Creating a Low-cost South African Heritage Portal

Honours Literature Review

Toshka L. Coleman
Department of Computer Science

University of Cape Town
South Africa
clmtos001@myuct.ac.za

Abstract

In recent years, digital heritage portals have been introduced with the purpose of preserving and accessing historical archives. Heritage portals that encompass archives from various external sources have been implemented internationally for projects such as the Europeana project, but this has not yet been developed for South African heritage archives.
Thus, this project aims to create a low-cost local heritage portal encompassing multiple local heritage archives on a central domain. Metadata harvesting would be used to gather metadata from multiple archives and store them in a central database. The Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) is a low-barrier mechanism for repository interoperability that would be the most fitting approach for the portal as it is a widely used protocol that can be implemented relatively simply.
In this review, various literature related to heritage portals are reviewed, covering system architectures, metadata harvesting, search and browse service implementation and low-cost practices implemented in related work.

CCS Concepts

• Heritage Portal, Metadata Harvesting, OAI-PMH, Digital Library Systems

Keywords

heritage portals, local, low-cost, architecture, search, browse, metadata harvesting

1. Introduction

South Africa possesses a unique heritage with renowned cultural significance, characterized by multiculturalism, post-colonialism, and the Apartheid era [10].
This heritage is the reason for the diverse culture seen in South Africa presently. One's knowledge of this heritage thus aids their understanding of our current society, its politics, historical monuments, and distinctive art [10]. For this reason, the preservation, as well as the accessibility of national heritage documents are essential for learning about and conducting research on South Africa's unique heritage.

In the past, South African memory institutions have made various attempts to develop archives for heritage content. The Bleek and Lloyd Collection [13] is an example of such an archive, but only contains a single domain of heritage.
Europeana [2] is an example of a high-cost heritage portal in Europe, encompassing a vast assemblage of European archives in a central system.
The South African National ETD Portal [14] is an example of a low-cost, local system developed for collating South African theses and dissertations resources.

There has not yet been a central system developed that contains multiple domains of local heritage archives. A central heritage system would allow historians, researchers, students and whoever else may be interested in South African heritage to search through a wide variety of focused historical documents. This is currently only possible through tools such as Google Search, proving impracticable due to its lack of and limitation in the specialization of focused, local heritage documents.

This project, therefore, aims to develop a South African heritage portal that uses the best practices in low-cost heritage archive systems, spans multiple local heritage archives in a central domain, and provides users with useful services like search and browse. We will be procuring our data from The Five Hundred Year Archive (UCT History), Bleek and Lloyd Collection (UCT Fine Art), and Metsemegologolo (Wits/UP).

This paper covers the following topics: system architectures of Digital Library Systems (DLS) [12] and portals, metadata harvesting through the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) [9], search and browse service implementation and low-cost practices implemented in related work.

2. System Architectures of Related Work

2.1 Background

The architecture of the heritage portal is constrained by various factors that influence the design decisions made for the system [15]. The heritage portal would need to be built to support multiple repositories and to allow for standardization in terms of formatting, functional structures, services and other constraints.
Additional factors [15] to consider when designing such a portal include interoperability, generality, usability by different populations, scalability and preservation.

Generally, Web Portals make use of DLS technology. A DLS [12] is an electronic collection of resources gathered from several repositories, encompassing a database of digital objects including text, images, audio, video and other digital media formats [12]. The core architectural components used in digital library systems include a digital object store and a metadata store implemented with databases and filesystems, and a collection of services which allows access to the digital object and metadata store [15].
With central DLS architectures, digital objects and metadata and services provided are stored in a central hardware system while a distributed DLS architecture stores digital objects, metadata and services in multiple locations [15].

In the early 2000's, the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) [9] was introduced and enforced the arising notion of interaction amongst components in a service-oriented architecture. Many projects have shown that components connected using well-defined protocols as the core architecture of a DLS has proven most viable and has thus become the standard in DLS architecture [15].

In this section, system architectures used in the context of various related projects will be reviewed. These architectures include multi-tiered, layered and distributed architectures.

2.2 Multi-tiered Architecture – NETD

Multi-tiered architectures implement client–server architecture and discriminate the presentation, processing and data management function components from each other [6].

The South African National Electronic Theses and Dissertations (NETD) portal [14] is an example of a local metadata aggregation system that uses a multitiered architecture. NETD provides the public with access to full-text records of South African theses and dissertations aggregated from various sources, and also facilitates the coordination of the development of ETD programs at tertiary institutions in South Africa.

Evidently, the portal's central repository architectures are connected to large repository environments, therefore a multi-tiered architecture where complex modules combined with an operating system framework has been used for its development [14].
This system's architecture has been componentized into multiple discrete components in order to allow for modifications to be isolated to the component concerned, instead of having to revise the entire system. Also, it prevents the entire system from failing if any element of the system is involved in a failure [14].

The South African National ETD's architecture is divided into the following three components, with their associated functions. These components, and their interactions are represented in Figure 1:

1. Harvester: The component that is responsible for retrieving metadata from ETD repositories. It uses the OAI-PMH [9] - a standard that facilitates the transfer of metadata from digital repositories and includes a database that incorporates a machine interface and an end-user management interface. The metadata retrieved is stored in the shared Harvester and Repository data store with MySQL used as the database layer. The Harvester performs initial validation checks to prevent malformed records from being sent to the Repository through the shared data store [14].

2. Repository: The component that provides a series of machine access points allocated to the metadata that is retrieved from the Harvester [14].

3. Web Portal: This component utilizes metadata and services from the Repository to allow for an end-user integrated interface including all repositories that had been harvested [14].

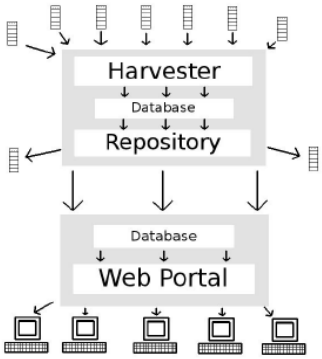

Figure 1: NETD High-Level System Architecture [14]

2.3 Distributed Architecture – NSDL

A subsequent example of an architecture that uses metadata aggregation to gather its content is the National Science Digital Library (NSDL) [16]. Alternatively, this system makes use of a distributed architecture as there are additional units involved in the processing of information; including user profiles, and access and rights management services.

Distributed Architecture [4] involves different units interacting with one another through a communication medium, with the

purpose of fulfilling a specific goal. This architecture does not restrict information processing to a single machine but rather distributes it over multiple independent machines.

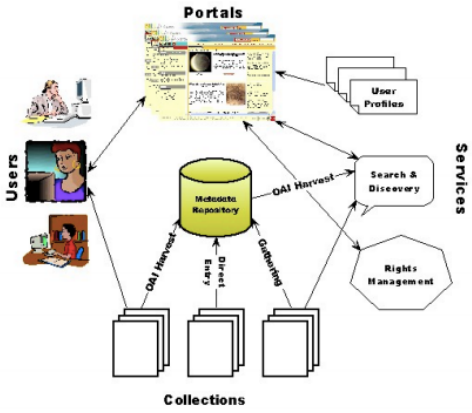

Figure 2: NSDL Components Architecture [7]

The NSDL [16], founded by the National Science Foundation, is an online library that provides structured public access to resources in science, technology, engineering and mathematics (STEM) and research collected from other digital libraries and NSF-supported projects. The portal is maintained using MySQL, PHP and Internet Scout Portal Toolkit which is functioning on Apache Servers [16].

The NSDL's system architecture [16] incorporates a distributed architecture structure and is centred around a focal metadata repository that stores metadata retrieved from various repositories. The high-level view of its architecture, as shown in Figure 2, includes the metadata repository, portals, services (search and discovery), user interfaces and user profiles, collections which are harvested and rights and access management.

Regarding the metadata repository, it's metadata is distributed with service providers through the OAI-PMH interface [9] and the repository consumes metadata in three ways. These include direct entry into its database, through Web-crawling and gathering from the Web, or by harvesting via OAI-PMH [16]. The latter will be described in the following section on metadata harvesting.

For access management [7], this component is responsible for ensuring that service and data providers conform to the set of requirements for the library. This involves authorization and authentication - through the use of standard protocols by administrators, User Profile Servers – which contains user attributes in order to customize a user's experience, and the Rights Management Broker – which coordinates decisions related to user access of the library, based on user and item attributes [7].

Lastly, the User Interface component allows for users of the NSDL to access collections and services through portals [16].

NSDL's Services component is covered in Section 4.4.

2.4 Layered Architecture - Bleek and Lloyd Collection

Contrary to the previous two architectures, The Bleek and Lloyd Collection [13] does not aggregate archives from different sources as it contains only its own collection. It uses a layered architecture to perform various processes on the data within the collection at the different layers.

Layered Architectures [6] involve a system of layered components that have specific responsibilities, interact with each other, and transfer data between the different layers at different levels of processing.
The Bleek and Lloyd collection [13] is a digital repository consisting of documents, notebooks and sketches recording the history and culture of the |xam and !kun people of Southern Africa. The collection is composed of work from Lucy Lloyd and Wilhelm Bleek and was initiated in 2003. The Lucy Lloyd Archive and University of Cape Town's Research centre proceeded to digitize and scan historical artefacts that were added to the online collection. This initiated the development of the Digital Library (DL) System with the features that facilitates storing, management and preservation of historical, digital records [13].
The architecture for this system includes three primary layers as seen in Figure 3:

1. Client Layer: Comprised of User Interface wherein end-users interact with the system to access its collection [13].
2. Service Layer: Comprised of services, "Search", "Browse" and "Index" shown in Figure 3, that allow information from the collection to be discovered [13].
3. Repository Layer: Applies hierarchical structure and stores metadata objects in XML plain text files and digital content in its collection on the filesystem. This structure was used as it aids in the preservation of the semantics of digital content and allows global operations to be applied to all metadata objects within the container [13].

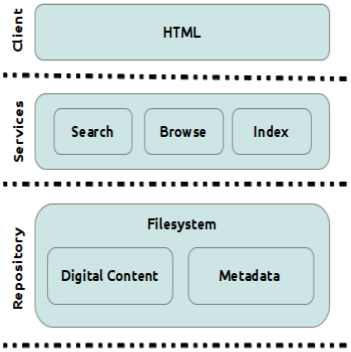

Figure 3: The Bleek and Lloyd Collection System Architecture [13]

As an overview, this digital collection was constructed from pre-generated hyperlinked static XML pages [13], which were

combined with the search component developed through Ajax and used JavaScript to facilitate query operations.

The abovementioned metadata objects consisting of XML text files have been encoded using the Dublin Core metadata scheme. They consist of three different types: container metadata objects, virtual objects and digital content metadata objects [13].
These container metadata objects outline the configuration of directories made up of other container metadata objects, the virtual objects determine the semantics of the illustrations and the collection using logical correlation and digital content metadata defines the raw images used [13].

The front-end of the collections includes the statically generated HTML files and the search feature that used Ajax for online and off-line searches [13]. This was generated through the use of XSLT style sheets added to the collection of metadata objects. It allowed the collection to be independent of a Web or Application server for retrieving results. Subsequently, a Web browser would only be required from end-users, allowing the collection to be stored and accessed from storage devices such as CD-ROMs and USB drives [13].

Having covered the architectural aspect, the processes that occur within the harvesting components of the metadata aggregators will be discussed in the following section, with reference to related work.

3. Metadata Harvesting via the OAI-PMH Protocol

3.1 Background

The content gathered from the heritage portal's data providers will be collated onto a central domain using metadata harvesting, implemented via the OAI-PMH protocol [20].

Metadata harvesting facilitates the identification and collation of resources based on their associated metadata, gathering metadata from numerous source archives or repositories and aggregating them to a single database destination [20].

In recent years, the OAI-PMH [9] has become the standard protocol used for metadata harvesting. The OAI-PMH implements unqualified Dublin Core as the metadata standard in order to maintain interoperability, thus Data Providers providing metadata have to ensure that their records are in the same format defined in XML. Service providers include the establishments that harvest data from Data Providers. They enhance metadata records and allow for a richer environment and more advanced services such as search engines [21].
Furthermore, the OAI-PMH contain requests which are sent and defined as HTTP (Hypertext Transfer Protocol) requests using GET and POST operations - the repositories are thus required to support these methods. The requests are all stored in a base URL which outlines the network host and port of the HTTP server that is the repository [21].
Thereafter, the responses to OAI-PMH requests are represented as XML sheets and are encoded using UTF-8 representation of Unicode with character references instead of entity references. This is used in order for XML responses to be independent of external entity declarations [9].

These requests and their associated responses [9] include:
Identify – returns description of archive, *ListMetadataFormats* - returns a list of the supported metadata formats, *ListSets* - returns a list of subsets of the repository, *ListIdentifiers* returns a list of record headers with associated date and set, *GetRecord* - returns a single metadata record for an associated identifier, *ListRecords* – returns list of metadata record as combination of the *ListIdentifiers* and *GetRecord* requests [9].

The application of the OAI-PMH within related metadata aggregation projects will be discussed in the following subsections.

3.2 Application of OAI-PMH Protocol in Related Work

The NETD relies on the OAI-PMH to retrieve metadata from several ETD repositories [14]. The system's aforementioned architectural components make use of the protocol as follows.

Following the Harvester's retrieval of metadata and storage of it in the data store, the Repository uses the OAI-PMH to make them accessible. The OAI-PMH interface is provided to the public and allows its records to be accessed with a standard protocol [14].

This interface satisfies the scalability constraint as it able to facilitate harvesting from continental and international repositories while minimizing network traffic [14]. The Web Portal then provides a user-centric interface for the metadata collection. It uses the OAI-PMH interface from the Repository to access its metadata records and harvests them into its local database. These metadata records stored in the local database, as seen in Figure 1, are processed using Lucene for search indexes and MYSQL for browse indexes.
Fundamentally, the Web Portal encapsulates and allows standard end-user services, as well as Web Services such as OAI-PMH and RSS, to be available to end-users in a single location [14].

The Networked Digital Library of Theses and Dissertations (NDLTD) [17] has also opted for the OAI-PMH, deviating from the federated search system previously used.
The OAI-PMH was chosen because it varies from former approaches to interoperability as it involves minimal interaction among external sites and the central site. Also, the protocol transfers increments of metadata thus a remote search facility is not required [17].

In the NDLTD, data is acquired via the OAI-PMH from a data provider and is stored locally, providing services to users using the response data.
Evidently, the response data in XML format has shown to be a more accurate structural specification language than Document Type Definition (DTD) [17].
Additionally, the Repository Explorer [17] used within their OAI-PMH architecture has the ability to perform machine validation of responses, and the CGI mechanism used for encoding requests has shown to have greater stability as a Web technology.

Moreover, the NDLTD claimed that metadata harvesting via the OAI-PMH has shown to be effective for their project due to the ease with which it can be modified to support NDLTD's specific requirements. These requirements include *simplicity* – through the use of toolkits - and its *ability to support arbitrary metadata formats* – where NDLTD had implemented a standard for metadata exclusively for ETDs that are supported by the OAI-PMH [17].

The NSDL [8] is another example of a metadata aggregator that has implemented the OAI-PMH for metadata harvesting, but, with the primary motivation being that it is simpler to implement and to maintain.
Furthermore, the protocol's standard of using metadata in Dublin Core format allows the prerequisites set for the NSDL metadata records to be met prior to them being harvested [8].

The aforementioned metadata repository is implemented as a relational Oracle database. Java Database Connectivity (JDBC), a Java Interface, is used by Oracle for interactions with the database [8].

As stated above, it supports the OAI-PMH as well as other methods of ingest. The harvested metadata from collections are encoded in XML sheets and are stored in an interim staging area. The records in the staging area are then involved in the following three stages [8].
Firstly, they are involved in the clean-up stage which includes tasks integrating *ListRecords* responses and often removing the OAI-PMH wrapping. A crosswalk follows in order to produce a metadata record in the standard and normalized format: nsdl_dc. These crosswalks are performed in XSLT. XML files containing sets of records are then generated. Lastly, using Java, the XML files are uploaded to the database, storing native metadata as well as the nsdl_dc records.
The relational tables within the repository are configured to the OAI server. SQL queries are submitted to the repository and the relevant records are submitted to the OAI server when harvesting requests are received from the NSDL service providers [8].

The search and browse services that are provided using the data collated as a result of harvesting will be covered in the next section.

4. Search and Browse Services

4.1 Background

The heritage portal will support search and browse services in order for uses to specify, identify and gather required content.

Searching in digital libraries involves inputting items related to an identifier and retrieving relevant content from various remote databases containing digital objects. These databases may include metadata for relevant objects or entire objects such as a an article, book or video [15].

Browsing is an information-seeking activity relatively less directed than searching as the resulting content is not necessarily predetermined. Browsing digital libraries is usually done through browsing by index such as browsing by date or author [15].

Search and browse are the core services in digital repositories and involve incorporating aspects of information retrieval. These services are typically implemented through a server-side index such as Apache Solr. The collections can be stored on the client's side therefore browser-based search and browse can also be implemented with the use of standard operations within the browser [15].

The common information retrieval technology, Federated searching [15], concerns services that access various distributed sites with the intention of fulfilling specific requests.

A federated search involves queries sent to a DLS that are directed to multiple external sites. The external sites' results are then merged into a single result set using result fusion. Additionally, source selection can be used to optimize the sets of remote sites queried by removing sites that do not contain relevant results [15]. Moreover, Federated Search requires that the DLS supports a remote search protocol such the Search/Retrieval via URL (SRU). SRU refers to a client-server protocol in which the client sends URL-encoded query parameters to the server and acquires an XML-encoded list of results in return [15].

In the following subsections, the implementation of search and browse services within a client-side system, a lightweight system and a metadata aggregation system will be discussed.

4.2 Implementation of Search and Browse Services in Related Work

4.2.1 NSDL's Implementation of Search Service

The NSDL [1] implements its search service using the OAI-PMH and additionally, Lucene, thereafter. Its discoveries services were developed with a server that returns ranked lists of items that match submitted queries.

In the original implementation of the NSDL search service, a commercial version of InQuery (search engine developed at the University of Massachusetts Center for Intelligent Information Retrieval) was integrated for its search engine [8].
A list of searchable items is acquired for the search operation by gathering the contents contained in Metadata Repository using the OAI-PMH. Through the application of this protocol [8], the repository's contents can to be acquired initially, and the search component's indexes are updated frequently by harvesting new and modified items. Search interfaces can easily be imported into databases through the use of an open protocol interface.
The search engine interacts with the portals with the use of the Simple Digital Library Interoperability Protocol (SDLIP) [7] protocol which specifies the methods for which queries are sent between clients and servers, as well as how results are returned.
The search's results are then structured as a ranked list of items. Through the SDLIP, the portal receives information of the item and a pointer to the item. These pointers enclose information which enables the Rights Broker to determine whether the user searching has access to the retrieved items [7].

In the latest implementation of NSDL [1], a more high-performance technique for its search engine was implemented - a search service that uses Apache Lucene for indexing and query processing was introduced. Through the use of the OAI-PMH and Metadata Repository, Lucene indexes the metadata as well as the full-text resources provided in the metadata records. In this way, the search service translates the metadata-centred data model to a resource-centred model [1].

4.2.2 In-Browser DLS' Implementation of Search and Browse Services

A study conducted by Suleman [18] aimed to assess the performance of a search and browse system based solely within a browser and without a network connection and a software installation.
Interestingly, it was concluded that this system shows potential for efficiency with the use of JavaScript and pre-indexed data stored in XML files. Additionally, it displayed reasonable performance for basic operations varying from small to medium sized collection [18].

This system's implementation of the search service applies information retrieval principles and uses an extended Boolean model [18] while the browse service is supported by database formatted indices containing items that match a particular term.
This system includes two sets of indices that are stored as XML documents which allow them to be processed using built-in browser services. The search index for a term will contain a list of identifiers of all items including the term. The browse index for a field name with a certain value will include a list of identifiers of all items in which the field has that specific value. The fields for searching and browsing are specified in a configuration file [18].
Essentially, the search system contains a Perl script that creates the indices [18] needed for search/browse and a JavaScript file that performs search or browse operations and displays the relevant results on the web browser window.
The relevant search and browse indices are loaded and the necessary processes are performed to produce results ordered by relevance and filtered by the particular browse fields [18].

4.2.2 SimplyCT's Implementation of Search Service

SimplyCT [3] is a more lightweight system approach, developed with the purpose of creating a Digital Library System (DLS) using simple architecture. Its framework is made up of a collection of archive data, services and indices within a hierarchical structured directory. The archive files are outlined in XML metadata files and its services adhere to a server-instance model [3].

Similarly to the previous project, indices and query operations for the search service, however, they are also implemented it using the Xapian information retrieval library as follows [3].

When a user searches for content within a particular archive, a GET request is sent from the XML HTTP Request in the Python CGI scripts [3]. These CGI scripts are instances of the search service allocated to the particular archive. The search query is conducted using the shared code in the lib/search directory. This code generates the paths for the archive and interfaces with Xapian. Xapian manages the productions and searching operations of the indices. The XML-formatted search results are sent to the JavaScript on the client-side and is transformed into a XHTML sheet that is presented to the user [3].

Including services, various aspects of heritage portals have been covered thus far. However, in order for these to be implemented successfully, the cost constraint of the portal should be considered.

5. Practices for Low-Cost Project Implementation

This Heritage Portal will include a large number of resources from various external repositories that would need to be managed, thus low-cost and lightweight practices need to be considered as a constraint in this project.

5.1 Low-cost System Practices

5.1.1 Low-cost Project Constraints

According to the principles outlined in The Bleek and Lloyd Project, [13] in order for electronic text to be low-cost, it should be implemented in plain ASCII as it is capable of reaching more users and has shown to withstand other formats and standards more cost-effectively.
In line with the lightweight SimplyCT project, low-cost projects should adhere to the following constraints [19]:

- Only the minimum system infrastructure that is required for a system should be integrated in order to minimize the cost of maintenance.
- It should not be a requirement for users to conform to system requirements for formats, identifiers etc, and access to files should be a primary form of accessing data as opposed to using a Web Services interface.
- Data should be capable of being accessible through a network and without one.
- Preservation should be implemented by copying relevant files and directories rather than using a database in order to maintain simplicity.

5.1.2 The Five Hundred Year Archive's Implementation of Lightweight Principles

The Five Hundred Year Archive [19] is an archive system including a collection of books and articles encompassing the heritage of KwaZulu-Natal and its surrounding areas. It is a lightweight system implementing cost-effective practices.
The system has been implemented as lightweight by following the principles outlined by The Bleek and Lloyd Project [13] as follows [19]:

- **Minimalism –** The amount of software was minimized through the implementation of short scripts as opposed to multi-layered abstractions and Web Service-based architectures.
- **Imposition on users –** The metadata in the spreadsheets are imported in batches. The data items already have identifiers assigned to them and are reused.
- **No Internet** - Read-only access is implemented without a Web application, but instead via direct access to relevant HTML files. JavaScript applications that read static files are used for the system's search and browse services. On online format of the archive is only required when data is added to the archive; in all other cases the read-only archive can be used and distributed through an offline medium.
- **Simple Preservation** - The original data and metadata files are preserved in their XML files and only requires to be copied. Migration is implemented by transformation of metadata files.
- **Any Objects** - All data file formats that can be accessed with a Web browser are supported by the system;
- **Flexibility -** Search and browse services are able to be defined to function with any metadata repository.
- **Platform agnostic –** The system is composed of HTML files that can be viewed on any operating system which has a Web browser available. The system can therefore be accessed from any device without modifications to the data or the system needed.
- **Collection building** - When new items are acquired, all data is pre-processed. The system is therefore permanently in a static state with the quickest possible access to relevant items [19].

5.2 Lightweight Digital Library Implementation Using Automated Linking

Using an alternative, more technical approach to those previously mentioned, the Digital Library Integration Infrastructure (DLII) [11] implemented an organized lightweight technique for the integration of digital library repositories and services. This system achieves a candid, lightweight and sustainable infrastructure for integrating through the use of automated linking [11].

In this approach, an anchor produces a list of links related to metainformation including structural and knowledge-sharing relationships, and metadata.
The DLII [11] links their collections automatically to relevant services and related collections, and provides the objects in collections (and other services) with automatic and direct access to these services.
Thereafter, the DLII filters and orders this set of generated links to user preferences and tasks.

This approach has been described as "lightweight" and "non-intrusive" [11] as its integration requires minimal to no modifications to its system's source code, and repositories and services are able to function independently of DLII after integration. DLII's approach has also shown to be more cost-effective with regards to time and resources due to its lack of system's documents and code changes required. [11].

An issue with many digital library systems, outlined by the developers of the DLII, is that they are not designed to allow for open access to their data and functional architecture [11]. This arises as the digital library systems are integrated with a variety of autonomous organizations' repositories. It can become difficult and more costly for integration architectures as inter-organizational systems are not able to enforce standardization constraints and thus rely on mechanisms and protocols that do not require compliance to a set of standards [11]. For this reason, loosely coupled approaches that prevent existing components from being affected when new components are added, and also avoid systems from needing to comply to a strict set of standard can be seen as the more lightweight approach. It also allows library systems sourcing data from various archives to provide users with ubiquitous content [11].

7. Summary

In this paper, various aspects related to developing a heritage portal were considered and discussed. These aspects include system architectures, metadata harvesting via the OAI-PMH [9], portal services (search and browse) and low-cost practices used in related work.

Table 1: High -Level Comparison Summary of Projects

Project	Architecture	OAI-PMH implemented	Search/Browse Implementation Tool	Cost
Bleek and Lloyd Collection	Layered	No	Ajax	Low
NETD	Multi-tiered	Yes	OAI-PMH	Low
NSDL	Distributed	Yes	OAI-PMH & Lucene	High
NDLTD	Distributed	Yes	Lucene & SOLR	High

Table 1 presents a comparison table that summarises the key aspects of the projects discussed in this paper. It includes the architectures used for the projects, whether metadata harvesting via the OAI-PMH was implemented in the project, the tool or technology used for implementing search and browse and the project's cost level.

For the heritage portal that we plan to implement, the following has been considered and concluded in the paper.

Due to multiple archives being aggregated from various external sources, the architecture would need to enforce interoperability, generality, usability by different populations, scalability and preservation [15].
In terms of implementation, it was found that OAI-PMH [9] has been construed as the most widely used interface for metadata harvesting as it has shown to be the most efficient way of facilitating data distribution while enforcing standardization and simplicity [20].

Regarding architectures, the NETD's architecture [14] and implementation of the OAI-PMH have shown to be the most promising approach as it allows for modifications as well as errors to be isolated to the component concerned instead of having to revise the entire system [14]. It has also shown to enforce scalability successfully as it is able to facilitate harvesting from continental and international repositories whilst minimizing network traffic. With scalability being a crucial constraint for this project, this was an important outcome [14].

In terms of portal services, using a search engine toolkit such as Apache Lucene for indexing and query processing as used in NSDL has shown to be a high- performance services approach [1].
For low-cost approaches, various principles can be implemented to allow for a more cost effective system. These include minimalism of software, imposition on users, flexibility and collection building [19]. Using a loosely coupled system approach with automated linking has also shown to enforce a more lightweight system [11].

References

[1] William Y. Arms, Naomi Dushay. Dave Fulker, and Carl Lagoze, 2002. A Case Study in Metadata Harvesting: the NSDL, Library Hi Tech.

[2] Sally Chambers and Wouter Schallier, 2010. Bringing Research Libraries into Europeana: Establishing a Library-Domain Aggregator, *Liber Quarterly*.

[3] Azhar Desai, 2010. SimplyCT Online Search, *CS10-04-00*, Department of Computer Science, University of Cape Town.

[4] Mirjana Ivanović, Milan Vidaković, Zoran Budimac and Dejan Mitrović , 2016. A scalable distributed architecture for client and server-side software agents. *Vietnam Journal of Computer Science 4*, 2, 127-137.

[5] Bimal Kumar. 2016. Layered Architecture for Mobile Web Based Application. *International Journal of Software Innovation 4*, 3, 51-64.

[6] Pradeep Kumar and Yogesh Singh, 2010. A Software Reliability Growth Model for Three-Tier Client Server System. *International Journal of Computer Applications 1*, 13, 9-16.

[7] Carl Lagoze, Walter Hoehn, David Millman, William Arms, Stoney Gan, Dianne Hillmann, Christopher Ingram, Dean Krafft, Richard Marisa, Jon Phipps, John Saylor, Carol Terrizzi, James Allan, Sergio Guzman-Lara, and Tom Kalt, 2002. Core Services in the Architecture of the National Science Digital Library (NSDL)", *In Proceedings of Second ACM/IEEE-CS Joint Conference on Digital Libraries*, 201-209.

[8] Carl Lagoze , Dean Krafft, Tim Cornwell, Naomi Dushay, Dean Eckstrom, and John Saylor, 2006. Metadata aggregation and 'automated digital libraries': a retrospective on the NSDL experience, in Nelson, M. L., and Marshall, C. C. (eds): *Proceedings of the 6nd ACM/IEEE-CS Joint Conference on Digital Libraries, ACM,* 230-239. doi:10.1145/1141753.1141804

[9] Carl Lagoze, Herbert Van de Sompel, Simeon Warner and Michael Nelson, 2001. The Open Archive Initiative Protocol for Metadata Harvesting, Open Archives Initiative. Available http:// www.openarchives.org/OAI/openarchivesprotocol.htm

[10] Joann McGregor. and Lyn Schumaker, L, 2006. Heritage in Southern Africa: Imagining and Marketing Public Culture and History. *Journal of Southern African Studies 32*, 4, 649-665.

[11] Nenchi Nnadi and Michael Bieber, 2004. Towards Lightweight Digital Library Integration. University Heights, Newark, NJ 07102, USA.

[12] Kay Oddone, 2015. Exploring Digital Libraries: Foundations, Practice, Prospects. *Australian Academic & Research Libraries 46*, 3, 219-220.

[13] Lighton Phiri and Hussein Suleman, 2012. In Search of Simplicity: Redesigning the Digital Bleek and Lloyd, *DESIDOC Journal of Library & Information Technology*, 32, 306-312, DESIDOC, Ministry of Defence, India.

[14] Hussein Suleman, Tatenda Chipeperekwa and Lawrence Webley, 2011. Creating a National Electronic Thesis and Dissertation Portal in South Africa. *Proceedings of 14th International Symposium on Electronic Theses and Dissertations*. Cape Town, South Africa.

[15] Hussein Suleman, 2012. Design and architecture of digital libraries. *Facet Publishing.* Cape Town, South Africa.

[16] Hussein Suleman, Edward A Fox and Devika Madalli, 2003. Design and Implementation of Networked Digital Libraries: Best Practices, *Proceedings of DRTC Workshop on Digital Libraries: Theory and Practice,* Bangalore, India.

[17] Hussein Suleman, Edward A Fox , 2002. Towards Universal Accessibility of ETDs: Building the NDLTD Union Archive, *Proceedings of The Fifth International Symposium on Electronic Theses and Dissertations* (ETD 2002), Provo, Utah, USA, Available at http://www.wvu.edu/~thesis/proceedings.html

[18] Hussein Suleman, 2019. Investigating the effectiveness of client-side search/browse without a network connection, *Proceedings of 21st International Conference on Asia-Pacific Digital Libraries (ICADL),* Kuala Lumpur, Malaysia, Springer.

[19] Hussein Suleman, 2019. Reflections on Design Principles for a Digital Repository in a Low Resource Environment, *Proceedings of HistoInformatics Workshop 2019, 13 September 2019,* Oslo, Norway, CEUR.

[20] Herbet Van de Sompel, Michael Nelson, Carl Lagoze and Simeon Warner, 2004.Resource Harvesting within the OAI-PMH Framework. *D-Lib Magazine 10*, 12.

[21] Mary Woodley, 2008. Crosswalks, metadata harvesting, federated searching, metasearching: Using metadata to connect users and information. In M. Baca (Ed.), Introduction to metadata 38–62, Los Angeles, CA: Getty Research Institute