An Active Learning Approach to Software Testing with Online Support

Alexandra Martinez
Department of Computer and Information Science
Universidad de Costa Rica, Costa Rica
alexandra.martinez@ecci.ucr.ac.cr

Marcelo Jenkins
Department of Computer and Information Science
Universidad de Costa Rica, Costa Rica
marcelo.jenkins@ecci.ucr.ac.cr

Abstract. We have taught software engineering-related courses at undergraduate and graduate levels at the Universidad de Costa Rica for several years, but next year we will face the challenge of teaching a graduate-level course in Software Testing. This will be the third time the course is offered in the master's program but it will be the first time we will teach it, and this time we want to incorporate active learning into the course, using an online support platform. In this paper we will present some of the active learning activities we plan to incorporate into the course and provide support for them based on previous experience using similar activities in other courses. We conclude with our contributions and future directions.

1. INTRODUCTION

Testing is a critical activity in software engineering. It has been estimated that testing accounts for 30%-90% of the total labor expended in producing a program [30]. Yet software testing remains an under-estimated activity in the Computer Science training curricula [35].

According to Kaner and others [35, 20], the purpose of software testing is to find errors (or problems) in the software, which in turns improves the quality of the final product because [most of] the errors get fixed [20] before the software is released to its final users. Others [40, 30] argue that testing that do not reveal errors is also useful, and therefore consider software testing essentially equivalent to verification, whose purpose is to determine whether the implementation is consistent with the specification [31].

Numerous books have been written on the subject of software testing [29, 20, 19, 31] but little has been published on the teaching of software testing [11, 22, 21, 24, 35]. Interestingly, we found that a common element in these works was the use web-based teaching materials [11, 21] or online collaborative platforms [24, 35]. Another interesting finding was a shift from traditional teaching styles towards student-centered approaches [22, 21, 24] that foster the skills a good tester must have: creative and critical thinking [19, 29]. Our contribution is to combine these ideas in the design of a pedagogical approach based on active learning with online support for a graduate course in software testing at our university.

McConnell [26, pp. 37] defines active learning as "(…) *the use of techniques that will get students involved in the learning process beyond what they get from passively listening to a lecture*." Students can actively learn by reading, writing, discussing, solving a problem, or responding to challenging questions [25, 26]. Active learning has been shown to improve student comprehension and retention of material [6], as well as to increase student motivation and higher order thinking [26, 2]. In computer science particularly, active learning techniques have been successfully used in a variety of courses ranging from 'Introduction to Computer Programming' to 'Social Issues in Computing', but mostly at the undergraduate level [25, 32, 6, 12, 37, 8, 23, 38, 16, 18, 17, 10, 15, 5]. We believe that the active learning approach can be extended to graduate level education and for this purpose we designed an active learning approach to teach software testing in a graduate course at our university. Our teaching approach requires support from online tools like *Moodle* [1, 36] and

Bug Hunt [13], as well as from other software development and testing tools like *Visual Studio* 2010 *Ultimate* [28] and *Team Foundation Server* 2010 [27]. The key principles underlying our approach are cooperative learning, peer assessment, and self reflection and assessment.

The rest of the paper is organized as follows. Section 2 describes the context of our course. Section 3 presents our active learning approach with online support. Section 4 summarizes our previous experience using similar activities in other courses. Section 5 presents our concluding remarks and outlines the future work.

2. THE COURSE CONTEXT

The Software Testing course (PF-3866) is part of a set of software engineering courses offered in the Master of Science program in Computer and Information Science at Universidad de Costa Rica (Figure 1 shows these courses). This paper is particularly concerned with the Software Testing course, which has been offered twice since 2009 and will be taught in Spring 2011 using an active learning approach with online support. This is a 4-credit-hour course with 64 hours of class time in a 16-week semester. The class meets once a week for 4 hours, with 10-minute breaks every hour. The course also has a corequisite 2-credit-hour lab course (PF-3867), where students put theory into practice typically through a large project with several deliverables.

The Students. All of our master's students work fulltime in different industries, from small software organizations to large IT Departments of non-IT companies such as banks and government agencies. Most of them are programmers, software developers, and project managers. We also get students with other job profiles such as customer service and support desk professionals. Because this is the only course we offer on the subject of software testing, and because it is until recently that local software industry started to open QA and Tester positions, we do not expect our students to have a sound background in software testing, not even to be practitioners. However, we do expect them to have some background and experience in software development, which will be essential to tackle on the course project. We also expect our students to be responsible for their learning, to be self-motivated, and to be self-critical about their learning progress, which are all important characteristics of the student's role within the active learning approach.

The Teacher. The teacher's role is to facilitate, guide, clarify, and integrate students' ideas and work. We expect the teacher to motivate students to learn and promote collaboration between peers and inside each team. The teacher should also assess the work made by the teams as well as the individual learning progress of each student in the class.

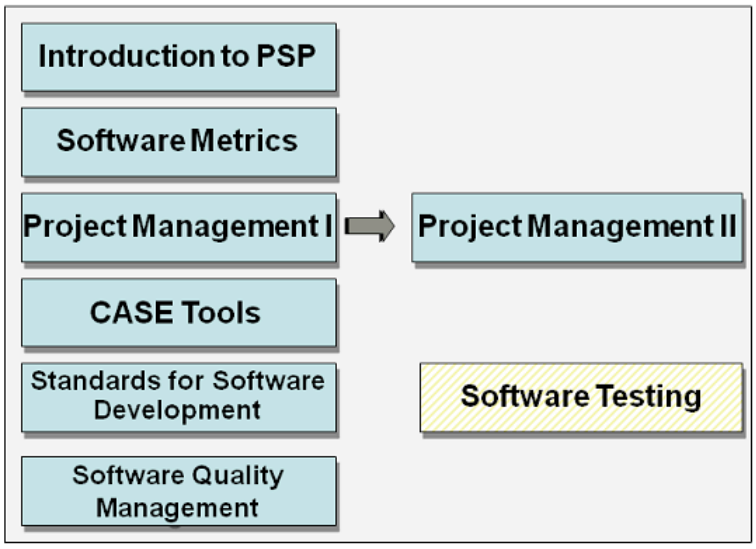

Figure 1. The context of our course in the MS program in Computer and Information Sciences.

3. THE PROPOSED APPROACH

3.1. Active Learning Activities

We propose to involve students in a set of active learning activities throughout the course, particularly, students will work on:

1. A **learning portfolio** where students document their learning and self-reflect about it. This portfolio will be in the form of a wiki page, and has to be updated frequently (at least once a week). It will contain short self-reflections on the learning taking place, commentaries on selected readings (for the session they will facilitate), and the results of the research conducted on a specialized topic in software testing. The use of (experiential) learning portfolios to document, display and analyze student learning process and outcomes has been studied in [9] and [7]. Leveraging wikis to develop learning portfolios (also called personal learning diaries) in computer science courses has been described in [38, 16]. Although this will be our first experience using portfolios in a course, we foresee it will be very valuable as a way of "making learning visible" [33] to the student, the teacher, and the peers.

2. A **team project** that enables cooperative learning. The use of cooperative learning in computer science has been addressed by [10, 17], and we too have used project-based approaches in the past. Depending on the amount of students registered in the course, team size can range from 2 to 4 people. The project will be part of the corequisite lab course, but will be tightly coupled with the main course. The project entails a single software product (e.g., a web-based application with a database component) that will be developed and tested jointly by all students, making use of modularization to delineate each team's work. The idea behind this whole-class project is to involve students in a highly realistic experience of software development in the industry, normally carried out in several phases (with milestones) by large teams working on the same code base. The tools we will use for this project are *Visual Studio 2010 Ultimate* [28] and *Team Foundation Server* 2010 [27], which together provide full support for the testing processes that we will study and apply. Through this project, students will engage in the different phases of the testing process, and will be able to apply commonly used testing techniques (black box testing, white box testing,

automated UI testing, load and performance testing, etc.) to a software product that is under development.

3. **Pre-class readings** and online assessment. For most classes, there will be reading assignments and students will be required to complete those readings prior to the class. The purpose of this type of pre-instruction is to reduce the amount of time needed for lecturing (passive learning) in class, while increasing the amount of time devoted to discussing, writing, solving and thinking (active learning). In order to ensure that students are actually doing the readings and come prepared to class, we will ask students to complete short online assignments prior to class. These assignments will resemble the "warmups" in Just-in-Time Teaching (JiTT) [39], having a few conceptual questions plus a section where students describe the aspects of the reading they found most unclear or difficult. Examples of pre-class instruction and online quizzes to assess pre-class readings can be found in [8, 26] and [39] presents a combination of JiTT exercises with Peer Instruction. We have experience using online assignments to assess pre-class readings in other courses, and the results from a student survey indicate that there are opportunities for improvement in this area.

4. **Facilitating discussion and activities in class**. Students will pair to serve as facilitators for one class session, which will be chosen early in the semester. The facilitators have two tasks to do: to post their commentaries of the assigned readings in the online portfolio, and to prepare and coordinate group activity and discussion for the first hour of class. This idea of students being facilitators was borrowed from Resnick's MAS 714 course at MIT Media Lab [34]. In the past, we have asked students to conduct a short investigation, in pairs, on a topic of their interest that is related to the subject of course, and we have allocated typically two class periods (about 30 minutes per pair) for the students to present their work. Although this activity is not the same as the one we proposed, it is similar in the sense that students have to study, understand, and present (teach) to the class some material that is relevant to the class context.

5. **Peer-reviewing** their classmates' work. This will be done in several ways and for different activities. On the project, students will constantly give and receive informal feedback to/from its teammates regarding their progress; and they will formally provide feedback to the teacher on their teammates learning at specific project milestones [15]. On the reading's commentaries, which are available on each student's online portfolio, peers can voluntarily leave comments and provide feedback to the authors, but formally some peers (selected by the teacher) will asses each commentary. On the facilitator role, students will assess their peers (those facilitating the class session) based on creativity and understanding of the material. On the research, students will rate their peers work based on content and presentation, and the two best research projects will get some bonus points. We have used peer-reviews to evaluate oral presentations and projects in other courses.

6. **Online tutorials**. Students will complete all lessons of the *Bug Hunt* online tutorial [13], for which access has already been granted by the University of Nebraska-Lincoln. *Bug Hunt* is hands-on web-based tutorial designed to engage students in the learning of software testing strategies [14]. It is still not yet clear whether these online tutorials will be assigned as homework or will be worked out in class, but we know we will use them in the course because they are a valuable learning tool and one of the best interactive tutorials we know of.

3.2. Online Support

Many of the learning activities mentioned above certainly require online support. The main online platform we will use is *Moodle* [1, 36] because it has been officially adopted by our department since some years ago, our students are familiar with it, and we know it can support all the online activities we plan to do, namely wiki creation, online assignment posting and submission (which can be automatically graded, if needed), linking to other online resources, and course content management.

Another online tool we will use is *Bug Hunt* [13], an online tutorial to instill good testing practices. One of the advantages of using *Bug Hunt* is that it continuously provides students with feedback (assessment) on the work they are doing, as opposed to traditional assignments that get graded only after all students have turned in their homework, resulting in delayed feedback. Moreover, using *Bug Hunt* in-class (in a lab session, for example) has an additional benefit: it allows students to know how they are doing relative to the rest of the class, i.e., it compares student's performance to the average class performance. We believe this can be exploited as a motivational factor, maybe in the form of a class contest. There has been some research in the area of online teaching [3, 4], but we believe this topic deserves further study.

The other tools we will need for the team project are *Visual Studio* 2010 *Ultimate* [28] (client side) and *Team Foundation Server* 2010 [27]. Together, these tools provide support for the testing processes we will study and apply in the course. The Team Foundation Server offers a version control repository for code (including testing code) and other artifacts (e.g., documents), which will be used to teach students about the configuration management process, and to help organize the teams work (it allows multiple students to be working on the same code at the same time). The *Visual Studio IDE* and the *Microsoft Test Manager* (part of the Ultimate edition) both offer an integrated view of Test Cases (both manual and automated), Test Plans, and Test Runs, as well as the possibility to link Bugs to Test Cases and Test Cases to User Stories (or requirements), facilitating traceability. Students will be able to learn about the management processes of user stories, test cases and bugs. In addition, both the teacher and the students can manage and monitor each team's work by means of Work Items assigned to individual team members. The *Visual Studio* 2010 tool will also enable students to do automated UI testing, load and performance testing, among others. One challenge we are still working on is to be able to provide our students with secure remote access to the Team Foundation Server. Since most of our master's students work fulltime outside the university campus, commuting to campus daily just to work for a couple of hours on the project is not convenient for many of them, but having remote access to the server will allow them to work from home and will be very useful.

We also plan to complement the readings with online resources like videos or presentations related to the subject of the readings. Those resources will be linked from the *Moodle* platform.

4. FINDINGS FROM PREVIOUS EXPERIENCE

We surveyed students from two different groups of the Databases I (CI-1312) course, taught by the first author of this paper during the fall semester of 2010. CI-1312 is a mandatory course in the 3rd year of the Bachelor of Science's program at our Department. During the fall semester of 2010, enrolment in group 1 was 16 and enrolment in group 2 was 16, for a total of 32 students, out of which 28 completed the survey. Participation in the survey was totally voluntary and anonymous. The survey was conducted online from December 2nd to December 9th, 2010, and it was created using the *LimeSurvey* tool. The survey had a total of 35 questions organized in 10 sections (pages). Not all

the survey questions are relevant to the purpose of this paper, so we will present here the results of eleven questions that provide support for some of the activities and ideas we have proposed.

About pre-class readings. Three questions were posed regarding pre-class readings and the way they were assessed. The first of those questions was "Do you deem the experience of developing a team project useful for your professional training?" All answers were positive. The second question was "When you individually read the material before class and come to class with a basic knowledge on the subject, does your learning improve, not change, or worsen compared to when you do not read the material in advance and are faced with it for the first time in class?" All students unanimously answer that their learning improved. The third question was "Do you think that the online reading reports are an effective way to verify that a student did the assigned reading?"Here, 62% of the students responded 'Yes' and 38% responded 'No'. The major criticism students had was that the questions were generally easy to answer straight from the reading, so students would typically just look for the specific answer without doing a thorough reading of the material. Based on this feedback, we know we need to improve the type of questions we ask about the readings, and we plan to substitute the multiple-choice questions for more open-ended thought-provoking questions.

About independent study with oral presentation. Only one question targeted this aspect, "Do you deem appropriate the investigation and oral presentation of a topic of interest in databases?" Student responses were 93% positive and 7% negative. The main complaints were that the investigation was time-consuming and that it was difficult to understand academic papers in a specific database area at their level. On the positive side, many students mentioned that they learned about interesting topics that they would not have studied otherwise (i.e., they were not covered in the regular class material). Some noted that doing this investigation motivated them to learn more about databases, and others pointed out that working on oral presentations was an important skill for their careers.

About *Moodle* and other online resources. Four questions were asked on this subject, two specifically related to the *Moodle* platform, and two related to other online resources. The first of those questions, "Did you find the *Moodle* platform useful for this course?" obtained 97% positive responses and only 3% negative responses. We must note that *Moodle* was heavily used in that course to manage content, post assignments and news, send coursework reminders, and enable cooperation between peers through virtual forums. The second question was "Did you find useful the online forums created in *Moodle* for each of the projects in the course?" to which 79% of the students answered 'Yes' while 21% answered 'No'. Responses to the third question, "How much did you use the teacher videos as an educational source?" were distributed as follows: 83% of the students chose 'A lot', 14% chose 'A little', and 3% chose 'Not at all'. These videos were home-made by the professor of the course on specific topics that were not studied in class, and they were uploaded to *You tube*. Students accessed them through a link available from the *Moodle* platform. Responses to the fourth question, "How much did you use other educational sources (internet, online documentation for Oracle and SQL Server)?" were distributed as follows: 52% of the students responded 'A lot', 41% 'A little', and 7% 'Not at all'. It is worth mentioning that 5 of the 22 students who responded the last optional and open-ended question of the survey ("Write any additional comments, suggestions or observations concerning the course, the teacher, the teaching assistants, or any other aspect.") expressed that the teacher videos were very useful for them as a learning resource, which was an interesting finding for us.

About cooperative learning activities. One question dealt with an assessment of the learning that took place during the group and collaborative activities: "Do you think you learned more, equal, or less from the group activities conducted in class (discussion, exercise solving, etc.) than from a teacher's lecture?" Student responses were as follows: 45% answered 'More', 31% answered 'Equal', and 24% answered 'Less'. Combining the first two categories, we see that 76% of the students learn more or equal from group and cooperative activities involving their peers than from a teacher's lecture.

About learning and teaching. Two questions were asked towards the end of the survey, aimed at assessing the learning that took place as well as the teaching methodology, both from the student's perspective. The first of those questions was "How much do you think you learned in this course?" to which 97% of the students answered 'A lot', 3% answered 'A little', and 0% answered 'Nothing'. The second question was "How do you rate the course teacher and her teaching?" Here 97% of the students chose 'Good', 3% chose 'So-so', and 0% chose 'Bad'.

In summary, the results from this survey are encouraging and offer evidence of the student satisfaction and learning when using an active learning approach to teaching combined with online resources.

5. SUMMARY AND FUTURE WORK

We designed an active learning approach for teaching software testing in a graduate course, which requires support from online tools like *Moodle* and *Bug Hunt*, as well as from other software development and testing tools like *Visual Studio* 2010 *Ultimate* and *Team Foundation Server* 2010. The key principles underlying our approach are cooperative learning, peer assessment, and self reflection and assessment. We provided evidence, based on previous experience in other courses, for many of the active learning strategies proposed. We strongly believe that active learning is a powerful and effective teaching approach, and we plan to explore the effect of its use in a graduate level software testing course. We are going to implement the proposed approach in the PF-3866 and PF-3867 courses during the spring semester of 2011, and we plan to report on the results of applying this strategy once the course is finished.

6. REFERENCES

[1] Al-Ajlan, A., Zedan, H. Why Moodle. *12th IEEE International Workshop on Future Trends of Distributed Computing Systems* (2008).

[2] Bonwel, C.C. Active Learning: Creating Excitement in the Classroom. *Active Learning Workshops*, CO, USA (2003). Available at http://www.ydae.purdue.edu/lct/hbcu/documents/Active_Learning_Creating_Excitement_in_the_Classroom.pdf (Accessed December 13, 2010).

[3] Bower, M. Groupwork activities in synchronous online classroom spaces. *Proceedings of the 38th SIGCSE technical symposium on Computer science education* (2007).

[4] Bower, M. Independent, synchronous and asynchronous an analysis of approaches to online concept formation. *Proceedings of the 12th annual SIGCSE conference on Innovation and technology in computer science education* (2007).

[5] Boyeena, M., Goteti, P. Promoting Active Learning through case driven approach: An empirical study on database course. *IEEE Students' Technology Symposium* (2010).

[6] Briggs, T. Techniques for active learning in CS courses. *Journal of Computing Sciences in Colleges*, 21, 2 (2005).

[7] Brown, J.O. Know Thyself: The Impact Of Portfolio Development On Adult Learning. *Adult Education Quarterly*, 52, 3 (2002).

[8] Carter, P. An experiment with online instruction and active learning in an introductory computing course for engineers: JiTT meets CS1. *Proceedings of the 14th Western Canadian Conference on Computing Education* (2009).

[9] Chang, C.C. Construction and Evaluation of a Web-based Learning Portfolio System: An Electronic Assessment Tool. *Innovations in Education and Teaching International*, 38, 2 (2001).

[10] Chase, J.D., Okie, E.G. Combining cooperative learning and peer instruction in introductory computer science. *Proceedings of the thirty-first SIGCSE technical symposium on Computer science education* (2000).

[11] Collofello, J., Vehathiri, K. An Environment for Training Computer Science Students on Software Testing. *Proceedings of the 35th Annual Conference Frontiers in Education* (2005).

[12] Cordes, D., Parrish, A. Active learning in computer science: impacting student behavior. *Frontiers in Education*, 1 (2002).

[13] Elbaum, S. et al. Bug Hunt. Available at http://esquared.unl.edu/BugHunt/ (Accessed December 14, 2010).

[14] Elbaum, S., Person, S., Dokulil, J., Jorde, M. Bug Hunt: Making Early Software Testing Lessons Engaging and Affordable. *29th International Conference on Software Engineering* (2007).

[15] Gehringer, E.F., Chinn, D.D., Perez-Quiñones, M.A., Ardis, M. Using peer review in teaching computing. *Proceedings of the 36th SIGCSE technical symposium on Computer science education* (2005).

[16] Hamalainen, H., Ikonen, J., Porras, J. Using Wiki for Collaborative Studying and Maintaining Personal Learning Diary in a Computer Science Course. *Ninth IEEE International Conference on Digital Advanced Learning Technologies* (2009).

[17] Iding, M.K., Crosby, M.E., Speitel, T., Shimabuku, T., Nguyen, T. Cooperative and Collaborative Learning in Computer-Based Science Instruction. *Proceedings of the 34th Annual Hawaii International Conference on System Sciences* (2001).

[18]Juedes, D. Work In Progress - Anytime, Anywhere Active Learning in Computer Science. *Frontiers In Education Conference* - Global Engineering: Knowledge Without Borders, Opportunities Without Passports (2007).

[19] Kaner, C., Bach, J. Pettichord, B. Lessons Learned in Software Testing, John Wiley & Sons, 2001.

[20] Kaner, C., Falk, J., Nguyen, H.Q. Testing Computer Software, 2d ed., John Wiley & Sons, 1999.

[21] Kaner, C., Fiedler, R. Adaptation and Implementation of an Activity-Based Online or Hybrid Course in Software Testing: Proposal to the National Science Foundation, 2007. Available at http://www.kaner.com/pdfs/CirculatingCCLI2007.pdf (Accessed December 30th, 2010).

[22] Kaner, C., Padmanabhan, S. Practice and transfer of learning in the teaching of software testing. *Conference on Software Engineering Education & Training* (2007).

[23] Ludi, S., Natarajan, S., Reichlmayr, T. An Introductory Software Engineering Course that Facilitates Active Learning. *Proceedings of the 36th SIGSCE technical symposium on computer science education* (2005).

[24] Mao, C. Towards a Question-Driven Teaching Method for Software Testing Course. *International Conference on Computer Science and Software Engineering* (2008).

[25] McConnell, J. J. Active learning and its use in computer science. *Proceedings of the 1st conference on Integrating technology into computer science education* (1996).

[26] McConnell, J. J. Active and Cooperative Learning: Tips and Tricks (Part I). *ACM SIGCSE Bulletin*, 37, 2 (2005).

[27] Microsoft. Team Foundation Server 2010. Available at http://www.microsoft.com/visualstudio/en-us/products/2010-editions/team-foundation-server (Accessed December 14, 2010).

[28] Microsoft. Visual Studio 2010 Ultimate. Available at http://www.microsoft.com/visualstudio/en-us/products/2010-editions/ultimate (Accessed December 14, 2010).

[29] Myers, G.J. The Art of Software Testing. John Wiley & Sons, 1979.

[30] NIST. The economic impacts of inadequate infrastructure for software testing, May 2002. Available at http://www.nist.gov/director/planning/upload/report02-3.pdf (Accessed December 30th, 2010).

[31] Pezze, M., Young, M. Software Testing and Analysis, John Wiley & Sons, 2008.

[32] Pollard, S., Duvall, R.C. Everything I Needed to Know About Teaching I Learned in Kindergarten: Bringing Elementary Education Techniques to Undergraduate Computer Science Classes. *Proceedings of the 37th SIGCSE technical symposium on computer science education* (2006).

[33] Project Zero. MLV Home. Available at http://24web.harvard.edu/mlv/ (Accessed December 13, 2010).

[34] Resnick, M. Technologies for Creative Learning - Syllabus. Available at http://mas714.media.mit.edu/syllabus (Accessed December 14, 2010).

[35] Talon, B., Leclet, D., Lewandowski, A., Bourguin, G. Learning Software Testing Using a Collaborative Activities Oriented Platform. *Ninth IEEE International Conference on Advanced Learning Technologies* (2009).

[36] The Moodle community. Moodle Documentation. Available at http://docs.moodle.org/en/Main_Page (Accessed December 13, 2010).

[37] Timmerman, B., Lingard, R., Barnes, G.M. Active learning with upper division computer science students. *Frontiers in Education Conference*, 1 (2001).

[38] Tsai, W.T., Li, W., Elston, J., Chen, Y. Collaborative Learning Using Wiki Web Sites for Computer Science Undergraduate Education: A Case Study. *IEEE Transactions on Education*, 99 (2010).

[39] Watkins, J., Mazur, E. Using JiTT with Peer Instruction. In *Just in Time Teaching Across the Disciplines*, Ed. Scott Simkins and Mark Maier, pp. 39-62 (Stylus Publishing, Sterling, VA, 2009).

[40] Whittaker, J. A. What is software testing? And why is it so difficult, *IEEE Software*, 17, 1 (2000).

Alexandra Martinez has been working since 2009 as an Invited Professor in the Department of Computer and Information Science at the Universidad de Costa Rica, teaching Database and Bioinformatics courses as well as doing applied research in software testing at the Research and Development Center on Information and Communication Technologies, affiliated with the Universidad de Costa Rica. Previously, she worked as a Software Design Engineer in Test for Windows 7´s Media Player at Microsoft Corporation in Redmond, WA. She also worked as a Software Engineer at ArtinSoft in San Jose, Costa Rica. She received her Ph.D. in Computer Engineering from the University of Florida in 2007, her M.S. in Computer Engineering from the University of Florida in 2006, and her B.S. in Computer and Information Science from the Universidad de Costa Rica in 2000. She also received a scholarship to study in the Pre-Doctoral Program in Computer Science at the Ecole Polytechnique Fédérale de Lausanne, Switzerland, from 2001 to 2002. Her research interests include bioinformatics, software testing, data quality, and databases.

Marcelo Jenkins obtained a B.S. degree in Computer and Information Sciences at the University of Costa Rica in 1986 and a M.Sc. and Ph.D. degrees from the University of Delaware, USA, in 1988 and 1992 respectively. Since 1986 he has been teaching computer science at the University of Costa Rica. From 1993 until 1998 he coordinated the Graduate Committee and from 1998 through 2001 he was the Chairman of the Department of Computer and Information Sciences. His research interests are in software engineering, software quality assurance, project management, and object-oriented programming. He has authored more than 30 technical papers on these subjects. As an independent consultant, he has worked with some of the largest software companies in the Central America region in establishing software quality management systems. In the last 15 years, he has taught several seminars on software quality assurance and software project management in 7 different countries. Dr. Jenkins is an ASQ Certified Software Quality Engineer (CSQE) and a member of the ASQ Software Division, the IEEE Computer Society, and the Project Management Institute (PMI).