The Art of Getting Our Students Involved

Wolf-Dieter Otte
Department of Computer Science, Northern Arizona University

Every professor will have had this experience: after thoughtfully taught classes an exam is written. The exam tests the material that was covered in the classes. However, a mystical "loss of knowledge" seems to set in shortly after the exam. This phenomenon gains speed at a breathtaking rate towards the end of the semester.

Why do teachers "own" the material, but only very few students? Why is it that our students more believe than know and understand? And, most importantly, what can we do about it?

In his higher-division classes, the author of this paper introduced a radical teaching paradigm shift from the traditional "teacher-lectures-class" methodology to a student-driven approach. The single intention of this change was to get students involved in the class as much as possible. The paper presented reflects on the results of this successful experiment. It describes how to create a "conversational environment" using a combination of techniques that foster discussion, thinking and understanding.

Important roles in creating a conversational environment play modern technologies like videotaping, podcasting and wikis, which are discussed in detail.

"Tell me and I will forget.
Show me and I will remember.
Involve me and I will understand."
Chinese proverb

The frustrations of everyday teaching

In the school system where I grew up, I had to learn Russian starting in 5th grade. When I eventually received my Ph.D., I had taken more than 10 years of Russian. Still, if I were to ask a Russian native speaker the time, I would hardly be able to understand the answer. I was never really motivated to learn this language. Being mostly interested in the natural sciences, I was equally weak in English. Often I received C-grades, and looking back, most of those were not even justified. I was not interested in that language until, at the age of 14, I made up my mind to become a radio officer on a merchant vessel. I was told I would have to speak English fluently, which really scared me, but I wanted to become a radio officer so badly that I took all the schoolbooks from the previous years and started to review them, book by book. Gradually, I became involved because I forced myself. Involvement became interest; and interest became true motivation. I will never forget the moment when we got our first test back after a long summer break. I received an A, the only one in the class. My strong motivation made all the difference.

When I became an educator I tried my best to pass this knowledge on to my students. But I had a very hard time figuring out actually how to do that. Like all of my colleagues, I desperately tried to make my lectures as easy to understand as possible. I “powerpoint-erized” my presentations heavily and made them available to my students electronically. I made a deliberate effort to be engaging, full of energy, entertaining and funny. But the harder I tried, the more frustrating were the results. The students seemed to be bored and didn't do well in exams. Many of them didn't even come to class, especially in the early-morning hours. The student assessments did not reflect my level of effort either. Even though my teaching evaluations scored among the highest in my department, I was constantly frustrated. It finally dawned on me, that just explaining how important motivation and involvement are is not enough. Instead, mechanisms have to be put in place to inspire enthusiasm, support and even enforce motivation and involvement.

An eye-opening experience

One day a friend and colleague of mine, who also worked in my department, quit his position unexpectedly. As this happened half way through the semester, he asked me to take over one of his classes. This upper-division class was about algorithm design and analysis, which is known to be one of the hardest classes in computer science. He told me that – out of the same frustrations that I had – he had changed his teaching style. He also invited me to continue with this new methodology, as I might find it interesting.

The first day I came to teach this class I could not believe my eyes. It was 8 a.m. and all thirty students were there. Everybody was highly awake, discussing the chapter of the book that was going to be covered. As I assigned parts of the chapter to students to present, more often than not students would volunteer to take over this or that part. Most of the presentations were of high quality, even on very demanding subject areas. Some of the students whom I remembered as mediocre students from my own classes did amazingly well, clearly demonstrating that they understood what they had read and prepared. Even more surprising, they could engage professionally in the discussions that followed each presentation. I was blown away.

The radical shift to student-centered teaching

The teaching methodology that I just outlined and eventually adopted and adapted was first introduced by Prof. Robert Lee Moore[1] about 1932, and since then, it has been applied mostly in Mathematics, mainly for proofing theorems[2]. Even though there are many parallels in thinking among engineers and mathematicians, and despite the fact that engineering disciplines employ mathematical knowledge and methodologies all the time, Moore’s method unfortunately never really found its way into Engineering.

But how does this method work? Professor Moore summarized his method in these words: “That student is taught the best who is told the least.” The method is a “commitment to teaching by letting students discover the power their own minds have.”[3] In a "classical lecture", we deprive our students of the chance to advance in any other field outside of mere "knowledge in-take". A lecture certainly demonstrates the brilliance of the teacher, but it does not directly further the brilliance of a student. Students get only very little chance to discuss, to argue, to brainstorm, or to be proud of their intellectual achievements. Struggling between a disciplined presentation of knowledge and laid-back discussions with students, the responsibility for knowledge transfer always lies with the educator. Probably the most significant difference of Moore's method compared to traditional lectured teaching is that this responsibility is shifted where it actually belongs: on the student. The class becomes a conversational environment, in which the students interact constantly. Students become active participants in the learning process, where interest and motivation are almost automatic.

During a typical class, I start out with a quiz. The quiz checks on the basic understanding of the topic that is going to be covered on that day, and also on topics covered in the recent past. After this, I hand out problems or parts of the material that the students prepared for the class. Most of the time, students volunteer, but in the end, I reserve the right to assign any part to any student. During the next couple of minutes, the students have the chance to prepare their presentations on the white board, using their notes.

Now the class actually starts. As the students take turns presenting their part of the material, all remaining students will help, ask questions, give comments, clarify or ask for clarification. Essentially, the presentations are just the vehicle, the "entry point" into interaction. Oftentimes, students get so excited that they jump up, go to the whiteboard and take over the presentation of a fellow student.

From the outside, a class in progress looks more like a competition of professionals than a class of students needing instruction. It is my responsibility as the educator to orchestrate the whole process. In the ideal case, my intervention is not required at all; indeed my presence would be unnecessary.

When I said "typical class" above, this was meant to describe what happens most of the time, however, the reality is not black and white. Oftentimes and as appropriate/necessary, I will give introductions, especially when a new subject area starts. I will contrast, correct and give perspective. In summary, I will create the background in which the individual student presentations will fit.

Normally, computer science classes are accompanied by lab assignments. In a traditional class the educator would introduce the students to practical material in the same way as theoretical instruction. Then the students would submit a programming assignment outside of the class, which is graded and returned, again outside of the class. In this approach students get a programming project assigned. They acquire the knowledge to do the project by self-study. During a lab class they present their project to the other students, discussing their program design, the difficulties they ran into and finally demonstrate the product, i.e. the compiled program.

Assessment and retention

On the surface, assessment doesn't seem to be all that different from a traditional class. There are basically three parts that can be assessed: presentations, exams/quizzes and programming assignments (lab). These three parts can be given weights, and eventually will comprise the final grade the student receives. However, the reality is much more subtle than this.

In this student-driven approach, the student doesn't actually need grades any more. Grades are a comparatively weak currency paid to a student in exchange for the demonstration of understanding the material. But the gratification and recognition a student receives in the classroom for his work are by far stronger incentives. All the time both the students and I know exactly how an individual student is doing.

Similarly, this teaching style allows an instructor to do something never allowed otherwise. Students have a right to privacy, and publishing grades that they get in exams and quizzes is not permitted. Now, the assessment of students becomes public in a very natural way, visible by everybody. Surprisingly, students take this challenge. It is my experience that they do feel comfortable with being seen and judged by their fellow students. This positive peer pressure is accepted and even desired by students; it becomes a major driving force in their personal development, helping them to rise to new levels of mastering knowledge. Once the grade privacy premise is given up, it becomes obvious what an immense hindrance it is to striving for excellence.

Likewise, grade privacy helps procrastination. Procrastination is a natural tendency in all humans – our survival mechanism to deal with unknown, scary or frightening situations. Time management seminars are specifically tailored towards graduate students and faculty to help them defeat procrastination. How can we have any reason to believe that our undergraduate students don't have this problem? But the anonymity of assessment makes it much harder to overcome procrastination. Here again, the positive peer pressure of public assessment solves this problem. In reality, it doesn't even become a problem, once the students adopt the daily routine of class preparations.

In the recent past, much has been said about grade inflation. In fact, once the students adopted this teaching methodology, grade inflation is a very natural tendency here as well. In my experience, a student prepares the material either very well or not at all. This is the reason why once a student is able to present a subject, he usually presents it with a very good result.
On the other hand, a strong polarization of results can be observed. The majority of the class is doing very well, but a small part has to struggle - not with the material, but with discipline. Most of these students will eventually drop out. I saw very intelligent students with a lack of self-discipline drop the class. But I also saw students, who were being thought of as mediocre students, press forward and excel to new levels way beyond what was thought to be possible for them.

In summary, the class retains exactly those students that – if you were a manager – you would want to employ in industry.

Technology and the role of podcasting

When I prepared this paper, I was tempted to call it the "zero technology class room”. But I use technology all the time, just in a different way. Usually, technology is being employed directly to support or even drive classroom activities. In my classroom, I use technology regularly to have students face themselves. During each class I videotape every presentation of every student. After class I transform the clips into video files and make them available. Each student has access only to his or her own video clips. The format is chosen so that students can watch them on their ipods. Usually this is the very first time in the lives of students that they get an idea about how they are performing in class. The inner self-critic is woken up, with amazing effects.

To make presentations available, I use my own server.[4] Each student has an account on that server and can access his home directory using a free client program like WinSCP. For the less technology-savvy teacher, I would not recommend this approach. Quite a few companies provide elaborate services to support podcasting in education. One example for this is Apple's iTunes-U[5], which allows anyone to make podcasts available over the Internet. Many leading universities adopted iTunes-U to publish complete courses, e.g., UC Berkeley,[6] Stanford University,[7] etc.

I also use technology to support my classes in any way imaginable. I have a web site that provides all the background information necessary to master the class. Reading assignments, lab assignments, literature references, access to podcastings, useful web links, etc., are all published on this site. Also, the teaching methodology itself requires thorough explanation. Clear rules have to be set up in order to make it work – this information is published on the web site as well.[8]

The role of an industry partner

In an earlier paper that I presented at this conference[9], I commented on the importance of an industry partnership, whenever applicable. Indeed, this student-driven teaching approach is very well complemented by the occasional presence of a representative of the industry. Not only is the latest knowledge injected into the class, it also gains in credibility, which immediately translates into higher motivation. Ideally, industry partners would come into class after the first half of the semester, when students have already gained enough self-confidence to act as equal counterparts, i.e. as professionals.

As an example, in one class of "Advanced Software Engineering", I had two industry partners come in to present on how software engineering processes work in industry and what technologies are being used in the processes. This project, which was funded by a small grant, was very well appreciated by the students.

Some practical considerations - When not to use this approach

Despite the many obvious benefits, this methodology is not a "one-size-fits-all" approach. There are a number of applicability considerations to take into account before teaching a class this way.

In the nature of this approach, it doesn't scale well. In each of my classes, on average, five to ten students present. In classes exceeding thirty students, this becomes impractical because the ratio between students who present versus those who don’t become too large. This is one reason why this style of teaching should be applied in upper division and graduate level classes only – they are usually smaller. Another reason is the level of maturity of the students. Most probably, students need to grow into taking over responsibility for their education gradually. One might expect that the dropout rate would be too high and student retention too low in lower division classes.

One time not to adopt this teaching style is when preparing a new class. On the surface, this kind of teaching seems to be much easier on the educators. In reality, they have to be very flexible, very focused and know the material "inside out". They also have to know exactly how much material can be covered over time. For a new preparation class, it might be more appropriate to "hide" behind a lecture-style class, before switching over.

An important aspect in applicability considerations is the nature of the material in a class. Content with algorithmic material – in which there is a clear emphasis of the thought process on how to arrive at conclusions and facts over the facts themselves – lends itself much better to the concept than factual content. For example, computer science classes including algorithms, distributed systems or automata theory are suitable, but not introductory programming with an emphasis on syntax.

Finally, be prepared to cover more material than you anticipated. This is because most of the time in the classroom is used to clarify and deepen the understanding of material, not necessarily to explain it from scratch. As a rule of thumb, in my classes a 2:3 ratio applies, i.e. the same amount of material I used to cover in a fifteen-week, lecture-style class I can cover in a ten-week, Moore-style class.

Summary

I found that in the student-driven, conversational environment, students truly "own" the material and their attitude towards learning changes from "I-wanna-be-entertained" to "I-take-control". In comparison to a traditional lectured class, the material is better understood, remembered and applied. Students become ready for learning in an industry environment; indeed, they become lifelong learners.

Let me conclude this paper with a quote by Bill Sanders: “Our job as teachers is to get the students to realize that they don't need teachers.”[3]

Bibliography

1. W. Ted Mahavier et al.. *A quick-start guide to the Moore method.* http://www.discovery.utexas.edu/rlm/reference/quick_start-3.pdf
2. William S. Mahavier. *What is the Moore method?* http://www.discovery.utexas.edu/rlm/reference/mahavier1.html
3. G. Edgar Parker. *Getting more from Moore.* http://www.discovery.utexas.edu/rlm/reference/parker.html
4. Wolf-Dieter Otte. *Example of a class web site.* http://flagstaff.cse.nau.edu/Courses/CS499%20-%20Enterprise%20Web%20Computing/index.html
5. Apple Inc. iTunes-U. http://www.apple.com/education/products/ipod/itunes_u.html
6. UC Berkeley. iTunes-U Portal. http://itunes.berkeley.edu/
7. Stanford University. iTunes-U Portal. http://itunes.stanford.edu/
8. Wolf-Dieter Otte. *Teaching Methodology: Rules.* http://flagstaff.cse.nau.edu/Courses/CS499%20-%20Enterprise%20Web%20Computing/Organisation/Teaching%20Methodology/index.html
9. Wolf-Dieter Otte. *Teaching Software Engineering with Industry Partnership*. 2006 ASEE PSW.