

# The Use of Open Source Software in Education

Jason H. Sharp

[jsharp@tarleton.edu](mailto:jsharp@tarleton.edu)

Computer Information Systems, Tarleton State University  
Stephenville, Texas 76401, USA

Jason B. Huett

[jasonhuett@charter.net](mailto:jasonhuett@charter.net)

Technology and Cognition, University of North Texas  
Denton, Texas 76203, USA

## Abstract

This paper reviews the current literature regarding the use of Open Source software in education. The material is presented in thematic order and includes a brief history of the Open Source movement and provides a general definition and philosophy of the movement. Several key areas are covered including the strengths and limitations of Open Source software, its diffusion into education, and research on its actual use in educational settings. The review concludes by providing possible ideas for the development of Open Source software and raises questions for future research.

**Keywords:** open source software, educational use, open source philosophy, pedagogy

## 1. INTRODUCTION

The purpose of this paper is to examine the literature written on the use of Open Source software in education. Several reasons have prompted educators to take a serious look at Open Source software as an alternative to proprietary software which is currently the most predominant type of software used in education. These reasons include philosophical, financial, practical, and pedagogical considerations. In order to better understand the overall concept of Open Source, a brief history will be provided, as well as a summary of its definition and philosophy. After this foundation has been established the major areas of research dealing with the educational use of Open Source will be presented. These include strengths and limitations, diffusion in education and educational uses. In conclusion, questions concerning how the transition to Open Source in education might occur are addressed.

## 2. HISTORY OF THE OPEN SOURCE MOVEMENT

Bretthauer (2002) contends that although the term itself was only coined in 1998, the roots of Open Source software can be traced back at least 30 years. For O'Dell (2004) the major stepping off point for the Open Source movement began in 1989 with the release of the World Wide Web specification by Tim Berners-Lee and the team at CERN. This was soon followed up by the release of Linux 1.0 in 1994 by Linus Torvalds and the Open Source movement was truly on its way to entering the mainstream. O'Dell sites the impact of the Internet as possibly the most important factor for the success of the Open Source movement. In fact, O'Dell states, "I think it is fair to say that without the Internet, the Open Source movement would not exist today" (p. 3).

The second factor that O'Dell attributes to its success is its economic viability which has been accomplished based on one of three

models: (1) income derived from providing support services, (2) vested interest which means by using Open Source solutions, companies save money and pass the savings on to the customer, or (3) the concept that a product will continue to be used while it has sufficient interest and a community to work on it.

Bretthauer summarizes the history of the Open Source movement by commenting that it "began as an assumption without a name or a clear alternative"; but, over the last thirty years, "it has produced some of the most stable and widely used software packages ever produced" (p. 3).

### 3. DEFINITION OF OPEN SOURCE

Although definitions vary slightly from author to author, the majority agree that Open Source software can be defined as software whose source code is freely available to the user to examine, modify, and redistribute (Bretthauer, 2002; Hart, 2003; Herbert, 2001; Kim, 2002; Moyle, 2003; O'Dell, 2004; Remidez, Laffey & Musser, 2001; Tong, 2004; Warger, 2002). Moyle adds another component to the definition by including that it is "open, unrestricted, and available by downloading from the Internet" (p. 4).

The definition for Open Source software is in sharp contrast to the definition of proprietary or closed source software, which by nature does not make the source code available. Moyle makes this contrast by including in the definition of closed source software that it "cannot be opened, viewed, customized or reauthored to meet specific requirements of individual locations nor can it be manipulated in order to fix bugs" (p. 10). Proponents are also quick to point out what Open Source is not. It is not shareware, public-domain, or freeware (Moyle; Bretthauer; EDUTECH Report, 2004). These types of software, although they may be downloaded for free, do not typically allow access to the source code, the very definition of Open Source software.

### 4. PHILOSOPHY OF OPEN SOURCE

Remidez et al. (2001) explicitly state that "the open-source model is not a set model or procedure for developing software. It is closer to a philosophy than a process" (p. 2).

At the heart of Open Source software philosophy is the concept of "free", not necessarily in financial terms, although many Open Source software applications are free of charge, but more so in the sense of freedom to examine the source code, to make modification, and to redistribute the software to others who have the exact same freedom (Hart, 2003; Moyle, 2003). In its purest form, Open Source is an approach that is more interested in serving the public good than it is in the bottom line (Hart; Moyle). The goal is to create a quality piece of software that will find broad application among users (Hart).

Ultimately, the philosophical difference between Open Source and proprietary software is the issue of control. Where by definition Open Source software allows for access, modification, and redistribution, proprietary software does not (Coppolla & Neelley, 2004). In fact the makers of proprietary software have gone to great lengths to restrict access to the source code by creating intricate and complicated license agreements. Herbert (2001) contends that Open Source as a philosophy is applicable to a wide range of technology areas.

### 5. STRENGTHS OF OPEN SOURCE SOFTWARE

#### Cost

The advocates of Open Source software in education point to its many strengths when presenting it as an alternative to proprietary software. With the rising price of proprietary software and its associated license fees, one of the primary strengths for Open Sources is its low cost of entry (Chauhan, 2004; EDUTECH Report, 2004; Moyle, 2003; Tong, 2004). O'Dell (2004) asserts that for many the decision to adopt Open Source simply boils down to the price, he states, "most budgets can handle free quite nicely" (p. 5). Gonsalves (2003) follows up this argument by commenting that "for cash-strapped schools, open source technology may offer an appealing choice over costly proprietary software" (p. 9).

Finally, Hart (2003) points to lack of money as a significant problem facing a large number of schools today. He concludes that, "a school can save a huge amount of money by using open source" (p. 12). Consequently,

the savings can be used for other technology related expenses such as hardware upgrades and staff development.

### **Quality**

Herbert (2001) goes as far to say that "it is the quality of the software that is the primary reason for the ongoing success of Open Source, not that it doesn't cost anything" (p. 4). This emphasis on quality comes from the collaborative nature of Open Source and the idea of what Eric S. Raymond (1999) calls Linus' Law which states, "given enough eyeballs, all bugs are shallow" (p. 6). That is, the more people, "eyeballs", involved in the development, the better the end product. Chauhan (2004) and Tong (2004) corroborate with Raymond in agreeing that the quality is due to the overall philosophy of Open Source which allows many skilled individuals to modify and then redistribute the code.

### **Support**

Support has always been available via the Internet through various development communities, but traditional support, similar to that provided by proprietary software distributors, in which a fee is paid is now available and growing (Hart, 2003; Herbert, 2001; Moyle, 2003).

### **Flexibility**

Gonsalves (2003) sites flexibility as a significant for Open Source in that it will effectively run on older computers ranging from 486s to early Pentium machines which are still found in many schools. In addition to hardware flexibility, Herbert (2001) points out that flexibility extends into the software area as well due to the nature of the Open Source licensure which does not lock the user into any contractual obligations.

### **Bridging the Digital Divide**

Referring back to Hart's (2003) statement above, a major problem for many schools is lack of money which creates a separation between those who have access to technology and those who do not. His assertion is that due to its lower cost of entry and maintenance, "open source does a lot to bridge this gap" (p. 15).

### **Control**

As also mentioned above, control is the primary philosophical difference between Open Source and proprietary software. Warger (2002) comments that "one of the alluring promises of open-source is return of control" (p. 19).

### **Pedagogy**

The whole philosophy of Open Source fosters the concepts of sharing, collaboration, the common good, and quality of products. Hart (2003) argues that "it is far better to teach the concepts behind world processing than it is to teach a particular program" (p. 20). Tong (2004) relates the pedagogical advantage of Open Source to the teaching of computer literacy. He asserts that "it is not important which operating system, word processor, email client, Web browser and spreadsheet used" (p. 30). What is important is the emphasis placed "on the teaching of the basic principles and concepts and avoid narrow exposure to only proprietary software from specific vendors" (p. 31).

## **6. LIMITATIONS OF OPEN SOURCE SOFTWARE**

### **Support**

One often cited limitation of Open Source is the lack of support. There is no single organization responsible for support and thus no 800 number to call when a problem is encountered (Moyle 2003; Gonsalves, 2003). A closely associated limitation is the lack of documentation in comparison to documentation available for proprietary software (O'Dell, 2004). The issue of support is not limited to external support, but also includes the local technology staff as well. Assuming that a local technology staff exists, it is not just a matter of the staff possessing the technical skills to support an Open Source software environment, but also a matter of the attitude of the staff. Individuals working in an Open Source environment must possess an attitude of collaboration and a willingness to ask questions when confronted with an unknown. Many support staff members are reluctant to do this for fear of looking incompetent (EDUTECH Report, 2004).

It has also been pointed out that although the entry cost for Open Source is low, this is not the case with total cost of ownership. This is due primarily to the fact that technical staff will require more extensive training and the time and money involved in this process may exceed the initial savings of software purchases (EDUTECH Report; Warger, 2002).

### **Negative Perception**

Another weakness that may not be as apparent is the negative perceptions of Open Source (Moyle, 2003). Many parents want their children learning the software they will be using in the "real world" and may feel their children are being placed at a disadvantage (Hart, 2003). This becomes a pedagogical issue which can be argued either as a strength or weakness (Hart; Moyle).

### **Other Limitations: Learning Curve and User Interface**

Other limitations include a steep learning curve for some Open Source applications and the fact that Open Source applications lack the slick user interfaces of many proprietary software packages (EDUTECH, 2004). Although Open Source software does possess some limitations, Gonsalves (2003) asserts that "the potential cost savings is attractive enough to consider Open Source, especially if a school has a strong technology team in place" (p. 9). Warger (2002) echoes these sentiments by stating, "despite all the obstacles, open-source has the potential to strongly influence the future of software development and support in the academic world" (p. 20).

## **7. DIFFUSION OF OPEN SOURCE IN EDUCATION**

Remidez et al. (2001) believe that the Open Source development model "provides a means for teacher educators and educational software developers in general, to create software that is sustainable, continuously improving, and sharable" (p. 2). However, they point out that currently there is not a wide-spread vehicle by which educators can share ideas and collaborate on the development of educational software. They propose that through the use of the Open Source development model, educators have the possibility of creating an environment much

like the scholarly journal system. That is, a system by which ideas can be peer-reviewed, revised, and improved upon.

Borrowing from ideas developed by Rogers in "Diffusion of Innovation" they apply the principles of relative advantage, compatibility, complexity, trialability, and observability to Open Source and teacher education software. In relation to relative advantage, Remidez et al. contend that "it is reasonable that a more attractive product will result" through the collaborative nature of Open Source development with its sharing of code and invitation to make improvements (p. 3). This collaborative aspect of Open Source development also has the potential to create projects that are more compatible with the needs of a wider range of users due to the fact that more people were involved in the process. When only a few individuals are involved in a project, they often fail to take into consideration the needs of a broader audience.

A possible weakness, however, of Open Source is tied to this concept of collaborative effort. Some time having too many opinions and ideas can make decision making too complex and overwhelm those who are involved in the process as well as those potential users of the software.

Open Source software by definition is available for download and use, if not entirely free, then typically for a reasonable cost. This allows the user to try out the software for an indefinitely time period, unlike many freeware applications, and see if it will meet the users need. If it does not sufficiently meet the users' need, the financial investment is at worst minimal. O'Dell (2004) reiterates this point by saying that "the thing I love most about Open Source is price related - its unlimited free-trial period" (p. 5).

Observability is the one area where Open Source and proprietary software share a common difficulty. Remidez et al. observe that "because software programs in general are relatively hard to observe being used, neither open nor closed source projects have a clear advantage in observability" (p. 3).

## 8. PEDAGOGICAL APPLICATION

### Pedagogical Strengths

One of the strengths of Open Source software is found in its pedagogical application. Allbritton (2003) uses Open Source software as a means to teach graduate and advanced undergraduate psychology students how to develop computing solutions for research. Rather than simply utilizing the computer as a tool by using proprietary software, the students actually write the code and build the Web interfaces for customized research applications. The uniqueness of this approach is the utilization of Open Source as the development tool. Albritton makes the assertion that this methodology for "teaching programming to psychology students may be more useful than ever, because the open-source model provides a mechanism for the entire research community to benefit from the skills that students develop" (p. 251). Several advantages of using Open Source include:

- 1) A model and a mechanism for collaborative knowledge construction.
- 2) Control, freedom, and flexibility for the instructor.
- 3) Freedom and accessibility for the students.
- 4) Increased learning opportunities for the students.
- 5) Price (p. 253-254).

These advantages parallel the strengths of Open Source software as outlined previously.

### Pedagogical Weaknesses

However, the disadvantages are pointed out as well. These include:

- 1) The instructor is also the system administrator.
- 2) Training students to use commercial software can be more immediately applicable to their work as psychologists. (p. 254)

### Philosophical Advantage

Allbritton stresses that not only are there practical and pedagogical advantages, but a philosophical reason as well, specifically that "open-Source software is created collabora-

tively and distributed freely" (p. 254). In conclusion he states that, "for institutions that see the free exchange of ideas as central to their mission, open-source software is not only a practical alternative, it is also a good fit" (p. 254). In line with this pedagogical application of Open Source software Chauhan (2004) mentions that working with Linux in an operating systems course can enhance student learning by allowing the students to actually see the code in which the operating system was written.

## 9. APPLICATION IN UNDERDEVELOPED COUNTRIES

Another educational use of Open Source software can be seen in underdeveloped countries (Halse & Terzoli, 2002; Tong, 2004). Two case studies conducted by Halse and Terzoli demonstrate the educational use of Open Source software in South Africa.

### Case 1: Nathaniel Nyaluza Secondary School

The first case chronicles the implementation of Open Source software at Nathaniel Nyaluza Secondary School in Grahamstown. At the time, the school had sixteen new computers (500 MHz Celeron processors with 64 MB of RAM) and the necessary equipment to set up a local area network. The school had two goals: (1) to print from any of the computers and charge the appropriate fee based on usage and (2) to have a centralized storage space with authentication protocols in place. In order to keep cost down, the school decided on a completely Open Source approach included the operating system, file sharing, e-mail, Internet access, system administration and printing.

### Case 2: Nombuelo Senior Secondary School

The second case took place at Nombuelo Senior Secondary School where the computers ranged from 486s to early Pentiums. Basically, the school wanted to establish a local area network and Internet access. It was found that e-mail was too difficult because of limited space on the server. In the case of Nyaluza the study shows "what can be done with the most basic entry level computers of today", Nombuelo, on the other hand, shows "what can be done with the bare minimum of computing power" (p. 5-6). The fact that the system requirements

and cost of Open Source solutions are lower is "particularly important in helping us bridge the digital divide in countries where computers are a scarce and treasured commodity" (Halse et al., p. 8).

### **10. APPLICATION IN EDUCATIONAL ENVIRONMENTS**

Dunlap, Wilson and Young (2002) illustrate how Open Source philosophy has impacted their educational environments and present three specific areas where its influence has been felt: self-publishing, knowledge sharing, and self-organized learning and performance support groups.

#### **Project 1: NOVations**

The first of the three web-based projects developed using Open Source software was an online journal entitled NOVations. This environment allowed the students a place to publish articles and book reviews and interact with other doctoral students. The students also serve on the publishing board providing them with invaluable experience of reading and critiquing other's work.

#### **Project 2: Web Resource Collaboration Center**

The second project is the Web Resource Collaboration Center which consists of a Discussion Forum, Link Manager, and Resource Construction System. Each of the tools is coded in Perl. The authors state although other tools like these exist, "the impact is in the use and integration of the tools, and the fact that they are Open Source and support learner-centered knowledge sharing" (Dunlap et al., p. 4).

#### **Project 3: Electronic Knowledge Base**

The third project is called the Electronic Knowledge Base. It was developed by a Master's program cohort group for the "common purpose of learning how to best integrate technology into their instruction, classrooms, and schools" (p. 6). The environment was built using Perl scripts and did not cost the university any additional money. The authors conclude that "the Open Source movement is a reenergizing catalyst for our reclamation of learner control. Influencing how we think about supporting collaboration, knowledge sharing, and teaching and learning in general, we are

embracing the message of Open Source with open arms" (p. 7).

### **11. COLLABORATION**

A final topic of research in the educational implication of Open Source is in the area of collaboration. This area ties back to the one of the strengths of Open Source in that communities evolve to develop, refine, and distribute quality software. Warger (2002) suggest that the academic community is an "ideal place" for experimenting with Open Source development model (p. 20).

Edwards (2001) building on the work of Haas and Holzner and Marx, has placed Open Source software development within the context of epistemic communities and situated learning. The crux of this work is that to understand the Open Source phenomenon one must also have an understanding of the communities in which the software is developed. As Edwards applies the tool of Legitimate Peripheral Participation to the context of Open Source software development communities, the idea is that the learner, the individual new to the community, must be viewed as a legitimate participant with all the rights and privileges as the other members.

Now, this does not necessarily mean the learner will have the same responsibilities or authority as insiders. What it does mean, however, is that in order for the learner to become a practitioner, the learner must be given the chance not only to observe the work that is happening in the community, but to participate in the work. Edwards asserts, "completion of this process and transformation of the learner into an insider requires that the insider *must* allow the learner legitimate peripheral participation" (p. 20).

### **12. CONCLUSION**

Tinker (2000) asserts that "software tools have revolutionized business and hold the key to breakthroughs in education as well" (p. 9). He then asks the all important question, "how are we going to produce the tools that could revolutionize education" (p. 12)? He argues that educational grants to fund this type of development are rare and that industry is not very interested in this area because of its small market and the risk involved. The bottom line for industry is

profit, and there is not a significant profit to be made from developing educational software. Tinker's conclusion is that Open Source can solve the problem and that it "provides the best way to develop the needed educational tools" (p. 12). If Open Source is the solution as Tinker and many others suggest, one looming question still arises, which is, "what is it that will motivate individuals to develop this free software?"

Borrowing from Bonaccorsi and Rossi (2003) studies in the context of a business model, several motivation factors have been identified. These include: (1) intellectual gratification, (2) demonstration of an art form, (3) pleasure of creativity, (4) notice by software firms, and (5) satisfaction of a demand. It would not necessarily be a huge logical leap to see the same motivational factors as influencing the creators of educational software. Edwards (2001) corroborates these findings by listing influence, recognition, and reputation as motivating factors. He also proposes an additional factor as personal need. Simply put, programmers truly have a need to solve problems, even if it is for free.

To reiterate, the philosophy of the entire Open Source movement is to produce quality products for the good of the community, not for the bottom line. Tinker's (2000) vision is to see an "international educational Open Source movement" that could "unleash the kind of global creativity that has created and expanded Open Source software for business" (p. 13).

In conclusion, Tinker asserts that "there is certain to be thousands of bright, creative programmers through the world who would be thrilled to have the chance to contribute to improved education everywhere" (p. 13).

Future research questions include who will step up to the challenge of developing educational software using Open Source? What will be the motivating factors for those individuals and communities? Who will develop the curriculum to accompany the software? The majority of the research contained in this literature review tends to be informational or anecdotal. A significant amount of empirical studies and actual evaluation of the effectiveness of Open Source software as opposed to proprietary software was not found.

### 13. REFERENCES

- Allbritton, David W. (2003) "Using Open-Source Solutions to Teach Computing Skills for Student Research," *Behavior Research Methods, Instruments, & Computers* (35:2), pp. 251-254.
- Bretthauer, David (2002) "Open Source Software: A History," *Information Technology and Libraries* (21:1), pp. 3-10.
- Bonaccorsi, Andrea and Cristina Rossi (2003) "Why Open Source Software Can Succeed," pp. 1-36. Retrieved November 5, 2004, from <http://opensource.mit.edu/papers/rp-bonaccorsirossi.pdf>
- Chauhan, Anand K. (2004) "Open Source and Open Standards in Higher Education," *ACET Journal of Computer Education and Research* (2:1), pp. 1-3. Retrieved November 4, 2004, from [http://www.hsu.edu/~csc\\_tjm/ACET\\_Journal/ACETJournalVol2No1\\_files/FINAL%20open\\_new.pdf](http://www.hsu.edu/~csc_tjm/ACET_Journal/ACETJournalVol2No1_files/FINAL%20open_new.pdf)
- Coppola, Chris and Ed Neelley (2004) "Open Source - Opens Learning: Why Open Source Makes Sense for Education," pp. 1-14. Retrieved October 14, 2004, from <http://www.e-learningcentre.co.uk/eclipse/Resources/open.htm>
- Dunlap, Joanna C., Brent G. Wilson and David L. Young (2002) "Xtreme Learning Control: Examples of the Open Source Movement's Impact on Our Educational Practice in a University Setting," *Proceedings of ED-MEDIA 2002 World Conference on Educational Multimedia, Hypermedia & Telecommunications*, June 24-29, pp. 2-7. (ERIC Document Reproduction Service No. ED477000)
- Edwards, Kasper (2001) "Epistemic Communities, Situated Learning and Open Source Software Development," pp. 1-24. Retrieved November 5, 2004, from <http://opensource.mit.edu/papers/kasperedwards-ec.pdf>
- Gonsalves, Antone (2003) "The Linux Alternative," *Technology & Learning* (23), pp. 9-12.

- Halse, Guy A. and Alfredo Terzoli (2002) "Open Source in South African Schools: Two Case Studies," pp. 1-8. Retrieved November 4, 2004, from [http://www.schoolnetafrica.net/fileadmin/resources/Open\\_Source\\_in\\_South\\_African\\_Schools.pdf](http://www.schoolnetafrica.net/fileadmin/resources/Open_Source_in_South_African_Schools.pdf)
- Hart, Timothy D. (2003) "Open Source in Education," pp. 1-26. Retrieved October 14, 2004, from <http://www.portfolio.umaine.edu/~hartt/OS%20in%20Education.pdf>
- Herbert, Malcolm (2001) "Open Source in Education: An Overview," Retrieved November 4, 2004, from <http://people.redhat.com/mherbert/papers/RHPaper1.pdf>
- Kim, Anne (2002) "Open Source Presents Benefits to Educators," *T.H.E. Journal* 30(1), pp. 1-3. Retrieved November 4, 2004, from <http://www.thejournal.com/magazine/vault/A4088.cfm>
- Moyle, Kathryn (2003, August) "Open Source Software and Australian School Education: An Introduction," pp. 1-49. Retrieved October 14, 2004 from [http://www.educationau.edu.au/papers/open\\_source.pdf](http://www.educationau.edu.au/papers/open_source.pdf)
- O'Dell, Robert (2002, September) "Using Open Source in Education," *T.H.E. Focus*, pp. 1-7. Retrieved November 4, 2004, from <http://www.thejournal.com/thefocus/12.cfm>
- Raymond, Eric S. (1999) "The Cathedral and the Bazaar," Retrieved September 28, 2004, from [http://www.firstmonday.dk/issues/issue3\\_3/raymond/](http://www.firstmonday.dk/issues/issue3_3/raymond/)
- Remidez, Herbert, James Laffey and Dale Musser (2001) "Open Source and the Diffusion of Teacher Education Software," Retrieved October 14, 2004, from <http://sns.internetschools.org/~schools/research/pubs/index.cgi>
- Tinker, Robert (2000) "Ice Machines, Steamboats, and Education: Structural Change and Educational Technologies," Proceedings of The Secretary's Conference on Educational Technology: Measuring Impacts and Shaping the Future, September 11-12. (ERIC Document Reproduction Service No. ED452834)
- Tong, Tan W. (2004) "Free/Open Source Software in Education [pre-publication copy]," pp. 1-55. Retrieved November 4, 2004, from <http://www.iosn.net/education/foss-education-primer/fossPrimer-Education.pdf>
- Warger, Thomas (2002) "The Open-Source Movement," *EDUTECH Report* (3), pp. 18- 20. (ERIC Document Reproduction Service No. ED479029)
- "Working With Open Source," (2004, March) *EDUTECH Report* (20:3), pp.1-7.