Experience with shared teaching materials for software engineering across countries

Klaus Bothe, Kay Schützler, Zoran Budimac, Zoran Putnik, Mirjana Ivanovic, Stanimir Stoyanov, Asya Stoyanova-Doyceva, Katerina Zdravkova, Boro Jakimovski, Dragan Bojic, Ioan Jurca, Damir Kalpic, Betim Cico[1]

ABSTRACT

Shared teaching materials are a means to save effort for its development, to transfer methodological and technical knowledge between different university staff, and to exchange experience in practical application. However, does it really pay off considering the diversity of different educational environments and the difficulties of using externally produced materials rather than dedicated individual ones? This paper reports on the extensive experience gained in a multi-country project.

Categories and Subject Descriptors

D.2 [SOFTWARE ENGINEERING]

General Terms

Management, Experimentation

Keywords

International cooperation, shared teaching materials, software engineering

1. INTRODUCTION

Since 2002, teachers from 10 universities in 7 countries - Germany, Serbia, Bulgaria, FYR Macedonia, Croatia, Romania, and Albania – have developed teaching materials in the field of software engineering within the JCSE (Joint Course on Software Engineering) project. The purpose was to jointly develop, discuss, share and reuse these materials to save capacity and to transfer technical and methodological knowledge. The consortium covers groups experienced in software engineering, as well as groups without any previous experience in the field. It was decided that English was the common language for the development of course materials. There were fundamental questions connected with this project:

- Is it profitable to reuse existing teaching materials instead of producing them individually from scratch?
- Is it really possible for a staff, without deeper experience in the field, to establish a new course only on the basis of materials (mainly) developed by other parties?
- Do existing shared teaching materials allow for enough flexibility to satisfy different educational environments?
- Since teaching materials are usually developed in the national language, there was the question of how challenging are teaching materials developed in English?

This paper will focus on the answers to these questions by evaluating the experience gained in the (re-)use of the JCSE teaching materials at 10 universities. To summarise, it turned out that the project has been successful by contributing to a high-quality delivery of software engineering courses at institutions with rather different traditions and educational environments.

In the literature, similar projects can be found in which shared teaching materials have been built: Ariadne [1], MuSoft [7], Merlot [8], ISEUC [9] and Swenet [10]. Some of them are even devoted to the particular field of software engineering [7, 9, 10]. As opposed to our approach, all of these follow a different philosophy: a set of rather independent modules have been produced. Each lecturer can select and combine those modules in a way that is found to be useful. Our philosophy, on the other hand, is to provide a complete course in software engineering consisting of a sequence of dependable modules. This form is easier to reuse, in particular for lecturers not so familiar with the field. In addition, this dependence allows for more coherent materials. For example, the same case study may be used in all of the modules, and the knowledge in the field can be developed step by step with the modules. On the other hand, with our approach, the flexibility of combination is naturally reduced.

The rest of the paper is organised as follows: in the next section, we give an overview of the process of teaching material development. In section 3, the different components of our teaching materials are introduced. After that, in section 4, we describe the different deliveries of the course at 10 universities. Section 5 presents key issues of diversity found in the delivery. Section 6 focuses on the experience, and section 7 provides the summary.

2. DEVELOPING SHARED TEACHING MATERIALS: MANAGEMENT ASPECTS

On the basis of a project supported by DAAD (German Academic Exchange Service), in 2002, a group of 5 universities (Humboldt University Berlin, Novi Sad, Plovdiv, Skopje and Belgrade) started the development of shared teaching materials for software engineering compliant to general recommendations [6]. In 2003, the first stable version was completed. During the next few years, there was a continuous evolution of the materials through improvements, extensions and new topics. From 2004 to 2007, new groups joined the project (Universities of Tirana, Timisoara, Zagreb, Sarajevo, Podgorica, Kragujevac, Rijeka and Polytechnic University Tirana). All of them contributed to the project by diverse activities: producing materials, discussing materials, writing review reports, applying slides in lectures and writing usage reports, translating materials to national languages, and others [3]. Joint development of shared materials was challenging and had to be supported by dedicated measures:

---

[1] Authors addresses at the end of the paper.

- **Slide style guides:** these were necessary to assure a unified appearance of the slides throughout all course topics. For example, the master slide had to be defined, the form of headlines, the kind of colours, etc.
- **Update management:** since all participants were requested to contribute to the materials, a restrictive update procedure had to be defined. A very simple one was adopted: at any particular time, there was only one current slide file for each topic which was on the project server. Only one party was allowed to modify a topic at any one time. The website administrator was the only one who could physically exchange the current version for a newer one.
- **Roles:** in addition to the website administrator, other roles with dedicated rights were introduced: project manager, developer, user and modifier.
- **Review reports:** as special kinds of documents, course materials could be assessed by the well-defined review technique. This process was supported by a review report form which was the basis of the discussion forum.

All in all, these measures were not too sophisticated. However, they were sufficient to guide a distributed development process. Details of the development process can be found in [3].

3. TEACHING MATERIAL COMPONENTS FOR THE SOFTWARE ENGINEERING COURSE

Figure 1 provides an overview of teaching materials for the shared course (cf. JCSE website [12].

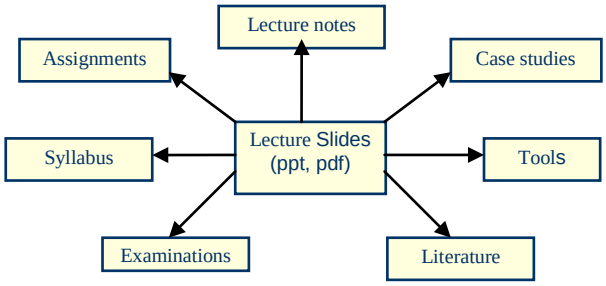

Figure 1. Overview of JCSE teaching materials

Based on the syllabus, the lecture slides constitute the core of the teaching materials by providing the basic technical knowledge. This knowledge is illustrated by case studies: software projects with documents covering all development phases (requirements specification, software architecture, code, test cases, cost estimation, etc.). Lecture notes provide additional information for lecturers, i.e. technical information in addition to the slide content and hints on the methodology.

Assignments are closely connected with the content delivered in lectures and refer to software tools as well as case studies.

The following Tables 1- 4 detail the information in Figure 1.

In Table 1, all the topics (modules) covered in the course are listed. All in all, 28 topics collected in five parts are offered. The general structure follows classical software engineering text-books.

Table 2 exhibits the 8 basic assignment types of the course which refer to main topics and tools used. Details of team-based assignment solutions can be found in [5]. The software tools presented in the lectures and used by the students in their assignments can also be seen in Table 2. These tools are chosen as examples to illustrate the necessity of tool-supported software development. The following fields are supported by tools: OOA (UML), formal specifications, metrics, testing and version control.

Table 1. Topics of the software engineering course.

Part I: Introduction to software engineering 1. What software engineering is 2. Quality criteria for software products 3. Software process models 4. Basic concepts for software development documents	*Part III: Software Design* 15. Overview of design activities 16. Structured design 17. Object-oriented design
Part II: 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	*Part IV: Implementation and testing* 18. Implementation 19. Systematic structured testing 20. Functional testing *Part V: Advanced problems* 21. Software metrics 22. Maintenance 23. Reverse engineering 24. Quality of software development process and its standardisation 25. Introduction to software ergonomics 26. User manuals 27. Project management 28. Configuration and version management

In SE education, case studies belong to the heart of the course. Case studies have to illustrate the delivered knowledge and should be close to real-life ones. In our case, two software projects were selected to be used as case studies (Table 3) throughout the lectures and in assignments.

Table 4 gives an impression of examination questions that can be used for written exams as well as for oral ones.

In our compilation of teaching materials, there are fixed dependencies in the sequence of topics as well as strong connections between the topics and other parts (case studies, assignments and tools). Nevertheless, there is the freedom to leave out parts (topics, assignments, tools or case studies). For example, two default recommendations were provided for a long variant as well as a short variant of the syllabus. Moreover, as is pointed out in sections 4 and 5, there is still much more flexibility for SE teachers.

Table 2. Assignments with reference to tools and course topic.

Type of assignment	Tool support	Refer to topic
1: Review of a requirements specification	-	5
2: Application of the function point method	-	6
3. Review of a structured analysis model	-	10
4: Development of an OOA model	UML tool	13
5: Formal specification of a (sub-) system	Z Tool	14
6: Develop functional test cases with a tool	Functional testing tool	20
7: Develop GUI-oriented automated regression test cases	Regression testing tool	20
8. Derive and interpret metrics	Eclipse plug-in	21

Table 3. Case studies used in the course.

	Seminar Organisation	XCTL
Origin	Textbook	Real customer
Field	Commercial application	Technical case study: experimental physics
Real-world?	Real-world example adapted to a textbook	Real-world example used by physicists
Usage in course	Illustrates all required documents throughout the course: requirements specification, cost estimation, OOA (UML) model, design, test cases, and metrics	Demonstrates treatment of special aspects (unknown software, reverse engineering, requirements specification, and metrics)

4. DELIVERY OF THE COURSE AT DIFFERENT SITES

Meanwhile, the course has been presented at 10 universities in 7 countries (Figure 2) based on our JCSE teaching material repository. It has been presented as a whole, in parts, and in variants. At some universities, the materials have even been applied in different internal variants, e.g. for different audiences

Table 4. Examination questions (some examples).

Collection of 200 examination questions, such as

- What is the subject of software engineering?
- Define what a software metric is.
- Given the following Java method …
  Derive the control flow graph and the cyclomatic complexity.
- Given the following Java method …
  and the test cases …
  What is the statement or branch test coverage for this situation?

or curricula. It turned out that there is a wide range of usage and enough flexibility to satisfy different educational environments.

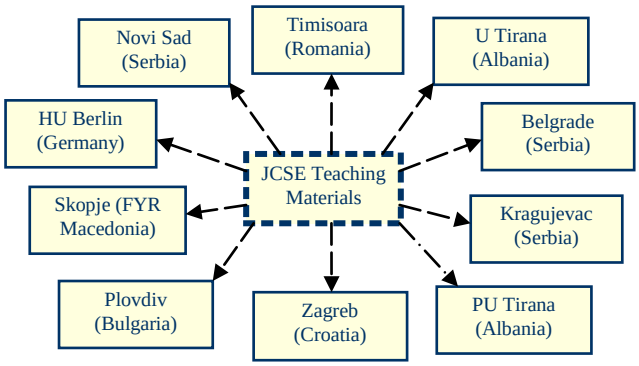

Figure 2. Universities using JCSE teaching materials.

Software engineering as an educational discipline was introduced at several universities only after our teaching materials were produced: Novi Sad, Skopje, Plovdiv, Tirana, Belgrade and Kragujevac.

The following is a brief summary of the delivery of courses based on the shared materials.

Humboldt University Berlin: an early variant of the course was introduced in 1996. Since 2002, the developed materials have been used for about 80 students per year. The complete materials are presented, i.e. all topics, assignments, tools and case studies. Exams are oral ones.

University of Novi Sad: this university was the first one of the South Eastern European partners that has applied the materials since 2002. From 2002 – 2004, the course was offered to a special audience (e.g. from industry for vocational training). Since 2004, software engineering has been introduced as a compulsory course in Bachelor studies [4]. Slides are used in English, and the oral delivery is given in Serbian. Most of the topics and assignments are presented. Only two of the five tools are used. Exams are in a written form using our project material.

University 'Paisii Hilendarski' Plovdiv: this group has presented all the course topics since 2003, basically in Bachelor studies and as induction modules for Master studies. About 100 students take part each year. Plovdiv decided to translate the English slides into Bulgarian. The assignments are not taken from the project. Instead of this, emphasis is placed on OO in UML and on a UML tool. Exams are in a written form using their own (private) questions. The group derived a Bulgarian textbook from the JCSE materials.

University 'Sts. Cyril and Methodius' Skopje: our English materials were selectively used in two existing general CS courses between 2004 and 2006. In 2007, there was the first official software engineering course where the JCSE materials were used selectively (only 15 out of 25 topics, and 3 of 8 assignments). During the next years, the contents has been extended to 21 topics and 6 assignments, and a new own case study was introduced.

University of Belgrade: at this faculty, software engineering was introduced in 2005. 12 topics were taken from the JCSE material and combined with their own (local) materials. Slides are used in Serbian translation. About 130 students take part every year.

University of Kragujevac: from an initial version of our JCSE, half of the modules were taken and translated into Serbian. After that, the Kragujevac group did not contribute any more to the project.

Polytechnic University Tirana: the delivery of the course at this university is unique in our project. The JCSE was delivered as an intensive course within 6 days by guest lecturers from Berlin and Novi Sad. From the Master studies "Computer Engineering", 17 students took part in 2007 and 32 students in 2008. Both the slides and presentation were in English. Afterwards, in the weeks after the lectures, 4 assignments were given, solved and submitted via email to the lecturers. Even the exams were assessed in a distance education mode: written in Tirana, and corrected in Berlin/Novi Sad. In this way, it was the first time that software engineering has been taught at a university in Albania. The goal is that the local staff takes over the course step by step. To that end, at the second delivery in 2008, two assistants from Tirana presented selected topics.

University Tirana: for a special curriculum ("Informatics in Economics"), 12 selected topics have been presented by local teachers without any external support, only by using the materials from the project website.

University Zagreb, Technical University Timisoara: only some of the selected JCSE topics have been presented as part of other established CS courses.

5. ISSUES IN DIVERSITY OF DELIVERY

In the last section, we saw that the original JCSE materials have been applied in nearly the original produced form as well as in several variants. The problems and issues connected with the diversity in the delivery of the course, which was originally intended to be presented in a fixed form, should be considered.

a) Extent of presented topics

The number of topics included in the local course was dependent on rather different factors: available time (credits), preferences of lecturers, and other related courses. To be able to cancel particular topics and select only special ones for presentation, it was necessary to document dependencies between different topics (which one is a prerequisite of the actual one).

b) Language of teaching materials

Several of the participating groups took the English materials in their original English form (Novi Sad, Skopje, Tirana, Berlin, Zagreb and Timisoara); others translated the slides to their national languages (Plovdiv, Kragujevac and Belgrade). The decision was made based on the ability of the students as well as of the staff.

The main disadvantage of a translation was a reduced version management capability: the actual translated native language materials were based on a special English version. However, the shared English version has been continuously improved and extended. Consequently, it was not an easy task to find out differences between the current native language version and the improved English version. In many cases, for that reason, the development of national language materials was frozen at the state of translation, i.e. there was no further evolution.

c) Differences in the audience

The course materials have been presented to quite different audiences: Bachelor students, Master students, and staff from industry. Studies with different directions (computer science, engineering, mathematics) have been covered. Depending on that, it was necessary and possible to leave out certain parts of the course. For example, "Formal Specification" has been dropped in some cases (staff from industry and certain kinds of Bachelors).

d) Tools

In the basic version at Humboldt University Berlin, five kinds of tools were selected for presentation in the lectures and for assignments: UML tool, metrics tool, functional testing tool, GUI-oriented regression testing tool, and Z tool. Most of the other groups took some UML tools (Rational Rose, ObjectiF, ARGE, etc.) because of the importance of OOA/OOD. Only a few groups (Berlin, Novi Sad and Tirana) introduced a metrics tool. A testing tool was applied only in Berlin. The reduced usage of tools seems to be a serious drawback since modem software development without tool-support cannot be successful. There are two main reasons for not using tools so intensively at partner universities: it takes staff capacity (time) to become familiar with the tool, and tools might not be free of charge even for universities.

e) Assignments

Eight types of JCSE assignments could be taken. Some groups excluded assignments and others chose their own. Several groups decided to take the assignments originally provided. In that case, it was possible to discuss – across the groups – the kind of assignment and the solution [5]. The most important reason to drop a suggested assignment was the non-availability of tools, i.e. testing tools. Another one was the decision to concentrate on OO development (Plovdiv) with UML in the practical assignments only. To cope with assignments in software engineering and their solutions requires a lot of experienced staff, for example, to assess a textual review report of a given requirements specification is not an easy task for the novice educator.

f) Examinations

A pool of examination questions, which is mainly dedicated to written exams, is available in the JCSE. These questions have been reused at only three universities (Novi Sad, Tirana and Skopje). The reasons are many: other personal preferences of lecturers, exams in oral form with a different style of questions (more in a form of a discussion), and inclusion of the solutions

to assignments in oral exams (e.g. “Explain your solution to assignment 3”).

g) Familiarity of the staff with the subject of software engineering

Of course, the easiest way to reuse the JCSE materials was possible using local staff with software engineering background. However, if only staff with less SE experience was available, the delivery was possible using lecture notes which were an integral part of the materials [13]. If there was no staff available at all for such a subject, there was still the possibility of guest-lecturing which worked successfully, accompanied by “distance education” for assignments and exams.

h) Other issues of diversity

Additional freedom or differences between the groups concerned the following points:

- Course staff: is there an assistant available who takes care of the assignments?
- Team work in assignments or not
- Both written and oral examinations are possible
- Additional case studies in lectures and/or assignments (used in Plovdiv)
- Mode of final grading: might be a combination of final examination, course work (assignments), and additional tests
- Own topics added: e.g. RUP (Belgrade), Extreme Programming (Novi Sad), i.e. the course is open to new subjects
- Publishing slides for students before/after the lectures
- Delivery of handouts before the lectures (most imported slides with detailed information have to be available in the lecture)
- Value of the course (how many credits, e.g. in ECTS).

6. EXPERIENCE & GENERALIZATION

After five years of reusing JCSE teaching material at different sites, the following experiences could be recorded:

- Sharing of teaching materials pays off: it saves capacity for the production of teaching materials and it allows for the transfer of technical and educational knowledge.
- Additional efforts are necessary to also allow lecturers with less background in the field to apply teaching materials properly and competently: lecture notes with supplementary information on the subject and on teaching tips to the slides was attached – information which is normally in the mind of the experienced lecturer. In this way, we have several examples where a new course on software engineering was introduced with less experience in the field only by using the JCSE materials [13].
- At the beginning of the project, there was agreement on the philosophy to establish a “complete course”. This is different from other projects such as Swenet [9] and Musoft [7], in which a set of (rather) independent modules were provided. In our case, this complete course made it manageable for teachers who were new to that field to take the materials as they were and deliver it (Skopje, Plovdiv and U Tirana).
- Teaching materials do not only have to concentrate on lectures, but have to cover materials for assignments, examinations, case studies, tools, and literature as well. The whole course would have been incomplete otherwise and unmanageable for the novice staff.
- Although a whole course in which teaching material components (single topics, case studies, assignments, etc.) depend on each other was prepared, software engineering courses have been introduced in a couple of variants. There are groups which strictly followed (nearly) the complete original material (Berlin, Novi Sad, Plovdiv, Skopje and PU Tirana) and others that reduced the topics considerably (Belgrade, Kragujevac, University Tirana, Zagreb and Timisoara). To reduce the material, i.e. drop topics in the sequence of dependent topics, it was necessary to document connections between the components. Reduced variants were necessary for different reasons: available time for the course was different (different credits), pre-knowledge of students was limited (e.g. for formal specifications, and pre-knowledge of logic and mathematics was necessary).
- While the substantial knowledge presented in lectures was rather similar at different sites, there were big differences between the groups in the delivery of assignments, usage of software tools, and in the examinations. From the pool of prepared assignments, several groups took the basic ideas (Berlin, Novi Sad, Skopje and Tirana). Others concentrated on their own style of assignments (Belgrade and Plovdiv). Fundamental differences were connected with the use of software tools because of the difficulty for the staff.
- Having the same materials, it is possible and desirable to discuss their content and application methodology. To that end, review reports and usage reports have been introduced and entered onto the project website as a means to save the individual experiences.
- According to the level of knowledge of the English language, several groups decided to translate the materials into their national languages. Although English is generally quite well understood since it is the technical language in informatics, the use of English in everyday lecturing is challenging and demanding. The use of a national language allows for more subtle teaching and discussion with students. However, materials which exist in the original English variant as well as in several national languages complicate maintenance and version control dramatically. Thus, when there was translation to the national language, the development life at that site very often stopped and did not include further improvements of the English baseline materials. Other points connected with the translation are its technical quality and the effort. To support the process of technical translation, a tool was implemented and used in our project [2]. By means of that tool, the translation process was supported by an internal technical multilingual dictionary which guarantied the technical correctness of the translation. This tool, however, is not in everyday usage. Its benefit is the standardization of translating technical terms, however, translating by hand is simply faster.

7. CONCLUSIONS

The primary purpose of the project was to disseminate experience and to reduce effort in creating a new course on

software engineering. To that end, shared teaching materials have been developed covering all parts (lectures, assignments, case studies, examinations and tools). The materials have been maintained and have evolved over the years.

This paper focused on the experience in using these materials at 10 universities. At several sites, the developed materials made it possible to introduce software engineering as a new teaching discipline. The course has been delivered as it is, i.e. as a complete package of topics with associated materials (slides, assignments, case studies, etc.) as well as in diverse variations. The pool of material allows for enough flexibility in that process. Thus, the course exists in several variants: different in the extent of presented topics, in assignments, used tools, examinations, experience of teachers in that field, kinds of delivery (local teachers / guest lecturers in international courses), basic curriculum, and others.

The influence of the shared SE teaching materials on the development of SE as a teaching discipline at several universities in South Eastern Europe cannot be overestimated.

8. ACKNOWLEDGMENTS

We acknowledge the support of DAAD (German Academic Exchange Service), under the auspices of the Stability Pact for South Eastern Europe, through which the work reported here has been funded.

9. REFERENCES

[1] Ariadne Project, http://www.ariadne-eu.org

[2] Bothe K. and Joachim S. 2005. Interactive Tool-based Production of Multilingual Teaching and Learning Materials. In Proceedings of the 5th IEEE International Conference on Advanced Learning Techniques, ICALT (Kaohsiung, Taiwan, 2005).

[3] Bothe, K., Schützler, K., Budimac, Z., Zdravkova, K. 2005. Collaborative Development of a Joint Web-Based Software Engineering Course across Countries, In Proceedings of the 35th ASEE/IEEE Frontiers in Education Conference (Indianapolis, IN, October 19 – 22, 2005).

[4] Budimac, Z., Putnik, Z. 2006. A short report on yet another run of the JCSE in Novi Sad, 6th Workshop "Software Engineering Education and Reverse Engineering" (Ravda, Bulgaria, September 18 - 23, 2006).

[5] Budimac, Z., Putnik, Z., Ivanovic, M., Bothe, K., Schützler, K. 2008. Conducting a Joint Course on Software Engineering Based on Teamwork of Students. Informatics in Education, vol. 7, No. 1, 2008.

[6] CC2001, "Computing Curricula 2001", ACM and the Computer Society of the IEEE, http://www.acm.org

[7] Doberkat, E.-E., Kopka, C., Engels, G. 2004. MuSofT – Multimedia in der Softwaretechnik. In: Softwaretechnik-Trends, Volume 24, Issue 1, 2004

[8] Merlot project, http://www.merlot.org

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

[10] Hilburn, T., Hislop, G., Lutz, M., Mengel, S., Sebern, M., Software Engineering Course Materials Workshop, 16th CSEET (Madrid, 2003).

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

[12] JCSE course website, http://www2.informatik.hu-berlin.de/swt/intkoop/jcse/

[13] Zdravkova, K. Experience with the delivery of JCSE in Skopje in 2007/08, 8th Workshop "Software Engineering Education and Reverse Engineering" (Durres, Albania, September 8 - 13, 2008).

AUTHORS ADDRESSES

Klaus Bothe, Kay Schützler
Humboldt University Berlin
Institute of Computer Science
Berlin, Germany
{bothe, schuetzl}@informatik.hu-berlin.de

Zoran Budimac, Zoran Putnik, Mirjana Ivanovic
University of Novi Sad
Institute of Mathematics and Informatics
Novi Sad, Serbia
{zjb,putnik,mira}@dmi.uns.ac.rs

Stanimir Stoyanov, Asya Stoyanova-Doyceva
University "Paisii Hilendarski"
Institute of Informatics
{S.Stojanov, a.stojanova}@isy-dc.com

Katerina Zdravkova, Boro Jakimovski
University "Sts. Cyril and Methodius"
Institute of Informatics
Skopje, FYR Macedonia
{keti,boro}@ii.edu.mk

Dragan Bojic
University of Belgrade
Faculty of Electrical Engineering
Belgrade, Serbia, bojic@etf.bg.ac.yu

Ioan Jurca
Technical University Timisoara
Institute of Informatics
Timisoara, Romania
ionel@cs.utt.ro

Damir Kalpic
University of Zagreb
Faculty of Electrical Engineering and Computing
Zagreb, Croatia
damir.kalpic@fer.hr

Betim Cico
University Tirana
Faculty of Electrical Engineering
Tirana, Albania,
bcico@abcom-al.com