Free/Open Source Software: New Opportunities, New Challenges

Brenda Chawner
Senior Lecturer
School of Information Management
Victoria University of Wellington
brenda.chawner@vuw.ac.nz

Abstract:
The free/open source (F/OS) software model makes source code available to users, who can change the software to tailor it more closely to their own requirements. With many F/OS software applications now available for library and information management, organisations have a new option for acquiring and implementing systems, plus new opportunities for participating in F/OS projects. Examples of such systems include Koha, Greenstone, and MyLibrary. Factors associated with the successful adoption of F/OS applications for library and information management include the match with an organisation's culture, technical infrastructure, staff skills, software functionality, and the extent of community support available.

Introduction

The free/open source (F/OS) software development model gives organisations a new option for acquiring and implementing systems, as well as new opportunities for participating in F/OS projects. What does this mean in practice for libraries and information centres, which have specialised requirements and make extensive use of technology to provide services to their users? This paper begins with a description of F/OS concepts, followed by a discussion of the benefits and issues associated with this approach. Three library-related F/OS projects are described to illustrate the approach in practice. The paper concludes with a discussion of factors associated with the successful implementation of F/OS software.

Free/open source concepts

If asked, most computer-literate people would say that free/open source software includes the original code for the program, whatever language it is written in (C, C++, Perl, PHP, etc.) and that it also may be modified for local use and then subsequently redistributed for "free". However, the official definitions of "free software" and "open source" cover other aspects of software use and distribution that are important in understanding how this type of software differs from commercial or proprietary software.

The Free Software Definition

The Free Software Foundation (FSF) maintains a formal definition of free software (Free Software Foundation 2003). It identifies four aspects of freedom, from the software users' perspective:

- the freedom to run the program, for any purpose (freedom 0)
- the freedom to study how the program works, and adapt it to local needs (freedom 1)
- the freedom to redistribute copies so others can benefit from the software (freedom 2)
- the freedom to improve the program, and release the improved version to the public, so that the community can benefit (freedom 3).

In order for software to qualify as "free" under freedoms 1 and 3, users must have access to the source code. The FSF maintains a list of licenses that comply with the Free Software Definition at http://www.fsf.org/licenses/license-list.html

The Open Source Definition

The Open Source Initiative (OSI) maintains the Open Source Definition (OSD), version 1.9 at 12 September 2003, which has 9 clauses:

1. Software must be able to be freely distributed, without requiring a royalty or fee for sale.
2. The source code for the program must be available, and, if not included in a distribution, must be easily available (for example, downloadable from a web page) in a form which allows a programmer to modify the program.
3. Modifications and derived works must be allowed, and these must be able to be redistributed under the same terms as the original software.

4. The integrity of the original source code must be able to be maintained, either by requiring modifications to be distributed as “patch files”, or by requiring modified versions to have a different name or version number.
5. There must be no discrimination against persons or groups.
6. There must be no discrimination against any field of endeavour.
7. The license must apply to anyone receiving a copy of the program, without requiring them to agree to another license.
8. The license must not be specific to a particular product or distribution.
9. The license must not apply to other software distributed along with the licensed program(s) (Open Source Initiative 2003).

The OSI maintains a list of approved licenses at http://www.opensource.org/licenses/ and has a certification mark that can be used on any software that is distributed under an OSI-certified license.

In September 2003, the OSI listed 45 OSD-compliant licenses, and the FSF listed 56 “free” licenses. Most of these, like the GNU General Public License (GPL) and the GNU Library or Lesser General Public License (LGPL), appear on both lists. The GPL/LGPL are by far the most popular of F/OS licenses, used by 31,280 and 4,764 SourceForge projects respectively on 14 September 2003.

Free software vs open source software

Though these two definitions represent very different philosophical positions, with the OSI being more pragmatic and the FSF less compromising, in practice they have a broad overlap. The OSD is more explicit about certain conditions of the license—such as clause 1, which specifies that the source code must be available without requiring license or royalty fees, and clause 9, which covers other software being distributed at the same time, but in practice the two definitions have much the same effect.

The key points covered by both licenses are that software may be freely modified and subsequently redistributed, and it must be available to anyone to use for any purpose. Clause 6 of the OSD is particularly important because it allows commercial use of the software — licenses that only allow non-commercial or educational use of the software are not fully “open”.

In a spirit of compromise, this paper will use the hybrid term “free/open source” (F/OS) to incorporate both positions.

A wide range of F/OS software is available, including the Linux operating system, the Apache web server, and the Sendmail utility that is used by many mail servers. Open source programming languages include Perl and PHP, and open source database management software such as MySQL and PostgreSQL is also widely used.

One particularly important point about both definitions is that they are about terms and conditions for distributing software, and say nothing about the methods and processes used to develop it—but because source code is available and can be modified by anyone who has the skill and interest, F/OS software can evolve differently from commercial packages. One effect

of the use of F/OS software licensing is the development of communities of developers and users around specific F/OS projects.

Support for F/OS projects

The FSF and the OSI are not the only organisations that promote and support F/OS activities. SourceForge.net (http://sourceforge.net/), owned by the Open Source Development Network, Inc. (OSDN), provides a repository for open source software projects, with over 72,000 individual projects listed in early December 2003. An online database of registered projects, called the Trove, allows people to search by keyword, as well as by development status, intended audience, platform, licence, programming language, interface language, and software category. SourceForge.net also provides a range of other services to open source communities, including project web space, mailing list and discussion forums, and software release management tools.

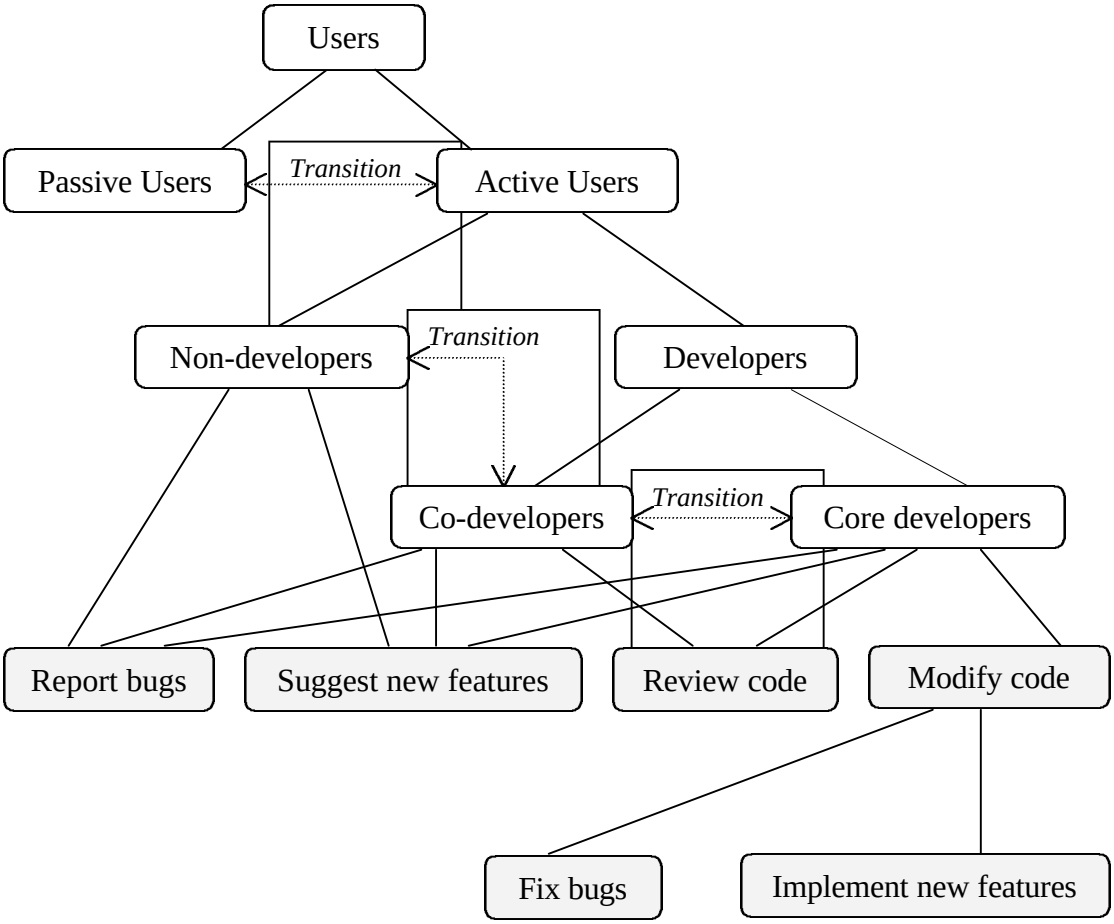
The OSDN hosts 11 other F/OS resources, of which Slashdot (http://slashdot.org/) is perhaps the most widely known, at least in the developer community. Subtitled “News for Nerds. Stuff that Matters”, Slashdot is a news portal with an emphasis on technology, in particular Linux and F/OS topics. Readers are able to comment on stories, making it a dynamic and at times controversial site.

O’Reilly and Associates, Inc., best known as a publisher of technical books, also provide a range of resources to support F/OS developers. They organise an annual Open Source Convention (OSCON), and have an open source portal at http://opensource.oreilly.com/.

F/OS project structures and roles

Scacchi (2002) described open source project organisation as “a loosely knit community of interested developers and end-users” (2002, p. 11). This is particularly applicable to projects to develop library and information management applications, as many librarians lack the skills to be active developers, but have extensive knowledge of their specialised requirements, while experienced developers are unlikely to have significant experience with library requirements.

Gacek, Lawrie, and Arief (2002) illustrate a typical F/OS community structure in a generalised, hierarchical model based on case studies of the Apache, Cocoon, and NetBSD projects. The activities carried out by the various types of participants are shown at the bottom of the diagram.



This model shows the different project-related tasks done by people in the different roles, and it also shows that people's involvement in a particular project may change over time. In a library and information context, the developers would not necessarily also be users of the software. The model shows only activities related to the actual software development, and would be a better representation of open source communities and what people do if it also showed other activities carried out by project members, such as preparing documentation or testing new releases.

F/OS software in library and information management

Free/open source software has had an increasingly high profile in the library and information management profession since 1999. A September 1999 meeting of 80 senior American academic library managers created three "keystone principles" to set a foundation for future developments of library services. One of these, "libraries are responsible for creating innovative information systems for the dissemination and preservation of information and new knowledge regardless of format" had an action item to "create interoperability in the systems they develop and create open source software for the access, dissemination, and management of information" (Keystone principles, 2000, p. 104). Members of the profession have continued to have a high level of interest in F/OS software, with over 200 articles and conference papers published in the last 4 years to describe specific projects. The oss4lib portal (http://www.oss4lib.org), originally set up in 1999, listed over 80 library-related projects in September 2003.

Frumkin (2002) suggests that the F/OS movement gives librarians an opportunity to become more active in determining the future development of the software they use, rather than letting vendors keep control. As Brandt (2001) notes, librarians have long been active not only in taking advantage of technological innovations, but also in experimenting with new approaches (using Peter Scott's HyTelnet as one example), and the F/OS approach should increase the opportunities for such activity. The following section summarises points made in the professional literature about the potential benefits and issues associated with using F/OS software in libraries.

The benefits of F/OS software for libraries

The low start-up cost associated with using F/OS software is frequently mentioned as its main attraction (Altman 2001, Geard 2001, Cervone 2003). F/OS licenses mean that software can be run on multiple machines (or at home or on laptops) with no extra licensing fees (Cervone 2003). However, this is not its only advantage.

F/OS software is also seen as a solution to the issues many libraries experience with commercial integrated library system software that is slow to evolve and expensive to upgrade (Chudnov 1999, p. 41). The expected benefits of a F/OS approach would also include more flexibility and a better match to libraries' requirements (Hattery 1999, p. 5). Schlumpf (1999) suggested that it would result in a need-driven development process, rather than the market-driven one implicit in the commercial model.

In examining the Open Source Digital Library System (OSDLS), as one example of a library-specific F/OS development, Clarke (2000) commented that the developers announced the project at an early stage, hoping to involve potential users in the design of the system, implying that this is a key benefit of the F/OS approach. He observed that the F/OS model also means that one project has the potential to generate other, related projects, because a successful idea or approach can be replicated in a slightly different context. One example of this is WIBS (Windsor Internet Booking System), developed by Art Rhyno at the University of Windsor. This project allows libraries to schedule use of public computers, and is based on an earlier F/OS project, MRBS (Meeting Room Booking System) (Bretthauer 2002b). Similarly, Prospero, a software package that allows libraries to deliver electronic interlibrary

loan documents directly to requesters, based its architecture on an earlier open source project, EDD (Schnell, 2002).

Poynder (2001) commented that F/OS software also has reduced vulnerability to viruses, and Cervone (2003) suggests that it is also technology neutral, meaning that applications will run on more than one platform (such as Windows, Linux, Unix, and MacOS X). This is true when they are written in languages such as Perl and PHP that are available for a number of platforms.

Bretthauer (2001, p. 9) comments that vendors of proprietary software can "withdraw the product, discontinue its support, go out of business, and/or can be acquired by another vendor who can do any of the above", and notes that this is not a risk when using open source software. Members of a F/OS community often set up mutual support mechanisms, sometimes based on an email discussion list. It is possible for another organisation to take on the role of project "co-ordinator" if the project originator is unable to continue in the role. This happened with the MyLibrary project, with support shifting from North Carolina State University to Notre Dame University when the project's main developer, Eric Lease Morgan, changed jobs.

Morgan (2002) identifies an increase in staff expertise through involvement in new developments, and an easier evaluation process (just download it and see what it's like to use) as other benefits of using F/OS software.

There are also technical advantages: the F/OS operating system Linux is stable, and does not need regular re-booting (Sisler 2002). Brice (2002) noted that the Meadville Public Library (MPL) benefited by being able to determine its own upgrade path when using a F/OS environment. James (2003, p. 32), comments that F/OS removes the requirement to constantly upgrade hardware in order to be able to run newer, more sophisticated software being promoted by commercial organisations, as there is no requirement to upgrade to new software versions.

Having access to source code means that users can develop better knowledge of software behaviour, with one example of this given by Brice (2002), where staff could view the blacklist of banned sites in the open source filtering software the library used, something that is not necessarily possible with commercial filtering software.

Issues relating to the use of F/OS software

The main issues raised in the literature relate to the level of support available and the degree of technical knowledge required to install and use F/OS software. With no vendor responsible for the software, support for F/OS applications can vary, and often depends on the user/developer community's commitment to the project. Schlumpf (1999) noted that F/OS software can be difficult to install and use, and this is still sometimes the case, as an analysis of postings to many F/OS mailing lists would confirm. The level of technical knowledge needed to install and maintain F/OS software can also be a barrier to its use (Poynder 2001, Bretthauer 2001). Clarke (2000, p. 36) acknowledged that some libraries choose proprietary software because they lack the necessary technical skills to support F/OS in-house. Sisler (2002) acknowledges the steep learning curve for a library implementing the Linux operating system, if staff have no previous experience with it. Technical support which relies on someone responding to a listserv request for help is also a potential limitation (Bretthauer 2001), though F/OS

communities are often very responsive to requests for help or advice, with replies to questions sent within minutes rather than days of the original request.

While the ability to use old PCs to run the F/OS operating system Linux is often portrayed as an advantage (Jacoby 2002, Lewis 2002), these are unlikely to be covered by warranties, and may be more likely to fail than newer hardware (Bretthauer 2001).

Other drawbacks of F/OS software include poor quality documentation (Murray 2002), fragmentation into multiple projects through "forking" (Altman 2001), less user-friendliness than commercial software (Altman 2001, Cervone 2003), and lower functionality than commercial equivalents (Breeding 2002, Cervone 2003).

There are also conflicting opinions on which types of libraries will be in the best position to take advantage of F/OS software: Tennant (2000) suggests that large libraries are more likely to have staff with the necessary skills and experience, while Breeding (2002b) comments that small libraries are more likely to be satisfied with the limited functionality found in currently available F/OS integrated library management systems.

Current status of F/OS software in library and information management

Dorman says that F/OS methodology is making "slow but steady progress within the library community"(2002a, 51). In an analysis of the initiators of the projects listed on the oss4lib portal, he notes that academic institutions are responsible for the largest number of projects, with not-for-profit organisations the next biggest category (2002b, 70). There are many examples of libraries using Linux for back-room operations such as web servers, database servers, or terminal services (Orr 1998, Bains and Richardson 2000, Sisler and Smith 2000).

Currently available F/OS library-related applications include not only integrated library management systems (for example Avanti, Koha, and PhpMyLibrary—not to be confused with the MyLibrary personalisation software described below), but also a range of innovative functionality, such as:

- DSpace, a digital library system to capture, store, index, preserve, and redistribute the intellectual output of a university's research faculty in digital formats
- MOSST—Modular Online Software for Self-paced Tutorials, to create web-based tutorials
- OSCR—Open Source Course Reserve, to manage electronic course reserve material, either PDF or URLs
- RAKIM—a web-based real-time virtual reference environment

Selected examples

This section of the paper will describe 3 examples of F/OS projects in more detail, to illustrate the outcomes of this approach.

Greenstone

Greenstone is a software package for constructing and implementing digital libraries that include documents in a variety of electronic formats. It was originally developed at the University of Waikato in New Zealand in 1995, and is available under the GPL. The current version is 2.42 (September 2003). Greenstone can be run under Windows, Linux, Solaris, Unix, and MacOS X, and works with standard web servers such as Apache.

Greenstone features include:

- browser-based access
- both full-text and field-specific searching
- browseable lists of authors, titles, dates, classification numbers, etc.
- use of Dublin Core and other metadata schema
- advanced data compression techniques that lower response times when searching large collections
- a customisable interface based on a configuration file
- multilingual interface, available in Arabic, Chinese, Dutch, French, German, Maori, Portuguese, and Spanish, as well as English
- extensive use of plug-ins to convert documents in different formats such as MS Word, PDF, HTML, or email
- administrative features that support access control and user activity logs

Greenstone uses Unicode for all documents and interfaces, making it suitable for use with material in in any language. It stores all documents in a standard XML form. The software is available for downloading from SourceForge. Documentation includes an installation manual, a user manual, a developer manual, a guide to customising the Greenstone interface, a description of MGPP (the Greenstone text compression and indexing system), and an online FAQ. There are two electronic discussion lists, a general one, and one for Greenstone developers. A web-based form is available for questions which are not covered in the documentation or FAQ.

The main Greenstone web site (http:/www.greenstone.org) lists selected examples of digital libraries created using the Greenstone software. A number of organisations have adopted the software to provide access to their collections of digital information, and Witten (2003) gives examples from China, Germany, India, Russia, the United Kingdom, and the United States, which cover a range of subject areas and include different source documents (text, images, pictures, and sound). One of the examples, the Mercy Corps, is not public, and illustrates the use of Greenstone to provide access to internal documents.

Greenstone's development continues to be largely based in Hamilton, New Zealand, but there is beginning to be more interaction between other Greenstone users on the general discussion list. Witten et al. (2001) give a comprehensive description of Greenstone components and processes.

Koha

Koha is an integrated library management system that was originally developed by Katipo Communications Limited of Wellington, New Zealand for the Horowhenua Library Trust (HLT), a regional library system located in Levin, some 100 kilometres north of Wellington. In 1999, HLT was looking for a new system to replace its DOS-based one, and found that commercially-available options would incur substantially increased communications charges. Katipo proposed developing a new system using open source tools (Perl, MySQL, and Apache) that would run under Linux and use Telnet to communicate with the branches. The software was in production on 3 January 2000, and released under the GPL for other people to use in July 2000. There has been a high level of interest in Koha internationally, and it is currently being used in New Zealand, Australia, Canada, the United States, India, Thailand, the United Kingdom, and France. Many of the libraries using Koha are small, mainly school and special libraries. Koha has just been implemented at the Nelsonville Public Library in Ohio. The Koha project has attracted developers in a number of different countries, with release 1.2.2 being coordinated from Canada and the current release, 2.0, from France.

The initial Koha release was an early version, with functionality for circulation, simple cataloguing and acquisitions, and OPAC searching. New versions have been released by the Koha developers irregularly, and the system is gradually gaining enhanced functionality. In most cases, Koha users either undertake the development themselves and contribute the changes back to the project, or they commission a developer to undertake specific enhancements. For example, Philanthropy Australia (based in Melbourne) hired Katipo (in Wellington) to add an abstract field to the main catalogue record, and to integrate URLs into the catalogue (Arkles, 2002). In August 2002, the Nelsonville Public Library in Ohio issued an RFP for a developer to add support for catalogue records in MARC 21 format to Koha, and this change is a major part of the new functionality in release 2.0. Other local customisations have included changing the language used in the Koha user interface.

Libraries considering implementing Koha have an option to hire Katipo staff to help with the implementation, and there is also a list of other organisations who could be hired at http://koha.org/installation/support.html.

The Koha project uses a number of channels to allow members of its community to communicate with each other — there is a general mailing list, as well as separate ones for developers, Windows users, French-speaking Koha users/developers, and German-speaking Koha users/developers.

In addition, the developers use Internet Relay Chat (IRC), a real-time message facility for scheduled meetings and less formal conversations. Bugzilla, also open source software, is used to manage bug reports and enhancement requests from users; it allows bugs to be assigned to a particular developer and allows their status to be tracked (such as reported, tested, signed off).

More information about Koha is available at http://www.koha.org.

MyLibrary

In 1998, Eric Lease Morgan, then employed at North Carolina State University Libraries developed a customisable, web-based interface to a database of information resources, called MyLibrary@NCState. It allows library users to create their own views of information resources, based on a user profile. Library staff maintain a database of descriptions of information resources that are matched to user profiles and create individual MyLibrary pages. The software is written in Perl and uses MySQL as the database engine. Source code for MyLibrary was released under the GPL in February 2000, and many libraries have since implemented it, including Wellington City Libraries in New Zealand, and Lund University in Sweden. Morgan and Reade (2000) say that nearly a dozen people were participating in MyLibrary development in the first year it was available, and people continue to contribute changes as the user base for MyLibrary grows. The first MyLibrary user group meeting was held at the LITA National Forum in October 2003, and some 20 people attended. The main activity of the meeting was the development of a list of new features for future releases. An email discussion list is the main forum for members of the MyLibrary community to communicate with each other. Leamon (2001) contains a good description of what is involved in implementing a MyLibrary portal, noting that it is "a good public relations tool for the library".

More information about MyLibrary is at http://dewey.library.nd.edu/mylibrary/.

Discussion

These three examples show that F/OS software works well in the library and information management context. These very different projects are all "alive", have growing user communities, and release new versions to their users in the same way commercial software vendors do. All of them have mechanisms to allow members of their communities to contact each other, with email discussion groups the most common. Koha has the widest range of communication channels, perhaps because the software is the most complex, with more core modules and a wider range of functionality and modules. Greenstone and MyLibrary have a more limited scope, and complement traditional library systems, making it easier for them to be installed on a trial or experimental basis.

While it is too early to say what the long term future of these projects may be, they have all made a good start, and so far their futures look promising. Their development paths over the next 2-4 years will be the real test of their viability.

Success factors for adopting F/OS software

In many ways, adopting open source software requires the same type of evaluation as purchasing commercial software, but with some modification. One analogy that is frequently used to describe using F/OS software is that it is like getting a "free" puppy or kitten. While the initial cost is zero, supporting the software over its lifetime may involve considerable costs, and organisations may find it difficult to estimate these costs when the software is initially adopted. To some organisations, the advantages of having more control over the software's future development will outweigh the perceived risks of using it, but others may come to the opposite conclusion.

Libraries thinking of adopting a F/OS software application need to consider the following factors.

Organisational culture

What support is there within your library and its parent organisation for the F/OS philosophy, and at what level? Is the organisation willing to take some responsibility for the future development of the software? Is there a "champion" who will actively promote the project, and keep momentum going during implementation?

Technical infrastructure

What operating system(s) does the software run on, and does your organisation support these? Is the programming language (Perl, PHP, etc.) installed and available? Is the database application (MySQL or PostgreSQL for example) available? What additional technologies does the software require (e.g. Apache web server)? Is there adequate documentation about the requirements and installation procedures? Can you try the software without affecting your current systems?

Staff skills

Are staff familiar with the programming language and operating environment? Will the organisation support any staff training and development needd to support the F/OS application? If not, do you have access to an external organisation who can provide the necessary support?

Software functionality

How mature is the software? How stable is it? How well does its functionality match your requirements? If it will need changes to meet your specific needs, do your staff have the skills and time to make these changes? Is there a process to have these changes incorporated into the package for future releases? What is the perceived risk of using the software? Using F/OS software to provide core functionality, such as a library management system, might be seen as too risky, while using F/OS software that provides stand-alone functionality that complements current systems might be acceptable.

Project community

How active is the project's community, and what resources are available to support new users of the software? Is there a coordinated effort to work on new releases, as in the Koha project? What would you be able to contribute back to the community? Where are community resources such as email archives and documentation hosted?

Conclusion

F/OS software has much potential for libraries and information centres, and there are a number of projects, including Greenstone, Koha, and MyLibrary, that demonstrate its viability in this context. It gives library staff an option to be actively involved in development projects, and this involvement can take many forms, such as reporting bugs, suggesting enhancements, and testing new versions. Organisations adopting F/OS software will need to provide their staff with additional development and training to enable them to take on these new roles effectively, and will need to have a long-term commitment to the projects. Currently available F/OS projects cover application areas ranging from the traditional library management systems to innovations like Greenstone and MyLibrary, which complement traditional systems. F/OS software is well worth considering, particularly for stand-alone applications that complement traditional commercial library management systems. Systems librarians and library managers should watch this trend for future developments.

List of references

Arkles, Louise. 2002. Open source library software in action. *inCite* 23 (11):12.

Bains, Simon, and Howard Richardson. 2000. Linux and CD-ROM networking: an academic library's DIY solution. *Online* 24 (2):54-56.

Brandt, D. Scott. 2001. Technologists and tinkerers. *Computers in Libraries* 21 (10):55-57.

Breeding, Marshall. 2002a. The open source ILS: only a distant possibility. *Information Technology and Libraries* 21 (1):16-18.

———. 2002b. An update on open source ILS. *Information Today* 19 (9):42-43.

Bretthauer, David. 2001. Open source software in libraries. *Library Hi Tech News* 18 (5):8-9.

———. 2002a. Open source software: a history. *Information Technology and Libraries* 21 (1):3-10.

———. 2002b. Open source software in libraries: an update. *Library Hi Tech News* 19 (5):20-22.

Brice, John J., III. 2002. Open source @ Meadville Public Library. Paper read at American Library Association, at Atlanta.

Cervone, Frank. 2003. The open source option. *Library Journal NetConnect* (Summer):8-12.

Chudnov, Daniel. 1999. Open source software: the future of library systems. *Library Journal* 124 (13):40-43.

Clarke, Kevin S. 2000. Open software and the library community. M.I.L.S. research paper, School of Information and Library Science, University of North Carolina - Chapel Hill, Chapel Hill, NC.

Dorman, David. 2002a. Open source software and the intellectual commons. *American Libraries* 33 (11):51-54.

———. 2002b. Technically speaking: open source sources. *American Libraries* 33 (11):70.

Free Software Foundation. 2003. “The free software definition.” Available: http://www.fsf.org/philosophy/free-sw.html (Accessed: 2003, December 2)

Frumkin, Jeremy. 2002. Guest editorial: balancing the playing field. *Information Technology and Libraries* 21 (1):2.

Gacek, Cristina, Tony Lawrie, and Budi Arief. 2002. Interdisciplinary insights on open source. Paper read at Open Source Software Development Workshop, 25-26 February 2002, at Newcastle upon Tyne.

Geard, Jennifer. 2001. A matter of mindshare: open source software for libraries. *Library Life* (256):29-31.

Hattery, Maxine. 1999. The free digital library. *Information Retrieval and Library Automation* 34 (11):1-7.

Jacoby, Christopher. 2002. How I built a failure-resistant web server for free. *Computers in Libraries* 22 (2):24-28.

The Keystone Principles: an action plan for values-based librarianship. 2000. *College & Research Libraries News* 61 (2):103-104.

Leamon, Pat. 2001. MyLibrary at Lund University. *Tidskrift for Dokumentation* 56 (4):103-111.

Lewis, Paul H. 2002. Why Linux works for libraries. *Computers in Libraries* 22 (10):28-30, 32-35.

Marmion, Dan. 2001. The open source movement and libraries. *Information Technology and Libraries* 20 (4):171.

Mickey, Bill. 2001. Open source and libraries: an interview with Dan Chudnov. *Online* 25 (1):23-24, 26-28.

Morgan, Eric Lease. 2002. Possibilities for open source software in libraries. *Information Technology and Libraries* 21 (1):12-15.

Morgan, Keith, and Tripp Reade. 2000. Pioneering portals: MyLibrary@NCState. *Information Technology and Libraries* 19 (4):191-198.

Murray, Robin. 2002. You pays your money — Open source software is free, but open standards are the way forward. *Information World Review* (178):14.

Open Source Initiative. 2003. "The open source definition." Available: http://www.opensource.org/docs/definition.php (Accessed: 2003, December 2)

Orr, Giles. 1998. Building a library web server on a budget. *Library Software Review* 17 (3):171-176.

Poynder, Richard. 2001. The open source movement. *Information Today* 18 (9):66-67, 69.

Scacchi, Walt. 2002. Exploring open software system acquisition processes and architectures. Irvine, California: Institute of Software Research, University of California, Irvine.

Schlumpf, Peter. 1999. Open source library systems. *Library Computing* 18 (4):323-326.

Schnell, Eric H. 2002. Prospero. In *Open source software for libraries*. Chicago: LITA, 19-29.

Sisler, Eric. 2001. Linux in your library? *Library Journal, NetConnect* 126 (17):12-14.

Sisler, Eric, and Veronica Smith. 2000. Building a library network from scratch: Eric and Veronica's excellent adventure. *Computers in Libraries* 20 (9):44-48.

Witten, Ian H. 2003. Examples of practical digital libraries: collections built internationally using Greenstone. *D-Lib Magazine* 9 (3).