Creative Commons Attribution-Noncommercial-Share Alike 3.0 United States

A Survey and Evaluation of Open-Source Electronic Publishing Systems

Mark Cyzyk and Sayeed Choudhury
Library Digital Programs
The Sheridan Libraries
The Johns Hopkins University
Baltimore, Maryland. USA

April 28, 2008

Preliminary Note

The research for this study, commissioned by the Open Society Institute (OSI), was performed from roughly November 2006 through July 2007. Since that time, the following electronic publishing systems have had the following releases:

DPubS version 2.1
GNU EPrints version 3.0.5RC1
Open Journal System version 2.2

Introduction

This study provides a high-level survey and evaluation of open-source electronic publishing systems ("ePublishing systems") most suitable for supporting publishing in a predominantly scholarly, scientific, or academic culture. Hence, this study is not concerned with ePublishing systems whose code bases are proprietary or are geared primarily toward purchase for use typically by for-profit corporations. This does not, of course, change the fact that the systems reviewed here could just as easily be of use in for-profit corporate settings, but this study emphasized a current evaluation of systems most useful in a non-profit or academic setting.

With the relatively recent call for "open access" to research and publications in the scholarly and scientific communities,[1] this survey and evaluation becomes arguably more

[1] The best single, comprehensive source of information on this timely topic is the Website of the Association of Research Libraries, Scholarly Publishing and Academic Resources Coalition (SPARC): http://www.arl.org/sparc/ . Professor Peter Suber's compilation of the SPARC Open Access Newsletter (http://www.earlham.edu/~peters/fos/) as well as the numerous other materials on his personal Website, especially his Open Access Overview (http://www.earlham.edu/~peters/fos/overview.htm), are likewise

important. University presses, scholarly/scientific/professional societies, libraries, and individual researchers and faculty themselves have become increasingly interested in providing open and easy access to scholarly works and scientific research, and they are increasingly finding that providing such access in electronic format via the Web can be the simplest, most economical, and most powerful way to accomplish this[2] -- hence the need for an up-to-date survey and evaluation of the various means toward accomplishing this goal in the current technological environment.

While this survey does not delve as deeply, it is inspired by a previous evaluation effort conducted by the Library Digital Programs at Johns Hopkins University. With funding from the Andrew W. Mellon Foundation, Johns Hopkins University conducted an evaluation of the repository software systems DSpace, Fedora, and Digital Commons.[3] Both of these evaluation efforts rest upon the premise that use cases or scenarios provide the best means for determining relevant functionalities for software systems. While the Mellon-funded repository analysis included a more in-depth analysis, the methodology from that analysis inspired the current evaluation of ePublishing systems. In the Mellon-funded repository analysis, which included multiple members of the Library Digital Programs team at Johns Hopkins, a community-wide effort resulted in a listing of dozens of scenarios. Each of these scenarios was mined for insights into specific repository functionalities that would support a range of content types and services. This analysis highlighted the particular importance of application programming interfaces (APIs) and ease of use and installation of the various systems.

It is worth noting that the aforementioned repository analysis reflected a great deal of initial investigation and evaluation that led to the more in-depth analysis. This ePublishing system review reflects this type of initial investigation and evaluation phase. Based on an initial review of several open-source ePublishing systems, the authors of this report developed a list of existing functionality and desiderata. This list was shared with colleagues at the Johns Hopkins University Press (JHUP) who provided feedback regarding a "canonical" list of features that would be required to support electronic publishing. While JHUP is known most prominently for Project Muse, which is primarily a humanities and social sciences set of publications, every effort was made to think more broadly and comprehensively. Having said this, undoubtedly, there is room for additional consideration.

---

enlightening and invaluable for learning about the Open Access movement. And the single best statement of purpose can be found at the Budapest Open Access Initiative site: http://www.soros.org/openaccess/ .

[2] For a fascinating account of the evolution of scholarly publishing in this regard, see: GueÏdon, J. (2001). *In Oldenburg's long shadow: Librarians, research scientists, publishers, and the control of scientific publishing.* Washington, D.C.: Association of Research Libraries.

[3] https://wiki.library.jhu.edu/display/RepoAnalysis/ProjectRepository

In addition to evaluating the features that any ePublishing system would typically support (peer review management; client access to final documents), this study offers special focus on the APIs provided by each system. Such APIs allow the system to interact with various other systems, e.g., institutional repositories, Websites, portals, learning management systems, content management systems, and digital asset management systems. Insofar as the ePublishing system typically exists and functions within the context of a larger IT enterprise, knowing how it can interact with other systems within that enterprise is important. At its simplest, batch import and export of data into and out of the ePublishing system is one example of an API. But as our work here at Hopkins with regard to institutional repositories has shown, APIs are not limited to this. The study seeks out, explores, and enumerates these APIs, all in the context of ePublishing systems.

Methodology

A preliminary review of the literature was performed as well as a significantly deeper scan of the Web in search of any ePublishing system that meets the criteria of being: (1) open-source, and (2) seemingly useful in an academic setting. The initial goal was to compile as comprehensive as possible a list of such systems. The results of this effort are listed in Figure One.

After delving deeper, we chose four systems for further, detailed investigation. These four systems were:

- DPubS (Digital Publishing System) (Cornell and Penn State)
- GNU EPrints (University of Southampton)
- Hyperjournal (Net7 and University of Pisa)
- Open Journal System (University of British Columbia and Simon Fraser University

Three other systems, while not fully evaluated here (for reasons discussed below), merit special mention:

- Connexions/Rhaptos (Rice University)
- DiVA (Digitala Vetenskapliga Arkivet) (Uppsala University)
- Topaz (The Topaz Project)

The evaluation of these first four systems—Dpubs, EPrints, Hyperjounral and OJS—consisted of local installation, reading supporting documentation, and consideration of four broad areas:

- Institutional affiliation and other indicators of the viability of the open-source project
- Technical requirements, maintenance, scalability, and documented APIs

- Submission, peer review management, and administrative functions
- Access, formats, and electronic commerce functions

The specific criteria for evaluation within these four broad areas were as follows:[4]

- Institutional affiliation and other indicators of the viability of the open-source project
    - Name of system
    - Current version of system
    - Tested version of system
    - URL of project homepage
    - Institutional affiliation
    - Age of project
    - Notes on long-term viability of project
    - Degree of deployment
    - Type of open-source license
    - Licensing notes
    - Other documentation (Webliography)

- Technical requirements, maintenance, scalability, and documented APIs
    - Local install or ASP?
    - Operating system requirements
    - Hardware requirements
    - Application server requirements
    - Primary programming language
    - Auxiliary programming language
    - Application framework
    - Database server requirements
    - Other software requirements
    - Required skills
    - Internal backup and restore functions
    - Scalability: Application
    - Scalability: Data
    - API: Batch ingest
    - API: Batch ingest formats
    - API: Batch export
    - API: Batch export formats
    - API: Support for JSR 170
    - API: Support for OAI harvesting
    - API: Support for eduSource Communication Layer (ECL)
    - API: Support for other Web services
    - Security notes

[4] While already deep into the evaluation phase of this project, the author learned of the 2006 work of Goh, Chua, et. al., at Singapore's Nanyang Technological University in which they arrived at a similarly useful instrument for evaluating digital library software. See: Goh, Dion Hoe-Lian, et. al. (2006) "A checklist for evaluating open source digital library software." *Online Information Review*, v30 (4).

- Submission, peer review management, and administrative functions
    - Support for multiple, discrete publications
    - Multiple administrative roles
    - Administrative roles configurable
    - Submission into system initiated by authors
    - Editorial workflow configurable per publication
    - Automated email alerts to authors
    - Automated email alerts to editors
    - Automated email alerts to reviewers
    - Stylesheets, customizable look and feel per publication
    - Versioning
    - Archiving

- Access, formats, and electronic commerce functions
    - Accessibility of system
    - Accessibility of document output
    - Internationalization support
    - Output in multiple document formats
    - Document formats supported
    - Plug-in requirements
    - Usability notes
    - Citation linking
    - OpenURL resolver
    - RSS feed
    - Digital rights management
    - Full-text search and retrieval
    - Federated searching
    - Authentication mechanisms
    - Subscription services
    - Electronic commerce functions
    - Context-sensitive Help support

In all cases, each system was installed locally and the ease of installation was noted. In some cases, publicly available demonstration installations of the systems were used for evaluation of system functionality and usability. In all cases, supporting documentation was consulted in an effort to determine the range of services and functionalities each system provides and the manner in which it provides them. In a few cases, the developers of the system under consideration were consulted directly, most notably to assist in solving installation issues.

Summary Results and Analysis

A summary of each system is provided below:

Connexions/Rhaptos

Connexions/Rhaptos, a project of Rice University, is offered either through a freely-available hosted service running on Rice servers ("Connexions"), or the software underlying this hosted service ("Rhaptos") can be downloaded and locally installed. The Connexions project began in 1999. Its goal is to provide easy and free access to various educational "modules" and learning objects, including articles and monographs, but also multimedia files and presentations. Such modules can then be stitched together to form larger collections and courses. Connexions is somewhat a cross between an electronic publishing system and a system like Sakai. Connexions from its inception has supported the sharing of many units of educational content; Sakai has emphasized a collaboration and learning environment that incorporate general-purpose groupware applications, so a comparison between these two systems would be worthwhile.

Data collected for Connexions/Rhaptos are listed in Figure Two.

DiVA

DiVA (Digitala Vertenskapliga Arkivet) was founded in 2000 by the Electronic Publishing Centre at Uppsala University, Sweden. The purpose of DiVA is to support and provide an online repository of local materials, most notably electronic theses and dissertations (ETDs). The DiVA Consortium was founded in 2002, and as of 2006 15 Scandinavian universities had become members. The future direction and development of DiVA is governed by this consortium.

Data collected for DiVA are listed in Figure Three.

DPubS

DPubS began as Project Euclid in the Cornell University Libraries in 2000. Cornell and the Penn State Libraries joined together on this project in 2004 to launch DPubS. This evaluation focused on the second (Spring 2007) version, noting that there is a new major release that is now available. DPubS provides a customizable, skinnable, repository-style application for storing and providing access to multiple, discrete publications.

Strengths

DPubS, along with Open Journal Systems, was one of two systems under consideration that made provision for subscription services. It also appears to be very well architected and capable of significant customization at a deep level, e.g., it supports multiple custom metadata schemas, UI configurations, and file formats on a per-publication basis.

Weaknesses

The installation of DPubS presented notable challenges that resulted in two multi-day attempts on Apache 2 and one multi-day attempt on Apache 1.4, with multiple email interactions with the developers. Ultimately, the Apache 2 instance installed properly. Problems related to the slightly different requirements for installing the application on Apache 1.4 versus Apache 2, and to the application's reliance on many external open-source Perl libraries, each of which presented its own potential installation problems. The cumulative and interactive effect of these dependencies led to the multi-day installation attempts. Once installed, configuration of the application required running Perl scripts at a command line level. If an organization or group wished to publish multiple, distinct publications, the need for centralized administration via a command line would make it difficult to distribute administrative tasks out to journal editors, etc. without technical system staff support.

The DPubS documentation at the time of this evaluation was inconsistent or incomplete, and some of the wiki entries were either out-of-date or inaccurate. Clear, concise documentation is always invaluable, especially if one encounters installation challenges. The DPubS project team has indicated that they intend to hire a technical writer to develop updated documentation.

Data collected for DPubS are listed in Figure Four.

GNU EPrints

The GNU EPrints project was founded in 2000 in the Department of Electronics and Computer Science at the University of Southampton, U.K. Of the systems reviewed here, it has probably the largest community of adopters throughout the world, perhaps because it provides an easy-to-use repository-style application with the main purpose of provision to scholarly materials in a free and open manner.

Strengths

EPrints runs on multiple platforms including, with its latest release, Windows. Many features are customizable on a per-publication basis. It provides easy, author-initiated submission into the repository. It has a large deployment of supportive user and developer communities.

Weaknesses

Installation and overall configuration is accomplished at the command-line via Perl scripts. These processes would be ideally modeled by a GUI-installer utility, and all post-installation creation and configuration of individual archives would be ideally accomplished from within a Web-based GUI.

EPrints is not really a full-scale electronic publishing system in the same sense as some of the other systems in this review. EPrints is a repository system for providing easy and

open access to previously published works. As such, it does not attempt to model the whole peer review and journal production process.

Data collected for EPrints are listed in Figure Five.

Hyperjournal

One of the interesting features of Hyperjournal is that it was the first ePublishing system to employ an RDF metadata repository on the backend. The 2006 report from Barbera and DiDonato from that year's ELPUB: International Conference on Electronic Publishing makes for interesting reading in this respect.[5] The Hyperjournal model, intent on publishing both accepted *and* rejected articles in its repository is interesting because it acknowledges and accepts the fact that "the notion of quality varies and changes; it is affected by time, space, and cultural factors". That the Hyperjournal project as a whole embraces such a relativistic stance toward the value of research literature (and by extension toward the nature of truth itself) is intriguing. The fact that it then models a software system upon this belief is bold, providing evidence of unconventional and creative thought.

Strengths

Hyperjournal had one of the most appealing default user interfaces of the systems under review. Also, built on top of its RDF backend, its "contextualization" features quickly allow users of the system to jump from article to relevant article. Editorial workflow is completely customizable. Administrative roles can be added.

Weaknesses

Hyperjournal was a challenge to install. There is no full-text search capability. The application appears to only support a single publication per instance, i.e., if one wanted to use it to support five scholarly journals one would have to run five separate instances of the application.

Data collected for Hyperjournal are listed in Figure Six.

Open Journal System

Like EPrints, the Open Journal System (OJS) enjoys widespread community adoption and a relatively long history of development. Designed and developed by Canada's Public Knowledge Project, it is well supported by two major Canadian universities

[5] Barbera, Michele and Di Donato, Francesca (2006) Weaving the Web of Science : HyperJournal and the impact of the Semantic Web on scientific publishing. In Martens, Bob and Dobrova, Milena, Eds. Proceedings ELPUB : International Conference on Electronic Publishing (10th : 2006 : Bansko), pp. 341-348, Bansko (Bulgaria). http://eprints.rclis.org/archive/00007561/

(University of British Columbia and Simon Fraser University) as well as significant sponsorship by the Canadian government. Version 1.0 was released in November 2002; the most current version is 2.1.1; development is ongoing. OJS models the entire scholarly and scientific journal production and publication process, from initial submission to final archiving.

Strengths

OJS runs on multiple platforms, including Windows, and it is not Web server dependent, i.e., it runs on either Apache or IIS. It is easy to install and had the best, most comprehensive and clear documentation of any of the systems under consideration. It provides support for multiple discrete publications, all from within a single instance of the application. Each publication is separately skinnable. It appears to be highly extensible via a well-defined plugin API. It has a large deployment and an active developer and user community. OJS models the entire scholarly publications process, from author-initiated account generation and article submissions, through peer-review, editing, copy-editing, production, publication, and archiving. It includes well-thought-out administrative roles and default workflow. Its selection of bibliographic "reading tools" is interesting and useful.

Weaknesses

Based on this review, potential improvements for OJS would be support for an outside authentication mechanism, e.g., CAS, SiteMinder, WebAuth, Shibboleth; perhaps, like Hyperjournal and Topaz, integration with external RDF repositories; and the facility for using an external repository for persistent storage. Such additions are probably suitable for development as plugins, yet might be central enough for the main developers of OJS to consider making more closely coupled as part of the application architecture.

Data collected for Open Journal System are listed in Figure Seven.

Topaz

The Topaz Project originated as a commissioned work for the Public Library of Science (PLOS). It is now a separate, non-profit corporate entity. Topaz is interesting because it has a Service Oriented Architecture (SOA) against a Fedora repository backend and because it uses the Mulgara RDF database for bibliographic/bibliometric linkages.

Data collected for Topaz are listed in Figure Eight.

Special Cases

DiVA is a special case because the nature of its current licensing model is somewhat uncertain. As of this writing, it is not open-source and never has been. However, there is

currently some discussion of the future of its licensing. Insofar as it is a major European ePublishing project, sponsored by a consortium of Scandinavian universities and freely-available at least among those universities, it deserves a role in this study. The data included in this study was gleaned from documentation on the DiVA Website (http://www.diva-portal.org/).

Connexions/Rhaptos is currently undergoing a major rewrite, and the code was not available for analysis and evaluation at the time of this writing. The data included in this study was gleaned from documentation on the Connexions Website (http://cnx.org/). Nevertheless, all indications are that Connexions could become a major player in the ePublishing and learning materials space, especially looking forward to its next major release.

Topaz is a special case and is included here because of growing interest, especially given its connection to the Fedora Commons that has recently received a major grant from the Moore Foundation. Topaz has officially not even been released, and even when it is released its deployment will essentially be managed by the organization responsible for its original commission: The Public Library of Science. Much of our information about Topaz was gleaned from a telephone interview with its lead architect, Amit Kapoor, on May 4, 2007. At that time, Topaz was in the process of undergoing major architectural changes and had not yet been released.

Conclusion

Regarding APIs, most systems supported the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH), thereby supporting federated searching across OAI-compliant repositories as well as providing an open API for programmatic bulk extraction of metadata. Supporting OAI-PMH is therefore doubly useful. The other APIs noted in the list of attributes against that we evaluated each system were not nearly as prevalent. Based on our previous repository analysis, we considered the Edusource Communication Layer (ECL) and from the Java realm, JSR-170, both protocols governing content communication between repository systems.. From this current study, it is not clear that any of the systems support ECL. Regarding JSR-170, four of the systems evaluated were written in languages other than Java (including PHP, Python and Perl). It will be interesting to note whether the two Java-based applications, DiVA and Topaz, include support for JSR-170 in the future.

Both DPubS and Open Journal System support code extensibility: DPubS through the creation of a special directory in which to hold Perl code that is then configured to plug into their modular, service-based architecture; and Open Journal System via a well-defined plugin API for PHP developers.

One lesson from the Mellon-funded repository analysis is that the ease of installation of a new system is often a key indicator of its power and ease of use by both technical and non-technical users . There is nothing necessary about this relationship – it's certainly

possible that a system that is extraordinarily difficult to install is intuitively pleasing and usable when put to practical use – but such experiences are not consistent with our experience. It is a more typical case that software developers who take it as their mission to provide detailed, complete, elegant, and intuitive installation procedures for their systems also provide a system that as a whole reflects those same virtues. It is more typical that a group of developers who attempt to streamline and minimize the number of individual mental operations one must perform to install a system will likewise have streamlined and distilled such things as the user interface, workflow rules, administrative roles, etc. down to the simplest representations and procedures needed to accomplish the task at hand. "Everything should be made as simple as possible, but not simpler." This study supports this notion.

With this notion in mind, and noting the other criteria of this evaluation including the APIs available for each system, it is worth mentioning Open Journal System's ease of installation and comprehensive functionality to support the goal of modeling and implementing the operations of a scholarly publication, from author-initiated submission, through peer review, to editing, production, public publication and final archiving.

It should be noted that other systems possessed unique and useful features as well. For example, if one's objective is to provide quick and easy access, with a minimum of workflow process, to publications that have already been vetted, formatted, and are ready to be made public, then GNU EPrints provides this functionality very well.

Also notable are the deep customization features of DPubS, and the RDF features of both Hyperjournal and Topaz. Hyperjournal was the first such application to build upon an RDF engine (Sesame), and now Topaz appears to be following the same path (with the Mulgara engine). If other projects follow this approach, they would also be able to support and facilitate even more sophisticated bibliometric and citation-linked functionalities from within.

Some of the systems provide functionality to support authentication through an external authentication service such as CAS or WebAuth. The systems that do not support this functionality, the ones that rely exclusively on an internal database for authentication, should consider optionally providing such a service. Retaining the functionality for authenticating users against a local store allows authors the world over to self-submit articles for review; optionally enabling authentication against an external provider facilitates better enterprise-wide integration of the ePublishing system with other systems (portals, directory services, learning management systems, content management systems, etc.) at one's local institution.

Finally, it is important to note that the systems under consideration here were all at varying stages of evolution and degrees of adoption at the time of this writing. For this reason, Johns Hopkins University has developed a wiki to promote a continuing discussion of these ePublishing systems into the future. We encourage both the development teams and the users of these ePublishing systems to participate in this dialogue.

This study was made possible by a grant from the Open Society Institute (OSI).

Mark Cyzyk is the Scholarly Communication Architect in the Library Digital Programs Group of The Sheridan Libraries at Johns Hopkins University, Baltimore, Maryland, USA.

Sayeed Choudhury is the Associate Dean for University Libraries and Hodson Director of the Digital Research and Curation Center of The Sheridan Libraries at Johns Hopkins University, Baltimore, Maryland, USA.

Figure One

Article System
http://freshmeat.net/projects/artsys/

BioMed Central
http://www.biomedcentral.com/

CDS Invenio (formerly CDSware)
CDS Software Consortium (CERN)
http://cdsware.cern.ch/invenio/index.html

Connexions
Rice University
http://cnx.org/

DiVA
Electronic Publishing Centre, Uppsala University Library, Uppsala University, Sweden
http://www.diva-portal.se/
http://www.dlib.org/dlib/november03/muller/11muller.html

DPubS (Digital Publishing System)
Cornell University Library, in partnership with Pennsylvania State University Libraries and Press
http://dpubs.org/
http://www.arl.org/newsltr/237/opensource.html
http://projecteuclid.org/Dienst/UI/1.0/Home

Editoral Express
University of Maryland
http://gemini.econ.umd.edu/e-editor/
Fee-based

Epress
University of Surrey
http://www.epress.ac.uk/
Fee-based

Eprints
School of Electronics and Computer Science, University of Southampton
http://www.eprints.org/

ePublishing Toolkit
Living Reviews
https://dev.livingreviews.org/projects/epubtk/

Espere
UK Electronic Libraries Programme
http://www.espere.org/
Free for consortium institutions

GAPworks
German Research Foundation, DFG
http://gapworks.berlios.de/

HyperJournal
Net7, Italy
http://www.hjournal.org/
http://sourceforge.net/projects/hyperjournal/

Journal Management System
Scholarly Publishing Office, University of Michigan
http://spo.umdl.umich.edu/tools.html

Open Journal System
Public Knowledge Project, University of British Columbia and Simon Fraser University, Canada
http://pkp.sfu.ca/ojs/
http://www.pkp.ubc.ca/OJS_Sheet.html
http://pkp.sfu.ca/ojs/OJSinanHour.pdf
http://research2.csci.educ.ubc.ca/eprints/archive/00000047/01/Library_Hi_Tech_DRAFT.pdf

OSPRey (Online Submission and Peer Review system)
National Research Council of Canada
http://pubs.nrc-cnrc.gc.ca/rp/rptemp/rp2_news4_e.html

Roquade
Utrecht University and Delft University of Technology
http://www.roquade.nl/
http://www.library.uu.nl/staff/savenije/publicaties/RoquadeProject.htm

SciX Open Publishing Services (SOPS)
Scientific Information Exchange (SciX)
http://www.scix.net/sops.htm

Scribus
http://www.scribus.net/

Temple Peer Review Manager
Fox School of Business and Management, Temple Univeristy

http://peerreview.temple.edu/

Topaz
http://www.topazproject.org

Valet for ETDs
VTLS (Visionary Techology in Library Solutions)
http://www.vtls.com/Products/

Figure Two

Connexions/Rhaptos

Institutional affiliation and other indicators of the viability of the open-source project

Name of system:

Connexions/Rhaptos

Current version of system:

1.5.1

Tested version of system:

URL of project homepage:

http://rhaptos.org/

Institutional affiliation:

Rice University

Age of project:

Connnexions project started by Rice University in 1999.

Notes on long-term viability of project:

Due to the fact that this project was started in 1999 and has every appearance of going strong to this day, as well as the sponsorship of a major North American University, with additional funding from The National Science Foundation, National Instruments, the Hewlett-Packard Corporation, the George R. Brown Endowment for Undergraduate Education, and The CLASS Foundataion, this project is clearly viable for now and into the future.

Degree of deployment:

There are thousands of modules already in the Connexions system.

Type of open-source license:

Creative Commons Attribution License.

Licensing notes:

Other documentation (Webliography):

> Linda L. Biggs, "Open Source Connects Courseware at Rice University." Campus Technology. June 26. 2007. http://campustechnology.com/articles/48874/

Technical requirements, maintenance, scalability, and documented APIs

Local install or ASP?:

> This application is interesting because it can be run locally, in which case it is called "Rhaptos", or the existing installation at Rice University can be used, with its local look and feel. "Connexions" is the name given to this application when serving as an application service provided by Rice University.

Operating system requirements:

> Debian or Ubuntu

Hardware requirements:

Application server requirements:

> Zope (2.7.6 or 2.7.7)/Plone

Web server requirements:

Primary programming language:

> Python.

Auxiliary programming language:

> Educational content served by Connexions/Rhaptos is in CNXML, the Connexions markup language, and MathML, the XML-based markup language for mathematical equations.

Application framework:

> Zope

Database server requirements:

> PostgreSQL (version 8.2+) and psycopg, Python bindings for PostgreSQL. libxml, libsxlt, cnxml, mathml2, bibtexml,

qml -- xml libraries packaged and provided by Connexions. LaTex, including cjk-latex, tetex-extra, latex-ucs, latex-ucs-contrib, hbf-kanji48. Ghostscript. gif2png. Java Runtime Engine (JRE). OpenOffice 1.1X. HTML tidy library and its Python bindings.

Other software requirements:

CVS, and pycvs, the Python bindings for CVS.

Required skills:

Significant skills as a system administrator are required to install and configure this software. At a minimum, one should feel comfortable installing such things as Zope/Plone, PostgreSQL, and configuring a database connection between the two.

Internal backup and restore functions:

Scalability: Application:

Scalability: Data:

API: Code extensibility:

API: Batch ingest:

API: Batch ingest formats:

API: Batch export:

API: Batch export formats:

API: Support for JSR 170:

API: Support for OAI harvesting:

Content contained in this application is fully exposed via OAI-PMH.

API: Support for eduSource Communication Layer (ECL):

API: Support for other Web services:

Connexions/Rhaptos supports various REST-based Web Services. Individual content modules are directly addressable via URL. And attributes of individual content modules are directly addressable via URL. For example, to return just the title of the module posted to http://cnx.org/content/m11359/latest/, one simply calls its

getTitle method: http://cnx.org/content/m11359/latest/Title

Security notes:

Submission, peer review management, and administrative functions

Support for multiple, discrete publications:

Yes.

Multiple administrative roles:

At a minimum, it appears that the application ships with five hardcoded roles: Authors; Maintainers; Copyright Holders; Editors; and Translators.

Administrative roles configurable:

It does not appear that the administrative roles are configurable, i.e., it looks like they are hardcoded into the application and that additional administrative roles cannot be added.

Submission into system initiated by authors:

Yes.

Metadata fields configurable:

There is a short list of metadata fields available to all content items ("title"; "created"; "revised"; "abstract"; "keywords"; "license"; "authors"; "maintainers"; "licensors"), and the Website indicated that this list may be expanded on a content type by content type basis.

Editorial workflow configurable per publication:

Connexions was designed from the start so that authors could self-publish their works. However, in a July 19th, 2007 paper (http://rhaptos.org/docs/architecture/design/lenses/CNX%20Lens%20Functional%20Design%20Draft.pdf), Katherine Fletcher of Connexions proposes the introduction of something she calles "lenses", essentially a peer-review and workflow process for the Connexions/Rhaptos software. Interestingly, "Each lens may have a different focus; examples include lenses controlled by traditional editorial boards, professional societies, or informal groups of colleagues as well as automated lenses based on popularity,

the amount of (re)use, the number of incoming links, or other metrics." In this manner, a single piece of content could be viewed through several different "lenses".

Automated email alerts to authors:

Automated email alerts to editors:

Automated email alerts to reviewers:

Stylesheets, customizable look and feel per publication:

Rhaptos can be skinned to more closely match a local institution's look and feel. Skinning is accomplished at the Plone or Zope layers of this application.

Versioning:

The application maintains separate versions of each piece of content.

Archiving:

Access, formats, and electronic commerce functions

Accessibility of system:

Accessibility of document output:

Internationalization support:

For the most part, internationalization in Rhaptos is provided by the underlying Plone application.

Output in multiple document formats:

Document formats supported:

Browser plug-in requirements:

No browser plugins are required.

Usability notes:

Citation linking:

OpenURL resolver:

RSS feed:

Digital rights management:

Full-text search and retrieval:

Full text searching is supported.

Federated searching:

Federated searching is supported via OAI-PMH as well as OpenSearch (http://opensearch.a9.com/).

Authentication mechanisms:

Presumably, Connexions/Rhaptos has at its disposal all the functionalities of its underlying Plone foundation. Such functionality would include the use of, e.g., the PloneLDAP library for authenticating against an external LDAP or Active Directory service.

Subscription services:

It does not appear that any sort of subscription service has been implemented, although there is a document on the developer's Wiki making a proposal for the inclusion of primitive subscription services in a future release.

Electronic commerce functions:

Context-sensitive Help support:

Figure Three

DiVA (Digitala Vetenskapliga Arkivet)

Institutional affiliation and other indicators of the viability of the open-source project

Name of system:

DiVA (Digitala Vetenskapliga Arkivet)

Current version of system:

Tested version of system:

URL of project homepage:

http://www.diva-portal.org/about.xsql

Institutional affiliation:

Electronic Publishing Centre, Uppsala University

Age of project:

Project founded in 2000

Notes on long-term viability of project:

Project founded in 2000 by the Electronic Publishing Centre at Uppsala University, Sweden. The DiVA consortium was founded in 2002 and as of 2006 15 Scandinavian universities have joined. These institutions now collaborate on development and future direction of the DiVA application. Two user-group meetings are sponsored every year.

Degree of deployment:

Type of open-source license:

Licensing notes:

Other documentation (Webliography):

DiVA Publishing System: The Community's Collaborative Development Approach. http://epc.ub.uu.se/files/ELPUBfinal.pdf The DiVA Project - Development of an Electronic Publishing System. http://www.dlib.org/dlib/november03/muller/11muller.html

Technical requirements, maintenance, scalability, and documented APIs

Local install or ASP?:

Local installation

Operating system requirements:

Any operating system capable of running a servlet container, e.g., Tomcat

Hardware requirements:

No specific hardware requirements

Application server requirements:

Tomcat

Web server requirements:

Apache

Primary programming language:

Java

Auxiliary programming language:

XML

Application framework:

Database server requirements:

Oracle

Other software requirements:

Required skills:

Internal backup and restore functions:

Scalability: Application:

Scalability: Data:

API: Code extensibility:

API: Batch ingest:

API: Batch ingest formats:

API: Batch export:

API: Batch export formats:

API: Support for JSR 170:

API: Support for OAI harvesting:

OAI-PMH harvesting is supported.

API: Support for eduSource Communication Layer (ECL):

API: Support for other Web services:

RSS feeds from DiVA are supported.

Security notes:

Submission, peer review management, and administrative functions

Support for multiple, discrete publications:

Multiple administrative roles:

Administrative roles configurable:

Submission into system initiated by authors:

Metadata fields configurable:

Documents are natively stored in the "DiVA Document Format", and XML-based document format consisting of 99 elements. However, the metadata structures can be configured to support other metatdata standards, e.g., Dublin Core, METS, etc.

Editorial workflow configurable per publication:

Automated email alerts to authors:

Automated email alerts to editors:

Automated email alerts to reviewers:

Stylesheets, customizable look and feel per publication:

Versioning:

Archiving:

Access, formats, and electronic commerce functions

Accessibility of system:

Accessibility of document output:

Internationalization support:

Output in multiple document formats:

Document formats supported:

PDF (via Apache FOP)

Browser plug-in requirements:

Usability notes:

Citation linking:

OpenURL resolver:

RSS feed:

Digital rights management:

Full-text search and retrieval:

Full text search and retrival is supported via the Apache Lucene engine.

Federated searching:

Authentication mechanisms:

Subscription services:

Electronic commerce functions:

Context-sensitive Help support:

Figure Four

DPubS (Digital Publishing System)

Institutional affiliation and other indicators of the viability of the open-source project

Name of system:

DPubS (Digital Publishing System)

Current version of system:

2.0

Tested version of system:

2.0

URL of project homepage:

http://dpubs.org/

Institutional affiliation:

Cornell and Penn State

Age of project:

Started as Project Euclid in 2000. Morphed into DPubS, and Cornell and Penn State joined forced on this project in 2004.

Notes on long-term viability of project:

DPubS has two strong institutions backing it. Development is active and ongoing. The next major version of DPubS is currently (spring 2007) under development.

Degree of deployment:

It is unclear how widely deployed DPubS is. Their wiki lists five major projects using it.

Type of open-source license:

Educational Community License

Licensing notes:

Other documentation (Webliography):

Ehling, Terry. DPubS: The Development of an Open Source Publishing System. Publishing Research Quarterly, 2005, v20(4), p 41.

Technical requirements, maintenance, scalability, and documented APIs

Local install or ASP?:

Local installation.

Operating system requirements:

Solaris or UNIX variant. We installed it under Ubuntu.

Hardware requirements:

Minimum 256MB of RAM recommended.

Application server requirements:

Apache with mod_perl.

Web server requirements:

Apache with mod_perl, mod_rewrite

Primary programming language:

Perl 5.8+

Auxiliary programming language:

Java Runtime Environment (JRE) is required if using the Lucene engine for fulltext indexing.

Application framework:

There is no specific application framework, per se. But the application seems to be nicely structured around various well-defined, internal services, e.g., there is a service that handles repository transactions, one that handles transactions related to subscriptions, a User Interface Service, an Admin Service, etc.

Database server requirements:

The application uses SQLite for persistent storage. Alternatively, the application can be configured to interact with external data stores such as Fedora and DSpace.

Other software requirements:

Apache mod_perl and mod_rewrite. Perl libraries and modules: XML::LibXML, XML::LibXSLT, XML::Writer, DB_File, Bundle::LWP, Unicode::String, Digest::SHA1, MIME::Tools, MIME::Lite, Archive::Zip, IO::Scalar, CGI::Session, Archive::Tar, Date::Manip, IO::Zlib, Mail::Address, DBI, DBD::SQLite, File::Copy::Recursive, Compress::Zlib, Time::ParseDate, HTML::Template, SOAP::Lite, ModPerl::Registry

Required skills:

DPubS requires significant skills as a UNIX system administrator to install. If installing on a shared server, among other Web sites and applications, one must be able to configure multiple Virtual Hosts under Apache. One must be able to troubleshoot problems related to Apache configuration and startup. One must be able to install and troubleshoot mod_perl under Apache. One must be able to troubleshoot problems with the Berkeley Database libraries.

Internal backup and restore functions:

It is not clear that there is/is not any sort of internal backup/restore functionality in this application.

Scalability: Application:

Scalability for this application would be handled at the Apache/mod_perl layer.

Scalability: Data:

The use of SQLite is odd. The assumption here is that to be scalable the application would have to be configured to run against either Fedora or DSpace.

API: Code extensibility:

It appears that the application codebase is significantly extensible. One must create a directory outside the root of the application codebase in which to hold one's new code. This is so that local, custom code is not overwritten upon future updates to the application proper. Then, there is a special config file within the application's directory tree that can be modified such that local code is initialized and incorporated into the application upon startup.

API: Batch ingest:

There is a somewhat convoluted process involved in getting data into this application. One must manually, at the command-line, create a subdirectory within a main directory holding data for a particular publication. This subdirectory will hold data for a particular journal "issue". Properly formatted content and metadata files are then placed into this directory. A command-line Perl script is run which does the job of importing the data into DPubS. Another command-line Perl script is run to update any indexes. Finally, Apache must be restarted.

API: Batch ingest formats:

Interestingly, the application can be configured to accept any file format. By default is accepts many common file formats, and it enables a system administrator to configure the acceptance of a new file format through the creation of XML format definition files. The application also provides support for something called "dynamic formats", i.e., formats derived from other formats.

API: Batch export:

API: Batch export formats:

API: Support for JSR 170:

The application is written in Perl and so does not support JSR 170.

API: Support for OAI harvesting:

The application fully supports OAI harvesting of metadata. Insofar as OAI requires metadata to be provided in Dublin Core format, if the metadata schema you are using does not include these DC fields you must first create a derived format, "crosswalking" from your idiosyncratic metadata schema to the Dublin Core standard. The resulting crosswalked fields are then exposed to OAI metadata harvesting.

API: Support for eduSource Communication Layer (ECL):

ECL does not appear to be supported.

API: Support for other Web services:

Security notes:

Submission, peer review management, and administrative functions

Support for multiple, discrete publications:

> Yes. One must decide first on the metadata schema to be used, create a new internal "authority" (unique identifier) for this schema, and at the command-line, edit XML configuration files accordingly. Apache must be restarted for these configuration files to take effect. At this point, a new publication with the specified metadata schema has been created, content can be loaded, and the UI can be customized.

Multiple administrative roles:

> It appears that there are only two roles modeled by this application: Editor and User.

Administrative roles configurable:

> The Editor and User roles of this application do not appear to be configurable, i.e., the Editor in one publication appears the have the same privileges as in another.

Submission into system initiated by authors:

> It is unclear how author-initiated submissions are handled. The wiki indicates how entire issues of properly-formatted content can be imported into the application, but no mention is made of direct author submissions.

Metadata fields configurable:

> One of the great strengths of this application is that it supports multiple custom metadata schemas, i.e., each individual publication can have its own idiosyncratic metadata schema.

Editorial workflow configurable per publication:

> It is unclear how workflow is handled. On the one hand, it appears that new User Interface "pages" can be created and incorporated into the application. On the other hand, it is unclear just how much of the logic of the application can be manipulated via these pages. It may be that a programming could create new User Interface pages which then call the

various underlying services of the application and thereby alter or create a new workflow procedure for use within the application framework. But this is something a developer/programmer would be doing, not an administrator of this application.

Automated email alerts to authors:

It is unclear whether automated alerts to authors are included as part of this application. It appears that the peer review process is not modeled by this application.

Automated email alerts to editors:

Automated email alerts to reviewers:

The application does not understand the role of "reviewer".

Stylesheets, customizable look and feel per publication:

The look and feel of individual publications is customizable via XSL stylesheets. At the command-line, the default directory structure containing the default stylesheets must be copied to a new directory, one that maps to the publication under consideration. The default stylesheets are then edited until the desired look and feel is attained. In addition to creating custom skins, the application makes provision for creating entirely new UI pages as well.

Versioning:

It does not appear that the application maintains separate versions of documents.

Archiving:

Access, formats, and electronic commerce functions

Accessibility of system:

Accessibility of document output:

Internationalization support:

Output in multiple document formats:

It does not appear that the application itself generates output formats. Rather, the application can accept multiple input formats and so the format in which a document is initially

submitted remains the format in which is is ultimately provided.

Document formats supported:

Browser plug-in requirements:

No browser plugins are required to use this application.

Usability notes:

Citation linking:

Citation linking or other bibliometric utilities do not appear to be present in this application.

OpenURL resolver:

RSS feed:

Digital rights management:

Full-text search and retrieval:

Yes, via the Lucene engine.

Federated searching:

The application fully supports OAI metadata harvesting and therefore supports federated searching.

Authentication mechanisms:

Authentication appears to be entirely internal, i.e., there is no provision for authentication against an external service.

Subscription services:

The application provides for subscription services and provides access control function on a per IP, per domain, or per user basis.

Electronic commerce functions:

This was the only application under consideration by this study whose documentation even mentioned eCommerce functions. Presumably, one could use this application in an eCommerce setting by controlling access via the subscription services it models. Most notably, the subscription services can control access by domain.

Context-sensitive Help support:

Summary data

Strengths:

> Impressive, well-thought-out service-based application architecture. Provision for subscription services. Highly customizable metadata schema.

Weaknesses:

> Platform dependent. Web server dependent. Extraordinarily difficult to install. Primitive initialization script. Installation documents assume that DPubS will be the only application running on the server, i.e., it's not intended to run in tandem with any other application. Installation assumes one is installing Apache from source. Installation assumes one is installing mod_perl from source. Documentation significantly incomplete/incorrect. It does not appear that this application is intended to model/facilitate the entire peer review process. Rather, it looks like the application is intended to provide a repository for already completed publications and to then provide a Web-based interface to them.

Figure Five

GNU EPrints

Institutional affiliation and other indicators of the viability of the open-source project

Name of system:

GNU EPrints

Current version of system:

3.0.1 beta

Tested version of system:

3.0

URL of project homepage:

http://www.eprints.org/

Institutional affiliation:

School of Electronics and Computer Science, University of Southampton

Age of project:

Project founded in 2000

Notes on long-term viability of project:

Formal community programme. Fee-based EPrints Services unit. EPrints seems to be widely-deployed and well-supported.

Degree of deployment:

EPrints is perhaps the most widely deployed of the open-source ePublishing systems under consideration by this study. As of this writing, the application's wiki page lists 223 separate, known archives actively using the software in production.

Type of open-source license:

GNU General Public License (GPL), Version 2 or later

Licensing notes:

Other documentation (Webliography):

Ruth Martin, "ePrints UK: Developing a national e-prints archive." Ariadne, 35, March/April 2003. http://www.ariadne.ac.uk/issue35/martin/ Peter Millington and William J. Nixon. "EPrints 3 Pre-launch Briefing." Ariadne, 50, January 2007. http://www.ariadne.ac.uk/issue50/eprints-v3-rpt/

Technical requirements, maintenance, scalability, and documented APIs

Local install or ASP?:

Local installation

Operating system requirements:

GNU/Linux. Also Solaris and MacOS. Version 3 now runs under Apache on Windows.

Hardware requirements:

No specific hardware requirements are mentioned.

Application server requirements:

Apache, with mod_perl

Web server requirements:

Apache, with mod_perl

Primary programming language:

Perl 5.6.1

Auxiliary programming language:

Application framework:

Database server requirements:

MySQL

Other software requirements:

Perl modules: Data::ShowTable; DBI; Msql-Mysql Module; MIME::Base64; Unicode::String; XML::Parser; Apache; CGI; Carp; Cwd; Data::Dumper; Digest::MD5; File::Basename; File::Copy; File::Find; File::Path;

Getopt::Long; Pod::Usage; Sys::Hostname Additional modules required for GDOME (XML) support: libxml2; libxml2-devel; XML-LibXML-Common; XML-NamespaceSupport; XML-GDOME Other modules/utilities: wget; tar; gunzip; unzip; xpdf (for PDF indexing); wvware (for MS Word indexing); lynx (for HTML indexing); latex/dvips/convert for display of latex equations

Required skills:

While version 3 of this application can run under Windows, all instances of it must run under Apache, so experience setting up Apache is required as is experience setting up the many Perl modules that are required. Unfortunately, the installation documents seem to assume that EPrints will be the only application running in Apache, i.e., that it will not be running on a shared server. This is a bad assumption, which led to problems during the installation phase. Moreover, as I painfully found out, the order of operations in which the installation occurs must be followed to the letter, even if there are one or two steps that seem, on the surface, like the order in which they execute would not be relevant. Still, the documentation for installing EPrints on Ubuntu provides a step-by-step installation procedure that, if followed and not deviated from in the slightest, results in a successful install. A GUI-based installer would be nice. As it stands, installation is handled via a somewhat primitive Perl script.

Internal backup and restore functions:

There is no internal backup and restore feature. To backup a set of archives one must back up all files under the EPrints root and use the backup features native to MySQL to backup all metadata.

Scalability: Application:

The layer here that must be scaled up is the Apache layer, so all the usual methods for scaling Apache, e.g., usage of a front-tier mod_proxy instance, apply.

Scalability: Data:

Since the content files for each individual archive is fully contained within its own directory tree, these directory trees could easily be distributed across multiple physical servers via, e.g., NFS shares. More, the metadata for each archive is

contained in a MySQL database which itself can now be clustered.

API: Code extensibility:

The application provides a defined API for the creation of plugins. It also provides support for packaged "extensions", basically entire sets of plugins all installed as a single package.

API: Batch ingest:

The wiki page for this application mentions Import Plugins, but no other information is available. It is unclear whether import plugins are shipped with the product at this time, or whether provision of them is something that will be present in a future release. The application, however, also provides an XML-based import and export format whereby the XML structure itself is relative to the fields of the individual repository/archive being used. It is not clear, though, from the documentation precisely how one would go about using this format to import and export data.

API: Batch ingest formats:

API: Batch export:

The application supports the export of records via plugin modules. Batch export of records can occur via the EPrints Web interface or via a command-line script.

API: Batch export formats:

Export of data can be accomplished via plugins for, e.g., Dublin Core, EndNote, METS, OAI, RSS, Atom, HTML, etc.

API: Support for JSR 170:

The application is written in Perl and so does not support JSR170.

API: Support for OAI harvesting:

OAI-PMH harvesting supported. EPrints was created to support OAI-PMH from the start.

API: Support for eduSource Communication Layer (ECL):

The application does not appear to support the eduSource

Communication Layer.

API: Support for other Web services:

Security notes:

Submission, peer review management, and administrative functions

Support for multiple, discrete publications:

Multiple archives are repositories are supported, each housing multiple documents, files, etc.

Multiple administrative roles:

The application provides four distinct roles: The main Administrator; the Repository Administrator; the Editor within a given repository; and the individual User.

Administrative roles configurable:

The roles provided by the application appear for be hard-coded, i.e., you cannot add to the number of these roles.

Submission into system initiated by authors:

A self-signup is provided for new authors. Once an account is generated, authors may submit to a particular respository. Their submission enters the idiosyncratic workflow for that repository where it may be reviewed and approved by, e.g., a repository editor before being put on public display.

Metadata fields configurable:

The metatdata is alterable on a per-archive basis. This is accomplished via editing of two configuation files on the command-line. More, if a field is added within these configuration files, it must likewise be manually added/configured in the database. The wiki indicates that work is underway to create a "tool" which will make this whole process much easier.

Editorial workflow configurable per publication:

Separate workflows can be created on a per-repository basis using XML configuration files contained in the directory tree for that particular repository instance. It appears that segments ("stages") of the custom workflow can be restricted per user type, i.e., to Repository Administrators,

to Editors, to regular Users.

Automated email alerts to authors:

Alerts can be configured such that Users of the each individual repository can sign up to receive notification of added items meeting their specified search criteria.

Automated email alerts to editors:

Insofar as the application supports the notion of a repository Editor, and insofar as it also supports custom notification based on specified criteria, it appears the application can be configured to send out automated email notification to Editors in the case of, e.g., new submissions awaiting review. Within the default configuration, the application then supports a Move to Repository, Return item (with notification), and Destroy item (with notification).

Automated email alerts to reviewers:

Stylesheets, customizable look and feel per publication:

The look and feel is configurable on a per-repository basis. Again, this is controlled via command-line manipulation of configuration files and contents of repository directories. With each change, such things as static pages must be regenerated, the default configuration for the archive must be reloaded, and ideally the Web server must be restarted.

Versioning:

New versions of documents can be submitted. The old version is retained and linked to the new version. More, the record for a document can be used as a "template" for creating an exactly similar, though unlinked and formally unrelated, record in the application. This new record can then be edited as needed.

Archiving:

In addition to its internal archive of documents, the application maintains a complete history of every digital object as it enters the repository. It can then provide this log, along with all metadata associated with a given object, all related objects and metadata, and all licensing information, as a piece to an outside preservation service. That is, if EPrints is just being used to provide online access to documents, it may be part of a larger effort where

longterm preservation is addressed by a separate system. In this case, EPrints provides that system not only with its digital objects themselves, but with their associated metadata as well.

Access, formats, and electronic commerce functions

Accessibility of system:

Accessibility of document output:

Internationalization support:

The application and database fully supports Unicode encoding (utf-8). Locale files are installed and configured at the command-line.

Output in multiple document formats:

Insofar as the application accepts multiple documents formats for input, it likewise provides those documents to the user in the same format in which they were submitted.

Document formats supported:

The default document formats supported include: Plain text; HTML; PDF; Postscript; MS Powerpoint; MS Word; JPEG; PNG; GIF; TIFF; BMP; MPEG; Quicktime; AVI.

Browser plug-in requirements:

No browser plugins are required to interact with this application.

Usability notes:

The application running under the default configuration was usable, its streamlined interface made perfect sense, was easy to navigate, and was attractive. Insofar as the main goal of this application is to provide quick and easy access to entire repositories of documents, the fact that the main links on the homepage include "Latest Additions", "Search Repository", and "Browse Repository" are apt and useful.

Citation linking:

There do not appear to be any sort of citation linking or other bibliometric utilities or services built in to the default configuration of this application. However, a sister project -

- CiteBase (http://www.citebase.org) -- is intended to be "a semi-autonomous citation index for the free, online, research literature" such as that provided by EPrints.

OpenURL resolver:

EPrints version 3.0 provides an OpenURL resolver.

RSS feed:

By default, plugins for both RSS and Atom feeds from individual repositories are provided.

Digital rights management:

There is no provision for file-level digital rights management. There is, however, metadata attached to each record in the repository denoting the particular license attached to it, e.g., various flavors of Creative Commons. The whole point of this software is to make documents openly available.

Full-text search and retrieval:

Yes. The following are required: xpdf (for PDF indexing); wvware (for MS Word indexing); lynx (for HTML indexing)

Federated searching:

All metadata is exposed to OAI harvesting and federated searching.

Authentication mechanisms:

The application can be configured to support authentication against an external LDAP server. By default, it authenticates against its internal authentication store. Interestingly, the application can be configured to be a Login-Only repository (where all interations with it must first be authenticated) or as a repository in which user registration is not even required.

Subscription services:

Insofar as this application is not a electronic publishing system in the same sense as the other systems under consideration by this study are, it does not provide subscription services. In another sense, though, it supports RSS and Atom feeds, so at least in that sense the notion of

"subscription" is provided.

Electronic commerce functions:

The application does not appear to provide any sort of ecommerce functions. Its main bent, in fact, is to provide fully open access to materials.

Context-sensitive Help support:

The application provides clickable Help buttons for each field in the various forms throughout. The default screens, though, are so streamlined, well-designed, and self-explanatory that further help facilities appear to be unneeded.

Summary data

Strengths:

The application nicely provides facility for controlled-vocabulary indexing of documents using the Library of Congress Subject Headings and/or the organizational structure of one's local institution. The default application is simple, yet powerful. The administrative roles and default, streamlined workflow are well-thought-out and useful. The workflow, branding, and import/export is all configurable, though all at the command-line by a system administrator and not particularly easy or straightforward.

Weaknesses:

Installation procedures assume that ePrints is being installed on its own server. Web server dependent (Apache). Primitive installation script. The configuration of the application as a whole, as well as of each individual archive, is performed at the command-line by a system administrator, using text-based configuration files. The EPrints wiki indicates that administrative tools, presumably GUI in nature, are currently under development.

Figure Six

Hyperjournal

Institutional affiliation and other indicators of the viability of the open-source project

Name of system:

Hyperjournal

Current version of system:

0.5b (beta)

Tested version of system:

0.5b (beta)

URL of project homepage:

http://www.hjournal.org/

Institutional affiliation:

Net7 and the University of Pisa

Age of project:

Project started in 2004.

Notes on long-term viability of project:

The longterm viability of this project is uncertain. It was initially supported by the University of Pisa Political Science department as well as a small Italina software firm, Net7. Something called the "Hyperjournal Association" was then formed in an effort to create an organization suitable for longterm planning and growth of the project. The Hyperjournal Association appears be an organization that accepts fee-based memberships for funding. It is uncertain if this funding strategy will work in the long term.

Degree of deployment:

Type of open-source license:

GPL2

Licensing notes:

Interestingly, Hyperjournal also makes an author specify an open-source license to apply toward his individual article submission.

Other documentation (Webliography):

Barbera, Michele and Di Donato, Francesca (2006) Weaving the Web of Science : HyperJournal and the impact of the Semantic Web on scientific publishing. In Martens, Bob and Dobrova, Milena, Eds. Proceedings ELPUB : International Conference on Electronic Publishing (10th : 2006 : Bansko), pp. 341-348, Bansko (Bulgaria). http://eprints.rclis.org/archive/00007561/

Technical requirements, maintenance, scalability, and documented APIs

Local install or ASP?:

Local installation.

Operating system requirements:

Linux, or other UNIX variant. We installed it under Ubuntu.

Hardware requirements:

No specific hardware requirements.

Application server requirements:

Tomcat is required to run the Sesame RDF repository.

Web server requirements:

Apache, with mod_rewrite.

Primary programming language:

PHP

Auxiliary programming language:

Application framework:

Database server requirements:

MySQL

Other software requirements:

The Sesame RDF repository running within a Java servlet container with appropriate JDBC driver for connectivity to MySQL installed in the proper place.

Required skills:

In contrast to the claims made on the Hyperjournal Website, significant skills are required to install this application. There are many steps along the installation path where things can, and will, go wrong. One must make educated guesses along the way. Must be able to install Apache, with mod_rewrite enabled. (The documentation does not mention this.) Must be able to install and configure, on a UNIX host, a mail transport agent such as Sendmail or Postfix. Must be knowledgeable about UNIX permissions issues. Must know how to install TrueType fonts on a UNIX host. Must be able to install and configure Tomcat on a UNIX host and the Sesame RDF repository on Tomcat. Must be able to install and configure MySQL. Must be able to install PHP under Apache on a UNIX host and to use the PEAR utility to install various required libraries. Must be able to troubleshoot connectivity to MySQL server via JDBC driver. Must be able to troubleshoot connectivity to Sesame repository. Must be able to troubleshoot sourcecode configure scripts and make files. Installing and configuring Hyperjournal is not a trivial task. In the end, despite hints in the documentation to the contrary, the only way I was able to get Hyperjournal installed and configured properly with MySQL and a local, not remote, Sesame repository was to let the Hyperjournal GUI installation utility actually create MySQL users and databases for use by Hyperjournal and Sesame, and to let the GUI utility create the Sesame repository under Tomcat as well. Even after doing this, though, the configuration required significant tweaking in order to get such things as the automated email messages, the "captcha", and JDBC connectivity from Sesame to MySQL to work.

Internal backup and restore functions:

Backup and restore appears to be performed at the database and file system. There does not appear to be an internal mechanism present for bulk export or import of data.

Scalability: Application:

Application scalability is handled by the Web server.

Scalability: Data:

Data scalability is handled by the database platform.

API: Code extensibility:

There does not appear to be an API for extending the capabilities of this application.

API: Batch ingest:

There does not appear to be an internal mechanism present for bulk import of data.

API: Batch ingest formats:

API: Batch export:

There does not appear to be an internal mechanism present for bulk export of data.

API: Batch export formats:

API: Support for JSR 170:

Insofar as this application is written in PHP, not Java, it is not extensibile via JSR170.

API: Support for OAI harvesting:

This application fully exposes its meta-data via OAI-PMH.

API: Support for eduSource Communication Layer (ECL):

This application does not provide an API for the eduSource Communication Layer.

API: Support for other Web services:

Other than exposure of meta-data via OAI-PMH, no other Web Services appear to be provided.

Security notes:

Submission, peer review management, and administrative functions

Support for multiple, discrete publications:

Publication of multiple journal titles are not supported. Only a single journal title per application instance is

provided.

Multiple administrative roles:

The application provides the following roles by default: Authors; Administrators; Reviewer; Editors.

Administrative roles configurable:

Administrative roles can be added, with custom sets of permissions for each.

Submission into system initiated by authors:

Yes, all submissions are initiated by the author.

Metadata fields configurable:

The meatdata, which appears to be based on Dublin Core, does not appear to be configurable, i.e., it does not look like additional fields can be added to what is already present by default.

Editorial workflow configurable per publication:

Workflow is fully customizable.

Automated email alerts to authors:

The submission and peer review process within Hyperjournal truly is "blind". Authors sign up and are issued Hyperjournal accounts. Then communications take place within drop boxes inside the Hyperjournal application itself. Authors must periodically check back to see what the current status of their submission is. Only once a submission has been fully approved can author attribution be added to the record.

Automated email alerts to editors:

The system alerts Editors of the status of submissions all along the workflow path.

Automated email alerts to reviewers:

The system alerts Reviewers of the status of submissions all along the workflow path.

Stylesheets, customizable look and feel per publication:

The logo that appears thoughout the UI is configurable from

within the Administrative screens. Custom "Interface Themes" can be created with Cascading Stylesheets (CSS) and can be registered with the application by placing them in a specified directory on the underlying filesystem.

Versioning:

"Revisions" are generated and maintained for each submitted manuscript.

Archiving:

There does not appear to be an archiving function, or hooks into external repositories.

Access, formats, and electronic commerce functions

Accessibility of system:

Accessibility of document output:

Internationalization support:

A configurable list of acceptable languages is presented to the author upon as part of the submission process.

Output in multiple document formats:

Document formats supported:

Browser plug-in requirements:

No browser plugins are required to run and use this application. The entire application runs within a standard, modern Web browser.

Usability notes:

The user interface (UI) of this application was clear, intuitive, and a pleasure to use. Labels were clear, and application functions seemed well thought out.

Citation linking:

Via Hyperjournal's unique RDF-backed "contextualization" repository, links between individual articles are provided for Cited authors; Citing authors; Cited works; and Citing works.

OpenURL resolver:

RSS feed:

Digital rights management:

> There is no provision for digital rights management. Promotiong of open access is one of the central goals of this software.

Full-text search and retrieval:

> There does not appear to be a fulltext index of the entire article. Titles are keyword searchable. Author names are searchable. There is a controlled-vocabulary subject search as well.

Federated searching:

Authentication mechanisms:

> Authentication is provided internally, by the application itself. There appears to be no provision for authentication against an external store or service.

Subscription services:

> No subscription services are provided.

Electronic commerce functions:

> No ecommerce services are provided.

Context-sensitive Help support:

Summary data

Strengths:

> The user interface (UI) of the application is well-laid-out and easily-understood. It was appealing and a pleasure to work with. The default administrative roles and workflow were well-thought-out. Hyperjournal is the first example of its kind: A Semantic-Web-Aware electronic publishing system. All of its data is exposed as RDF for harvesting and use within the Semantic Web rubric. Its "contextualization" features provide powerful and useful bibliometric tools and allow users to quickly enter a stream of relevant, linked, bibliographic data.

Weaknesses:

> A challenge to install. Installation documentation slightly,

yet significantly, out of date. Platform dependent. Data import/export is missing. No defined APIs for code extensibility/development of extensions or plugins.

Figure Seven

Open Journal Systems

Institutional affiliation and other indicators of the viability of the open-source project

Name of system:

Open Journal Systems

Current version of system:

2.1.1

Tested version of system:

2.1.1

URL of project homepage:

http://pkp.sfu.ca/?q=ojs

Institutional affiliation:

Public Knowledge Project: University of British Columbia and Simon Fraser University

Age of project:

Version 1.0 released in November 2002.

Notes on long-term viability of project:

OJS is a subproject of the federally (Canadian) funded Public Knowledge Project, a partnership between the University of British Columbia Faculty of Education, the Simon Fraser University Library, and the Simon Fraser University Canadian Centre for Studies in Publishing. The various projects of the PKP have been funded by: British Columbia Teachers Federation; International Network for the Availability of Scientific Publications; Canadian Association of Research Libraries; Social Sciences and Humanities Research Council of Canada; International Development Research Council; John D. and Catherine T. MacArthur Foundation; Open Society Institute, Soros Foundation; Max Bell Foundation; Government of Canada, Office of Learning Technologies.

Degree of deployment:

Type of open-source license:

GNU General Public License 2+

Licensing notes:

Exact sub-version of the GPL2 is up to the user to decide.

Other documentation (Webliography):

Willinsky, J. (2005). Open Journal Systems: An example of Open Source Software for journal management and publishing. Library Hi-Tech 23 (4), 504-519. http://pkp.sfu.ca/node/433 da Fonseca, R.M.S. (2004, June). Open Journal Systems. Paper presented at the ICCC 8th International Conference on Electronic Publishing, Brasilia, Brazil. http://pkp.sfu.ca/node/473

Technical requirements, maintenance, scalability, and documented APIs

Local install or ASP?:

Local installation.

Operating system requirements:

Windows, Unix, or Linux. Unix-like OS recommended.

Hardware requirements:

No specific hardware requirements.

Application server requirements:

An auxiliary application server is not required.

Web server requirements:

Apache 1.3.2+ or 2.0.4+ or IIS 6+ (We were, however, able to install it for testing purposes under IIS 5.1 on the WinXP platform.)

Primary programming language:

PHP 4.2+ (IIS requires PHP 5.0+)

Auxiliary programming language:

None.

Application framework:

None used.

Database server requirements:

MySQL 2.23+ or PostgreSQL 7.1+

Other software requirements:

The following helper applications must be present on the server if PDF, Postscript, and Microsoft Word documents are to be fulltext indexed: pstotext; pdftotext; ps2ascii; antiword; catdoc.

Required skills:

Required skills for setup and administration include the following: Ability to set up, configure, administer, and secure a Web server, either Apache or IIS; ability to set up and configure PHP with either the MySQL or PostgreSQL connector; ability to set up and administer either the MySQL or PostgreSQL database server.

Internal backup and restore functions:

There does not appear to be an internal backup or restore function. Backup and restore would therefore happen on the database server and on the local file system in the usual way.

Scalability: Application:

Application scalability is handled at the Web server or network content switch; it is not specifically addressed by this application.

Scalability: Data:

Data scalability is handled at the database server; it is not specifically addressed by this application.

API: Code extensibility:

The application provides a robust plugin API. Examples of community-produced plugins include: An RSS/Atom feed plugin; a WYSIWYG editor plugin; an LDAP authentication plugin; a PubMed XML export plugin; a

Google Scholar Gateway plugin; etc. Plugins are written in object-oriented PHP and typically extend one of the four provided base classes: Generic; importexport; auth; and gateways. The Open Journals Systems Technical Reference provides ample instruction and examples on how to write plugins for the application.

API: Batch ingest:

Supported through plugin modules.

API: Batch ingest formats:

The application ships with a custom DTD, "native.dtd", that enables import of valid XML documents representing single articles, multiple articles, single issues, and multiple issues.

API: Batch export:

Supported through plugin modules.

API: Batch export formats:

By default, export plugins are provided for the following XML formats: CrossRef; Erudit; PubMed. The CrossRef and PubMed plugins support export of article metadata whereas the Erudit plugin supports export of the fulltext of the articles themselves.

API: Support for JSR 170:

This application is written in PHP and so does not support the JSR-170 API.

API: Support for OAI harvesting:

Metadata from each installation of OJS is fully exposed to OAI harvesting. Interestingly, the Public Knowledge Project has another product, the OAI Harvester, that aggregates, indexes, and provides a public search interface to OAI-enabled repositories, including OJS.

API: Support for eduSource Communication Layer (ECL):

It does not appear to be the case that the application natively supports the eduSource Communication Layer protocol, although it could in the future via a plugin. This is ironic considering ECL was created, promoted, and supported by the Laboratory for Ontological Research (LORE) at Simon Fraser University -- one of the supporting

institutions of this application.

API: Support for other Web services:

No other default Web Services are provided, other than what is listed above. Additional, custom, Web Services could be provided in the future via community-submitted plugins.

Security notes:

SSL encryption can be enforced for the entire application or just the login portion of the application. Invalidating a user session if the user's IP address changes is a global option. Data encryption can use either the MD5 or SHA1 algorithms.

Submission, peer review management, and administrative functions

Support for multiple, discrete publications:

Yes, multiple publications (journals, and issues of those journals) are supported.

Multiple administrative roles:

Yes, the application provides multiple administrative roles. These roles include: Author; OJS Superuser; Journal Manager; Editor; Section Editor; Copy Editor; Layout Editor; Proofreader.

Administrative roles configurable:

The provided administrative roles do not appear to be configurable, i.e., there does not appear to be any way to add an administrative role to the system. That said, the administrative roles provided appear to be comprehensive and very well thought out.

Submission into system initiated by authors:

Submissions are author-initiated, and file uploading is done via the application. Metadata is supplied by the author at the time of submission. Resubmissions can occur at the editor's request.

Metadata fields configurable:

Metadata fields in this application do not appear to be

configurable, e.g., they cannot be added to.

Editorial workflow configurable per publication:

While the workflow is not configurable, a lot of thought was put in to the hardcoded workflow and editorial processes hardcoded into this application. The main OJS documentation ("OJS in Ten Minutes") provides a very nice chart of the workflow modeled by the OJS application. This chart nicely illustrates the movement of a submission through the workflow process and the various interactions between authors, editors, reviewers, and other editoral staff along the way.

Automated email alerts to authors:

The application by default supplies many "prepared emails" that are used to notify authors of the status of their submission as it moves through the workflow. These prepared email messages are editable by the Journal Manager. The sender of these automated messages is one of the configured Users of the application, e.g., Section Editor; Copy Editor; etc.

Automated email alerts to editors:

Both authors and editorial staff are automatically alerted via prepared email messages as a submission moves through the workflow.

Automated email alerts to reviewers:

Automated alerts to Reviewers can be set up by the Journal Manager. These automated alerts will trigger in two cases: If a Reviewer has not responded to a request to review a work in X number of days; if a Reviewer has failed to submit a review of a work X number of days after its due date.

Stylesheets, customizable look and feel per publication:

The Journal Manager controls the look and feel of each individual journal via stylesheets and custom HTML header and footer files.

Versioning:

Archiving:

The application provides two types of internal archiving: It

provides a submission archive which maintains copies of records for all submissions, accepted or declined; and it provides a journal archive, preserving the structure, layout, and content of all published journal issues. The application supports external archiving via cross-institutional LOCKSS (Lots of Copies Keep Stuff Safe) archives.

Access, formats, and electronic commerce functions

Accessibility of system:

Accessibility of document output:

Internationalization support:

The backend database must support UTF-8 (Unicode) encoding in order for special characters to be stored, retrieved, and then displayed properly. The en_US locale is installed by default. Locale files for the following also ship with the default configuration and can be activated after initial installation: es_ES; fr_CA; it_IT; pt_BR; ru_RU; tr_TR. In addition to these, locale files for the following languages are currently under development: Arabic; Catalan; Chinese; Croatian; Farsi; Hindi; Norwegian; Thai; Vietnamese.

Output in multiple document formats:

Document formats supported:

Browser plug-in requirements:

Browser plugins are required to view certain document formats, e.g., PDF files. No browser plugins are required to use the application itself.

Usability notes:

Citation linking:

The application provides a wide array of "Reading Tools", including linkages between citations. These tools provide internal links to such things as: Abstract; About the Author information; a formatted bibliographic citation in a specified format; display of author-submitted metatdata; links to author-submitted files that accompany the publication; a link to a formatted Print Version of the publication; ability to click on terms in the text and have

them looked up automatically in an external dictionary utility; a Notify a Colleague function for quick email; an Email the Author function; and a small utility that allows a reader to add comments to the publication. Each of these items can be activated/deactivated on a per journal basis by the Journal Manager.

OpenURL resolver:

RSS feed:

Digital rights management:

Full-text search and retrieval:

Full-text indexing is supported for the following file formats: Text; RTF; Microsoft Word; PDF; Postscript.

Federated searching:

Authentication mechanisms:

Authentication can occur against either the backend database or against an external LDAP server. The plugin for LDAP authentication is provided with the default OJS software package.

Subscription services:

There is an entire administrative module to manage subscription services per individual journal that can be activated by the Journal Manager. Such subscription-related attributes as Subscription Type (e.g., individual or institutional); Subscription Policies; Subscription Expiry Reminders; and Delayed Open Access for Subscription Journals are included. Journal Managers are provided an administrative interface for created subscriptions. This interface includes such things as Subscription Type; start and end dates; Membership requirements of the subscribing party; Domain, if access to subscribed publications are to be restricted by domain; and IP ranges, if access to subscribed publications are to be restricted by IP range.

Electronic commerce functions:

Context-sensitive Help support:

Context-sensitive, pop-up Help files are liberally provided throughout the application.

Summary data

Strengths:

> Easy installation. Platform independent. Excellent and comprehensive documentation. Well-implemented plugin support. Well-thought-out workflow and administrative roles. Solid support for internationalization via UTF-8 encoding and locale files.

Weaknesses:

> It would be nice if an authentication plugin would be provided to allow Shibboleth authentication as well as authentication against other single sign-on utilities, e.g., CAS; WebAuth; SiteMinder.

Figure Eight

Topaz

Institutional affiliation and other indicators of the viability of the open-source project

Name of system:

Topaz

Current version of system:

0.6

Tested version of system:

URL of project homepage:

http://topazproject.org/

Institutional affiliation:

Independent non-profit.

Age of project:

Nov 2005

Notes on long-term viability of project:

Degree of deployment:

Public Library of Science as only client right now. Topaz was commissioned by PLOS. "Not quite ready yet."

Type of open-source license:

Apache 2.0

Licensing notes:

Other documentation (Webliography):

Technical requirements, maintenance, scalability, and documented APIs

Local install or ASP?:

Local install

Operating system requirements:

Any OS supporting Java. Only tested on Linux. RPM packages.

Hardware requirements:

Mulgara RDF DB tier should be on 64-bit machine.

Application server requirements:

Tomcat 5.5 (anything Fedora runs on).

Web server requirements:

Apache or IIS (Apache preferred).

Primary programming language:

Java

Auxiliary programming language:

Groovy; XSLT

Application framework:

WebWorks (Struts 2.0); Watermark templating language; Dojo AJAX framework

Database server requirements:

Mulgara (for metadata); Fedora (for articles/content)

Other software requirements:

Image conversion library, similar to ImageMagik

Required skills:

"Very good developer required." Could install from RPM.

Internal backup and restore functions:

Scalability: Application:

Scalability: Data:

API: Code extensibility:

API: Batch ingest:

Yes. Copy files to a specified directory. Files appear from

within app. Pick and choose and Ingest.

API: Batch ingest formats:

Zip file comprising article, images, etc. Packaged in specified format, similar to PubMed format.

API: Batch export:

No.

API: Batch export formats:

API: Support for JSR 170:

API: Support for OAI harvesting:

Yes. Via Fedora.

API: Support for eduSource Communication Layer (ECL):

API: Support for other Web services:

Security notes:

Submission, peer review management, and administrative functions

Support for multiple, discrete publications:

Not right now. Adding in future release. "The same article can belong to multiple journals."

Multiple administrative roles:

Admin and regular User.

Administrative roles configurable:

No.

Submission into system initiated by authors:

Author can submit directly into ingestion directory via FTP.

Metadata fields configurable:

Multiple ingestion applications, each with its own idiosyncratic metadata schema, can be configured.

Editorial workflow configurable per publication:

Not really. Article ingested, Admin approves. One step

workflow.

Automated email alerts to authors:

Yes.

Automated email alerts to editors:

No

Automated email alerts to reviewers:

No

Stylesheets, customizable look and feel per publication:

Yes, via the Watermark templating engine. Only works across entire application. Next version will have "skin types".

Versioning:

Yes. Linking between versions of articles.

Archiving:

All data written into Mulgara is also being logged in a transaction log. Able to rollback application data in case of corruption.

Access, formats, and electronic commerce functions

Accessibility of system:

Accessibility of document output:

Internationalization support:

Unicode compliant, yet no language packs as of yet.

Output in multiple document formats:

No.

Document formats supported:

All submissions must be in NLM DTD 2.0+ format.

Browser plug-in requirements:

No.

Usability notes:

Citation linking:

OpenURL resolver:

RSS feed:

Yes.

Digital rights management:

Full-text search and retrieval:

Lucene

Federated searching:

None.

Authentication mechanisms:

Single signon capability, against CAS.

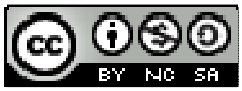
Subscription services:

Email subscription.

Electronic commerce functions:

Context-sensitive Help support:

No.



Creative Commons Attribution-Noncommercial-Share Alike 3.0 United States

License

THE WORK (AS DEFINED BELOW) IS PROVIDED UNDER THE TERMS OF THIS CREATIVE COMMONS PUBLIC LICENSE ("CCPL" OR "LICENSE"). THE WORK IS PROTECTED BY COPYRIGHT AND/OR OTHER APPLICABLE LAW. ANY USE OF THE WORK OTHER THAN AS AUTHORIZED UNDER THIS LICENSE OR COPYRIGHT LAW IS PROHIBITED.

BY EXERCISING ANY RIGHTS TO THE WORK PROVIDED HERE, YOU ACCEPT AND AGREE TO BE BOUND BY THE TERMS OF THIS LICENSE. TO THE EXTENT THIS LICENSE MAY BE CONSIDERED TO BE A CONTRACT, THE LICENSOR GRANTS YOU THE RIGHTS CONTAINED HERE IN CONSIDERATION OF YOUR ACCEPTANCE OF SUCH TERMS AND CONDITIONS.

1. Definitions

a. **"Collective Work"** means a work, such as a periodical issue, anthology or encyclopedia, in which the Work in its entirety in unmodified form, along with one or more other contributions, constituting separate and independent works in themselves, are assembled into a collective whole. A work that constitutes a Collective Work will not be considered a Derivative Work (as defined below) for the purposes of this License.

b. **"Derivative Work"** means a work based upon the Work or upon the Work and other pre-existing works, such as a translation, musical arrangement, dramatization, fictionalization, motion picture version, sound recording, art reproduction, abridgment, condensation, or any other form in which the Work may be recast, transformed, or adapted, except that a work that constitutes a Collective Work will not be considered a Derivative Work for the purpose of this License. For the avoidance of doubt, where the Work is a musical composition or sound recording, the synchronization of the Work in timed-relation with a moving image ("synching") will be considered a Derivative Work for the purpose of this License.

c. **"Licensor"** means the individual, individuals, entity or entities that offer(s) the Work under the terms of this License.

d. **"Original Author"** means the individual, individuals, entity or entities who created the Work.

e. **"Work"** means the copyrightable work of authorship offered under the terms of this License.

f. **"You"** means an individual or entity exercising rights under this License who has not previously violated the terms of this License with respect to the Work, or who has received express permission from the Licensor to exercise rights under this License despite a previous violation.

g. **"License Elements"** means the following high-level license attributes as selected by Licensor and indicated in the title of this License: Attribution, Noncommercial, ShareAlike.

2. Fair Use Rights. Nothing in this license is intended to reduce, limit, or restrict any rights arising from fair use, first sale or other limitations on the exclusive rights of the copyright owner under copyright law or other applicable laws.

3. License Grant. Subject to the terms and conditions of this License, Licensor hereby grants You a worldwide, royalty-free, non-exclusive, perpetual (for the duration of the applicable copyright) license to exercise the rights in the Work as stated below:

a. to reproduce the Work, to incorporate the Work into one or more Collective Works, and to reproduce the Work as incorporated in the Collective Works;

b. to create and reproduce Derivative Works provided that any such Derivative Work, including any translation in any medium, takes reasonable steps to clearly label, demarcate or otherwise identify that changes were made to the original Work. For example, a translation could be marked "The original work was translated from English to Spanish," or a modification could indicate "The original work has been modified.";

c. to distribute copies or phonorecords of, display publicly, perform publicly, and perform publicly by means of a digital audio transmission the Work including as incorporated in Collective Works;

d. to distribute copies or phonorecords of, display publicly, perform publicly, and perform publicly by means of a digital audio transmission Derivative Works;

The above rights may be exercised in all media and formats whether now known or hereafter devised. The above rights include the right to make such modifications as are technically necessary to exercise the rights in other media and formats. All rights not expressly granted by Licensor are hereby reserved, including but not limited to the rights set forth in Sections 4(e) and 4(f).

4. Restrictions. The license granted in Section 3 above is expressly made subject to and limited by the following restrictions:

a. You may distribute, publicly display, publicly perform, or publicly digitally perform the Work only under the terms of this License, and You must include a copy of, or the Uniform Resource Identifier for, this License with every copy or phonorecord of the Work You distribute, publicly display, publicly perform, or publicly digitally perform. You may not offer or impose any terms on the Work that restrict the terms of this License or the ability of a recipient of the Work to exercise the rights granted to that recipient under the terms of the License. You may not sublicense the Work. You must keep intact all notices that refer to this License and to the disclaimer of warranties. When You distribute, publicly display, publicly perform, or publicly digitally perform the Work, You may not impose any technological measures on the Work that restrict the ability of a recipient of the Work from You to exercise the rights granted to that recipient under the terms of the License. This Section 4(a) applies to the Work as incorporated in a Collective Work, but this does not require the Collective Work apart from the Work itself to be made subject to the terms of this License. If You create a Collective Work, upon notice from any Licensor You must, to the extent practicable, remove from the Collective Work any credit as required by Section 4(d), as requested. If You create a Derivative Work, upon notice from any Licensor You must, to the extent practicable, remove from the Derivative Work any credit as required by Section 4(d), as requested.

b. You may distribute, publicly display, publicly perform, or publicly digitally perform a Derivative Work only under: (i) the terms of this License; (ii) a later version of this License with the same License Elements as this License; or, (iii) either the unported Creative Commons license or a Creative Commons license for another jurisdiction (either this or a later license version) that contains the same License Elements as this License (e.g. Attribution-NonCommercial-ShareAlike 3.0 (Unported)) ("the Applicable License"). You must include a copy of, or the Uniform Resource Identifier for, the Applicable License with every copy or phonorecord of each Derivative Work You distribute, publicly display, publicly perform, or publicly digitally perform. You may not offer or impose any terms on the Derivative Works that restrict the terms of the Applicable License or the ability of a recipient of the Work to exercise the rights granted to that recipient under the terms of the Applicable License. You must keep intact all notices that refer to the Applicable License and to the disclaimer of warranties. When You distribute, publicly display, publicly perform, or publicly digitally perform the Derivative Work, You may not impose any technological measures on the Derivative Work that restrict the ability of a recipient of the Derivative Work from You to exercise the rights granted to that recipient under the terms of the Applicable License. This Section 4(b) applies to the Derivative Work as incorporated in a Collective Work, but this does not require the Collective Work apart from the Derivative Work itself to be made subject to the terms of the Applicable License.

c. You may not exercise any of the rights granted to You in Section 3 above in any manner that is primarily intended for or directed toward commercial advantage or private monetary compensation. The exchange of the Work for other copyrighted

works by means of digital file-sharing or otherwise shall not be considered to be intended for or directed toward commercial advantage or private monetary compensation, provided there is no payment of any monetary compensation in connection with the exchange of copyrighted works.

d. If You distribute, publicly display, publicly perform, or publicly digitally perform the Work (as defined in Section 1 above) or any Derivative Works (as defined in Section 1 above) or Collective Works (as defined in Section 1 above), You must, unless a request has been made pursuant to Section 4(a), keep intact all copyright notices for the Work and provide, reasonable to the medium or means You are utilizing: (i) the name of the Original Author (or pseudonym, if applicable) if supplied, and/or (ii) if the Original Author and/or Licensor designate another party or parties (e.g. a sponsor institute, publishing entity, journal) for attribution ("Attribution Parties") in Licensor's copyright notice, terms of service or by other reasonable means, the name of such party or parties; the title of the Work if supplied; to the extent reasonably practicable, the Uniform Resource Identifier, if any, that Licensor specifies to be associated with the Work, unless such URI does not refer to the copyright notice or licensing information for the Work; and, consistent with Section 3(b) in the case of a Derivative Work, a credit identifying the use of the Work in the Derivative Work (e.g., "French translation of the Work by Original Author," or "Screenplay based on original Work by Original Author"). The credit required by this Section 4(d) may be implemented in any reasonable manner; provided, however, that in the case of a Derivative Work or Collective Work, at a minimum such credit will appear, if a credit for all contributing authors of the Derivative Work or Collective Work appears, then as part of these credits and in a manner at least as prominent as the credits for the other contributing authors. For the avoidance of doubt, You may only use the credit required by this Section for the purpose of attribution in the manner set out above and, by exercising Your rights under this License, You may not implicitly or explicitly assert or imply any connection with, sponsorship or endorsement by the Original Author, Licensor and/or Attribution Parties, as appropriate, of You or Your use of the Work, without the separate, express prior written permission of the Original Author, Licensor and/or Attribution Parties.

e. For the avoidance of doubt, where the Work is a musical composition:

    i. **Performance Royalties Under Blanket Licenses.** Licensor reserves the exclusive right to collect whether individually or, in the event that Licensor is a member of a performance rights society (e.g. ASCAP, BMI, SESAC), via that society, royalties for the public performance or public digital performance (e.g. webcast) of the Work if that performance is primarily intended for or directed toward commercial advantage or private monetary compensation.

    ii. **Mechanical Rights and Statutory Royalties.** Licensor reserves the exclusive right to collect, whether individually or via a music rights agency or designated agent (e.g. Harry Fox Agency), royalties for any phonorecord You create from the Work ("cover version") and distribute,

subject to the compulsory license created by 17 USC Section 115 of the US Copyright Act (or the equivalent in other jurisdictions), if Your distribution of such cover version is primarily intended for or directed toward commercial advantage or private monetary compensation.

f. **Webcasting Rights and Statutory Royalties.** For the avoidance of doubt, where the Work is a sound recording, Licensor reserves the exclusive right to collect, whether individually or via a performance-rights society (e.g. SoundExchange), royalties for the public digital performance (e.g. webcast) of the Work, subject to the compulsory license created by 17 USC Section 114 of the US Copyright Act (or the equivalent in other jurisdictions), if Your public digital performance is primarily intended for or directed toward commercial advantage or private monetary compensation.

5. Representations, Warranties and Disclaimer

UNLESS OTHERWISE MUTUALLY AGREED TO BY THE PARTIES IN WRITING, LICENSOR OFFERS THE WORK AS-IS AND ONLY TO THE EXTENT OF ANY RIGHTS HELD IN THE LICENSED WORK BY THE LICENSOR. THE LICENSOR MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND CONCERNING THE WORK, EXPRESS, IMPLIED, STATUTORY OR OTHERWISE, INCLUDING, WITHOUT LIMITATION, WARRANTIES OF TITLE, MARKETABILITY, MERCHANTIBILITY, FITNESS FOR A PARTICULAR PURPOSE, NONINFRINGEMENT, OR THE ABSENCE OF LATENT OR OTHER DEFECTS, ACCURACY, OR THE PRESENCE OF ABSENCE OF ERRORS, WHETHER OR NOT DISCOVERABLE. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES, SO SUCH EXCLUSION MAY NOT APPLY TO YOU.

6. Limitation on Liability. EXCEPT TO THE EXTENT REQUIRED BY APPLICABLE LAW, IN NO EVENT WILL LICENSOR BE LIABLE TO YOU ON ANY LEGAL THEORY FOR ANY SPECIAL, INCIDENTAL, CONSEQUENTIAL, PUNITIVE OR EXEMPLARY DAMAGES ARISING OUT OF THIS LICENSE OR THE USE OF THE WORK, EVEN IF LICENSOR HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

7. Termination

a. This License and the rights granted hereunder will terminate automatically upon any breach by You of the terms of this License. Individuals or entities who have received Derivative Works (as defined in Section 1 above) or Collective Works (as defined in Section 1 above) from You under this License, however, will not have their licenses terminated provided such individuals or entities remain in full compliance with those licenses. Sections 1, 2, 5, 6, 7, and 8 will survive any termination of this License.

b. Subject to the above terms and conditions, the license granted here is perpetual (for the duration of the applicable copyright in the Work). Notwithstanding the

above, Licensor reserves the right to release the Work under different license terms or to stop distributing the Work at any time; provided, however that any such election will not serve to withdraw this License (or any other license that has been, or is required to be, granted under the terms of this License), and this License will continue in full force and effect unless terminated as stated above.

8. Miscellaneous

a. Each time You distribute or publicly digitally perform the Work (as defined in Section 1 above) or a Collective Work (as defined in Section 1 above), the Licensor offers to the recipient a license to the Work on the same terms and conditions as the license granted to You under this License.

b. Each time You distribute or publicly digitally perform a Derivative Work, Licensor offers to the recipient a license to the original Work on the same terms and conditions as the license granted to You under this License.

c. If any provision of this License is invalid or unenforceable under applicable law, it shall not affect the validity or enforceability of the remainder of the terms of this License, and without further action by the parties to this agreement, such provision shall be reformed to the minimum extent necessary to make such provision valid and enforceable.

d. No term or provision of this License shall be deemed waived and no breach consented to unless such waiver or consent shall be in writing and signed by the party to be charged with such waiver or consent.

e. This License constitutes the entire agreement between the parties with respect to the Work licensed here. There are no understandings, agreements or representations with respect to the Work not specified here. Licensor shall not be bound by any additional provisions that may appear in any communication from You. This License may not be modified without the mutual written agreement of the Licensor and You.