

# From Open Source to Open Libraries

by Thomas Krichel

## Open Source Software in Libraries

**M**ost 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](mailto: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

KRICHEL, continued

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

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

KRICHEL, continued

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/](http://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. ■