CHAPTER

DIGITAL LIBRARY MANAGEMENT SYSTEMS

6

INTRODUCTION

The design of systems for management and delivery of digital library content is an interdisciplinary area where research on digital libraries intersects with software development, database management, information retrieval, and human-computer interaction. Digital library management systems (DLMS) share some similarities with web content management systems but are also different because of the required support for digital library standards, especially in regard to information organization and interoperability. DLMS represent a specialized category of software systems that integrate functionality for building, managing, storing, providing access to, and preserving digital objects and collections. They are part of a broader category of digital asset management systems that are used in practice for acquisition, indexing, storage, management, preservation, and delivery of digital objects. In a distributed digital library environment, DLMS also provide platforms for aggregating digital content and metadata.

The concept of DLMS is used here according to the definition proposed in the DELOS Manifesto as "a generic software system that provides the appropriate software infrastructure both to produce and administer a Digital Library System incorporating the suite of functionality considered foundational for Digital Libraries and to integrate additional software offering more refined, specialized, or advanced functionality" (Candela et al., 2007b). The DELOS Manifesto makes a distinction between a digital library system (DLS) and a DLMS. A DLS offers functionality for a particular digital library, including support for end user interactions. A DLMS provides a platform for producing and administering digital collections and services by ensuring essential functionality and incorporating additional software components for more refined and advanced features (Tramboo et al., 2012). DLMS enable an instantiation and management of digital collections and services that become part of a centralized or distributed DLS.

The design of DLMS has been an area of active development since the early days of digital libraries and has included efforts to provide conceptual models as well as platform solutions. Many of the early systems were custom-built, designed for single projects in order to meet the needs of a particular community (Suleman and Fox, 2001). The late 1990s saw the development of the first architectural models for repositories and the emergence of the dedicated content management systems for cultural heritage digital collections. Greenstone was released as open source software and has been widely adopted throughout the world. CONTENTdm became a popular choice in the category of proprietary software among the US public and academic libraries. Early 2000s marked the construction of the first digital repositories for scholarly publication with EPrints developed at the University of Southampton, UK, and DSpace and Fedora released in the United States. Currently, there are number of open source systems and commercial software packages available for building digital library systems and meeting the needs and requirements of specific communities. Many organizations, however, especially those building

Discover Digital Libraries. http://dx.doi.org/10.1016/B978-0-12-417112-1.00006-5

large-scale systems such as Europeana, HathiTrust, or National Science Digital Library (NSDL) have developed custom platforms (Concordia et al., 2010; Henry, 2012). In recent years, libraries and archives have been migrating from the first generation of open source or proprietary software, such as DSpace or CONTENTdm toward more robust and scalable open source solutions (Gilbert and Mobley, 2013; Stein and Thompson, 2015). The new generation of DLMS is comprised of several open source technologies and often integrate Fedora with other customizable platforms, such as Hydra and Islandora (Awre and Cramer, 2012; Cramer and Kott, 2010; Moses and Stapelfeldt, 2013).

The terminology of DLMS and DLS is used according to the DELOS Manifesto (Candela et al., 2007b). However, it is important to note that other terms are also used for DLMS, including digital content management systems (Han, 2004), digital collection management systems (Zhang and Gourley, 2009), repository platforms (Henry, 2012), or digital asset management systems (Breeding, 2013, 2015; Kaplan, 2009; Stein and Thompson, 2015). The term "digital asset management system" is being used widely and often interchangeably with DLMS. In practice, its usage is broad and often encompasses a wide range of software, including digital collection systems (e.g., CONTENTdm, Omeka), repositories (e.g., DSpace, Fedora, Digital Commons), digital preservation systems (e.g., Rosetta, Preservica), discovery layers (e.g., Blacklight), or even databases, such as FileMaker Pro (Stein and Thompson, 2015).

This chapter presents both theoretical and practical perspectives on developing and implementing DLMS. The focus of the review is primarily on software systems designated for building and managing digital objects and collections and on multifunctional open source repository systems. Digital repositories designed for providing long-term preservation of digital content are covered in the chapter on digital preservation (Chapter 9). This distinction is not always clear as increasingly new systems are designed to meet both preservation and access needs. The second generation of DLMS addresses the challenge of maintaining separate systems for access and preservation and offers multipurpose repository systems (Awre, 2012; Awre and Cramer, 2012; Cramer and Kott, 2010).

DESIGN AND ARCHITECTURE OF DIGITAL LIBRARY SYSTEMS

The complexity of DLS as information systems is widely acknowledged (Candela et al., 2007b; Chowdhury and Chowdhury, 2003; Concordia et al., 2010; Henry, 2012). DLS serve diverse groups of users from scientific, educational, archives, and museum communities. They provide access to scholarly publications, data sets, archival documents, and cultural heritage objects in multiple formats and languages that are described by a variety of metadata standards. Obviously, there is no single, universal software system that could meet the needs of all user communities and support the variety of data types and metadata schemas. In addition to user requirements and functionality, the design of a DLMS has to address the technical aspects of system reliability, scalability, and sustainability. Most digital libraries are created independently by content holders in research, library, archives, and museum communities using a range of standards and software solutions. Interoperability has become one of the most important issues in the development of digital libraries. The goal of interoperability is to enable the exchange of data between independent digital libraries and to provide services for easier discovery and interaction with digital library resources in the network environment (Arms et al., 2002).

The initial systems were often built "from scratch" or incorporated existing software components but offered limited modularity and interoperability. Early DLMS were standalone systems and had typical features of the system-centered design, which meant they were difficult to install, customize, or configure (Ioannidis et al., 2005). Some researchers argue that the lack of common definitions and conceptual frameworks led to such ad hoc system development and hindered interoperability (Candela et al., 2010; Gonçalves et al., 2008).

The large-scale digital libraries that have emerged in recent years pose new challenges for interoperability and system reliability, scalability, and sustainability (Henry, 2012). The aggregation of content and services can take place on consortial, regional, national, or international levels. These large-scale digital library systems are built either as centralized aggregators of content, metadata, and services or use a distributed network of content and service hubs with a service layer to facilitate access to harvested metadata and links to objects. HathiTrust is an example of a centralized model, while Europeana, the Digital Library of America, and the National Science Digital Library represent large-scale distributed systems.

The first two decades of digital library research efforts have concentrated on defining the components of digital library systems, delineating the relationships among them, and developing conceptual models that would enable interoperation between individual DLS. This section provides an overview of the research surrounding digital library architectural models, reviews functionality and other system requirements, and discusses interoperability approaches.

ARCHITECTURAL MODELS

The concept of architecture in the context of digital library systems refers to "a consistent and comprehensive set of software components necessary for a DLS and the interaction between them" (Candela et al., 2007a, p. 23). An architectural model serves as a starting point and a conceptual framework for designing flexible and interoperable systems. It is an abstract framework for identifying components and describing the relationships among them, usually independent of specific standards, technology, and implementations. The goal of an architectural model is to provide a set of common basic elements and to serve as a blueprint for an integration of subelements supporting specialized functionality (Candela et al., 2007a).

The research community has proposed several models of digital library architecture of various levels of complexity. Three core components can be identified across the frameworks, although the names of the components often vary (Candela et al., 2010; Suleman, 2012):

- Data store (also referred to as content files, data repository, or archive)
- Metadata registry (also referred to as metadata catalog, metadata store, or index)
- A set of services (e.g., index, store, manage, copy, authorize, etc.)

The separation of digital content from the structured information describing it (metadata) is a fundamental principle of digital library architectural models. A data store contains digital content files in a variety of formats and structures, such as images, audio, text files, or complex structures consisting of multiple files. A metadata registry includes associated metadata to identify stored content files and to provide information about their properties and context. The two components represent the system's repository and interact with each other through a range of services, such as submit, index, store, manage, copy, authorize, retrieve, import, export, etc. This generic model identifies only the core components and services to assure basic digital library functionality. The core components can have subcomponents that provide specialized functionality. For example, metadata store can include a selection of controlled vocabulary tools and metadata schemas. Models of open, component-based architecture provide a framework for creating and managing large distributed digital libraries.

The foundational research on architecture of digital library systems took place in the mid-1990s. In their seminal work, (Kahn and Wilensky, 2006) propose a reference model of open architecture that envisions an interoperation of multiple repositories and supports distributed digital information services. They define the concepts of digital objects, unique identifiers (handles), and repositories, and specify the method for depositing and accessing digital objects in repositories. Based on the work of Kahn and Wilensky, Arms (1995) outlines general principles and elements of digital library architecture, including the separation of the underlying architecture from the content stored in the library, names and identifiers as building blocks of digital libraries, and the concepts of digital objects and repositories.

Researchers at the Cornell Digital Library Research Group have advanced the concepts of open architecture through the development of a protocol for distributed document libraries (Dienst) and a component-based digital library architecture called CRADDL—Cornell Reference Architecture for Distributed Digital Libraries (Davis and Lagoze, 2000; Lagoze and Fielding, 1998). CRADDL identifies a set of core components of a digital library infrastructure, such as repository, index, naming, collection, and defines the interactions between them (Lagoze and Fielding, 1998). The functionality of CRADDL is defined in terms of five core services:

- The repository service that provides the mechanism for the deposit, storage, and access to digital objects
- The naming service that provides a registry of unique names for identifying digital objects
- The index service that includes information about digital objects or sets and the mechanism for their discovery via query
- The collection service that provides a method of aggregating digital objects and services into meaningful collections
- User interface services or gateways that provide entry points for collection creators and administrators to build and manage collections and for end users to search and access objects in collections

Lagoze and Fielding (1998) highlight the open architecture of this model, and note that other services can be added to enhance the core functions. The authors focus on the collection services, selection criteria, and specifications for administering collections and the dynamic nature of collections that allows for the possibility of a single object to belong to multiple collections.

The introduction of the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) has presented new possibilities for the interoperation among standalone digital library systems and prompted the development of new architectural models. Using OAI-PMH as a foundation, Suleman and Fox (2001) propose a framework for building open digital libraries. The model combines the capabilities of OAI-PMH as a protocol to transfer metadata with the concept of extended services and local open archives as self-contained but interoperable components. Open digital libraries are envisioned as a network of components or building blocks (open archives) with data providers sharing metadata through a union catalog and providing extended services through a common interface, such as search, browse, and recommend. The researchers provide examples of successful implementation of this model, including the Networked Digital Library Theses and Dissertations project at Virginia Tech (Suleman et al., 2003).

The component-oriented approach is proposed for large-scale digital libraries serving more than one community and whose requirements may evolve over time. The Digital Library System Reference Architecture was developed by the DELOS Network of Excellence on Digital Libraries to provide a framework for cooperative and distributed development of European digital library systems. This model presents digital library architecture as a modular and flexible structure of components and layers (Candela et al., 2007a). It identifies the core components and their subsystems; for example, the information component consists of subcomponents of data files (Archive), metadata, and controlled vocabulary tools, such as thesauri and ontology. The layers organize the components according to functionality. This model assumes easy design through component selection and replacement, sharing and reuse of components in different contexts, distributed installation and maintenance, and easy support for component modification or addition. The proposed reference model, although designed specifically for the digital library community, utilizes many concepts from computing and demonstrates that digital library system design is truly an interdisciplinary endeavor.

The design of digital library systems has been informed by architectural models in computing, especially service-oriented architecture (SOA) (Henry, 2012; Suleman, 2005, 2012). SOA is a relatively new model of software construction where tasks and services are subdivided and performed by independent components that interact with each other through standard interfaces and communication protocols (Suleman, 2005). SOA supports reusability, subtraction, and substitution of components and services and offers a potential for the development of evolving and expanding digital library systems. Suleman (2005), however, notes that, as of 2005, very few open source systems, including Greenstone, EPrints, or DSpace, applied SOA design principles.

Fedora (Flexible Extensible Digital Object Repository Architecture) represents a reference model for DLMS that is based on SOA. Fedora was originally developed by researchers at Cornell University as an architecture framework for storing, managing, and accessing digital content (Payette and Lagoze, 1998; Payette et al., 1999). Fedora's open architectural model was inspired by Kahn and Wilensky's work on digital objects. The concept of a digital object is a fundamental block of Fedora's architecture. Its object model supports many kinds of digital content including documents, images, e-books, multimedia learning objects, datasets, and other complex objects (Lagoze et al., 2006; Payette and Lagoze, 1998). In addition to defining digital objects, Fedora also provides a theoretical foundation for repository architecture focusing on extensibility and interoperability. Payette and Lagoze (1998) outline the key features of the architecture:

- Support for heterogeneous data types
- Accommodation of new types as they emerge
- Aggregation of mixed, possibly distributed, data into complex objects
- The ability to specify multiple content disseminations of these objects
- The ability to associate rights management schemes with these disseminations (Payette and Lagoze, 1998)

The Fedora reference model also identifies a set of core functions, including (1) repository services for depositing, storing, and accessing digital objects; (2) index services for discovering digital objects; (3) collection services that provide the means of aggregating sets of digital objects and services into meaningful collections; (4) naming services that register and resolve globally unique, persistent names for digital objects; and (5) user interface services that provide a human gateway into the other services

(Payette and Lagoze, 1998). Fedora offers a highly flexible architecture of relationships among digital objects and an ability to extend its components and integrate new services.

The prototype digital library system using the Fedora architecture was built in 2000 at the University of Virginia Library (Staples et al., 2003). The first version of Fedora open source software (Fedora 1.0) was released to the public in 2003. Since then, Fedora has been adopted by more than three hundred institutions worldwide (Fedora, 2015). Because of its open and flexible architecture, Fedora is used as a framework for a variety of digital library systems, including digital collections, institutional repositories, digital preservation systems, and large-scale distributed digital library networks. Fedora provides a foundation for the new generation of multifunctional platforms, including those built with Hydra or Islandora (Awre and Cramer, 2012; Cramer and Kott, 2010; Jettka and Stein, 2014; Kent, 2014; Moses and Stapelfeldt, 2013). Fedora, as an open source software, is supported by an active community of users under the stewardship of DuraSpace, a nonprofit organization (Fedora, 2015). Fedora 4.0.0 was released in 2014. The implementation of Fedora software is discussed further in the section on open source repository systems.

FUNCTIONALITY AND OTHER SYSTEM REQUIREMENTS

Designing DLMS is an extremely challenging task, as it requires the integration of architectural models, technologies, and standards. It begins with a conceptual model but also involves a range of technologies, standards, and applications. It is a combination of those elements that contributes to a flexible and usable system design and provides the required functionality for creating, managing, and using a digital library.

Functionality refers to system's capabilities in building and managing digital libraries and providing end user support. Functionality is expressed in terms of services and is often divided into fundamental and value-added services (Gonçalves et al., 2008). The set of core functions relates to object- and collection-building, managing, disseminating, and/or preservation capabilities (Gonçalves et al., 2008; Zhang and Gourley, 2009). Researchers sometimes distinguish preservation services from those focused on content creation and management (Gonçalves et al., 2008), but since many systems integrate preservation and access services, the following list includes preservation in the core functions:

- Creation of digital objects and collections, which includes ingesting and/or processing of digitally-born or digitized materials and associated metadata records; creation of collections of objects based on predefined selection criteria
- Management, which involves adding, modifying, and deleting objects; management of user rights and permissions
- Access, which includes indexing, searching, browsing, and harvesting services plus presentation of objects and collections through a web interface and tools for user interaction
- Preservation, which includes services to store and manage digital objects and archival master files

Although preservation is recognized as a core function, it needs to be noted that many systems, especially those focused on building digital heritage collections, separate access and preservation activities. In practice, preservation is often managed by an archival information system, such as a dedicated digital preservation repository, or "dark archive," that may or may not be integrated with a primary access DLMS. An example of an integrated DLMS and archival information system is OCLC CONTENTdm, where licensed users have an option of depositing their digital master files in the OCLC

Digital Archive. Many institutional repositories built with open source systems, such as DSpace or Fedora, represent hybrid, multifunctional environments with various levels of access and preservation functions. HathiTrust is an example of a custom-built system that provides both access and preservation services to its partner institutions.

In addition to core functions, DLMS can provide a wide range of advanced and/or value-added services. Some of the services, although not included in the core, are critical to building digital libraries and maintaining them over time. For example, Export service is a function of the system that provides a means of retrieving objects and/or metadata and depositing them outside of the system. Export function becomes critical when institutions decide to migrate to a different platform. There is no comprehensive list of digital library services, although some efforts have been undertaken to define services based on theoretical models. Using the 5S framework, Gonçalves et al. (2008) provide a list of services and their informal definitions, from annotating to recommending, translating, and visualizing.

Value-added services can include additional functionality, such as page turners, high-resolution image viewers, integration of thesauri, visualization tools, geo-tagging, social tagging, etc. Tools for visualization of search results are particularly useful in large-scale digital libraries. Europeana and the DPLA offer timeline and map views of their results. The expansion of functionality is enabled by modular architecture and the use of open application programming interfaces (APIs) or plug-ins. APIs provide powerful tools for integrating the components of DLMS (Zhang and Gourley, 2009). The DPLA has also opened its data and API to external software developers, researchers, and others to create novel environments for learning, tools for discovery, and engaging apps. Plug-ins are software components that add new functionality to an application. The open source Omeka system, for example, provides an array of plug-ins to expand its basic functionality in creating digital collections and exhibits.

DLMS serve two primary groups of users: (1) digital library content providers, designers, and administrators, who produce and manage digital library collections and systems, and (2) digital library end users, who search, browse, and interact with digital objects and collections. DLMS need to support complex tasks and workflows of both groups through two separate user interfaces. Fig. 6.1 demonstrates an administrative interface in the open source system, Omeka, where administrators can configure the site, add and edit items, create metadata records, group items in collections, and customize the end user interface. Fig. 6.2 provides an example of an end user interface for a digital collection created in the Omeka system. The collection of documents from the Laura Hershey Collection, a disability rights activist, was built by a group of library and information science graduate students at the University of Denver. The simple interface allows end users not only to search and browse items in the digital collection but also to share objects through a variety of social media and to contribute their own materials to the collection. As demonstrated in the Omeka example, open source software enables relatively easy development of new services to enhance the system's functionality while simultaneously leveraging the contributions of the user community.

Functionality is a key requirement of DLMS, as it supports content creation, management, and user access. In addition, there is also a range of related system features and capabilities that impact system performance, user satisfaction, maintenance, and the ability to interact with other digital library systems. The most important capabilities are outlined as:

- *Extensibility* relates to a system's capabilities of adding new components and services to accommodate the continuous expansion of digital libraries and to incorporate new technological solutions. Models of open and modular architecture that are being gradually adopted in digital library environments support system flexibility and extensibility (Suleman, 2012; Yeh et al., 2009).

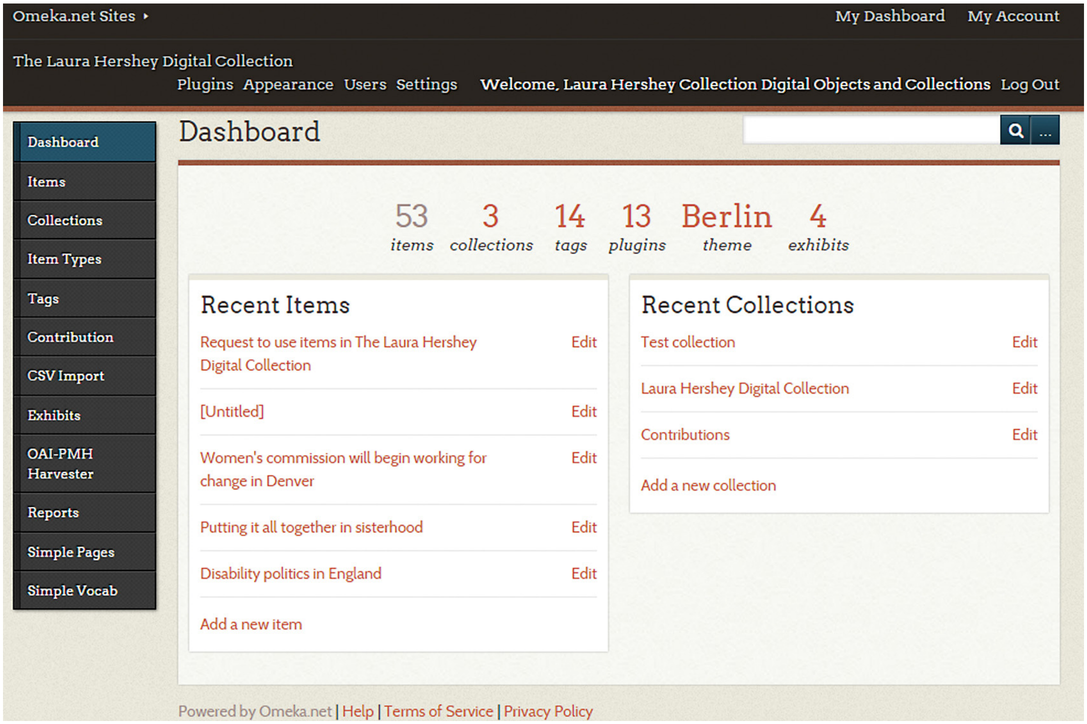

FIGURE 6.1 Administrative Interface in Omeka

- *Reliability* is a measure of system performance and relates to hardware and software failures and errors. Reliability of servers can be addressed by redundant configurations (Henry, 2012; Zhang and Gourley, 2009). Reliability decreases in a distributed large-scale digital library environment where it is more challenging to monitor systems of local content providers. Centralized large-scale digital libraries, such as HathiTrust or World Digital Library, have more control over content and the availability of the overall system.
- *Scalability* refers to a system's ability to accommodate (1) the expansion of content in the terms of the growing number and/or size of objects and collections and (2) an increasing number of users. Zhang and Gourley (2009) point out that scalability refers to the entire system: hardware, network, and software. In large-scale distributed digital library systems, scalability needs to address the growing number of content and service providers joining the network.
- *Sustainability* refers to a system's ability to provide robust management of collections and services over time. Henry (2012) notes that one needs to think of system sustainability not only in terms of hardware and software but also of the entire organization responsible for creating, managing, and maintaining the DLS over time.

Reliability, scalability, and sustainability relate to overall system performance and represent important factors influencing users' perceptions and acceptance of the system. Interoperability is a less visible but nonetheless important feature, as it offers the potential for independent digital libraries to cooperate and share content with a wider audience.

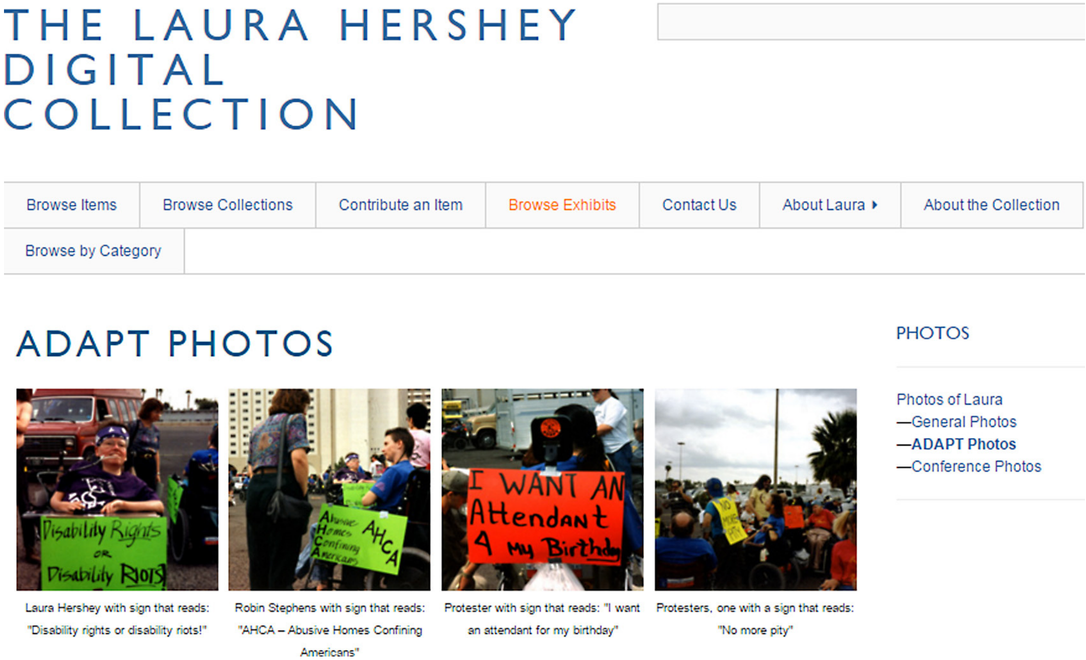

FIGURE 6.2 End user Interface for a Digital Collection Built in the Omeka System

The collection is available at: http://laurahershey.omeka.net/.

INTEROPERABILITY

Interoperability refers to a system's ability to communicate with other digital library systems using standard protocols in order to exchange data. Interoperability has many aspects including uniform naming, metadata formats, document models, and access protocols (Lagoze and Van de Sompel, 2001). It has been recognized as a critical problem and a fundamental challenge since the early days of digital library development (Arms, 2000; Paepcke et al., 1998). The goal of interoperability is to build a set of services for users "from components that are technically different and managed by different organizations" (Arms et al., 2002). The challenge lies in heterogeneous content, multiple data formats, different protocols, and the variety of metadata schemas used by individual organizations. Establishing an interoperability framework is not only a technical but also organizational issue, as it requires a variety of content providers to cooperate and agree on common standards.

Three basic approaches have been identified in the "spectrum of interoperability" with different levels of engagement from content providers (Arms et al., 2002):

- *Federation* provides the strongest form of interoperability, but it also places the highest demands on the participating institutions. It requires that content providers agree that their services will conform to certain specifications. Federation is a well-established form of exchanging data in the library world. Examples of federated services include the sharing of online catalog records using Z39.50 protocol or metasearching of multiple journal databases.

- *Harvesting* represents a less rigorous approach. Participating institutions have to agree to expose and share their data, but they don't have to adopt a formal set of agreements.
- *Gathering* represents the least demanding approach for content providers. Resources openly available on the web are gathered by web crawlers, and no formal agreement may be necessary between organizations holding digital content and a digital library service provider collecting it.

Digital library service providers may select one of the approaches or a combination thereof in an effort to aggregate metadata and/or content from multiple independently operated libraries or other content providers.

Metadata harvesting with the OAI-PMH has become the most widely adopted solution to interoperability in the digital library environment. Metadata harvesting provides a model of interoperability where participating content providers agree to expose and share their metadata. The exposure of metadata allows other organizations to harvest it, aggregate it, and provide access services. Users can search across the body of aggregated metadata and link to digital objects held by original content providers. The transfer of metadata is defined by the OAI-PMH, a protocol that is easy to adopt and implement. The researchers involved in developing the OAI-PMH standard note that this low-barrier approach has contributed to its widespread adoption (Lagoze and Van de Sompel, 2003). The idea of developing a metadata harvesting standard originated in the scientific community, which was interested in a more efficient dissemination of scholarly publications (Lagoze and Van de Sompel, 2001). However, once OAI-PMH was proposed, it has been quickly accepted by digital libraries across domains and by a wide range of communities including cultural heritage organizations.

The OAI-PMH provides an interoperability framework based on metadata harvesting (Open Archives Initiative, 2002). It distinguishes between two classes of participants:

- *Data providers* adopt the OAI technical framework, and agree to open their servers for metadata harvesting.
- *Services providers* harvest metadata by employing the OAI protocol, and use the aggregated metadata as the basis for providing access services to users.

OAI-PMH uses basic Dublin Core as a common element set and requires data providers to expose metadata in that format, which poses a number of challenges for organizations using different metadata schemas. The issues of semantic interoperability and cross-walks are discussed in the chapter on Metadata (Chapter 5). From the perspective of DLMS, it is important to examine the system's support of interoperability and compliance with OAI-PMH. Most of the currently available DLMS from open source software to proprietary systems support OAI-PMH as metadata transfer protocol.

OAI-PMH was introduced in its first version in 2001. The second version, OAI-PMH 2.0, was released in 2002. Open Archives Initiative, a nonprofit organization dedicated to developing and promoting interoperability standards, is responsible for the maintenance of OAI-PMH (Open Archives Initiative, 2015). It also maintains Object Reuse and Exchange (OAI-ORE), a standard for the description and exchange of aggregations of web resources.

OAI-PMH provides a foundation for connecting independently operated digital libraries and the creation of service providers on multiple levels. The following example illustrates the multilayered interconnectivity of content and service providers. The Mountain West Digital Library, a regional aggregator, harvests metadata from multiple digital library systems in the western US and provides an interface for cross-collection searching. Moreover, a regional service provider, like the Mountain West Digital

Library, also exposes the aggregated metadata for harvesting, becoming a de facto data provider for a larger entity like the DPLA. In the decade since its release, OAI-PMH has enabled the formation of large national and international distributed digital library systems, such as the DPLA, Europeana, and NSDL, and the significance of this standard in the evolution of digital libraries cannot be overstated.

CURRENT LANDSCAPE OF DLMS

The exponential growth of digital libraries and their evolution from single, standalone projects to large-scale systems has increased the demands for developing flexible and interoperable DLMS. Two decades of research and development efforts have resulted in a diversified DLMS landscape with multiple open source solutions, proprietary software, and custom-built systems. The new generation of repositories often includes Fedora as a framework and a stack of other open source technologies. The array of choices reflects a variety of perspectives on managing and delivering digital content as well as different user requirements in regard to the functionality and technical support. Large-scale digital library systems present an impressive assemblage of records, but it is worth remembering that the actual digital content building takes place on the ground in a variety of organizations, from large research institutions to academic or public libraries, archives, historical societies, and museums. These organizations have different requirements and traditions of organizing, managing, and preserving content and provide varying levels of technical support. Large research institutions often choose to develop their own platforms, utilizing open architectural models, digital library standards, and a stack of open source technologies. However, smaller institutions that don’t have the support of programmers and technical staff turn to open source or proprietary software packages that are easy to install and manage.

The diversification of DLMS is also reflected in a growing number of systems that provide specialized functionality and are developed specifically to manage certain types of digital content and/or serve different user communities. The distinctions among systems are not always clear, but a number of specialized platforms have emerged from digital repositories in academic communities to systems dedicated to managing digital heritage collections in libraries, archives, and museum settings.

Digital repositories serve primarily as platforms for preserving and providing open access to scientific papers and other forms of scholarly output, but they are also used for hosting digital heritage collections. The Open Access (OA) movement has spurred the growth of systems for open dissemination of scholarly publications including repositories and e-publishing solutions. Many digital repositories have been developed with open source software, including DSpace, EPrints, and Fedora, but there are also hosted licensed systems available in this category, such as Digital Commons, available from bepress. DigiTool provided by Ex Libris, which is an example of proprietary software used for both institutional repositories and digital collections. Open Journal Systems (OJS) represents an open source e-publishing system that provides support not only for depositing scholarly papers but also for their management through peer review and editorial processes (Public Knowledge Project, 2015).

DLMS for building digital collections of cultural heritage materials represent a distinct category because of requirements for managing and presenting heterogeneous multimedia content as well as support for digitization workflows. A wide range of options are available for cultural heritage institutions and include well-established platforms like open source Greenstone and proprietary CONTENTdm or LUNA. There is also a growing number of open source solutions including CollectiveAccess, Collection Space, and Omeka that are used by the members of the LAM (libraries, archives, and museums) community. In addition, open-source repository systems, such as DSpace, Fedora, Hydra, and Islandora are used for hosting digital collections.

The museum community has a different tradition of organizing and presenting materials from libraries or archives and requires dedicated systems to manage its unique workflows and to present digital representations of artifacts through an exhibit function. Large museum organizations often select dedicated proprietary systems that provide capabilities for managing large-scale collections and support for workflows in managing and curating objects. Museum proprietary systems include Proficio, provided by Re:discovery Software and the EMu (Electronic Museum), developed by KE Software.

The current landscape presents many alternatives and types of DLMS with new solutions continuously being added, especially in the open source category. The results of recent surveys demonstrate a wide range of systems being implemented in practice for institutional repositories and digital collections. Andro et al. (2012) compared the features and functionality of ten DLMS used in France and internationally. The authors identified Invenio, Greenstone, DSpace, Omeka, EPrints, and ORI-OAI in the open source category and, for proprietary software, DigitTool, CONTENTdm, and two products used primarily in France: Mnesys and Yoolib (Andro et al., 2012). Moulaison et al. (2015) conducted a nationwide survey among US-based repositories registered with the Directory of Open Access Repositories (OpenDOAR) and found that DSpace was the most common system used for open repositories, followed by Digital Commons (bepress), Fedora, ExLibris DigiTool, Hydra, Islandora, Omeka, and CONTENTdm. Stein and Thompson (2015) surveyed institutions migrating from old digital asset management systems to new platforms. Since the category of digital asset management systems is broad, the list of software identified in this study was more varied. Among the top currently used systems that the survey participants considered abandoning were two proprietary systems, ExLibris DigiTool and CONTENTdm, and one open source system, DSpace. Stein and Thompson (2015) noticed a trend in the migration pattern where institutions were more often than not moving away from proprietary systems towards open source solutions. A desire for more local control was cited by respondents as a primary reason for this migration. Islandora and Hydra were identified as the top choices that institutions were selecting as their new open source platforms.

It is impossible to review all available options within the limits of this chapter. There is also the risk of the information becoming outdated quickly, a risk inherent in any type of discussion about digital technology. The following section compares the benefits and limitations of open source versus proprietary software, and provides a brief review of selected systems. The focus of this review is on software systems used for building and managing digital collections and on multipurpose systems that are used for institutional repositories and digital collections. Platforms in the category of digital repositories that serve primarily preservation functions are discussed in the chapter on digital preservation (Chapter 9). Finally, this chapter concludes with a brief overview of approaches to the selection of DLMS.

OPEN SOURCE VERSUS PROPRIETARY SOFTWARE

Open source software refers to any software that provides free distribution and redistribution as well as access to source code. The Open Source Initiative defines open source software as "software that can be freely used, changed, and shared (in modified or unmodified form) by anyone" (Open Source Initiative, 2015, para. 4). The code source is available under a GNU public license, which allows developers to modify and redistribute it. As Goh et al. (2006) note, open source is different from freeware in that it is freely released but without licenses for modification and redistribution. Shareware, on the other hand, is free only for a limited period of time. The availability of the source code represents a great potential for modifications, improvements, and further software development. Sustainability of open source software, however, requires an active developer community.

Researchers point out a natural affiliation between the open source movement and the library world because of libraries' long history dealing with licensed content, a tradition of sharing and collaboration, and positive perceptions of open source software (Krishnamurthy, 2008; Palmer and Choi, 2014; Payne and Singh, 2010; Rafiq, 2009). Librarians not only select open source software for library applications but also are becoming active participants in its development (Fox, 2006; Payne and Singh, 2010; Samuels and Griffy, 2012). Some of the benefits of open source software for DL applications include:

- Low cost of implementation, which is particularly important because of shrinking library budgets
- Ability to modify software and adapt to meet specific user needs
- Ability to improve functionality of the software
- Institutional autonomy and freedom of commercial licensure
- Support of a large user community

However, open source software has its limitations. A recent study identified some challenges for adopting open source software in libraries, including:

- The need for highly skilled staff that could provide support for the open source system
- Poor documentation
- The need for additional training or expertise
- Substandard development practices (Thacker and Knutson, 2015)

Open source solutions lack the formal technical support and training offered by commercial vendors. Adopting open source software requires a commitment to invest time and resources in learning the software and maintaining it, which can slow down the actual digital library project. Samuels and Griffy (2012) state that the economic benefits of low initial costs can be canceled out by expenses involved in trouble-shooting and learning cumbersome workflows. The authors recommend considering "total cost of ownership," which includes not only the initial investment but also direct and indirect costs throughout the entire software lifecycle. The study, examining the cost of operating institutional repositories, found almost no difference between annual operating expenses for institutions that use open source software and institutions that use proprietary solutions (Burns et al., 2013).

Proprietary software packages offer relatively easy solutions for building and managing digital collections, but this ease-of-use often has a price tag to match. Proprietary DLMS are sold or licensed without access to the software code and with restrictions on how the system may be modified. The license may have provisions for the number of software instances, the size of collections, and/or the number of collections. In addition to the license cost, institutions are also required to pay an annual maintenance fee, which guarantees access to upgrades and documentation. Ease of installation and use, documentation, technical support, system stability, and integrated functionality represent clear benefits of proprietary digital library software. The cost and limited opportunity for customization are the obvious disadvantages.

The decision to adopt open source software or purchase proprietary software for building and managing digital collections is quite complex and requires the consideration of multiple factors. Institutions need to weigh the benefits and costs, evaluate and test potential candidates, and select the solution that meets the organizational needs and user requirements. As mentioned before, there is a general trend for libraries to select open source solutions, but there are also some institutions that choose to migrate to proprietary systems after using open source for several years (Corbett et al., 2016).

The following section provides a brief overview of currently available open source and proprietary DLMS and compares some of their features. The focus of the review is primarily on software used for building digital collections. Open source repository software, including DSpace, Fedora, Hydra, and Islandora are presented separately. It is difficult to compare these systems to the "out-of-the-box" software because of their unique structure, the requirements for additional development of the interface layer, and the integration of other open source solutions on top of the repository platform. The DLMS reviewed below are in the "turnkey" category of systems that are relatively easy to install and manage. In addition, several of the programs like CONTENTdm, LUNA, and Omeka offer hosted solutions, making collection building possible for small institutions with limited access to servers. The selected systems are well established and widely adopted by their designated user communities. The selection criteria for systems included in this section are: (1) a minimum of five years of development and use, and (2) a user base with a minimum of 100 active institutional users. Obviously, there are more than six DLMS that meet these criteria, but the limitations of this chapter dictate reducing the number. In addition, an attempt was made to present DLMS used in a variety of cultural heritage institutions, including archives and museums. The review of the systems is by no means comprehensive, nor is it meant to serve as a recommendation or evaluation. The goal of the review is purely informative and meant to provide a description of a sample of DLMS in open source and proprietary categories.

OPEN SOURCE SYSTEMS

CollectiveAccess (http://www.collectiveaccess.org/) is an open source content management system designed for cataloging, managing, and publishing museum and archival collections. It is also increasingly used by libraries, nonprofit organizations, private collectors, artist studios, and performing arts organizations (CollectiveAccess, 2015). The software was created in 2006 by Whirl-i-gig, a software development and consulting company, and was released to the public under the open source GNU Public License in 2007. It has been adopted by a variety of cultural institutions including archives, historical societies, libraries, and museums. CollectiveAccess provides users with highly configurable features including integrated metadata standards and controlled vocabularies, batch uploading of a variety of file formats, and customizable interfaces. The range of supported audio, video, and multimedia formats is impressive and probably one of the reasons why this software has been adopted by several film archives. The system includes two modules: *Providence*, the backend cataloging application and *Pawtucket*, the public-access interface. Both modules can be customized to obtain additional functionality. CollectiveAccess takes a flexible, cooperative approach to metadata standards, allowing users to import and share a variety of standards from user-contributed installation profiles (CollectiveAccess, 2015). It supports Dublin Core, PBCore, and VRA Core, as well as several archival and museum metadata standards including CDWA, CCO, DACS, and DarwinCore. The software also provides multilingual support in seven languages. Fig. 6.3 demonstrates an example of a bilingual German/English collection of photographs and videos of the fall of the Berlin Wall and German reunification. This software is highly configurable, but does require some customization and programming knowledge. Support is provided through a user forum and a wiki available to all institutional users.

Greenstone (http://www.greenstone.org/) represents the first generation of digital library software, but it has been upgraded several times and is actively used throughout the world. It was developed at

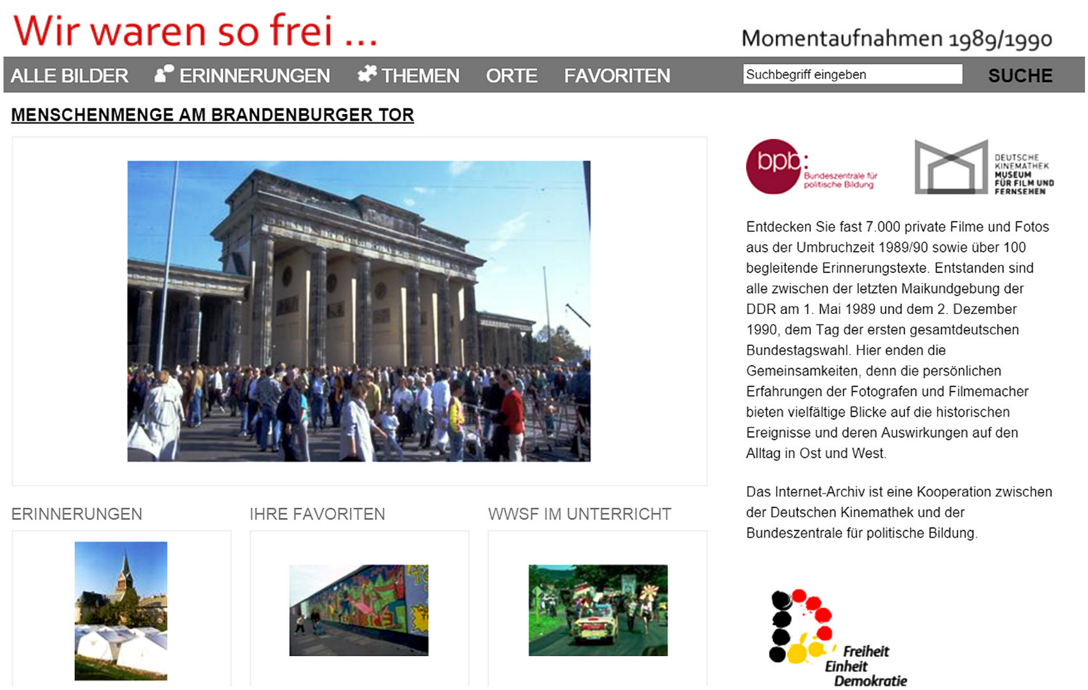

FIGURE 6.3 ***Wir waren so frei*—Built in CollectiveAccess by the Deutsche Kinemathek**

The collection is available at: http://www.wir-waren-so-frei.de.

the University of Waikato, New Zealand, in 1997, and became available as open source software under the GNU General Public License in 1998 (Witten and Bainbridge, 2003). It has been distributed in cooperation with UNESCO and the Human Info NGO in Belgium. Greenstone provides capabilities to create fully searchable collections of documents, books, photographs, newspapers, audio (mp3 files), and video. It supports the Dublin Core metadata standard (both unqualified and qualified). In addition, plug-ins can be used to ingest externally prepared metadata in different formats. Most collections are distributed on the web, but several collections of documents with humanitarian information have been produced on CD-ROM for distribution in developing countries (Witten, 2008).

Greenstone has a strong international and humanitarian focus and has been used as a platform for building digital collections representing indigenous cultures and social and environmental issues such as community development, poverty, sustainability, globalization, etc. (see Fig. 6.4 for a collection of UNESCO documents built in Greenstone). Multilingual support is one of Greenstone's strengths. For many years, Greenstone was the only digital library software providing Unicode support and the capability to process and display documents in non-Latin characters (Matusiak and Myagmar, 2009). Greenstone is one of the most widely adopted DLMS worldwide, with collections in more than 50 languages.

Omeka (http://omeka.org/) (hosted version: http://www.omeka.net/) provides a lightweight solution to building digital projects for cultural institutions and individuals. Omeka was created by the Center

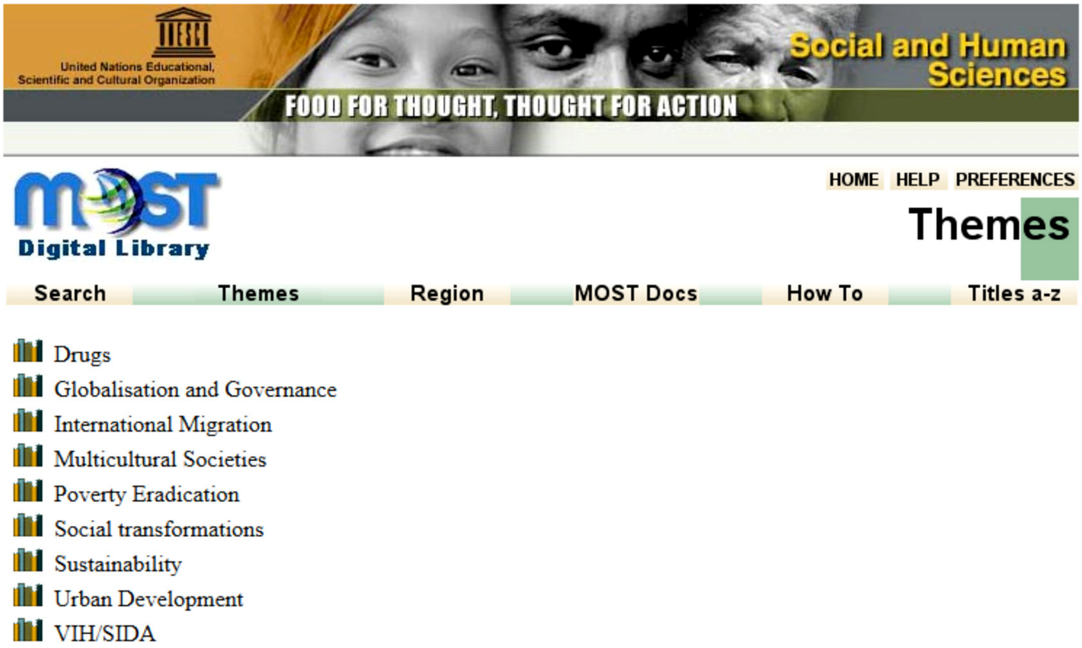

FIGURE 6.4 UNESCO Collection of Documents Documenting Social Science Research of the MOST Programme Built with Greenstone Software

UNESCO MOST Digital Library is available at: http://digital-library.unesco.org/shs/most/gsdl/cgi-bin/library?c=most&a=p&p=about.

for History and New Media at George Mason University in 2008. It is designed with non-IT specialists in mind to offer the opportunity to create digital collections and exhibits to those with limited infrastructure and/or technical skills (Omeka, 2015a). Omeka offers an easy-to-use platform to small archives and museums as well as to scholars and students in digital humanities and education. Fig. 6.5 provides an example of a digital history project created in Omeka by the Center for History and New Media in partnership with the Gulag Museum in Perm and the International Memorial Society in Russia. Users can choose to install an Omeka instance on their own servers from omeka.org, or they can use a hosted account with omeka.net. The installed version can be enhanced with the addition of plug-ins and has fewer customization restrictions than the omeka.net version. Omeka.net, however, provides an option of building digital collections to institutions and individuals who do not have access to the servers. As the documentation on the Omeka site indicates, the software has been adopted to teaching digital history, English, and library science courses (Omeka, 2015b). Fig. 6.2 provides an example of a digital collection built by graduate students.

Omeka accepts most formats for text, image, audio, and video files including jpg, jp2, PDF, mp3, mp4, and tif. The file size in the hosted version is restricted to 64 MB. The support of metadata standards is limited to basic Dublin Core. Some digital collection builders feel restricted by the limited fields in unqualified Dublin Core and the lack of the metadata template customizations (Kucsma et al., 2010). The “out-of-the-box” software provides very basic functionality, which can be expanded by the use

FIGURE 6.5 Gulag: Many Days, Many Lives

The project is available at: http://gulaghistory.org/.

of plug-ins. A selection of plug-ins is available for downloading to expand collection-building functions and to incorporate Web 2.0 tools. In addition to building objects and collections, Omeka offers capabilities for designing exhibits. The exhibit plug-in is one of the strengths of Omeka. The software is used sometimes as the digital exhibition platform in conjunction with other DLMS (Gilbert and Mobley, 2013).

PROPRIETARY SYSTEMS

CONTENTdm (http://www.oclc.org/en-US/contentdm.html) represents one of the first proprietary content management systems dedicated to building digital collections. This Windows-based software package was developed at the University of Washington in the late 1990s, and eventually acquired by OCLC (Zick, 2009). It was originally designed as an image management system, but through subsequent upgrades, the software has offered support for compound objects, PDFs, and streaming of audio

FIGURE 6.6 A Video Clip of Martin Luther King, Jr., Speech

The collection was built at the University of Wisconsin-Milwaukee Libraries with CONTENTdm software. The record is available at: http://collections.lib.uwm.edu/cdm/ref/collection/wtmj/id/49.

and video (see Fig. 6.6 for an example of a video collection). The software is licensed to participating institutions with a variety of license options and levels that depend on the size of collections. A hosted version is available for an additional fee (OCLC, 2015). CONTENTdm is intended to be scalable, enabling institutions to upgrade license levels as they increase capacity. CONTENTdm has been adopted by a wide range of cultural heritage institutions, from large academic libraries to smaller public libraries and archives. OCLC provides an entry-level hosted version to OCLC FirstSearch subscribers at no additional charge. OCLC members can take advantage of CONTENTdm's integration with Connexion cataloging tool and harvesting services through WorldCat Digital Collection Gateway. The software provides a wide range of services for streamlined digital collection building with metadata template customization, batch upload of content files and metadata, and automated creation of derivatives. Interface customization is limited, although users can make some changes through the CONTENTdm API.

KE EMu (http://www.kesoftware.com/) is an electronic museum content management system developed by the KE company based in Melbourne, Australia. The software is designed primarily for the museum market to manage internal workflows and to provide a platform for building and presenting digital collections online. The software has a diverse user base from small to large museums including American Museum of Natural History, National Museum of Australia, and National Museum of the American Indian (see Fig. 6.7 for an example of the implementation of EMu). EMu is designed to

FIGURE 6.7 The National Museum of the American Indian Use of EMu Software

The record is available at: http://www.nmai.si.edu/searchcollections/item.aspx?irn=15330&hl=443.

facilitate multiple functionalities with modules for collection management, cataloging, interpretation, and online publishing. The collection management module is designated to record object details and to track activities, such as exhibitions, loans, and acquisitions. The module is compliant with SPECTRUM, a guide to good practice for museum documentation. The software supports Dublin Core and Darwin Core and integrates a number of controlled vocabulary tools. EMu allows for the customization of metadata fields so that the fields match the discipline and attributes of the object (KE Software, 2015). EMu also provides an interpretation module for capturing experts' knowledge and presenting a collection within a cultural, historical, or scientific context. Digital objects and their online publication are managed with a digital asset management module. With a highly modular structure, strong support of multiple media formats, and several metadata standards, EMu provides a powerful tool for managing unique museum workflows and building online collections. The licensing structure depends on the number of concurrent users, types, and sizes of legacy databases that need to be converted, and training required.

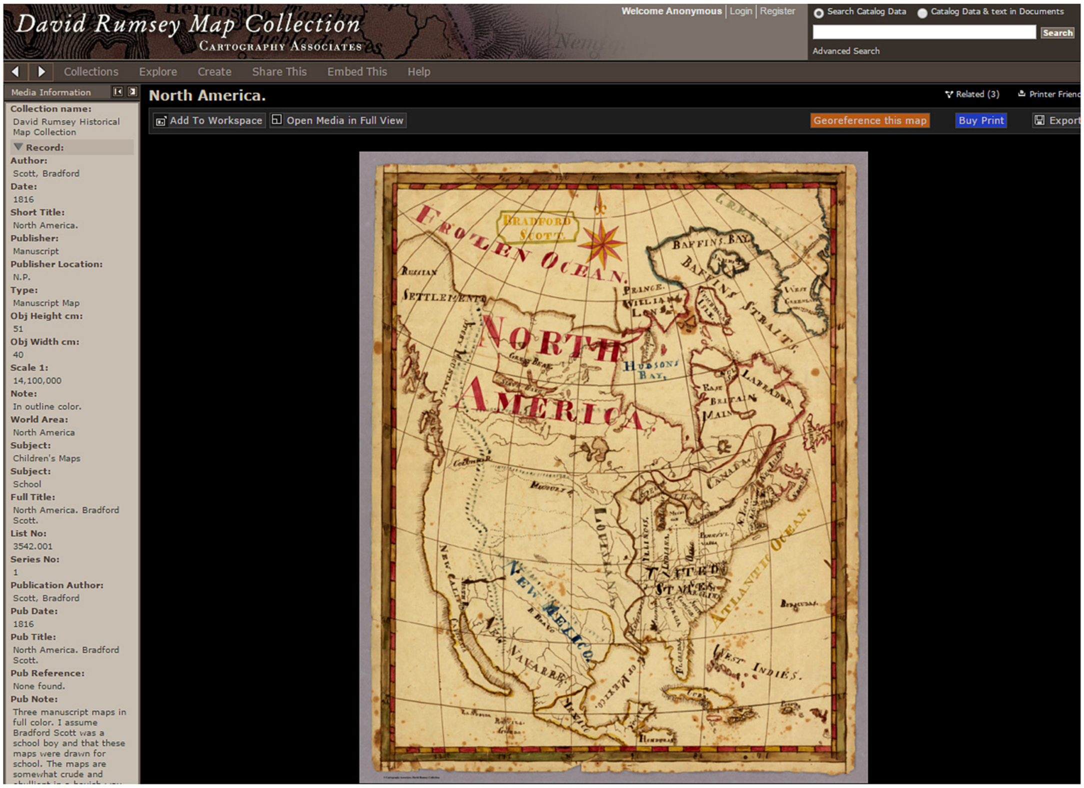

FIGURE 6.8 A High-Resolution Map Image in the David Rumsey Map Collection Built with LUNA Software

Available at: http://www.davidrumsey.com/luna/servlet/s/163gby.

LUNA software is produced and licensed by Luna Imaging, Inc., a company formed with support from the J. Paul Getty Trust and Eastman Kodak Company. In addition to maintaining and supporting the software, Luna Imaging, Inc., also provides imaging and consulting services to cultural heritage institutions. LUNA software is primarily used for managing and presenting visual collections, although it is capable of managing other types of content as well, including text, audio, and video. However, its powerful image viewer with zoom and pan capabilities makes is particularly suitable for presenting high-resolution visuals, such as art images, maps, or medical and scientific imagery (see Fig. 6.8 for an example of a map from the David Rumsey Collection). The software provides additional tools to annotate images, export to PowerPoint, and create dynamic presentations and slide shows. Because of its strong capabilities in presenting high-resolution images and integrating with presentations software, LUNA has been adopted by museums and academic libraries with visual collections. As of November 2015, Luna Imaging offered a site license for installing the software on the local server and a hosted version, Luna Solo, for smaller institutions and individuals. The hosted option requires an annual fee and allows 100 GB of data (Luna Imaging, 2015). The software provides some Web 2.0 tools, including integration with Flickr images.

Table 6.1 provides a summary of selected features of the reviewed DLMS based on openly available documentation. The review was conducted in 2014 and updated in 2015; it captures the features of

Table 6.1 Comparison of Selected Features of Open Source and Proprietary DLMS

	Open Source			Proprietary		
Features	**Collective Access**	**Greenstone**	**Omeka**	**CONTENTdm**	**EMu**	**LUNA**
File formats	jpg, jp2, PDF, mp3, mp4, tiff, plus many audio and video formats[a]	jpeg, PDF, gif, jif, tiff, mp3, mpeg, midi, and others	jpg, jp2, PDF, mp3, mp4, tiff, and others	jpg, jp2, PDF, mp3, mp4, tiff,	jpg, PDF, avi, wav, mpeg, and others	jpg, PDF, png, gif, bmp,mp3, tiff, flv
File conversion	√	U	N/A	√	√	√
Batch upload of files	√	√	√	√	√	√
Batch upload of metadata	√	√	√	√	√	√
Metadata schemas	Dublin Core, Darwin Core, and archival and museum standards[b]	Dublin Core	Dublin Core	Dublin Core, VRA Core, EAD, METS	Dublin Core, Darwin Core, XMP, IPTC	Dublin Core
Metadata template customization	√	√	N/A	√	√	√
Controlled vocabulary	AAT, TGN, GeoNames, LCSH, your own vocabulary	N/A	LCSH, your own vocabulary	AAT, TGM, TGN, ULAN, CSH, your own vocabulary, and more	AAT, LCSH, TGN, ULAN, many others, your own vocabulary	Your own vocabulary
Persistent object identifier	N/A	N/A	N/A	√	U	√
Allow multiple collections	√	√	√	√	√	√
Global update of metadata	√	U	N/A	√	√	√
Multilingual support	√	√	√	√	√	√
High-resolution Image Viewer	√	N/A	N/A	√	√	√
Interoperability	OAI-PMH	OAI-PMH, Z39.50	OAI-PMH	OAI-PMH, Z39.50	OAI-PMH, Z39.50	OAI-PMH
Platform	Windows, Unix/Linux, Mac OS-X	Windows, Unix/Linux, Mac OS-X, FreeBSD	Windows, Unix/Linux, Mac OS-X	Windows, Unix/Linux	Windows, Unix/Linux, Mac OS-X	Windows Unix/Linux, Solaris
Hosted platform	N/A	N/A	√	√	N/A	√

U, unknown; NA, not available/applicable.
[a]*For a complete list of formats, see: http://docs.collectiveaccess.org/wiki/Supported_Media_File_Formats.*
[b]*For a complete list of supported metadata standards, see: http://docs.collectiveaccess.org/wiki/Metadata_Standards.*

Table 6.2 Web 2.0 Features and Visualization Tools of Selected Open Source and Proprietary DLMS

	Open Source			Proprietary		
Features	**Collective Access**	**Greenstone**	**Omeka**	**CONTENTdm**	**EMu**	**LUNA**
Web 2.0 Features and Visualization Tools						
Geo-referencing	√	N/A	√	N/A	√	√
Visualization tools	√	N/A	√	N/A	√	√
Social media sharing	√	N/A	√	√	N/A	√
User tagging	√	N/A	N/A (only those logged in can tag, public users cannot)	√	N/A	N/A
User contribution	√	N/A	√	N/A	N/A	√

NA, not available/applicable.

the systems at that time. It is very likely that the systems will evolve and add new functionality in the future. The comparison in Table 6.1 focuses on a small number of core functionality related to content creation.

Table 6.2 examines a handful of Web 2.0 features and visualization tools. It is by no means a comprehensive comparison, as it does not address the functionality related to content management, discovery, presentation, user interface, or administration. Nonetheless, the tables provide a snapshot of differences and similarities between currently available open source and proprietary DLMS, and to a certain extent between early systems like Greenstone or CONTENTdm and the next generation of DLMS, such as CollectiveAccess or Omeka.

As Tables 6.1 and 6.2 demonstrate, there is no single system that will provide all features (even if the list is narrowed to one category, such as content creation and ingest). In fact, this small-scale comparison reveals considerable differences between the systems in both open source and proprietary categories. All systems support Dublin Core and OAI-PMH, but beyond basic Dublin Core, options for selecting other schemas and customizing metadata templates vary. CollectiveAccess offers the largest selection of metadata standards, but at the same time it requires more technical skills in configuring the system. Open source DLMS, like CollectiveAccess or Omeka, provide support for the widest range of file formats, especially in audio, video, and multimedia, but all reviewed systems support at least one standard audio and video file format. The biggest difference between newer open source systems and proprietary software is in the integration of Web 2.0 tools for sharing objects through social media, user tagging, and user contribution. The generation of DLMS developed in the open source culture of collaboration and sharing, such as CollectiveAccess or Omeka, are much better at integrating Web 2.0 features into their systems.

The systems reviewed in this section focus primarily on providing access to digital cultural heritage content from libraries, archives, and museums. The functionality is centered on managing and ingesting digital objects, their discovery, and online delivery. Long-term preservation of digital assets, especially of master files, is typically not supported in digital collection software. Thus, institutions need to purchase or build separate digital preservation systems, and integrate them with digital collection platforms. In addition, institutions may need to maintain digital repositories to provide access and preservation of open access scholarly publications and research data. The multitude of systems creates a complex environment with challenges in streamlining operations and providing integrated services and discovery experiences.

OPEN SOURCE REPOSITORY SYSTEMS

Digital repositories are multifunctional information systems that support a wide range of digital curation activities, including content management, submission, ingesting, storing, discovery, access, and preservation (Rieger, 2007). The first digital repository systems were developed in the early 2000s to enable open access to scholarly publications. In support of scholarly communication and open access, repositories were designed to provide an infrastructure and services for capturing and managing open access scholarly publications and electronic dissertations and theses. Digital repositories are a key component of the Open Access (OA) movement (Swan, 2012). Repositories provide access to intellectual output of an institution or in a specific discipline. Institutional repositories represent the most common type (OpenDOAR, 2015). Lynch (2003) defines institutional digital repositories as “a set of services that a university offers to the members of its community for the management and dissemination of digital materials created by the institution and its community members” (p. 328). Early repository software, such as DSpace and EPrints were developed to support self-deposits of open access scholarly publications and to enable resource discovery and retrieval in the open web environment. The first generation of open-source repository systems was designed primarily as open access publication databases with limited support for other types of digital objects and services (Fay, 2010).

The content and roles of digital repositories, however, have become more diversified with time. In addition to providing open access to scholarly articles and dissertations and theses, repositories began to host diverse content of digitized archival and special collections, geospatial data, research data, audiovisual materials, and complex multimedia objects. In the case of institutions participating in the Google Book Project, institutional repositories also serve as preservation systems for master copies of digitized books (Cramer and Kott, 2010). Rieger (2007) outlines a long list of repository roles. The top purposes include:

- Enable digital asset management
- Offer preservation services
- Provide institutional visibility
- Support learning, teaching, and research
- Facilitate discovery of content
- Enable reuse and repurposing of content

Increasingly, repositories are also seen as platforms for curating faculty’s digital scholarship and supporting access, preservation, and reuse of digital assets created or collected by faculty as part of their research and teaching activities. Digital repositories that aggregate diverse content are especially valuable for research and instruction (Kutay, 2014).

Digital preservation has emerged as an equally important goal as access for repository systems. The challenges, strategies, and technologies for preserving born digital and digitized objects are described in more detail in chapter on digital preservation (Chapter 9). Digital repositories are discussed there in terms of providing preservation services. As mentioned before, the early repository systems, such as EPrints and DSpace, offer limited support for digital preservation. The need for more robust preservation services is cited as one of the reasons for migrating from the early generation repository software toward newer and more versatile systems (Cramer and Kott, 2010).

The new generation of open source repository systems addresses the limitations of the early repositories by offering stronger support for features essential to digital preservation. The emerging open source solutions also address the challenge of maintaining more than one system in support of institutional curatorial processes. The new systems serve multiple purposes and offer multiple functions with user interface for deposit, description, discovery, and retrieval of digital content as well as a range of specialized functions to make deposited assets available over the long term. Adherence to digital preservation standards and integration of tools, such as checksums and format validation applications, ensure integrity, authenticity, and reusability of digital objects (Awre and Cramer, 2012; Cramer and Kott, 2010; Kent, 2014). Fig. 6.9 demonstrates a multipurpose model of the Stanford Digital Repository (SDR) with three distinct yet integrated areas: (1) management, (2) discovery and delivery, and (3) preservation.

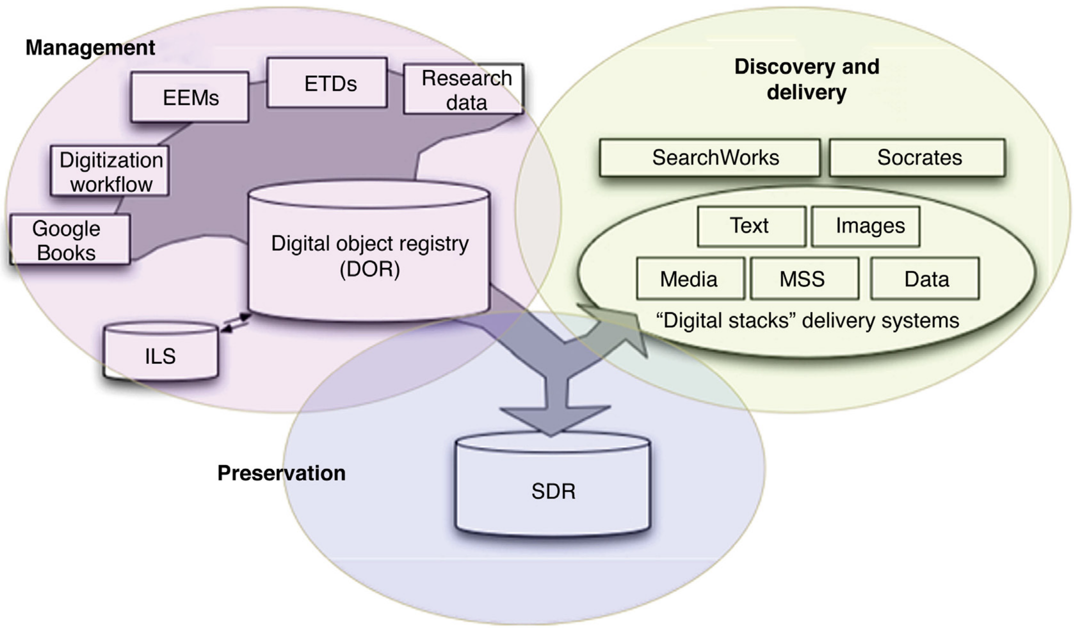

FIGURE 6.9 Model of the Stanford University Digital Repository System with Three Main Spheres: Management, Preservation, and Access (Cramer and Kott, 2010)

The new repository serving the Stanford University community is designed to accommodate diverse content and serve multiple functions. It is based on modular architecture and constructed with multiple open source technologies, including Fedora as an object management framework (Cramer and Kott, 2010). More recently, Stanford joined the Hydra Project and incorporated Hydra into a suite of the repository technologies (Awre and Cramer, 2012).

The emerging generation of multipurpose repository systems adopts modular architecture and integrates a suite of open source applications. As outlined by Fay (2010), modular architecture allows:

- Independence of functional components
- A separation of digital objects from particular software installations
- An iterative approach to developing capacity in different functional areas
- Flexible design according to evolving demands and requirements

Modular architecture and the integration of several open source components enable customization and ensure flexibility and extensibility.

The following section reviews four open source repository systems: DSpace, Fedora, Hydra, and Islandora. DSpace and Fedora are listed in the survey research as widely used repository systems (Moulaison et al., 2015; Stein and Thompson, 2015). DSpace is an example of the early repository software. Fedora is central to the digital library development not only as an architectural model but also as a software used in many operational repositories, and a foundation for the new repository systems. Hydra and Islandora represent the new generation of DLMS.

DSpace (http://www.dspace.org/) is one of the first open source repository solutions supporting not only the dissemination of scholarly publications but also the access to digital collections. It was developed under the leadership of the Massachusetts Institute of Technology (MIT) and in collaboration with the Hewlett Packard Corporation. Since its release in 2002, DSpace has been widely adopted and is used by over 1000 institutions worldwide (DSpace, 2015; OpenDOAR, 2015). Its usage accounts for 43% repositories registered with the Directory of Open Access Repositories (OpenDOAR, 2015). Currently, DSpace is supported by a community of users and maintained through DuraSpace (DSpace, 2015).

DSpace supports Dublin Core metadata standard but does not include other metadata schemas. It is compliant with the OAI-PMH interoperability standard (Smith et al., 2003). Metadata may be entered by end users as they submit content, or it might be derived from other metadata as part of an ingest process (Tramboo et al., 2012). It uses a handle system for identifiers. The software is relatively easy to install and customizable. However, some limitations of its early architecture are noticed especially in comparison to Fedora. Fay (2010) compared DSpace, EPrints, and Fedora. The author identified a number of DSpace disadvantages, including its monolithic and restrictive data model, limited support for complex objects and digitized collections, lack of support for identifier schemas, and the lack of attention to preservation storage. The limited support of DSpace for preservation services was also found in another comparative study (Madalli et al., 2012).

Fedora (http://fedorarepository.org/) has emerged as one of the most versatile solutions among open source digital repositories. Built on the Fedora flexible and extensible architecture, the repository software supports a wide range of applications including institutional repositories, electronic records archives, e-publishing, trusted repositories for digital preservation, digital collections, and distributed digital libraries (Lagoze et al., 2006). In comparison to DSpace and EPrints, Fedora supports a wider range of objects and multiple metadata schemas and can be customized to local requirements (Fay, 2010). Fedora offers a flexible data model and a robust repository but requires additional tools to build an interface layer. Fedora requires significant local development in order to achieve any useful functionality (Fay, 2010). The user and web services interfaces need to be developed with other open source applications. Fedora provides a very flexible environment, but it requires substantial technical skills to implement—thus it is often adopted by larger institutions with sufficient programming and technical staff.

Fedora serves as a foundation for major scientific digital libraries, including the Public Library of Science (PLoS ONE), the National Library of Medicine (NLM), and the National Science Digital Library (NSDL). The NSDL served as a testing environment for Fedora repository structure and additional open source development (Krafft et al., 2008). NLM selected Fedora after extensive testing and comparing it to ten other open source and proprietary systems (Marill and Luczak, 2009). Fedora also provides a repository platform for cultural heritage collections, including Open Vault, a moving image archives project of the WGBH Media Library and Archives (WGBH Educational Foundation, 2015). Like DSpace, Fedora is supported by a community of users and maintained through DuraSpace (Fedora, 2015). Increasingly, Fedora is integrated with the new generation of front-end applications, such as Islandora and Hydra. As of Nov. 30, 2015, the Fedora user registry listed 73 Islandora/Fedora and 35 Hydra/Fedora registered repositories (DuraSpace, 2015a).

Hydra (http://projecthydra.org/) is a multipurpose open source repository solution that has been developed as a collaborative project by several organizations (Hydra Project, 2015). The Hydra Project started in 2008 with the University of Hull in the UK, Stanford University, University of Virginia, and Fedora Commons (now DuraSpace) as foundational partners (Awre, 2012; Awre and Cramer, 2012; Green and Awre, 2009). Since then other institutions have joined the Hydra Project as partners and have formally committed themselves to supporting Hydra's open source development (Hydra Project, 2015).

The collaborative Hydra Project was initiated in light of the recognition that no single system can provide the full range of repository-based solutions and that no single institution can support such large-scale development (Awre, 2012; Awre and Cramer, 2012). Thus, the goal of the Hydra Project participants was to develop jointly a common repository framework upon which flexible solutions can be built and shared. An African proverb, "If you want to go fast, go alone, if you want to go far, go together" became the project's motto. Multiinstitutional collaboration has become a vital aspect of the project.

Building on Fedora, Hydra is designed as a flexible application framework that can support the development of multiple systems tailored to local needs. As Awre and Cramer (2012) point out, the use of the term "Hydra" deliberately indicates one body and many heads. Hydra provides a common and reusable framework, which can be adopted, extended, and modified. Each instance of Hydra adoption can become its own Hydra head. Hydra developers and adopters also contribute to the wider Hydra community (DuraSpace, 2015b). Hydra software is released as open source under the Apache 2.0 license.

As a technical framework, Hydra provides a set of reusable open source components that can be combined and configured to meet different needs. The major components include:

- Fedora as a repository layer to support object management
- Apache Solr, indexing software to provide access to indexed objects
- Blacklight: a next-generation discovery interface that provides faceted search and customized views
- Hydra plugin, a collection of components that facilitate workflow in managing digital content
- Solrizer, a component that indexes Fedora-held content into a Solr index (Awre, 2012)

Several of the components use Ruby on Rails as the coding language. Hydra software or its components have been implemented by a number of large research institutions worldwide. In addition to the initial Hydra partners, the software has been adopted by Northwestern University, Indiana University, Columbia University, London School of Economics, and many others (Hydra Project, 2015). The list of Hydra partners and adopters is growing steadily. Hydra offers a multipurpose, multifunctional DLMS by combining the flexibility of the repository structure with versatile functionality and rich applications. However, it requires substantial development work and expertise in the Ruby on Rails programming language.

Islandora (http://islandora.ca/) is an open source software framework designed to help institutions collaboratively manage and discover digital assets. It was originally developed by the University of Prince Edward Island's Robertson Library but is now adopted by an international community of users (Islandora, 2015). Since its initial release in 2006, Islandora has been upgraded to support diverse content and extend its functionality (Moses and Stapelfeldt, 2013). Islandora has evolved into a multipurpose repository system serving as a platform for open access publishing as well as a digital asset management system for cultural heritage collections (Jettka and Stein, 2014; Kent, 2014; Moses and Stapelfeldt, 2013). Islandora is released as an open source software under a GNU license.

Islandora's technical framework is built using a modular architecture approach. It is based on Fedora and integrates additional open source applications. The core components include:

- Fedora as a repository framework to support data storage, RDF relationships, and metadata harvesting
- Drupal as a front-end content management application to provide user interface and extend Islandora elements
- Solr for indexing services

Islandora's functionality can be further extended by incorporating other open source applications (Moses and Stapelfeldt, 2013). Islandora uses Drupal module to create solution packs for different content types and formats, such as audio and video or PDFs. Each solution pack has its own set of tools to support automatic processing during ingest. For example, while a batch of master files in the TIFF format is uploaded for archival storage, JPEG or JPEG 2000 derivatives are created automatically for access (Kent, 2014). Islandora supports the MODS schema but accepts other metadata standards at ingest, which are then mapped to MODS. It offers support for digital preservation standards and tools, including PREMIS metadata standard and checksums.

Islandora has been implemented as an open access scholarly repository with a suite of unique services supporting digital scholarship (Moses and Stapelfeldt, 2013). It has also been used as a solution for presenting diverse scholarly and cultural heritage content in digital collections and digital humanities projects. University of Hamburg adopted Islandora to construct a repository for a multilingual collection of spoken language resources (Jettka and Stein, 2014). A consortium of academic libraries in Minnesota selected Islandora as a platform for providing access and supporting preservation of digital collections (Kent, 2014). Islandora is supported collaboratively by a community of users who interact through Islandora Camps and Google Groups (Islandora, 2015).

The reviewed open source repository systems accommodate diverse content and formats, and seek to serve multiple repository purposes. In contrast to DSpace, the new generation of systems offers more flexible platforms and stronger support for complex objects and digital preservation. The new systems are constructed using a stack of open source technologies. While Fedora provides a core repository framework, the systems also incorporate other open source components. They can be modified and extended by integrating additional applications to meet new requirements. Collaborative, multiinstitutional development is a distinct feature of the new generation of open source repository systems. Shared development contributes to the system functionality and sustainability, but can also help in sharing other digital library standards and best practices.

SELECTION OF DLMS

The design of DLMS has evolved in the past decade with the introduction of low-barrier systems, robust and flexible open source repository systems, and more specialized options available to specific user communities like archives and museums. Selection of an appropriate system among so many alternatives, however, is not easy. The variety of DLMS and their features makes the selection and evaluation process challenging. In addition to the "out-of-the-box" solutions, institutions have the option of customizing available open source software or building their own custom systems. The choice really depends on institutional needs and requirements, types of resources, scale of collections, technical infrastructure and expertise, long-term goals, and budget. As discussed above, the selection between open source and proprietary systems needs to be considered not only in terms of initial cost but also in light of total cost of ownership (Samuels and Griffy, 2012). Selecting a well-established proprietary system can make perfect sense for an institution with limited technical and programming staff, especially if the system meets other functionality requirements. Selecting the right system requires a considerable amount of research, evaluation of options, and testing the performance of selected candidates. Zhang and Gourley (2009) identify four steps in the selection process:

- Identifying organizational requirements and resources
- Developing selection criteria
- Researching available systems
- Evaluating candidates

Organizational requirements depend on the types and characteristics of resources, needs of intended users, traditions of information organization and resource sharing, strategic goals, consortia agreements, and the technical infrastructure of individual institutions. Selection criteria need to be developed in light of organizational requirements. Developing a list of selection criteria and system requirements can be an overwhelming task. There is no single checklist that would include all the requirements for content creation and management, online presentation and access, user interface, preservation, user and system administration, etc. The lack of evaluation tools has been noted by the researchers examining open source DLMS (Goh et al., 2006; Samuels and Griffy, 2012) and is further discussed in Chapter 10.

Several case studies focus on the process of selecting and evaluating open source DLMS and report their methodology and findings. Although these studies concentrate exclusively on open source software, their approaches and selection criteria can be useful for evaluating proprietary systems as well. In search of a suitable DLMS, the researchers at the University of Arizona Library adopted the systems analysis process (Han, 2004). They identified four major areas of functional requirements: information organization focusing on content and its associated metadata, presentation, access including interfaces for both internal and external users, and preservation. For the purpose of their study, the researchers developed detailed criteria within those categories and conducted a comparative study of three DLMS: Greenstone, Fedora, and DSpace.

Goh et al. (2006) conducted a study in order to develop a standardized checklist for evaluating open source DLMS. The researchers identified 12 categories and developed a weighted list of criteria within those categories. The checklist categories include content management, content acquisition, metadata, search support, access control and privacy, report and inquiry capabilities, preservation, interoperability, user interfaces, standard compliance, automatic tools, system support, and maintenance. The evaluation checklist generated as a result of this research was used in evaluating four DLMS CERN CDSware, Greenstone, Fedora, and EPrints. Greenstone emerged as a system that consistently fulfilled the majority of the criteria and obtained full scores in five of the 12 categories.

Based on the case studies of Han (2004) and Goh et al. (2006) as well as their own experiences, Zhang and Gourley (2009) proposed a model called "FITS to Organizations" or "FITS to O" to be used in selecting and evaluating DLMS. This model groups the core requirements into four categories, functionality, interface, technology, and support, and then identifies subcategories within each group. The categories along with suggested weights are compiled into a relatively comprehensive yet flexible checklist. As the authors point out, the checklist can be adopted to individual institutional needs or used in evaluating selected DLMS components like functionality or interface (Zhang and Gourley, 2009).

A number of studies discuss the selection of open source repository software and provide criteria for their evaluation (Fay, 2010; Marill and Luczak, 2009; Rieger, 2007). Rieger (2007) discusses the selection process in light of multiple purposes of repositories and presents an evaluation model that involves several steps, including stakeholder analysis, needs assessment, service definition, and identification of use cases and governance-related matters. Marill and Luczak (2009) report on the process of selecting a repository system for NLM. In this case, ten repository systems were evaluated according to the established functional and nonfunctional evaluation criteria. Three systems that ranked high, DigiTool, DSpace, and Fedora, were tested further. Fedora achieved the highest rank in this rigorous evaluation process and was recommended for implementation at NLM. Likewise, Fedora emerged as the most flexible repository in a comparative study conducted by Fay (2010). The NLM case study includes a useful list of evaluation criteria that can be applied in other settings (Marill and Luczak, 2009). DLMS evaluation categories and criteria are also part of larger digital library evaluation studies that are discussed in Chapter 10.

The landscape of DLMS has evolved from the early standalone systems to flexible, open models and collaborative, multipurpose systems. The development of open source software and the competing systems from commercial vendors provide digital library developers with many options to create and manage digital content. The development of lightweight solutions like Omeka has dramatically changed the DLMS landscape, and opened the participation in digital content creation to smaller institutions and individuals. Building digital collections no longer has to be tied to the institutions and large-scale DLMS. The range of specialized DLMS offers differing features that can be tailored to the needs of user communities and different types of content. The selection process, although still not easy, is informed by a growing body of research on evaluation models and criteria.

The increasing diversity of DLMS does not mean that the currently available systems meet all user requirements. While there has been significant progress in interoperability and building flexible systems, user-level functionality still leaves much to be desired. DLMS function for the most part as databases of objects and associated metadata with limited capabilities for building layers to present contextual information. Further, they are limited in their ability to provide workspaces for end users to manipulate digital objects, contribute their own materials, or collaborate with others. Ioannidis et al. (2005) describe a number of hypothetical scenarios for advanced digital library functionality to support users in cultural heritage and scientific communities. Many concepts proposed in those scenarios, such as real-time construction of collections, personal annotations, or collaborative spaces, are still not supported by most standard DLMS. The design of DLMS is still a work in progress.

REFERENCES

Andro, M., Asselin, E., Maisonneuve, M., 2012. Digital libraries: comparison of 10 software. Lib. Collect., Acquisit. Tech. Serv. 36 (3), 79–83.

Arms, W.Y., 1995. Key concepts in the architecture of the digital library. D-Lib Mag. 1 (1).

Arms, W.Y., 2000. Digital Libraries. MIT Press, Cambridge, MA.

Arms, W.Y., Hillmann, D., Lagoze, C., Krafft, D., Marisa, R., Saylor, J., Van de Sompel, H., 2002. A spectrum of interoperability: the site for science prototype for the NSDL. D-Lib Mag. 8 (1), 9. Available from: http://www.dlib.org/dlib/january02/arms/01arms.html.

Awre, C., 2012. Hydra UK: flexible repository solutions to meet varied needs. Ariadne 70. Available from: http://www.ariadne.ac.uk/issue70/hydra-2012-11-rpt.

Awre, C., Cramer, T., 2012. Building the Hydra together: enhancing repository provision through multi-institution collaboration. J. Digital Inf. 13 (1), 1–11.

Breeding, M., 2013. Technology alternatives for special collections. Comp. Lib. 33 (9), 19–22.

Breeding, M., 2015. The Future of Library Resource Discovery. A White Paper Commissioned by the NISO Discovery to Delivery (D2D) Topic Committee. NISO. Available from: http://www.niso.org/apps/group_public/download.php/14487/future_library_resource_discovery.pdf.

Burns, C.S., Lana, A., Budd, J.M., 2013. Institutional repositories: exploration of costs and value. D-Lib Mag. 19 (1–2). Available from: http://www.dlib.org/dlib/january13/burns/01burns.html.

Candela, L., Castelli, D., Pagano, P., 2007a. A reference architecture for digital library systems: principles and applications. In: Digital Libraries: Research and Development, pp. 22–35. Berlin: Springer.

Candela, L., Castelli, D., Pagano, P., Thano, C., Ioannidis, Y., Koutrika, G., Schuldt, H., 2007b. Setting the foundations of digital libraries: the DELOS manifesto. D-Lib Mag. 13 (3), 4. Available from: http://www.dlib.org/dlib/march07/castelli/03castelli.html.

Candela, L., Castelli, D., Fox, E.A., Ioannidis, Y., 2010. On digital library foundations. Int. J. Digital Lib. 11 (1), 37–39.

Chowdhury, G., Chowdhury, S., 2003. Introduction to Digital Libraries. Facet Publishing, London.

CollectiveAccess, 2015. About CollectiveAccess. Available from: http://www.collectiveaccess.org/support/library.

Concordia, C., Gradmann, S., Siebinga, S., 2010. Not just another portal, not just another digital library: a portrait of Europeana as an application program interface. IFLA J. 36 (1), 61–69.

Corbett, H., Ghaphery, J., Work, L., Byrd, S., 2016 forthcoming. Choosing a repository platform: open source vs. hosted solutions. In: Burton, B., Callicott, D.S., Wesolek, A. (Eds.), Making Institutional Repositories Work. Charleston Insights in Library, Archival, and Information Sciences. Purdue University Press. Available from: http://scholarscompass.vcu.edu/cgi/viewcontent.cgi?article=1036&context=libraries_pubs.

Cramer, T., Kott, K., 2010. Designing and implementing second generation digital preservation services: a scalable model for the Stanford digital repository. D-Lib Mag. 16 (9), 1. Available from: http://www.dlib.org/dlib/september10/cramer/09cramer.html.

Davis, J.R., Lagoze, C., 2000. NCSTRL: design and deployment of a globally distributed digital library. J. Am. Soc. Inf. Sci. 51 (3), 273.

DSpace 2015, DSpace Home. Available from: http://www.dspace.org/.

DuraSpace, 2015a. Fedora user registry. Available from: http://www.duraspace.org/registry/fedora.

DuraSpace, 2015b. Hydra community framework. Available from: https://wiki.duraspace.org/display/hydra/Hydra+Community+Framework.

Fay, E., 2010. Repository software comparison: building digital library infrastructure at LSE. Ariadne 64. Available from: http://www.ariadne.ac.uk/issue64/fay.
Fedora, 2015. Fedora Repository Home. Available from: http://fedorarepository.org/.
Fox, R., 2006. The digital library in the bazaar. OCLC Syst. Serv. Int. Digital Lib. Persp. 22 (2), 100–106.
Gilbert, H., Mobley, T., 2013. Breaking up with CONTENTdm: why and how one institution took the leap to open source. Code4lib J. 20 (4). Available from: http://journal.code4lib.org/articles/8327.
Goh, D.H.-L., Chua, A., Khoo, D.A., Khoo, E.B.-H., Mak, E.B.-T., Ng, M.W.-M., 2006. A checklist for evaluating open source digital library software. Online Inf. Rev. 30 (4), 360–379.
Gonçalves, M.A., Fox, E.A., Watson, L.T., 2008. Towards a digital library theory: a formal digital library ontology. Int. J. Digital Lib. 8 (2), 91–114.
Green, R., Awre, C., 2009. Towards a repository-enabled scholar's workbench. D-Lib Mag. 15 (5/6). Available from: http://www.dlib.org/dlib/may09/green/05green.html.
Han, Y., 2004. Digital content management: the search for a content management system. Lib. Hi Tech. 22 (4), 355–365.
Henry, G., 2012. Core Infrastructure Considerations for Large Digital Libraries. Council on Library and Information Resources, Digital Library Federation, Washington, DC.
Hydra Project, 2015. About Hydra. Available from: http://projecthydra.org/.
Ioannidis, Y., Maier, D., Abiteboul, S., Buneman, P., Davidson, S., Fox, E., Weikum, G., 2005. Digital library information-technology infrastructures. Int. J. Digital Lib. 5 (4), 266–274.
Islandora, 2015. About. Available from: http://islandora.ca/about.
Jettka, D., Stein, D., 2014. The HZSK repository: implementation, features, and use cases of a repository for spoken language corpora. D-Lib Mag. 20 (9/10). Available from: http://www.dlib.org/dlib/september14/jettka/09jettka.html.
Kahn, R., Wilensky, R., 2006. A framework for distributed digital object services. Int. J. Digital Lib. 6 (2), 115–123.
Kaplan, D., 2009. Choosing a digital asset management system that's right for you. J. Archival Org. 7 (1/2), 33–40.
KE Software, 2015. Research. Available from: http://emu.kesoftware.com/about-emu/overview/research.
Kent, A., 2014. Islandora. Comp. Lib. 34 (9), 12–32.
Krafft, D.B., Birkland, A., Cramer, E.J., 2008. NCore: architecture and implementation of a flexible, collaborative digital library. In: Proceedings of the Eighth ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 313–322. ACM.
Krishnamurthy, M.M., 2008. Open access, open source and digital libraries: a current trend in university libraries around the world. Prog. Electron. Lib. Inf. Syst. 42 (1), 48–55.
Kucsma, J., Reiss, K., Sidman, A., 2010. Using Omeka to build digital collections: the METRO case study. D-Lib Mag. 16 (3/4). Available from: http://www.dlib.org/dlib/march10/kucsma/03kucsma.html.
Kutay, S., 2014. Advancing digital repository services for faculty primary research assets: an exploratory study. J. Acad. Librar. 40 (6), 642–649.
Lagoze, C., Fielding, D., 1998. Defining collections in distributed digital libraries. D-Lib Mag. 4 (11).
Lagoze, C., Van de Sompel, H., 2001. The open archives initiative: building a low-barrier interoperability framework. In: Proceedings of the First ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 54–62.
Lagoze, C., Van de Sompel, H., 2003. The making of the open archives initiative protocol for metadata harvesting. Lib. Hi Tech. 21 (2), 118–128.
Lagoze, C., Payette, S., Shin, E., Wilper, C., 2006. Fedora: an architecture for complex objects and their relationships. Int. J. Digital Lib. 6 (2), 124–138.
Luna Imaging, 2015. Luna Solo. Available from: http://www.lunaimaging.com/solo.
Lynch, C.A., 2003. Institutional repositories: essential infrastructure for scholarship in the digital age. Portal Lib. Acad. 3 (2), 327–336.
Madalli, D.P., Barve, S., Amin, S., 2012. Digital preservation in open-source digital library software. J. Acad. Librar. 38 (3), 161–164.

Marill, J.L., Luczak, E.C., 2009. Evaluation of digital repository software at the National Library of Medicine. D-Lib Mag. 15 (5–6). Available from: http://www.dlib.org/dlib/may09/marill/05marill.html.

Matusiak, K.K., Myagmar, M., 2009. Newspaper digitization project in Mongolia: creating a digital archive of Mongolian rare periodicals. Ser. Librar. 57 (1), 118–127.

Moses, D., Stapelfeldt, K., 2013. Renewing UPEI's institutional repository: new features for an Islandora-based environment. Code4lib J. 21 (9). Available from: http://journal.code4lib.org/articles/8763.

Moulaison, H.L., Dykas, F., Gallant, K., 2015. OpenDOAR repositories and metadata practices. D-Lib. Mag. 21 (3/4), 1. Available from: http://www.dlib.org/dlib/march15/moulaison/03moulaison.htm..

OCLC, 2015. CONTENTdm. Available from: http://www.oclc.org/en-US/contentdm.html.

Omeka, 2015a. Omeka: Project. Available from: http://omeka.org/about/.

Omeka, 2015b.Teaching with Omeka. Available from: http://omeka.org/codex/Teach_with_Omeka.

Open Archives Initiative 2002. The open archives initiative protocol for metadata harvesting. Available from: http://www.openarchives.org/OAI/2.0/openarchivesprotocol.htm.

Open Archives Initiative (OAI), 2015. Home. Available from: https://www.openarchives.org/.

Open Source Initiative, 2015. Open Source Definition. Available from: http://opensource.org/.

OpenDOAR: Directory of Open Access Repositories, 2015. Usage of open access repository software—worldwide. University of Nottingham, UK. Avaialble from: http://www.opendoar.org/find.php?format=charts.

Paepcke, A., Chang, C.C.K., Winograd, T., García-Molina, H., 1998. Interoperability for digital libraries worldwide. Commun. ACM 41 (4), 33–42.

Palmer, A., Choi, N., 2014. The current state of library open source software research: a descriptive literature review and classification. Lib. Hi Tech 32 (1), 11–27.

Payette, S., Lagoze, C., 1998. Flexible and extensible digital object and repository architecture (FEDORA). Second European Conference on Research and Advanced Technology for Digital Libraries, Heraklion, Crete, Greece, September 21–23. Available from: http://arxiv.org/abs/1312.1258.

Payette, S., Blanchi, C., Lagoze, C., Overly, E., 1999. Interoperability for digital objects and repositories. D-Lib Mag. 5 (5). Available from: http://www.dlib.org/dlib/may99/payette/05payette.html.

Payne, A., Singh, V., 2010. Open source software use in libraries. Lib. Rev. 59 (9), 708–717.

Public Knowledge Project, 2015. Open Journal Systems. Available from: http://pkp.sfu.ca/ojs/.

Rafiq, M., 2009. LIS community's perceptions towards open source software adoption in libraries. Int. Inf. Lib. Rev. 41 (3), 137–145.

Rieger, O.Y., 2007. Select for success: key principles in assessing repository models. D-Lib Mag. 13 (7–8). Available from: http://www.dlib.org/dlib/july07/rieger/07rieger.html.

Samuels, R.G., Griffy, H., 2012. Evaluating open source software for use in library initiatives: a case study involving electronic publishing. Portal Lib. Acad. 12 (1), 41–62.

Smith, M., Barton, M., Bass, M., Branschofsky, M., McClellan, G., Stuve, D., Walker, J. H., 2003. DSpace: An open source dynamic digital repository. D-Lib Mag. 9(1). Available from: http://www.dlib.org/dlib/january03/smith/01smith.html.

Staples, T., Wayland, R., Payette, S., 2003. The Fedora Project: an open-source digital object repository management system. D-Lib Mag. 9 (4). Available from: http://www.dlib.org/dlib/april03/staples/04staples.html.

Stein, A., Thompson, S., 2015. Taking control: identifying motivations for migrating library digital asset management systems. D-Lib Mag. 21 (9). Available from: http://www.dlib.org/dlib/september15/stein/09stein.html.

Suleman, H., 2005. Analysis and evaluation of service oriented architectures for digital libraries. In: Peer-to-Peer, Grid, and Service-Orientation in Digital Library Architectures, Springer, Berlin and Heidelberg, pp. 130–146.

Suleman, H., 2012. The design and architecture of digital libraries. In: Chowdhury, G.G., Foo, S. (Eds.), Digital Libraries and Information Access: Research Perspectives. Neal-Schuman, Chicago, IL, pp. 13–28.

Suleman, H., Fox, E.A., 2001. A framework for building open digital libraries. D-Lib Mag. 7 (12). Available from: http://www.dlib.org/dlib/december01/suleman/12suleman.html.

Suleman, H., Fox, E.A., Kelapure, R., Krowne, A., Luo, M., 2003. Building digital libraries from simple building blocks. Online Inf. Rev. 27 (5), 301–310.

Swan, A., 2012. Policy Guidelines for the Development and Promotion of Open Access. UNESCO, Paris.

Thacker, C., Knutson, C., 2015. Barriers to initiation of open source software projects in libraries. Code4Lib J. 29. Available from: http://journal.code4lib.org/articles/10665.

Tramboo, S., Shafi, S.M., Gul, S., 2012. A study on the open source digital library softwares: special reference to DSpace, EPrints and Greenstone. Int. J. Comp. App. 59 (16), 1–9.

WGBH Educational Foundation, 2015. Open Vault. Available from: http://openvault.wgbh.org/.

Witten, I.H., 2008. The development and usage of the Greenstone digital library software. Bull. Am. Soc. Inf. Sci. Technol. 35 (2), 31–38.

Witten, I.H., Bainbridge, D., 2003. How to Build a Digital Library. Morgan Kaufmann, San Francisco.

Yeh, J.H., Sie, S.H., Chen, C.C., 2009. Extensible digital library service platform. In: Theng, Y.-L., Foo, S., Goh, D., Na, J.-C. (Eds.), Handbook of Research on Digital Libraries: Design, Development, and Impact. Information Science Reference, Hershey, NY, pp. 27–40.

Zhang, A.B., Gourley, D., 2009. Creating Digital Collections: a Practical Guide. Oxford, England, Chandos.

Zick, G., 2009. Digital collections: history and perspectives. J. Lib. Adm. 49 (7), 687–693.