.88%		60.21%
2008	42.00%	57.50%	43.00%	41.33%	45.96%
2009	52.86%	58.89%	53.33%	49.44%	53.63%
2010	59.70%	50.51%	45.15%	44.17%	49.88%
2011	62.93%	44.73%	47.93%	39.20%	48.70%
Average	54.37%	52.91%	47.35%	43.54%	49.54%

Results for this group are significantly worse than for the other group of students from Novi Sad. Much less pre-knowledge in informatics is exhibited here, so the average number of points gained is hardly over 50%, which is the minimum for passing the test. And again, the difference of the results from last year is not too high, and we are of the opinion that it is for the same reason: the fact that a large number of those students in each year is from previous generations, studying by pre-Bologna principles.

Let us now present the results that students from Tirana obtained. They had only two tests and, what is probably more important and influencing the results, tests were performed 'on the distance.' The professor and assistant sent tests to the local organizer, who organized an exam in Tirana. Both tests were performed at the same time, which is yet another difference compared to other institutions, where tests are scheduled throughout the semester. Questions were chosen from the same repository, and two tests of 30 questions were formed. Test results are presented in Table 8.

Table 8

Test Points for Tirana Master Students

Tirana	Nr of Students	Average Points Test 1	Average Points Test 2	Total Points Tests
2009	17	58.33%	50.33%	54.33%
2010	15	64.23%	56.43%	60.33%
2011	15	67.33%	52.00%	59.67%
Average		63.30%	52.92%	58.11%

The number of points gained is much lower than the number of points gained by Novi Sad students of computer science and only slightly higher than points gained by Novi Sad students of the 'Professor of Geography and Informatics' programme. The only reasonable explanation, which was also confirmed through consultations with master students, was the problem of the English language. Usage of non-mother tongue in tests and answers presented the greatest problem to the students, greater than the expert knowledge required. During tests English had to be used on-site, both for understanding the questions and for answering, which created a lot of problems. The additional problem was the fact that the test was performed on the distance. So, ambiguousness within questions, even lingual ones, could not be solved.

In Skopje, the final part of the exam was organized through three tests. As in Novi Sad, students that failed a certain part of the exam, or were unable to take it, had one additional chance to take it. The results of tests obtained by students from Skopje are given in Table 9.

Table 9

Test Points for Students from Skopje

Skopje		Nr of Students	Average Points Test 1	Average Points Test 2	Average Points Test 3	Total Points Tests
2010	3-year	46	45.59%	56.70%	54.85%	52.38%
	4-year	75	59.98%	67.63%	66.60%	64.74%
2011	3-year	51	50.38%	52.20%	56.90%	52.90%
	4-year	118	62.66%	64.07%	61.10%	62.61%
	"old"	80	57.46%	58.19%	57.59%	57.75%
Average			56.83%	58.15%	58.53%	57.75%

Similar to the assignments, students of the 4th year gained significantly more points than students of the 3rd year. The distribution of points between tests is quite regular, the first one being 'the most difficult one' which is the common practice, before students get acquainted with the material.

As far as the percentages are concerned, actual comparison is not easy, because of the different number of tests in each country. Yet, knowing the topics presented, we can try to give a rough comparison: Students of the 4-year studies from Skopje are somewhat below the percentages of students both from Novi Sad and from Tirana. It is not surprising that results of the students of 3-year studies from Skopje are comparable to Novi Sad students of 'Geography & Informatics' programme (Table 10). We already discussed reasons for that in the previous section.

Table 10

Final Test Percentages for all Countries

	Test 1	Test 2	Test 3	Test 4	All tests
Novi Sad CS	68.29%	66.53%	60.47%	58.16%	63.36%
Novi Sad Prof	54.37%	52.91%	47.35%	43.54%	49.54%
Tirana	63.30%		52.92%		58.11%
Skopje III year	47.99%	54.45%	55.88%		52.77%
Skopje IV year	60.03%	63.30%	61.76%		61.70%
Average	58.80%	59.30%	55.68%	50.85%	57.10%

If we just extract the data for comparable groups, the results are more similar, as we think they should be (Table 11).

Table 11

Final Test Percentages for the Final Year students of CS Programme from Different Countries

	Test 1	Test 2	Test 3	Test 4	All tests
Novi Sad CS	68.29%	66.53%	60.47%	58.16%	63.36%
Tirana	63.30%		52.92%		58.11%
Skopje IVy	60.03%	63.30%	61.76%		61.70%
Average	63.87%	64.91%	58.38%	58.16%	61.06%

Discussion

We can note that all of the institutions used different methods for determining the final grade.

In Serbia, the final grade is based on three inputs:

points gained for solving assignments,

points gained at tests, and

bonus points for activity during the course.

In Albania, the final grade was based on two inputs:

points achieved for assignments, and

points achieved at tests.

In FYR of Macedonia, the final grade is based again on three inputs:

points obtained for assignments,

points obtained at tests,

extra points obtained for attendance, activity, or on-time homework delivery.

In Germany, the final grade is based only on the results of the final, oral exam, while points for the assignments present only the necessary prerequisite to approach the oral exam.

In the end, let us present students' average final grades for all courses conducted (Table 12; the scale of the grades is from 6 to 10).

Table 12

Final Grades

Course	Final mark
NS CS 2007	8.24
NS CS 2008	7.94
NS CS 2009	8.24
NS CS 2010	8.29
NS CS 2011	7.63
NS Prof 2007	6.50
NS Prof 2008	6.00
NS Prof 2009	7.25
NS Prof 2010	6.50
NS Prof 2011	7.00
Tirana 2010	8.24
Tirana 2011 I	8.40
Tirana 2011 II	8.20
Skopje 2010 3-year	8.11
Skopje 2010 4-year	8.31
Skopje 2011 3-year	6.56
Skopje 2011 4-year	6.82
Skopje 2011 4-year - old	7.64

In our opinion, final grades represent steady and expected trends. It is not just the expected average, it is also within the normal (Gaussian) distribution for each group. Grades are significantly higher for students of computer science than for the programme 'Professor of Geography and Informatics' and for the 3-year studies.

Conclusions

The general opinion of all project members is that the project was very successful and useful, first of all for students. During project realization we also went a step ahead and developed and intensively used several educational tools within several programming courses. Also, we conducted several questionnaires in order to obtain students' opinion about different educational issues and influences like gender influences on studying, usefulness of wikis for students' practical team work, privacy and usability aspects of using LMS, and so on. The main achievements and results of these activities are:

- Time for course preparation is drastically shortened.

- Students are enabled to learn in accordance with contemporary contents, principles (Klašnja Milićević et al., 2011; Vesin et al., 2013), and European standards.

- Course compatibility, both general and concrete, is achieved.

- An excellent base for usage of distance learning principles is created (Putnik et al., 2013; Ivanović et al., 2013a).

- Experiences, methods, and learning activities and styles of lecturers from several different countries are adopted (Ivanović et al., 2013).

- Possibilities for different kinds of future cooperation among the project participants are promoted and recognized. What is more, based on excellent experiences, and using the same technique, we developed several other courses and introduced them into curricula of a number of universities (Ivanović et al., 2013b).

The central attention in this paper has been devoted to the software engineering course. The same course was conducted in four different countries over multiple years and the same pool of assignments was used. Furthermore, the lecturer in Berlin and Tirana was the same, as well as the assistant in Novi Sad and Tirana. In all countries we had students of computer science and with significant background in informatics, while in two countries we had students with lower background in informatics ('professors' in Novi Sad and 3-year students in Skopje). All this created a good environment to draw some conclusions based on numerical results.

- To be successful (student) in software engineering, one must have a certain background in informatics. This justifies the position of this course at the end of studies, rather than near the beginning.

- Assistants' experience in conducting the course is an important factor – every time the assistant changed, the results showed inconsistency with general patterns and trends in results.

- Although students of 'Professor' and '3-year' studies are getting significantly fewer points than students of computer science, their results are still proportional and follow the general pattern. Furthermore, comparable groups of students from different countries have quite similar results. This leads us to the conclusion that problems and issues of software engineering are universal, and that cultural, historical, and other differences do not influence students' achievements nor should influence teaching methods.

We also have to note that students studying according to Bologna principles exhibited significantly lower achievements. While we tried to explain it in the paper, we are still not ready to draw formal conclusions. Further investigation of this phenomenon is needed.

Acknowledgment

This work was supported by the Ministry of Education and Science of the Republic of Serbia within project no. OI174023: “Intelligent techniques and their integration into wide spectrum decision support”.

References

Ariadne Project homepage. Retrieved from http://www.ariadne-eu.org

Balzert, H. (1998). *Lehrbuch der Software-Technik, Band 1 und 2*. Heidelberg: Spektrum Akademischer Verlag.

Bielikova, M., & Navrat, P. (2004). Experiences with designing a team project module for teaching teamwork to students. *Journal of Computing and Information Technology*, *13*(1), 1–10.

Bothe, K., Schuetzler, K., Budimac, Z., Zdravkova, K., Bojić, D., & Stoyanov, S. (2003). Technical and managerial principles of a distributed cooperative development of a multi-lingual educational course. *1st Balkan Conference in Informatics* (pp. 307-316). Thessaloniki, Greece.

Bothe, K., Schuetzler, K., Budimac, Z., & Zdravkova, K. (2005). Collaborative development of a multi-lingual software engineering course across countries. *35th ASEE/IEEE Frontiers in Education Conference* (pp. T1A-1 – T1A-5). Indianapolis, USA.

Bothe, K., Schützler, K., Budimac, Z., Ivanović, M., Putnik, Z., Stoyanov, S., Stoyanova-Doyceva, A., Zdravkova, K., Jakimovski, B., Bojić, D., Jurca, I., Kalpić, D., & Cico, B. (2009). Experience with shared teaching materials for software engineering across countries. *Informatics Education Europe IV* (pp. 57 - 62). Freiburg, Germany.

Bothe, K., Budimac, Z., Cortazar, R., Ivanović, M., & Zedan, H. (2009a). Development of a modern curriculum in software engineering at master level across countries. *Comput. Sci. Inf. Syst*, *6*(1), 1-21.

Bran, A., Bourque, P., Dupuis, R., & Moore, J. W. (Eds.) (2001). *Guide to the software engineering body of knowledge SWEBOK*. New Jersy: IEEE Computer Science Press.

Budimac, Z., Putnik, Z., Ivanović, M., Bothe, K., & Schuetzler, K. (2008). Conducting a joint course on software engineering based on teamwork of students. *Informatics in Education, An International Journal, Institute of Mathematics and Informatics, Lithuanian Academy of Sciences*, *7*(1), 17-30.

Budimac, Z., Putnik, Z., Ivanović, M., & Bothe, K. (2009). Common software engineering course: Experiences from different countries. *1st International Conference on Computer Supported Education* (pp. 375-378). Lisboa, Portugal.

Budimac, Z., Putnik, Z., Ivanović, M., Bothe, K., & Schuetzler, K. (2011). On the assessment and self-assessment in a students' teamwork based course on

software engineering. *Computer Applications in Engineering Education*, *19*(1) 1–9.

Caplinskas, A., & Vasilecas, O. (2003). Development of modern curriculum in information systems at master level. *CompSysTech '03 Proceedings of the 4th international conference conference on Computer systems and technologies: e-Learning* (pp. 585-590*)*. Sofia, Bulgaria. DOI: 10.1145/973620.973718

Computing Curricula. (2001). *ACM and the Computer Society of the IEEE*. Retrieved from http://www.acm.org/education/curric_vols/cc2001.pdf

Doberkat, E.-E., Kopka, C., & Engels, G. (2004). MuSofT – Multimedia in der Softwaretechnik. *Softwaretechnik-Trends*, *24*(1).

Hilburn, T., Hislop, G., Lutz, M., Mengel, S., & Sebern, M., (2003) Software engineering course materials workshop, *16th CSEET*. Madrid.

Ivanović, M., Putnik, Z., Budimac, Z., Bothe, K., & Zdravkova, K. (2013) Gender influences on studying computer science – Balkan case. *Proceedings of the 6th Balkan Conference in Informatics* (pp. 171-178). Thessaloniki, Greece . DOI: 10.1145/2490257.2490286

Ivanović, M., Putnik, Z., Komlenov, Ž., Welzer, T., Hölbl, M., & Schweighofer, T. (2013a) Usability and privacy aspects of Moodle - students' and teachers' perspective. *Informatica, An International Journal of Computing and Informatics, 37*(3), 221-230.

Ivanović, M., Budimac, Z., Mishev, A., Bothe, K., & Jurca, I. (2013b). Java across different curricula, courses and countries using a common pool of teaching material. *Informatics in Education, International Journal. Institute of Mathematics and Informatics, Lithuanian Academy of Sciences*, *12*(2), 153-179.

Joint SE course homepage. Retrieved from http://www2.informatik.hu-berlin.de/swt/intkoop/jcse/

Klašnja Milićević, A., Vesin, B., Ivanović, M., & Budimac, Z. (2011). Integration of recommendations and adaptive hypermedia into Java tutoring system. *Comput. Sci. Inf. Syst*, *8(*1), 211-224.

Merlot project homepage. Retrieved from http://www.merlot.org

Modesitt, K. (2002). International Software Engineering University Consortium (ISEUC), A Glimpse into the Future of University and Industry Collaboration". *15th CSEET*. Covington, Kentucky, USA.

Putnik, Z., Budimac, Z., Ivanović, M., & Bothe, K. (2013). Analysis of students' behaviour based on participation and results achieved in a wiki-based team assignments. *Proceedings of the 6th Balkan Conference in Informatics* (pp. 179-186). Thessaloniki, Greece. DOI: 10.1145/2490257.2490277

TEMPUS course homepage. Retrieved from http://perun.dmi.rs/msc-se/

Tibaut, A., Rebolj, D., Menzel, K., & Hore, A. (2013). ITC-Euromaster Course Pool for AEC Engineers. *iJET, 8*(2), 36-40.

Vesin, B., Ivanović, M., Klašnja Milićević, A., & Budimac, Z. (2013). Ontology-based architecture with recommendation strategy in java tutoring system. *Comput. Sci. Inf. Syst., 10*(1), 237-261.

Zdravkova, K., Bothe, K., & Budimac, Z. (2003). SETT-Net: A network for software engineering training and teaching. *ITI-Information technology interfaces* (pp. 281 - 286). Cavtat, Croatia.

Athabasca University