Mobile Distance Learning Driven Software Development Education

Sigrid Schefer-Wenzl and Igor Miladinovic

***Abstract*—Software development education requires students to be engaged in realistic software projects, which are increasingly developed in geographically distributed teams. Therefore, we started to include distance learning based projects into our software development courses. Combining distance learning and mobile technologies opens possibilities for new teaching methods, facilitating personalized, informal, situated, contextualized and ubiquitous learning. In this paper, we introduce a novel course module design, gradually increasing mobile distance learning activities. It leverages different teaching methods, including elements from Just-in-Time Teaching, project-based learning and peer assessment.**

***Index Terms*—Distance learning, just-in-time teaching, mobile learning, software development education.**

I. Introduction

Teaching software development courses is a non-trivial task. Software development courses typically have high dropout rates as well as high failure rates. One of the reasons frequently stated for students' lack of motivation is that software development courses typically focus on small, structured and well-defined problems that do not provide a realistic setting. This is mainly due to high requirements concerning the development of cognitive skills of students within a short time. In addition, state of the art tools and techniques in computer science are permanently changing.

As Frailey stated, we are currently witnessing a widespread change in how education is delivered which should also motivate us to use innovative methods in software development education [1]. Learning is known to be most effective when it is active, experiential, situated, problem-based, and provides immediate feedback [2]. On the other hand, software projects are increasingly developed in distributed and international teams. This requires students to develop collaboration and self-organizing skills [3].

To meet the demands of the industry, the University of Applied Sciences Campus Vienna offers a Bachelor program in Information Technology and Telecommunication. Key skills gained with this degree program include a solid understanding of software development as well as comprehension of modern software engineering principles. However, as experience has shown, students' capability concerning the practical application of programming skills and software engineering principles leaves room for improvement. We therefore currently undergo a substantial update and refurbishment of this study program in order to strengthen these skills. One key element is to design and implement more motivating and learner-centered course concepts. In this paper, we present a modern, distance learning based concept of a software development course module, composed of three consecutive software development courses. The main goals are enhancing learner engagement, improving learning outcomes, and strengthen collaboration and self-organizing skills. Several authors state that mobile distance learning can contribute to these goals (see, e.g., [4]-[8]). Moreover, mobile distance learning fosters the collaboration among students and teachers [9].

We designed the distance learning course concepts by integrating a set of selected teaching methods, gradually increasing the level of distance learning activities. In particular, we combine mobile distance learning with Just-in-Time Teaching (JiTT) elements [10], industry-like student projects and peer assessment. In order to investigate the course concepts, we performed qualitative interviews with all participants, focusing on the progress in programming and software engineering skills.

The remainder of this paper is structured as follows. Section II presents related work on distance learning in software development courses. In Section III, we introduce our course design for a Bachelor-level software development course module. In Section IV, we show the students' evaluation results as well as an example student project. Finally, Section V concludes this paper.

II. Related Work

A. Distance Learning in Software Development Education

Several authors analyze the challenges in teaching software development topics [11]-[16]. Amongst others, these challenges include the vast amount of material that needs to be taught in short time in order to enable students to start their own software projects, the limited quality of most programming textbooks, as well as the dissatisfying learning outcomes of such courses. It is also important that students receive proper feedback on their practical programming tasks in a timely manner which is often not realized in software development courses [17].

With the advent of distance learning and especially mobile learning, several authors started to argue that the teaching of programming should be done directly on the mobile devices itself [4]-[7]. Some authors also advocate for combining traditional, distance, and mobile learning strategies in order to successfully teach programming contents [18]. Another

Manuscript received September 19, 2017; revised October 26, 2017.

The authors are with the University of Applied Sciences Campus Vienna, IT and Telecommunication, Favoritenstr. 226, A-1100 Vienna, Austria (e-mail: sigrid.schefer-wenzl@fh-campuswien.ac.at, igor.miladinovic@fh-campuswien.ac.at).

doi: 10.18178/ijiet.2018.8.5.1058

positive effect of integrating mobile distance learning in software development education is that it allows simulating distributed software projects and including new technologies to make students more prepared for their future work environment [3], [19].

B. Mobile Distance Learning

The advancements in communication technology enable learners to use mobile technology anytime and anywhere to access up-to-date educational resources and individualize learning [20]-[24]. In addition, it facilitates the interaction with other students to share information or work collaboratively on a task or project. Thus, mobile learning is both formal and informal. Situated and contextualized learning, i.e. the application of knowledge and skills to specific contexts, is also supported, since learners can apply what they learn immediately [20], [25].

Different kinds of challenges in mobile learning scenarios have been addressed in [25]. Some students may experience frustrations in mobile learning scenarios that may result, for example, from the reluctance of instructors to provide learning material that is optimized for mobile devices. Moreover, other applications on the device may distract the students from learning. Some studies also argue that mobile learning leads to constant connectivity and thereby fosters digital addiction of students as their personal identity merges with their mobile devices [25].

C. Just-in-Time Teaching

Just-in-Time Teaching (JiTT) was introduced in [10]. The main aim of this teaching method is to combine distance learning assignments with active in-class hours. Students prepare themselves via online material and web-based assignments for the subsequent lecture, the instructor reads and evaluates the student submissions just-in-time, i.e. shortly before the next in-class lesson, to adjust the lecture according to the students' needs. Thus, the feedback loop between the students' self-study preparation influences which contents are discussed in more detail in the upcoming in-class lesson. Jonsson presented a course concept for integrating JiTT into a programming course [26]. He then compared the outcomes of the course with a programming course teaching the same contents in a traditional lecture form. The share of students passing the course as well as the percentage of good grades increased substantially in the JiTT-based course. Another example for integrating elements of JiTT into software development education has been introduced in [27]. In a pre-lecture phase background information and warm-up exercises are prepared by the students, which are reviewed by the lecturer and then discussed in class, followed by post-lecture activities. In course evaluations, students reported better learning performance due to the introduced JiTT elements.

III. Course Module Design

In this section, we present the design for our newly developed software development course module tailored to cover the identified shortcomings regarding insufficient programming and software engineering skills. Students take this module in the 4th and 5th semester of their Bachelor study and it is awarded with 18 ECTS credits, which corresponds to approximately 450 working hours. This module consists of a lecture (4 ECTS), a tutorial (4 ECTS) and a project (10 ECTS).

A. Module Overview

When designing our course module concept, we had to address several challenges. One challenge is that students enter our course with very diverse levels of programming skills. This is mainly because we also offer an evening study allowing students to pursue a study while working full time. Thus, our student groups include professional software developers next to students with very limited software development experience. Our combination of mobile distance learning and JiTT concepts addresses this issue by enabling individualized learning speed and intensity.

Another challenge is to support students in finding appropriate project ideas for their own software projects. These projects need to be settled in a realistic environment close to industry standards. However, students need to be able to finish these projects with reasonable effort within one semester. Thus, we also want to ensure a continuous progress tracking of the students' projects. We addressed these topics with a hackathon event, regular coaching units throughout the course as well as tool support for project management and collaboration.

The third challenge is to develop self-organizing skills of students, which are at very different levels among students. To address this challenge we gradually increased the amount of distance learning elements throughout the module from basic in the lecture to high in the project.

B. Learning Objectives

Our module concept is based on knowledge levels according to Bloom's Taxonomy [28]. For each part of the course module we defined the targeted knowledge levels (see Fig. 1). Thus, the main aim of the lecture is to provide a solid theoretical background that enables them to apply the taught concepts in tutorial exercises and later on to create their own projects. Moreover, students have to analyze and evaluate other projects. We achieve these levels by gradually increasing the amount of distance learning activities (see Section III.C).

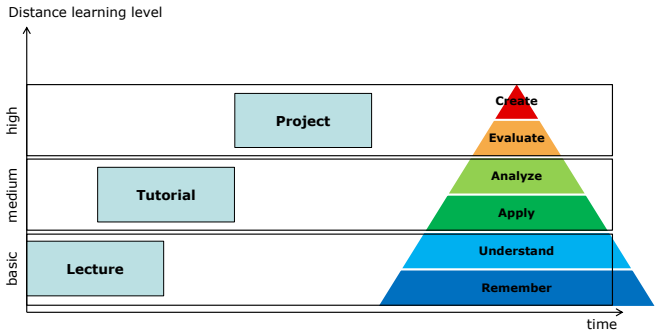

Fig. 1. Knowledge levels.

We derived the following learning objectives for our "Software Development" course module:

- Understand and apply an effective software development process, based on knowledge of widely used software

process models.

- Employ team working skills including organized planning, time management and inter-group negotiation.
- Capture, document and analyze requirements.
- Translate a requirements specification into an implementable design, following a structured process.
- Make effective use of the Unified Modeling Language (UML) and apply software design strategies.
- Implement the software according to the specification
- Design a testing strategy for a software system, employing techniques such as unit testing, test driven development and functional testing.
- Evaluate the final projects by checking compliance with the requirements, and analyze the design and implementation.

C. Applied Teaching Methods

In order to achieve the learning outcomes stated above, we designed a distance learning module concept by integrating a set of different teaching methods (see Fig. 2).

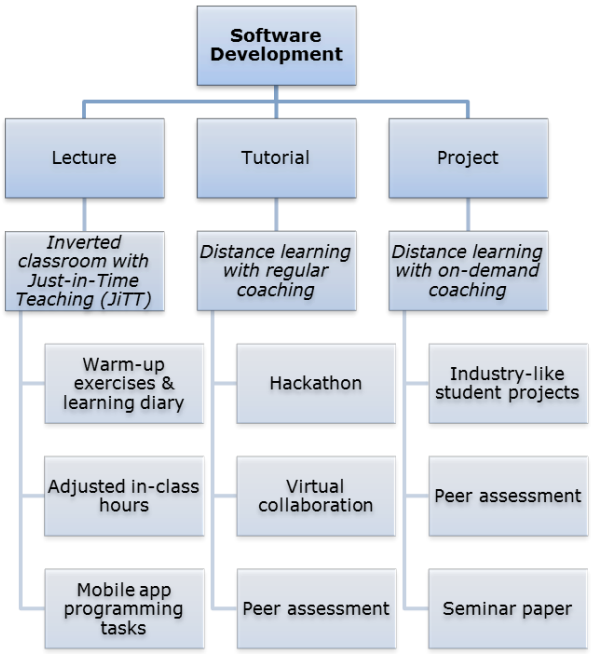

Fig. 2. Method mix.

In the lecture, we apply an inverted classroom setting [29], [30] combined with elements of JiTT and mobile learning. In an inverted classroom, lecture content is provided over some asynchronous medium. Students access this content remotely. During the class hours students are involved in learning activities such as in-class assignments, laboratories, and discussions. The learning content is provided in short learning snippets that can be consumed fast and from anywhere via a smartphone or tablet. We also supply short duration videos, usually lasting between two and four minutes, explaining and illustrating complex lecture contents.

In the lecture, we combine the inverted classroom concept with JiTT elements. Students prepare for each lecture by reading and watching learning material as well as doing warm-up exercises. Subsequently, the students reflect each self-study phase in a learning diary, where they answer questions like: "What did I learn? What was new to me? Was there something that changed my views and why? Focus on and analyze the themes important to you." and "What did I not understand? What went against my own ideas and why? What was less comprehensible and why? Focus on and analyze questions that left you puzzled." The instructors provide feedback for each learning diary entry before the next in-class lesson and adjust the contents of the lesson accordingly. Based on the learning diary entries, the in-class hours start with clarification of open questions and then focus on further deepening the contents, especially those which were not well understood in the self-study phase. The combination of inverted classroom and just-in-time teaching enables a strong learner-orientation as well as individualization of course contents during class hours. The distance learning aspect further supports location- and time-independent learning as well as immediate interaction between students about learning contents.

The tutorial part of this module is partly intertwined with the lecture. It aims at applying the concepts from the lecture within a student project, i.e. the design and development of a mobile application. The student groups consist of two to three people who work on these projects on their own. In addition, they are supported by regular meetings with their instructors as well as a range of software tools for project management, code version management and tools supporting the virtual collaboration of students during the software development process.

We officially start the tutorial part with a hackathon event. A hackathon (hacking marathon) can be described as a problem-focused computer programming event, where people come together to develop prototype applications by collaborating intensively over a short period of time [31]. Organizations use hackathons to test their new products and services as well as to generate new ideas [32]. They either focus on attracting external developers or requesting own employees to focus on a specific problem. In accordance with [33], the main goals of our hackathon event are to support team formation, provide an opportunity to creatively develop and assess software project ideas via face-to-face interactions and thereby increase motivation of the students to put a lot of effort in realizing their ideas.

Within the hackathon event, students first collect in a brainstorming manner as many project ideas as possible, then form teams and subsequently reduce the number of ideas, until each team has decided on three particular ideas, which they are going to present to the others. Immediately after the presentation the other students rank the presented project ideas by online voting. Based on this ranking, students select the idea which they are going to implement. Following [34], each team defines the requirements for their project in four prioritization categories using the MoSCoW system - Must have, Should have, Could have, and Won't have but would like [34]. After the hackathon event, students start working on their projects with regular coaching meetings with their instructors. When the projects are finished, the final projects are evaluated via peer assessment. The projects are distributed between the different student groups, which then check the conformance between the originally stated MoSCoW requirements and the behavior of the final

implementation.

In the project we simulate an industry-like software project, where the acceptance of a software is done by persons not being involved in the development. Students have to come up with own ideas and implement them within the defined schedule. To stimulate distance learning, coaching by lecturers is provided on-demand in contrast to regular coachings during the tutorial. Depending on the self-organizing skills of the group the coaching meetings occurred more or less frequently. The main goal of this course part is on the one hand to introduce students to the nature and complexity of real software projects, and on the other to stimulate the development of self-organizing skills of the students through distance learning elements.

IV. Results

A. Qualitative Interviews

At the end of the lecture part qualitative interviews were performed with the participants ($n = 21$). The interviews were conducted by non-lecturers, each interview lasting for approximately one hour. The main goal of this qualitative evaluation was to provide students with the opportunity to reflect on their progress in programming and software engineering skills through our course. In future work, we will conduct further qualitative and quantitative evaluations and compare these results.

In the first part of the interview, we asked the students to rate their individual increase in different skill areas by using a scale from 1, i.e. no knowledge at all, to 10, i.e. expert knowledge. One of the main goals of our course was to further develop software development and software engineering skills. Fig. 3 presents the students' average self-rating of software development and software engineering skills before and after the course, while Fig. 4 shows the respective standard deviation for each rating. The average rating of software development skills increased from 4.5 points before our software development course to 6.9 points after our course with a standard deviation of 1.8 and 1.31 points respectively. Software engineering skills were rated with 3.7 points before and 6.4 points after the course with a standard deviation of 1.47 and 2.12 points. For more information about detailed evaluation results please refer to [35].

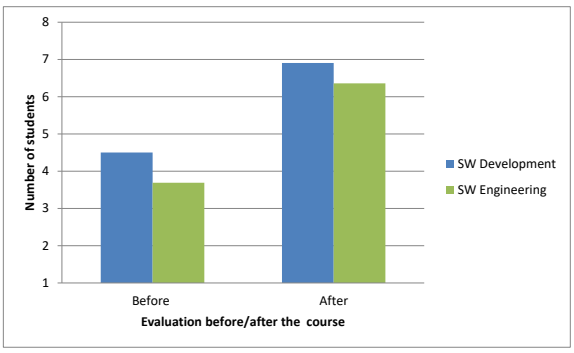

Fig. 3. Average students' rating.

B. Distance Learning Project

In the distance learning Os.car student project, geographically separated student teams designed and developed a mobile app for the interdisciplinary industry project, called "Formula Student". "Formula Student" is the worldwide competition for engineers, which is held annually at different locations around the world. More than 600 teams from different universities around the world compete with their self-constructed cars in various disciplines. This project involves knowledge in the areas of manufacturing and telecommunication. The outcome of the project is a new architecture for monitoring engine-conditions and car's information during the race, as illustrated in Fig. 5.

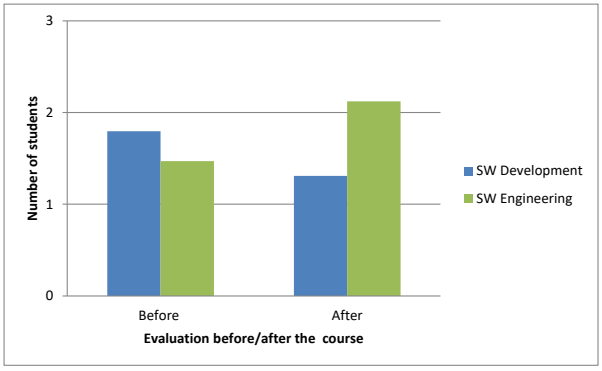

Fig. 4. Standard deviation of students' rating.

Fig. 5. Os.car architecture.

Three groups – each composed of two students – have been working on different parts of the architecture, as indicated in Fig. 5. The first group implemented the connection to the car system and the sender unit in the car based on XBee technology [36]. The second group realized the base station including a receiver unit for information from the car, a processing unit based on an Arduino board [37], as well as the wireless LAN access point for distributing this information among the visitors' devices. Finally, the third group designed and developed a mobile application to display this information on visitors' devices.

In qualitative interviews presented in Section IV.A, we have noticed in average 21.2% higher progress in the Os.car group than in the other groups. The students stated several reasons for their high learning progress, including:

- the highly motivated Formula Student team that acted as a customer to our students,
- working on real problems with tough deadlines,
- competition character,
- success stories visible fast after solving an issue, and

- lecture, tutorial and coaching concept as a platform for continuous and flexible distance learning.
- More information about this project can be found in [38].

V. Conclusion

In this paper, we present a module design for teaching software development topics. We gradually increased distance learning activities, and combined them with another educational trend called mobile learning, where mobile devices are used as the primary learning medium. The goal of this course module was to enable students to learn software development in an industry like project experiencing all knowledge levels according to Bloom's Taxonomy.

The students' evaluations as well as the lecturers' impressions confirm that the mobile distance learning concept has been well-received by most participants. In future research, we will further apply and evaluate this module concept in different student settings.

References

[1] D. J. Frailey, "The times, they are changing," *Software Engineering Education and Training*, pp. 1-2, Klagenfurt, Austria, 2014.

[2] E. A. Boyle, T. M. Connolly, and T. Hainey, "The role of psychology in understanding the impact of computer games," *Entertainment Computing*, vol. 2, no. 2, pp. 69–74, 2011.

[3] S. Beecham, T. Clear, D. Damian, J. Barr, J. Noll, and W. Scacchi, "How best to teach global software engineering? educators are divided," *IEEE Software*, vol. 34, no. 1, pp. 16-19, Jan.-Feb. 2017

[4] M. Pasamontes, J. L. Guzman, F. Rodriguez, M. Berenguel, and S. Dormido, "Easy mobile device programming for educational purposes," in *Proc. the 44th IEEE Conference on Decision and Control*, pp. 3420–3425, Dec 2005.

[5] N. Tillmann, M. Moskal, J. de Halleux, M. Fahndrich, J. Bishop, A. Samuel, and T. Xie, "The future of teaching programming is on mobile devices," in *Proc. the 17th ACM Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE '12*, pp. 156–161, New York, NY, USA, 2012.

[6] Y.-C. Hsu and Y.-H. Ching, "Mobile app design for teaching and learning: Educators' experiences in an online graduate course," *The International Review of Research in Open and Distance Learning*, vol. 14 nr. 4, pp. 117–139, 2013.

[7] A. Esakia and D. S. McCrickard, "An adaptable model for teaching mobile app development," *2016 IEEE Frontiers in Education Conference (FIE)*, pp. 1–9, Oct. 2016.

[8] H. Tseng and E. J. Walsh, "Blended versus traditional course delivery: Comparing students' motivation, learning outcomes, and preferences," *Quarterly Review of Distance Education*, vol. 17, no. 1 pp. 43-52, 2016.

[9] E. Vazqzez-Cano, "Mobile Distance learning with smartphones and apps in higher education," *Educational Sciences: Theory and Practice*, vol. 14 no. 4, pp. 1505-1520, 2014.

[10] G. M. Novak, E. T. Patterson, A. D. Gacrin, and W. Christian, *Just-In-Time Teaching: Blending Active Learning with Web Technology*, Prentice Hall, 1999.

[11] K. Gary, T. Lindquist, S. Bansal, and A. Ghazarian, "A project spine for software engineering curricular design," in *Proc. the 26th International Conference on Software Engineering Education and Training (CSEE T)*, pp. 299–303, May 2013.

[12] C. Ghezzi and D. Mandrioli, *The Challenges of Software Engineering Education*, pp. 115–127, Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[13] A. Fox, D. A. Patterson, R. Ilson, S. Joseph, K. Walcott-Justice, and R. Williams, "Software engineering curriculum technology transfer: Lessons learned from moocs and spocs," Technical Report UCB/EECS-2014-17, EECS Department, University of California, Berkeley, March 2014.

[14] E. Lahtinen, K. Ala-Mutka, and H.-M. Järvinen, "A study of the difficulties of novice programmers," *SIGCSE Bull.*, vol. 37 no. 3, pp. 14–18, June 2005.

[15] A. Pears, S. Seidman, L. Malmi, L. Mannila, E. Adams, J. Bennedsen, M. Devlin, and J. Paterson, "A survey of literature on the teaching of introductory programming," *Working Group Reports on ITiCSE on Innovation and Technology in Computer Science Education, ITiCSE-WGR '07*, pp. 204–223, Dundee, Scotland. ACM, 2007.

[16] A. Robins, J. Rountree, and N. Rountree, "Learning and teaching programming: A review and discussion," *Computer Science Education*, vol. 13, no. 2, pp. 137–172, 2003.

[17] T. Staubitz, H. Klement, J. Renz, R. Teusner, and C. Meinel, "Towards practical programming exercises and automated assessment in massive open online courses," in *Proc. the IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)*, pp. 23–30, Dec 2015.

[18] G. Venugopal-Wairagade, "Study of a pedagogy adopted to generate interest in students taking a programming course," in *Proc. the International Conference on Learning and Teaching in Computing and Engineering (LaTICE)*, pp. 141–146, March 2016.

[19] L. Briz-Ponce and J. A. Juanes-Méndez, "Mobile devices and apps, characteristics and current potential on learning," *J. Inf. Technol. Res.*, vol. 8, no. 4, pp. 26–37, Oct. 2015.

[20] M. Ally and J. Prieto-Blazquez, "What is the future of mobile learning in higher education?" *Revista de Unicersidad y Sociedad del Conocimiento (RUSC)*, vol. 11, no. 1, pp. 142–151, 2014.

[21] L. F. Motiwalla, "Mobile learning: A framework and evaluation," *Computers & Education*, vol. 49, no. 3, pp. 581-596, 2007.

[22] J. B. Roberts, *Handbook of Mobile Learning, Chapter Accessibility in m-Learning: Ensuring Equal Access*, pp. 427–435, Routledge, New York, NY, 2013.

[23] P. R. H. Ramos, F. J. G. Penalvo, and M. A. C. Gonzalez, "Towards mobile personal learning environments (mple) in higher education," in *Proc. the 2nd International Conference on Technological Ecosystems for Enhancing Multiculturality (TEEM)*, Salamanca, Spain, October 2014.

[24] M. Sarrab, M. Elbasir, and S. Alnaelic, "Towards a quality model of technical aspects for mobile learning services: An empirical investigation," *Computers in Human Behavior*, vol. 55(Part A), pp. 100–112, 2016.

[25] J. Gikas and M. M. Grant, "Mobile computing devices in higher education: Student perspectives on learning with cellphones, smartphones & social media," *The Internet and Higher Education*, 19, pp. 18-26, 2013.

[26] H. Jonsson, "Using flipped classroom, peer discussion, and just-in-time teaching to increase learning in a programming course," *IEEE Frontiers in Education Conference (FIE)*, pp. 1–9, Oct 2015.

[27] Y. Tao, G. Liu, J. Mottok, R. Hackenberg, and G. Hagel, "Just-in-time-teaching experience in a software design pattern course," in *Proc. the IEEE Global Engineering Education Conference (EDUCON)*, pp. 915–919, March 2015.

[28] B. S. Bloom, "Taxonomy of educational objectives, handbook I: Cognitive domain," New York: David McKay, 1956.

[29] M. J. Lage, G. J. Platt, and M. Treglia, "Inverting the classroom: A gateway to creating an inclusive learning environment," *The Journal of Economic Education*, vol. 31, no. 1, pp. 30-43, 2000.

[30] G. Gannod, J. Burge, and M. Helmick, "Using the inverted classroom to teach software engineering," in *Proc. the ACM/IEEE 30th International Conference on Software Engineering*, pp. 777–786, May 2008.

[31] H. Topi and A. Tucker, *Computing Handbook. Information Systems and Information Technology*, CRCPress, 2014.

[32] B. Rosell, S. Kumar, and J. Shepherd, "Unleashing innovation through internal hackathons," *IEEE Innovations in Technology Conference*, pp. 1–8, May 2014.

[33] E. H. Trainer, A. Kalyanasundaram, C. Chaihirunkarn, and J. D. Herbsleb, "How to hackathon: Socio-technical tradeoffs in brief, intensive collocation," in *Proc. the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing, CSCW '16*, pp. 1118–1130, New York, NY, USA, 2016.

[34] D. Clegg and R. Barker, *Case Method Fast-Track: A RAD Approach*, Addison-Wesley, Boston, MA, USA, 1994.

[35] S. Schefer-Wenzl and I. Miladinovic, "Game Changing Mobile Learning Based Method Mix for Teaching Software Development," in *Proc. the mLearn 2017, 16th World Conference on Mobile and Contextual Learning*, 2017.

[36] Digi International. (2017). [Online]. Available: https://www.digi.com/lp/xbee/

[37] Arduino. (2017). [Online]. Available: https://www.arduino.cc/

[38] I. Miladinovic and S. Schefer-Wenzl, "Pawing the way for interdisciplinary learning in software development education," in *Proc.*

of the mLearn 2017, 16th World Conference on Mobile and Contextual Learning, 2017.

Sigrid Schefer-Wenzl is a senior re searcher and lecturer at the University of Applied Sciences Campus Vienna, WU Vienna, and the University of Salzburg. Sigrid has worked as a software analyst and developer in several companies and received the Ph.D. degree (with honors) in Information Systems from WU Vienna. Sigrid's current research and teaching activities focus on the fields of software engineering and IT-security.

She has published the results of her work in top ranked journals and presented her work at various international conferences.

Igor Miladinovic is the head of the degree program Information Technologies and Telecommunication at the University of Applied Science Campus Vienna. Igor received the Ph.D. degree (with honors) in electrical engineering from Vienna University of Technology in 2003. Igor worked for more than 10 years on leading positions at Alcatel-Lucent (later Nokia) in area of telecommunication software and in parallel as a lecturer at two universities.

His research interests cover telecommunication networks, software engineering and IoT, with over 30 publications in international journals, conferences and as book chapters.