Experiences Teaching GIS with Open Source Software

Ian Turton,

John A. Dutton e-Education Institute
EES Building
Pennsylvania State University
University Park
PA 16802
USA
Email:ijt1@psu.edu

Introduction

This paper intends to provide a data point for geographic teaching by recounting the experiences of the author in teaching a masters level courses in the Pennsylvania State University's e-education program. The author teaches "GEOG585: Open Web Mapping", the course has been released as an Open Educational Resource and can be viewed freely on the WWW. The course teaches the principals and practice of web mapping using open source software (OSS) and open standards. In this paper OSS is referenced often it means (in this paper) software that is freely distributable and modifiable (which implies the source code must be available). The Open Source Initiative (OSI) provides a list of 10 key properties at http://www.opensource.org/docs/osd.

While the use of OSS is widely known in geographic (and most scientific fields) the field of teaching with OSS is not well documented in the literature. The area with most examples is Computer Science for example Nelson & Ng (2000) discuss the use of OSS for networking, Cagiltay (2007)Warren (2005) both show how the availability of OSS help in the teaching of software development. In other fields Cataloglu (2006) is one of the few examples where he discusses the use of OSS in Physics.

The paper will outline the software packages used in teaching the course, the types (stereotypes) of the students taking the course and the problems they experience, we will then look at the pros and cons of using open source in teaching. Finally we will conclude by considering what a prospective user of OSS in teaching should know before starting.

GIS Software used in the course

The main requirements for building an open web map are a map server (or WMS in OGC speak) and a means of drawing the map (a WMS Client). There are two main choices in map servers in the open source world, either a server written in Java or one written in C. For the C language the sole choice is MapServer, while in the Java world there are two choices GeoServer and Deegree. For thin clients (those that run in a browser) the language choice is between AJAX or Flash for modern solutions, in terms of tool kits there is OpenLayers in the JavaScript space, while Flash has WorldKit and OpenScales. For a thick client (that is a client that runs directly on the computer desktop) both well known choices are Java based: uDig and WorldWind.

For a normal (face to face) course teaching with software usually involves letting the local computer service what you will require on which lab computers by when (or installing the software yourself). For a distance education course the process is a little more difficult – each student must download and install the software on the computer they intend to use, as we will see below this leads to more excitement. As a result of this consideration the author decided to only make use of Java for it's portability and cross operating system abilities. Thus students are required to download and install GeoServer as the WMS, uDig as the thick client and OpenLayers as the thin client (thin clients run in the browser so are impertinently more portable). To make the installation of Geo-

Server more fun for the students they are required to use Tomcat (a servlet container) to host their map server[1].

Pros and cons of using open source software

The benefits of using OSS include that the software are all free (in terms of beer and speech) so there are no complex licence problems to be dealt with, this means that if the student wishes to keep and publish their work after the course ends they can. It even means that students can take the software to work and show their boss. These are all good but of course there are problems too; Students are generally unfamiliar with the software. As alluded to above students can run a variety of operating systems (Windows XP, Windows Vista, Mac OS and Linux varieties are all encountered) on their machines. Some students are using a computer provided by their employers, this leads to issues with the student not having sufficient privileges to install software.

1 Range of Students Taking the Course

The course attracts a wide variety of students who's skill level ranges from hardcore hackers to managerial types who have their email printed out (well not quite).

- **Knowledgeable Students**

  These are the students who understand how their machine connects to the Internet. They tend to ask questions about their LAMP[2] box in basement that they plan to start their new consulting business on. In general they just get on with it and help the less able students out on the discussion forums.

- **Less Knowledgeable Students**

  These are the students that know how to install software, they mostly get on with the projects but can become completely stumped if their machine is not like the example in any way. The course is developed on a Windows XP machine so the pictures of the install process are slightly different for a student using a different operating system.

- **Clueless Students**

  These are the students who are completely stumped at all times, often they have never installed anything on machine since it arrived new. In many cases this “computer illiteracy” has been trained in the to student by their IT support group and when the process is explained more slowly (along with lots of “you can't break it doing this” support) they can succeed on the course.

2 Range of Problems Encountered

There are four main steps to the use of open source programs in the course:

- **Install a program**

First the students must download the latest version of the program and install it on their machine. As mentioned above the latest version of the program is a more flexible concept for OSS than proprietary software. This could be avoided by downloading and hosting the “right” version on a university server, however OSS projects usually issue new versions to fix problems not “enhance” features so the latest version is a good thing to use. So while you can wait until the start of the next

1 GeoServer comes with a perfectly good installer which includes Jetty (another servlet container) that works with one click in windows. However if they ever apply their knowledge in the real world an understanding of Tomcat (and servlet containers in general) will be useful.

2 Linux, Apache, MySQL and PHP – a standard set of open source server software.

academic year to install the latest version of ArcMap in the labs, open source sites often hide (or remove) the older version you used to write the course.

- Use the program

  Once they are happy the server is installed and working, they have to make use of it. Some students complain that GeoServer is "harder" to use than, say, ArcView and ArcGIS Server, which have a common interchange format to help with set up. However students soon seem to get the hang of using uDig to style their data and then copying the style file to GeoServer. They also like the way you can drag data layers from USGS's site to uDig directly to build a set of WMS layers.

- **Write a program**

  The final and hardest step in the course is to develop their own thin web map client using the OpenLayers library. The aim of this step is to encourage the students to go beyond the default client provided by GeoServer and tie together several layers of data. The data can either come completely from the local server or be a combination of local and remote data. In many cases this is the first experience of programming (beyond drag and drop Arc Objects) for the students. However once they have followed a worked example and over-come their initial panic all the students have completed this portion of the course successfully.

Conclusions

As can be seen there are benefits of using OSS in your teaching which in the author's opinion easily outweigh the disadvantages of using OSS. Student feedback indicates that while the course is considered one of the hardest in the programme, they all feel they have learned something beyond button pushing and the "standard" software packages.

References

Cagiltay, N.E., 2007. Teaching software engineering by means of computer-game development: Challenges and opportunities. *British Journal of Educational Technology*, 38(3), 405-415.

Cataloglu, E., 2006. Open Source Software in Teaching Physics: A Case Study on Vector Algebra and Visual Representations. *Turkish Online Journal of Educational Technology*, 5(1).

Nelson, D. & Ng, Y.M., 2000. Teaching computer networking using open source software. *SIGCSE Bull.*, 32(3), 13-16.

Warren, I., 2005. Teaching patterns and software design. In *Proceedings of the 7th Australasian conference on Computing education - Volume 42*. Newcastle, New South Wales, Australia: Australian Computer Society, Inc., pp. 39-49.