AJAX Repositories

Matthew Hirst

Department of Computer Science, University of Cape Town
Private Bag, Rondebosch, 7701 South Africa
e.sixis@gmail.com

Abstract. Digital repositories are systems used to store, access and manage digital collections. Most often these systems are Webbased but in places where there is limited or no Internet connectivity a distributed system on a digital medium such as a CD or DVD would be preferred. In order to create a compatible and lightweight digital repository that can be distributed this paper investigates and finds that using AJAX to create a system to browse a digital repository is a practical solution. This paper also looks at existing applications like Greenstone and DSpace and identifies what can be learnt from each implementation.

1. Introduction

This paper is a literature synthesis of papers related to the task of creating a system to browse a digital repository using AJAX or Asymmetric JavaScrip and XML [4].

Traditionally digital collections have been accessed by more heavyweight Webbased applications [10]. This makes them difficult to access in places where Web connectivity is not available or limited and leads to a need for a more lightweight and distributable system.

There have been systems created that could be distributed on CD and DVD, such as Greenstone. However systems like Greenstone need to install software on the computer to function [13]. This leads to the system being incompatible with certain architectures, limiting the reach of the information in the repository.

In light of this and in order to address these issues this paper proposes and discusses using AJAX to create a system to manage, browse and search a digital collection, specifically focusing on the browsing and storing of the information within the repository. AJAX is lightweight and simply needs an Internet browser to function. This makes it an ideal technology for creating a universal and lightweight application.

Firstly, in order to gain a better understanding of what the problem of creating a digital repository entails, this paper will first look at research done in the area of digital repositories.

Secondly, this paper investigates the AJAX technology, what it is, its capabilities and its limitations. An example of the last criteria is that AJAX is not able to access a filestore [6].

Lastly this paper will look at systems already created in the area of digital repositories, the problems they were made to solve and their implementations.

2. Digital Repositories

Digital repositories are obviously stores of information. However, like digital libraries, they tend to be difficult to define.

Digital repositories and digital libraries are similar in definition and it could be argued that the terms could be applied interchangeably [3]. However in a library the librarians have control over what resources are to be placed in the library whereas repositories on the other hand place an emphasis on people being able to contribute to the collection [3].

Because the terms are difficult to define there are many descriptions of digital repositories. One example would be to describe it as a shared database of information with a repository manager that supports various actions geared towards accessing and managing the information. [2]

Another general description of a digital repository is a system used to store and retrieve any digital material. These materials can consist of text, hypertext, images, videos and many other digital formats. [3]

Digital repositories are not just about safely storing digital materials, but more about sharing and reuse [3].

In general, a digital repository is a digital collection with some or other system to access and manage the collection a similar definition to a digital library [11].

Now that there is a definition of a digital repository to work with, this paper will look at the systems that make up a repository, namely the storage system and the system to access the objects/data in storage.

2.1 Storage

Storage in a repository is an integral part of the system and can be done in a variety of ways. Examples include file systems, database systems or even hard-copy filing cabinets. A variety of information about sources in the repository can be stored as well, such as a source's location, its revision history, the tools and processes that were used to build it and even who is able to access the file [2].

More than one system may be used to store information in the repository. For example, a document or source code program may be stored in a file system, while the descriptive attributes of the object such as its location and other information mentioned before are stored in a DBMS [2]. (See figure 1)

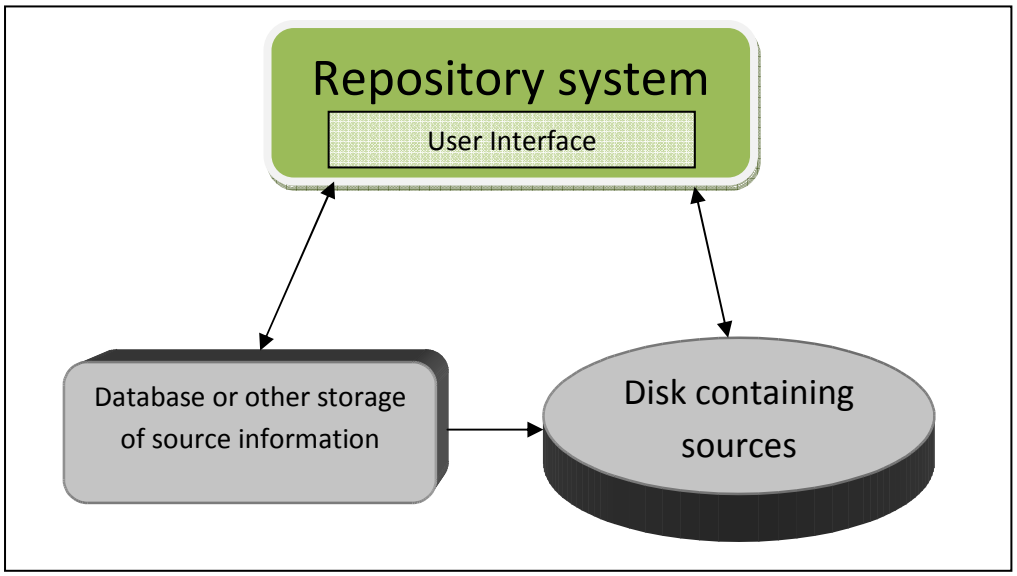

Figure 1: Diagram showing the layout of a repository system using two storage systems.

This is similar to the method one would have to use when creating a repository system with AJAX. As the repository will need to be distributable, the objects will need to be stored on a file system, but one cannot access the file system directly using AJAX. In order to solve this problem we will need a secondary listing of the information, such as location, which we will need to store somewhere where AJAX can access it.

2.2 Data Access System

Once information is stored in a repository, one needs a system to access the objects and manage the repository. A repository manager must provide services for modelling, retrieving and managing the objects in a repository. The repository manager uses the repository's storage manager to store and retrieve the objects. So using the previous example with the file system and the DBMS, when a user asks to retrieve an object, the repository uses the DBMS to look up the object's current location attribute and then uses that to load the object from the file system. [2]

Beyond simple storage and retrieval, tools are needed for managing metadata. These services are the main added features of a repository and provide workflow control, version control, configuration control, content management and checkout/check in[2].

In the example of an AJAX repository this would be the AJAX interface and the pre-processing to create AJAX-readable information.

The tasks needed to be performed by a repository system can finally be summarised by looking at a table from the IMS Digital Repositories White Paper.

Task defined by use	Response of Digital Repository
Search, gather, alert, browse	Expose
Configure	Interface change
Request	Deliver
Publish	Store
Deliver (from one repository)	Store (in another)

Table 1: Tasks and the way a digital repository might respond (based on Digital Repository Interoperability Problem Space from IMS Digital Repositories White Paper [3]

Now that a basis for the functionality and storage needed for a repository has been defined, this paper can look into the AJAX technology to see how it can be applied to the task of creating a repository.

3. AJAX

Web applications for a long time have been less rich and less responsive than their desktop counterparts; however with the introduction of Web 2.0 and technologies such as AJAX this gap is closing. AJAX stands for Asynchronous JavaScript and XML and it represents a shift in what is considered possible on the Web [4].

AJAX isn't really a new technology. It is several technologies working together in powerful new ways [4].

AJAX incorporates [4]:

- standards-based presentation using XHTML and CSS;
- dynamic display and interaction using the Document Object Model;
- data interchange and manipulation using XML and XSLT;
- asynchronous data retrieval using XMLHttpRequest; And
- and JavaScript binding everything together.

In the classic Web interaction model the communication is one way with the user performing an action which then posts back to the server which does some processing and then returns an HTML page to the client [4]. This limits the interactivity and feedback of a website [5].

AJAX allows two way communication to occur between the browser and the Web server without having to install additional software [5]. It does this by introducing an AJAX engine which acts as an intermediary between the user and the server. Instead of loading a webpage, when a session is started the browser loads an AJAX engine written in JavaScript. In cases where AJAX is used, this AJAX engine is then responsible for rendering the user interface as well as communicating with the server [4]. (see figure 2)

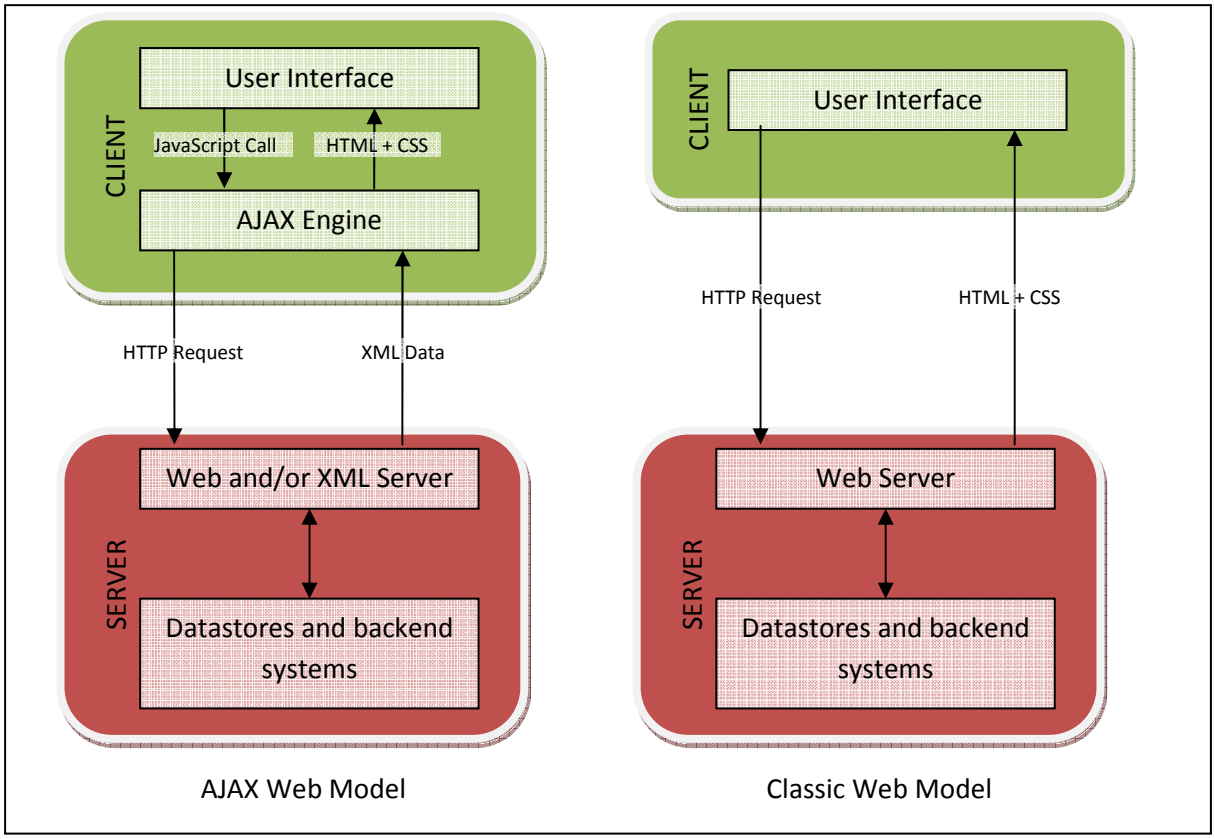

Figure 2 : The different models for AJAX (left) and traditional (right) Web applications.

This enables creating a Web application that works without using a Web server, the application is able to just do all the work through the AJAX engine, never having to contact a Web server. This allows for creating a distributable repository that uses a Web browser without any extra software needed.

Furthermore, using the AJAX engine, one can manipulate the Document Object Model or DOM which is basically the data structure that represents the current webpage [4]. By manipulating the DOM it is possible to create functionality without a Web server. Manipulating the DOM allows an application to create new elements in a Web page, remove them, reorder them and change their attributes. It basically gives the application the ability to directly edit the source of a page [4].

AJAX also is already a very widely used technology with large companies like Google using it in many of their applications such as Google Maps, Google Groups and Google Suggest. The core AJAX technologies are mature, stable and well understood [4]. As such almost all modern browsers have support for AJAX, making any application written with it widely compatible with a variety of operating systems and architectures.

One limitation as specified before is that the JavaScript used is not able to access a local filestore due to security issues. This makes it more difficult for AJAX to access information from a repository and will need a workaround such as embedding the information describing the sources directly into the webpage and then using this to fetch the information. This paper will look into ways of overcoming this limitation by identifying what other systems using AJAX offline have done.

Using AJAX with an intermediate data representation of the digital collection one could use the AJAX engine to browse through the digital collection by modifying the DOM and styling the data using some styling method, such as CSS.

4. Previous work performed in the area

Although no repository has been created using AJAX, previously there have been systems created with similar objectives, either to be a distributable medium for information or to create a lightweight system that is easy to use. In order to gain background information and to learn from what the other systems have done, this paper will investigate several previously created systems, including Greenstone [13], A Student friendly repository for teaching [1], DSpace [7], DotWikIE/TiddlyWiki [6] and The Bleek and Lloyd collection [8], focusing specifically on how the systems browsed and stored information.

4.1 System to access the repository

On the topic of browsing of digital repository systems, this paper will look at 2 well established digital repository systems: Greenstone and DSpace. These systems were created with slightly different objectives. DSpace was created in an attempt to address the problems that the MIT faculty were having in collecting, preserving, indexing and distributing research materials and scholarly publications [7]. Greenstone was created to help people build their own collections [13].

One commonality between the systems was the goal to design the system to make participation by contributors easy [7] [13]. Another commonality is that both systems use a Webbased interface for the users to interact with the digital repository. This would suggest that using a Web interface is the easiest or at least most common or most comfortable way for a user to interact with the system.

Using AJAX to build a digital repository may be natural step in creating a user friendly system. This can be further motivated by the fact that in the student friendly repository for teaching, AJAX was chosen specifically to emphasize the ease-of-use of the system [1].

Another observation is that the systems keep the different interfaces separate, i.e. the interface for users to browse the data is kept separate from the interface to manage the repository [13] [1] [7]. This makes it simpler and easier for the users to browse through the data as well as making the application to browse the data more lightweight and easier to distributable.

In greenstone, data in the repository is browsed based on the metadata. The data in the repository can therefore be browsed in a multitude of ways, e.g. by author, by titile, etc. The browsing facilities are created during the building process by accessing the metadata for each object in the collection [13].

DSpace on the other hand allows data to be added to the repository after it has been created, so statically creating the browse functionality at build time was not an option. DSpaces solution was to have users browse based on a specific index. It does this by providing an API that allows a user to specify an index and a subsection of that index. The indices that may be browsed are item title, item issue date and authors [12].

The information within a distributed repository is likely to be static or not frequently updated, so building the browsing facilities at creation is an option.

4.2 Storage

Often the storage system of choice for a digital library system is a database [8]. This can be seen in systems such as DSpace [7] where mySQL is normally used [8]. This is because databases provide an efficient and easy way to manage the data in the collection [8]. However in order to use AJAX one would have to find another way to store the data as AJAX is unable to connect to a database [6].

One technology that has already been suggested and tested is XML [8]. It has been used in collecting data in the Greenstrone system [13] and it also fits well into the AJAX schema as XML is one of the core technologies AJAX works with. However storing files in XML still requires the XML files to be on a filestore and AJAX cannot directly access a filestore.

The solution TiddlyWiki used was to store the data embedded in the Webpage in the form of text and hyperlinks [6]. In TiddlyWiki the sources are known as 'tiddlers'. 'tiddlers' Are created, edited and deleted through the use of JavaScript embedded in the same file. By editing the DOM as stated in the AJAX section of this paper we can hide, display and load information through hyperlinks onto the page without having to have direct access to browse the filestore.

4. Conclusion

This paper has looked at background information on building an AJAX Repository. It has looked at what a repository entails and the functionality that is expected when a repository is created. It has been seen that AJAX is a proven and widely compatible technology that could be applied to the task of creating a distributable and lightweight repository.

This paper has also looked at the other work that has been done in the area of creating lightweight applications and repositories, and analysed how these systems have been built. In conclusion it has been seen that AJAX is a practical and fitting solution to creating a lightweight distributable repository.

5. References

[1] P. Achananuparp and R. B. Allen, "Developing a student-friendly repository for teaching principles of repository management" In *DigCCurr: International Symposium in Digital Curation,* 2007

[2] P.A. Bernstein and U. Dayal, "An overview of repository technology" in Proceedings of the 20th International Conference on Very Large Data Bases, Morgan Kaufmann Publishers, San Francisco 1994, pp. 705–713.

[3] C. Duncan, "Digital Repositories: e-Learning for Everyone," presented at eLearnInternational, Edinburgh, February 9-12, 2003

[4] J.J. Garrett, “Ajax: A New Approach to Web Applications”, [Online]. Available: http://adaptivepath.com/publications/essays/archives/000385.php. [Accessed: May. 5, 2009]

[5] M. Greensade, “Development of web applications using AJAX”, BSc Thesis, University of Portsmouth.

[6] M.J. Rees, “Ultra Lightweight Web Applications: A Single-Page Wiki employing a Partial Ajax Solution”, [Online], Available http://ausweb.scu.edu.au/aw06/papers/refereed/rees/paper.html. [Accessed: May. 5, 2009].

[7] M. Smith et al., “DSpace An Open Source Dynamic Digital Repository”, D-Lib Magazine, vol. 9, no.1, [Online]. Available: http://www.dlib.org/dlib/january03/smith/01smith.html. [Accessed: May. 5, 2009].

[8] H. Suleman, “Digital Libraries Without Databases: The Bleek and Lloyd Collection”, [Online], Available: http://pubs.cs.uct.ac.za/archive/00000433/01/ecdl_2007_dlwd.pdf. [Accessed: May. 5, 2009].

[9] H. Suleman, “in-Browser digital library services”, [Online], Available: http://pubs.cs.uct.ac.za/archive/00000434/01/ecdl_2007_ajax.pdf. [Accessed: May. 5, 2009].

[10] H. Suleman, “AJAX Repository – Department of Computer Science”, March. 9, 2009. [Online]. Availiable: http://www.cs.uct.ac.za/teaching/honours/honours-projects-2009/ajax-repository/. [Accessed: May. 5, 2009].

[11] H. Suleman, CSC4000W DL lectures, University of Cape Town, 2009

[12] R. Tansley et al., “The DSpace Institutional Digital Repository System: Current Functionality” In Proceedings of *The 3rd ACM/IEEE-CS joint conference on Digital libraries*, 2003, pp. 87-97

[13] I.H. Witten, D. Bainbridge, S.J. Boddie, “Power to the People: End-user Building of Digital Library Collections” In Proceedings of JCDL, 2001