

BULLETIN

of the American Society for
Information Science and Technology

asis&t

December/January 2009

Volume 35, Number 2

ISSN: 1550-8366

“Open source software is about community—the type of community that is only possible in a globally networked computer environment.”

SPECIAL SECTION

Open Source Software in Libraries

- 8]** **Introduction**
Eric Lease Morgan, guest editor
- 10]** **Explaining Free and Open Source Software**
by Scot Colford
- 15]** **The Thick of the Fray: Open Source Software in Libraries in the First Decade of this Century**
by K.G. Schneider
- 20]** **The Viability of Open Source ILS**
by Marshall Breeding
- 26]** **Evergreen in Context**
by Robert E. Molyneux
- 31]** **The Development and Usage of the Greenstone Digital Library Software**
by Ian H. Witten
- 39]** **From Open Source to Open Libraries**
by Thomas Krichel

DEPARTMENTS

- [2]**
Editor's Desktop
- [3]**
President's Page
- [4]**
Inside ASIS&T
- [5]** Don Kraft
Named Editor Emeritus
of *JASIST*

COLUMNS

- [47]**
The Student Scene
Students as
Technology Leaders
by Laura Krier
- [48]**
What's New
Selected Abstracts
from *JASIST*

NEXT PAGE >



IRENE L. TRAVIS

Editor

*Bulletin of the American Society
for Information Science and
Technology*

[Bulletin<at>asis.org](mailto:Bulletin@asis.org)

Few of our special sections are as truly comprehensive state-of-the-art surveys as the one we are pleased to present in this issue. Our guest editor, Eric Lease Morgan, has assembled an outstanding group of experts to address open source software in libraries, particularly open source integrated library systems (ILS). The six authors not only give us a good understanding of the open source movement, but also the pros and cons of open source library applications and the state of their development.

There are many variations and nuances in the definitions of open source software, but broadly, it is software that is distributed under one of a number of licensing arrangements that grant the licensees the right to modify the source code, which is included with the software, to fit their needs – provided that, if they distribute their modifications, they do so under an open source license. Open source software is not necessarily non-commercial, but many widely used examples, such as the web browser Firefox, are distributed without charge. Open source ILS systems are still young relative to the long-established commercial systems, but their functional capabilities and their established base are growing.

Technology in libraries is also the focus of a column by Laura Krier of Simmons College, a student member of the *Bulletin* Advisory Board for 2008. Her thoughtful and reflective contribution considers the library's problem in discovering how to integrate Web 2.0 applications in ways that will be useful and helpful to patrons.

The IA Column will return in the next issue, which will also include selected papers from the 2008 European IA Summit.

As I write this Editor's Desktop, the ASIS&T Annual Meeting has just concluded in Columbus, Ohio, with a good turnout despite the economy. As has become our custom, the February/March issue of the *Bulletin* will feature much many of the substantive and social work events of the meeting through photographs and articles drawn from the activities.

But in this issue of the *Bulletin*, Don Case, who assumed the ASIS&T presidency at the end of the meeting, draws from his inaugural address presented at the Annual Business Meeting to provide a taste of what's to come during his term. Check it out in Don's first President's Page.

I cannot complete these remarks without paying tribute to Don Kraft, my friend of 40 years, as he leaves the editorship of the *Journal of the American Society for Information Science and Technology*. Under his 24-year stewardship of the *Journal*, our flagship publication has retained its high ranking among journals in the field and has grown in size, scope and international stature. No one who has attended an ASIS&T Annual Meeting during Don's tenure has been exempt from his exhortation: "Publish in *JASIST*" (nor from Kraftian humor). Fortunately, Don will no doubt continue in this vein as editor emeritus – retired or not. I join other members of ASIS&T in thanking him for his great contribution to the Society and to the field. ■

**DONALD O. CASE**

2009 ASIS&T President

University of Kentucky

859-257-8415

[dcase<at>email.uky.edu](mailto:dcase@email.uky.edu)

Having served on the ASIS&T Board for several years before becoming president, I know that a continuing concern has been the erosion of our membership base over the last two decades. Of course we are far from alone in facing this problem. We live in an era of declining support for voluntary organizations. Certainly many societies and associations of various stripes – professional, scholarly and scientific – have experienced membership declines in recent years. I have examined some of the statements made by officers of these groups. The reasons typically given for membership declines implicate factors such as the rise of the Internet and computer-mediated communication, competition from similar organizations and a more general decrease in group participation across society.

Ironically, while our numbers have declined, the financial health of ASIS&T is better than ever. The recent renegotiation of our publishing contract provides the Society with enough revenue to improve what it has to offer to our membership. I believe that a substantial portion of any new monies ought to go back to the membership by lowering costs and adding benefits. Along with improvement of our meetings this strategy is one way to retain and increase membership.

The Board has recently debated a variety of initiatives toward this end, and some of them were already in place for the 2008 Annual Meeting in Columbus. For example we provided a free, one-year membership to all non-members who registered at the regular, full-conference rate. Because we believe

that future growth is partly tied to the internationalization of ASIS&T we have also provided free registration to the presidents of information science organizations outside of North America. We are discussing other incentives for members from lesser-developed nations and ways to attract and retain more student members.

In terms of enhanced benefits, you may have noticed an improved mechanism for employee recruitment at the Annual Meeting, a web-based system for exchanging information about jobs and applicants. Our goal is to make ASIS&T more of a place for recruiting than it has been in recent years. The system – which is very basic in its initial year – is the result of a task force headed by Abby Goodrum and the programming efforts of Dick Hill. We hope that this increased emphasis on recruitment will also attract entirely new attendees to future Annual Meetings.

These are just a few of the things that we have accomplished this year. There are more innovations to come.

I understand that a common way to conclude my first column as your president would be to ask you to “join me in this adventure called ASIS&T.” However, I am reminded of the words of one of my heroes, the explorer Roald Amundsen, who once said “Adventure is just bad planning.” We are lucky to have in our Board a group of dedicated members who will not leave the future of the organization to chance.

I thank all of you for your support of the American Society for Information Science and Technology. ■

New Officers and Directors Join ASIS&T Board

Every year at the ASIS&T Annual Meeting, a new administrative year begins, and the first official order of business is the introduction of new faces to the ASIS&T Board of Directors. In October in Columbus, the annual changing of the guard took place, and new officers and directors took their seats.

Each of the positions filled through recent balloting is for a three-year term. Those elected to the Board are **Gary Marchionini**, president-elect; and **Debbie Barreau** and **Peter Morville**, directors-at-large. In addition, **Amy Wallace** joined the Board as Chapter Assembly Director as a result of balloting conducted among chapter representatives.

As the new members took their seats, **Donald Case**, elected last year as president-elect, assumed the presidency from **Nancy Roderer**, who remains on the Board as past president for one year.

Gary Marchionini, professor of information science at the University of North Carolina at Chapel Hill, has served ASIS&T in numerous ways, including on the Board of Directors; on various standing and ad hoc committees; and on the editorial board of *JASIST*.

In his position statement issued in the ballot materials, Marchionini focused on the importance of



Gary



Donald Case



Debbie Barreau



Peter Morville

growing the ASIS&T membership and in strengthening ASIS&T's role and participation in summits and meetings devoted to significant topics in information science.

Deborah Barreau, associate professor at the University of North Carolina at Chapel Hill, has been active in ASIS&T chapters and SIGs since joining in 1988. She has served on numerous committees and awards juries and has worked on ASIS&T annual meetings. In 2002 she received the ASIS&T Outstanding Information Science Award. Her election platform and position statement focused primarily on increasing membership, including student membership, and engaging more members in the life of the association.

Peter Morville is widely recognized as a founder of the information architecture field. His best-selling books include *Information Architecture for the World Wide Web*, co-authored with Louis Rosenfeld, and *Ambient Findability*. He is called upon to deliver seminars and speeches at conferences and leading institutions throughout the world. Morville's position statement emphasized the importance of

maintaining a strong, enduring relationship between ASIS&T and the information architecture community. ■

ASIS&T 2008 Annual Meeting Coverage

As this issue of the *Bulletin of the American Society for Information Science and Technology* is being prepared for online access, members of ASIS&T gathered in Columbus for the 2008 Annual Meeting. Complete coverage of the meeting will be included in the February/March 2009 issue of the *Bulletin*. To provide a brief glimpse of meeting coverage for those who didn't attend the meeting, here are the winners of the 2008 ASIS&T Awards. More details will be provided next issue.

Award of Merit – Clifford A. Lynch
Watson Davis Award – Samantha Hastings
Thomson ISI/ASIS&T Outstanding Information Science Teacher Award – Eileen Abels
Best Information Science Book Award –
Scholarship in the Digital Age: Information, Infrastructure, and the Internet, by **Christine L. Borgman** (published by MIT Press)
John Wiley & Sons Best JASIST Paper Award –
Teresa M. Harrison, Theresa Pardo, J. Ramon Gil-Garcia, Fiona Thompson and Dubravka Juraga for "Geographic Information Technologies, Structuration Theory, and the World Trade Center Crisis," *JASIST*, 58(14), pp. 2240-2254
James M. Cretsos Leadership Award –
Elise Lewis and Phillip M. Edwards
Pratt-Severn Best Student Research Paper Award – Ann Irvine for "Natural Language Processing and Temporal Information Extraction in Emergency Department Triage Notes"

CONTINUED ON PAGE 5

Don Kraft Named Editor Emeritus of *JASIST*

At the meeting of the Editorial Board of the *Journal of the American Society for Information Science and Technology (JASIST)*, incoming editor Blaise Cronin announced that Donald Kraft has been named editor emeritus of the *Journal*. Kraft retires as editor-in-chief after 24 years of service on January 1, 2009. Cronin was named as his replacement in June.

Kraft's numerous accolades from both ASIS&T and the information profession include the 2007 ASIS&T Award of Merit, recognizing his contributions to *JASIST*, among other achievements. In the words of the citation:

That *JASIST* continues to reign as a top-ranked journal in quantitative and qualitative assessments is a tribute to Dr. Kraft's leadership. He has steered the *Journal* into electronic publication and more than doubled the number of issues and quadrupled the number of pages published each year. Dr. Kraft's editorship has been characterized by expanded international reach and receptivity to new approaches and areas of research.

Throughout his editorship, Dr. Kraft has had the vision to recognize these trends and encourage a very wide range of authors from many fields to publish in the *Journal*. He has thus promoted the formation, clarification and extension of the field of information science as we move through times of rapid and profound social and technological transformation. As a consequence of his leadership and considerable people skills, the *Journal* has grown

substantially under his direction, bringing a wider understanding of the field to the discipline of information science.

On accepting the editorship, Cronin also paid tribute to Kraft's tenure:

JASIST is the preeminent journal of its kind in the world, and the enduring record of our field's intellectual focus and evolution. Under Don Kraft's editorship *JASIST* has flourished, growing in terms of size, number of annual issues and breadth of subject coverage. The institutional, geographic and disciplinary affiliations of contributing authors are more varied than ever, reflecting the internationalizing of interest in information science and also the perceived attractiveness of *JASIST* as a publication outlet of first, not last, resort. In short, the maturation of our field, both scholastically and professionally, is mirrored in the pages of the journal. ■



Flanking new *JASIST* editor-for-life Don Kraft at the recent *JASIST* Advisory Board meeting are, left to right, Katherine McCain, Carol Barry, Linda Schamber, Kraft, Nancy Roderer and Blaise Cronin, new editor of ASIS&T's flagship publication.

COVERAGE, FROM PAGE 4

Thomson ISI Citation Analysis Research Grant – Isola Ajiferuke and Dietmar Wolfram for *Citer Analysis as a Measure of Research Impact*

ProQuest/ASIS&T Doctoral Dissertation Award – Eric Meyer for *Socio-Technical Perspectives on Digital Photography: Scientific Digital Photography Use by Marine Mammal Researchers*

Chapter-of-the-Year – Los Angeles Chapter of ASIS&T (LACASIS)

Student Chapter-of-the-Year – University of Washington Information School Student Chapter and the European Student Chapter

Chapter Member of the Year – Rachael Green Clemens, Carolinas Chapter, Christine Quirion, New England, and Bo-Gay Salvador, Los Angeles

Chapter Event-of-the-Year – Working Together, Working Differently: How Millennials Are Changing the Way Other Generations Learn, Interact and Do Commerce, Potomac Valley Chapter, and Tag You're It: A Dialog Between Social Tagging and Traditional Classification, Los Angeles Chapter

Chapter Innovation-of-the-Year – Los Angeles Chapter of ASIS&T for its wiki

Chapter Publication-of-the-Year – Los Angeles Chapter's *Observation of the American Society for Information Science and Technology*

SIG-of-the-Year – SIG/International Information Issues (SIG/III) and SIG/Information Needs, Seeking & Use (SIG/USE)

SIG Member-of-the-Year – Karen Fisher, SIG/USE

SIG Publication-of-the-Year – *Information and Emotion: The Emergent Affective Paradigm in Information Behavior Research and Theory*, SIG/USE ■

The International Calendar of Information Science Conferences (<http://icisc.neasist.org/>) is a nonprofit collaboration between the Special Interest Group/International Information Issues (SIG/III) and the European (ASIST/EC) and New England (NEASIST) chapters of the American Society for Information Science and Technology, with the additional support of Haworth Press.

Call for Papers for 2009 ASIS&T Annual Meeting
**Thriving on Diversity –
 Information Opportunities in a Pluralistic World**
 November 6-11, 2009 Vancouver, British Columbia, Canada

We live in a culture where countries, organizations and individuals have never been so closely linked politically, economically and socially, linkages that are founded on rapid and efficient information transfer and access. Yet we also co-exist in a world that displays its rich cultural diversity and relies upon information sharing to reinforce its plurality. ASIS&T 2009 offers participants the opportunity to explore how information research and practice can promote global communication while maintaining diversity.

Submissions by researchers and practitioners are sought across the spectrum of information science and technology. Possible topics include, but are not limited, to the following:

- Individual identities and how they are transformed by the impact of information technologies
- The societal archive – is it disappearing and/or being marginalized?
- Societal attentions and how emphasis on information technology either allows or hinders these
- Openness, access and privacy issues
- Generational, economic and socio-cultural dimensions of the impact of information on people’s lives
- Cognitive and emotional aspects of interactions with information

- Reshaping the boundary between personal and public information space
- The effect of collective information creation on authority and trust
- Information by the people for the people
- Role of information in connecting people and community building

Types of Submissions

The program committee will accept the following types of submissions:

Contributed papers present original, unpublished papers reporting on research projects, theoretical developments or innovative practical applications are invited. These papers should be reports of completed or well-developed projects on topics suitable for publication in scholarly and professional journals.

Contributed posters may be of either of two types: contributed research posters presenting new and promising work or preliminary results of research projects or contributed “best practices” posters presenting the results of design projects, practical implementations of an organization’s practices or industry innovations.

Technical sessions and panels present topics for discussion such as cutting-edge research and design, analyses of emerging trends, opinions on controversial issues, reports by practitioners on current

2009 AM Conference Committee

The following people are serving ASIS&T as members of the 2009 AM Conference Committee:

Andrew Large, McGill University, chair; **France Bouthillier**, McGill University, and **Corinne Jorgensen**, Florida State University, contributed papers co-chairs; **KT Vaughan**, University of North Carolina, and **Pascal Calarco**, University of Notre Dame, panels and technical sessions co-chairs; **Heidi Julien**, University of Alberta, and **Valerie Nettet**, State University of New York at Buffalo, posters co-chairs; and **Karen Fisher**, University of Washington; **Grant Campbell**, University of Western Ontario; **June Abbas**, University of Oklahoma; **Luanne Freund**, University of British Columbia; **Sandra Hirsh**, Microsoft Corporation; and **Tao Jin**, Louisiana State University.

information science and technology projects, and contrasting viewpoints from experts in complementary professional areas. Innovative formats that involve audience participation are encouraged. **Pre-conference sessions** present topics such as theoretical research, management strategies and new and innovative systems or products, typically for purposes of concept development or continuing education.

For more information, including submission guidelines and deadlines, please see the complete Call for Papers at <http://www.asis.org/conferences/am09/am09cfp.html> ■

News about ASIS&T Chapters

The Northern Ohio ASIS&T chapter (NORASIST) featured **Ed Dale**, who runs the Ernst & Young intranet, at its annual meeting in October. Dale spoke about information retrieval in an intranet environment and the specific challenges the technology presents.

News about ASIS&T Members

Don Kraft, outgoing editor of the *Journal of the American Society for Information Science and Technology* has been named an ACM Distinguished Scientist for 2008, an ACM membership level recognizing members who have achieved significant accomplishments or have made a significant impact on the computing field.

Amanda Spink, professor at the Queensland University of Technology Faculty of Science and Information Technology, has been appointed associate editor of the Elsevier journal *Information Sciences*. The journal publishes basic investigations in the areas of information science focusing on informatics and computer science, Information technologies, intelligent systems and applications. ■

New ASIST Titles from Information Today, Inc.

Call for our current catalog or visit www.infotoday.com

Information and Emotion: The Emergent Affective Paradigm in Information Behavior Research and Theory

Edited by Diane Nahl and Dania Bilal
ASIST member price \$47.60

ARIST 41

Edited by Blaise Cronin
ASIST member price \$99.95



Covert and Overt: Recollecting and Connecting Intelligence Service and Information Science

Edited by Robert V. Williams and Ben-Ami Lipetz
ASIST member price \$39.60

Theories of Information Behavior

Edited by Karen E. Fisher, Sanda Erdelez, and Lynne (E. F.) McKechnie
ASIST member price \$39.60



Note: Prices do not include shipping and handling.

143 Old Marlton Pike • Medford, NJ 08055 • Phone: (800) 300-9868 or (609) 654-6266 • Fax: (609) 654-4309
E-mail: custserv@infotoday.com • Order online: www.infotoday.com

Introduction

by Eric Lease Morgan, guest editor for special section

Open Source Software in Libraries

It is a privilege and an honor to be the guest editor for this special issue of the *Bulletin of the American Society for Information Science and Technology* on open source software. In it you will find a number of articles describing open source software and how it has been used in libraries. *Open source software* or *free and open source software* is defined and viewed in a variety of ways, and the definition will be refined and enriched by our authors. However, very briefly, for those readers unfamiliar with it, open source software is software that is distributed under one of a number of licensing arrangements that (1) require that the software's source code be made available and accessible as part of the package and (2) permit the acquirer of the software to modify the code freely to fit their own needs provided that, (3) if they distribute the software modifications they create, they do so under an open source license. If these basic elements are met, there is no requirement that the resulting software be distributed at no cost or non-commercially, although much widely used open source software such as the web browser Firefox is also distributed without charge.

In This Issue

The articles begin with Scot Colford's "Explaining Free and Open Source Software," in which he describes how the process of using open source software is a lot like baking a cake. He goes on to outline how open source software is all around us in our daily computing lives.

Eric Lease Morgan is with the University of Notre Dame and can be reached via email at emorgan@nd.edu

Karen Schneider's "Thick of the Fray" lists some of the more popular open source software projects in libraries and describes how these sorts of projects would not have been nearly as feasible in an era without the Internet.

Marshall Breeding's "The Viability of Open Source ILS" provides a balanced comparison between open source software integrated library systems and closed source software integrated library systems. It is a survey of the current landscape.

Bob Molyneux's "Evergreen in Context" is a case study of one particular integrated library system, and it is a good example of the open source adage "scratching an itch."

In "The Development and Usage of the Greenstone Digital Library Software," Ian Witten provides an additional case study but this time of a digital library application. It is a good example of how many different types of applications are necessary to provide library service in a networked environment.

Finally, Thomas Krichel expands the idea of open source software to include open data and open libraries. In "From Open Source to Open Libraries," you will learn that many of the principles of librarianship are embodied in the principles of open source software. In a number of ways, librarianship and open source software go hand-in-hand.

What Is Open Source Software About?

Open source software is about quite a number of things. It is about taking more complete control over one's computer infrastructure. In a profession that is a lot about information, this sort of control is increasingly necessary. Put another way, open source software is about "free." Not *free*

MORGAN, continued

as in gratis, but *free* as in liberty. Open source software is about community – the type of community that is only possible in a globally networked computer environment. There is no way any single vendor of software will be able to gather together and support all the programmers that a well-managed open source software project can support. Open source software is about opportunity and flexibility. In our ever-dynamic environment, these characteristics are increasingly important.

Open source software is not a panacea for libraries, and while it does not require an army of programmers to support it, it does require additional

skills. Just as all libraries – to some degree or another – require collection managers, catalogers and reference librarians, future-thinking libraries require people who are knowledgeable about computers. This background includes knowledge of relational databases, indexers, data formats such as XML and scripting languages to glue them together and put them on the web. These tools are not library-specific, and all are available as open source.

Through reading the articles in this issue and discussing them with your colleagues, you should become more informed regarding the topic of open source software. Thank you for your attention and enjoy. ■

Explaining Free and Open Source Software

by Scot Colford

Open Source Software in Libraries

Communicating the benefits and limitations of free and open source software to the less technically experienced can be (to say the least) challenging, so before attempting to do so, the smart professional will prepare himself or herself with three things:

- A thorough (but concise) description
- The ability to correct misconceptions
- A list of open source applications an organization can (and may already) benefit from.

What It Is

In the introductory technology course I teach for graduate LIS students at Simmons College, students usually grasp the concept of compiled software (vs. scripting) fairly easily, so I often approach the topic of open source as a metaphor.

Imagine that it's your job to buy a cake for a co-worker's birthday. You go to the bakery to see what they've got on display and you find a lovely white cake with a beautiful yellow icing with "Happy Birthday!" flowing across it. It's even kosher. But there's a problem. The frosting is pink and the birthday person just can't stand that color. There are other pre-made cakes available but they have even more things wrong with them, like the chocolate cake with "Sto Lat!" written on it. (That's Polish.) You can order a custom cake, of course, but it's more expensive, takes a week and you have your doubts that the bakery can meet your specific demands. What do you do?

Scot Colford is web services manager at the Boston Public Library. He can be reached by email at scolford@bpl.org

Why don't you bake your own cake? It would be pretty hard if you had to guess at the ingredients and just experiment with various ratios of flour, butter and sugar. You might be better off with the pink cake from the bakery. But that's why Betty Crocker invented new recipes. It's a proven way to create a cake. (And a list of ingredients is freely redistributable [1] by the way.) You can fiddle with the recipe to your heart's content to make the exact cake you need. And when you're done, you can give the list of ingredients to anyone else facing a similar situation.

Once the penny drops, I go on to explain that free and open source software is a lot like the cake that you made from the recipe. It's a creation that owes a lot to the person or people who created the original recipe, but since you were given the building blocks to create this end result, you had the opportunity to alter them to suit your needs. Ultimately, your modifications should help others who have needs similar to yours, and often, they might even be the people from whom you got the recipe in the first place. It's an evolutionary process that feeds back into itself.

Often, a student will then ask where open source software may be purchased. God bless him. I have heard colleagues ask the same thing. While the concept of direct access to application source code can be described in metaphor, the topic of licensing involves taking a look at licenses, those things we all like to click past in our rush to install software. It is necessary to broach the topic, however, since the license is truly what makes free software "free" and open source software "open." It is also useful to understand the difference between free software, as defined by the Free Software Foundation (FSF), and open source software, as defined by the Open Source Initiative (OSI).

The free software definition was published in 1986 by Richard Stallman,

president then and now of the FSF. The definition codifies four essential freedoms that computer software users should be entitled to:

- The freedom to run the program for any purpose.
- The freedom to study how the program works and adapt it to your needs.
- 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 [2].

With the emphasis on these four freedoms, free software is software that end users have freedom to alter, run and redistribute as they see fit. The label “free software” often causes confusion. “Free,” of course, also carries the connotation of “without cost,” when in reality, cost is not a criterion for free software. To address the ambiguity between “free as in free speech” versus “free as in free beer” mentioned in the free software definition, over the years a number of alternative terms have been suggested, mostly using non-English words that have unequivocal definitions. Software *libre*, for example, uses the Spanish and French adjective meaning “free” in the same sense as “liberty.” Similarly, software licensed free of charge is sometimes labeled software *gratis*. While these (and similar) terms have been adopted in non-English speaking countries – and while the FSF officially supports any term conveying the concept of liberty – software *gratis* and software *libre* have not been widely adopted in the United States. Instead, cost-free software is generally labeled “freeware” and the main FSF-approved label for liberty-infused software in use is “free software.”

This confusion, along with the confrontational activist stance of the FSF in defending these freedoms, led to the formation of the Open Source Initiative in 1998 and the open source definition [3]. Ten criteria must be met in order for a software distribution to be considered open source:

1. Free redistribution – The license must allow end users to redistribute the software, even as part of a larger software package and may not charge royalties for this right.
2. Source code – The distribution must make the source code freely available to developers.

3. Derived works – The license must permit modifications to be made to the software for redistribution under the same license.
4. Integrity of the author’s source code – The license may require that modified distributions be renamed, or that modifications be made via patch files rather than modifying the source code.
5. No discrimination against persons or groups
6. No discrimination against fields of endeavor – This includes commercial or controversial endeavors.
7. Distribution of license – The same license must be passed on to others when the program is redistributed.
8. License must not be specific to a product – A program may be extracted from a larger distribution and used under the same license.
9. License must not restrict other software – The license cannot prescribe the terms of other software with which it is distributed.
10. License must be technology-neutral – The license cannot restrict the use of the program to any individual interface or platform [4].

While the Open Source Initiative has approved over 50 different licenses as meeting the criteria of the organization, a list of the nine most widely used licenses is a sufficient sample to get an overview of the different restrictions and freedoms they provide:

- Apache Software License 2.0 (www.apache.org/licenses/LICENSE-2.0.html)
- New BSD License (www.opensource.org/licenses/bsd-license.php)
- GNU General Public License (GPL) (www.gnu.org/licenses/gpl.html)
- GNU Lesser General Public License (LGPL) (www.gnu.org/licenses/lgpl.html)
- MIT License (www.opensource.org/licenses/mit-license.php)
- Mozilla Public License 1.1 (MPL) (www.mozilla.org/MPL/MPL-1.1.html)
- Common Development and Distribution License (www.sun.com/cddl/cddl.html)
- Common Public License 1.0 (www.ibm.com/developerworks/library/os-cpl.html)
- Eclipse Public License (www.eclipse.org/legal/epl-v10.html) [5].

Despite the few ideological differences between open source and free software, for practical purposes they provide the same basic advantages (and challenges) in a library or information science setting. For this reason, they are often referred to under a collective term such as “free and open source software,” FOSS, F/OSS or other terms.

What It Is Not

A more difficult task than defining free and open source software for novices is combating the inevitable assumptions and misinformation that materialize. The most common misconception, alluded to above, is that since the source code is freely distributed without royalty or licensing fee, open source applications are free of cost. While it is possible for a library or other organization to avoid buying a proprietary software package, open source may carry a plethora of hidden costs in development and maintenances, particularly if any customization is to be made to the software. These costs may translate into salaries for additional technical staff or possibly external support, development and/or hosting services such as the consulting service LibLime.

In my experience, the staff most susceptible to the “free lunch” myth are overeager novice technicians or new librarians hoping to stretch dwindling budgets as far as possible. Veteran administrators, on the other hand, seem to regard free and open source solutions with suspicion. While some of their caution is of the “you get what you pay for” variety, the absence of an accountable vendor causes distress to some of the old library guard. Without a contract to point to, non-technical administrators seem to feel that development of library services is in the hands of an unknown middle-aged, unemployed social misfit coding in his parents’ basement at three a.m. between reruns of *Stargate* and *Star Trek: Deep Space Nine*. It is an image closely associated with hacker culture and all the bugaboos associated with it, such as Bill Gates’ attempt to frame FOSS advocates as “modern day Communists.” [6]. It is a valid point – not the image of the basement coder, but rather that the people currently working on any given open source application are not doing so to win customers; they are attempting to solve specific problems.

Richard Stallman clearly illustrates this clever problem-solving ability of developers in his 2002 article “On Hacking.” [7]. “Playfully doing something

difficult” is hacking, according to Stallman, as opposed to the criminal connotation sometimes associated with the word. A hacker enjoys puzzles and finding efficient solutions to seemingly insurmountable problems, something that libraries and information centers seem to have no lack of. An organization must hire either staff or consultants with this special ability to contribute to the larger problem-solving community if its unique concerns are to be addressed in an open source application. It is an entirely new and exciting paradigm to invite these creative people into our space, rather than knowing they are sequestered out of reach behind a vendor’s competitive gate.

Along the same lines, non-technical staff are often concerned about the perceived absence of technical support available for open source projects. Library vendors charge an incredible amount of money to support the software they license, typically in the form of yearly maintenance fees. A significant portion of a technology operating budget can be spent this way in exchange for the privilege of calling the vendor when the software fails (one hopes 24/7, but sometimes only 9-5 weekdays), when a bug is identified (that perhaps the vendor has already documented but about which has not thought to inform customers) or when documentation proves incorrect or inadequate (sometimes referred to as a “training issue” by vendors). On occasion, the maintenance fee even entitles customers to conversations such as this one.

Me: ... So, the end result is that the online catalog informs all our patrons that they will be notified by phone when their holds arrive.

Vendor Representative: Okay.

Me: Even if the individual patron will actually be notified by email or snail mail.

Vendor Rep: I see. But the default notification method for the location is “phone.”

Me: I know, but the patron’s choice overrides this when the notice is sent.

Vendor Rep: Yes. But this is the way it was designed to work.

Me: Incorrectly?

Vendor Rep: Well, you’re certainly welcome to fill out an enhancement request...

While not all vendor support experiences are as fruitless as the tongue-in-cheek examples above, technical staff will recognize that the promise of support from a proprietary software vendor rarely matches value – monetary or quality – attributed to it by the vendor or non-technical library staff. Support from a hardware vendor can be exemplary without much effort. If a component breaks, swap it out with a working piece. Software packages, however, are complex systems that usually do not function in such a modular fashion. Well, not in proprietary applications, at any rate.

Free and open source software application users, on the other hand, must rely on development communities for support. Users and developers produce documentation, write installation guides and answer specific support questions in forums, not because they are bound by contract, but because they can learn more about the software that they, too, are using. One can see this type of activity in proprietary library software user groups as well. Indeed, many systems librarians around the world find user group listservs much more illuminating about how an application works than vendor-supplied documentation or training. Many systems librarians also spend a large amount of time writing scripts, developing external applications or finding unintended creative uses for application features. Vendors sometimes even celebrate these accomplishments at annual user group conferences. That recognition is nice, but the upshot of this type of grassroots support of proprietary software is summarized most succinctly by a meme I hear frequently repeated by my colleague Michael Klein: “The workaround has become the work.” When practical issues with open system software are addressed by those using the software, however, the solutions can be immediately returned to the user community as a new release. A user’s contribution is not just a workaround. It is the essential work. No vendors are needed and it will not matter if you missed the user group conference.

Misconceptions such as those mentioned above can transform into terrifying specters sure to doom the success of any open source project at an organization. Nevertheless, a well-prepared technical staff member at a library or information-centric organization can circumvent misunderstandings and turn stakeholder anxiety into excitement, provided that an open source alternative is truly the best option. A full cost benefit analysis should be performed taking in account

all of the factors mentioned above for both proprietary and open source alternatives, including the following:

- Licensing
- Recurring fees, such as maintenance
- Personnel costs for development and maintenance cycles
- Amount and time of additional *development* required for missing features
- Amount and time of *workarounds* required for missing features
- The benefit of contributing to the support community
- The “lock in” aspect of committing to a proprietary model
- The ease with which one can (or can’t) migrate to a new platform, if necessary.

If after this analysis, free or open source software seems to offer significantly more benefits, developing a quick fact sheet for administrators comparing a specific FOSS application to a known proprietary equivalent can quickly impress. Above all, a functional prototype created in the FOSS application will dispel concerns that the software is somehow rudimentary or experimental. Another oft-repeated meme among my colleagues is “working code works,” and it is true. Nothing illustrates your point better than illustrating your point.

You’re Soaking In It

If conversations with non-technical staff veer off into the uncomfortable realm of doubt and uncertainty, they may start asking questions like “Are we sure that we are ready to *invest* in open source software?” or “Do you think we have investigated enough to *commit* to open source?” This moment is an excellent opportunity to pull out your best Madge, the Palmolive manicurist impression, and quip, “Commit to open source? Why, you’re soaking in it!”

The pervasiveness of the World Wide Web guarantees that nearly every information organization is using free or open source software to perform some function. For example, 43.7% of web browsing is being done with Firefox, an open source application, and Internet Explorer is steadily losing its lead. It only has 50.5% of the market currently [8]. Similarly, 49.82% of web servers are running Apache, which has retained its first-place spot over Microsoft IIS for 12 years [9]. It seems that every month, another visible

COLFORD, continued

library announces its website redesign in Drupal. And if an organization hosts a blog or a wiki, the chances that it is an open source package are pretty good. In fact, the chance that your organization's hosted blog is powered by WordPress is pretty good simply because it is supported by one of the most active open source communities in cyberspace.

Widely used open source applications	
Operating Systems	Web and Proxy Servers
GNU/Linux	Apache
FreeBSD/OpenBSD	Squid cache
Web Browsers	Blogs
Firefox	WordPress
Lynx	Movable Type
Office Software	Wikis
OpenOffice.org	MediaWiki
PDFCreator	TikiWiki
Instant Messengers	Content Management Systems
Gaim	Drupal
Pidgin	Joomla

Realizing that your organization is already a hybrid environment helps administrators and staff realize that a relationship with open source can be more like a respectful, close friendship than a toxic, codependent marriage. Open source will let you develop relationships with other software packages, open or proprietary. It is true that there are some great philosophical justifications for using FOSS. Just as many of us cannot ride a bicycle to every destination and instead opt to buy a hybrid car as a compromise, one can consider those noble justifications while being "as open as possible." ("AOAP" is the meme, as Mr. Klein reminds me.)

Open source library applications	
ILSs	OPACs
Evergreen	Blacklight
Koha	Fac-Back-OPAC
Repositories	MARC Module for Drupal
Digital Asset Factory	Scriblio
DSpace	SOPAC
Fedora	VuFind
Metasearch Resolvers	
CUFTS	
LibraryFind	

A commitment need only be as deep as required by the organization and exploration can be done without an intention to replace proprietary software currently in place. Much has been written about the open source integrated library systems Koha and Evergreen, but if an organization currently has an

ILS in place, it may still be worthwhile to install an alternate web OPAC like Scriblio, SOPAC or VuFind instead. The installation can live alongside the current system and, if presented as a public beta, can provide useful data regarding what users prefer from a catalog interface. There are hundreds of library applications in development, though it is advantageous to choose a project with an active development community.

Conclusion

Open source software can unnerve staff and administrators who do not have a full understanding of the concept, the myths and the all-around usefulness of it. Developers and technical staff who can communicate these three things will find it much easier to integrate some truly innovative software into their organization's technical environment. ■

Resources Cited in the Article

- [1] U.S. Copyright Office (2008). *Recipes*. Retrieved August 24, 2008, from www.copyright.gov/fls/fl122.html
- [2] Free Software Foundation (2007). *The free software definition*. Retrieved August 24, 2008, from www.fsf.org/licensing/essays/free-sw.html
- [3] Tiemann, M. (2006). *History of the OSI*. Retrieved August 24, 2008 from www.opensource.org/history/
- [4] Coar, K. (2006). *The open source definition*. Retrieved August 24, 2008, from www.opensource.org/docs/osd/
- [5] Nelson, R. (2006). *Open source licenses by category*. Retrieved August 24, 2008, from www.opensource.org/licenses/category/
- [6] Brown, A. (2005, January 11). The war on copyright communists: Bill Gates wants software patents to protect his profit, not the public. *The Guardian [London, England]*, p.22.
- [7] Retrieved August 24, 2008 from www.stallman.org/articles/on-hacking.html
- [8] W3Schools. (2008). *Browser statistics*. Retrieved September 1, 2008, from www.w3schools.com/browsers/browsers_stats.asp
- [9] Netcraft, Ltd. (2008). *August 2008 Web server survey*. Retrieved September 1, 2008, from http://news.netcraft.com/archives/2008/08/29/august_2008_web_server_survey.html

The Thick of the Fray: Open Source Software in Libraries in the First Decade of this Century

by K.G. Schneider

Open Source Software in Libraries

The library community has been a-buzz of late about open source software (OSS). The air is thick with wildly divergent opinions of its value and utility for libraries even as wikis, blogs, conferences and journal articles about OSS flood the library attention-economy.

The Body Count

To no one's surprise, a raw "body count" of libraries using open source integrated library systems indicates that the vast majority of libraries continue to rely on legacy proprietary systems. However, awareness of OSS as a potentially viable approach for library technology is much more widespread and growing, particularly as libraries use and become comfortable with reliable, high-performing OSS programs such as Firefox.

Libraries are even reengaging with software development projects for the first time in decades, and there are at least a dozen active OSS projects based in or with their genesis in library organizations, including such better-known projects as the integrated library systems Evergreen, Koha, OPALS, OLE and xCatalog; discovery layers such as LibraryFind, Blacklight, VuFind, Fish4Info, Scriblio and SOPAC; and component projects such as Umlaut, the OpenURL resolver and iVia, a search engine/portal.

Major grants through LSTA and Mellon have helped fuel some of these projects, while library systems committed to OSS development have donated sweat equity, hardware and the gift of moral support.

Even OCLC – an organization which for years has zealously protected its data and which in a public relations faux pas in the 1990s sicced its lawyers on New York's Library Hotel for using Dewey decimal numbers on its hotel

rooms – has gotten into the game. OCLC's Developer's Network now collaborates in an "open source, code-sharing infrastructure" in which library developers, among other things, "share software code with other network members and the community-at-large in an open source environment" [1].

What Is Free?

The chattering classes have been busy with OSS. At one extreme, skeptics condemn OSS as a low-grade, poor-man's substitute for licensed software and argue that libraries are better served with off-the-shelf proprietary products. At the other extreme, OSS evangelists intimate it can cure cancer, dissolve cellulite and always out-perform any other software. Between both extremes float misconceptions, assumptions, anxieties and wish fulfillment.

Somewhere amid the fog and friction of myth lie the simple facts of open source software. What makes OSS different from proprietary software is that it is free in every sense of the word: free as in "no cost," free as in "unencumbered" and free as in "not locked up." OSS is free to download, free to use and free to modify (though the process for committing modifications to the core code has a slight air of mystery, as will be discussed below). OSS code is also freely viewable.

Each definition of "free" has its own significance for the value of OSS in libraries, and each definition has also led to some confusion.

For software to be free as in "no cost" means that the software itself has no licensing fees. This definition of "free" has led library technologists such as Eric Lease Morgan to observe that free software is free as in "free kittens" – that is, that these programs still require support and maintenance. It is a matter of debate how freedom from licensing fees directly factors into the total cost of ownership for software.

Marshall Breeding, the director of Innovative Technologies and Research

K.G. Schneider, community librarian with Equinox Software, Inc., can be reached by email at kgs@freerangelibrarian.com

for the Jean and Alexander Heard Library at Vanderbilt University and author of the popular website, Library Technology Guides, has frequently questioned whether OSS is overall less expensive than its proprietary counterparts and has called for libraries to look hard at cost factors. He tells me that it is important to question whether OSS offers a lower overall TCO (total cost of ownership) than its proprietary counterparts and to not base decisions on philosophical preferences.

Meanwhile, organizations such as the Georgia Public Library Service (GPLS) have reported significant first- and second-year savings in the implementation and maintenance costs for their automated systems. In the case of GPLS, funding from other organizations has underwritten all or part of new services for their Evergreen system, including acquisitions, internationalization, a low-cost non-SIP (session initiative protocol) self-check alternative and other features.

The “free-as-in-no-cost” characteristics of OSS have led to concerns that libraries would have to provide their own support for OSS. This is a legitimate concern, as few libraries (or for that matter, few organizations in any profession) are in a position to provide most or all of the support and development required to maintain and develop software.

However, “free-as-in-no-cost” does not preclude commercial support models for OSS – and in fact, unlike the proprietary software world, because OSS is open to anyone to use, this vaguely communistic approach to software development has led to a strong free market for software support and development services. In the world beyond libraries, numerous companies have arisen to provide support for various OSS products, such as Red Hat for Linux and Acquia for Drupal.

The Net Under the Tighrope

Within LibraryLand, at least five companies now provide support for OSS. Several of these companies are associated with a specific software program (Equinox for Evergreen, Liblime for Koha and Media Flex for OPALS), while other companies such as Alpha-G and Galecia Associates provide consulting, and some companies, such as IndexData, provide a little of each.

Another common misconception about the “free-as-in-no-cost” nature of

OSS is the belief that none of the code is produced through paid development. This misconception has been supported by some of the more woo-woo philosophizing found in works such as Eric Raymond’s *The Cathedral and the Bazaar* [2], the first half of which tosses around speculative theorizing about “hacker milieus,” “gift cultures” and reputation game analyses to create a myth of the noble OSS developer, contributing code for the good of the community for nothing more than an enhanced reputation. Volunteer” and many OSS projects are thriving communities with leaders, followers, contributors, audiences and reputation systems. The library community, with its strong ethos of sharing and openness, is well-suited for volunteer development.

However, the pragmatic investment of money or sweat equity into OSS development is equally if not more significant than the role of volunteer contributions – particularly in librarianship, where developers are scarce to begin with and usually involved with far too many projects to donate the gift of development time. Companies such as Equinox, Liblime and Media Flex use the proceeds from service or special-project contracts to fund future development in the products they support, while library organizations contribute developer hours to developing services they will need, as is happening with Project Conifer, a consortium of Canadian academic librarians contributing to the development of acquisitions and internationalization for Evergreen.

This blended development model is unique to OSS – and is directly a result of the other ways OSS is “free.” Like so many things librarians hold dear – information, books and library buildings themselves – OSS is open, available and visible for all to see. This consonance with librarian values is a philosophical advantage that should not be downplayed.

Leave the Cloak and Daggers at Home

However, this openness is also a significant strategic advantage. Even if OSS were a financial wash compared to proprietary software, or only neutrally consonant with librarian values, there are benefits to OSS that make OSS preferable to proprietary products.

Vendors for proprietary software have a ready excuse for cloak-and-dagger development: if they developed their code where everyone could see it, they would be compromising the source of their revenue. OSS completely

removes that problem from the table. When anyone can see the code, the code itself has no monetary value, so the economic model for OSS is dependent on the services that software companies can deliver – a point made quite well in the second (and far superior) half of *The Cathedral and the Bazaar*. There is no motivation to reduce maintenance support when that is the company's primary revenue stream.

With OSS, it is also much easier for customers to perform due diligence about the products they are selecting – in other words, no more false promises. Many librarians have lived through years of reassurances that the companies they were working with would deliver the next version Real Soon Now, only to learn through terse press releases that the long-promised product would never be materializing. (Some librarians recall being “trained” on Taos, the never-to-emerge vaporware from DRA.) As Raymond also makes clear in the second half of *The Cathedral and the Bazaar*, “when your key business processes are executed by opaque blocks of bits that you can't even see inside (let alone modify), *you have lost control of your business*” [2, p. 152] (Raymond's rather needless emphasis).

Put on Your Big-Girl Britches

There was a time, decades ago, in the early days of library automation, when libraries built the software that drove their systems. Melvyl, NOTIS and LRS share the proud heritage of systems that were designed and written by libraries and for libraries.

It has been suggested that if the Internet had existed in those days, and those early developers had ready access to file-sharing and online community-building, librarians might have invented open source. Instead, companies founded by people with the best of intentions took over fledgling software efforts and struggled to build viable businesses. While some vendors proved scurrilous, and some customers proved exasperating, in most cases the vendor-customer relationships were riddled with endemic and unavoidable “no-fault” problems. Library software vendors found themselves dealing with customers who in a chronically under-funded profession were never able to pay realistic licensing fees, while librarians were dealing with vendors who, unable to earn enough revenue through licensing, scrimped on maintenance and development.

“Learned helplessness” is what Lori Ayre of The Galecia Group calls the outcome of this folie à deux. Ayre writes, “It's ridiculous that libraries are stuck with the systems they've got without options to determine what changes get made or even the access or privileges that would allow them to make the changes for themselves.” [3]

“Learned helplessness” has resulted in library automation software that is generally years if not decades behind development found outside LibraryLand with the result that library services are often cramped by the limitations of aged library software that all too often falls short both in features (how many catalogs still do not perform spell-check? How many do not allow patrons to pay fees by credit cards?) as well as in functionality (such as poor reindexing or transaction load capabilities). In an era of budget reductions, the last thing libraries need is software that positions them poorly with their communities.

Ayre has further commented on the stagnation in ILS development. “Library system admins simply stopped asking their ILS vendors for the changes they needed after seeing their requests end up in some future release black hole. Library staff soon learned that “it wasn't possible with their system” and stopped asking system admins for changes. Eventually, people just stopped thinking about how things could be improved. Somehow, we now have to reverse this trend.”

Ayre has also shared her experiences trying to get proprietary software to communicate with other vendor products. In this area – interoperability – OSS again presents distinct advantages. This superiority becomes sharply clear in discussions about emerging standards, where vendors for proprietary products can spend years in stalemate, because to become compliant for a particular product means that the software vendor must make its code open and available. Foot-dragging about standards compliance may often have much less to do with the vendor's reluctance to hew to a shared model than to open its code for the world to see.

Debunking the Hype

As discussed earlier in this article, there are numerous strong cases to be made for the adoption of open source software in libraries. But in some circles the hype has far exceeded what any software can deliver. It's

important to underscore that beyond the core characteristic of openness, nothing else can be inferred about OSS. The quality of OSS ranges from superior, industrial-strength, ever-adapting programs such as Firefox, Linux, Apache and PostgreSQL, to what can be charitably called GIAG, or Guy in a Garage software – the poorly documented, badly maintained programs about which the most we can say is that it’s “free” (less the many hours lost struggling within its limitations). Some OSS programs have grown huge contributor communities – thousands of developers have contributed to the latest Linux releases. Some have one or a handful of developers involved and will never grow beyond that number.

Quite a few OSS programs exhibit the latest characteristics of good software – reliance on modern programs and service-oriented architecture – but this desirable attribute is in part an accident of the relatively recent history of OSS. The history of software makes it clear that at some point not all that far in the future, all OSS known today will be obsolete. For example, among larger applications, there is an argument to be made that PostgreSQL is replacing MySQL for serious database developers.

So while it is beneficial that so much OSS leaves behind the ratty, tatty code used in some of the more notorious examples of elderly library software, it is still incumbent on the library OSS development community to evolve its products in concert with larger changes in software development outside LibraryLand. An open, collaborative model can facilitate healthy code evolution, but it cannot guarantee it.

Not only that, but OSS occasionally lives up to its stereotypes and even some of the FUD (rumors based on Fear, Uncertainty and Doubt) spread about it.

OSS can be developer-centric, with an emphasis on bare-metal interfaces and text-heavy script configurations that privilege the technically savvy at the expense of those who are inexperienced or who prefer to showcase their mettle through other means than hand-editing lengthy scripts. Software that is more time-consuming or complex for general users than its proprietary counterparts can feel like software of last resort, even when the software has superior characteristics that may endear it to power-users or make it the better choice for high-end needs.

Join these problems with some websites associated with OSS projects –

with their overly bright web pages reminiscent of mid-1990s web development, cartoonish software “mascots,” confusing and vaguely cultic invitations to “join the community” and grudging invitations to download “unsupported” Windows “binaries” (perhaps with a jab at “Micro\$oft”) – and hesitation about OSS becomes understandable, particularly for library administrators accustomed to products that are superficially more polished and who direct their appeal at the people writing checks, rather than the developers.

Also, while some OSS is exhaustively documented – MySQL is an example – overall documentation is a pervasive problem. Stuart Yeates from the University of Oxford and a contributor to the Educause blog, Open Source in Higher and Further Education, observes in his article, “Documentation Issues in Open Source,” that “[m]any open source projects face significant challenges generating and maintaining high quality, end-user documentation.” [4]

Yeates traces these challenges to nine issues, including the fact that documentation is rarely considered “sexy.” Yeates’ recommendations include requiring documentation as part of the code roll-out and underwriting the cost of professional documentation, which are practices embraced in whole or in part by several library software development projects.

One issue Yeates leaves off the table is that most people vastly underestimate the skill and resources required to write good documentation – or for that matter, to produce good writing at all. In an environment dominated by brilliant, dedicated coders, it can be easy to devalue the humble efforts of right-brainers and the skills they bring to writing, graphics design, project management and information architecture.

OSS may always trend to these limitations; these problems may be ones that are in constant mitigation, but never fully resolved. That said, the sometimes messy, poorly documented, developer-centric world of open source development, for all its foibles (endemic or otherwise), is a healthy improvement on the “learned helplessness” of the last two decades.

Oh Brave New World...

OSS presents important opportunities for libraries – though in most ventures, *opportunity* is also a synonym for *risk*. We can take back

SCHNEIDER, continued

ownership of our future, returning software development to its early glory days, when software development was intimately intertwined with, and helped drive, rapid changes in library services. As this author wrote in a biography of the technology pioneer Anne Lipow, there was a time when librarians envisioned major new services such as document delivery, and developers working right in the same library wrote the code to help make these services happen.

This is the world we want to be in again. It will not always be easy, and there will be a few spectacular failures. But there will also be spectacular successes – and this time, they will happen in the open. ■

Resources Mentioned in the Article

- [1] WorldCat. (2008). *WorldCat Developer's Network*. Retrieved October 15, 2008, from http://worldcat.org/devnet/wiki/Main_Page.
- [2] Raymond, E.S. (2001). *The cathedral and the bazaar: Musings on Linux and open source by an accidental revolutionary*. Cambridge, MA: O'Reilly.
- [3] Ayre, L. B. (2008, July 10). Ten years of learned helplessness coming to an end. *Mentat* [blog]. Retrieved October 15, 2008, from www.galecia.com/weblog/mt/archives/cat_especially_for_libraries.php.
- [4] Yeates, S. (2005-2008). Documentation issues in open source. *OSS Watch*. Retrieved October 30, 2008, from www.oss-watch.ac.uk/resources/documentation.xml.

The Viability of Open Source ILS

by Marshall Breeding

Open Source Software in Libraries

Following an era lasting more than two decades where companies offering integrated library systems (ILS) under traditional closed-source license arrangements, the library automation industry has seen a burst of activity in the last few years involving open source ILS alternatives. As libraries consider the field of ILS options today, a thorough investigation includes both traditionally licensed and open source alternatives. This essay focuses on questions regarding to what extent open source ILS products can be considered viable alternatives.

We will look at open source ILS viability from four perspectives: market acceptance, support options, product development and functionality and risk factors.

Market Perspective

We're seeing a great deal of market acceptance of ILS products in the open source arena. This does not necessarily mean that they offer all the nuances of functionality found in their commercial counterparts, only that libraries seem willing to adopt them. In broad terms, open source options are now well represented in the ILS products to which libraries are migrating.

In the current library automation marketplace, news of libraries selecting open source ILS products has become routine. In the United States and Canada, three open source ILS products dominate – Koha, OPALS and Evergreen. While Evergreen and OPALS have not yet found wide adoption outside the United States and Canada, Koha finds use in libraries worldwide.

The demographics of the libraries that have so far chosen to adopt these

Marshall Breeding is director of Innovative Technologies and Research for the Jean and Alexander Heard Library at Vanderbilt University. He can be reached by email at marshall.breeding@vanderbilt.edu

open source ILS products provide interesting information. This data reveals the track record already established and helps us identify the categories of libraries in which the current slate of ILS products can be expected to adequately serve. The following figures illustrate the level of adoption of the three major open source ILS products by libraries in the United States, as represented in the lib-web-cats database of libraries (www.librarytechnology.org/libwebcats/). This database provides the most comprehensive data available regarding the automation systems used by U.S. libraries. The table shows the numbers of libraries that have made official commitments to implement systems, not necessarily the number fully in production.

Within certain bounds, open source ILS products are making great strides in adoption in libraries within the

United States. Koha, while it attracts far more public libraries than other types, serves the most diverse audience. Evergreen finds use primarily in public libraries with a strong orientation to consortia. The original PINES implementation alone includes 49 library systems spanning 272 individual library facilities. OPALS appeals primarily to K-12 school libraries. The numbers provided in the table variously represent individual libraries, school districts and consortia of districts. The number of individual school libraries represented in the numbers is difficult to calculate but totals several hundred.

We use U.S. figures to gauge the demographics of libraries gravitating

Libraries Selecting Open Source ILS in the United States			
System	Koha	Evergreen	OPALS
Public	126	58	
Academic	23		3
School	32		51
Museum	12		
Medical	3		
Church	2		2
Other Special	10		3

Figures current as of September 2008.

BREEDING, continued

to open source ILS. The patterns are somewhat different in Canada, with both academic and public libraries showing interest in Evergreen. British Columbia has begun a phased, opt-in province-wide implementation of Evergreen while a group of academic libraries is working toward enhancing Evergreen for eventual adoption through the Conifer Project and many libraries have adopted Koha and OPALS.

The upper bound of the size and complexity of libraries moving to open source ILS products continues to increase. Currently, smaller and medium-sized public libraries have made decisions to implement Koha or Evergreen. Two members of the Urban Libraries Council have joined the open source ranks, with Koha at the Howard County Public Library in Maryland and the recent implementation of Evergreen at the Grand Rapids Public Library in Michigan. Orange County Libraries in Florida have elected to use Koha as its web-based catalog while still retaining the Millennium system for other functions. (See “Orange County Library System Board of Trustees Meeting, Board Packet for January 2008 – www.ocls.info/About/BOT/PDFs/Meetings/2008/Packets/080110Up.pdf.) No members of the Association of Research Libraries have given an open source ILS the nod.

Evergreen has proven itself capable of providing automation support for consortia with large numbers of participating libraries. The consortia using

Libraries in the United States Adopting Open Source ILS

	Koha	Evergreen	OPALS
2000			
2001			
2002	2		
2003			6
2004			2
2005	2		4
2006	22	46	5
2007	102	3	18
2008 (thru Aug)	101	11	35

Evergreen tend to comprise small to mid-sized public libraries and do not include those supporting highly populated urban centers.

The numbers of libraries adopting open source ILS products show some fairly dramatic rates of

increase, especially in the last two years. These products have already reached a level where they have made a large impact on the overall industry.

When considering the position of open source ILS in the competition to

provide library automation systems to libraries, we see a high probability that the numbers will continue to increase. At least within the small and medium-sized library arena, open source ILS isn't so much a matter of viability as inevitability. We have entered a phase of library automation where open source and proprietary ILS options will compete vigorously.

Support for Open Source ILS

A very important factor in the adoption of open source ILS products by libraries involves the role of commercial companies. Each of the open source ILS products is closely tied to a commercial business that markets, develops and supports it.

Each of the companies involved in commercial support of ILS products shares a number of important characteristics. They include the primary, if not original

personnel, that developed the product, and they offer support contracts for installation, data conversion,

system maintenance and hosting options. These companies include LibLime providing services related to Koha; Equinox Software, Inc., providing services related to Evergreen; and Media Flex providing services related to OPALS.

Although in theory any library can implement an open source ILS completely on its own, the vast majority of libraries choose to work with commercial companies. Of the libraries that have chosen to implement Koha, Evergreen or OPALS in the United States, we have found only a few examples of independent implementation without contracting for services from a commercial support companies. A group of school libraries in Utah, for example, implemented Koha independently through the Southwest Educational Development Center.

	LibLime	Equinox Software	Media Flex
Product	Koha	Evergreen	OPALS
Company Website	www.liblime.com	esilibrary.com	www.mediaflex.net
When Founded	January 2005	January 2007	1985
Background	Involved with the implementation of Koha at the Nelsonville Public Library	Development team that created Evergreen for the Georgia Public Library Service	Previously created Mandarin automation system. Created OPALS as a new open source product.

BREEDING, continued

For both the proprietary and the open source ILS products, one of the main trends involves vendor-hosted implementations, marketed as software-as-a-service (SaaS). These SaaS implementations relieve the library of the need to maintain local server hardware, operating system and network administration. The vendor assumes responsibility for the maintenance of the ILS software, including such tasks as installing new versions, patches and performing backups. The vendor provides – often through a leasing arrangement with a higher-level provider – data center facilities, including redundant levels of protected power, network and hardware. The library pays an annual or monthly subscription fee that includes all these hardware and software components. These predictable subscription costs offset the upfront capital expenditures involved when the library hosts its own servers. The only substantive difference between the SaaS offering of the open source and the proprietary ILS products involves the portion of the subscription fees associated with the software license fee.

In the current environment, almost all libraries implementing open source ILS products do so with paid support contracts. The availability of these support options makes the open source ILS products possible for almost any library regardless of the level of in-house technology support available. Libraries generally need not employ more technically trained staff to implement an open source ILS than they would for the traditionally licensed products.

As the open source ILS products experience a surge of rapid adoption through commercial support contracts, a key concern involves the capacity of the companies to absorb these new projects and deliver adequate levels of service, deliver on-time implementations and achieve a high level of customer satisfaction. In the history of library automation, there have been times where a company has achieved a rapid influx of libraries selecting its product and faltered in its ability to live up to its customer's expectations. As libraries consider open source ILS support options, they should evaluate the company's capacity to provide adequate service levels.

The companies involved in open source ILS products exhibit varying levels of maturity and growth. Media Flex, led by industry veteran Harry Chan, has been in business since 1985 and has been involved in multiple generations of proprietary library automation products for the school library

market prior to its involvement with OPALS. LibLime has been in business since early 2005 and has demonstrated constant growth in personnel, including a number of hires of individuals previously employed by companies involved with proprietary ILS products. Equinox Software, Inc., while still a very small company, has also increased its ranks, expanding the original GPLS development team with additional personnel from the ranks of the ILS industry and from libraries.

Product Development

Open source ILS products are gaining functionality at a relatively rapid pace. Through in-house library development, sponsored development and volunteer programming, Koha and Evergreen have gained major extensions in functionality not present in their original versions.

Each of the ILS products has cultivated a community of supporters highly motivated to help them succeed. The free and open source software movement has found fertile ground in the library community. The number of libraries willing to allocate institutional resources and the commitment of individuals interested in contributing both personal and work time to open source ILS projects have resulted in greater momentum than would have otherwise been possible.

Proprietary ILS products have been available for a long time relative to their open source alternatives. Proprietary products have achieved a very high level of maturity and have evolved a very rich and nuanced set of features for each aspect of functionally addressed. A key consideration for open source ILS products involves whether they have achieved a high enough level of functionality for any given type of library and if the development models in place will result in an adequate pace of advancement.

In the classic proprietary development model, the company that owns the software assumes complete control of the development agenda and its execution. Only developers directly associated with the company have access to the source code and can materially participate in its development. In the ILS industry, the development roadmap includes initiatives spearheaded within the company and in response to enhancement requests from current library customers.

BREEDING, continued

Open source ILS products follow a different model of development, involving multiple threads. By definition, open source ILS products allow anyone to gain access to their source code and make modifications. The key challenge of open source ILS involves harnessing a variety of support efforts, channeling them into a single rapidly evolving product.

The basic approach used in the open source arena involves establishing an organization surrounding a product that shepherds its development. With open source software everybody, regardless of their understanding of the product or their programming proficiency, can make modifications. Of course, people can make the changes they choose for their local version. The growth of the software, however, involves finding ways to funnel improvements back into the official distributions so that the broader community benefits.

In most cases a single individual or organization takes the role of a release manager adjudicating what modifications will be incorporated into future distributions. In order to ensure the integrity of a product, some sort of vetting process establishes which individuals enter the circle of trust of those allowed to commit new or modified code into the official distribution.

In the open source ILS arena, each of the companies has established itself as the responsible agent for its flagship product. While open source licensing theoretically enables multiple companies to compete for the support of a given product, within the United States we're seeing a one-to-one relationship between companies and products. Outside the United States, other companies have emerged to support Koha within their geographic regions (see www.koha.org/support/pay.html).

A major part of the development of open source ILS occurs through a model of sponsored development. The basic premise of this approach involves libraries interested in a given ILS identifying a particular feature, module or component that they are willing to pay to have developed. If in the consideration of implementing the product a library finds a feature missing that they must have, rather than rejecting the project, they enter into a contract for its development. In most cases, functionality created by one library through a sponsored development contract will be cycled into the next distribution so that any other library will receive the enhancement without cost. This model results in an accumulation of improvements each

of which is paid for by a single library benefiting the entire installed base.

The model of sponsored development can be seen as the key factor in the development of the Koha ILS. The initial development of the software by Katipo Communications for the three libraries involved in the Horowhenua Library Trust in New Zealand involved a fairly simple data model and did not implement many of the standards and functional modules needed by larger libraries. As Nelsonville Public Library in Athens County, Ohio, considered the implementation of Koha, they chose to contract for the development of features such as support for MARC21 and Z39.50. The Crawford County Libraries in Pennsylvania chose to support the integration of the Zebra XML storage and retrieval environment into Koha. The Westchester Academic Libraries Director Organization (WALDO) contracted to introduce many features into Koha required for academic libraries. These represent but a few examples where sponsored development has led to the expansion of Koha from a very simple ILS to one with a competitive set of features suitable for many small and medium-sized libraries.

Volunteer development also plays a role in open source ILS. The Georgia Public Library Service specifically developed the original version of Evergreen for public libraries. While the functionality of the system was geared toward public libraries, the system was designed to scale to large implementations. Larger libraries interested in open source ILS have therefore focused their attention more on Evergreen, which relies on OpenSRF middleware layer, rather than on Koha, which is based on a more monolithic perl-based infrastructure. OpenSRF, specifically developed for Evergreen, uses the Jabber protocol to distribute messages in a more services-based environment and has given this system stronger appeal to those concerned with enterprise-level implementations.

Yet the absence of components such as acquisitions, serials control, multilingual support and course reserves in Evergreen has prevented its adoption by large academic libraries. A group of libraries in Ontario, including McMaster University, University of Windsor and Laurentian University, under the name Project Conifer, have begun an effort to develop these components (see conifer.mcmaster.ca). These libraries have chosen to allocate internal resources toward this project rather than directly contracting with an external

BREEDING, continued

support company such as Equinox. Nonetheless, the work of Project Conifer has been coordinated with Equinox, working toward incorporating the project's contributions into Evergreen Version 2.0, which is being positioned as the "academic version" of Evergreen.

The forward development of open source ILS includes volunteers working for libraries or independently and sponsored development. These development channels have resulted in products capable of competing with the longer-established closed-source ILS in some market segments.

An area of concern regarding development trajectories of open source ILS products involves long-term development strategies. Much of the development energies currently focus on catching up with functionality already present in proprietary products. While the success of open source ILS depends on achieving parity with their proprietary competitors, there is some risk that too much of a focus on reinventing existing functionality comes at the cost of using open source software to deliver greater innovation.

Functionality Thresholds

Prior to the last two years, one of the greatest concerns with open source ILS products centered on whether they offered adequate functionality. Proprietary products currently hold many advantages in functionality for most staff-side modules and are capable of supporting a much wider range of libraries. Yet as we noted above, open source products have achieved functionality capable of supporting all but the tier of larger libraries.

Open source ILS products compete especially well on their web-based catalog interfaces. Both Koha and Evergreen offer some of the features expected in next-generation interfaces such as faceted browsing, relevancy-ranked results and display of cover art. In an environment where the traditional library OPAC has been at least somewhat discredited, a new genre of discovery interfaces has emerged better suited to web-savvy library users, including AquaBrowser from Medialab Solutions; Encore from Innovative Interfaces; Primo from Ex Libris; BiblioCommons; and open source alternatives such as VUFind and SOPAC. The online catalogs of Koha and Evergreen have just as much in common with these new interfaces as with traditional library OPACs.

The functionality of open source ILS products already falls well within the needs of a very large number of libraries. Many public libraries serving small communities and rural populations currently rely on outdated automation products or have no automation. Open source ILS products represent at least equal functionality but also provide an entry into more sustainable web-based computing. While proprietary closed source ILS vendors have struggled to provide automation solutions that small public libraries could afford and implement, open source ILS products seem well positioned to serve this large segment of the library automation market.

Open source systems must survive on the merits of their features and functionality, scalability and performance. The standards for these characteristics vary according to each sector of the library market. Today, only proprietary systems stand ready to support the largest and most complex tier of libraries. Given the trajectories of development described previously, we can expect the capabilities of open source products to rise gradually. If functionality increases past the threshold that now precludes adoption by large libraries, we can expect at least some large public and academic libraries to shift from proprietary to open source ILS products.

Risk Issues

Many libraries currently seek automation strategies that make them less vulnerable to the increasing harsh business environment. Recent rounds of business consolidation and corporate buy-outs by private equity firms have resulted in a great deal of uncertainty for libraries about the business environment. Some companies have been strengthened through these transitions and stand more capable than ever to create and support library automation products. Yet many libraries remain very skeptical of the current business environment of the incumbent vendors, which opens an opportunity for new products, business models and companies.

The movement toward open source ILS comes during a time in the library automation industry when libraries find themselves frustrated with traditional vendors. One event in particular eroded library confidence in the status quo slate of vendors. The business decision of SirsiDynix not to develop future generations of Horizon and to concentrate on Unicorn as the

BREEDING, continued

basis for its strategic library automation platform left many libraries scrambling for new options. Even for libraries outside the SirsiDynix fold, many libraries came to realize their vulnerability to business decisions far removed from their control.

One factor in the appeal of open source ILS involves a perception that it frees libraries from dependence on any given vendor. In theory, once a library has adopted an open source ILS, it will have viable options for support even if the company with which it originally contracted either goes out of business or fails to deliver adequate support. In the absence of license restrictions imposed by proprietary products, a library could choose to contract with another firm for support, or it could take on support in-house.

Conclusions and Observations

The open source ILS movement has progressed past the point where its viability can seriously be questioned. The current momentum of open source ILS adoption makes it almost inevitable that it will represent an increasing portion of the library automation landscape. A set of companies has emerged to provide support options. Each of the products has already achieved a level of functionality suitable for its current target market. Current open source ILS products have demonstrated a history of increasing functionality with models in place that promise reasonable levels of future development.

It is not at all clear what proportion of the ILS market will be represented by open source alternatives. Although we see increased acceptance of open source ILS products, the market for proprietary systems remains strong,

especially for larger libraries. We see higher levels of adoption of open source ILS in the United States and Canada than in other regions of the globe.

Current trends do not necessarily presage the demise of the companies offering closed source ILS products. Some of these companies continue to foster a positive reputation with their current library customers and continue to make announcements of new sales. The open source movement most affects the companies that have failed to maintain the confidence of their customer libraries either through diminished levels of customer support, sluggish product development or strategies that force premature migrations.

Although some of the companies involved with closed source ILS have experienced a downturn in sales in the last year, some appear to be attracting new libraries to their products. In 2007, open source contracts represented less than 10% of the ILS contracts for academic and public libraries, according to the Library Journal "Automation System Marketplace" report. 2008 stands to be a pivotal year in the heightened competition between the companies offering proprietary versus open source ILS.

The open source ILS movement has produced other benefits for the industry. At least some of the companies offering closed source library automation products have reacted to the presence of open source products by introducing new levels of openness in their closed-source products, usually through increased availability and documentation of APIs (application programming interfaces). These APIs provide new opportunities for interoperability with external systems and give libraries better access to the underlying data and functionality of their systems without access to source code. ■

Evergreen in Context

by Robert E. Molyneux

Open Source Software in Libraries

The plan to develop Evergreen, the open-source consortial library software, was announced in June 2004 by Lamar Veatch, state librarian and head of the Georgia Public Library Service (GPLS). Evergreen went live in 2006. That two-year period was filled with activity.

It was a major decision by GPLS to develop new software of a scope never before attempted, but it was undertaken only after it had exhausted all other alternatives to run PINES, the resource-sharing public library consortium in Georgia. Evergreen was to be designed from the ground up as a new thing: a consortial library system, that is, an integrated library system for a large, geographically dispersed, resource-sharing consortium. Why did GPLS take this path and how did it go about doing it?

A Bit of History

The decision came about after a long chain of events made over many years in Georgia as it, like all states, wrestled with how best to provide public library services to its citizens. Policy decisions over many years followed a consistent direction.

Local library organizational patterns commonly follow a model of treating each library separately resulting in what today are commonly called information silos, resulting in largely separate collections with weak communications among them. Resource sharing among the libraries in most state systems is through an interlibrary loan (ILL) process.

Georgia, on the other hand, has consistently chosen another direction:

Bob Molyneux is vice president of business development with Equinox Software, Inc. He can be reached by phone: 877- 673-6457, ext. 710 or 678-269-6112; by email at bob<at>esilibrary.com; or by Skype: ESI-Bob. The company website is at www.esilibrary.com

joining the many small public libraries into larger resource-sharing regional libraries. Other states have worked on the idea, abandoned it, taken it up again and so on. It was an idea that the library community liked in theory, but before recent developments in computer technology, Georgia's path was unusual.

The fundamental problem that all states must wrestle with in providing public library service – indeed for all entities providing any library service – is the great disparity of size, wealth and resources of the many libraries that exist. There are a few very rich libraries; however, most have only modest resources.

Consider these figures from the latest national-level data we have for U.S. public libraries (FY 2005). Table 1 shows that the top quartile of these libraries by their total expenditures – that is, the 25% of public libraries that spent the most – spent \$7.9 billion that year while the bottom three quartiles (the 75% that spent the least) spent \$1 billion in total. Indeed, to press the point, the top 10% spent \$6.5 billion. Think about that: Ten percent of the public libraries in the United States spent about 6.5 times as much money as the smallest 75% of the libraries.

The big libraries are very big, and the small libraries are very small, and the disparities in expenditures that these figures demonstrate are consistent across other variables such as collections, resources and staff. Moreover, they are consistent with the pattern for other types of libraries. However, our story is centered on public libraries in the United States because of decisions made by GPLS.

TABLE 1. Total expenditures by U.S. public libraries in FY 2005

	\$ Millions	Percent
Top Quartile: (Largest 25% of Total Expenditures)	7.9	89
Bottom Three Quartiles: (Smallest 75% of Total Expenditures)	1.0	11

It is useful to note that the two largest open source library systems running in the United States, Evergreen and Koha, were developed for small public library settings. If you look at the expenditure figures above, it is not hard to imagine why – the large proprietary vendors prefer to concentrate their efforts where the money is. Within the open source framework, however, Evergreen and Koha have different approaches, with Evergreen being consortial for reasons that are discussed here.

The disparities in resources outlined by the data present a profound information policy question for a consensual democracy: In a system of government that relies on an informed citizenry, are such disparities acceptable? Is it acceptable for the “haves” to have access to so much more information than the “have nots”? If not, how do we ameliorate the results of this disparity?

Public library service as it is configured in the United States today developed before Google and other modern software developments that have relentlessly broken down information silos in other fields. Their configuration cannot adequately provide the vast pools of information and materials that are now available to the citizenry. One step in the right direction is consortial resource sharing, and another is to use these consortia to pool money to buy access to enhanced content for all the libraries in the consortium – especially those that could not otherwise have afforded this content – the “everything everywhere” library consortium.

In its history (at least by the 1940s), Georgia decided to follow its subsequently consistent policy – with the normal stops and starts involved with politics – to bind the small public libraries in the state into larger, regional entities for resource sharing. That strategy was Georgia’s way to get library resources to the “have nots.” The state provided both funding and direction toward that end. Although Georgia’s largest libraries are typically independent, the smaller ones today are mostly in regional systems.

PINES

In 1998, David Singleton, then at GPLS and now at Charlotte-Mecklenburg Public Library in Charlotte, North Carolina, suggested a universal borrower card for Georgia’s public library users. Such a card

would allow all citizens of Georgia to borrow an item from any library. Given both the amount of state funding and state strategic practices to group libraries into regions, such an idea was possible. Of course, the idea was not new to Georgia, but it fell on receptive Georgia soil since the infrastructure and the general acceptance of the idea of sharing resources were in place. Because of years of preparation, state guidance and vision, the idea would work.

TABLE 2. PINES timeline

1999	PINES goes live with 26 library systems and 96 outlets
June 2004	Work begins on Evergreen
September 2006	Evergreen goes live over the Labor Day weekend. 44 Georgia public libraries with 252 outlets are now running an open source consortial library system.
September 2008	On Evergreen’s second birthday, it is running in 62 public library systems with 294 outlets in five U.S. states and one Canadian province. It is also running on its first academic library. PINES has 50 systems with 275 outlets and regularly circulates 100,000 a day and will reach about 16 million circulations for the year. There are nearly 10 million items in the PINES online catalog.

Y2K – the computer problem that existed because of the way computers handled dates – in the end was what led to the creation of PINES (the Georgia public library resource sharing network – the network of Georgia’s regional library systems). Y2K money was allocated by the state, and an RFP for a statewide system was issued. One vendor’s product stood out, and PINES was born in 1999 with Phase 1 linking 26 library systems with 90 outlets (central libraries, branches and bookmobiles).

PINES was an immediate success. PINES users liked being able to borrow books from a larger set of libraries with a richer set of collections than one local library. They liked access to the longer tail, and PINES added more libraries and more users. Library users altered their behavior, and many bypassed their local libraries to have access to the larger, virtual collection. PINES grew until it started having problems with the underlying software that, famously, resulted in the system’s production servers having to be rebooted during the middle of the day in order to be able to handle control of the transactions load. The software had reached a previously unknown limit.

Limits

Given that the software that ran PINES is running other systems that have higher circulations and larger bibliographic databases than PINES, what limit had been reached?

There were several, but the central one was the system could not keep up with the update load. The software was not designed for a consortium spread over a large area with updates such as checkouts, check-ins and cataloging changes to the central database coming from so many different places. In such a system, there are many people logged into the system changing the database at any one time, and the database, to be useful, has to be current. It should reflect what is checked in or out (transactions) and what the system owns (inventory) so that users searching will know if an item is available and librarians will know the state of items in the system. These functions are basic to a modern inventory control system.

In PINES, there are thousands of terminals across the state logged into the database making changes, and each change has to be reflected in the database by “reindexing” it. While rebooting in this old system allowed the system to catch up with the changes, it could not be done frequently enough. The limit reached was a hard one, and there was no solution using that software. Meanwhile, more libraries wanted to join PINES. All the while, the circulation load was large and increasing.

Design Architecture of Evergreen

Any large piece of software has a design architecture: how it organizes its elements, what tools it uses, that is, how it does what it does. The design architecture, in turn, is a function of various factors, the two most important being system architecture and computer costs.

Many old versions of old software still litter the ILS ecosystem. The software then running PINES began development in the era when mainframes ruled the earth. This legacy software has been adapted and changed and added to over the years. Elements had been bolted on and kludged, but, it is a fact that updating deployed software has proven to be an almost impossible task almost every time it has been tried. There are a thousand problems, and what most vendors of library software have done is start over using the more modern updated practices of the time.

When the original PINES software was developed, computers were immensely expensive and software comparatively primitive. Better practices and more capable small computers existed when Evergreen was envisioned, and the developers used more modern software development processes to run on more capable computers that no longer cost a million dollars each to buy. An examination of alternatives in the market found none that had the capabilities necessary to run PINES then or in the future. What was Georgia to do? Limit PINES? Give up and abandon a policy direction it had followed so consistently and that had produced such benefits? Over software?

After much discussion, GPLS embarked on a revolutionary path – an open source, consortial library system to run PINES.

Development Begins

Work began on Evergreen mid-2004. The later a software system is developed, the more modern practices can be used in that development. Evergreen, for instance, was web-aware and used Unicode for text storage and representation from the beginning. To implement these capabilities, Evergreen did not require the costly and difficult rewrites to the foundational code necessary for ILSs that had their basic code written before Unicode had been developed. Unicode is important because it allows a system to handle non-Roman scripts, including characters in Chinese and Arabic.

Technology – The Underlying Software Architecture

Evergreen’s development began with practices then in place in the business world. These practices today result in software that is modular, scalable and relatively easy to update – at least compared to legacy software that is even as little as five years old.

Evergreen uses a service-oriented architecture (SOA) with representational state transfer (REST) and “n-tier” architectural design concepts. Let us explore these concepts a bit.

The development philosophy behind these elements is that software is based on distributed services that communicate and collaborate through a set of standards, that is, it should be “service oriented.” In Evergreen, the open scalable request framework (OpenSRF, pronounced “open surf”) is such an SOA framework.

OpenSRF is, in many ways, the key to Evergreen because it does so many things. OpenSRF provides load balancing and is robust and scalable. It also allows the development of software without requiring a detailed knowledge of Evergreen's structure. In effect, the Evergreen layer consists of a number of applications each of which rides on top of OpenSRF. The result is that the developer need only know the APIs and what information the program elements require and what they will reply to write new capabilities into Evergreen. OpenSRF handles the details of implementing a stateful, decentralized service architecture while the Evergreen code supplies the interface and business logic. OpenSRF is now a separate open source project.

Networking follows a design philosophy that is often introduced to students with the OSI Reference Model. The model describes seven layers that pass information among each other to provide network services in a manner similar to what one finds with SOA. "N-tier" application architecture is a related idea where the tiers communicate and collaborate with each other by established standards. We see these operations in our daily lives. If, for example, a new version of Firefox comes out, we can change the old version for the new one without changing the operating system or reformatting our hard drives. We usually do not even have to reboot. Why? Because of the underlying architecture where each program communicates with the other parts of the network software by established rules – application programming interfaces (APIs) – that specify how the various elements involved exchange needed information. So, to change a part or one tier does not require a massive rewrite of the entire code, just what needs to be adjusted. As mentioned, the Evergreen middle layer rides on top of OpenSRF, and the user interfaces (the OPAC and staff client) ride on top of Evergreen.

In short, REST is a set of principles that establishes an architectural style for large, scalable applications.

Evergreen also uses other open source projects – there are no proprietary pieces in Evergreen – to supply needed functionality. One advantage of open source applications is that a developer working on one project can use what is already tested and proven by other projects. PostgreSQL ("Postgres") is an enterprise-grade relational database system with over 15 years of development.

It was chosen for Evergreen because it was virtually the only open source database system that had the capabilities to support a system with the database structure and the transactions load the size of PINES. Another key application is XUL (pronounced "zool") which stands for XML user interface language. Developed by the Mozilla project, XUL allows fully featured cross-platform web applications. Perhaps the best known XUL-driven application is the Firefox web browser.

There are, as you see, a number of pieces to the Evergreen puzzle. These modules work together, and each can be changed when necessary without affecting the others as long as the changes do not affect the communications among them. Using these practices, coupled with the open source framework, provided a flexible method for very rapid development of Evergreen in the early days and continues to this day.

The Interface

Focus groups were formed to ask library staff and patrons what features they would like to see and how they would like the OPAC to perform. How well they did that job is indicated by the increased usage that followed the introduction of Evergreen. For instance, year-over-year, holds increased by 30% – same libraries, same network, but a new interface that users found easier to use, and they did.

Development Process

The general process for development was "release early, release often," a process followed to this day. Small, iterative changes were made and released for comment and testing – not rare mega updates. If there were a problem, the software could be fixed relatively quickly. In addition, the software was open source so that it was developed in the open, not secretly by a small group. Many eyes looking at software makes better code.

Another aspect of Evergreen's development is "scratching an itch." That is, development tends to focus on the problems that people want fixed, not necessarily theoretical problems or capabilities dreamt up by product managers in response to real or imagined competitive pressures. Any kind of software development will have its strengths and weaknesses. Scratching

an itch will solve problems people really have, but the community does need a mechanism to guide development strategically to make sure that while solving this or that problem the software stays a coherent whole.

Development Philosophy

As a result of what was learned about the first iteration of PINES, there were several key results expected from Evergreen. The original charge was to build something “scalable, robust and fault-tolerant.” And, of course, open source. What do those three new terms mean?

Scalable means that the software can be deployed in different-sized libraries and that adding capacity is relatively easy. At the point where the decision was being made to move from the proprietary system that ran PINES, the server upgrade for the old system would have cost \$1.5 million for a large server. Evergreen development cost the GPLS much less than that.

Learning that lesson, Evergreen can be upgraded by adding “commodity” servers – that is, servers from any manufacturer or any do-it-yourself builder, and they can be added as needed. OpenSRF, clearly, allows Evergreen to scale easily.

Because Evergreen is easy to use – remember the focus groups – as a library’s users learn the software, they use it more, and growth can be done by adding relatively cheap, redundant servers, not million dollar boxes running proprietary software. Add a server or two, load OpenSRF, and you have upgraded. Evergreen runs on laptops, is currently running on a home library, and runs PINES with 16 million circulation transactions.

Robust. Evergreen was designed to be able to keep functioning in dire circumstances. When a backhoe severed the PINES network, the Evergreen staff clients allowed libraries that were disconnected from the central databases to continue checking out books. When the network was restored, the database was updated by an established procedure. There was no need to

write down transactions on paper. Consider: PINES has had a number of days recently with 100,000 circulations. If half the network were down for a half a day, that could be 25,000 sheets of paper to be managed. Not with Evergreen.

Fault-tolerant. Servers can fail, and the system will keep on functioning. The use of redundant servers allows one to fail and the others to keep on functioning. Evergreen was designed as much as possible to never go down.

The Future

Reflecting its history, Evergreen does not have acquisitions nor serials modules. Work on these two services is ongoing with releases expected by Spring 2009. It is a development project, but one coming from the library world to solve library problems.

As was observed above, it has been quite difficult in the past to update software to reflect new demands on it. It did prove relatively easy to add web awareness to most existing library software but rather more difficult to add Web 2.0 capabilities and Unicode. Consortial capabilities have proved daunting. And as a result of Evergreen such capabilities are now touted by software in the market that may soon have those capabilities, too.

How does Evergreen avoid these kinds of pitfalls of obsolescence? Maybe it doesn’t but Evergreen’s practices reflect the current best guess about how to engineer software so updating it is never as painful as it has been in the past with code bases even a few years old.

Welcome to the future, version 1. ■

Resources Mentioned in the Article

- [1] NCES/NCLIS. Enhanced Longitudinal Public Library Data File (PLDF3). Retrieved October 20, 2008 from <http://68.163.78.28/statsurv/NCES/pldf3/index.html>

The Development and Usage of the Greenstone Digital Library Software

by Ian H. Witten

Open Source Software in Libraries

Greenstone is a suite of software for building and distributing digital library collections. It is not a digital library but a tool for building digital libraries. It provides a way of organizing information and publishing it on the Internet (or on removable media) in the form of a fully searchable, metadata-driven collection. It has been used to create fully searchable and browsable collections of all kinds of documents, books, photographs, newspaper images, metadata such as library catalogues of MARC records, audio (MP3 files) and video – as well as mixed collections. Most are distributed on the web, but several collections of humanitarian information have been produced on CD-ROM for distribution in developing countries. Greenstone has been used for collections of a small handful of documents up to collections of several million newspaper articles (20 GB of raw text, 2 billion words, 60 million unique terms).

The New Zealand Digital Library project began 13 years ago. Two years later the name Greenstone was adopted for the software that was produced. It has been developed and distributed in cooperation with UNESCO and the Human Info NGO in Belgium. It is distributed under the GNU General Public License and runs on all popular operating systems. For more details see the book *How to Build a Digital Library* [1] and the website www.greenstone.org. Today Greenstone's user base hails from 70 countries, and the reader's interface has been translated into between 50 and 60 languages. Downloads from SourceForge exceed 200 per day.

This article recounts how Greenstone's international, humanitarian focus arose from a few essentially serendipitous events involving a local

Ian H. Witten is in the Department of Computer Science at the University of Waikato, Hamilton, New Zealand. He can be reached by phone at +64 7 838-4246 or by email at ihw@cs.waikato.ac.nz

collection in New Zealand's Māori language, a chance contact with a small humanitarian organization and a formal link with UNESCO. In retrospect, these events conspired to set the project's direction. We then review the immense importance of digital libraries in developing countries and the special requirements imposed by the conditions that prevail there. Finally we discuss efforts to establish regional support organizations for Greenstone in India and Africa.

First, however, let us begin by summarizing salient aspects of this open source software package and its user population.

Platforms. Greenstone runs on all popular operating systems: all Windows versions, Linux, Mac – even the iPod. It is very easy to install. For the default Windows installation absolutely no configuration is necessary, and end users routinely install Greenstone on their personal laptops or workstations. Institutional users run it on their main web server, where it interoperates with standard web server software such as Apache.

User base. As with most open source projects, the user base for Greenstone is unknown. It is distributed on SourceForge, a leading distribution center for open source software. Table 1 gives relevant download statistics;

TABLE 1. Usage Statistics

Distributed via SourceForge since:	Nov 2000
Average downloads since then:	4500/month
Currently running at:	6500/month
Proportion of downloads that are documentation:	44%
Proportion of downloads that are software:	56%
Of these,	82% are Windows binaries
	10% are Linux binaries
	4% are MacOS binaries
	4% are source
Number of people on Greenstone email lists:	600
Number of countries represented:	70
Number of messages (excluding spam):	150/month

WITTEN, continued

it also shows the number of people who contribute to the Greenstone mailing lists and the volume of traffic. The website www.greenstone.org points to a representative selection of examples of public Greenstone collections. The institutions they belong to are shown in Table 2. A survey of Greenstone users was undertaken in 2004-2005 [2].

TABLE 2. Sample Collections (see www.greenstone.org for URLs)

Afghanistan Research and Evaluation Unit	Marshall Foundation, Virginia
Almaty Municipal Library, Kazakhstan	National University of Science and Technology, Zimbabwe
Arafura Digital Archive for East Timor	
Argentina Secretary of Human Rights	New York Botanical Garden
Association of Indian Labour Historians, Delhi	New Zealand National Library
Balearic Islands Scientific Library	Oxford University, UK
Biblioteca Obispo Angelelli, Argentina	Pacific Archive for Learning and Education
British Columbia Indian Chiefs Union	Slavonski Brod Public Library, Slovenia
Cape Town University, South Africa	State Library of Tasmania
Centro de Información de Recursos Naturales, Chile	Sudanese Association of Libraries and Information
Charles Darwin University, Australia	Sudan Open Archive
Consejo Latinoamericano de Ciencias Sociales, Argentina	UNESCO
Fundación Escuela de Gerencia Social, Venezuela	United Nations in Pakistan
Council of Independent Colleges, Washington DC	Universities of Auburn (Al), Chicago, Detroit, Illinois, Illinois Wesleyan, Iowa, Lehigh, Tulane, Yale (in US)
Hawaiian Electronic Library	University of KwaZulu Natal, South Africa
iArchives, Utah	University of Malaya
Ionian University, Greece	University of Namibia
Indian Institute of Management, Kozhikode	University of the South Pacific, Fiji
Indian Institute of Science, Bangalore	Universidad Nacional de La Plata, Argentina
Institute of Rural Management, Pakistan	Vietnam National University
Kabul University	Vimercate Public Library, Milan, Italy
Kazakhstan Human Rights Commission	Washington Research Library Consortium
Kyrgyz Republic National Library	Welsh Books Council
Latin America and Caribbean Network of Social Science	

Educational usage. Greenstone forms a popular basis for practical work in U.S. library and information science training programs, and several leading institutions employ it for this purpose. Indeed the book *How to Build a Digital Library*, which contains extensive material on Greenstone, is the most frequently assigned text in U.S. digital library courses [3].

Interfaces. Greenstone has separate interactive interfaces for readers and librarians. End users access the digital library through the reader interface, which operates within a web browser. The librarian interface is a Java-based graphical user interface (also available as an applet) that makes it easy to gather material for a collection (downloading it from the web where necessary), enrich it by adding metadata, design the searching and browsing facilities that the collection will offer the user, and build and serve the collection. There is also a separate editor for creating new metadata sets and adding elements to them.

Standards. Greenstone is strongly standards-compliant. It incorporates a server that can serve any collection over the Open Archives Protocol for Metadata Harvesting (OAI-PMH), Z39.50 and SRW (Search/Retrieve via the Web). Greenstone can harvest documents over any of these protocols and include them in a collection. Collections can be exported to METS (METadata harvesting and Transmission Standard) in the Greenstone METS Profile, approved by the METS editorial board, and Greenstone can ingest documents in METS form. Any collection can be exported to DSpace [4] ready for DSpace's batch import program, and any DSpace collection can be imported into Greenstone [5]. There is also a close connection with Fedora [6], and a modified form of the librarian interface can be used to build Fedora collections [7].

Ingesting documents. An extensible plugin scheme is used to ingest documents. There are plugins for most common formats of textual documents, listed in Table 3, including PowerPoint and Excel documents.

TABLE 3. Metadata and Document Formats

Predefined metadata sets
Dublin Core (qualified and unqualified)
RFC 1807
NZGLS (New Zealand Government Locator Service)
AGLS (Australian Government Locator Service)
Metadata plugins
XML, MARC, CDS/ISIS, ProCite, BibTex, Refer, OAI, DSpace, METS
Document plugins
PDF, PostScript, Word, RTF, HTML, Plain text, Latex, ZIP archives, Excel, PPT, Email (various formats), source code
Multimedia plugins
Images (any format, including GIF, JIF, JPEG, TIFF), MP3 audio, Ogg Vorbis audio
Generic plugin

WITTEN, continued

There are also plugins for most image formats and some audio and video formats. There is a generic plugin that can be configured for other multimedia formats such as MPEG and MIDI.

Metadata. The librarian interface includes flexible facilities for adding metadata to documents. Where externally prepared metadata is available it can be ingested using plugins. Plugins for about 10 widely used standard metadata formats are listed in Table 3. (There are, in addition, some plugins for non-standard metadata).

Metadata sets. Four predefined metadata sets are provided with the software (Table 3). New metadata sets can be created interactively using Greenstone's metadata set editor.

Languages. One of Greenstone's unique strengths is its multilingual nature. The reader's interface is available in the 50 languages shown in Table 4, with another 10 in progress. The librarian interface is available in 10 languages, and the full Greenstone documentation (which is extensive) is

TABLE 4. Languages of Greenstone

Reader's Interface is available in:

Arabic, Armenian, Bengali, Catalan, Chinese (both simplified and traditional), Croatian, Czech, Dari, Dutch, English, Farsi, Finnish, French, Gaelic, Galician, Georgian, German, Greek, Hebrew, Hindi, Hungarian, Indonesian, Italian, Japanese, Kannada, Kazakh, Kirghiz, Latvian, Malayalam, Maori, Marathi, Mongolian, Polish, Portuguese (both European and Brazilian versions), Pashto, Romanian, Russian, Serbian, Slovak, Spanish, Tamil, Telugu, Thai, Turkish, Ukrainian, Urdu, Vietnamese

Other languages in progress:

Amharic, Azeri, Bulgarian, Burmese, Gujarati, Kiswahili (Swahili), Khmer (Cambodian), Malay, Nepali, Oneida (Iroquoian), Sinhala (Sinhalese), Samoan

Librarian Interface available in:

Arabic, English, French, Marathi, Spanish, Romanian, Russian Chinese (simplified), Latvian, Vietnamese in progress

Full documentation (four manuals) available in:

English, French, Spanish, Russian (three of the four also in Kazakh, Vietnamese)

available in English, French, Spanish, Russian and Kazakh.

International, humanitarian focus. Three formative, serendipitous events described in the following paragraphs had a major impact in making Greenstone the system of choice for internationalized collections of indigenous and humanitarian information.

Niupepa: The Māori Newspapers. Early on we embarked on a large collection of Māori-language newspapers ("Niupepa"), sourced from New Zealand's Turnbull Library [8]. We made an initial demonstration with a full Māori interface and sought funding from the NZ Ministry of Education to continue the work. This activity had two formative effects: we focused from the beginning on multiple-language interfaces and the ability to very quickly build small but fully functional demo collections became a valued feature of Greenstone. The full Niupepa, which was officially launched in March 2002, is still the largest collection of online Māori-language documents. It is extensively used for historical, social, legal and linguistic research, and in a moving ceremony in November 2000 the Māori people presented the Greenstone project with a ceremonial *toki* (adze) as a gift in recognition of our contributions to indigenous language preservation.

Humanitarian collections. Also in the early days, Human Info NGO sought help for producing fully searchable CD-ROM collections of humanitarian information. These collections were the vision of a Belgian medical doctor who had worked in Africa, witnessed a desperate need for such information in developing countries and hit upon electronic distribution as the solution. Unfortunately he had encountered difficulties in developing appropriate software. To bring Greenstone into line with his needs we had to make our server (and in particular its full-text search engine), which had been developed under Linux, run on Windows machines – including Windows 3.1 and 3.11 because, although by then obsolete, they were still prevalent in developing countries. This task was demanding but largely uninteresting technically: we had to develop expertise in long-forgotten software systems. However, it focused our attention on the need to run on all platforms, not just Linux.

The first *Humanity Development Library* CD-ROM was issued in 1998, closely followed by UNESCO's *Sahel point Doc*. In the latter all documents, along with the entire interface, help text and full-text search mechanism, are in French, further underscoring the project's focus on multilingual interfaces and processing non-English documents. The first multilingual collection soon followed: a Spanish/English *Biblioteca Virtual de Desastres/Virtual Disaster Collection* aimed at South America. We also began to develop interfaces in

WITTEN, continued

TABLE 5. Humanitarian CD-ROMs

2006	Appropriate Technology Knowledge Collection	En	2002	Community Development Library for Sustainable Development and Basic Human Needs v2.1	En
2005	Gender and HIV/AIDS Electronic Library	En		Food and Nutrition Library v2.0	En
	Textes de Base sur L'Environnement au Sénégal	Fr		UNDP Energy for Sustainable Development Library	En
	Educational Aids/Lehr- und Lernmittel/ Moyens didactiques/Material didáctico v3.0	En/De/Fr/Es	2001	UNAIDS Library v1.1	En/Fr/Es/Ru
2004	Africa Collection for Transition: From Relief to Development v1.01	En		East African Development Library	En
	UNECE Committee for Trade, Industry and Enterprise Development	En/Fr/Ru		Safe Motherhood Strategies	En/Fr/Es
	INEE Technical Kit on Education in Emergencies and Early Recovery	En		Researching Education Development	En
	Educational Aids/Lehr- und Lernmittel/ Moyens didactiques/Material didáctico v2.0	En/De/Fr/Es		Biblioteca Virtual de Salud para des Desastres	Es/En
2003	Education, Work and the Future/ Education Travail et Avenir v2.0	En/Fr		WHO Medicines Bookshelf	En
	Revised Curricula for Technical Colleges	En		Africa Collection for Transition	En
	UNAIDS Library v2.0	En/Fr/Es/Ru	2000	World Environmental Library v1.1	En
	Biblioteca Virtual de Salud para des Desastres/ Health Library for Disasters v2.0	Sp/En		Sahel point Doc v2.0	Fr
	Food and Nutrition Library v2.2	En		Food and Nutrition Library v1.0	En
	Educational Aids/Lehr- und Lernmittel/ Moyens didactiques/Material didáctico v1.0	En/De/Fr/Es	1999	Medical and Health Library v1.0	En
	ICT Training Kit and Digital Library for Africa	En		Bibliothèque pour le Développement Durable et des Besoins Essentiels v1.0	Fr
				Biblioteca Virtual de Desastres	Es/En
				UNU Collection on Critical Global Issues v2.0	En
				Sahel point Doc	Fr
				Humanity Development Library v2.0	En
			1998	UNU Collection on Critical Global Issues v1.0	En
				Humanity Development Library v1.3	En

collections into the hands of librarians and other non-IT specialists the world over. (In New Zealand by the way, they say, “Give a man a fish, and he will eat for a day. Teach a man to fish, and he’ll sit in a boat and drink beer for the rest of his life.”)

We began by packaging and documenting our perl scripts, and slowly and painfully came to terms with the fact that operating at this level is anathema for librarians. In 2001 we produced a web-based system for building digital libraries called the *Collector* [9]. However, this system was never a great success. Today web-based submission to institutional repository systems (including Greenstone collections) is commonplace, but back then we were trying to allow users to design and configure digital libraries over the web as well as populate them. Shortly thereafter we began a Java development that became known as the Greenstone Librarian Interface, which grew over the years into a comprehensive system for designing and building collections [10].

non-European languages such as Chinese and Russian. To date, about 40 humanitarian CD-ROM collections, listed in Table 5, have been published. We were heavily involved with the first few and then transferred the technology to Human Info’s people in Romania so that they could proceed independently.

The UNESCO connection. Human Info introduced us to UNESCO. Although UNESCO supports the idea of producing humanitarian CD-ROMs and distributing them in developing countries, they are really more interested in sustainable development. They stress the value of empowering non-technical people in a more primitive computing milieu than ours to produce and distribute their own digital library collections, following that old Chinese proverb about giving a man a fish versus teaching him to fish. We had transferred our collection-building technology to Human Info, but UNESCO envisaged a completely different proposition: to put the power to build

Requirements for Digital Libraries in Developing Countries

Digital libraries are the killer app for computing technology in developing countries. Priorities here include health, agriculture, nutrition, hygiene, sanitation and safe drinking water. Computers are not a priority, but simple, reliable access to targeted information meeting these basic needs certainly is [11]. UNESCO taught us the importance of recognizing the special conditions that prevail throughout the developing world.

Working without the Web. Digital library projects invariably presuppose usage over the web. But Internet access varies widely across the globe. Schools and hospitals in the developing world are poorly connected; even universities often have appalling access by western standards. In this environment, Greenstone’s ability to create CD-ROM-based collections is

WITTEN, continued

crucial. However, we had to develop our own installer, for we could not distribute the commercial installer that Human Info and we used to produce these CD-ROMs – and this development was before the days of comprehensive open-source installers.

Software distribution. From the outset, UNESCO's goal was to lead by example, producing CD-ROMs containing the entire Greenstone software (not just individual collections plus the run-time system, as in Human Info's products) for use by those without ready access to the Internet. Although we continue to produce them annually, they are more of symbolic than actual significance because they become outdated by new releases of the software that appear on the Internet.

Instructional material. When we and others started to give workshops, tutorials and courses on Greenstone, we adopted a policy of putting all instructional material – PowerPoint slides, exercises, sample files for projects – on a workshop CD-ROM, and we began to include this auxiliary material on the UNESCO distributions too.

Multilingual documentation. UNESCO saw good documentation as crucial. They helped us make the entire Greenstone technology available in Spanish, French and Russian – and, later, in the other two official UNESCO languages, Arabic and Chinese. We already had versions of the interface in these (and many other) languages, but UNESCO wanted everything to be translated – not just the documentation, which was extensive (four substantial manuals) but all the installation instructions, README files, example collections, warning messages from perl scripts, etc. We might have demurred had we realized the extent to which such a massive translation effort would threaten to hobble the potential for future development, and we have since suffered mightily in getting everything – including last-minute interface tweaks – translated for each upcoming CD-ROM release.

Interface translation. The cumbersome process of maintaining up-to-date translations in the face of continual evolution of the software – which is, of course, to be expected in open source systems – necessitated a scheme for maintaining all language fragments [12]. We chose to use a version control

TABLE 6. Greenstone Workshops in Developing Countries

2008	May	Mahé, Sechelles	Aug	Tirunelveli, India	
	May	Kuala Lumpur, Malaysia	Mar–Apr	Madras, India	
	Apr–May	Addis Ababa, Ethiopia	Mar	Durban, South Africa	
	Mar	Kathmandu, Nepal	Feb	Bangkok, Thailand	
	Mar	Jaipur, India	2005	Nov	Cape Town, South Africa
	Feb	Maseru, Lesotho	Nov–Dec	Arusha, Tanzania	
2007	Dec	Bangalore, India	Sep	Suva, Fiji	
	Dec	Hanoi, Vietnam	Aug	Bangalore, India	
	Dec	Bangalore, India	May	Ho Chi Minh City, Vietnam	
	Dec	Bulawayo, Zimbabwe	May	Kozhikode, India	
	Nov–Dec	Kozhikode, India	2004	Dec	Bombay, India
	Nov	Lilongwe, Malawi	Oct	Havana, Cuba	
	Oct	Windhoek, Namibia	Sep	Trirandom, India	
	Sep	Morelia, Mexico	Aug–Sep	Windhoek, Namibia	
	Jul	Kuching, Malaysia	Jul	Suva, Fiji	
	Jun	Suva, Fiji	Jun	Cape Town, South Africa	
	May	Port of Spain, Trinidad	Mar	Dakar, Senegal	
	Mar	Colombo, Sri Lanka	Mar	Cape Town, South Africa	
	Feb	Vellore, India	Feb	Gaborone, Botswana	
2006	Dec	Calcutta, India	Feb	Almaty, Kazakhstan	
	Dec	New Delhi, India	2003	Nov	Dakar, Senegal
	Nov–Dec	Kozhikode, India	Nov	Suva, Fiji	
	Oct	Vladimir, Russia	May	Bangalore, India (IISC)	

system in order to automatically determine what needs updating. This decision resulted in the Greenstone Translator's Interface, a web portal where officially registered translators can examine the status of the language interface for which they are responsible and update it. Today the interface has been translated into many languages (see Table 4), most of which have a designated volunteer maintainer.

Training workshops. Training is a bottleneck for widespread adoption of any digital library software. With UNESCO's encouragement and sponsorship we have worked to enable people to take advantage of digital library technology by running hands-on workshops. Many Greenstone workshops have been given in developing countries, ranging from half a day to six days. Table 6 lists ones given by people closely associated with the project, but there have been many others.

WITTEN, continued

Other training. The United Nations Food and Agricultural Organization (FAO) and UNESCO's Institute for Information Technology in Education have also produced training material on Greenstone. Furthermore, we have been active in conducting Greenstone tutorials at major digital library conferences – JCDL, ECDL, ICADL, ICDL (on several occasions in each case) – and library conferences such as the LITA, DLF and ALA conferences. The Payson Institute of International Development at Tulane University has run courses that use Greenstone collections as a resource in dozens of locations in Africa (Burkina Faso, Cameroon, Cote d'Ivoire, Democratic Republic of Congo, Ghana, Rwanda, Senegal, Sierra Leone, Togo) and Latin America (Argentina, Bolivia, Colombia, Ecuador, Guatemala).

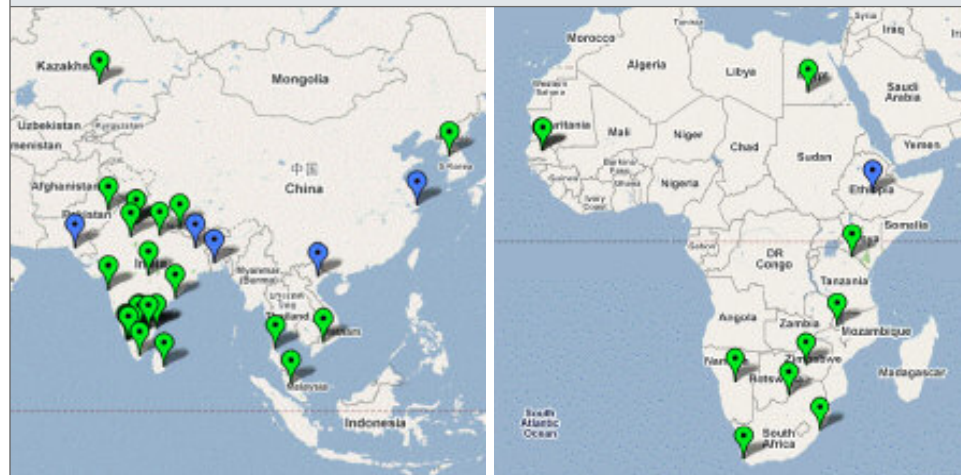
Regional Support

Recognizing that devolution is essential for sustainability, we are striving to distribute Greenstone training, maintenance and support by establishing regional support groups. There are around 60 registered volunteer language translators who provide a natural focus for informal local support. User groups for Spanish and French users have existed for some time, and a Greenstone blog for Arabic users (in Arabic) has appeared recently. However, more formal support is needed, especially in developing countries. We have concentrated our efforts so far on South East Asia – in particular India – and on Africa, particularly southern Africa. Figure 1 shows where Greenstone workshops and tutorials have been held in these parts of the world.

South Asia support group. Greenstone has been widely adopted in India, a country that is renowned for its strong library community. One of the earliest training courses was held (in 2003) at the Indian Institute of Sciences, and many other centers emerged and began to arrange locally run training courses as well as ones led by project members from New Zealand. A tutorial at the International Conference on Asian Digital Libraries in Bangalore attracted 200 attendees, and robust interest has been shown at international conferences on digital libraries in New Delhi.

In April 2006 a Greenstone Support Group for South Asia was launched, centered in Kozhikode, India. It has established its own website and an accompanying electronic discussion and support forum. It is coordinating

FIGURE 1. Greenstone workshops in South East Asia and Africa



the development of a network of Greenstone users under the supervision of an advisory committee. It has organized several workshops in the region, most recently an advanced training workshop at the Indian Institute of Sciences in Bangalore, and a further workshop is planned for early 2009 as part of an intensified Greenstone skill-development program in different parts of India.

The support group is surveying Greenstone usage in India and studying the feasibility of a more comprehensive and sustainable organization that relies less on individual volunteers and will function as a springboard for a broader participative collaboration in South Asia. In the meantime, it has initiated the production of a basic Greenstone handbook for use in schools of library and information science and is making efforts to bring other Asian countries into the network.

An important activity, particularly in this region, concerns the use of local languages. Greenstone interfaces have been established in several Indian languages, including Hindi, Tamil, Malayalam, Marathi and Telegu, with Nepalese and Sinhalese interfaces underway. The support group is also promoting the acquisition and processing of digital library content in local languages.

The South Asia support group is seen as a model for Greenstone organizations in other regions, and although it has not yet reached full self-sustainability it is probably not far off.

WITTEN, continued

Southern Africa support group. In 2005 UNESCO sponsored a study of the feasibility of setting up a Greenstone Support Organization for Africa [13]. This study began with a survey questionnaire that was widely circulated to African professionals, which underlined what we already suspected: Africa is far less developed in terms of basic awareness of the need for digital libraries than India. In order to foster the construction of digital libraries we decided to take a proactive approach. We were fortunate to receive funding from a private foundation, and we began a one-year pilot project. Greenstone User Support in Southern Africa began in mid-2007 coordinated by eIFL.net, a not-for-profit organization that advocates the wide availability of electronic resources by library users in transitional and developing countries.

The project, which focuses primarily on Namibia, Malawi, Zimbabwe and Lesotho and neighboring countries, is just approaching completion. Workshops were run at the University of Namibia Library, which was designated the regional coordination center, and at three national coordination centers: Bunda College Library in Malawi, the National University of Science and Technology Library in Zimbabwe, and in Lesotho where the National University of Lesotho Library and the Lesotho College of Education Library share responsibility. A total of 66 specialists from 10 countries were trained in basic Greenstone use and digital library techniques, with further advanced training for 13 of them from four countries. Each of these workshops has been a learning experience for the organizers and the southern African resource persons, so that there is now a pool of technical and methodological expertise to extend the training effort throughout the region.

The national centers are in the process of developing their own initial digital library applications and organizing basic Greenstone training to support the development of national Greenstone networks in their own and neighboring countries. The regional center has set up a website for the project and is hosting an electronic user support service. This is developing well, with the user discussion list providing a lively forum for technical exchange.

As the pilot project comes to an end, the national centers are being helped to evolve as digital library centers of excellence, and the participating institutions and specialists are further developing their capacities through cooperation and self-help. A survey of potential and

actual Greenstone users is being organized. The next step is for users to consider how to continue and reinforce their cooperation as a sustainable regional support network. There is some way to go before the network reaches the same degree of potential self-sustainability as the South Asian group, and we have secured funding for a follow-up project to continue the work.

Conclusions

We would like to underscore the enormous importance of digital libraries for the developing world. Most digital library research is conducted in libraries whose purpose is scholarship, and from most people's perspective such libraries often seem esoteric. But they are not necessarily so. Digital libraries are the killer app for information technology in developing countries: they provide a low-cost way of distributing organized information widely throughout the vast Internet-challenged regions of the world. In comparison, digital library technology is relatively unimportant in developed countries, according to some, because there are so many alternative sources of information.

Sustainability is one of the greatest challenges for open-source projects with international user populations – particularly when the users are not programmers and when much of the usage is in poor countries. Our approach is to foster the establishment of regional support centers that provide a variety of functions: training, technical support, documentation, software internationalization and localization, discussion, inspiration and visibility. Whether these efforts will result in a truly sustainable infrastructure for Greenstone has yet to be seen.

We have been asked the secret to success when striving to build a community around a piece of open source software. Some think we must have found it since we have a wealth of activity spread around the globe, volunteers who translate the interface into local languages, associations with UNESCO and other international organizations, and support activities moving into autonomous regional centers. In fact, luck and serendipity played a major role. And, indeed, there have been many failures, too: initiatives that did not come off, sources of support that we have been unable to tap, opportunities that knocked but were unheeded. The secret of

success (as a thousand self-help books on financial management will tell you) is that there is no secret of success. Open source software projects are built upon a compelling vision and an excellent implementation. But equally important is the need to communicate that vision and what the software can do – clearly, widely and enthusiastically. Perhaps that necessity is the hardest part.

Acknowledgements

I warmly acknowledge the entire New Zealand Digital Library Project team for their unstinting work in providing an environment that makes this kind of research meaningful – and enjoyable. John Rose, formerly of UNESCO, has been an inspiration. Eric Morgan contributed a key insight. And thanks to our users for making it all worthwhile. ■

Resources Mentioned in the Article

- [1] Witten, I.H., & Bainbridge, D. (2003). *How to build a digital library*. San Francisco, CA: Morgan Kaufman.
- [2] Sheble, L. (2006). *Greenstone user survey*. Detroit, MI: Wayne State University.
- [3] Pomerantz, J., Oh, S., Yang, S., Fox, E.A., & Wildemuth, B.M. (2006, November). The core: Digital library education in library and information science programs. *D-Lib Magazine*, 12(11). Retrieved October 21, 2008, from www.dlib.org/dlib/november06/pomerantz/11pomerantz.html.
- [4] Smith, M., Bass, M., McClella, G., Tansley, R., Barton, M., Branschofsky, M. Stuve, D., & Wakler, J. (2003, January). DSpace: An open source dynamic digital repository. *D-Lib Magazine*, 9(1). Retrieved October 21, 2008 from www.dlib.org/dlib/january03/smith/01smith.html.
- [5] Witten, I.H., Bainbridge, D., Tansley, R., Huang, C.Y., & Don, K. (2005, September). Stoned: A bridge between Greenstone and DSpace. *D-Lib Magazine*, 11(9). Retrieved October 21, 2008, from www.dlib.org/dlib/september05/witten/09witten.html.
- [6] Lagoze, C. Payette, S. Shin, E., & Wilper, C. (2006). Fedora: An architecture for complex objects and their relationships. *International Journal on Digital Libraries*, 6(2).
- [7] Bainbridge, D., & Witten, I.H. (2008). A Fedora librarian interface. *Proceedings of the ACM/IEEE-CS Joint Conference on Digital Libraries*, 8, 407-416.
- [8] Apperley, M., Keegan, T.T., Cunningham, S.J., & Witten, I.H. (2002). Delivering the Māori-language newspapers on the Internet. In J. Curnow et al. (Eds.). *Rere atu, taku manu! Discovering history, language and politics in the Māori-language newspaper* (pp. 211-232). Auckland, New Zealand: Auckland University Press.
- [9] Witten, I. H., Bainbridge, D., & Boddie, S.J. (2001). Power to the people: End-user building of digital library collections. *Proceedings of the ACM/IEEE-CS Joint Conference on Digital Libraries*, 1, 94-103.
- [10] Bainbridge, D., Thompson, J. and Witten, I.H. (2003). Assembling and enriching digital library collections. *Proceedings of the ACM/IEEE-CS Joint Conference on Digital Libraries*, 3, 323-334.
- [11] Witten, I.H., Loots, M., Trujillo, M.F., & Bainbridge, D. (2002). The promise of digital libraries in developing countries. *The Electronic Library*, 20(1), 7–13.
- [12] Bainbridge, D., Edgar, K.D., McPherson, J.R., & Witten, I.H. (2003). Managing change in a digital library system with many interface languages. *Proceedings of the European Conference on Digital Libraries ECDL2003*, 7, 350-361.
- [13] Peters, D.P. (2006, January). *Feasibility study on the establishment of a Greenstone Support Organization for Africa*. Retrieved October 21, 2008, from www.greenstone.org/docs/GSOA%20Feasibility%20Study.pdf.

From Open Source to Open Libraries

by Thomas Krichel

Open Source Software in Libraries

Most contributions in this issue are concerned with open source software (OSS) in libraries. Their basic angle is to look at what is being done with OSS in libraries – or what can be done. This contribution takes a broader look. It outlines a number of direct correlations between the functions of libraries and the characteristics of OSS, and by extension, how the principles of OSS can be applied to the distribution of “open libraries” as a future direction for librarianship. Software is nothing but information. The OSS communities create and maintain a bundle of highly structured information for free. What are the implications for the library community? Can they learn something for the open source communities? In other words, I want to look at what can be learned from the OSS software to understand the changing nature of libraries. Libraries are changing dramatically at this time because we are moving from print storage to digital storage and from slow physical transport to fast transport via computer networks.

Some Theory

I will start by drawing a parallel between software and libraries. It may appear to be far fetched, but I hope it is nevertheless interesting. To draw the comparison, we have to put software and libraries on an equal footing. A library is most commonly thought of as a service. In our context this description is especially true when we are referring to digital libraries.

Thomas Krichel is affiliated with both the Palmer School of Library and Information Science at CW Post Campus of Long Island University and the Information Systems Division of the Faculty of Information Technology, Novosibirsk State University. He can be reached by email at krichel@openlib.org or at his website:

<http://openlib.org/home/krichel>

Normally, software will be thought of as something enabling a service like a digital library. But I would like to look at the software itself as a service. Thus, I will treat both libraries and software as services.

Let us start with the software service. Conceptually, a software service can be thought of as three things. First there is something that a user can use. I can open a document in, say, OpenOffice, and I see a bunch of interface elements such as a prompt for keyboard input, some buttons and some icon that moves with the mouse. These interface elements allow me to manipulate a document. In principle, I can imagine another interface to the software service. And often enough, the same piece of software supports slightly different interfaces depending on what computer system it runs on. Second, there is the code that makes the software work. This code is usually some textual data called the “source code.” OSS is software for which the customer who acquires the software also gets the source code. Finally, there are the objects that the software service manipulates. Some may protest that the objects manipulated by the software service are separate from the software itself, but surely, every piece of software is tailored to the objects it manipulates. For example, picture-editing software needs to know the structure of the picture that is its underlying object. If that structure would change, the software would be next to useless, and the software service would be broken. Of course, OSS is not about making manipulated objects freely available. But generally, if many manipulated objects are freely available that’s good for the software itself, because it is cheaper to get hold of existing objects to manipulate.

Let us turn to the library service. In a similar way as I have explained for the software service there are three elements to a library. There is an interface through which the collection can be accessed. Whether the library is a building or whether the library is a digital collection accessible via a

KRICHEL, continued

website does matter to the interface. Both types have very different interfaces. What is important here is that the interface can be thought of as a separate component of the library. For example, we can move a physical collection from one building to another. Only the interface changes. Second there is the description of the collection. Like the middle component of the software, this description is the central part of the library. It contains descriptions of the objects held, as well as links between the objects and the users. Finally, there are the objects that the library holds. In a digital library these objects are useably referred to as “full-text files.” In a physical library they are physical books and periodicals. Again, as in the case of the software service, the objects that are manipulated by the library do not necessarily have to be freely

TABLE 1. Comparison of software services and library services

Elements	Software Service	Library Service
Interface	Mouse button, text field	Building, staff
Organization	Source code	Catalog
Objects of interest	Files	Books

available, but it will help the library if they have liberal licensing conditions.

Thus we can think of the source

code as the core of the software and the description of objects as being the heart of the library. Then we can draw a parallel between open source software and open libraries as shown in Table 1.

“Open Library” as a Term

I have been working on practical open library development. But as sometimes I am asked what I actually do, I have had to do some thinking about the topic.

I suspect that I was the first to think of an open library as a freely available collection of descriptions of digital objects that people can reuse and/or change, just like developers can reuse and change open source computer code. At the PEAK meeting at the University of Michigan in March 2000, I presented a paper “RePEc, an Open Library for Economics,” archived at <http://eprints.rclis.org/archive/00014408/>. In September 2004 Michael E.D. Koenig and I presented a paper “From open access to open libraries: Claims and visions for Open Academic Libraries,” archived at <http://eprints.rclis.org/archive/00002202>. These papers have some early

thinking about the concept. My concept of an open library is probably best described in these papers. It reflects the creation of freely available digital libraries that are independent of end-user services or any specific usage an end user might make of them. Still, the idea is quite concrete because OSS movement has inspired it.

A Look at OSS from a Social Perspective

The concept of an open library is very closely aligned to what OSS is about. OSS is really not a project that one organization runs. Rather it is very large set of small-scale projects, many of which achieve great things because they are compatible with others.

A bit of history helps. Since the 1980s Richard M. Stallman has called for GNU. GNU stands for “GNU is not UNIX.” It is a free replacement of UNIX. UNIX was a popular operating system. The way the UNIX operating system is built helps the work to replace it. UNIX is not a monolithic system. Instead, it has a lot of components that all work together. Thus GNU project participants did not have to write the whole thing from scratch. Instead, they could start by rewriting utilities such as “ls,” a program that lists file names. The GNU version of “ls” was a drop-in replacement for a standard version in (almost) any UNIX system. GNU versions of such utilities were very often faster and always more full-featured than the standard version. There are lots of ways to write out a list of files. Modern versions of GNU “ls” cover all the most useful ones.

Still, I guess that in the early days, most people were skeptical about whether the GNU project could be successfully completed. And it is not really quite complete. But the spirit of GNU lives on, and it lives more strongly than most people ever expected. Free operating systems for computers are a reality. They may not be called GNU systems, but they are free in the sense that Stallman envisioned.

Nowadays computers have a lot more functionality than they did in the 1980s. Putting many of these components together on a single computer generates a very complicated system. Let me illustrate this point with a simple example. My operating system of choice is Debian GNU/Linux. The system consists of a set of packages. When I looked at it in May 2008, there

were 22456 packages available in my (typical) installation. Each package provides a particular functionality. When I want to add a new functionality to my computer, I add a package, say package *A*. But packages are not independent. More often than not, when I add a new package, I am told that I have to install a bunch of other packages as well because without these, my desired package *A* will not run. And there are also other packages that are suggested by package *A*. I am told that when I run package *A*, I may also install package *B* and *C* that are just friends of package *A*. Actually, on a technical level things are even more complicated. Each package comes with a version number. Package *A* version 1.0 may require package *B* version 2.0 or higher. It may be incompatible with version 2.1 of package *B*. And so on. You get the idea.

Before I bore you with more technical details, let's move away from technology and look at people. Let's look at people who package the software. Let us call them packagers. Most packagers are not the authors of the software they package. They know the software well, and they know the operating system well. They work at the interface between the software and the operating system. They take the software as an input and contribute to the operating system. In doing that task, they may change some aspects of the software. They make these changes because the software can be changed. After all, this feature is what open source is all about. It is open not only for reading but also for writing. Thus it can be adapted to the requirements of the operating system. Such requirements are, for example, that the operating system requires packages to work together. Or, for another example, that a user may change aspects of the software but still want to have it update gracefully when the latest and greatest version of the software comes out at the operating system level.

So you start to understand how come all this OSS is available for free. A person who packages a piece of software will spend only a few hours a week on this activity. She can do this, essentially, in her spare time. So the key to making a huge piece of complex information available is to split the task into small bits and assign a volunteer to each bit. This strategy is a key to success.

Another key to success is reuse. And reuse in software comes in the form of libraries. All pieces of software rely on libraries. Yes, geeks use libraries too. But their libraries are actually computer files. These special files contain

compiled pieces of code that have already been written by somebody else. When I write software for my digital library systems, I use a language called Perl. The code that I write in Perl is a simple text file. But I am not writing all my software from scratch using the commands that Perl provides. Instead I use some structures of Perl code called modules that contain Perl code that has already been written by somebody else to enable common tasks. These modules form a library of code. So modules are one way we can facilitate reuse. In a similar way, Perl itself is written in a language called C, as immortalized in the Beatles song "Write in C." C code itself relies on C libraries to achieve common tasks such as showing a character on the screen. The maintainers of Perl reuse these libraries. In the same way, when you use a website, the web server, most likely Apache, will answer your requests. The web server software is also written in C and uses the very same libraries that Perl uses to do common tasks, for example reading a file.

An Example

Can information professionals create and maintain open libraries, just like computer professionals create and maintain freely available software? From what we have learned in the previous paragraph, it should appear infeasible. In the same way that Richard M. Stallman has challenged computing professional to create free software, I challenge information professionals to maintain open libraries. I would not do it had I not already created one, the RePEc open library (see <http://repec.org>) for research in economics.

I formally co-founded RePEc in 1997, but it really goes back to efforts I made in 1993 to collect information about scientific papers in economics and make it freely available. Today, RePEc is based on over 900 RePEc archives. These sites furnish bibliographic information that can be harvested. User services are separate from these archives. They provide aggregates of the bibliographic data for end users. Thus the provision of archives and the provision of interfaces of the data in the archive are separated. Archive maintainers supply data that is used in a variety of interfaces. The more widely the data is seen, the more interest they have in keeping it up to date. User services then have a wide set of data to show that attracts wide usage. But as a whole the most important thing about RePEc is that it is sustainable

KRICHEL, continued

without external subsidy. RePEc has no budget, no officers, no meetings and no formal decision making process. And it no longer relies on a single person to keep things together, as it did when I got things started.

How did RePEc come about? If you start with empty archives, you have empty user interfaces. Who will start the first archive, and who will build the first user interface? Well, I did a lot of that in the first year, 1993. In 1994 José Manuel Barrueco Cruz joined me. By the time we got funding from the Joint Information Systems Committee of the United Kingdom's higher education funding bodies, in May 1996, we already had collected 2500 descriptions of online papers, mainly manually. Although I can't prove this observation scientifically, just having a few people compile bibliographic data is not how things have really scaled up. In our case of an academic open library the thing that really worked to bring in community involvement – yes, that is what is needed – was to build an author registration system. In RePEc's case, there is now a special interface that authors use to register the works that they have written and that are catalogued in the RePEc data. In 1998 I supervised a student, Markus Johannes Richard Klink. We got the registry to run 1999. The system is now called the RePEc Author Service.

What's the big deal about author registration? Well, on its own, nothing much. Author registration has to be seen within the wider context of an academic digital library. Author registration allows us to put the papers of a person on a common page. This page starts to look like part of a CV, but there are two crucial differences. Instead of the researcher having to compile and format the bibliographical information on her own, here she can just say what papers she has written and the system will find and format the data. Second, the publication list is reusable across further services. The key is reusability. A CV is not reusable, but the author registration record is. Because the data is reused, authors have strong incentive to maintain the data. They also have incentives to add papers to the RePEc dataset in order to keep their document record up data. This benefit has been the key to getting authors involved in populating RePEc archives, sometimes even building new ones.

Author registration allows for a battery of author rankings to be calculated from various measures of usage. Typically such measures involve the number of abstracts viewed and the number of full-text downloads.

Interestingly, although such usage is distributed across RePEc services, the services collaborate to build a cross-service usage database. This dataset is called LogEc. It was built by Sune Karlsson. He also maintains it. Other measures of usage include citations. CitEc, a system built and maintained by José Manuel Barrueco Cruz, is the citations database of RePEc.

Author registration illustrates the parallels between open library and open source. I earlier identified decentralization and reuse as key for open source software. Clearly RePEc could not register authors through the use of its dedicated staff: there is no staff. But there are tons of authors to register and even more papers to associate with them. Authors have to do it themselves. They will do it because the profiles they create are used in various forms across a battery of RePEc services.

From that experience, I believe that building an author registration service is a crucial component of future academic digital libraries. It is particularly relevant when it works with the open access repositories. Therefore, I am working on an interdisciplinary author registration system called AuthorClaim at <http://authorclaim.org>. In parallel, I am working on a list of academic institutions at <http://ariw.org>. *Ariw* stands for “academic and research institutions in the world.”

Reasons to be Cheerful

At first some readers may object that digital libraries contain objects that are produced with a profit motive in mind. Such readers should remind themselves that I am writing about digital libraries as descriptions of objects. If the objects themselves are freely available for use, that's all the better. But they don't have to be free. Recall that OSS does not prevent its users from creating objects with a profit motive. You can use OSS to create a piece of poetry and charge people to download a copy of the poem. Similarly the open library as envisioned here works at the level of the description of the objects. And remind yourself that not all underlying objects in libraries have been written with an immediate profit-from-sale purpose. For example, I have not been paid to write this paper. If the object is available for a fee, its description is less valuable. But on the other hand, the copyright holder has better incentives to contribute to the open library

because the library advertises the copyrighted material, which should help to bring revenues to the copyright holder.

At some universities library professionals have already started to build repositories for the institution. Such repositories don't qualify as libraries. They really are publishers. But they do help to build libraries because some metadata about these documents is readily available. The data is not precise, but it can be used to, say, put together an author registration service. For such a service, you only need author name expressions, titles, some publication information and a link to further information, be it an abstract or full text. I am currently working on integrating such institutional archives into AuthorClaim.

In recent years a lot of progress has been made in the technologies that enable the reuse of textual data. First, XML has established itself as a lingua franca of textual data. There is a large set of technologies that surround XML. Most of those are well implemented in OSS.

An example of such a technology is XML Namespaces. The NameSpace allows mixing of elements from different metadata systems (or namespaces). In simple terms, it allows you to use different vocabularies of terms in a single document. This facility is useful when combining or extending sets of metadata. For example, Peter and Paul can describe a bunch of objects, each using a different set of properties.

Here is an example from my own work. Most of the data collection used a format called AMF (<http://amf.openlib.org/doc/amf.html>). Much of the design of it I did myself in 2000. The format has been very useful. It allows me to describe objects of my interest, in the scholarly communications domain, in the way I want to describe them. But sometimes I have some special requirements that I can't describe in AMF as it stands. I can extend AMF but there is no sense in extending a vocabulary in ways that are only useful to solve a particular issue that I have with a specific system. AMF is built as an open format. It accepts foreign vocabularies from other name spaces pretty much anywhere. Thus I can pick and mix my metadata. This feature is very useful when merging datasets. XML Namespaces are to digital information what DNS (Domain Name Service) is to networks, but without the registration hassle. However, it is still difficult, even for me, to escape the idea of a record having a fixed structure, with a number of fields

that may be optional or required, single-occurrence or repeatable.

Technological developments will fuel the developments of open libraries by reducing costs. Disk space, computer power and network throughput have become a lot cheaper and will become a lot cheaper. Network access is more widespread. Text on computer screens is becoming more readable, and prices of high quality screens continue to come down. Hosting remains a significant expense, but, for my own work I don't have too many problems relying on donated hosting.

Another piece of good news is that the primary publishing business, the provision of content, also has made great strides in the provision of open-access information in a loosely structured form. The best example is Wikipedia. Making Wikipedia a lot more structured is difficult because of the general level that it operates at. But the example illustrates well that free information can be of a very high quality indeed. This positive example is important because free is still viewed as cheap and cheap as bad.

Metadata Example

Decentralization of metadata maintenance is a business issue. But the business model has to be accompanied by changes in the way the metadata is encoded. We need to consider more metadata about relationships among entities.

Let us consider an example from RePEc. I am cutting the examples a bit to conserve space. Here is a working paper from the International Monetary Fund.

```
<text id="RePEc:imf:imfwpa:04/17">
<type>preprint</type>
<title>Interest Rate Volatility and Risk in Indian Banking</title>
<abstract>The easing of controls on interest rates has led ,Ä¶</abstract>
<keywords>Interest rates , India , Banks</keywords>
<status>The text is part of a series Working Papers Number 04/17 27
pages</status>
<date event="created">2004-02-13</date>
<file>
<url>http://www.imf.org/external/pubs/ft/wp/2004/wp0417.pdf</url>
<format>application/pdf</format>
</file>
```

KRICHEL, continued

```

<hasauthor>
<person>
<name>Ila Patnaik</name>
</person>
</hasauthor>
<hasauthor>
<person>
<name>Ajai Shah</name>
</person>
</hasauthor>
<ispartof>
<collection ref="RePEc:imf:imfwpa"/>
</ispartof>
</text>

```

The collection in which the paper was published is described in a separate record:

```

<collection id="RePEc:imf:imfwpa">
<type>series</type>
<title>IMF Working Papers</title>
<description>International Monetary Fund Working Papers</description>
<haspublisher>
<organization ref="RePEc:edi:imffus"/>
</haspublisher>
</collection>

```

The IMF, as the publisher of the collection, is described in a separate record. This data is collected by EDIRC, a central service that registers all economics department and research institutions.

```

<organization>
<name>International Monetary Fund (IMF)</name>
<email>publicaffairs@imf.org</email>
<phone>(202) 623-7000</phone>
<postal>700 19th Street, N.W., Washington DC 20431</postal>

```

```

<homepage>http://www.imf.org/</homepage>
<fax>(202) 623-4661</fax>
<postal>Washington, District of Columbia (United States)</postal>
</organization>

```

This paper has been claimed by an author to be hers. The author produced this data using the RePEc Author Service.

```

<person id="RePEc:per:1964-04-27:ila_patnaik">
<email>ilapatnaik@gmail.com</email>
<postal>National Institute of Public Finance and Policy Satsang Vihar
Marg New Delhi 110067 INDIA</postal>
<homepage>http://openlib.org/home/ila</homepage>
<name>Ila Patnaik</name>
<familyname>Patnaik</familyname>
<givenname>Ila</givenname>
<ispartof>
<organization ref="RePEc:edi:nipfpin"/>
</ispartof>
<isauthorof>
<text ref="RePEc:ind:icrier:114" />
</isauthorof>
<isauthorof>
<text ref="RePEc:ind:icrier:108" />
</isauthorof>
<isauthorof>
<text ref="RePEc:wpa:wuwpri:0501003" />
</isauthorof>
<isauthorof>
<text ref="RePEc:nbr:nberwo:11387" />
</isauthorof>
<isauthorof>
<text ref="RePEc:imf:imfwpa:04/17" />
</isauthorof>
</person>

```

KRICHEL, continued

The paper was mentioned in the *NEP: New Economics Papers* report on financial markets, October 22, 2005. This data is contributed by that service.

```
<collection id="RePEc:nep:repsum:nepfmk:2005-10-22">
<haseditor>
<person ref="RePEc:per:2005-12-06:carolina_valiente">
<name>Carolina Valiente</name>
</person>
</haseditor>
<ernad:time>1130233735</ernad:time>
<haspart>
<text ref="RePEc:imf:imfwpa:05/88" />
<text ref="RePEc:imf:imfwpa:04/181" />
<text ref="RePEc:imf:imfwpa:04/17" />
<text ref="RePEc:geo:guwopa:gueconwpa~05-05-17" />
<text ref="RePEc:imf:imfwpa:05/162" />
<text ref="RePEc:imf:imfwpa:04/86" />
<text ref="RePEc:imf:imfwpa:05/181" />
<text ref="RePEc:imf:imfwpa:04/196" />
</haspart>
</collection>
```

There is also a different version of the last set of data that contains all papers in the report with as full descriptions as were available at the time of the report. This latter set of data is important for internal housekeeping at NEP and to feed the statistical learning procedures that help editors compose report issues. In addition, reports also have collection metadata attached to them, which shows the current editor of the report. The record above, which is specific to an issue of the report, shows data about the editor who prepared the issue in which the paper was published, but this editor is not the current editor.

The Obstacles

There are powerful obstacles to achieving open libraries. I have three for you here. First, there is technical incompetence; then, there are two more sophisticated problems that I call the “myth of industry” and the “myth of the full text.” Let me elaborate on the three obstacles in turn.

Technical incompetence is a huge problem. Unicode, XML and its related technologies such as XML Schema and XSTL, CSS, SQL, OAI-PMH and OAI-ORE, operating system skills, basic knowledge of networking, and above all, knowledge of a scripting language such as Perl or PHP – it all adds up to a large body of knowledge. While it is not required that every digital library builder have a deep knowledge of each of these areas, a deep understanding of at least a few of them, as well as having the programming skills, is required. Without this foundation, we can’t get started. Usually none of these technologies is taught in library schools. For years, I have been battling to introduce at least a small part of this body into the curriculum of my school, without much success. As a result the average library school graduate has almost no chance of getting involved in digital library building. One may argue that this work can be left to technical staff and that library staff only need to design the system. To characterize how absurd this idea is, I use the analogy of a person who wants to be a singer but has no voice. You can’t turn to this person and say, “OK. No problem. You can’t sing, but you can just imagine how a piece should be sung, and somebody else will sing for you.” Without having studied the technical underpinnings, library staff lack the analytical reasoning skills that are required even to get started with the design of new systems. All they will be able to say is, “Oh, it should be user friendly.” As a result, innovation in libraries is stifled. There is a tendency to contract out all developments that involve digital information skills. As I wrote in a mailing list recently, “Libraries are outsourcing to their death.”

A second problem is what I call the “myth of industry.” It is a term that I coined myself. It is the tendency of people involved in digital library work to protect their work. They put up obstacles to its reuse. Their idea is that they have built the data, and therefore they want to keep tight control over its usage. As a result you have to ask them to get a copy of the data, and more often than not, the answer is no. For example, it has not been possible for me to get a copy of the Astrophysical Data Service to use in the AuthorClaim registration service. What industry mythers do not understand is that by giving the structured data away freely, they encourage reuse of the data. It means that their own contributors have better incentives to contribute

data since it is more widely used. In other words, by giving away their structured data, open access publishers and digital library builders add value to their collections. It will still take a long time until this point sinks in.

Finally, there is the worship of the full text. There is too much emphasis in libraries on the problem of users reaching full text – where I take a wide view of what “full text” actually is. Many people place the full text of resources at the center of their collection development. If the full text is really a textual object, and if it is freely available as it should be, it can be quite easily indexed by a full-text engine, say Google. Thus textual metadata attached (in some form) to the full-text is not really important. The same textual string can also be found in the full text. So people put documents on a website, have them indexed by Google and say “That’s it, I am done.” This approach works if the text is an announcement of your next birthday party, but it appears insufficient when we deal with important documents such as scientific papers, legal codes, technical documentation or works of art. In these fields we are typically not only interested in getting access to the full text, but, in fact, we are also interested in the links among these object. For example, in patent data, we are interested in such things as citation links between patents, who applied for the patent and whether the patent was approved. In academic work we need to know who the author is. In preservation we need to know what general class a full-text object belongs to so that we can reach a decision about whether and – if yes – how to preserve it. All these concerns require registries. And these registries have to be compiled partly at least by hand. This registry creation is the job of digital librarians. Digital librarians are required to set up registries, to monitor their contents and, sometimes, even to populate them. Registries make a digital library more than just a collection of http-available computer files. But work on registries cannot progress unless more people realize their importance. Thus, in digital libraries, the full text should not be considered of central importance. Rather, it should be considered to be a metadata attribute.

Conclusions

Libraries traditionally have been working with non-free information. They have argued that resources should be pooled to purchase access to

such information for community members. Their promotion of free information has been hypocritical. They have advocated free access to information as long as it requires paying libraries to provide it.

The most important trend libraries are facing is the increase of free access information resources. Nowhere is this more obvious than on the web. More and more serious information is being made available for free on websites. Project Gutenberg was an early starter. Many newspapers, for example, have been building websites and offer much of their content for free on these sites. Many institutions offer important information about themselves on websites. Encyclopedic knowledge is more widely available than ever thanks to Wikipedia.

Generally, we see societies moving from an economy of information to an economy of attention. In the economy of information, information is rare and attention is plentiful. In the economy of attention, it is the opposite. The fact that we now have a freely available computer operating system is a part of the attention-economy trend. So far the library sector is stuck in the economy-of-information track. It will wither if it does not get out of there.

Libraries have the opportunity to participate in the creation of open libraries that provide structured information on behalf of community members for free reuse by others, which can be a value-added business model for them. Building open libraries requires technical skill current librarians generally don’t have. It requires a business sense they have problems perceiving. And it requires a change in purpose that they are slow to accept. Therefore, I am not optimistic about the future of the formal library sector. But, of course, open libraries that are modeled after the open source movement are here to stay.

Acknowledgements

Some of this paper was written while I enjoyed the hospitality of Siberian Federal University www.sfu-kras.ru/. I am grateful to Eric Lease Morgan for comments that have helped to improve the paper, including the suggestion to add the metadata example. ■

Students as Technology Leaders

by Laura Krier

Laura Krier is systems library assistant in the Simmons College Beatley Library. She can be reached by email at laura.krier@simmons.edu or at her website: www.lauraek.net.

Editor's note: Each year ASIS&T student chapters compete for ASIS&T Student Chapter-of-the-Year honors. I ask the winning chapter to recruit a student to serve for a year on the *Bulletin* Advisory Board and to write a column for us. In 2007 the winners were the student chapters at UCLA and Simmons College. Laura Krier is our current student member from Simmons.

Last summer I attended Podcamp Boston, an un-conference dedicated to the myriad uses of social media. As I sat through sessions on marketing, increasing your followers, monetizing your personal brand and the best equipment for podcasting, I tried to uncover the bits and pieces that would be useful for libraries. I wanted to try to answer the question that so many librarians seem unable to answer: Just what is all this social software for, and how are we supposed to be using it in libraries?

I've been reading about social media and libraries since my first day in library school, and most of what I read raises more questions than answers. Librarians are certainly interested in social media, and we know we should be using it, but we're not yet sure how. Libraries are experimenting with Flickr, blogs, wikis and Facebook accounts, but are presented with the same questions with each new Web 2.0 application: What are we doing here? And do our users even want us here? Do college students want to befriend the library in a virtual world? Do people need to see pictures of librarians on Flickr? And what are we supposed to be writing about on all these blogs? Despite the work of forerunners like Meredith Farkas, whose book *Social Software in Libraries* is quickly becoming required reading, a lot of uncertainty remains, and with every blog post and article, the phrase Library 2.0 seems to become more muddled.

I hoped, as I sat through sessions at Podcamp and met new media aficionados, that one of these sessions would help me think of new and effective ways for libraries to get

involved in this online social realm. And finally, near the end of the weekend, someone did.

Kabren Levinson is a remarkably self-possessed 18-year-old, who had just graduated from high school and was at Podcamp to talk about the technology group he'd founded as a senior project (www.kabrenlevinson.com). This group was a consulting group of sorts, made up of students and staff whose task it was to research technologies and figure out the best ways to make use of them in their particular educational environments. The key to this group, and the element that stood out to me, was the integral involvement of students.

We talk about our current generation of students as technology leaders. We discuss these millennials or digital natives or N-genners, whatever name they're going by today, as though they were born in a social media bubble. We consider them the experts, and yet, we rarely involve them in our decisions about software implementation. Kabren Levinson's presentation that July afternoon made me realize that if we want to know where and how our students want us to be in their online social networks, we should ask them.

This isn't a particularly new idea. The focus on user-centered services is prominent in almost everything I read and for every librarian I talk to. We want to put our patrons first, and we want to ensure that we're providing the tools patrons need to work better and more efficiently. But in the year I've worked as a systems assistant in my college's library, I haven't heard anyone ask the students what they want, and I haven't read about too many other libraries asking students about their needs, either. Sure, we test the usability of our

websites with patrons, and we assess students when we provide information literacy training. But we don't talk to them about their ideas for how the library could serve them better.

This kind of research can take place on a small scale or a much larger one, depending on the resources available at your institution and the amount of buy-in there is for social software in the first place. A roving librarian can ask students in the library what tools they think would be useful and how the library could better meet their needs online. A focus group could be convened to introduce various tools and interfaces to find out what students think of them and whether they could be improved or should be forgotten all together. A simple survey could go out asking students what tools they'd like to use.

But I think it would be really interesting to see a library form a student committee dedicated to exploring the role of social software in libraries. Many students are engaged and active on campus, and it might not be as hard as we sometimes think to get them interested in contributing. A group like this could play with new tools, talk about what works and doesn't work for them and get additional input from their friends and classmates. They could beta test any new social media tools the library is planning to implement to determine whether they're worth fully developing. They could provide the information libraries need about how and where they should be on the web. A group like this would provide valuable input for the library and good extra-curricular fodder for students' resumes.

The point, no matter how it's undertaken, is to engage students in conversations. If we are going to continue to call our students technology experts, we should take advantage of their expertise and rely on our students as our technology leaders. Maybe then we can stop asking ourselves what they want us to do. ■

Selected Abstracts from *JASIST*

Authors who choose to do so prepare and submit these summaries to the editor of the *Bulletin*.

From *JASIST* v. 59 (12)

Hartley, J. & Betts, L. (2008). *Revising and polishing a structured abstract: Is it worth the time and effort? (1870-1877)*.

Study and Results. This study tested whether readers can detect tiny changes made to text in order to improve its clarity. Over 200 information scientists and authors of academic articles rated electronically an original and a revised version of a structured abstract. The results showed that the revised aims, results and conclusion sections, and the abstract as a whole, were all rated significantly clearer than were the original texts.

What's New? Few studies address the effectiveness of revising texts despite the fact that authors spend a good deal of time doing it. Here the reasons for making the revisions are discussed in detail and tested. The findings – that changes are detectable and are thus worth doing – have clear relevance for information science and the scientific community.

Limitations. This research is a close study of work with one abstract. Further research is needed with different kinds of text, participants and texts from different cultures. Such quantitative studies also need to be complemented by qualitative ones.