TAMPERE UNIVERSITY OF TECHNOLOGY

Applying third-party moocs in programming education: a case study

APPLYING THIRD-PARTY MOOCS IN PROGRAMMING EDUCATION: A CASE STUDY

Mika Saari, Timo Mäkinen, Petri Linna

Tampere University of Technology – Pori (FINLAND)

Abstract

Utilizing MOOCs developed by universities worldwide seems to be a viable option for organizing online education. However, the implementation of a third-party MOOC is not straightforward, because it may cause major changes in teaching processes. This case study describes some experiences gathered from the deployment of a third-party MOOC on a programming course in a small university department. The topic is discussed from the perspectives of instructor and students alike. The analysis of the results of the study shows that the students' attitudes to the refined way of organizing the course were not solely positive. The change in the instructor's total workload is significant while there are also changes in the types of work items to be performed. The results were encouraging enough for us to continue experimenting with third-party MOOCs on our courses.

Keywords: MOOC, programming education, student feedback, case study.

1 INTRODUCTION

Nowadays, it is becoming more and more common practice for universities to arrange education for students using massive open online courses (MOOCs) [1]. MOOCs have open access via the web and participation is unlimited. The expenses of developing a MOOC are quite high [2], and therefore small units may not have enough resources to implement high quality online courses of their own. For that reason, utilizing other universities' MOOCs constitutes a considerable option. However, the implementation of a third-party MOOC is not straightforward, because it may cause major changes in teaching processes [3].

Based on a case study carried out in the fall of 2016, this paper discusses what should be taken into account when deploying a third-party MOOC on a programming course. There are three questions:

1 How did the teaching process change after implementing the MOOC?

2 How did the change affect the workload concerning students and teacher alike?

3 How did the students experience the new way of implementing the course?

To answer these questions, a case study was arranged at the Pori Department of the Tampere University of Technology (TUT/Pori). The programming MOOC (CS/MOOC) of the University of Helsinki Department of Computer Science (Helsinki/CS) was used in this study.

In TUT/Pori, the student intake is relatively small, at about 60 students each year. The unit is one of the 19 departments of the Tampere University of Technology, but is located over 100 hundred kilometers away from the main campus, which has about 8 300 students. TUT/Pori offers education leading to a Master's degree in Management and Information Technology. Postgraduate studies leading to the degrees of Doctor of Science and Doctor of Philosophy in Technology can also be completed. In the department, there are three major subjects to study: Software Engineering and Data Management, Network Management and Information Security, and Industrial Engineering and Management. TUT/Pori is part of the University Consortium of Pori (UCPori), which is a center of 2500 students, 170 experts and four universities, operating as a network in a multi-science environment. As far as education is concerned, the focus of UCPori is on undergraduate and postgraduate degree programmes, multidisciplinary studies, and adult education.

TUT/Pori started a gradual transition to blended learning, formerly known as multiform learning [3], methods in the fall of 2013. Blended learning was seen as a solution offering flexible education for students and a solution to the competition for new students. Blended learning can also represent a method and technologies that make it easier to combine study, work, and family life. During previous years, a variety of blended learning implementation methods and technologies have been tested [3-8].

In practice, the transition to blended learning adds a large number of new issues to address. Furthermore, for the teaching staff, the transition to blended learning creates a great opportunity to develop new kinds of teaching pedagogies and use new technologies, although resistance to change is a challenge. Here are some examples of questions that need to be solved: How are the courses scheduled? How are the problems of physical distances solved? How are the materials distributed? How is communication secured? What is the commitment of students and teachers like? How are the needs for pedagogical change established and utilized? How are (remote, network, electronic) examinations arranged? How are the training sessions and laboratories arranged? How are students motivated to use eLearning? The focus of this paper is on the last two questions of the list.

This case study was arranged in the first programming course for students starting a course leading to a Master's degree in Management and Information Technology. On this course, the student will learn to change a problem into a computer program. The students obtain basic programming skills that can be used as a tool in their studies and research work. The students will reach a baseline from which they will be able to develop into skilled programmers through practice and study. It also aims to help students to feel justifiably confident of their ability to write small programs that allow them to accomplish useful goals. The course uses the Java programming language [8].

In section 2, the background of the programming course and the University of Helsinki Department of Computer Science programming MOOC (CS/MOOC) is introduced. The combination process is also described. In section 3, we will review the survey and the feedback given by the students. Also, self-evaluation is handled. The last section is reserved for conclusions and final discussion.

2 STARTING POINTS

First, this section describes our previous programming course arrangement. Then, we describe the CS/MOOC used in this case and briefly review its design and composition.

2.1 Course prior to Utilization of MOOC

The previous programming course had been similar since the year 2002. Then the Java programming language [9] was introduced on this basic studies level course. During the years, minor developments had taken place but the main idea had been the same. The course was carried out mostly by means of lectures. The course included exercises and two exams. Students returned exercises to the training assistants and received points. The course rating was based on collected points from exercises and exams. Exams were organized so that students received information on their own level of competence. The course was the first programming course for information technology students in our department.

In the fall of 2015, this course was a six-credit course with 60 students. It consisted of three-hour-long lectures once a week for 13 weeks. There were two exercise groups of two hours per week. This course had only one instructor, who both gave the lectures and supervised the exercises. As in the previous implementations, the grade of the course was based on points from the exercises and the examination. The students returned the exercises by email. Therefore all the contact hours were optional. After the course, an exam was arranged using an electronic examination system, which was discussed in our previous study [8].

2.2 MOOC Applied in this Study

The Department of Computer Science of University of Helsinki arranged a "Ohjelmoinnin perusteet"(Programming basics) MOOC for the first time in January 2012 where they omitted lectures and only provided learning material and exercises on the Internet [10, 11]. This course was on the basics of computer programming, and was arranged only in Finnish. This course was based on web material, exercises, the Test My Code NetBeans (TMC) plugin, and server [12]. The whole CS/MOOC material takes 14 weeks, and students receive 10 credits for completing the course. There is one webpage per week, which contains both the theory and exercises. The whole set of material includes over 200 programming exercises, which are designed for implementation with NetBeans IDE (Integrated Development Environment) [13] and TMC programming tools. Each exercise includes automated tests and if the answer code passes all the tests, the student can send the answer code to the TMC server with the TMC tool. The TMC server counts the points and presents them in the webpage scoreboard. The architecture of MOOC arrangements is depicted in Figure 1.
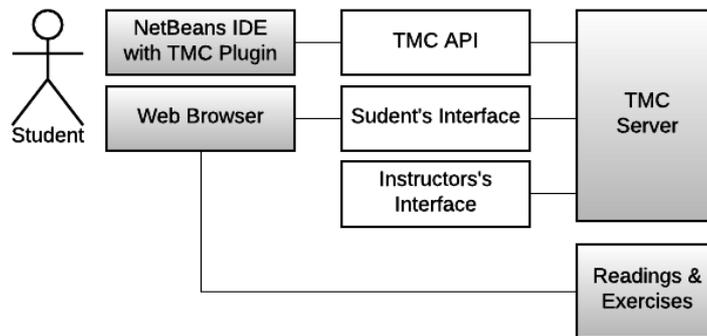
Figure 1. Architecture of MOOC.

CS/MOOC material is in Finnish and licensed under Creative Commons BY-NC-SA [14]. The University of Helsinki offers the maintenance of the course for free and therefore the courses are widely used, in 32 educational institutes in Finland.

3   COURSE REVISION FROM INSTRUCTOR'S PERSPECTIVE

In this section, we will answer the first question "How did the teaching process change after implementing the MOOC?" At first the implementation procedure is handled by explaining how we combined the CS/MOOC and the programming course. Then the instructor's self-evaluation is discussed, which is the partial answer to the second question "How did the change affect the workload concerning the students and teacher alike?"

3.1   Introduction of the MOOC in the Course

The implementation of the new teaching material for the course had several phases: Need for teaching material, qualification of chosen material, studying the material, introducing and teaching the material. In this case study, we did not focus on the selection process. However, if one of the selection criteria is the Finnish language, there are not many alternatives. The second phase was material qualification and test suitability for our use. This was performed by participating in the open implementation of the course managed by the Helsinki/CS. The results of this were the time consumption and suitability of exercises. This also showed the technical usability of the course - whether there were any technical problems or how much technical support was needed during the course. No issues of this kind were found.

The course construction phase was made easy, because the TMC server was maintained by the Helsinki/CS. The instructors of our programming course were given access for maintaining the material, choosing the exercises, and managing the deadline dates. The registering to the course was open to everybody so the students can register themselves to the course.

The arrangement of our programming course included contact hours twice a week for the whole of the fall semester. The first contact time was the starting lecture, which dealt with the pass criteria of the course, the way to work with the exercises and the examination method. During the first and second session, hands-on training was given in using NetBeans and TMC tools.

Table 1. *Grade of student from MOOC points*

Grade	students	percentage
5 (90% - 100% of exercises)	14	19.72%
4 (80% < 90% of exercises)	8	11.27%
3 (70% < 80% of exercises)	11	15.49%
2 (60% < 70% of exercises)	9	12.68%
1 (50% < 60% of exercises)	15	21.13%
0 – Fail (<50%)	14	19.72%
Total	71	

The students received points from past exercises as mentioned above. Table 1 shows the grading of the course collected from the MOOC grading system. This is not the final result, because each student also has to pass the examination. The purpose of the exam arrangement is to prevent cheating on the exercises, because if the student has not actually done the exercises he or she will not usually pass the exam.

3.2   Effects on Instructor's Work

As previously mentioned, we decided to perform a comparison of this programming course to answer the presented research questions - before and after, as shown in Table 2. There was no formal method for this self-evaluation. The overall schedule consisted of course planning, preparing laboratories, preparing lectures, enrollment for the course examination, and grading. These phases were introduced in the previous sections so in this section we will focus on the change in the use of time. The course planning, study, and evaluation of the course took a lot more time than before, because the material was not familiar. In 2015, old lecturing material was used. Furthermore, exploring the exercises took time although they were known well. We arranged only one three-hour lecture on this course. The starting lecture included information about how the students would undertake this programming course. The rest of the course in the fall was less time consuming than the implementation of previous years, because of the lack of lectures. Naturally, the weekly meetings required preparation for the scheduled exercises. On the other hand, the evaluation of the students' exercise solutions did not take so much time thanks to the automated testing. The examination took the same time as before. We use an automated exam system, which was introduced in our previous study [8]. Most of the students passed the examination without remarks, and the grade of the course was determined on the basis of the returned exercises.

Table 2. *Comparison of instructor's workload in course implementation.*

	2015	2016
Study and evaluation of course	0	35
Overall course arrangements	20	20
Preparation of lectures	36	3
Preparation of laboratories	50	0
Lectures	36	3
Laboratories	48	57
Evaluation of students' solutions	100	0
Email communication	10	30
Examination arrangements	20	20
Exam review and grading	20	20
Total estimated workload (h)	**340**	**188**

The overall conclusion is that utilizing this Programming MOOC decreased the need for lecturing resources on courses while introducing new creative ways to arrange high-level education for students.

4   STUDENTS' EXPERIENCES OF THE REVISED COURSE METHODOLOGY

This section describes research approaches related to student feedback and lecturers' self-evaluation. Feedback was gathered from the students using a standard questionnaire used in most Tampere University of Technology courses. The interviewing was not formal; instead it was in the form of a discussion held during the course's weekly meetings.

4.1   Feedback Questionnaire

After the course implementation, feedback was gathered from the students using a questionnaire and by interviewing a few students. The questionnaire consists of a multiple-choice survey (Table 3) and open-ended questions (Table 4). The questions were of common type, because we did not want to

focus on the MOOC implementation. In this kind of arrangement, we tried to elicit the overall opinion of MOOC-type courses. This same questionnaire had been used before this course, so there were results from several years for comparison.

Table 3. *Summary of the answers to the multiple-choice questions of the feedback questionnaire.*

Question	2015 (n=27)				2016 (n=41)			
	mo	md	$\overline{x}$	s	mo	md	$\overline{x}$	s
1. Overall rating of the course and its implementation. (0-5)	4	4	4.10	0.66	4	4	3.81	0.94
2. Please evaluate the appropriateness of the course in relation to the intended learning outcomes. (0-5)	4	4	4.05	0.84	4	4	3.92	0.87
3. Did the workload correspond to the number of credits awarded for the course? (1:light, 2: comparable, 3: laborious)	2	2	2.05	0.22	2	2	2.44	0.50

Legend: n - number of answers, mo - mode, md - median, $\overline{x}$ - mean, s - standard deviation.

Table 3 presents the multiple choice survey results from 2015 and 2016. The questionnaires used the same questions in both years. As mentioned above, in 2015 the implementation of the programming course was traditional. However, in 2016 we used the MOOC implementation.

The first and second questions were scaled from zero to five. The zero was "Very Poor" to five "Excellent." The third question was scaled from one to three. One means "The course workload was light compared to the number of credits," two means "The course workload corresponded to the number of credits" and three was "The course workload was heavy compared to the number of credits". Table 3 presents the mode, median, mean, and standard deviation values calculated from the answers. The first row also shows the number of answers in both years.

There is no significant change between the grades in Table 3. 2016 has a slightly lower grade for questions one and two concerning the mean. The standard deviation change between 2015 and 2016 may be due to the overall change of the implementation method; in 2015 there were 12 three-hour lecture sessions and 30 exercises, but 2016 there were 134 exercises and no lectures.

Table 4. *Number of answers to open questions in 2016.*

Question	2016
4. What worked well during the course?	18
5. How would you develop the course?	12

4.2 Open Ended Questions and Oral Feedback

The number of answers to the open questions are presented in Table 4. The results of the previous 2015 questionnaire are omitted, because of the different kind course implementation. The purpose of Table 4 is to show that more than half of the students did not answer the open questions. As shown in our previous research [8], a reward for responding may raise the answer percentage, but this was not possible in this case.

The answers to the fourth and fifth questions were mostly positive; of course the fourth question was positively worded. The aspects raised by the fourth question were as follows: Ability for distance learning. Determining the grade based on the tasks performed by the course. The study material was good. Possibility of classroom teaching was good. Programming is interesting. A few students also mentioned some negative aspects - The material was laborious. This same issue came up a few times in the answers to the fifth question, where students were asked about developing the course. The other development suggestions were: More hands-on exercises at the beginning of the course. More example solutions. The course was too laborious. One reason for the "laborious" comments might be

that the course was also open to students other than information technology students. In our department, the programming techniques course is obligatory for all first year students. The obligatory and comprehensive aspects have been taken for further development. In addition, one development aspect was the timing of the course. The 2016 course had strict deadlines for the exercise packages, but in the future we will also offer the course without deadlines. This also provides the possibility to arrange the course as a continuous implementation, for example during the summer season.

The interviews with students were informal; held as a discussion during the course's weekly meetings. These discussions brought out needs like: exercise coding in the group - the instructor leads the coding session and the attempt is to solve some exercise from MOOC material. The need for overall theories of programming - because there were no lectures, only written material, some of the students asked the instructor to clarify some terms and constructions of the programming language.

The interviews revealed the students' collaborative working using Facebook groups. The students had set up a group where they could ask for solutions to problems with the programming exercises of the course. The University of Helsinki maintains an IRC channel for questions, but according to the students, this was not commonly used. Instead the Facebook group had about 45 members and the discussions were active during the course. The usage of the group was typically a photo of the code and the question "This won't pass the tests, what is wrong? " A Facebook group is a good tool from the instructor's point of view, because the whole answer code is difficult to share compared a special question with a photo.

The analysis of the survey data and interviews showed that the students had positive attitudes to the new way of organizing the course using MOOC. The interviews revealed for instance that the students had established a Facebook group to collaborate on the exercises of the course. The change in the instructor's total workload is significant while there are also changes in the types of work items to be performed. The programming exercises of the MOOC come attached with automated tests, and using them required a considerable amount of briefing. Also, overall guidance by email was given much more than on previous course implementations.

5   SUMMARY AND CONCLUSIONS

The main outcome of this study is proven practices of utilizing third-party MOOCs in programming education. The analysis of the survey data shows that the students had positive attitudes to MOOCs in programming education.

The results encouraged us to continue with the experiments of third-party MOOCs on our courses. Based on this case study, we expect that, by utilizing MOOCs, it will be possible to decrease the amount of human resources required in courses while introducing new creative ways to arrange high-level education for students. There is currently strict timing of exercise packages, but in the future we will offer a course without timing. Also, the survey analysis shows that if the student's major subject is something other than information technology, there is a need for a narrower programming course. From the instructor's view and for equal treatment of students, there is a need to develop the examination. Whether the development will be to arrange more exams will be the main question.

ACKNOWLEDGEMENTS

REFERENCES

[1]   I. Allen, J. Seaman. "Grade level: Tracking online education in the United States". Babson Survey Research Group and Quahog Research Group. 2014 Retrieved from http://www.onlinelearningsurvey.com/reports/gradechange.pdf

[2]   F. M. Hollands, D. Tirthali. "Resource requirements and costs of developing and delivering MOOCs" in The International Review of Research in Open and Distributed Learning. vol. 15, 2014.

[3]   P. Linna, K. Ahonen, and J. Multisilta. "Mooc Utilization in Blended Learning: Prospects for the Future From the Perspective of a Small University Unit", EDULEARN16 Proceedings, pp. 952–957. 2016

[4]   P. Linna, H. Keto, and T. Makinen. "Towards Blended Learning: A Case Study". In 2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), pp. 685–689. 2015.

[5]   P. Linna, T. Mäkinen, and H. Keto. "Utilizing MOOCs in the Development of Education and Training Programs." 2016 39th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), pp. 861–864. 2016.

[6]   P. Linna, F. Cameron, and J. Tanttu. "Experiences With Blended And Flipped Learning in University - Case Study." EDULEARN16 Proceedings, pp. 7265-7273. 2016.

[7]   J. Harjamäki and J. Leppäniemi. "Towards MOOC Based Training Programs for Small and Medium Size Enterprises." EDULEARN16 Proceedings, pp. 4873–4879. 2016.

[8]   M. Saari and T. Mäkinen. "Utilizing Electronic Exams in Programming Courses: a Case Study". In EDULEARN16 Proceedings, pp. 7155–7160. 2016.

[9]   Java programming language, www.oracle.com/java/index.html, accessed 7th May 2017.

[10]   A. Vihavainen, M. Luukkainen and J. Kurhila. "Multi-faceted support for MOOC in programming". In Proceedings of the 13th annual conference on Information technology education - SIGITE '12. Vol. 68, p. 171. 2012

[11]   Ohjelmoinnin MOOC. (Object oriented programming with Java), University Of Helsinki, MOOC In Finnish, http://mooc.fi/courses/2016/ohjelmoinnin-mooc/, accessed 11th May 2017.

[12]   Test My Code NetBeans plugin, https://github.com/testmycode/tmc-netbeans, accessed 7th May 2017.

[13]   NetBeans IDE, https://netbeans.org/, accessed 11th May 2017.

[14]   Attribution-NonCommercial-ShareAlike 4.0 International (CC BY-NC-SA 4.0) https://creativecommons.org/licenses/by-nc-sa/4.0/, accessed 9th May 2017.