"Open Source Library Automation: Overview and Perspective"

A chapter from

Library Technology Reports

Expert Guides to
Library Systems and Services
by Marshall Breeding

Chapter 1

Open Source Library Automation

Overview and Perspective

This is a time of major transformation in the library automation industry, and the open source software movement has found fertile ground among libraries. Many libraries are moving away from proprietary integrated library systems in favor of open source software. The dynamics of the industry have changed dramatically in recent years—until recently, libraries had largely acquired propriety automation systems from a clique of specialized vendors following the traditional software licensing models. The open source movement has disrupted long-established patterns, introducing a new way of thinking about the development and distribution of software, new products, and a new set of companies seeking to compete against the status quo.

In this issue of *Library Technology Reports* we provide extensive information about the emerging open source software movement as it applies to integrated library systems. As libraries make decisions about what software to use when automating their operations, it is vital for decision-makers to have a solid grasp of the available options. In the past, our options were differentiated on the basis of features, functionality, price, and performance of the software and on the perceived ability for a given company to develop its products into the future and provide adequate support. Do these factors differ with open source ILS products? As we explore open source software, we hope that readers will become well equipped to make informed decisions regarding whether or not this approach benefits their library.

The marketing efforts of the companies involved in open source software evangelize its benefits, while the incumbent companies warn of its dangers. We must look beyond the marketing for the most objective information on this complex issue. On discussion lists and blogs, opinions flow in all directions on the role of open source software in libraries. This report is not meant to advocate for or against the open source approach, but rather to describe in some detail what is different about the open source approach and to provide information about some of the products and companies involved. Readers can then draw their own conclusions.

This report focuses on open source issues specifically relating to integrated library systems. We will provide some general information about open source software and its use in other domains in order to provide some background for the discussion.

ILS in a Nutshell

The Integrated Library System, or ILS, provides computer automation for all aspects of the operation of a library. These products are generally organized into modules that address specific functional areas. Standard modules include cataloging for creating bibliographic records that represent works in the library's collection, circulation that automates tasks related to loaning items to patrons, serials control for managing periodicals and serials, acquisitions to handle the procurement process for new items added to the collection, and the online public access catalog to allow library users to search or browse through the library's collection. Each of these modules offers a very detailed suite of features to accommodate the complex and nuanced routines involved in the library work.

Integrated library systems rely on databases shared among the functional modules. The bibliographic database stores descriptive information about each work in the collection, ideally consistent with the MARC21 standard. A database

of authority records ensures consistent forms of names and subject terms and provides references to related terms. Another database tracks information about each item, linking each record for a copy to the appropriate bibliographic record. A patron database manages data for each registered library user. The acquisitions module relies on multiple databases in support of procurement-related functions such as vendors, orders, invoices, and funds. The circulation module involves transactions linking patron and item records when an item is checked out and unlinking them when it is returned. A set of configuration tables, built according to the library's policies on the loan period for each type of material and category of borrower, controls the behavior of the circulation module. The online catalog draws from almost all of the databases and policy tables to provide an interface for library users that enables them to locate items in the library's collections and take advantage of other services offered by the library.

A number of standards have been developed to ensure interoperability among library automation components and to allow the interchange of data. These standards include Z39.50 for the search and retrieval of bibliographic information; SRW/U, a variant of Z39.50 expressed as a Web service; MARC21 for the structure of bibliographic records; AACR2 for consistent syntax for each field within bibliographic records; MARC holdings to represent the issues held for each serial or periodical title; SI for circulation related functions; and P2 or NCIP (NISO Circulation Interchange Protocol) for standard messaging and transactions.

Almost all libraries in the developed world make use of an ILS. In the United States, only very small public or academic libraries, often in rural communities, operate without them.

Some of the major proprietary ILS products currently available include Symphony from SirsiDynix, Millennium from Innovative Interfaces, Aleph from Ex Libris Group, Voyager from Ex Libris Group, Polaris from Polaris Library Systems, Library.Solution from The Library Corporation, Carl.X from The Library Corporation, Spydus from Civica, and many others. The proprietary products have been available for many years, have reached a high level of maturity, and remain the dominant approach used for library automation.

What Is Open Source?

Open source software is free software. It's not necessarily cost-free, but is free to use, free to modify, and free to share. It's a model of dealing with software that presents an alternative to the commercial licensing that imposes many layers of restrictions.

Please note that the term *free software* tends to be used synonymously with *open source software*. In this report we will use *open source software* since it tends to be used a bit more widely in the library community. All of the ILS products in this space promote themselves as open source rather than free software.

The open source software movement is one of the major alternatives for professionals who work with computer software. On one level, it involves a specific set of software license terms that specify who gets access to the source code that underlies programs, who can change them, what can or must be done with changed versions of the software, and other issues related to modifying the program. But open source also stands for a broader philosophical approach to software that aims to give its users more freedom and allow them to break free from constraints associated with the traditional proprietary model.

Open source software has been a growing part of the overall landscape for the last decade or so. In the broader information technology arena, open source software alternatives have become well-established in key areas of infrastructure from operating systems to web servers. Open source operating systems include many varieties of Linux that compete with proprietary systems like Microsoft Windows. The classic polemic casts Microsoft as a monopolistic domineering company against the open source alternatives that free the world from its stranglehold. In the real world, many individuals continue to choose the proprietary option, and others prefer open source alternatives. These two approaches coexist in the market.

In almost all aspects of computer infrastructure, open source and proprietary software are both available. Table 1 lists some well-known examples of open source and proprietary products available in several categories of computing infrastructure and applications.

Whether a library uses an open source ILS or not it may make use of open source software in other parts of its computing environment.

Open Source versus Traditional Licensing

Open source software is governed by a family of software licenses that embody a philosophy of software freedom,

Category	Closed Source Examples	Open Source Examples
Server operating system	Windows Server 200x	Linux variants (Red Hat, Ubunto, Debian, SUSE Linux)
Database engines	Oracle, DB2, Windows SQL Server	MySQL, PostgreSQL
Programming languages	Microsoft C++	Perl, PHP, Ruby, Python
Desktop operating system	Windows Vista / XP; Mac OS X	Linux + desktop environments (e.g., GNOME or KDE)
Web server	Microsoft Internet Information Server	Apache
Web browser	Microsoft Internet Explorer	Firefox, Mozilla, Opera, Chrome
Office productivity	Microsoft Office	Open Office

Table 1
Common open source infrastructure components

appropriate attribution, and ensuring that no one has an unfair advantage. No single set of rules applies to all—many different flavors of open source licenses have emerged to accommodate many different business models, legal concerns and philosophical standards.

The label *open source* refers to a key principle—that the source code for the software must be made available to its users. Programmers write software using languages like C, C++, Java, or Perl. The code written by the programmer will usually be compiled into a binary form that can be run on a computer. It is this binary form that is most commonly distributed for use, even with open source applications. Distributing the binaries saves the user from the work of recompiling the software and makes for a much easier process of installation.

An Explanatory Note

In some programming environments, the discussion of source code versus binaries will not apply. Programs written in interpreted languages such as Perl, PHP, and JavaScript exist only as source code and are dynamically converted into binary machine instructions upon execution. Some environments compile Perl scripts into a binary form for faster execution, bringing back the distinction. Programs written in C or C++ must be compiled into binary form before they can be installed and executed.

The binary form of the software, while it runs well on a computer, cannot be read or understood by a human. In order to read, understand, and modify code, a programmer needs access to the original source code from which the binaries were created.

In the realm of proprietary software, only the binary form of the program is distributed to users. The original source code is held as confidential proprietary information, made available only to programmers of the organization that created the software application. In a business model that is dependent on revenue from licensing fees and prohibits use by anyone not paying for the product, it's important to control access to the source code, lest unauthorized versions become freely available. In this realm, the way the software works as expressed in the source code is usually a closely guarded trade secret.

In contrast, open source software requires that the source code underlying a computer program be made available to its users. With the source code available, other programmers can study how the software works, fix errors, and make modifications. If the software isn't exactly suited for a given use, it can be adjusted or improved.

The open source model of software development values the inspection of the source code by other programmers. It proposes that when more programmers have the ability to view and study the code, the more likely it is that errors will be discovered and repaired.

The open source approach does not necessarily require that the source code be distributed automatically to each user. The vast majority of users are not programmers and will never have need for the source code. The open source approach requires, however, that there be a convenient way to access the source code on request, even if only binary versions are routinely distributed. In practice, it's common for the download page of an open source application to offer binaries for each of the common hardware platforms or operating systems, with an additional option to select a version that also includes the source code. Some distribution sites offer downloads only

for binary versions, with a notice that the source code can be obtained through an e-mail request. Open source software can also be distributed on media like CD or DVD.

Many open source software applications make use of other open source components. A common approach involves LAMP: Linux, Apache, MySQL, and Perl (or PHP). These components form the basis for many open source products. The requirement to make the source code available extends to the prerequisite components. Most open source developers avoid the use of any proprietary components. It is allowable, however, to mix open source and proprietary components under some of the open source licenses. Many commercial proprietary software products, including integrated library systems, make use of open source components.

Open source software, with its inherent requirement for access to source code, comes with the freedom to make changes or derivative versions. If a programmer wants to make changes to an application, it is permissible to do so under any of the open source licenses.

The freedom to modify open source software introduces some complexities related to version control. Ideally, if a programmer discovers an error or makes an improvement in an open source program, those changes can be attributed to the individual or organization that oversees the development of that application and incorporated into future releases and distributions. As more users of the software make more improvements, the application grows in functionality and stability over time. The community of programmers involved in using and improving the software often forms some kind of organization that deals with issues of quality assurance, testing, and version control and might establish a road map for future development.

Another requirement for open source software is the freedom to share. If I have access to an open source software application, I can share it with someone else. If I modify the software, I'm free to share that modified version, provided that I meet certain requirements like giving proper attribution to the original version and making available the source code associated with the modified version. Open source software precludes users from passing off someone else's work as their own. While any user

> The Free Software Foundation offers a definition widely accepted within the open source software community:
>
> Free software is a matter of the users freedom to run, copy, distribute, study, change and improve the software. More precisely, it refers to four kinds of freedom, for the users of the software:
>
> - The freedom to run the program, for any purpose (freedom 0).
> - The freedom to study how the program works, and adapt it to your needs (freedom 1). Access to the source code is a precondition for this.
> - The freedom to redistribute copies so you can help your neighbor (freedom 2).
> - The freedom to improve the program, and release your improvements to the public, so that the whole community benefits (freedom 3). Access to the source code is a precondition for this.
>
> A program is free software if users have all of these freedoms. Thus, you should be free to redistribute copies, either with or without modifications, either gratis or charging a fee for distribution, to anyone anywhere. Being free to do these things means (among other things) that you do not have to ask or pay for permission.[1]

Issue	Proprietary Software	Open Source
Source code	Not distributed to customers.	Available to anyone that uses the software.
Form of software distributed	Binaries / object code only.	Binaries and source code. In some cases, only the source is distributed. If binaries are distributed, source must be available on request.
Who can make changes?	Only the original developer or designates.	Anyone that uses the software.
Sharing—redistribution	Users may not share, resell, or further distribute software.	Users may share the software.
License scope	Licenses apply to a specific product.	Generalized: must not be specific to a given product.

Table 2
Major open source principles

of an open source program is allowed to share derivative versions, there is no requirement to do so.

Open source and proprietary software represent two ends of a spectrum of options (see table 2). Other license variants that fall between these extremes represent a compromise between the two. Some companies and organizations have specific concerns that prevent them from using a completely free approach.

Open source software is not synonymous with "public domain" software. Copyrights apply to open source software, whereas *public domain* generally implies no claim to copyright. Given the implied nature of copyrights, saying that software is in the public domain does not ensure the protections given by open source software licenses.

The Free Software Foundation uses the term *copylefted* for software whose license specifies that no additional restrictions can be added when new versions are created and distributed. Open source software can also be non-copylefted, meaning that it is possible to add some restrictions as it is redistributed. With non-copylefted software, the original free version may be compiled and distributed only as a binary. The original version remains free, but the modified version may not be.

In recent news, open source licenses have been upheld in court rulings. According to Lawrence Lessig, the Court of Appeals for the Federal Circuit ruled that breaking the terms of restrictions specified in an open source license amounts to copyright infringement. This ruling reinforces these licenses as legally binding agreements.[2]

Open source programs adhere to a variety of different licenses. Two of the most popular are the GPL General Public License and the Apache Software License, but there are many widely used alternatives in the field. Each of these licenses has evolved over time. The GPL Public License, given its adoption by all of the Open Source ILS projects, is of particular interest to this report. The Apache license tends to be used more with proprietary commercial software that uses open source components internally. Many of the proprietary ILS products make use of the Apache license for their internal open source components.

Full terms of the GPL General Public License
www.fsf.org/licensing/licenses/gpl.html

Full terms of the Apache Software License
www.apache.org/licenses/LICENSE-2.0.html

The GNU General Public License, now in Version 3, is a full copyleft license that requires software to be free to use in any way, share, and modify. It requires that the source code be made available.

The GNU GPL does not prohibit commercial activity. For instance, you can charge a fee to allow someone to download copies of the software. You cannot require that others charge for downloading or pay you anything if they share it. Charging for downloading GNU GLP software is rare in practice, given that there are always ways for others to get the software without paying a fee. As we will see later, many companies do find business models surrounding open source software. The opportunities for income rely more on value-added services related to the software rather than for basic access to or use of the software itself.

The Apache Software License offers terms more amenable to commercial use. While it is a free license, and compatible with GPLv3, it allows for open source software to transition to a proprietary model. The Apache license does not require that changed versions of an open source software program be distributed under the same terms as the original version. It is possible for the changed version not to be distributed as open source, free software. The Apache Software License allows open source components to be incorporated into proprietary software, provided that certain requirements regarding attribution and licenses notices are met.

The Apache Software Foundation supports the development of some of the most commonly used infrastructure components, like the Apache web server, the Lucene search engine, the Solr search server, the Apache Tomcat Java Servlet environment, and many others.

Apache components are very widely adopted throughout the IT industry. According to Netcraft, the Apache web server ranks as the most widely used web server (49.49%). The proprietary Microsoft Internet Information Server comes in second with 34.88%.[3]

Open Source Initiative
www.opensource.org

Apache Software Foundation
www.apache.org

Free Software Foundation
www.fsf.org

GNU General Public License
www.gnu.org/copyleft/gpl.html

Notes

1. Lawrence Lessig, "Huge and Important News: Free Licenses Upheld," Lessig.org website, Aug. 13, 2008. www.lessig.org/blog/2008/08/huge_and_important_news_free_l.html (accessed Sept. 22, 2008).

2. Netcraft, "August 2008 Web Server Survey," Netcraft website, Aug. 29, 2008. http://news.netcraft.com/archives/web_server_survey.html (accessed Sept. 22, 2008).
3. From "The Free Software Definition," Free Software Foundation website. www.fsf.org/licensing/essays/free-sw.html [accessed Sept. 22, 2008].