Building an ICT Library

The most commonly used open source software for libraries
& their most important functions and uses

Workshop Handouts

Dr Edmund Balnaves
ejb@prosentient.com.au

Contents

Introduction to the Workshop

Background to Open Source

The "open source" movement emerged as a systematic method of distributing software in full source code in a manner that ensured it's ongoing availability in open source. The success of this movement has hinged on the ease of collaborative programming in an Internet environment, and service-based and reputation-based business models for software development.

Libraries themselves have an established history in systematic development of standards and the implementation of data interchange systems. The MARC (MAchine Readable Cataloguing) standard has been a critical pre-XML standard for open interchange of bibliographic data. The Z39.50 standard has enabled open inter-networking of library catalogues.

The first comprehensive suite of software released in open source for libraries was the Koha Library Management system. It has an active developer community internationally and has been translated for use in a multi-lingual environment (1).

The first experiments in open source library management systems have also helped evolve the sophisticated database schemas supporting current open source library management systems such as Greenstone, Evergreen and Koha 3(2). Even before this there have been numerous supporting tools developed by library institutions. These include open source modules for:

- text indexing and searching
- barcode generation
- Z39.50 services.

Software provided with source code is not new (for example the release of the IBM operating system source code in the 1960's as a result of anti-trust action in the US). However a long-term model for sustaining software development and improvement in an open source framework is new. The concept of open source has its origins in the evolution of the Internet and the Unix operating system - all on a C programming language base and often in the collegial environment of the University. Low cost access to personal computers and networking has inspired a new generation of computer users engaged in “social networking” and “social media”, and capable of building and contributing to their own systems, and mashups of other systems.

Open Source library management systems have been available for nearly a decade now. Reviews of open source systems have moved from cautious (3) to optimistic (4). During this time they have gradually evolved in both functionality and stability to the point where they are credible alternatives to commercial systems and in some cases provide a framework for earlier adoption of Web 2.0 features than might be otherwise available through commercial products(5). The evolution of open source Library Management has been energised by the Library 2.0 era. Other software that facilitates communication between librarians and their clients has emerged with the second generation of internet software - the Web 2.0 and Library 2.0. Web 2.0 capitalises on the internet framework to provide means for online collaboration, networking and

communication. This communication is facilitated by the availability of tools which make sharing and distribution of content and information easier, including community and tagging systems, and mashups using web services.

Open Source for the ICT Library

During the first session we identified the following core technologies for building an ICT-based library:

- Library Management System
- Digital Library System
- Document Delivery Systems
- Content Management System
- Digital Archives
- Document Management (Media Management System)

(see Figure 1)

Open source options are now available for the core systems needed to provision the ICT library. They vary considerably in functionality, capability and levels of support. This workshop reviews the most commonly used open source software for libraries & their most important functions and uses, focussing on the specific example of Koha and Linux.

One of the challenges in implementation of open source is the selection of a sustainable support model. This involves the scrutiny of the levels of professional support available internally and externally to support an open source implementation.

There are also a range of tools that are freeware, shareware or low cost which are not addressed in this article. They include MARC Edit (software for MARC data conversion) and EzProxy (widely used for single sign-on to database services).

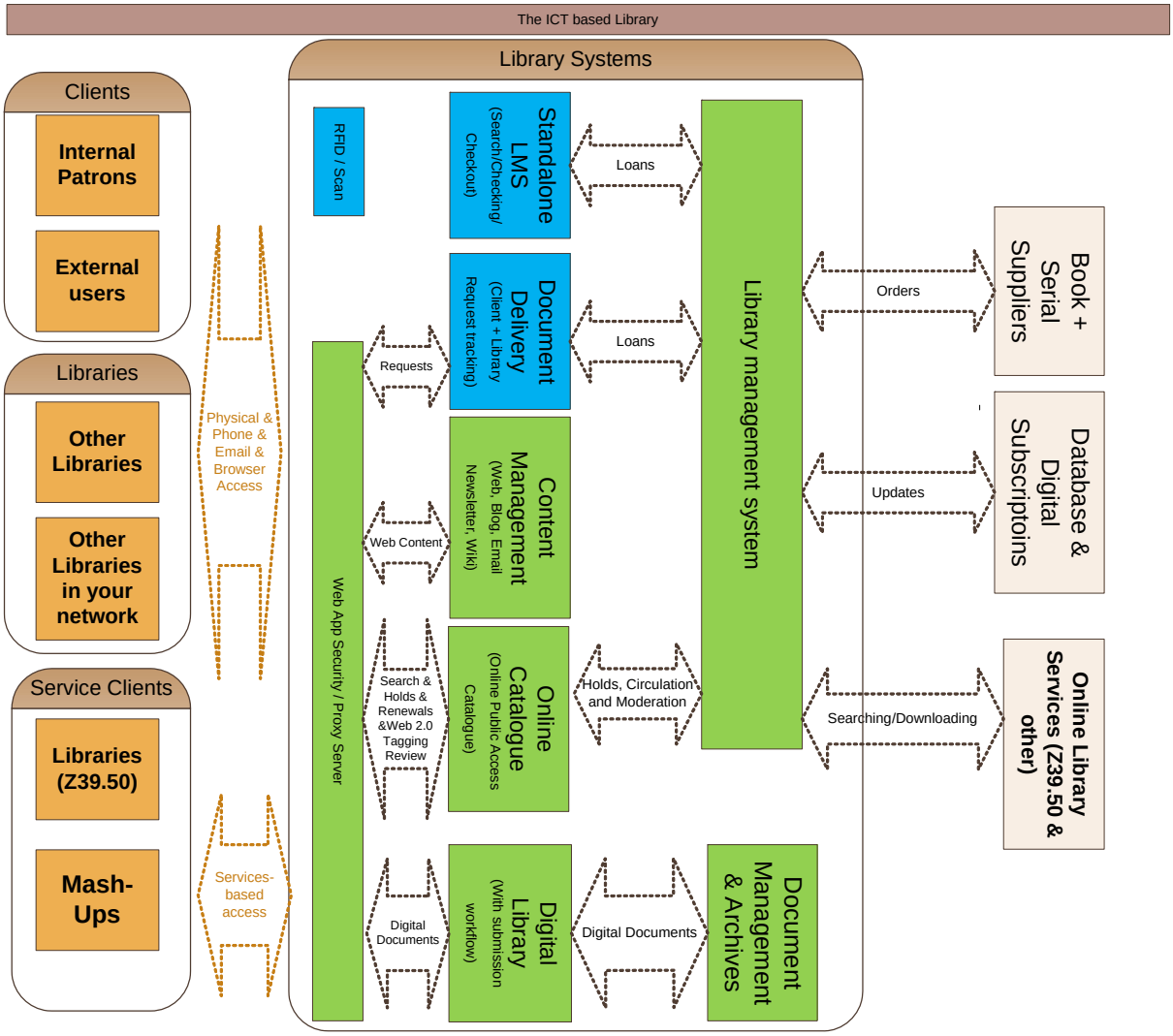


Figure 1: The ICT-based Library

Green objects are systems for which robust open source options are available

Workshop: Virtualisation, Linux & Koha

Introduction to Workshop

Koha is only one of several established open source library management systems. There are already over 20 open source library management systems projects of varying sizes visible through SourceForge (http://sourceforge.net). It is not uncommon for OSS projects to last a few years only and falter either for lack of ongoing patronage or lack of take-up. Systems such as Koha and PMB have a strong presence in Europe. Evergreen, one of the newest OSS entrants, has been built from ground up for scalability in large networks of libraries.

Being the oldest multi-branch open source LMS, the latest version of Koha (version 3) has a rich set of features The architecture, based on PERL, MYSQL and Apache is stable in a Linux environment, and the recent release of Koha3 brings a more stable framework for Windows also. The Koha development community is active, and this is reflected in solid wiki resources for developers and strong development and code management guidelines. Several commercial services provide implementation and ongoing support(6). Koha also has a diversified installed base, which gives an assurance of ongoing viability. The concerns regarding scalability of Koha Version 2 (when addressing catalogues of hundreds of thousands of items) have been reduced with the integration of a scalable search engine using Zebra. The release of version 3.0 looks to address many cross-platform issues and better multi-lingual support. The new version extends the use of Web 2.0 features, and the adoption of Zebra to enhance searching using facets. The Perl language base may not be popular in the long term among developers and implementers of the system, but the system has a healthy developer community.

The workshop will focus on a walkthrough of Koha from an evaluative point of view, giving in the process a close look at:

Open Suse Linux V.11 (an open source alternative to Windows)

Koha

MYSQL

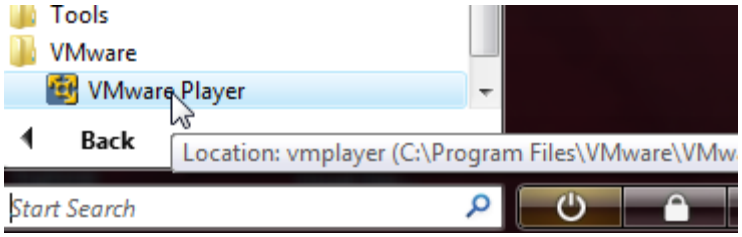
Perl scripts

These are all open source projects, bought together to provide a highly functional Integrated Library Management System.

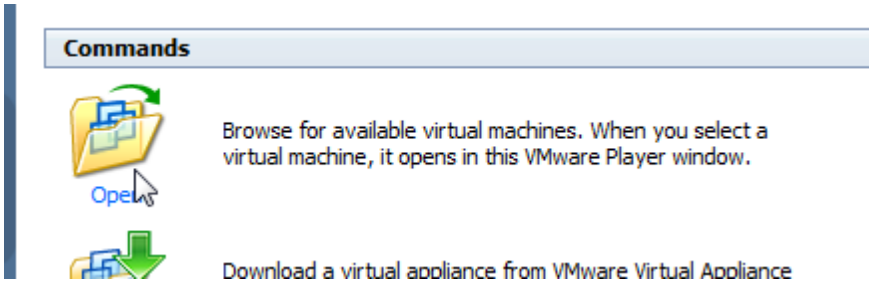
Workshop environment

Virtualised image: Open-Suse 11
Koha: 3.01
Username: vmplanet
Password: demo

Handout 1: Architecture & the Koha virtual image

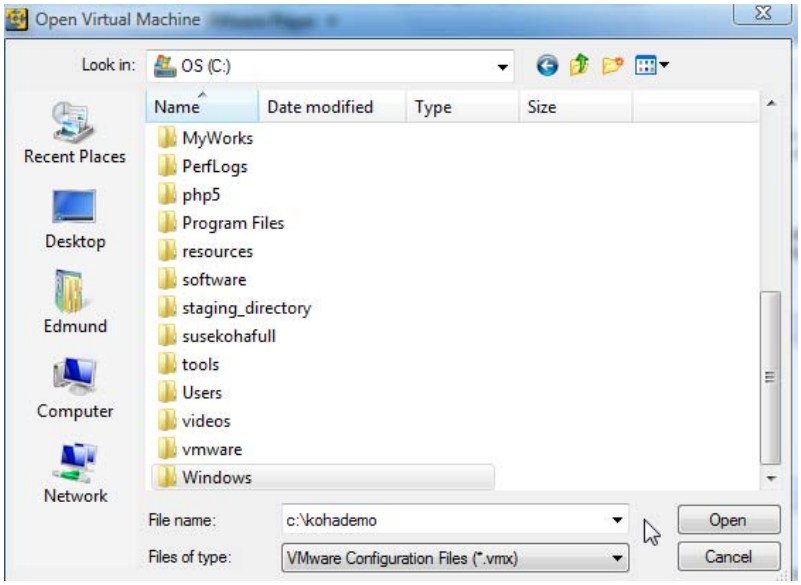


Click the OPEN button

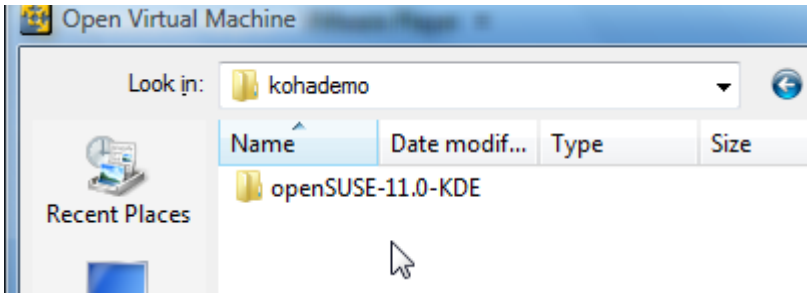


Against “file name” type in the location of the virtual image:

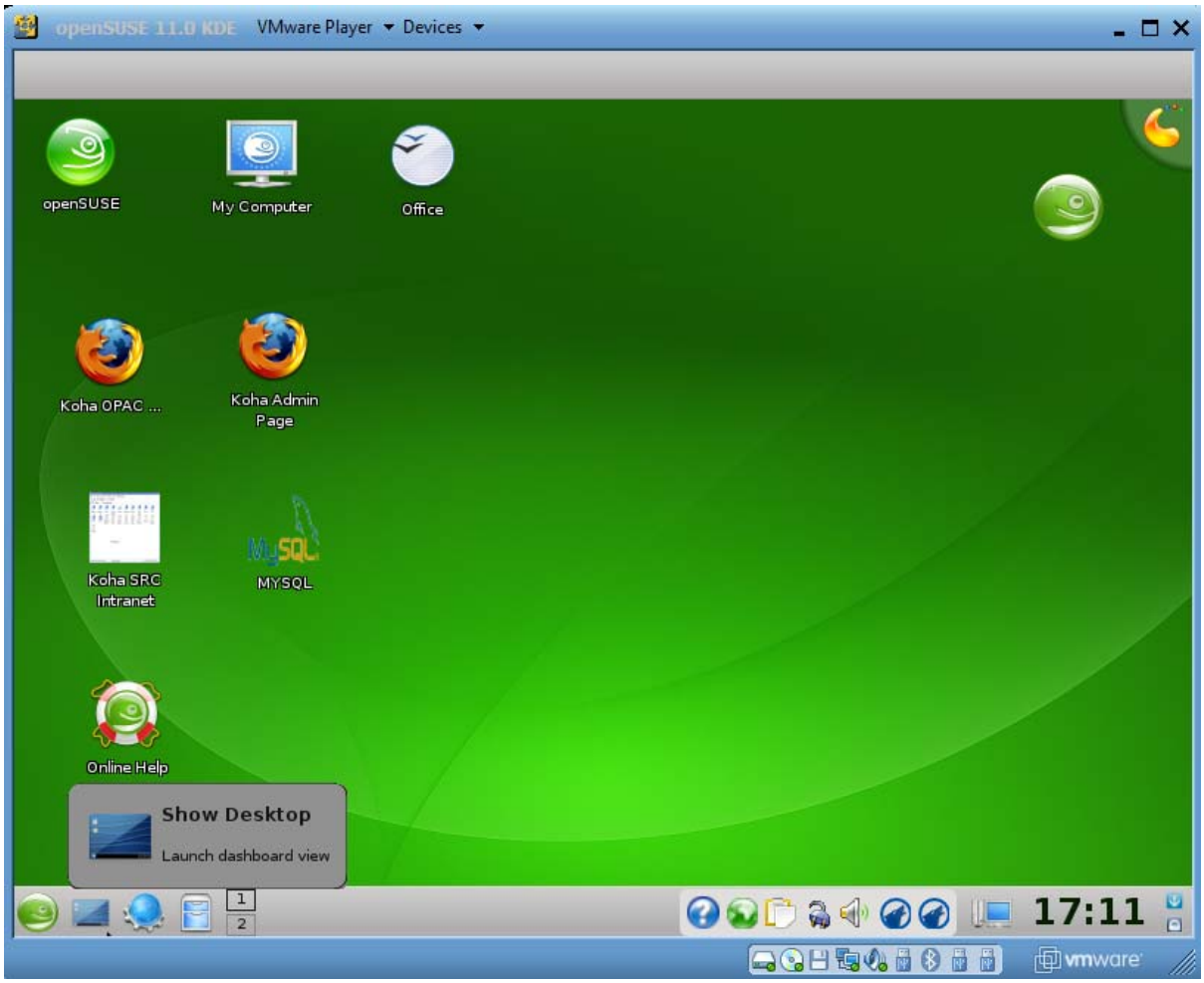
C:\kohademo



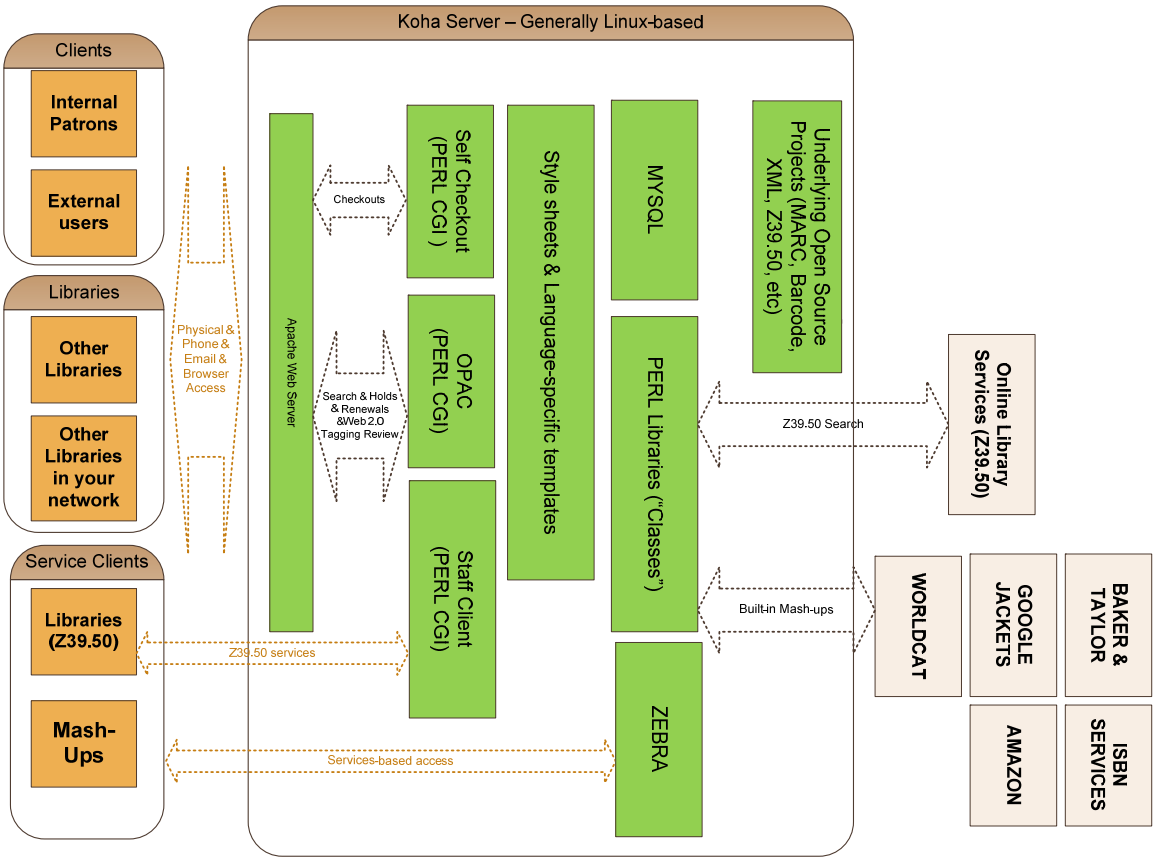
Double Click on the **openSUSE-11.0-KDE** folder and then double click the **openSUSE-11.0-KDE.wmv** file.



You should now have a Suse window open with your LINUX desktop



Koha Architecture



Handout 2: Evaluating the functionality

Example: (guided) Logging in & looking around the OPAC & Intranet

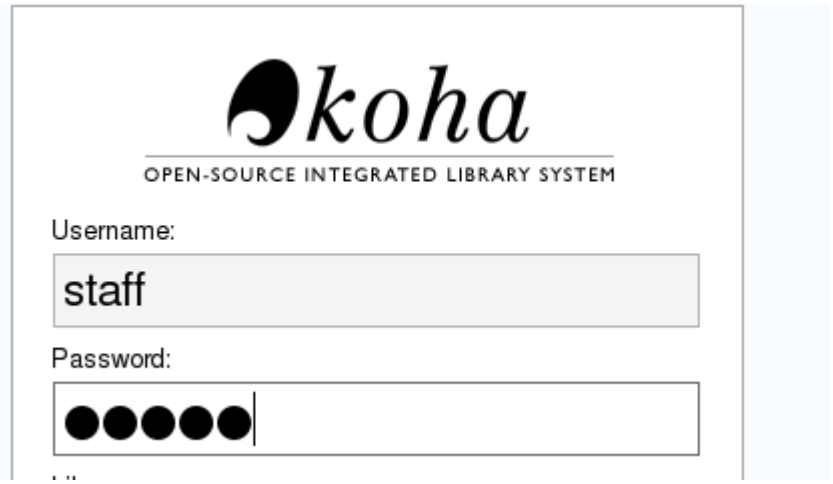
Sample login – patron: demo password demo

Sample login – staff : staff password staff

The staff client

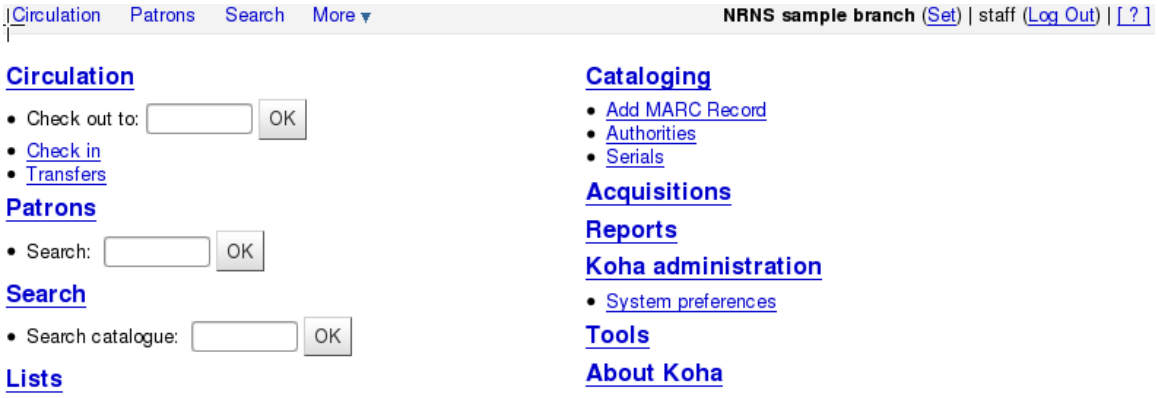
1) Double click on the staff client icon

2) Log into the staff client



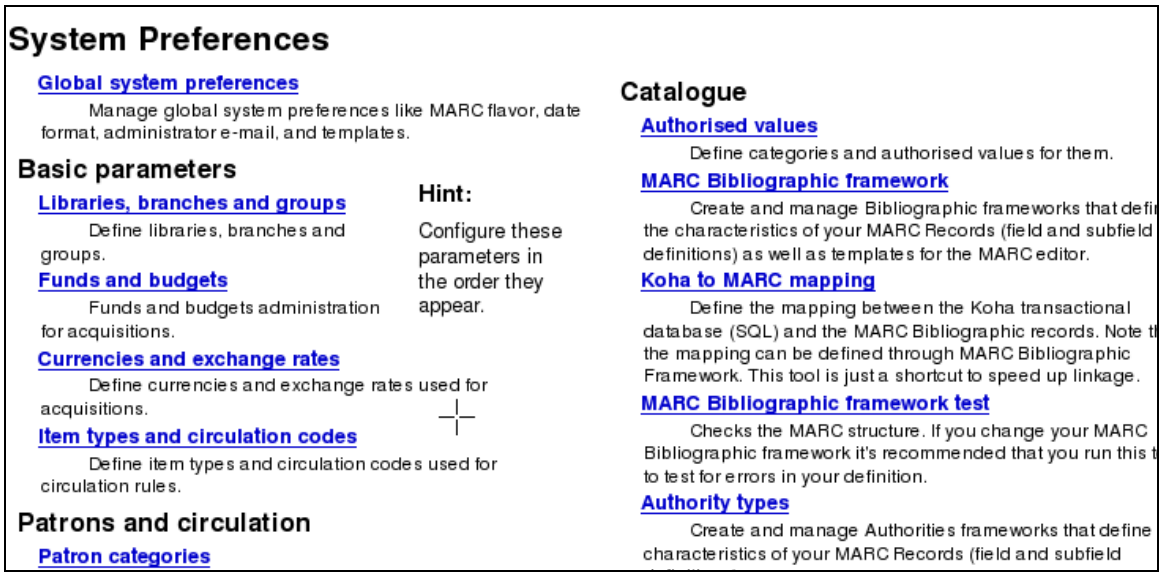
3) The staff client home page

Note: top right-hand the ? for page-sensitive help
Note: the More drop down tab allows movement between staff client modules
Note: access to different modules can be constraint on a login-by-login basis
Note: Cataloguing supports Z39.50 for copy cataloguing

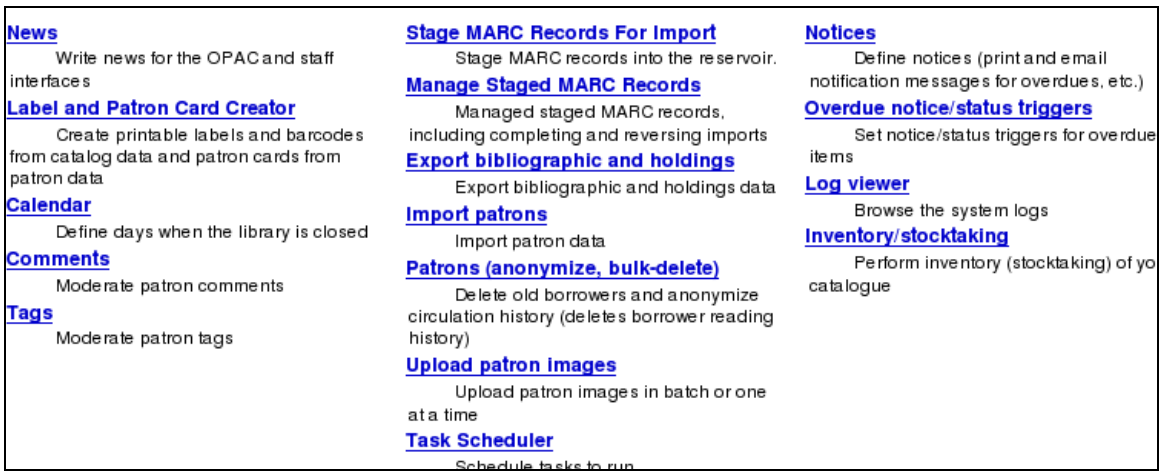


4) Self-Management

Koha Administration – for basic settings, including content management of the OPAC, circulation rules, fines, patron types, MARC configuration



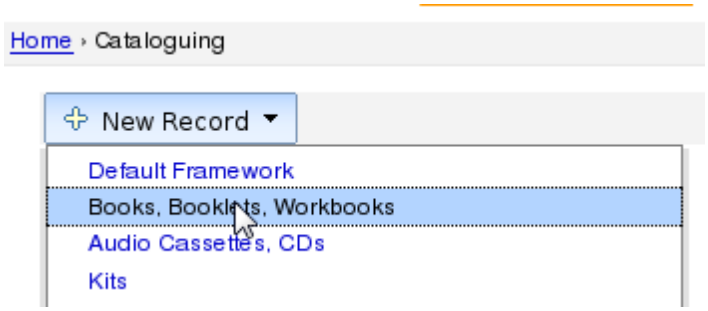
Tools – for setting up form letters (circulation reminders, claims, etc), News and content moderation (tags, reviews), MARC import & export.



5) Cataloguing

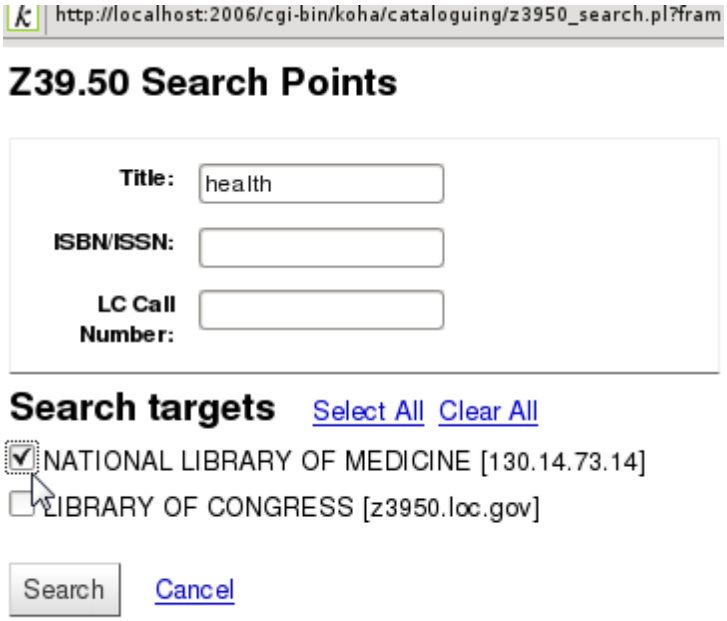
Note: MARC-oriented data entry – not loved by all!
Note: Z39.50 copy cataloguing search – loved by all!

More then Cataloguing. Create a new record by clicking the drop down arrow against "new record" and select Books



Click the Z39.50 search button

Search for a title



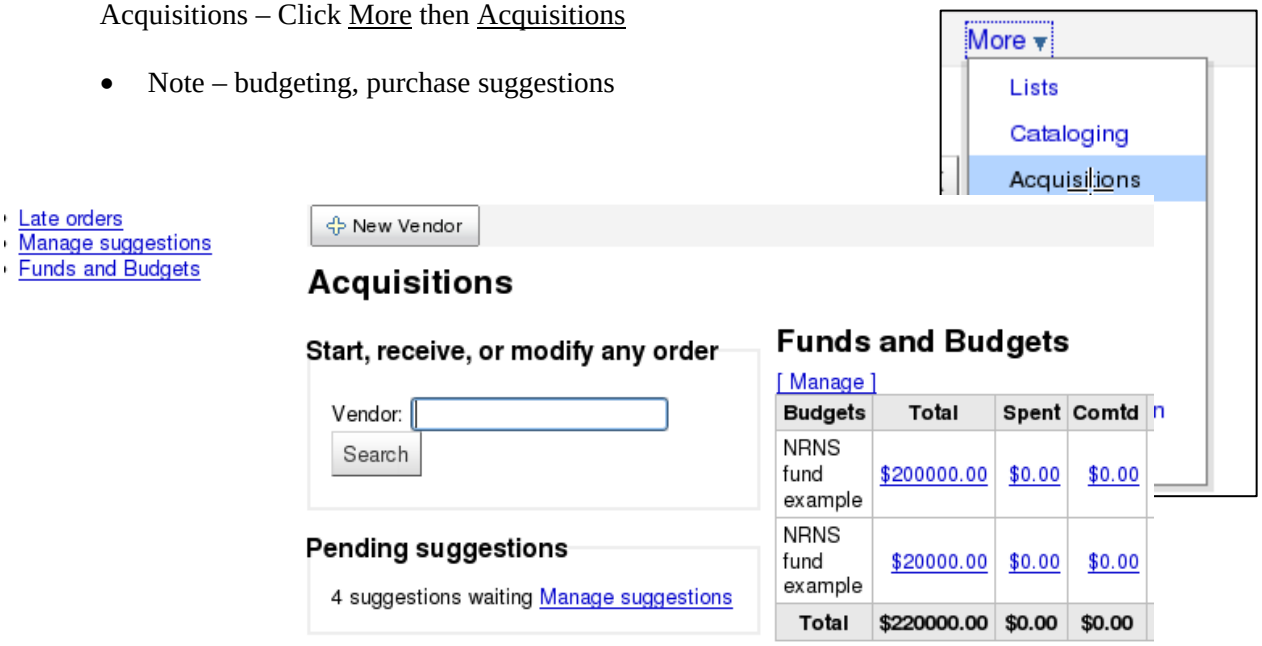
From the list of matching titles click the Import link (scroll right!)

The bibliographic record will be bought in. You can review the description by clicking on the links at the top (1 = MARC 100's = Author) (2 = MARC 200's = Title + description), etc)

6) Acquisitions

Acquisitions – Click More then Acquisitions

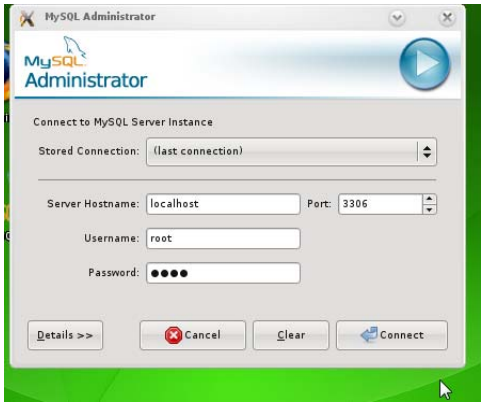
- Note – budgeting, purchase suggestions



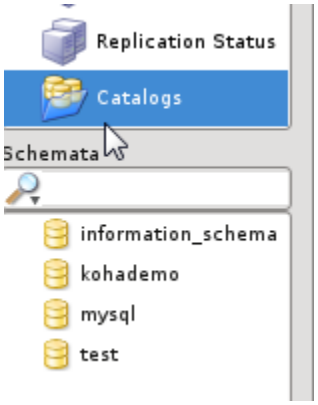
Handout 3: Evaluating the Database

(1) Double click on the MYSQL icon

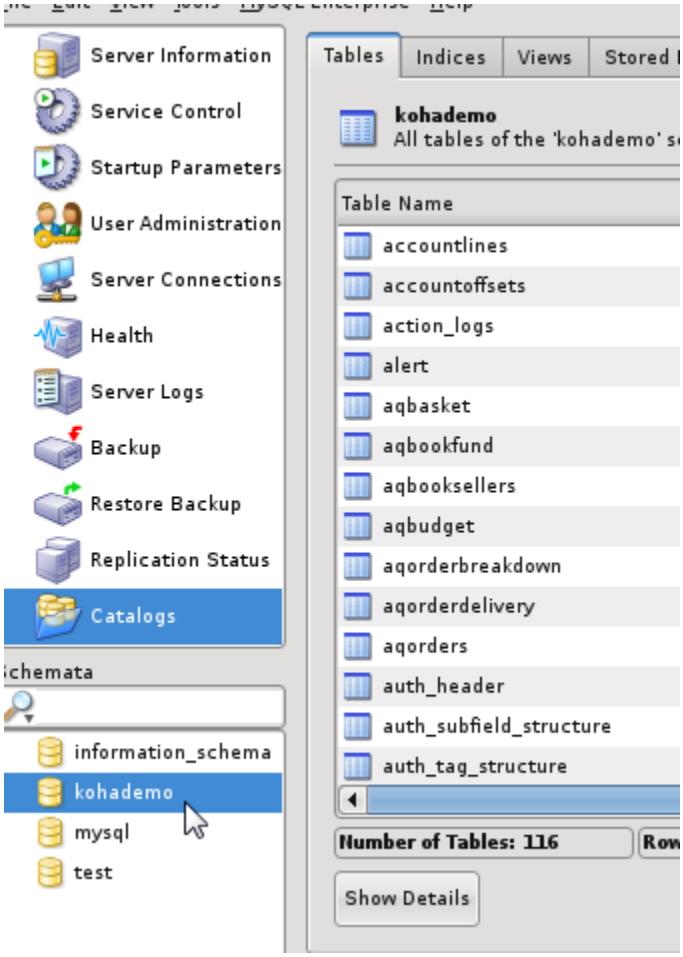
(2) On the connection screen, type in the password “demo” and click CONNECT



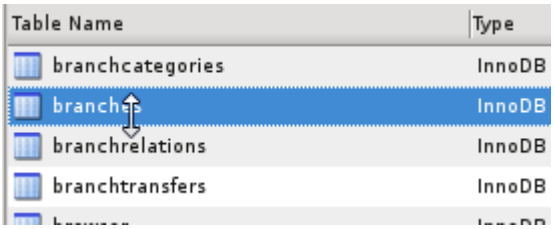
(3) Click on CATALOGS to see the list of databases



(4) Double click on the database **kohademo** to see a list of database tables



(5) Double click on any table to examine its structure



(6) Questions you might ask of a Library management system

- Is there functional support for complex bibliographic data
    - **Have a look at biblio, biblioitems, items tables**
- Does it support multi branch operation?
    - **Koha: have a look at the relationship between items and branches**
- Does the system have reasonable support for acquisitions
- How well normalised is the database design?

Handout 4: Evaluating the Code

Click on the Koha SRC button to look at the staff client source code folder (guided walk through)

Handout 5: Community

Example: Koha community review and source code release history. Evaluating the frequency of releases. Evaluating the degree of community engagement in code support. Evaluating the support options.

How to implement Koha in your library:

Discussion

Hosted-solutions

Internal hosting with support

Further information

Content Management:

Drupal: http://drupal.org/

Joomla http://www.joomla.org/

Canberra Hospital/ACT Health Library http://tchdev.anu.edu.au/

Digital Library:

Evergreen http://evergreen-ils.org/

Greenstone: http://www.greenstone.org/

Dspace: http://www.dspace.org/

Implementing an institutional repository:
http://www.dspace.org/implement/leadirs.pdf

Library Management:

Koha http://www.koha.org/

Koha manual https://sites.google.com/a/liblime.com/koha-manual/Home

Prosentient Systems http://www.prosentient.com.au/

PMB http://www.sigb.net/index.php?page=secteurs&id_rubrique=1&lang=en

Source Forge http://sourceforge.net/

WinISYS - Information:
http://portal.unesco.org/ci/en/ev.php-URL_ID=2071&URL_DO=DO_TOPIC&URL_SECTION=201.html

WinISYS - Distributers:
http://portal.unesco.org/ci/en/files/27278/12145786843cds_isis_distributors.pdf/cds_isis_distributors.pdf

Appendix One: Multi-dimensional Evaluation of Open Source (pre-publication draft)

Published in: Balnaves, E (2008). "Open source library management systems: a multidimensional evaluation" Australian Academic & Research Libraries, AARL March 2008 vol 39 no 1 pp1-13.

Dr E. Balnaves

Prosentient Systems

Abstract: The last five years have seen steady improvement in the features of open source library management systems, and they now present a credible option to small to medium libraries and library networks. A key merit of open source systems is the visibility of their design, their current constraints and their roadmap for enhancement. In additional to their comparative features, these systems will be analysed from three dimensions that only open source can offer: the developer & support community, the source code characteristics and the information schema. These additional dimensions allow for extended comparative product review not generally possible with proprietary software. An approach for detailed evaluation of open source library management systems is presented across five dimensions. This approach is applied to seven commonly deployed open source systems. The review highlights the variety of design approaches and the different library segments that these systems currently address.

Introduction

The last decade has witnessed a period of significant innovation in open source systems (OSS) for integrated library management. However, over the decade since the release of Koha, open source systems have shown sustained development of features, and significant cross-fertilisation of ideas based on shared experiences. Dorman[1] observed the difficulty that open source library management systems (LMS) may face in the context of a mature proprietary system marketplace. Nevertheless, libraries are showing increasing patronage of OSS, for integrated LMS and a range of other functions[2], and OSS will increasingly features as a mainstream option in the software selection process for libraries. The technologies of Web 2.0 (dubbed Library 2.0) have also added new energy to the development of open source integrated LMS.

Evaluation of proprietary software has necessarily focused on the evaluation across two dimensions: the features of the system and the operational platform. Software features can be compared, and the merits, scalability and capability of target platforms can be evaluated.

With OSS, it is not only the source code which is open to inspection. Open source systems generally attempt to enlist the widest possible collaboration in the software development process. As a result, their code change processes, change methodology and planning processes are also published in repositories such as SourceForce. Also available for inspection is the

underlying data design and the degree to which this endows flexibility to the application. The open development process therefore also opens perspectives on:

- the community dimension: the nature of the development community, the roadmap for enhancement and the level of active participation in development
- the source code dimension: the robustness of the coding, its level of code reuse, and the level of code documentation;
- the schema dimension: the database schema and design

Where to begin?

There are many published reviews of existing proprietary LMS systems, and a growing base of technical literature and blogs that review open source along these lines also, most notably Sturman[3] and Breeding[4]. The objective of this paper is to discern the degree to which open source systems provide additional dimensions for software review and selection of Library Management Systems.

Wheeler's blog on OSS evaluation points to issues of supportability, installed base and licensing model as important non-functional factors in OSS system selection[7]. The Business Readiness Rating (BRR) proposed by Carnegie Melon[5] takes a metrics-based approach that looks at open source community software. Cau, Concas, & Marchesi argue that two key metrics for open source projects are age and status– which taken together measure the success of an open source projects[6]. However, equally important to the durability of an open source project is the existence of an installed base – indeed many LMS open source projects have achieved a critical mass through the patronage of one or more library networks.

Both BRR[5] and Wheeler[7] suggest initial candidate filtering to limit the selection of systems to those with the best apparent fit based on functional and licensing issues – a process not dissimilar to short listing of proprietary systems.

This paper adopts an initial candidate selection with the following criteria:

- Source code is provided
- The application is released under a recognized OSS license
- The development processes are public.
- The system has a track record of adoption in libraries

There are already over 20 open source library management systems projects of varying sizes visible through Source Forge. It is not uncommon for OSS projects to last a few years only and falter either for lack of ongoing patronage or lack of take-up [6], and this is certainly evident in the case of LMS in open source.

Projects that are clearly moribund or have no current installed base are not included in this analysis. The seven systems selected for analysis in this paper are:

- Emilda[8] – a small open source project with an attractive interface and a good installed base in small libraries

- Evergreen[9] – a more recent open source system with a good installed base and strong "hype" in the blogs

- Gnuteca[10] – a Brazilian open source system with strong non-English installed base

- Koha[11] – the oldest open source system, with a strong installed base

- OpenBiblio[12] – a system popular in small libraries, with simple installation and an effective presentation.

- PhpMyLibrary[13] – developed in the Philippines and released not long after Koha. This system championed the use of MARC to facilitate adoption.

- PMB[14] – a French open source system with strong European installed base

While there are a range of open source licensing approaches, the popularly adopted LMS examined here all use the GNU General Public License. Drummond has a good review of alternative approaches to open source licensing[15].

Other projects, such as the UNESCO WEBLIS[16], are provided as free-ware but without source-code or visibility into the software development processes. A Koha branch called LearningAccess ILS[3, 17], much discussed in several open source blogs, has not yet been released to open source. While these systems have strong functional merits, they provide no visibility in their development processes and underlying code and so are not included in this analysis.

Multi-Dimensional Evaluation Of Open Source Software

For obvious competitive reasons, proprietary software suppliers generally give minimal visibility to:

a) their internal development capacity;

b) their detailed roadmap for development and strategic intentions;

c) the source code for their systems;

d) the internal development dialogue; and,

e) the detailed database schema design.

A functional evaluation of library management systems, for instance, may look at the features supporting:

> Cataloguing & OPAC – authority control, record import and creation (and record export), duplicate checking and thesaurus management,. Support for the OPAC, the public catalogue, search capability, documentation, web-based visibility, saved search capability.

Circulation – client and lending capabilities, tracking, returns, fines, notices, holds, recalls and reservations, and inter-library loans

Serials management – journal ordering, receipting, missing issue handling, claims and binding

Acquisitions – and the capabilities for ordering, making claims, receipting, budgeting and associated reporting and tracking.

Management & reporting, auditing

Documentation & end-user assistance

A technical architecture evaluation might examine:

The available platforms (e.g. Linux vs. Windows)

Scalability

Reliability

Support infrastructure & costs.

This paper proposes five dimensions for the analysis of open source library management systems:

The Functional dimension – and the degree to which OSS LMS provide well known LMS features such as OPAC, circulation, acquisitions and serials management. An element of the functional dimension is the end user documentation.

The Architectural dimension – the application architecture, the platform longevity, supportability and scalability.

The Community dimension – the level of activity in the development community, the patronage and installed base of the system, the open source licensing methodology, the existence of service organizations and the roadmap for development

The Code dimension – the code design, commenting and level of code reuse through Object oriented programming (OOP) or code inclusion.

The Schema dimension – the sophistication of the database schema and the capability and documentation of the schema design.

These dimensions differ from the BRR metrics both in their emphasis on the community dimension of OSS support and in the inclusion of the schema in the OSS analysis.

A poorly designed schema can be a significant impediment to integration with OSS systems with other reporting and information services and the sophistication of the product schema sheds a considerable light on the product capability. Similarly the design approach of the code (good code reuse, object orientation of design and schema) can facilitate extension of the system in new areas and by third parties. The level of code documentation can be a factor in the ease of participation and support for a system.

Multi-dimensional analysis

Table 1 below presents comparative summaries of the seven systems in each of the five dimensions. These systems serve different audiences, a factor that would be difficult to accommodate in metrics-based evaluation. A contingent factor in adoption of systems has been the degree to which the systems have a support base and are flexible to implementation across language boundaries. A further contingent factor is the complexity of the system against the size of the operation it is intended for – for instance Emilda clearly has a level of popularity based on its ease of implementation despite functional deficiencies.

All of the systems depend on other open source projects – for their database systems (MYSQL and PostgreSQL), for their web services (Apache) and for programming (PHP, Perl and C). All favour a Linux platform for implementation. While Windows implementations are possible, they are generally more complex to deploy in this environment and none offer a complete installer for Windows. Those based on a LAMP foundation (Linux, Apache, MYSQL and PHP) have the advantage of simplicity of code deployment in an "out of the box" Linux environment. Some, like Emilda, are more kindly to a Windows deployment, supporting IIS as well as Apache, but still require installation of PHP and MYSQL.

The most active projects, Koha, Evergreen and PMB, are innovating in areas of Web 2.0 support as well as extending the core set of features in the applications. All projects have some way to go in full multi-lingual implementation. Most implement a utf-8 unicode schema and some form of "skinning" to allow user interface language customization. All systems fall short of implementing double-byte unicode (utf-16) in their default schema and would not support Asian-language character sets (for example Cantonese, Mandarin, Japanese) for complex searching. Those libraries looking for capability on this front would face the prospect of customisation.

These contingent factors make a purely metric-based selection problematic. Where a system needs additional customisation to suite a particular library setting, understanding the additional dimensions will help selection of the most suitable system.

KOHA
www.koha.org

Koha is well reputed as the oldest open source LMS, with a rich set of features. Koha has not won a lot of friends in the Windows community, with a many reports of implementation difficulties[18]. The architecture, based on PERL, MYSQL and Apache is stable in a Linux environment. Their development community is active and diversified and several organisation services for implementation and ongoing support. Koha also has a diversified installed base, which remains a strength with respect to long term viability. It is suitable in a multi-branch operation. The beta release of version 3.0 looks to address many cross-platform issues and better utf-8 unicode support. The new beta also introduces Web 2.0 features. The Perl language base may not be popular in the long term among developers and implementers of the system. While concerns have been expressed regardng scalability, there are main techniques for load balancing and mult-server implementation that would support a Koha base.

EVERGREEN

www.open-ils.org

One of the newest open source applications, Evergreen has created a very positive "buzz" in the LMS open source blogs. Its schema and code design include strong OOP philosophy. It does not yet have the functional maturity of Koha and PMB, but shows promising potential in its underlying code and schema design. It has a large network installed base with the Georgia Library PINES network of over 200 libraries. Evergreen, like Koha, has a CGI (Common Gateway Interface) mode of server operation, but has clearly been able to support the PINES network.

GNUTECA
www.gnuteca.org.br

A Brazilian open source system with strong Portuguese installed base. Recently translated to French by University of Lyon. The code exhibits good OOP design traits but minimal code commenting and documentation. Functionality has yet to fully mature, but exhibits a good design foundation. This is an interesting project that could benefit from further internationalization work.

PMB
www.pizz.net

PMB has functional richness equivalent to Koha, with a better code and schema design framework. It has some internationalization support, but has largely been implemented in European libraries, and is strongest in its French support base – the default schema does not support utf-16. With a PHP/MYSQL base and good use of classes for code reuse, it is probably the most elegant choice of the current offerings for a multi-library network setting. Further implementation examples in English will improve its adoption outside a European context. As with Koha a range of Library 2.0 features are being explored.

EMILDA, OPENBIBLIO & PHPMYLIBRARY
www.PhpMyLibrary.org
www.emilda.org
obiblio.sourceforge.net

These three systems address a very different "market" from their larger open source equivalents. PhpMyLibrary is nearly as old as Koha, and has influenced the direction of open source development by demonstrating the effectiveness of integrating MARC support. These smaller systems have attractive interface and use easily deployed PHP/MYSQL platforms. Their schema complexity underline the functional gulf between these systems and their larger peers. However their small size has also lent these systems a certain nimbleness in simplicity of installation and attractiveness of user interface.

While these systems have weaknesses in support for acquisitions, serials management and more complex circulation control, they have a track record of simpler installation. The lack of multi-branch networking, acquisitions and other advanced features may not be problematic in a small-library setting (for example school libraries, or small research libraries) where the

principle focus is circulation control. Of these systems, Emilda shows the most active release profile. These systems particularly emphasise the contingent factors in suitability of open source systems – most particularly library size and type of library.

Conclusions

There is a healthy variety of open source LMS serving both individual libraries and large library networks. In 2002, Breeding, while optimistic about the prospects for open source LMS, indicated that "the handful that use open source systems cannot yet be noted as a trend" [4]. The six years since his review have seen considerable evolution in the functionality and installed base of open source library management systems. Interest in Web 2.0 functionality has opened an opportunity for these systems in an otherwise mature LMS market. These OSS projects show a variety of design approaches, have solid patronage and good user and developer community engagement. The emergence of support organisations such as LibLime that provide several different OSS systems is a further indicator of the growing maturity of the open source LMS marketplace.

Table 1: Multi-dimensional comparison of open source library management systems

DIMENSION	**KOHA**	**EMILDA**	**PHPMYLIBRARY**	**OPENBIBLIO**	**EVERGREEN**	**GNUTECA**	**PMB**
FUNCTIONAL	Koha has a wide adoption and strong support for OPAC, circulation, serials, acquisitions and reporting, including a range of and third party "add-ons. Has MARC support.	Single-branch software. Has OPAC and circulation. Limited support for acquisitions. No serials management and principally book-oriented schema. Clean user interface. Has MARC support (via Zebra).	Principally oriented to a single-branch library requiring OPAC and circulation management. Minimal acquisitions & serials management. Has MARC support.	Comments: the schema is principally oriented to OPAC and circulation support for a book-based service in a single-library setting. Minimal acquisitions & serials management. Has MARC support.	Has a strong OPAC and circulation system very suitable for multi-branch operation. Limited Acquisitions and Serials management. OOP design has facilitated integration of Web 2.0 features and MARC support.	Supports a multi-branch library network. Has OPAC, and some elements of circulation, serials, acquisitions, and a range of reporting functions. Documentation in Portuguese. Has MARC support.	Supports a multi-branch library network. Has OPAC, circulation, acquisitions, serials management and a range of reporting functions. Good internationalization support. Has MARC support.
ARCHITECTURE	*Perl/CGI* Installation & stability issues in Windows. Linux implementation popular & stable. They have adopted *Zebra* for search engine in v3.0. CGI-based web design could limit scalability.	*PHP//XML/Perl* *With some:* *Zebra, Yaz, Perl* *Requires Apache* Straightforward implementation – best on Linux. Windows installation possible, but requires multi-step installation.	*PHP* *Requires Apache* Straightforward implementation & best on Linux(Apache & MYSQL preinstalled). Windows installation possible, but requires multi-step installation.	*PHP* Straightforward implementation & operates comfortably in Linux (Apache & MYSQL preinstalled). Windows installation straightforward but multi-step.	*C++, Perl, Python* *Requires Apache* Straightforward implementation, best on Linux. CGI-based server design but has scaled to a network of several hundred libraries.. Has a Windows installer for the staff client.	*Perl/PHP* *Requires Apache* Best on Linux, but CGI-based web design could limit scalability. Windows installation possible, but requires multi-step installation.	*PHP. Requires Apache* Straightforward, multi-stage implementation. Principle installations currently in French/European.
COMMUNITY	Appears to have a very active developer community & several support companies – notably *LibLime*. First release was 2000 with 10 version/sub-version releases since. Installation appears to be a point of focus in the ver.3 beta. *Has an active roadmap showing progress against planned features.* Google Scholar references: 76	An active project. Developed & supported by *RealNode* First release 2004-01-22. Latest release 2005-06-29 version 1.2.3 (used for evaluation) with 5 version/subversion releases. It as a wish-list based through discussion lists Google Scholar references: 11	Developed in the Philippines, it is a marginally active project with limited participants, but a significant installed base. First release 2001. The current release is 2.2.1-3 (2007). Minimal activity between the 2003 & 2007 release. Roadmap skeleton in place – with a release 2.0 focus on Serials and Acquisitions Google Scholar references: 17	An active project with marginal activity and limited participants The first release was on 2002 into beta on SourceForge. The latest release was on 0.6.0 in 2007. Has a roadmap (no tracking) through wiki. Google Scholar references: 18	Appears to have an active developer community. Installation and support is available through *LibLime*. The first release ws on 1.2.0.1 in 2007. It has had 3 major releases since. They have a roadmap (wiki-based, no tracking) - http://open-ils.org/dokuwiki/doku.php?id=feature_list Google Scholar references: 69	An active project Has a discussion-list based feature request. First released: gnuteca-0.1 in 2001. The current release is Gnuteca 1.7 RC2 in 2007 with 9 major version/sub-versions since inception Google Scholar references: 7	An active project. Several support organizations exist, principally the French *PMB Services*. Has a roadmap (wiki-based, no tracking). Installation and support service organizations exist. Google Scholar references: 33

DIMENSION	**KOHA**	**EMILDA**	**PHPMYLIBRARY**	**OPENBIBLIO**	**EVERGREEN**	**GNUTECA**	**PMB**
CODE	*GNU General Public License* Code structure: procedural design but with extensive use of core set of classes. Some abstracted database classes. The v.3 beta addresses Unicode multi-lingual issues in the default schema and user interface "skinning", and supports additional databases. They have an active documentation project. Average changes per annum: 887. Code comments averaging 49 lines/function/page	*GNU General Public License* Procedural design with some inclusion-based code reuse and an "API". No abstracted database classes (embedded SQL). Uses: SourceForce - average changes released to SourceForge: 110 per annum. Some Page/Function and code documentation averaging 32 lines per function/page.	*GNU General Public License (GPL)* Code structure: some reuse classes but minor code reuse. Good use of templating for user interface. No abstracted data classes but use of separate db update functions. Some Page/Function comments and some code comments. average comments per function/page: 34 SourceForge *and* code.google.com	*GNU General Public License (GPL)* Code structure: OOP Use of classes to achieve code reuse, and good use of abstract data classes. Some code comments – mainly function descriptions Average comments per class/page: 36	*GNU General Public License* OOP philosophy evident in code and schema. Average comments per page/function – 10 Comments are principally inline, many functions with minimal code documentation. Mixture of embedded SQL & supporting database classes..	*CC-GNU General Public License* Good OOP design and use of classes for code reuse. Good use of abastract classes for database access. Minimal code documentation (average 1 line per class excluding standard headers)	*GNU General Public License (GPL)* Some OOP design with good use of PHP classes for effective code reuse. Some embedding of SQL in application pages & some use of abstract db classes. Some code documentation (average 57 lines per class).
SCHEMA	*Databases supported: MYSQL.* 74 tables in the schema. The default schema does not use double-byte Unicode. Schema design & code MSSQL specific. Have adopted Zebra for search which may limit double-byte search support. . Table schema is partially documented with a number of sample data sets.	*Databases supported: MYSQL.* 24 tables in the schema. The default schema does not use double-byte Unicode. Schema design & code MSSQL specific. No abstract database classes. Schema oriented to book-based OPAC/circulation.	*Databases supported: MYSQL.* 23 tables in the schema. The default schema does not use double-byte Unicode. Schema design & code are MSSQL specific. The schema is principally oriented to OPAC and circulation support for a book-based service in a single-library setting.	*Databases supported: MYSQL.* 24 tables in the schema. The default schema does not use double-byte Unicode. Schema design & code MSSQL specific. The schema is principally oriented to OPAC and circulation support for a book-based service in a single-library setting.	*Databases supported: PostgreSQL.* 104 tables in the schema. The default schema does not use double-byte Unicode. Schema design & code PostgreSQL specific, and PostgreSQL has some Unicode issues. The table schema is well documented and has an object-oriented philosophy.	*Databases supported: PostgreSQL.* 31 tables in the schema. The default schema does not use double-byte Unicode. Schema design & code PostgreSQL-specific. A schema diagram available.	*Databases supported: MYSQL* 127 tables in the schema. The default schema does not use utf-16. Schema design & code MSSQL specific. Documented principally in French, but with supporting documentation for alternative language installation. The table schema is partially documented with a number of sample data sets

Koha, Evergreen and PMB demonstrate very active developer communities with solid institutional backing. Their code base and schema show solid technical evolution and these systems have been nimble in adopting new Library 2.0 innovations. PMB and Gnuteca have a strong non-English language installation and support base.

While Koha shows the most sophistication in managing the open source development process, Evergreen and Gnuteca exhibit strong object oriented design philosophy in their code and schema design. Emilda, OpenBiblio and PhpMyLibrary continue to serve a role for the small library community for their relative easy of implementation and simple, intuitive interfaces.

A multi-dimensional view of open source systems will enable better decision making at the point of system selection. The success of software has never been solely down to the technical merits of a system. The adopted base of software is, of course, one of the most significant elements of software, both in sponsorship for further enhancement and development, and in assuring a degree of technical awareness of the product that encourages adoption. In this respect particularly the analysis of all five dimensions suggested in this report opens a more complete insight into the current state of each system.

As an afterword, many organizations are bound to their existing systems due to the complexity of transferring their current information base. However, MARC has provided a core common semantic base to facilitate the adoption of different systems through MARC export/import processes. It is a credit to this long-standing ontology that it still serves such a functional purpose.

NOTES

1. Dorman, D. (2004) The Case for Open Source Software in the Library Market. Ubiquity Volume,

2. Herbert, E. How Open Source Software Can Improve Our Library. Online, cited 2008 18/01/2008; Available from: http://www.degreetutor.com/library/managing-expenses/open-source-library.

3. Sturman, R., Il software open source per la gestione integrata delle biblioteche: una nuova risorsa?, . BollettinoAIB, 2004. 44(3): p. 257-270.

4. Breeding, M., An update on Open Source ILS. Information Today, 20002. 19(9): p. 42-43.

5. Wasserman, A., M. Pal, and C. Chan. The Business Readiness Rating:a Framework for Evaluating Open Source. in EFOSS - Evaluation Framework for Open Source Software. 2006. Grand Hotel, Como, Italy.

6. Cau, A., G. Concas, and M. Marchesi. Extending OpenBRR with Automated Metrics to Measure Object Oriented Open Source Project Success. in EFOSS - Evaluation Framework for Open Source Software. 2006. Grand Hotel, Como, Italy.

7. Wheeler, D. How to Evaluate Open Source Software / Free Software (OSS/FS) Programs. Online 2008 Revised as of January 15, 2008 cited 2008 18/01/2008; Available from: http://www.dwheeler.com/oss_fs_eval.html.

8. Realnode Ltd. Emilda. 2008 cited 2008 18/01/2008; Available from: http://www.emilda.org/.

9. Georgia Public Library Service. Evergreen. Online cited 2008 18/01/2008; Available from: http://www.open-ils.org/.

10. Cooperativa de Soluções Livres. Gnuteca. Online 2008 cited 2008 18/01/2008; Available from: http://www.gnuteca.org.br/.

11. Koha Development Team and Katipo Communications Ltd. Koha. Online 2008 cited 2008 18/01/2008; Available from: http://www.koha.org/.

12. Stevens, D. and M. Stetson. OpenBiblio. Online 2008 cited 2008 18/01/2008; Available from: http://obiblio.sourceforge.net/.

13. Babao, P. PhpMyLibrary. Online 2008 cited 2008 18/01/2008; Available from: http://www.phpmylibrary.org/.

14. PMB Services. PMB. Online 2008 cited 2008 18/01/2008; Available from: http://www.pizz.net/.

15. Drummond, J. Open Source Software and Documents: A Literature and Online Resource Review Online 2000 cited 2008 21/01/2008; Available from: http://www.omar.org/opensource/litreview/.

16. UNESCO. Logiciel de base de données CDS/ISIS: WEBLIS. Online 2007 cited 2008 18/01/2008.

17. Learning Access Institute. Online 2008 cited 2008 12/01/2008; Available from: http://www.learningaccess.org/tools/ils.php.

18. Chalon, P.X., et al. Open your mind! Selecting and implementing an integrated library system:the open-source opportunity. in 10th European Conference of Medical and Health Libraries. 2005. Cluj-Napoca , Romania, 11th-16th September 2005.