Using Open Source Software to Engage in Real-Life Problem Solving

Stewart Adam, Dawson College

ABSTRACT (Press Here for Sound)
This paper reviews the impact of computers and technology on the learning process of a 17-year-old student. It highlights the key IT experiences that transformed a curious boy into an open source software developer—one with an opinion on teaching practices. In contrast to a culture of ownership and copyright, the paper acknowledges the importance of the concept of using, remixing and sharing information to meet the needs of every learner. This concept is described as creating the pedagogy needed to motivate and engage students in the classroom.

My name is Stewart Adam and I am currently studying pure and applied sciences at Dawson College. Pure and applied science is a two-year pre-university program in Quebec that focuses on physics, mathematics (calculus) as well as chemistry and biology. I have always had a great interest for knowing how things work, which is why I like learning about science, physics and most of all, technology. We use technology daily, but we rarely get the chance to understand how its underlying components work. When my family purchased a new computer a few years ago, I was extremely excited and, over several months, I was slowly able to determine how the system worked by tinkering with it. Since then, I have started designing computer software. I would like to explain my learning experience and how I think it can be applied to the school classroom.

My fascination with computers and technology started the moment I put my hands on a keyboard at the age of four. I remember looking at the screen and being amazed at how I could tap "a" on the keyboard and a small "a" would appear on the screen. How did the computer know which key I was pressing? From that day on, I was interested in computers and more specifically, how they worked. When I was in my last year of elementary school, students were asked to design and program a small game using software called "MicroWorlds" as a computer science project. MicroWorlds is a program that uses the "logo" programming language to control one or more turtle icons on the screen. These turtle icons can be given various commands that enable them to move around, draw shapes on the screen or even change in appearance. In my game, the user navigated through a maze I had designed. If the turtle touched any part of the maze wall, the player would have to start over from the beginning. I used multiple turtles together in order to create various obstacles, such as a power-up that increased the turtle's movement speed and a moving maze wall which required precise timing to pass through. Our teacher was very knowledgeable and communicated her passion for technology through such well-designed class projects. By the end of that project, I was proud not only of being able to use a computer, but also of having designed something that other people could use. That accomplishment gave me the push to write computer software.

One of the things I learned quickly about commercial software is that in most cases, the consumer's rights are very restricted. For example, commercial software is typically distributed in a small box with the product manual inside, as well as a CD-ROM to install the program on a computer. Unfortunately, installing the program is just about all one is authorized to do. In most cases, one is not permitted to share software, view its inner working nor create new software based on the original. I found this very annoying, as I wanted to know what went on between the time I pressed the power button on the computer and when I was ready to enter my system password. However, because of the licensing terms of my operating system, reverse engineering any part of this process was illegal. I was shocked. How could it be illegal to view how something works, even if the objective is to enhance it? If I buy an old videocassette recorder, I may not have its blueprints, but nothing stops me from taking it apart. How was this not the case for computer software? I found the answer to these questions in open source software. Unlike the typical piece of commercial software, the source code (programming code used to create any software) for all open source software is published. Any programmer can examine the source code of an open source program to determine how the program functions, tweak it if so desired, or even publish his or her own program based on the original software. *Use*, *remix* and *share* captures the essence of open source in three words; one can use open source software, *remix* it to make it better for their uses, and *share* copies of it—both the original copy and the "remixes"—with anyone else.

The year after I had created my “MicroWorlds” game in grade six, I decided to install Fedora® Linux, a free and open source operating system. Similar to Microsoft's® Windows XP or Apple's® OS X, Fedora Linux let me perform all of the usual tasks like checking my e-mail or browsing the Web. However, because it was formed entirely of free and open source software, I was able to view precisely how any part of it worked! This was how one of my software projects was born. I was curious as to how an audio converter program I used at home worked, so I examined the program's source code which presented me with its logic flow statements and the various commands it executed to convert audio files. By learning from this code, I was able to create my own program which I uploaded to my Web site for others to use, share and remix. This example shows how the open source software I was using enabled limitless creativity, as any software could be remixed until it was something completely new and innovative. I had discovered a cycle of creativity and learning; a new idea that required challenging software programming led to the acquisition of knowledge and skills, which in turn enabled me to transfer this new knowledge to programming more advanced software and develop still new ideas. The user community was also a highlight of the experience. Other Linux users, like me, were volunteering their knowledge via online forums. I was able to learn something new about Linux, and then share the same knowledge with another new user from whom, in turn, I would learn something new. For instance, when I joined the forums I wanted to know how to install certain hardware drivers that would let Linux use my computer's hardware to its full potential. After reading the forum posts and finding this information, I noticed that many new users also asked this very same question. I wrote a small tutorial on installing these drivers and posted it to the forum's “how-to” section so that new users could find instructions on how to install the drivers in an easy manner. So once again, I found myself in another cycle of learning. I was constantly engaged in the wonderful online learning experience of passing on knowledge through the forums. There was such a diversity of people from all over the world because anyone could join and play a part in this community. Age and geographical location was of no importance. We were all coming to the same place to learn about Linux. For the most part, we all participated during our free time.

My terrific learning experience about Linux and open source software in general left me thinking about how we could transfer this type of learning to the classroom. Too many students perceive their classes as boring, but perhaps that can be changed completely by just modifying the manner in which the course content is delivered. The children of my generation are much more technology oriented, and I think teaching practices should take this into account. Students need to be able to *use*, *remix*, and *share* information. These three words that describe open source software are key to learning. Social learning environments are not only more enjoyable, but they can also let students engage in and take some control over their learning experience. I think students would enjoy learning and retain more content if they found it pleasurable. Students need to be able to use material, remix it to make it their own, and then share it once they have finished their work. Achieving a competency, or developing competencies in any discipline, involves learning to make appropriate choices to solve a problem and/or meet a challenge. Applying the open source principles to learning helps students do just this—acquire knowledge, transfer that knowledge to another learning situation when appropriate and to problem-solve along the way using one's personal and external resources. It gives students a sense of purpose and fulfillment which leads to further motivation to learn. Once the work (or "remix") is finished, it can be used again by the community of other teachers and students.

One example I would like to give is about waves and optics studied in physics classes. There are quite a few formulas to remember, and often the textbook questions are rather dry. For example, "if the value of *a* is this and the value of *b* is that, use this formula to calculate the value of *c*." What if the students, using the signal intensity (the "signal bars") displayed on a cell phone, decided to calculate the distance to the nearest cell phone tower? To apply abstract content to real-world problems in this manner challenges the students and gives them a sense of control over their learning. Additionally, it deepens their understanding of the physics behind the problem as they encounter obstacles in the calculations, and discover how to overcome them. The abstract content comes to life in authentic, student-created examples that develop the very competencies that teachers want their students to acquire.

Upon reflection, my learning experience with open source software has been one of engagement in real-life problem solving and knowledge acquisition that is continuously used and transferred to other problems. I think that by applying authentic problems to the classroom, students will not only enjoy learning more, but will also be proud of what they have accomplished afterwards. This was certainly the case for me.

Stewart Adam is currently attending Dawson College in Montreal. He has been creating open source software for the last five years and at the age of 14, started Diffingo Solutions Inc. In 2008, he received the Millennium Excellence Award for outstanding community leadership and in 2009, received the prestigious provincial Claude Masson award for volunteer hours committed to producing tutorials on open source software. He is the youngest community manager at Fedora-Forum, an online Linux forum with over 130,000 members.

LINK TO:
http://www.diffingo.com
http://www.firewing1.com