The Flipped Classroom Method: Lessons Learned from Flipping Two Programming Courses

Antti Knutas, Antti Herala, Erno Vanhala, Jouni Ikonen

Abstract: *The flipped classroom teaching method, which emphasizes independent learning of theory and practical, in-depth exercises in the classroom, is gaining foothold in teaching. The method is increasingly being applied at university level. It has been implemented with varying approaches and guidelines, and a single unified process has not been described. In this article we compare existing literature to two case studies where flipped classroom was introduced to teaching. We discuss the lessons learned in these cases and present recommendations based on our experiences. Flipping the classroom has been found to be more efficient than traditional lecture-exercises method and the findings in this study support this. Therefore we recommend teachers to explore the possibility of utilizing the flipped classroom method in their courses.*

Key words: *Flipped classroom, computer science education, computer-supported education, programming*

INTRODUCTION

Since the introduction of the Internet, teaching and learning have changed significantly [9]. This has happened in all environments from elementary school to university level. The traditional university lecturing with blackboard and overhead projector is competing with online courses and digital learning material. At the same time paper exams and individual essays are replaced with online exams and peer learning. This has put pressure to teachers to improve their teaching perspective, methods and skills. At the same time the focus on research has pressured to move the responsibility of learning from the lecturer to students. The new task of the lecturer is to guide and facilitate the learning, while the students take care of their individual learning based on their own learning styles.

Flipped classroom method means teaching where students learn theory by themselves and in classroom learn by applying the previously learned theory. Recently the method has become a target of studies [2] and has been found to improve learning [7] and reduce teachers' workload [12]. However, the guidelines for its application in programming are not clearly defined. The flipped classroom method has been utilized at various levels of education [3, 11] and has been found useful for teaching programming [6]. While the method is gaining popularity, there is still a lack of guidelines on how and why the method should be applied.

In this article we first review what is the current status of flipped classroom teaching in computer science and then present two case studies where flipped classroom method was used in two advanced programming courses. The courses were taught by different teachers and they used different ways to implement the idea of flipped classroom. The goal for this study is to present the experiences on these two courses and draw on those

This is a pre-print version of an article. The actual version will be published in ACM DL at http://dl.acm.org/event.cfm?id=RE248 by late 2016. Please use the official version of the paper and the following reference when citing: Knutas, A., Herala, A., Vanhala, E., Ikonen, J., 2016. The Flipped Classroom Method: Lessons Learned from Flipping Two Programming Courses. CompSysTech 2016.

experiences to build a common baseline for applying the flipped classroom method in the teaching programming. More specifically the research question is *which parts of the flipped classroom teaching method are best suited for programming courses at university level?* The article follows with related research discussing flipped classroom method. After that the research process is discussed and cases are detailed. In discussion section the benefits and drawback of the method are presented, along with recommendations. Conclusion wraps up the study.

RELATED WORK ON FLIPPED CLASSROOMS

The concept of flipped classroom was developed in step with technological advancements. Technology has allowed the provision of wide amount of learning materials through easily accessible channels. This gives room to a philosophy where teacher's and student's contact time is aimed to be used as effectively as possible [15]. This means that theory can be studied as homework and exercises, which are traditionally given as homework, are done when the instructor is present.

The method was introduced in an economics course [7] in the early '00s and it has been used in multiple different disciplines and at different levels, such as in industrial engineering [16], biology [13], introductory business [14], mechanical engineering [12] and mathematics [1, 10].

In computer science the method has been used to teach introductory programming [6], [8] and also advanced topics, such as computer architecture [3]. The strategies for flipped classroom in computer science education has been introduced in a study by Maher et al. [11], where four different courses were reconstructed with flipped classroom. The strategies in Table 1 concentrate on three individual topics: workflow, video instructions and in-class activities. The workflow presents how the students are encouraged to study and apply the theory individually. Different approaches can be applied to video instruction creation: the videos can be created from scratch, curated from separate existing sources or from a massive open online course (MOOC). The third topic is the in-class activities, which determines the application of theory while the instructor is present.

Table 1. Three topics for flipped classroom strategies

Workflow		**Video instructions**			**In-class activities**		
Temporal flow	Preparatory work	Creating videos	Curating videos	Wrapping a MOOC	Pair programming labs	Group problem solving activities	Flexible quiz activities
Students use the online material for theory / instructions and join into group activities in class.	Students are required to do graded preparatory work before the class.	Creating lecture videos from scratch.	Collecting and curating videos from multiple sources, such as YouTube and other instructional websites.	Using a MOOC course with the permission from the publisher, providing added content for additional topics.	Pair work to complete the assigned tasks, that were constructed to be completed within one session, if the student was prepared.	Small groups, that work on problems based on the topic of the course, such as design problems.	The students complete quizzes in class either individually or in groups; the quizzes are graded or used as a base for topics of short lectures.

RESEARCH SETUP

This study uses two case studies as the main sources of data. These two cases are programming courses held in Lappeenranta University of Technology in 2014, both of which were 12 weeks long medium sized courses. Details of the courses are presented in the Table 2. The data gathered from the courses consists of grading data and the end of course feedback survey.

Table 2. Course specifics from 2014

Enrolled	Passed	Duration	Learning goals
54	26	12 weeks	"*Student learns to use object-oriented programming methods to solve typical programming problems and familiarizes himself with Java and its features in programming. Student knows how to read and describe Java code and UML diagrams.*"
69	30	12 weeks	"*The goal of the course is to introduce student to web programming, architecture and tools used in development. The course provides readiness to design and implement interactive web applications for different use.*"

Case 1: Object-Oriented Programming

The object-oriented programming course was remade in 2014 to use flipped classroom [4]. Reasons are many, but the industrial shift from Symbian to game development in Finland gave the first kick to move from C++ to Java. As the university changed semester length from fourteen to twelve weeks there was a clear room for complete improvement of the course instead of small changes once in a while. The first version of the course consisted of short lecture videos (31 items, average length of 15-30 minutes), weekly exercises, larger course project and an exam online or on paper. The course was focused solely on the object-oriented paradigm and most of the students' work contained programming in-class. There was one session in the course, where students were asked to create a design with unified modeling language (UML), in other words a design-based problem.

Physical lectures were dropped completely and only short half an hour introduction was given where the new course structure was described. After that the weekly workflow started with tasks before coming to in-class event such as reading chapters from programming manuals and watching the corresponding theory videos. After that students started the programming tasks and continued them in the exercises. Some started the work on the exercises as it was also accepted method but gave them less time to finish the tasks. The students were expected to watch the videos and read the material before starting the programming; the work was not tested with quizzes or other forms of evaluation and depended solely on students' own motivation.

Case 2: Webbed applications

Webbed applications course was changed to utilize some aspects of the flipped classroom method. The new elements include sample solutions as videos, technical learning material for independent learning and online-based peer review on project work. The overall structure of the course is a weekly cycle, with tasks introduced at the start and returned online at the end of the week. A diagram of the weekly cycle is presented in the Figure 1. During the first three weeks of the course students were given task to study basic concepts from an online service codecademy.com at a fast pace. During these weeks students were able to get help at the lectures if they had some problems understanding the material. Unlike in previous years, the course material was made immediately available at the start of the week and had small, individual tasks that controlled the learning of the material. Additionally, there were larger weekly exercise tasks with online return at the end of the week. Before the end of the week there were exercise classes with an emphasis on collaborative problem solving and discussion, with the aim of helping the students to understand the problem and solve issues together.

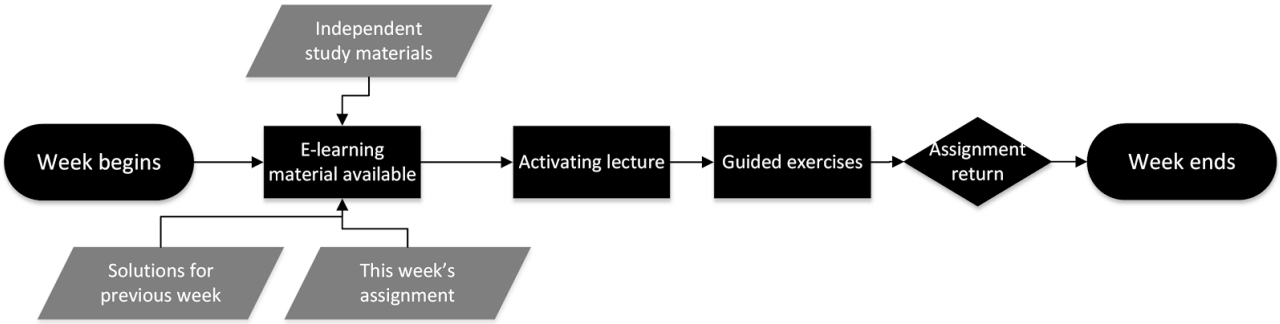

Figure 1. Weekly structure of the webbed applications course

Summary of the used flipped classroom methods

Table 3 presents a list of the elements used in the courses. The flipped classroom method focuses on the in-class session where students have active learning instead of passive one. Thus the table presents the independent work required before the classroom session and material sources as independent studying. On the other hand, in-class activities and group work methods are the other side of flipped classroom method.

Based on the strategies presented earlier, the object-oriented course used temporal workflow, the videos and materials were created from scratch and the in-class activities were pair programming labs. The webbed applications course was more complicated: concentrating on temporal workflow and flexible quiz activities for in-class teaching, but using all the presented strategies for video instructions. Some of the videos were curated from other sources, while the course also partially wrapped a MOOC. The correct answers for homework were videos created from scratch by the course assistant, covering that strategy as well.

Table 3. Elements of teaching in the flipped classroom (adapted from Maher et al. [11])

	Independent studying		**Non-independent studying**	
Course	**Work required before class**	**Sources of material**	**In-class activities**	**Cooperative methods**
Object-oriented programming	Watch videos and read corresponding parts from manuals	31 videos and two manuals created by the instructors	Solving problems with programming and design tasks	Students were encouraged to talk with each other in the exercises and project work could be done in pairs
Webbed applications	Watch the video solutions for the previous week assignments, read dedicated theory and start work on assignments	External MOOC-materials and available tutorials	Activating lectures and solving programming and design tasks in the exercises	Programming projects were peer reviewed

Feedback

The object-oriented programming course was developed from scratch and the flipping was only a part of the change. However, most of the positive student feedback concentrated on the lecture videos and the active learning environment in the exercise sessions. The lecture videos were commended on being an "*excellent invention especially in teaching programming. They made the schedule changes possible*", and they made following the lecture easier: "i*t was easier to follow the programming examples, when it was possible to test them yourself while watching the video*". In the exercise sessions the students felt that there was a "*good climate, it was not hard to approach the teacher, if there was something to ask about*".

The webbed applications course got also positive feedback and student felt that self-study and exercises are suitable methods as "*one does not learn to code while sitting in*

the lectures". Though the course was generally success, also voices were heard requiring more material and teaching to support learning. As the course was not yet completely moved to flipped classroom method, there was shortcomings on video material and readable guidelines. Notes like "*It took considerable amount of time to get all the necessary help*" and "*there should be a manual of some kind to support assignments and project*" indicate that amount of material still needs to be increased. The overall feeling was still highly positive and students felt that the course was "*one of the best ones*".

Especially the fact that programming course did not have an exam of any kind was mentioned as a positive feature (evaluation was based on two assignments, and online and class activity).

Both of the courses were evaluated quantitatively by students with a scale from one to five. The webbed applications course got an average course feedback of 4.6 (out of 5), while object-oriented programming received an average grade of 4.5. The evaluation of learning methods and tools was 4.7 to webbed applications and also 4.7 to object-oriented programming. While the quantitative feedback gives high averages, the number of students who gave feedback was significantly low (N=7 and N=19 respectively) causing the data to not have statistical significance. While the feedback does not provide significant improvements, the observations made by the lecturers and assistants of the courses provide more promising statistics. In webbed applications the percentage of passing students rose by 18%, while in object-oriented programming none of the students dropped the course after a successful submission of the project or passed exam. In previous years some of the students did not fix their project after completing the exam or did not take the exam after submitting an accepted project.

All the selected methods were accepted by students and especially the collaborative exercise lessons got a lot of positive feedback. The individual feedback gained from online peer review and especially from the online grading process was better and more personal in the students' experience, even though the online grading process used boilerplate improvement suggestions. It was experienced as personal communication directly from the teacher despite the partial automation and valued over general discussion in the classroom. What students felt that could be improved was the fragmented nature of the learning material (some students asked for a single textbook) and some more strict online exercises were considered annoying or unclear.

DISCUSSION

In the beginning we set up research question *which parts of the flipped classroom teaching method are best suited for programming courses at university level?* Here we explore answers to these questions.

Observed pros

The main observation in the object-oriented course was the high acceptance of the new learning material and the praises for the new teaching methods. Student feedback described the videos as the way of teaching in this millennium and not a single complaint or desire to return back to physical lectures were given. This was surprising as the course was completely flipped upside down changing everything; students graded the course higher than ever.

Responsible teacher of the object-oriented course reported to have spent less than 40 hours of creating videos and other materials, but the number of actual teaching hours decreased, leaving the total to approximately 115 hours. Although this was not the aim it was interesting to notice that the efficiency was also increased with the new teaching methods. The amount and depth of material could be increased without any observed negative impacts on students. Although the new setup required more initial work from the teachers, the online setup can be used with little effort in subsequent years.

In our experience even a hybrid approach utilized in the webbed applications course can provide clear benefits especially in the aspects of student satisfaction and student retention. Implementing a partial approach was shown to help struggling students to understand and apply the taught concepts. The cause was not only the increased teacher contact, the changed course structure provided more avenues for collaborative support and critical thinking instead of memorization.

The in-class activities provided more educational support, since the students were able to communicate with their peers as well as with the instructor. The in-class conversations concentrated mostly on problem solving and discussion, between peers and with instructor. In webbed applications the students also gave and received feedback about their projects from other students through peer review, giving them more personalized feel of the feedback. This would suggest, that the students should be utilized in the checking process, which would give them experience about checking programs and reading code but also gives them more feedback without increasing the workload of the instructor. This suggestion would be only applicable in programming or in similar activities.

Observed cons

Although our cases provided very little negative comments regarding flipped classroom method, some comments underline the possible pitfalls.

"*There could have been more videos about UML*", a student in object-oriented programming course.

"*I would have preferred more studying on lectures because homework and tasks were challenging. It took a lot to find information from the Internet*", a student in webbed applications course.

These kind of quotations describe the importance of appropriate self-learning material. Although students can ask questions and clarification in the exercises, the basic learning material should be satisfactory in the first place. While the videos provide the lecturer an easy method for holding the lectures, the importance of covering everything relevant should always be considered. In this light of these findings we recommended that the videos and materials are reviewed by other experts in the field before actually using them. Student feedback should be considered, but only as a secondary source. Students do not know what to expect from the material, which may leave important topics unaddressed. Additionally, the theory presented in the material should be integrated well to the course material. In the case studies not all students considered theory as an important component of the study material and used only practical example from course material and the Internet, showing clear signs of not understanding of all concepts in the exam. In hindsight activating students with graded preparatory work might have addressed this issue.

Another point of concern were the video materials. In our experience the lecturer should use care when selecting the video material and especially consider the flow between not originally related videos. In webbed applications the students complained that the material was not as coherent as possible because of the different sources. Similar experiences were reported by Baldwin [1] where students found the curated videos difficult to understand and the process of watching unpleasant. The problem with curated videos is the number of different lecturers, who teach what they see as important. Consistency is why the video creation could be more preferable method, unless the instructor will be lecturing to provide the consistency between videos.

Guidelines for applying the flipped classroom method

According to the preliminary study [5] it is recommended to start with presenting the independent study material, control learning afterwards with quizzes and then proceed with in-depth learning by guided exercise tasks in the classroom. The classroom work

worked well with a proportion of teacher-facilitated individual work and a proportion of cooperation and discussion between the students.

The split between independent learning of theory and learning practice together worked well for learning programming. Practical learning of programming centers heavily on writing code and so the time spent in classroom is filled with problem solving and programming. The learning outside classroom was focused on reading programming literature, tutorials, manuals, and watching videos. The mandatory reading and watching tasks were then controlled with quizzes that present questions on the topic of the week. The quizzes ensured that when students come to classroom they already have the basic theoretical knowledge to start working with the exercise tasks. In these courses the main task in the exercises was to solve the presented problem by programming an application.

In both courses there was a beneficial phenomenon: some of the students had started to work on the exercise before classes and some had already completed the task. We recommend that students are encouraged to start independently and then come to continue and then continue at class where they can collaborate and get guidance on finishing the week's exercise.

The following recommendations for applying the flipped classroom method in teaching programming at university level were identified from existing literature, and the pros and cons experienced in the case studies.

- Create or curate videos in addition to text-based material
    - Video curating suggested, if the instructor intends to hold small lectures
- Use weekly quizzes to evaluate the level of understanding and satisfaction
- Strictly integrate the theory and material to the course
- Encourage students to engage peers in-class and to review each other's work
- Require students to start the weekly tasks before the exercises as preparatory work

CONCLUSION

In this article we discussed the role of flipped classroom teaching method in university level programming courses. Based on the literature and experiences gained from two of our courses we presented our experiences and listed several recommendations on how to apply the flipped method to teaching CS.

Related research has pointed out several benefit gained through flipped classroom. Students can independently focus on reviewing the theory as long as they need, teachers can concentrate on helping with actual problems in the classroom and with repeated courses videos save time from lecturing. In the end what matters is that students' learning results can be improved, with our results aligning with earlier studies. We note that in the cases the computer science students accepted the flipped classroom method and gave positive feedback. While there is no a single way to implement the method, we recommend teachers to try the method while considering the lessons learned in these case studies.

Lecturing is still the de facto method of teaching in university level and more research on converting lecture-exercise-model to flipped classroom method is needed. As we are putting more and more effort on flipped classroom teaching we are also creating more research on the topic and encourage others to write on their experiences.

ACKNOWLEDGEMENTS

We thank Liikennevirasto and FUUG for supporting this study.

REFERENCES

[1] Baldwin, D. 2015. Can We "Flip" Non-Major Programming Courses Yet? *Proceedings of the 46th ACM Technical Symposium on Computer Science Education* (New York, 2015), 563–568.

[2] Bishop, J.L. and Verleger, M.A. 2013. The flipped classroom: A survey of the research. *ASEE National Conference Proceedings, Atlanta, GA* (2013).

[3] Gehringer, E.F. and Peddycord, I., Barry W. 2013. The Inverted-lecture Model: A Case Study in Computer Architecture. *Proceeding of the 44th ACM Technical Symposium on Computer Science Education* (New York, 2013), 489–494.

[4] Herala, A. et al. 2015. Object-oriented Programming Course Revisited. *Proceedings of the 15th Koli Calling Conference on Computing Education Research* (New York, NY, USA, 2015), 23–32.

[5] Herala, A. et al. 2015. Teaching Programming with Flipped Classroom Method: A Study from Two Programming Courses. *Proceedings of the 15th Koli Calling Conference on Computing Education Research* (New York, NY, USA, 2015), 165–166.

[6] Horton, D. and Craig, M. 2015. Drop, Fail, Pass, Continue: Persistence in CS1 and Beyond in Traditional and Inverted Delivery. *Proceedings of the 46th ACM Technical Symposium on Computer Science Education* (New York, 2015), 235–240.

[7] Lage, M.J. et al. 2000. Inverting the Classroom: A Gateway to Creating an Inclusive Learning Environment. *The Journal of Economic Education*. 31, 1 (Jan. 2000), 30–43.

[8] Latulipe, C. et al. 2015. Structuring Flipped Classes with Lightweight Teams and Gamification. *Proceedings of the 46th ACM Technical Symposium on Computer Science Education* (New York, 2015), 392–397.

[9] Lee, U. 1997. Using Internet Tools as an Enhancement of C2 Teaching and Learning. *Foreign Language Annals*. 30, 3 (Oct. 1997), 410–427.

[10] Lockwood, K. and Esselstein, R. 2013. The Inverted Classroom and the CS Curriculum. *Proceeding of the 44th ACM Technical Symposium on Computer Science Education* (New York, 2013), 113–118.

[11] Maher, M.L. et al. 2015. Flipped Classroom Strategies for CS Education. *Proceedings of the 46th ACM Technical Symposium on Computer Science Education* (New York, NY, USA, 2015), 218–223.

[12] Mason, G.S. et al. 2013. Comparing the Effectiveness of an Inverted Classroom to a Traditional Classroom in an Upper-Division Engineering Course. *IEEE Transactions on Education*. 56, 4 (Nov. 2013), 430–435.

[13] Moravec, M. et al. 2010. Learn before Lecture: A Strategy That Improves Learning Outcomes in a Large Introductory Biology Class. *CBE-Life Sciences Education*. 9, 4 (Dec. 2010), 473–481.

[14] Schullery, N.M. et al. 2011. Toward solving the high enrollment, low engagement dilemma: A case study in introductory business. *International Journal of Business, Humanities and Technology*. 1, 2 (2011), 1–9.

[15] Talbert, R. 2012. Inverted classroom. *Colleagues*. 9, 1 (2012), 7.

[16] Toto, R. and Nguyen, H. 2009. Flipping the work design in an industrial engineering course. *Frontiers in Education Conference, 2009. FIE'09. 39th IEEE* (2009), 1–4.

ABOUT THE AUTHORS

Antti Knutas, M.Sc., Lappeenranta University of Technology, Phone: +358-0294-462-111, E-mail: antti.knutas@lut.fi.

Antti Herala, M.Sc., Lappeenranta University of Technology, Phone: +358-0294-462-111, E-mail: antti.herala@lut.fi.

Erno Vanhala, D.Sc., University of Tampere, Phone: +358-0294-462-111, E-mail: erno.vanhala@staff.uta.fi.

Jouni Ikonen, D.Sc., Lappeenranta University of Technology, Phone: +358-0294-462-111, E-mail: jouni.ikonen@lut.fi.