Novi Sad J. Math.
Vol. 43, No. 1, 2013, 167-177

TRANSNATIONAL COOPERATION IN HIGHER EDUCATION IN BALKAN COUNTRIES[1]

Zoran Budimac[2], Zoran Putnik[3], Mirjana Ivanović[4] and Klaus Bothe[5]

Abstract. Successful international educational project entitled "Software Engineering Computer Science Education and Research Cooperation" lasts since year 2001, gathering participants from 9 countries and 15 universities. The project is funded by DAAD (German Academic Exchange Service) and exists under support of the "Stability Pact of South-Easter Europe". At first, the project began with the aim of creation of joint course in the field of "Software Engineering". Over the years the cooperation expanded into the development of several additional common courses: "Object-oriented Programming", "Software Project Management", "Compiler Construction", and "Data Structures and Algorithms" being the most developed ones, and already used at several universities. Some other courses are still under development by interested parties from various countries. Over the years, the aims of the project were extended to other educational and research goals, and the most of the educational resources were developed in a web-based form, available for the usage in the eLearning environment.

AMS Mathematics Subject Classification (2010): 97B40, 97Q20

Key words and phrases: Software Engineering, Joint Courses

1. Introduction

The project under the name "Software Engineering Computer Science Education and Research Cooperation" was created in 2001 by a group of lecturers and researchers who still make the core of it, coming from Germany, Serbia, FYR Macedonia, and Bulgaria. Over the years, participants from all other Balkan countries joined the project, namely: Croatia, Romania, Bosnia and Herzegovina, Albania and Montenegro. Originally, the project was intended to last for three years, yet the excellent educational and research results in creation of common educational resources, had a consequence that since 2004

[1]The work is partially supported by the Ministry of Education and Science of the Republic of Serbia, through project no. OI174023: "Intelligent techniques and their integration into wide-spectrum decision support"

[2]Department of Mathematics and Informatics, Faculty of Science, University of Novi Sad, e-mail: zjb@dmi.uns.ac.rs

[3]Department of Mathematics and Informatics, Faculty of Science, University of Novi Sad, e-mail: putnik@dmi.uns.ac.rs

[4]Department of Mathematics and Informatics, Faculty of Science, University of Novi Sad, e-mail: mira@dmi.uns.ac.rs

[5]Institute of Informatics, Humboldt University Berlin, Germany, e-mail: bothe@informatik.hu-berlin.de

the project manages to justify its existence, prove its success, and receives continuation year after year. The project is currently supported until the end of the year 2013, while future continuations are still possible and wishful by all of the participants, including even several newly interested. About successful management of the project aims the DAAD foundation published reports on several occasions, for example in [8] and [22], while the basics about the project can be found on its homepage [19].

In the beginning, the project was created in order to help the development of a joint course in the field of "Software Engineering", yet over the years, the collaboration was extended to creation of several more courses. In addition, the goals of the project were broadened to the development of further supplementary educational resources, such as: complex case-studies, joint team and individual assignments, or pool of exam questions. The majority of newly produced teaching and learning materials is developed in a form appropriate for the use on the Internet, and *is* used within local learning management systems of the participating institutions. For these purposes, even some specialized tools have been developed, or are currently in the phase of development. Examples of two important ones are a translator between languages of the participating countries (which also incorporates tools for often needed adjustments of shapes, pictures, or text clouds at individual slides) and a principal model for the conversion of teaching materials between classic teaching resources and learning objects, suitable for use in distance learning environments. Another valuable goal of the project was at first introduction, and later development of research cooperation in the field of distributed software engineering, widened further to other research fields. This goal is also successfully implemented and conducted over the years, which is proven by a large number of research papers published in journals and conference proceedings over the years.

Some other projects of a similar type and intentions can be found in the literature - for example [1], [15], [16], [20], or [21]. Three of those are also concerned with the field of software engineering, and are also created for educational purposes. Still, the main difference between them and the project presented here lies in the approach to creation of teaching resources. The mentioned projects created an aggregate of relatively independent modules that all of the participants combine according to their needs. Our project managed to create a complete course and entire teaching material as a unity of interconnected resources. The idea was to enable easier reusability of such material, even for those lecturers who do not have software engineering as a main field of teaching and research interest.

In the beginning, the course was founded on presentations used at the Humboldt University, based on [2]. Following the trends and recommendations of the most important expert associations such as [7] and [14], those materials were translated into English, and further refined and developed. This way, it was ensured that the course contains all of the introductory topics from the field, but also a wide spectrum of advanced topics suggested by the ACM and IEEE associations, and other important expert bodies. Later on, perfected teaching resources have been adjusted to the needs of individual participants, and usually translated to local languages.

2. An overview of the project development

Using the first and the most developed cooperation example - the course in "Software Engineering", we will present the methodology and practice of joint work, but also the basic assumptions and goals of the project. Over the years, the mentioned course went through three, often overlapping, phases:

- in the first phase, the existing German slides based on [2] were translated into English, and through the participation of interested members further refined and developed over several years, based on experiences in the course realization at certain institution;
- in the second phase, new (both basic and advanced topics) that were not part of the original selection, were created and continuously innovated;
- in the third phase, final versions of the teaching materials, agreed by all participants, were translated into local languages [5]. For those purposes, a specialized multilingual tool for translation was developed [4].

As elementary learning objectives, participants defined that teaching materials should enable student who successfully attended the course the abilities to work in a team, to develop both analytical and synthetic approach to decision making, to apply acquired knowledge in practice of realistic environment, to renew, expand, and continually build knowledge from the field, and to be able to make adequate decisions during software development process [11].

In order to fulfill all of the above, additional teaching resources were developed for the course, which we will present in more details here.

3. Teaching materials for "Software Engineering" topics

3.1. Presentations/Topics

The first pillar on which the course is based are 28 topics, organized in 5 areas. Each topic is a self-sufficient PowerPoint presentation, enriched with the addition of "lecture notes". Lecture notes are used for exchange of advices between lecturers, contain answers to the questions posed on the slides, enable creation of printed versions used for studying, and the like. The topics are organized as follows:

- **Area 1: Introduction to Software Engineering**
  - 1. What is software engineering?
  - 2. Quality criteria for software products
  - 3. Software process models
  - 4. Basic concepts for the software development documents
- **Area 2: Requirements Engineering**
  - 5. Results of the "analysis and definition" phase

  - 6. Cost estimation
  - 7. Function-oriented view
  - 8. Data-oriented view
  - 9. Rule-oriented view
  - 10. Structured analysis
  - 11. State-oriented view
  - 12. Scenario-oriented view
  - 13. Object-oriented analysis
  - 14. Formal software specification and program verification
- **Area 3: Software Design**
  - 15. Overview of design activities
  - 16. Structured design
  - 17. Object-oriented design
- **Area 4: Implementation and Testing**
  - 18. Implementation
  - 19. Systematic structured testing
  - 20. Functional testing
- **Area 5: Advanced Problems**
  - 21. Software metrics
  - 22. Maintenance
  - 23. Reverse engineering
  - 24. Quality of software development process
  - 25. Introduction to software ergonomics
  - 26. User manuals
  - 27. Project management
  - 28. Configuration and version management

From the given list of topics, each lecturer can select those suitable for the usage, those that fit into curriculum as defined at the given university, and time-frame given to the course in it. As a consequence, the lecturers *do* apply various teaching methodologies. For example:

- Lecturers in Germany and in Bulgaria, use a whole set of 28 topics. The course is conducted in the final, fourth year of studies, and lasts two semesters;
- Lecturers in Novi Sad, Serbia and Skopje, FYR of Macedonia, use about 24 topics, omitting a (different) few that local professors select. The course is held in the final year of different study programs, i.e. in the third and fourth year of studies, lasting one semester;

- In Belgrade, Niš and Kragujevac, Serbia; Zagreb and Rijeka, Croatia; Sarajevo and Banja Luka, Bosnia and Herzegovina; Podgorica, Montenegro; Tirana, Albania, and Timisoara, Romania, the lecturers selected from 5 to 12 topics to present, and combined those into the existing courses of "Software Engineering";

- A special case is the Polytechnic University of Tirana, Albania, where the course is not held regularly during the school-year, but instead in the form of a one-week crash-course, where 18 topics are presented by guest lecturers from Berlin and Novi Sad.

3.2. Case-studies for the "Software Engineering" course

A second important course component connected to both theory lectures and practical exercises is the usage of complex case-studies for illustration of the concepts presented. During the creation of a course, two case-studies were used:

- "Seminar organization" - a software system intended to manage work of a company dealing with organizing seminars and their presentations to clients. This case-study is used as a part of 10 topics to illustrate theoretical concepts presented during lectures;

- "XCTL" - real, existing system to control the work of measuring instruments at the Institute of Physics of the Humboldt University in Berlin, Germany. The system was analyzed by basic methods of reengineering, so the students are introduced to realistic results of measuring of the quality, testing, and reengineering within 4 topics.

Later, some additional case-studies have been developed and used for various reasons, most often as a way to prevent cheating with the assignments that students have to solve, which we will explain bellow.

3.3. Team assignments for the course

The third essential component of the course is the assignments, created and prepared to be solved by teamwork effort of students. Over the course duration, students are required to solve certain number of assignments, and to gain at least 50% of points for their solving, in order to fulfil the exam prerequisites.

With the assignments, the situation is similar as with teaching topics - a pool of assignments is created that exceeds the needed quantity for the realization of the practical part of the exam. So, the lecturer can select those assignments (s)he finds the most appropriate according to presented teaching material; quality and affinities of students; previously attended courses; need to acquire certain software tools (requiring license); available time, etc.

This kind of flexibility enables the usage of subsets of assignments both for the short courses (crash-course/one-week or one-semester course), and for longer, two-semester courses. A larger number of assignments and the fact that they are parameterized enables changes, which decreases the likelihood of cheating and plagiarism. A list of the assignments follows:

- **Assignment 1**: Reading, checking and "repairing" of the requirement specification for a case-study "Seminar organization". Students are supposed to find errors, ambiguities and obscurities, and suggest how those should be fixed.

- **Assignment 2**: Application of the "function point" method to estimate the cost and human resources needed to create a software for the "Seminar organization" case-study.

- **Assignment 3**: Analysis of a product model created as a result of structured analysis. Again, the "Seminar organization" case-study is used, and the students are given data-flow diagrams containing some errors, which they must identify and suggest corrections.

- **Assignment 4**: Development of a part of static model by the creation of a class diagram and use-case diagram. Here, students are given a new, small case-study that also tests their creativity.

- **Assignment 5**: Creation of formal specification for several new operations, based on the specifications presented at theoretical classes.

- **Assignment 6**: Analysis and critique of another teams' solution for the assignment 4, facing students with the different views on a same problem.

- **Assignment 7**: Software quality measurement for a given source code, with a given tool.

- **Assignment 8**: Specification of a regression test, where students are required to specify set of test cases ensuring "branch coverage".

- **Assignment 9**: Creation of a "classification tree" for software testing. Again, using "Seminar organization" case-study and tool for functional testing, the students have to create an appropriate set of test-cases and check the correctness of a program.

In practice, for all assignments the same procedure is conducted: teams get their task and deadline (of at least two weeks) to solve it. Team members are required to read and understand the assignment, think about it and create the idea of a solution *before* the first team meeting. Over several meetings, team members discuss individual views and create a joint solution.

From time to time, but obligatory after the first assignment, a class is organized at which team creating the most interesting and provoking solution, presents it to other teams. Faced with the different solutions, students contemplate, analyze, discuss and evaluate the presented solution.

The trend is that part of students participate less, or do not participate at all in teamwork, while other team members cover for them. This caused some institutions to experiment with the usage of wiki technology in assignment solving with the idea to closely follow *individual* participation of team members, and more precisely determine their personal contribution. This technique also influences the type and style of meetings the team members organize, which

does not necessarily have to be in person anymore, but can also be conducted virtually. Still, this technique has not be conducted long enough, and is still new for definite conclusions, so we will not comment it further here.

The second characteristic problem with the assignments is universal, and has been noted at all participating universities. Namely, it was observed that less ambitious students *do* participate, yet abandon their teams as soon as they reach a minimal required number of points for the assignments. Grading methodology varies between institutions, so this can mean that students are either satisfied with the minimal passing grade, or that they simply achieved the necessary 50% of the points, and are *not* interested to learn further methods and techniques presented. In each case, this results in much larger burden for the students persisting in solving *all* of the assignments. A satisfactory solution for this problem has not been found up to now, even though the problem is researched at several institutions, because higher grades do not reflect adequately the larger workload enforced to students.

Let us mention one more time that not all of the assignments are used each year, at each institution. The decision about which assignments will be used is based both on subjective - selection and ideas of a lecturer - and objective factors. Some of the assignments are based on the usage of tools that require purchase/registration by the faculty, which is not always possible. At some institutions, the course is much shorter, so it is impossible to manage all of them. This fact proves one more time that created the teaching resources are of high quality and great flexibility. Joint experiences of the project members show that they have the ability to satisfy teaching goals even when various methods for knowledge assessment were used.

4. Other courses developed within the project

Based on excellent experiences and successful cooperation with the joint course on "Software Engineering", the project members decided to extend mutual aid to other courses. So far, four courses have been developed, while the work on some other courses is still in progress:

- "Joint Teaching materials on Object-oriented Programming using Java" [18] is a subproject started during 2004. The subproject deals with the development of joint teaching materials (presentations, case-studies, exam and practice assignments, and similar) for several institutions. Since the course already existed at curricula of all participants, effort was dedicated only to educational and research cooperation. The administrative burden of introducing the course into the curricula this time was not needed (as in the case of the "Software Engineering" course).

- "Software Project Management" [23] is another subproject started in 2004. The goal was development of additional materials for this very important subfield of "Software Engineering". The course was mainly developed by the participants from Berlin, Germany and Novi Sad, Serbia and has been conducted in Novi Sad since 2005.

- "Compiler Construction" is the third subproject started in 2004. The course with the similar name and content existed already in Berlin, Germany and Novi Sad and Belgrade, Serbia so the main point was making them compatible and improve them through the exchange of the existing and joint creation of new material.

- "Data Structures and Algorithms" is a subproject started in 2006. and mostly involved participants from Novi Sad, Serbia and Skopje, FYR Macedonia. Since similar course exist at all other universities, other members joined the development occasionally, mostly through consultations.

5. Conclusions

More than ten years of experience in the cooperation, development and use of joint courses and teaching resources can be summarized in several main outcomes [10]:

- All of the courses are created as an entity containing presentations, but also lecture notes for lecturers, assignments, case-studies and other needed materials. Experience shows that each course can be adjusted to various curricula, but also style and needs of lecturers, and that they can be used in various ways, at different universities;

- The courses have been presented at different universities in various ways and quantity of topics, but still in each variant the created resources proved to be extremely useful;

- The exam and assessment materials were also used in different ways, and again it was proven that the assignments and exam questions satisfied all differences;

- The exchange of teaching resources is worth the effort! Time for creation of teaching material is shortened, while the quality and contemporariness of the material is raised, exchange of experiences and teaching and technical knowledge is also enabled;

- The creation of "lecture notes for teachers" enables the usage of the same material and teaching even by professors with less experience in the field in question;

- The existence and usage of joint material enabled exchange of experiences between lecturers, conduction of surveys and improvement of courses through adoption of students ideas and wishes;

- Thongtime experience was published in several research papers [3–6, 9–13, 17, 24].

As the most important goals of the project were the exchange of experiences, raising quality of teaching, and reducing the efforts for creation of new courses; the conclusions given above show that these goals were not only fulfilled, but by

far surpassed. Based on the excellent experiences with the development of the first course in "Software Engineering", the cooperation was extended to creation of additional courses, both used in practice, and continuously improved.

Some of the problems that the project participants encountered over the years were mentioned in this paper, yet none of those were particular and caused by the project itself. Students' cheating with assignment solving, copying solutions from previous years, or not being ambitious, and dropping out on their teams before the end of the semester are *not* a specialty of this project, but unfortunately a common phenomenon. One, and perhaps the only problem particular for the project itself, is worth mentioning as a recommendation for future projects of a similar type, so we will dedicate more time to it here.

Teaching materials that are used in parallel at several universities as a whole, and at more than 10 universities to some extent, will naturally be constantly innovated through research of individual lecturers. While this is one of the major advantages of being a member of a project, this fact carries some problems too. The coordination and configuration management of teaching materials becomes a major issue. All of the participating institutions use local, translated versions of presentations, perhaps accompanied with the "original", English versions. Innovations are usually performed on those local versions. Their collection from participants, translation to English and their incorporation into master version of presentations, in a situation where several lecturers performed various changes with the same presentations, becomes a huge problem and requires high precision and care. Besides, the noticed errors, even those typographical which still arise every now and then (rarely, but with the body of more than 1600 slides, inevitably) should be carefully corrected in each of the local versions, for each language. So far, these corrections were performed based on the enthusiasm of participants and their wish to prolong project successful duration. Still, it is obvious that such activities take a vast amount of time and attention, and it also usually means that they will be prolonged as long as possible. Recommendation that would definitely relax the obligations within the project, but also ensure prompt raise in quality of teaching materials is existence of at least one purposefully employed person, some sort of a technical secretary of a project. This person would be in charge of improvement, innovation, and refinement of teaching resources. Unfortunately, such position is rarely envisioned in the organizational structure of the project that donators predict, so the projects still must rely on eagerness and passion of the participants.

The lecturers gathered within a project have not stopped just on educational elements. Excellent experiences in cooperation with the development of teaching materials were further extended with research cooperation, overcoming the borders of courses they came from. Additionally, at autumn time each year, the project participants gather to exchange ideas and experiences in person, to communicate and consult about the further educational and research cooperation. Each year, these workshops include young assistants and students who are embraced by the project, so that after 11 years, more than ten assistants were employed at various universities.

References

[1] Ariadne project homepage. Available at: http://www.ariadne-eu.org/ (accessed 1 March 2013)

[2] Balzert, H., Lehrbuch der Software-Technik. Spektrum Akademischer Verlag. 1998.

[3] Bothe, K., Schuetzler, K., Budimac, Z., Zdravkova, K., Bojić, D., Stoyanov, S., Technical and Managerial Principles of a Distributed Cooperative Development of a Multi-Lingual Educational Course. In: Proceedings of the 1^{st} Balkan Conference in Informatics BCI'2003, pp. 307-316. 2003.

[4] Bothe, K., Joachim, S., Tool Support for Developing Multi-Lingual Course Materials. Presented at ONLINE EDUCA Berlin, 10^{th} International Conference on Technology Supported Learning & Training, Berlin, Germany. 2004.

[5] Bothe, K., Schuetzler, K., Budimac, Z., Zdravkova, K., Collaborative Development of a Multi-Lingual Software Engineering Course Across Countries. In: Proceedings of 35^{th} ASEE/IEEE Frontiers in Education Conference, pp. T1A-1 - T1A-5. 2005

[6] Bothe, K., Putnik, Z., Cico, B. International Educational Cooperation - One Possible Model. In: Proceedings of the 5^{th} Balkan Conference in Informatics BCI'2012, pp. 76-81. 2012.

[7] Bourque, P. and Dupuis, R., Guide to the Software Engineering Body of Knowledge SWEBOK. IEEE Computer Science Press. 2001.

[8] Bringing Curriculums and Equipment Up To Date. Daad Newsletter, 3, September, 2002.

[9] Budimac, Z., Putnik, Z., Ivanović, M., Bothe, K., Schtzler, K., Conducting a Joint Course on Software Engineering Based on Teamwork of Students. Informatics in Education, Vol. 7, (Issue 1). 2008. 17-30.

[10] Budimac, Z., Putnik, Z., Ivanović, M., Bothe, K., Common Software Engineering Course: Experiences from Different Countries. In: Proceedings of the 1^{st} International Conference on Computer Supported Education, pp. 375-378. 2009.

[11] Budimac, Z., Putnik, Z., Ivanović, M., Bothe, K., Schtzler, K., On the Assessment and Self-assessment in a Students Teamwork Based Course on Software Engineering. Computer Applications in Engineering Education, Vol. 19, (Issue 1). 2011. 1-9.

[12] Budimac, Z., Ivanović, M., Putnik, Z., Bothe, K., Studies in Wonderland - Sharing of Courses, Lectures, Tasks, Assignments, Tests and Pleasure. In: Proceedings of the 22^{nd} EAEEIE Annual Conference, pp. 213-219. 2011.

[13] Budimac, Z., Putnik, Z., Ivanović, M., Bothe, K., A View on a Successful International Educational Project in Software Engineering. e-Informatica Software Engineering Journal, Vol. 6, (Issue 1). 2012. 47-59.

[14] Computing Curricula 2001. ACM and the Computer Society of the IEEE Available at: http://www.acm.org/education/curric_vols/cc2001.pdf (accessed 1 March 2013)

[15] Doberkat, E.E., Kopka, C., Engels, G., MuSofT - Multimedia in der Softwaretechnik. Softwaretechnik-Trends, Vol. 24, (Issue 1). 2004.

[16] Hilburn, T., Hislop, G., Lutz, M., Mengel, S., Sebern, M., Software Engineering Course Materials Workshop. Presented at the 16^{th} CSEET Conference, Madrid, Spain. 2003.

[17] Ivanović, M., Budimac, Z., Putnik, Z., Bothe, K., Short Comparison of Tasks and Achievements of Different Groups of Students with the Common Software Engineering Course. In: Proceedings of the International Conference on Software Engineering Theory and Practice (SETP-09), pp. 84-91. 2009.

[18] Java course homepage. Available at: `http://perun.pmf.uns.ac.rs/java` (accessed 1 March 2013)

[19] JCSE project homepage. Available at: `http://www2.informatik.hu-berlin.de/swt/intkoop/daad/` (accessed 1 March 2013)

[20] Merlot project homepage. Available at: `http://www.merlot.org` (accessed 1 March 2013)

[21] Modesitt, K., International Software Engineering University Consortium (ISEUC), A Glimpse into the Future of University and Industry Collaboration. Presented at the 15^{th} CSEET Conference, Covington, Kentucky, USA. 2002.

[22] Networks for Sustainability. Daad Newsletter, 1, October, 2009. 2-3.

[23] Software Project Management course homepage. Available at: `http://perun.pmf.uns.ac.rs/moodle` (accessed 1 March 2013)

[24] Zdravkova, K., Bothe, K., Budimac, Z., The structure of SETT-Net. In: Proceedings of Eurocon 2003, pp. 126-129. 2003.

Received by the editors February 8, 2013