I nternational
O pen
S ource
N etwork
An Initiative of the
UNDP's Asia-Pacific
Development Information
Programme

APDIP

Free/Open Source Software in Education

Tan Wooi Tong

Pre-publication Copy

Asia-Pacific Development Information Programme
e-Primers on Free/Open Source Software

Published by
the United Nations Development Programme's
Asia-Pacific Development Information Programme (UNDP-APDIP)
Kuala Lumpur, Malaysia

Web: http://www.apdip.net/
Email: info@apdip.net

Table of Contents

Preface

Free and Open Source Software (FOSS) is a recent phenomenon that has the potential of revolutionizing the software industry. It has already gained a strong foothold in the server software segment, with leading market share worldwide in some software categories. It is also gaining ground in desktop applications and it has been predicted that its use on the desktop will become significant in the near future.

Interest in FOSS is growing globally particularly in developing countries. Governments are considering policies to promote its use, businesses are recognizing its potential, and various sectors are giving increasing attention to the opportunity for localization that it presents.

The impact of FOSS will be felt in many areas. In this primer we focus on FOSS in education and the role it can play in schools, colleges and universities.

Information and communication technologies (ICTs) have the potential to improve the quality of education. However, educational institutions are often faced with financial constraints. Competing demands for resources and the high costs of ICTs can be a major obstacle to providing ICT facilities in educational institutions. FOSS has the potential to help lower the cost barrier by reducing the cost of software, which is an important component of ICT facilities. Besides the cost benefits, there are numerous other advantages in using FOSS in education, including pedagogical benefits.

This primer is intended to help policy- and decision-makers understand the potential use of FOSS in education—where and how it can be used, why it should be used, and what issues are involved. In particular, officials in ministries of education, school and university administrators, academic staff and researchers should find this primer useful.

This primer is part of a series of primers on Free/Open Source Software brought to you by the International Open Source Network (IOSN), an initiative of the UNDP's Asia-Pacific Development Information Programme (APDIP). We would like to thank all those who have been involved in the creation of this primer. We would also like to thank the International Development Research Centre (IDRC) of Canada for their generous financial support without which this primer would not have been written.

1 Introduction

Free/Open Source Software is software that is made freely available with the distribution of the source code as a distinctive feature. It is often available at no cost. Users can use and distribute the software. And if they so wish, they can study the source code and modify it to suit their needs. The modified version of the software can also be redistributed. In contrast, proprietary software is licensed to users for a fee and the source code is usually closely guarded and not made available to users. It is illegal to make copies and distribute proprietary software without paying additional licensing fees.

There is a fine distinction between Free Software and Open Source Software. The Free Software movement focuses on moral and ethical issues relating to the freedom of users to use, study, modify and redistribute software. Open Source advocates take a more pragmatic approach, focusing on the advantages of the Open Source software development method. For most purposes, Free Software and Open Source Software can be considered to be the same and we refer to it as Free/Open Source Software (FOSS). For more information on the general aspects of FOSS, please refer to the companion primer "Free/Open Source Software–A General Introduction"[1] which is available at http://www.iosn.net/.

FOSS can play an important role in education, especially in developing countries. The reasons for this are listed below. In Chapter 2, we go into more detail on how FOSS can be used in setting up the ICT infrastructure in educational institutions, the server software and desktop applications available, and the potential cost savings resulting from the use of FOSS. Chapter 3 focuses on FOSS for the administration of academic institutions, in particular the Library Management Systems and Learning Management Systems available. Chapter 4 looks at how FOSS can be used in the teaching of Information Technology in schools and universities. Open Content is described in Chapter 5. Although Open Content is not directly related to FOSS, it results from the application of similar principles in the publication of content and is important in education. The role of FOSS in research is covered in Chapter 6. Training and certification in FOSS is not normally part of formal education but due to its importance in building human resource capacity in FOSS, it is covered in Chapter 7. In the last chapter, we list policy issues for decision-makers to consider in implementing FOSS in education.

Why FOSS for Education?

Lower Costs

One of the main issues that policy-makers have to contend with in making decisions on the use of ICTs in education is the cost. The cost of providing communication infrastructure, computing and networking hardware and the necessary software can be daunting not only for developing countries but also for underprivileged sectors in developed countries.

FOSS can lower the barriers to access to ICTs by reducing the cost of software. The initial acquisition cost of FOSS is negligible. Indeed, it is usually possible to download FOSS without any cost. If there is limited bandwidth, it may be more

convenient to get the software in a CDROM for a nominal fee. But there is no licensing fee for each user or computer and it can be freely distributed once a copy is downloaded or made available on a CDROM. Hence, the initial cost of acquiring FOSS is much lower than the cost of acquiring proprietary software for which license fees have to be paid for each user or computer. Upgrades of FOSS can usually be obtained in a similar way, making the upgrade costs negligible as well. In contrast, proprietary software upgrades normally have to be paid even though the upgrade costs may be lower than the initial cost.

Reliability, Performance, Security

Lower cost is not the only reason why the use of FOSS for servers is prevalent. FOSS is considered to have better reliability, performance and security. The administrators of educational institutions should take these into account when making decisions on the IT infrastructure of their institutions. This is especially important in the larger institutions.

The development methodology of FOSS tends to assure high quality of the software. Bugs are rapidly removed with the help of large numbers of developers, and the resulting software is more reliable. This is especially true of the more mature FOSS for servers. For example, in a quantitative analysis of database software carried out by Reasoning Inc., it was found that the FOSS database MySQL has six times fewer defects than proprietary databases[2].

Some studies also suggest that FOSS perform better than their proprietary counterparts. For example, performance tests for file servers were carried out by *PC Magazine* in 2001 and 2002 to compare Samba running on Linux and Windows 2000 (Samba is a FOSS file server that can run on the Linux platform and work seamlessly with workstations running Windows). It was found that Samba significantly outperforms Windows 2000[3] by about 100% in the 2002 tests. Tests carried out by IT Week Labs in 2003 indicate that the later version of Samba has widened the performance gap when compared to Windows[4]. More information on other performance comparison studies is available in the paper by Wheeler[3].

It is very difficult to make comparisons between the security of FOSS and that of proprietary software. However, there have been a number of attempts to do so and these are summarized by Wheeler[3]. Many of the comparisons suggest that FOSS is often superior to proprietary software in terms of security. One of the reasons cited is the availability of the source code, which allows vulnerabilities to be identified and resolved by third parties. An independent audit of code is possible only with FOSS and not with proprietary software.

Build Long-term Capacity

There are clear indications that the use of FOSS in government, industry and other institutions is growing and that there will be a need for graduates who are familiar with FOSS. Hence, concerted efforts should be made to ensure the use of FOSS in the IT curriculum wherever possible. It is important that students are not only exposed to the predominant proprietary software but also have the opportunity to use a wider array of software, including FOSS.

Companies recognize the importance of the education market because the students of today are tomorrow's employees in the ICT sector. They will also be the users of technologies either on a personal basis or in the workplace. Hence, if they are exposed to certain products during their education, they will tend to continue to use

them in the future. For this reason, companies will go out of their way to provide incentives, such as hefty discounts, to capture the education market.

Open Philosophy

The open philosophy of FOSS is consistent with academic freedom and the open dissemination of knowledge and information common in academia. “The advances in all of the arts and sciences, indeed the sum total of human knowledge, is the result of the open sharing of ideas, theories, studies and research. Yet throughout many school systems, the software in use on computers is closed and locked, making educators partners in the censorship of the foundational information of this new age.”[5]

Computer software is often used in research work and the use of proprietary software and operating systems in such work is inconsistent with the principle of verifiability as the computation of results by closed-source software is not open to scrutiny. Because the workings of open source software can be examined, the validity of research results arrived at using such software can be verified.

Encourage Innovations

Much innovation originates from universities and many of the FOSS were initially developed in an academic environment. For example, in 1984 Richard Stallman started developing a free operating system called GNU in the Artificial Intelligence Laboratory at the Massachusetts Institute of Technology (MIT). Linus Torvalds started the work that resulted in Linux in the University of Helsinki in Finland.

An academic environment where FOSS is prevalent will encourage academic staff and students to tinker and experiment with and to participate in the development of FOSS that may eventually lead to innovative solutions.

Alternative to Piracy

Educational institutions that cannot afford to pay for licensing fees may resort to using unlicensed copies of the proprietary software. With FOSS, educational institutions can use as many copies of the software as required regardless of whether it is for academic or administrative purposes.

The use of FOSS also discourages piracy by students, many of whom can ill-afford the purchase of licensed copies of proprietary software. If proprietary software were used as the basis for teaching, students would have no choice but to use pirated copies of the software to allow them to do homework and assignments at home or on their laptop computers. In contrast, there is no restriction against making copies of FOSS for use outside institutions.

Possibility of Localization

Educational institutions in non-English speaking countries may not be able to benefit from the use of FOSS as most of the original software is developed in English. However, the open nature of FOSS is such that it can be localized. Such localization need not involve the original developer. With proprietary products, localization is constrained by commercial interests. When the size of the market is too small, there is no incentive for localizing proprietary products for that market.

Learning from Open Source Code

One of the main characteristics of FOSS is that the source code is available for users to examine and to modify. This gives students the opportunity to learn from studying high quality real-life programs. In contrast, proprietary software is normally provided in binary form and the source code is seldom released for users to study.

2 Infrastructure

Requirements of educational institutions

Different educational institutions have different ICT infrastructure requirements depending on the level of education, the nature of the courses they offer, and the funding available. Invariably, however, there is a need for computer laboratories in educational institutions for conducting basic computer classes, allowing students to do their assignments, conducting specialized IT classes, providing access to the library system, giving access to learning management systems, and email communications. The number of computers required depends on the student population and the student-computer ratio that the institution considers as desirable taking into account various factors.

Networking

Computers within laboratories are normally connected via a Local Area Network (LAN). In an education environment most of the student users are “nomadic” and do not have designated computers. Hence, the set-up must be such that they are able to work from any of the computers on the network.

Internet Connectivity

Ability to connect to the Internet is a basic requirement that educational institutions should strive to provide as it allows both students and academic staff to access the numerous digital resources available on the Web. It also enables the use of email, which has become an important means of communication. For educational institutions, the Internet is a service that facilitates effective administration of the institution and provides a channel of communication between educators and students that can lead to more effective learning. It is also necessary for the implementation of elearning and distance learning.

Security

With Internet connectivity, an institution’s network is accessible from the Internet. Thus it is essential to have a firewall to protect against intrusion by unauthorized users. This is especially so if there is high-speed 24-hour Internet access. User authentication is required to ensure that only authorized users can access the network.

Web Publishing

There are often requirements in an academic setting for publishing information and providing access to materials via the Web. These include teachers and lecturers providing course materials online, students putting up Web pages, administrators publishing information on the intranet, and the institution maintaining a public website. All these require appropriate Web server hardware and software.

Server Software

A main component of the IT infrastructure of an organization is the servers that provide various services such as email, file and print services. Appropriate software is required to provide these services. FOSS have been found to be appropriate for

this purpose. They compare favourably with the proprietary equivalent in terms of features and in certain cases have a higher market share in specific categories[3].

Email

The provision of email services requires setting up an email server that controls the sending and delivery of emails. Once such a server is set up, it is expected to run continuously without any problems. An interruption will affect communications and the work of many in the organization. Hence, reliability is a major concern for email server software. Other concerns are security and performance. Several robust FOSS email servers such as Sendmail (http://www.sendmail.org/) and Postfix (http://www.postfix.org/) are available and are being deployed successfully in many organizations, including educational institutions. They compare favourably with proprietary email servers such as Microsoft Exchange, are simpler to deploy, and require fewer computing resources.

File and Print Services

One characteristic of IT usage in an educational institution is the category of "nomadic" users—i.e., students who use the network services from different computers. This requires a file server that would allow them to save their work and configurations in a central server instead of in the local workstations. The possible need to print from any workstation they are working on also requires server software that allows this. Samba (http://www.samba.org/) is a FOSS file server that runs on Linux and works seamlessly with workstations running Windows at the same time.

Network services

The Domain Name System (DNS) is needed to translate domain names to Internet Protocol (IP) addresses. The most widely used DNS server is BIND (Berkeley Internet Name Domain), which is Free Software.

Every computer connected to a network needs to be assigned an IP address. This can be done manually but it is most common to have the IP address assigned dynamically by a DHCP (Dynamic Host Configuration Protocol) server. Most Linux distributions include a DHCP server.

Linux has a utility called iptables that can be used to implement firewalls to protect against security intrusion. Some ready-made FOSS firewalls are also available, such as Shorewall (http://www.shorewall.net/). Intrusion detection tools are used to detect any security breach and one such FOSS is a program called Snort http://www.snort.org/).

Web Server

The most popular Web server is Apache (http://www.apache.org/), which is an Open Source Software. It is reputed to have a 67% share (April, 2004) of the total Web server market worldwide (http://news.netcraft.com/archives/web_server_survey.html). It can be used to host public websites for educational institutions and to host intranets within institutions. It is often used with Linux as the operating system, MySQL as the database server and PHP as the scripting language. All of these are FOSS. Their combination is often referred to as LAMP–**L**inux, **A**pache, **M**ySQL and **P**HP (or **P**erl or **P**ython).

Other Server Software

There are numerous other Free/Open Source server software that can be useful in an education environment, such as database management systems, content management systems and mailing list servers.

The most well known FOSS database management systems are MySQL (http://www.mysql.com) and PostgreSQL (http://www.postgresql.org/). As mentioned earlier, MySQL is often used for building dynamic websites. It is suitable as a database management system for many other applications and is used in business-critical enterprise applications and packaged software. Various educational applications such as Learning Management Systems and Library Management Systems also use MySQL.

A Content Management System (CMS) facilitates the creation, publishing and management of Web content by providing a platform that can be used by individuals without their having to be skilled in the underlying technologies. Examples of FOSS Content Management Systems are PostNuke (http://www.postnuke.com/) and Plone (http://plone.org/). PostNuke is based on PHP and MySQL while Plone is based on the Zope application server which is written using the Python programming language. In an educational institution, the availability of such FOSS CMSs enables staff and students to develop Internet and Intranet websites for various purposes with relative ease.

The mailing list is an older application that enables online discussions and collaboration to take place. When an email is addressed to a mailing list, it is broadcast to the individuals subscribed to the list by email. Although newsgroups and Web-based discussion forums serve a similar function, the mailing list may still be useful in the education environment especially where Internet connectivity is still rudimentary. The more popular FOSS mailing list servers are Majordomo (http://www.greatcircle.com/majordomo/) and Mailman (http://www.list.org/).

These FOSS server applications make possible the development of a collaborative environment in educational institutions at minimal cost. They allow academic staff, students, parents and administrators to interact in a way that was not possible earlier due to technological and cost constraints.

Linux Terminal Server Project (LTSP)

Desktop applications such as a browser, an email client and a productivity suite are basic requirements in education settings. Even a small computer laboratory will require these applications to be installed in all computers. Instead of installing these applications on every workstation, it may be easier and less expensive to use “thin clients”. These are computers with a network card, graphics card, monitor, keyboard and mouse, and without a hard disk, CDROM drive, and operating system. The server handles all of the computing tasks, including the running of applications, provision of storage space and management of files. This means that cheaper hardware (or old and donated hardware) can be used for the clients. Only the server needs to be installed with the necessary peripherals and software, which means maintenance will be easier.

The Linux Terminal Server Project (LTSP - http://www.ltsp.org/) started in 1999 provides the necessary software to set up such a network of diskless workstations. It supports various Linux distributions and over 100,000 sites are using LTSP. During

boot up, the diskless workstation obtains the necessary network information from the server and the operating system is downloaded from the server. Any program supported by the server can be run from the workstation. It should be noted that a network setup using LTSP will be a Linux-only network and is primarily suited for a new installation.

Example: Kerala, India

In 2002, a project to introduce computer facilities in schools in Kannur, Kerala was initiated by the local Member of Parliament and the district administrators. To cut cost it was decided that a Linux Terminal Server Project (LTSP) solution would be used to set up the facilities.

The hardware was supplied by a state-owned enterprise and the LTSP solution was implemented by a private company. Since the project involved the installation of new hardware, it was possible to use identical computing components (computers, video cards, network cards, hubs, and other accessories), which simplified the whole setup. The LTSP server was a Pentium IV machine with CDROM drive and hard disk. The disk-less workstations were Celeron machines without any hard drive or CDROM drive. All of the computers were configured and tested off site before they were sent to the schools. Each school initially received one server and 3-5 workstations.

The server ran a customized Red Hat distribution and LTSP. The other FOSS software installed to run from the server were OpenOffice suite, multimedia and Internet applications, programming tools and other educational software.

Using the LTSP to set up the computer facilities resulted in substantial savings as more computer facilities could be set up in more schools. Forty-three government schools now have access to their own computer facilities running GNU/Linux with numerous FOSS. The teachers were given basic training on GNU/Linux systems and the computer facilities have been well received by both teachers and students.

(More information about this project is available at the following URL: http://s2s2net.netfirms.com/project.html)

Workstation Software

Although there has been increasing adoption of FOSS for desktops, its penetration is still relatively shallow. Microsoft Windows still dominates the desktop environment, with an estimated 90% market share. However, it is believed that the desktop based on Linux is now good enough for many users[6]. The increasing availability of applications on the Linux desktop with features comparable to proprietary software will only encourage more widespread adoption. A long list of FOSS desktop applications that are equivalent to the applications running on Windows is available at the following URL: http://linuxshop.ru/linuxbegin/win-lin-soft-en/table.shtml.

However, it should be noted that in order to use FOSS on the desktop it is not necessary to discard the proprietary desktop operating systems. For example, programs such as OpenOffice, Mozilla, and GIMP are available for the Windows platform. Hence, FOSS programs can be easily downloaded, installed and

experimented with without having to replace the existing proprietary operating system.

In fact, there are various options when considering the introduction of FOSS on the desktop, namely:

- retain the use of Windows and run FOSS applications for this platform;
- replace Windows with Linux;
- set up dual boot systems, which allows users to choose between Windows or Linux during startup; or
- run Windows within Linux or run Linux within Windows using appropriate software.

Productivity Suite

In educational institutions, both staff (administrative and academic) and students require the use of a suite of office productivity software consisting of a word processor, spreadsheet and presentation software. Microsoft Office is currently the most widely used productivity suite. However, the FOSS productivity suite, OpenOffice (http://www.openoffice.org/), is gaining popularity as its features are becoming comparable to the proprietary Office suite. As noted earlier, OpenOffice can run on various platforms and a complete migration to Linux before it can be used is not necessary. In fact, it can be run on Windows side-by-side with Microsoft Office. A prominent feature of the latest version of OpenOffice is the ability to export documents directly to pdf format. This feature is not available in its proprietary counterpart.

Although interoperability with the existing proprietary productivity suite is not perfect, OpenOffice is an attractive option for educational institutions. The look and feel are similar to that of Microsoft Office. In most cases, only the basic features of the productivity suite are utilized by students and staff and these are available in the OpenOffice suite.

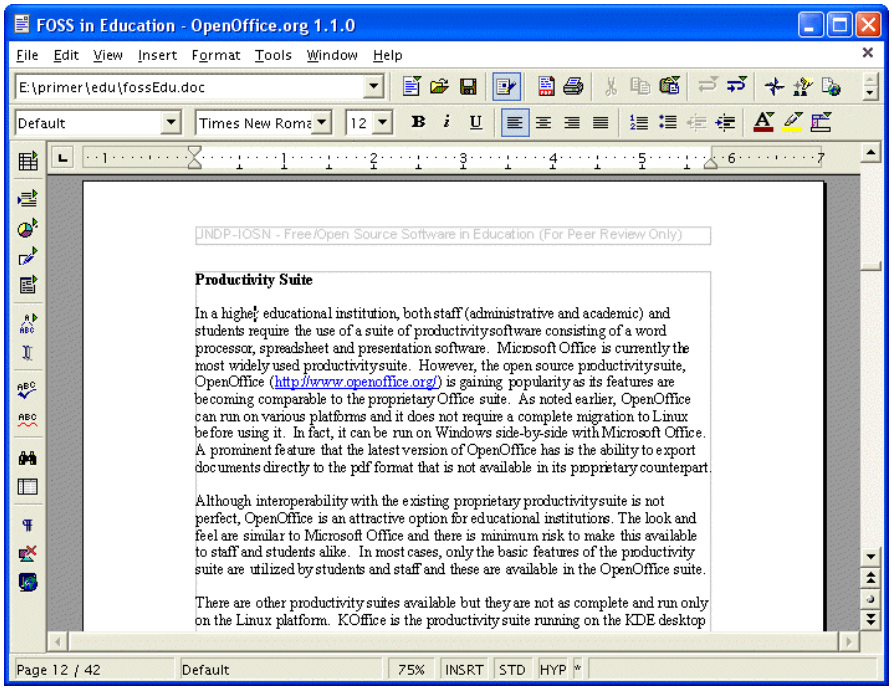

Figure 1. Word processor in OpenOffice

There are other FOSS productivity suites available but they are not as complete as OpenOffice. KOffice is the productivity suite running on the KDE desktop and GNOME Office is the suite running on the competing desktop GNOME. The word processor (Abiword) and spreadsheet (Gnumeric) components in Gnome Office are considered to be of high quality.

AbiWord (http://www.abiword.com/) works on most major operating systems, including Windows, and it supports many languages. It can read and write most documents in Word format and has the same look and feel as Word. AbiWord's native document format uses Extensible Markup Language (XML), which is an open standard. This means that an AbiWord document can be read by any other software using an appropriate XML parser.

Gnumeric (http://www.gnome.org/projects/gnumeric/) is a fast and complete spreadsheet program that is being actively developed. It can support various spreadsheet file formats and has good support for Excel files. Many of the worksheet functions available in Excel are supported in Gnumeric. However, it currently does not run on Windows. Work is being carried out to make this possible.

Web Browser

There are a number of Open Source browsers available such as Mozilla, Galeon and Konqueror. Mozilla (http://www.mozilla.org/) is a popular browser that is based on source code released by Netscape. Apart from a browser it also contains an email client, a Web authoring tool and other utilities. It is cross-platform and a version for Windows is available that can be downloaded and installed without affecting existing Internet Explorer installation. Unlike Mozilla, Galeon (http://galeon.sourceforge.net/) is purely a Web browser. It runs on the GNOME desktop. Konqueror (http://konqueror.kde.org/) is a Web browser that runs on the KDE desktop.

Figure 2. Mozilla, a FOSS Web browser

Multimedia

A wide range of multimedia FOSS is available, including graphics editors and video players that can serve as tools for enhancing educational content and its delivery.

GIMP (GNU Image Manipulation Program - http://www.gimp.org/) is the most well known FOSS for image editing and graphic design. It is a desktop application that can be used for various purposes by both academic staff and students. It is available for Linux and other Unix systems as well as Windows. As it supports various image file formats, interoperability with other programs should not be an issue. GIMP is considered to be the FOSS equivalent of the proprietary Photoshop software.

A program that allows the drawing of diagrams will find good use in an education setting. There are some good proprietary software that serve this purpose, such as Visio and Smartdraw. A FOSS equivalent called Dia (http://www.lysator.liu.se/~alla/dia/) has been designed to provide similar functions. It can be used to draw many different kinds of diagrams and has special objects to facilitate the drawing of flowcharts, network diagrams and simple circuits. The program is available for the Windows platform in addition to FOSS operating systems.

Audacity (http://audacity.sourceforge.net/) is a FOSS audio editor through which you can record sounds, play sounds, and import and export files in various formats. It can be used to edit your audio, mix tracks together, or apply effects to your recordings. This software will be useful when there is a need to digitize audio or make recordings for incorporation into multimedia educational content. It runs on most Unix systems (including Linux) and Windows.

A media player for workstations is necessary for playback of videos and other multimedia content. This is commonly available on proprietary platforms. On FOSS platforms, Mplayer (http://www.mplayerhq.hu/) is a program that provides similar functionalities. It is available for Linux and many other Unices and it supports many video and audio formats.

For a listing of other multimedia FOSS available, please refer to the following URL: http://linuxshop.ru/linuxbegin/win-lin-soft-en/table.shtml.

Other Educational Software

Aside from FOSS desktop applications for general use, there are a lot of Free/Open Source educational software that can be used for teaching specific subjects or courses in schools, colleges and universities. These range from drawing programs for young students (e.g., Tux Paint -http://www.newbreedsoftware.com/tuxpaint/) to programs for learning geometry (e.g., Kig - http://edu.kde.org/kig/), chemistry (e.g., Ghemical - http://bioinformatics.org/ghemical/) and physics (e.g., Open-Source Physics Education project - http://www.opensourcephysics.org/). For higher education, there is QCAD (http://www.ribbonsoft.com/qcad.html), a program for Computer-Aided Drafting that may be used in technical drawing classes. Scilab (http://scilabsoft.inria.fr/) is a full-featured scientific software package that may be used in numerical analysis or engineering courses at the university level.

These represent only a very small sample of FOSS available for education. There are various useful online resources available for locating other educational software some of which are mentioned below.

Schoolforge (http://www.schoolforge.net/) is a website for projects that use Free and Open Source solutions in education. It is conceived as a site where resources are made available that can help schools to develop affordable and dependable software and educational content.

SEUL/edu (http://richtech.ca/seul/) is an education portal of Simple End-User Linux promoting the use of Linux and other open resources in education. It covers various aspects of educational uses of Linux by teachers, parents and students. It has a directory of school-related Open Source software.

The Organization for Free Software in Education and Teaching (OFSET) has developed Freeduc (http://www.ofset.org/freeduc/), which provides a catalog of educational software. It has also created a live CD-ROM of FOSS for schools. The idea behind a live Freeduc system on CD-ROM is that no installation is required for it to be used and students and teachers can easily use the applications made available.

The KDE Edutainment Project (http://edu.kde.org/) aims to develop educational software for the KDE desktop. Its main focus is young schoolchildren but there are also programs that cater for university students and teachers.

Survey of OSS Use in Tertiary Institutions[7]

To gauge the extent of the use of Open Source Software (OSS) in tertiary institutions, staff of the School of Computer Science and Software Engineering of the University of Western Australia conducted a survey and reported the results in February 2004. Thirty-four tertiary institutions in Australia, New Zealand and the U.K. provided feedback for the survey. The number of systems that the respondents were responsible for ranged from 10 to 18,000.

Seventy-eight percent of the respondents reported having staff with skills in OSS. All of the institutions surveyed had deployed OSS in servers, 50% of the institutions had deployed OSS in administration, 53% of the institutions are using OSS in teaching, 56% are using it in laboratories, and 50% are using it in research.

Eighty-seven percent of the respondents said there is equivalent or better support available for OSS. Sixty-eight said that the support requirements of OSS are not higher than that for proprietary software and that OSS is sometimes easier to support.

The main benefit (84%) cited for the adoption of OSS is the lower Total Cost of Ownership (TCO) although this may not be supported by a thorough assessment of the TCO in the institution. Seventy-eight percent said the benefit of using OSS is less reliance on a specific vendor.

This survey shows that OSS has already made significant inroads into tertiary institutions in Australia, New Zealand and the U.K., with 94% of the respondents indicating that they are already using OSS.

Cost Savings

As shown earlier, FOSS in backend servers is mature and is equivalent or better than the proprietary counterpart. Applications for the desktop are increasingly available and some of them are suitable for production use. Given that this is the case, it is imperative for administrators of educational institutions, especially those that are publicly funded, to give due consideration to the use of FOSS in their institutions. Even though it is not uncommon for heavy discounting of proprietary software to be available for academic institutions, consideration should still be given to use of FOSS. In the long run the choice to use FOSS instead of proprietary software can result in bigger cost savings.

Proprietary software designed specifically for the education market is generally very expensive since vendors must recoup their development costs from very small markets. These may be administrative software such as Library Management Systems or Learning Management Systems. However, even for such specialized applications, high quality Open Source equivalents are now becoming available. In other areas such as Student Information Systems, Human Resource Management and Financial Management Systems the FOSS alternatives are not mature enough for production use. But it is likely that FOSS for these and other education-specific applications will become available in the future.

As mentioned earlier, the initial cost of FOSS is negligible. The cost of upgrades of FOSS is also insignificant. However, it has been argued that Total Cost of Ownership (TCO) should be used in making comparisons between FOSS and

proprietary software. TCO also includes maintenance, support and training costs and these may be higher for FOSS. However, in various comparisons, the TCO for FOSS is still lower than that for proprietary software.

In countries where labour costs are lower, the cost of maintenance, support and training will be a smaller percentage of the TCO, in which case the TCO of FOSS will be much more favourable. The availability of the source code also results in more companies being able to provide maintenance and support, which will drive down the maintenance or support costs. In an academic setting, there is often the possibility of getting assistance from the FOSS community without any cost involved.

Figure 3 is a comparison between the TCO of FOSS and the TCO of proprietary software.

In some situations, the availability of funds or lack of it is such that it is not a choice between proprietary software and FOSS but a choice between FOSS and nothing. By using inexpensive or donated hardware with FOSS, some institutions may be able to provide computing facilities to their students that would otherwise be impossible.

Even if funds are available for purchasing proprietary software, the savings resulting from using FOSS alternatives can be used for better purposes such as buying more computers, providing training for administrative and academic staff, or developing non-ICT related infrastructure for the institution.

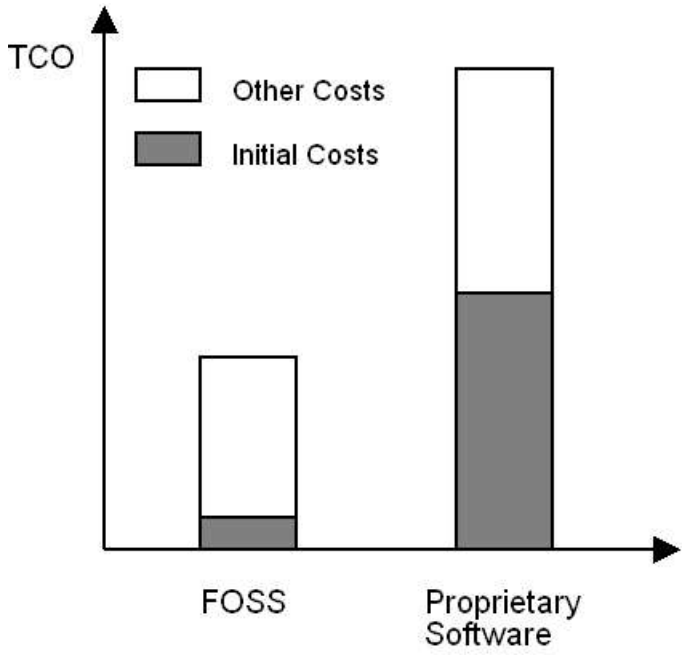

Figure 3. Comparison between TCO of FOSS and proprietary software (not based on actual figures)

Example: Goa, India

Another example of the use of FOSS technology in setting up computer facilities in schools is the Goa Schools Computer Project (GSCP). Starting from 2000, the GSCP helped to deploy 425 used computers shipped from the United States in 125 schools in Goa. The Linux Terminal Server Project (LTSP) solution was used to network the computers in the school laboratories. A cost analysis was made to estimate the cost savings resulting from the use of recycled computers and FOSS software[8]. It was found that there was a cost savings of 77% when compared to the use of new equipment and proprietary software. The cost of maintenance was taken into account in the analysis. Even if new equipment were used, there was still a savings of 64% because the new equipment was of lower specifications when used with the FOSS software.

One of the issues that arose was the need to train the computer teachers to facilitate the move to the Linux environment. But once this barrier was overcome, it was not difficult to use the FOSS desktop applications as they are quite similar to their proprietary counterparts. Another issue was lack of technical support provided by firms locally, pointing to the need for in-house staff that can provide the support internally. More details of this case study are provided by Martyris.[8]

3 Administration

Proprietary software tailored for education administration has been dominated by a small group of companies. They usually have a niche market for specialized systems such as a Library Management System or a student information system. The costs of these types of software are often very high because of the small market. Usually only the well-endowed universities or schools can afford such systems. Many schools even in developed countries are not able to afford these. Needless to say, they are beyond the reach of most educational institutions in developing countries.

In recent years, FOSS catering to this segment has appeared and in some categories like Library Management Systems and Learning Management Systems, good systems have been developed and are available for use by academic institutions.

Library Management Systems

For any school, college or university with a decent-sized library, a computerized system to automate the management of the library is essential. The most well-known FOSS Library Management System is Koha. There are other systems being developed that are not as mature as Koha, such as PhpMyLibrary (http://phpmylibrary.sourceforge.net/) and OpenBiblio (http://obiblio.sourceforge.net/). These are at various stages of development. Anctil provides a comparison of the various systems.[9]

Koha

Koha (http://www.koha.org/) was developed in 1999 in New Zealand by Katipo Communications Ltd with funding from the Horowhenua Library Trust. It is available under the GNU General Public License. It is now supported by a growing developer community from various countries and has been ported to several languages. It runs on Linux and uses the Apache Web server and MySQL database and was developed in Perl. It is also possible to use other Web servers or databases to run the program. Koha is used by the Horowhenua Library Trust for their three library branches, which have a total of 80,000 books and 25,000 patrons. Another library that is using Koha is in Coast Mountain School District, British Columbia, Canada; it has eight branches with 2,000-8,000 books per branch and 1,000 patrons.

Koha is full-featured and has modules for cataloguing, reserves, Online Public Access Catalogue (OPAC), circulation, patron management and acquisitions. The circulation component includes issues, renewals, returns and fines, and it can be set up to use bar code scanners. The acquisitions module includes budgets, pricing and supplier information.

Koha is a Web-based system where both library patrons and staff access the system using a Web browser. It provides a simple and clear interface to allow searches to be done easily via OPAC over the Internet.

The earlier version of Koha lacks some features such as support for international cataloguing standards. However, the latest version supports the international cataloguing standards *MAchine Readable Cataloging* (MARC). There are also plans to implement a serials module in future releases.

In May 2002, Koha was awarded the prize in the category for “Software for Public Administration” in an international competition organized in France. The competition was established to highlight Open Source Software that has the potential to benefit consumers and business.

Koha: Key Features

- Circulation module includes issues, renewals, returns, fines, use of barcode scanners, generation of overdue list
- Full acquisitions including budgeting, pricing and supplier information and tracking of items ordered and received
- Simplified acquisitions features for smaller libraries
- Patron (membership) management
- Reserves with possibility of self-service reservation in the library or via the Internet
- Online Public Access Catalogue (OPAC) in the library or via the Internet
- Simple and clear interface for librarians and members
- Customizable search by keyword, author, title, subject, class number or combinations
- Ability to catalogue websites
- Support for MAchine Readable Cataloging (MARC)
- Stock rotation through branch libraries

(source: http://www.koha.org/)

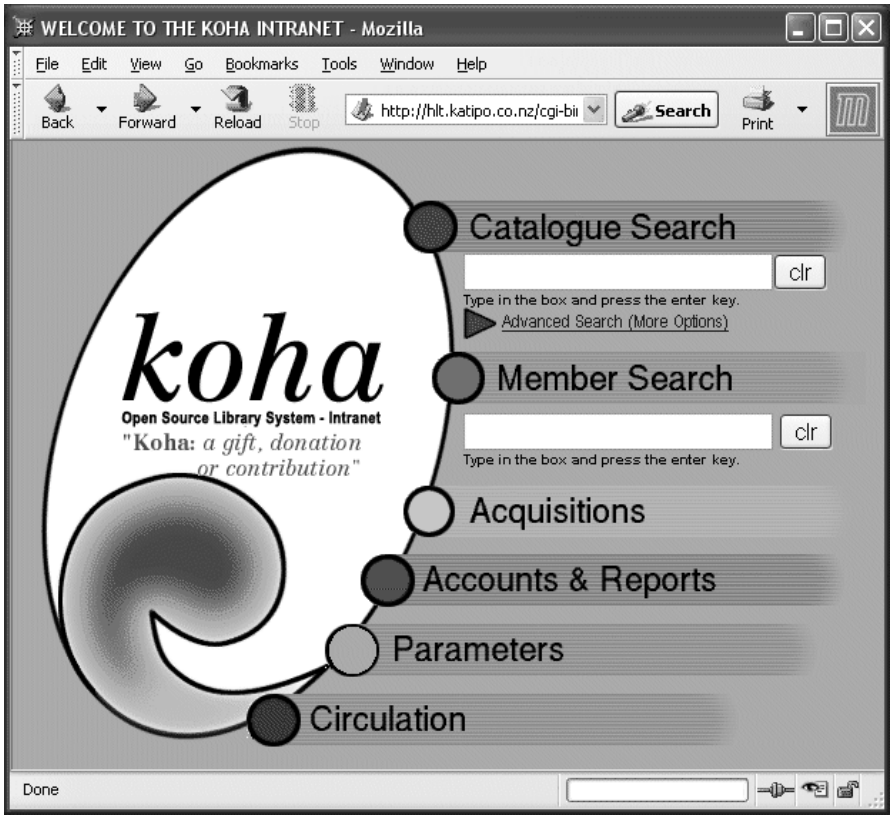

Figure 4. FOSS Library Management System - Koha

Learning Management Systems

A Learning Management System is a software application or a Web-based system that provides an instructor with a way to create and deliver online content, monitor student participation and assess student performance. A Learning Management System may also support collaboration and provide features such as chat facilities and discussion forums. Learning Management Systems are sometimes referred to as Course Management Systems.

The availability of such a system in a school or university will help to achieve the pedagogical improvements that ICTs are envisaged to bring to education. Its availability is also essential for implementing elearning. However, the existing proprietary systems such as WebCT and Blackboard are too expensive and beyond the reach of many academic institutions especially in developing countries. Fortunately, several FOSS Learning Management Systems are now available. In a report published by the Commonwealth of Learning in June 2003, 35 Open Source Learning Management Systems were identified and evaluated[10]. Ranked highest was ATutor, which will be described later in this primer.

Standards

To facilitate the interoperability of different Learning Management Systems, the content created should conform to a standard. There are a number of initiatives to establish standards for Learning Management Systems.

The Open Knowledge Initiative (OKI) was initiated at Massachusetts Institute of Technology (MIT) in 2001. It is a collaboration among a number of leading universities, with MIT and Stanford leading the initiative. This initiative has resulted in two Learning Management Systems—Stellar, developed at MIT, and CourseWork, developed at Stanford University. CourseWork was released as Open Source in June 2003.

The objective of OKI is to define an open and extensible architecture for learning technology. It is targeted specifically to the needs of the higher education community. It provides specifications for interfaces among components within a learning management system and facilitates communication with other systems, including existing enterprise systems. Commercial and non-commercial developers of products for the higher education market can use the OKI architecture which is fundamentally Open Source.

IMS (Instructional Management Systems) Global Learning Consortium is a non-profit organization supported by a worldwide consortium that includes educational institutions, software companies and publishers. It develops open technical specifications to support distributed learning. Several of these specifications are being adopted internationally as standards for learning technology. The goal of these specifications is to allow different course management systems and content from different authors to work together or interoperate. For example, content produced using a proprietary course management system such as Blackboard can be made accessible to another system such as WebCT and used in content authored in the latter system.

A specific implementation of IMS is the Sharable Content Object Reference Model (SCORM) developed by Advanced Distributed Learning (ADL). It incorporates elements from IMS and other specifications to provide elearning capabilities that allow interoperability, accessibility and reusability of Web-based learning content.

Both IMS and SCORM use XML as a common language to enable communication between disparate systems.

Stanford's Coursework

Stanford University developed its own Learning Management System called CourseWork (http://aboutcoursework.stanford.edu/) that has been used on its campus to provide instructional websites since January 2002. It has been rapidly adopted by its faculty in various academic disciplines, supporting over 400 courses with more than 12,000 users by spring of 2003.

CourseWork has been released as Open Source software, providing academic institutions non-proprietary, open access to a flexible, scalable Learning Management System. It allows institutions to integrate their course websites with other systems such as a student information system, library management system, and other education-specific infrastructure systems. The tools in CourseWork can be customized to suit the needs of a particular institution and the interface can be modified to be consistent with the institution's websites.

CourseWork is designed to be user-friendly so that academic staff can set up a course website without having to be skilled in the underlying technologies. But it has features that allow faculty with greater expertise in Web technologies to build a more complex course website.

Using CourseWork academic staff can put up announcements, syllabi and course schedules. Course materials can be uploaded to the server and managed by Coursework. Students can easily access these by following the appropriate links. Links to appropriate external resources can also be added as part of the online course readings. A discussion forum can be set up for a course to facilitate online discussions among students and with the course instructor.

Tools are available for instructors to create homework sets, assignments and quizzes. Students can submit their completed assignments online directly over the Web. Multiple-choice quizzes can be automatically graded. CourseWork manages the distribution of assignments, collection of student work and provision of feedback to students. Student grades for online work can be displayed and final grades can be computed.

Moodle

Moodle (http://moodle.org/) was first developed by an Australian, Martin Dougiamas. Moodle is an acronym for Modular Object-Oriented Dynamic Learning Environment. It is released under the terms of the GNU General Public License and currently has a very active group of developers working on it. Developed using PHP, Moodle is cross-platform: it runs without modification on Linux, Windows, Mac OS X and any other system that supports PHP. It supports the FOSS databases MySQL and PostgreSQL and can also be used with other databases. Moodle is multilingual and is currently available in 34 languages, including Chinese, Indonesian, Japanese and Thai.

Moodle has numerous features for site management, user management and course management. It has modules for assignment, quiz, discussion forum and chat. Instructors can put up assignments with due date and maximum grade, allow students to upload completed assignments, and provide feedback to students on their assignments. Various types of quiz questions can be created using Moodle—multiple choice, short answers, true/false and fill-in-blanks. The quiz can be graded automatically and feedback can be given to explain the answers. The questions can be shuffled to make it more difficult for students to cheat, and the maximum number of times students can take the quiz can be specified. All grades for assignments and quizzes can be viewed and downloaded for further processing if necessary.

Discussion forums can be set up using Moodle to facilitate student interaction in a particular course, with the option to email copies to the students. Moodle also has a chat function for live discussions. Chat sessions can be logged for later viewing by both staff and students.

Moodle: Key Features

Overall design

- Suitable for courses conducted entirely online as well as for supplementing classroom learning
- Uses a simple browser interface
- Easy to install on almost any platform that supports PHP and requires only one database
- Full database abstraction supporting all major database systems
- Emphasis on strong security

Site management

- Plug-in "themes" allow the site to be customized
- Plug-in activity modules can be added to existing Moodle installations
- Plug-in language packs allow full localization to any language
- The code is written in PHP under a GPL license which allows modifications to suit particular needs

User management

- Supports a range of authentication mechanisms allowing easy integration with existing systems
- Each person requires only one account for the whole server
- Teachers may have editing privileges removed so that they cannot modify the course (e.g., for part-time tutors)
- As a security feature teachers can add an "enrolment key" to their courses to keep out non-students

Course management

- Choice of course formats such as by week, by topic or a discussion-focused format
- Flexible array of course activities available via various modules--Forums, Journals, Quizzes, Resources, Choices, Surveys, Assignments, Chats, Workshops
- All grades for Forums, Journals, Quizzes and Assignments can be viewed and downloaded as a spreadsheet file
- Full user logging and tracking with activity reports for each student
- Teachers can define their own scales to be used for grading

(source: http://moodle.org/)

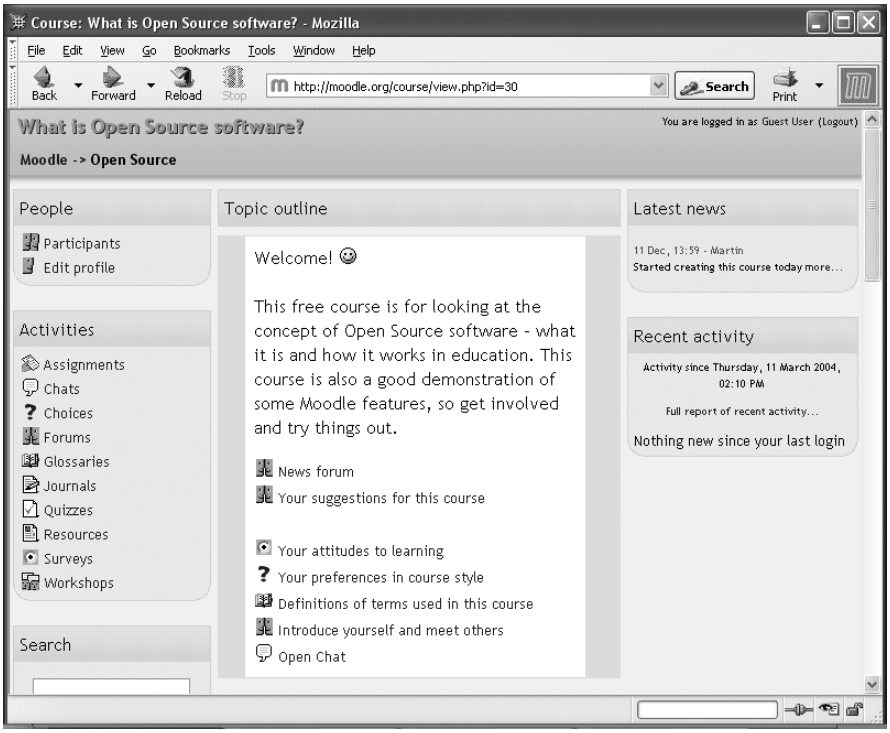

Figure 5. Moodle, a FOSS Learning Management System

ATutor

The Commonwealth of Learning report[10] evaluates Open Source Learning Management Systems and recommends two products. The criteria used for evaluation include features and functionality, cost of ownership, maintainability, usability, standards compliance and scalability. The report's top recommendation is ATutor.

ATutor (http://www.atutor.ca) has a variety of tools to allow the instructor to manage the online content, such as a built-in content editor, a resources database, a forum manager, course statistics and assessment support. It has good collaborative support with standalone modules for collaboration and chatting. It provides good documentation and online help with a tutorial for new users. However, the user interface may not be very intuitive.

ATutor supports IMS/SCORM specifications, allowing content to be imported from and exported to other Learning Management Systems that also conform to IMS/SCORM specifications. However, it is relatively new and does not have a large installed base. ATutor is released under the GNU General Public License.

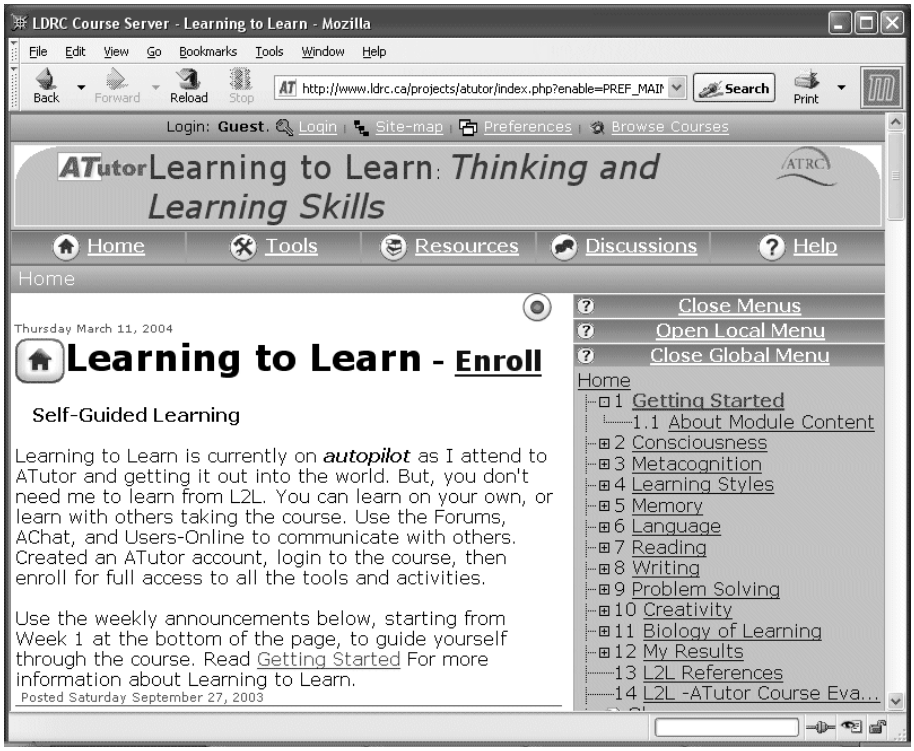

Figure 6. ATutor, another FOSS Learning Management System

Others

Among the other requirements for the administration of educational institutions is a Student Information System for the management of student records and subject offerings, timetabling, registration, management of academic and financial records, and so on. As Student Information Systems are often customized to the needs of particular institutions, there are not many proprietary systems available. So far, there is no production quality Open Source student information system available. However, there are some initiatives in developing such a system.

The SchoolTool (http://www.schooltool.org) project is one such initiative. The objective of SchoolTool is to develop a system for school administration that can be used globally and which is suitable both for schools and for higher educational institutions. It would incorporate best practices in school administration and would be easily customized for local needs. The system will be made available under an Open Source license. This project was started in South Africa in 2000 with funding from the Shuttleworth Foundation. The project was temporarily suspended in 2002 but was restarted in 2003. It is still in the early stages of development and the software is not yet available for use.

4 Teaching IT with FOSS

In the earlier sections we looked at the use of FOSS in the setting up of the IT infrastructure of educational institutions and its use in administrative functions. In this section, we examine the role of FOSS in the teaching of Information Technology (IT). Most software currently used in teaching IT, including basic productivity software for teaching computer literacy, compilers for programming courses and relational database management systems, are proprietary. However, there are FOSS equivalents available that can be suitable replacements. In addition to the cost savings, there are other advantages to using FOSS in the teaching of IT.

Computer Literacy

At the very basic level, teaching IT involves teaching computer literacy. Computer literacy means having acquired the skills to make use of the computer for common tasks. It usually implies competency in using common desktop applications such as a word processor, spreadsheet, email client and Web browser. The skills in Table 1 are considered essential. The order of the skills in the list is not indicative of their relative importance and will change with time as technologies change; the relative

Table 1. IT Skills

1. Setting up a personal computer
2. Using basic operating system features
3. Using a word processor
4. Using a graphics and/or artwork package
5. Connecting a computer to a network
6. Using the Internet
7. Using a computer to communicate with others
8. Using a spreadsheet
9. Using a database system
10. Using instructional materials for new applications

(*source: U.S. National Research Council's Committee on Information Technology report*[11])

importance of skills also varies depending on individuals.

Computer literacy should be taught not only to students in schools but also to university students who may not have acquired the skills associated with computer literacy in their earlier schooling. It is a common assumption that university students today should be computer literate and have the skills to use desktop applications for their academic work regardless of the field of study they are in. In some universities computer literacy courses are offered to ensure that the students have these skills. In some cases, there may not even be formal courses and students are simply expected to learn on their own. Whatever the case, computer literacy programmes

usually use the dominant proprietary software. Even at the lower education levels it is not uncommon to find students being taught to use Microsoft Windows and Office as part of computer literacy courses.

There are some problems with this method of teaching computer literacy. The first is that skill in the use of a particular version of proprietary software is usually short lived. Even though it will be easier to learn how to use a new version of the software from the same vendor (relative to learning an entirely new software), re-training will still be necessary unless the user has the ability to self-learn. A different approach to teaching computer literacy should be used in order to equip students with the ability to learn, unlearn and relearn. The emphasis should be on generic skills that should not be dependent on software from a specific vendor.

The second problem with using specific proprietary software in the computer literacy curriculum is that it encourages illegal copying of licensed software. While the students are studying, the need to use the same software as that which is available in their schools or universities for doing their homework and assignments would lead many of them to use pirated copies at home or on their laptop computers. Schools and institutions that have financial constraints may even go to the extent of using unlicensed copies of proprietary software in their enthusiasm to provide computer literacy training to their students.

In teaching computer literacy it is not important which operating system, word processor, email client, Web browser and spreadsheet are used. Linux, together with the appropriate Graphical User Interface (GUI) such as GNOME or KDE, is a FOSS operating system that can be used to teach the basics of operating system features.

OpenOffice has word processor, spreadsheet, presentation and drawing programs that can replace the proprietary equivalents. These software should be sufficient for teaching the basic features available in office productivity software. Other FOSS software such as the Abiword word processor or the Gnumeric spreadsheet can also be used in a computer literacy curriculum if necessary.

To teach students how to access the Web or to use email, the FOSS Web browser and email client that is part of Mozilla can be used. Again, the features of Mozilla are comparable to the browser and email client that come together with Windows and should be sufficient for use in a computer literacy curriculum.

The FOSS database systems MySQL and PostgreSQL are full-feature systems that can certainly be used to teach the basics of database systems. The GUI available for these databases may not be as user-friendly as the proprietary equivalent but it should not be an obstacle to learning the basic principles that should be taught.

Michael Surran, a Computer Science teacher in the US states the following[12]:

> *People sometimes ask me, ``Is teaching our students Linux preparing them for the workplace?'' This question is based on the fact that Microsoft is the current dominating presence in operating systems and office software. It is a question I have thought over a long time, and the answer I always come up with is, ``Yes, most definitely.'' The basic principles of any type of operating system, office application or other similarly grouped software are the same. A student who becomes proficient in Linux will not find themselves (sic) lost in a Windows environment. I have found Linux to be the more advanced of the two operating systems, yet our students are very quickly and easily learning*

it. The process of copying a file or formatting a paragraph is not so different between one operating system and the other.

Using FOSS software as the basis of the computer literacy curriculum also results in cost savings for the school or university. It obviates the need to ensure that sufficient licenses are purchased as FOSS software can be legally installed in as many computers as necessary. Students can also install the FOSS software in their own computers without restrictions, and piracy of proprietary software is not necessary to ensure access to software for their academic work outside the school or university's premises. More importantly, it would encourage placing emphasis on the teaching of the basic principles and concepts and avoid narrow exposure to only proprietary software from specific vendors.

However, it may be necessary to train teachers and lecturers in the use of Linux and FOSS before they can be competent enough to conduct classes using these software. Modification of the curriculum is necessary and some effort will have to be put into developing suitable teaching materials.

Schools

Computer literacy is usually the main focus of teaching IT to students in schools. This would equip them with the ability to use computers to enhance their learning, access the Internet, use email, and so on.

But as emphasized earlier, this should not be confined to teaching students to use specific proprietary software. Students who have not been exposed to computers are likely to be more receptive to Linux and FOSS and it would be desirable to start using FOSS as early as possible.

For example, for pedagogical reasons FOSS was introduced into some non-governmental schools in Australia. The students in these schools developed the ability to use IT without assuming that computing can be based only on one predominant computing platform. Teachers at the Sydney Church of England Girls Grammar School (SCEGGS) deliberately expose students to more than one suite of office applications, giving them the opportunity to use both FOSS and proprietary software and increase their understanding of the principles of these applications[13].

Apart from using FOSS for teaching IT, there is a lot of FOSS available that can be used to teach non-IT subjects in schools. Some of these have been mentioned in Chapter 2 and the online resources where more information on such educational software can be found are also given.

Higher Education

Programming

Learning how to write computer programs is invariably a part of Computer Science or Information Technology programmes. It is often taught to students in other disciplines as well since it may be necessary for them to develop some computer programs for their projects or research work.

At a fundamental level, the choice of programming language to be used as the basis for teaching programming concepts is not important. There are numerous computer languages available on the Linux platform that can be used for this purpose. The GNU Compiler Collection (GCC) is a collection of programming language compilers that is included in most Linux distributions. It currently supports computer languages such as C, C++ and Java. Work is in progress to include other languages such as Pascal and Cobol. There are also other computer language compilers available such as for dialects of BASIC.

Many educational institutions currently base their programming courses on the Windows platform, resulting in a dependence on proprietary programming tools. These tools are usually in the form of an Integrated Development Environment (IDE) that simplifies some of the tasks involved in programming. However, there are IDEs available for the FOSS platforms such as KDevelop (http://www.kdevelop.org/), and if it is necessary to expose students to the use of IDE for programming these can be used instead of the proprietary tools.

Availability of Source Code

At the more advanced level, the teaching of programming will be facilitated by the availability of source code of FOSS. The best way to learn how to write good computer programs is to study what has been written by others, especially real-life high quality programs. Take writing a novel: a writer cannot be expected to write a good novel if he/she has not read numerous works of others. In contrast, programmers are expected to write programs by just mastering the syntax and construction of a particular computer language without having much opportunity to study good code written by others.

In the past there has been a lack of real-life high quality source code available for all to look at and to study. The source code of proprietary software is protected as a trade secret and is seldom released publicly. FOSS has made available the source code of thousands of programs, some of which are market leaders in their particular market segment. The importance of the availability of source code of FOSS and its role in building capacity in software development is still not widely recognized.

Most users of FOSS do not have an interest in looking at the source code and neither is it necessary for them to do so. However, where students are being taught programming, the availability of the source code should be utilized to enhance teaching and learning. This is analogous to the sudden availability of great novels for reading to students in a writing class where there was none previously. For example, in programming projects undertaken for IT courses students can be encouraged to study the available source code and to build on it by adding features or making improvements. If they are sufficiently prepared, they can even participate

in various FOSS projects such as those hosted in Sourceforge.net by submitting their code contributions.

Programming Languages Used in FOSS

Although at the basic level it is not important which programming language is used as the basis of teaching, at the more advanced level institutions should equip students with competency in the use of programming languages relevant to their future work. If there is a likelihood that students who take up a career in software development in the future will be involved in FOSS projects, then the curriculum should be designed to take this into consideration.

Table 2 lists the computer languages used in open source projects included in the SourceForge (http://sourceforge.net) repository as of December 2003.

Table 2. Computer languages used in Open Source projects

Programming Language	Number of projects	Percentage of projects
C	12,329	19.3
C++	12,173	19.0
Java	10,594	16.5
PHP	8,023	12.5
Perl	5,141	8.0
Python	2,873	4.5

(Source: http://sourceforge.net)

There are many more languages being used to develop Open Source software. We have included only the top few in the list above. C and C++ are predominantly used in Open Source projects. However, the use of Java, PHP, Perl and Python is growing. C and C++ are already commonly taught in the traditional Computer Science curriculum. With the growing importance of Java, PHP, Perl and Python, there should be an increased effort to include these in the IT curriculum.

Software Engineering

FOSS Development Methodology

With the increased importance of FOSS, the teaching of software engineering should be modified to take into account the processes and methodology used in developing FOSS.

In his essay "The Cathedral and the Bazaar", Eric Raymond describes the characteristics of FOSS development and explains the metaphors he used to compare the methodologies used in traditional software development (cathedral model) and FOSS development (bazaar model) as follows:[14]

> *I believed that the most important software (operating systems and really large tools like the Emacs programming editor) needed to be built like cathedrals, carefully crafted by individual wizards or small bands of mages working in splendid isolation, with no beta to be released before its time.*

> *Linus Torvalds 's style of development—release early and often, delegate everything you can, be open to the point of promiscuity—came as a surprise. No quiet, reverent cathedral-building here—rather, the Linux community seemed to resemble a great babbling bazaar of differing agendas and approaches (aptly symbolized by the Linux archive sites, who'd take submissions from anyone) out of which a coherent and stable system could seemingly emerge only by a succession of miracles.*

A FOSS project tends to be initiated as a result of the needs of the developer or developers. They will then develop the software to a stage where it is usable with the basic functionalities before releasing it to the community as an Open Source project. The program can be bug-ridden, incomplete and poorly documented. But it must at least run and convince potential contributors that it has the promise of evolving into a good piece of software in the near future. This will attract other developers interested in solving the same problem to contribute to the project. However, it is not always the case that the development starts from scratch. In fact, many of the successful projects such as Apache are built on existing partial solutions. It should be noted that a project cannot be initiated from scratch in a bazaar mode.

Unlike a proprietary software project, a FOSS project is released as soon as it is usable and updated versions are released whenever there are modifications to the software. "Release early, release often" is an important characteristic of a FOSS project. By doing so, many more users can participate in the testing of the software under varying conditions. If there are any bugs in the software, they will be quickly discovered and fixed. The availability of the source code allows the users to be co-developers in the sense that they can fix the bugs instead of only reporting them. In contrast, for proprietary software it takes much longer for the small group of paid developers to fix the reported bugs through a beta-testing process.

This model of development is not entirely without structure especially for the larger projects. For example, Linux has a hierarchical structure based on the delegation of responsibilities by Linus Torvalds to different people. They each have responsibility and authority for different parts of Linux although Linus Torvalds will always have the last word. The high degree of modularity of FOSS projects has given rise to a specific organizational and decision-making process. Generally the FOSS development model tends to be more informal since the developers perform their tasks voluntarily and for no direct financial gain.

It is now quite common for FOSS projects to rely on tools for issue tracking, source code management, design, automated testing, and packaging and deployment. The concurrent versions system (CVS) is the most widely used version control system in open source projects. Its features include a central server containing the latest versions of the software that makes them accessible to anyone over the Internet. The CVS can be configured to send email notifications to project developers whenever changes are made so that the new source code can be tested and reviewed. Bugzilla was developed for use in the Mozilla project for issue tracking and technical support. It is now used in many FOSS projects.

The FOSS development methodology is different from traditional software engineering in many ways. Some researchers who are studying it are finding that FOSS development methodology “can be faster, better and cheaper than the textbook software engineering often used in corporate settings.”[15] However, they are not concluding that very large and complex software applications customized for a small market such as the aviation industry can be developed using the open source model; for these traditional software engineering approaches are still required.

5 Open Content

The success of the Open Source phenomenon has prompted efforts to apply similar principles in publication of content. The idea behind publishing Open Content is that anyone can use the content, distribute it freely, modify it and redistribute it. In this way, the content can be improved upon and knowledge is made freely available for the common good. The term “Open Content” was coined by Dr. David Wiley who launched the OpenContent project in 1998 and provided the Open Content License. The Open Content License is now superseded by the Creative Commons licenses.

Creative Commons (http://creativecommons.org), which is based at the Stanford Law School, provides various options for licensing Open Content. It should be noted that publishing content using one of these licenses does not mean that the author is giving up copyright to the work. Rather, some rights are offered to users of the work under certain conditions. The various Creative Commons options are summarized below:

1. **Attribution** – Gives permission to copy, distribute, display, and perform work and derivative works based upon it but only if credit is given.

2. **Noncommercial** – Gives permission to copy, distribute, display, and perform work and derivative works based upon it but for noncommercial purposes only.

3. **No Derivative Works** – Gives permission to copy, distribute, display, and perform only verbatim copies of work but not derivative works based upon it.

4. **Share Alike** – Gives permission to distribute derivative works only under a license identical to the license that governs the original work.

These license options include a common set of other rights and restrictions.

Open Content is particularly important to education and there are a number of initiatives to provide Open Content for educational use, the most notable of which is the OpenCourseWare initiative by MIT.

MIT OpenCourseWare

In April 2001, MIT announced the OpenCourseWare (OCW) project through which it will make available course material used in 2,000 of the courses taught at MIT. These will be available online and educators, students and self-learners from anywhere in the world can access the material without any restrictions. In May 2004, materials for 700 courses from virtually all academic disciplines were published on the OCW website (http://ocw.mit.edu).

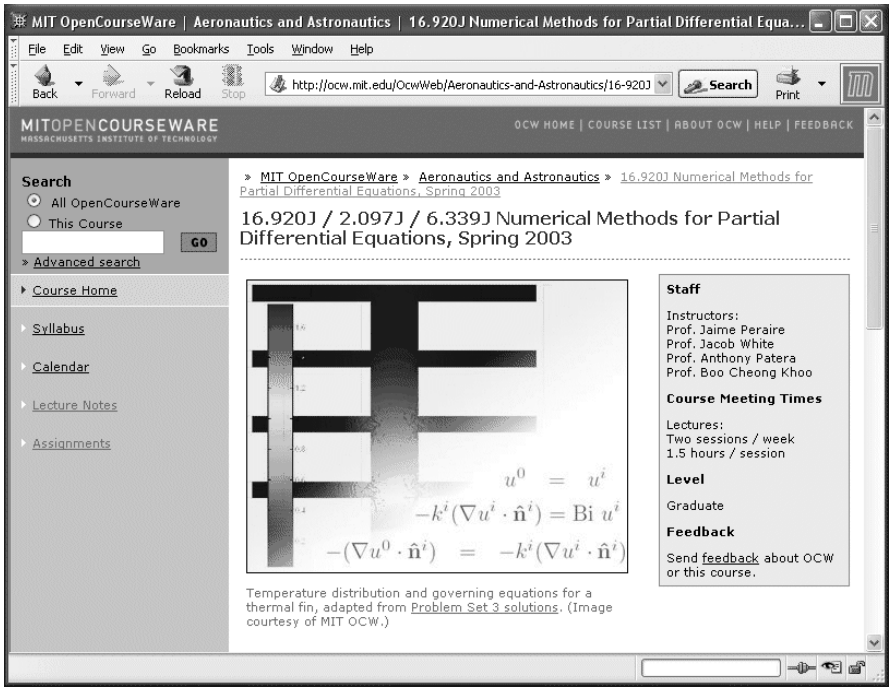

Figure 7. One of the courses offered by MIT OpenCourseWare

Educators from all over the world can use the materials as a basis for curriculum development in their own institutions. Students can use the materials for self-study or as supplementary materials for the courses they are undertaking in other institutions. The availability of such a repository of educational materials can stimulate innovations in teaching and can lead to other collaborative efforts.

OCW is not about providing an MIT education. Neither is it a distance education initiative. According to Phillip Long, "OpenCourseWare is not an online teaching environment; it is the opportunity to have faculty at MIT present their view of good teaching material, the sequencing of teaching material, good problem sets, and appropriate types of activities. It is a representation of content and sequencing and thoughtful selection and juxtaposition of materials. It is an exposure to a public audience of the decisions and processes that faculty members go through to come to the point of having a collection of resources and materials to use when teaching a particular course."[16]

OpenCourseWare materials are licensed under a Creative Commons License with the attribution, noncommercial and share alike options mentioned earlier.

Wikipedia

Wikipedia (http://www.wikipedia.org/) is a free Web-based encyclopedia that is available under the GNU Free Documentation License. The encyclopedia's contents are written collaboratively by readers and are not subjected to any formal peer review. Readers can also edit the articles written by someone else.

Wikipedia was founded by Jimmy Wales and Larry Sanger who initially started another free Web-based encyclopedia called Nupedia where the articles were peer reviewed. Progress for Nupedia was slow and the number of articles available is limited. With Wikipedia, contributors make edits and create new articles rapidly. In December 2003 it had 185,785 articles listed, covering a wide range of subjects. It is also multilingual, with articles in various other languages. The founders of Wikipedia

believe that the continuous process of editing articles will improve the content until a stable state with high quality content is reached.

Public Library of Science

The Public Library of Science (PloS – http://www.plos.org/) is a non-profit organization founded in October 2000 with the aim of making the world's scientific and medical literature a freely available public resource. Over 30,000 scientists from over 180 countries, including 13 Nobel Laureates, endorsed the setting up of PloS, which is based in San Francisco. The rationale for PLoS is that unrestricted and open access to scientific ideas, methods, results and conclusions will speed up the progress of science and medicine.

With the help of grants from the Gordon & Betty Moore Foundation and the Irving A. Hansen Foundation, PLoS launched in October 2003 the first open access journal, called PLoS Biology. The journal is available online and full-text articles can be freely accessed, downloaded, printed and distributed. Although it is an open access journal, PLoS Biology still follows a process of rigorous peer review and selection similar to the current practice for conventional journals. The Public Library of Science plans to launch a medical journal, PLoS Medicine, in 2004 using the same open access model.

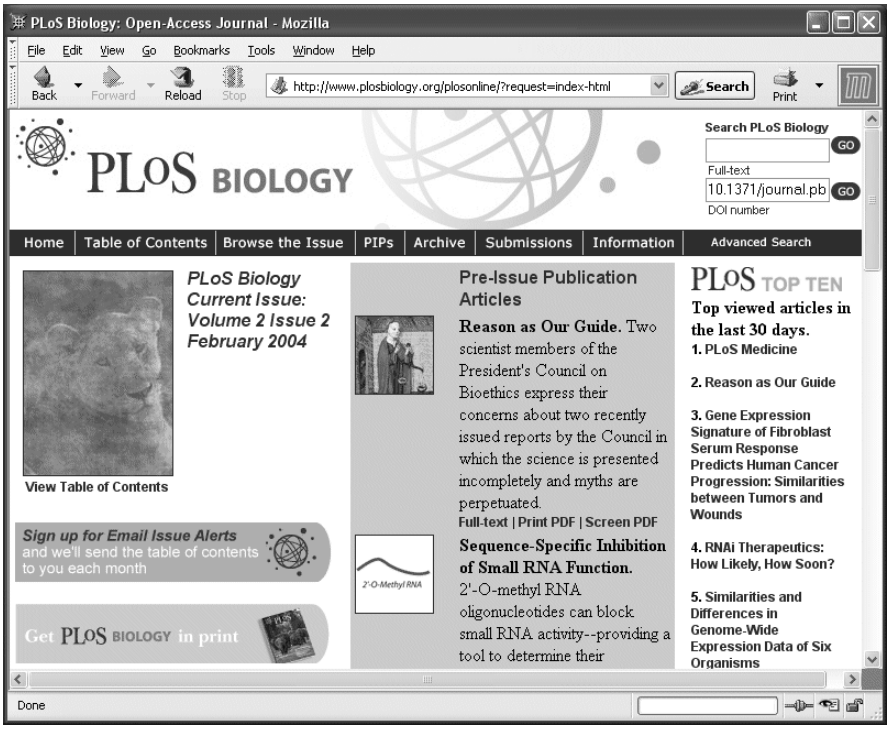

Figure 8. PLoS Biology, an open access journal

6 Research using FOSS

Traditionally, academic research is carried out in an open manner where publication of research findings is first subjected to a peer review process. All of the assumptions, calculations and experiments that lead to the results are scrutinized before the findings are accepted by journals for publication. The researchers do not usually acquire ownership of their findings and discoveries and they are expected to publish these.

Computer software is often used not only in Computer Science and ICT research but also in research in many other fields. “However, scientists rarely make their software available to other scientists for scrutiny—and even if they did, they often used closed-source programs in which the underlying source code is protected by copyright and trade secrecy claims. But this practice strikes at the heart of science, namely, the notion of *verifiability.* To be accepted as valid, all calculations and assumptions that go into a given scientific assumption must be open to public scrutiny. Yet closed-source software makes such scrutiny impossible.”[17]

In contrast, the open philosophy of FOSS is entirely consistent with the process of academic research since the source code of the software is also available for examination, if necessary. Researchers should use FOSS as a tool in their work as far as possible. Bryan Pfaffenberger goes further to argue that, “It’s not enough for scientists to use open-source software; they must also use an open-source operating system.”[17]

As mentioned earlier, Free/Open Source programming languages, database systems, spreadsheet software and other applications that can be used for computations and data analysis in research are available. More specialized FOSS is also available. For example, Scilab (http://scilabsoft.inria.fr/) is a FOSS package for numerical computations that has features equivalent to that of proprietary packages. Scilab is a full-featured scientific package, with hundreds of built-in functions for matrix manipulation, signal processing, Fourier transforms, graphics, and the like. It can be used as a research tool in a broad range of science and engineering disciplines. Another example is the GNU Scientific Library (GSL), a FOSS generic mathematical library with a rich set of functions that is available for use under the GPL. The use of these library modules facilitates development of computer programs for research purposes.

Bioinformatics

Bioinformatics, in general, is the use of computers to handle biological information. It is the use of computers to characterize the molecular components of living things (computational molecular biology). The most prominent achievement of bioinformatics is the Human Genome Project, an attempt to map the complete set of human genes. A tremendous amount of data needs to be handled in molecular biology and this is clearly possible only with the aid of computers and software.

Open source software features prominently in Bioinformatics. Ewan[18] argues that “open source makes sense because it follows good and well-known scientific principles. Traditionally, scientific practice has involved openly sharing and discussing results, and providing enough information to allow third-party confirmation of results. Clearly open source software fits well into this model.” The second reason

for using Open Source software is that the "actual data matters much more than the tools used to process it." Sharing the software used to conduct research reduces duplication of effort to develop the software.

The Bioinformatics Organization, Inc. (http://bioinformatics.org/) was founded in 1999 to facilitate worldwide communications and collaborations in bioinformatics research and to provide free and open access to methods and materials in such work. Its website hosts extensive resources, including software and databases, and provides a forum for activities that facilitate the development of such resources

High-end Computing

Linux and Open Source software have been used in projects to provide affordable high-end computing capabilities. This is done by combining the processing power of multiple low-cost servers and workstations into a system that can deliver supercomputer power. According to Cook, "The reason these systems are so effective is that there are a great many very big, very complicated problems that naturally break down into a bunch of iterations of the same, much simpler, problem. That describes everything from forecasting the weather to doing computer animation."[19]

Beowulf is the name of the architecture used for building a massively parallel system constructed out of commercially available PCs. The computers used for building the system can be 486 systems, Pentium systems and Alpha computers; the computers need not be homogeneous. Even old PCs that would otherwise be discarded can be used to build such a system. In Oak Ridge National Laboratory in the US, the Stone SouperComputer was built using a combination of old PCs connected together using a standard Ethernet network and was used to solve a mapping problem[20]. The system has a theoretical peak performance of 1.2 gigaflops[1]

Another example is the supercomputer launched by the State University of New York (SUNY) which consists of over 2,000 computers running Linux to conduct drug research to combat cancer, Alzheimer's disease and AIDS.

[1] FLOPS stands for floating point operations per second. It is used as an approximate measure of computing speed. A gigaflops is one billion FLOPS.

7 Training in FOSS

Although training in FOSS is not normally part of formal education, educational institutions can play a role in providing this service as part of a professional training or adult education program.

One of the ways to promote the adoption of Linux and FOSS in government, educational institutions, organizations and the corporate sector is to ensure that suitable human resource capacity is available. A short-term measure for building human resource capacity in FOSS is to provide a path for current IT professionals to acquire the necessary skills and certification. Although many system administrators, network engineers and other IT professionals should be able to learn on their own, a more structured training program will ensure systematic and adequate coverage of the various topics. A certification process will assess the competence of Linux and FOSS professionals and will give confidence to employers and facilitate the hiring process in an organization.

Training in Linux and FOSS is primarily to enhance job-related skills. Since FOSS is now predominantly used in back office servers, the areas for training would be in operating systems, servers, security and Web application development.

Since desktop applications are beginning to mature, there is also a growing need to train users to use these applications. Some governments and corporations are already implementing policies to migrate to FOSS desktop applications either on a mandatory or voluntary basis. However, such policies can succeed only if they are accompanied by a concerted effort to train the users who are affected by the move to use FOSS.

Certification

Lack of technical support is often cited as one of the reasons for not considering the adoption of FOSS. There is, of course, support provided informally through various newsgroups and mailing lists and vendors like Red Hat also provide support options that can be purchased by institutions. However, the existence of a pool of Linux-certified professionals will go a long way to allay the fears of organizations that are considering the adoption of Linux and FOSS.

A certification programme also helps training centers in deciding on the training curriculum. Instead of having to develop its own curriculum, a training center can adopt a widely recognized certification programme. There are other advantages of certification, such as industry recognition of Linux and provision of a path of study for professionals desiring to acquire skills in Linux.

A certification programme should have some form of examination to assess students. Examinations should be designed in such a way as to reliably assess the competencies of students. The training centers should be certified and the instructors themselves should also be certified to be suitable to conduct the training.[21]

Linux Professional Institute (LPI)

The Linux Professional Institute (LPI - http://www.lpi.org/) is a non-profit organization established in 1999. It is vendor-independent and through its activities it aims to promote the use of Linux and FOSS.

The LPI certification programme consists of three levels and is designed to certify the competency of system administrators, system engineers and other IT professionals in the use of Linux and other associated servers and utilities. It is not based on any particular Linux distribution although inputs to the programme are provided by major hardware and Linux distribution vendors.

LPI tests for well-rounded skills that are usable on any Linux distribution and to ensure validity, reliability and high quality. LPI does not provide the training directly; neither does it provide the training materials. A wide range of preparation options are supported and training and testing centers for LPI certification are available in many countries.

Red Hat Certified Engineer (RHCE)

Red Hat (http://www.redhat.com/training/) has two certification examinations—the Red Hat Certified Engineer (RHCE) and the Red Hat Certified Technician (RHCT). The training and testing emphasize practical skills and the exams measure competence with live equipment. The RHCE is aimed at two groups of IT professionals. The first group consists of system administrators, network engineers and other IT staff who already possess experience and knowledge in UNIX or Linux. The second group includes IT professionals who have little or no prior experience with UNIX or Linux.

The RHCE and RHCT certifications are meant to assess competencies in installing and configuring Red Hat Linux, configuring basic networking and file systems, essential system administration and configuring basic security. The RHCT is certification to a technician level and focuses more on client-side services and supporting Red Hat Linux systems on an existing network. RHCE focuses on server services and assesses competence in managing Red Hat Linux servers.

CompTIA Linux+

Computing Technology Industry Association (CompTIA) Linux+ (http://www.comptia.org/certification/linux/default.asp) certification is another vendor-neutral programme that validates the knowledge and abilities of technicians with at least six months of practical Linux experience. The CompTIA Linux+ certification exam measures competencies in planning and implementation, installation, configuration, administration, maintenance and troubleshooting of Linux systems. This is considered to be an entry-level certification on Linux.

8 Policy Issues

As the earlier chapters have shown, FOSS has an important role to play in education. We have seen how it can be used in the setting up and running of the ICT infrastructure of academic institutions. It can be used to meet some of the specialized administrative needs of these institutions, such as the management of libraries and the setting up of learning management systems. Its use can potentially lower the costs of providing ICT facilities to educational institutions. The use of FOSS also makes possible improvements in the teaching of computer literacy, programming, software engineering and other non-IT subjects. FOSS has a role to play in academic research and it has influenced and contributed to a more open dissemination of academic and research content.

Nevertheless, in considering the adoption of FOSS in education, policy- and decision-makers should be aware of the various issues that may arise.

Software procurement

Since there are numerous advantages to using FOSS in educational institutions, including lower costs, reliability, better performance and, arguably, better security, strategic plans or policies for education at the national or institutional level should have guidelines for procurement of software that give due consideration to FOSS. These guidelines should also apply to decisions on software acquired for use in various curricula.

The different approaches that can be taken for developing guidelines for the procurement of software are:

- Making it mandatory to use FOSS unless no suitable FOSS that is equivalent to the proprietary software is available;
- Recommending that FOSS be used whenever possible; and
- Ensuring that FOSS is given due consideration and not excluded in favour of proprietary software.

Migration

In many situations, educational institution may already be using proprietary software for both backend servers and desktops. In these cases, a strategy should be developed to migrate to the use of FOSS. The first place to start is usually the backend servers since the migration will be transparent to users and a wide range of high-quality FOSS is already available for servers. The exception will be certain applications such as financial management systems where good FOSS equivalents are not yet available. In such cases, servers running proprietary operating systems in support of such applications can be maintained and can coexist with other servers based on FOSS platforms on the same network.

For desktop applications, the adoption of FOSS can potentially result in greater cost savings. However, a migration policy will have to take into account the existing use of proprietary software and the need to maintain the use of some proprietary applications for academic requirements. A gradual approach can be taken, for example by first introducing and supporting FOSS applications that run on Windows, followed by the introduction of Linux as part of a dual boot system. There may be a transitional period where dual or multiple operating systems have to be maintained, which may result in additional support costs.

The available FOSS expertise within the institution will determine the training requirements for system administrators and other IT support staff. User training may also be required for other administrative users.

Curricula in Schools

More and more schools are being equipped with computer facilities and many have already implemented curricula to teach computer literacy to their students. These curricula should be examined to ensure that they are not based on specific proprietary software. If necessary, modifications should be made to the curricula so that the emphasis is on teaching concepts and generic skills. As we saw earlier, FOSS that is suitable for use as the basis of teaching computer literacy is available and it should be used wherever possible. It has cost advantages; discourages software piracy; raises the awareness of the students regarding the availability of FOSS solutions; and avoids over-dependence on one proprietary platform. This does not necessarily mean that proprietary software should be excluded entirely. If the resources are available, proprietary software can be used to demonstrate the range of software available to accomplish certain tasks.

We have seen that a lot of FOSS educational software for specific subjects are available and teachers should be encouraged to use these to enhance teaching and learning. If they have the skills, the teachers should also be encouraged to develop appropriate software for their classes and make these available as FOSS.

To introduce FOSS in the curricula of schools, appropriate training for teachers is likewise required. The emphasis in this case will be on training the teachers to use the appropriate desktop FOSS, such as OpenOffice, Mozilla and GIMP.

Curricula in Tertiary Institutions

At the tertiary level, the policies of institutions with respect to curricula for Computer Science or Information Technology programmes should encourage the incorporation of FOSS. Academic staff should examine course syllabi and modify them where necessary. Student projects that leverage on the availability of source code of FOSS and encourage participation and contribution to ongoing FOSS development efforts,should be introduced. Consideration should also be given to the introduction of computer languages that are increasingly being used in FOSS development such as PHP, Perl, Python and Java, in addition to the traditional languages. FOSS development methodology and the tools commonly used in developing FOSS should be incorporated in software engineering courses.

Non-IT students should be taught computer literacy using curricula that is FOSS-enabled, with emphasis on acquisition of generic skills. It may be useful to expose students to a wide selection of software, including both FOSS and proprietary software if the resources are available. FOSS for teaching specific subjects should be identified and used wherever possible. Academic staff who have the skills should be encouraged to develop software for enhancing the teaching of courses and to release these as FOSS.

Development of FOSS for education

The numerous FOSS available for educational use range from Learning Management Systems to software that can be used to teach specific subjects in schools or universities. However, in order for academic institutions in a particular country to use these software, it may be necessary to modify them to suit local educational requirements. In countries where English is not used as the medium of instruction, there is also a need to localize the software so that they can be used in the local language. Where there are no suitable FOSS available such as a Student Information System or software applications for specific academic subjects, then there may also be a need to develop new applications.

To encourage the customization, localization and development of FOSS for education, the relevant government agencies should consider establishing incentive schemes for the private sector and academic institutions to undertake these activities. This can be in the form of grants that would help to alleviate the risks involved in investing in the development of educational FOSS that may not have direct commercial returns for the private sector.

Research grants

Earlier we gave some examples of the use of FOSS in research and the reason why it should be used instead of proprietary software where possible. To promote the use of FOSS in research activities, agencies providing research grants can consider making the use of FOSS one of the criteria for the award of the grants. They can also consider specifying that any software developed as part of research activities should be released as FOSS. These conditions should apply regardless of whether the research is related to ICTs, since computer software are very often used as research tools in many other areas of research.

Training

Immediate measures need to be taken to build the human resource capacity required to implement and support FOSS. This will require the setting up of training centers that conduct programmes preferably leading to certification such as the LPI or the RHCE certification. The training centers may be government-run, set up by the private sector or set up in partnership with universities or colleges. To act as a catalyst and to ensure the availability of sufficient trainers to run the training programmes, “training the trainers” programmes can be established.

Glossary

BIND
BIND is the acronym for Berkeley Internet Name Domain. It is a computer program developed to facilitate the resolution of domain names to Internet Protocol (IP) addresses on the Internet. It is the most widely used DNS server software.

DHCP
DHCP is the acronym for Dynamic Host Configuration Protocol. Every computer connected to a network needs to be assigned an IP address. This can be done manually but it is most common to have the IP address assigned dynamically by a DHCP server.

DNS
An abbreviation for Domain Name System, an Internet service that translates domain names into Internet Protocol (IP) addresses. Domain names (e.g., www.sample.com) are easier to remember and to use but the Internet is actually based on cryptic IP addresses (e.g., 198.101.208.15). Hence, a translation between the two is required.

Free Software
The word “free” in Free Software refers to the users' freedom to run, copy, distribute, study, change and improve the software. It does not refer to the price of the software. More precisely, a program is Free Software if users have the four freedoms:

- The freedom to run the program, for any purpose.
- The freedom to study how the program works, and adapt it to your needs. Access to the source code is a precondition for this.
- The freedom to redistribute copies so you can help your neighbor.
- The freedom to improve the program, and release your improvements to the public, so that the whole community benefits. Access to the source code is a precondition for this.

The definition of Free Software and a more detail explanation is available at http://www.fsf.org/philosophy/free-sw.html.

GCC
The GNU Compiler Collection (GCC) is a collection of front ends for programming languages that is included in most Linux distributions. It currently supports computer languages such as C, C++, Objective-C, Fortran, Java and Ada and contains libraries for these languages. Work is in progress to include other languages such as Pascal and Cobol.

GIMP
GIMP (GNU Image Manipulation Program) is the most well known FOSS for image editing. It is available for various operating systems and many languages. It supports various image file formats and is considered to be the FOSS equivalent of the proprietary Photoshop software.

GNOME
GNOME (GNU Network Object Model Environment) is one of the two main desktop environments with a graphical user interface for the Linux operating system. It is intended to make the Linux operating system easy to use.

GNU
There are many references to GNU in this paper and other materials on FOSS. It is a recursive acronym for "GNU's Not Unix". In 1984, a project was started by Richard Stallman to develop a complete UNIX style operating system that is available as Free Software. This is called the GNU operating system.

GPL
The General Public License (GPL) was originally used as the license for "Free Software" distributed by the Free Software Foundation (FSF). Under the GPL, users may run, copy and modify the software, and distribute the modified software. However, users are not allowed to add their own restrictions and the modified software must be released under the same licensing terms.

IDE
An acronym for Integrated Development Environment. It refers to an integrated computer programming environment that usually has a user-friendly graphical user interface and that provides the necessary tools for developing computer programmes.

IMS
IMS originally stood for Instructional Management Systems but it is now used to refer to the organization IMS Global Learning Consortium, a non-profit organization supported by a worldwide consortium that includes educational institutions, software companies and publishers. It develops open technical specifications to support distributed learning. Several of these specifications are being adopted internationally as standards for learning technology. The goal of these specifications is to allow different course management systems and content from different authors to work together or interoperate.

KDE
K Desktop Environment (KDE) is one of two main desktop environments with a graphical user interface for the Linux operating system. It is intended to make the Linux operating system easy to use.

LAN
A Local Area Network (LAN) is a local computer network for communication between computers, typically covering a small area such as an office building or a group of buildings like a campus. A LAN may be connected to the Internet or it may be a separate distinct network. LAN is commonly used for sharing of resources such as files, printers and disk storage.

Localization
In the context of software, localization is the process of adapting, translating and customizing a product for a specific market. This means the modification of the interface so that it is meaningful and comprehensible to the local user of the product. Apart from the linguistic issues, localization also needs to address content and cultural issues as well as technical issues.

LTSP
The Linux Terminal Server Project (LTSP) provides the necessary software to set up a network of diskless workstations or thin clients to connect to a Linux server. It supports various Linux distributions and many sites are using LTSP. During boot up, the diskless workstation obtains the necessary network information from the server

and the operating system is downloaded from the server. Any program supported by the server can be run from the workstation.

OCW
In 2001, MIT announced the OpenCourseWare (OCW) project through which it would make available course material used in the courses taught at MIT. These would be available online for use by educators, students and self-learners from anywhere in the world. Materials for 700 courses from virtually all the academic disciplines are currently available on the OCW website (http://ocw.mit.edu).

OKI
The Open Knowledge Initiative (OKI) was initiated at MIT in 2001. It is a collaboration among some leading universities with MIT and Stanford leading the initiative. The objective of OKI is to define an open and extensible architecture for learning technology and it is targeted specifically to the needs of the higher education community. It provides specifications for interfaces among components within a learning management system and facilitates communication with other systems, including existing enterprise systems.

Open Source
Open Source software does not only mean access to the source code. To qualify as Open Source software, the distribution terms of the software must comply with the following criteria:

- Free Redistribution
- Availability of Source Code
- Possibility of Derived Works
- Integrity of The Author's Source Code
- No Discrimination Against Persons or Groups
- No Discrimination Against Fields of Endeavor
- Distribution of License
- License Must Not Be Specific to a Product
- License Must Not Restrict Other Software
- License Must Be Technology-Neutral

For further explanation of the definition of Open Source please refer to:
http://www.opensource.org/docs/definition.php

Operating System
The Operating System (OS) is the collection of software that controls the hardware and software applications on a computer. The OS manages and allocates the physical resources (CPU processing time, hard disk space, inputs from the keyboard, etc.) among the different applications that run on it. Examples of an OS are Microsoft Windows, GNU/Linux, Solaris and Mac OS X.

Perl
Perl is an acronym for Practical Extraction and Report Language. It is an interpretive programming language designed to process text and is typically used for CGI scripts. Perl has been developed as an open source project and was started in 1987 by Larry Wall. Developed originally for UNIX it is now available for different operating systems.

PHP
PHP originally stood for “Personal Home Page”. Today it is a recursive acronym for "PHP: Hypertext Preprocessor". PHP is an Open Source server-side scripting

language for Web programming. You can use it to add dynamic features to HTML pages or to create entire sites that generate HTML dynamically. PHP is executed on the server and the client cannot view the PHP code. PHP is compatible with many types of databases.

Python
Python is an interpreted, interactive and object-oriented programming language developed by Guido van Rossum in 1991. Python is portable and runs on most operating systems. It is suitable for rapid prototyping and as an extension language for applications that need a programmable interface. Although Python is copyrighted, the source code is freely available and distributable even for commercial use.

SCORM
An acronym for Shareable Content Object Reference Model, SCORM is an XML-based framework that allows interoperability, accessibility and reusability of learning content. Learning content created using SCORM can be easily shared among different learning management systems.

Source Code
The source code of the software is the set of programming instructions that is written by the programmer using a particular computer language. In order for the computer to understand and run the software, the source code must be compiled or "translated" into machine code (also referred to as binary code, executable code or object code). To modify the software, the source code must be available for modifications, as the machine code is not human-readable.

TCO
TCO stands for Total Cost of Ownership. This includes all of the costs involved in a technology or business solution. In addition to the initial investment cost, such costs include maintenance, support, replacement costs, and the like. In the case of software, the TCO should include the initial cost of the software; upgrade cost; and maintenance, support and training costs.

References

1. Wong, Kenneth and Sayo, Phet, ""Free/Open Source Software–A General Introduction", UNDP-APDIP, 2003; available from http://www.iosn.net.

2. "How Open Source and Commercial Software Compare: MySQL 4.0.16", Reasoning Inc. Whitepaper, 2003; available from http://www.reasoning.com/downloads.html.

3. Wheeler, David A., "Why Open Source Software / Free Software (OSS/FS)? Look at the Numbers!", December 2003; available from http://www.dwheeler.com/oss_fs_why.html.

4. Howorth, Roger, "Samba 3 extends lead over Win 2003", Oct 2003, *IT Week*; available from http://www.itweek.co.uk/News/1144312.

5. Vessels, Terry, "Why should open source software be used in schools?", 2001; available from http://edge-op.org/grouch/schools.html.

6. Decrem, Bart, "Desktop Linux Technology & Market Overview", Open Source Applications Foundation (OSAF), July 2003; available from http://www.osafoundation.org/desktop-linux-overview.pdf.

7. Glance, David, Kerr, Jeremy, Reid, Alex, "Factors affecting the use of open source software in tertiary education institutions", *First Monday*, Volume 9, No. 2, Feb 2004; available from http://firstmonday.org/issues/issue9_2/glance/index.html.

8. Martyris, Daryl, "Community–government partnerships and open source technology for low cost IT access in India–A case study", Harvard University, July 2003; available from http://www.developmentgateway.com/node/133831/sdm/blob?pid=5474.

9. Anctil, Eric, "The Open Source Integrated Library System: An Overview", 2003; available from http://www.anctil.org/users/eric/oss4ils.html.

10. "COL LMS Open Source", Commonwealth of Learning, 25 June 2003; available from http://www.col.org/Consultancies/03LMSOpenSource.pdf.

11. "Being Fluent with Information Technology", U.S. National Research Council, Committee on Information Technology, National Academy of Sciences, 1999.

12. Surran, Michael, "Linux from Kindergarten to High School", *Linux Journal*, February 2003; available from http://www.linuxjournal.com/article.php?sid=6349.

13. Moyle, Kathryn, "Open source software and Australian school education", August 2003; available from http://www.educationau.edu.au/papers/open_source.pdf.

14. Raymond, Eric, "The Cathedral and the Bazaar", 2000; available from http://catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/.

15. Hart, David, “Faster, Better, Cheaper: Open-Source Practices May Help Improve Software Engineering”, NSF press release, NSF PR 03-132, December 2003.

16. Long, Phillip D., "OpenCourseWare: Simple Idea, Profound Implications", *Syllabus Magazine*, Jan 2002; available from http://www.syllabus.com/article.asp?id=5913.

17. Pfaffenberger, Bryan, “Linux in Higher Education: Open Source, Open Minds, Social Justice”, *Linux Journal*, March 02 2000; available from http://www.linuxjournal.com/article.php?sid=5071.

18. Stewart, Bruce, “Ewan Birney’s Keynote: A Case for Open Source Bioinformatics”, O’Reilly Network, 2002; available from http://www.oreillynet.com/lpt/a/1511/.

19. Cook, Rick, “Supercomputers on the cheap”, April 2000; available from http://www.cnn.com/2000/TECH/computing/04/13/cheap.super.idg/.

20. Hargrove, William W., Hoffman, Forrest M. and Sterling, Thomas, "The Do-It-Yourself Supercomputer", Scientific American.com, August 16, 2001; available from http://www.sciam.com/article.cfm?articleID=000E238B-33EC-1C6F-84A9809EC588EF21.

21. York, Dan, “Creating a Linux Certification and Training Program”, *Linux Gazette*, Issue 33, Oct 1998; available from http://www.linuxgazette.com/issue33/york.html.

Further Readings

1. Moody, Glyn, “Rebel Code–Linux and the Open Source Revolution”, Penguin Books, 2002.

2. St. Onge, Peter D., “Linux in Education: Two Years Later”, August 17 2002; available from http://freshmeat.net/articles/view/533/.

3. Kegel, Dan, “The Case for Linux in Universities”, Oct 2002; available from http://www.kegel.com/linux/edu/case.html.

4. Kegel, Dan, “The Undergrad CS Program, Linux, and Open Source”, Mar 2003; available from http://www.kegel.com/linux/edu/curriculum.html.

5. Howland, John E., “Software Freedom, Open Software and the Undergraduate Computer Science Curriculum”, Department of Computer Science, Trinity University, April 14 2000; available from http://www.cs.trinity.edu/~jhowland/ccsc2000/ccsc2000/ccsc2000.html.

6. Massey, Bart, “Open Source Software Development in the Unix Environment”, course in Portland State University; available from http://www.cs.pdx.edu/course.php?cid=110.

7. Cesarini, Paul, “A Monocultural Alternative: The OpenCD”, Fall 2003, Computers and Composition Online; available from http://www.bgsu.edu/cconline/reviews/cesarini_review.htm.

8. Hart, Timothy D., “Open Source in Education”, May 2003, University of Maine; available from http://portfolio.umaine.edu/~hartt/OS in Education.pdf.

9. Lineweaver, Rob, “Cost savings of open source software in the server room —An informal case study in K-12 education”, 2002; available from http://staff.harrisonburg.k12.va.us/~rlineweaver/.

10. Carmichael, Patrick and Honour, Leslie, "Open Source as Appropriate Technology for Global Education", School of Education, University of Reading, UK; available from http://www.google.com.my/url?sa=U&start=1&q=http://www.ellak.gr/pub/osdocs/education/carmichael.pdf&e=7413.

11. Robbins, Jason E., “Adopting OSS Methods by Adopting OSS Tools”, CollabNet, Inc.; available from http://opensource.ucc.ie/icse2002/Robbins.pdf.

12. González-Barahona, Jesús M. and Robles, Gregorio, "Free Software Engineering: A Field to Explore", *UPGRADE - European Journal for the Informatics Professional*, Vol. IV, No. 4, August 2003; available from http://www.upgrade-cepis.org/issues/2003/4/up4-4Gonzalez.pdf.

13. Gilbert, Steven W. and Long, Phillip, "Open Knowledge and Open CourseWare Initiatives: An Interview with MIT's Phil Long", The Technology

Source, March/April 2002; available from http://ts.mivu.org/default.asp?show=article&id=979.

14. Siemens, George, “Free and Open Source Movements: Part 1 - History and Philosophies”, March 2003; available from http://www.xplana.com/whitepapers/archives/open_source_part1/.

15. Siemens, George, “Open Source Content in Education: Part 2 - Developing, sharing, expanding resources”, March 2003; available from http://www.xplana.com/whitepapers/archives/open_source_part2/.

16. Bretthauer, David, “Open Source Software: A History”, Information Technology and Libraries, Vol. 21, No. 1; available from http://www.lita.org/Content/NavigationMenu/LITA/LITA_Publications4/ITAL__Information_Technology_and_Libraries/2101_Bretthauer.htm.

17. Lakhani, Karim R. and Wolf, Robert G., "Why Hackers Do What They Do: Understanding Motivation Effort in Free/Open Source Software Projects", MIT Sloan School of Management, September 2003; available from http://freesoftware.mit.edu/papers/lakhaniwolf.pdf.

About the Author

Tan Wooi Tong is Programme Specialist for the International Open Source Network (IOSN) at UNDP's Asia-Pacific Development Information Programme (APDIP). He obtained his Bachelor and Master degrees from the Massachusetts Institute of Technology, USA. He has more than 20 years of working experience in the Information Technology field, academic institutions and engineering firms.

Acknowledgement

The author would like to thank the reviewers of this primer, Dr. M. Sasikumar (India), Dr. Nah Soo Hoe (Malaysia) and Dr. Onno Purbo (Indonesia), for their valuable comments and suggestions that have contributed to the improvement of the primer. Many thanks also go to colleagues at UNDP-APDIP—Shahid Akhtar and Kenneth Wong, for their valuable input, and all the others for their efforts that have made the creation of this primer possible.

APDIP

The Asia-Pacific Development Information Programme (APDIP) is an initiative of the United Nations Development Programme (UNDP) that aims to promote the development and application of new Information and Communication Techno-logies (ICT) for poverty alleviation and sustainable human development in the Asia-Pacific region. It does so through three core programme areas, namely, Policy Deve-lopment and Dialogue; Access; and Content Development and Knowledge Management.

In collaboration with National Governments, APDIP seeks to assist national and regional institutions in the Asia-Pacific through activities that involve awareness-raising and advocacy, building capacities, promoting ICT policies and dialogue, promoting equitable access to tools and technologies, knowledge sharing, and networking. Strategic public-private sector partnerships and opportunities for technical cooperation among developing countries (TCDC) are APDIP's key building blocks in implementing each programme activity.

http://www.apdip.net

IOSN

The International Open Source Network (IOSN) is an initiative of UNDP's Asia-Pacific Development Information Programme (APDIP). Its overall objective is to serve as a Center of Excellence and a Clearinghouse for Information on Free and Open Source Software (FOSS) in the Asia-Pacific region. IOSN seeks to raise awareness of FOSS, facilitate networking of people involved in FOSS, strengthen capacities in FOSS, and conduct R&D on FOSS.

The beneficiaries of IOSN are governments, IT professionals, software developers, the FOSS R&D community, academics and the NGO community. IOSN serves as a resource center to help policy- and decision-makers in the public sector, educational institutions, businesses and others develop policies and plans for the use of FOSS in their respective organizations. Much of IOSN’s activities are undertaken online and the IOSN portal (www.iosn.net) has been developed for this purpose and to serve as a comprehensive online resource center on FOSS. The IOSN portal also provides a means for the FOSS community in the region to contribute to its efforts and to interact.

http://www.iosn.net